OPTIMIZING URBAN SAFETY AND TRAFFIC MANAGEMENT: A MACHINE LEARNING APPROACH TO SENSOR PLACEMENT FOR INTELLIGENT TRANSPORTATION AND CRIME DETECTION SYSTEMS

DENIS NEDELJKOVIC

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF ARTS

GRADUATE PROGRAM IN INFORMATION SYSTEMS AND TECHNOLOGIES

YORK UNIVERSITY

TORONTO ONTARIO

MARCH, 2024

Abstract

This thesis explores the use of machine learning algorithms for optimizing sensor placement in urban areas to improve crime prevention and traffic management. Focused on the City of Toronto as a case study, it integrates traffic sensor data with vehicular crime statistics to propose a model predicting potential hotspots for traffic violations and optimal locations for crime-prevention sensors. This research employs the use of Random Forest, Long Short-Term Memory, Fourier Series Neural Networks, Support Vector Machine, and the Feed Forward Neural Network, to provide insights that are actionable for safer, smart cities. Providing these results for government officials, law enforcement agencies, and research with an ease of access cloud-based tool to serve as a Software as a Service format. The study underscores the importance of leveraging advanced data analytics in urban planning, suggesting a direction for future research and implementation in intelligent transportation systems.

Acknowledgements

First and foremost, I extend my deepest gratitude to my thesis supervisor, Dr. Manar Jammal, whose expertise, understanding, and patience have been the bedrock of this journey. Without Dr. Manar Jammal's invaluable guidance, this thesis simply would not have seen light. Her insights and wisdom have been instrumental throughout the entire process of research and writing. Dr. Manar Jammal's commitment to excellence and her unwavering support have encouraged me to push my boundaries and strive for the highest standards in my work. Her ability to challenge my thinking, while providing constructive feedback, has fostered an environment of growth and learning. The trust and confidence she placed in me were paramount to my ability to navigate the complexities of this research. I am immensely thankful for the privilege of her mentorship, which has been a cornerstone of my academic and personal development. Working under her guidance has been an enriching experience that I will cherish as I move forward in my career.

I would also like to express my profound gratitude to my parents, Svetozar Nedeljkovic and Zeljka Kupek, and my sister, Isidora Nedeljkovic, for their unwavering support and encouragement. Their belief in my abilities and constant encouragement have been a driving force behind my dedication and perseverance. Their influence has not only propelled me to pursue my Master’s degree at York University but has also been a cornerstone in the completion of this work. It is with their support that I have navigated the challenges of this journey, reaching this final milestone with a work that I am proud to present. Their encouragement has been one of the pivotal reasons for my pursuit of higher education and this achievement is as much theirs as it is mine.

Table of Contents

List of Tables

List of Figures

Preface

The research documented in this thesis represents the original works of Denis Nedeljkovic, undertaken in supervision by Dr. Manar Jammal. This thesis adheres to the paper-based format, in line with the prescribed guidelines for such submissions. Portions of this report have been accepted for the peer reviewed publications indicated below:

- D. Nedeljkovic, N. Y. Fares, and M. Jammal (2023). A Novel Machine Learning-based Approach to City Crime Sensor Placement Prediction. IEEE World Forum on Internet of Things, vol. 9.

Parts of this report are still undergoing review by the following venues:

- D. Nedeljkovic, N. Y. Fares, and M. Jammal (2024). Enhancing City Traffic Safety through Machine Learning-Optimized Sensor Placement for Intelligent Transportation Systems (To be Submitted)
- D. Nedeljkovic, and M. Jammal (2024). Cloud-Based Perspective for Intelligent Transportation and Crime Prevention: A Web Deployment Solution (Submitted to The 33rd International Conference on Computer Communications and Networks)

Abbreviations

AI	Artificial Intelligent
ASE	Automated Speed Enforcement
CPU	Central Processing Unit
C	Regularization Parameter
DBLA	Dynamic Load Balancing
FNN	Feed Forward Neural Network
FSNN	Fourier Series Neural Network
GPS	Global Positioning System
GPU	Graphical Processing Unit
HTML	HyperText Markup Language
GUI	Graphical User Interface
LSTM	Long Short-Term Memory
ITS	Intelligent Transportation System
IOT	Internet of Things
kPCA	Kernal Principal Component Analysis
L1	Lasso Regularization
L2	Ridge Regularization

ML	Machine Learning
MLDF	Machine-Learning Data Fusion
MSE	Mean Squared Error
PCT	Percentile Speed Observed
R2	R-Square
RBF	Radial Basis Function
RAM	Random Access Memory
RF	Random Forest
SSH	Secure Shell
SHAP	SHapley Additive exPlanations
SAAS	Software as a Service
SPD	Count of Vehicles Observed
SVM	Support Vector Machine
VRAM	Virtual Random Access Memory
vGPU	Virtual Graphical Processing Unit
WYS	Watch Your Speed
WYSP	Watch Your Speed Program

1. Introduction

Road traffic and its accompanying effects wield significant influence over the quality of urban life. Studies have highlighted the correlation between traffic congestion and negative behavioral traits such as impulsiveness, lack of self-control, and a predisposition towards criminal behavior [1]. The repercussions of such behaviors are not trivial; for instance, the City of Toronto reported 78 individual fatalities stemming from 76 fatal traffic collisions in 2016 [2], underscoring the public's concern over traffic safety. This concern has prompted several initiatives, including the Vision Zero Plan (2017) aimed at eliminating traffic fatalities. However, Toronto's struggles are not unique; in 2020, Portugal witnessed 27,725 accidents with victims, leading to 536 deaths [3], indicating a global crisis in road safety.

In response to these challenges, there is a growing momentum towards digitalization and the transformation of urban centers into smart cities. Such a transition advocates for the adoption of Intelligent Transportation Systems (ITS) and the integration of smart technologies, including Machine Learning (ML), to address traffic-related concerns. The utilization of ML not only aids in the identification of high-priority areas plagued by traffic violations but also facilitates the allocation of resources to mitigate the impact of traffic-related crimes and offenses.

Building upon this foundation, my research delves into the development of a more sophisticated investigative approach through the application of machine learning algorithms, transitioning towards a predictive methodology for the strategic placement of traffic-related sensors. By harnessing the power of advanced data analysis and non-linear predictive modeling, this study aims to furnish government and law enforcement agencies with actionable insights, thereby elevating public safety standards to unprecedented levels. The focus is on innovating

urban safety strategies by deploying crime prevention sensors and resources in areas identified through cutting-edge AI technology as requiring utmost attention.

This thesis sets out to explore these innovative methodologies with the City of Toronto serving as a case study. By integrating traffic sensor data with vehicular crime statistics, we propose a model that not only predicts potential hotspots for traffic violations but also suggests optimal locations for the deployment of crime-prevention sensors. The implications of this research are far-reaching, offering a blueprint that can be adapted by other municipalities in Ontario, Canada, and globally, to foster safer and more intelligent cities. This only possible by deploying this work in a cloud-based solution capable of being accessed by all on any device as long as the appropriate files are ready.

1.1 Motivation and Objectives

This study's central research question is: How can machine learning algorithms be effectively utilized to optimize the placement of sensors in urban areas for improved crime prevention and traffic management? For this purpose, we ask several sub-questions:

- What are the most significant features that influence the predictive placement from the historical sensor data?
- Would the classic machine learning approach suit, or can a deep learning algorithm perform better?
- Could the tool be properly integrated in a cloud solution with the ML pipeline as a key focus?
- Furthermore, how can this application of machine learning for crime-based sensor predictive placement be accessible to many, regardless of technical expertise?

As the work involves a novel idea, exploring these questions requires an expansive search to understand an ideal outcome best.

It is widely evident that collecting suitable data and focusing on the best technology will yield powerful results on screen and records. However, effort must be added for proper human resources and policy management to achieve an impactful outcome on real-life streets. Therefore, this work proposes a potential solution. It comprises several data stems, each forming all possible relevant features and sensors. Predictive approaches, rather than detective ones, in advanced technology are preferable in the case of crime and offences, as the end goal is to preclude traffic violations and minimize resulting risks. These questions aim to explore the multifaceted nature of applying machine learning in an urban safety context, encompassing technical and practical considerations.

1.2 Outline

The structure of the remainder of this thesis is outlined as follows:

- Chapter 2 reviews the existing literature on the integration of machine learning and sensory data in traffic Internet of Things (IoT) systems. This chapter highlights the broader context of the field, even though it does not directly touch upon the specific niche area of this research.
- Chapter 3 introduces the exploration of different machine learning models that could solve the proposed challenge.
- Chapter 4 discusses the optimization of the pre-existing approaches, improving the final results with further data and model exploration, and a development and implementation of a graphical user interface (GUI).

- Chapter 5 presents a cloud-based tool solution for a ML pipeline from inputting data to examining the final results.
- Chapter 6 discusses the final conclusions drawn from the research. It discusses their implications for the field of traffic IoT systems in a smart city, and suggests directions for future research.

2. Background

2.1 Literature Review

Various research involving sensory data and ML in traffic related approaches have been conducted. Chen believes that due to high accuracy and less human involvement, the machine learning approach in traffic flow prediction has become a hot topic that officials must consider [4]. Sandim and others discuss how new models and algorithms that can process location traces are used to predict traffic congestion [5]. Yildirim uses several sensory measurements and computes vehicle speed predictions using ML methods [6]. Tang et al. propose an improved version of the Markov Chain framework to estimate traffic flow data series considering spatial-temporal correlation links [7]. Al Mamlouk studies the importance of employing accurate machine-learning models to predict traffic accident severity for transportation systems [8]. They use a variety of supervised machine-learning algorithms and achieve 75.5 percent accuracy. Their limitation was mainly the lack of relevant data. Mostafa's paper proposes a machine learning-based data fusion (MLDF) model to perform cumulative traffic analysis using decision-making techniques and classification algorithms to automate traffic violation detection and categorize its types online [9]. Moses's research aims to develop an automobile crime prediction system to effectively predict crimes and notify responsible authorities in advance to prepare them before any offensive occurrence [10]. In their comprehensive review of previous research and studies, Nama provides what can be considered a three-tier solution for issues regarding deploying machine learning in the transportation system [11]. They discuss proper data collection tools, explain machine learning algorithms and their accuracy, and emphasize the importance of adequate traffic police scheduling strategies.

It is widely evident that collecting suitable data and focusing on the best technology will yield powerful results on screen and records. However, to achieve an impactful outcome on real-life streets, a must is to add proper human resources and policy management. Therefore, this study proposes a potential solution. It comprises several data sets, each forming all possible relevant features and sensors. We believe that predictive approaches, rather than detective ones, in advanced technology are preferable in the case of crime and offenses, as the end goal is to preclude traffic violations and minimize resulting risks.

To the best of this study's knowledge, there is no work in such an area and most of the work focus on traffic prediction. This report covers the niche area of crime sensor placement based on a machine-learning predictive approach, emphasizing vehicle-related crime and speed cameras.

2.2 Internet of Things and Intelligent Transportation Systems

This work has an application research area that is focused around current and upcoming Smart Cities. To that extent it can be broken down into two different sub-areas of research; Internet of Things (IoT), and Intelligent Transportation Systems (ITS). One handles the cloud-based operations, while the other integrates traffic related challenges.

The phrase Internet of Things (IoT) refers to connecting various physical devices and objects throughout the world via internet [12]. To address this in terms related to this work, IoT is incorporated as it is the handling of data from many sensor units from two types of sensors with different objective functionality. The collection of this data is then publicly available from official government databases accessible via the internet.

The capability of providing valuable directives to drivers, as well as to elements of the transportation infrastructure (e.g., traffic lights), catering for a more efficient and safer mobility. [13]. In terms of the relation to this work, each of the previously mentioned sensors are in some capacity apart of a traffic safety. One tickets the driver for a speeding infraction relative to excess speed above the posted limit. The other informs drivers of their current speed if it is near or above the posted limit.

3. A Novel Machine Learning-based Approach to City Crime Sensor Placement Prediction

3.1 Introduction

In an era where urban safety and efficient traffic management are paramount, the quest for innovative solutions has led to the exploration of data-driven approaches to prevent and predict traffic violations. The intersection of technology and urban planning presents an opportunity to leverage the vast amounts of data generated by city infrastructures and law enforcement agencies.

The research in this chapter was conducted in search to find a far more efficient method of placement of crime related traffic sensors, in terms of prediction vs the current method of investigation. The approach described in this thesis covers merging pre-existing traffic sensor data from two sources with vehicular crime data. Then with the use of three ML regression models, we predict the latitude and longitude of possible new hotspots and sort them by predicted ticket count. The contributions are summarized as follows.

- We formulate the problem of predictive placement based on historical ticket measurements from the Automated Speed Enforcement locations, Toronto police crime statistics, and Watch Your Speed sign historical data. we choose such a formulation of data to determine if this novel predictive method can perform or assist the current investigative method.

- We developed three models for each regression learning approach, for target features of latitude, longitude, and ticket count, to allow predictive placement of hotspots based on the greatest possible ticket counts.
- We propose the Fourier Series Neural Network, Long Short-term Memory, and Random Forest approaches to determine which method provides the most efficient and accurate method of placement.

3.2 Problem Statement

Traffic violations and crimes can have a negative impact on civilians and the community as a whole. Speeding, reckless driving, and running past red lights increase accident risks, resulting in injuries, fatalities, and property damage. Traffic violations can also cause congestion, leading to increased travel times and delays for commuters and environmental pollution. Moreover, traffic offenses result in fines, points on a driver's license, and even jail time in some cases. Drivers who commit traffic crimes may also see their insurance rates increase, making driving more expensive [14]. Therefore, aiming for a reduction of traffic violations would have a positive impact on all aspects of society.

Efforts to curb the traffic crime rate are facing several challenges and limitations. These include the availability and suitability of data, choosing the best model/algorithms, determining the end goal (detection vs. prediction), and applicability in real life.

Therefore, we study the predictive potential of three machine learning approaches; a single non-neural network and two neural network machine learning architectures, to compare performances and determine the most effective. To accomplish trustworthy street safety measures, we then use the results to determine the most common speed camera location hotspots

and, thus, support governmental officials and policymakers in assigning adequate resources to the places that need them the most in hopes of replacing or assisting a cumbersome investigative approach.

3.3 Methodology

The approach to this problem requires multiple related data sets that must be merged, cleaned, and standardized into two final groups. In this chapter, the City of Toronto is the use case. Still, the approach can be applied to multiple cities, as the initial data sets are not unique to the town and can be found in numerous government entities. These data sets require features related to various crime rates, different related volumetric data of vehicles, and various volumetric data of speeding tickets served from the Automated Speed Enforcement, also known as ASE. In addition, they contained locations in a descriptive or Global Positioning System, also known as GPS, standard found within the City of Toronto.

From these data sets, we select the relevant features of the problem. With these selected features, we aggregate the data to represent better any outliers found in the collections. Afterward, the data is standardized as the final step before splitting the results into training and prediction data sets. This will be further discussed in this chapter.

The training data set runs through three different machine learning regression models as the data set is nonlinear. These models consist of two neural network architectures, which are the Long Short-term Memory Network (LSTM), and the Fourier Series Neural Network (FSNN). In contrast, the remaining model is a non-neural network, the Random Forest. We are looking for a Mean Squared Error (MSE) score of below 0.05 as it would suggest that model provides a strong enough forecast to the actual result and the highest R-square (R2) score possible. The primary

focus is the MSE score, as the lower the value, the less skewed the results are, and below 0.05 is considered the most optimal. While the R2 score is the secondary focus, this suggests how well the features are related

A. Data-sets

- As mentioned earlier, the training and final prediction data come from three different data sets merged into one. The first data set, Automated Speed Enforcement Monthly Charges, comes from the City of Toronto's Open Data Portal, an open-source delivery tool [15]. This set contains the locations in a descriptive manner of an ASE historical site and the related number of tickets issued during each month of its active use as features we are interested in.
- The second data set, neighborhood-crime-rates –4326, came from the City of Toronto's Open Data Portal [16]. This set contains the various crime rates and yearly counts across all of Toronto's Neighborhoods. The features we considered are the most recent Auto Theft crime rate, Theft from Motor Vehicle crime rate since these rates have already been normalized by the most recent year of population prediction. In addition, we also took each neighborhood's geometry based on the given GPS coordinates.
- The third data set, Watch Your Speed (WYS) Stationary monthly summary, comes from the City of Toronto's Open Data Portal [17]. This set contained monthly summary data of the Watch Your Speed Program or WYSP signs. The data includes all recorded instances of vehicles going over the posted speed limit beyond the sign's initial threshold. In short, it contains the sign's historical location paired with its percentile speed observed at the sign location from the 5th percentile to the 95th percentile, the count of vehicles observed within an additional 5 km/h, from 5km/h over to 100 km/h in 5 km/h increments, and the

measure over the speed limit volumetric vehicle count. In addition, the GPS coordinates of the sign are given.

B. Preprocessing

The preprocessing is carried out in four main steps:

- Data aggregation,
- Location Determination,
- Location Pairing,
- Data Standardization

Data is aggregated to a monthly measure across all available data found with a yearly range of 2020 to 2022. For features pct xx, spd xx, and Sign Volume, the monthly median is calculated per unique WYS location. Crime rates of Auto Theft and Theft from Motor Vehicles are found first as the median yearly rates, then divided by 12 to get the monthly rates. The monthly median is taken per location instead of the average for the monthly ASE ticket charges, as some sites indicated outliers, such as the first month and last month.

Most features that indicated a location provide the correct standard of GPS coordinates; however, the sites provided by the ASE are in a descriptive form. To overcome this issue, we use the ArcGIS platform to request the appropriate GPS coordinates. The resulting location transformation would move the initial location to the closest intersection, i.e., Royalcrest Rd. Near Cabernet Circle == 43.75114900234675, - 79.60429802518539. Another issue arrives when dealing with the GPS coordinates involving the neighborhoods, as instead of a single point, it provides coordinates of a polygonal shape. Each district's center point was calculated to standardize this large set of numbers.

Location pairing is used to relate each data set to the other correctly. For the training file, the neighborhood center point and WYS location are paired up with the closest ASE location and each instance's related information. For the hotspot data set prediction, WYS instances are paired up with the closest neighborhood center points, as this final does not contain any target features. A geodesic algorithm is used to measure the distance between points [18].

The final step to preprocessing is the standardization of the data. It is conducted to prevent the interpretation of data with extreme outliers. The same Min Max Scalar is used with a min of 0 and a max of 1 on both data sets. This step significantly improved the stability of the machine-learning models.

The final training data set consists of 42 independent features with three target features, 298 instances, and 13,112 data points. The features used in the training data set can be found in Table 3.1.

Table 3.1: Definition of all of the Features in the Training Data Set

Feature	Definition
pct_05	5-th percentile speed observed at this location while the sign was installed monthly median
pct_10	10-th percentile speed observed at this location while the sign was installed monthly median
pct_15	15-th percentile speed observed at this location while the sign was installed monthly median
pct_20	20-th percentile speed observed at this location while the sign was installed monthly median
.	.
.	.
.	.
pct_95	95-th percentile speed observed at this location while the sign was installed monthly median
spd_05	Count of vehicles observed within this 5 km/hr speed bin, from x km/hr to x + 4 km/hr inclusive. 5-9 monthly median
spd_10	Count of vehicles observed within this 5 km/hr speed bin, from x km/hr to x + 4 km/hr inclusive. 10-14 monthly median
spd_15	Count of vehicles observed within this 5 km/hr speed bin, from x km/hr to x + 4 km/hr inclusive. 15-19 monthly median
spd_20	Count of vehicles observed within this 5 km/hr speed bin, from x km/hr to x + 4 km/hr inclusive. 20-24 monthly median
.	.
.	.
.	.
spd_100	Count of vehicles observed above 100km/hr monthly median
sign_volume	Total number of vehicles observed for monthly median
AutoTheft_Rate_monthly	Crime rate of auto theft monthly median
TheftfromMotorVehicle_Rate_monthly	Crime rate of theft from vehicle monthly median
Monthly_Median_Tickets	Median tickets charged per month
Sensor_Latitude	Latitude coordinates of ASE sensor
Sensor_Longitude	Longitude coordinates of ASE sensor

The final three features represent the targets, which will be conducted under supervised learning. For each regression model used, three outcome models will be created for each target feature, totaling 9 models for this chapter.

C. Regressions

This chapter suggests three different approaches to training and solving this problem. The first approach is a non-neural network, and the last 2 are neural networks. These models would target output features Sensor Latitude, Sensor Longitude, and Monthly Median Tickets.

This approach with a non-neural network machine learning model uses the Random Forest method. Such a method deals well with the limited number of features found in the data set. At the same time, the choice of Random Forest assists in preventing possible overfitting of the training model.

The Random Forest Regression formula is given by:

$$\hat{y}_i = \frac{1}{N_{\text{trees}}} \sum_{j=1}^{N_{\text{trees}}} f(x_i; \theta_j) \qquad \textit{Equation 1}$$

The first approach with a neural network machine learning model involves using the FSNN. Fourier Series Neural Network is chosen as it provides high efficiency with nonlinear regression data [16].

The FSNN formula is given by:

$$f(x) = \sum_{k=1}^{K} (a_k \cos(k\omega x) + b_k \sin(k\omega x))$$

Equation 2

The second approach with a neural network machine learning model involves using the LSTM. A Long Short-Term Memory architecture is chosen for this approach as it is used in deep learning [16].

D. Final Method

Once all 9 feature models are created from the three regression approaches, they would run against the unused portion of the data set, the final prediction of hotspots. With this, the most significant latitude and longitude pairs based on monthly ticket predictions would be graphed on a map of the City of Toronto in increasing ranges. These graphs would display the top 10 at once and all locations for each regression model with the top 100 to visually determine how each would make its prediction.

3.4 Evaluation

Each regression model is trained thrice with the same independent features, once per target feature. These target features are Monthly Median Tickets, Sensor Latitude, and Sensor Longitude. A single output approach is chosen as the goal is to achieve the best possible results for each target and then pair them together for the final predictions.

A. Random Forest

There was room for hyperparameter tuning for modeling the Tickets coordinates. The best results come from a testing range of 0-100 for max tree depth and max-leaf nodes, where 11 and 53 are respectfully selected. This has resulted in an MSE score of 0.010128 and a R2 score of 0.016278. This can be seen in Figure 3.1 demonstrating the predicted results, actual or true results, where the red line would be the perfect prediction. However, the low positive R2 score indicates a poor correlation between the features

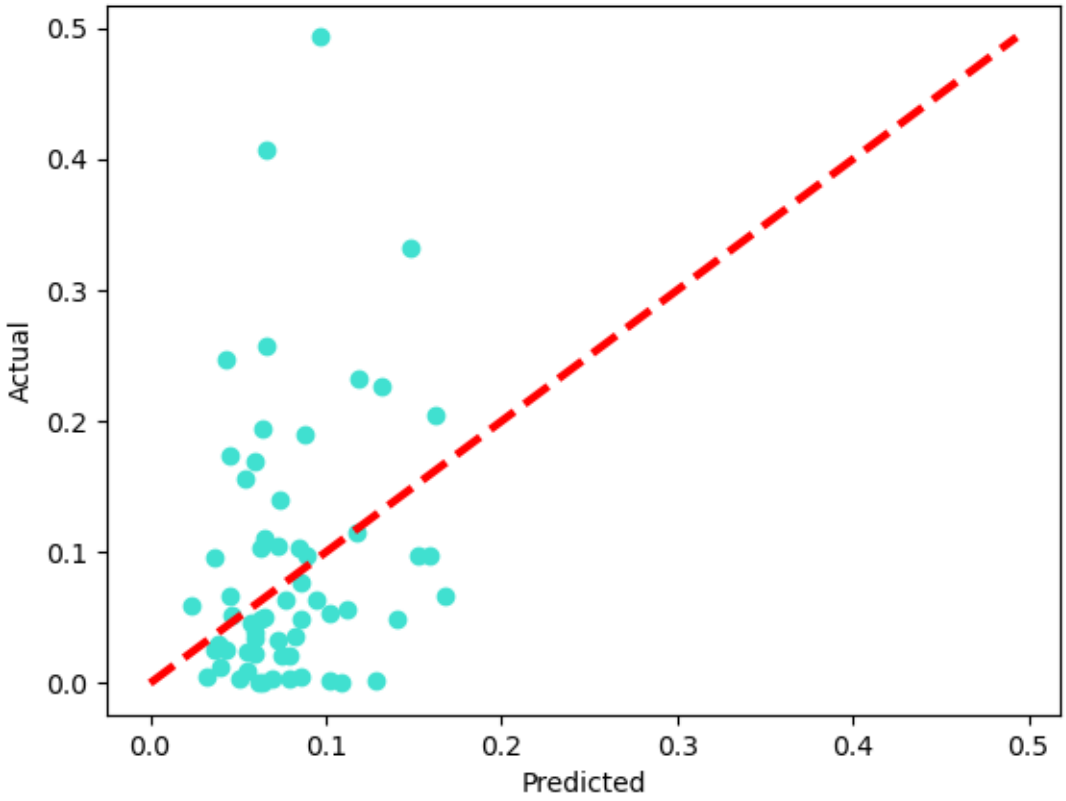

Figure 3.1: Predicated vs Actual results for ticket model from Random Forest Regression Approach.

There was also room for hyperparameter tuning for modeling the latitude coordinates. The best results come from a testing range of 0-100 for max tree depth and max-leaf nodes, where 16 and 90 are respectfully selected. This has resulted in an MSE score of 0.019701 and an R2 score of 0.610186. This can be seen in Figure 3.2 demonstrating the predicted results, actual or true results, where the red line would be the perfect prediction. However, the positive R2 score indicates a sound correlation between the features.

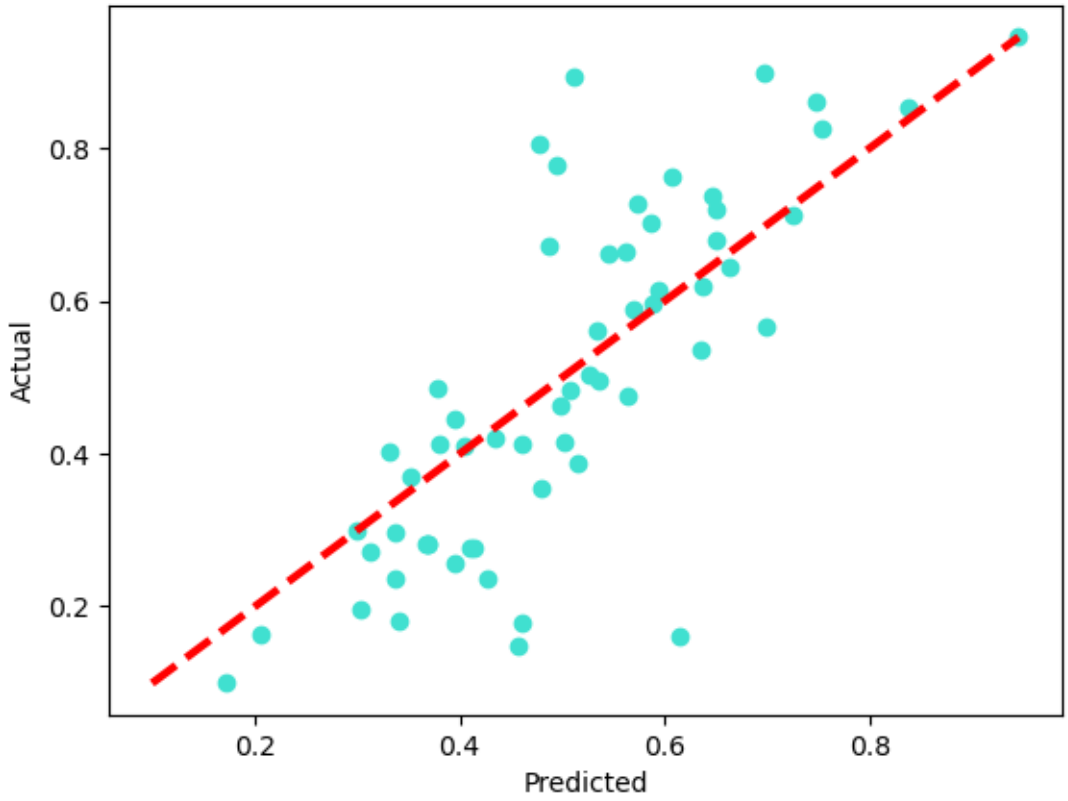

Figure 3.2: Predicated vs Actual results for latitude model from Random Forest Regression Approach

We also performed tuning for hyperparameters for modeling the longitude coordinates. The best results come from a testing range of 0-100 for max tree depth and max-leaf nodes,

where 14 and 92 are respectfully selected. This has resulted in an MSE score of 0.023591 and an R2 score of 0.552474. This can be seen in Figure 3.3 demonstrating the predicted results, actual or true results, where the red line would be the perfect prediction. However, the positive R2 score indicates a healthy correlation between the features.

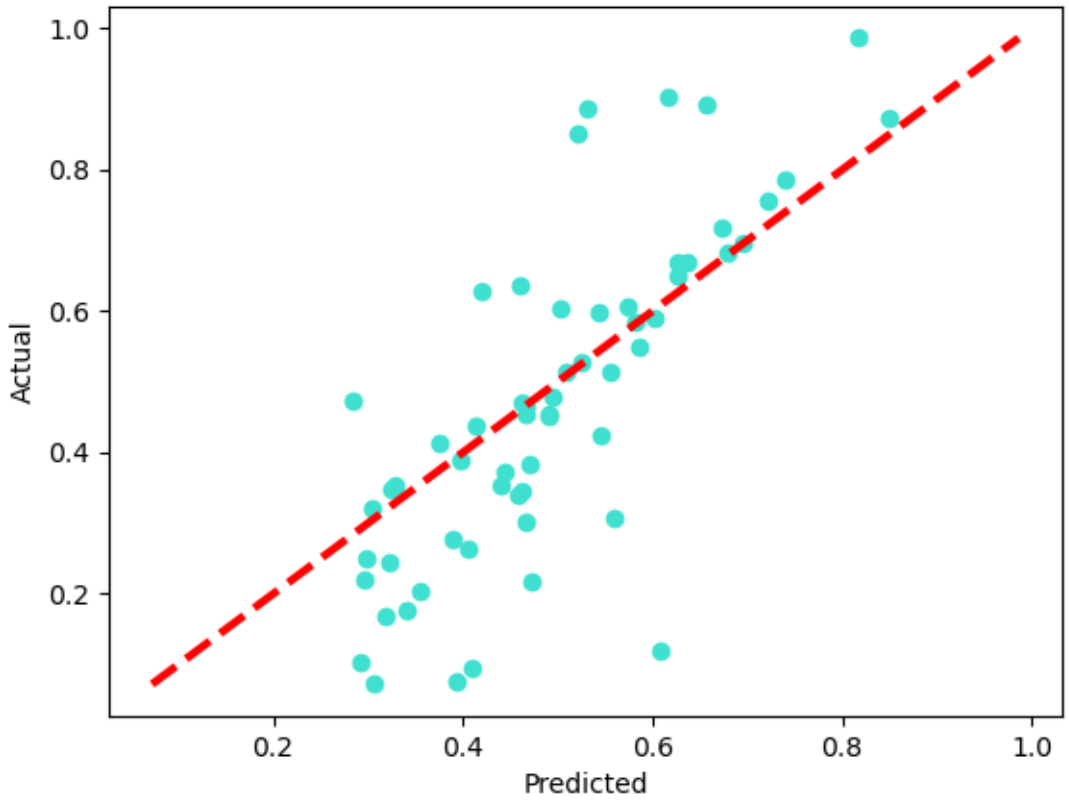

Figure 3.3: Predicated vs Actual results for longitude model from Random Forest Regression Approach

B. Long Short-Term Memory

Each model trained has an architecture built with two hidden layers, each with 100 nodes, with the swish activation function. This function is selected as it provides better results than the

rectified linear unit activation function [19]. Additionally, with a dropout of 0.2 and return sequences.

There was room for hyperparameters to tune for modeling the Tickets. The best results come from a testing range of 0-100 for batch size and epoch, where 5 and 15 are respectfully selected. This has resulted in an MSE score of 0.010293 and an R2 score of 0.000225. This can be seen in Figure 3.4 demonstrating the predicted results, actual or true results, where the red line would be the perfect prediction. However, the low positive R2 score indicates a poor correlation between the features.

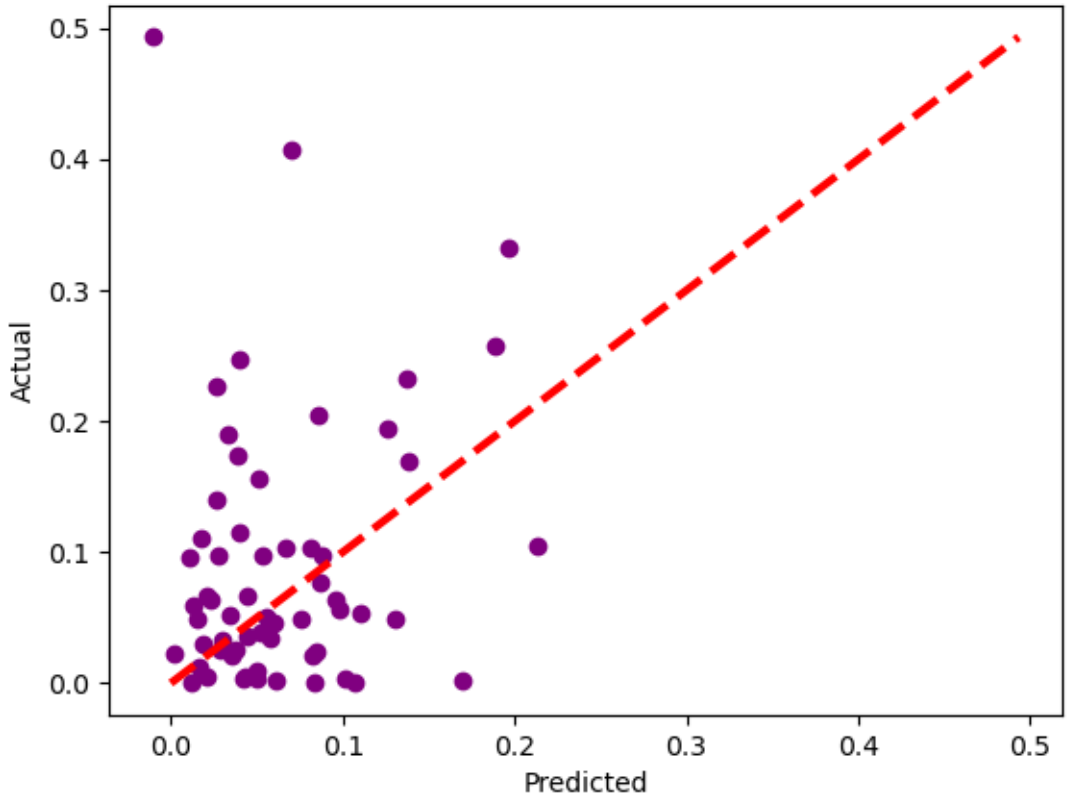

Figure 3.4: Predicted vs Actual results for Ticket model from LSTM Regression Approach

There was also room for hyperparameters to tune for modeling the latitude coordinates. The best results come from a testing range of 0-100 for batch size and epoch, where 10 and 60 are respectfully selected. This has resulted in an MSE score of 0.028736 and an R2 score of 0.431404. This can be seen in Figure 3.5 demonstrating the predicted results, actual or true results, where the red line would be the perfect prediction. However, the moderate positive R2 score indicates a good correlation between the features.

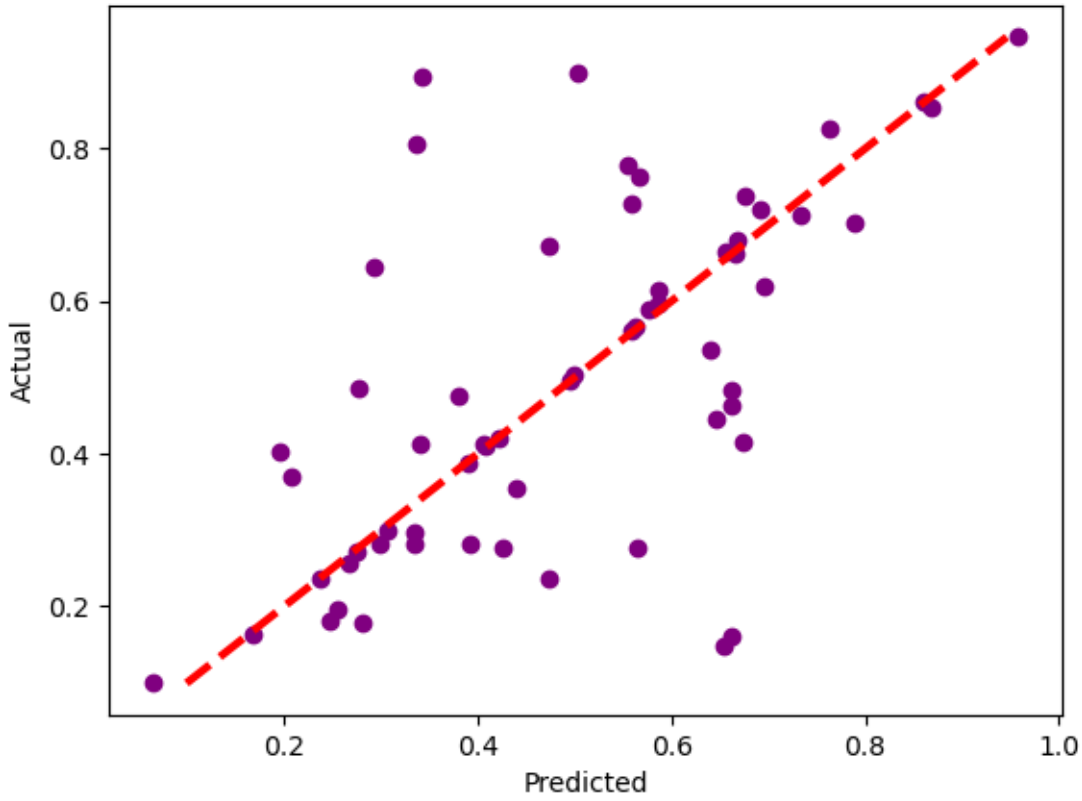

Figure 3.5: Predicted vs Actual results for Latitude model from LSTM Regression Approach

We have also performed hyperparameter tuning for modeling the longitude coordinates. The best results come from a testing range of 0-100 for batch size and epoch, where 0 and 75 are respectfully selected. This has resulted in an MSE score of 0.038648 and an R2 score of

0.266857. This can be seen in Figure 3.6 demonstrating the predicted results, actual or true results, where the red line would be the perfect prediction. However, the low positive R2 score indicates a poor correlation between the features.

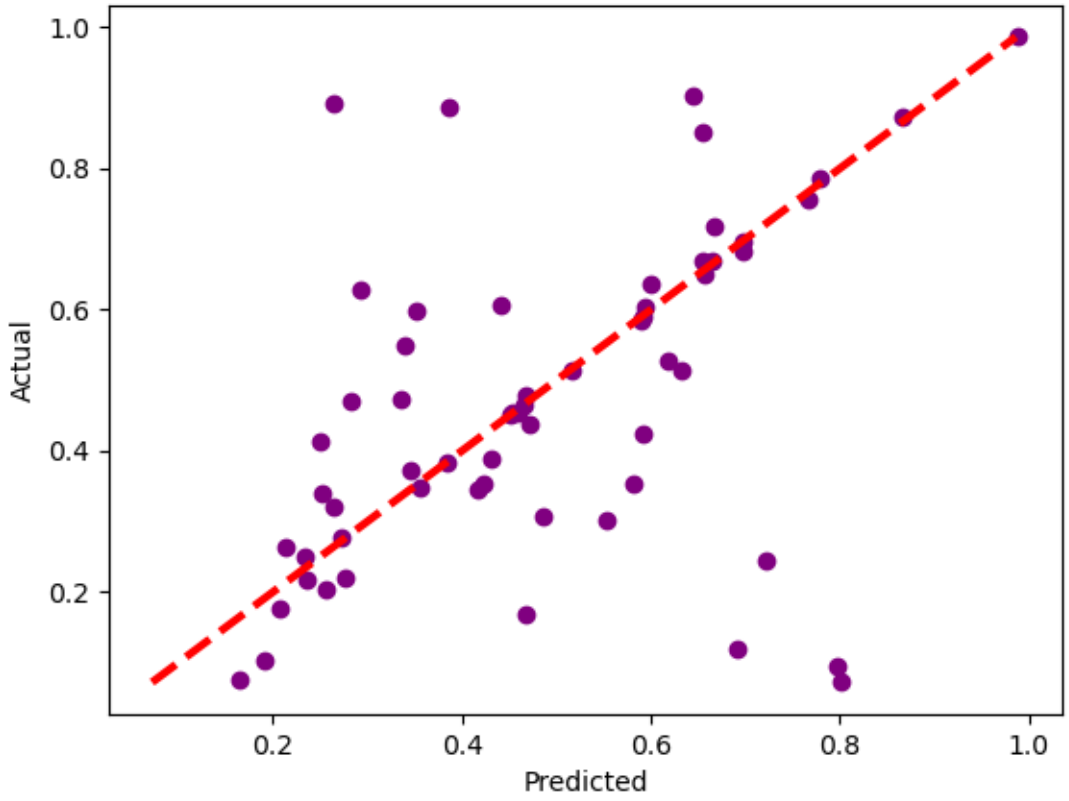

Figure 3.6: Predicted vs Actual results for Longitude model from LSTM Regression Approach

C. Fourier Series Neural Networks

There was room for hyperparameters to tune for modeling the Tickets. The best results come from a testing range of 0-100 for batch size and epoch, where 50 and 60 are respectfully selected. This has resulted in an MSE score of 0.012508 and an R2 score of -0.21492. This can be seen in Figure 3.7 demonstrating the predicted results, actual or true results, where the red line

would be the perfect prediction. However, the low negative R2 score indicates a poor correlation between the features.

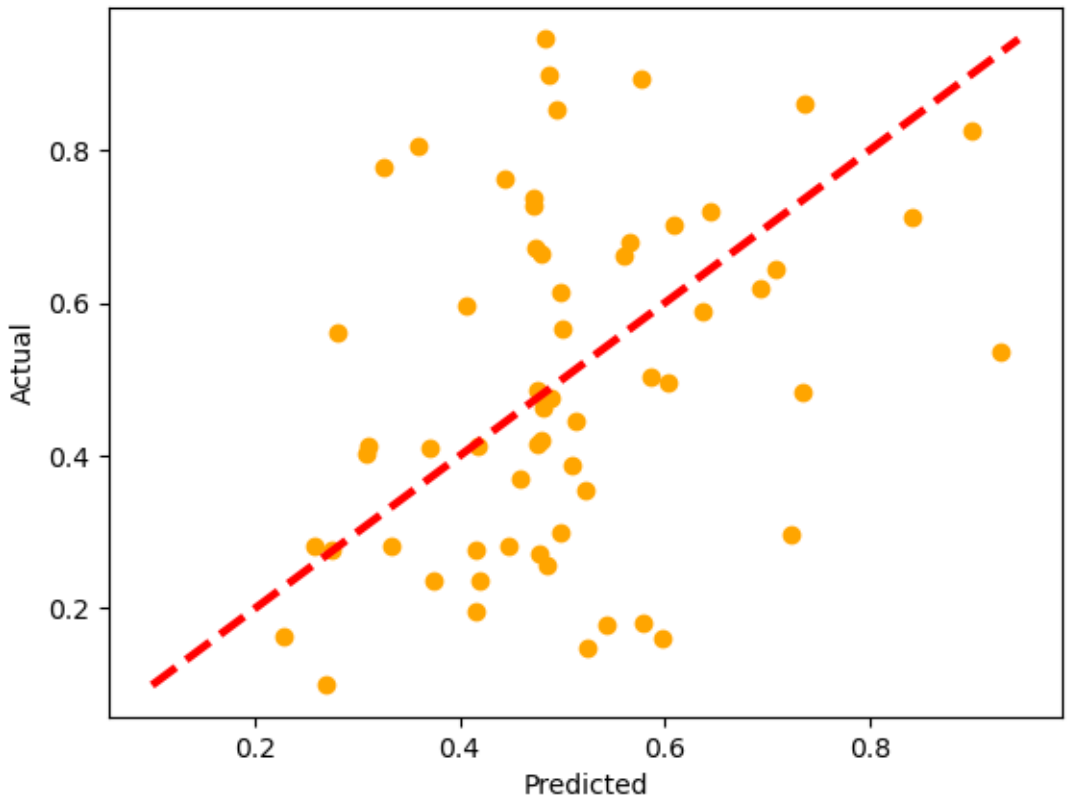

Figure 3.7: Predicted vs Actual results for Ticket model from FSNN Regression Approach

There was room for hyperparameters to tune for modeling the latitude coordinates. The best results come from a testing range of 0-100 for batch size and epoch, where 20 and 45 are respectfully selected. This has resulted in an MSE score of 0.021548 and an R2 score of 0.573629. This can be seen in Figure 3.8 demonstrating the predicted results, actual or true results, where the red line would be the perfect prediction. However, the positive R2 score indicates a sound correlation between the features.

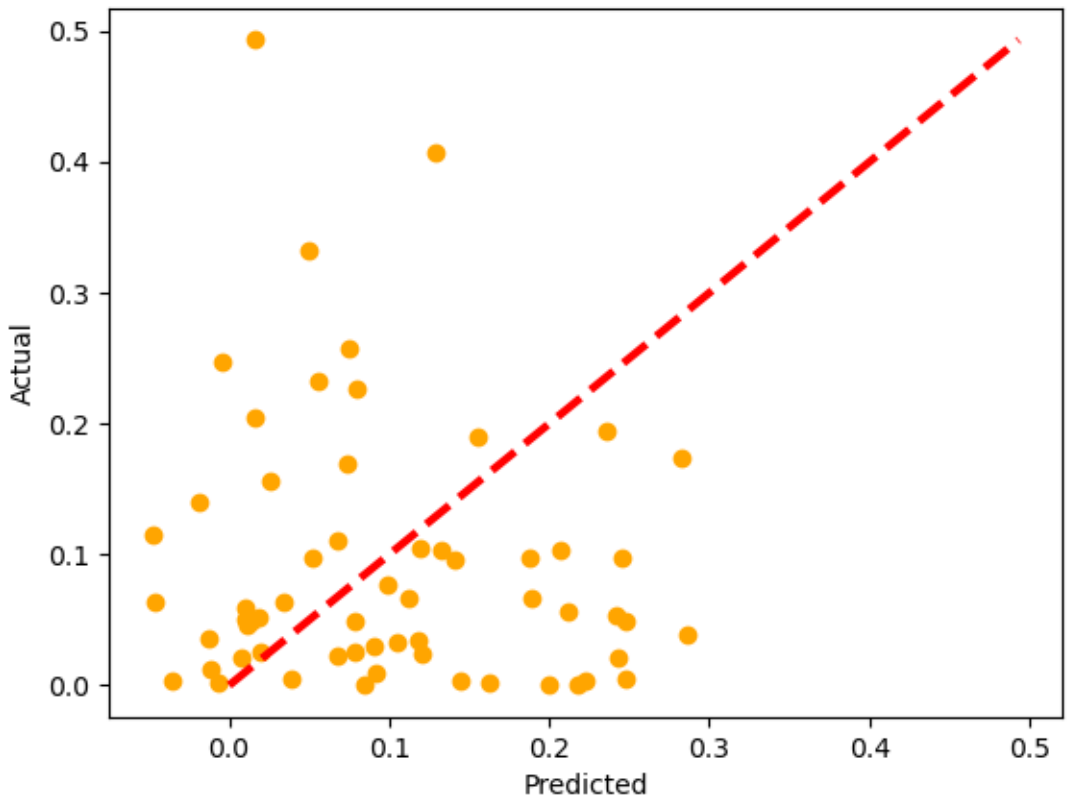

Figure 3.8: Predicted vs Actual results for Latitude model from FSNN Regression Approach

There was room for hyperparameters to tune for modeling the longitude coordinates. The best results come from a testing range of 0-100 for batch size and epoch, where 5 and 60 are respectfully selected. This has resulted in an MSE score of 0.031154 and an R2 score of 0.409014. This can be seen in Figure 3.9 demonstrating the predicted results vs. actual or true results, where the red line would be the perfect prediction. However, the good positive R2 score indicates a good correlation between the features.

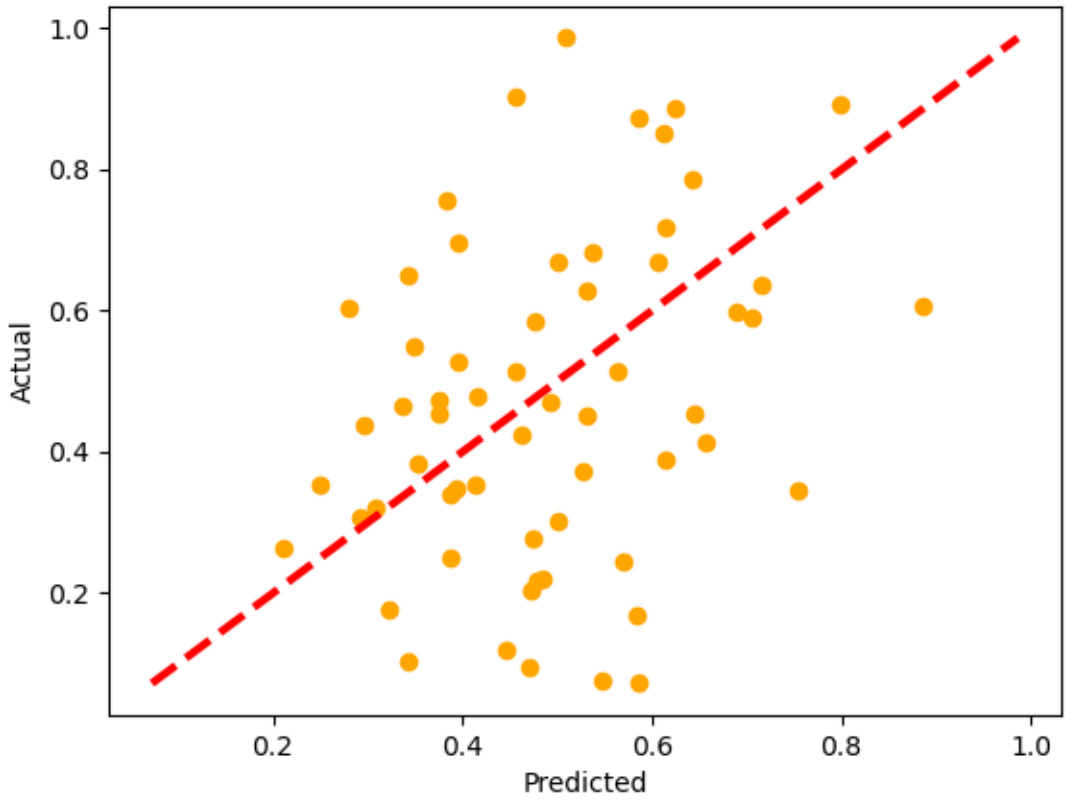

Figure 3.9: Predicted vs Actual results for Longitude model from FSNN Regression Approach

D. Comparison

Figure 3.10 shows that all machine-learning approaches have achieved a score of 0.05 or less, which is a good assessment. The neural network models have the lowest scores for the target features across the board, while the Random Forest performed the best. Overall, all three machine learning approaches have achieved ideal scores across each of the target features, however, scores are marginally different.

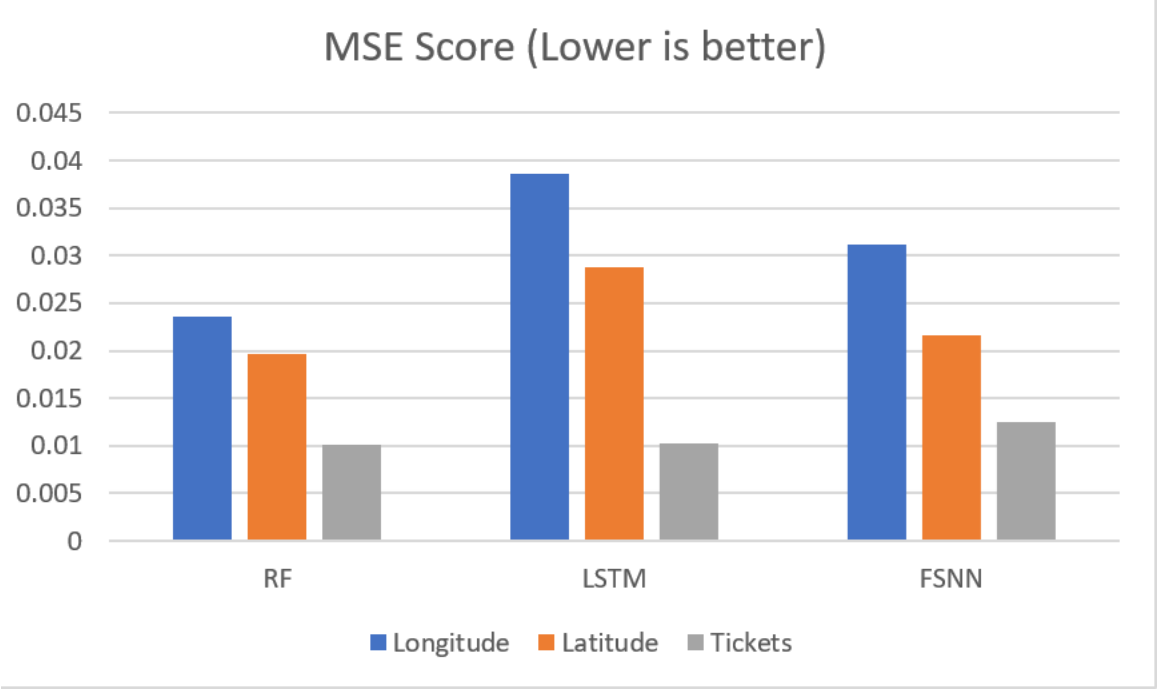

Figure 3.10: MSE Scores

Figure 3.11 shows that the Random Forest and the FSNN have performed well for Longitude and Latitude target features. Only the FSNN has achieved the highest score for the Tickets target feature achieved a mediocre score, while the other approaches have not performed the best, especially LSTM achieving a negative score. There is a greater emphasis on the ticket target feature as it can provide better ASE locations based on the highest predicted ticket count. Overall, the FSNN has performed the best when considering all three target features.

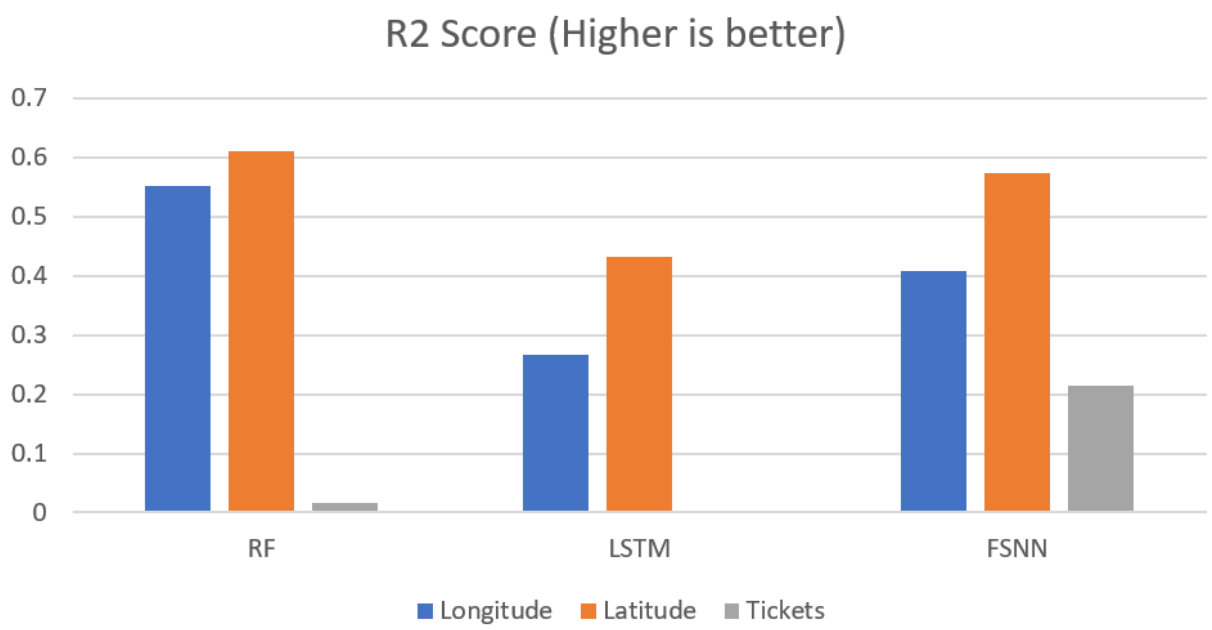

Figure 3.11: R2 Scores

This demonstrates that FSNN outperforms all other approaches when considering Figures 3.10 and 3.11. This is seen by all MSE scores below 0.05 and all R2 scores with at least an acceptable score above 0.2.

E. Final Prediction

In Figure 3.12, we present the top 10 locations based on the most significant value of the Monthly Median Ticket counts. Random Forest is in turquoise, LSTM is in purple, FSNN is in orange, and green is the original location of the ASE sensors. Upon initial analysis of the map, there appears to be some overfitting of the LSTM hotspots seen in the downtown core and to the east. However, the other approaches favor locations of high traffic, such as schools and increased

pedestrian traffic locations.

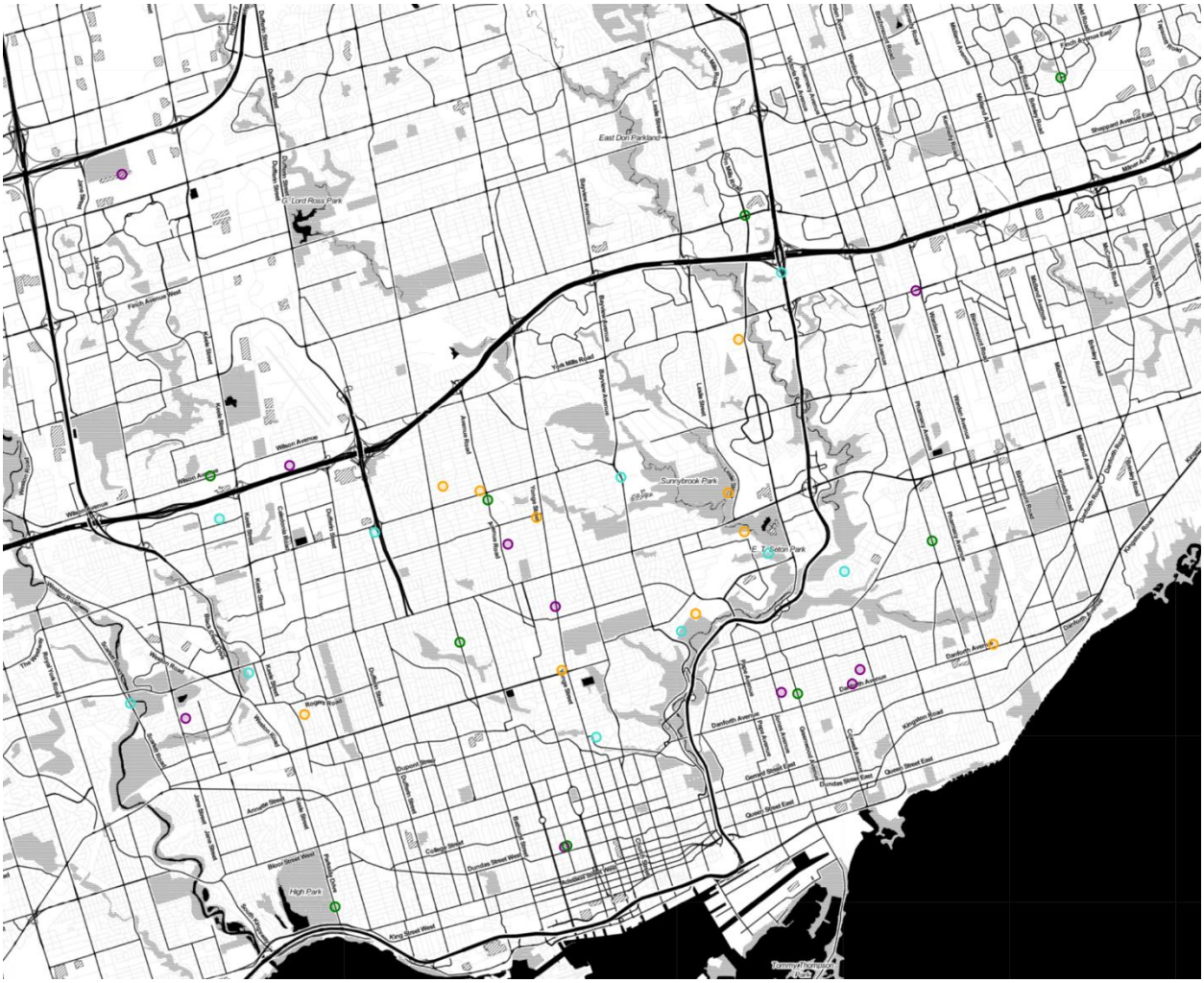

Figure 3.12 Final predictions of the 10 ASE hotspots with historical locations

In Figure 3.13, we present the top 100 locations of the non neural network approach provided in this chapter based on the most significant values of the Monthly Median Tickets counts. The turquoise points represent the Random Forest approach. It is clustered in the center with a bias towards the east of the city, with some minor overfitting seen with the historical points.

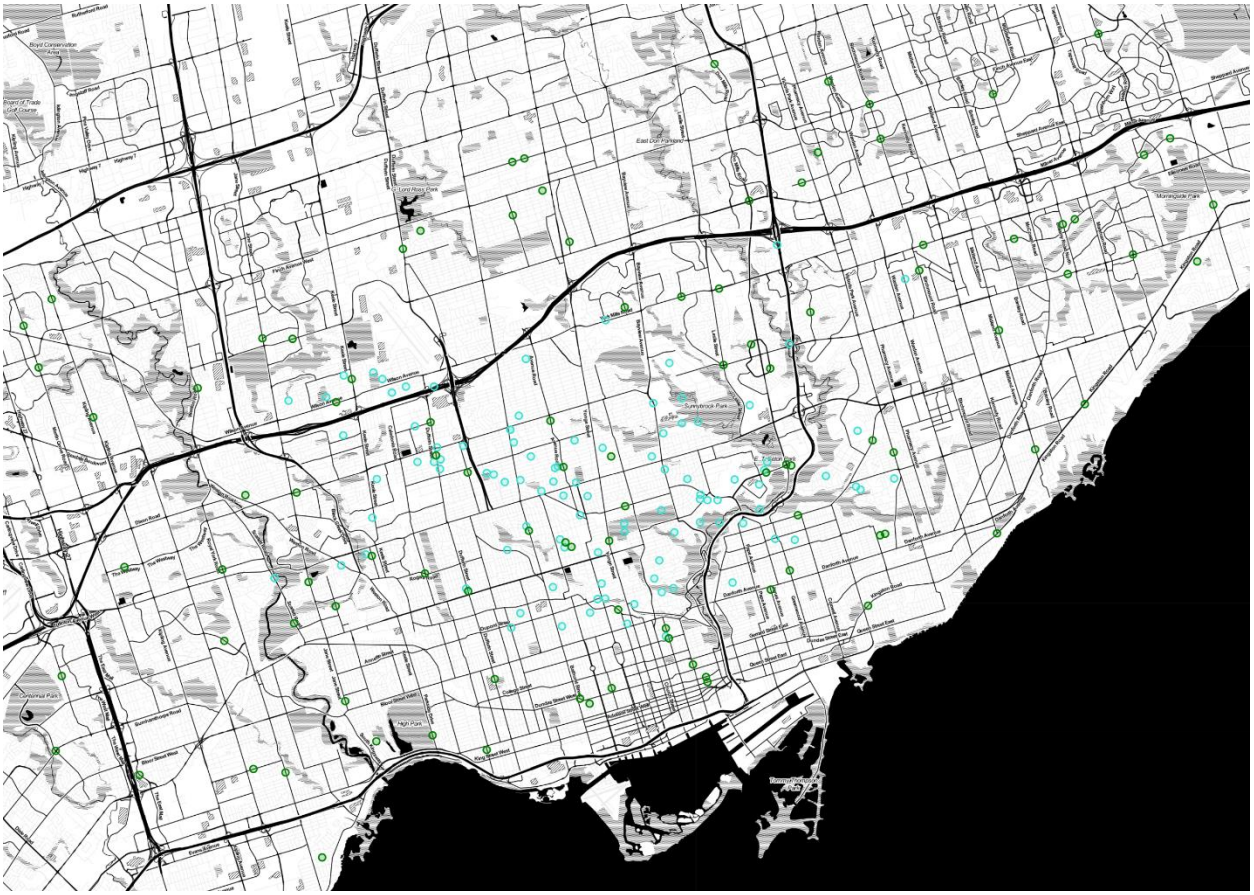

Figure 3.13 Final predictions of the top 100 ASE hotspots for Random Forest and Historical

In Figure 3.14, we present the top 100 locations of the neural network approaches provided in this chapter based on the most significant values of Monthly Median Tickets counts. The purple represents the LSTM, and the orange represents FSNN. It is noticeable that the points provided by the LSTM are out of the bounds of the City of Toronto. This suggests that the more points added, the more underfitting bias appears. However, FSNN demonstrates clustering towards the east with an appropriate spread across the city. As for LSTM, it demonstrates an additional overfitting bias in some locations that can be seen when compared to historical points. Indicating top locations have an overfitting bias and on the lower end of the list, there is an underfitting bias, all the while FSNN indicates neither.

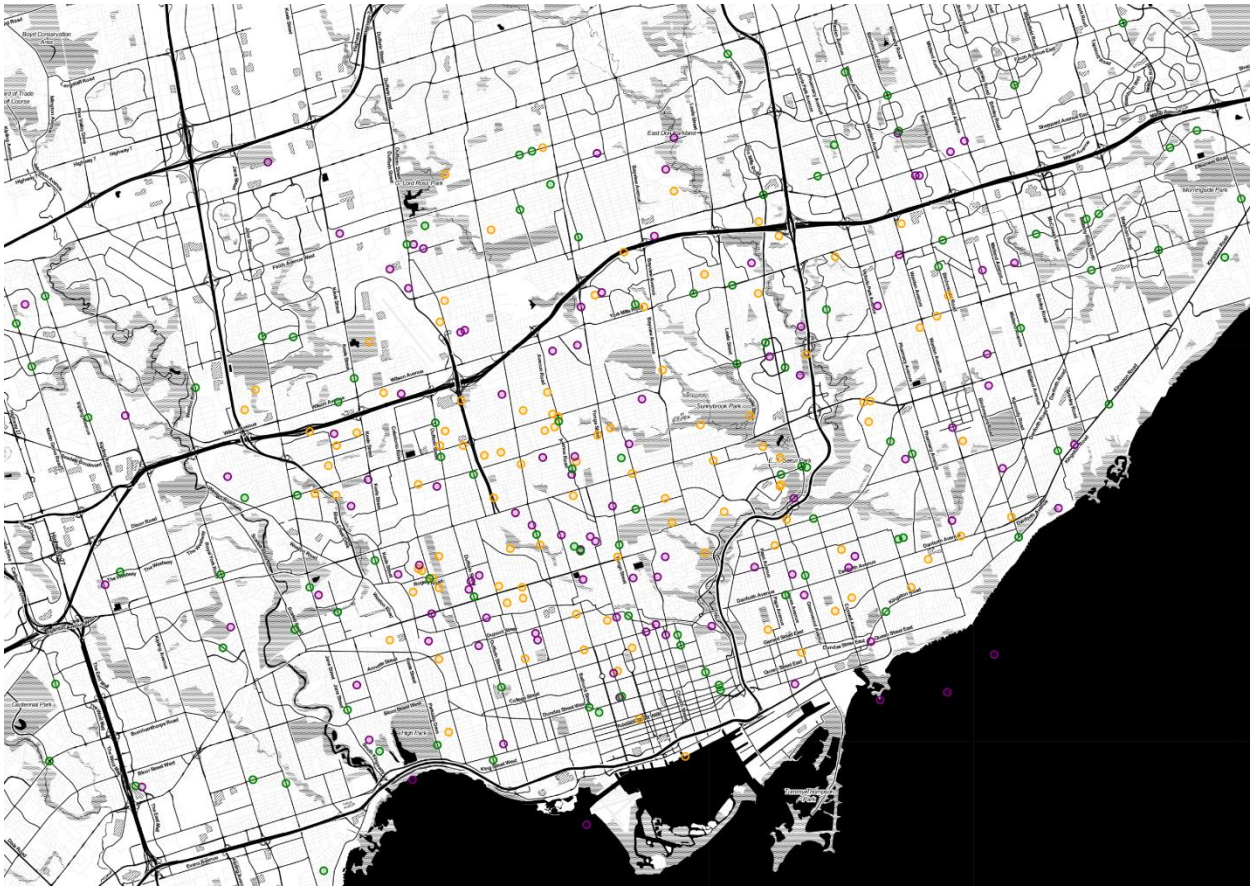

Figure 3.14: Final predication of the top 100 ASE hotspots for FSNN, LSTM, and Historical

F. Pre kPCA Ablation Analysis

An ablation analysis is pivotal for understanding how individual features within a dataset influence the predictions made by a machine learning model. For in-depth interpretability, we employ the SHapley Additive exPlanations (SHAP) method, which quantifies the impact of each feature. In this study, we apply SHAP to analyze three previously mentioned machine learning models to assess the significance of their features. This chapter focuses on a dataset to which kernel Principal Component Analysis (kPCA) has not yet been applied; the integration of kPCA is detailed in the following chapter.

The first SHAP summary plot analyzed is for the FSNN Ticket model which can been seen in Figure 3.15. It indicates that the top 20 features had minimal influence on the model's predictions, indicative by the trend of the clustering around the center point. In addition, the points on the far edges of the scale, only have a value of at most around -0.002 to 0.002, representative of its low R2 score.

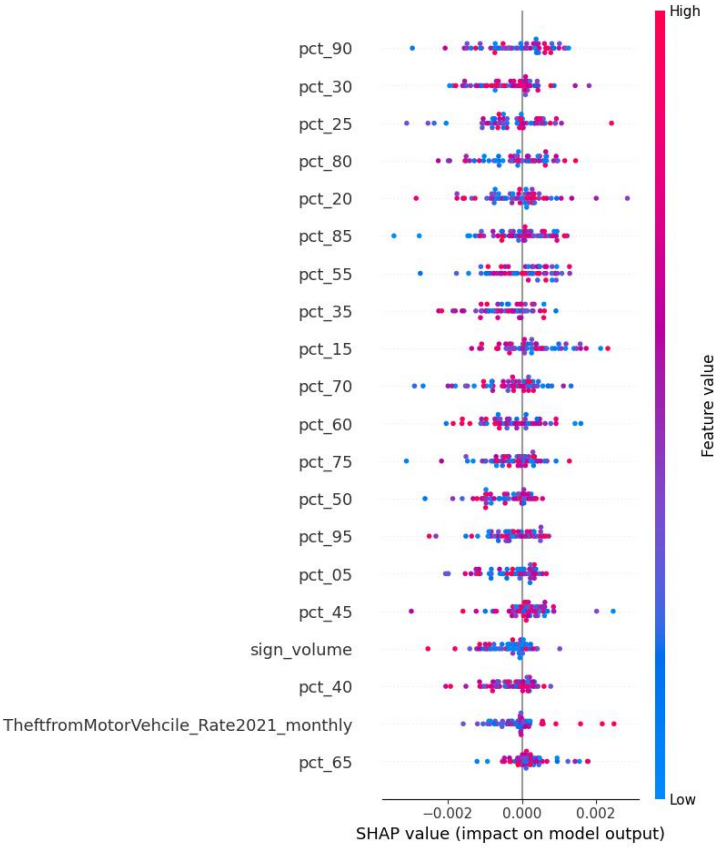

Figure 3.15: FSNN Ticket model SHAP Summary Plot

The second SHAP summary plot analyzed is for the RF Latitude model which can been seen in Figure 3.16. It indicates that the top 6 features had some influence on the model's predictions, particularly with the top 2. These features are in relation to criminal statistics which

suggest that crime in itself is a heavy influence for this model. In addition, the points on the far edges of the scale, have a value of at most around -0.1 to 0.2, representative of its R2 score.

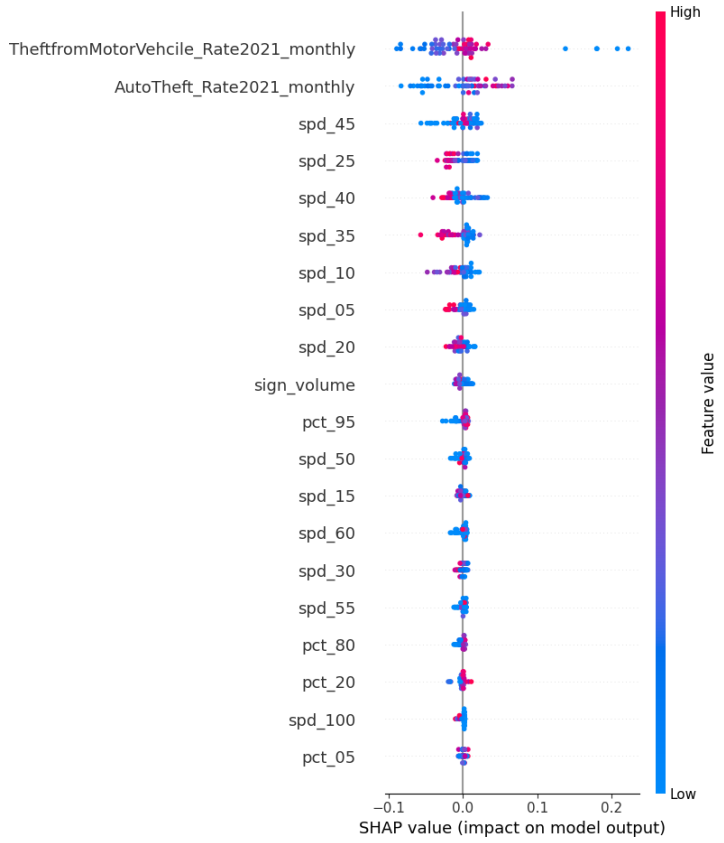

Figure 3.16: RF Latitude model SHAP Summary Plot

The final SHAP summary plot analyzed is for the LSTM Longitude model which can be seen in Figure 3.17. It indicates that the top 6 features had some influence on the model's predictions, particularly with the top 3. These features are in relation to criminal statistics which suggest that crime in itself is a heavy influence for this model, with the exception of its top feature which indicates relative volumetric vehicle count. In addition, the points on the far edges of the scale, have a value of at most around -0.2 to 0.1, representative of its R2 score.

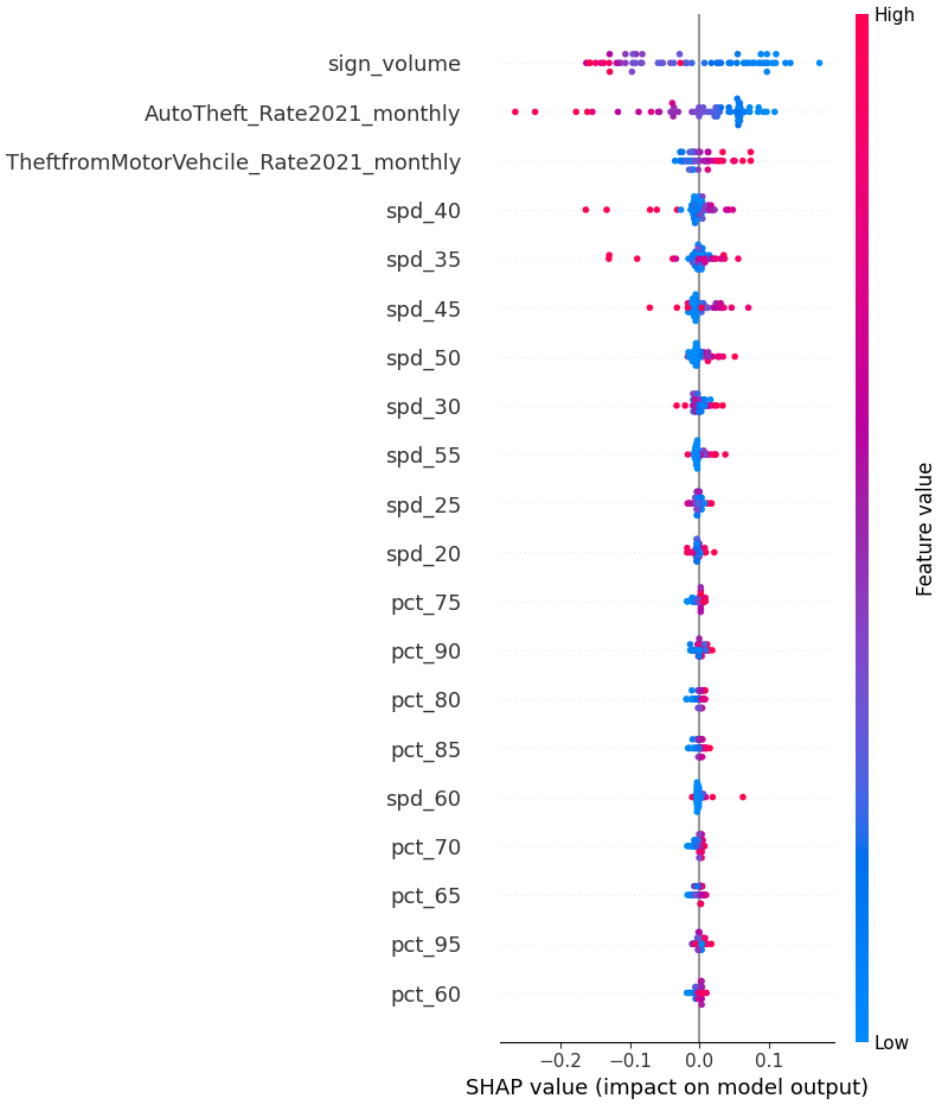

Figure 3.17: LSTM Longitude model SHAP Summary Plot

More pre kPCA SHAP Summary Plots can be found in the Appendix.

3.5 Summary

As populations grow, more automobiles are being purchased since people aim for comfort. However, this can lead to serious traffic incidents and violations. As a result, people's awareness is hindered, expenses are increased, properties are damaged, and people are either

injured or killed. These circumstances necessitate developing traffic violation prediction and detection systems to automate traffic regulations and eliminate unawareness within the human population [20].

Incorporating intelligent transportation systems (ITS) applications and smart technology techniques such as machine learning can be beneficial in resolving traffic concerns and improving road safety in Toronto. By leveraging data from various sources such as traffic sensors, cameras, and GPS devices, ML algorithms can analyze traffic patterns, identify congestion, predict traffic volume, and detect traffic violations.

This thesis demonstrated a generalized novel approach to ASE hotspot prediction, providing similar placements based on the current descriptive investigative method. Three approaches were used; Random Forest, LSTM approach, and FSNN approaches. The metric results indicated that the FSNN outperformed the rest. This is more evident visually, looking at a map of the City of Toronto with the ASE hotspot predictions. Undoubtedly, more research is needed, and more models are to be evaluated to fully identify the best ML-enabled traffic crime prediction approach.

Consequently, not only can we decrease the adverse impacts of traffic across the streets of Toronto, but we can also enhance the overall protection standards and road safety measures for citizens in the city and other cities. In the following chapters, we will explore an expanded selection of machine learning models with an addition to a principal component analysis to determine if the dimension of the training data can be shrunk to improve the performance of the models.

4. Enhancing City Traffic Safety through Optimized Machine Learning Models for Sensor Placement for Intelligent Transportation Systems

4.1 Introduction

Through extensive feature engineering techniques, we extract highly relevant features with a confidence level of 99%, reducing overall computational load. Subsequently, we explore various machine learning (ML) models to predict the latitude and longitude of potential new hotspots, sorting them by predicted ticket count through comprehensive hyperparameter tuning. The contributions are summarized as follows.

- Here, new models are explored and old models such as the Fourier Series Neural Network (FSNN), Long Short-term Memory (LSTM), and Random Forest (RF) are optimized further [21]. On top of this we developed two additional models for each regression learning approach, ensuring all undertake an in depth hyperparameter tuning while incorporating the architecture design as one of the lists of parameters.
- We formulate the problem of predictive placement based on historical ticket measurements from the Automated Speed Enforcement locations, Toronto police crime statistics, and Watch Your Speed sign historical data.
- We develop a smart framework that contributes to the reduction of model processing time and prediction duration without compromising accuracy, ensuring efficient performance.

- We develop a graphical user interface (GUI) that government officials, law enforcement agencies, and researchers, regardless of their technical expertise, can use as a tool for uploading data and determining areas of high and low traffic crime rates
- We implement a rudimental load balancer with multi threading and parallel processing to the primary computational server to reduce overall working time.
- We implement feature engineering with Lasso Regularization (L1), Ridge Regularization (L2), Min-Max scaling, a kPCA, geographical encoding, and a custom feature construction to decrease computational resource demands and to improve CPU times of the hyper-parameter tuning.
- We propose a comprehensive comparative analysis among different models including the FSNN, LSTM, Random Forest, Feed Forward Neural Network (FNN), and a Support Vector Machine (SVM) to determine which method provides the most efficient and accurate method of placement.

4.2 Problem Statement

In the quest to refine urban safety and traffic management, this study builds upon the preceding work to further enhance the precision and efficacy of sensor placements for detecting traffic violations. The initial research laid a strong foundation, employing machine learning techniques to predict potential hotspots for traffic violations, thereby contributing significantly to the intelligent transportation system's domain. However, despite these advancements, the dynamic and complex nature of urban traffic, coupled with the ever-evolving urban environments, necessitates continuous optimization and refinement of the predictive models.

Hence, the central objective of this phase of research is to critically analyze the outcomes of the previous efforts, identify any limitations or areas of improvement, and leverage the latest advancements in machine learning, data processing, and algorithm optimization. The problem statement evolves to focus on enhancing the predictive accuracy, computational efficiency, and practical applicability of these models. This involves a meticulous examination of the previously utilized datasets, models, and feature engineering techniques, with a particular emphasis on improving model robustness against overfitting and underfitting, and exploring new or refined machine learning algorithms that could offer superior performance.

Moreover, recognizing the importance of real-world applicability, this study aims to ensure that the refined predictive models are not only more accurate and efficient but also adaptable to various urban settings and scalable across different jurisdictions. This involves addressing the challenges related to data availability, differing urban layouts, and legal and ethical considerations pertinent to the deployment of surveillance technologies in public spaces.

Ultimately, this refined problem statement guides the research towards developing a more sophisticated, versatile, and user-friendly system for predicting and preventing traffic violations through optimized sensor placement. The goal is to provide city planners, law enforcement agencies, and policymakers with a robust tool that supports proactive measures in enhancing road safety and traffic management, thereby contributing to the broader objectives of public safety and urban sustainability violations and crimes can have a negative impact on civilians and the community as a whole. Speeding, reckless driving, and running past red lights increase accident risks, resulting in injuries, fatalities, and property damage. Traffic violations can also cause congestion, leading to increased travel times and delays for commuters and environmental pollution. Moreover, traffic offenses result in fines, points on a driver's license, and even jail

time in some cases. Drivers who commit traffic crimes may also see their insurance rates increase, making driving more expensive [14]. Therefore, aiming for a reduction of traffic violations would carry a positive impact on all aspects of society.

Efforts to curb the traffic crime rate are facing several challenges and limitations. These include the availability and suitability of data, choosing the best model/algorithms, determining the end goal (detection vs. prediction), and applicability in real life. Therefore, we extend the previous work [21]. to explore four areas of interest. This includes the addition of two new machine learning approaches, the in depth hyperparameter tuning for each with consideration of their own architecture, an extensive application of feature engineering, and design of GUI to stream line the user experience with a black box concealment approach.

4.3 Methodology

To optimize and refine the previous work on predicting sensor placements for traffic violations using machine learning, the methodology evolves to focus on advanced data integration, enhanced feature engineering, and sophisticated model development. We aim to build upon the initial findings, addressing limitations, and incorporating new insights to significantly improve the predictive accuracy and applicability of the models.

Feature engineering takes a central role in the refined methodology. We delve deeper into selecting the most predictive features using sophisticated techniques, and we explore the construction of new features that can reveal complex patterns in the data. Additionally, we apply advanced dimensionality reduction techniques to streamline these models without losing critical information, aiming for a balance between model complexity and computational efficiency.

Model development is approached with a focus on refinement and exploration. We revisit these initial models, including the Fourier Series Neural Network (FSNN), Long Short-term Memory (LSTM), and Random Forest, applying extensive hyperparameter tuning to optimize their performance. Furthermore, we explore additional models that may offer improvements in prediction accuracy or efficiency. These models are the Support Vector Machines (SVM) and the Feed Forward Neural Network (FNN). Each model is rigorously evaluated through a comprehensive set of metrics, going beyond traditional measures to include those specific to the challenges of predicting geospatial hotspots for traffic violations.

A significant part of the methodology is the implementation of a robust evaluation framework. This framework not only assesses model performance on a static dataset but also simulates real-world scenarios to understand how the models' predictions would perform in dynamic urban environments. This approach allows us to gauge the practical applicability of the models and make necessary adjustments to ensure they can be effectively used by urban planners and policymakers.

In summary, the optimized methodology is designed to build on the successes and learnings of the previous work, employing a more sophisticated approach to data processing, feature engineering, and model development. This iterative process is aimed at producing highly accurate, efficient, and adaptable models for predictive sensor placement, contributing to safer and more efficient urban traffic systems.

A. Graphical User Interface

Seen in Figure 4.1 is a prototype graphical user interface (GUI) that handles the demands of the interacting user within set limits, and provides visual feedback. These come in forms of a

progress bar and the final predicted results in a dynamic scalable map that can be manipulated by the user for an in-depth examination of the results. The user would be able to select the desired excel or csv files for the system to process. Once the files are chosen, the user starts the system, and select a desired graph based on either the best combination of models, only neural networks, or non neural networks.

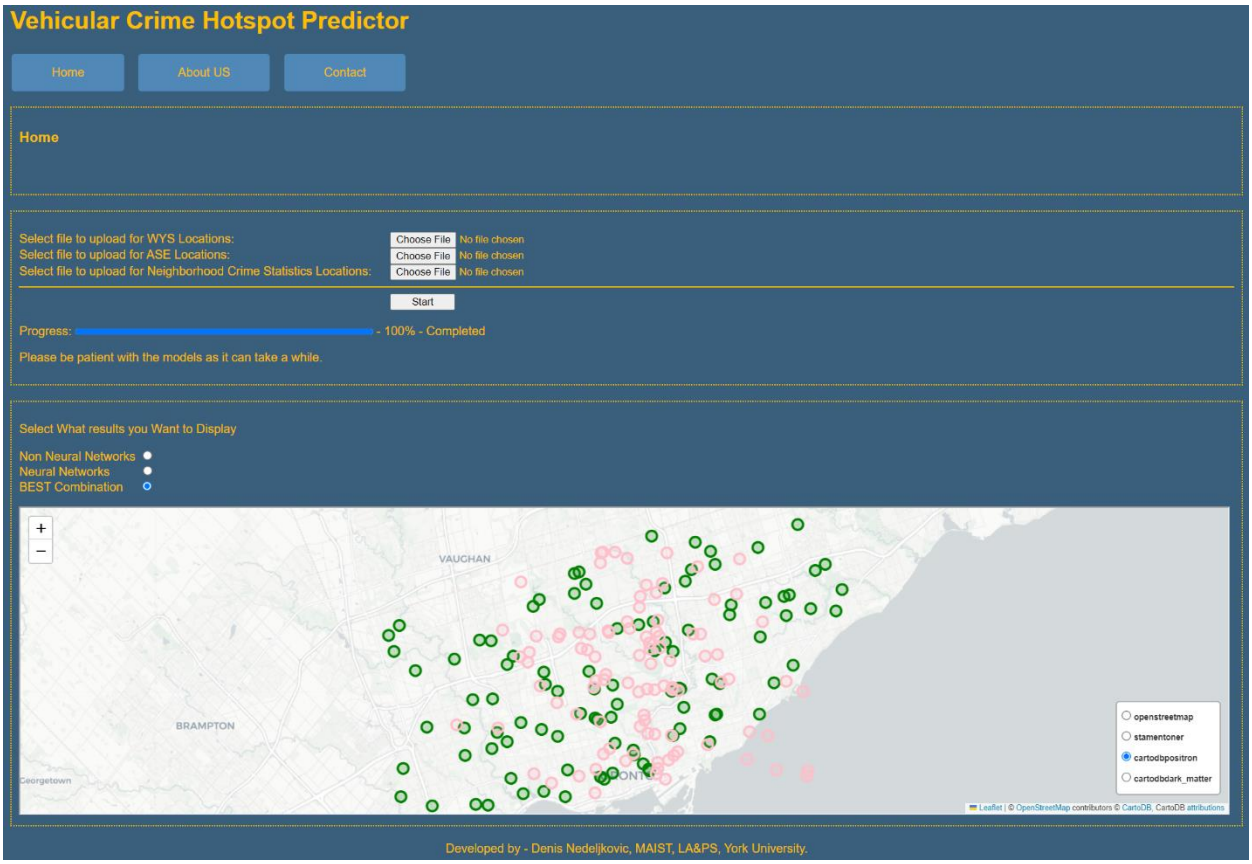

Figure 4.1: Graphical User Interface for the controller

This GUI provides a streamlined user experience where the system would operate in a black box capacity. A typical user would primarily be one with limited technical knowledge in machine learning, stakeholder, government official, or researcher.

B. Computational Architecture

Seen in algorithm 1, is the approach for allocation of process incorporating parallel processing and multi-threading. Whether or not a neural network is selected defines the logic on handling multiple process requests. When a non neural network is chosen, the controller splits the requests by the number of available threads on the target CPU by dividing the largest range of hyper-parameter testing for a single parameter. On the other hand, when a neural network is chosen, it splits the work of each individual target feature to its own thread, while incorporating dynamic memory allocation on the single GPU, as each job uses a virtual GPU that can grow its VRAM usage to fit its needs.

Algorithm 1 Pseudo-code for Controller

```
IMPORT necessary libraries
DEFINE constants and parameters
INITIALIZE memory tracking
procedure F_SELECTION
    DISPLAY options to select a target feature
    GET user input
    RETURN selected feature or error
end procedure
procedure ML_SELECTION
    DISPLAY options for machine learning approach
    GET user input for ML approach
    CALL multi with selected ML, parameters, and target
end procedure
procedure MULTI(ML task, parameters, target)
    DEFINE threading or processing logic
    CALL terminal command with ML task, args, thread
end procedure
CALL utility to create training and testing datasets
CALL ml selection function
```

As depicted in Algorithm 2, the approach involves sending commands for the jobs. This setup comprises a controller device that automatically initiates commands based on two simple selections in the controller menu, directing them to the server or laboratory machine for computation using the most appropriate hardware available for the job. These commands are transmitted over the web using SSH and the proprietary communication protocol. The hardware

requirements for the laboratory hardware specifications are: i7-12700KF, RTX 4070 Ti with 12 GiB of VRAM, 126 GiB of RAM and 128 GiB of swap to support, running in a headless Linux server with an Ubuntu distribution.

Algorithm 2 Pseudo-code for Sending Remote Commands

```
IMPORT os module
procedure SEND COMMAND(ML task, args, thread)
        SET user, server, and remote dir constants
        CONSTRUCT script with ML task, args, and thread
        DISPLAY the constructed script (for debugging)
        CONSTRUCT the SSH command using user, server, core, and script
EXECUTE the SSH command locally to run the script on the remote server on indicated thread and core
end procedure
```

Shown in Figure 4.2, is a high-level overview of the system's working environment from the user's local machine to the computational server. The user's environment handles the outgoing pre-process data files provided by the user as well as the processing logic and the incoming final predicted results from the server. On the back-end, the server receives the files with the provided processing logic, computes the requests, and provides the final predictions back to the user.

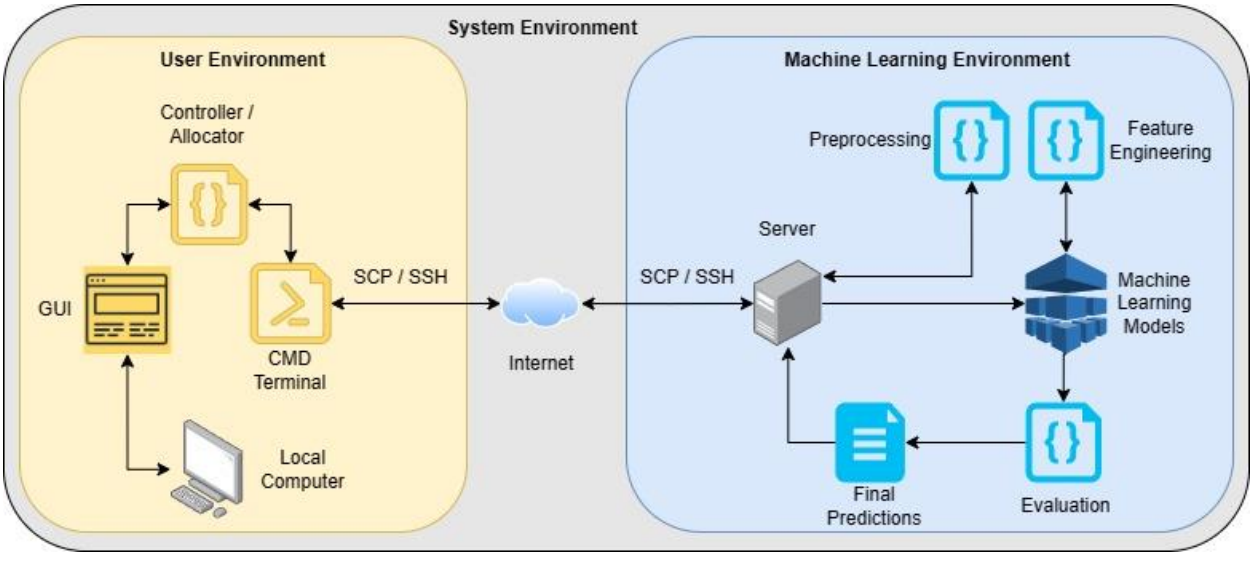

Figure 4.2: Graphical representation of the System's working environment

We extend this work to provide, an in-depth approach to training these models. we focus on testing the parameters for optimization, activation, kernel initializer, hidden layers, L1, L2, epoch, and batch size for all neural networks. In terms of SVM and the Random Forest, we run through their own unique set of parameters. This means that 1,263,360 combinations were tested for neural networks, 2,912,000 for SVM, and 1,555,000 for Random Forest, resulting in a total 1,910,120 combinations for a single target feature, and 5,730,360 combinations for all 3 targets. Henceforth, this system had a real time approximate of 65 hours and CPU time of 672 hours, achievable only by the allocation of jobs logic.

C. Load Balancing Strategy

As mentioned in the previous section of computational architecture, managing the load balancing is handled through the efficient distribution of computational tasks. This is paramount, especially given the complexity and volume of operations required for optimal machine learning model training. This necessitates a sophisticated load balancing strategy to ensure that this system infrastructure is utilized to its full potential, thereby minimizing bottlenecks and maximizing throughput. This load balancing approach is trifold, focusing on dynamic resource allocation, multithreading, and parallel processing to accommodate the diverse needs of the computational tasks, including deep learning, SVM, and Random Forest model optimization.

Dynamic resource allocation is pivotal for adapting to fluctuating workloads in real-time. By continuously monitoring system demand, resources are dynamically distributed across tasks, ensuring that no single node is overwhelmed. This approach not only optimizes hardware utilization but also enhances the system's responsiveness and reliability.

Multithreading is leveraged to concurrently execute multiple operations within a single job. By splitting the job into several subtasks that run simultaneously, it reduces execution time and improves the application's overall efficiency. This is particularly beneficial for processing large volumes of data from traffic sensors, where timely data analysis is crucial.

Parallel processing takes this a step further by distributing the jobs across multiple computing cores. This strategy is essential for processing complex machine learning algorithms and large datasets efficiently. By running multiple operations in parallel, the system can handle more data in less time, significantly speeding up the analysis and decision-making processes.

Together, these techniques form the backbone of an advanced computational architecture designed to support the demands of intelligent transportation systems. Through dynamic resource allocation, multithreading, and parallel processing, the system ensures optimal performance, enabling faster, more accurate traffic management and safety measure

D. Data-sets

As mentioned earlier, the training and final prediction data come from three different data sets merged into one. The first data set, Automated Speed Enforcement Monthly Charges, comes from the City of Toronto's Open Data Portal, an open-source delivery tool [15]. This set contains the locations in a descriptive manner of an ASE historical site and the related number of tickets issued during each month of its active use as features we are interested in.

The second data set, neighborhood-crime-rates –4326, comes from the City of Toronto's Open Data Portal [16]. This set contains the various crime rates and yearly counts across all of Toronto's Neighborhoods. The features we considered are the most recent Auto Theft crime rate, Theft from Motor Vehicle crime rate since these rates have already been normalized by the most

recent year of population prediction. In addition, we also took each neighborhood's geometry based on the given GPS coordinates.

The third data set, Watch Your Speed (WYS) Stationary monthly summary, comes from the City of Toronto's Open Data Portal [17]. This set contained monthly summary data of the Watch Your Speed Program or WYSP signs. The data includes all recorded instances of vehicles going over the posted speed limit beyond the sign's initial threshold. In short, it contains the sign's historical location paired with its percentile speed observed at the sign location from the 5^{th} percentile to the 95th percentile, the count of vehicles observed within an additional 5 km/h, from 5km/h over to 100 km/h in 5 km/h increments, and the measure over the speed limit volumetric vehicle count. In addition, the GPS coordinates of the sign are given.

E. Pre-processing

The pre-processing conducted in this chapter follows the same pipeline process previously mentioned in this thesis. Therefore, the final training data set consists of 42 independent features with three target features, 298 instances, and 13,112 data points.

The features used in the training data set can be found in Table 3.1.

The final three features represent the targets, which will be conducted under supervised learning. For each regression model used, three outcome models will be created for each target feature, totaling 15 models for this chapter.

These three target features are the paired Latitude coordinate and Longitude coordinate, with their associated total monthly median ticket count.

See Algorithm 3, for the pre-processing pseudo-code:

Algorithm 3 Pseudo-code for Data Merging and Cleaning

```
IMPORT required modules and libraries
procedure GET LAT_LONG_V2(location)
        PARSE location to extract relevant details
        RETRIEVE latitude and longitude for the location
        RETURN coordinates
end procedure
procedure SET_MONTHLY_MEDIAN_TICKETS(sensor)
    EXTRACT monthly ticket numbers
    COMPUTE median of ticket numbers
    RETURN sensor details with median
end procedure
procedure CLEAN_SENSOR_DATA(sensor list)
    PROCESS and clean sensor data
    RETURN cleaned list of sensors
end procedure
procedure CLEAN_CRIME_RATE(crime list)
    PROCESS and clean crime rate data
    RETURN cleaned list of crime rates
end procedure
procedure CLEAN_SIGNS(signs, lists)
     PROCESS and clean sign data based on list pairings
     RETURN cleaned list of signs
end procedure
procedure GET _LIST(data)
    CONVERT CSV data to list format
    RETURN list of data
end procedure
procedure RUN(sensor, crime, sign)
    LOAD data from provided sources
    CLEAN and process the loaded data
    GENERATE training and testing data-sets
end procedure
```

F. Feature Engineering

Before the training data is handled by the machine learning approaches, an extensive list of feature engineering components where applied. This list consists of:

- Feature Construction: Merging of the files
- Feature Transformation: Min-Max Scaling

- Feature Encoding: GPS Encoding
- Feature Selection: L1 and L2
- Dimensionality Reduction: kPCA

We explored featured engineering in depth as we wanted to improve this system's working time when handling requests, while concurrently maintaining respectable scores for MSE and R2. Such a balance is needed, since this system would need to handle multiple requests at once, and without it, working time to achieve desired results would exponentially increase.

The first three components are handled by the pre-processing logic and are applied seamlessly, which can be seen in the aforementioned subsection.

During the hyper-parameter tuning of the machine learning neural networks, the feature selection of the L1 and L2 passed through. These parameters carry the values of [None, 0.001, 0.01, 0.1, 1] and are passed through to reduce CPU time per process.

However, this finalized training data set has a kPCA applied to it, with a target variance of 0.99. This choice of a confidence level of 99% stems from the purpose of this work, where the confidence level is correlated to a lower error in accuracy for generalization at the cost of increasing computational time and resources. However, the primary reasoning behind this choice is that it would directly affect public safety and their standards.

Once the kPCA transforms the training set, the total independent features shrink from 42 to 17 for the minimum number of components needed to achieve a 99% confidence level, this can be said true for all three target features. The kernel that is utilized for this feature extraction is the Radial Basis Function (RBF), as it performs well on multitude of different data sets. The total amount of data points with the extracted data set of 17 independent features for over all

three target features results in 5,960 points of data, demonstrating a 54.55% reduction in overall size.

Yet, in contrast despite having a 99% confidence level, feature extraction technique of the kPCA has reduced demand for computational resources and time by more then 50%, when considering only the pre-processed data as the only factor. This further indicates that the original training data set contains 25 features that are deemed unnecessary for these models to achieve a near identical result, when compared to using the full data set.

G. Regressions

This chapter suggests five different approaches to training and solving this problem. The first two approaches are non neural networks, and the last three are neural networks. These models would target output features: Sensor Latitude, Sensor Longitude, and Monthly Median Tickets.

The approach with the first non-neural network machine learning model uses the Random Forest method. Such a method deals well with the limited number of features found in the data set. At the same time, the choice of Random Forest assists in preventing possible overfitting within the training model.

The Random Forest Regression formula is given previously by equation 1.

The second approach with a non-neural network machine learning model uses SVM, particularly a Support Vector Regression. we move forward with this model as it is highly efficient and develops fast results outside large data sets.

The SVM formula for regression is given by:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \qquad \textit{Equation 3}$$

With the following for the constraints:

$$y_i - (w \cdot \phi(x_i) + b) \leq \varepsilon + \xi_i, \qquad \textit{Equation 4}$$

$$(w \cdot \phi(x_i) + b) - y_i \leq \varepsilon + \xi_i^*, \qquad \textit{Equation 5}$$

$$\xi_i, \xi_i^* \geq 0, \quad i = 1, 2, \ldots, n, \qquad \textit{Equation 6}$$

The first approach with a neural network machine learning model involves the Fourier Series Neural Network FSNN. It is chosen as it provides high efficiency with nonlinear regression data [16].

The FSNN formula is given previously by equation 2.

The second approach with a neural network machine learning model involves the LSTM. A Long Short-Term Memory architecture is chosen for this approach as it is used in deep learning.

The LSTM formula is given by:

$$f_t = \sigma(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f) \qquad \textit{Equation 7}$$

$$i_t = \sigma(W_{xi} \cdot x_t + W_{hi} \cdot h_{t-1} + b_i) \qquad \textit{Equation 8}$$

$$\tilde{C}_t = \tanh(W_{xc} \cdot x_t + W_{hc} \cdot h_{t-1} + b_c) \qquad \textit{Equation 9}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \qquad \textit{Equation 10}$$

$$o_t = \sigma(W_{xo} \cdot x_t + W_{ho} \cdot h_{t-1} + b_o) \qquad \textit{Equation 11}$$

$$h_t = o_t \odot \tanh(C_t) \qquad \textit{Equation 12}$$

The third approach with a neural network machine learning model involves using the FNN mentioned earlier. A Feed Forward Neural Network architecture is chosen for this approach of the design's speed and the neural network's dynamic nature. The idea behind selecting FNN is to see whether a provided general neural network approach can result in more outstanding scores and ASE hotspot placements.

The FNN formula is given by:

$$\hat{y} = f\left(\sum_{i=1}^{n} w_i^{(2)} \cdot h\left(\sum_{j=1}^{m} w_j^{(1)} \cdot x_j + b^{(1)}\right) + b^{(2)}\right)$$

Equation 13

To give a better understanding on how the ML models are implemented, See Algorithm 4 for pseudo-code:

Algorithm 4 Pseudo-code for ML Implementation

```
Import necessary libraries for machine learning and data handling
function GPUCHECK
    Check and configure GPU settings for Virtual GPU in TensorFlow
end function
function CUSTOM _SCORER(y test, y pred)
    Compute custom scoring metric
end function
function CREATE _MODEL(xShape, yShape, parameters)
    Create and Return an ML model based on specified parameters
end function
function PROGRESS(parameters, trail, time)
    Track and Display progress of model training
end function
function FUNCTIONFINDBESTPARAMS(parameters, name, layers)
    Find and Return the best hyperparameters for the LSTM model
end function
function START(f, t, H1, H2, l)
    Initiate the model training, hyperparameter tuning process and log the work
end function
Read user inputs for feature and hidden layers
Determine the type of model to train based on input
Call START with the specified parameters
```

H. Final Method

Once all 15 feature models are created from the three regression approaches, they would run against the unused portion of the data set, the final prediction of hot spots. With this, the most significant target features of the predicted latitude and longitude value pairs, with their associated predicted monthly ticket predictions, would be graphed on a map of the City of Toronto in increasing ranges. These graphs will display the top 50 ’best’ hot spot locations based on the top-performing machine learning models for each target feature, as well as the top 50 hot spot locations for non-neural networks and neural networks, respectively. They will help visualize each regression model’s prediction-making process. See Algorithm 5 for pseudo-code.

Algorithm 5 Final Prediction and Map Plotting

```
Import libraries for web browsing, data processing, machine learning, and mapping
Load machine learning models and scalars from specified paths
Set up a map with initial location and zoom level
Add tile layers to the map for visualization
function INITIALIZE
    Load and process data from CSV files
    Return processed data and true values
end function
Call INITIALIZE
Store returned data for further use
for each machine learning model (SVM, FNN, FSNN, LSTM, RF) do
    Predict tickets, latitude, and longitude using the model
    Process and scale the predictions
    Map the results on the map with unique colors and labels
end for
Save the map as an HTML file
```

4.4 Evaluation

To gain a better understanding of the final predicted work with the unused and unpaired data, further in-depth graphical analysis has been created for each of the 15 models. These graphs

are the representation of the predicted point to the real point, using the test segment of the training data. The closer the points are to the red line, the greater the accuracy and performance of the models. Scores shown on the Predicted and Actual axis are merely arbitrary as they represent the standardization of the min-max scale of 0 to 1.

Once we have established the final graphical performance representation of each machine learning approach for each respective target feature, we hand-select the models with the highest R2 score for each feature. These selected models are then utilized collectively to predict the best hot spots based on the highest median ticket count. For each prediction, we determine the latitude and longitude coordinates. Additionally, we plot locations for all neural networks and non-neural networks, respectively.

A. Random Forest

The first non neural network approach that was explored is the random forest. As mentioned earlier, this approach was selected to lower the chances of receiving an over-fitting bias. For each target feature, hyper-parameter testing was conducted with max depth range of 1 to 400, a max node range of 2 to 400, number of estimators of [100, 500, 1000], mean samples split of [2, 6, 10], a min sample leaf of [1, 5, 10], and a ccp alpha of [0.0, 0.05, 0.1]. This approach averaged approximately 5 hours across all three target features in real time following the above-mentioned distribution of job processes. When taking into consideration the target features with the distribution logic, a total of approximately 1,555,200 parameter combinations were tested with a total CPU time of 300 hours.

The hyper-parameter testing resulted in an MSE score of 0.009408759 and an R2 score of 0.08611237. This is depicted in Figure 4.3, where the clustering of points closer to the red line in

the corner can be attributed to the MSE score, while the outliers near the top of the graph can be explained by the poor R2 score. However, considering that this is the best R2 score for the ticket target feature in any of the approaches, as demonstrated by a handful of points placed further right compared to the cluster, these scores were achieved with a maximum depth of 15 and 1000 estimators.

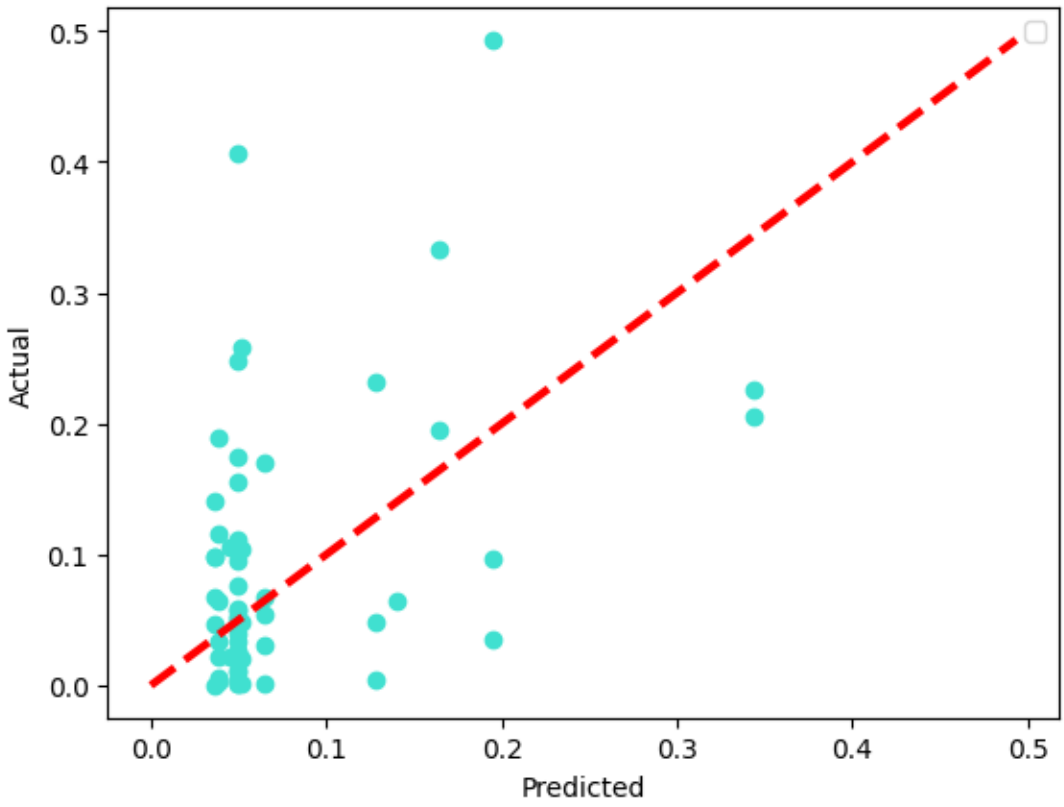

Figure 4.3: Predicted vs Actual results for Ticket model from Random Forest Regression Approach

The hyper-parameter testing resulted in an MSE score of 0.036615895 and an R2 score of 0.241720172. This is illustrated in Figure 4.4, where the red line indicates the MSE scores, and the spread of the points across the x-axis represents the R2 score. Interestingly, the points are grouped in vertical columns, likely due to the tree architecture of the approach. Where each

visual column is possibly described by a dominate path and or branch of the tree. These scores were achieved with a maximum depth of 10 and 100 max estimators.

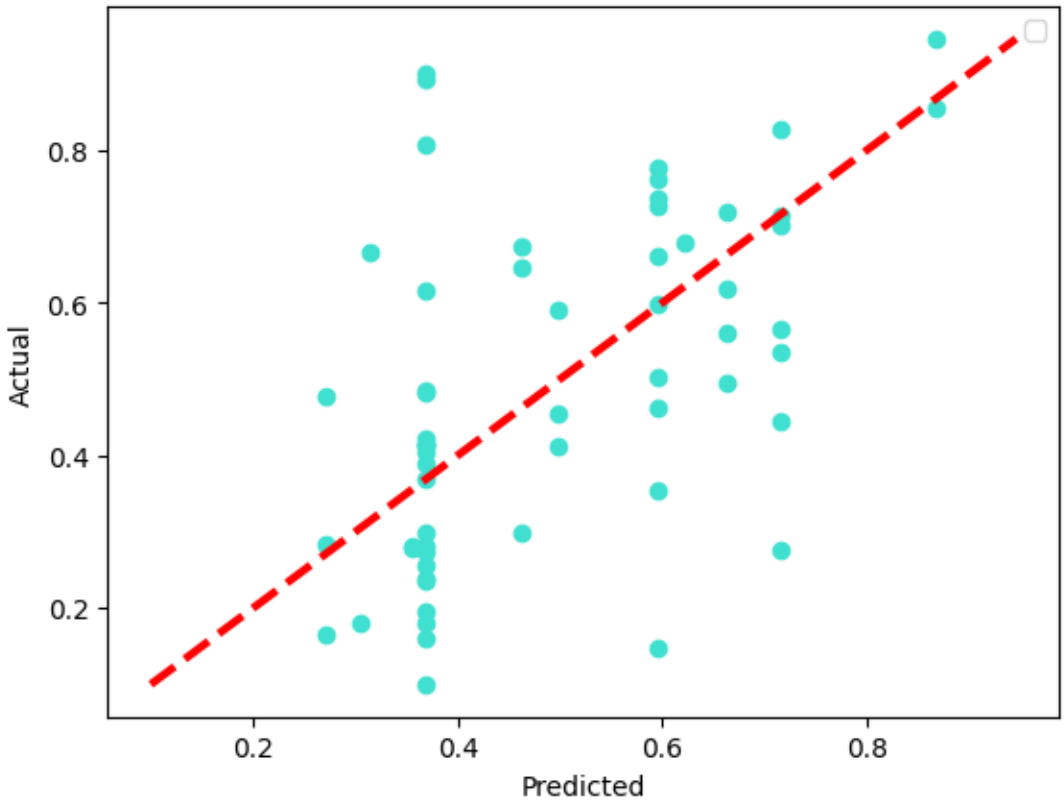

Figure 4.4: Predicted vs Actual results for Latitude model from Random Forest Regression Approach

The hyper-parameter testing resulted in an MSE score of 0.030842846 and an R2 score of 0.379894144. This is illustrated in Figure 4.5, with proper clustering along the red line indicating the MSE score, and more variance in the results compared to the Latitude model, as explained by the R2 score. This graph shows the same unique pattern of vertical columns spread across the horizontal axis, explained by the architecture of the machine learning approach. These scores were achieved with a maximum depth of 10 and 500 estimators.

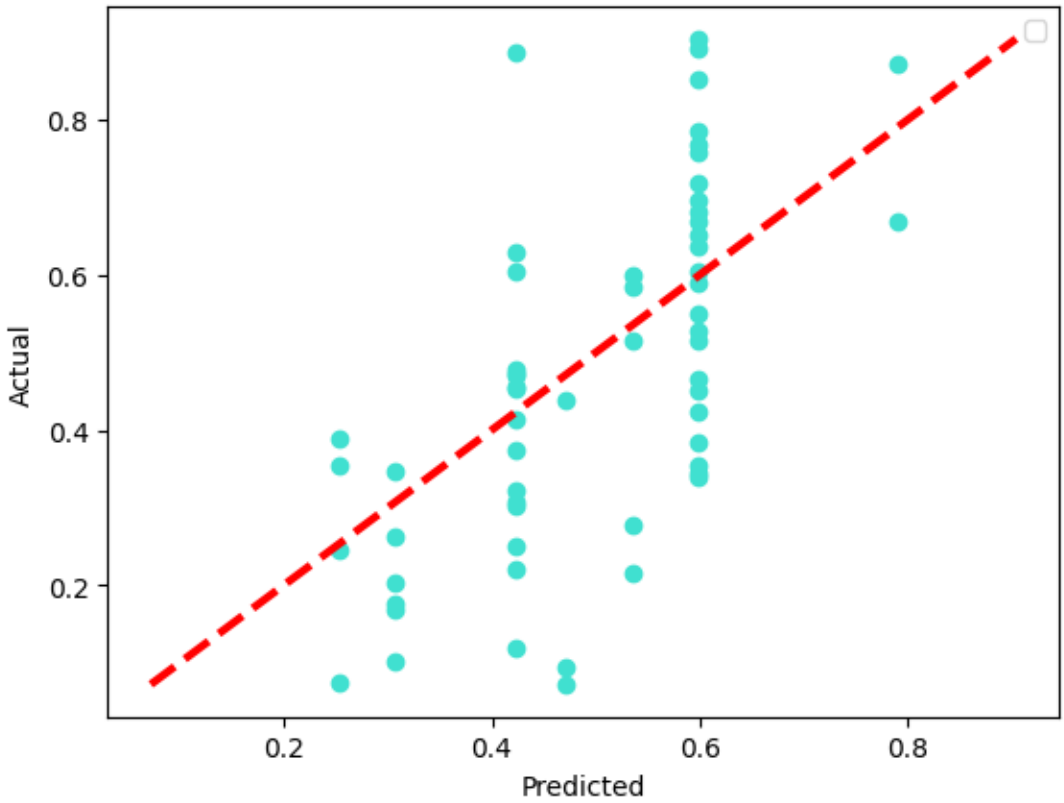

Figure 4.5: Predicted vs Actual results for Longitude model from Random Forest Regression Approach

B. Support Vector Machine

The second non neural network approach that was explored is the SVM, more particularly the support vector regression. As mentioned earlier, this approach was selected for its rapid computation of regression datasets. For each target feature, hyper-parameter testing was conducted with a value of C from 0 to 1 increment by 0.019, kernel of [’rbf’, ’poly’, ’sigmoid’, ’linear’], degree of range 1 to 10, gamma of [0.0001,0.001, 0.01, 0.1, 1, 10, 100], a coef0 of [0,1], a epsilon of [0.1, 0.2, 0.3, 0.4,0.5,0.6,0.7,0.8,0.9,1], shrinking of true or false, tol of [0.0001,0.001, 0.01, 0.1, 1], a cache size of 1000 and max iteration that is unlimited. This approach averaged approximately 50 minutes of real time across all three target features

following the above-mentioned distribution of job processes. When taking into consideration the target features with the distribution logic, a total of approximately 2,912,000 parameter combinations are tested with a total CPU time of approximately 12 hours.

The hyper-parameter testing resulted in an MSE score of 0.0099721 and an R2 score of 0.03139416. This is illustrated in Figure 4.6, where the clustering of points closer to the red line in the corner can be explained by the MSE score, while the outliers near the top of the graph can be attributed to the poor R2 score. These scores were achieved with a degree of 5, a kernel of poly, a C of 0.20754717, gamma of 1, coef0 of 1, an epsilon of 0.1, and a tol of 0.1.

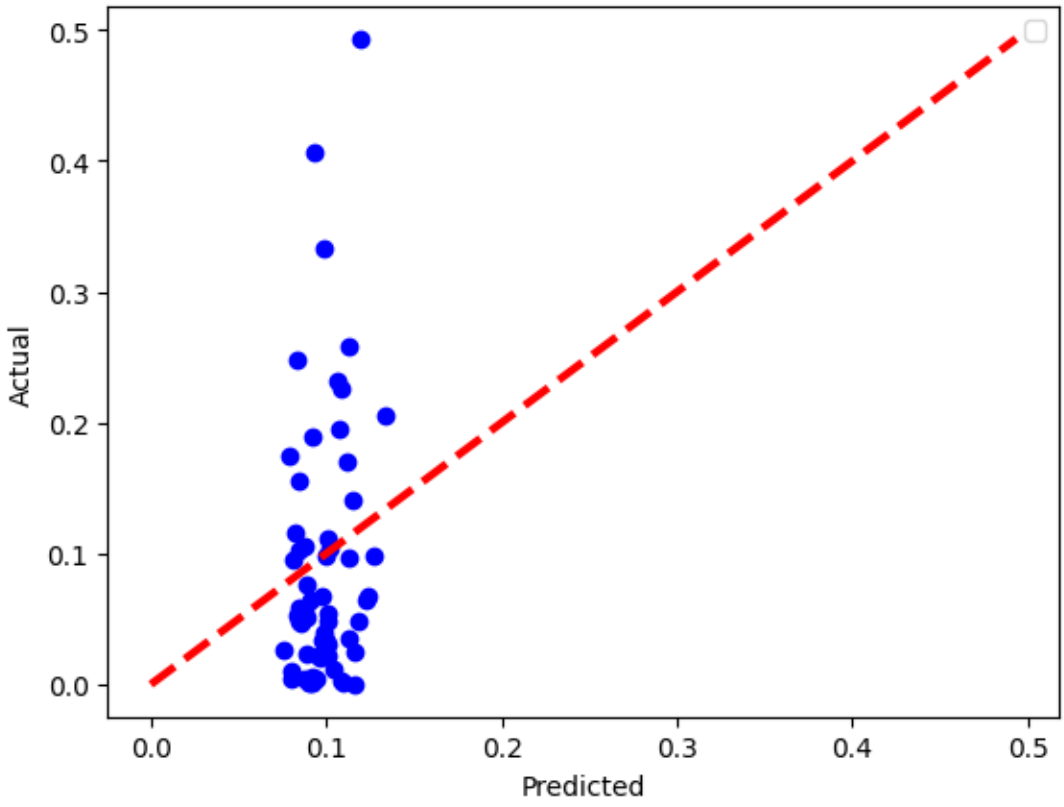

Figure 4.6: Predicted vs Actual results for Ticket model from SVM Regression Approach

The hyper-parameter testing resulted in an MSE score of 0.02448489 and an R2 score of 0.49294157. This is illustrated in Figure 4.7, where the proximity of the points to the red line indicates the MSE score, while the R2 score is reflected in the few outliers. To achieve these scores, a degree of 1, a kernel of RBF, a C of 0.41509434, gamma of 10, coef0 of 0, tol of 01, and an epsilon of 0.1 were selected.

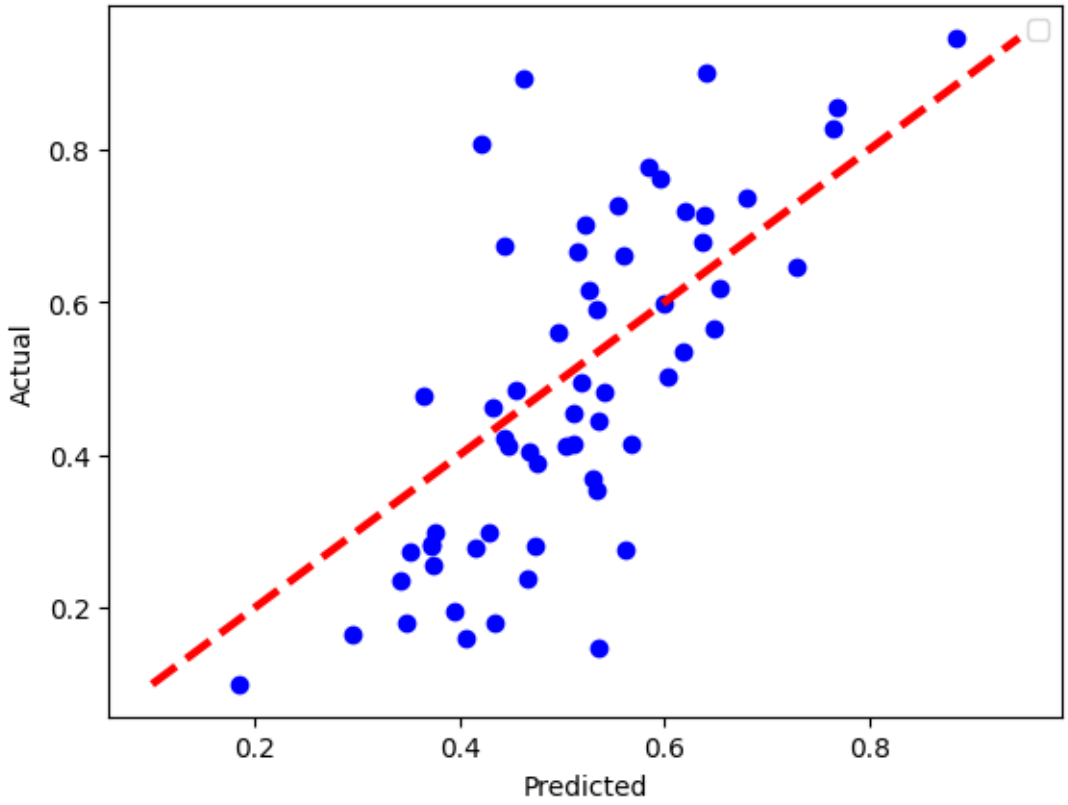

Figure 4.7: Predicted vs Actual results for Latitude model from SVM Regression Approach

The hyper-parameter testing resulted in an MSE score of 0.030107609 and an R2 score of 0.39467634. This is demonstrated in Figure 4.8, with proper clustering along the red line indicated by the MSE score, and a peculiar spread primarily vertical, reflecting the R2 score. To achieve these scores, a degree of 1, a kernel of RBF, C of 0.698113208, gamma of 100, coef0 of 0, tol of 0.1, and epsilon of 0.1 were selected.

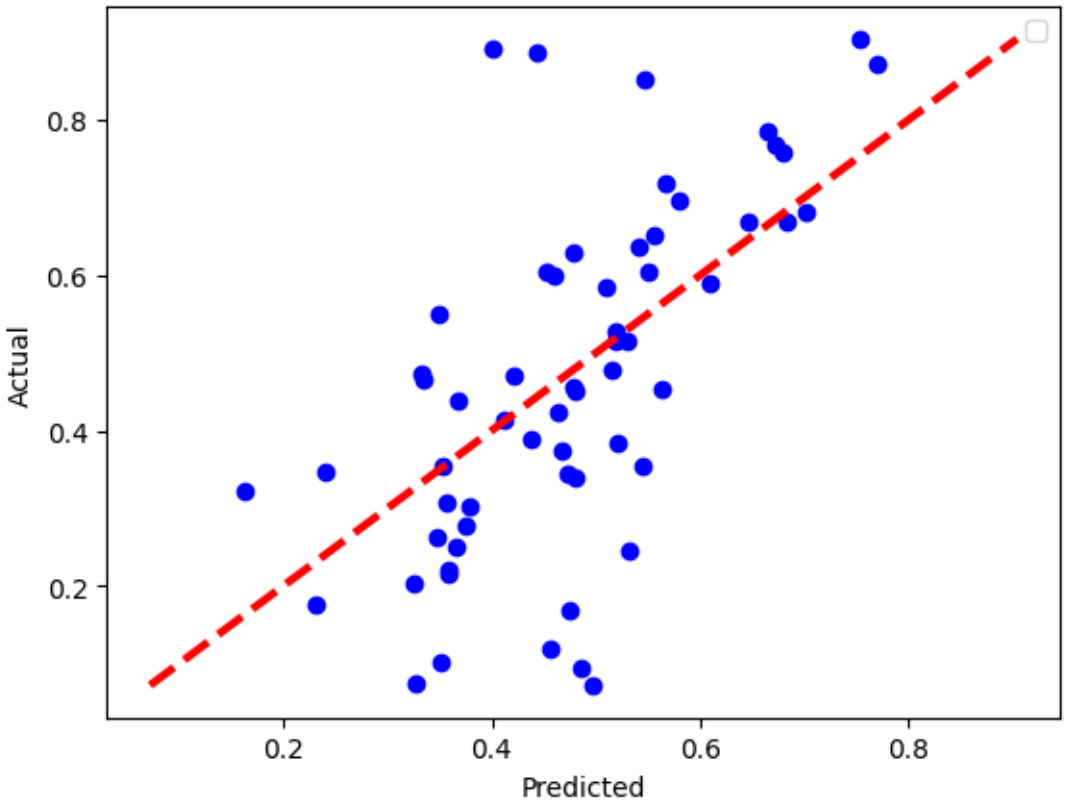

Figure 4.8: Predicted vs Actual results for Longitude model from SVM Regression Approach

C. Long Short-Term Memory

The first neural network approach that was explored is the LSTM. Each model trained has an architecture built with n hidden layers plus an initial hidden layer, each with 17 nodes. This function is selected as it provides better results when compared to the rectified linear unit activation function [18]. Additionally, it results with a dropout of 0.2 and return sequences. The parameters tested are batch size of [4,8,16,32,64,128], epoch of [25,50,75,100,125,150,175,200], optimizer of [’Adam’], activation of [’ReLU’, ’tanh’, ’sigmoid’], kernel initializer of [None, ’glorot uniform’, ’LecunNormal’, ’LecunUniform’], L1 of [None, 0.01, 0.1, 1], L2 of [None, 0.01, 0.1, 1], number of hidden layers of [0,1,2,3,4], drop rate of [0.2,0.3,0.4,0.5,0.6], and units of [10, 50, 100]. This approach averaged approximately 20 hours of real time across all three

target features following the above-mentioned distribution of job processes. When taking into consideration the target features with the distribution logic, a total of approximately 691,200 parameter combinations were tested with a total CPU time of approximately 225 hours.

The hyper-parameter testing resulted with a MSE score of 0.009743074 and a R2 score of 0.053639812. This is demonstrated in Figure 4.9, where the points clustered towards the red line in the corner can be explained by the MSE score, and the points roughly grouped in a vertical fashion can be explained by the low R2 score. To achieve this score, a batch size of 4, epoch of 100, with 2 intermediate hidden layers, activation of tanh, kernel initializer of none, drop rate of 0.4, 100 units, L1 of None, and a L2 of None were selected.

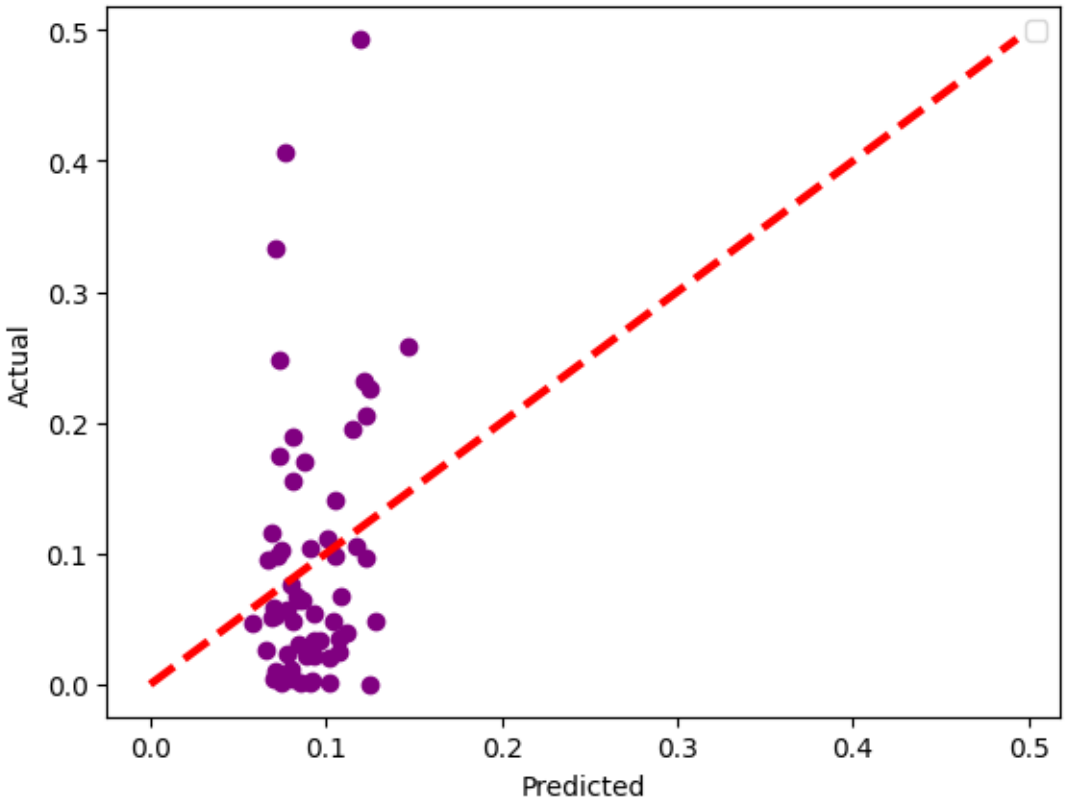

Figure 4.9: Predicted vs Actual results for Ticket model from LSTM Regression Approach

The hyper-parameter testing resulted with a MSE score of 0.026634357 and a R2 score of 0.448428179. This is demonstrated in Figure 4.10, where the points clustered horizontally towards the red line can be explained by the MSE score, and those spread across the line of perfect prediction is explained by the R2 score. To achieve this score, a batch size of 4, epoch of 200, with only the initial hidden layer, activation of tanh, kernel initializer of none, drop rate of 0.5, 100 units, L1 of None, and a L2 of None were selected.

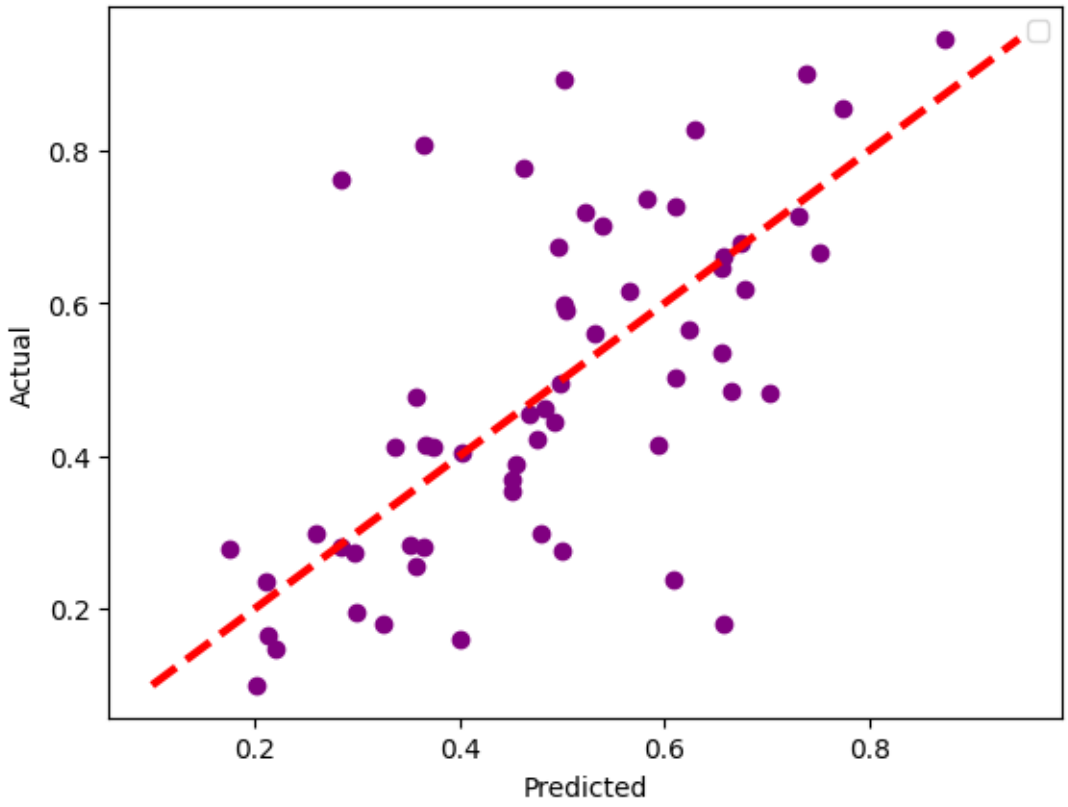

Figure 4.10: Predicted vs Actual results for Latitude model from LSTM Regression Approach

The hyper-parameter testing resulted with a MSE score of 0.028160558 and a R2 score of 0.433822456. This is demonstrated in Figure 4.11, where the points are clustered close to the red line wit some visual vertical grouping can be explained by the MSE score, and all of the points in a singular line of points can be explained by the approximate zero R2 score. To achieve this

score, a batch size of 128, epoch of 25, with only the initial hidden layer, activation of tanh, kernel initializer of none, drop rate of 0.6, 50 units, L1 of None, and a L2 of None were selected.

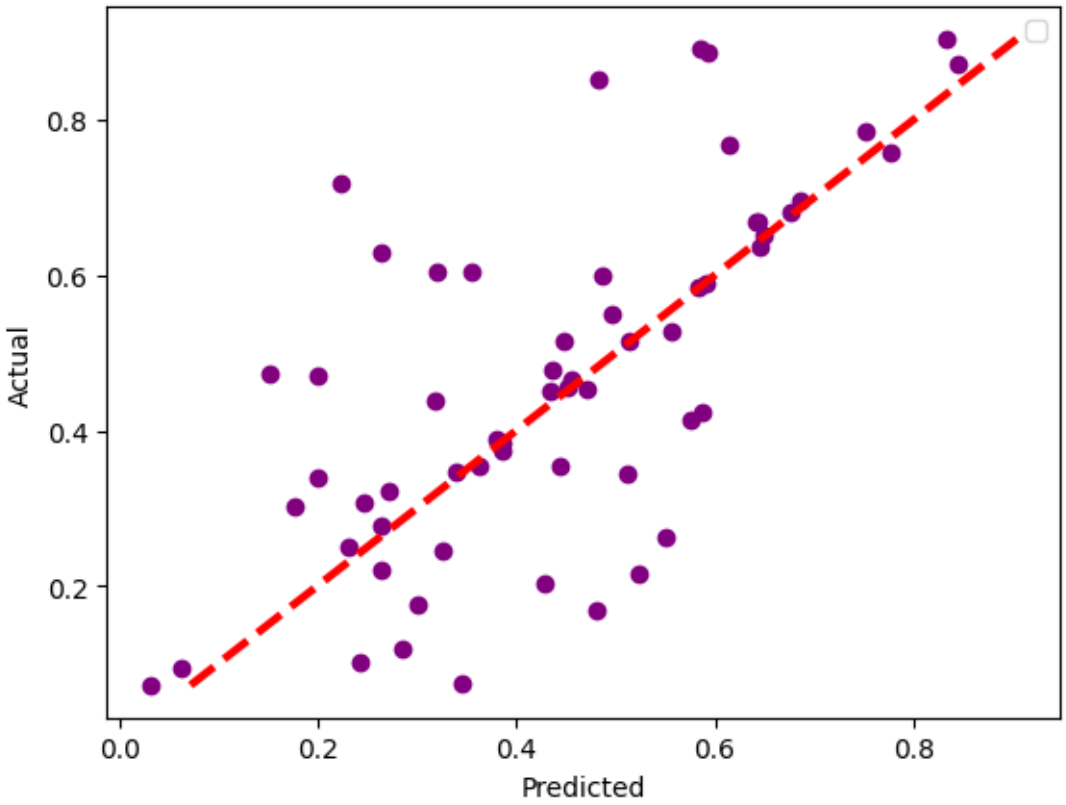

Figure 4.11: Predicted vs Actual results for Longitude model from LSTM Regression Approach

D. Fourier Series Neural Networks

The second neural network approach that was explored is the FSNN. Each model trained has an architecture built with n hidden layers, each with 17 nodes per layer, with a custom sine activation objection function. The parameters tested are batch size of [1,2, 4, 6, 8, 16, 32, 64, 128, 256, 512, 1024], epoch of [0,10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160, 170,180,190,200], optimizer of [’Adam’, ’Nadam’], activation of [1,2,3,4,5,6,7,8,9,10], kernel initializer of [None, ’glorot uniform’, ’normal’, ’Uniform’], L1 of [None,0.001], L2 of [None,

0.001] and number of hidden layers of [0,1,2,3,4]. This approach averaged approximately 5 hours of real time across all three target features following the above-mentioned distribution of job processes. When taking into consideration the target features with the distribution logic, a total of approximately 403,200 parameter combinations were tested with a total CPU time of approximately 45 hours.

The hyper-parameter testing resulted with a MSE score of 0.010020296 and a R2 score of 0.026712757. This is demonstrated in Figure 4.12, where the points clustered towards the red line in the corner can be explained by the MSE score, and all of the points in a rough vertical line can be explained by the R2 score. To achieve this score, a batch size of 1, epoch of 10, with 1 intermediate hidden layer, activation of Adam, kernel initializer of None, L1 of None, L2 of None, and a n activation value of 7 for the custom sine activation were selected.

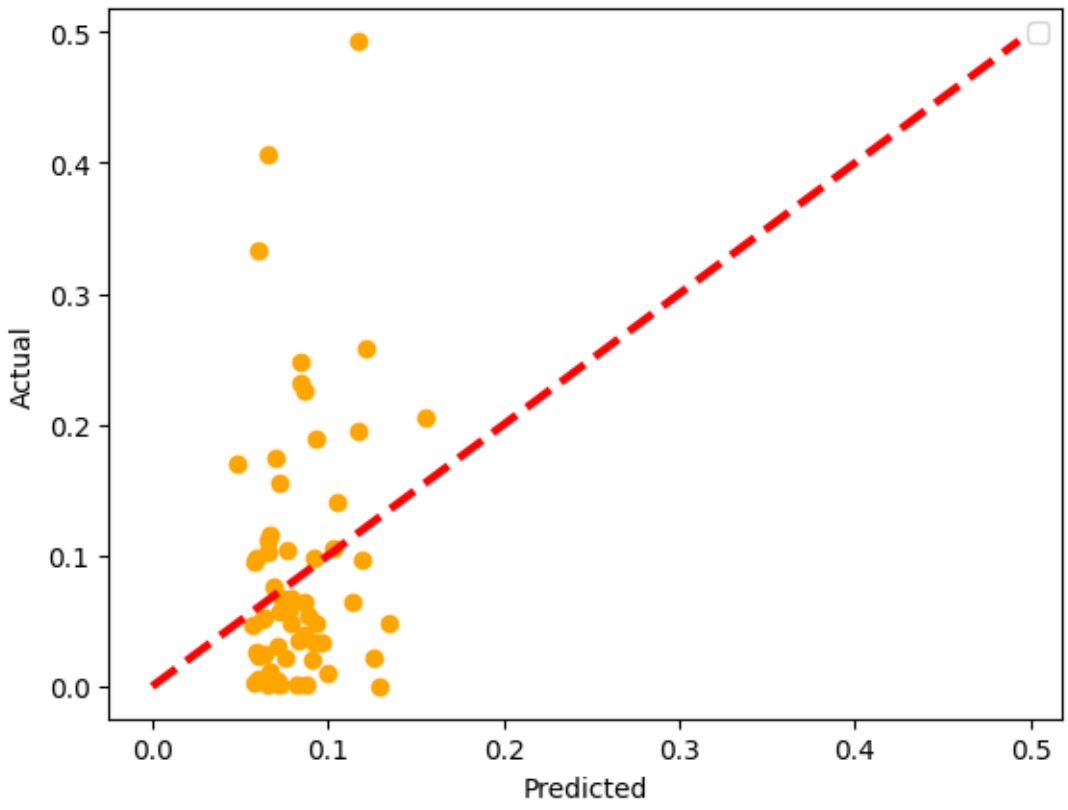

Figure 4.12: Predicted vs Actual results for Ticket model from FSNN Regression Approach

The hyper-parameter testing resulted with a MSE score of 0.025443776 and a R2 score of 0.473083963. This is demonstrated in Figure 4.13, where clustering of the points towards the red line can be explained by the MSE score, and the points following the line of perfect prediction are explained by the R2 score. To achieve this score, a batch size of 1, epoch of 20, with 0 intermediate hidden layer, activation of Adam, kernel initializer of None, L1 of None, L2 of None, and a n activation value of 9 for the custom sine activation were selected.

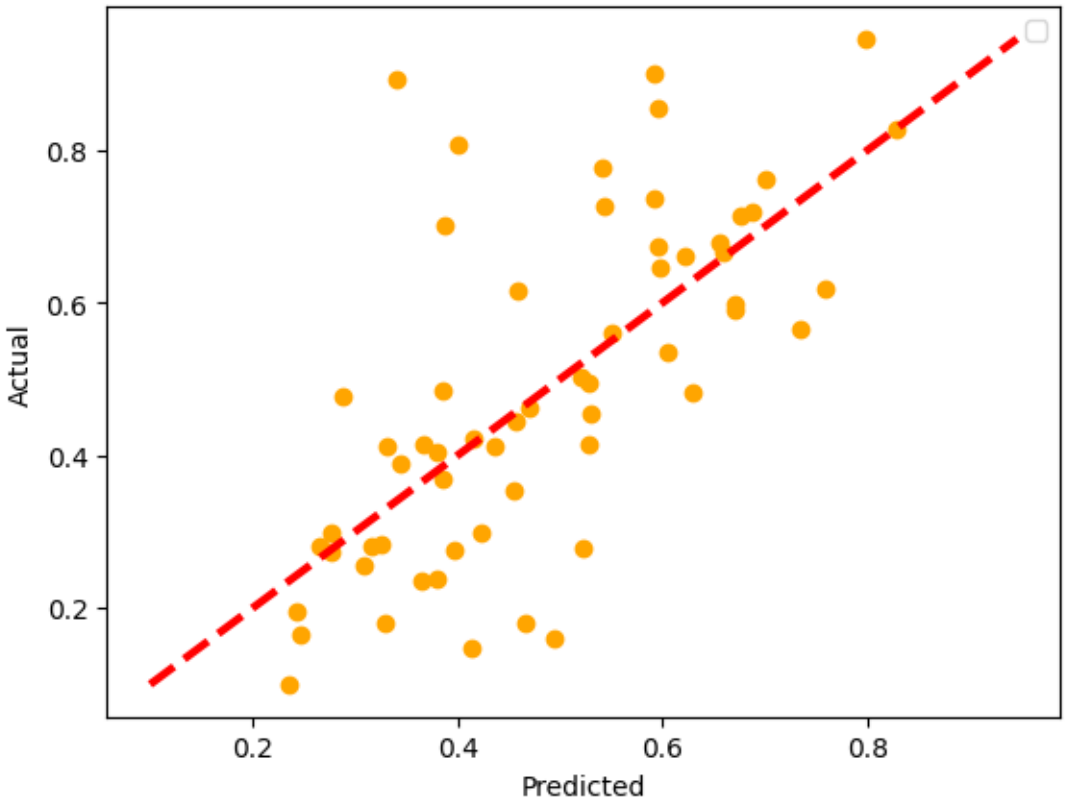

Figure 4.13: Predicted vs Actual results for Latitude model from FSNN Regression Approach

The hyper-parameter testing resulted with a MSE score of 0.031902056 and a R2 score of 0.358598372. This is depicted in Figure 4.14 - the way points are clustered in the closer to the

red line can be explained by the MSE score, and the points following line of perfect prediction are explained by the R2 score. To achieve this score, a batch size of 1, epoch of 50, with 0 intermediate hidden layer, activation of Adam, kernel initializer of None, L1 of None, L2 of None, and a n activation value of 7 for the custom sine activation were selected.

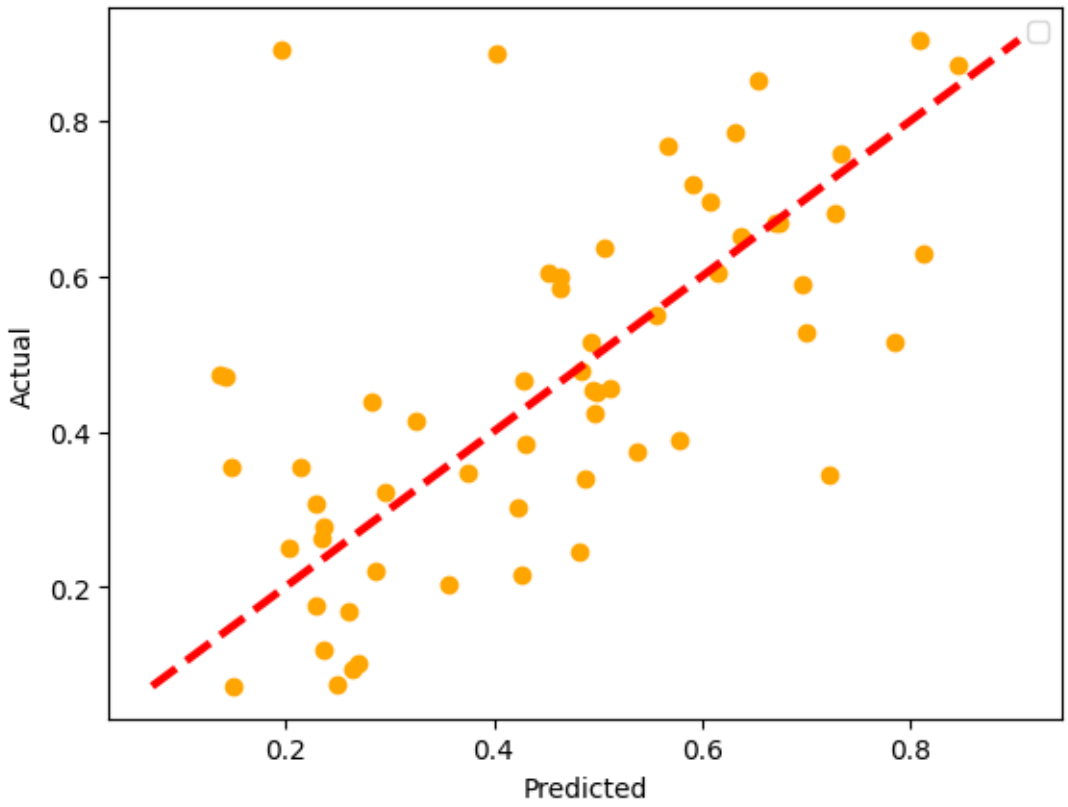

Figure 4.14: Predicted vs Actual results for Longitude model from FSNN Regression Approach

E. Feed Forward Neural Network

The third neural network approach that was explored is the FNN. Each model Trained has an architecture built with n hidden layers, each with 9 nodes per layer. This approach averaged a real time of approximately 340 minutes on all three target features following the above-mentioned distribution of job processes. The parameters tested are batch size of

[2,4,6,8,16,32,64,128,256,512, 1024], epoch of [25,50,75,100,125,150,175,200], optimizer of [’Adam’, ’Nadam’], activation of [’relu’, ’selu’, ’linear’], kernel initializer of [None, ’glorot uniform’, ’normal’, ’uniform’], L1 of [None, 0.01, 0.1, 1], L2 of [None, 0.01, 0.1, 1] and number of hidden layers of [0,1,2,3,4]. When taking into consideration the target features with the distribution logic, a total of approximately 168,960 parameter combinations were tested with a total CPU time of approximately 90 hours.

The hyper-parameter testing resulted with a MSE score of 0.009912769 and a R2 score of 0.037157062. This is demonstrated in Figure 4.15, where the clustering of the points towards the red line in the corner can be explained by the MSE score, and the outliers near the top and bottom of the graph can be explained by the poor R2 score. To achieve these scores a batch size of 8, epoch of 50, optimizer of Nadam, activation of relu, kernel initializer of uniform, L1 of None, L2 of 0.01 and 1 hidden layer were selected.

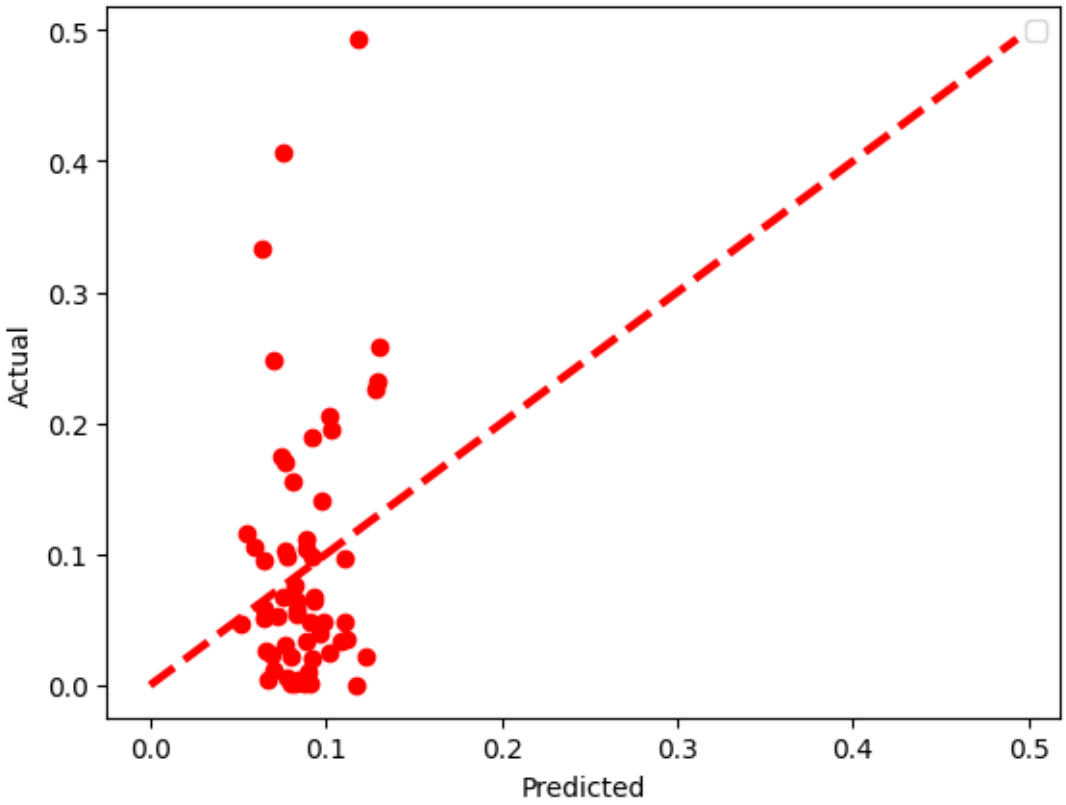

Figure 4.15: Predicted vs Actual results for Ticket model from FNN Regression Approach

The hyper-parameter testing resulted with a MSE score of 0.026405682 and a R2 score of 0.453163814. This is demonstrated in Figure 4.16, indicated by the red line and the proximity of the points to it demonstrating the MSE score, while the R2 score represents the few outliers. To achieve these scores, a batch size of 8, epoch of 50, optimizer of Adam, activation of relu, kernel initializer of uniform, L1 of None, L2 of None and 1 hidden layer were selected.

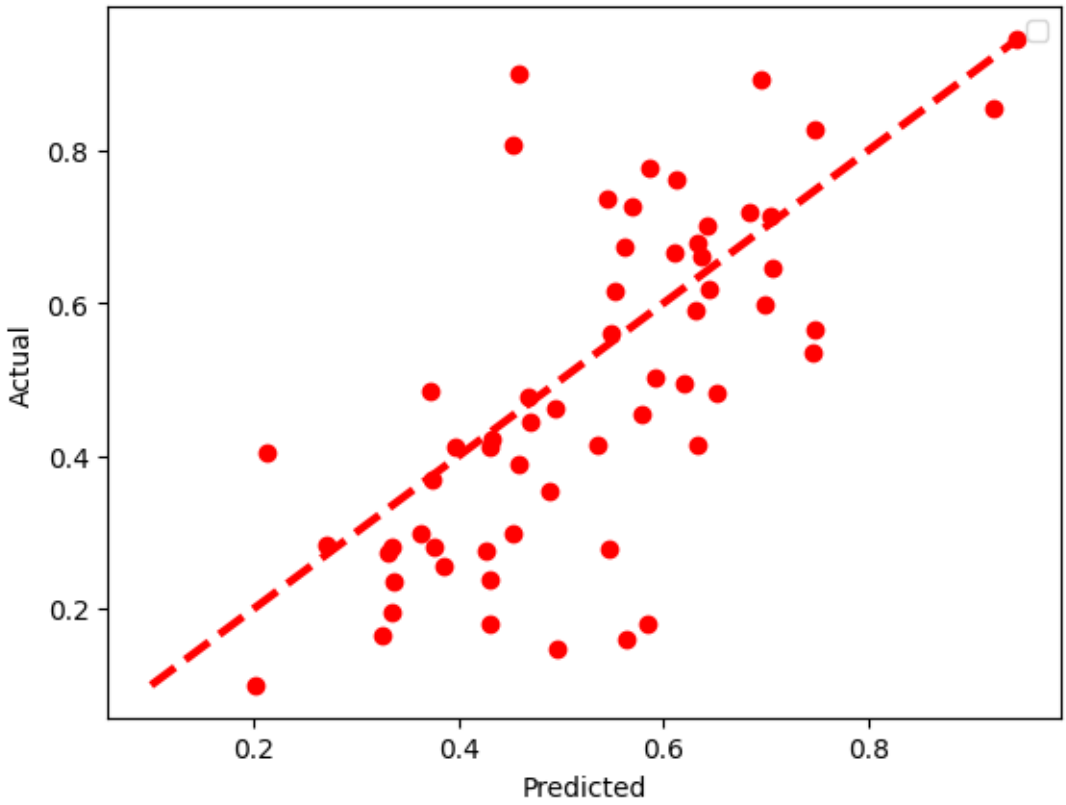

Figure 4.16: Predicted vs Actual results for Latitude model from FNN Regression Approach

The hyper-parameter testing resulted with a MSE score of 0.032136028 and a R2 score of 0.353894276. This is demonstrated in Figure 4.17, where the points clustered close to the red line in the corner can be explained by the MSE score, and the outliers near the top and bottom of the graph can be explained by R2 score. To achieve these scores, a batch size of 8, epoch of 100, optimizer of Nadam, activation of relu, kernel initializer of uniform, L1 of None, L2 of 0.01 and 0 intermediate hidden layers were selected.

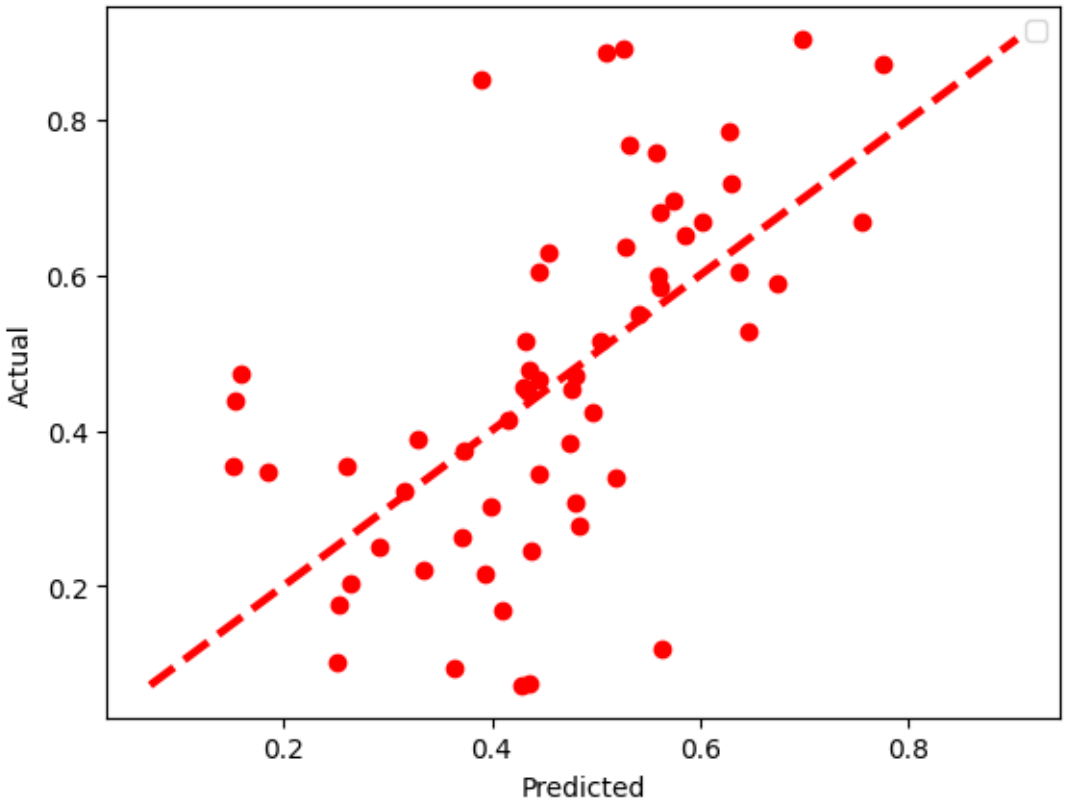

Figure 4.17: Predicted vs Actual results for Longitude model from FNN Regression Approach

F. Comparison

The MSE scores represent the average accuracy of the predicted results compared to their actual test counterparts. We aim to achieve a score lower then 0.05% to provide a minimized error in prediction while still ensuring a maximized R2 score. Seen in Figure 4.18, the MSE scores across the board provide results that satisfy the requirement. Overall, LSTM provides the best MSE scores while SVM provides the lowest for the Ticket target feature. Table 4.1 demonstrates the change in MSE results when compared with those of the previous study [4].

We aim to achieve a MSE score of at least 0.05 or lower which We have gotten across all three target features for each machine learning approach, and can be seen in Figure 4.18. A low

score assures us that these models' accuracy is within a 5% threshold without running into an over-fitting bias. As shown in the graph, LSTM performed the worst while FNN performed the best.

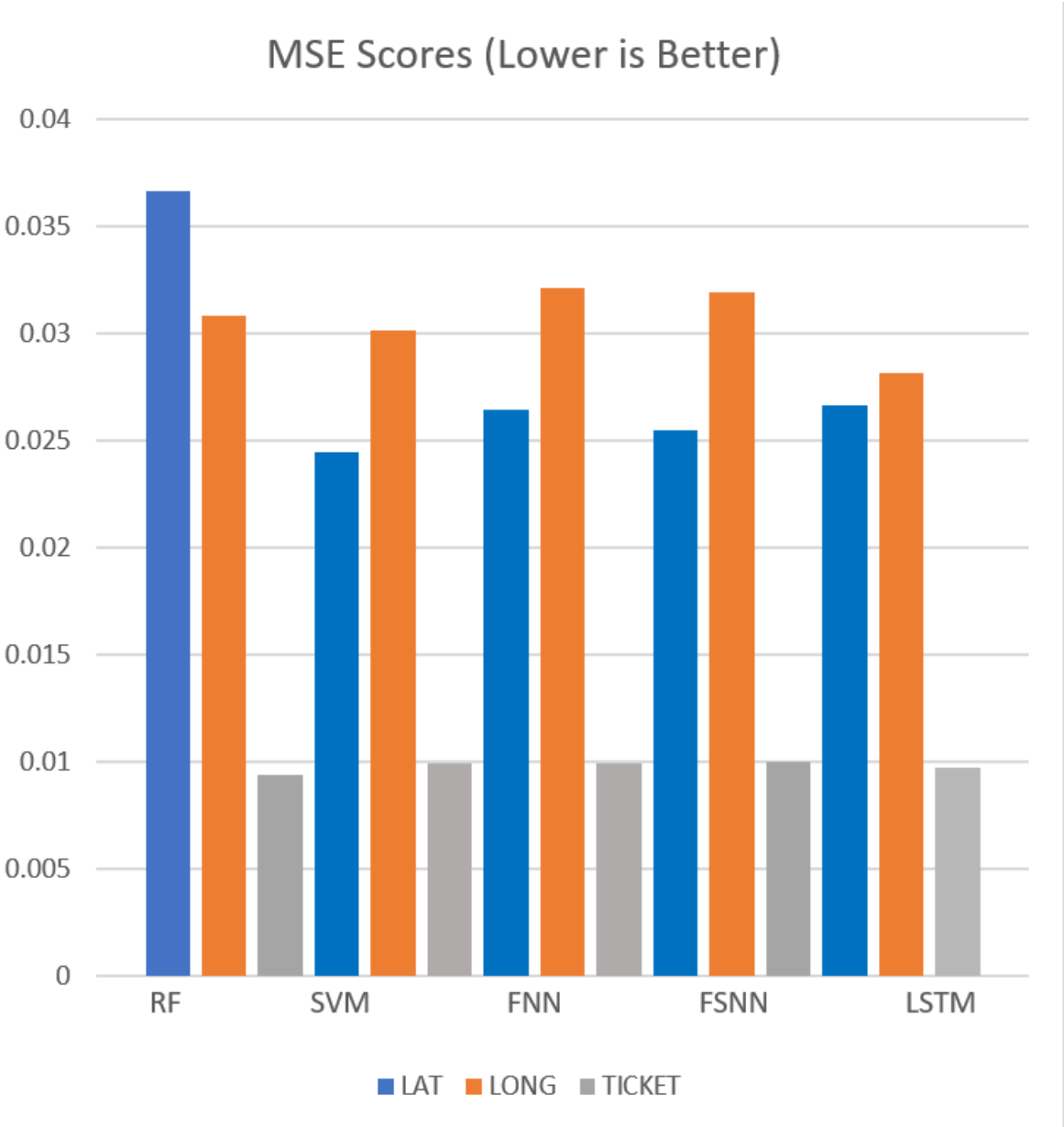

Figure 4.18: MSE Scores

We aim to achieve a R2 score of the greatest possible value once a MSE of 0.05 or lower is given for the related model. The higher the R2 score the better the relationship between the independent features and the certain target feature. A very low or near zero score can lead to extreme results which can be seen in all of the machine learning approaches for their target feature of Monthly Median Ticket Count. We believe that the LSTM approach overall did so well due to its complex nature which allows for more room to form better pathways between neurons. On the other hand, RF preformed the worst because of the reduced sized of the dataset due to the kPCA. This can be seen in Figure 4.19.

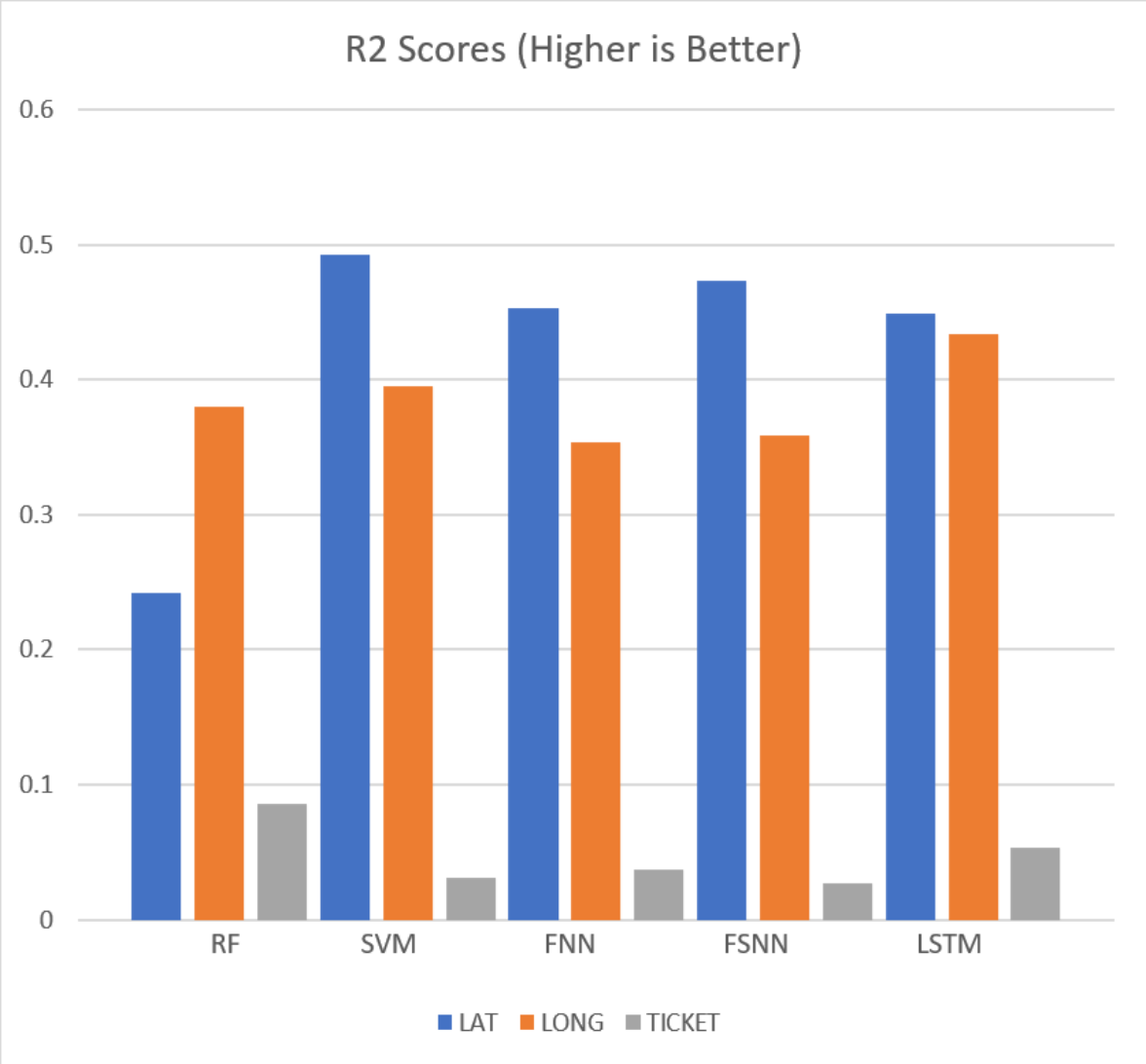

Figure 4.19: R2 Scores

Table 4.1 depicts the change in MSE results compared to those in the previous chapter [4]. There seems to be significant improvement with some mild degradation in the change of the accuracy scores. The notable improvements are for RF, and FSNN in target features Lat and Long.

Table 4.1: Change in MSE Scores Percentage Pre and Post kPCA and New Hyper-Tuning Parameters

ML Approach	**Lat**	**Long**	**Ticket**
RF	86%	31%	-7%
LSTM	-7%	-27%	-5%
FSNN	18%	2%	-20%

Table 4.2 shows the change in R2 results compared to those in the previous chapter [4], where there are significant improvements in the ticket column especially for LSTM staggering an over 23000% improvement. Overall, LSTM improved well across all three target features.

Table 4.2: Change in R2 Scores Percentage Pre and Post kPCA and New Hyper-Tuning Parameters

ML Approach	**Lat**	**Long**	**Ticket**
RF	-90%	-31%	429%
LSTM	4%	63%	23740%
FSNN	-18%	-12%	112%

The negative values seen in Tables 4.1 and 4.2 merely represent the degradation of the models rather than an improvement. However, this is expected as the kPCA applied reduced the size of the data leading to the reduction of the overall performance while improving working time.

G. Final Prediction

In Figure 4.20, we present top 50 non neural network approaches based on the most significant value of the Monthly Median Ticket counts. The blue color indicates the SVM locations, the turquoise indicates RF locations, and the green depicts historical locations. Upon initial analysis, only a single approach is well spread out indicating no particular bias or specific pattern. However, the SVM visually provides an inaccurate generalization as it indicates over-fitting bias through heavy grouped plots just above the downtown core. This does not at all represent the previously-seen scatter plot graphs.

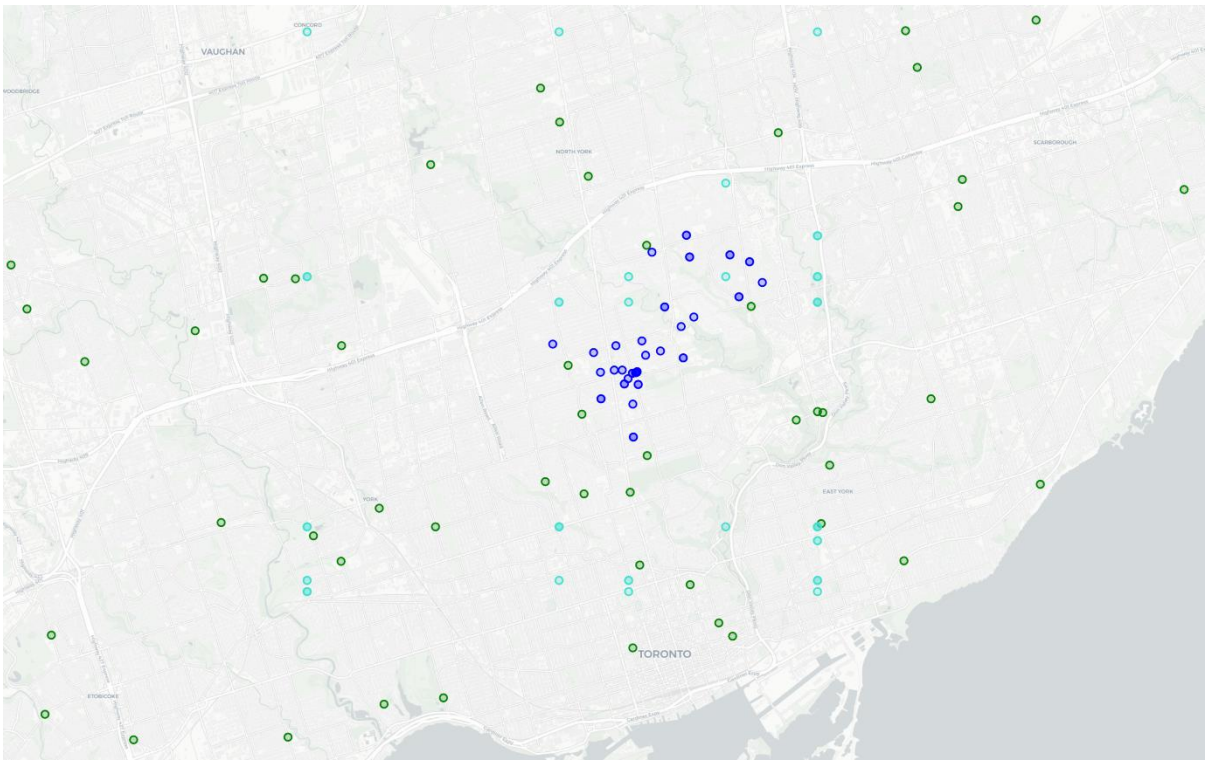

Figure 4.20: Final predictions of the top 100 ASE hotspots for SVM, RF, and Historical

In Figure 4.21, we present the top 50 neural network based on the most significant value of the Monthly Median Ticket counts. The red color represents the FNN locations, the purple color represents the LSTM locations, the orange color represents the FSNN locations, and in green is the historical locations. Upon initial analysis, the FSNN and FNN plot in Lake Ontario indicating under-fitting bias. In addition, the FNN presents a light over-fitting bias through the placements of grouped points. The FSNN provides good generalization towards the north west. Interestingly, LSTM suggests heavy over-fitting with some significant clusters grouped together while also indicating minor under-fitting bias as it places some points in the lake. This behaviour is the exact opposite result seen in the previous chapter where it performed the best.

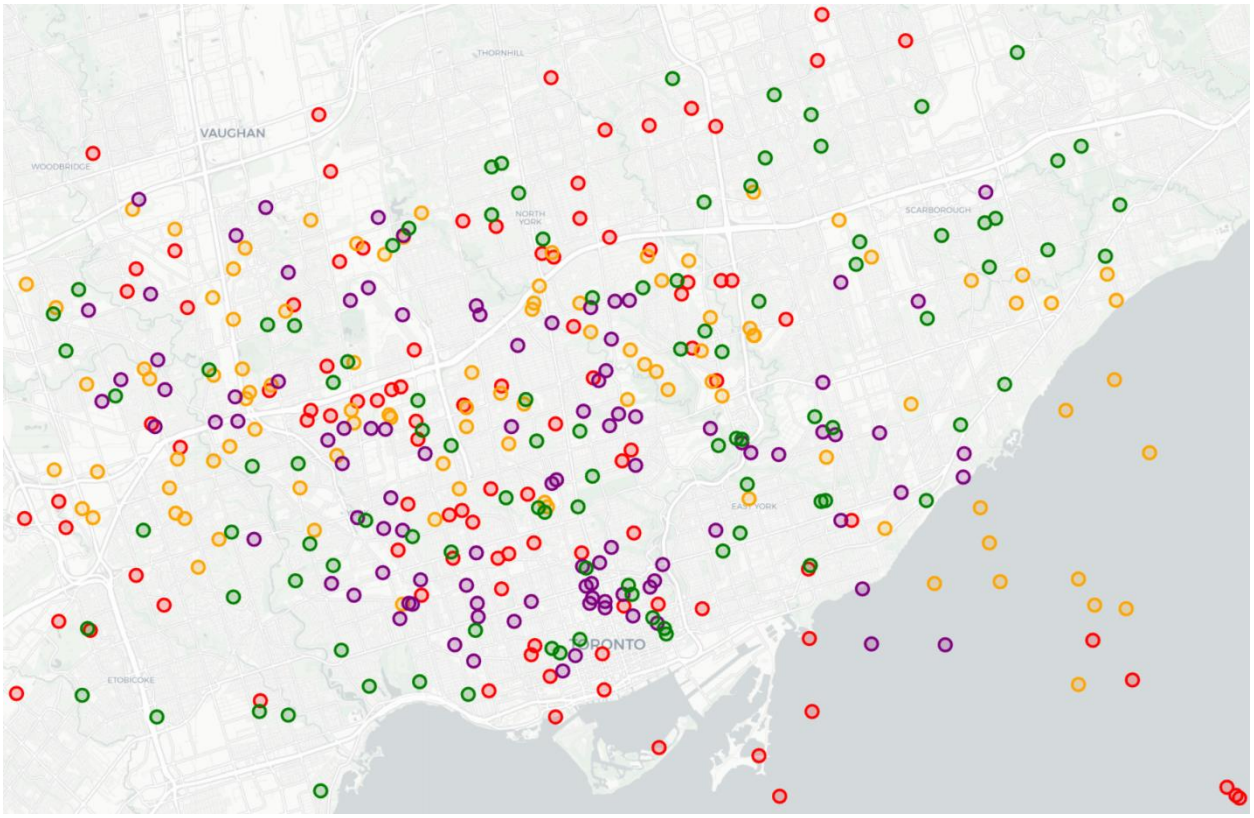

Figure 4.21: Final prediction of the top 100 ASE hotspots for FNN, FSNN, LSTM, and Historical

In Figure 4.22, we present the top 50 best combination of ML of the target features Ticket, Latitude, and Longitude in pink. The best latitude model is the LSTM Lat, the best longitude model is the LSTM Long, and the best ticket model is the FSNN Ticket. This combination of models was selected based on the highest R2 scores, along with an initial visual analysis of previously plotted graphs depicting the strongest relationships between dependent and independent features. Additionally, all of these models achieved a MSE score of 0.05 or lower. Overall, this model has the best generalization of the predictive placement seen in this and the previous chapter, as it spreads towards the east end of the city with no overlapping [4]. Only

2 outliers are plotted, and these are demonstrated through under-fitting bias since they are placed outside the bounds of the city, both found in Lake Ontario.

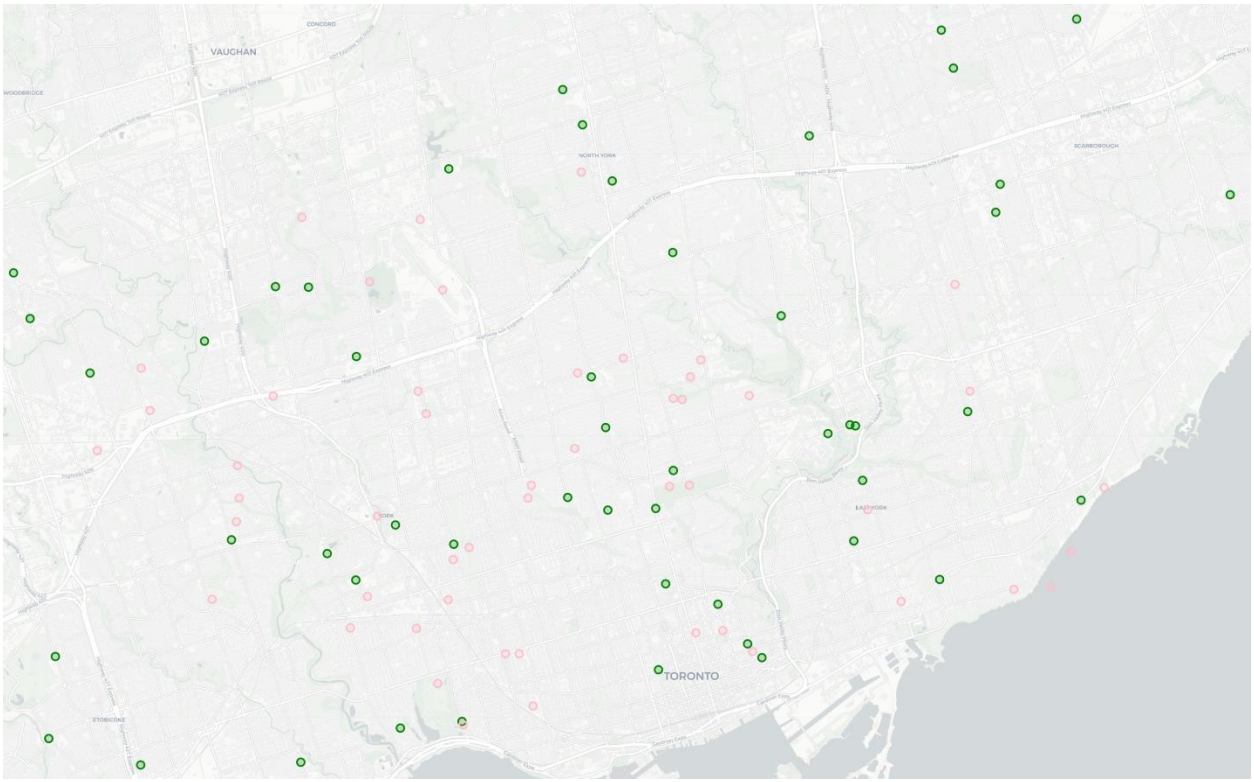

Figure 4.22: Final predictions of the top 100 ASE hotspots of combination of LSTM Lat, LSTM Long, FNN Ticket and additionally Historical

H. Post kPCA Ablation Analysis

In this chapter, we apply SHAP to analyze three previously mentioned machine learning models, assessing the significance of their features post-application of the kPCA. Which has been applied to the dataset, transforms the original feature space into a higher-dimensional space to capture non-linear relationships among the features. This transformation often reveals hidden patterns and enhances model interpretability. Notably, after kPCA transformation, the features are no longer distinguishable in their original form, complicating direct interpretations of their

influence but potentially uncovering more complex interactions. The effects of kPCA on feature significance and model performance are explored through SHAP, providing a deeper understanding of how transformed features impact predictions.

The first SHAP summary plot analyzed is for the FSNN Ticket model which can been seen in Figure 4.23. It demonstrates that the all 17 features had some influence on the model's predictions, far better then what the pre kPCA indicated. Where in particular the top 2 features have the most influence. However, what each feature represents is unknown, therefore it is difficult to infer the reasoning behind the influence, such is the pattern for the rest of the summary plots. In addition, the points on the far edges of the scale, have a value of at most around -0.3 to 0.2, representative of its R2 score.

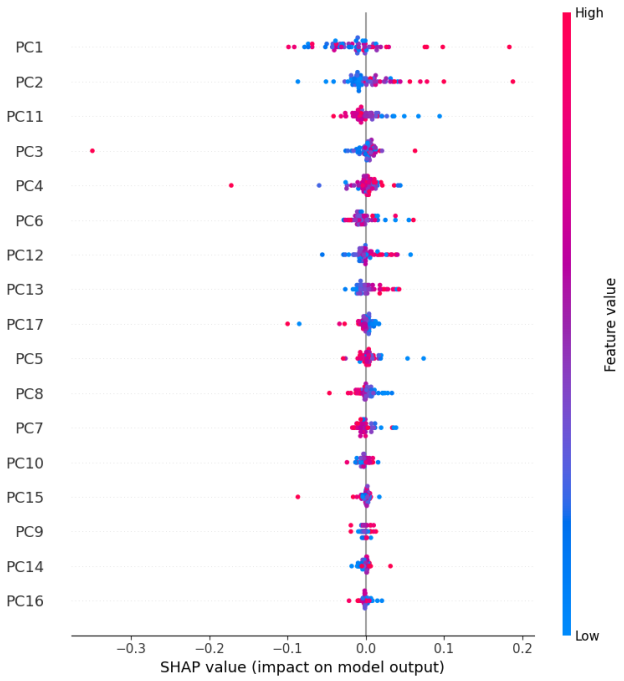

Figure 4.23: FSNN Ticket model SHAP Summary Plot

The second SHAP summary plot analyzed for the RF Latitude model which can been seen in Figure 4.24. It demonstrates that the all 17 features had some influence on the model's predictions, far better then what the pre kPCA indicated. Where in particular the top 4 features have the most influence, where the last feature had minimal. In addition, the points on the far edges of the scale, have a value of at most around -0.15 to 0.1, representative of its R2 score.

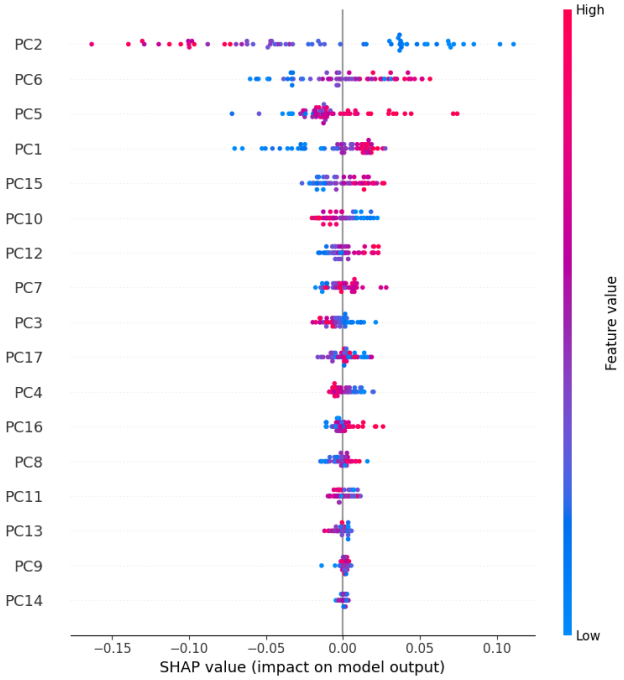

Figure 4.24: RF Latitude model SHAP Summary Plot

The third SHAP summary plot analyzed for the LSTM Longitude model which can been seen in Figure 4.25. It demonstrates that the all 17 features had some influence on the model's predictions, far better then what the pre kPCA indicated. Where in particular the top 4 features have the most influence. However, the points on the far edges of the scale, have a value of at most around -0.015 to 0.01, representative of its poor R2 score.

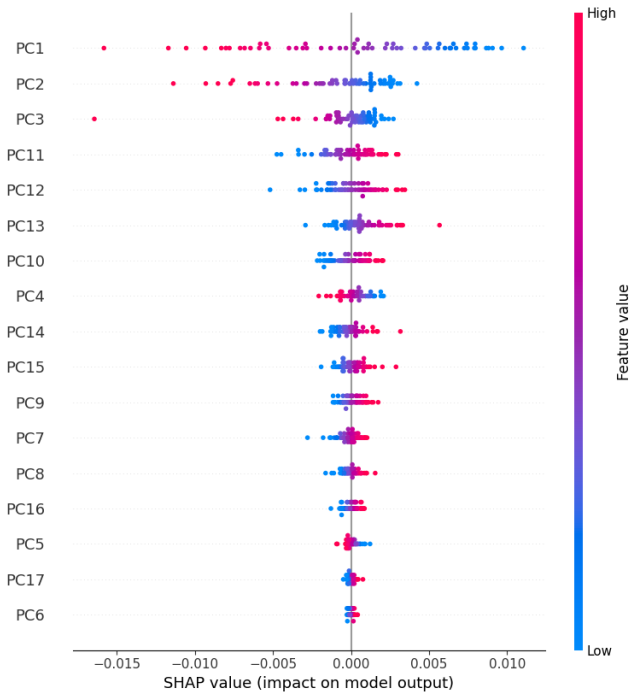

Figure 4.25: LSTM Longitude model SHAP Summary Plot

The fourth SHAP summary plot analyzed for the SVM Ticket model which can been seen in Figure 4.26. It demonstrates that the all 17 features had some influence on the model's predictions. Where in particular the top 9 features have the most influence. However, the points on the far edges of the scale, have a value of at most around -0.01 to 0.04, representative of its poor R2 score.

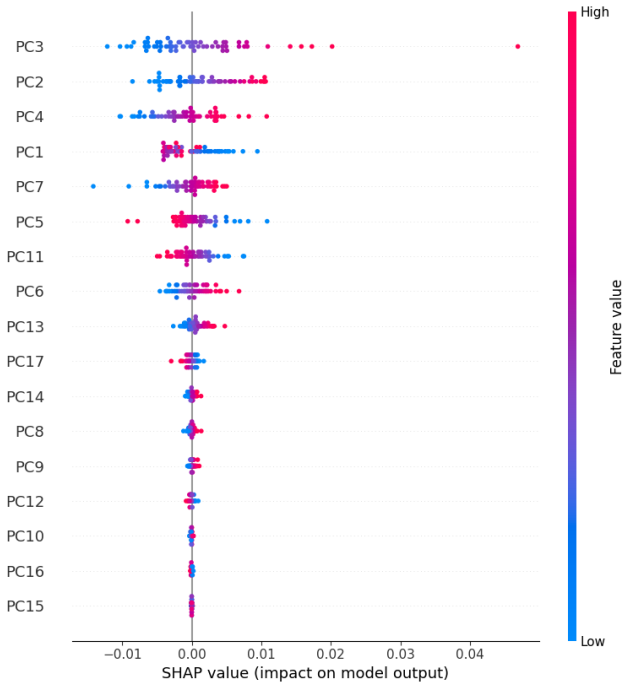

Figure 4.26: SVM Ticket model SHAP Summary Plot

The fifth SHAP summary plot analyzed for the FNN Longitude model which can been seen in Figure 4.27. It demonstrates that the all 17 features had some influence on the model's predictions. Where all but the last feature has had some noticeable level of influence. However, the points on the far edges of the scale, have a value of at most around -0.2 to 0.2, representative of its R2 score.

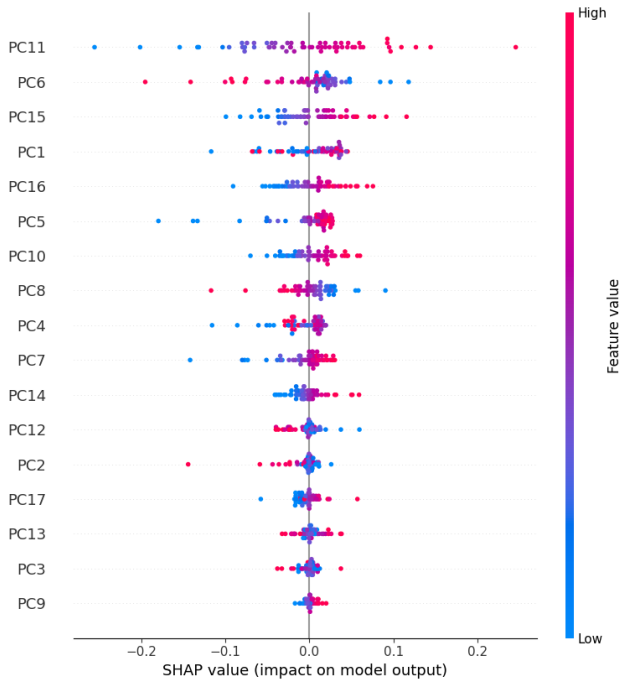

Figure 4.27: FNN Longitude model SHAP Summary Plot

More post kPCA SHAP Summary Plots can be found in the Appendix.

4.5 Implementation and Significance of Study

The imperative of prioritizing safety and accident prevention underscores the significance of strategic sensor placements and predictive technologies, not only within Toronto but also across other urban landscapes. Central to this project is the adoption of an action plan aimed at fostering the integration of high-priority technologies to advance societal well-being. An integral aspect entails robust engagement with citizens, businesses, and stakeholders, necessitating accessible and well-publicized public forums to elucidate the potential and limitations of smart technology. Collaboration with municipal authorities, key stakeholders, and technology providers is indispensable for crafting comprehensive implementation strategies, encompassing timelines, budgets, and resource allocation amid competing priorities. Furthermore, thorough scrutiny encompassing technical, legal, ethical, and sociocultural dimensions is paramount to ensure

seamless progress and adherence to standards. By meticulously navigating these facets, successful execution in Toronto can serve as a blueprint for replication in diverse urban contexts.

In a practical sense, such a blueprint has the design capacity for implementation in municipalities that contains collected data for sensors near identical to what has been used in this chapter. A requirement of an automated ticketing system for speeding, a warning sign for speeding system, both following an internet of things architectural approach, and the historical collection of vehicular crime theft statistics based by neighborhoods or wards. Such is the requirement to effectively deploy the system for other urban environments found national and internationally.

4.6 Summary

As populations grow, more automobiles are being purchased since people aim for comfort. However, this is associated with serious traffic incidents and violations. As a result, people's awareness is hindered, expenses are increased, properties are damaged, and people are either injured or killed. These circumstances necessitate developing traffic violation prediction and detection systems to automate traffic regulations and eliminate unawareness within the human population [20].

This work demonstrates a more specialized method for ASE hotspot prediction for the target urban environment of the City of Toronto. Five machine learning approaches are used: SVM, Random Forest, LSTM, FSNN, and FNN. However, to improve the performance of these models compared to some of its predecessors in the previous work, significant feature engineering and hyper-parameter optimization are applied to the training data set and machine learning approaches. Metric results of MSE and R2 indicate that FNN provides the best overall

performance. When combining the coordinate models of FNN with the ticket model RF, it demonstrates the best observed generalization of these traffic sensor hotspot predictions, placing points east of the downtown core.

5. Cloud-Based Perspective for Intelligent Transportation and Crime Prevention: A Web Deployment Solution

5.1 Introduction

In this study, we aim to develop a more effective strategy for deploying traffic sensors that focus on predicting rather than merely investigating traffic-related crimes. Building upon the foundation laid by the previous preceding [21], we have progressed to the operational deployment of this research in the form of a Software as a Service (SaaS) model, facilitated through an advanced digital platform accessible via the Internet. This evolution signifies a pivotal transition from theoretical exploration to the practical application of the findings, aiming to provide scalable solutions that cater to a broader user base. This work would represent a sense of a cloud tool for a ML pipeline from inputting data to examining final results. This in turn would operate in black-box format such that to minimize the concern for the user the current process.

A. Contributions

The key findings of the research are outlined as follows:

- Here, we provide the user the ability to tune the models and we provide a sample tunning set as a default setting.
- We provide a template to the strategic deployment of traffic monitoring internet of things sensors for world widescale, making it a universally applicable solution to enhance road safety and law enforcement efficiency.

- The research introduces a sophisticated framework designed to decrease both the processing time of models and the duration of predictions, while maintaining high levels of accuracy. This ensures the framework operates efficiently without sacrificing performance quality.
- We have created a user-friendly graphical user interface (GUI) that enables government officials, law enforcement personnel, and researchers to easily upload data and identify areas with high and low traffic crime rates, regardless of their technical background. This tool is designed to be accessible and straightforward, facilitating data analysis and decision-making for users from various professional backgrounds
- We have incorporated a cloud-based Dynamic Load Balancing (DLBA) [22] strategy, mindful of multithreading and parallel processing capabilities, into the primary computational server. This approach is designed to decrease the total processing time by efficiently distributing tasks across multiple processing units.
- We propose the initiation of an elaborate and detailed comparative analysis, encompassing a range of distinct modeling techniques This analysis aims to be provided in a report like form factor, allowing users to comprehensively evaluate the efficiency, accuracy, and performance of their results.

5.2 Problem Statement

Traffic infractions and crimes have a detrimental impact on both individuals and the broader community. Engaging in behaviors like speeding, reckless driving, and ignoring red lights escalates the likelihood of accidents, resulting in injuries, deaths, and property destruction. Furthermore, committing traffic offenses can result in fines, points on a driver's license, and in

certain cases, imprisonment. Drivers found guilty of traffic crimes may also experience an increase in their insurance premiums, making the cost of driving significantly higher [14]. Therefore, efforts to reduce traffic violations are crucial for the improvement of societal conditions. However, not everyone is capable of having access to a system that could assist in such a manner. Henceforth a cloud-based system, such as a Software as a Service (SaaS) is needed, but to prevent overload a properly implemented load balancing must handle the incoming requests especially if its a system with limited resources.

In the pursuit of lowering traffic crime rates, several challenges and limitations are encountered. These include the availability and appropriateness of data, the selection of the most effective models and algorithms, defining the ultimate goal (whether for detection or prediction), and the feasibility of practical application. To address these challenges, we build upon the previous research [21], aiming to explore four key areas: the finalization machine learning methods, the optimized hyper-parameter tuning specific to each method's architecture, the introduction of load balancing to handle job requests, the creation of a user-friendly graphical interface (GUI) to enhance user interaction while employing a black box approach to simplify complexity, and offering of this framework as cloud service in the format of a SaaS.

5.3 Methodology

A. Optimized Hyperparameter Tunnig

An optimized set of parameters was identified to achieve the most favorable Mean Squared Error (MSE) and R2 scores across various machine learning models. With the transition of this project into a production environment, there arises a critical need to further refine or optimize the set of parameters for hyperparameter tuning. The production context is characterized by its demand

for high throughput and reduced operational time, necessitating a streamlined approach to parameter testing.

To facilitate the integration of this parameter optimization process within production environments, several assumptions have been posited:

- It is presumed that the incoming merged and preprocessed data maintains a confidence level of 99%, ensuring a high degree of reliability in the data inputs.
- Parameters that have a direct impact on the architecture of the machine learning models remain unchanged from their optimized values identified in prior analyses. For example, the number of hidden intermediate layers is fixed at two, and the activation function is designated as "Adma". This approach ensures stability in the model architecture while allowing for adjustments in other areas.
- Hyperparameter tuning efforts are specifically focused on parameters that do not alter the model's architecture, such as batch size and the number of epochs. This constraint is intended to balance the necessity of model optimization with the practicalities of production constraints.

In addition to, if the default set of parameters are not up to par with advanced user, they may opt in to adjust what test to their objective within reason. Such that the tunning of the hyperparameters would be allowed in the cloud-based tool.

B. Data Sets

As elucidated in the preceding work, the datasets utilized in this study were sourced from the public domain, courtesy of the City of Toronto [15]. However, it is important to acknowledge that such data availability may not be uniform across all governing municipalities. To address

this variability, it is imperative to classify the incoming datasets into three distinct categories: (1) the primary sensor dataset, which is earmarked for prediction purposes as depicted in Table 6.1; (2) a comprehensive volumetric sensor dataset aimed at maximizing the identification of potential locations, as illustrated in Table 6.2; and (3) a dataset encompassing neighborhood crime statistics pertinent to vehicular-related offenses, as shown in Table 6.3 where each data instance in each dataset is annotated with GPS coordinates or descriptive location. This methodological stratification is essential for ensuring the adaptability and applicability of the study's findings across diverse municipal contexts.

Table 5.1: Snippet of ASE median monthly tickets dataset features

Site Code	Location*	Enforcement Start Date	Enforcement End Date	Jul-20	Aug-20	Sep-20	Oct-20
A001	Royalcrest Rd. Near Cabernet Circle	06-Jul-20	11-Nov-20	1,117	479	487	354
A002	Harefield Dr. Near Barford Rd.	06-Jul-20	25-Oct-20	43	21	10	16
A003	Renforth Dr. Near Lafferty St.	06-Jul-20	25-Oct-20	2,428	1,665	1,037	268

Table 5.2: Snippet of WYS volumetric dataset features

_id	sign_id	month	pct_05	pct_70	pct_75	pct_80	pct_85	pct_90	pct_95	spd_00	spd_05	spd_10	spd_95	spd_100_a	volume
244847	3	2020-09-01	9	25	27	28	31	33	37	0	5023	8418	0	0	83065
244848	9	2020-09-01	5	22	23	24	25	27	29	0	17232	3264	0	0	36154
244849	10	2020-09-01	5	23	25	27	29	30	33	0	748	423	0	0	2986
244850	15	2020-09-01	19	43	44	46	48	51	55	0	1204	1873	5	1	144494

Table 5.3: Snippet of Neighborhood crime statics with vehicular features highlighted

_id	OBJECTID	HoodName	HoodID	AutoTheft	AutoTheft_2021	AutoTheft_Rate2014	Theftfrom	Theftfrom	geometry
1	1	Yonge-St.Clair	97	2	14	15.8768	8	63.50718	{'type': 'Polygon', 'coordinates': (((-79.3911775309057, 43.6810827793252), (
2	2	York University Heights	27	109	172	379.5265	142	494.429	{'type': 'Polygon', 'coordinates': (((-79.5052740392145, 43.759873278756), (-
3	3	Lansing-Westgate	38	18	60	111.8985	22	136.7649	{'type': 'Polygon', 'coordinates': (((-79.4399692149635, 43.7615581933539), (
4	4	Yorkdale-Glen Park	31	67	92	440.8765	105	690.9258	{'type': 'Polygon', 'coordinates': (((-79.4396716176683, 43.7056110727847), (

C. Graphical User Interface

Figure 6.1 showcases the fully developed graphical user interface (GUI) that effectively caters to user interactions within predetermined constraints, offering visual feedback mechanisms. These include a progress bar and the presentation of final predicted outcomes on a

dynamic, scalable map. This interactive map allows users to thoroughly explore the results by manipulating the view. Users have the capability to upload their data through excel or csv files, which the system then processes. Upon file selection, the user can initiate the system and choose a preferred type of graph, selecting from options such as the best model combination, exclusively neural networks, or solely non-neural networks.

This GUI is designed to deliver a seamless user experience, functioning as a comprehensive solution that simplifies complex processes for users with varying degrees of technical expertise, including stakeholders, government officials, or researchers. The interface operates as a sophisticated tool, presenting complex machine learning operations as accessible and manageable tasks.

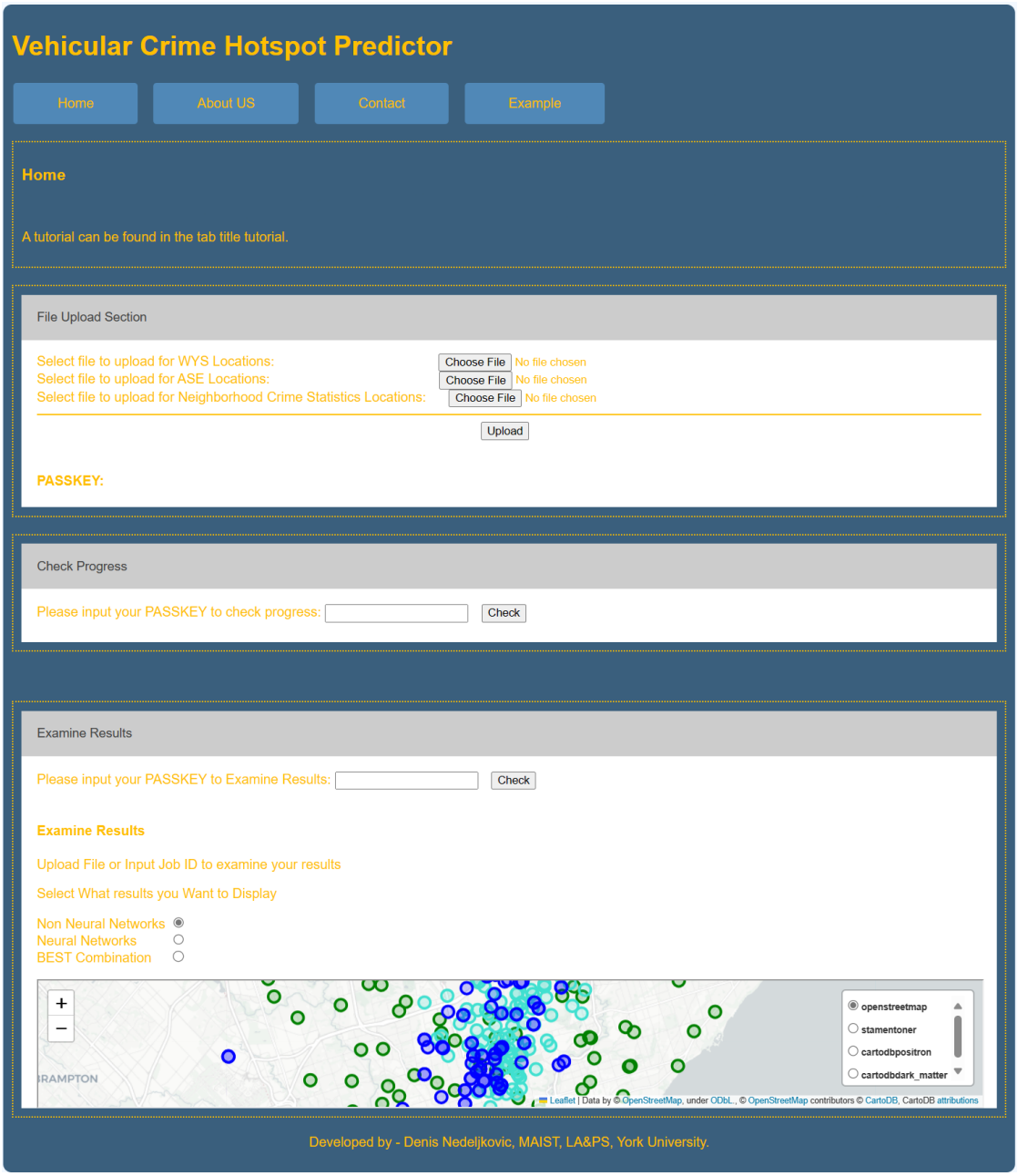

Figure 5.1: Graphical user interface for production

D. Smart Framework

Figure 6.2 presents a schematic representation of the advanced smart framework, which is compartmentalized into three primary segments: data pre-processing (highlighted in blue), hyperparameter tuning (depicted in green), and the ultimate generation of the report alongside its

associated graphical representations. To optimize performance, the framework eschews the development of a singular, monolithic model for predicting three target features. Instead, it adopts a more tailored approach, assigning a distinct model for each target feature within the specified machine learning methodology. This stratified design philosophy ensures a precision-focused and efficient predictive outcome for each target variable.

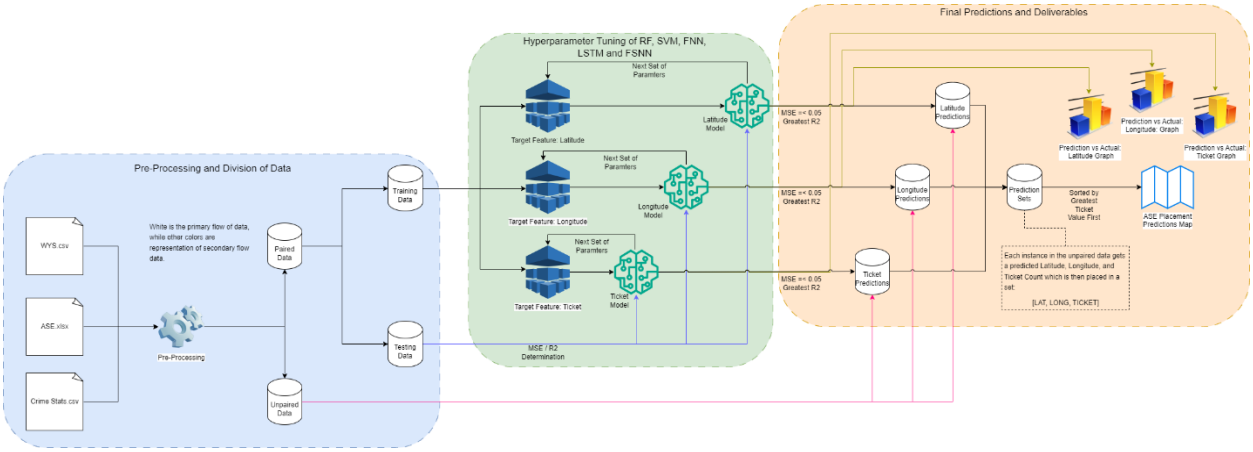

Figure 5.2: The flow chart on how the the system operates

E. Load Balancing

In the precious work, we highlighted the need for efficient task distribution to manage load effectively, crucial for training complex machine learning models. The strategy here involves three key approaches: DLBA, multithreading, and parallel processing, catering to tasks like optimizing deep learning, SVM, and Random Forest models.

DBLA adjusts resources in real-time to prevent overload, optimize hardware use and enhance system reliability [22]. Multithreading speeds up jobs by running multiple subtasks

concurrently, ideal for quick data analysis from traffic sensors. Parallel processing distributes tasks across multiple cores, speeding up complex algorithm processing and large data handling.

Together, these approaches ensure the computational infrastructure supports intelligent transportation systems effectively, enabling faster and more accurate traffic management and safety measures. For a clearer insight into how this is implemented, Figures 6.3 and 6.4 illustrate the division of data within the hardware architecture and provide a surface level flowchart of the process.

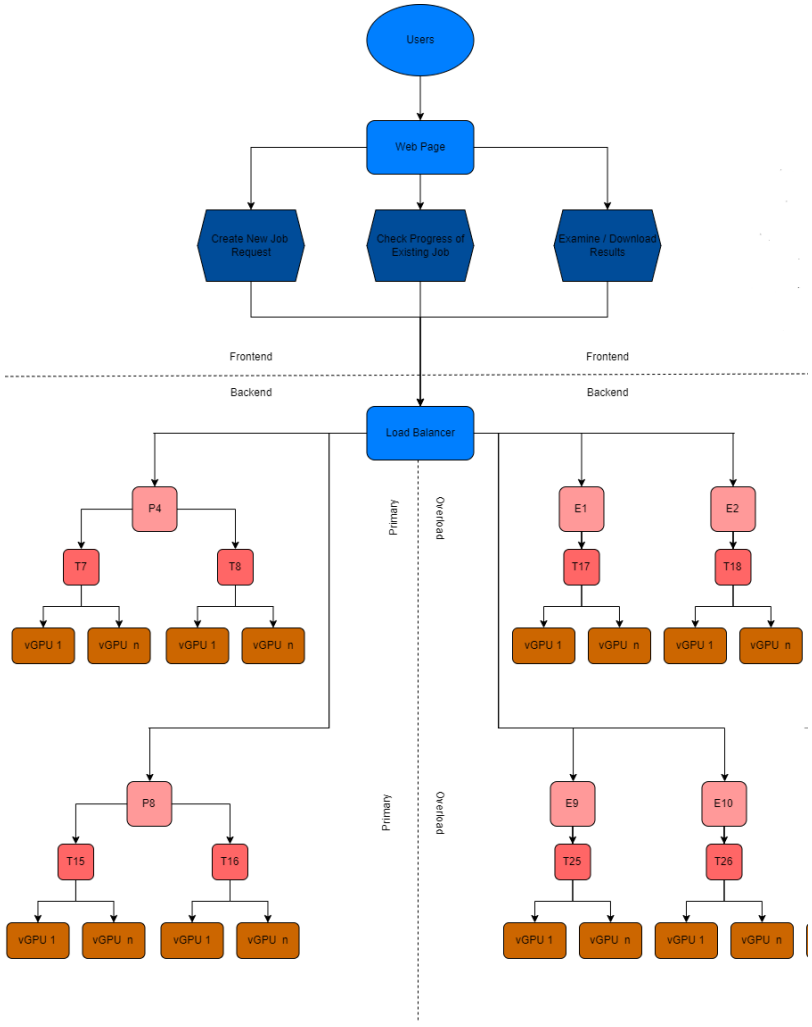

Figure 5.3: A snippet of the low-level overview in a top-down architecture of the current system

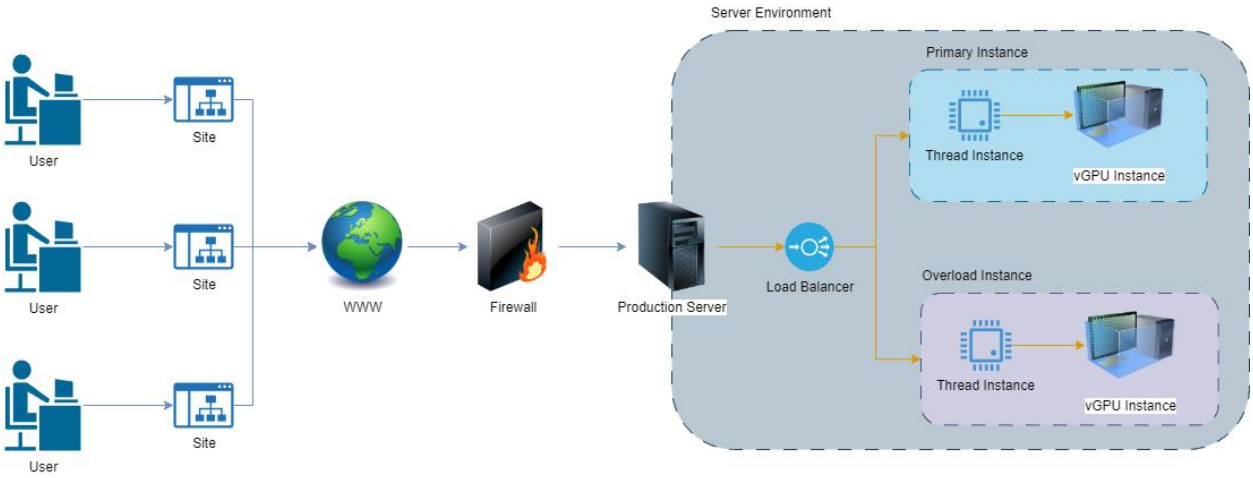

Figure 5.4: A high level overview in a left to right diagram of the current system

5.4 Evaluation

The overarching aim of this chapter is to articulate the design, development, and implementation of a sophisticated Graphical User Interface (GUI) that strives to be universally accessible, providing a seamless and enriching experience for users of diverse technical backgrounds. This interface is pivotal in enabling users to harness predictive machine learning technologies for deeper analytical insights. The GUI is meticulously crafted to serve the distinct needs of its primary audience, which includes researchers, government officials, and law enforcement agencies, while also accommodating the interests of stakeholders within the private sector. The structure of the GUI is segmented into four main components, each designed to facilitate specific aspects of user interaction and engagement with the tool:

- Home Page: Positioned as the epicenter of the research tool, the Home page is where the tool's functionalities come to life. It is equipped with three principal features aimed at optimizing user interaction:

- File Upload Section: This feature allows users to upload three essential files, following which they are assigned a unique passkey. This passkey serves as an identifier for their specific job request, ensuring a personalized and secure user experience. Advanced users with a high degree of technical knowledge may change a set limit of parameters to fit their objectives.
- Progress Monitoring: Users can enter their passkey at any given time to access real-time updates regarding the status of their submission. This includes information on the percentage of completion and an estimated timeline for completion, thereby keeping users informed and engaged throughout the process.
- Results Examination: This critical feature offers two distinct methodologies for reviewing outcomes. Users can either input their previously received passkey to access interactive maps and data visualizations—allowing for an in-depth analysis of predictive outcomes across various models (including neural networks, traditional machine learning algorithms, or a hybrid approach) and historical data—or opt to upload results in an HTML format. This bifurcated approach is specifically designed to cater to users with varying levels of technical expertise, ensuring that the tool is accessible and user-friendly

- About Us Page: This section serves as an informational hub, offering an exhaustive overview of the project's lifecycle—from its conceptualization to its fruition, and every pivotal moment in between. It not only sheds light on the purpose and ambitions of the tool but also introduces users to the minds behind its creation, fostering a sense of community and transparency.

- Contact Page: Recognizing the importance of user feedback and collaboration, the Contact page is dedicated to facilitating open lines of communication between the users and the development team. It is designed as a platform for addressing technical queries, resolving issues, and exploring potential collaborations, whether with individuals or organizations, thereby nurturing a collaborative ecosystem.
- Example Page: With a focus on user education and empowerment, the Example page is crafted to guide users through the operational nuances of the tool. It features detailed step-by-step instructions, specifies input file formats, and includes a video demonstration to ensure that users are well-prepared to leverage the tool to its full potential.

A huge improvement in the overall processing time, Fig. 4.3. Low level top-down architecture snippet because of the implemented load balancing, parallel processing and multi threading. we saw on average a CPU intensive work load for those classic ML models, a significant improvement of about 5900% with an average single thread time of 300 hours vs an average of 5 hours with this method [21]. Work loads that are GPU intensive had significant improvement as well where it was 938% with an average single thread time of 670 hours vs an average of 64 hours in a using the method and incorporating vGPUs. This framework demonstrates great potential in scalability, just adding additional physical GPUs, we could greatly improve the processing times, not to mention that an implementation of some form distributed computing would be essential for large scale production

The formula used to determine percentage improvement:

$$Improvement = \left(\frac{CPU\ Time - Real\ Time}{Real\ Time}\right) x\ 100 \qquad \textit{Equation 14}$$

Where CPU Time represent the single thread time and the Real Time represents this method.

By integrating these components into a coherent and user centric GUI, the project aims to democratize access to advanced machine learning predictions. This not only enhances data-driven decision-making capabilities among key stakeholders but also paves the way for innovative research and strategic initiatives across various areas of interest. Through this comprehensive approach, the GUI stands as a statement to the project's commitment to accessibility, usability, and technological advancement. In extension to this, this work can be explained as a SaaS simply how would be providing access to the custom software for other to use without having them install it to their local machines.

5.5 Conclusion

This study presents a groundbreaking approach to the strategic deployment of traffic sensors, leveraging advanced machine learning techniques and a Software as a Service (SaaS) model to predict traffic-related crimes effectively. By integrating data from Automated Speed Enforcement, Toronto Police crime statistics, and "Watch Your Speed" signs, we have developed a framework that not only predicts future crime hotspots with high accuracy but also offers a scalable, universally applicable solution for enhancing road safety and law enforcement efficiency across the globe.

Key achievements include the development of an optimized machine learning model architecture through rapid hyperparameter tuning, leading to quicker and more accurate predictions. we have also introduced a user-friendly graphical user interface (GUI) that simplifies complex data analysis for users of all technical backgrounds, promoting wider adoption and ease of use. Furthermore, the incorporation of Dynamic Load Balancing significantly improves the efficiency and reliability of the system.

In conclusion, this study marks a significant advancement in the use of technology to address traffic-related issues, offering a blueprint of a cloud tool in a ML pipeline for future developments in the field of intelligent transportation systems and smart city initiatives.

6. Conclusion and Future Directions

In this thesis, we have delved into the transformative potential of machine learning in enhancing urban safety and traffic management, with a specific focus on the strategic placement of sensors. The comprehensive research methodology, grounded in the analysis of urban data from Toronto, has provided significant insights into the optimization of sensor networks for crime detection and traffic control. This study not only underscores the pivotal role of machine learning in urban planning but also opens new avenues for the application of advanced analytics in public safety strategies.

By leveraging sophisticated machine learning algorithms, we have demonstrated the feasibility of predicting high-risk areas for traffic violations and crime, thereby enabling more targeted and effective urban safety measures. The empirical evidence presented supports the argument for integrating technology-driven solutions into urban infrastructure, highlighting the benefits of such approaches in terms of resource allocation and policy effectiveness.

Furthermore, this research contributes to the broader discourse on smart cities by showcasing how data-driven decision-making can lead to more resilient and responsive urban environments. It calls for a collaborative approach among urban planners, policymakers, and technologists to harness the power of machine learning in crafting cities that are not only safe but also adaptable to the evolving needs of their inhabitants.

As we look to the future, the scalability of this model suggests its applicability beyond Toronto to other urban contexts, promising a global impact on urban safety and traffic management practices. However, the journey does not end here. The rapidly advancing field of

machine learning presents ongoing opportunities for refinement and innovation in sensor placement strategies.

6.1 Future Directions

Expanding on the notion of future research, it becomes crucial to delve deeper into the integration of real-time data feeds, which would significantly enhance the responsiveness of urban safety and traffic management systems to dynamic urban environments. The incorporation of more diverse data sets, including socio-economic, environmental, and transportation data, would provide a holistic view of urban dynamics, facilitating more nuanced and effective interventions. Additionally, the development of more sophisticated predictive models, possibly through the application of deep learning and artificial intelligence, promises to refine the understanding and prediction of urban phenomena. To such of an extent that direct integration with an ITS system to provide frequent updates with live data to result in far more up to date predictions. Hence forth a direct collaboration with the City of Toronto would provide the utmost benefit to this research potential having access to greater amounts of data and resources. This progression towards more advanced, data-driven solutions will not only improve urban safety and efficiency but also contribute to the creation of more humane, sustainable, and adaptable urban spaces. This research direction emphasizes the importance of interdisciplinary collaboration and technological innovation in shaping the future of urban living.

7. Bibliography

[1] Junger, M., West, R., Timman, R. (2001). Crime and Risky Behavior in Traffic: An Example of Cross-Situational Consistency. Journal of Research in Crime and Delinquency, 38(4), 439–459.

[2] Bavcevic, Z., Harinam, V. (2020) Targeting Fatal Traffic Collision Risk from Prior Non-Fatal Collisions in Toronto. Camb J Evid Based Polic 4, 187–201. https://doi.org/10.1007/s41887-020-00054-z

[3] P. Infante et al., "Prediction of road traffic accidents on a road in Portugal: A multidisciplinary approach using artificial intelligence, statistics, and Geographic Information Systems," Information, vol. 14, no. 4, p. 238, Apr. 2023. doi:10.3390/info14040238

[4] X. Chen et al. (2020). "Sensing Data Supported Traffic Flow Prediction via Denoising Schemes and ANN: A Comparison," in IEEE Sensors Journal, vol. 20, no. 23, pp. 14317-14328

[5] M. Sandim, R. J. F. Rossetti, D. C. Moura, Z. Kokkinogenis and T. R. P. M. Rubio, (2016). "Using GPS-based AVL data to calculate ´ and predict traffic network performance metrics: A systematic review," 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 1692-1699

[6] U. Yildirim and Z. Cataltepe. (2008). "Short time traffic speed prediction using data from a number of different sensor locations," 2008 23rd International Symposium on Computer and Information Sciences, , pp. 1-6

[7] J. Tang, J. Hu, W. Hao, X. Chen and Y. Qi, ”Markov chains based route travel time estimation considering link spatio-temporal correlation”, Phys. A Stat. Mech. Appl., vol. 545, May 2020.

[8] R. E. AlMamlook, K. M. Kwayu, M. R. Alkasisbeh and A. A. Frefer, ”Comparison of Machine Learning Algorithms for Predicting Traffic Accident Severity,” 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Amman, Jordan, 2019, pp. 272-276, doi: 10.1109/JEEIT.2019.8717393.

[9] Mostafa, S.A. et al (2022). A Machine Learning-Based Data Fusion Model for Online Traffic Violations Analysis. International Conference on Innovative Computing and Communications. Advances in Intelligent Systems and Computing, vol 1394

[10] Moses, T., Obi, H. E.(2022). Review on Automobile Crime Prediction Model. International Journal of Darshan Institute on Engineering Research and Emerging Technologies, 11(1), 08-13.

[11] Nama, M et al. (2021). Machine learning-based traffic scheduling techniques for intelligent transportation system: Opportunities and challenges. International Journal of Communication Systems, 34(9), e4814.

[12] P. Gokhale, O. Bhat, and S. Bhat, “Introduction to IOT,” *International Advanced Research Journal in Science, Engineering and Technology*, vol. 5, no. 1, Jan. 2018. doi:DOI: 10.17148/IARJSET.2018.517

[13] G. Dimitrakopoulos and P. Demestichas, “Intelligent Transportation Systems,” *IEEE Vehicular Technology Magazine*, vol. 5, no. 1, pp. 77–84, Mar. 2010. doi:10.1109/mvt.2009.935537

[14] NHTSA. Speeding. Retrieved from https://www.nhtsa.gov/riskydriving/speeding

[15] Transportation Services(2023). About Automated Speed Enforcement (ASE) Charges.https://open.toronto.ca/dataset/automated-speed-enforcement-ase-charges/

[16] Toronto Police Services (2022). About Neighbourhood Crime Rates. https://open.toronto.ca/dataset/neighbourhood-crime-rates/

[17] Transportation Services. About Safety Zone Watch Your Speed Program – Monthly Summary (2023).https://open.toronto.ca/dataset/safety-zone-watch-your-speed-program-monthly-summary

[18] K. Esmukov, “GeoPy,” GitHub. 2022

[19] Y.-M. Wang and L.-J. Huang, “Fourier series neural networks for regression,” 2018 IEEE International Conference on Applied Sy

[20] CR-T. (2022). How to Implement New Technology in Your Organization. Retrieved form https://www.cr-t.com/blog/how-toimplement-new-technology-in-your-organization/:˜:text=Step%203.- ,Develop%20a%20Plan%20for%20Implementing%20Your%20New%20Technology.,details%2C%20and%20manage%20conflicting%20priorities

[21] D. Nedeljkovic, N. Y. Fares, and M. Jammal, “A Novel Machine Learning-based Approach to City Crime Sensor Placement Prediction,” IEEE World Forum on Internet of Things, vol. 9, 2023.

[22] M. Kumar and S. C. Sharma, “Dynamic load balancing algorithm to minimize the makespan time and utilize the resources effectively in cloud environment,” International Journal of Computers and Applications, vol. 42, no. 1, pp. 108–117, Nov. 2017. doi:10.1080/1206212x.2017.1404823

8. Appendix

A. Additional Figures

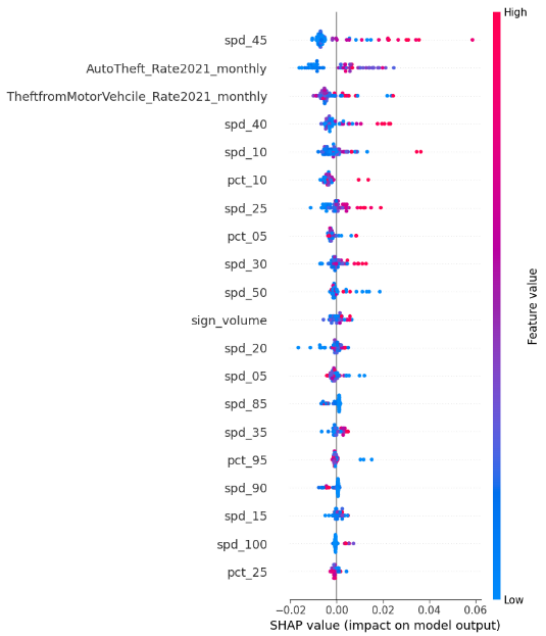

Figure 8.1: Pre kPCA RF Ticket model SHAP Summary Plot

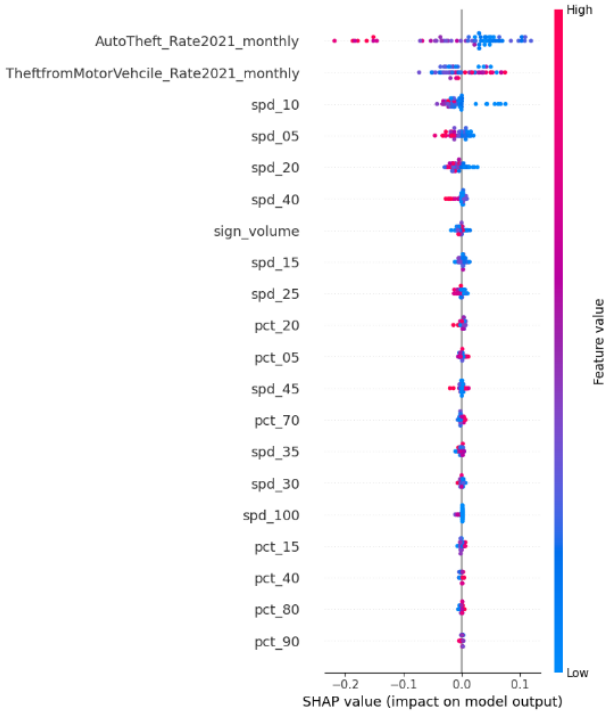

Figure 8.2: Pre kPCA RF Longitude model SHAP Summary Plot

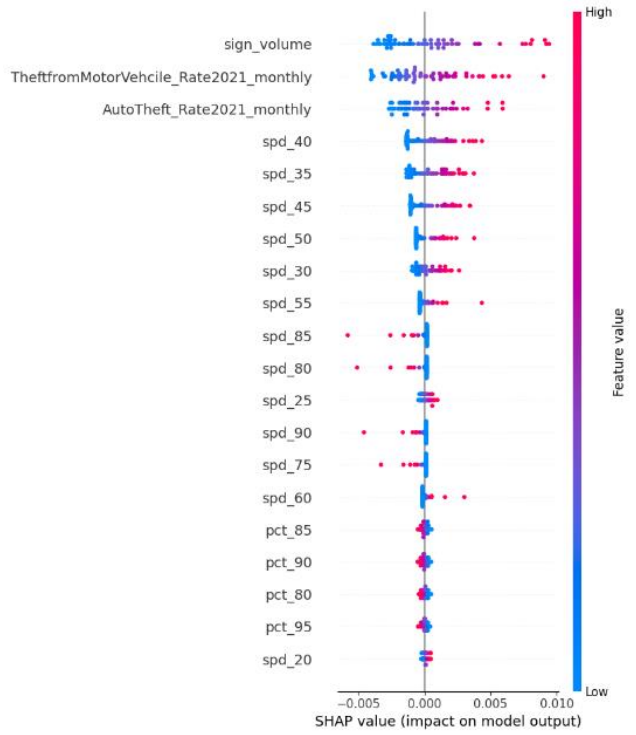

Figure 8.3: Pre kPCA LSTM Ticket model SHAP Summary Plot

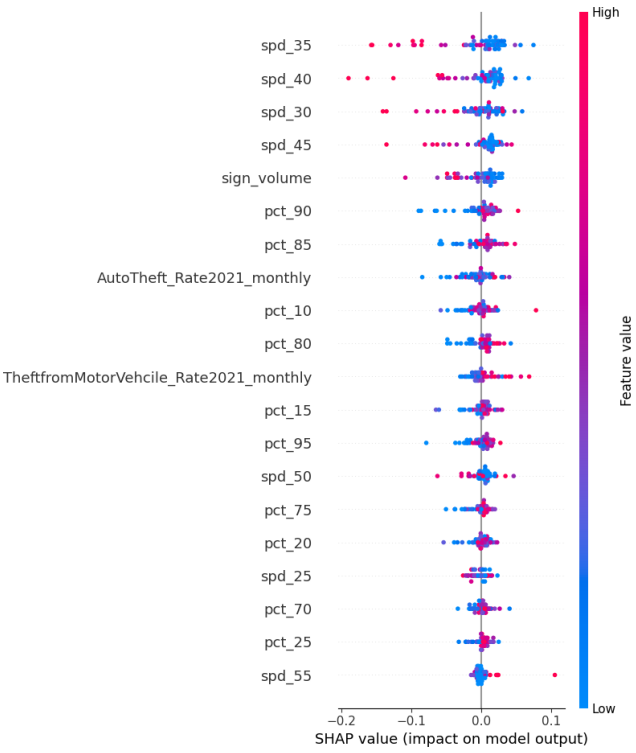

Figure 8.4: Pre kPCA LSTM Latitude model SHAP Summary Plot

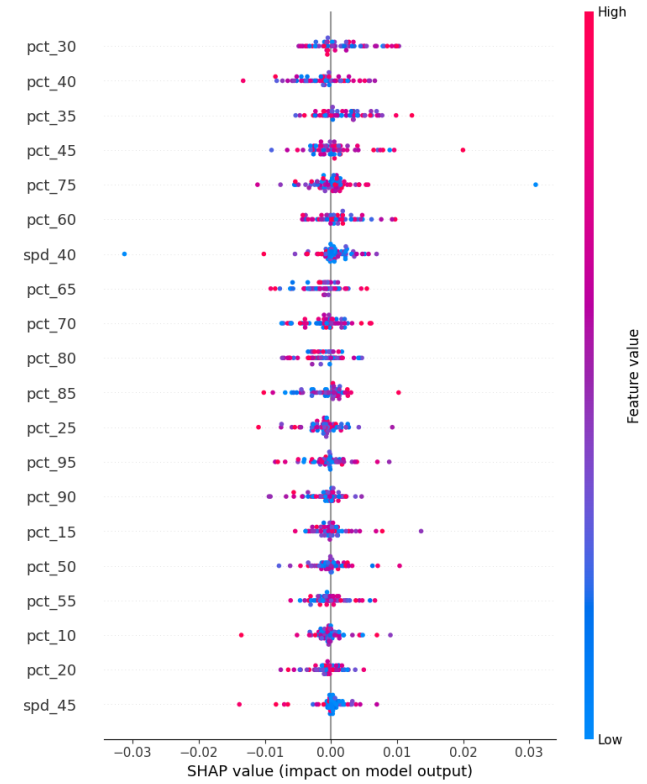

Figure 8.5: Pre kPCA FSNN Longitude model SHAP Summary Plot

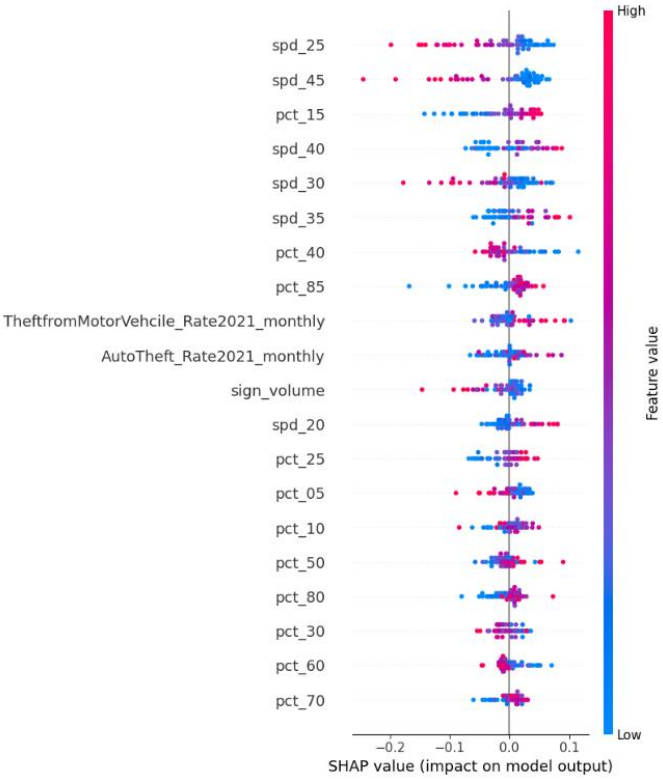

Figure 8.6: Pre kPCA FSNN Latitude model SHAP Summary Plot

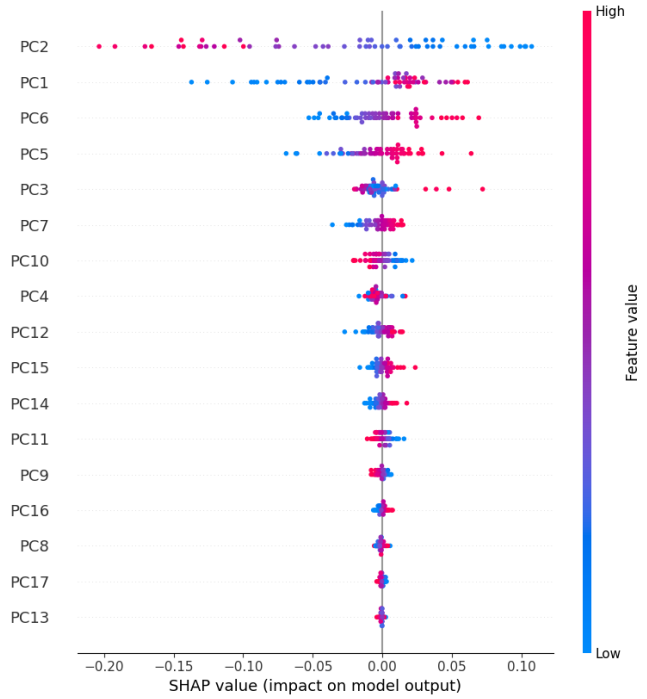

Figure 8.7: Post kPCA SVM Latitude model SHAP Summary Plot

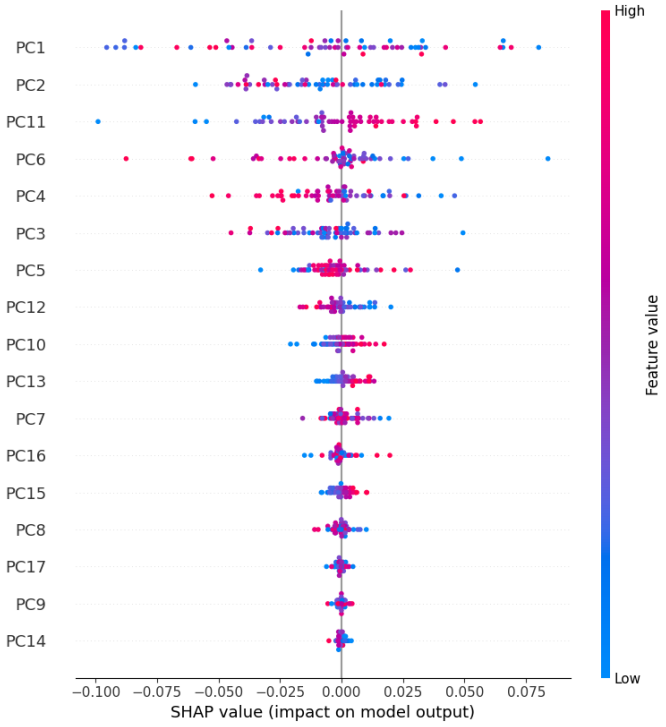

Figure 8.8: Post kPCA SVM Longitude model SHAP Summary Plot

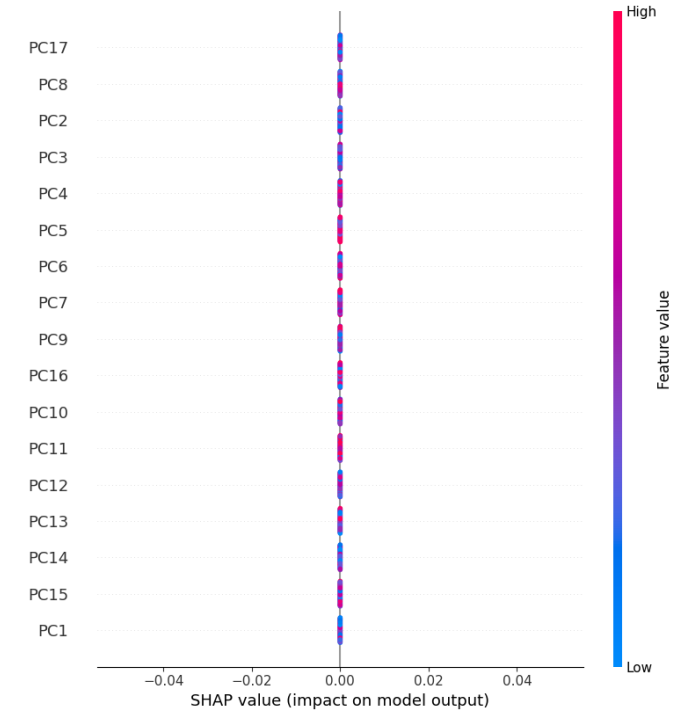

Figure 8.9: Post kPCA RF Ticket model SHAP Summary Plot

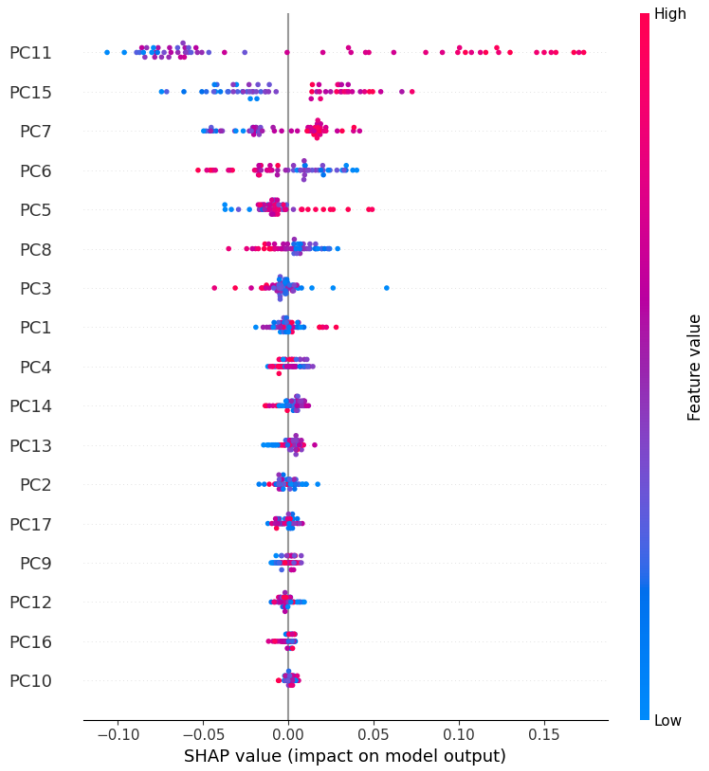

Figure 8.10: Post kPCA RF Longitude model SHAP Summary Plot

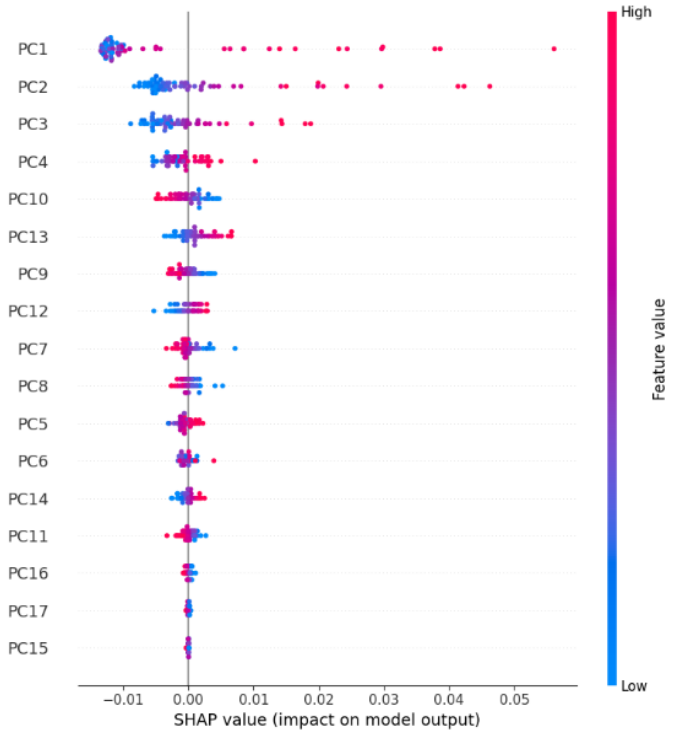

Figure 8.11: Post kPCA LSTM Ticket model SHAP Summary Plot

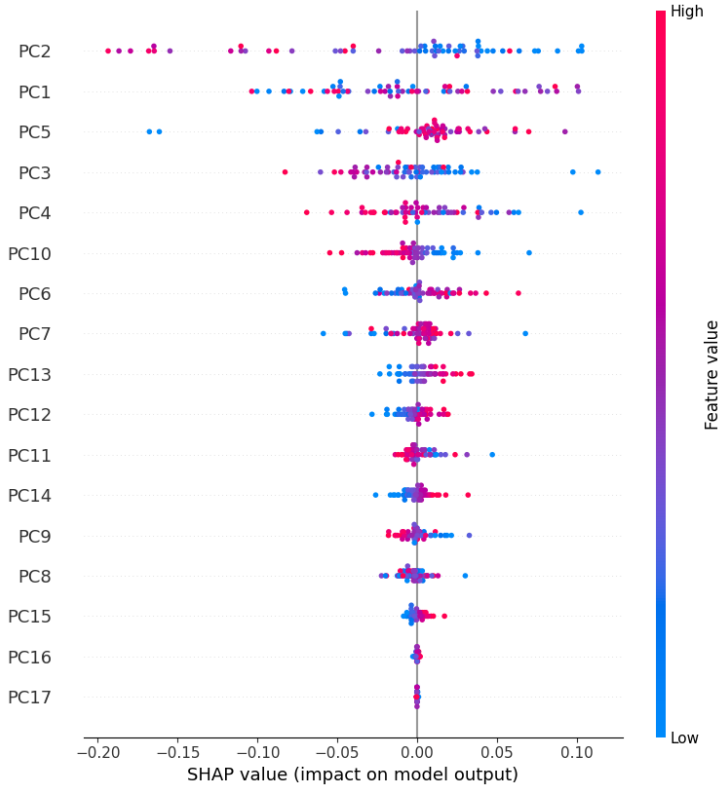

Figure 8.12: Post kPCA LSTM Latitude model SHAP Summary Plot

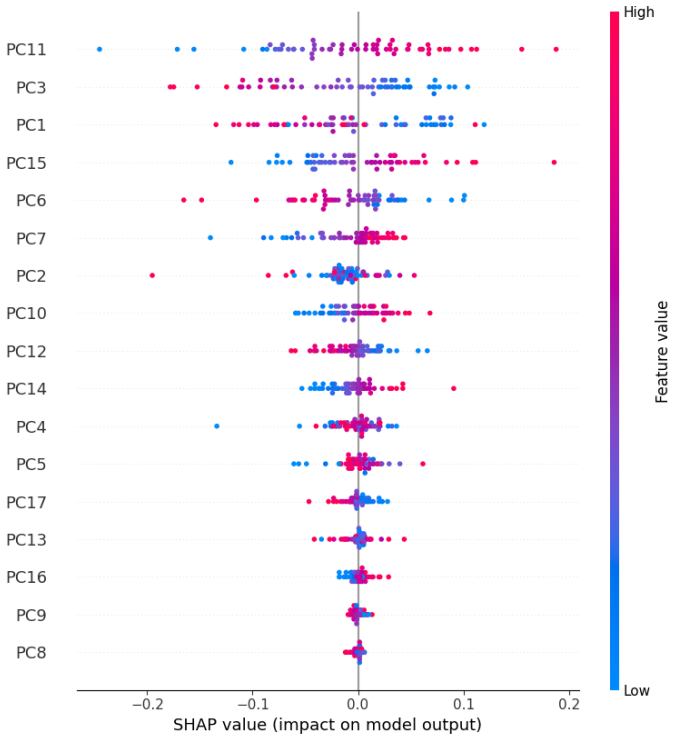

Figure 8.13: Post kPCA FSNN Longitude model SHAP Summary Plot

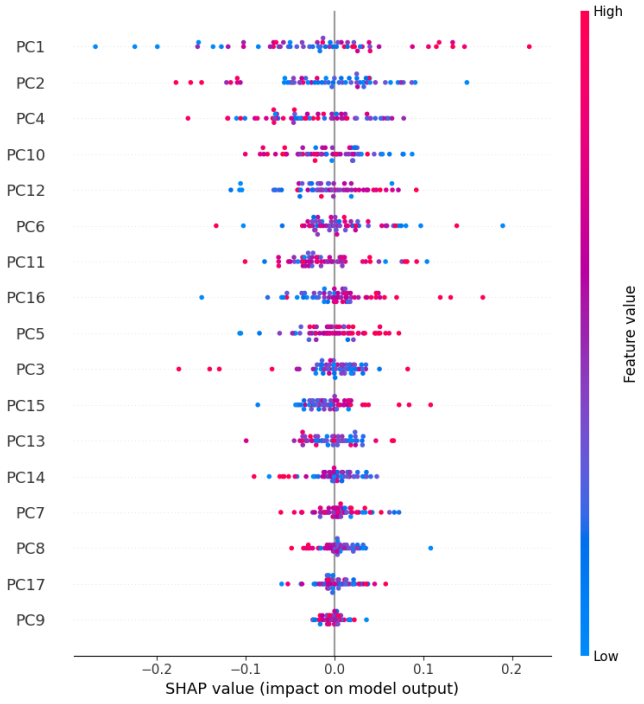

Figure 8.14: Post kPCA FSNN Latitude model SHAP Summary Plot

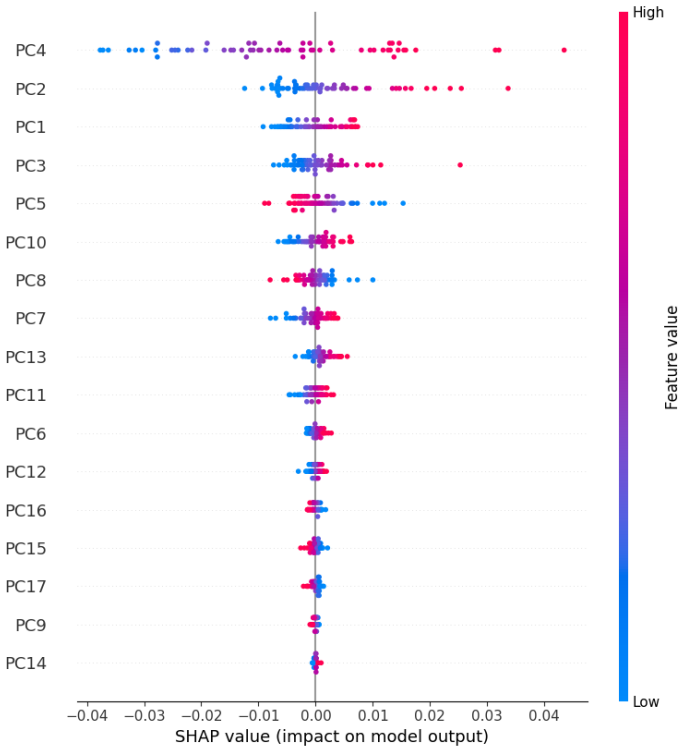

Figure 8.15: Post kPCA FNN Ticket model SHAP Summary Plot

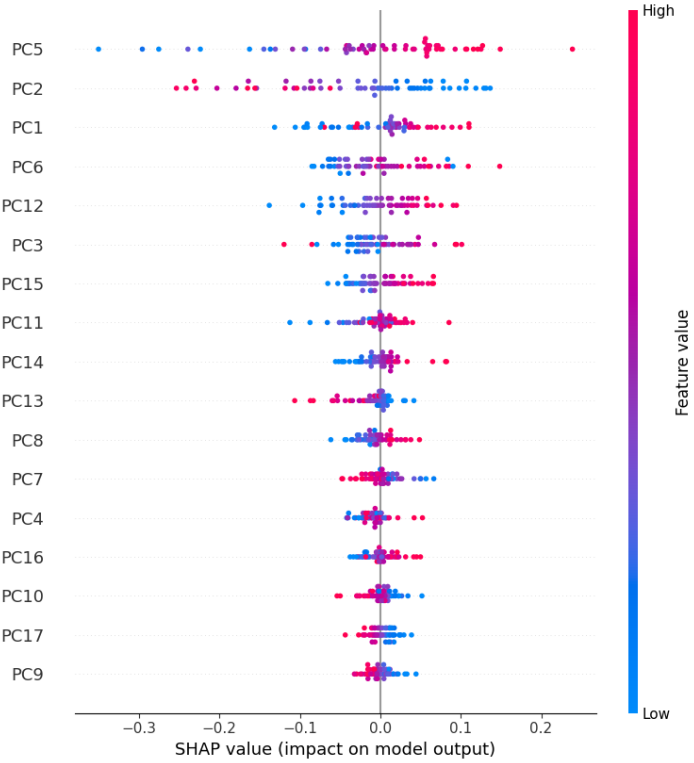

Figure 8.16: Post kPCA FNN Latitude model SHAP Summary Plot