INTERNATIONAL HELLENIC UNIVERSITY

Traffic Prediction in Smart Cities, Featuring the Impact of COVID-19

Liapis Stergios

SID: 3308190015

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2021

THESSALONIKI – GREECE

INTERNATIONAL HELLENIC UNIVERSITY

Traffic Prediction in Smart Cities, Featuring the Impact of COVID-19

Liapis Stergios

SID: 3308190015

Supervisor: Assoc. Prof. C. Tjortjis

Supervising Committee Members: Dr D. Karapiperis
Dr N. Serketzis

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2021

THESSALONIKI – GREECE

Abstract

This dissertation was written as a part of the MSc in Data Science at the International Hellenic University.

Smart cities emerge as highly sophisticated bionetworks, providing smart services and ground-breaking solutions. This dissertation presents a novel method using classification in the context of Smart City projects, focusing on traffic prediction. A methodical literature review identifies the main topics and methods used, emphasizing on various Smart Cities components, such as data harvesting and data mining. It addresses the research question whether traffic loads can be forecasted based on past data, meteorological conditions, seasonality, and time intervals, as well as COVID-19 related restrictions. We propose reliable models by utilizing smaller data partitions. Apart from feature selection, we incorporate features related to movement restrictions, forming a pioneering data model. Our approach explores the optimal degree of data aggregation. Results have shown that various models can be developed with varying levels of success. The best outcome was achieved when factoring in all relevant features. Accuracy improved significantly compared to previously published work.

Acknowledgements

Firstly, I would like to thank my supervisor, Professor Christos Tjortjis, for providing invaluable advice and guidance and for encouraging me to publish my research.

I would like to express my special gratitude to Konstantinos Christantonis for his willingness to assist me and for sharing his valuable domain knowledge.

I am thankful to Victor Chazan–Pantzalis for providing important insights on data sources and for his useful suggestions. Additionally, I would like to thank DOTSOFT S.A. for providing the sensor data required for our experiments.

Lastly, I am grateful to my family and my close people for their faith in me and for giving me strength throughout this dissertation.

Stergios Liapis

03 January 2021

Contents

1 Introduction

Smart Cities are designed to solve urban challenges. The growing urban population leads to more complex city problems. Thus, Smart Cities should focus on providing proactive solutions and improving liveability for citizens. Undoubtedly, Smart Governance is a key factor to manage cities more efficiently. Hence, Smart Cities should have data awareness and apply data-driven decision making for Smart Governance. The transformation of cities to Smart Cities is tightly connected with efficiency and transparency. Their function requires innovation in problem solving, keeping in mind citizen needs. As a result, managing cities more efficiently empowers communities and augments sustainability.

A sustainable smart city utilizes city data for enhanced decision making. Data intelligence enables the identification of insights to fulfil the needs of the urban ecosystem. This framework of data creation and data usage improves operational efficiency. Thus, local government should develop the city's data culture and could promote innovation in decision making. However, creating a robust urban governance requires a responsible management of data security and data privacy. Consequently, the concept of Smart Cities is responsible for effective resource management driven by solving complex urban problems.

The deployment of modern smart cities increasingly gains attention, as large urban centers present numerous challenges for citizens over time. Traffic is a stressful and time-consuming factor affecting citizens. Recently, many local authorities attempt to design and create smart infrastructures and tools in order to collect data and utilize models for better decision making and citizen support. Such data are often obtained from sensors collecting information about Points Of Interest (POIs) in real time. Data mining techniques and algorithms can then support getting useful insights into the problem, whilst forming appropriate strategies to counter it.

This dissertation focuses on analyzing different approaches regarding data manipulation in order to predict day-ahead traffic loads around two cities, mainly based on weather conditions, but also including seasonality and the impact of COVID-19 related restrictions. Prediction efforts regard classification tasks aiming at highlighting factors that affect traffic forecast. The novel method we propose utilizes weather data collected from sensors located in Athens and Thessaloniki, Greece. This work is an extension of prior efforts [1], when three different time slots were introduced (Morning, Afternoon, Evening) and compared, while subsets of trimesters were also tested. The current work further expands the analysis to investigate the impact of a fourth time slot, namely Noon. In addition, 4-month periods of time are also tested with the introduction of new features.

The work also includes an analysis on how COVID-19 impacted traffic loads, with COVID-19 used as a factor supplementary to weather conditions. Additionally, it contributes to the understanding of how unexpected events can modify traffic loads on a temporary/semi-permanent basis.

This dissertation differs from standard approaches for traffic prediction in several ways, mainly associated to the methodology. To the best of knowledge, the transformation of the problem into classification and the inclusion of movement restrictions is an approach that has not been used for predicting traffic loads. Our contribution is part of an internal layer that assists those predictions in the day-ahead case.

To summarize, traffic prediction is a problem with extended seasonality. Constructing a general model needs numerous resources. Our focus is on investigating whether we can build reliable models by utilizing smaller partitions of data. In addition, feature selection is essential to solve the classification problem. The incorporation of features related to movement restrictions, led to identifying abnormal traffic conditions and formed a pioneering data model. Furthermore, our approach attempts to ascertain the optimal degree of data aggregation, which is regarded as an open research question. Our computational approach led to a significant improvement of accuracy.

This dissertation is structured as follows: Chapter 2 provides a methodical literature review on Smart Cities, focusing on traffic and accidents prediction. Chapter 3 provides the problem definition and describes the approach we followed for traffic prediction, including data selection and feature engineering. Chapter 4 presents the 2-classes experimental results on a variation of time periods, focusing on the lockdown impact and the comparison between Athens and Thessaloniki. Chapter 5 presents the respective 3-classes experimental results, while Chapter 6 discusses and evaluates the research findings extracted from this dissertation. Chapter 7 summarizes the most remarkable findings and concludes with suggestions for future work.

2 Literature Review

In this section a methodical literature review on the concept of Smart Cities is presented. It comprises all dimensions of Smart Cities, such Energy, Healthcare, Economy, Education and Governance, but mainly focuses on Smart Mobility. More specifically, our focus on Smart Mobility and Transportation is concentrated on traffic prediction and accident prediction.

2.1 Smart Cities

The concept of Smart Cities originally appeared in the literature in the late 1990's, but latest state-of-the-art technologies and evolved human-device interaction have led to a contextual shift in Smart Cities. A general definition of the term is that "Smart City is an innovative city that uses ICT and other means to improve quality of life, efficiency of urban operation and services, and competitiveness, while ensuring that it meets the needs of present and future generations with respect to economic, social and environmental aspects" [2].

The smartness of a city refers to its ability to combine different resources aiming to achieve goals in a more effective way. The Smart City concept focuses on providing sustainable solutions for economic growth, reduced emissions and more importantly improved life conditions for citizens. Interestingly, Figure 1 showcases a wordcloud based on the abstracts of research works on Smart Cities [3]. As expected, "cities" and "data" are the most frequent words. Then, "system", "service", "big data", "technologies" and "application" occur repeatedly. Additionally, we can identify words related to the dimensions of Smart Cities such as "governance", "energy", "economic" and "traffic".

Smart cities are mainly linked with smart mobility, smart living, smart governance and smart environment. Figure 2 indicates that Smart Cities should be considered as a "data engine" in order to effective. This "data engine" is consisted of a cyclic flow of data production, data storage, data analytics and service design [4]. According to Soomro, the main domains of a smart city are governance, environment, transport, economy and energy and each one faces various challenges and limitations, so different solutions could be implemented [5]. Smart cities present big data applications such as smart education, smart traffic lights, smart electrical grid system. In addition, the main challenges in smart cities could be summarized in data sources and characteristics, information sharing, data quality, security and privacy, cost and city population [6].

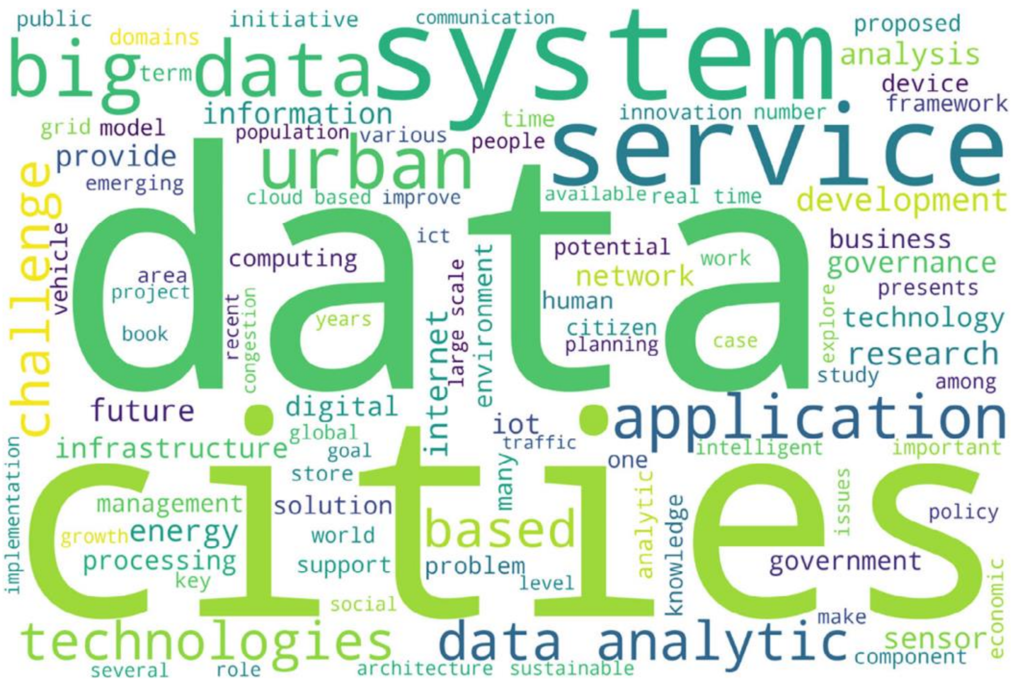


Figure 1: Most frequently used words mined from selected abstracts [3].

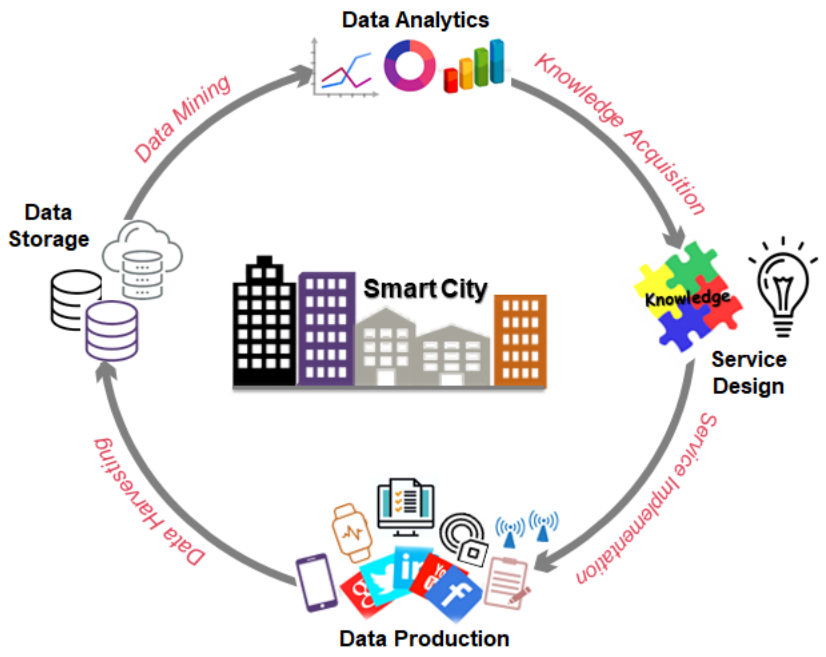


Figure 2: Smart City as a “data engine” [4].

A detailed presentation of each dimension of Smart Cities presented in Figure 3, follows in the next subsection.

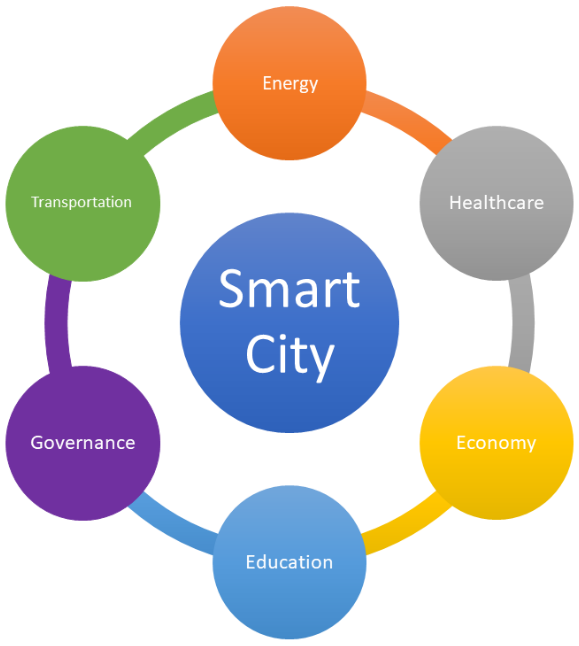


Figure 3: Dimensions of Smart Cities.

2.1.1 Smart Energy

An important dimension of the Smart Cities environment is Smart Energy. The main challenge is to produce cleaner energy, promote lower energy consumption and smooth energy consumption peaks. First, it is urgent to encourage renewable sources of energy such as wind and solar energy. Changing the energy generation from fossil fuels to alternative methods, could reduce air pollution and emissions of CO_2, and improve air quality and make environment cleaner.

Smart Energy concept is based on Smart Buildings and Smart Homes. Smart buildings emphasize on enhancing energy efficiency along with high levels of comfort for their occupants. For instance, smart sensors will provide information about energy usage, reveal load patterns, predict peak demand and measure air quality. Thus, it is possible to detect the least energy efficient buildings and regulate energy improvement measures in order to comply with the energy policy. In addition, real-time energy consumption is available and billing information is simplified.

Smart sensors could identify the energy demand patterns in households [7],[8]. There is a variation between energy supply and demand throughout day, so dynamic pricing could balance the energy management. This could transform the most energy consuming habits. For example, avoid operating the washing machine when energy demand is at peak level due to the higher pricing [9]. It also has the potential to optimize the energy usage. As a result, implementing the smart building concept could benefit owners with reduced energy bills and lower energy consumption.

2.1.2 Smart Healthcare

Smart Cities applications can contribute to transiting from preventing healthcare systems to treating systems. The level of deployment of Smart Healthcare applications in 50 cities is presented in Figure 4. Smart healthcare focuses on preventing diseases before they occur, reducing the possibility of dangerous complications or the need for a patient to be hospitalized [10]. This can be achieved by gathering real time data of patients through smart devices, which will monitor for instance blood pressure, blood sugar levels, sleep patterns and heart rate. Collecting all this information on a daily basis will give more accurate diagnoses and even detect health issues that are difficult to be recognized by a health expert.

Also, patient history records could be beneficial for insurance companies and some government agencies which will suggest more personalized health and treatment plans for each individual patient [6]. Additionally, smart healthcare can be a tool for accessing rural areas that suffer from poor connectivity and limited supply of doctors by providing personalized treatment, early detection of illness, imaging and AI driven diagnostics [11].

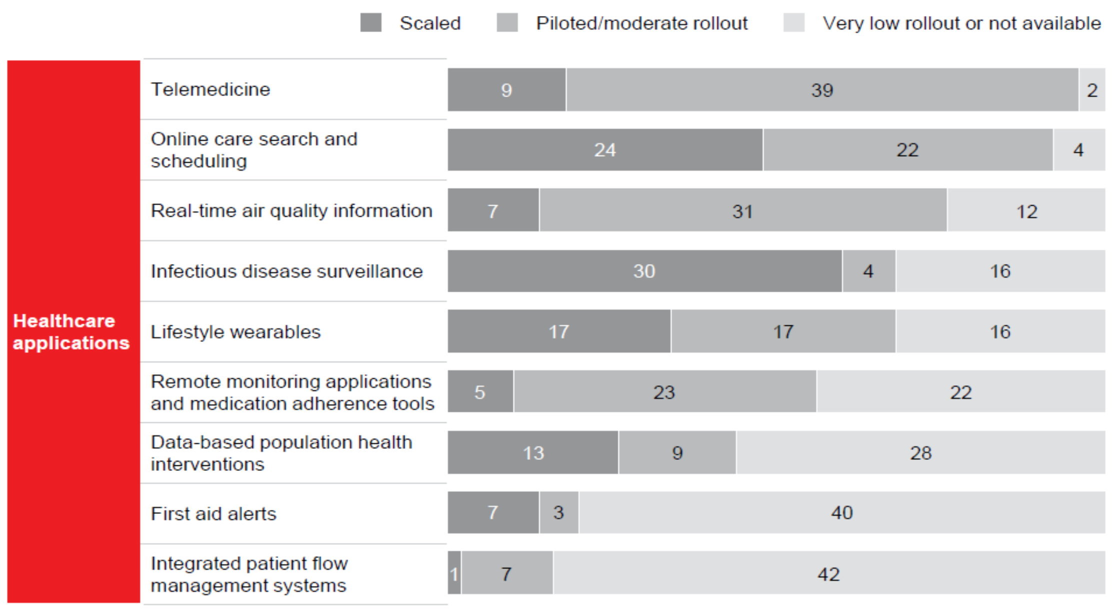


Figure 4: Rollout status of healthcare applications [10].

2.1.3 Smart Economy

Smart economy applications are a vital part of the Smart City concept. The increasing amount of data combined with AI technologies and analytics could provide better results in risk assessment. It is true that access to detailed data of peoples' needs, preferences and behavior is a key step to maintain competitive advantage. In addition, insurance companies could benefit from IoT and valuable available data to improve risk assessments and offer their clients more personalized services and products.

Also, a smart dynamic pricing is one of the most important infrastructures of modern cities, where efficiency is maintained while demand and supply are aligned. Thus, smart technologies like sensors and payment systems could produce real-time data about the use of infrastructure and contribute to create a real dynamic pricing in smart cities. Moreover, the implementation of new technologies has already changed the way of payments and transactions in everyday life. Indeed, mobile payments via smartphones combined with biometric authentication are leading to an almost complete elimination of cash money for payments which are now digitalized [9].

2.1.4 Smart Education

One of the expectations of the digital revolution is to improve the educational system in all levels from primary schools to universities and lifelong learning. Smart education aims to deploy all potential useful data and combine them in a way that the learning outcomes will be improved.

Nowadays, education systems treat all students with the same way, ignoring the uniqueness of every student or learner and their personal skills or special talents. Personal learning is the primary goal of this concept which aims to adjust the content, the speed and the style of instructions in order to meet the needs of each student individually.

Also, taking advantage of data and analyzing them could give insights about faculty performance, course progression, or students tuition fees which will benefit colleges and universities. Among others, predictive analytics could make accurate predictions about students who are about to drop out or face severe difficulties and support them using different methods and material.

Another application of smart education is lifelong learning which focuses on effective and usually short-term training programs for midcareer workers or people with no working experience. These programs help learners to advance their skills or gain new ones, find their way into new occupations and expand their knowledge. Digital online platforms could also contribute to a more efficient labor market, mitigating unemployment levels. Finally, smart education could enable easy access to students in disadvantaged populations or inaccessible areas to high-quality education, resources and expert support [10].

2.1.5 Smart Governance

A significant component of a Smart City is Smart Governance which aids planning and decision making, by taking advantage of data-driven useful information. One of the most challenging applications of the smart concept for governments is to integrate different agencies and combine their processes. This inter-departmental collaboration will result in more efficient operations, enhance policy making and enforcement and data will be easier to handle [6]. Indeed, the need for a big data strategy for smart cities is urgent, where privacy is preserved in open data, which are securely gathered and processed to support effective and timely decision making [3].

In addition, the smart concept will help governments to focus on citizens’ issues and concerns about healthcare, transportation, education, or economy and develop citizen-centric public services, which are accessible anytime anywhere. Furthermore, smart governance aims to improve business decisions by supporting firms with big data analytics tools. Data concerning a firm’s behavior in comparison with its competitors, along with economic and environmental factors can lead to more accurate decisions about employment, production, sales and strategies [6]. However, smart governance solutions require city administrations to reform their processes and adopt these solutions, as well as IT experts to design and implement these technologies [3].

2.1.6 Smart Mobility and Transportation

The growth of population in the last decades has led to a significant increase of vehicles driving on the roadway every day, especially in big cities. Therefore, traffic congestion and accidents have increased, since transportation systems have not evolved, and strategies and policies have not been agreed to solve these problems. Thus, predicting traffic and dealing with it has gained great attention and became a vital issue in smart cities.

Traffic data can be collected from different sources such as sensors and GPS devices, but they are difficult to be handled, processed and analyzed because of their huge size. Data mining techniques could extract useful information and discover hidden patterns from large amount of traffic data, which can help governments, companies and researchers to make better decisions. Analyzing and predicting traffic data could also help drivers organize their trips better, find the best route for their destination or even avoid risky areas and drive safe.

Smart Transportation could benefit a city by predicting traffic congestion and controlling traffic flow. For instance, recognizing traffic patterns enables efficient transport management. By analyzing traffic data, local government it able to regulate the supply chain delivery system. As a result, transport optimization is essential to reduce delivery time and energy cost. The challenge here is to facilitate data creation and data usage. In that case, control of smart traffic lights could be coordinated with traffic information collected through sensors [6].

Additionally, finding a free parking spot is a great challenge especially in large cities. Smart solutions can be used to optimize the occupancy of parking spaces by detecting if a spot is occupied or not via sensors. This data could be beneficial for users to reduce their time spent on searching for parking space and provide with several alternatives according to price and nearest solution. Smart Mobility also includes solutions to individuals needing or selling a ride. There are multiple smart apps and digital platforms which have recently gained attention and their systematic use could lead to reduced congestion and emissions [9].

2.2 Traffic Prediction

Traffic data analysis could be used to predict traffic flow and local speed at several locations examined and also locate areas with high traffic congestion level. A recent study [12] presents a methodology of short-term prediction of traffic state and local speed using data-driven approaches, as seen in Figure 5. Data has been collected in Irvine CA and Tel Aviv, Israel through sensors which are numerous, and facilitate researchers to retrieve information about traffic flow. The results indicate that the more the clusters, the lower the performance. It is proposed that a lower number of clusters related to the traffic states leads to a better performance. On the contrary, a big number of clusters leads to overfitting problems. This data-driven approach could be expanded with additional explanatory variables so that system is fed with more information, which helps with the understanding and prediction of the traffic system.

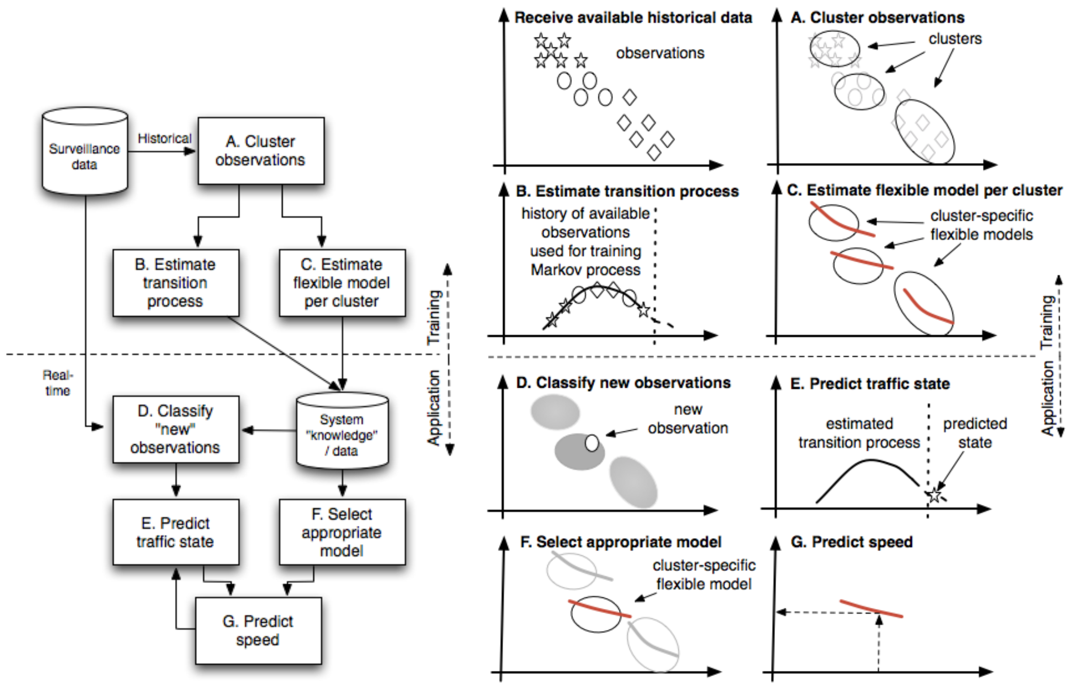

Surveillance data
Historical
A. Cluster observations
B. Estimate transition process
C. Estimate flexible model per cluster
Training
Real-time
Application
D. Classify "new" observations
System "knowledge" / data
E. Predict traffic state
F. Select appropriate model
G. Predict speed
Receive available historical data
observations
A. Cluster observations
clusters
B. Estimate transition process
history of available observations used for training Markov process
C. Estimate flexible model per cluster
cluster-specific flexible models
D. Classify new observations
new observation
E. Predict traffic state
estimated transition process
predicted state
F. Select appropriate model
cluster-specific flexible model
G. Predict speed

Figure 5: Overall local traffic state prediction framework [12].

Similarly, Kalvapalli et al. investigated the influence of weather conditions on taxi rides in New York City and applied XGBOOST technique to predict the mean waiting time for taxi pooling [13]. Especially, it is claimed that the average travel speed is mainly affected by the hour and the day of the week, but not by the month. It concludes that XGBoost is the optimal algorithm to predict the trip duration given the origin and the destination coordinates.

In another work [14], authors developed a model that supports spatio-temporal queries on taxi trips in New-York City. The model can detect patterns and create visual representations of the traffic system. Moreover, pickup and drop-off clusters were created. The traffic flow to main hubs, such as train stations and airports was researched and traffic state changes over time were investigated. The dataset was also used to predict the cost and duration of taxi trips, by applying Linear Regression and Random Forest. The results were satisfactory and comparable with Google Maps prediction [15].

Hashemi et al. employed classification techniques to predict the future traffic state. The following classification algorithms have been applied: Classification Tree, Naïve Bayes, Random Forest and CN2 [16]. The target variable was the Level Of Service (LOS) in short time intervals; it was better predicted by Classification Tree and Random

Forest. Another purpose of that paper was to extract if-then rules, which were used for studying traffic patterns. This was accomplished by applying the CN2 model. The rule with the highest quality seemed to be "IF Time Duration (min)<=240 AND Flow (veh/h) <=774 AND Time Duration (min)>30 AND Speed (km/h)>69 THEN nextState=A". Extracting these patterns and understanding the traffic system parameters such as speed, flow and travel time could benefit the traffic management.

Similar research aimed to predict traffic congestion using association rules to achieve higher accuracy in classification problems, taking into consideration the traffic environment with DBSCAN (Density Based Spatial Clustering of Application with Noise) [17]. The main idea was to predict the congestion level having the temporal association rules of the traffic environment, which are extracted by Genetic Algorithms. The classification levels of traffic congestion included the following states: free, moderate and severe based on the traffic density. The proposed method achieved high accuracy close to 95%.

Theodorou et al. focused on traffic prediction in California based on abnormal conditions, such as accidents and road maintenance [18]. The proposed model was more accurate and achieved generalization ability as more information about abnormal conditions was inserted. Another study conducted based on data from Los Angeles [19], proposed a feature selection model to investigate the impact of traffic incidents. According to this study, it is urgent to include non-recurring incidents, such as accidents, music concerts, road reconstruction, because these events are possibly related to traffic jams. Considering all these factors, their prediction model combines historical data and event information, and outperforms well known algorithms in terms of accuracy, for different traffic conditions.

Figure 6 represents a scalable distributed stream mining system for highway traffic data, which can perform traffic analysis and detect real time abnormal events, traffic congestion and predict flow speed [20]. The limitations of this proposal include inadequate research, limited distributed data mining algorithms and infrastructure issues related to sensors. An experiment was conducted based on the proposed system with promising results and it was regarded as a successful trial, which could improve the transportation system. Vlahogianni et al. established that further research is needed to ascertain the optimal degree of data aggregation [21], which further motivated our work.

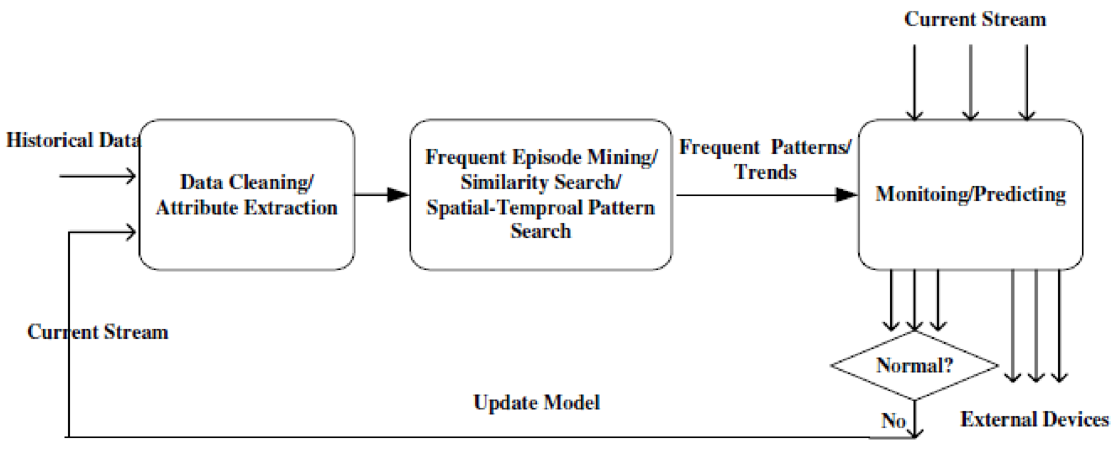

Current Stream
Historical Data
Data Cleaning/ Attribute Extraction
Frequent Episode Mining/ Similarity Search/ Spatial-Temproal Pattern Search
Frequent Patterns/ Trends
Monitoing/Predicting
Current Stream
Normal?
Update Model
No
External Devices

Figure 6: Framework of the distributed traffic stream mining system [20].

Gecchele et al. presented a comparison between various clustering methods in order to estimate the Annual Average Daily Traffic (AADT) in the Province of Venice [22]. The study was carried out using data from Automatic Traffic Recorders (ATRs) which were capturing data for each traffic direction. Vehicles were categorized in two classes in order to lower the complexity of the problem. The two categories used were Passengers Vehicles (PV) and Truck Vehicles (TV), depending on the vehicle’s length. Results have shown that 7 clusters of roads should be formed. The model-based clustering methods, such as the VEI (Variable Equal Identity) model, achieved marginally better results than hierarchical and partitioning methods. This work suggests that traffic prediction is more demanding on weekdays and during the summer and depends on vehicle-type for truck vehicles.

In another research, street segments in New Haven, Connecticut were studied based on GPS data from 10000 vehicles. Necula et al. tried to explore street segments which present a similar traffic flow over time [23]. Data mining techniques, such as K-means, were applied to generate patterns based on temporal and spatial characteristics, resulting to an optimal model with 4 clusters. Interesting insights were discovered by clustering the data and constructing Speed-Density, Speed-Flow and Flow-Density diagrams. One of the conclusions indicated that lower density implies higher speed values and vice versa. In addition, peak hours are identified between 7:45 and 8:45 in the morning, as well as between 17:15 and 18:15 on weekdays.

Another research on traffic prediction claimed that the optimal process of collecting travel data for different transportation modes could be with GPS travel surveys [24]. Then Bolbol et al., applied an SVM model in order to predict the transportation mode (car, bus, train, tube, walk) based on attributes such as speed and acceleration and achieved accuracy of 88%. Interestingly, research efforts simulated the traffic system based on data from connected automated vehicles (CAVs) in order to reduce energy consumption and traffic levels up to 20% with penetration rate at 0.5 [25].

Another application of data mining aims to predict traffic in a Bike-Sharing System. Li et al. developed a model to predict the incoming flow of bikes to each station and the outcoming flow of bikes to each station. The main idea is to optimize the reallocation of bikes. First, stations were clustered based on their location. Each station's flow is affected by numerous factors such as time of the day, weather conditions, events and there is correlation between stations, which should not be ignored. The experiments were conducted on four datasets from D.C. and NYC. RMLSE and Error Rate metrics were employed to measure the performance of methods, such as hierarchical prediction (HP), ARMA, HP-KNN (based on K-NN), geographical clustering (GC). The results showed that as the number of clusters increases, prediction accuracy is affected negatively. Moreover, HP-MSI (prediction based on multi-similarity inference) achieved the best performance for common hours and P-TD (Prediction based on clustering Trip Duration) for anomalous hours [26].

At the IEE ICDM 2010 Contest "TomTom Traffic Prediction for Intelligent GPS Navigation" data mining algorithms were presented that could efficiently predict the traffic flow in order to provide smart navigation to the drivers and optimize city planning. The contest included three tasks, the 1st task was about prediction of traffic congestion based on historical timeseries data from traffic recorder devices and evaluation on RMSE. The 2nd task was related to predicting traffic jams during peak hours due to road maintenance based on radio information; precision was considered as evaluation metric. The 3rd task referred to short-term traffic prediction based on real-time data acquired by sensors installed on vehicles, having as evaluation metric the RMSE [27].

Especially, for the case of Thessaloniki Mystakidis and Tjortjis investigated the traffic system and suggested that the size and the quality of the dataset affects the model's accuracy and a Decision Tree Classifier has been identified as the best performer [28]. Boufidis examined the traffic conditions in Thessaloniki based on a dataset generated by GPS sensors on taxi vehicles [29]. After the implementation of map-matching algorithm, the Random Forest regression algorithm outperformed the other models in all evaluation metrics such as Mean Square Error (MSE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE). In an alternative approach for traffic prediction in Thessaloniki, Adam used classification and regression algorithms on floating car data combined with the weather status [30].

The most recent study presents a traffic prediction approach based on historical traffic loads data and weather data collected from 8 sensors in Athens and 2 sensors in Thessaloniki during a six-month period. Moreover, data collection consisted of averaging sensor values and assigning them to the following intervals: Morning, Afternoon and Evening. The target variable is the Jam Factor and it is investigated as a binary and a three-class (high, medium, low) variable. For the classification task, the Random Forest classifier is applied, with satisfactory accuracy results in both cases, especially for the binary experiments. Thus, beneficial insights were produced about the traffic system, such as that the traffic load is heavier in the afternoon. Interestingly, there is a fluctuation in accuracy during time periods and the highest performance was presented in July-October 2019 [1]. We considered this research work remarkable and regarded it as a baseline for this dissertation. We aimed to extend this research. A detailed analysis is presented in Chapter 3.

2.3 Accident Prediction

Traffic states are affected significantly by unexpected events, such as accidents. Research efforts focus on the causes of traffic accidents and the injury severity. Specifically, Najada et al. investigated the main causes of traffic accidents happening daily and cause congestion especially in big cities [31]. It is shown that human behavior plays a significant role in traffic causation and attributes such as age, type of driver (pedestrian, cyclist, car occupant) were selected for prediction of new cases of road accidents. The algorithms that gave the best results were Naïve Bayes and Decision Tree Classifier C4.5 and data mining tools like H20 and WEKA were used in order to design and run the experiments. In the future we could use real time traffic data to adjust traffic flow and reduce traffic accidents.

Another study proposed that other factors causing serious road accidents could be alcohol consumption and collision type [32]. Moreover, the fatal rate is not significantly affected by weather conditions, light conditions or road surface type. The association rules discovered were produced by Apriori, Naïve Bayes classifier was applied and clusters were formed by k-means. By clustering it is obvious that some geographical regions have higher fatality rate compared to others, so this information led to advise drivers to drive more carefully when driving through higher risk areas. To extend this research we could incorporate traffic data of both fatal and non-fatal accidents, so we could be more accurate in detecting patterns and extracting useful information.

Unfortunately, road accidents are linked with injuries, casualties and fatalities daily. The main idea to reduce road accidents, is to detect the most influential factors and try to mitigate them. Data mining techniques were used in order to predict the severity of injuries caused by traffic accidents [33]. The algorithms used in this research include Decision Tree, Rule Induction, Naïve Bayes and Multilayer Perceptron. The study indicated that the main factors affecting mostly the severity of an injury were age, gender, nationality, casualty status and collision type. The results have shown that young drivers tend to be involved more frequently in road accidents. Also, drivers are more prone to accidents than pedestrians and male drivers more so than female ones.

Additionally, a study based on crash data during (2006-2008) from traffic crashes on two-lane, two-way roads in Iran [34], tried to identify key factors which affect traffic injury severity with data mining, as seen in Figure 7. The dataset contained 14 variables of which injury severity is the target. The target variable is categorized in three levels: 1st level no-injury, 2nd level injury and 3rd level fatality. Classification and Regression Tree (CART) classifier was applied for the classification problem. According to CART, the most important features were improper overtaking and not using the seat belt, which increase the probability of injury and fatality. Future direction could focus not only on rural roads, but also on urban areas.

Krishnaveni et al. focused on several classification models in order to predict the severity of injuries caused by traffic. Many features were taken into consideration, such as the weather conditions, the number of vehicles involved in the crash, the role of casualty, the driver’s sex and age. Feature reduction was implemented with Genetic Algorithm. Then, three experiments were conducted based on accident information, vehicle information and casualty information and in each case the respective attributes were involved. Random Forest was the best performing model in all experiments [35].

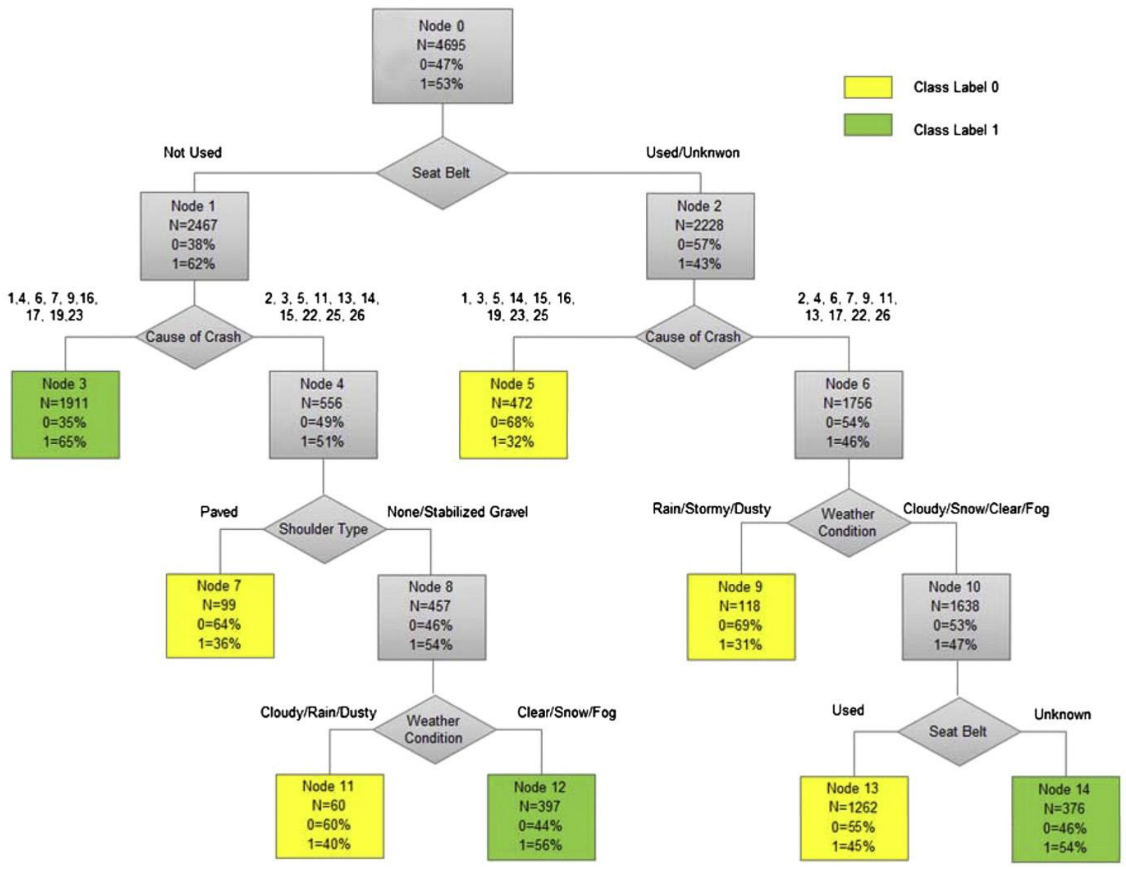


Figure 7: Decision Tree of traffic accident severity prediction [34].

Another research focused on detecting patterns appeared in car crashes and developing a model that can predict the class of injury severity [36]. Chong et al. compared the performance of neural networks, decision trees and a concurrent hybrid model involving both. The state of the target variable can be no injury, possible injury, non-incapacitating injury, incapacitating injury and fatal injury. The models were applied on the GES automobile accident dataset from 1995 to 2000, which is consisted of 417.670 instances with a variety of attributes about driving, road condition, weather and injury details. The hybrid model outperformed the other models in predicting the state and it is shown that the attribute affecting the level of injury the most is vehicle speed. We could extend this research by applying more advanced machine learning models.

Moreover, Shanthi et al. implemented data mining techniques and more specifically classification algorithms such as C4.5, CR-T, ID3, CS-CRT, CS-MC4, Naïve Bayes and Random Tree on data about road traffic accidents [37]. The training dataset consisted of 77125 samples with 33 attributes related to time, driver's characteristics, accident description, alcohol test, location and having as target attribute Injury Severity which takes the following values: Possible Injury, Non-incapacitate, Incapacitate and Fatal. Since the misclassification rate was high, authors applied feature selection algorithms to select the most important features and then incorporated a meta classifier algorithm Arc-X4 to improve accuracy. Results showed that Random Tree classifier using meta classifier achieved the highest performance with 99.73% in accuracy. According to feature importance results, injury severity is mainly affected by collision type, seating positing, harmful event and protection system.

Ramya et al. also attempted to identify the risk factors causing traffic accidents that can lead to fatalities [38]. The classification of the type of accident severity included 3 classes: fatal, serious and slight and features were selected based on Information Gain and Gain Ratio. The models were developed using different combinations of attributes affecting them like weather and light conditions, speed limit and road type and surface. The target classes were imbalanced and oversampling and undersampling methods were applied to eliminate bias. The proposed classification algorithm is Random Forest classifier, which outperformed other algorithms tested, such as J48 Decision Tree and Artificial Neural Networks.

In an alternative approach, incident situations are investigated using data mining tools [39]. Lee suggested that data mining techniques could help the user to define the impact area of the incident temporally and spatially. The technique proposed the clustering of data based on time, traffic density and flow. The information was retrieved and visualized through DataScope, so that traffic engineers could tackle traffic problems and instantly intervene when an incident occurs. The algorithms used for this purpose were Fuzzy C-Means (FCM) and some built-in algorithms known as relation finder and decision support. Future improvements could include the application of data mining tools to establish a management traffic system.

To conclude with, road traffic accidents and traffic congestion constitute a serious problem and prediction of its magnitude has become a necessity. Data mining techniques could be very beneficial in order to predict and give solutions to these problems, in combination with the application of the right policies and measures to reduce them.

3 Methodology

Weather data can be exploited in different smart cities problems and scenarios. Collecting accurate weather data can benefit numerous daily problems which associate weather with human decisions. Traffic is affected by various factors, only some of which are predictable. As mentioned in Chapter 2, traffic is directly affected by weather conditions, however the scale differs across cities, cultures and climate zones.

Also, traffic is affected by emergency incidents, which cannot be predicted. On the other hand, it would be very useful to predict the impact that those events may have on traffic. This is a challenging task. It is also necessary to discriminate between unexpected events with temporary effects and those with mid-term or long-term effects. COVID-19 is such a new circumstance affecting everyday life as well as traffic.

This chapter presents the problem we address with a description of our approach, followed by a detailed description of the data we collected, the necessary preprocessing and feature selection we conducted prior to experimentation presented in Chapters 4 and 5 respectively.

3.1 Problem Definition

This section tests the intuition presented above, about traffic prediction. Traffic is a multi-dimensional problem; researchers mostly focus on either predicting traffic loads on a certain time interval or selecting the optimal route based on real-time adaptations, in order to minimize travel time. Accidents and social events can shortly disrupt normal traffic in specific areas, while seasonality and weather conditions can affect traffic on a larger scale. Based on that, weather data were used, collected from sensors installed around carefully chosen city spots, for predicting the day-ahead traffic volume.

To select the most appropriate locations for installing sensors that either measure traffic loads or collect weather data, it is crucial to define their objective in advance. Research efforts focus on the question: “How can one exploit sensor data that are not personalized and create meaningful conclusions for the general public?” Deployment of smart city infrastructure requires a deep understanding of the traffic problem. Roads that are busier than others do not always provide more information as compared to more isolated ones. The number of alternative roads or the location of busy buildings can affect the necessity of measuring traffic for a specific road.

3.2 Approach

The approach followed, besides examining traffic predictability based on weather data, aims also to clarify the differences among two cities, namely Athens and Thessaloniki, about which we collected data. Our effort focused on creating two new features that describe "abnormal" conditions throughout the year, one for the separation of weekdays and weekends and one related with movement restrictions. Moreover, a series of experiments were conducted regarding different monthly periods serving the objective to understand which months contribute positively to the deployment of such models. The approach we followed is elaborated in Chapter 4 for 2-classes and in Chapter 5 for 3-classes.

In addition to the main problem we addressed, the outbreak of COVID-19 added a new dimension to our research. Two main questions were integrated into the problem definition: The first question was: "in what way did lockdown restrictions imposed because of COVID-19 affect traffic?" The second question was: "in what way is a model for traffic prediction affected by emergency situations like the COVID-19 pandemic?" Following this, we focused on creating another subset of data related to the lockdown period and examined the model on it.

Reformulating the problem from one requiring regression to one suitable for classification by categorizing the target variable into two and later into three classes, was an important decision. Ideally, the outcome should be the exact Jam Factor value, requiring regression techniques, but this research is conducted with a broader motivation to identify factors that affect predictions positively or negatively. An investigation was conducted on how effective the model would be for each set of parameters and which reasons lead to differentiation in accuracy. Therefore, the model was initially transformed into a binary one. The split point was the median value for each dataset, aiming at a fully balanced task, which would grant safer conclusions.

Thus, by addressing the problem as a classification one, the whole process could be easily automated. Forecasting Jam Factor fluctuation has a totally different purpose than building a model that focuses on the exact Jam Factor value.

The two initial labels were “High” and “Low”, with regards to the Jam Factor. The median values reflect the median Jam Factor throughout predetermined time periods. In a different set of experiments, the target variable was split into three categories of equal size (namely, “High”, “Medium” and “Low”), following the same splitting strategy. The metric used for evaluation was accuracy since classes are balanced and of equal interest.

After training the model, two types of model evaluation were used. One involved using a test set, containing 20% of the records available, and another using a validation set. Both evaluation types produced matching results with high accuracy. The results presented in Chapter 4 and in Chapter 5 pertain to the type using the validation set, as it emerged after a typical 10-fold cross validation.

Finally, apart from splitting the time based on COVID-19 related restrictions, seasonal based segregations were also performed. Models based on both 3-month and 4-month intervals were used and their results were compared with the results from the whole dataset. Interesting conclusions were extracted and are discussed in Chapter 6.

3.3 Dataset

In this section, the selection of traffic and weather data is presented. Follows an analysis of the preprocessing and the feature engineering steps. Then, an exploratory analysis is performed to investigate data and extract insights.

3.3.1 Dataset Description

The data source we used was ppcity.eu [40], an integrated system of platforms, which promotes contextual engineering and citizen involvement in decision making and life quality improvement. Multiple environmental, geospatial, and traffic data have been integrated into ppCity platforms. These data are open and may be exploited for experimental purposes. They are collected from various sources and serve the analysis purposes that have been pre-established.

We focused on data extracted from several sensors located at central points around the cities of Athens and Thessaloniki, Greece. These two large urban centers host almost half the Greek population. In general, there are numerous city spots in both cities, where the environmental and traffic conditions are measured. Initially, locations were chosen according to data available regarding traffic and weather conditions. The focus of our attention was on 26 reliable spots that produce data, i.e. sensors with the wider range of recording and most compact flow of data. The dataset comprises data collected between 01.08.2019 and 31.08.2020.

Weather related attributes we used for our analysis, involved the following: relative Humidity (%), Atmospheric Pressure (hPa), Temperature (^oC), Wind Direction (degrees), Wind Speed (m/s) and Ultraviolet Radiation (UV index). The target variable (traffic load), named Jam Factor, represents quality of travel. It ranges from 0 to 10, where 0 describes a completely empty road and 10 stopped traffic flow. Data samples are given in Appendix A.

3.3.2 Dataset Pre-processing

The way the data were processed is conceptually straightforward. Regarding the data structure, three distinct time intervals within a day were originally defined and their predictability was examined [1]. Initially the focus was set on specific hours, which were recognized as the time intervals when people drive to or depart from their jobs, and the time interval that they usually go out for entertainment. The first interval, named Morning, includes data from sensors between 07:00 and 10:00. The second one named, Noon, includes data between 12:00 and 15:00. The third interval, named Afternoon, includes data between 15:00 and 18:00 and the last one, named Evening, includes data between 19:00 and 22:00.

Therefore, the average value for the given data was computed. For example, if a sensor captures information every hour (e.g. at 07:10, 08:10, 09:10 etc.), the average value for each weather metric and the traffic load for that specific interval was computed.

Fortunately, the quality of the dataset is high, thus missing values are minimal. It is worth noting that the averaging strategy implemented does not replace missing values. If there is a missing value in weather metrics, averaging is performed using the remaining two available values, instead of averaging and labeling three values. We used data collected from 26 sensors over 13 months for our experiments. Thus, the volume of data available is significantly large.

Moreover, the imbalance between the sensors located in Thessaloniki and Athens was addressed. In Christantonis et. al [1], 8 sensors were in Athens and 2 in Thessaloniki. Following an increase to sensor availability, data were collected from 10 sensors located in Athens and 16 in Thessaloniki. The location of those sensors is depicted in Figure 8. With this distribution of sensors, a straightforward comparison of accuracy regarding the Jam Factor between the two cities is now possible.

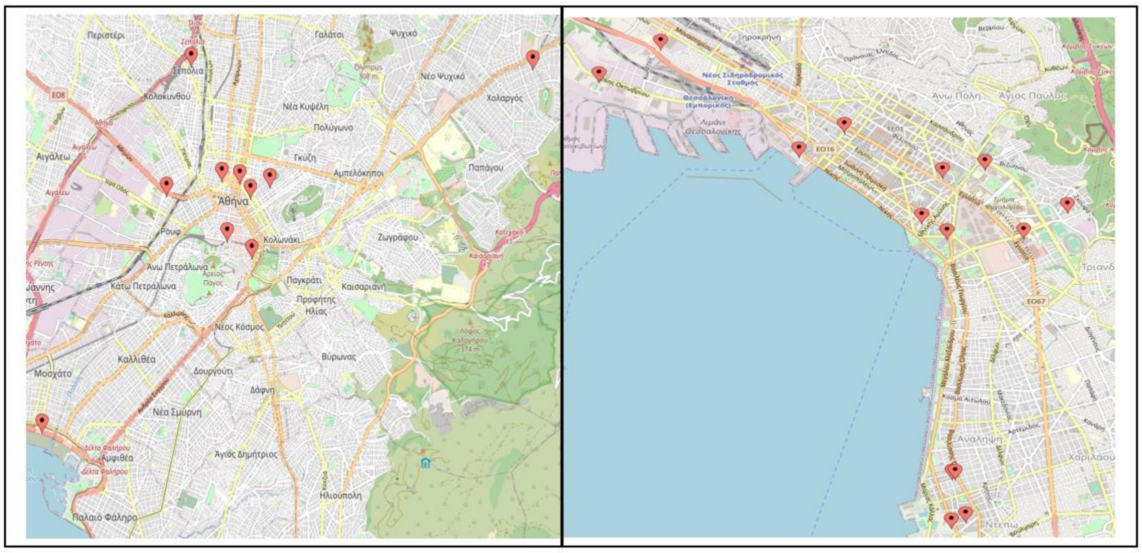


Figure 8: Sensor location in Athens (left) and Thessaloniki (right).

3.3.3 Feature Engineering and Selection

The dataset comprises data collected between 01.08.2019 and 31.08.2020, including time periods with movement and transportation restrictions due to COVID-19. The Greek government took the following measures to hinder the spread of COVID-19:

– On March 23, 2020 they implemented a general lockdown and introduced an SMS based system to provide movement permits only within regional units.

– On May 4, 2020 they abolished the SMS system but preserved regional restrictions regarding transportation.

– On May 18, 2020 they permitted free and unconditional movement and transportation.

Consequently, based on movement and transportation restrictions due to COVID-19, we introduced a new feature, named “status”, and our research focused on three time periods of equal number of values (56 days):

– August 1, 2019 to March 22, 2020, the pre-lockdown period, with unrestricted transportation.

– March 23, 2020 to May 17, 2020, the lockdown period, with restrictions on transportation.

– May 18, 2020 to August 31, 2020, the post-lockdown period, with unrestricted transportation.

The aforementioned information is depicted in Figure 9. The red lines signify the beginning and the end of the government restriction measures. It is obvious that restrictions and public concerns over COVID-19 had a huge impact on the Jam Factor. It is also important to mention that there is a dramatic drop in the Jam Factor a few days before the implementation of the movement and transportation restrictions. This can be explained by the fact that coffee shops, bars and shopping centers were forced to close on March 14, 2020, exactly where the black line is. We can additionally observe that the gradual increase in the Jam Factor after the COVID-19 period, actually begins from the green line, which signifies the day that the SMS system was abolished by the government (May 4, 2020), and movement became easier.

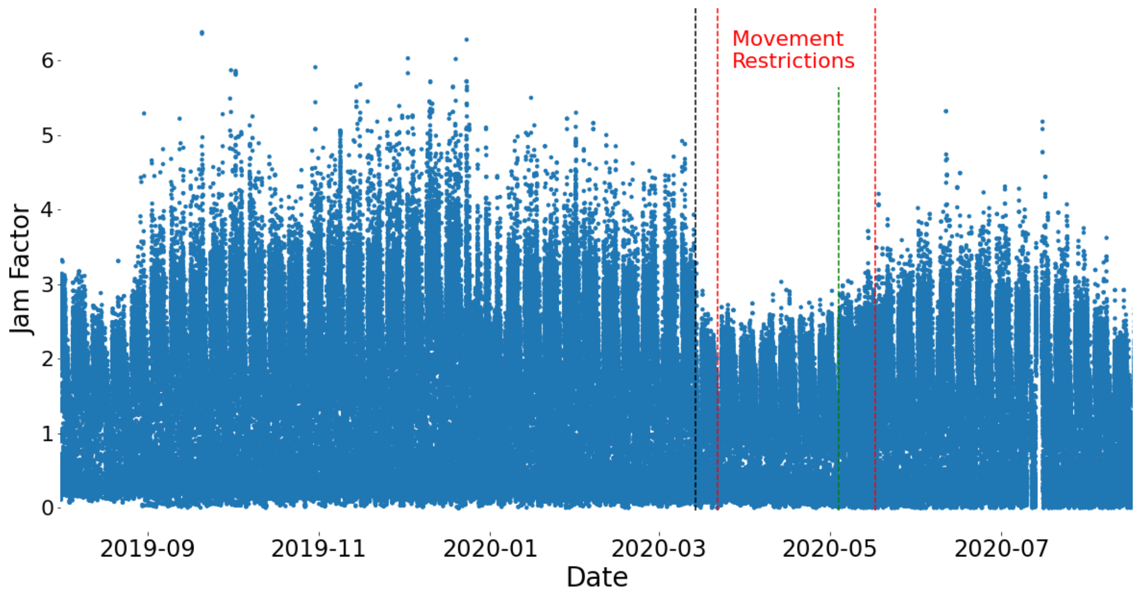


Figure 9: Jam Factor during 07.2019-08.2020.

Another special feature that was incorporated in our research was derived from the segregation of the days examined into weekdays and weekends. Figure 10 illustrates a significant drop of the Jam Factor during weekends, which deepens on Sundays. Thus, a new Boolean attribute was created, to examine the effect that weekends have on predicting the Jam Factor.

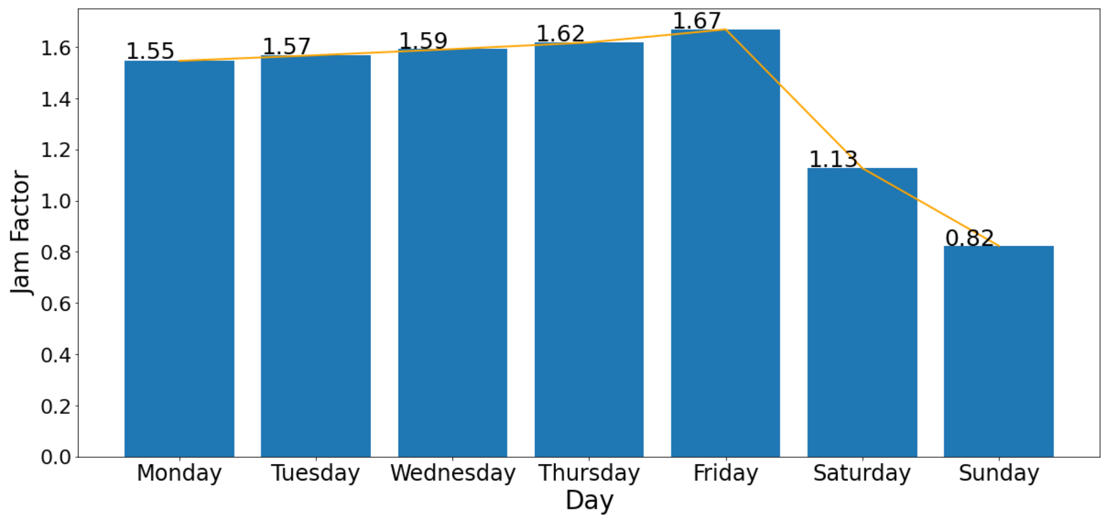


Figure 10: Average Jam Factor per Day.

The feature “wind direction” was transformed from numerical (degrees) into categorical (cardinal points). To convert degrees to cardinal points, compass was divided into 4 sectors of 90 degrees each and represented by a dummy variable. For instance, if “wind direction” ranges between 315^o and 45^o is mapped to “North”, if ranges between 45^o and 135^o is mapped to “East”, if ranges between 135^o and 215^o is mapped to “East” else to “West”.

Additionally, recursive feature elimination was implemented to select the optimal subset of features. The least important features were wind direction cardinal points and were removed. According to Figure 11, “weekends” have the highest importance score by far from the rest. Then, “uv” and “status” follow with equally high importance scores. The remaining features that contribute to model performance, such as temperature and atmospheric pressure, refer to weather conditions. These features are related with weather phenomena that severely affect the traffic system. For instance, rain and snow often limit movement and cause traffic congestions. Interestingly, grouping together features related to weather conditions, reaches feature importance score at 0.5. Thus, we could support the significance of weather information for our classification problem. More details about feature correlation and recursive feature elimination are given in Appendix B.

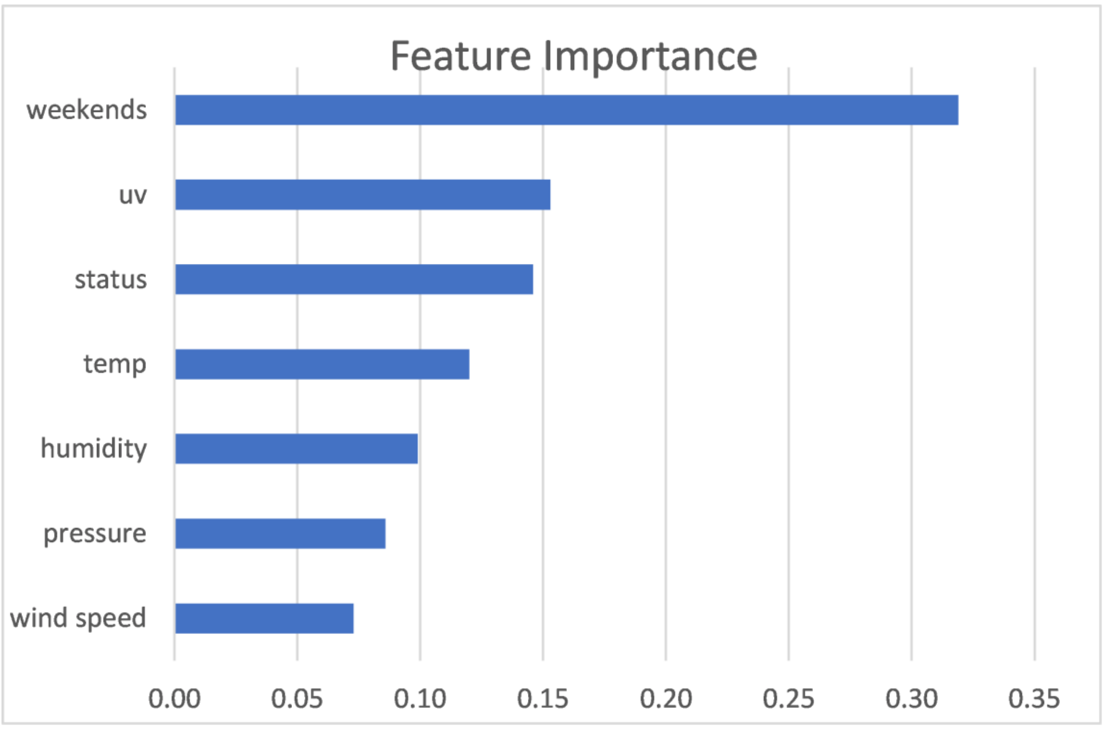


Figure 11: Feature Importance using Recursive Feature Elimination.

3.3.4 Exploratory Analysis

Considering the importance of the temperature feature “temp”, we investigated how it ranges every month. As expected, Figure 12 indicates that temperature is maximized during summer months, fluctuates at the same level in Autumn and Spring and significantly decreases during winter months. Overall, it seems that there is high variance between months and this could enhance model performance.

Another important feature according to feature importance analysis is humidity, which highly related to rainfall and therefore affects traffic conditions. Figure 13 represents the average percentage of relative humidity for each month of the year. November has the highest percentage of relative humidity and August the lowest. We could claim from May to September humidity is limited to low levels and from October to April reaches high levels.

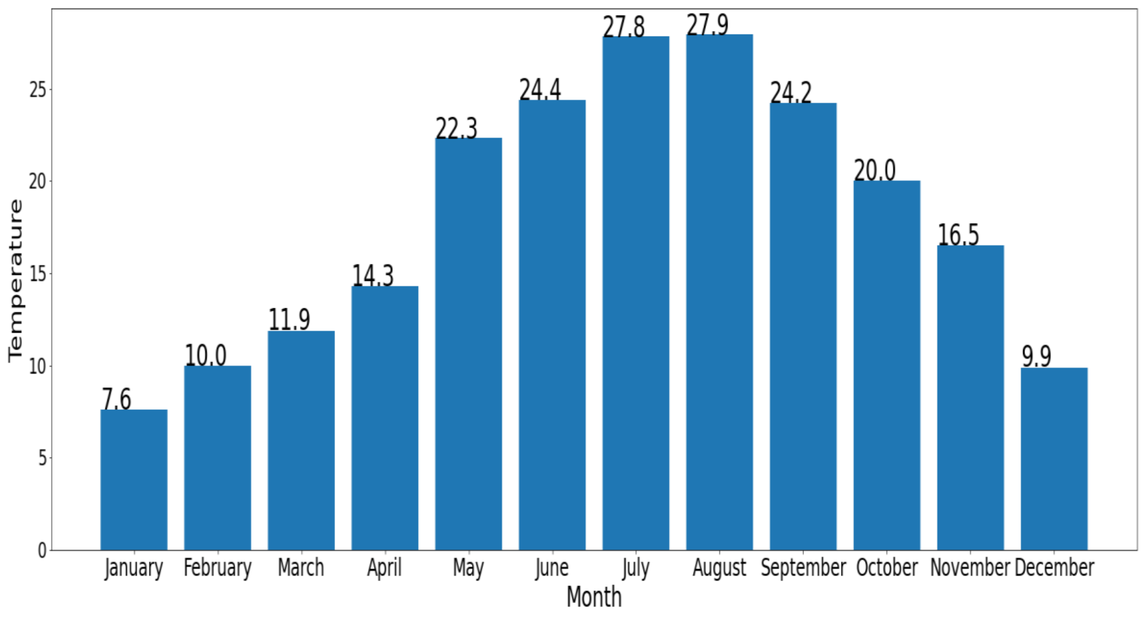


Figure 12: Temperature monthly average.

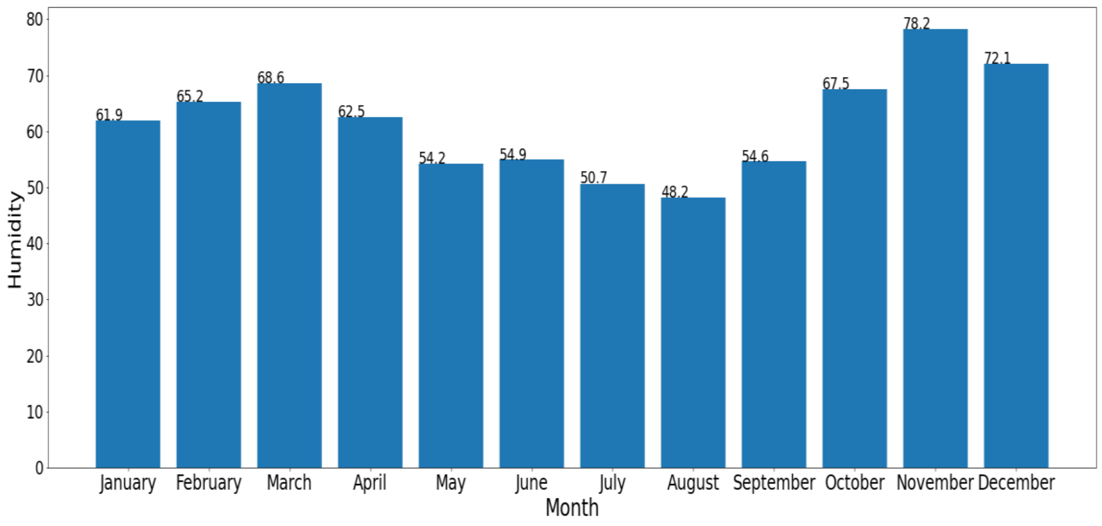


Figure 13: Humidity monthly average

Extending previous research efforts [1], we added a new time interval to the existing ones. The interval is named Noon and includes data from sensors between 12:00 and 15:00. This time interval was deemed to be important for our research. As seen in Figure 14, interestingly, the Jam Factor during Noon is slightly higher than any other hour within the day.

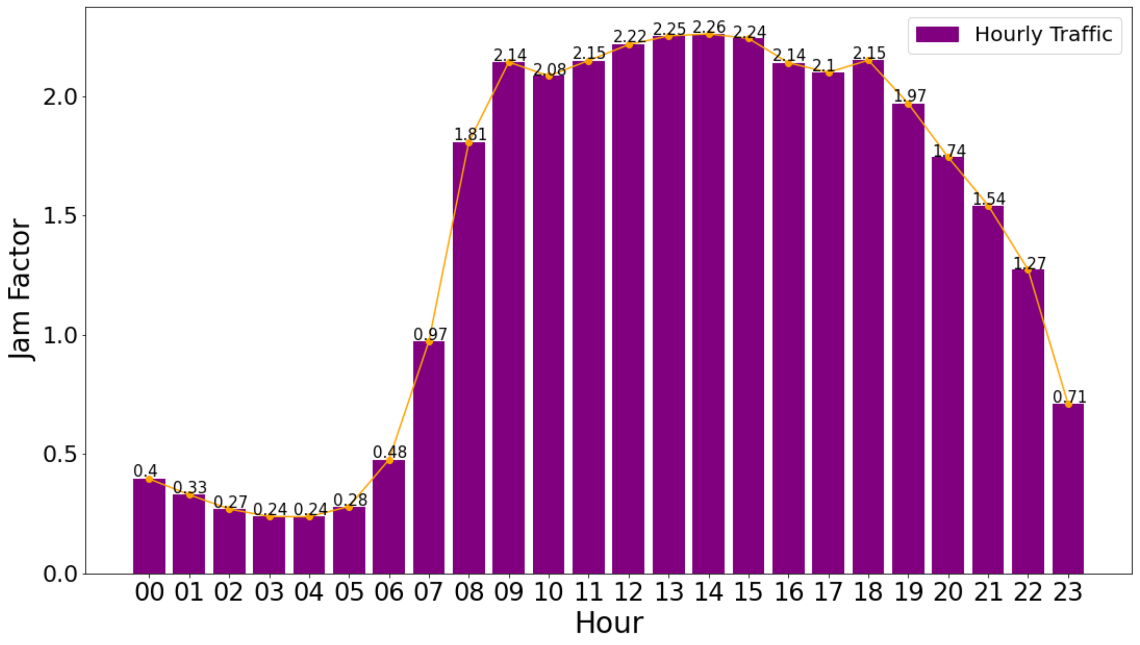


Figure 14: Jam Factor Hourly Average.

The time periods when people drive to their jobs and when they drive back home are generally considered as rush hour. Usually rush hours are before 12:00 and after 15:00. However, in big cities like Athens and Thessaloniki, there is a high volume of traffic even during noon, when people need to move through the city for tasks like school runs, shopping, business meetings, urban freight transport etc. Thus, the time interval called Noon, can also be regarded as rush hour. Additionally, rush hours often differ during the year. More details about Jam Factor hourly average values for each day of the week are given in Appendix D.

It has already been mentioned that the Jam Factor and our predictions were influenced by the outbreak of COVID-19. Therefore, it is essential to investigate the impact on each time interval, as Figure 15 depicts. As already mentioned, the red lines indicate the beginning and the end of the lockdown period, the black line signifies when Ho.Re.Ca and retail shops shut down and the green line denotes the abolishment of SMS system. For all periods, it is clear that traffic conditions were heavier at Noon and in the Afternoon and lower traffic levels were observed in the Morning and in the Evening. Remarkably, during lockdown traffic levels were decreased in all time intervals, as a result the gaps between them were diminished. After lockdown, the gaps between time intervals reach pre-lockdown levels. We can detect the valleys every weekend, so weekly seasonality is identified. Surprisingly, in the period oriented between the green line and the second red line, we can observe that on weekend mornings traffic is boosted, although movement restrictions were still present. The source code of all graphs is given in Appendix D.

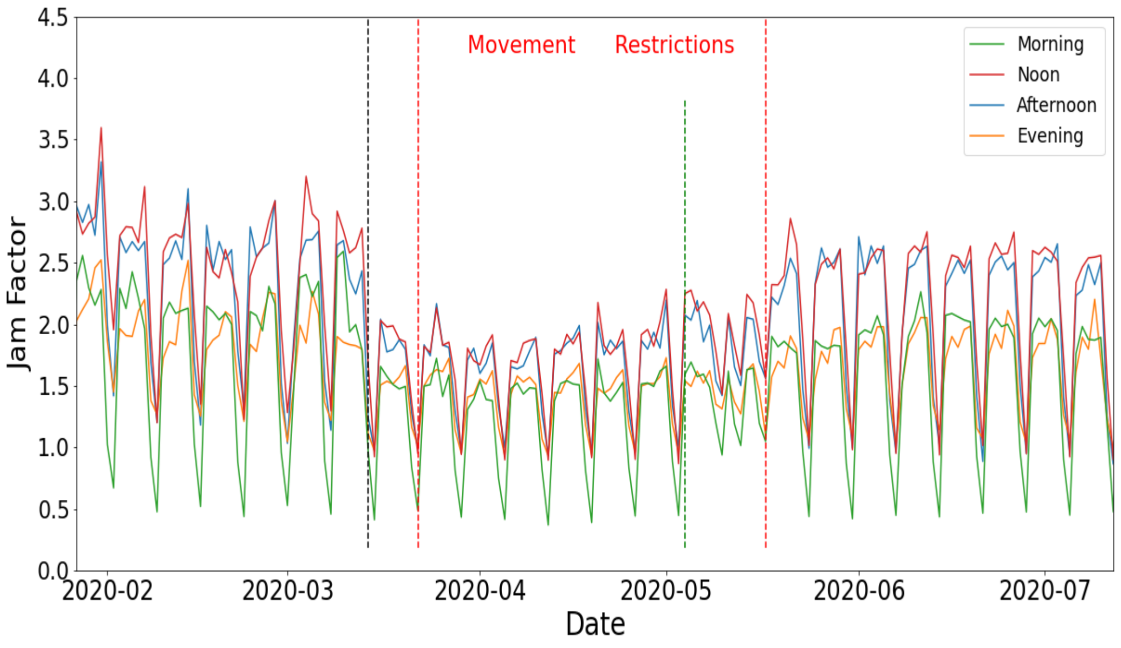


Figure 15: Jam Factor before, throughout and after lockdown, featuring all time intervals.

4 Experiments 2-classes

This section presents 2-classes experiments we conducted on our data, starting with the baseline, extended by model improvements involving the introduction of an attribute related to the type of day (weekday vs. weekend) the type of period related to COVID-19 (pre- during and post-lockdown) as well as a combination of both. It also presents experimental results related to 3 or 4 month splits of the data. It concludes with a comparison between results achieved for Athens vs. Thessaloniki.

4.1 Baseline and Model Evolution 2-classes

As described in Chapter 3 both types of generalization (test set and validation set) have insignificant variation and almost identical trends. Using the validation set and test set display minor differences, with the validation set producing slightly better results in terms of accuracy. Two classifiers were used Random Forest and Logistic Regression, because these classifiers were recommended the most in the literature review. Grid search was executed for both, in order to specify the optimal parameters of each given model. Figure 16 shows the accuracy achieved by Random Forest (RF-2) and Logistic Regression (LR-2) for the 2-class problem. For the 2-classes the two classifiers consistently achieved accuracy over 80% with best results for Afternoon. RF-2 with 10-fold cross-validation achieved slightly better results than LR-2.

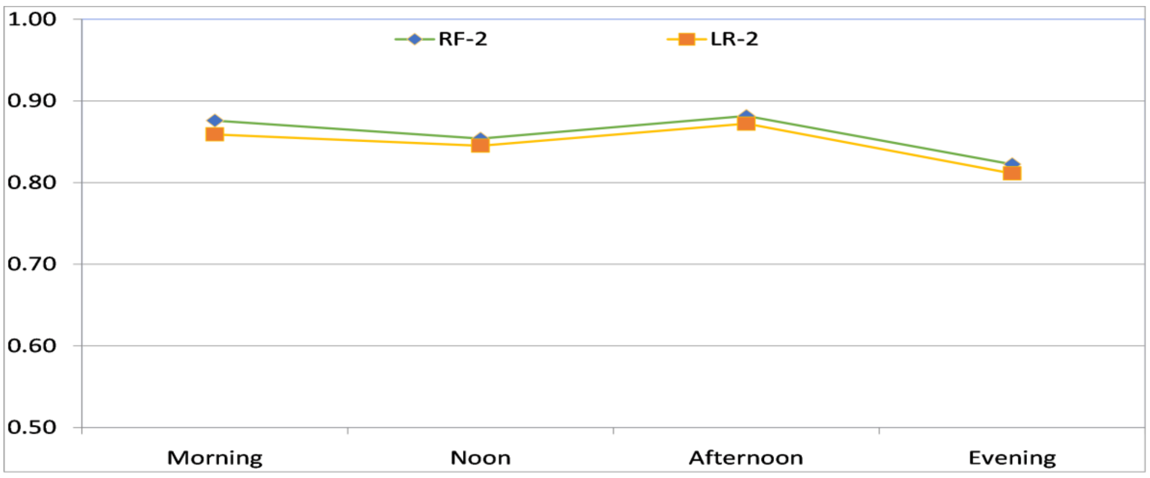


Figure 16: Comparing Logistic Regression and Random Forest classifiers in 2-classes experiments.

Thus, all remaining figures in the main body of this chapter refer to results achieved using a validation set, for models trained with Random Forest and for target attribute with two classes.

The baseline model is regarded as the model developed by Christantonis et al [1]. It was not trained with special circumstances, such as the COVID-19 pandemic. Thus, when evaluated on the whole dataset and its accuracy fluctuated around 64%, as can be seen in the first set of four bars, labeled "BASELINE" in Figure 17.

With the incorporation of the feature "status", related to the lockdown due to COVID-19 (having three distinct values: pre-lockdown, lockdown, post-lockdown) into the original model, a slight improvement was observed, achieving accuracy close to 67% and over 70% for the Evening time interval. This can be observed in the second set of four bars labeled "STATUS" in Figure 17.

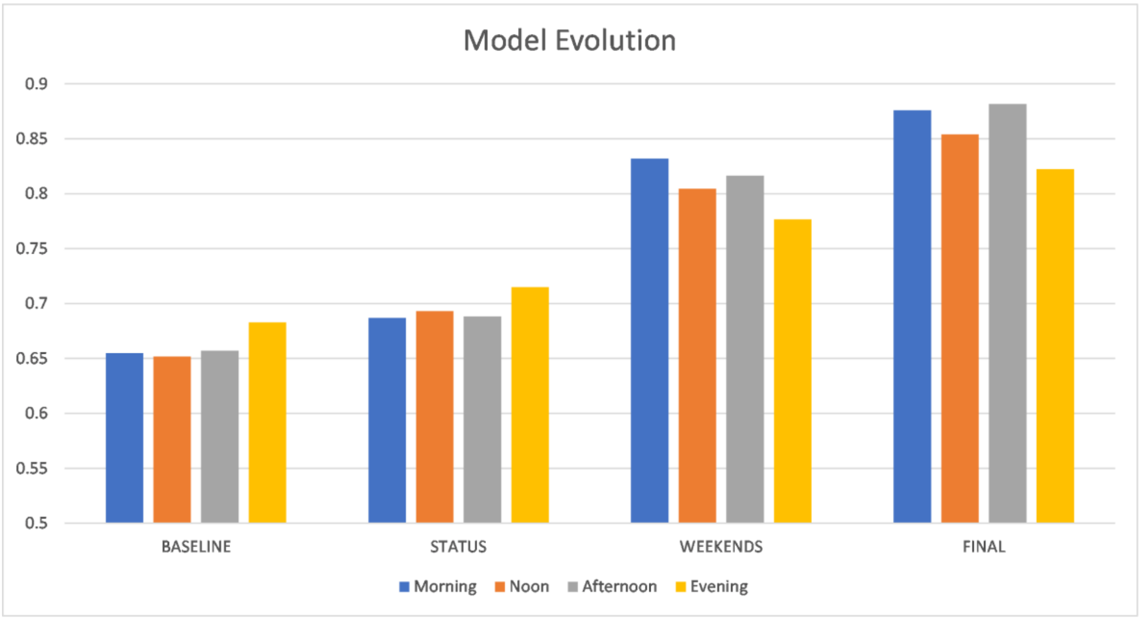


Figure 17: Model evolution featuring all time intervals.

Moreover, with the incorporation into the baseline model of the feature "weekends", which categorizes days into weekdays and weekends, the new model achieves significant higher accuracy, even above 80% for Morning, as can be seen in the third set of four bars labeled "WEEKENDS", in Figure 17.

Finally, both these two features, i.e. status and weekends are used conjointly and form the final model, illustrated by the fourth set of four bars in Figure 17, labeled "FINAL". Compared to the baseline approach, an undoubted improvement was achieved on accuracy, which increased to over 80% for all periods and approximated 90% for certain day intervals. Additionally, it is worth mentioning that the feature “status” contributes more to the classification task when combined with the feature “weekends” rather than combined just with the original model. “Weekends” actually offered higher improvement than that introduced by “status”.

Another observation derived from Figure 17 is that the two new features, namely “status” and “weekends” have a different impact on each time interval. In particular, “status” has a homogenous impact on each time interval, by slightly improving accuracy. On the other hand, the feature “weekends” seems to affect evening time intervals less than it affects the other three. It is observed that the Evening time interval has the best accuracy in the baseline model but has the worst accuracy when the feature “weekends” is incorporated into the model. One possible explanation for that observation lies in the fact that evenings’ traffic congestion is related with transportation for recreation rather than for business purposes. Since weekends are rest days, it is normal for traffic congestion to be reduced during morning, noon and afternoon on those days, but this does not reflect the evening traffic loads. Therefore, while the prediction about the three other time intervals is heavily affected by the feature “weekends”, this is not the case about Evening time interval.

This assumption is further confirmed by Figure 18, where it is obvious that, regarding the Jam Factor, the difference between weekdays and weekends is significantly lower during the evenings (19:00 - 22:00) than any other time interval.

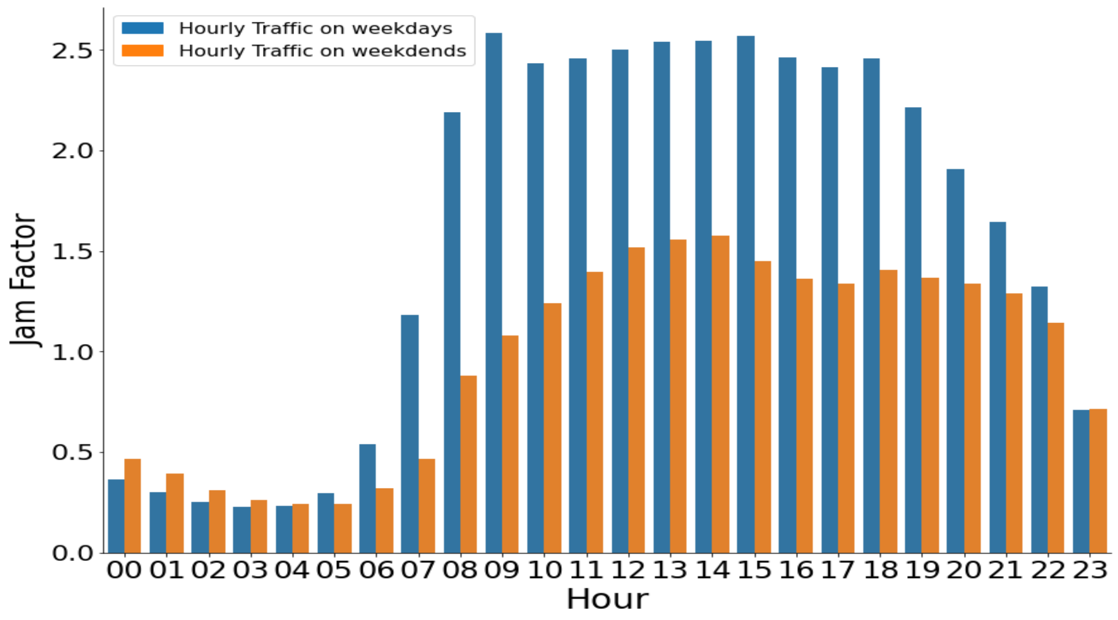


Figure 18: Average Hourly Jam Factor on weekdays vs. weekends.

4.2 3-month split 2-classes

By examining the dataset segregated using 3-month periods, it emerges that Morning and Afternoon are more predictable intervals than Noon and Evening. The model trained on the whole dataset performs better than the other splits except for one: the best performing training period (01.05.20-31.07.20) contains both movement restrictions and free movement, which seems to be a key factor in data selection. Those observations are depicted in Figure 19.

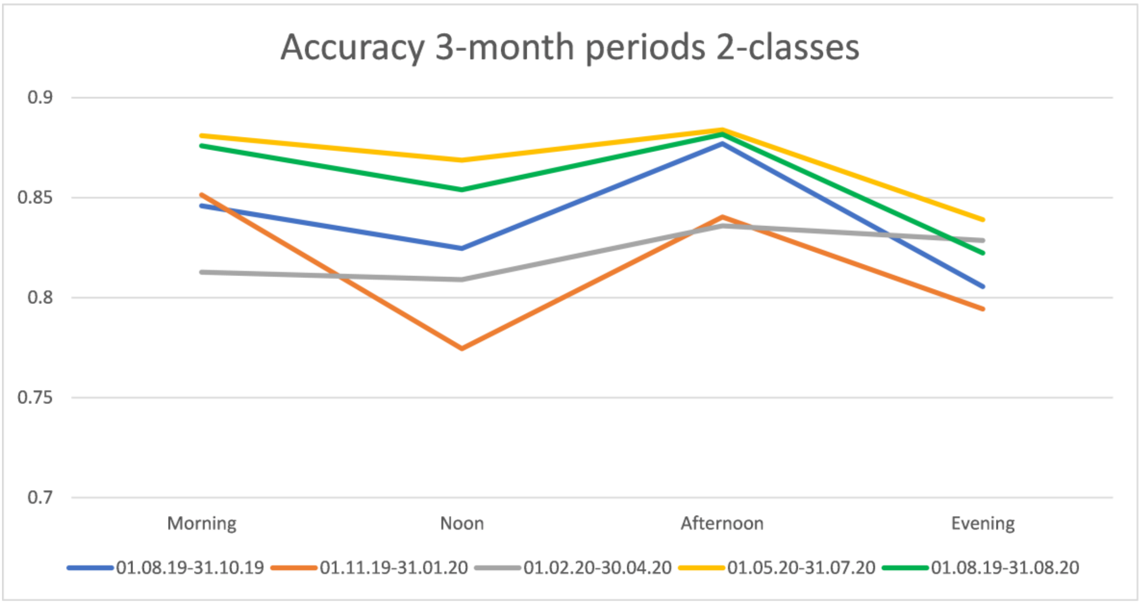


Figure 19: Accuracy for 3-month periods and 2-classes.

4.3 4-month split 2-classes

The model is also trained and tested based on 4-month periods and compared to the training case of the whole dataset. The 3rd 4-month split (01.04.20-31.07.20) outperforms all others for every time slot. Observing Figure 19 and comparing it to Figure 20, the same patterns emerge as in the case of 3-month periods. For instance, the model performs better in the Morning and in the Afternoon. The model trained with the whole dataset (01.08.19-31.08.20) achieves higher accuracy (86%) than the first two 4-months intervals (84%-81%) but is being outperformed (87%) by the last one (01.04.20-31.07.20). This remark also applies to Figure 19.

It seems that a range containing both normal and special (COVID-19) traffic conditions is an optimal training period for the model. Moreover, since in this case there is a subset that outperforms the whole dataset, we assume that it is recommended to include normal and special traffic conditions to our model. In any case, more data records do not guarantee higher accuracy.

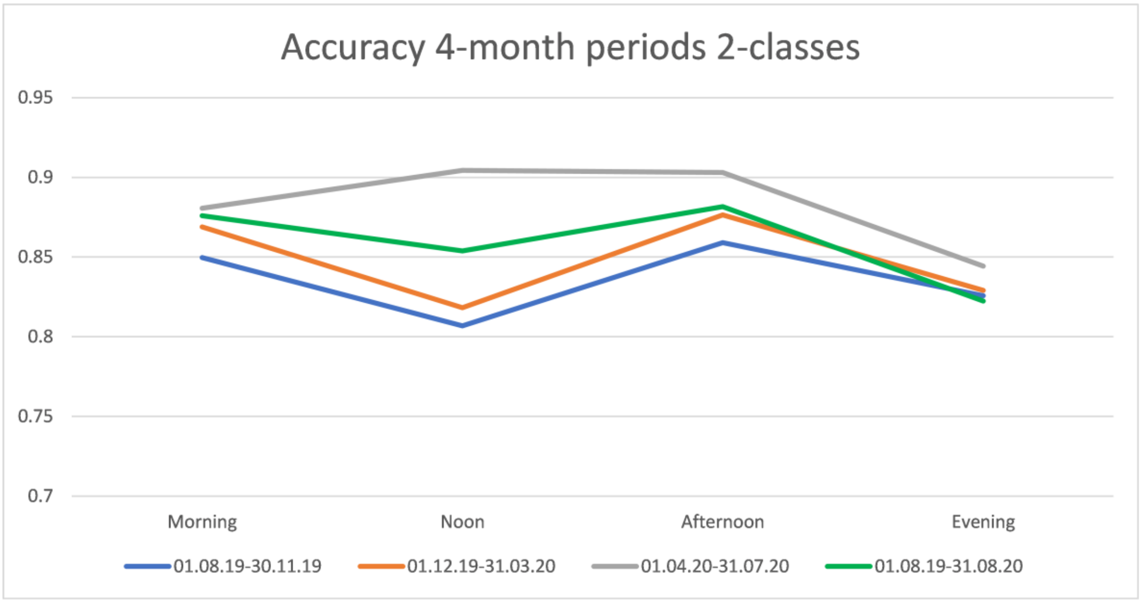


Figure 20: Performance on 4-month periods for 2-classes.

4.4 COVID-19 2-classes

It has already been mentioned that the Jam Factor and our predictions were influenced by the outbreak of COVID-19 (see Figure 9). To investigate that influence, the dataset was initially split into three periods of equal size (pre-lockdown, lockdown, post-lockdown) and the results were compared with those for the whole dataset. Later, we merged those intervals and added one more time period, from 27.01.2020 to 12.07.2020, which proved to be revealing for this research. Results are presented in Figure 21, after aggregating sensor outcomes for every time interval during the day.

Lockdown and post-lockdown periods turned out to be the most difficult to predict, demonstrating that COVID-19 has caused unexpected changes to every-day life. The pre-lockdown split appeared to be the easiest to predict. As previously exhibited, in pre-lockdown days, traffic conditions were normal. Afterwards, during the lockdown, traffic dropped and then, in post-lockdown days, rose but not up to normal levels. Subsequently, it could be assumed that accuracy is affected by traffic conditions (i.e. movement restrictions) and it is easier to predict traffic during normal traffic conditions.

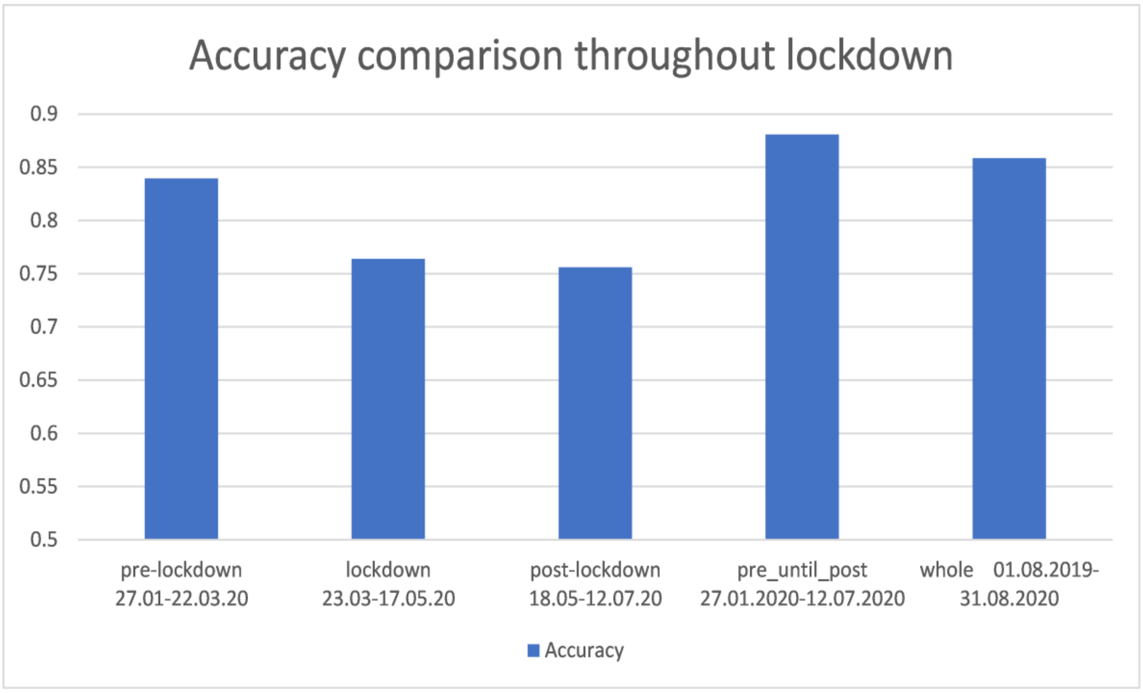


Figure 21: Accuracy comparison related to the lockdown.

Surprisingly, though, the whole dataset produced better results than any time intervals. The explanation of this ostensible abnormality is given by inspecting the time interval that was added later. It is clear that the model trained from 27.01.2020 until 12.07.2020 achieved the highest performance with accuracy close to 90%. This can be explained by the fact that the feature “status”, which is one of the most important features of the model, during this period takes exactly equal number of values from each category. Consequently, there is a perfect balance on the values of this feature, and this contributes to the model training and as a result to the classification ability. Balance between days with restrictions and days without restrictions is absent for all three time periods (pre-lockdown, lockdown, post-lockdown), so “status” is constant and does not have a big contribution to the baseline model.

The Figure 22 is an extension of Figure 21, with the addition of time intervals. As already noted, Morning and Afternoon traffic is predicted with higher accuracy for most of the cases, except for the lockdown period, when Noon and Evening traffic has higher predictability.

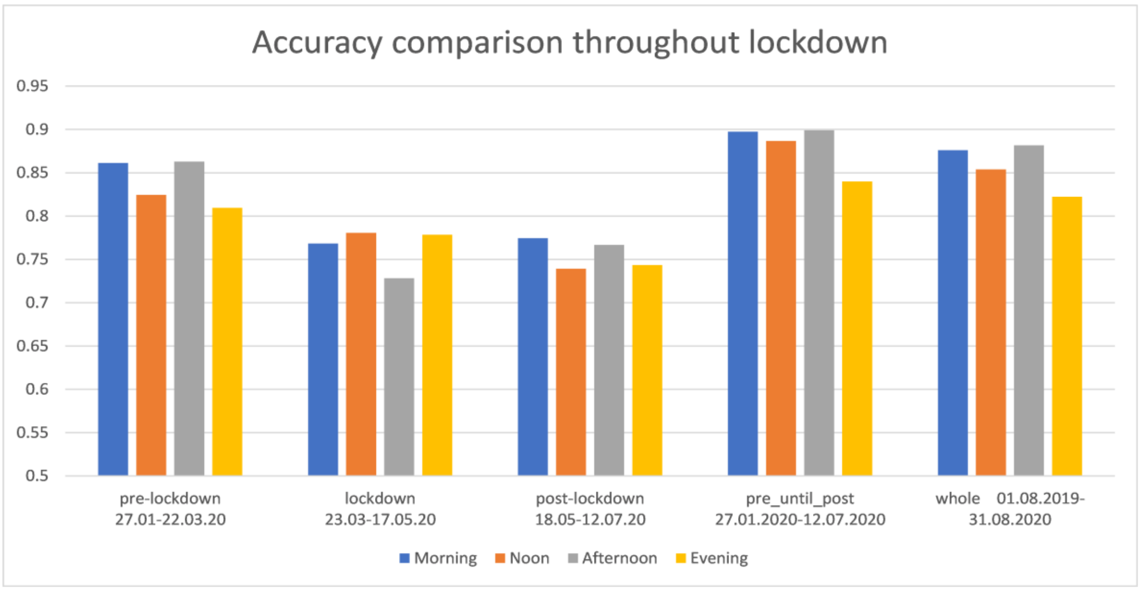


Figure 22: Accuracy comparison related to lockdown periods, featuring all time intervals.

4.5 Athens-Thessaloniki comparison 2-classes

Finally, we compare and contrast experimental results for Athens and Thessaloniki. Table 1 presents the 2-classes experiments Athens outperforms Thessaloniki in all time intervals, having the model trained on the whole dataset. On average accuracy is 2.25% higher in Athens. The difference in accuracy rises in the Morning and converges in the Evening. We should also highlight that the model performs better in both cities in the Morning and in the Afternoon than the rest intervals.

Table 1: Accuracy comparison between Athens and Thessaloniki for 2-class experiments for the whole dataset.

Time Interval	Athens	Thessaloniki
Morning	89.52%	86.39%
Noon	87.26%	84.21%
Afternoon	89.36%	87.41%
Evening	82.79%	81.89%
Average	87.23%	84.98%

Furthermore, Figure 23 shows the fluctuation of the Jam Factor, during the whole 13-month period for which data were collected. Both cities hit a peak on July 15, 2020, which incidentally was the day when direct flights from Great Britain to Greece were resumed.

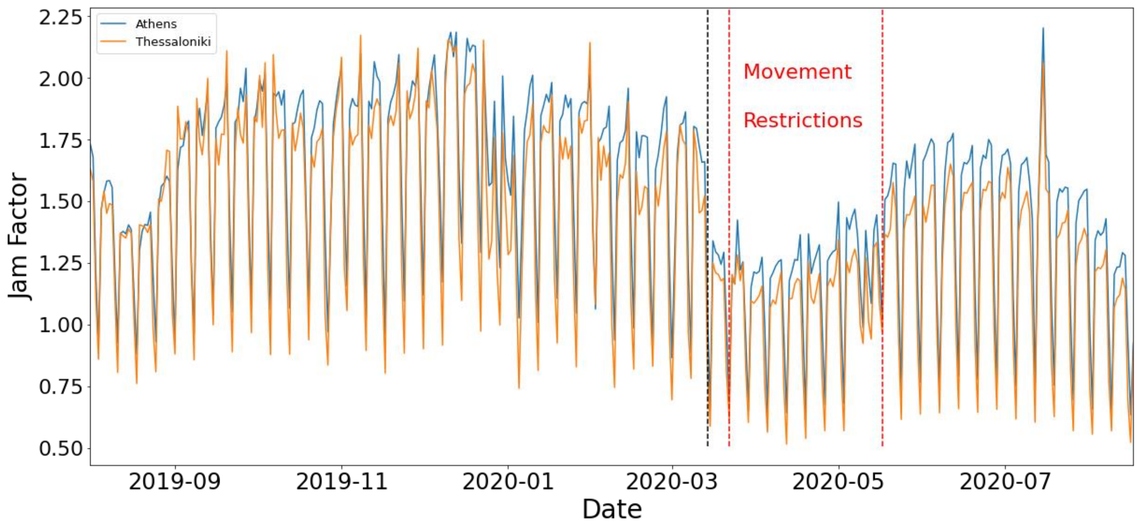


Figure 23: Average Jam Factor in Athens and Thessaloniki during 07.2019-08.2020.

Focusing closer to the Jam Factor for the period between 27.01.2020 and 12.07.2020 in Figure 24, one can observe the similar distribution for traffic in Athens and Thessaloniki. It is obvious that Athens has heavier traffic conditions than Thessaloniki. It is worth mentioning that the gap between the two cities is higher during the days with free movement after lockdown compared to other periods. One reason behind that might be road construction work in the city center of Athens. In one case, for Panepistimiou Str., travel time almost doubled. This street is close to sensors 18, 25, 31 and surely it affected traffic congestion. The beginning of the construction works was on May 21, 2020 and the end was on June 14, 2020.

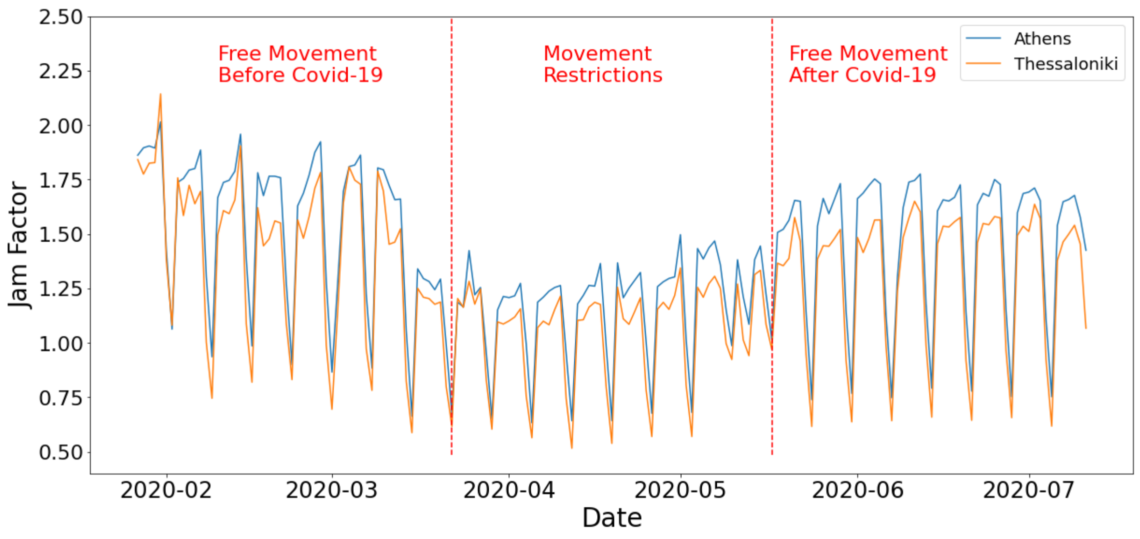


Figure 24: Average Jam Factor in Athens and Thessaloniki before, during and after the lockdown.

As far as accuracy is concerned, no major differences were observed between the two cities. Figure 25 showcases that the accuracy was slightly better for Thessaloniki during pre-lockdown and lockdown periods, and better for Athens during the post-lockdown era. Evidently, pre-lockdown was the best period for both cities, regarding accuracy. For each city, as traffic drops due to movement restrictions, accuracy decreases as well. Interestingly accuracy does not improve enough post-lockdown to reach pre-lockdown levels, similarly to traffic levels. Especially in Thessaloniki accuracy post-lockdown slightly falls, although traffic levels rise.

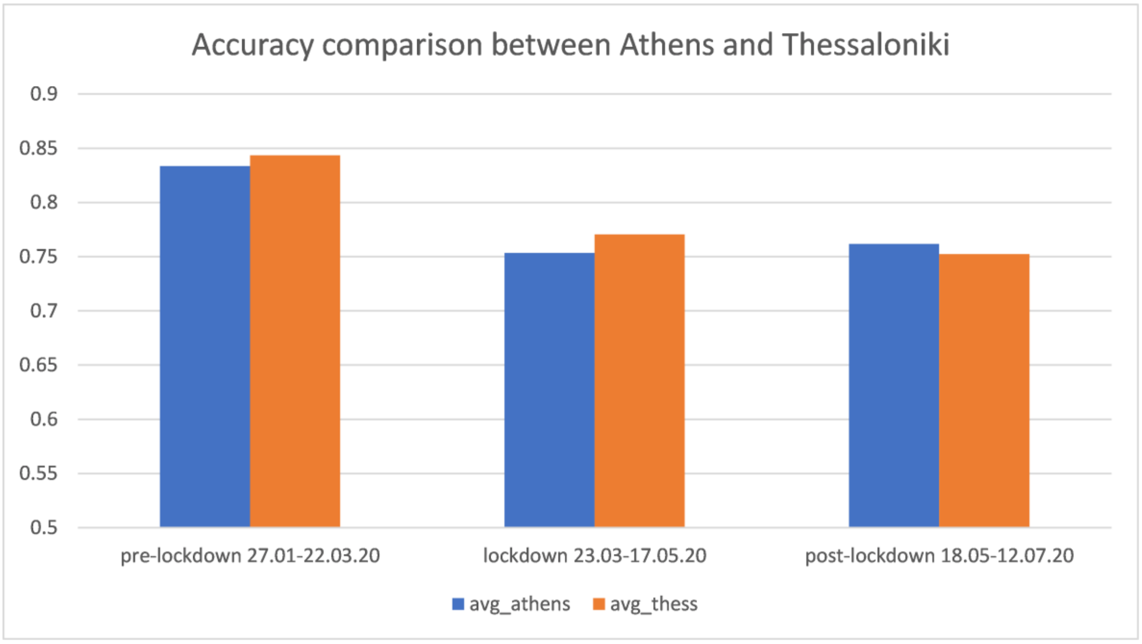


Figure 25: Accuracy comparison between Athens and Thessaloniki before, during and after lockdown.

Focusing on time intervals during the day, Figure 26 depicts that for Athens the best time interval in terms of accuracy is Afternoon, at any stage; before, during or after the COVID-19 restrictions. On the other hand, in Thessaloniki, the best predicted time interval is Morning, except for the lockdown period. It must also be noted that the worst predicted time interval is Evening for Athens and Afternoon for Thessaloniki.

The source code of the 2-classes experiments is given in Appendix C. The detailed results of each sensor are presented in Appendix E.

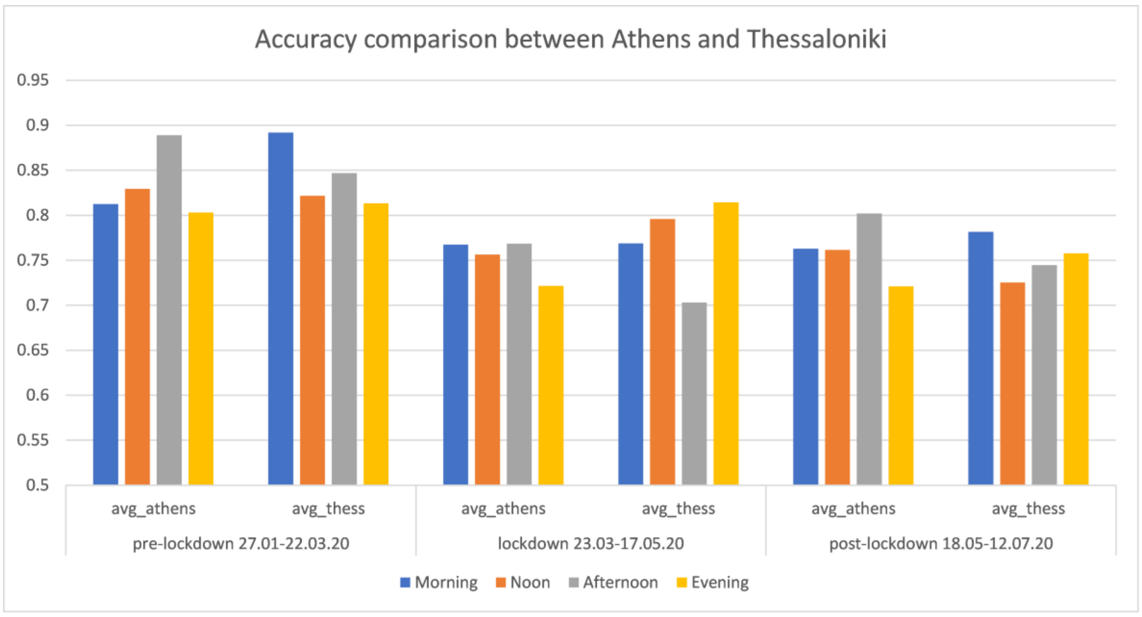


Figure 26: Accuracy comparison between Athens and Thessaloniki before, during and after lockdown, featuring all time intervals.

5 Experiments 3-classes

This section presents 3-classes experiments we performed on our data, beginning with the baseline, extended by model changes involving the inclusion of an attribute related to the type of day (weekday vs. weekend), the type of period related to COVID-19 (pre-, during and post-lockdown) as well as a mixture of both. It also presents experimental results findings relating to data splits of 3 or 4 months. Then, follows a contrast of the outcomes of Athens vs. Thessaloniki. The detailed results of each sensor are presented in Appendix F.

5.1 Baseline and Model Evolution 3-classes

Similarly to the 2-class problem, two classifiers were used: Random Forest and Logistic Regression. Grid search was executed for both, in order to specify the optimal parameters of each given model. Figure 27 shows the accuracy achieved by Random Forest (RF-3) and Logistic Regression (LR-3) for the 3-class problems. For the 3-classes the model achieved accuracy over 80% in the Morning, close to 75% for Noon and Afternoon and lower than 70% in the Evening. Random Forest with 10-fold cross-validation achieved slightly better results than Logistic Regression.

Thus, all remaining figures in the main body of this chapter refer to results achieved using a validation set, for models trained with Random Forest and for target attribute with three classes.

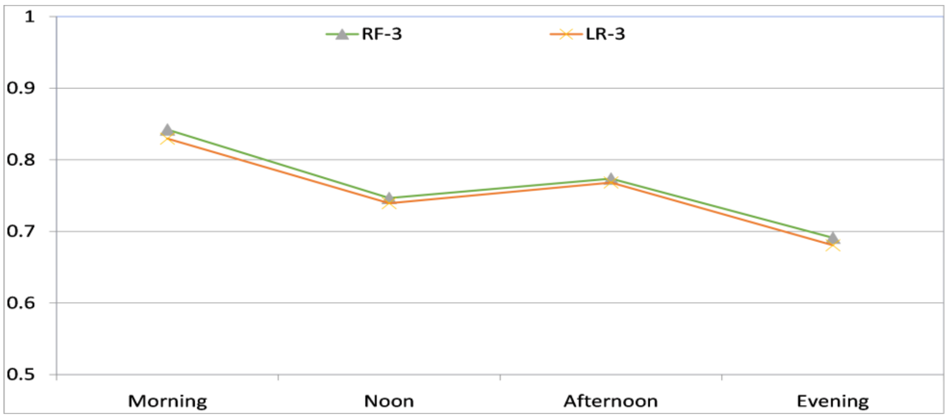


Figure 27: Comparing Logistic Regression and Random Forest classifiers in 3-classes experiments.

Respectively, the model developed by Christantonis et al. [1] is called the baseline model for the 3-classes experiments. Special cases, such as the COVID-19 pandemic, were not included. Hence, when assessed on the whole dataset, performance was lower than 55% for all time intervals as Figure 28 showcases.

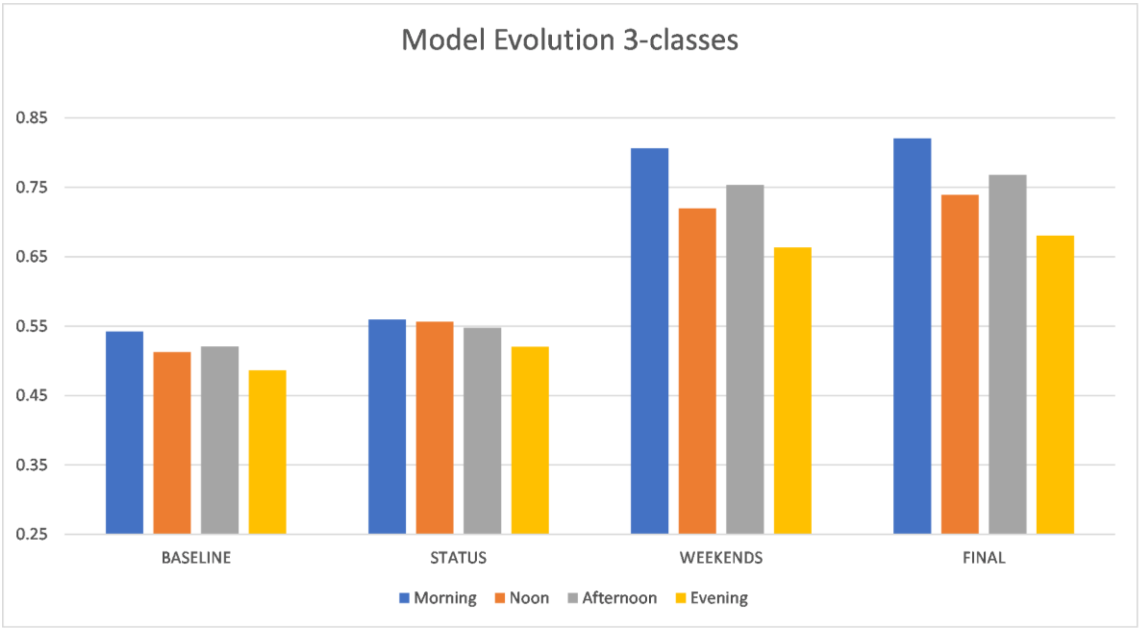


Figure 28: Model evolution for 3-classes experiments, featuring all time intervals.

This model was extended with the inclusion of the feature “status”, related to the lockdown movement restrictions. A small improvement was observed on the performance of the model “STATUS”. Accuracy in the Morning, Noon and Afternoon interval is almost equal at 55%, with Evening being lower.

Another approach, led to the introduction of the feature “weekends” into the baseline model, forming the “WEEKENDS” model. In comparison with the baseline model, it achieved accuracy close to 80% in the Morning interval and even above 65% for the rest. Comparing it to the baseline model, we observe a homogenous impact on each time interval. For instance, Morning remained the most predictable interval and Evening the worst performing.

The final edition of the 3-classes model incorporates as features both “weekends” and “status”. As can be seen in the fourth set of bars in Figure 28, the “FINAL” model has clearly outperformed the baseline model. It is remarkable that “weekends” assisted more than “status” on model’s performance.

5.2 3-month split 3-classes

In another experiment, we split the dataset into on 3-month periods and trained the model with each of them. As Figure 29 showcases, the best performing model was trained with data from 01.02.20 until 30.04.20. Remarkably, it outperforms even the model trained on the whole dataset (01.08.19-31.08.20). The performance peaks in the Morning interval in all cases, then drops noticeably during Noon and collapses in the Evening. Despite that models are trained on periods with completely different characteristics, there is a similar pattern between the accuracy lines.

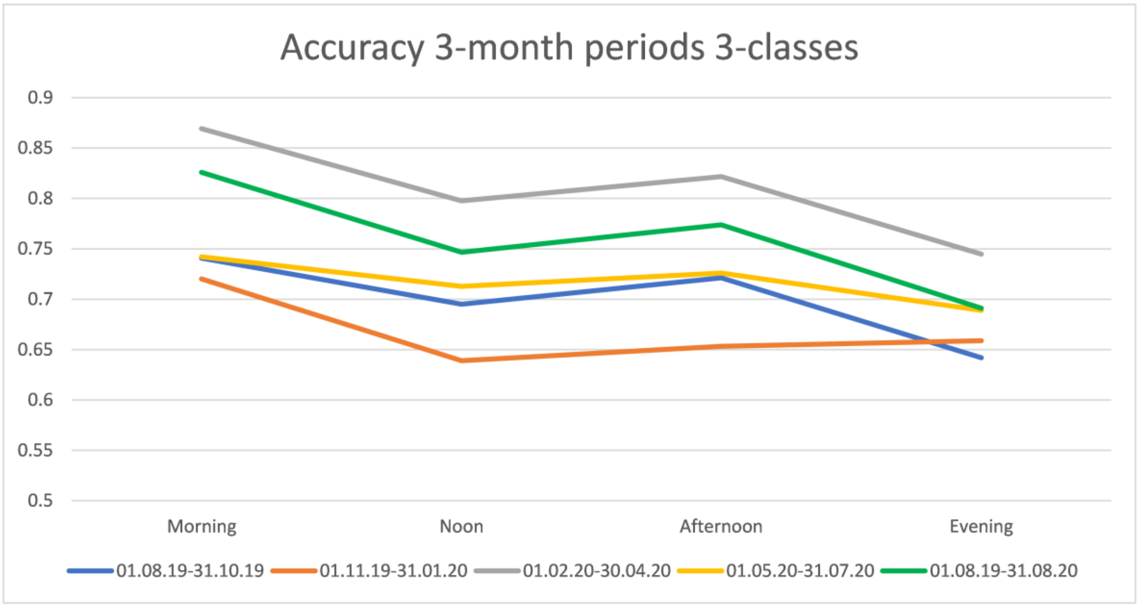


Figure 29: Results on 3-month periods for 3-class experiments.

5.3 4-month split 3-classes

Afterwards, the dataset was split into 4-month periods which were utilized for training the model conducting the 3-classes experiments. Observing Figure 30, the third 4-month period (01.04.20-31.07.20) achieved the highest performance among all other periods. Comparing it with the model trained on the whole dataset, it is obvious that the latter outperforms during the Morning interval, at Noon equal accuracy score is accomplished, while in the rest intervals underperforms.

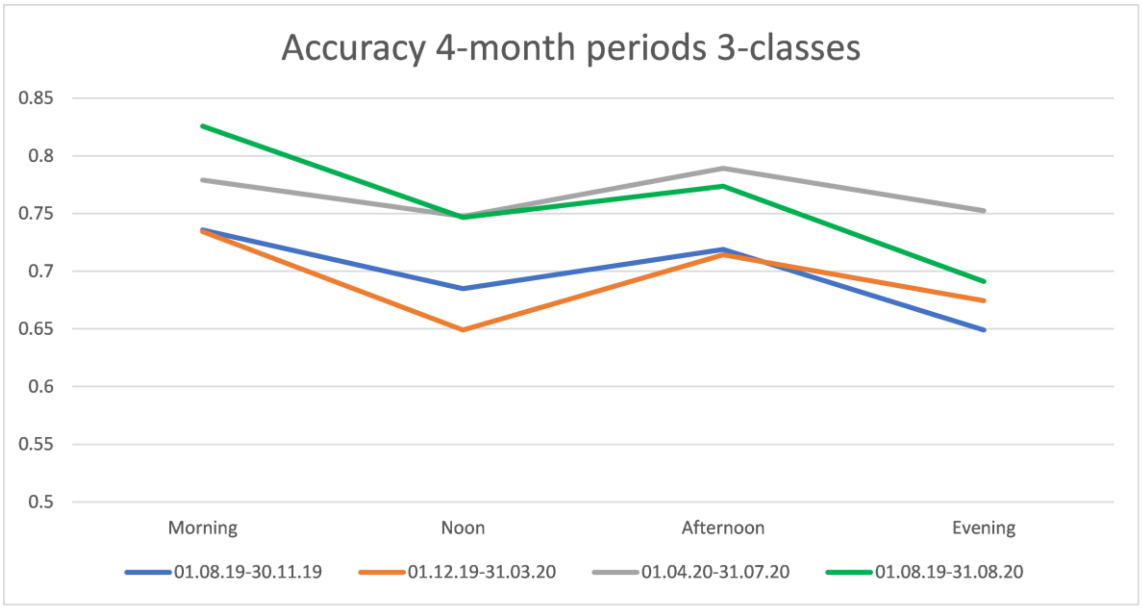


Figure 30: Results on 4-month periods for 3-class experiments.

5.4 COVID-19 3-classes

Afterwards, our research concentrated on the effect of COVID-19 pandemic into our 3-classes experiments. As already presented for the 2-classes experiments, our model was evaluated on periods of equal size. Each period before, throughout and after lockdown period lasts 56 days, as many as the lockdown period from 23.03.2020 to 17.05.2020. The merge of these periods is called “pre until post” period from 27.01.2020 to 12.07.2020 and was compared with results from the “whole” dataset. Figure 31 indicates that accuracy was marginally higher before lockdown (67%) than throughout (65%) and after (66%). Interestingly, as Figure 24 showcases, Jam Factor follows a similar pattern with accuracy. Specifically, traffic was at peak levels before lockdown, then during lockdown fell dramatically and after lockdown surged but not up to normal levels. Therefore, there is evidence that traffic levels affect model’s performance.

Additionally, the results based on the merged period “pre until post” reveal that accuracy rose up to 75%. Moreover, the whole dataset achieved marginally even better results, with accuracy at 75,9%. This suggests that model should be trained on data both with and without movement restrictions.

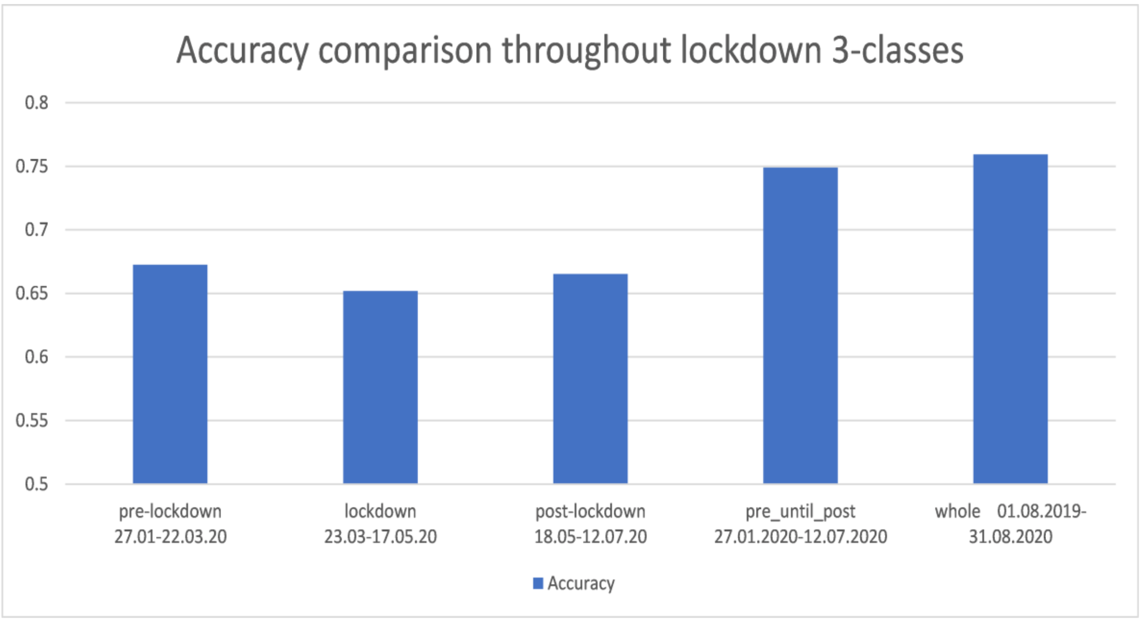


Figure 31: Accuracy comparison related to the lockdown for 3-class experiments.

Figure 32 focuses on each time interval of the day for the 3-classes experiments. For instance, it is clear that model performs better in the Morning and it is confirmed in all cases. In the second place of predictability stands Afternoon for most of the cases, except from the post lockdown period. Then, follow Noon and Evening with changes in the ranking, but for the "whole" model, Noon is more predictable (74.6%) than Evening (69.1%).

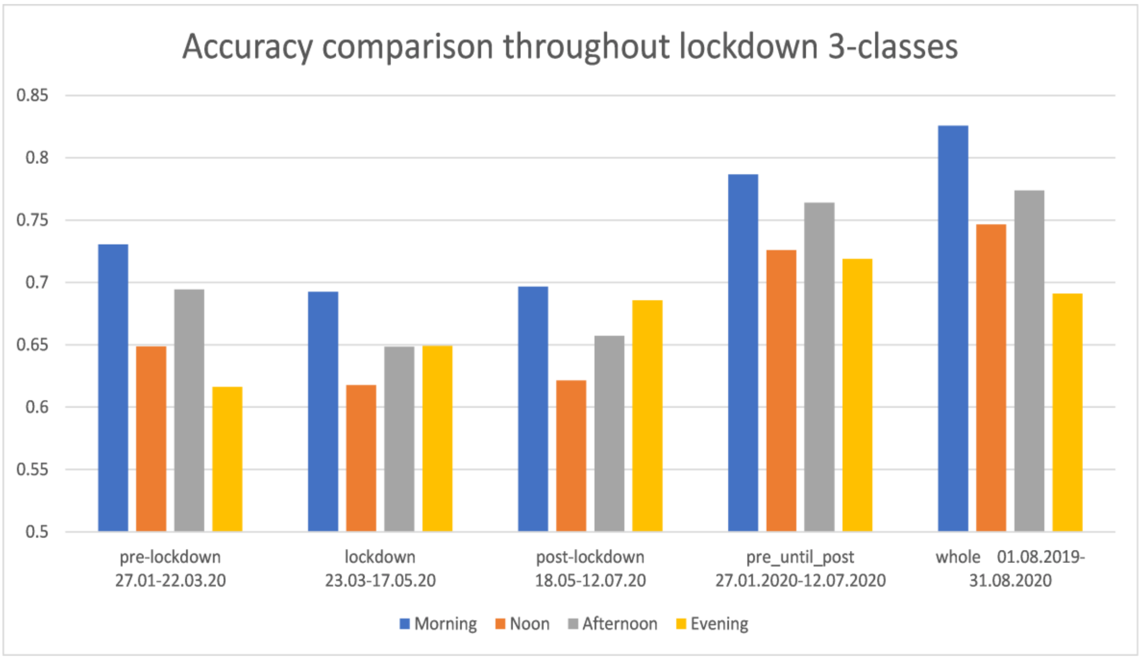
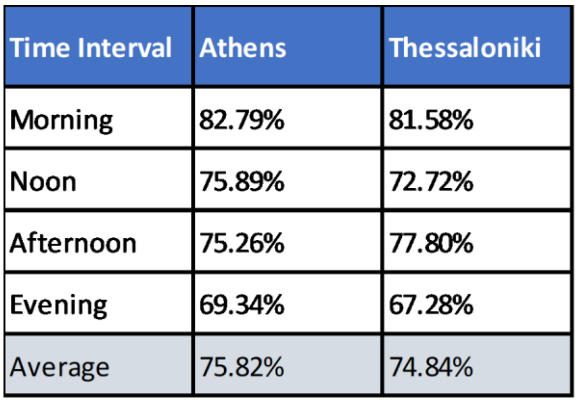


Figure 32: Results before during and after lockdown periods for 3-class experiments.

5.5 Athens-Thessaloniki comparison 3-classes

Next, we focus our research on a city level comparison between Athens and Thessaloniki. These 3-classes experiments are based on a model trained on the whole dataset (01.08.2019-31.08.2020). The results are listed in Table 2 and contain all time intervals. It is clear that Morning is the most predictable interval in both cities, with Athens at the top. In fact, only in the Morning accuracy exceeded 80%. On the contrary, in the Evening Athens and Thessaloniki underperformed with accuracy lower than 70%. Moreover, Athens scored higher at Noon with accuracy at 75.89% and Thessaloniki in the Afternoon (77.80%). On average accuracy is 1% higher in Athens and the difference in accuracy it not characterized by sharp fluctuations. Overall, results can be considered as satisfactory for a 3-class classification problem.

Table 2: Accuracy comparison between Athens and Thessaloniki for 3-class experiments for the whole dataset.



Time Interval	Athens	Thessaloniki
Morning	82.79%	81.58%
Noon	75.89%	72.72%
Afternoon	75.26%	77.80%
Evening	69.34%	67.28%
Average	75.82%	74.84%

Another 3-class experiment concentrated on effect of COVID-19 on our predictions, keeping in mind that movement restrictions significantly decreased traffic levels. Figure 33 presents an accuracy comparison between Athens and Thessaloniki before, during and after lockdown. We observe that there are alternations in best performing city. Specifically, in Thessaloniki accuracy is higher in the pre-lockdown and post-lockdown period. In both cities, accuracy fluctuates between 65% and 68% in all time periods. Indeed, COVID-19 mainly affected Jam Factor in Thessaloniki, as mentioned previously according to Figure 24. As a result accuracy is influenced as well, following the same direction with traffic levels.

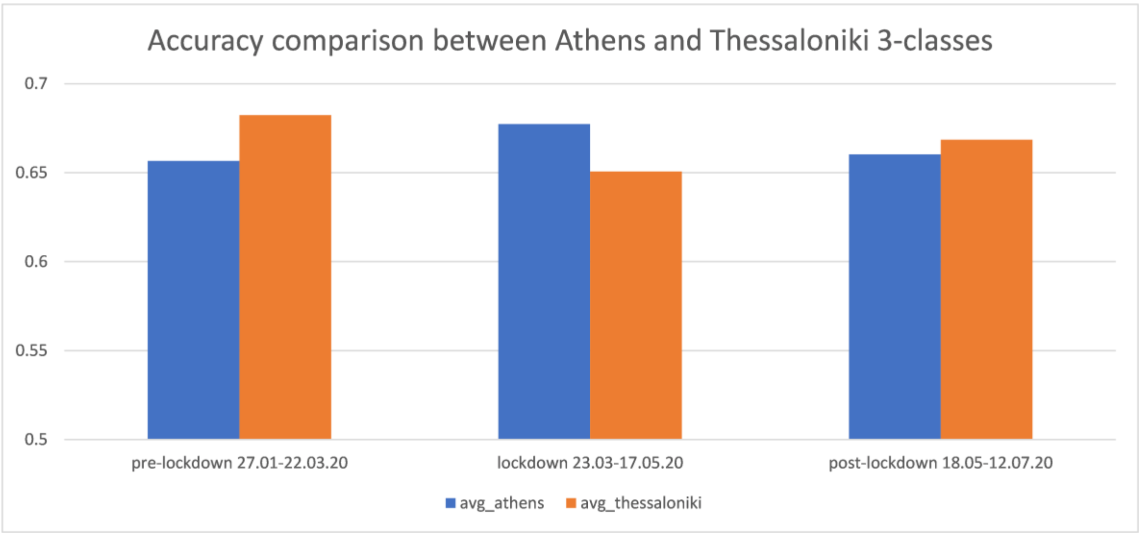


Figure 33: Accuracy comparison between Athens and Thessaloniki before, during and after lockdown for 3-classes experiments.

Focusing on time intervals during the day, Figure 34 depicts that for Athens the best predicted time interval is Morning before and after lockdown, and Afternoon during the COVID-19 restrictions, reaching a peak at 76.8%. On the other hand, in Thessaloniki, in each period there is different interval with a higher predictability. Before lockdown, the best predicted time interval is Morning, during lockdown period is Afternoon and after lockdown is Evening. Overall, model underperforms at Noon in both Athens and Thessaloniki, even though accuracy never drops lower than 60%.

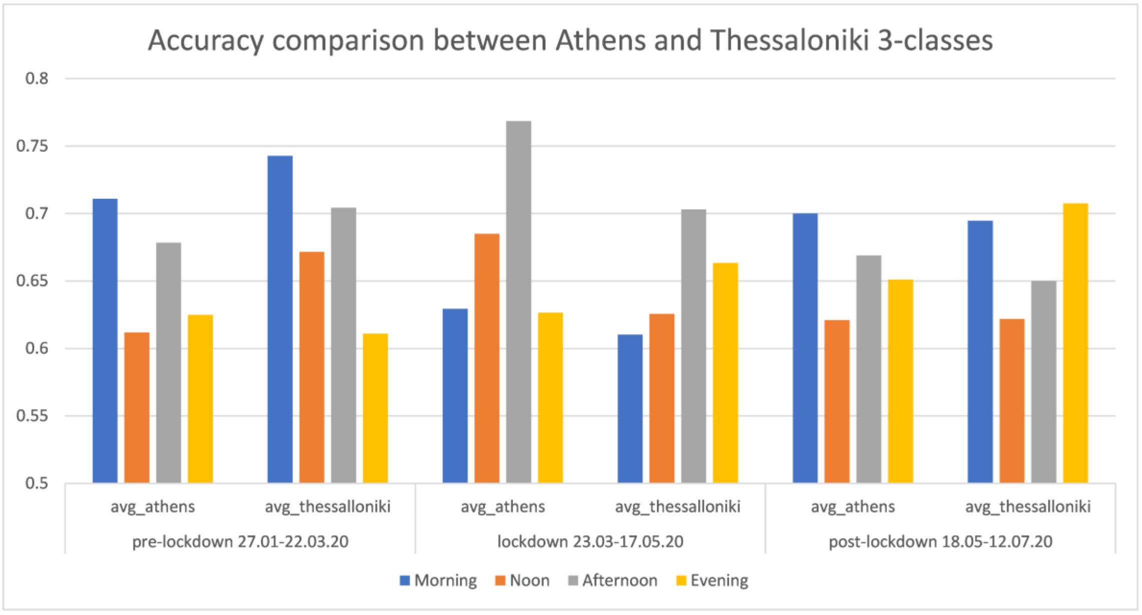


Figure 34: Accuracy comparison between Athens and Thessaloniki before, during and after lockdown, featuring all time intervals for 3-classes experiments.

6 Evaluation and Discussion

This chapter examines and discusses the findings of the current work, whilst comparing these with literature findings. While previous research work focused on analyzing periods with the main goal of extracting conclusions regarding individual sensors, current work examines the sensors as a whole and provides a comparison between the two cities. A major conclusion that previous effort unfolded [1], was that the subsets containing abnormal (whether expected or not) traffic periods (mostly holidays) were performing better than the corresponding trimester with the lowest expected abnormalities (15.09.2019-20.12.2019). Based on that, it was decided to exploit the heavily disruptive period of the COVID-19 related imposed lockdown. The initial model contained only weather metrics and was not discriminating weekdays from weekends. The dataset range was approximately six months, from August 2019 to January 2020. The current work includes a binary variable that distinguishes weekends as well as a new feature regarding the lockdown, while expanding the range until August 2020. The following subsections briefly compare primary conclusions for the initial work regarding the main technical aspects and concepts.

6.1 Discussion on early results

Combining the information of Figure 16 and Figure 27, Figure 35 was created. It was established that the model constructed with 2 classes for the target variable outperforms the 3-classes model. Figure 35 shows the accuracy achieved by Random Forest (RF-2, RF-3) and Logistic Regression (LR-2, LR-3) for the 2 and 3 class problems, respectively. RF-2 being the best option followed by LR-2. For the 2-classes the model consistently achieved accuracy over 80% with best results for Afternoon, while the 3-classes fared better in the Morning. Random Forest with 10-fold cross-validation achieved slightly better results than Logistic Regression. The trade-off between them is that Random Forest was 10 times slower than Logistic Regression. Given the amount of data used, Random Forest was chosen due to its higher accuracy. However, for a vast amount of data, Logistic Regression might be the appropriate classifier, due to its speed.

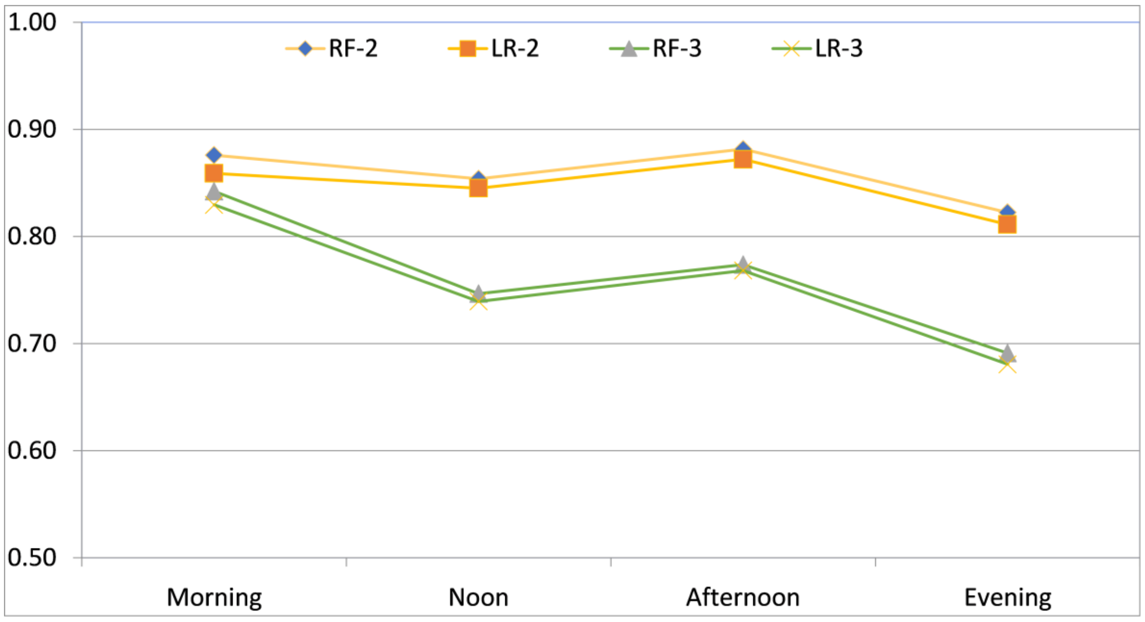


Figure 35: Comparing Logistic Regression and Random Forest classifiers in 2-classes and 3-classes.

Other efforts utilize the same two classifiers: Random Forest and Logistic Regression [1]. The difference in performance among the two classifiers was deemed insignificant with minor deviations. The same conclusion emerges from the current findings. On top of that, the Random Forest classifier needs approximately x10 times tuning, regarding hyper-parameters. Thus, the selection can be based on individual needs in terms of training and adaptation speed vs. accuracy. Respectively, related works suggest either Random Forest [23] or Decision Trees [28] as top performers.

Another important part was that in the 3-classes case, the accuracy of the baseline model barely exceeded a satisfactory level. However, processing more data and the addition of a new variable that distinguishes weekdays from weekends resulted in an impressive accuracy leap. The same conclusion applies for both cases (2 and 3-classes). In fact, for Morning the accuracy for 3 classes was surprisingly close to that of 2 classes. In both works, time intervals other than Morning resulted in a significant drop in accuracy.

One further verification of earlier findings the current work offers is that adding more data does not always result in higher accuracy. Similarly to the literature [1], the best scores were not achieved by training the whole dataset.

6.2 Evaluation 3-4 months

By selecting multiple splits to the dataset reveals points worth discussing, after observing the results emerging from a 10-fold cross-validation. The first point is that quadrimester splits produce better results for most of the cases. The line indicating the whole dataset (13 months) is common in Figure 19 and Figure 20, however it does not achieve better results than the corresponding subset of the lockdown and post-lockdown period at any interval during the day.

An interesting fact is that the newly introduced day interval named Noon, gives always worse results than the corresponding Afternoon one, while it only once scores higher than Morning. Afternoon produces the best results for the case of two classes in both splits (3 and 4 months) and seems to be the more predictable interval among those examined. The Evening interval has the lowest traffic load and for both splits it fails to withstand.

When the dataset is split into 3 classes the results during the day differ. The Morning interval gives better results than the rest, but still Afternoon outperforms Noon. Even though Morning and Evening intervals have the lowest loads of traffic throughout the dataset range, they do not share similar predictability. Evening usually underperforms all others, however in the case of the 4-month split it achieves accuracy similar to Noon.

6.3 Weekday vs. weekend

A period of undoubted traffic variation is weekends. The literature [1] is unclear if weekdays should be processed without weekends or models should consider them as a whole.

Besides the fact that another work [1] segregated data as such, weekends as a binary feature was the dominating factor of this work. In general, this tagging was expected to show increased accuracy in advance, because those instances (weekends) present significantly less traffic congestion. Thus, the target variable on those instances is expected to be lower than the median value (class Low), bearing in mind that weekends are fewer than weekdays. However, based on applied recursive feature elimination, it is impressive that it achieves approximately three times higher importance than any weather condition metric. The evaluation of this addition is positive and it is considered of high significance for future research as it is not an expensive and hard to capture feature like the rest utilized by the model.

6.4 Morning / Noon / Afternoon / Evening

Time intervals of interest during a day can vary. In small cities normally the focus is mostly up to the afternoon. In this case both cities are quite large (Athens with over 4 million population, and Thessaloniki with over 1 million) and considered busy until very late. Thus, multiple splits were examined. In addition to another work [1], a new subset of the same length (3 hours), named Noon was introduced and captured data from 12:00 to 15:00. It is identified as peak hours but differs from other findings [23].

Prior observations [1] had shown that the load of traffic congestion was lower for Morning, while the new data which contain information for a whole year reflect a slightly different conclusion. Indeed, the current dataset reveals that the lowest traffic load is linked with Evening, followed by Morning. By having a bigger dataset, Noon and Afternoon are still the most heavily loaded intervals during a day. Another significant observation for the 2-classes experiments, is that for the Evening split the final model struggles to compete with the rest intervals, while for the baseline model, without the addition of the new features, it was the only prominent one. Overall, Morning and Afternoon intervals outperform the rest for both 2-classes and 3-classes experiments, even for the cases of the whole model and the lockdown segmentation.

6.5 COVID-19 / Lockdown

Unfortunately, 2020 contained a concurrent lockdown for both cities which -inevitably- heavily affected traffic. Indicatively, the levels of traffic were even lower than these in the middle of August which traditionally is considered in Greece to be the peak of vacations, thus traffic congestion is extremely low in both cities.

For this approach we decided to form a time range of interest based on the length of the lockdown period (56 days). In other words, we chose one subset of 56 days for the pre-lockdown period and another for the post-lockdown with the goal to compare the model for all three.

Based on that, clearly, the periods during the lockdown and after it scored approximately 10% worse than pre-lockdown. Interestingly, combining all three subsets the best accuracy achieved, was even higher than the score for the whole dataset, as seen in Figure 21 and Figure 22 for the 2-classes experiments. Similarly to the previous subsection, the same experiments were performed for all time intervals during the day for the 3-class split (see Figure 31 and Figure 32). During the lockdown period

(23.03.2020-17.05.2020) the algorithm regarding Afternoon had the worst accuracy among all intervals, while Noon was slightly the best. Finally for the case of 2-classes, a validation of the trends is that the model trained on the whole dataset has very similar (in terms of deviation) results to the whole COVID-19 related split of data from 27 January to 12 July. For the case of 3-classes, again, the Morning interval scores higher accuracy while the whole model also has very similar results to the merged lockdown-split.

6.6 Athens vs. Thessaloniki

Undoubtedly, Athens is more populated than Thessaloniki and attracts more tourists throughout the year. Considering that traffic is positively correlated to the population, higher jam factor values for Athens are explainable. However, the similarities between the two cities in daily activities lead to analogous patterns before and after COVID-19 restrictions. The results of the 2-classes experiments indicate that the model performs better with data from Thessaloniki before and during the lockdown (see Figure 25). However, after the lockdown, traffic in Thessaloniki does not return to normal levels and it also results in a drop in accuracy. Consequently, it can be claimed that the model's accuracy is affected by the traffic levels. This claim is supported during and after the lockdown transition. However, in the case of 3-classes (see Figure 33) the accuracy levels before, during and after the lockdown range between 65% and 68%, as a result not significant variations are detected and there is not a drop during the lockdown period.

6.7 Threats to validity

Many scientific efforts analyze and examine their questions based on several assumptions. Most of the time those assumptions are raised due to unknown factors, limited data, or personal estimations. The validity of results and conclusions with assumptions is highly connected.

A crucial assumption of this work relies on the fact that the model developed needs accurate predictions of day-ahead weather conditions, which is not always feasible. Another threat is that 2020, due to the pandemic, seems to affect the post-lockdown load of traffic and thus the results may differ from previous years. In addition, transforming the problem into balanced classification may slightly affect the accuracy of the model since the introduction of abnormal instances may not affect the split point the same way the mean value might do.

7 Conclusions and Future Work

The purpose of this dissertation was to examine how different intervals during the day and year can affect traffic forecasting. The main aspects of the proposed model are weather metrics, combined with features describing weekends and the lockdown. In general, the fragmentation of the dataset was extensive. The dataset was initially split into four intervals during a day (Morning, Noon, Afternoon, Evening) and consecutive trimesters and quadrimesters along the whole range. On top of that, findings led to the decision for a further split adjusted for the lockdown period which lasted 56 days in Greece.

7.1 Conclusions

The major factor that boosted the model's performance was the distinction of weekends. Besides that, having a variable that marks the status of days, regarding lockdown presence, also rewarded the model with a small increase.

In between, transforming the target variable into three classes resulted in remarkably lower accuracy (besides the Morning subset), however it justified some findings regarding day intervals.

A clear conclusion is that the model was more successful for Morning and Afternoon. This is probably because these splits coincide with the beginning and the end of work for most of the working population. Also, the Evening interval does not stand out in comparison with the rest. This can be speculated to be because weather metrics such as UV and humidity do not add much variation. The Evening interval though has the lowest load of traffic and thus its predictability may be affected in a negative way.

The splits regarding Spring achieved higher accuracy, but this rather happened because of the explosion of the pandemic. The constructed subset based on the lockdown era, consisting of 168 days, clarified that datasets with equally mixed data had scores consistently higher, and Morning and Afternoon intervals were again the most reliable. Consequently, in case of special events, it is suggested that training of models should be equal for each period.

It seems that higher traffic load does not guarantee higher accuracy, since even though the Noon split had higher traffic load, it did not outperform the rest at any point. On the other hand, the Evening interval which marked the lowest traffic load, performed worse for almost every scenario. Also, testing the model on the whole dataset did not result in better performance, thus it is concluded that a bigger training set does not directly improve accuracy. This means that similar models in order to remain competitive in terms of accuracy should be deployed with a clear target period and not arbitrarily be trained on all available data.

The comparison of the two cities did not yield concrete results. It seems that the model is better adjusted on sensors based in Thessaloniki, but without significant differentiation and certainly not for the post-lockdown period.

Overall, we could claim that our research effort could enhance Smart Mobility and Transportation and therefore the Smart Cities concept. Traffic prediction is vital for efficient traffic management and could reduce traffic congestion. Our approach is concentrated on predicting traffic levels based on weather conditions, day of the week and traffic restrictions. Our findings could provide trip advisory to travelers in order to plan travel accurately. Another contribution of our work could be on traffic management system. For instance, smart traffic lights could operate based on our traffic level prediction in order to reduce congestion, limit air pollution and improve travel time.

7.2 Future Work

The insights this work concluded were very interesting, and while some were expected, there are others that were not. Our understanding suggests that the analysis regarding the day intervals has not much space to be further exploited.

On the other hand, the splits throughout the year may reveal significant points in order to adopt the model and the potential is considered substantial. Feature-wise it would be interesting to also focus on the weather icon (snow, rain etc.). This information could enhance our model’s performance. Partially, our model already categorizes the weather icon in the background, based on weather conditions such as humidity and atmospheric pressure. Possibly, weather icon could lead to a different feature selection. Furthermore, it could improve our model’s explainability.

Furthermore, there is room for improvement in our data modeling approach. Special events and holidays should be incorporated in our model and the current conclusions should be examined. Lastly, possible future movement restrictions could assist on further validating existing conclusions.

References

[1] Christantonis, K., Tjortjis, C., Manos, A., Filippidou, D. E., Mougiakou, E., & Christelis, E. (2020, June). Using Classification for Traffic Prediction in Smart Cities. In *IFIP International Conference on Artificial Intelligence Applications and Innovations* (pp. 52-61). Springer, Cham.

[2] Anthopoulos, L. G. (2017). *Understanding smart cities: A tool for smart government or an industrial trick?* (Vol. 22). Cham: Springer International Publishing.

[3] Soomro, K., Bhutta, M. N. M., Khan, Z., & Tahir, M. A. (2019). Smart city big data analytics: An advanced review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(5), e1319.

[4] Moustaka, V., Vakali, A., & Anthopoulos, L. G. (2018). A systematic review for smart city data analytics. *ACM Computing Surveys (CSUR)*, *51*(5), 1-41.

[5] Soomro, K., Bhutta, M. N. M., Khan, Z., & Tahir, M. A. (2019). Smart city big data analytics: An advanced review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(5), e1319.

[6] Al Nuaimi, E., Al Neyadi, H., Mohamed, N., & Al-Jaroodi, J. (2015). Applications of big data to smart cities. *Journal of Internet Services and Applications*, *6*(1), 25.

[7] Christantonis, K., & Tjortjis, C. (2019, July). Data mining for smart cities: predicting electricity consumption by classification. In *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)* (pp. 67-73). IEEE.

[8] Christantonis, K., Tjortjis, C., Manos, A., Elizabeth, D., & Filippidou, E. C. (2020). Smart Cities Data Classification for Electricity Consumption & Traffic Prediction. *Automatics & Software Enginery*, *31*(1), 49-69.

[9] Dubbeldeman, R., & Stephen, W. (2015). Smart cities: How rapid advances in technology are reshaping our economy and society. *Delodtte*. Available from: https://www2.deloitte.com/content/dam/Deloitte/tr/Documents/public-sector/deloitte-nl-ps-smart-cities-report.pdf

[10] Woetzel, J., Remes, J., Boland, B., Lv, K., Sinha, S., Strube, G., ... & Von der Tann, V. (2018). Smart cities: Digital solutions for a more livable future. *McKinsey Global Institute: New York, NY, USA*, 1-152.

[11] NITI Aayog (2018, June). National strategy for artificial intelligence. *Discussion Paper*. Available from: http://niti.gov.in/writereaddata/files/document_publication/NationalStrategy-for-AI-Discussion-Paper.pdf

[12] Antoniou, C., Koutsopoulos, H. N., & Yannis, G. (2013). Dynamic data-driven local traffic state estimation and prediction. *Transportation Research Part C: Emerging Technologies*, *34*, 89-107.

[13] Kalvapalli, S. P. K., & Chelliah, M. (2019). Analysis and Prediction of City-Scale Transportation System Using XGBOOST Technique. In *Recent Developments in Machine Learning and Data Analytics* (pp. 341-348). Springer, Singapore.

[14] Ferreira, N., Poco, J., Vo, H. T., Freire, J., & Silva, C. T. (2013). Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips. *IEEE transactions on visualization and computer graphics*, *19*(12), 2149-2158.

[15] Antoniades, C., Fadavi, D., & Amon, A. F. (2016). *Fare and duration prediction: A study of New York city taxi rides*. Tech. Rep.

[16] Hashemi, S. M., Almasi, M., Ebrazi, R., & Jahanshahi, M. (2012). Predicting the next state of traffic by data mining classification techniques. *International Journal of Smart Electrical Engineering*, *1*(03), 181-193.

[17] Wen, F., Zhang, G., Sun, L., Wang, X., & Xu, X. (2019). A hybrid temporal association rules mining method for traffic congestion prediction. *Computers & Industrial Engineering*, *130*, 779-787.

[18] Theodorou, T. I., Salamanis, A., Kehagias, D. D., Tzovaras, D., & Tjortjis, C. (2017, April). Short-term traffic prediction under both typical and atypical traffic conditions using a pattern transition model. In *International Conference on Vehicle Technology and Intelligent Transport Systems* (Vol. 2, pp. 79-89). SCITEPRESS.

[19] Pan, B., Demiryurek, U., & Shahabi, C. (2012, December). Utilizing real-world transportation data for accurate traffic prediction. In *2012 IEEE 12th International Conference on Data Mining* (pp. 595-604). IEEE.

[20] Liu, Y., Choudhary, A., Zhou, J., & Khokhar, A. (2006, September). A scalable distributed stream mining system for highway traffic data. In *European Conference on Principles of Data Mining and Knowledge Discovery* (pp. 309-321). Springer, Berlin, Heidelberg.

[21] Vlahogianni, E. I., Karlaftis, M. G., & Golias, J. C. (2014). Short-term traffic forecasting: Where we are and where we’re going. *Transportation Research Part C: Emerging Technologies*, *43*, 3-19.

[22] Gecchele, G., Rossi, R., Gastaldi, M., & Caprini, A. (2011). Data mining methods for traffic monitoring data analysis: A case study. *Procedia-Social and Behavioral Sciences*, *20*, 455-464.

[23] Necula, E. (2015). Analyzing traffic patterns on street segments based on GPS data using R. *Transportation Research Procedia*, *10*, 276-285.

[24] Bolbol, A., Cheng, T., Tsapakis, I., & Haworth, J. (2012). Inferring hybrid transportation modes from sparse GPS data using a moving window SVM classification. *Computers, Environment and Urban Systems*, *36*(6), 526-537.

[25] Makridis, M., Mattas, K., Ciuffo, B., Raposo, M. A., & Thiel, C. (2018). Assessing the impact of connected and automated vehicles. A freeway scenario. In *Advanced Microsystems for Automotive Applications 2017* (pp. 213-225). Springer, Cham.

[26] Li, Y., Zheng, Y., Zhang, H., & Chen, L. (2015, November). Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems* (pp. 1-10).

[27] Wojnarski, M., Gora, P., Szczuka, M., Nguyen, H. S., Swietlicka, J., & Zeinalipour, D. (2010, December). Ieee icdm 2010 contest: Tomtom traffic prediction for intelligent gps navigation. In *2010 IEEE International Conference on Data Mining Workshops* (pp. 1372-1376). IEEE.

[28] Mystakidis, A., & Tjortjis, C. (2020). Big Data Mining for Smart Cities: Predicting Traffic Congestion using Classification. In *2020 IEEE 11th International Conference on Information, Intelligence, Systems and Applications*.

[29] Boufidis, N. (2019). *Distributed traffic forecasting using Apache Spark*. [Master's dissertation, International Hellenic University].

[30] Adam, K. (2018). *Real-time Traffic Prediction using Multiple Data Sources - a Neural Network Approach*. [Master's dissertation, Aristotle University of Thessaloniki].

[31] Al Najada, H., & Mahgoub, I. (2016, September). Big vehicular traffic data mining: Towards accident and congestion prevention. In *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)* (pp. 256-261). IEEE.

[32] Li, L., Shrestha, S., & Hu, G. (2017, June). Analysis of road traffic fatal accidents using data mining techniques. In *2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA)* (pp. 363-370). IEEE.

[33] Taamneh, M., Alkheder, S., & Taamneh, S. (2017). Data-mining techniques for traffic accident modeling and prediction in the United Arab Emirates. *Journal of Transportation Safety & Security*, *9*(2), 146-166.

[34] Kashani, A. T., & Mohaymany, A. S. (2011). Analysis of the traffic injury severity on two-lane, two-way rural roads based on classification tree models. *Safety Science*, *49*(10), 1314-1320.

[35] Krishnaveni, S., & Hemalatha, M. (2011). A perspective analysis of traffic accident using data mining techniques. *International Journal of Computer Applications*, *23*(7), 40-48.

[36] Chong, M., Abraham, A., & Paprzycki, M. (2004, August). Traffic accident data mining using machine learning paradigms. In *Fourth International Conference on Intelligent Systems Design and Applications (ISDA'04), Hungary* (pp. 415-420).

[37] Shanthi, S., & Ramani, R. G. (2012, October). Feature relevance analysis and classification of road traffic accident data through data mining techniques. In *Proceedings of the World Congress on Engineering and Computer Science* (Vol. 1, pp. 24-26).

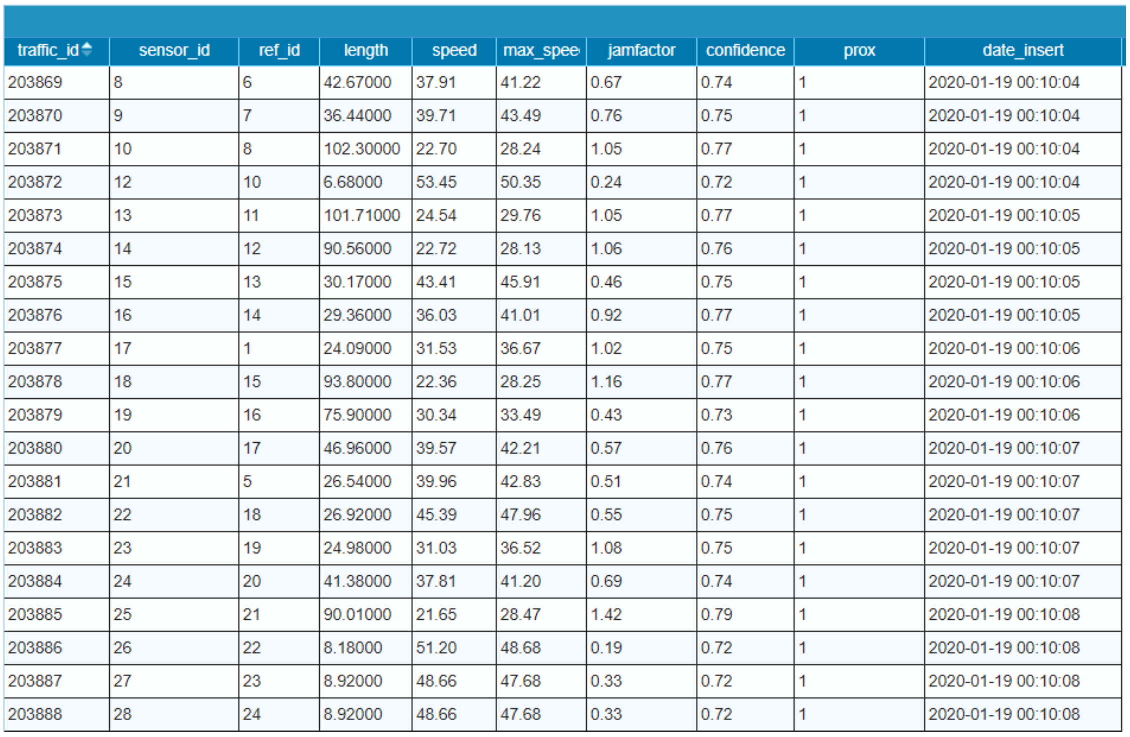
[38] Ramya, S., Reshma, S. K., Manogana, V. D., & Saroja, Y. S. (2019). Accident Severity Prediction using Data Mining Methods. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, *5*(2).

[39] Lee, D. H., Jeng, S. T., & Chandrasekar, P. (2004). Applying data mining techniques for traffic incident analysis. *Journal of the Institution of Engineers*, *44*(2), 90-101.

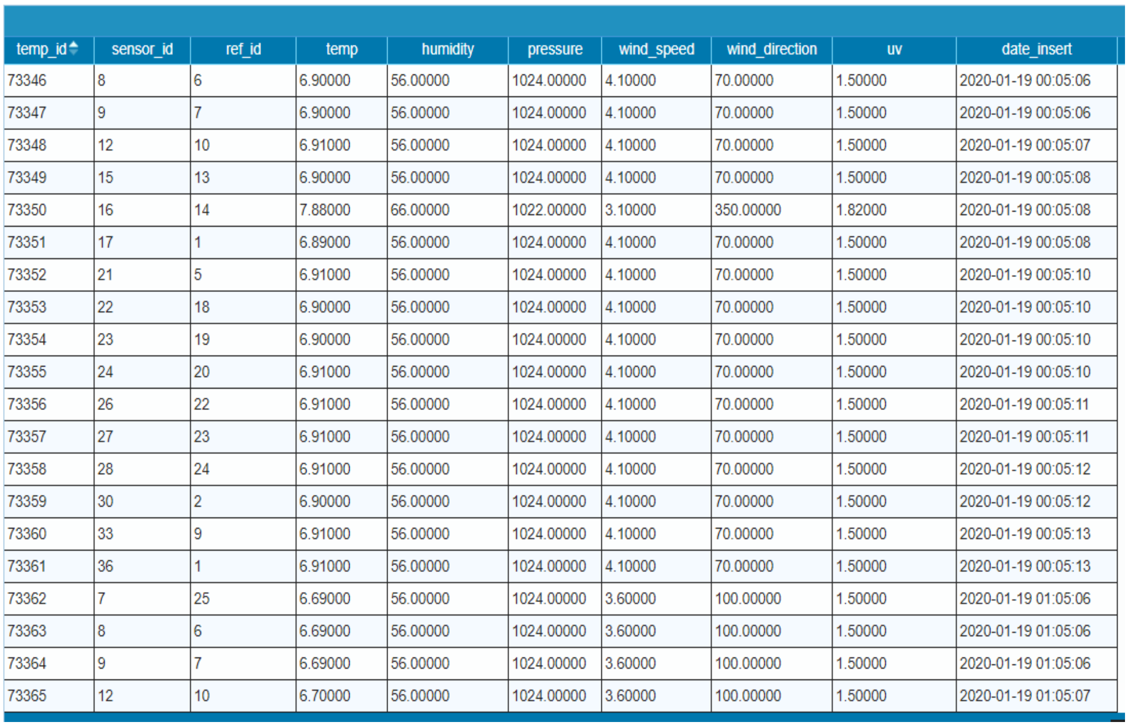
[40] *Public Participation CITY*. PPCITY. Retrieved November 28, 2020, from https://panel.ppcity.eu/platform3/

Appendices

Appendix A: Traffic and Weather Data Samples



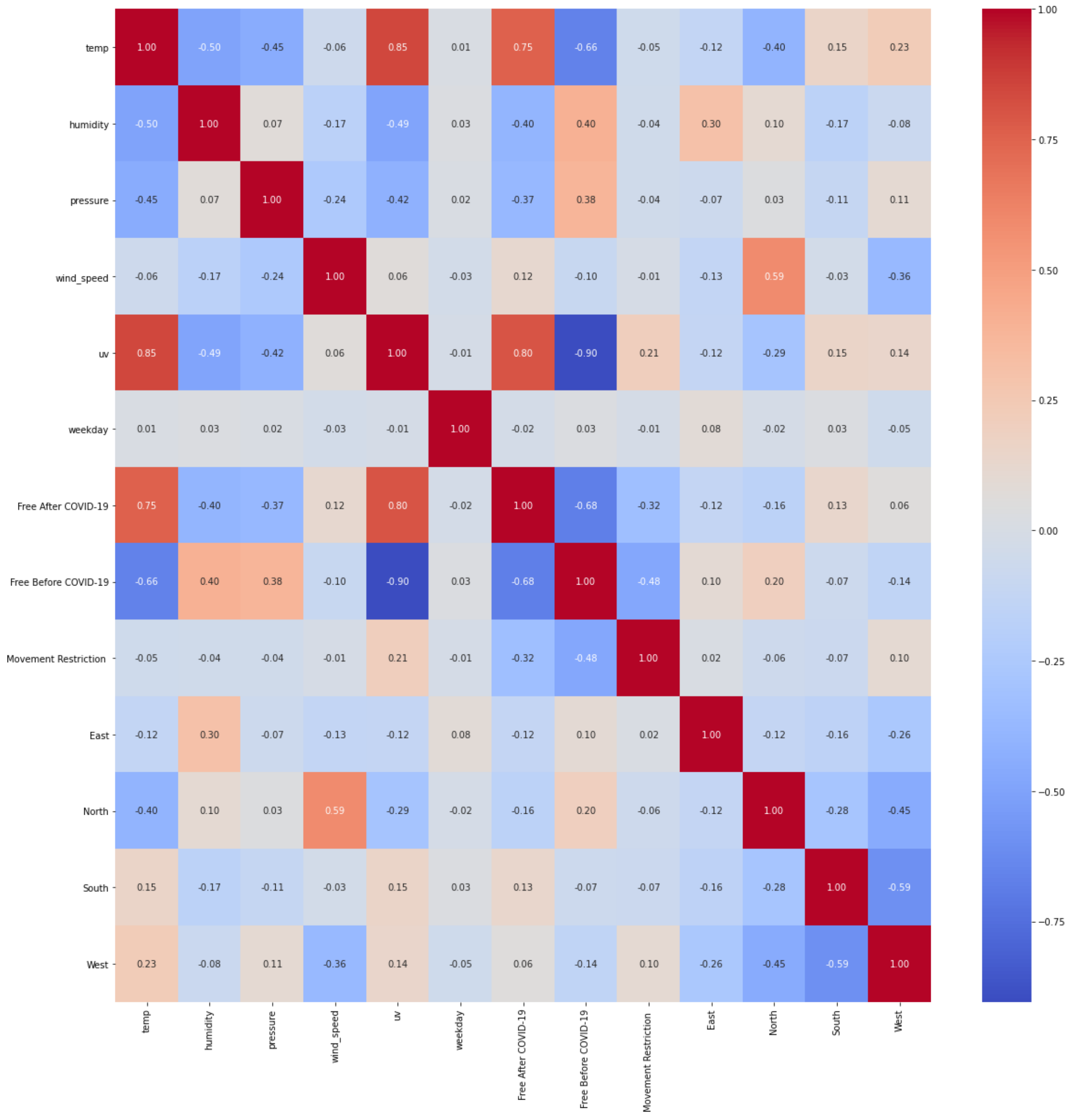
traffic_id	sensor_id	ref_id	length	speed	max_spee	jamfactor	confidence	prox	date_insert
203869	8	6	42.67000	37.91	41.22	0.67	0.74	1	2020-01-19 00:10:04
203870	9	7	36.44000	39.71	43.49	0.76	0.75	1	2020-01-19 00:10:04
203871	10	8	102.30000	22.70	28.24	1.05	0.77	1	2020-01-19 00:10:04
203872	12	10	6.68000	53.45	50.35	0.24	0.72	1	2020-01-19 00:10:04
203873	13	11	101.71000	24.54	29.76	1.05	0.77	1	2020-01-19 00:10:05
203874	14	12	90.56000	22.72	28.13	1.06	0.76	1	2020-01-19 00:10:05
203875	15	13	30.17000	43.41	45.91	0.46	0.75	1	2020-01-19 00:10:05
203876	16	14	29.36000	36.03	41.01	0.92	0.77	1	2020-01-19 00:10:05
203877	17	1	24.09000	31.53	36.67	1.02	0.75	1	2020-01-19 00:10:06
203878	18	15	93.80000	22.36	28.25	1.16	0.77	1	2020-01-19 00:10:06
203879	19	16	75.90000	30.34	33.49	0.43	0.73	1	2020-01-19 00:10:06
203880	20	17	46.96000	39.57	42.21	0.57	0.76	1	2020-01-19 00:10:07
203881	21	5	26.54000	39.96	42.83	0.51	0.74	1	2020-01-19 00:10:07
203882	22	18	26.92000	45.39	47.96	0.55	0.75	1	2020-01-19 00:10:07
203883	23	19	24.98000	31.03	36.52	1.08	0.75	1	2020-01-19 00:10:07
203884	24	20	41.38000	37.81	41.20	0.69	0.74	1	2020-01-19 00:10:07
203885	25	21	90.01000	21.65	28.47	1.42	0.79	1	2020-01-19 00:10:08
203886	26	22	8.18000	51.20	48.68	0.19	0.72	1	2020-01-19 00:10:08
203887	27	23	8.92000	48.66	47.68	0.33	0.72	1	2020-01-19 00:10:08
203888	28	24	8.92000	48.66	47.68	0.33	0.72	1	2020-01-19 00:10:08



temp_id	sensor_id	ref_id	temp	humidity	pressure	wind_speed	wind_direction	uv	date_insert
73346	8	6	6.90000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:06
73347	9	7	6.90000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:06
73348	12	10	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:07
73349	15	13	6.90000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:08
73350	16	14	7.88000	66.00000	1022.00000	3.10000	350.00000	1.82000	2020-01-19 00:05:08
73351	17	1	6.89000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:08
73352	21	5	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:10
73353	22	18	6.90000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:10
73354	23	19	6.90000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:10
73355	24	20	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:10
73356	26	22	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:11
73357	27	23	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:11
73358	28	24	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:12
73359	30	2	6.90000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:12
73360	33	9	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:13
73361	36	1	6.91000	56.00000	1024.00000	4.10000	70.00000	1.50000	2020-01-19 00:05:13
73362	7	25	6.69000	56.00000	1024.00000	3.60000	100.00000	1.50000	2020-01-19 01:05:06
73363	8	6	6.69000	56.00000	1024.00000	3.60000	100.00000	1.50000	2020-01-19 01:05:06
73364	9	7	6.69000	56.00000	1024.00000	3.60000	100.00000	1.50000	2020-01-19 01:05:06
73365	12	10	6.70000	56.00000	1024.00000	3.60000	100.00000	1.50000	2020-01-19 01:05:07

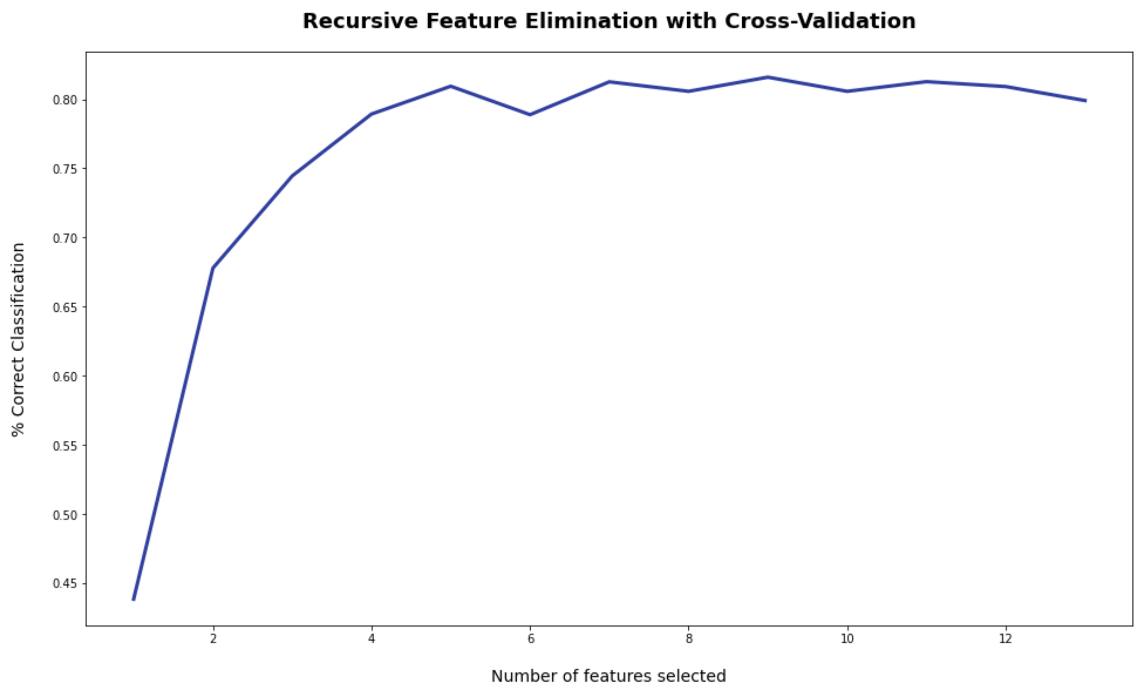
Appendix B: Feature Correlation and Feature Selection

```python
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(20,20))
sns.heatmap(final_df.corr(),fmt=".2f", cmap='coolwarm',annot=True)
```



```python
from sklearn.feature_selection import RFECV
from sklearn.model_selection import GridSearchCV, cross_val_score, StratifiedKFold, learning_curve

X = final_df.drop(['Target'], axis=1)
target = final_df['Target']

rfc = RandomForestClassifier(random_state=0)
rfecv = RFECV(estimator=rfc, step=1, cv=StratifiedKFold(10),
       scoring='accuracy')
rfecv.fit(X, target)

print('Optimal number of features: {}'.format(rfecv.n_features_))
```

```python
plt.figure(figsize=(16, 9))
plt.title('Recursive Feature Elimination with Cross-Validation',
          fontsize=18, fontweight='bold', pad=20)
plt.xlabel('Number of features selected', fontsize=14, labelpad=20)
plt.ylabel('% Correct Classification', fontsize=14, labelpad=20)
plt.plot(range(1, len(rfecv.grid_scores_) + 1),
         rfecv.grid_scores_, color='#303F9F', linewidth=3)

plt.show()
```



```python
import numpy as np

#features to drop
print(np.where(rfecv.support_ == False)[0])

X.drop(X.columns[np.where(rfecv.support_ == False)[0]], axis=1, inplace=True)

print("We eliminate these features: East, North, South, West")
```

```
X.columns
```

```
Index(['temp', 'humidity', 'pressure', 'wind_speed', 'uv', 'weekday',
       'Free After COVID-19', 'Free Before COVID-19', 'Movement Restriction '],
      dtype='object')
```

Appendix C: Scripts for Traffic Prediction

```python
# Traffic Prediction

#connect to Google Drive

from google.colab import drive
drive.mount('/content/drive')

#imports

import pandas as pd
import copy
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score

# Read csv files

df_temp_raw = pd.read_csv('/content/drive/My Drive/thesis_colab/temp_02.09.20.
                                                  csv',sep=';', encoding="ISO-8859-1")
df_traffic_raw = pd.read_csv('/content/drive/My Drive/thesis_colab/traffic_02.
                                         09.20.csv',sep=';',encoding="ISO-8859-1")

# Processing 1

def day_intervals(
    sensor_no, prox_km, interval_start, interval_stop , weekday_check):

    df_traffic_raw['date_insert'] = pd.to_datetime(
                                        df_traffic_raw ['date_insert'])
    df_temp_raw['date_insert'] = pd.to_datetime(df_temp_raw['date_insert'])

    #create weekends feature
    if weekday_check == True:
      df_temp_raw['weekday'] = df_temp_raw['date_insert'].dt.dayofweek
      df_temp_raw['weekday'] = (df_temp_raw['weekday']// 5 == 1).astype(int)

    #select sensor_id
    df_traffic = df_traffic_raw.loc[(df_traffic_raw['sensor_id'] == sensor_no)

     & (df_traffic_raw['prox'] == prox_km)]

    df_temp = df_temp_raw.loc[df_temp_raw['sensor_id'] == sensor_no]

    #select time interval
    df_traffic = df_traffic[(df_traffic['date_insert'].dt.hour <=
    interval_stop)& (df_traffic['date_insert'].dt.hour >= interval_start)]

    df_temp = df_temp[(df_temp['date_insert'].dt.hour <= interval_stop) &
                           (df_temp['date_insert'].dt.hour >= interval_start)]


    df_temp = df_temp.groupby([df_temp['date_insert'].dt.date]).mean()
    df_traffic = df_traffic.groupby(
                 [df_traffic['date_insert'].dt.date])['jamfactor'].mean()

    #join dataframes
    df_combined = pd.concat([df_traffic, df_temp], axis=1, join='inner')
    df_combined = df_combined.drop(['temp_id', 'sensor_id', 'ref_id'], axis=1)


# Processing 2 for 2-class classification

def two_classes_classification(df_combined):
    jammax = df_combined['jamfactor'].max()
    jammin = df_combined['jamfactor'].min()
    jamrange = jammax - jammin
    jammedian = df_combined['jamfactor'].median()
    jammean = df_combined['jamfactor'].mean()
    print(' ')
    print('Max of Jamfactor equals to: ', jammax)
    print('Min of Jamfactor equals to: ', jammin)
    print('Median of Jamfactor equals to: ', jammedian)
    print('Range of Jamfactor equals to: ', jamrange)
    print('Mean of Jamfactor equals to: ', jammean)

    #class limits
    Targetlist = df_combined['jamfactor'].tolist()
    emptylist = []
    for i, elem in enumerate(Targetlist):
        if elem > jammedian:
            emptylist.append('High')
        else:
            emptylist.append('Low')

    dfentry = pd.Series(emptylist)
    df_combined['Target'] = dfentry.values
    print("No. of High values is: ", emptylist.count('High'))
    print("No. of Low values is: ", emptylist.count('Low'))

    # Wind Direction feature engineering
    DirectionList = df_combined['wind_direction'].tolist()
    emptylist = []
    for i, elem in enumerate(DirectionList):
        if elem < 45.0 or elem > 315.0:
            emptylist.append('North')
        elif elem < 135.0:
            emptylist.append('East')
        elif elem < 225.0:
            emptylist.append('South')
        else:
            emptylist.append('West')

        dfentry = pd.Series(emptylist)
        df_combined['wind_direction'] = dfentry.values

        initial_df_ = pd.get_dummies(df_combined['wind_direction'])
        final_df = pd.concat([df_combined, initial_df_], axis=1)

        del final_df['jamfactor']
        del final_df['wind_direction']

        #feature selection suggests:
        del final_df['East']
        del final_df['North']
        del final_df['South']
        del final_df['West']

        return final_df

    # Classification

    def classification_task(dataframe, algorithm, scale_option):
        Y = dataframe['Target']
        X = dataframe.drop('Target', 1)

        X_train, X_test, y_train, y_test = train_test_split(X, Y,
                                           test_size=0.2, random_state=42)

        if scale_option == True:
            scaler = StandardScaler()
            scaler.fit(X_train)
            X_train = scaler.transform(X_train)
            X_test = scaler.transform(X_test)

        if algorithm == 'RandomForest':
            clf = RandomForestClassifier(random_state=42)
            n_estimators = [5, 10, 30, 50, 100]
            min_samples_split = [2,  5]
            min_samples_leaf = [2, 3, 4]
```

```python
            max_depth = [2, 3]
            max_features = ['auto']
            param_grid = {'n_estimators': n_estimators,
                          'max_features': max_features,
                          'min_samples_split': min_samples_split,
                          'min_samples_leaf': min_samples_leaf,
                          'max_depth': max_depth}
        else:
            clf = LogisticRegression(penalty='l2', random_state=42)
            max_iter = [10, 20, 50, 100, 150]
        C = [0.001, 0.01, 0.1, 0.5, 1, 2, 3]
        param_grid = {'max_iter': max_iter, 'C': C}

        CLF = GridSearchCV(clf, param_grid=param_grid, cv=10, verbose=10)
        CLF.fit(X_train, y_train)

        preds = CLF.predict(X_test)
        accuracy = accuracy_score(y_test, preds)
        params = CLF.best_params_

        print('Best parameters: ', CLF.best_params_)
        print('Average score: ', CLF.best_score_)

        return CLF.best_score_, accuracy , params

    # 3-class classification

    def three_classes_classification(dataframe):

        Targetlist = dataframe['jamfactor'].tolist()
        Flaglist = copy.deepcopy(Targetlist)
        Flaglist.sort()
        Flag1 = round(len(Flaglist) / 3)
        Flag2 = Flag1 * 2
        print(Flag1)
        print(Flag2)

        #class limits
        emptylist = []
        for i, elem in enumerate(Targetlist):
            if elem <= Flaglist[Flag1 - 1]:
                emptylist.append('Low')
            elif elem <= Flaglist[Flag2 - 1]:
                emptylist.append('Medium')
            else:
                emptylist.append('High')

        dfentry = pd.Series(emptylist)
        dataframe['Target'] = dfentry.values

        # Wind Direction feature engineering
        DirectionList = dataframe['wind_direction'].tolist()
        emptylist = []
        for i, elem in enumerate(DirectionList):
            if elem < 45.0 or elem > 315.0:
                emptylist.append('North')
            elif elem < 135.0:
                emptylist.append('East')
            elif elem < 225.0:
                emptylist.append('South')
            else:
                emptylist.append('West')

        dfentry = pd.Series(emptylist)
        dataframe['wind_direction'] = dfentry.values

        # Dummies for Wind Direction
        initial_df_ = pd.get_dummies(dataframe['wind_direction'])
        final_df = pd.concat([dataframe, initial_df_], axis=1)
        del final_df['wind_direction']
        del final_df['jamfactor']

        #feature selection suggests:
        del final_df['East']
        del final_df['North']
        del final_df['South']
        del final_df['West']

        return final_df

    def year_intervals(dataframe):

        # selecting time period for experiments
        startdate = pd.to_datetime('2019-08-01').date()
        enddate = pd.to_datetime('2020-08-31').date()

        final_df = dataframe.loc[startdate: enddate]

        print(final_df.head(70))
        print(final_df.shape)

        return final_df

    # create "status" feature

    def create_status(df_combined):
      df_combined['date_insert'] = df_combined.index
      df_combined['date_insert'] = pd.to_datetime(df_combined['date_insert'],
                                   format='%Y-%m-%d')

      status = {'Free Before COVID19': ('29 July 2019', '22 March 2020'),
                'Movement Restriction ':('23 March 2020', '17 May 2020'),
                'Free After COVID-19': ('18 May 2020', '02 September 2020')}
      status_map = {date.date(): status for status,
                    dates in status.items() for date in pd.date_range(*dates)}

      df_combined['status'] = df_combined['date_insert'].dt.date.map(
                              status_map).fillna('Free After COVID-19')
      pd.set_option('display.max_rows', df_combined.shape[0]+1)
      del df_combined['date_insert']
      df_dum_status = pd.get_dummies(df_combined['status'])
      df_combined = pd.concat([df_combined, df_dum_status], axis=1)
      del df_combined['status']

      return df_combined

    #Experiments

    ids=[]
    for i in range(7,37): #all sensors
      if i not in [11,32,34,35]: #sensors dropped
        ids.append(i)
    for i in ids:
      sensor_id = i
      start_time = 19
      end_time = 21 #21 means until 21:59

      weekday_option = True #"weekends" feature
      df_combined = day_intervals(sensor_id, 1, start_time, end_time,
                    weekday_option)
      df_combined = year_intervals(df_combined) #time period
      df_combined = create_status(df_combined) #"status" feature

      #select 2 or 3 classes
      classes = 3
      if classes ==2:
        final_df = two_classes_classification(df_combined)
      else:
         final_df = three_classes_classification(df_combined)

      #select classifier
      algorithm = 'RandomForest'
      # algorithm = 'LR'

      #scale option
      if algorithm == 'LR':
        scale_option = True
      else:
        scale_option = False

      classification_task(final_df, algorithm, scale_option)
      result= classification_task(final_df, algorithm, scale_option)
      score =result[0] #validation score
      accuracy = result [1] #test score
      params = result [2]  #grid search parameters

      print('Algorithm:', algorithm,  'Score=',  "{:.4f}".format(score),
            'Accuracy=', "{:.4f}".format(accuracy), 'Sensor_id=',
            sensor_id ,'Time=', start_time, ', end_time+1,
            'params',params,'length=',len(final_df) ,
            file=open("output_3_3.txt", "a"))
```

Appendix D: Graphs

```
df_new['date_insert'] = pd.to_datetime(df_new['date_insert'])

mask = (df_new['date_insert'] > '2019-8-1') &
       (df_new['date_insert'] < '2020-9-1')
df_new = df_new.loc[mask]

df_new = df_new[(df_new.sensor_id != 4)  & (df_new.sensor_id != 5) &
                (df_new.sensor_id != 6)  & (df_new.sensor_id != 11) &
                (df_new.sensor_id != 34)  & (df_new.sensor_id != 35)]

df_new = df_new[(df_new.prox == 1)]
df_new['date_insert'] = df_new['date_insert'].dt.floor('30min')

df_graph= df_new.loc[:, ['date_insert', 'jamfactor', 'sensor_id']]
df_graph['date_insert'] = pd.to_datetime(df_graph['date_insert'],
                          format='%Y-%m-%d')
```

```
df_temp= pd.read_csv('/content/drive/My Drive/thesis_colab/
                     temp_02.09.20.csv',  sep=';', encoding="ISO-8859-1")
mask = df_temp['date_insert'] < '2020-9-1'

df_temp = df_temp.loc[mask]

df_temp = df_temp[(df_temp.sensor_id != 4)  & (df_temp.sensor_id != 5) &
                (df_temp.sensor_id != 6)  & (df_temp.sensor_id != 11) &
                (df_temp.sensor_id != 34)  & (df_temp.sensor_id != 35)]

city_dict = {10:"Athens", 13:"Athens", 14:"Athens",16:"Athens",18:"Athens",
             19:"Athens",20:"Athens",25:"Athens",30:"Athens",31 :"Athens",
```

```python
                7:"Thessaloniki",8:"Thessaloniki",9:"Thessaloniki",
                12:"Thessaloniki",15:"Thessaloniki",17:"Thessaloniki",
                21:"Thessaloniki", 22:"Thessaloniki", 23:"Thessaloniki",
                24:"Thessaloniki",26:"Thessaloniki",27:"Thessaloniki",
                28:"Thessaloniki",29:"Thessaloniki",33:"Thessaloniki",
                36:"Thessaloniki"}
df_temp ['city'] = df_temp["sensor_id"].map(lambda x:city_dict[x])

df_temp['date_insert'] = pd.to_datetime(df_temp.date_insert)
df_temp['date_insert'] = df_temp['date_insert'].dt.floor('30min')

df_temp['month'] = df_temp['date_insert'].dt.strftime('%B')

```

Code for Figure 9: Jam Factor during 07.2019-08.2020.

```python
plt.figure(dpi=1200)
df_graph.plot(x ='date_insert', y='jamfactor', kind = 'scatter',
              figsize=(20, 10))

plt.xlim(dt.datetime(2019, 8, 1), dt.datetime(2020, 8, 17))
plt.xlabel('Date', fontsize=28)
plt.ylabel('Jam Factor', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)

plt.axvline(dt.datetime(2020, 3, 14), color ='black', linestyle="--", ymin=0.04)
plt.axvline(dt.datetime(2020, 3, 22), color ='red', linestyle="--", ymin=0.04)
plt.axvline(dt.datetime(2020, 5, 4), color ='green', linestyle="--", ymin=0.04, ymax = 0.85)



plt.axvline(dt.datetime(2020, 5, 17),
            color ='red', linestyle="--",ymin=0.04)
plt.text(dt.datetime(2020, 3, 27),6.2,'Movement ', color ='red', size =22)
plt.text(dt.datetime(2020, 3, 27),5.9,'Restrictions', color ='red', size =22)


plt.savefig('year')
plt.show()
```

Code for Figure 10: Average Jam Factor per Day.

```python
df_graph_day = df_graph.groupby(['day'])['jamfactor'].mean()
print(df_graph_day.head(7))
df_graph_days = pd.DataFrame(df_graph_day)
df_graph_days['day'] = df_graph_days.index
df_graph_days.index.names = ['Weekday']
print(df_graph_days.head(7))

cats = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
df_graph_days['day'] = pd.Categorical(df_graph_days['day'], categories=cats, ordered=True)
df_graph_days = df_graph_days.sort_values('day')

plt.figure(dpi=1200)

plt.figure(figsize=(20,9))
sns.barplot(data = df_graph_days, x='day',y='jamfactor')
sns.lineplot(df_graph_days['day'], df_graph_days['jamfactor'], color='orange',
linewidth = 2.0)
bars = plt.bar(df_graph_days['day'], height=df_graph_days['jamfactor'], width=
.8)

plt.xlabel('Day', fontsize=28)
plt.ylabel('Jam Factor', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)
for bar in bars:
    yval = round(bar.get_height(),2)
    plt.text(bar.get_x(), yval + .005, yval,fontsize=25)
plt.xlabel('Day', fontsize=28)
plt.ylabel('Jam Factor', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)
plt.savefig('days')

plt.show()
```

Code for Figure 12: Temperature monthly average.

```
plt.figure(dpi=1200)
plt.figure(figsize=(30, 10))
bars = plt.bar(df_graph_months['month'],
                height=df_graph_months['temp'], width=.8 )


for bar in bars:
    yval = round(bar.get_height(),1)
    plt.text(bar.get_x(), yval + .005, yval, fontsize=30 )

plt.xlabel('Month', fontsize=28)
plt.ylabel('Temperature', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)

plt.savefig('temp_month')
plt.show()
```

Code for Figure 13: Humidity monthly average

```
df_graph_month = df_temp.groupby(['month'])['humidity'].mean()
print(df_graph_month)
df_graph_months = pd.DataFrame(df_graph_month)
df_graph_months['month'] = df_graph_months.index
df_graph_months.index.names = ['Month']
print(df_graph_months)

cats = ['January', 'February', 'March', 'April', 'May',
        'June', 'July', 'August', 'September',
        'October', 'November', 'December']
df_graph_months['month'] = pd.Categorical(df_graph_months['month'],
                           categories=cats, ordered=True)
df_graph_months = df_graph_months.sort_values('month')
print(df_graph_months)

plt.figure(dpi=1200)
plt.figure(figsize=(30, 10))

bars = plt.bar(df_graph_months['month'],
                height=df_graph_months['humidity'], width=.8 )

```

```python
for bar in bars:
    yval = round(bar.get_height(),1)
    plt.text(bar.get_x(), yval + .005, yval, fontsize=20 )

plt.xlabel('Month', fontsize=28)
plt.ylabel('Humidity', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)

plt.savefig('month_humidity')
plt.show()
```

Code for Figure 15: Jam Factor before, throughout and after lockdown, featuring all time intervals.

```python
df_graph ['city'] = df_graph["sensor_id"].map(lambda x:city_dict[x])
df_interval= df_new
df_interval['hour'] = df_new['date_insert'].dt.strftime('%H')
cat = {
    "07" :"Morning",     "08" :"Morning",     "09" :"Morning",
    "12" :"Noon", "13" :"Noon", "14" :"Noon",
    "15" :"Afternoon",     "16" :"Afternoon",     "17" :"Afternoon",
    "19" :"Evening", "20" :"Evening", "21" :"Evening"
}
df_interval["interval"] = df_interval['hour'].map(cat)
df_interval = df_interval[df_interval['interval'].notna()]

df_interval['date_insert']=df_interval['date_insert'].dt.date
df_interval_group = df_interval.groupby(['date_insert','interval']).mean()

import matplotlib.patches as mpatches

plt.figure(dpi=1200)

plt.figure(figsize=(20,9))
sns.lineplot(data = df_interval_group, x='date_insert',y='jamfactor',
             hue='interval')

plt.ylim(0, 4.5)
plt.xlim([dt.date(2020, 1, 27), dt.date(2020, 7, 12)])
plt.xlabel('Date', fontsize=28)
plt.ylabel('Jam Factor', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)

plt.axvline(dt.datetime(2020, 3, 14),
            color ='black', linestyle="--", ymin=0.04)
plt.axvline(dt.datetime(2020, 3, 22),
            color ='red', linestyle="--", ymin=0.04)
plt.axvline(dt.datetime(2020, 5, 4), color ='green',
            linestyle="--", ymin=0.04, ymax = 0.85)

plt.axvline(dt.datetime(2020, 5, 17),
            color ='red', linestyle="--",ymin=0.04)
plt.text(dt.datetime(2020, 3, 30),4.2,
            'Movement      Restrictions', color ='red', size =22)

handles, labels = plt.gca().get_legend_handles_labels()
myorder = [2, 3, 0, 1]
labels = [labels[i] for i in myorder]
handles = [handles [i] for i in myorder]

plt.legend(fontsize='18', handles = handles, labels = labels )

plt.savefig('intervals_corona')
plt.savefig('intervals_corona.tif', dpi=600)

plt.show()
```

Code for Figure 18: Average Hourly Jam Factor on weekdays vs. weekends.

```
df_weekends_hour['weekend']= df_weekends_hour['date_insert'].dt.dayofweek
df_weekends_hour['weekend'] = (df_weekends_hour['weekend']
                                // 5 == 1).astype(int)
df_weekends_hour['hour'] = df_weekends_hour['date_insert'].dt.strftime('%H')

plt.figure(dpi=1200)

plt.figure(figsize=(35,30))

sns.catplot(x = 'hour', y='jamfactor',
            hue = 'weekend',data=df_weekends_hour,
            kind='bar',  height=8.27, aspect=11.7/8.27, palette=["C0", "C1"])

red_patch1 = mpatches.Patch( label='Hourly Traffic on weekdays', color='C0')
red_patch2 = mpatches.Patch( label='Hourly Traffic on weekdends', color='C1')

plt.legend(handles=[red_patch1,red_patch2],fontsize=16)
plt.xlabel('Hour', fontsize=28)
plt.ylabel('Jam Factor', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)

plt.savefig('myimage_weekday_weekends_hour')

plt.show(sns)
```

Code for Figure 23: Average Jam Factor in Athens and Thessaloniki during 07.2019-08.2020.

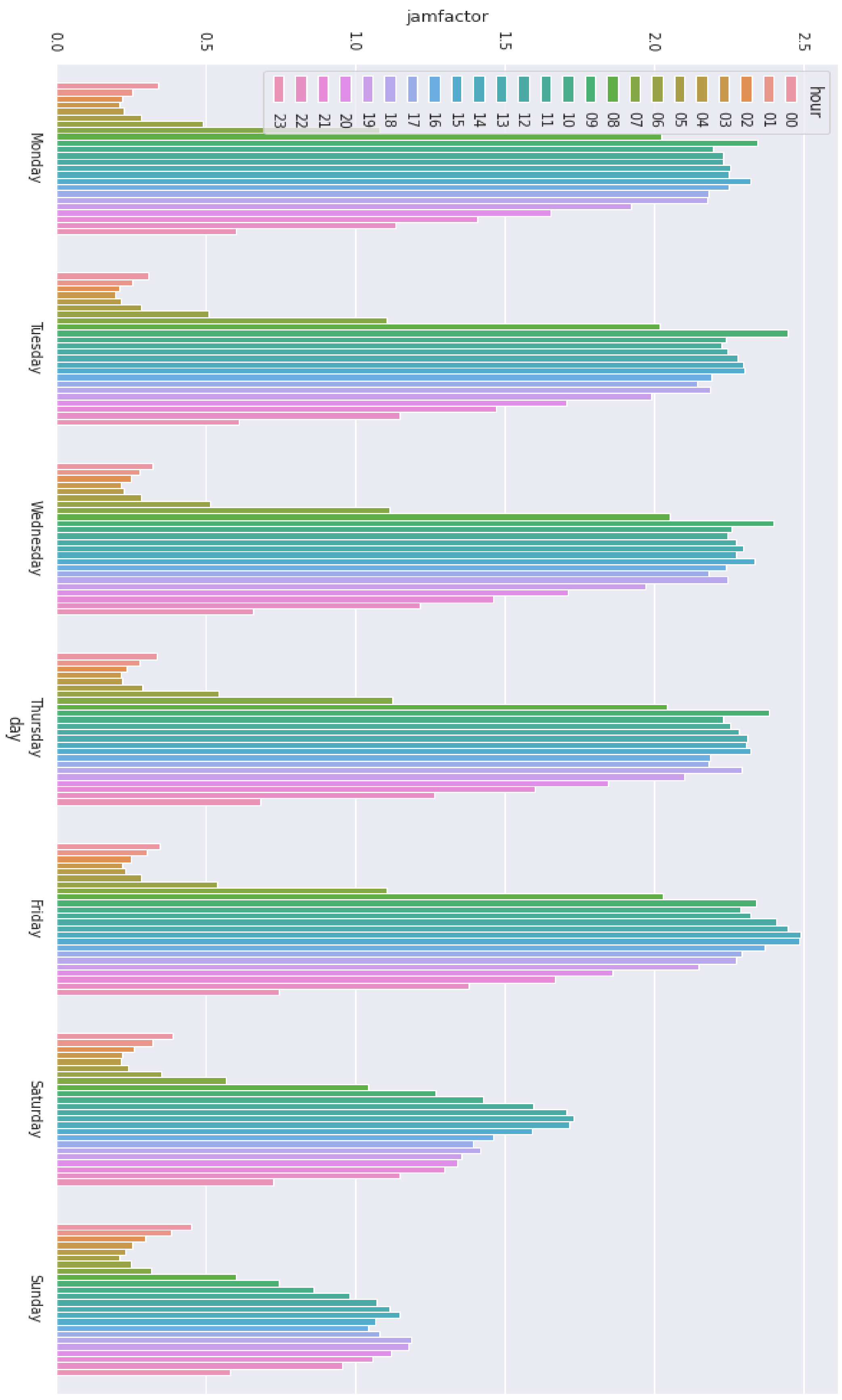
```
df_city = df_graph.groupby(['date_insert','city']).mean()

plt.figure(dpi=1200)
plt.figure(figsize=(20,9))
sns.lineplot(data = df_city, x='date_insert',y='jamfactor', hue='city')
plt.xlim(dt.datetime(2019, 8, 1), dt.datetime(2020, 8, 17))

plt.xlabel('Date', fontsize=28)
plt.ylabel('Jam Factor', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)

plt.axvline(dt.datetime(2020, 3, 14), color ='black',
            linestyle="--", ymin=0.04)
plt.axvline(dt.datetime(2020, 3, 22), color ='red',
            linestyle="--", ymin=0.04)
plt.axvline(dt.datetime(2020, 5, 17),  color ='red',
            linestyle="--",ymin=0.04)
plt.text(dt.datetime(2020, 3, 27),2,'Movement ', color ='red', size =22)
plt.text(dt.datetime(2020, 3, 27),1.8,'Restrictions', color ='red', size =22)
plt.legend(fontsize='13')

plt.savefig('cities_year')
plt.show()
```

Code for Figure 24: Average Jam Factor in Athens and Thessaloniki before, during and after the lockdown.

```python
mask = (df_city['date_insert'] > '2020-1-26') &
       (df_city['date_insert'] < '2020-7-12')
df_city_corona = df_city.loc[mask]

plt.figure(dpi=1200)
plt.figure(figsize=(20,9))
sns.lineplot(data = df_city_corona, x='date_insert',y='jamfactor', hue='city')

plt.ylim(0.4, 2.5)
plt.xlabel('Date', fontsize=28)
plt.ylabel('Jam Factor', fontsize=26)
plt.xticks(fontsize=24)
plt.yticks(fontsize=22)

plt.axvline(dt.datetime(2020, 3, 22), color ='red',
             linestyle="--", ymin=0.04)
plt.axvline(dt.datetime(2020, 5, 17),  color ='red',
             linestyle="--",ymin=0.04)
plt.text(dt.datetime(2020, 4, 7),2.3,'Movement ', color ='red', size =22)
plt.text(dt.datetime(2020, 4, 7),2.2,'Restrictions', color ='red', size =22)

plt.text(dt.datetime(2020, 2, 10),2.3,'Free Movement ', color ='red',
         size =22)
plt.text(dt.datetime(2020, 2, 10),2.2,'Before Covid- 19',
         color ='red', size =22)

plt.text(dt.datetime(2020, 5, 20),2.3,'Free Movement ', color ='red',
         size =22)
plt.text(dt.datetime(2020, 5, 20),2.2,'After Covid-19',
         color ='red', size =22)

plt.legend(fontsize='18')
plt.savefig('cities_year_corona.tif', dpi=600)

plt.show()
```

Code for Figure: Average Jam Factor per day per hour

```python
df_graph_day_hour = df_graph.groupby(['hour', 'day'])['jamfactor'].mean().sort_values()
day_order = ["Monday", "Tuesday", "Wednesday", "Thursday",
             "Friday", "Saturday", "Sunday"]
plt.figure(dpi=1200)

plt.figure(figsize=(25,10))
df_graph_day_hour = df_graph.groupby(['hour', 'day'])['jamfactor'].mean().reset_index().sort_values('hour')
sns.barplot(x = 'day',  hue ='hour',y = 'jamfactor',data = df_graph_day_hour, order = day_order, edgecolor="white")

plt.show()
```


jamfactor
0.0
0.5
1.0
1.5
2.0
2.5
hour
00
01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
day

Appendix E: Results 2-classes

Model evolution results:

whole dataset 2-class																
01.08.19-31.08.20	07--10				12--15				15--18				19--22			
Sensors	RF SCORE															
	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL
Sensor 7	0.6591	0.7143	0.8178	0.8611	0.6432	0.6713	0.8443	0.8654	0.6271	0.659	0.7922	0.8854	0.714	0.736	0.751	0.8217
Sensor 8	0.6218	0.6735	0.8168	0.8686	0.6217	0.6745	0.8304	0.8764	0.6343	0.6844	0.8067	0.8816	0.6766	0.7209	0.7772	0.8587
Sensor 9	0.6527	0.6818	0.8137	0.8842	0.6355	0.6949	0.8256	0.9037	0.6264	0.6875	0.8218	0.9132	0.6725	0.7382	0.7889	0.7966
Sensor 10	0.6974	0.7379	0.8525	0.9226	0.6543	0.7132	0.796	0.8756	0.701	0.7376	0.8403	0.9168	0.7375	0.7444	0.7905	0.8208
Sensor 12	0.6395	0.6722	0.8102	0.8462	0.6587	0.6907	0.7279	0.6979	0.6569	0.6755	0.7709	0.8448	0.6087	0.6587	0.755	0.8199
Sensor 13	0.6908	0.7075	0.8591	0.9095	0.6414	0.6872	0.8122	0.8916	0.6778	0.7011	0.847	0.8935	0.7274	0.7543	0.7808	0.8305
Sensor 14	0.6906	0.7446	0.8757	0.9259	0.636	0.7048	0.8068	0.8758	0.7009	0.7274	0.837	0.9034	0.7444	0.7545	0.7939	0.8374
Sensor 15	0.6142	0.6454	0.8248	0.8819	0.6177	0.6582	0.8276	0.8779	0.6141	0.6344	0.819	0.8903	0.6653	0.6932	0.7892	0.8105
Sensor 16	0.7171	0.7007	0.8234	0.8916	0.7186	0.7452	0.7249	0.7783	0.6819	0.7054	0.7841	0.8398	0.6376	0.6781	0.8258	0.8949
Sensor 17	0.6272	0.6422	0.803	0.8593	0.6616	0.6856	0.824	0.8609	0.6483	0.6755	0.8016	0.8766	0.6776	0.7209	0.742	0.8018
Sensor 18	0.6823	0.7438	0.8141	0.8955	0.6588	0.7053	0.8168	0.8857	0.69	0.7367	0.8433	0.9133	0.7444	0.7645	0.7939	0.8275
Sensor 19	0.6941	0.7074	0.849	0.9129	0.7005	0.74	0.7895	0.8585	0.6911	0.7375	0.8505	0.9169	0.6712	0.7142	0.8172	0.8837
Sensor 20	0.6438	0.6636	0.839	0.8424	0.677	0.7172	0.8257	0.8685	0.6746	0.7075	0.8437	0.8669	0.6047	0.6381	0.7408	0.7571
Sensor 21	0.6437	0.6742	0.7984	0.8365	0.631	0.6648	0.8101	0.8711	0.6031	0.6339	0.8045	0.8732	0.6392	0.6817	0.756	0.8159
Sensor 22	0.6194	0.6871	0.8214	0.8808	0.6748	0.6786	0.8381	0.8691	0.645	0.6626	0.7958	0.8852	0.7177	0.7509	0.7396	0.8104
Sensor 23	0.6293	0.6612	0.832	0.8565	0.6261	0.7058	0.8149	0.9053	0.6746	0.7142	0.8028	0.8831	0.6966	0.7534	0.7382	0.7762
Sensor 24	0.6413	0.6967	0.854	0.9056	0.6278	0.6782	0.8337	0.9034	0.638	0.6698	0.8232	0.8905	0.6735	0.7222	0.792	0.8205
Sensor 25	0.6906	0.7274	0.869	0.896	0.6713	0.7072	0.8255	0.882	0.7142	0.7309	0.8204	0.8835	0.7542	0.7711	0.7772	0.8075
Sensor 26	0.6378	0.683	0.8447	0.8777	0.6185	0.6963	0.7037	0.6889	0.6446	0.69	0.7718	0.854	0.6354	0.6698	0.7775	0.844
Sensor 27	0.6167	0.6641	0.8406	0.8707	0.6432	0.7055	0.756	0.7515	0.6485	0.6732	0.7863	0.842	0.6313	0.6643	0.77	0.8688
Sensor 28	0.6565	0.6637	0.8057	0.8422	0.6421	0.694	0.7159	0.782	0.6755	0.7085	0.7798	0.8322	0.6595	0.694	0.822	0.8454
Sensor 29	0.6442	0.6408	0.8333	0.8188	0.6818	0.7021	0.8117	0.856	0.5953	0.6368	0.8063	0.8406	0.602	0.5948	0.7262	0.733
Sensor 30	0.6692	0.6854	0.8048	0.8605	0.6452	0.6805	0.8473	0.8972	0.6857	0.7071	0.8	0.8786	0.7111	0.763	0.763	0.8
Sensor 31	0.6746	0.7131	0.8649	0.8955	0.6762	0.7026	0.8452	0.9132	0.7131	0.7202	0.8657	0.9232	0.7812	0.7927	0.8117	0.8195
Sensor 33	0.638	0.6632	0.8228	0.8618	0.6117	0.6538	0.8292	0.8951	0.5846	0.64	0.8662	0.9129	0.6768	0.71	0.766	0.8528
Sensor 36	0.634	0.6671	0.8377	0.8699	0.6752	0.6623	0.8321	0.8696	0.6413	0.6411	0.8424	0.8803	0.6943	0.703	0.8049	0.8269
Average	0.654842	0.686996154	0.831861538	0.875930769	0.651919	0.693069231	0.804426923	0.853869	0.657226923	0.688376923	0.816280769	0.881608	0.682873077	0.714880769	0.776557692	0.822385

whole dataset 2-class																
01.08.19-31.08.20	07--10				12--15				15--18				19--22			
Sensors	RF ACUURACY															
	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL
Sensor 7	0.5873	0.5873	0.8413	0.873	0.5775	0.6901	0.7887	0.8732	0.5571	0.6143	0.8429	0.8571	0.6912	0.7059	0.8088	0.8235
Sensor 8	0.6349	0.746	0.8571	0.8889	0.5352	0.7465	0.8169	0.9014	0.5714	0.5714	0.8714	0.8714	0.6765	0.7353	0.7353	0.7941
Sensor 9	0.6393	0.6721	0.8197	0.8525	0.7353	0.6471	0.75	0.8382	0.7015	0.597	0.7164	0.806	0.7344	0.7969	0.7188	0.8438
Sensor 10	0.5467	0.6267	0.8267	0.92	0.6316	0.6711	0.7105	0.8684	0.6184	0.6579	0.75	0.8947	0.8026	0.7895	0.7632	0.8289
Sensor 12	0.6129	0.6935	0.8226	0.9032	0.6087	0.6377	0.5942	0.6812	0.7206	0.5735	0.6765	0.7059	0.6061	0.6515	0.7121	0.7879
Sensor 13	0.5867	0.5867	0.8667	0.9067	0.6316	0.6711	0.8158	0.8947	0.6316	0.7105	0.8816	0.8816	0.7368	0.75	0.6974	0.7763
Sensor 14	0.5467	0.6267	0.8133	0.92	0.6364	0.6753	0.9091	0.9221	0.5526	0.6316	0.7763	0.8947	0.7368	0.7895	0.7632	0.8158
Sensor 15	0.6818	0.6515	0.7273	0.7727	0.5867	0.6	0.7867	0.88	0.6081	0.6486	0.8378	0.8514	0.7042	0.7606	0.7324	0.8873
Sensor 16	0.6351	0.6351	0.7973	0.9324	0.75	0.7763	0.6711	0.8289	0.7922	0.8052	0.8831	0.8701	0.5789	0.6184	0.8289	0.8947
Sensor 17	0.6667	0.697	0.7727	0.8485	0.5676	0.6081	0.7297	0.9054	0.5946	0.7297	0.8649	0.8649	0.7042	0.7746	0.7183	0.8169
Sensor 18	0.6533	0.6133	0.9067	0.9467	0.6753	0.6883	0.8961	0.9351	0.6842	0.7237	0.7368	0.8553	0.7632	0.8026	0.7368	0.8158
Sensor 19	0.6	0.6133	0.7467	0.9067	0.7105	0.6842	0.7105	0.8421	0.6184	0.6974	0.8158	0.8947	0.6579	0.6974	0.8026	0.8421
Sensor 20	0.6	0.68	0.76	0.7733	0.6842	0.6974	0.7105	0.7895	0.6447	0.7237	0.8158	0.8026	0.6447	0.5921	0.75	0.6974
Sensor 21	0.6061	0.6515	0.8333	0.9091	0.6216	0.5946	0.8108	0.9324	0.6081	0.6757	0.7703	0.9324	0.6761	0.7746	0.7746	0.8732
Sensor 22	0.5714	0.6349	0.7778	0.8571	0.5775	0.662	0.7746	0.8732	0.5714	0.6286	0.8	0.8143	0.6471	0.6765	0.7353	0.7941
Sensor 23	0.5645	0.6935	0.7581	0.871	0.6957	0.6377	0.7101	0.7826	0.5942	0.6522	0.8406	0.8696	0.697	0.803	0.8182	0.8333
Sensor 24	0.6949	0.7458	0.7966	0.8475	0.6154	0.5846	0.8615	0.8615	0.5312	0.6719	0.7344	0.8906	0.6452	0.6935	0.7903	0.7903
Sensor 25	0.6667	0.5733	0.88	0.9067	0.5974	0.6623	0.6623	0.8052	0.6316	0.6842	0.7895	0.8684	0.7632	0.8158	0.7632	0.8158
Sensor 26	0.6885	0.6557	0.8689	0.8525	0.7353	0.7206	0.7647	0.75	0.6866	0.6418	0.791	0.7463	0.6308	0.5077	0.7846	0.7846
Sensor 27	0.7797	0.7119	0.8305	0.8136	0.6615	0.6615	0.7231	0.7538	0.7188	0.8438	0.7812	0.875	0.5738	0.623	0.5902	0.6557
Sensor 28	0.5806	0.5645	0.7742	0.8871	0.6912	0.6618	0.7059	0.7353	0.7015	0.6866	0.7612	0.8657	0.5538	0.5692	0.7538	0.8308
Sensor 29	0.662	0.662	0.8873	0.8873	0.5753	0.6575	0.7671	0.8493	0.589	0.6301	0.7945	0.863	0.6438	0.7534	0.6849	0.7808
Sensor 30	0.5556	0.6667	0.8095	0.9206	0.6197	0.7042	0.8169	0.9155	0.5571	0.6429	0.8857	0.8714	0.7353	0.7941	0.7353	0.7794
Sensor 31	0.6462	0.6462	0.8615	0.8308	0.6716	0.6716	0.8358	0.8657	0.6364	0.6515	0.8636	0.8333	0.7424	0.697	0.7121	0.7727
Sensor 33	0.7119	0.6102	0.8475	0.8305	0.6769	0.7077	0.8154	0.8462	0.6719	0.7812	0.8281	0.8594	0.6557	0.6885	0.7049	0.8033
Sensor 36	0.6727	0.6909	0.7818	0.8364	0.6333	0.6333	0.9	0.9	0.6441	0.5593	0.8136	0.8644	0.7193	0.7193	0.7544	0.7719
Average	0.630469	0.651396154	0.817888462	0.872876923	0.642423	0.667407692	0.770692308	0.847342	0.632203846	0.67055	0.804730769	0.854008	0.681576923	0.714611538	0.744984615	0.804246

Whole dataset (01.08.2019-31.08.20) results:

whole dataset 2-class																
01.08.19-31.08.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8611	0.873	0.8409	0.8254	0.8654	0.8732	0.8552	0.8451	0.8854	0.8571	0.8819	0.8571	0.8217	0.8235	0.7991	0.8382
Sensor 8	0.8686	0.8889	0.8366	0.8889	0.8764	0.9014	0.8835	0.9014	0.8816	0.8714	0.878	0.8714	0.8587	0.7941	0.855	0.8088
Sensor 9	0.8842	0.8525	0.8633	0.8361	0.9037	0.8382	0.9037	0.8382	0.9132	0.806	0.9131	0.806	0.7966	0.8438	0.7768	0.8438
Sensor 10	0.9226	0.92	0.8892	0.8667	0.8756	0.8684	0.8427	0.8816	0.9168	0.8947	0.8871	0.8816	0.8208	0.8289	0.834	0.8026
Sensor 12	0.8462	0.9032	0.8462	0.9032	0.6979	0.6812	0.6976	0.6522	0.8448	0.7059	0.8484	0.7059	0.8199	0.7879	0.8084	0.7879
Sensor 13	0.9095	0.9067	0.8759	0.8667	0.8916	0.8947	0.8883	0.8947	0.8935	0.8816	0.8835	0.8816	0.8305	0.7763	0.8173	0.7895
Sensor 14	0.9259	0.92	0.8892	0.88	0.8758	0.9221	0.8726	0.9221	0.9034	0.8947	0.8804	0.8816	0.8374	0.8158	0.8206	0.7895
Sensor 15	0.8819	0.7727	0.8781	0.7727	0.8779	0.88	0.8576	0.88	0.8903	0.8514	0.8834	0.8649	0.8105	0.8873	0.7817	0.8451
Sensor 16	0.8916	0.9324	0.8847	0.9324	0.7783	0.8289	0.7814	0.8421	0.8398	0.8701	0.8394	0.8571	0.8949	0.8947	0.8883	0.8947
Sensor 17	0.8593	0.8485	0.8477	0.8485	0.8609	0.9054	0.8545	0.8784	0.8766	0.8649	0.8662	0.8649	0.8018	0.8169	0.7879	0.8028
Sensor 18	0.8955	0.9467	0.8789	0.9333	0.8857	0.9351	0.8559	0.9221	0.9133	0.8553	0.9	0.8289	0.8275	0.8158	0.8274	0.8158
Sensor 19	0.9129	0.9067	0.8928	0.8667	0.8585	0.8421	0.8518	0.8684	0.9169	0.8947	0.8937	0.9079	0.8837	0.8421	0.864	0.8421
Sensor 20	0.8424	0.7733	0.7987	0.8133	0.8685	0.7895	0.8087	0.7763	0.8669	0.8026	0.8208	0.7632	0.7571	0.6974	0.681	0.7368
Sensor 21	0.8365	0.9091	0.8175	0.8788	0.8711	0.9324	0.8647	0.9189	0.8732	0.9324	0.8732	0.9189	0.8159	0.8732	0.8158	0.8732
Sensor 22	0.8808	0.8571	0.8571	0.8095	0.8691	0.8732	0.8515	0.8873	0.8852	0.8143	0.8783	0.8429	0.8104	0.7941	0.7953	0.75
Sensor 23	0.8565	0.871	0.8523	0.8387	0.9053	0.7826	0.9091	0.7826	0.8831	0.8696	0.8611	0.8841	0.7762	0.8333	0.7755	0.8182
Sensor 24	0.9056	0.8475	0.888	0.7797	0.9034	0.8615	0.9034	0.8615	0.8905	0.8906	0.8785	0.8906	0.8205	0.7903	0.8247	0.7903
Sensor 25	0.896	0.9067	0.8657	0.9067	0.882	0.8052	0.8454	0.8182	0.8835	0.8684	0.8705	0.8816	0.8075	0.8158	0.8173	0.8158
Sensor 26	0.8777	0.8525	0.8697	0.8525	0.6889	0.75	0.737	0.75	0.854	0.7463	0.8463	0.7463	0.844	0.7846	0.8363	0.7846
Sensor 27	0.8707	0.8136	0.8707	0.8136	0.7515	0.7538	0.7629	0.7692	0.842	0.875	0.8342	0.8281	0.8688	0.6557	0.8688	0.6557
Sensor 28	0.8422	0.8871	0.842	0.9032	0.782	0.7353	0.7933	0.7353	0.8322	0.8657	0.847	0.8507	0.8454	0.8308	0.8451	0.8
Sensor 29	0.8188	0.8873	0.8083	0.8592	0.856	0.8493	0.8184	0.8493	0.8406	0.863	0.8268	0.8219	0.733	0.7808	0.7086	0.7808
Sensor 30	0.8605	0.9206	0.8326	0.8889	0.8972	0.9155	0.8658	0.9155	0.8786	0.8714	0.875	0.8714	0.8	0.7794	0.7815	0.8088
Sensor 31	0.8955	0.8308	0.884	0.8615	0.9132	0.8657	0.8906	0.8358	0.9232	0.8333	0.9154	0.8788	0.8195	0.7727	0.8385	0.8485
Sensor 33	0.8618	0.8305	0.8618	0.8305	0.8951	0.8462	0.8951	0.8462	0.9129	0.8594	0.9129	0.8594	0.8528	0.8033	0.8322	0.8033
Sensor 36	0.8699	0.8364	0.8608	0.8364	0.8696	0.9	0.8697	0.9	0.8803	0.8644	0.8803	0.8644	0.8269	0.7719	0.8093	0.7368
Average	0.875931	0.872877	0.85895	0.857427	0.853869	0.847342	0.844631	0.845092	0.881608	0.854008	0.872131	0.850431	0.822385	0.804246	0.811169	0.802446

3-month results:

3-month 2-class																
01.08.19-31.10.19	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.725	0.8333	0.745	0.8333	0.76	0.6	0.8267	0.8667	0.88	0.9333	0.85	0.9333	0.8	0.8571	0.8	0.7857
Sensor 8	0.87	0.7273	0.84	0.6364	0.7633	0.8	0.8133	0.8667	0.8633	0.9333	0.8667	0.9333	0.73	0.8571	0.76	0.7143
Sensor 9	0.6	0.7778	0.5833	0.6667	0.765	0.75	0.805	0.75	0.87	0.5	0.87	0.5	0.81	0.8182	0.735	0.7273
Sensor 10	0.9321	0.9474	0.9304	0.9474	0.9571	0.8889	0.9571	0.8333	0.9179	0.8947	0.9071	0.8947	0.9339	0.8947	0.8946	0.8947
Sensor 12	0.75	0.7	0.625	0.8	0.7133	0.5385	0.5433	0.7692	0.7067	0.6923	0.7633	0.6923	0.69	0.5	0.73	0.5833
Sensor 13	0.9321	0.9474	0.9304	0.9474	0.9571	0.8889	0.9429	0.8889	0.9179	0.8947	0.9071	0.8947	0.9464	0.9474	0.9071	0.9474
Sensor 14	0.9446	0.9474	0.9321	0.9474	0.9429	0.8947	0.9304	0.8947	0.9179	0.8947	0.9071	0.8947	0.9446	0.8947	0.8911	0.8947
Sensor 15	0.8867	0.9286	0.8933	0.7857	0.9429	0.8947	0.9429	0.7895	0.9321	0.8421	0.9054	0.8421	0.9	0.7778	0.8714	0.7222
Sensor 16	0.8857	0.8889	0.8857	0.7778	0.6786	0.6842	0.6929	0.8421	0.8804	0.8947	0.8375	0.8947	0.9036	0.8421	0.8929	0.8421
Sensor 17	0.9667	1	0.9	0.9333	0.8357	0.8421	0.8339	0.7368	0.9179	0.9474	0.9196	0.9474	0.8357	0.7222	0.8667	0.7778
Sensor 18	0.9304	0.9474	0.9304	0.9474	0.9429	0.9474	0.9304	0.8947	0.9161	0.8947	0.8893	0.8947	0.8929	0.8947	0.8911	0.8947
Sensor 19	0.9304	0.9474	0.9446	0.9474	0.9571	0.8333	0.9446	0.8333	0.9571	0.9474	0.8893	0.8947	0.9446	0.8421	0.8661	0.8421
Sensor 20	0.9446	0.8947	0.9161	0.8947	0.9304	0.8333	0.9304	0.8333	0.9321	0.8947	0.9304	0.8421	0.8375	0.7895	0.8071	0.7895
Sensor 21	0.93	0.8571	0.91	0.9286	0.8893	0.8333	0.9161	0.7778	0.9429	0.9474	0.9018	0.9474	0.7976	0.9444	0.8405	0.8333
Sensor 22	0.805	0.8333	0.775	0.8333	0.7633	0.7333	0.83	0.8	0.8633	0.8667	0.8333	0.8667	0.6767	0.8571	0.6767	0.7143
Sensor 23	0.675	0.8	0.7	0.8	0.7667	0.7857	0.75	0.7857	0.85	0.7143	0.8867	0.7857	0.73	0.7692	0.73	0.7692
Sensor 24	0.7833	0.7143	0.7	0.4286	0.8167	0.4444	0.8417	0.5556	0.8	0.7778	0.8	0.5556	0.7167	0.75	0.55	0.875
Sensor 25	0.9321	0.9474	0.9304	0.9474	0.9446	0.8889	0.9304	0.8333	0.9179	0.8947	0.9071	0.8947	0.8643	0.8947	0.8089	0.7895
Sensor 26	0.725	0.7	0.7167	0.6	0.625	0.5833	0.58	0.75	0.825	0.8333	0.755	0.75	0.655	0.8182	0.695	0.7273
Sensor 27	0.9	0.4286	0.9333	0.2857	0.6167	0.5556	0.7	0.4444	0.875	0.7778	0.8083	0.6667	0.65	0.25	0.6	0.25
Sensor 28	0.85	0.9	0.925	1	0.75	0.5833	0.765	0.75	0.735	0.8333	0.84	0.8333	0.84	0.6364	0.82	0.7273
Sensor 29	0.7867	0.8571	0.8233	0.8571	0.8833	0.9333	0.85	1	0.9	0.875	0.8667	0.8125	0.6905	0.75	0.7405	0.6875
Sensor 30	0.905	0.9167	0.865	0.8333	0.88	0.8	0.8767	0.7333	0.9333	1	0.9167	0.9333	0.77	0.7857	0.8	0.7857
Sensor 31	0.7417	0.6667	0.8417	0.8889	0.8583	0.6667	0.7917	0.6667	0.8167	0.2222	0.7833	0.7778	0.8167	0.5556	0.8167	0.6667
Sensor 33	0.85	0.1429	0.9	0.4286	0.825	0.5556	0.7333	0.6667	0.8667	0.4444	0.8333	0.5556	0.7833	0.5	0.7167	0.5
Sensor 36	0.81	0.5385	0.9	0.4286	0.675	0.8	0.7333	0.6667	0.8667	0.4444	0.8333	0.5556	0.7833	0.5	0.7167	0.5
Average	0.84585	0.799623	0.845258	0.781731	0.824623	0.752285	0.822769	0.778054	0.876996	0.799819	0.861858	0.807446	0.805512	0.755727	0.785569	0.740062

01.11.19-31.01.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8767	0.7857	0.87	0.8571	0.8405	0.7647	0.7643	0.7647	0.9262	0.875	0.9095	0.875	0.8167	0.6875	0.8167	0.6875
Sensor 8	0.82	0.7857	0.8333	0.7143	0.8262	0.7059	0.7762	0.8235	0.8595	0.875	0.8738	0.875	0.85	0.75	0.85	0.6875
Sensor 9	0.8133	0.7857	0.76	0.8571	0.8071	0.7647	0.7619	0.7647	0.9214	0.875	0.8881	0.875	0.85	0.5625	0.8667	0.625
Sensor 10	0.8286	0.7222	0.8286	0.8333	0.7643	0.9444	0.7911	0.7778	0.7714	0.5556	0.6714	0.8333	0.7857	0.6111	0.7429	0.5
Sensor 12	0.91	0.7857	0.8533	0.8571	0.5643	0.3529	0.6095	0.5294	0.7143	0.6875	0.7	0.8125	0.85	0.875	0.7833	0.6875
Sensor 13	0.8161	0.8333	0.8304	0.8889	0.8482	0.8889	0.75	0.8333	0.8	0.7778	0.7286	0.6667	0.7286	0.6111	0.7286	0.5556
Sensor 14	0.8429	0.8333	0.8286	0.8333	0.8321	0.8333	0.7321	0.8333	0.8286	0.5	0.7286	0.6667	0.7286	0.7222	0.7571	0.5556
Sensor 15	0.86	0.8571	0.87	0.7857	0.7976	0.7647	0.7643	0.7647	0.9429	0.8125	0.9095	0.875	0.8	0.6875	0.8167	0.6875
Sensor 16	0.7833	0.6667	0.7833	0.7222	0.6571	0.5556	0.6143	0.3889	0.8232	0.6316	0.8232	0.6316	0.8536	0.7778	0.8089	0.7778
Sensor 17	0.9067	0.7143	0.8467	0.8571	0.7929	0.7059	0.7619	0.8824	0.8429	0.875	0.7929	0.875	0.8	0.8667	0.7333	0.8
Sensor 18	0.7857	0.6111	0.8143	0.8333	0.8464	0.8889	0.7482	0.7778	0.7857	0.5556	0.7571	0.7778	0.7286	0.6667	0.7286	0.5
Sensor 19	0.8143	0.8333	0.8	0.8889	0.6821	0.6667	0.65	0.7778	0.7571	0.6667	0.8286	0.7778	0.7571	0.5556	0.7429	0.7222
Sensor 20	0.8589	0.8333	0.8161	0.7778	0.7643	0.6667	0.7518	0.7222	0.7714	0.5556	0.7714	0.8333	0.7429	0.7778	0.7143	0.7222
Sensor 21	0.89	0.8571	0.8533	0.7857	0.8381	0.8235	0.7333	0.8235	0.9262	0.875	0.9095	0.875	0.8	0.6875	0.8167	0.6875
Sensor 22	0.8367	0.6429	0.8	0.7143	0.8548	0.7059	0.7619	0.7059	0.9262	0.875	0.8762	0.875	0.7833	0.6875	0.7833	0.6875
Sensor 23	0.8633	0.8571	0.85	0.9286	0.7571	0.7059	0.7881	0.7647	0.7952	0.8125	0.7762	0.875	0.7167	0.6667	0.7	0.7333
Sensor 24	0.8133	0.7143	0.8367	0.7857	0.7643	0.7059	0.719	0.8824	0.8595	0.875	0.8595	0.875	0.85	0.6875	0.85	0.8125
Sensor 25	0.8161	0.8333	0.8304	0.8889	0.7929	0.8947	0.7089	0.9474	0.7857	0.6111	0.7286	0.7778	0.6429	0.5556	0.6429	0.5
Sensor 26	0.9067	0.7143	0.89	0.8571	0.6643	0.5882	0.6524	0.5882	0.8405	0.75	0.8262	0.9375	0.9	0.8125	0.85	0.75
Sensor 27	0.89	0.8571	0.8733	0.8571	0.6952	0.5882	0.6857	0.7647	0.8524	0.8125	0.7762	0.875	0.9	0.8125	0.85	0.75
Sensor 28	0.89	0.8571	0.8733	0.8571	0.6857	0.5294	0.6857	0.7647	0.8357	0.8125	0.7905	0.9375	0.9	0.8125	0.85	0.75
Sensor 29	0.8589	0.8333	0.8161	0.7778	0.7661	0.6667	0.7518	0.7222	0.7714	0.5556	0.7714	0.8333	0.7429	0.7778	0.7143	0.7222
Sensor 30	0.9033	0.7857	0.8467	0.8571	0.7952	0.6471	0.7619	0.8235	0.8429	0.875	0.7738	0.875	0.75	0.6875	0.7667	0.6875
Sensor 31	0.8	0.7222	0.8	0.7778	0.8321	0.8889	0.7607	0.8333	0.8	0.6111	0.7714	0.7778	0.7429	0.5556	0.7286	0.5
Sensor 33	0.8733	0.7857	0.87	0.7143	0.8262	0.8235	0.7786	0.8824	0.9238	0.875	0.9238	0.875	0.8333	0.6875	0.8333	0.6875
Sensor 36	0.8767	0.7857	0.8367	0.7857	0.8405	0.8235	0.7333	0.8235	0.9429	0.875	0.9095	0.875	0.8	0.6875	0.8167	0.6875
Average	0.851338	0.780508	0.835042	0.818973	0.774446	0.726719	0.73065	0.767958	0.840269	0.748392	0.810596	0.832254	0.794377	0.702681	0.780481	0.671688

01.02.20-30.04.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.781	0.9412	0.731	1	0.8262	0.7778	0.8405	0.7222	0.8875	0.7778	0.8429	0.7778	0.8667	0.8824	0.8238	0.8235
Sensor 8	0.8214	1	0.8619	0.9412	0.7976	0.8333	0.7952	0.7778	0.8732	0.8333	0.8714	0.7778	0.8643	0.7647	0.8095	0.7647
Sensor 9	0.7595	0.9412	0.7333	1	0.8405	0.7778	0.8405	0.8333	0.8607	0.7778	0.8589	0.7778	0.8667	0.8235	0.85	0.8824
Sensor 10	0.8667	0.8333	0.8667	0.8333	0.8161	0.7778	0.775	0.8333	0.7732	0.6667	0.8	0.7778	0.8446	0.7778	0.8	0.7222
Sensor 12	0.8643	0.9412	0.8786	0.9412	0.7786	0.8333	0.8214	0.8333	0.8143	0.7778	0.7875	0.7778	0.8667	0.8235	0.8524	0.7059
Sensor 13	0.8714	0.8889	0.8405	0.8333	0.8893	0.8333	0.9036	0.8333	0.8036	0.7778	0.7589	0.8333	0.7589	0.7778	0.7732	0.7778
Sensor 14	0.881	0.8889	0.8667	0.8889	0.85	0.8333	0.8375	0.7778	0.7875	0.6667	0.7857	0.7778	0.7875	0.7778	0.8	0.7222
Sensor 15	0.7929	0.8824	0.7762	0.8824	0.8119	0.7778	0.8405	0.7222	0.9	0.7222	0.8429	0.7778	0.8524	0.7647	0.8238	0.7059
Sensor 16	0.8952	0.8889	0.9095	0.8333	0.8286	0.8889	0.8286	0.7778	0.8071	0.7778	0.8054	0.7222	0.7964	0.8333	0.8232	0.8889
Sensor 17	0.6976	1	0.7738	0.8824	0.75	0.8235	0.75	0.9412	0.8875	0.8333	0.8161	0.7778	0.8548	0.8824	0.8238	0.8235
Sensor 18	0.869	0.8824	0.8238	0.8824	0.8304	0.7778	0.8589	0.8333	0.7446	0.6667	0.8	0.7778	0.8018	0.7222	0.8	0.7222
Sensor 19	0.8571	0.6111	0.8548	0.7778	0.8036	0.8333	0.8036	0.8333	0.8589	0.7778	0.8589	0.7778	0.8161	0.8889	0.8143	0.7778
Sensor 20	0.8381	0.8333	0.8238	0.8333	0.8321	0.7222	0.8607	0.7778	0.8321	0.8333	0.8161	0.8889	0.8607	0.6667	0.8036	0.7222
Sensor 21	0.7595	0.8824	0.7476	0.8824	0.8119	0.7778	0.8262	0.7778	0.8446	0.8333	0.8714	0.7778	0.8381	0.7059	0.8524	0.7647
Sensor 22	0.7786	0.8824	0.7381	0.8824	0.869	0.8889	0.8405	0.7778	0.8464	0.7778	0.875	0.7778	0.7952	0.8824	0.8071	0.8235
Sensor 23	0.7119	1	0.7286	0.9412	0.7357	0.8824	0.75	0.9412	0.8732	0.8333	0.7875	0.7778	0.8262	0.7647	0.8238	0.7647
Sensor 24	0.7762	0.9412	0.8048	0.9412	0.8429	0.8333	0.8262	0.8333	0.8875	0.8333	0.8304	0.8333	0.85	0.7059	0.8238	0.7059
Sensor 25	0.869	0.8333	0.8381	0.8333	0.8036	0.6667	0.8036	0.6667	0.8	0.6667	0.7714	0.7222	0.7893	0.8889	0.7732	0.8333
Sensor 26	0.8333	0.9412	0.8476	0.8824	0.8071	0.8333	0.8381	0.8333	0.8143	0.8889	0.7875	0.8889	0.8238	0.7059	0.8214	0.7059
Sensor 27	0.8071	0.9412	0.819	0.8824	0.7786	0.8889	0.8405	0.8333	0.7875	0.7222	0.7875	0.8889	0.7929	0.7059	0.781	0.7647
Sensor 28	0.7905	0.9412	0.819	0.8824	0.7667	0.8889	0.8405	0.8333	0.7875	0.7222	0.7875	0.8889	0.7929	0.7059	0.781	0.7647
Sensor 29	0.8381	0.8333	0.8238	0.8333	0.8321	0.8333	0.8607	0.7778	0.8321	0.8333	0.8161	0.8889	0.8607	0.7222	0.8036	0.7222
Sensor 30	0.7286	1	0.7286	0.7647	0.7643	0.8824	0.7643	0.9412	0.8875	0.8333	0.8161	0.7778	0.8548	0.8824	0.8238	0.8235
Sensor 31	0.8952	0.8333	0.8952	0.8333	0.8179	0.7778	0.8179	0.8333	0.7589	0.7778	0.8	0.7778	0.7875	0.7778	0.8	0.7222
Sensor 33	0.7881	0.8824	0.7476	0.9412	0.7357	0.7778	0.7667	0.7778	0.9	0.7778	0.8857	0.7778	0.8548	0.8824	0.7952	0.8824
Sensor 36	0.7595	0.8824	0.731	0.8824	0.8119	0.8889	0.8095	0.7778	0.8857	0.7778	0.8857	0.7222	0.8381	0.8235	0.8381	0.7059
Average	0.812723	0.897196	0.808062	0.881235	0.808935	0.819635	0.820796	0.811585	0.835977	0.775642	0.821019	0.797019	0.828535	0.789981	0.812385	0.770108

01.05.20-31.07.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8405	0.8889	0.8548	0.8889	0.8179	0.7778	0.8196	0.7778	0.8732	0.8333	0.9018	0.8333	0.7548	0.7222	0.7833	0.6667
Sensor 8	0.8786	0.7778	0.8786	0.7778	0.8732	0.7778	0.8732	0.7778	0.8732	0.8333	0.8875	0.7778	0.8	0.5	0.7548	0.5
Sensor 9	0.9119	0.8333	0.8976	0.8333	0.8875	0.8889	0.8732	0.8889	0.8589	0.7778	0.8589	0.7778	0.8405	0.7222	0.8119	0.7222
Sensor 10	0.8643	0.8889	0.8667	0.8333	0.9	0.8889	0.9	0.8889	0.9018	0.8333	0.9304	0.8333	0.8857	0.8333	0.8857	0.8333
Sensor 12	0.8548	0.7778	0.8548	0.7778	0.7214	0.6667	0.7339	0.5556	0.8893	0.8889	0.8893	0.8889	0.8548	0.9444	0.8119	0.9444
Sensor 13	0.9095	0.7778	0.8952	0.7778	0.9304	0.8333	0.9161	0.8333	0.9571	0.8333	0.9161	0.8333	0.869	0.7778	0.869	0.7778
Sensor 14	0.8952	0.8889	0.8952	0.8889	0.9018	0.8889	0.9018	0.8889	0.8446	0.8333	0.9143	0.8333	0.869	0.7778	0.869	0.7778
Sensor 15	0.869	0.8333	0.869	0.8333	0.8607	0.7222	0.8607	0.7222	0.9018	0.7778	0.9161	0.7778	0.7833	0.7778	0.7833	0.7222
Sensor 16	0.9095	0.9444	0.8952	0.9444	0.8589	0.8889	0.8446	0.8889	0.8714	0.6667	0.9	0.6667	0.9143	0.8333	0.9	0.8333
Sensor 17	0.8643	0.7778	0.8357	0.7778	0.8875	0.8333	0.8732	0.8333	0.8893	0.7778	0.8893	0.7778	0.8262	0.7778	0.7548	0.7222
Sensor 18	0.8976	0.8889	0.8833	0.8889	0.9	0.8889	0.9	0.8889	0.8875	0.8889	0.9018	0.8889	0.8857	0.8333	0.8857	0.8333
Sensor 19	0.8833	0.9444	0.869	0.9444	0.9161	0.7778	0.9018	0.7778	0.8857	0.9444	0.8714	0.8889	0.869	0.8889	0.869	0.8889
Sensor 20	0.9143	0.8333	0.9286	0.8333	0.9286	0.8889	0.9018	0.8333	0.8464	0.7222	0.8732	0.8333	0.8262	0.8889	0.8714	0.7778
Sensor 21	0.869	0.8889	0.8548	0.8889	0.8464	0.7778	0.8607	0.7778	0.9143	0.8333	0.9286	0.8333	0.7976	0.7222	0.7833	0.7222
Sensor 22	0.8857	0.8889	0.8857	0.8889	0.8339	0.7778	0.8482	0.7778	0.8589	0.7778	0.8875	0.7778	0.7976	0.6667	0.8	0.6667
Sensor 23	0.8643	0.8333	0.8643	0.8333	0.875	0.8333	0.8607	0.8333	0.9036	0.7778	0.8893	0.7778	0.8405	0.7778	0.8262	0.6667
Sensor 24	0.8643	0.7222	0.8357	0.7222	0.8732	0.7778	0.8732	0.7778	0.8446	0.7778	0.8446	0.7778	0.7857	0.6111	0.7571	0.6667
Sensor 25	0.9095	0.8889	0.881	0.8889	0.9143	0.8333	0.9143	0.8333	0.9018	0.8333	0.9018	0.8333	0.869	0.7778	0.869	0.7778
Sensor 26	0.8548	0.8889	0.8548	0.8333	0.8036	0.5556	0.775	0.6667	0.8893	0.8889	0.9179	0.8333	0.8548	0.9444	0.8119	0.9444
Sensor 27	0.8833	0.8889	0.8833	0.8889	0.8161	0.5	0.7893	0.6111	0.8893	0.9444	0.9036	0.8889	0.8548	0.8889	0.8119	0.9444
Sensor 28	0.8833	0.8889	0.8833	0.8889	0.8161	0.4444	0.7893	0.6111	0.9018	0.9444	0.9179	0.8889	0.869	0.8889	0.8119	0.9444
Sensor 29	0.9143	0.8333	0.9286	0.8333	0.9286	0.8889	0.9018	0.8889	0.8464	0.7222	0.8732	0.8333	0.8262	0.8889	0.8571	0.7778
Sensor 30	0.8643	0.7778	0.8357	0.7778	0.8875	0.8333	0.8732	0.8333	0.8893	0.7778	0.8893	0.7778	0.8262	0.7778	0.7548	0.7222
Sensor 31	0.8976	0.8333	0.8833	0.8889	0.9	0.8889	0.9	0.8889	0.8875	0.8889	0.9018	0.8889	0.8857	0.8333	0.8857	0.8333
Sensor 33	0.8667	0.8889	0.8524	0.8889	0.9018	0.7778	0.8875	0.7778	0.8732	0.8333	0.8857	0.8333	0.8429	0.5	0.8119	0.5
Sensor 36	0.8548	0.8889	0.8548	0.8889	0.8054	0.7778	0.8482	0.7778	0.9	0.8333	0.9286	0.8333	0.7857	0.6667	0.769	0.6111
Average	0.88095	0.852562	0.8739	0.850423	0.868688	0.784192	0.862358	0.792738	0.883854	0.824777	0.896919	0.822646	0.839008	0.777777	0.823062	0.760677

4-month results:

4-month 2-class																
01.08.19-30.11.19	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.7976	0.8125	0.781	0.75	0.7875	0.85	0.775	0.75	0.85	0.8	0.825	0.8	0.825	0.8421	0.7304	0.8421
Sensor 8	0.7619	0.5	0.7238	0.875	0.75	0.7	0.8	0.85	0.8625	0.8	0.8625	0.8	0.7964	0.7895	0.7696	0.7895
Sensor 9	0.7933	0.6429	0.7567	0.6429	0.7976	0.5882	0.7476	0.7647	0.8357	0.7647	0.8524	0.8235	0.7333	0.4375	0.75	0.875
Sensor 10	0.9567	1	0.9467	0.9583	0.9256	0.875	0.9256	0.7917	0.9478	0.8	0.9378	0.84	0.96	0.88	0.94	0.88
Sensor 12	0.7633	0.8667	0.7633	0.6	0.6786	0.5556	0.5821	0.4444	0.7089	0.8889	0.7893	0.6667	0.681	0.7647	0.7119	0.7647
Sensor 13	0.9478	1	0.9278	0.9583	0.9356	0.9583	0.9144	0.875	0.9367	0.84	0.9478	0.88	0.9189	0.92	0.8989	0.88
Sensor 14	0.9467	0.9583	0.9267	0.9583	0.9367	0.9583	0.9267	0.9167	0.9589	0.84	0.9589	0.88	0.9289	0.88	0.9189	0.88
Sensor 15	0.85	0.8421	0.8393	0.8947	0.8633	0.9583	0.8833	0.9167	0.9167	0.9583	0.9278	0.9583	0.9	0.913	0.8889	0.913
Sensor 16	0.9333	0.8696	0.9556	0.8696	0.6667	0.7917	0.6978	0.7917	0.9189	0.92	0.9189	0.88	0.8778	0.88	0.8678	0.84
Sensor 17	0.8786	0.85	0.8411	0.9	0.8622	0.875	0.84	0.875	0.9389	0.9167	0.9289	0.9167	0.8181	0.8636	0.7833	0.9091
Sensor 18	0.9467	0.9583	0.9267	0.9583	0.9467	0.9583	0.9367	0.9583	0.9589	0.8333	0.9589	0.875	0.94	0.88	0.91	0.88
Sensor 19	0.9567	1	0.9567	0.9583	0.9044	0.8333	0.9033	0.9167	0.97	0.88	0.96	0.96	0.9278	0.92	0.8878	0.92
Sensor 20	0.8944	0.9167	0.8644	0.8333	0.8178	0.875	0.8489	0.8333	0.9789	0.88	0.9589	0.92	0.8356	0.84	0.8056	0.84
Sensor 21	0.8964	0.8421	0.8964	0.7368	0.8911	0.9167	0.8911	0.8333	0.9356	0.9167	0.9044	0.9167	0.9	0.9091	0.8889	0.8636
Sensor 22	0.75	0.8125	0.781	0.6875	0.7	0.8	0.7875	0.75	0.7625	0.8	0.7875	0.9	0.7982	0.7368	0.7429	0.6842
Sensor 23	0.75	0.7333	0.6833	0.6	0.7786	0.5263	0.8339	0.6316	0.7554	0.7895	0.7714	0.7895	0.6643	0.7059	0.7048	0.7059
Sensor 24	0.87	0.75	0.785	0.75	0.7833	0.7857	0.8233	0.6429	0.8533	0.7857	0.8233	0.6429	0.72	0.8462	0.7	0.8462
Sensor 25	0.9267	0.9583	0.9067	1	0.9156	0.9167	0.9267	0.9167	0.9589	0.84	0.9489	0.84	0.8489	0.76	0.87	0.76
Sensor 26	0.8	0.7143	0.78	0.5714	0.6429	0.6471	0.6167	0.7059	0.6905	0.8235	0.7667	0.8235	0.7071	0.8125	0.7048	0.8125
Sensor 27	0.785	0.75	0.775	0.8333	0.6333	0.6429	0.69	0.8571	0.7767	0.6429	0.7767	0.7143	0.8	0.6154	0.74	0.6923
Sensor 28	0.9167	1	0.9333	1	0.7548	0.6471	0.6905	0.8235	0.7619	0.8889	0.7976	0.9444	0.8667	0.6875	0.85	0.75
Sensor 29	0.7964	0.7	0.7607	0.9	0.8597	0.8571	0.8708	0.7619	0.7958	0.8182	0.7514	0.7727	0.7667	0.6364	0.7417	0.6818
Sensor 30	0.8429	0.8824	0.7786	0.8824	0.85	0.8	0.85	0.85	0.8875	0.7619	0.8625	0.8571	0.8375	0.7368	0.8125	0.7368
Sensor 31	0.8233	0.6429	0.84	0.6429	0.8233	0.7333	0.7167	0.7333	0.7733	0.6	0.8067	0.8	0.7767	0.8	0.74	0.8
Sensor 33	0.805	0.5	0.825	0.5	0.7633	0.6429	0.8	0.6429	0.7567	0.6429	0.78	0.7143	0.875	0.6154	0.725	0.6154
Sensor 36	0.7	0.625	0.7333	0.625	0.7083	0.5556	0.6833	1	0.8417	0.5556	0.8	0.7778	0.7667	0.5	0.7667	0.25
Average	0.849592	0.812612	0.834158	0.803319	0.806804	0.778785	0.806227	0.801281	0.858946	0.807219	0.8617	0.834362	0.825792	0.775862	0.801938	0.785081

01.12.19-31.03.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8375	0.9524	0.825	0.9524	0.7789	0.913	0.8011	0.913	0.9208	0.8696	0.9097	0.8696	0.8278	0.8182	0.8375	0.8182
Sensor 8	0.8375	0.9048	0.8125	0.9048	0.8356	1	0.8356	1	0.8736	0.8696	0.8847	0.8696	0.8056	0.7727	0.8153	0.7727
Sensor 9	0.9125	0.9524	0.85	0.9524	0.8122	0.913	0.8233	0.913	0.9083	0.8696	0.8986	0.8696	0.8236	0.7273	0.8486	0.6818
Sensor 10	0.8278	0.7917	0.8078	0.9167	0.8756	0.875	0.8744	0.7917	0.8956	0.9167	0.9067	0.9167	0.8311	0.7083	0.82	0.75
Sensor 12	0.9	1	0.8	0.9524	0.68	0.6522	0.67	0.6957	0.7514	0.7391	0.7403	0.6087	0.7694	0.8636	0.8194	0.7273
Sensor 13	0.8278	0.875	0.8178	0.9167	0.8756	0.875	0.8644	0.7917	0.8844	0.9167	0.8644	0.8333	0.8489	0.75	0.83	0.75
Sensor 14	0.8367	0.7917	0.7856	0.9167	0.8656	0.88	0.8756	0.92	0.8956	0.875	0.8967	0.875	0.7533	0.7083	0.7844	0.7917
Sensor 15	0.85	0.9524	0.825	0.9524	0.8011	0.913	0.8344	0.7826	0.9208	0.8696	0.8986	0.7826	0.8278	0.8182	0.8375	0.8182
Sensor 16	0.9244	0.875	0.83	0.875	0.7789	0.625	0.7344	0.625	0.8967	0.88	0.8856	0.88	0.8978	0.7917	0.8633	0.7917
Sensor 17	0.8875	0.85	0.8232	0.85	0.8244	0.9565	0.8356	1	0.8986	0.913	0.8986	0.913	0.8611	0.6364	0.85	0.5909
Sensor 18	0.8567	0.913	0.8244	0.9565	0.8544	0.92	0.8644	0.92	0.8956	0.9167	0.8967	0.9167	0.8178	0.75	0.8089	0.7083
Sensor 19	0.9044	0.875	0.8078	0.9583	0.8222	0.875	0.8222	0.875	0.9167	0.8333	0.9078	0.875	0.84	0.9167	0.84	0.875
Sensor 20	0.8811	0.9167	0.8167	0.9167	0.8878	0.875	0.8644	0.875	0.83	0.8333	0.8544	0.8333	0.86	0.625	0.8489	0.7083
Sensor 21	0.85	0.9524	0.825	0.9524	0.8122	0.913	0.8122	0.8261	0.9097	0.8696	0.8986	0.8696	0.85	0.7727	0.8597	0.7727
Sensor 22	0.875	0.9524	0.8	0.9048	0.8011	0.7826	0.8122	0.7826	0.9083	0.8696	0.8986	0.8696	0.7903	0.7273	0.7917	0.8182
Sensor 23	0.8857	0.9	0.8107	0.9	0.7811	1	0.8033	1	0.8514	0.6957	0.8514	0.913	0.8722	0.7273	0.8611	0.6364
Sensor 24	0.875	0.9048	0.825	0.8571	0.8689	0.9565	0.8467	0.913	0.9097	0.8696	0.9097	0.8696	0.8486	0.6818	0.8611	0.7273
Sensor 25	0.9022	0.875	0.8167	0.875	0.8722	0.88	0.8722	0.88	0.8856	0.875	0.8967	0.8333	0.7667	0.75	0.7889	0.7083
Sensor 26	0.85	0.9524	0.8	0.9524	0.7367	0.8261	0.7256	0.6522	0.8556	0.6957	0.8097	0.8261	0.7694	0.9545	0.8194	0.7273
Sensor 27	0.8625	1	0.8	0.9524	0.77	0.8696	0.7922	0.8696	0.7958	0.6957	0.775	0.913	0.8264	0.8182	0.8347	0.7727
Sensor 28	0.85	0.9524	0.8	0.9524	0.77	0.8696	0.7922	0.8696	0.7958	0.6957	0.775	0.913	0.8264	0.8182	0.8347	0.7727
Sensor 29	0.8811	0.9167	0.8167	0.9167	0.8878	0.875	0.8644	0.875	0.83	0.8333	0.8544	0.8333	0.86	0.625	0.8489	0.7083
Sensor 30	0.9	0.9	0.8232	0.85	0.8356	1	0.8356	0.9565	0.8764	0.913	0.8986	0.913	0.8292	0.8182	0.8292	0.7727
Sensor 31	0.8278	0.8333	0.8078	0.875	0.8744	0.875	0.8622	0.7917	0.8956	0.875	0.8756	0.9167	0.8311	0.7083	0.82	0.75
Sensor 33	0.8625	0.9524	0.825	0.9524	0.8011	0.9565	0.8233	0.8696	0.8875	0.913	0.8875	0.913	0.9056	0.7727	0.8722	0.7273
Sensor 36	0.8875	0.9524	0.8625	0.9524	0.7689	0.913	0.8011	0.8261	0.8986	0.8696	0.8986	0.8696	0.8153	0.8182	0.8042	0.7727
Average	0.868969	0.911319	0.816862	0.919769	0.818165	0.884215	0.820885	0.854412	0.876465	0.845104	0.872008	0.865227	0.829054	0.764569	0.831908	0.748104

01.04.20-31.07.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8778	0.6522	0.8778	0.6522	0.8856	0.9167	0.8689	0.8333	0.9178	0.9583	0.9178	0.9583	0.8178	0.6957	0.7844	0.8261
Sensor 8	0.8667	0.913	0.8778	0.913	0.9467	0.9167	0.9378	0.875	0.8844	0.875	0.8333	0.9167	0.8122	0.6522	0.7711	0.6522
Sensor 9	0.8778	0.8261	0.8778	0.8261	0.9167	0.875	0.9278	0.875	0.8778	0.9167	0.8778	0.9583	0.8267	0.8261	0.8067	0.8261
Sensor 10	0.9233	0.9583	0.9344	0.9583	0.93	0.9583	0.92	0.9583	0.93	1	0.93	1	0.9056	0.8333	0.8844	0.875
Sensor 12	0.8444	0.7826	0.8444	0.7826	0.7967	0.8333	0.7956	0.7917	0.8678	0.8333	0.8867	0.875	0.8578	1	0.8367	0.8696
Sensor 13	0.9233	0.9583	0.9133	0.9583	0.94	0.9167	0.94	0.9167	0.93	0.9583	0.93	1	0.8	0.7083	0.78	0.9167
Sensor 14	0.9144	0.875	0.9244	0.875	0.9289	0.9167	0.9289	0.9167	0.93	1	0.93	1	0.8633	0.75	0.82	0.875
Sensor 15	0.8444	0.7826	0.8556	0.7391	0.8844	0.875	0.8856	0.875	0.9289	0.9583	0.9289	0.9583	0.8156	0.7391	0.7956	0.7826
Sensor 16	0.9256	0.913	0.9367	0.9565	0.8956	0.9167	0.9056	0.9167	0.93	1	0.93	1	0.9267	1	0.9167	1
Sensor 17	0.8556	0.913	0.8556	0.913	0.9378	0.8333	0.9378	0.875	0.8744	0.9167	0.8567	0.9583	0.7822	0.7826	0.7611	0.7391
Sensor 18	0.9244	0.875	0.9022	0.9167	0.94	0.9167	0.94	0.9167	0.93	1	0.93	1	0.8956	0.8333	0.8422	0.875
Sensor 19	0.9144	0.9167	0.9133	0.9583	0.94	0.9167	0.94	0.9167	0.94	0.9167	0.94	0.9583	0.84	0.7917	0.8433	0.8333
Sensor 20	0.8489	0.8333	0.7833	0.9167	0.8244	0.8333	0.7589	0.875	0.8111	0.875	0.7556	0.875	0.8522	0.7083	0.7111	0.7083
Sensor 21	0.8333	0.8261	0.8556	0.8261	0.8844	0.875	0.8856	0.875	0.9178	0.9583	0.9178	0.9583	0.8478	0.6087	0.8156	0.7826
Sensor 22	0.8889	0.7826	0.8889	0.7826	0.8522	0.9167	0.8644	0.875	0.9067	0.9167	0.9067	0.9167	0.8289	0.8696	0.7956	0.913
Sensor 23	0.8333	0.8696	0.8444	0.8696	0.9378	0.875	0.9378	0.875	0.8878	0.9583	0.8767	0.9583	0.8156	0.7826	0.74	0.6957
Sensor 24	0.8444	0.8696	0.8444	0.913	0.9567	0.875	0.9378	0.875	0.8944	0.875	0.8333	0.9167	0.8122	0.6522	0.7711	0.6522
Sensor 25	0.9133	0.875	0.9244	0.9583	0.95	0.875	0.95	0.875	0.93	1	0.93	1	0.8744	0.875	0.8856	0.875
Sensor 26	0.9	0.8261	0.8444	0.8696	0.8722	0.8333	0.8822	0.8333	0.9389	0.9167	0.9389	0.9583	0.8456	0.913	0.8356	0.8696
Sensor 27	0.9	0.8696	0.8667	0.7826	0.9056	0.875	0.9156	0.875	0.9267	0.9167	0.9178	0.9583	0.8467	0.913	0.8267	0.9565
Sensor 28	0.9	0.8696	0.8667	0.7826	0.9056	0.875	0.9156	0.875	0.9267	0.9167	0.9178	0.9583	0.8356	0.9565	0.8267	0.9565
Sensor 29	0.8489	0.8333	0.7833	0.9167	0.8233	0.8333	0.7589	0.875	0.8011	0.875	0.7556	0.875	0.8522	0.7083	0.7111	0.7083
Sensor 30	0.8556	0.8696	0.8556	0.913	0.9378	0.8333	0.9378	0.875	0.8744	0.9167	0.8567	0.9583	0.7922	0.7391	0.75	0.7391
Sensor 31	0.9133	0.875	0.9022	0.9167	0.94	0.9167	0.94	0.9167	0.93	0.9583	0.93	1	0.8856	0.875	0.8422	0.875
Sensor 33	0.8556	0.8261	0.8556	0.8261	0.9167	0.9167	0.9067	0.9167	0.8733	0.9583	0.8744	0.9583	0.8711	0.7391	0.8167	0.8261
Sensor 36	0.8667	0.8696	0.8667	0.8696	0.8633	0.8333	0.8756	0.875	0.9178	0.9583	0.9178	0.9583	0.8467	0.7391	0.7944	0.7826
Average	0.88055	0.856188	0.872904	0.868935	0.904323	0.883015	0.899804	0.883019	0.902992	0.935896	0.893088	0.955115	0.844242	0.795838	0.806331	0.823508

Lockdown results:

	covid 2-class															
pre-lockdown 27.01-22.03.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.925	0.9091	0.925	0.8182	0.885	0.7273	0.905	0.6364	0.79	0.9091	0.82	1	0.85	0.5455	0.825	0.7273
Sensor 8	0.88	0.7273	0.88	0.7273	0.815	0.5455	0.835	0.7273	0.89	0.8182	0.865	0.8182	0.825	0.7273	0.8	0.7273
Sensor 9	0.925	0.8182	0.925	0.8182	0.88	0.7273	0.88	0.7273	0.815	0.7273	0.82	0.9091	0.775	0.7273	0.825	0.8182
Sensor 10	0.82	0.7273	0.815	0.8182	0.79	0.8333	0.83	1	0.865	0.9091	0.915	1	0.77	1	0.795	1
Sensor 12	0.925	0.7273	0.9	0.7273	0.635	0.5455	0.68	0.7273	0.835	0.9091	0.865	0.8182	0.8	0.6364	0.825	0.7273
Sensor 13	0.81	0.8182	0.785	0.9091	0.84	1	0.83	0.9167	0.89	0.8182	0.865	1	0.795	0.9091	0.84	0.8182
Sensor 14	0.815	0.8182	0.815	0.8182	0.81	1	0.83	1	0.925	0.7273	0.91	0.9091	0.795	0.9091	0.79	0.8182
Sensor 15	0.925	0.9091	0.925	0.8182	0.885	0.7273	0.885	0.7273	0.795	0.8182	0.82	1	0.825	0.5455	0.8	0.7273
Sensor 16	0.905	0.9091	0.855	0.9091	0.795	0.5455	0.765	0.6364	0.925	0.8333	0.925	0.9167	0.885	0.6667	0.88	0.8333
Sensor 17	0.8	0.8182	0.8	0.8182	0.815	0.7273	0.765	0.7273	0.89	0.8182	0.89	0.7273	0.775	0.8182	0.75	0.8182
Sensor 18	0.805	0.8182	0.85	0.8182	0.86	0.9167	0.86	0.9167	0.865	0.9091	0.865	1	0.8	0.9091	0.795	0.9091
Sensor 19	0.78	0.8182	0.855	1	0.825	0.9167	0.8	0.9167	0.91	1	0.89	1	0.82	1	0.815	0.8182
Sensor 20	0.79	0.9091	0.78	0.9091	0.875	0.9167	0.85	0.9167	0.845	0.6364	0.795	0.6364	0.84	0.9091	0.775	0.7273
Sensor 21	0.925	0.9091	0.9	0.8182	0.91	0.6364	0.885	0.6364	0.77	1	0.82	1	0.775	0.6364	0.825	0.7273
Sensor 22	0.925	0.8182	0.925	0.8182	0.885	0.8182	0.835	0.6364	0.84	0.9091	0.89	1	0.825	0.2727	0.85	0.7273
Sensor 23	0.85	0.9091	0.85	0.9091	0.75	0.7273	0.7	0.8182	0.89	0.8182	0.865	0.8182	0.775	0.4545	0.75	0.7273
Sensor 24	0.9	0.7273	0.9	0.7273	0.875	0.6364	0.875	0.6364	0.89	0.8182	0.865	0.8182	0.825	0.9091	0.8	0.7273
Sensor 25	0.76	0.9091	0.785	0.9091	0.84	0.9167	0.86	0.9167	0.93	0.9091	0.885	0.9091	0.825	0.8182	0.845	0.8182
Sensor 26	0.9	0.7273	0.9	0.8182	0.68	0.7273	0.65	0.8182	0.88	0.9091	0.855	0.9091	0.825	0.8182	0.85	0.8182
Sensor 27	0.875	0.8182	0.88	0.7273	0.79	0.6364	0.765	0.5455	0.88	0.9091	0.885	0.9091	0.825	0.8182	0.85	0.8182
Sensor 28	0.875	0.8182	0.9	0.7273	0.81	0.6364	0.765	0.5455	0.88	0.9091	0.885	0.9091	0.825	0.8182	0.85	0.8182
Sensor 29	0.79	0.9091	0.78	0.9091	0.855	0.9167	0.85	0.9167	0.845	0.6364	0.795	0.6364	0.84	0.9091	0.775	0.7273
Sensor 30	0.8	0.9091	0.775	0.9091	0.8	0.8182	0.765	0.7273	0.89	0.8182	0.89	0.7273	0.725	0.7273	0.75	0.8182
Sensor 31	0.84	0.7273	0.815	0.8182	0.86	0.9167	0.86	0.9167	0.845	1	0.865	1	0.775	0.9091	0.795	0.9091
Sensor 33	0.925	0.6364	0.9	0.8182	0.835	0.4545	0.835	0.4545	0.89	0.9091	0.865	0.8182	0.85	0.7273	0.775	0.8182
Sensor 36	0.925	0.9091	0.925	0.8182	0.84	0.6364	0.84	0.6364	0.77	1	0.82	1	0.8	0.7273	0.825	0.7273
Average	0.861346	0.825192	0.859423	0.832185	0.824615	0.754104	0.815385	0.760808	0.863077	0.860735	0.862692	0.891912	0.809423	0.763419	0.809808	0.794308

lockdown 23.03-17.05.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.805	0.5455	0.805	0.4545	0.88	0.6364	0.855	0.5455	0.73	0.5455	0.755	0.7273	0.855	0.6364	0.83	0.6364
Sensor 8	0.86	0.5455	0.76	0.6364	0.845	0.4545	0.82	0.4545	0.725	0.8182	0.675	0.8182	0.815	0.7273	0.765	0.4545
Sensor 9	0.735	0.6364	0.68	0.8182	0.81	0.6364	0.81	0.6364	0.735	0.9091	0.75	0.8182	0.785	0.6364	0.79	0.6364
Sensor 10	0.73	0.7273	0.725	0.7273	0.77	1	0.8	0.7273	0.815	0.8333	0.77	0.8333	0.625	0.75	0.67	0.5833
Sensor 12	0.705	0.7273	0.69	0.8182	0.66	0.7273	0.69	0.7273	0.605	0.7273	0.72	0.7273	0.81	0.6364	0.83	0.6364
Sensor 13	0.705	0.6364	0.71	0.7273	0.79	0.9091	0.79	0.9091	0.77	0.9167	0.765	0.75	0.755	0.6667	0.735	1
Sensor 14	0.8	0.6364	0.795	0.6364	0.77	1	0.775	0.9167	0.785	0.8333	0.765	0.75	0.735	0.5833	0.785	0.8333
Sensor 15	0.83	0.4545	0.755	0.5455	0.88	0.6364	0.83	0.5455	0.71	0.4545	0.73	0.7273	0.855	0.6364	0.83	0.6364
Sensor 16	0.89	0.7273	0.815	0.7273	0.755	0.8333	0.745	0.9167	0.675	0.6667	0.7	0.6667	0.61	0.6667	0.72	0.6667
Sensor 17	0.66	0.7273	0.725	0.8182	0.785	0.6364	0.76	0.6364	0.665	0.9091	0.75	0.8182	0.885	0.7273	0.79	0.7273
Sensor 18	0.77	0.7273	0.785	0.6364	0.76	1	0.815	0.9167	0.815	0.6667	0.77	0.8333	0.675	0.8333	0.74	0.8333
Sensor 19	0.75	0.7273	0.76	0.7273	0.64	0.9091	0.655	0.9091	0.815	0.8333	0.765	0.8333	0.74	0.6667	0.785	0.6667
Sensor 20	0.745	0.7273	0.74	0.7273	0.775	0.7273	0.79	0.9091	0.74	0.9167	0.86	1	0.775	1	0.8	1
Sensor 21	0.755	0.7273	0.755	0.7273	0.795	0.6364	0.81	0.6364	0.735	0.3636	0.755	0.7273	0.83	0.7273	0.805	0.7273
Sensor 22	0.805	0.7273	0.775	0.6364	0.815	0.7273	0.81	0.6364	0.65	0.9091	0.75	0.8182	0.855	0.6364	0.81	0.6364
Sensor 23	0.74	0.7273	0.67	0.7273	0.74	0.7273	0.76	0.6364	0.665	0.9091	0.75	0.8182	0.79	0.6364	0.77	0.6364
Sensor 24	0.81	0.7273	0.715	0.7273	0.845	0.3636	0.82	0.4545	0.775	0.8182	0.675	0.8182	0.86	0.6364	0.79	0.8182
Sensor 25	0.795	0.6364	0.84	0.6364	0.79	0.9091	0.835	0.9091	0.79	0.75	0.79	0.75	0.745	0.75	0.76	0.8333
Sensor 26	0.74	0.4545	0.685	0.7273	0.715	0.6364	0.715	0.5455	0.635	0.7273	0.72	0.7273	0.81	0.6364	0.83	0.6364
Sensor 27	0.765	0.5455	0.785	0.7273	0.755	0.6364	0.805	0.7273	0.69	0.7273	0.745	0.5455	0.73	0.6364	0.78	0.6364
Sensor 28	0.79	0.8182	0.785	0.7273	0.755	0.6364	0.805	0.7273	0.69	0.8182	0.745	0.5455	0.755	0.6364	0.78	0.6364
Sensor 29	0.74	0.8182	0.78	0.7273	0.775	0.7273	0.765	0.7273	0.74	0.9167	0.86	1	0.775	1	0.8	1
Sensor 30	0.72	0.7273	0.705	0.7273	0.74	0.7273	0.76	0.6364	0.645	0.9091	0.75	0.8182	0.855	0.6364	0.79	0.7273
Sensor 31	0.77	0.6364	0.785	0.6364	0.775	0.9091	0.775	0.8182	0.835	0.6667	0.765	0.75	0.7	0.8333	0.74	0.8333
Sensor 33	0.78	0.5455	0.735	0.7273	0.81	0.6364	0.84	0.6364	0.77	0.7273	0.73	0.8182	0.785	0.6364	0.79	0.6364
Sensor 36	0.78	0.5455	0.78	0.6364	0.87	0.5455	0.835	0.5455	0.73	0.8182	0.725	0.7273	0.835	0.6364	0.835	0.6364
Average	0.768269	0.660865	0.751538	0.695831	0.780769	0.727873	0.787308	0.707192	0.728269	0.772738	0.751346	0.775654	0.778654	0.700196	0.782692	0.719419

post-lockdown 18.05-12.07.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.735	0.7273	0.78	0.6364	0.69	0.5455	0.69	0.6364	0.745	0.6364	0.715	0.8182	0.725	0.9091	0.55	0.7273
Sensor 8	0.8	0.7273	0.78	0.8182	0.665	0.7273	0.715	0.7273	0.81	0.7273	0.695	0.7273	0.765	0.9091	0.65	0.7273
Sensor 9	0.83	0.8182	0.83	0.7273	0.69	0.5455	0.74	0.5455	0.74	0.6364	0.695	0.6364	0.715	0.7273	0.65	0.7273
Sensor 10	0.76	0.7273	0.79	0.7273	0.765	0.5455	0.855	0.7273	0.805	0.7273	0.765	0.9091	0.765	0.6364	0.715	0.5455
Sensor 12	0.76	0.7273	0.735	0.9091	0.645	0.4545	0.685	0.5455	0.675	0.7273	0.705	0.7273	0.765	0.7273	0.65	0.7273
Sensor 13	0.805	0.7273	0.76	0.7273	0.81	0.4545	0.695	0.5455	0.785	0.7273	0.76	0.6364	0.77	0.5455	0.745	0.4545
Sensor 14	0.735	0.7273	0.78	0.6364	0.765	0.6364	0.79	0.8182	0.8	0.7273	0.715	0.8182	0.745	0.5455	0.74	0.4545
Sensor 15	0.78	0.6364	0.83	0.7273	0.725	0.5455	0.65	0.6364	0.725	0.6364	0.715	0.8182	0.815	0.6364	0.7	0.6364
Sensor 16	0.825	0.8182	0.9	0.8182	0.645	0.9091	0.585	0.5455	0.77	0.6364	0.765	0.5455	0.69	0.7273	0.735	0.6364
Sensor 17	0.78	0.7273	0.73	0.7273	0.79	0.7273	0.795	0.7273	0.76	0.7273	0.765	0.6364	0.705	0.9091	0.68	0.7273
Sensor 18	0.665	0.9091	0.76	0.8182	0.765	0.6364	0.88	0.7273	0.8	0.9091	0.72	0.9091	0.745	0.7273	0.72	0.5455
Sensor 19	0.855	0.6364	0.905	0.6364	0.755	0.7273	0.665	0.5455	0.855	0.8182	0.785	0.7273	0.745	0.7273	0.745	0.6364
Sensor 20	0.79	0.7273	0.79	0.7273	0.77	0.8182	0.74	0.7273	0.795	0.6364	0.765	0.6364	0.675	0.8182	0.705	0.6364
Sensor 21	0.76	0.6364	0.76	0.6364	0.68	0.4545	0.715	0.6364	0.715	0.7273	0.715	0.8182	0.775	0.6364	0.65	0.7273
Sensor 22	0.805	0.6364	0.75	0.8182	0.675	0.5455	0.695	0.5455	0.77	0.5455	0.74	0.7273	0.74	0.8182	0.665	0.7273
Sensor 23	0.8	0.6364	0.66	0.7273	0.785	0.6364	0.735	0.7273	0.76	0.7273	0.765	0.6364	0.655	0.9091	0.68	0.7273
Sensor 24	0.825	0.8182	0.805	0.7273	0.71	0.5455	0.715	0.8182	0.725	0.8182	0.715	0.8182	0.765	0.8182	0.65	0.7273
Sensor 25	0.76	0.8182	0.79	0.7273	0.765	0.5455	0.855	0.7273	0.85	0.7273	0.77	0.5455	0.725	0.7273	0.72	0.7273
Sensor 26	0.74	0.6364	0.685	0.7273	0.8	0.7273	0.775	0.7273	0.745	0.8182	0.775	0.7273	0.83	0.7273	0.76	0.7273
Sensor 27	0.735	0.7273	0.685	0.7273	0.74	0.6364	0.775	0.9091	0.75	0.7273	0.73	0.8182	0.86	0.6364	0.835	0.6364
Sensor 28	0.735	0.7273	0.68	0.9091	0.74	0.6364	0.775	0.9091	0.73	0.7273	0.685	0.7273	0.86	0.6364	0.835	0.6364
Sensor 29	0.79	0.7273	0.79	0.7273	0.77	0.8182	0.74	0.7273	0.795	0.6364	0.765	0.6364	0.675	0.8182	0.705	0.6364
Sensor 30	0.78	0.7273	0.73	0.7273	0.81	0.6364	0.815	0.7273	0.76	0.7273	0.765	0.6364	0.655	0.9091	0.68	0.7273
Sensor 31	0.655	0.7273	0.76	0.8182	0.765	0.5455	0.855	0.6364	0.8	0.7273	0.74	0.8182	0.695	0.6364	0.695	0.5455
Sensor 33	0.825	0.6364	0.875	0.8182	0.74	0.6364	0.785	0.7273	0.75	0.7273	0.765	0.5455	0.755	0.8182	0.705	0.7273
Sensor 36	0.805	0.7273	0.805	0.7273	0.76	0.5455	0.74	0.6364	0.72	0.7273	0.715	0.8182	0.72	0.6364	0.67	0.7273
Average	0.774423	0.723804	0.774808	0.748277	0.739231	0.622404	0.748462	0.688842	0.766731	0.716812	0.738846	0.723804	0.743654	0.741285	0.701346	0.660865

pre_until_post lockdown 27.01.2020-12.07.2020	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8628	0.9688	0.8224	0.9375	0.8769	0.9394	0.8615	0.9091	0.9236	0.9091	0.9236	0.9091	0.8333	0.875	0.7846	0.8438
Sensor 8	0.8942	0.9062	0.9032	0.8438	0.8994	0.8485	0.8994	0.8485	0.8934	0.8485	0.878	0.8788	0.8173	0.8438	0.7769	0.8125
Sensor 9	0.8808	0.9688	0.8647	0.9375	0.884	0.8788	0.884	0.9091	0.8934	0.9394	0.8857	0.9394	0.8006	0.8438	0.7929	0.9062
Sensor 10	0.9532	0.8485	0.9532	0.8485	0.939	0.8235	0.9236	0.8235	0.9011	0.9394	0.9016	0.9091	0.8698	0.8485	0.8698	0.8182
Sensor 12	0.8801	0.875	0.8801	0.875	0.8442	0.697	0.8058	0.8485	0.9005	0.9091	0.8929	0.9091	0.866	0.9062	0.866	0.9062
Sensor 13	0.9385	0.875	0.9385	0.875	0.9242	0.8824	0.9159	0.8529	0.894	0.9394	0.8934	0.9394	0.8401	0.8485	0.8626	0.7879
Sensor 14	0.9378	0.8485	0.9378	0.8485	0.9319	0.8235	0.9319	0.8235	0.9011	0.9091	0.9016	0.9394	0.8621	0.7879	0.8467	0.8485
Sensor 15	0.8558	0.9688	0.8301	0.9375	0.8686	0.9394	0.9071	0.9394	0.939	0.9091	0.939	0.9091	0.8494	0.8125	0.7846	0.8438
Sensor 16	0.9462	0.7812	0.9462	0.7812	0.8698	0.9091	0.8857	0.8182	0.9099	0.8529	0.8945	0.8529	0.9236	0.8485	0.9236	0.8485
Sensor 17	0.9109	0.8438	0.9032	0.8125	0.866	0.9697	0.8583	0.9697	0.9088	0.9091	0.9088	0.9091	0.8083	0.8438	0.8006	0.8438
Sensor 18	0.9276	0.9062	0.9276	0.9062	0.9319	0.8235	0.9319	0.8235	0.9088	0.8788	0.9016	0.9394	0.8698	0.8485	0.8544	0.8182
Sensor 19	0.9378	0.9375	0.9378	0.9375	0.9236	0.8235	0.9308	0.8235	0.9093	0.9091	0.8934	0.9394	0.8626	0.9091	0.8396	0.9091
Sensor 20	0.8577	0.875	0.8827	0.75	0.8258	0.8824	0.8181	0.9118	0.8258	0.8485	0.8258	0.8182	0.7637	0.6667	0.7637	0.6364
Sensor 21	0.9032	0.9062	0.8712	0.9062	0.9	0.8788	0.8923	0.8485	0.9313	0.9394	0.9313	0.9394	0.834	0.8438	0.7929	0.8438
Sensor 22	0.8712	0.9375	0.8385	0.9375	0.9	0.8485	0.8846	0.8788	0.8934	0.9394	0.8857	0.9394	0.841	0.7188	0.7853	0.8438
Sensor 23	0.9103	0.75	0.9032	0.7812	0.9071	0.8182	0.8994	0.8485	0.9088	0.9091	0.9088	0.9091	0.816	0.75	0.8244	0.8438
Sensor 24	0.8628	0.9062	0.8872	0.8438	0.8917	0.8485	0.8994	0.8485	0.8934	0.8788	0.878	0.8788	0.825	0.8125	0.8013	0.8125
Sensor 25	0.9224	0.875	0.9224	0.875	0.939	0.7941	0.9313	0.7941	0.894	0.9697	0.9088	0.9091	0.8473	0.8182	0.839	0.8788
Sensor 26	0.8571	0.8438	0.8564	0.8438	0.8288	0.8182	0.7904	0.9091	0.8929	0.9394	0.8929	0.9091	0.8583	0.9375	0.8423	0.9375
Sensor 27	0.8641	0.875	0.8558	0.7812	0.8442	0.7576	0.8141	0.9091	0.8929	0.9091	0.8929	0.9091	0.8737	0.9688	0.8423	0.9375
Sensor 28	0.8564	0.875	0.8564	0.8438	0.8442	0.7576	0.8141	0.9091	0.8929	0.9091	0.8929	0.9091	0.8814	0.9375	0.8423	0.9375
Sensor 29	0.8577	0.8438	0.8827	0.75	0.8258	0.8824	0.8181	0.9118	0.833	0.8485	0.8258	0.8182	0.7637	0.6667	0.7637	0.6364
Sensor 30	0.9109	0.8438	0.9032	0.8125	0.8583	0.9697	0.8583	0.9697	0.9088	0.9091	0.9088	0.9091	0.809	0.875	0.7923	0.8438
Sensor 31	0.9449	0.8485	0.9372	0.8485	0.9236	0.8529	0.9236	0.8529	0.9165	0.9394	0.9016	0.9091	0.8621	0.8182	0.8544	0.8182
Sensor 33	0.8878	0.9375	0.8641	0.9062	0.9224	0.8788	0.8994	0.8788	0.8934	0.9697	0.8934	0.9697	0.825	0.9062	0.841	0.9062
Sensor 36	0.9032	0.9062	0.8712	0.9062	0.8846	0.9091	0.8846	0.8788	0.9159	0.9394	0.9159	0.9394	0.834	0.8438	0.7846	0.8438
Average	0.897515	0.882762	0.891423	0.858715	0.886731	0.855965	0.879369	0.874573	0.899073	0.9116	0.895258	0.909269	0.839888	0.837685	0.821992	0.842565

Grouped results:

Grouped results 2-class																	
		07--10				12--15				15--18				19--22			
		RF		LR		RF		LR		RF		LR		RF		LR	
	Period	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
whole	01.08.19-31.08.20	0.876	0.873	0.859	0.857	0.854	0.847	0.845	0.845	0.882	0.854	0.872	0.850	0.822	0.804	0.811	0.802
4-month	01.08.19-30.11.19	0.850	0.813	0.834	0.803	0.807	0.779	0.806	0.801	0.859	0.807	0.862	0.834	0.826	0.776	0.802	0.785
	01.12.19-31.03.20	0.869	0.911	0.817	0.920	0.818	0.884	0.821	0.854	0.876	0.845	0.872	0.865	0.829	0.765	0.832	0.748
	01.04.20-31.07.20	0.881	0.856	0.873	0.869	0.904	0.883	0.900	0.883	0.903	0.936	0.893	0.955	0.844	0.796	0.806	0.824
3-month	01.08.19-31.10.19	0.846	0.800	0.845	0.782	0.825	0.752	0.823	0.778	0.877	0.800	0.862	0.807	0.806	0.756	0.786	0.740
	01.11.19-31.01.20	0.851	0.781	0.835	0.819	0.774	0.727	0.731	0.768	0.840	0.748	0.811	0.832	0.794	0.703	0.780	0.672
	01.02.20-30.04.20	0.813	0.897	0.808	0.881	0.809	0.820	0.821	0.812	0.836	0.776	0.821	0.797	0.829	0.790	0.812	0.770
	01.05.20-31.07.20	0.881	0.853	0.874	0.850	0.869	0.784	0.862	0.793	0.884	0.825	0.897	0.823	0.839	0.778	0.823	0.761
lockdown	pre_lockdown 27.01-22.03.20	0.861	0.825	0.859	0.832	0.825	0.754	0.815	0.761	0.863	0.861	0.863	0.892	0.809	0.763	0.810	0.794
	lockdown 23.03-17.05.20	0.768	0.661	0.752	0.696	0.781	0.728	0.787	0.707	0.728	0.773	0.751	0.776	0.779	0.700	0.783	0.719
	post-lockdown 18.05-12.07.20	0.774	0.724	0.775	0.748	0.739	0.622	0.748	0.689	0.767	0.717	0.739	0.724	0.744	0.741	0.701	0.661
	pre_until_post lockdown 27.01.2020-12.07.2020	0.898	0.883	0.891	0.859	0.887	0.856	0.879	0.875	0.899	0.912	0.895	0.909	0.840	0.838	0.822	0.843

Appendix F: Results 3-classes

Model evolution results:

whole dataset 3-class																
01.08.19-31.08.20	07--10				12--15				15--18				19--22			
Sensors	LR SCORE															
	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL	OLD	STATUS	WEEKENDS	FINAL
Sensor 7	0.5406	0.568	0.8263	0.8302	0.5119	0.53	0.8021	0.7881	0.5661	0.577	0.8352	0.8495	0.4791	0.5014	0.6724	0.6803
Sensor 8	0.5102	0.5222	0.8012	0.8132	0.5092	0.5335	0.7315	0.7456	0.5589	0.5771	0.8242	0.828	0.4608	0.49	0.7358	0.7209
Sensor 9	0.5252	0.5413	0.814	0.8305	0.5061	0.5689	0.7846	0.8034	0.5664	0.5823	0.8338	0.857	0.504	0.5317	0.6254	0.656
Sensor 10	0.5971	0.6177	0.8286	0.839	0.5565	0.6252	0.7762	0.796	0.5516	0.5748	0.7541	0.7408	0.5282	0.5649	0.6544	0.7043
Sensor 12	0.51	0.5505	0.7457	0.7702	0.4767	0.5347	0.5738	0.5923	0.4688	0.5462	0.6825	0.6899	0.4712	0.4902	0.6439	0.667
Sensor 13	0.5771	0.5905	0.839	0.8525	0.5463	0.5994	0.7928	0.7895	0.5448	0.5882	0.7806	0.7808	0.4949	0.5848	0.7176	0.7408
Sensor 14	0.5869	0.6108	0.8318	0.8353	0.5705	0.6267	0.7738	0.7938	0.5614	0.5782	0.774	0.7839	0.4982	0.5315	0.6808	0.694
Sensor 15	0.5117	0.5346	0.8433	0.8395	0.4734	0.493	0.7637	0.7734	0.526	0.498	0.8054	0.802	0.4712	0.5137	0.6872	0.6589
Sensor 16	0.5579	0.5787	0.8268	0.8539	0.5725	0.6323	0.6726	0.7025	0.5968	0.6068	0.7667	0.7834	0.4997	0.559	0.7133	0.7527
Sensor 17	0.5437	0.5359	0.8479	0.8558	0.4964	0.5301	0.7568	0.7667	0.5324	0.5463	0.7883	0.8017	0.5198	0.5336	0.6472	0.6469
Sensor 18	0.5643	0.5784	0.8275	0.8379	0.6032	0.623	0.7632	0.7931	0.5233	0.5467	0.7433	0.7567	0.5152	0.5451	0.6413	0.6973
Sensor 19	0.5838	0.5772	0.8523	0.8523	0.5134	0.5827	0.7301	0.7366	0.5014	0.5548	0.7606	0.7573	0.4814	0.5014	0.7672	0.7605
Sensor 20	0.4695	0.4898	0.6441	0.688	0.4108	0.4934	0.5655	0.6213	0.4053	0.4651	0.5581	0.6144	0.4056	0.4488	0.5348	0.558
Sensor 21	0.5712	0.5674	0.8333	0.8296	0.4849	0.5092	0.7792	0.7957	0.548	0.5516	0.8151	0.8356	0.4558	0.5195	0.6956	0.7028
Sensor 22	0.5529	0.5723	0.8303	0.834	0.5151	0.5334	0.7809	0.7703	0.5519	0.5807	0.8316	0.8495	0.5279	0.5315	0.6692	0.6655
Sensor 23	0.5593	0.5873	0.8085	0.8445	0.4894	0.5948	0.7648	0.7831	0.561	0.5791	0.8089	0.8238	0.5744	0.5779	0.6266	0.6342
Sensor 24	0.5562	0.5647	0.8509	0.8681	0.5392	0.5663	0.7598	0.7642	0.5615	0.5845	0.8005	0.8162	0.4982	0.5258	0.7225	0.7142
Sensor 25	0.5801	0.6003	0.8117	0.8154	0.5657	0.6086	0.7697	0.7926	0.5317	0.5881	0.7208	0.7406	0.5347	0.5552	0.6377	0.6875
Sensor 26	0.5123	0.5242	0.77	0.7862	0.4741	0.5037	0.5667	0.5852	0.4387	0.4952	0.611	0.6295	0.4485	0.4945	0.6498	0.6657
Sensor 27	0.4833	0.5219	0.7502	0.7668	0.4548	0.4935	0.5366	0.5989	0.3805	0.4383	0.6406	0.6522	0.3935	0.4062	0.639	0.6513
Sensor 28	0.5178	0.5265	0.7577	0.7697	0.4829	0.5237	0.55	0.6126	0.4477	0.4736	0.6614	0.6761	0.426	0.4729	0.6394	0.6665
Sensor 29	0.4622	0.5015	0.701	0.7223	0.459	0.4867	0.579	0.6162	0.4083	0.4775	0.6058	0.6506	0.4361	0.4876	0.5118	0.5704
Sensor 30	0.5258	0.5654	0.8248	0.8328	0.5214	0.5532	0.7409	0.7445	0.5714	0.5786	0.8143	0.8321	0.5222	0.5481	0.6185	0.6259
Sensor 31	0.5851	0.5851	0.8526	0.8722	0.5702	0.6345	0.762	0.8195	0.5632	0.5865	0.7207	0.7356	0.5403	0.5748	0.6516	0.7125
Sensor 33	0.5699	0.5868	0.837	0.8623	0.5217	0.5485	0.8251	0.8328	0.55	0.5335	0.8618	0.8697	0.4557	0.5288	0.7295	0.7297
Sensor 36	0.547	0.5556	0.8104	0.8292	0.5063	0.5406	0.8111	0.8067	0.5263	0.5389	0.7911	0.8168	0.5032	0.5123	0.7389	0.7342
Average	0.54235	0.559792308	0.80641923	0.820438	0.512754	0.556523077	0.719711538	0.739408	0.5209	0.547984615	0.753476923	0.768219	0.486376923	0.520430769	0.663515385	0.680692

Whole dataset (01.08.2019-31.08.20) results:

whole dataset 3-class																
01.08.19-31.08.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8342	0.7778	0.8302	0.7937	0.7915	0.7887	0.7881	0.7746	0.8495	0.7571	0.8495	0.7429	0.7024	0.6324	0.6803	0.6176
Sensor 8	0.8171	0.8889	0.8132	0.873	0.7525	0.7465	0.7456	0.7465	0.8388	0.7714	0.828	0.7857	0.7246	0.6324	0.7209	0.6029
Sensor 9	0.8347	0.918	0.8305	0.918	0.8071	0.7647	0.8034	0.7794	0.8531	0.7463	0.857	0.7612	0.6718	0.7188	0.656	0.6875
Sensor 10	0.8422	0.9067	0.839	0.84	0.7961	0.8026	0.796	0.7895	0.7505	0.75	0.7408	0.7237	0.7138	0.7237	0.7043	0.7105
Sensor 12	0.7778	0.7903	0.7702	0.8387	0.5927	0.5362	0.5923	0.5797	0.7011	0.5882	0.6899	0.6029	0.6746	0.5909	0.667	0.5909
Sensor 13	0.8525	0.9067	0.8525	0.88	0.793	0.8289	0.7895	0.8026	0.7939	0.8158	0.7808	0.75	0.7475	0.7237	0.7408	0.6842
Sensor 14	0.8453	0.9067	0.8353	0.88	0.81	0.8312	0.7938	0.8182	0.7705	0.7895	0.7839	0.7632	0.6808	0.6974	0.694	0.6711
Sensor 15	0.8434	0.8333	0.8395	0.8182	0.7734	0.8267	0.7734	0.8133	0.7986	0.8649	0.802	0.8784	0.666	0.7606	0.6589	0.6901
Sensor 16	0.827	0.7838	0.8539	0.7973	0.6658	0.7368	0.7025	0.7237	0.7929	0.7403	0.7834	0.7792	0.7592	0.7368	0.7527	0.6711
Sensor 17	0.8672	0.8636	0.8558	0.803	0.7667	0.7432	0.7667	0.7297	0.8054	0.8919	0.8017	0.9054	0.6292	0.6197	0.6469	0.6338
Sensor 18	0.838	0.8933	0.8379	0.88	0.8031	0.8182	0.7931	0.8182	0.7533	0.75	0.7567	0.7763	0.6973	0.7237	0.6973	0.6974
Sensor 19	0.859	0.8667	0.8523	0.8667	0.7437	0.7237	0.7366	0.6974	0.7473	0.7763	0.7573	0.7763	0.7739	0.7632	0.7605	0.7237
Sensor 20	0.6945	0.8	0.688	0.6933	0.6713	0.6447	0.6213	0.5921	0.6513	0.6184	0.6144	0.6447	0.608	0.5526	0.558	0.4868
Sensor 21	0.8484	0.8788	0.8296	0.8485	0.8094	0.7973	0.7957	0.7568	0.8254	0.9054	0.8356	0.8514	0.7101	0.7606	0.7028	0.7183
Sensor 22	0.8418	0.8413	0.834	0.8413	0.7775	0.7324	0.7703	0.7042	0.8388	0.7286	0.8495	0.7286	0.6876	0.6471	0.6655	0.6324
Sensor 23	0.8445	0.8548	0.8445	0.8548	0.8012	0.6377	0.7831	0.6522	0.8386	0.7971	0.8238	0.7681	0.6348	0.6061	0.6342	0.6364
Sensor 24	0.8639	0.7797	0.8681	0.7797	0.7951	0.6923	0.7642	0.7692	0.82	0.8594	0.8162	0.875	0.7145	0.6613	0.7142	0.6613
Sensor 25	0.8154	0.8933	0.8154	0.84	0.7958	0.7532	0.7926	0.7143	0.7439	0.7763	0.7406	0.7632	0.6977	0.7237	0.6875	0.7105
Sensor 26	0.7823	0.8033	0.7862	0.8525	0.6074	0.6471	0.5852	0.6324	0.645	0.6119	0.6295	0.597	0.6889	0.6	0.6657	0.6
Sensor 27	0.7754	0.7966	0.7668	0.7966	0.6143	0.4923	0.5989	0.5385	0.7075	0.5781	0.6522	0.5938	0.6682	0.6393	0.6513	0.623
Sensor 28	0.786	0.8226	0.7697	0.8226	0.6053	0.5441	0.6126	0.5735	0.695	0.6567	0.6761	0.6119	0.6783	0.6308	0.6665	0.6462
Sensor 29	0.7724	0.6901	0.7223	0.7183	0.6232	0.6164	0.6162	0.6438	0.6681	0.6575	0.6506	0.6849	0.6228	0.6301	0.5704	0.589
Sensor 30	0.8406	0.9683	0.8328	0.9365	0.7483	0.8451	0.7445	0.8028	0.8321	0.8429	0.8321	0.8143	0.6259	0.7353	0.6259	0.6471
Sensor 31	0.8682	0.7846	0.8722	0.7846	0.8231	0.7463	0.8195	0.7164	0.7316	0.7727	0.7356	0.8182	0.7239	0.6818	0.7125	0.6364
Sensor 33	0.8623	0.8305	0.8623	0.8305	0.8328	0.8154	0.8328	0.8154	0.8697	0.8125	0.8697	0.8125	0.7338	0.7049	0.7297	0.7049
Sensor 36	0.8383	0.8182	0.8292	0.7818	0.8109	0.85	0.8067	0.85	0.7955	0.8814	0.8168	0.8814	0.7346	0.6842	0.7342	0.6667
Average	0.825862	0.842227	0.820438	0.8296	0.746585	0.729296	0.739408	0.7244	0.773746	0.759254	0.768219	0.757315	0.691162	0.676196	0.680692	0.651531

3-month results:

3-month 3-class																
01.08.19-31.10.19	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.735	0.5	0.715	0.1667	0.69	0.4667	0.69	0.7333	0.7933	0.7333	0.7933	0.6667	0.54	0.7143	0.43	0.6429
Sensor 8	0.77	0.5455	0.795	0.5455	0.76	0.6	0.7433	0.6	0.71	0.6	0.7433	0.7333	0.7933	0.7143	0.7533	0.6429
Sensor 9	0.5417	0.5556	0.4167	0.3333	0.71	0.8333	0.69	0.6667	0.77	0.75	0.67	0.8333	0.535	0.7273	0.38	0.6364
Sensor 10	0.9036	0.9474	0.8893	0.9474	0.7446	0.6667	0.6893	0.7222	0.6536	0.6316	0.7232	0.7368	0.6446	0.5263	0.7518	0.4737
Sensor 12	0.55	0.6	0.55	0.4	0.42	0.5385	0.41	0.4615	0.55	0.4615	0.5433	0.4615	0.625	0.5833	0.685	0.5833
Sensor 13	0.9018	0.9474	0.9018	0.9474	0.7321	0.7222	0.7214	0.6667	0.7661	0.7895	0.7804	0.7895	0.7071	0.4737	0.6786	0.3684
Sensor 14	0.8911	0.9474	0.9036	0.9474	0.7036	0.7368	0.6768	0.7368	0.6804	0.7895	0.6946	0.7368	0.7107	0.6842	0.7071	0.4211
Sensor 15	0.7267	0.7857	0.71	0.7143	0.7196	0.6316	0.7179	0.6316	0.7393	0.7895	0.7679	0.7895	0.6405	0.6667	0.6548	0.5556
Sensor 16	0.8714	0.8889	0.9	0.9444	0.5929	0.4737	0.6071	0.5789	0.7946	0.7368	0.7518	0.6316	0.7821	0.7895	0.7268	0.4737
Sensor 17	0.7567	0.8	0.79	0.9333	0.7375	0.5789	0.7375	0.5789	0.7536	0.7895	0.7679	0.7895	0.6405	0.4444	0.6095	0.7222
Sensor 18	0.9036	0.9474	0.9036	0.9474	0.7179	0.7895	0.7196	0.7895	0.6625	0.5789	0.6911	0.7368	0.6839	0.4737	0.75	0.3684
Sensor 19	0.9	0.8947	0.8714	0.8947	0.6946	0.6667	0.65	0.7222	0.7679	0.7895	0.7643	0.7368	0.7964	0.5789	0.8071	0.6842
Sensor 20	0.8732	0.7368	0.8464	0.8421	0.6786	0.4444	0.6107	0.7222	0.7375	0.6842	0.7661	0.7368	0.5589	0.6842	0.5071	0.6316
Sensor 21	0.7633	0.7857	0.73	0.7857	0.6946	0.8889	0.6929	0.8333	0.775	0.7368	0.8036	0.7895	0.5905	0.6667	0.5333	0.6111
Sensor 22	0.685	0.3333	0.75	0.1667	0.7633	0.4667	0.7067	0.6667	0.7767	0.6667	0.76	0.6667	0.4633	0.5	0.4267	0.7143
Sensor 23	0.575	0.6	0.525	0.9	0.75	0.5	0.77	0.5714	0.7567	0.5714	0.7533	0.6429	0.61	0.5385	0.51	0.5385
Sensor 24	0.55	0.1429	0.6333	0.2857	0.7833	0.5556	0.7	0.5556	0.625	0.5556	0.6917	0.5556	0.575	0.5	0.5333	0.75
Sensor 25	0.875	1	0.8893	0.9474	0.7446	0.6667	0.6893	0.6111	0.6679	0.5789	0.6679	0.6842	0.6411	0.4737	0.7339	0.3158
Sensor 26	0.6417	0.7	0.6583	0.7	0.49	0.4167	0.415	0.0833	0.585	0.5	0.525	0.5	0.59	0.3636	0.63	0.6364
Sensor 27	0.5667	0.8571	0.5333	0	0.5417	0.2222	0.4917	0.3333	0.6667	0.2222	0.6583	0.2222	0.6167	0.25	0.55	0.375
Sensor 28	0.9	0.6	0.85	0.6	0.515	0.5	0.575	0.6667	0.67	0.5833	0.755	0.5	0.705	0.3636	0.655	0.4545
Sensor 29	0.77	0.7143	0.76	0.7857	0.8167	0.8	0.9	0.8	0.7881	0.875	0.7881	0.75	0.7048	0.5625	0.5905	0.4375
Sensor 30	0.735	0.75	0.69	0.9167	0.7133	0.6667	0.7267	0.7333	0.7	0.8	0.7	0.9333	0.6633	0.5	0.6167	0.5
Sensor 31	0.6083	0.5556	0.6333	0.5556	0.7333	0.6667	0.7667	0.6667	0.7333	0.5556	0.7583	0.6667	0.6	0.5556	0.6083	0.6667
Sensor 33	0.6	0.5556	0.6083	0.6667	0.85	0.4444	0.7667	0.6667	0.775	0.5556	0.7667	0.5556	0.6667	0.25	0.6333	0.25
Sensor 36	0.6667	0.25	0.6333	0.25	0.775	0.5556	0.7667	0.5556	0.85	0.4444	0.7667	0.6667	0.6083	0.5556	0.6333	0.5556
Average	0.740827	0.69005	0.734112	0.658619	0.695085	0.596123	0.678115	0.629008	0.721085	0.644973	0.725069	0.67355	0.6420269	0.543869	0.619054	0.538838

01.11.19-31.01.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.68	0.7143	0.6233	0.6429	0.6643	0.5294	0.6048	0.4706	0.6929	0.8125	0.6738	0.875	0.7167	0.6875	0.7167	0.5625
Sensor 8	0.73	0.5	0.71	0.4286	0.631	0.4706	0.531	0.5294	0.7429	0.875	0.7429	0.8125	0.6	0.6875	0.6667	0.625
Sensor 9	0.7467	0.5714	0.7	0.3571	0.6071	0.6471	0.6381	0.7647	0.6976	0.6875	0.7286	0.6875	0.7	0.875	0.7	0.6875
Sensor 10	0.7	0.5	0.7143	0.7222	0.6357	0.7778	0.5964	0.7778	0.6714	0.4444	0.6714	0.3889	0.6857	0.5	0.6286	0.5
Sensor 12	0.7733	0.5714	0.8033	0.4286	0.5405	0.4706	0.5405	0.3529	0.5167	0.3125	0.5476	0.5625	0.6167	0.5625	0.5333	0.5625
Sensor 13	0.7464	0.5	0.7054	0.5556	0.6518	0.8889	0.6518	0.8333	0.6857	0.6667	0.6571	0.6111	0.6143	0.2222	0.6	0.5556
Sensor 14	0.7286	0.5556	0.7143	0.6111	0.6786	0.9444	0.6536	0.8889	0.6857	0.6111	0.6571	0.7778	0.5857	0.5556	0.5429	0.4444
Sensor 15	0.7333	0.5	0.7	0.5	0.6643	0.4706	0.5833	0.5294	0.6452	0.75	0.6619	0.75	0.7	0.625	0.7	0.625
Sensor 16	0.6976	0.6667	0.681	0.6667	0.5429	0.3889	0.5	0.2778	0.6304	0.6842	0.6286	0.6316	0.7643	0.6111	0.7375	0.5556
Sensor 17	0.67	0.6429	0.6667	0.6429	0.6786	0.5882	0.6524	0.5294	0.7119	0.75	0.7286	0.75	0.5667	0.6	0.55	0.7333
Sensor 18	0.7286	0.6667	0.7429	0.6111	0.6946	0.8889	0.625	0.8333	0.6714	0.7222	0.6571	0.6111	0.6429	0.4444	0.5714	0.2778
Sensor 19	0.6571	0.5556	0.5643	0.5556	0.6554	0.7222	0.625	0.7778	0.6857	0.5556	0.7286	0.7222	0.7714	0.3889	0.7286	0.4444
Sensor 20	0.7589	0.6667	0.7179	0.6111	0.6536	0.6667	0.5821	0.6667	0.6571	0.5556	0.5286	0.3889	0.6429	0.4444	0.6857	0.5
Sensor 21	0.7367	0.5	0.7867	0.4286	0.6643	0.4706	0.6238	0.5882	0.7095	0.75	0.7119	0.75	0.75	0.625	0.7667	0.5625
Sensor 22	0.7467	0.5	0.7367	0.5714	0.7024	0.5294	0.5786	0.6471	0.7	0.75	0.6476	0.8125	0.65	0.75	0.6667	0.5
Sensor 23	0.72	0.6429	0.64	0.5714	0.6476	0.8235	0.7167	0.7647	0.6	0.5625	0.631	0.6875	0.6	0.6667	0.6	0.6667
Sensor 24	0.73	0.5714	0.6967	0.2143	0.6167	0.5294	0.531	0.5294	0.7286	0.75	0.8071	0.75	0.7167	0.625	0.6833	0.625
Sensor 25	0.775	0.6667	0.7339	0.7222	0.7393	0.8421	0.6554	0.8947	0.6286	0.4444	0.6286	0.5	0.5429	0.3889	0.4714	0.5
Sensor 26	0.7733	0.4286	0.7467	0.2857	0.5738	0.4118	0.4976	0.4706	0.5095	0.5625	0.4833	0.6875	0.7333	0.5	0.5167	0.5
Sensor 27	0.7	0.5714	0.6933	0.3571	0.5	0.5294	0.5238	0.5294	0.4667	0.375	0.4667	0.8125	0.5833	0.875	0.5167	0.625
Sensor 28	0.7	0.5714	0.7133	0.2857	0.5238	0.3529	0.5405	0.5882	0.4667	0.375	0.4667	0.8125	0.5833	0.875	0.5167	0.625
Sensor 29	0.7589	0.6667	0.7179	0.6111	0.6536	0.6667	0.5821	0.6667	0.6571	0.5556	0.5286	0.3889	0.6429	0.4444	0.6857	0.5
Sensor 30	0.67	0.6429	0.6333	0.6429	0.681	0.5294	0.6524	0.5882	0.7119	0.75	0.7286	0.75	0.5833	0.625	0.5333	0.6875
Sensor 31	0.7143	0.6667	0.7429	0.6111	0.6643	0.8889	0.625	0.7778	0.6857	0.6667	0.6714	0.6111	0.6857	0.2778	0.6143	0.2778
Sensor 33	0.6433	0.5	0.5533	0.6429	0.669	0.5882	0.6524	0.7647	0.7143	0.8125	0.7262	0.75	0.7167	0.75	0.7167	0.8125
Sensor 36	0.7033	0.5	0.7333	0.4286	0.6786	0.5882	0.5	0.7059	0.7095	0.8125	0.7262	0.8125	0.7333	0.6875	0.7167	0.6875
Average	0.720077	0.578462	0.6989	0.527173	0.638954	0.623262	0.594742	0.644138	0.653181	0.638231	0.647531	0.680542	0.6587962	0.588246	0.629473	0.563196

01.02.20-30.04.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8476	0.9412	0.831	0.9412	0.8833	0.6667	0.9119	0.6667	0.8732	0.7778	0.8857	0.8333	0.7929	0.7647	0.7929	0.7647
Sensor 8	0.8333	0.9412	0.8643	0.9412	0.8119	0.8333	0.8548	0.8333	0.8768	0.8333	0.8911	0.8333	0.7119	0.7059	0.7429	0.7059
Sensor 9	0.9119	0.8824	0.8976	0.8235	0.8381	0.7222	0.8667	0.7222	0.8446	0.7778	0.8446	0.7778	0.8048	0.7647	0.7952	0.7059
Sensor 10	0.8714	0.8889	0.8571	1	0.75	0.6667	0.7482	0.7222	0.8018	0.7778	0.8304	0.8889	0.7607	0.7222	0.7464	0.7222
Sensor 12	0.8214	0.8824	0.7429	0.7059	0.6476	0.5	0.7095	0.5556	0.7054	0.6111	0.7196	0.6667	0.7548	0.7647	0.6929	0.7647
Sensor 13	0.8548	1	0.8976	1	0.8607	0.7222	0.8893	0.7778	0.8857	0.7222	0.8179	0.8333	0.7893	0.7222	0.7768	0.6667
Sensor 14	0.8857	1	0.8857	1	0.8589	0.6111	0.8179	0.6667	0.8179	0.8333	0.8179	0.8889	0.6893	0.6111	0.6893	0.6667
Sensor 15	0.8762	0.9412	0.8286	0.9412	0.8833	0.6667	0.8976	0.7222	0.8732	0.7222	0.8857	0.7778	0.6929	0.7647	0.7071	0.7647
Sensor 16	0.8833	0.8889	0.9119	0.8889	0.7286	0.6111	0.7429	0.7222	0.7661	0.6667	0.6821	0.7222	0.8179	0.5556	0.8054	0.8333
Sensor 17	0.8952	0.8824	0.8476	0.9412	0.8095	0.8824	0.8238	0.8824	0.8732	0.7778	0.8607	0.6667	0.7476	0.6471	0.7476	0.6471
Sensor 18	0.9119	0.8235	0.9405	0.8824	0.8607	0.6111	0.8321	0.6667	0.775	0.7778	0.8161	0.8333	0.775	0.6667	0.7321	0.6667
Sensor 19	0.8833	0.8889	0.869	0.8889	0.7036	0.5	0.7339	0.7222	0.7607	0.6667	0.7732	0.8333	0.8732	0.6667	0.8732	0.7778
Sensor 20	0.7952	0.7222	0.65	0.8333	0.7036	0.6111	0.6321	0.6111	0.7018	0.5	0.7179	0.5556	0.4893	0.4444	0.4643	0.6111
Sensor 21	0.8952	0.8235	0.8786	0.8824	0.8833	0.6667	0.9119	0.7222	0.8857	0.7778	0.8857	0.7778	0.7333	0.8235	0.6929	0.7059
Sensor 22	0.8952	0.8235	0.8452	0.8824	0.8548	0.7778	0.8548	0.7778	0.8732	0.7222	0.8446	0.7222	0.7357	0.7647	0.7071	0.7647
Sensor 23	0.8667	0.9412	0.8619	0.9412	0.781	0.7647	0.8238	0.8824	0.8732	0.7778	0.8607	0.7222	0.7333	0.6471	0.7476	0.6471
Sensor 24	0.85	0.8824	0.8667	0.8824	0.8405	0.8333	0.8548	0.8333	0.875	0.8889	0.8911	0.8333	0.7786	0.7059	0.75	0.7059
Sensor 25	0.8548	1	0.8976	1	0.8357	0.7222	0.8339	0.7778	0.7875	0.6111	0.7875	0.8889	0.7196	0.7778	0.7036	0.7222
Sensor 26	0.8857	0.9412	0.8524	1	0.7214	0.5	0.7381	0.6111	0.7625	0.6111	0.7911	0.6667	0.781	0.7647	0.7643	0.7647
Sensor 27	0.8857	0.9412	0.8524	1	0.7333	0.5556	0.7238	0.6111	0.8018	0.6667	0.7893	0.7222	0.8238	0.8235	0.8095	0.7647
Sensor 28	0.8857	0.9412	0.8524	1	0.7333	0.5556	0.7381	0.6111	0.8161	0.6111	0.7893	0.7222	0.8238	0.8235	0.8095	0.7647
Sensor 29	0.7952	0.6667	0.65	0.8333	0.6893	0.5556	0.6321	0.6111	0.7018	0.5	0.7179	0.5556	0.45	0.5556	0.4643	0.6111
Sensor 30	0.9095	0.8824	0.8476	0.9412	0.8238	0.8824	0.8238	0.8824	0.8732	0.7778	0.8607	0.6667	0.7476	0.6471	0.7476	0.6471
Sensor 31	0.8571	1	0.8857	1	0.75	0.6667	0.7321	0.7778	0.775	0.7222	0.8161	0.8333	0.7625	0.6111	0.7321	0.6667
Sensor 33	0.8976	0.8824	0.8667	0.9412	0.8667	0.7222	0.881	0.7778	0.9	0.7778	0.9036	0.7222	0.8214	0.7647	0.7929	0.7059
Sensor 36	0.8476	0.9412	0.831	0.9412	0.8833	0.7778	0.8976	0.7778	0.8857	0.8333	0.8857	0.8333	0.75	0.7059	0.7643	0.7647
Average	0.869123	0.898081	0.846615	0.924346	0.797546	0.676354	0.804096	0.727885	0.821773	0.720088	0.821777	0.760681	0.7446231	0.700608	0.732762	0.712804

01.05.20-31.07.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.6857	0.6111	0.7357	0.5556	0.7232	0.6111	0.7107	0.5556	0.7071	0.5	0.7518	0.6111	0.7232	0.6111	0.7262	0.4444
Sensor 8	0.7524	0.7222	0.6786	0.5556	0.7232	0.6111	0.7286	0.6667	0.7071	0.5	0.6929	0.5556	0.7071	0.5	0.6952	0.5
Sensor 9	0.7524	0.6667	0.6762	0.7222	0.7018	0.5	0.7268	0.5556	0.7054	0.5556	0.8036	0.5556	0.7119	0.4444	0.6976	0.6111
Sensor 10	0.7214	0.5556	0.6929	0.5556	0.7768	0.6667	0.7589	0.7778	0.8321	0.7222	0.6625	0.6667	0.669	0.6111	0.6833	0.6111
Sensor 12	0.7048	0.6667	0.6167	0.5	0.5839	0.5	0.5821	0.4444	0.6518	0.6667	0.7321	0.6111	0.7	0.7222	0.6667	0.5556
Sensor 13	0.7357	0.5556	0.6452	0.8333	0.6107	0.7222	0.6375	0.6111	0.7768	0.5556	0.7893	0.6111	0.7262	0.5	0.681	0.4444
Sensor 14	0.6786	0.5	0.7619	0.7778	0.7625	0.7222	0.7482	0.8889	0.7911	0.7222	0.7786	0.6111	0.6548	0.3889	0.6667	0.5556
Sensor 15	0.7024	0.8333	0.7786	0.5556	0.7232	0.6111	0.7071	0.5	0.7107	0.5	0.7339	0.6111	0.6857	0.5	0.6357	0.6667
Sensor 16	0.719	0.7778	0.75	0.6111	0.7071	0.5	0.6946	0.5	0.7357	0.6111	0.7357	0.5	0.6833	0.6111	0.6548	0.5556
Sensor 17	0.8524	0.5556	0.7167	0.6111	0.7286	0.4444	0.7679	0.5556	0.75	0.4444	0.7875	0.5	0.6405	0.5556	0.7429	0.7222
Sensor 18	0.7476	0.5556	0.6524	0.6111	0.7643	0.7222	0.7321	0.7778	0.8179	0.5	0.7607	0.5	0.7429	0.6111	0.6095	0.5
Sensor 19	0.7905	0.3889	0.6452	0.7778	0.6929	0.5556	0.6804	0.6111	0.8036	0.7778	0.6393	0.5556	0.669	0.6111	0.681	0.3889
Sensor 20	0.6476	0.6111	0.719	0.7222	0.7214	0.3889	0.6643	0.5	0.6679	0.5	0.7482	0.8333	0.6095	0.6667	0.669	0.4444
Sensor 21	0.6762	0.7778	0.7976	0.6667	0.7089	0.6667	0.6839	0.6111	0.7482	0.5	0.7464	0.5556	0.6833	0.4444	0.6262	0.6111
Sensor 22	0.7071	0.6111	0.75	0.6667	0.7089	0.5556	0.6946	0.6111	0.6768	0.6667	0.6946	0.6111	0.669	0.5	0.6214	0.5
Sensor 23	0.8429	0.6667	0.7381	0.5556	0.7125	0.5	0.6964	0.4444	0.6946	0.7778	0.7089	0.5556	0.6143	0.5556	0.6976	0.6111
Sensor 24	0.8119	0.6667	0.7214	0.7222	0.8232	0.5	0.7821	0.5	0.7375	0.6111	0.7196	0.5	0.6381	0.5	0.6857	0.6111
Sensor 25	0.7643	0.6111	0.7167	0.6111	0.6821	0.6667	0.6357	0.5556	0.7911	0.6111	0.6768	0.5556	0.7571	0.6111	0.7262	0.6667
Sensor 26	0.7762	0.7222	0.7167	0.6111	0.6696	0.5	0.6268	0.5556	0.6339	0.5	0.7089	0.6111	0.7143	0.6667	0.7119	0.6111
Sensor 27	0.7476	0.6111	0.6524	0.6111	0.6964	0.6111	0.6839	0.6111	0.6375	0.3889	0.6964	0.6667	0.7143	0.5556	0.6095	0.5
Sensor 28	0.7762	0.6111	0.7786	0.5556	0.7107	0.6111	0.6839	0.6111	0.6536	0.5556	0.6536	0.4444	0.7143	0.5556	0.6357	0.6111
Sensor 29	0.6476	0.6111	0.7357	0.5556	0.7214	0.3889	0.6518	0.5	0.6661	0.5	0.7214	0.5556	0.6071	0.6111	0.6548	0.5556
Sensor 30	0.8357	0.5	0.7476	0.6667	0.7	0.4444	0.7679	0.6111	0.7357	0.4444	0.8018	0.4444	0.6524	0.5556	0.7714	0.6111
Sensor 31	0.7524	0.7222	0.6024	0.7222	0.7339	0.7222	0.7464	0.7778	0.8179	0.7222	0.7071	0.6667	0.7571	0.6667	0.6524	0.4444
Sensor 33	0.8048	0.7778	0.8048	0.7778	0.7536	0.6667	0.7536	0.6111	0.7054	0.5556	0.7196	0.7222	0.7571	0.5556	0.7571	0.5556
Sensor 36	0.6643	0.7222	0.6643	0.7222	0.6929	0.5556	0.6946	0.5	0.7196	0.4444	0.7214	0.4444	0.7119	0.5556	0.7119	0.5556
Average	0.742219	0.638896	0.711362	0.647446	0.712835	0.574788	0.701569	0.594023	0.725965	0.570515	0.726638	0.579065	0.6889769	0.564112	0.679669	0.555558

4-month results:

4-month 3-class																
01.08.19-30.11.19	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.6905	0.5625	0.5619	0.625	0.775	0.65	0.7625	0.7	0.775	0.65	0.7875	0.65	0.6357	0.5789	0.6339	0.5263
Sensor 8	0.6952	0.625	0.6952	0.75	0.6125	0.55	0.7625	0.55	0.775	0.55	0.775	0.5	0.7	0.5789	0.6964	0.5789
Sensor 9	0.6367	0.5714	0.6133	0.4286	0.719	0.7647	0.7024	0.8235	0.7024	0.4706	0.6667	0.4118	0.75	0.5625	0.6	0.5
Sensor 10	0.8522	0.6667	0.8533	0.7083	0.7233	0.75	0.6922	0.75	0.7289	0.6	0.6944	0.64	0.6167	0.52	0.7089	0.56
Sensor 12	0.5733	0.6	0.5733	0.6	0.4607	0.5	0.4268	0.4444	0.6536	0.3333	0.6643	0.2778	0.6048	0.4706	0.6238	0.4706
Sensor 13	0.83	0.7083	0.82	0.6667	0.7	0.7917	0.6789	0.9167	0.7478	0.68	0.7389	0.64	0.66	0.6	0.6678	0.6
Sensor 14	0.8522	0.7917	0.8411	0.7917	0.74	0.6667	0.6622	0.6667	0.7711	0.68	0.74	0.6	0.6078	0.52	0.6567	0.56
Sensor 15	0.6232	0.7895	0.6089	0.5789	0.7844	0.625	0.7956	0.6667	0.8089	0.7083	0.8511	0.7083	0.6722	0.5652	0.6278	0.3043
Sensor 16	0.8278	0.7391	0.8278	0.7391	0.6267	0.5417	0.6244	0.5417	0.7322	0.72	0.7622	0.68	0.7533	0.68	0.7133	0.56
Sensor 17	0.7143	0.65	0.7	0.75	0.7622	0.6667	0.75	0.7083	0.7056	0.75	0.7389	0.7083	0.6125	0.5455	0.5792	0.5
Sensor 18	0.8522	0.7083	0.81	0.7917	0.7278	0.7083	0.7067	0.7083	0.7278	0.7083	0.7578	0.625	0.6167	0.48	0.6911	0.48
Sensor 19	0.8433	0.75	0.8556	0.75	0.6622	0.5833	0.6944	0.8333	0.7622	0.72	0.7111	0.8	0.7833	0.64	0.7211	0.64
Sensor 20	0.8111	0.7083	0.8333	0.6667	0.6633	0.625	0.6989	0.75	0.7278	0.56	0.6956	0.56	0.6478	0.56	0.5967	0.44
Sensor 21	0.8089	0.4737	0.7536	0.5263	0.7811	0.8333	0.7711	0.7917	0.7867	0.8333	0.7756	0.7917	0.6375	0.5909	0.6153	0.4545
Sensor 22	0.6905	0.375	0.6286	0.3125	0.7375	0.6	0.7	0.65	0.825	0.65	0.825	0.65	0.5411	0.4737	0.5804	0.4211
Sensor 23	0.6467	0.6	0.63	0.3333	0.6804	0.7368	0.6839	0.6842	0.7036	0.7368	0.7696	0.6842	0.6167	0.4118	0.5881	0.4706
Sensor 24	0.775	0.5	0.735	0.5	0.6767	0.7143	0.64	0.5714	0.7	0.6429	0.6867	0.6429	0.64	0.4615	0.64	0.6154
Sensor 25	0.8389	0.6667	0.85	0.6667	0.7233	0.7083	0.7244	0.625	0.7056	0.48	0.6233	0.56	0.5667	0.6	0.67	0.44
Sensor 26	0.5933	0.5	0.5833	0.2857	0.4643	0.4706	0.5071	0.2941	0.5643	0.7059	0.5762	0.6471	0.6786	0.5	0.7071	0.5
Sensor 27	0.67	0.6667	0.66	0.5	0.4933	0.5714	0.5133	0.5	0.5933	0.4286	0.6433	0.6429	0.61	0.4615	0.565	0.5385
Sensor 28	0.7467	0.8667	0.7667	0.9333	0.619	0.4118	0.619	0.5294	0.6643	0.6111	0.6262	0.6111	0.6476	0.375	0.6452	0.4375
Sensor 29	0.7446	0.65	0.7464	0.65	0.7347	0.6667	0.7569	0.6667	0.7625	0.8182	0.7986	0.7727	0.6403	0.5	0.6139	0.4545
Sensor 30	0.719	0.5294	0.7786	0.5882	0.7875	0.7	0.7625	0.85	0.6625	0.6667	0.675	0.7619	0.6179	0.6316	0.5268	0.4211
Sensor 31	0.6967	0.3571	0.6633	0.5714	0.8233	0.5333	0.7	0.6667	0.81	0.5333	0.7867	0.6	0.7333	0.4	0.7367	0.4667
Sensor 33	0.73	0.5	0.69	0.1667	0.7233	0.5	0.78	0.5714	0.6767	0.6429	0.6267	0.5714	0.62	0.4615	0.62	0.3846
Sensor 36	0.6667	0.75	0.6	0.75	0.6083	0.6667	0.6667	0.7778	0.6167	0.4444	0.65	0.4444	0.6667	0.625	0.7	0.375
Average	0.735731	0.627158	0.718431	0.601185	0.684992	0.636012	0.683938	0.663	0.718827	0.627869	0.717169	0.622365	0.649123	0.530542	0.643277	0.488446

01.12.19-31.03.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.6625	0.7619	0.6125	0.7619	0.7033	0.4348	0.6589	0.4783	0.7722	0.6087	0.7153	0.7826	0.6875	0.4091	0.65	0.4091
Sensor 8	0.7375	0.8571	0.5875	0.8095	0.6156	0.5652	0.6167	0.5217	0.7611	0.6087	0.7389	0.6087	0.7722	0.5909	0.7597	0.5909
Sensor 9	0.65	0.8095	0.6	0.8095	0.7133	0.5652	0.6811	0.4783	0.6903	0.6087	0.625	0.7391	0.6167	0.5	0.6583	0.3636
Sensor 10	0.7833	0.6667	0.7822	0.75	0.6111	0.5833	0.6744	0.625	0.7567	0.7083	0.7256	0.625	0.6656	0.5	0.6878	0.7083
Sensor 12	0.7375	0.7619	0.6625	0.619	0.5478	0.4348	0.5367	0.3913	0.5931	0.6957	0.6069	0.6522	0.7347	0.5455	0.6528	0.4091
Sensor 13	0.76	0.75	0.7044	0.7083	0.6489	0.625	0.6767	0.625	0.7144	0.625	0.6322	0.7917	0.6444	0.4167	0.5933	0.5
Sensor 14	0.7711	0.5417	0.6633	0.8333	0.6556	0.68	0.6311	0.68	0.7544	0.5833	0.7344	0.5417	0.6344	0.4167	0.6333	0.375
Sensor 15	0.6625	0.6667	0.6125	0.8571	0.6911	0.5652	0.6589	0.4783	0.7611	0.7391	0.7278	0.6957	0.7208	0.5909	0.6722	0.5
Sensor 16	0.7078	0.7083	0.6633	0.7917	0.6344	0.4167	0.6333	0.5417	0.69	0.56	0.6911	0.6	0.6067	0.4167	0.6222	0.7083
Sensor 17	0.7232	0.7	0.6732	0.8	0.5811	0.7391	0.6044	0.7391	0.6819	0.6522	0.6611	0.6087	0.6264	0.5	0.6625	0.4091
Sensor 18	0.7456	0.7391	0.7144	0.8696	0.5922	0.68	0.6111	0.72	0.7111	0.5417	0.7033	0.5417	0.6644	0.4167	0.6778	0.7083
Sensor 19	0.77	0.875	0.7378	0.8333	0.7378	0.5	0.7167	0.5833	0.7556	0.7917	0.7456	0.7917	0.7178	0.625	0.6544	0.6667
Sensor 20	0.7189	0.875	0.7833	0.9167	0.7567	0.4583	0.7489	0.4583	0.7456	0.7917	0.7456	0.7917	0.6222	0.5417	0.6544	0.75
Sensor 21	0.6625	0.6667	0.6	0.8095	0.7233	0.5217	0.6811	0.4348	0.75	0.6522	0.6944	0.7391	0.6375	0.5455	0.6819	0.3636
Sensor 22	0.7625	0.8095	0.525	0.8571	0.7233	0.5217	0.7144	0.4783	0.7389	0.6957	0.7056	0.7826	0.65	0.4091	0.6361	0.3636
Sensor 23	0.8232	0.8	0.6857	0.8	0.5989	0.6957	0.6222	0.8261	0.6708	0.6087	0.6611	0.5652	0.6958	0.4545	0.6694	0.4091
Sensor 24	0.75	0.6667	0.575	0.8095	0.5933	0.6957	0.5833	0.6087	0.75	0.5652	0.7194	0.6522	0.7583	0.5455	0.7486	0.5909
Sensor 25	0.7522	0.7083	0.7167	0.7917	0.6144	0.4	0.59	0.68	0.7222	0.5	0.6822	0.5417	0.6333	0.4583	0.6333	0.3333
Sensor 26	0.775	0.7143	0.6875	0.7143	0.58	0.4783	0.6589	0.4348	0.6	0.6522	0.6264	0.5652	0.7	0.5455	0.6222	0.4545
Sensor 27	0.725	0.7143	0.7	0.7619	0.5933	0.3913	0.6256	0.4348	0.6347	0.5652	0.6042	0.5652	0.7083	0.3636	0.6597	0.3636
Sensor 28	0.725	0.7143	0.725	0.7619	0.6033	0.6522	0.6256	0.4348	0.6486	0.5217	0.6042	0.5652	0.7083	0.3636	0.6597	0.3636
Sensor 29	0.7189	0.875	0.7833	0.9167	0.7367	0.4167	0.7489	0.4583	0.7456	0.7917	0.7456	0.7917	0.6422	0.4583	0.6656	0.7083
Sensor 30	0.7607	0.75	0.6857	0.8	0.5811	0.6522	0.6489	0.6522	0.7042	0.6522	0.6736	0.6087	0.6667	0.5455	0.6472	0.4545
Sensor 31	0.7611	0.625	0.75	0.75	0.6133	0.5417	0.6489	0.5417	0.7222	0.5417	0.6933	0.5417	0.6678	0.5833	0.6878	0.7083
Sensor 33	0.8	0.6667	0.6125	0.8095	0.7022	0.6957	0.6589	0.6522	0.7486	0.6957	0.6486	0.6957	0.6708	0.5455	0.6583	0.5909
Sensor 36	0.65	0.8571	0.6125	0.8095	0.7244	0.5217	0.6811	0.4783	0.75	0.6522	0.6944	0.7391	0.6819	0.5909	0.6625	0.4545
Average	0.734462	0.741569	0.671377	0.798135	0.649092	0.555085	0.651412	0.555204	0.714358	0.638815	0.684838	0.658608	0.674412	0.495346	0.661962	0.509888

01.04.20-31.07.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.8111	0.6522	0.7778	0.6522	0.7578	0.7083	0.7378	0.75	0.8089	0.75	0.8189	0.8333	0.7533	0.6957	0.7411	0.6957
Sensor 8	0.7444	0.7826	0.7333	0.913	0.8189	0.7083	0.8189	0.7083	0.7889	0.7083	0.7889	0.7083	0.7044	0.5217	0.7267	0.4783
Sensor 9	0.7444	0.6957	0.7778	0.7391	0.8089	0.6667	0.7989	0.6667	0.8111	0.75	0.8122	0.75	0.76	0.6957	0.7489	0.6957
Sensor 10	0.8044	0.7917	0.8044	0.7917	0.7922	0.875	0.7922	0.875	0.8178	0.7917	0.83	0.75	0.8278	0.7917	0.8278	0.7917
Sensor 12	0.7778	0.5217	0.7333	0.6087	0.4744	0.6667	0.5033	0.7083	0.7133	0.75	0.6822	0.75	0.7189	0.8696	0.7067	0.7826
Sensor 13	0.7944	0.8333	0.8044	0.8333	0.8122	0.9167	0.7922	0.9167	0.83	0.75	0.8411	0.875	0.7656	0.7917	0.7122	0.7917
Sensor 14	0.8167	0.8333	0.8156	0.8333	0.8333	0.8333	0.8222	0.8333	0.8322	0.7917	0.8189	0.75	0.7133	0.7917	0.7222	0.7917
Sensor 15	0.8111	0.5217	0.7556	0.6522	0.7889	0.75	0.78	0.75	0.82	0.75	0.8189	0.8333	0.7644	0.7391	0.7633	0.7391
Sensor 16	0.8244	0.913	0.8244	0.913	0.6878	0.5	0.6656	0.5833	0.7278	0.7917	0.7278	0.7083	0.7967	0.75	0.7978	0.7917
Sensor 17	0.7444	0.6522	0.6889	0.6957	0.8189	0.7083	0.8178	0.6667	0.82	0.625	0.82	0.6667	0.7078	0.5652	0.7289	0.6087
Sensor 18	0.8156	0.7917	0.8156	0.7917	0.8333	0.8333	0.8222	0.8333	0.83	0.875	0.84	0.875	0.7856	0.7917	0.7856	0.7917
Sensor 19	0.8144	0.8333	0.8144	0.8333	0.6778	0.7083	0.6256	0.7917	0.8633	0.75	0.7789	0.7083	0.7733	0.7917	0.7733	0.7917
Sensor 20	0.6622	0.7917	0.6844	0.6667	0.5822	0.75	0.6133	0.75	0.61	0.625	0.6544	0.5417	0.6689	0.75	0.6256	0.6667
Sensor 21	0.8444	0.6522	0.8	0.6087	0.8089	0.75	0.7889	0.75	0.8089	0.7083	0.8511	0.7917	0.7433	0.6957	0.73	0.7826
Sensor 22	0.8	0.6522	0.8111	0.5652	0.8011	0.75	0.8211	0.6667	0.7878	0.7083	0.81	0.7917	0.7633	0.6522	0.7633	0.6087
Sensor 23	0.6889	0.6087	0.7	0.6522	0.8078	0.75	0.8178	0.7083	0.8211	0.625	0.8211	0.6667	0.6967	0.4348	0.7289	0.5652
Sensor 24	0.7111	0.8261	0.6889	0.7391	0.8089	0.7083	0.8089	0.7083	0.7989	0.75	0.7889	0.7083	0.7156	0.5652	0.7256	0.6087
Sensor 25	0.8244	0.8333	0.8256	0.7917	0.7922	0.875	0.7811	0.875	0.81	0.8333	0.81	0.8333	0.7778	0.7917	0.7444	0.7917
Sensor 26	0.8222	0.7391	0.7889	0.7391	0.6089	0.7083	0.62	0.75	0.7244	0.75	0.7244	0.75	0.8156	0.6522	0.8056	0.6087
Sensor 27	0.7667	0.6087	0.7667	0.8261	0.6622	0.6667	0.6944	0.625	0.7878	0.7917	0.7878	0.7917	0.7933	0.6087	0.7933	0.6522
Sensor 28	0.7667	0.7391	0.7667	0.8261	0.6711	0.6667	0.6944	0.625	0.7878	0.7917	0.7878	0.7917	0.7933	0.6087	0.7833	0.5652
Sensor 29	0.6522	0.75	0.6733	0.7083	0.5922	0.8333	0.6033	0.75	0.6211	0.6667	0.6322	0.5417	0.68	0.75	0.6256	0.6667
Sensor 30	0.7778	0.7391	0.7111	0.6522	0.8189	0.7083	0.8178	0.6667	0.82	0.625	0.82	0.6667	0.6967	0.4783	0.7289	0.6087
Sensor 31	0.8156	0.7917	0.8156	0.7917	0.8222	0.8333	0.8222	0.8333	0.84	0.8333	0.83	0.875	0.7956	0.7917	0.7956	0.7917
Sensor 33	0.8222	0.7826	0.7556	0.8261	0.7789	0.7083	0.7789	0.7083	0.8311	0.8333	0.8556	0.8333	0.8233	0.6957	0.8122	0.6957
Sensor 36	0.8	0.6957	0.7778	0.6522	0.7789	0.625	0.7789	0.4583	0.8089	0.75	0.8511	0.8333	0.7311	0.7391	0.7411	0.7391
Average	0.779135	0.732023	0.765815	0.742408	0.747646	0.738773	0.746835	0.729162	0.789273	0.745192	0.792392	0.754808	0.752523	0.692673	0.747612	0.696227

Lockdown results:

covid 3-class																
pre-lockdown 27.01-22.03.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.705	0.8182	0.735	0.7273	0.81	0.3636	0.81	0.2727	0.755	0.8182	0.68	0.8182	0.65	0.4545	0.65	0.8182
Sensor 8	0.77	0.7273	0.76	0.7273	0.735	0.7273	0.74	0.7273	0.84	0.4545	0.755	0.6364	0.7	0.9091	0.725	0.8182
Sensor 9	0.835	0.8182	0.835	0.8182	0.685	0.3636	0.66	0.4545	0.69	0.6364	0.63	0.6364	0.625	0.2727	0.45	0.8182
Sensor 10	0.7	0.6364	0.655	0.7273	0.57	0.6667	0.515	0.8333	0.68	0.6364	0.635	0.6364	0.695	0.6364	0.655	0.7273
Sensor 12	0.71	0.6364	0.755	0.5455	0.455	0.6364	0.505	0.4545	0.6	0.6364	0.535	0.6364	0.65	0.7273	0.625	1
Sensor 13	0.69	0.6364	0.59	0.8182	0.555	0.8333	0.555	0.75	0.715	0.6364	0.645	0.6364	0.69	0.5455	0.67	0.5455
Sensor 14	0.655	0.7273	0.59	0.5455	0.58	0.5833	0.655	0.6667	0.645	0.5455	0.625	0.7273	0.61	0.6364	0.66	0.6364
Sensor 15	0.685	0.6364	0.755	0.7273	0.715	0.4545	0.74	0.3636	0.735	0.6364	0.66	0.7273	0.575	0.1818	0.5	0.8182
Sensor 16	0.695	0.6364	0.67	0.6364	0.455	0.2727	0.455	0.7273	0.69	0.5	0.715	0.5	0.65	0.5833	0.56	0.5
Sensor 17	0.775	0.7273	0.775	0.8182	0.765	0.4545	0.74	0.5455	0.74	0.6364	0.685	0.5455	0.55	0.5455	0.525	0.5455
Sensor 18	0.69	0.8182	0.665	0.8182	0.66	0.8333	0.63	0.6667	0.625	0.5455	0.675	0.5455	0.61	0.5455	0.7	0.6364
Sensor 19	0.74	0.6364	0.69	0.8182	0.66	0.6667	0.695	0.75	0.705	0.7273	0.62	0.6364	0.535	0.7273	0.55	0.1818
Sensor 20	0.67	0.7273	0.59	0.6364	0.595	0.75	0.625	0.5833	0.735	0.7273	0.71	0.6364	0.625	0.6364	0.55	0.5455
Sensor 21	0.71	0.7273	0.76	0.7273	0.76	0.4545	0.74	0.3636	0.725	0.8182	0.665	0.9091	0.625	0.2727	0.525	0.8182
Sensor 22	0.81	0.8182	0.76	0.7273	0.76	0.4545	0.735	0.4545	0.69	0.7273	0.64	0.7273	0.425	0.2727	0.45	0.6364
Sensor 23	0.75	0.7273	0.725	0.7273	0.665	0.3636	0.6	0.4545	0.735	0.6364	0.685	0.6364	0.6	0.1818	0.525	0.7273
Sensor 24	0.73	0.8182	0.73	0.7273	0.665	0.5455	0.57	0.6364	0.84	0.4545	0.755	0.6364	0.675	0.8182	0.75	0.8182
Sensor 25	0.695	0.6364	0.62	0.6364	0.6	0.75	0.65	0.6667	0.695	0.5455	0.595	0.5455	0.65	0.7273	0.685	0.5455
Sensor 26	0.645	0.5455	0.745	0.6364	0.495	0.5455	0.53	0.5455	0.55	0.4545	0.515	0.7273	0.65	0.6364	0.6	0.8182
Sensor 27	0.825	0.6364	0.825	0.7273	0.505	0.6364	0.525	0.6364	0.595	0.4545	0.45	0.9091	0.7	0.6364	0.6	0.8182
Sensor 28	0.8	0.6364	0.825	0.7273	0.57	0.5455	0.545	0.5455	0.575	0.6364	0.52	0.8182	0.7	0.6364	0.6	0.8182
Sensor 29	0.67	0.7273	0.59	0.6364	0.59	0.75	0.625	0.5833	0.735	0.7273	0.71	0.6364	0.625	0.6364	0.54	0.7273
Sensor 30	0.825	0.6364	0.75	0.8182	0.785	0.4545	0.715	0.5455	0.715	0.6364	0.685	0.5455	0.55	0.6364	0.525	0.4545
Sensor 31	0.75	0.7273	0.585	0.8182	0.66	0.8333	0.605	0.6667	0.58	0.3636	0.655	0.7273	0.635	0.6364	0.7	0.6364
Sensor 33	0.735	0.8182	0.785	0.8182	0.835	0.4545	0.785	0.3636	0.73	0.6364	0.705	0.7273	0.525	0.2727	0.6	0.8182
Sensor 36	0.73	0.8182	0.71	0.7273	0.735	0.4545	0.74	0.2727	0.735	0.7273	0.705	0.8182	0.5	0.2727	0.55	0.8182
Average	0.730577	0.709819	0.710577	0.723804	0.648654	0.571085	0.641923	0.558858	0.694423	0.613654	0.648269	0.6801	0.616346	0.539931	0.595	0.694077

lockdown 23.03-17.05.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.71	0.6364	0.54	0.4545	0.68	0.5455	0.67	0.5455	0.655	0.4545	0.625	0.5455	0.7	0.6364	0.53	0.5455
Sensor 8	0.665	0.8182	0.69	0.6364	0.655	0.6364	0.68	0.6364	0.75	0.7273	0.695	0.8182	0.675	0.5455	0.465	0.5455
Sensor 9	0.705	0.6364	0.57	0.4545	0.655	0.4545	0.6	0.4545	0.735	0.5455	0.66	0.9091	0.695	0.6364	0.56	0.1818
Sensor 10	0.725	0.8182	0.75	0.8182	0.77	0.6364	0.745	0.6364	0.7	0.5833	0.72	0.6667	0.68	0.6667	0.6	0.5833
Sensor 12	0.79	0.3636	0.67	0.3636	0.45	0.4545	0.425	0.3636	0.53	0.7273	0.385	0.7273	0.635	0.5455	0.58	0.5455
Sensor 13	0.765	0.7273	0.735	0.7273	0.69	0.6364	0.69	0.5455	0.625	0.6667	0.62	0.5833	0.55	0.6667	0.525	0.3333
Sensor 14	0.72	0.7273	0.695	0.8182	0.67	0.75	0.67	0.75	0.695	0.5	0.69	0.5	0.61	0.75	0.56	0.5
Sensor 15	0.645	0.4545	0.6	0.4545	0.705	0.6364	0.655	0.6364	0.64	0.4545	0.665	0.6364	0.73	0.6364	0.5	0.1818
Sensor 16	0.795	0.7273	0.825	0.7273	0.36	0.6667	0.44	0.5833	0.64	0.75	0.615	0.75	0.645	0.6667	0.53	0.6667
Sensor 17	0.695	0.7273	0.72	0.7273	0.575	0.5455	0.57	0.7273	0.75	0.7273	0.665	0.7273	0.675	0.5455	0.66	0.5455
Sensor 18	0.73	0.7273	0.725	0.6364	0.71	0.6667	0.755	0.8333	0.715	0.6667	0.69	0.5	0.665	0.6667	0.585	0.6667
Sensor 19	0.68	0.6364	0.655	0.6364	0.555	0.9091	0.57	0.4545	0.745	0.5	0.715	0.5833	0.735	0.5	0.455	0.5833
Sensor 20	0.65	0.4545	0.67	0.5455	0.565	0.3636	0.49	0.8182	0.57	0.6667	0.565	0.6667	0.435	0.75	0.425	0.6667
Sensor 21	0.725	0.5455	0.725	0.6364	0.63	0.6364	0.67	0.6364	0.605	0.6364	0.475	0.2727	0.765	0.6364	0.515	0.2727
Sensor 22	0.615	0.5455	0.615	0.6364	0.7	0.4545	0.63	0.4545	0.66	0.9091	0.635	0.9091	0.7	0.5455	0.58	0.5455
Sensor 23	0.68	0.5455	0.755	0.6364	0.48	0.6364	0.5	0.7273	0.665	0.7273	0.615	0.8182	0.675	0.4545	0.635	0.5455
Sensor 24	0.64	0.6364	0.735	0.9091	0.635	0.6364	0.68	0.6364	0.77	0.5455	0.725	0.7273	0.63	0.7273	0.51	0.6364
Sensor 25	0.7	0.7273	0.765	0.7273	0.68	0.5455	0.7	0.6364	0.69	0.6667	0.645	0.6667	0.675	0.5833	0.535	0.5833
Sensor 26	0.675	0.4545	0.655	0.5455	0.445	0.3636	0.425	0.3636	0.54	0.6364	0.435	0.7273	0.635	0.5455	0.61	0.4545
Sensor 27	0.67	0.5455	0.62	0.4545	0.595	0.5455	0.595	0.6364	0.5	0.6364	0.46	0.6364	0.645	0.4545	0.595	0.4545
Sensor 28	0.67	0.5455	0.62	0.5455	0.595	0.5455	0.595	0.6364	0.5	0.6364	0.465	0.5455	0.62	0.4545	0.595	0.4545
Sensor 29	0.63	0.4545	0.67	0.5455	0.565	0.3636	0.49	0.8182	0.55	0.75	0.565	0.6667	0.44	0.75	0.45	0.6667
Sensor 30	0.685	0.7273	0.72	0.7273	0.57	0.5455	0.57	0.7273	0.755	0.6364	0.66	0.7273	0.63	0.4545	0.66	0.5455
Sensor 31	0.73	0.7273	0.75	0.6364	0.725	0.6364	0.775	0.5455	0.715	0.6667	0.69	0.5	0.64	0.6667	0.585	0.5833
Sensor 33	0.665	0.6364	0.625	0.6364	0.7	0.4545	0.695	0.5455	0.595	0.5455	0.445	0.8182	0.67	0.5455	0.6	0.3636
Sensor 36	0.645	0.4545	0.58	0.4545	0.7	0.6364	0.67	0.6364	0.565	0.6364	0.545	0.6364	0.725	0.5455	0.635	0.5455
Average	0.6925	0.6154	0.68	0.618896	0.617692	0.57315	0.613654	0.614815	0.648462	0.638423	0.602692	0.664062	0.649231	0.599085	0.556923	0.507581

post-lockdown 18.05-12.07.20	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.725	0.3636	0.75	0.3636	0.74	0.7273	0.615	0.7273	0.655	0.5455	0.625	0.3636	0.71	0.6364	0.55	0.3636
Sensor 8	0.665	0.5455	0.64	0.6364	0.625	0.5455	0.65	0.5455	0.485	0.6364	0.445	0.6364	0.71	0.7273	0.65	0.6364
Sensor 9	0.59	0.7273	0.64	0.7273	0.625	0.6364	0.45	0.5455	0.76	0.5455	0.665	0.6364	0.715	0.4545	0.67	0.4545
Sensor 10	0.71	0.7273	0.67	0.7273	0.66	0.6364	0.62	0.6364	0.695	0.9091	0.67	0.9091	0.695	0.4545	0.57	0.4545
Sensor 12	0.73	0.4545	0.75	0.5455	0.41	0.2727	0.385	0.0909	0.61	0.4545	0.54	0.6364	0.78	0.6364	0.695	0.5455
Sensor 13	0.71	0.6364	0.71	0.6364	0.62	0.6364	0.62	0.9091	0.68	0.8182	0.635	0.7273	0.64	0.5455	0.585	0.9091
Sensor 14	0.66	0.4545	0.71	0.7273	0.61	0.8182	0.61	0.8182	0.72	0.9091	0.655	0.9091	0.615	0.6364	0.67	0.6364
Sensor 15	0.7	0.5455	0.675	0.4545	0.67	0.6364	0.635	0.7273	0.655	0.4545	0.695	0.6364	0.64	0.4545	0.55	0.2727
Sensor 16	0.785	0.6364	0.705	0.6364	0.58	0.7273	0.54	0.4545	0.545	0.6364	0.605	0.8182	0.735	0.6364	0.665	0.7273
Sensor 17	0.615	0.5455	0.615	0.7273	0.675	0.6364	0.605	0.5455	0.72	0.6364	0.65	0.6364	0.65	0.6364	0.7	0.5455
Sensor 18	0.665	0.7273	0.78	0.7273	0.625	0.9091	0.585	1	0.67	0.9091	0.675	0.9091	0.595	0.6364	0.55	0.6364
Sensor 19	0.645	0.4545	0.52	0.7273	0.625	0.5455	0.6	0.6364	0.645	0.7273	0.62	0.5455	0.675	0.7273	0.71	0.5455
Sensor 20	0.765	0.6364	0.715	0.6364	0.675	0.8182	0.655	0.6364	0.62	0.5455	0.645	0.6364	0.77	0.7273	0.745	0.8182
Sensor 21	0.725	0.5455	0.775	0.6364	0.68	0.7273	0.64	0.8182	0.68	0.6364	0.605	0.5455	0.735	0.5455	0.62	0.5455
Sensor 22	0.725	0.6364	0.725	0.6364	0.65	0.6364	0.45	0.6364	0.7	0.4545	0.7	0.4545	0.605	0.6364	0.55	0.4545
Sensor 23	0.61	0.3636	0.66	0.5455	0.615	0.7273	0.65	0.6364	0.675	0.5455	0.605	0.6364	0.675	0.6364	0.645	0.6364
Sensor 24	0.61	0.7273	0.585	0.8182	0.635	0.4545	0.655	0.6364	0.65	0.6364	0.535	0.7273	0.655	0.5455	0.615	0.6364
Sensor 25	0.685	0.7273	0.695	0.6364	0.51	0.8182	0.535	0.5455	0.7	0.7273	0.615	0.5455	0.63	0.5455	0.625	0.8182
Sensor 26	0.71	0.5455	0.775	0.5455	0.475	0.5455	0.465	0.5455	0.635	0.7273	0.565	0.6364	0.785	0.7273	0.765	0.7273
Sensor 27	0.76	0.5455	0.68	0.6364	0.57	0.4545	0.65	0.4545	0.675	0.6364	0.63	0.6364	0.755	0.6364	0.74	0.7273
Sensor 28	0.705	0.4545	0.68	0.6364	0.57	0.4545	0.65	0.4545	0.675	0.6364	0.635	0.5455	0.755	0.6364	0.74	0.7273
Sensor 29	0.765	0.6364	0.715	0.6364	0.675	0.8182	0.655	0.6364	0.62	0.5455	0.645	0.6364	0.77	0.7273	0.745	0.8182
Sensor 30	0.635	0.7273	0.59	0.7273	0.68	0.6364	0.605	0.5455	0.72	0.6364	0.65	0.6364	0.65	0.6364	0.655	0.6364
Sensor 31	0.74	0.7273	0.76	0.7273	0.625	0.9091	0.56	1	0.695	0.9091	0.65	0.8182	0.505	0.7273	0.525	0.5455
Sensor 33	0.755	0.4545	0.75	0.4545	0.61	0.6364	0.555	0.6364	0.58	0.6364	0.55	0.5455	0.715	0.5455	0.58	0.6364
Sensor 36	0.725	0.7273	0.725	0.5455	0.725	0.6364	0.705	0.7273	0.625	0.4545	0.63	0.5455	0.665	0.6364	0.57	0.6364
Average	0.696731	0.587427	0.692115	0.632892	0.621538	0.653865	0.590192	0.636385	0.657308	0.650369	0.620769	0.650377	0.685769	0.618908	0.641731	0.6189

pre_until_post lockdown 27.01.2020-12.07.2020	07--10				12--15				15--18				19--22			
Sensors	RF		LR		RF		LR		RF		LR		RF		LR	
	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
Sensor 7	0.7923	0.6875	0.8077	0.7812	0.7359	0.7879	0.7827	0.8788	0.7621	0.697	0.7632	0.697	0.7288	0.7812	0.6968	0.75
Sensor 8	0.8013	0.875	0.8333	0.8438	0.791	0.8182	0.7756	0.8182	0.7698	0.7879	0.7401	0.8182	0.6628	0.75	0.6397	0.8438
Sensor 9	0.7846	0.875	0.8071	0.8125	0.759	0.8485	0.7288	0.8485	0.7923	0.8182	0.7698	0.8182	0.7212	0.75	0.6801	0.75
Sensor 10	0.7872	0.7576	0.8122	0.7576	0.7357	0.7647	0.75	0.8235	0.811	0.697	0.7951	0.6364	0.717	0.697	0.733	0.7879
Sensor 12	0.6897	0.8438	0.7237	0.875	0.5885	0.4848	0.5885	0.5152	0.7258	0.8182	0.7033	0.8182	0.7333	0.8125	0.7173	0.8125
Sensor 13	0.7974	0.7188	0.8205	0.8438	0.7588	0.7647	0.7736	0.7647	0.8418	0.697	0.8341	0.7273	0.7632	0.697	0.7401	0.697
Sensor 14	0.7545	0.8485	0.7885	0.8182	0.767	0.7941	0.7445	0.7647	0.7885	0.697	0.7879	0.7273	0.6868	0.7273	0.7176	0.7879
Sensor 15	0.8013	0.6875	0.8006	0.75	0.7744	0.8182	0.7904	0.8485	0.7775	0.7576	0.7703	0.7879	0.7218	0.7188	0.6737	0.7188
Sensor 16	0.8365	0.7812	0.8436	0.8438	0.6374	0.7273	0.6736	0.6061	0.6879	0.7647	0.6945	0.7353	0.7104	0.5758	0.6951	0.6061
Sensor 17	0.8237	0.7812	0.8154	0.7812	0.7282	0.7879	0.7359	0.8182	0.739	0.7879	0.7478	0.8485	0.7045	0.7812	0.6885	0.7812
Sensor 18	0.7949	0.9062	0.8353	0.8438	0.7813	0.7059	0.8044	0.7353	0.8104	0.6061	0.7956	0.6364	0.7093	0.697	0.7484	0.7879
Sensor 19	0.7968	0.8438	0.8288	0.9375	0.7203	0.5294	0.728	0.6471	0.8044	0.6364	0.7885	0.6667	0.8011	0.7273	0.7934	0.7576
Sensor 20	0.7494	0.7812	0.7583	0.8125	0.6879	0.7059	0.6863	0.6471	0.7209	0.8485	0.7511	0.8485	0.7489	0.7273	0.7929	0.7273
Sensor 21	0.8	0.7812	0.8071	0.8438	0.784	0.7576	0.791	0.7879	0.7786	0.7576	0.8011	0.7879	0.7615	0.7812	0.7128	0.75
Sensor 22	0.7769	0.7188	0.7853	0.7812	0.7442	0.8485	0.7436	0.8485	0.7308	0.7576	0.7544	0.7273	0.6808	0.6875	0.6564	0.6562
Sensor 23	0.8224	0.75	0.791	0.75	0.7603	0.697	0.7756	0.697	0.7775	0.697	0.7929	0.7576	0.6737	0.7188	0.6731	0.7188
Sensor 24	0.7679	0.9688	0.8244	0.9062	0.7442	0.8788	0.7679	0.7273	0.7549	0.8182	0.7621	0.7879	0.6705	0.75	0.6647	0.7812
Sensor 25	0.766	0.7812	0.8128	0.9062	0.7588	0.7353	0.7429	0.7353	0.7654	0.7273	0.75	0.697	0.7027	0.6364	0.7341	0.7273
Sensor 26	0.7929	0.875	0.8013	0.8438	0.6263	0.5758	0.5724	0.6061	0.6945	0.8485	0.7022	0.8182	0.7705	0.8125	0.7545	0.9062
Sensor 27	0.7679	0.875	0.7776	0.8438	0.6269	0.6667	0.6718	0.5758	0.7335	0.8788	0.7104	0.8485	0.7314	0.875	0.7	0.875
Sensor 28	0.7763	0.875	0.7769	0.875	0.6346	0.5758	0.6801	0.5758	0.7335	0.8788	0.7104	0.8485	0.7308	0.875	0.7154	0.875
Sensor 29	0.7487	0.7812	0.75	0.8125	0.6879	0.7059	0.6863	0.6471	0.7275	0.8485	0.7511	0.8485	0.7418	0.7273	0.8005	0.7273
Sensor 30	0.8071	0.75	0.8071	0.7812	0.7276	0.697	0.7359	0.8182	0.7313	0.8182	0.7484	0.8485	0.7045	0.7812	0.6968	0.8125
Sensor 31	0.7647	0.7879	0.8141	0.7273	0.7357	0.7059	0.7495	0.7353	0.8181	0.6364	0.7956	0.6667	0.7093	0.7273	0.7484	0.7879
Sensor 33	0.8231	0.875	0.8237	0.875	0.7987	0.7879	0.7987	0.7576	0.7934	0.7576	0.7852	0.7879	0.7192	0.8438	0.6801	0.8438
Sensor 36	0.8314	0.75	0.8154	0.7812	0.7833	0.7879	0.7904	0.8485	0.7929	0.7273	0.7709	0.7879	0.6897	0.6875	0.6974	0.75
Average	0.786727	0.806015	0.802373	0.824158	0.726073	0.729138	0.7334	0.733704	0.763973	0.760204	0.760615	0.768396	0.719058	0.744073	0.713492	0.769969

Grouped results 3-class

		07--10				12--15				15--18				19--22			
		RF		LR		RF		LR		RF		LR		RF		LR	
	Period	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy	Score	Accuracy
whole	01.08.19-31.08.20	0.826	0.842	0.820	0.830	0.747	0.729	0.739	0.724	0.774	0.759	0.768	0.757	0.691	0.676	0.681	0.652
4-month	01.08.19-30.11.19	0.736	0.627	0.718	0.601	0.685	0.636	0.684	0.663	0.719	0.628	0.717	0.622	0.649	0.531	0.643	0.488
	01.12.19-31.03.20	0.734	0.742	0.671	0.798	0.649	0.555	0.651	0.555	0.714	0.639	0.685	0.659	0.674	0.495	0.662	0.510
	01.04.20-31.07.20	0.779	0.732	0.766	0.742	0.748	0.739	0.747	0.729	0.789	0.745	0.792	0.755	0.753	0.693	0.748	0.696
3-month	01.08.19-31.10.19	0.741	0.690	0.734	0.659	0.695	0.596	0.678	0.629	0.721	0.645	0.725	0.674	0.642	0.544	0.619	0.539
	01.11.19-31.01.20	0.720	0.578	0.699	0.527	0.639	0.623	0.595	0.644	0.653	0.638	0.648	0.681	0.659	0.588	0.629	0.563
	01.02.20-30.04.20	0.869	0.898	0.847	0.924	0.798	0.676	0.804	0.728	0.822	0.720	0.822	0.761	0.745	0.701	0.733	0.713
	01.05.20-31.07.20	0.742	0.639	0.711	0.647	0.713	0.575	0.702	0.594	0.726	0.571	0.727	0.579	0.689	0.564	0.680	0.556
lockdown	pre_lockdown 27.01-22.03.20	0.731	0.710	0.711	0.724	0.649	0.571	0.642	0.559	0.694	0.614	0.648	0.680	0.616	0.540	0.595	0.694
	lockdown 23.03-17.05.20	0.693	0.615	0.680	0.619	0.618	0.573	0.614	0.615	0.648	0.638	0.603	0.664	0.649	0.599	0.557	0.508
	post-lockdown 18.05-12.07.20	0.697	0.587	0.692	0.633	0.622	0.654	0.590	0.636	0.657	0.650	0.621	0.650	0.686	0.619	0.642	0.619
	pre_until_post lockdown 27.01.2020-	0.787	0.806	0.802	0.824	0.726	0.729	0.733	0.734	0.764	0.760	0.761	0.768	0.719	0.744	0.713	0.770