How Negative Sampling Provides Class Balance to Rare Event Case Data Using a Vehicular Accident Prediction Project as a Use Case Scenario

by

Jeremiah Roland

Mina Sartipi
Professor of Computer Science
(Chair)

Osama Osman
Assistant Professor of Civil Engineering
(Committee Member)

Dalei Wu
UC Foundation Associate Professor
(Committee Member)

How Negative Sampling Provides Class Balance to Rare Event Case Data Using a Vehicular Accident Prediction Project as a Use Case Scenario

by

Jeremiah Roland

A Thesis Submitted to the Faculty of the University of Tennessee at Chattanooga in Partial Fulfillment of the Requirements of the Degree of Master of Science: Computer Science

The University of Tennessee at Chattanooga
Chattanooga, Tennessee

Dec 2020

ABSTRACT

Rare event case data occur at such an infrequent rate that even having high amounts of it can leave researchers starving for more information. There has always existed a tug and pull relationship among rare event case data, where a higher count of entries often leads to a lack of explanatory variables, and vice versa. In the research spectrum of rare event case probability prediction, several methods of data sampling exist to remedy the main issue of rare event case data: a lack of data to collect and learn from. The most effective methods often involve altering the distribution of the training samples in a data set. The least utilized of these methods is negative sampling, where positive entries in a data set are used to generate negative entries. To outline the utility of negative sampling, this work discusses the application of five types of negative sampling on a vehicular accident prediction project, where non-accident records are generated through manipulating the temporal and spatial attributes of existing accident records. Moreover, different methods of data manipulation, including feature selection and different negative to positive data ratios, are used to explore what types of explanatory variables are most important when predicting vehicular accidents. Additionally, two types of predictive models, a Multilayer Perceptron and a Logistic Regression model, are created and directly compared in terms of predictive capability. Ultimately, the best model for predictive performance is heavily dependent on the specific implementation and desired results.

ACKNOWLEDGMENTS

First and foremost I wouldn't be the stubborn perseverant fool I am today with out my mom, Jenny Roland. She raised me to the best of her abilities all by herself, and is responsible for the majority of the good traits I have. She's also responsible for some of my bad traits, but I'll skip those. She's always encouraged and supported me through life, and I simply can't thank her enough. I'd also like to thank Rodney Mitchell and Denise Stricklin, two of my high school teachers who originally introduced me into computer science and inspired me to take school more seriously.

I would also like to thank Mina Sartipi for giving me the initial opportunity to join her research group back in 2018. It has been a life changing experience that has proven invaluable to me. Additionally, I'd like to thank my research colleagues from CUIP in general, for being very supportive and helpful throughout my time working with them. I may not have been to most expressive person during our time together, but know I greatly value each and every one of you. Of note, Jose Stovall, Rebekah Thompson, and Jin Cho who all acted as the 'senior' members of the CUIP group who I and pretty much everyone else would always go to for questions. They not only were always willing to help, but often went out of their ways to help us all have an easier time with school, research, and life.

I would also like to thank Pete Way and Katie Rouse for being the best friends I could have asked for, and just being amazing in general. Of special note regarding Pete, he proved to be a valuable colleague and friend during the years I've spent doing research, acting as the blind optimist and overly cheery counterbalance to my more cynical and laid back methods and attitudes.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Motivation for Thesis Applications

Every day, there is a massive influx of new data being added to some collective pool of data. This data can range from the mundane goings on of the average citizen to the extraordinary discovery of a new celestial body. However, even with such a large collection of data, scientists and researchers still continue the hunt for additional knowledge. Of particular interest are rare event case data where there may exist a vast amount data, yet said data occurs so infrequently that the potential for analysis, exploration, and understanding is hindered. As the name implies, rare events are events which occur very infrequently, yet when then do occur, dramatic consequences can arise [2]. The common remedy to rare event case data issues is to apply a method of data sampling, which involves minimizing the effects of rareness by altering the distribution of the training samples in a data set. There are many commonly used forms of sampling, such as over sampling [3], under sampling [3], stratified sampling [4], or random sampling [4]. A rarely employed version of sampling is the process of negative sampling, where positive entries in a data set are used to generate negative entries. Negative sampling is primarily used for language based research projects, [5–8], but has seen promising results in vehicular accident prediction projects [9–12]. The slowly growing popularity of negative sampling as a rare event case balancing agent requires a more thorough understanding of how negatives are produced, the different methodologies of doing so, and the resulting impact of those different generation methodologies.

1.2 Negative Sampling

The definition of negative sampling depends on the context which it is applied to. The most popular and most common application of negative sampling is for language based projects, such as Word2Vec and Natural Language Processors. In these instances, negative sampling refers to the process of splicing out a subsection of a dataset for analysis. The lesser applied use of negative sampling is for incident creation in rare event data, where the process of negative sample generation refers to the creation of an opposing incident entry to act as a balancing agent for a dataset. An example of this would be accidents which are represented as positive entries, and non-accidents which are represented as negative entries. These non-accidents are generated from a dataset of accidents where temporal and spatial information is manipulated to create entries, or records, for non-accidents where each non-accident is simply a record for a time and location where an accident did not occur. While conceptually simplistic, the analytical power these negative entries provide the dataset cannot be understated.

1.3 Research Objective

Negative samples assist in learning from rare event classification incidents, as the positive entries display the factors at play when said incident occurs, while negative entries show what factors are at play when an incident does not occur. However, the introduction of negative entries brings with it the issue of balancing the number of positive entries with the number of negative entries, known as class balance. Different projects have attempted alternative methods for addressing class balance, such as stratified sampling [13], modifying sample weights [11], and over-sampling methods [14]. However, the application of negative sampling for the purpose of addressing class balance is underutilized. Through this, negative samples provide a more concrete understanding of the different factors behind an incident

occurrence. In the case of vehicle accidents, negative samples allow for deeper analysis into how strongly different temporal and spatial factors affect an accident occurrence. The complexity of negative sampling comes from its implementation: How does one properly generate negative entries and which methodology provides the best results? This project addresses those issues by discussing systematic and brute force generation approaches, data manipulation methods, and real world implementation result comparisons between a Logistic Regression model and a Multilayer Perceptron model.

1.4 Thesis Layout

The remainder of this work is structured as follows: Chapter 2 discusses similar projects pertaining to data sampling, negative sampling for rare event data, and negative sampling for language based projects. Chapter 3 reviews the basic conceptual information for the different topics discussed in this work, including rare events, data sampling, logistic regression, and multilayer perceptrons. Chapter 4 introduces and reviews the use case scenario for negative sampling, which is a vehicular accident prediction project, discusses the set up processes for the multilayer perceptron and logistic regression models, and finishes with a broad overview of how and why negative sampling is used for the use case scenario. Chapter 5 reviews the model creation results for the multilayer perceptron and logistic regression models, breaks down the overall performances of the different negative sampling methods, demonstrates how each model performs when predicting future vehicular accident occurrence, discusses the performance discrepancies seen between the different models, and finishes with what model had the overall superior performance. Lastly, Chapter 6 provides a quick summation of the project, some additional closing thoughts, the impact of the work presented, and discusses potential future work possibilities for negative sampling and the vehicular accident prediction project.

CHAPTER 2

Related Works

2.1 Data Sampling

The most commonly faced issues with analyzing and evaluating prediction attempts on rare event data are covered by [3]. The issues covered consist of a lack of data, class distribution, improper evaluation metrics, data fragmentation, and more. The authors also discuss different algorithmic techniques for assisting in rare event analysis, such as cost-sensitive learning, a threshold method, altering bias values, and utilizing human interaction and knowledge. The authors also discuss different data manipulation methods, including feature selection, basic data sampling and advanced data sampling, and kernel-based methods, along with the benefits and drawback of utilizing any of the previously mentioned methods.

The importance of sampling methods for assisting in rare event data analysis is outlined by [15]. "Rare events may occur only infrequently, but they are often the events that have the highest impact. The probability of occurrence and the dynamics leading to such a rare event are crucial information for understanding, predicting, or planning for them" [15]. To address this, the team focuses on rare event sampling methods by evaluating different algorithms for rare event sampling. These algorithms include altering the prediction formula used through *dynamic importance sampling*, a *splitting algorithm* which generates additional rare event observations, re-sampling observations in a dataset through *population dynamics*, and a *Markov Chain Monte Carlo algorithm*.

The team of [4] set out to address the issue faced by performing statistical analysis on rare events data. The first issue faced is how the mean of a binary variable is often

overlooked, preventing proper understanding of a data set's content. "The mean of a binary variable is the relative frequency of events in the data which, in addition to the number of observations, constitutes the information content of the dataset" [4]. The second issue faced is the process of data collection, and the trade off of more observations leading to fewer explanatory variables, and vice versa. To address these issues, the team studied various methods of data acquisition and manipulation. For data acquisition, different sampling strategies such as *random sampling*, *exogenous stratified sampling*, and *choice-based sampling* were tested with a Logistic Regression model. For data manipulation, the team discussed the pros and cons of prior correction and weighting methods for selection on Y, rare event and finite sample corrections, parameter estimation, and probability calculations. The team concluded that the effects of different methods would have the greatest impact when the number of observations was under a few thousand, and the rarity of events were under approximately 5%.

2.2 Negative Sampling for Rare Event Data

A study was conducted by [9] on predicting traffic accidents by comparing the prediction results of four different classification models of prediction. In this study, negative sampling was used to generate non-accident records. For each positive example, the value of one feature was changed among hour, day, and road ID. The resulting example was then checked for a positive (match found) or negative (no match found) result. When the negative sample process was finished, the team created triple the number of negatives than positives, roughly a 75/25 split of data.

The team of [10] performed similar tests with accident prediction and negative sampling. Contrasting the systematic approach, this team created their negatives through brute force. Every single possible combination of data were generated by altering the time and location information of the accidents. This method resulted in 2.3 million negatives for their dataset.

A unique method of negative sampling was performed by [12], resulting in a ratio of 1:4, with 4 non-accident entries per accident entry. The method of negative sampling used involved the process of freezing the day of the week, time, and location of the accident, and looking two weeks before, one week before, two weeks after, and one week after the date of the original accident.

2.3 Negative Sampling for Language Processing

In [8], four strategies of negative sampling were studied for language processing applications. While not applied to rare event case data, the concepts covered still yield valuable information into the process of negative sampling. *Local Sampling* refers to negatives that are close to the existing positive sample by some given measure of approximation. *Distance Sampling* refers to negatives that are as distinct from the positive entries as possible to an extent, ensuring the data is correctly clustered in the given space of study. *Uniform Sampling* refers to the random selection of negatives within the given space, ensuring the entire space to be explored is represented equally. *Refined Sampling* is simply the combination of Local and Distance sampling. In addition, [8] outlined rules for negative samples; negative samples should be 1) as similar as possible to positive samples to increase the model's discriminative abilities, 2) as different as possible to positive examples to avoid feeding the model conflicting information, and 3) representative of the entire space of negative samples.

[16] also provides three unique divisions of negative sample creation. *Incompatible Relations* are relations that always, or almost always, conflict with the relation wished to be extracted. In regards to rare event case data, an incompatible relation would be when a generated negative sample exactly matches a positive sample. *Domain Specific Rules* are negative samples that are highly specific towards the particular data being explored. *Random Samples* deal with marking some current data as negative evidence.

CHAPTER 3

Background Information

3.1 Rare Events

Rare events, by their very name, are events that take place with a significantly lower frequency compared to more common events [3]. These rare events include credit fraud, identity theft, natural disasters, wars, vehicular accidents, and much more. While these rare events may only occur infrequently, they are often events that have the highest impact [15]. For researchers, the issue with rare event data comes from two primary sources. The first is the utilization of rare events with probabilistic statistical methods, as they are more likely to underestimate the probability of a rare even occurring, leading to faulty model creation and rare event estimation errors [3, 15]. This logically follows given that probability based methods are more inclined to predict for events that have a higher observation count. The second issue is a quality vs quantity issue that has to do with the process of collecting rare event data. When collecting rare event data, there is an inherent trade off between more observations and more explanatory variables [3, 4]. In other words, the more rare event data collected, the more likely some observations will be lacking some explanatory variable, thus necessitating the researcher to either drop that observation or drop that explanatory variable.

The issues with rare event data also extend into the realm of machine learning, consisting of the following issues outlined by [3]:

- *Lack of Data*: when the number of observations associated with a rare event class of data is very small. For example, the number of people who die being struck by lightning.

- *Relative Lack of Data*: when the number of observations for an event are rare relative to other events, objects, or classes. For example, if out of 100,000 cancer patients 5000 of them have a mutated form of that cancer. In this case, the number of majority class examples far exceeds that of the minority class, while the minority class when considered on its own may not be seen as rare.

- *Class Distribution*: a classifier will typically assume a dataset is properly distributed between the training and testing set. Unfortunately, this is not always the case.

- *Improper Evaluation Metrics*: typically accuracy is used as an evaluation metric, yet it does not properly evaluate a model's performance [2, 3]. Since accuracy is computed from correctly classified observations, it is heavily biased towards the majority class at the expense of the minority class [3].

- *Inappropriate Inductive Bias*: when a machine learning model has a predisposition for one explanation rather than another.

- *Small Disjuncts*: an inductive set of rules for rule-based classifiers which correctly classify small training examples. It is argued that class imbalance leads to small disjuncts that are more error prone [3].

- *Data Fragmentation*: when using divide-and-conquer strategies that partition rare event data into smaller groups, a lack of data within a partition can become an issue.

- *Noise*: nonsensical or contradictory data within a dataset that can affect model performance.

3.2 Basic Data Sampling Methods

Data Sampling provides a means to address the class balancing issue inherently present in rare event data. *Under Sampling* is the process of balancing a training set by removing

observations from the majority class [3]. The inherent issue with under sampling is during the process of removing observations from the majority class, one could also be removing useful information. In contrast to under sampling, *Over Sampling* is the process of creating identical examples of the minority class to balance the training set [3]. This process of data duplication can come at the cost of increased computational times, as well as over-fitting due to the identical copies in the minority class. *Random Sampling* is where all observations in a dataset are selected at random for use [4]. *Case Based Sampling* is the process of collecting all available events and a small random sample of non-events [17]. Lastly, *Stratified Random Sampling* is when the observations in a dataset are broken up into different strata (groups), and random observations from each strata are taken for analysis [18]. These strata are formed based on characteristics shared between observations, such as age, gender, height, location, etc.

3.3 Logistic Regression

Logistic Regression is a statistical method used for binary classification problems. It utilizes the logistic function, which takes any real-value number and maps it to a new value between 0 and 1 [19]. The logistic function is an S-shaped curve shown in Figure 3.1, with an accompanying formula where e is the base of the natural logarithms, and z is the numerical value to be transformed.

The implementation of the logistic function for logistic regression involves taking input values x, and combining them in a linear fashion using weights or coefficients to predict some output y. Unlike linear regression, where the output is a numeric value, the output of logistic regression in a binary value (0 or 1) [19]. Equation 3.1 shows a simple logistic regression equation, where:

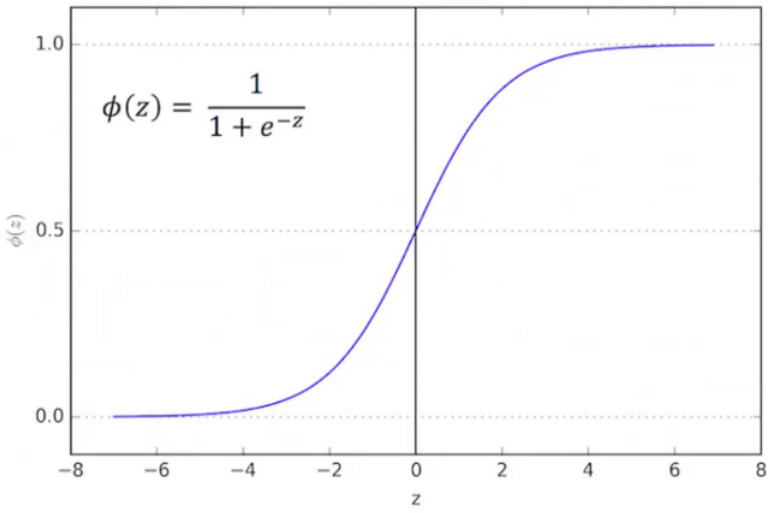

Figure 3.1 A Logistic Function S-Curve, [1]

- y is the output
- b is the bias
- c is the coefficient for the single input value x

$$y = e^{b+(c*x)}/(1 + e^{b+(c*x)}) \tag{3.1}$$

In implementation, logistic regression models the probability that an input belongs to the default class [19]. For example, if modeling a person's gender based on height with the default class being male, logistic regression would be predicting the probability of the gender (y) being male (1) based on the height (x).

3.4 Multilayer Perceptron Review

Neural Networks [20] are built upon the idea of artificial intelligence, where they mimic the natural learning process a human undergoes by being modelled after the human brain. In the same way that human brains are comprised of neurons sending electrical signals to each

other, neural networks are comprised of nodes sending information to each other. By doing this, a machine is essentially taught to learn similar to how a human learns. A multilayer perceptron (MLP) is a type of artificial neural network composed of an input layer, any number of hidden layers, and an output layer. An MLP is constructed with individual nodes which are clustered into layers. All computations happen in the hidden layers of the MLP, and there can be as many hidden layers as desired by the programmer. These multiple hidden layers are what distinguish an MLP network from other artificial neural networks. MLPs have proven useful for predicting continuous problems, yet are more frequently used for classification problems. Figure 3.2 shows the basic layout of an MLP with one hidden layer, where:

- $a_i^{(h)}$ is the i^{th} node in layer h
- $w_{i,j}^{(h)}$ represents the weight of node i going into node j in layer h
- g is the activation function
- b^h is the bias for layer h
- $\hat{y}$ is the prediction output

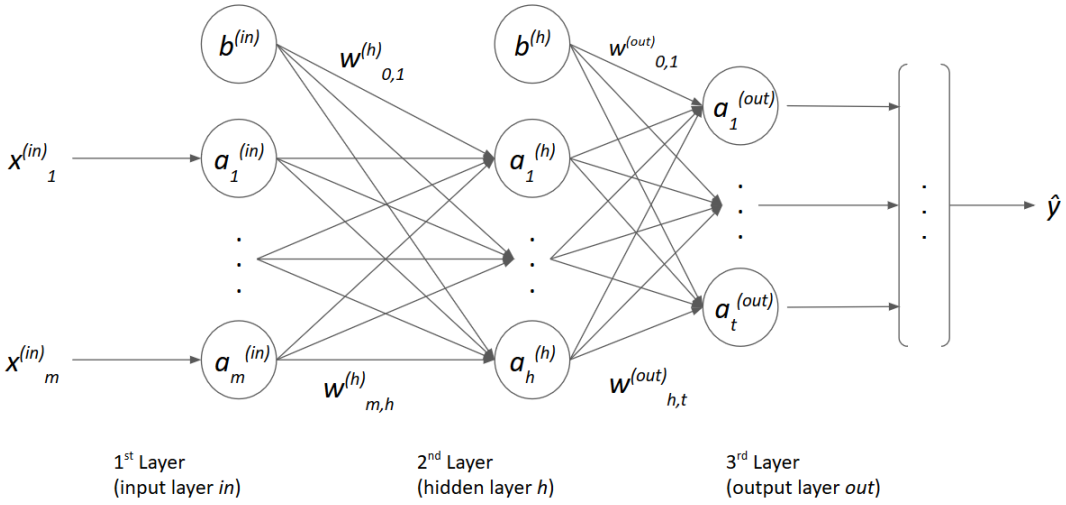

Figure 3.2 A Basic Multilayer Perceptron Network with One Hidden Layer

Activation functions, also called transfer functions, are mappings of summed weighted inputs to the output of the current node the function is being applied to [21]. While not directly present in Figure 3.2, the activation function is at the forefront of each node. The bias nodes present in Figure 3.2 are known as bias neurons, which are special neurons added to each layer of the MLP [22]. For neural network models, bias acts as a shifting agent for the activation function, similar to how a constant acts for a linear equation. The mathematical representation of Figure 3.2, with a single hidden layer, is shown in Equation 3.2.

$$\hat{y}_k = g(b^{(h)} + \sum_j g(b^{(in)} + \sum_i a_i^{(in)} w_{ij}^{(h)}) w_{jk}^{(out)}) \tag{3.2}$$

One of the main proponents of an MLP, and neural networks in general, is the back-propagation algorithm. This algorithm is comprised of the *feed forward pass* and the *backward pass*. The *forward pass* calculates the prediction by moving from the input layer, through the hidden layers (if any are present), and into the output layer. The *backward pass* then uses partial derivatives of the error function and back-propagates them through the network. This yields the gradient of error which can then be adjusted to find the minimum error rate. The back-propagation algorithm continues this optimization process to lower the error rate of the predictions until convergence is reached, where the network is no longer capable of improving given its current configuration. The training process of MLPs involves modelling the correlation between a set of input and output pairs and adjusting weights and biases of the model to minimize error. Think of this process as a game of pong, where there exists a constant back and forth, estimating some prediction calculation, receiving feedback, and making corrections/alterations accordingly.

CHAPTER 4

Methods

4.1 Vehicular Accident Prediction Project: A Use Case Scenario

To better elaborate the applicability and process of negative sampling, a vehicular accident prediction project [23], hence forth referred to as APP, is used as a use case. The purpose of APP is to predict accident hotspots within the Chattanooga area, with the end goal being to supply the local law enforcement a live service that can update first responders throughout the day where accidents are most likely to occur. A Multilayer Perceptron (MLP) has acted as APP's predictive model for the majority of its lifetime, as initial predictive attempts using other predictive models, such as logistic regression and logit/probit models, yielded poor results. However, recent predictive attempts were made using a simple logistic regression model which yielded superior results when compared to an MLP.

4.1.1 The Data

The accident data is provided by the Hamilton County Emergency Service Department, with the accidents ranging from October 2016 to present day being updated on a daily basis. While the total available data ranges from 2016 to present day, all prediction models created and shown in this work were created using data from 2017-2019, consisting of roughly 61,000 accident entries after data cleaning. These accident records include timestamps for the accident call report, the officer arrival time, the latitude and longitude coordinates of the accident, and the level of injury (no injury, injury, entrapment, mass casualty). By using the

time and location information provided by the accident records, the weather present during the time of the accident was fetched through DarkSky. DarkSky is a Python API library that collects weather reports from several different weather sources and returns the most appropriate weather report based on the location and time provided. Roadway geometric information was also obtained with ETRIMS and ArcGIS. ETRIMS is a database of Tennessee specific roadway information and ArcGIS acted as a mapping program to assist in manipulating and aggregating spatial information. Figure 4.1 shows the general outline of data collection, processing and aggregating, and modeling and visualizing.

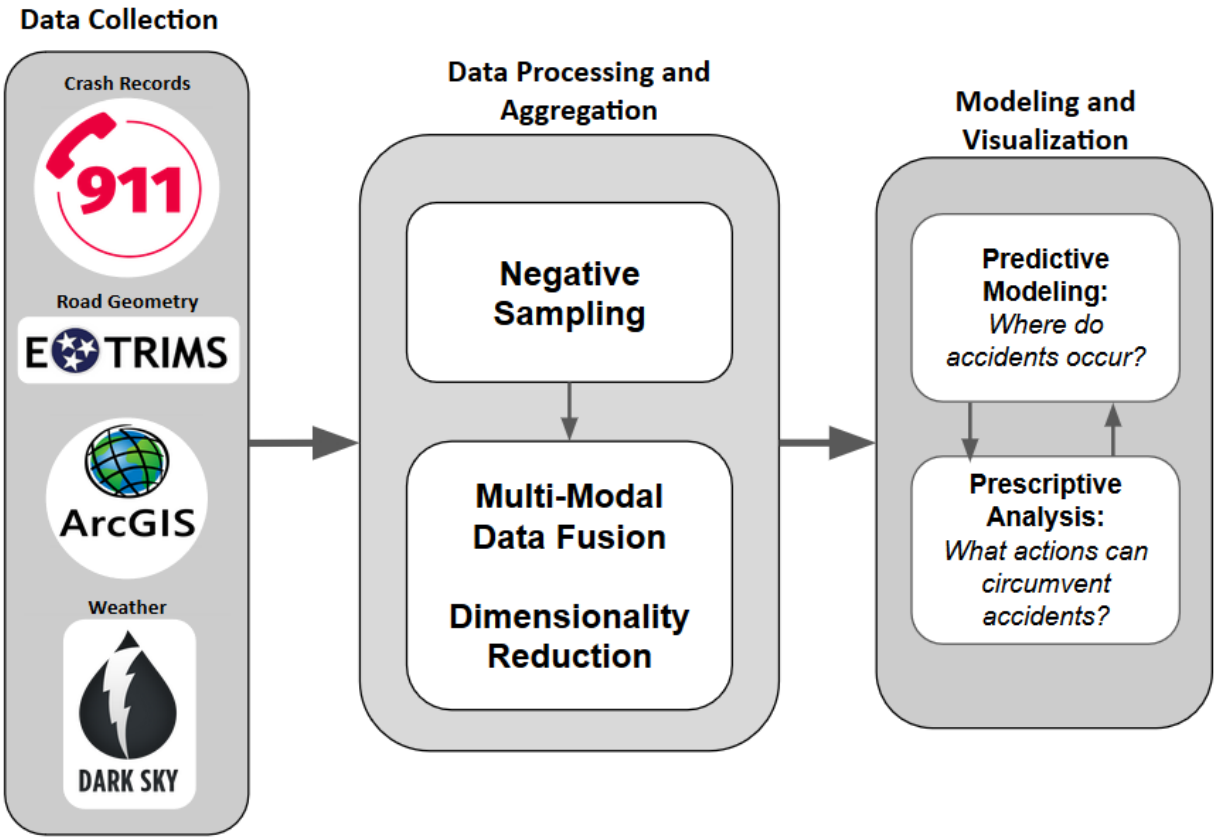

Figure 4.1 Illustration of Data Collection and Processing

Through DarkSky, weather specific information such as precipitation intensity, weather type, and visibility were added to the accident records. Through ETRIMS, roadway specific

information such as number of lanes, pavement type, and roadway function class were also added to the accident records. Finally, ArcGIS provided the project with the means to spatially aggregate the study area into 694 hexagons, each covering 0.2 square miles. This spatial aggregation resulted from previous accident prediction attempts yielding very high false positives on accidents, which indicate instances where the model thinks there will be an accident but there was not. Originally, APP had no spatial aggregation and attempted to predict accidents for each roadway segment along the roadways in Chattanooga. This lead to a very high count of accidents being predicted and excess noise, leading to the necessity for spatial aggregation. With the aggregated study area, the prediction process is greatly simplified. Furthermore, due to the spatial aggregation, the roadway geometrics also needed to be aggregated, resulting in the roadway geometrics present in the project being aggregated versions of their original values based on averages per hexagon. Table 4.1 displays all the variables present for a single accident entry. Figure 4.3 displays the hexagon layout covering the study area.

The different hourly aggregations for DayFrame can be seen in Table 4.2. The different hour splits were selected based on accident trends seen in Figure 4.2. Corresponding to the spikes seen in the Figure, each DayFrame covers a particular mindset that drivers may be in during the day. DayFrame 1 (orange) covers the overnight hours, DayFrame 2 (green) covers the morning rush hours, DayFrame 3 (blue) covers the lunch rush hours , and DayFrame 4 (purple) covers the evening rush hours.

Table 4.1 Variables Used in Study

APP Variables	Explanation
Accident	No Accident(0) or Accident (1)
Lat/Long	GPS Coordinates of the Accident
Unix	Timestamp of the accident in seconds
Hour	The hour of the day accident occurred
DayFrame	Aggregated Hour times of the day (see Table 4.2)
WeekDay	If accident was on weekend or weekday (binary)
DayOfWeek	Day of the week (0-6, Mon-Sun)
Roadway Variables	**Explanation**
Join Count	Historic Accident count for grid number
Grid_Num	Position in aggregated spatial hexagon layout
TY_Terrain	The type of land terrain (rolling, flat, etc.)
NBR_Lanes	Number of lanes
Func_Class	Function Class (municipal highway agency, etc)
Weather Variables	**Explanation**
cloudCover	Percentage of the sky covered by clouds (0 to 1)
dewPoint	Air temp required for water vapor saturation
humidity	Amount of water vapor in the air (0 to 1)
precipIntensity	Intensity of precipitation at time of record
pressure	Air Pressure
temperature	Temperature at time of record
uvIndex	Amount of Sunlight
visibility	Visible Distance (miles)
windSpeed	Speed of the Wind (mph)
Rain/Cloudy/Foggy/Snow/Clear	Precipitation conditions (binary)
Rain Before	Rain in previous hour (binary)

Table 4.2 DayFrame Breakdown

DayFrame	Hours Covered
DayFrame 1	0 - 4 and 19 - 23 (Overnight)
DayFrame 2	5 - 9 (Morning rush)
DayFrame 3	10 - 13 (Lunch hours)
DayFrame 4	14 - 18 (Evening rush)

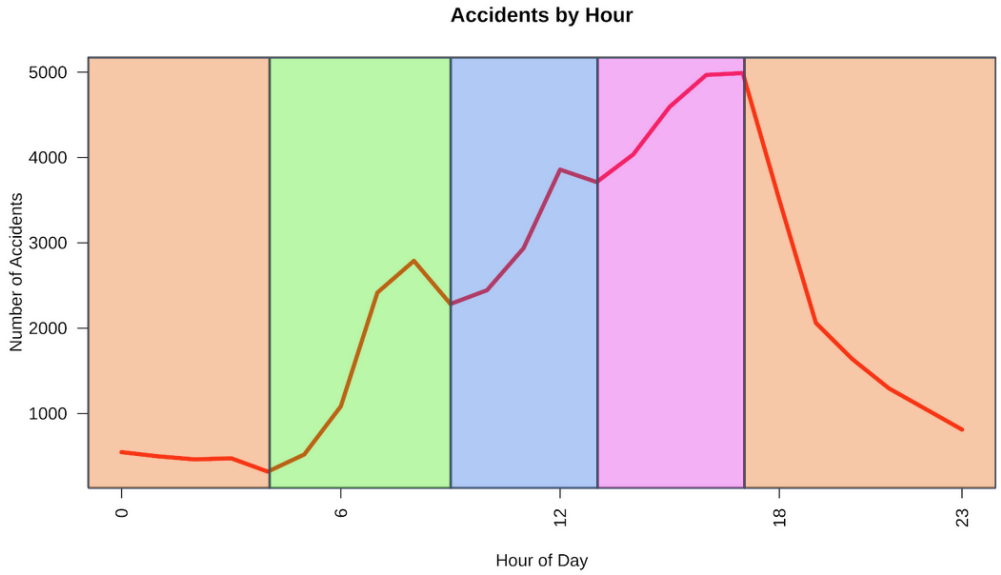

Figure 4.2 Distribution of DayFrame hours encompasses the various accident trends throughout the day

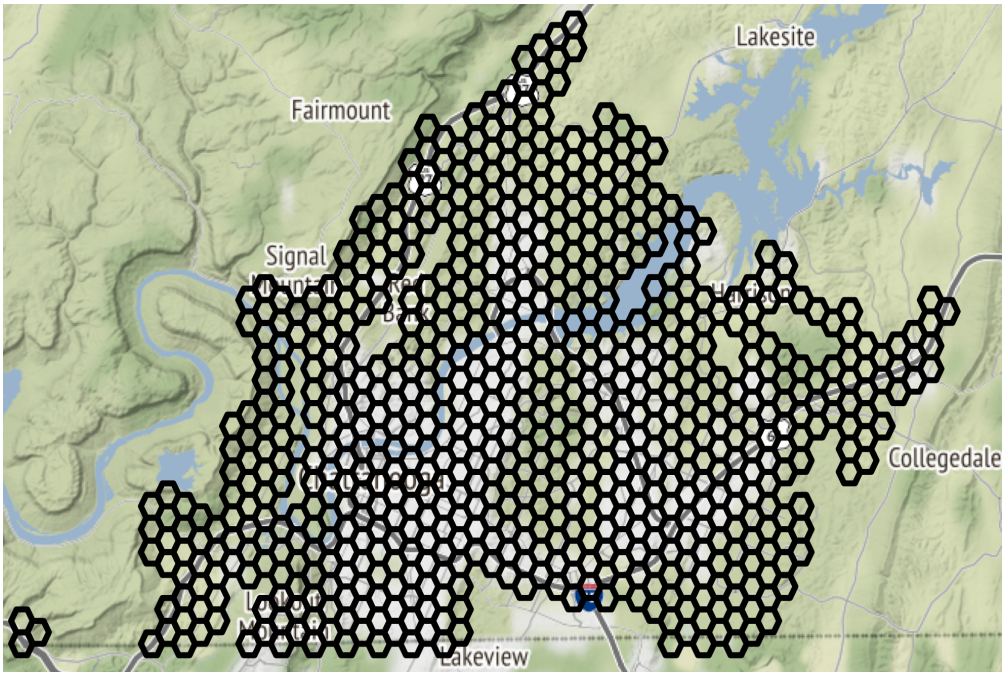

Figure 4.3 Hex Layout View of Chattanooga, TN

4.1.2 Neural Network Model

APP has primarily used a standard Keras Sequential Multilayer Perceptron (MLP) offered through a Python module in Pycharm. Several variations of the MLP model with different hyper parameters were tested, including different amounts of hidden dense layers and different dropout layers of varying size. MLPs are flexible with their use of data, which is of great benefit to APP as the dataset is complex and intricate. Furthermore, the data uses labelled inputs for classification prediction, which MLPs are suitable for. Nadam acts as the model architecture with mean squared error (MSE) providing compilation. Table 4.3 shows the architecture of the MLP used for APP. In the Node column, a formula is used to determine the number of nodes per layer. This was implemented due to the Test Types described below in Section 4.2.4, as each test has a different number of variables used. Additionally, for Layer 4 the number of variables is either reduced by 10 or 5. This addition was due to one of the Test Types resulting in a dataset with less than 10 variables, requiring an adjustment.

Early prediction models utilized binary cross-entropy for compilation, however testing with MSE for compilation provided a significantly lower loss score at a cost of only 2% accuracy. Sigmoid, acting as the model's activation function, out performed alternative activation functions including identity, tanh, and relu. This performance difference is due to sigmoid being a squashing function which limits a prediction model's output to a range of 0 to 1, making it particularly useful for probability prediction. Additionally, Nadam [24] provided superior performance when compared to alternative optimizers, such as sgd and adam. The model runs in 10 learning cycles where the output of the previous cycle acts as the input for the next cycle, allowing for more learning. Furthermore, different combinations of hidden layers and dropout layers of varying size were tested, with the layout present in Table 4.3 yielding the best performance. The best results were achieved with one dropout layer set to 0.1 with 2 dense hidden layers with a node counts of X-5 and X-10, with X

being the number of variables used by the data. Lastly, early stopping is used to prevent redundant learning, where the current learning cycle breaks if the model does not change in accuracy by some threshold for a set amount of epochs.

Table 4.3 MLP Neural Network Architecture

Layer	**Location**	**Type**	**Node**	**Activation**
1	Input	Dense	X	Sigmoid
2	Hidden	Dense	X - 5	Sigmoid
3	Hidden	Dropout	-	-
4	Hidden	Dense	X - 10 or 5	Sigmoid
5	Output	Dense	1	Sigmoid

4.1.3 Logistic Regression

Recent testing with Logistic Regression for APP yielded superior predictive performance compared to the MLP model which has been historically used for the majority of APP's lifetime. Different hyper parameters were tested with logistic regression, with the best performing model using newton-cg as the solver and setting the class weight to balanced. Newton-cg is used for multiclass problems, and outperformed other multiclass problem options including saga, sag, and lbfgs. Balanced acted best for class weight, as it uses the values of the prediction variable (y) to adjust weights inversely proportional to class frequencies in the input data (x) [25]. However, it is important to note that using balanced or any alternate class weight parameter yielded a small difference in predictive performance when compared to a logistic regression model with the class weight set as the default setting of none.

Additional parameter testing including altering the tolerance for stopping criteria. It was found that there was negligible change in increasing and decreasing the tolerance value, and that the higher the tolerance value the more inclined the model was to predict negative entries. In the case of APP, this meant a higher tolerance value would create a model which

predicted more non-accidents. The maximum number of iterations was also tested, but only proved necessary when using lbfgs as solver. Additionally, when using newton-cg as the solver, increasing the max number of iterations from the default 100 only yielded about a 0.01% increase in accuracy per additional 400 iterations.

4.2 Negative Sample Generation Methods

4.2.1 Original Inspiration

While the training and testing accuarcies of APP's model had greatly improved through the implementation of an MLP, forecasting yielded inconsistent results. The term 'forecasting' refers to the process of providing a model some day in the future, and predicting where accidents are most likely to occur. Note that all forecasts performed for this paper were done using data that covered 2017 to 2019, while predicting for 2020 dates. The inconsistent results were high false positives for accidents in some instances, and high false negatives for accidents in others. The original inspiration for negative sampling came from [9], where for each accident entry the team changed either the hour, the date, or the road ID of the accident entry. The resulting entry was checked against the entire dataset of accidents for a matching entry. If a match was found, the generated entry was discarded and the process was repeated for every positive entry in the dataset. This process resulted in the team having triple the count of negatives as positives, roughly a 75-25 (negative-positive) data split.

The APP team replicated this process to a degree, as at the time there was no roadway specific information used in the project. Therefore, the original attempt at negative sampling involved changing either the hour or the date of an accident entry. This process resulted in roughly a 66-33 data split. The addition of roadway specific information and additional testing led to several different methods of negative sample testing.

4.2.2 Class Balance

One of the primary benefits for creating negative entries for rare event case data is for the negatives to act as a balancing agent. If a dataset consists solely of positive examples of some event, any attempt to discover the important factors at play are hindered. In other words, even with a significant amount of accident entries to learn from, real world prediction attempts were failing. However, overwhelming a dataset with a significant amount of negative entries can also hinder attempts to discover important factors and lead to useless forecasts. Therefore, maintaining proper class balance is crucial for rare event case datasets utilizing negative examples [11, 26, 27].

To explore the effects of different positive to negative ratio splits, three distinct ratio splits were used in model creation for this project. The first is the **Rare Circumstance 90 - 10 Split** (referred to as No Split), where during the negative sample process the current accident entry being altered is altered 9 times before moving onto the next accident entry. This results in each accident entry generating 9 negative entries. This type of dataset was used to see how an overwhelming amount of negative samples would impact a model's performance, while fully embracing the rare nature of accident occurrence. The second split is the **75 - 25 Split**, taken directly from the team of [9]. Lastly, there is the **Even 50 - 50 Split** to test the impact of an even distribution of negatives to positives. For the 75-25 and 50-50 splits, the specified ratio was reached by altering what negatives were kept from the 90-10 split. For each of the 9 negatives that were generated per accident, the 75-25 split kept only 1/3 of them while the 50-50 kept only 1 negative per accident.

4.2.3 Negative Sampling: Brute Force and Systematic Approaches

There are two paths available for the process of negative sampling, brute force and systematic. Brute force negative sampling is similar to brute force password matching. For

accidents, every possible non-accident per hour of the day at every location is generated. For example, Figure 4.4 shows the process of brute force negative sampling for a single hexagon from Figure 4.3. In this example, there exists an accident for the given hexagon at hours 4, 13, and 20, so the remaining hours of the day for that hexagon are considered non-accident entries.

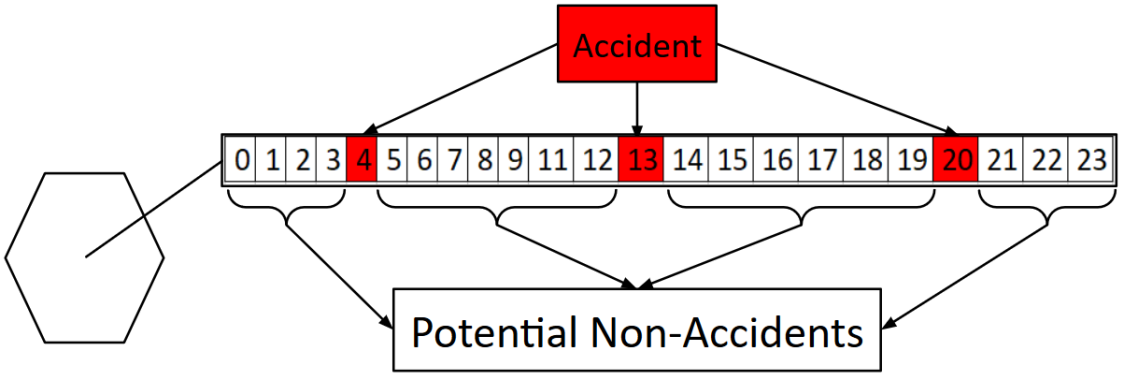

Figure 4.4 Brute Force Negative Sampling Outline

While an effective technique for mass negative sample generation, brute force negative sampling falls victim to the issue of class balance. With such an overwhelming amount of negative entries, any predictive model will yield a high count on negative entries, hindering the predictive capabilities of the model. Such a process of negative sampling was performed by [10], producing 2.3 million negative entries. However, the team only used 0.1% of the generated negatives for their training and testing. Additionally, APP attempted brute force negative sampling in past experiments yet ran into model performance issues when attempting to implement said negatives. Furthermore, the time and resources required to generate such a large amount of data proved inefficient. The process of generating all possible negatives, along with matching the negatives with the appropriate location and weather data, took a significant amount of time, even with the relatively small study area of Chattanooga. Some traffic accident analysis projects can span several counties, cities, or even an entire state. For such large areas, brute force negative sampling would be impractical without a very high end machine.

Systematic negative sample generation takes a more streamlined approach to negative sample generation, where specific variable combinations are altered to generate negatives. For accidents, the hour, date, and location of an accident entry are the variables that can be altered for generating negatives. In this project, five types of systematic negative sample methods are used: *Hour Shift (HS)*, *Date Shift (DS)*, *Spatial Shift (SS)*, *Grid Fix (GF)*, and *Total Shift (TS)*. Figure 4.5 displays the processes of the Hour, Date, and Spatial shifts.

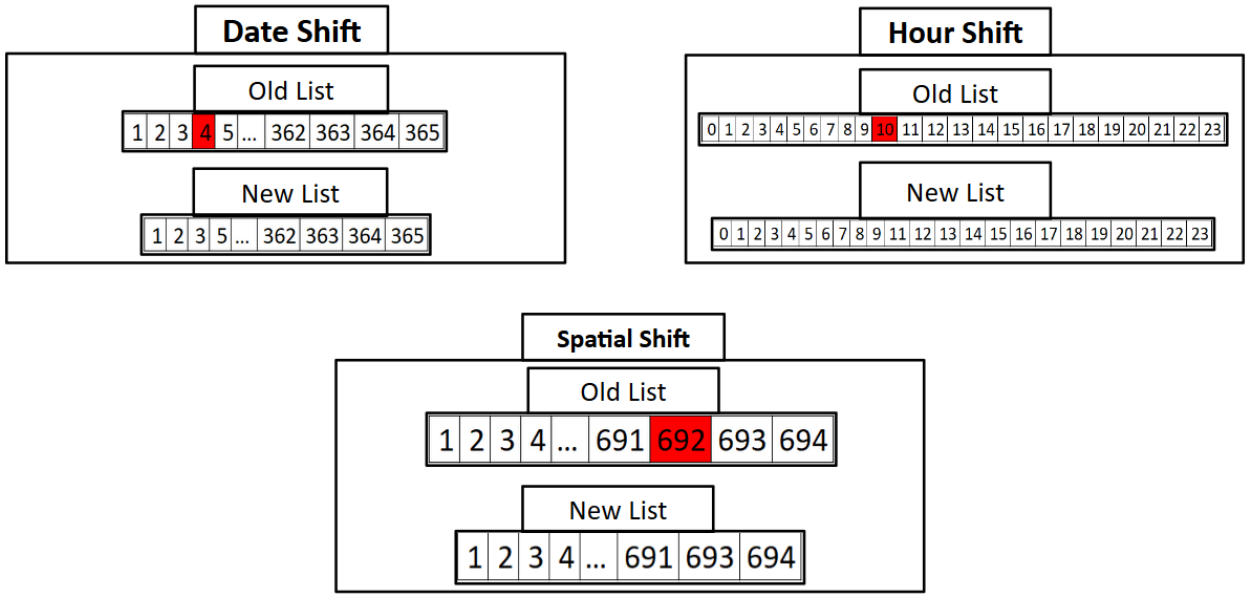

Figure 4.5 Systematic Negative Sampling Outline

For *Hour Shift*, the hour of the accident entry is changed to a different hour between 0 and 23, excluding the original hour of the accident. For *Date Shift*, a new day of the year between 1 and 365 is chosen, excluding the accident's original day of the year. Note for *Date Shift* negative sampling, if being performed on data that does not span an entire year, then the selected date should not extend to a date not covered by the data. For example, if creating negatives for accident data that covers January 1st to March 28th 2019, then any *Date Shift* negative should only be between January 1st and March 28th 2019. Additionally for *Date Shift* negative sampling, the newly selected date should be kept within the same

year of the accident entry. For *Spatial Shift*, the grid number of the accident is changed to a different grid number between 1 and 694, again excluding the original grid number of the accident. The remaining two negative sampling types, Grid Fix and Total Shift, are combination versions of the previous three sampling types. *Grid Fix* refers to the process of altering the date and the hour of the accident while freezing the grid number. *Total Shift* involves changing the hour, date, and location of an accident entry.

4.2.4 Data Manipulation Methods

Two types of data manipulation methods were used when creating models. The first is *Test Type*, which simply refers to one of four tests (A, B, C, D) that dictate what variables are used in model creation. Table 4.4 displays a quick summary of the different tests. Test A models have all available variables present in Table 4.1, and are used to test how an "all in" variable model performs. Test B has redundant variables such as dewPoint, unix, latitude/longitude dropped as they are represented by other variables. Test C models had their weather variables dropped to test how a spatially focused model performs. And lastly, Test D models had their spatial variables dropped apart from Grid_Num, as the grid number was needed for matching predictions to accidents.

Table 4.4 Test Types

Test	**Description**
A	All Variables Present
B	Dropped Redundant Variables
C	Dropped Weather Variables
D	Dropped Spatial Variables (except Grid_Num)

The second type of data manipulation is *Feature Selection*. An ExtraTreesClassifier algorithm was used to return the top 15 most significant variables per model. The exemptions to

this are Test C models, as they only have 8 variables. Amongst all of the feature importance testing, the most commonly seen significant variables included temporal variables (Hour, DayFrame, Unix) and spatial variables (Join_Count, Latitude, Longitude, and Grid_Num). These combinations of most significant variables being consistently reported as significant is counter intuitive to the initial assumption of what variables would be important. Many of these variables either represent the same data in an aggregated form (e.g. Hour and DayFrame or Lat/Long and Grid_Num) or are highly specific to the point of uniqueness (e.g. Lat/Long and Unix).

CHAPTER 5

Results

5.1 Model Training Results

5.1.1 MLP Training/Testing Results

Training/Testing results refer to the Training and Testing accuracies generated during the model creation process. Later sections discuss the accuracies of these models when properly implemented to predict future vehicular accidents. Recall and Specificity are used as metrics for evaluating model performance as they provide a more concrete performance analysis than the Training and Testing Accuracy values. Equations 5.1 and 5.2 [28] display the formulas for Recall and Specificity, respectively. Recall is the percentage of correctly predicted accidents and Specificity is the percentage of correctly predicted non-accidents. Two more commonly used metrics of evaluation for probability prediction projects are Precision and F1 Score. Precision represents the ratio of correctly predicted accidents to all of the predicted accidents, notated in Equation 5.3. F1 Score is a weighted average of Recall and Precision, notated in Equation 5.4, where the higher the number the better. The reason for using Recall and Specificity above Precision and F1 Score is due to the latter two having very low results despite Recall and Specificity remaining high. Note that this issue was only present during the forecasting results discussed later in Section 5.3, and will be further discussed in Section 5.4.4.

$$Recall = \frac{TruePositives}{(TruePositives + FalseNegatives)} \tag{5.1}$$

$$Specificity = \frac{TrueNegatives}{(FalsePositives + TrueNegatives)} \tag{5.2}$$

$$Precision = \frac{TruePositives}{(TruePositives + FalsePositives)} \tag{5.3}$$

$$F1\ Score = 2 * \frac{(Recall * Precision)}{(Recall + Precision)} \tag{5.4}$$

To further emphasize the importance of using Recall and Specificity for model evaluation, Table 5.1 displays the top 5 best performing models according to Training and Testing accuracy. Note for all values in the table, the Training and Testing Accuracy values are all above 90, while the Recall scores are very low. By using Recall and Specificity, it can be seen that regardless of how high the Training and Testing accuracy values are, the models in the table are inclined to predict more non-accidents than they are accidents.

Table 5.1 Top 5 MLP Models by Training and Testing Accuracy

Model	Train Acc	Test Acc	AUC	Recall	Specificity
TS No TA	92.82	92.56	0.70	0.41	0.99
TS No FS TA	92.54	92.39	0.68	0.38	0.99
SS No TA	92.26	92.06	0.65	0.32	0.99
TS No TB	92.03	91.95	0.66	0.33	0.99
TS No FS TB	92.01	92.00	0.66	0.34	0.99

Considering all of the above mentioned data manipulation methodologies, the five different negative sampling methods, the three different ratio splits, 4 test types to change what variables are used with each of those test types having their own reduced version through feature selection, 120 Multilayer Perceptron models were created. To simplify analyzing the results, only the top 10 best models are shown in this section. Table 5.2 shows the top 10

best performing models for training and testing. Using Recall and Specificity as performance indicators, the best performing model is Total Shift 50-50 Test A.

Table 5.2 Top 5 MLP Models by Recall and Specificity

Model	Train Acc	Test Acc	AUC	Recall	Specificity
TS 50-50 TA	**81.58**	**81.91**	**0.82**	**0.83**	**0.81**
TS 50-50 FS TA	80.54	80.78	0.81	0.82	0.80
TS 50-50 TB	79.78	79.64	0.80	0.80	0.79
SS 50-50 FS TB	80.16	79.64	0.80	0.83	0.76
TS 50-50 FS TB	79.27	79.49	0.79	0.80	0.79
SS 50-50 FS TA	78.92	79.10	0.79	0.84	0.74

5.1.2 Logistic Regression Training/Testing Results

Following the trend set by the previous subsection, Table 5.3 shows the top 5 models created, ordered by Recall and Specificity balance. While the best performing logistic regression model has a recall value around 10% less than the best performing MLP model, the results are close enough to warrant further investigation and proper implementation testing.

Table 5.3 Top 5 Logistic Regression Models by Recall and Specificity

Model	Recall	Specificity
TS 50-50 TA	0.74	0.80
TS 50-50 TC	0.73	0.79
SS 50-50 TA	0.71	0.82
SS 50-50 TC	0.68	0.81
HS 50-50 TC	0.67	0.63

Note that the test types used for the logistic regression models followed the same scheme as the test types used for the MLP models, which are those seen in Table 4.4. However, no logistic regression models were created using the B test type, as logistic regression makes finding important and non-important variables easy with the logit table. All logistic regres-

sion models had any variables dropped with a p-value of > 0.05, acting as a substitute for the feature selection method used for the MLP models.

5.2 Result Breakdown

5.2.1 Negative Sampling Method Breakdown

To better understand the effects of the different negative sampling methods applied from Section 4.2.3 on the MLP and LR models, the effects of the negative sampling methods are outlined below.

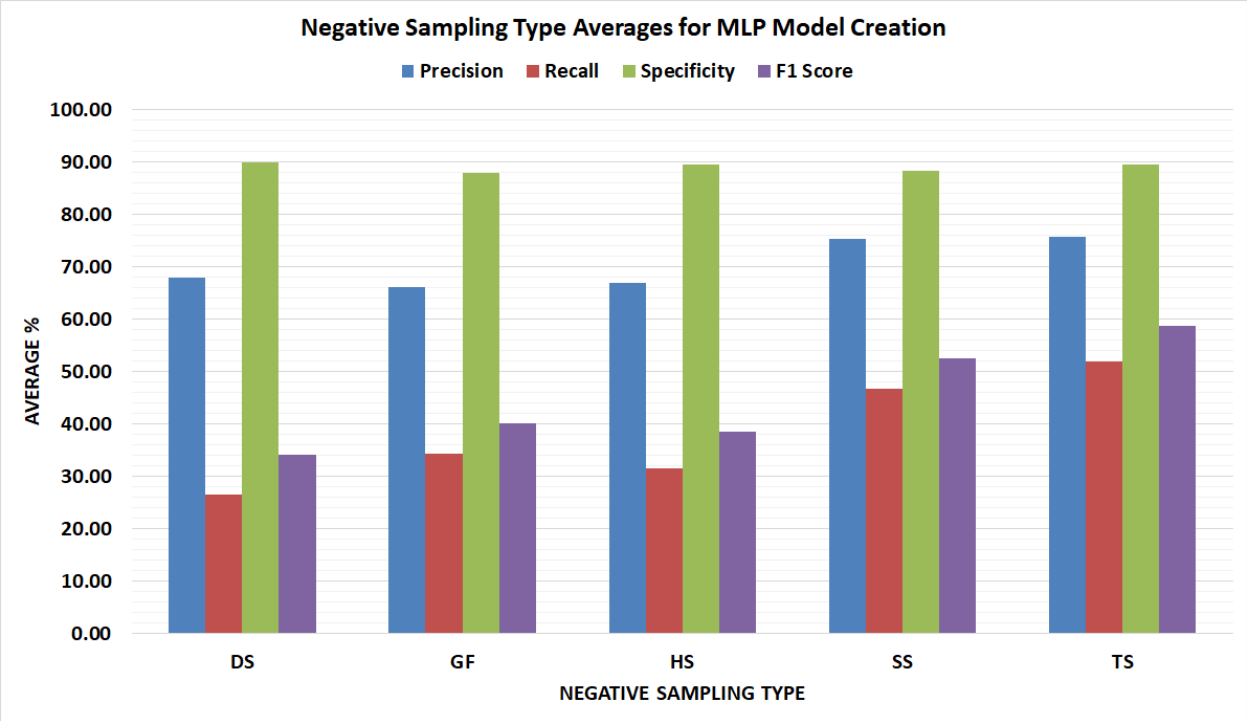

Figure 5.1 Negative Sampling Results on MLP Model Creation

Figure 5.1 shows how the five negative sampling types affected the MLP models' performance during model creation. Overall, Specificity remains high across the negative sampling

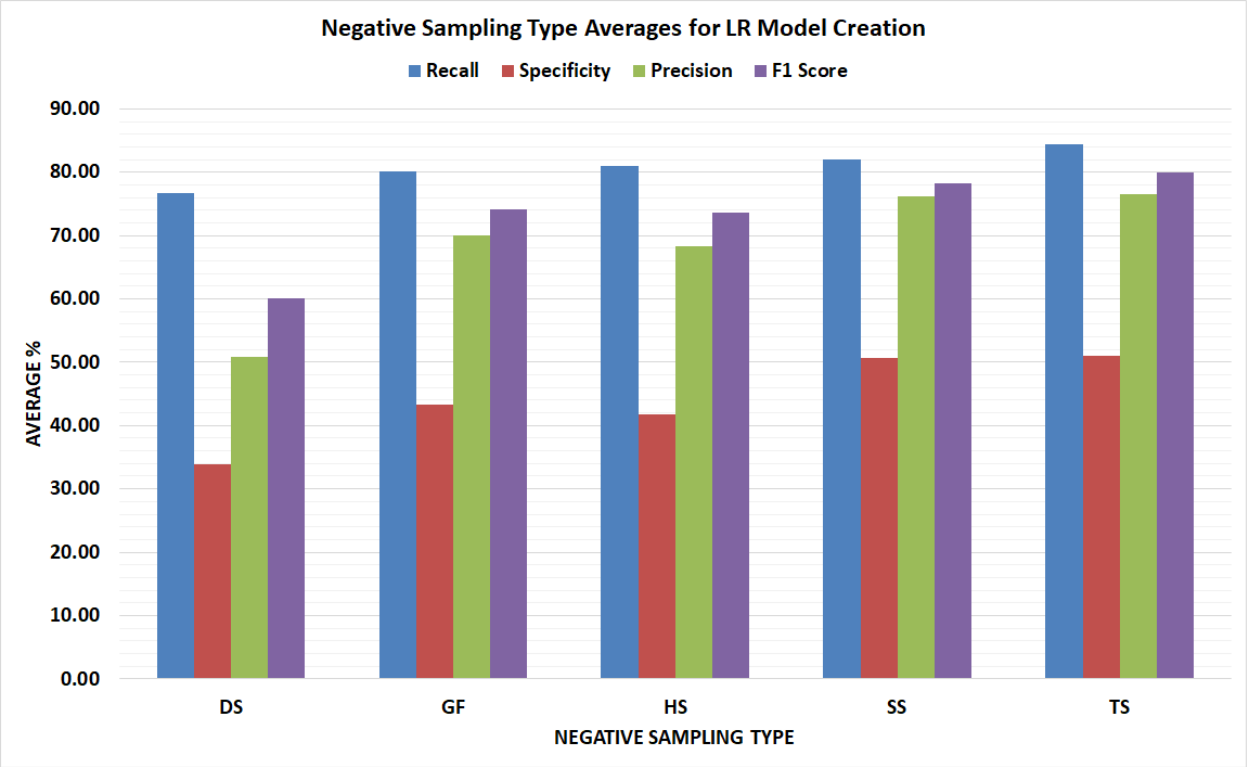

Figure 5.2 Negative Sampling Results on LR Model Creation

types. Recall is low for the Date Shift (DS), Grid Fix (GF), and Hour Shift (HS) models, and is slightly better for the Spatial Shift (SS) models, with the Total Shift (TS) negative sampling type having the highest overall average Recall. Precision remains decently high across the negative sampling types, while only the F1 Score averages reach above 50% in the SS and TS negative sampling types. Figure 5.2 shows how the five negative sampling types affected the LR models' performance during model creation. Overall, Recall, Precision, and F1 Score averages were high, with Specificity taking a large hit across the negative sampling types. From this, we can see that when compared to the MLP models, the LR models are statistically stronger at predicting accidents while being statistically weaker at predicting non-accidents.

5.2.2 MLP Breakdown

Tables 5.4, 5.5, and 5.6 show the top 5 models according to Recall and Specificity for the 50-50, 75-25, and No Split data splits, respectively. The 50-50 split models have the best balance between Recall and Specificity values, suggesting the strongest real world applicability. The 75-25 split models are slightly more bias towards non-accidents, reflected by the higher specificity values and reasonable Recall scores. Lastly, the No Split models have the heaviest bias towards non-accidents, reflected by the near perfect Specificity scores and the low Recall scores. The No Split models provide the perfect example of poor class balance mentioned in Section 4.2.2, as the ratio for the No Split models are 90% negatives and 10% positives, roughly. Due to the overwhelming amount of negatives, the resulting model is heavily bias towards predicting those negatives, leading to the high count of non-accident predictions and low count of accident predictions.

Table 5.4 Top 5 MLP Model Results for 50-50 Split

Model	Train Acc	Test Acc	AUC	Recall	Specificity
TS 50-50 TA	81.58	81.91	0.82	0.83	0.81
TS 50-50 FS TA	80.54	80.78	0.81	0.82	0.80
TS 50-50 TB	79.78	79.64	0.80	0.80	0.79
SS 50-50 TA	80.16	79.64	0.80	0.83	0.76
TS 50-50 FS TB	79.27	79.49	0.79	0.80	0.79

Table 5.5 Top 5 MLP Model Results for 75-25 Split

Model	Train Acc	Test Acc	AUC	Recall	Specificity
TS 75-25 FS TA	85.17	85.39	0.78	0.62	0.94
TS 75-25 TA	85.12	84.94	0.77	0.61	0.93
TS 75-25 TB	84.25	84.25	0.76	0.57	0.94
SS 75-25 TA	83.89	83.36	0.75	0.59	0.92
TS 75-25 FS TB	84.05	83.87	0.75	0.56	0.94

Table 5.6 Top 5 MLP Model Results for No Split

Model	Train Acc	Test Acc	AUC	Recall	Specificity
TS No TA	92.82	92.56	0.70	0.41	0.99
TS No FS TA	92.54	92.39	0.68	0.38	0.99
TS No FS TB	92.01	92.00	0.66	0.34	0.99
TS No TB	92.03	91.95	0.66	0.33	0.99
SS No TA	92.26	92.06	0.65	0.32	0.99

5.2.3 Logistic Regression Breakdown

Following the breakdown of the MLP models, the following breaks down how different negative to positive ratio splits affected model performance for logistic regression. Tables 5.7, 5.8, and 5.9 displays the top 5 models for 50-50, 75-25, and No Split respectively. Of particular interest are the overall high recall scores and low specificity scores, particularly for the 75-25 and No Split models. Recall the results from Tables 5.5 and 5.6, where the MLP model results show Recall decreasing as Specificity remains upwards of 90% for the 75-25 and No Split models. While the 75-25 and No Split ratios for the MLP models resulted in models predicting higher counts of non-accidents, the 75-25 and No Split ratios for the logistic regression models show the opposite, an inclination to predict higher counts of accidents, resulting in overall low Specificity scores. This is due to the class weight parameter of the logistic regression model being set to balanced, as it attempted to balance out the difference between the class types. Because the 75-25 and No Split models have a higher amount of non-accidents (negative class), the model places greater weights on the accidents (positive class) to compensate. Clearly this compensation was too strong, resulting in models which predicted more accidents than non-accidents.

Table 5.7 Top 5 Logistic Regression Model Results for 50-50 Split

Model	Recall	Specificity
TS 50-50 TA	0.74	0.80
TS 50-50 TC	0.73	0.79
SS 50-50 TA	0.71	0.82
SS 50-50 TC	0.68	0.81
HS 50-50 TC	0.67	0.63

Table 5.8 Top 5 Logistic Regression Model Results for 75-25 Split

Model	Recall	Specificity
TS 75-25 TA	0.89	0.57
TS 75-25 TC	0.89	0.56
SS 75-25 TA	0.88	0.59
SS 75-25 TC	0.87	0.58
TS 75-25 TD	0.85	0.41

Table 5.9 Top 5 Logistic Regression Model Results for No Split

Model	Recall	Specificity
SS No TA	0.96	0.32
SS No TC	0.95	0.31
TS No TA	0.96	0.30
TS No TC	0.96	0.29
HS No TC	0.94	0.18

5.3 Prediction Implementation Results

To better see how these prediction models would perform when forecasting, the first week of January 2020 was used as a prediction sprint. Note that the data used for these models covers all accident records in Chattanooga from 2017 to 2019. Forecasting results refers to the proper implementation of a predictive model to predict, or forecast, future vehicular accidents.

5.3.1 MLP Forecasting Results

Figures 5.3, 5.4, and 5.5 show the results of utilizing the models listed in Tables 5.4, 5.5, and 5.6. The upper graph in each figure shows the Recall scores for each model across the 7 days, while the lower graph in each figure shows the Specificity scores for each model. For Figure 5.3, the overall best performing model is Total Shift 50-50 with Feature Selection on Test A, notated by the red line. Regardless of not having the consistently highest Recall and Specificity scores across the 7 days, the TS 50-50 FS TA results show consistently acceptable performance abilities across the week long prediction period, while the other models have inconsistent results. For Figure 5.4, the overall best performing model is Spatial Shift 75-25 with Feature Selection on Test A, notated by the blue line. As noted above, the best performing model is selected by the most consistently acceptable performance abilities. Lastly for Figure 5.5, there is no best performing model as every model showed has such poor performance abilities that none of the models would be viable for implementation.

Figure 5.3 MLP Forecasts for 50-50 Models from Table 5.4

Figure 5.4 MLP Forecasts for 75-25 Models from Table 5.5

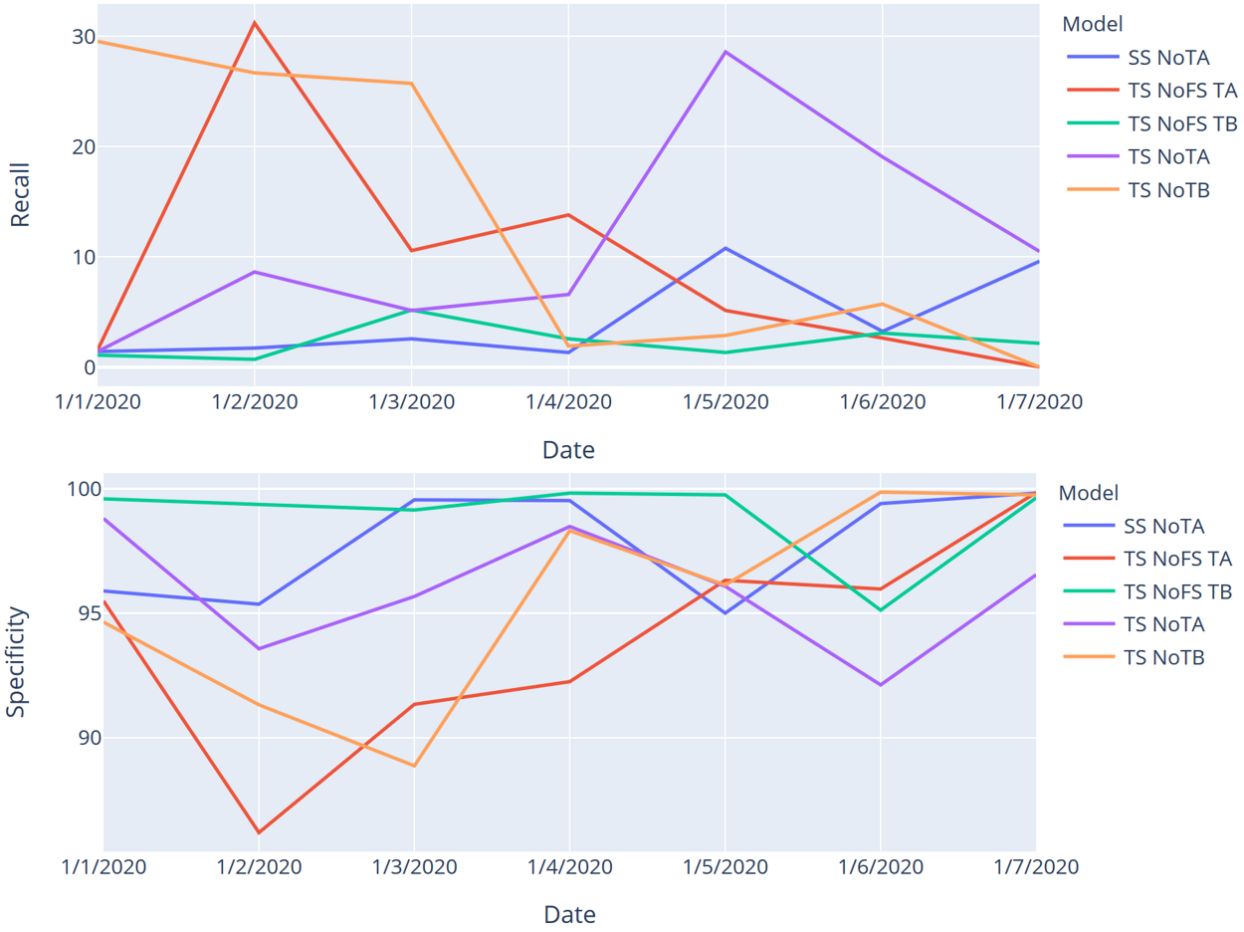

Figure 5.5 MLP Forecasts for No Split Models from Table 5.6

5.3.2 Logistic Regression Forecasting Results

Figures 5.6, 5.7, and 5.8 show the forecasting results for the first week of January for the 50-50, 75-25, and No Split logistic regression models, respectively. Using the same evaluation methods as the MLP models in the previous section, the best overall performing model for logistic regression is the Spatial Shift 5050 TC, notated by the green line in Figure 5.6. For Figure 5.7, the best performing model is Spatial Shift 75-25 TC, notated by the red line. Lastly, for Figure 5.8, the best performing model is Spatial Shift No Split TC, notated by the green line.

Figure 5.6 Logistic Regression Forecasts for 50-50 Models from Table 5.7

Figure 5.7 Logistic Regression Forecasts for 75-25 Models from Table 5.8

Figure 5.8 Logistic Regression Forecasts for No Split Models from Table 5.9

Of interest concerning the logistic regression models is the very high similarity between the Recall and Specificity scores of Figure 5.6 and Figure 5.8. The reported Recall and Specificity values are nearly identical across the prediction period, with only minor deviations in reported values. Additionally, the reported Recall and Specificity values in Figure 5.7, while easily distinguishable from the other two logistic regression forecast figures, are similar enough to the other logistic regression forecast figures that it can be concluded that the split between positive and negative values in the dataset does not yield as significant of a change as it does in the MLP models. Note in Figures 5.3, 5.4, and 5.5, the differences between the reported Recall and Specificity values are significantly altered based on which data ratio split is used.

5.4 Performance Discrepancies

5.4.1 MLP Model Performances

Ultimately, the Total Shift 50-50 Split with Feature Selection on Test A (TS 50-50 FS TA) model yielded the most acceptable prediction results for the MLP models. Table 5.10 shows the top 15 most significant variables for the TS 50-50 FS TA model. Of note is that the most significant variables are those related to location and time specific information, namely Join_Count and Hour. Join_Count acts as a historical accident count per hex number, therefore its significance is not surprising. These most significant variables bring to light a potential issue with the variables used: why bother with weather if it is not significant? To test this, a model using only the non weather variables from Table 5.10 was created and used to predict accidents for January 1st to 7th 2020, henceforth known as Weather-Exclusion. Table 5.11 shows the results of the Weather-Exclusion model in comparison to the best performing model for prediction implementation, TS 50-50 FS TA. While the TP and FN averages for the Weather-Exclusion model are more preferable to the TS 50-50 FS TA model, they are statistically outweighed by the larger differences in TN and FP. The overall performance of the TS 50-50 FS TA model is preferable to the Weather-Exclusion model, as the TS 50-50 FS TA model has 240 more True Negatives and 241 fewer False Positives, at the cost of 9 fewer True Positives and 10 more False Negatives across the week of predictions. Therefore, by including weather information the model is more balanced in terms of prediction accuracy, as it is less inclined to have a higher prediction count for accidents.

Additionally, there are a high amount of variables which are ranked significant, yet contain the same or similar information. For example, DayFrame is simply an aggregated version of Hour, so the presence of both variables so high on the list of significance is perplexing. The inclusion of potentially redundant variables is not detrimental to a model's usability, as

Table 5.10 Top 15 Most Important Variables for TS 50-50 FS TA

Rank	TS 50-50 FS TA
1	Join_Count
2	Hour
3	DayFrame
4	Latitude
5	Longitude
6	Grid_Num
7	Unix
8	humidity
9	windSpeed
10	uvIndex
11	temperature
12	dewPoint
13	pressure
14	visibility
15	cloudCover

those variables present information in inherently different ways. In other words, the potentially redundant variables use different scales. For example, latitude and longitude use the standard GPS coordinate scale, while Grid_Num is an integer from 1 to 694. Of additional concern is the contradictory nature of the TS 50-50 FS TA model, as according to [11,26,27], having a proper class balance is crucial for prediction performance. One would assume that for accident predictions, having a higher number of negative samples would appropriately balance the accident dataset and assist in reflecting the rare nature of accident occurrence. Indeed, this is not the case for the above mentioned best MLP model, and it is likely due to the inherently random nature of accidents. There can be one to one hundred different causes behind an accident happening, and even the most seemingly insignificant factor can be the determining cause of an accident. When a prediction model is created, the model can only take into consideration what factors are provided, resulting in the model assuming the present features are the only features at play. However in real world scenarios, there can be several dozen more factors as play that the prediction model was not prepared to handle, such as driver sex, age, mental state, vehicle age, condition, etc. Additionally, there are

scenarios where the factors the model was created with are not the strongest contributing factors behind the accident. Ultimately, the applicability of the accident prediction model is limited by the historical trends and currently available information of accidents.

Table 5.11 MLP Model Forecast Averages for Jan 1 to Jan 7 2020

Model Name	TP	FN	TN	FP	Recall	Specificity
TS 50-50 FS TA	49	35	3753	668	60.18	84.89
Weather-Exclusion	58	25	3513	909	70.92	79.44

5.4.2 Logistic Regression Model Performances

The most perplexing discoveries of the logistic regression models are the types of models returned as best performing as well as the overall performance of the different ratio types. Note in Tables 5.7, 5.8, and 5.9, only the models with a ratio split of 50-50 yielded well balanced Recall and Specificity values. With the best performing model, in terms of testing and training, was the TS 50-50 TA model, which was also the best performing MLP model. However, the implementation of the logistic regression models shows quite contradictory information regarding initial performance.

Table 5.12 Logistic Regression Forecast Averages for Jan 1 to Jan 7 2020

Model Name	TP	FN	TN	FP	Recall	Specificity
SS 5050 TC	57	26	3586	712	70.01	83.43
SS 7525 TC	58	25	3564	735	71.21	83.43
SS No TC	58	25	3574	725	71.11	83.13
SS 7525 TA	49	35	3586	713	60.68	83.36
SS 5050 TA	49	35	3640	659	60.45	84.65

Table 5.12 show the averages for the top 5 best models for logistic regression. The best overall model for logistic regression is the Spatial Shift 5050 TC. The reason for this is due to the specific average confusion matrix values, similar to the Weather-Exclusion model versus the TS 5050 FS TA model. SS 5050 TC has on average fewer False Positives and more True Negatives at the small cost of one less True Positive and one more False Negative when compared to the other two best performing logistic regression models. Furthermore, and of considerably more interest, is the contradictory nature of the reportedly best test type. Recall from the previous section that discussed the importance of weather variables for MLP model performance, and how the inclusion of said weather variables led to a more balanced prediction model. The logistic regression forecasts go completely against this notion as seen by models SS 7525 TA and SS 5050 TA from Table 5.12 having a lower Recall average, a lower True Positive average, and a higher False Negative average. Additionally, the logistic regression models show the significance of location related explanatory variables, meaning even simplistic regression models are capable of viable vehicular accident prediction.

5.4.3 The Superior Model

Table 5.13 shows a direct comparison between the best performing MLP model and the best performing Logistic Regression (LR) model in terms of forecasting capability. Note these values are the average scores for forecasts covering January 1st to January 31st 2020. Between the two, the LR model has on average 7 more True Positives, 6 fewer False Negatives, 188 fewer True Negatives, and 61 more False Positives. Overall, the LR model provides superior performance for vehicular accident prediction. However, it could be argued that the MLP model still has viability dependent on the desires of the end user. While the overall performance is better for the LR model, the MLP model does have the benefit of being more accurate at predicting non-accidents, while the LR model tends to have on average a higher prediction count of accidents.

Table 5.13 MLP vs Logistic Regression Forecast Averages for Jan 1 to Jan 31, 2020

Model Name	TP	FN	TN	FP	Recall	Specificity
SS 5050 TC (LR)	59	30	3641	721	66.72	83.43
TS 5050 FS TA (MLP)	52	36	3829	660	59.79	85.29

Figures 5.9 and 5.10 show the Recall and Specificity scores for the two models from Table 5.13, respectively. Of note in Figure 5.10, while the MLP model had overall higher Specificity values, those values varied wildly with many deep dips in Specificity across the month of prediction, whereas the Specificity scores for the LR model are much more consistent.

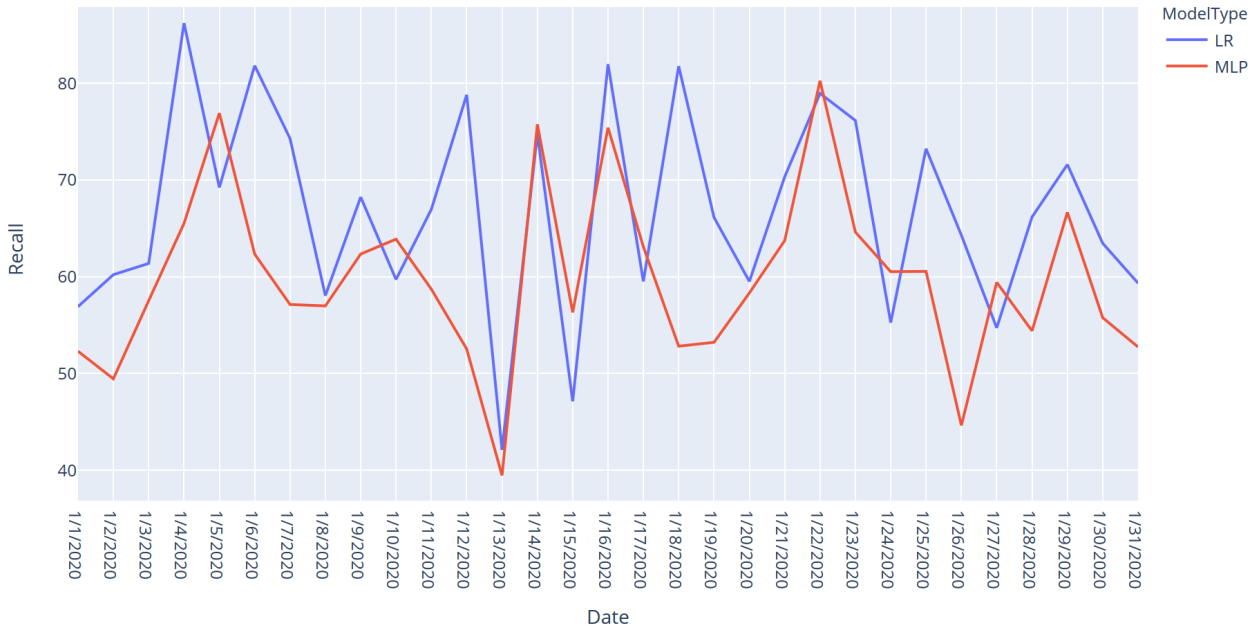

Figure 5.9 January Forecast Recall Scores for Models from Table 5.13

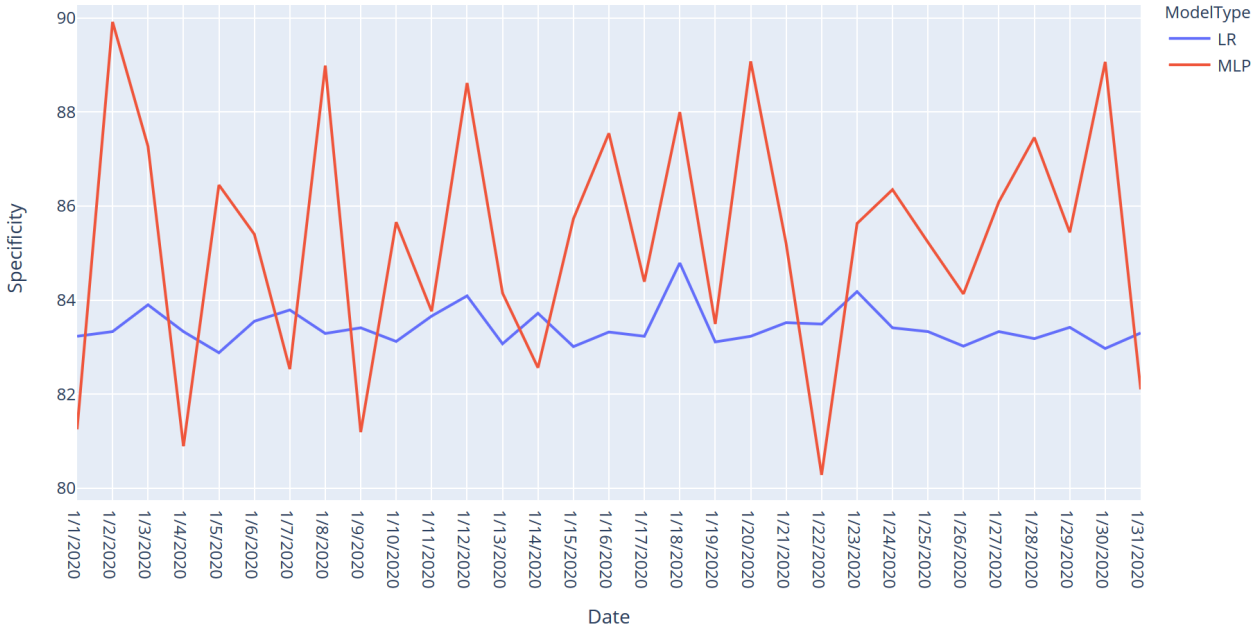

Figure 5.10 January Forecast Specificity Scores for Models from Table 5.13

5.4.4 Evaluation Metric Choices

Regarding the decision to use Recall and Specificity over Precision and F1 Score, the latter two scores for the forecasting predictions were very low across most models explored. There are two reasons for the F1 Score and Precision values being so low for forecasting. The first is that F1 and Precision do not take into consideration negative predictions, those being False Negatives and True Negatives. The predictive model for APP predicts, overall, more non-accidents than accidents as there are going to be more non-accidents than there are accidents in a given day. This feature led to some models that indeed did have high Precision and F1 Score values, yet very low Specificity values, negating the their viability. The second reason for low F1 and Precision values is due to how predictions are used for APP. For the two models discussed above in Table 5.13, there is an average of 691 False Positives (FP). While there are other predictive models which had lower FP, they had lower TP and higher FN as a result. For the purposes of APP, the forecasting scheme does not need to be so highly specific that it says where an accident *will* be with exact precision. The goal of APP is to provide local law enforcement with a service that shows them where

accidents are most *likely* to occur, with the effect of reducing the time it takes for emergency responders to reach these accident sites. Typically, police officers are assigned rather large areas to patrol during the day. APP would provide these officers with a smaller area of patrol coverage, reducing their overall response time and, ideally, having these officers a mere few streets away from an accident site.

CHAPTER 6

Conclusion

6.1 Closing Thoughts

A researcher has many tools at his/her disposal when working with rare event case data. These tools assist the researcher in balancing the classes present in rare event data, allowing for greater analytical accuracy and data exploration. The main tool for rare event data class balancing is data sampling, which involves altering the distribution of the training samples in a dataset. The least utilized version of sampling for rare event data is negative sampling, which involves generating negative entries from altering the spatial and temporal information of positive entries. While primarily and historically used in natural language processors and word2vec projects, negative sampling is steadily growing in the field of smart cities applications in regards to rare event case data, primarily for vehicle accident analysis. Where the former application of negative sampling involves the splicing of data subsections for analysis, the latter involves generating non-inherently present data. In other words, for rare event data, negative sampling provides the means of generating data records where none previously existed. For example as seen through this paper, *existing* yet *not present* non-accident records are generated by altering the spatial and temporal information of *existing* and *present* accident records.

As demonstrated in this work, there exists a plethora of methods for generating these negative entries, all of which stem from simple variable manipulation. These methods range from the brute force approach of generating literally any and every possible negative entry available, to the more streamlined and systematic approach. In regards to vehicular acci-

dents, these systematic approaches involve altering the hour, date, and location variables for accident records. Additionally, the inclusion of negative to positive data ratio splits and variable importance algorithms provided significant insight into what types of variables are most important for vehicular accident prediction, as well as what types of predictive models work best with different variables. Ultimately, this work demonstrates the relative ease of implementing negative sampling for vehicular accident prediction. Furthermore, it explores the effects of different data manipulation strategies and negative sampling methods on the predictive capabilities of different prediction models. Additionally, because no highly specific or difficult to obtain data was used in the creation of APP, most other cities or counties are able to implement this project for their area of study.

Overall, the best performing predictive model was a Logistic Regression model, with Spatial Shift negative sampling, with the positive to negative data ratio set to an even 50-50 split, while using only the temporal and spatial variables surrounding an accident entry. Despite the common idea of having a greater number of negative entries to act as a balancing agent for rare event case data, it was found that for predicting accidents, having an even split between negative to positive entries yielded the greatest potential for vehicle accident prediction. This can be seen by the Training/Testing results for the MLP and LR models in Tables 5.2 and 5.3, as well as the Forecasting results for MLP and LR models in Figures 5.3 through 5.8.

While the LR model won in overall predictive performance, the MLP model still holds viability for accident prediction given its slightly higher accuracy for predicting non-accident. For the MLP model, similar to the LR model, it was found that using an even 50-50 positive to negative data ratio was the best in predicting accidents. However, the Total Shift MLP models saw the greatest predictive capabilities, with all possible variables being present. Furthermore, the application of an ExtraTreesClassifier Feature Selection algorithm enhanced the model's general performance without restricting the its real world predictive capabilities.

Finally, regardless of the performance during a predictive model's creation, implement-

ing a predictive model in a real world setting will yield varying results. This is due to two concepts: the chaotically random nature of vehicular accidents and variable presence assumption. Chaotic randomness is the main draw back of studying vehicular accidents. There can be a single or even countless factors at play which can cause a vehicular accident. This can be a simple case of a distracted driver, or a mass collection of different minute mistakes from multiple parties that culminate together to cause an accident. Regardless of how fine tuned a model is, how many explanatory variables are used, and how powerful a machine is, it is simply impossible to perfectly predict something as chaotically random as vehicular accidents, at least with today's technology. Variable presence assumption is the concept of a predictive model assuming all variables provided during creation are the only variables at play during a rare event. For example, if a predictive model is created using only location and rain, that model will only be able to use location and rain to predict vehicular accidents, as those are the only variables it has to learn from. Essentially, a model will assume the variables present during creation are the only factors at play which can cause a vehicular accident, when in reality that is far from the case.

6.2 Future Work

While this work focused on the implementation of negative sampling through the lens of a vehicular accident prediction project, negative sampling has yet to find implementation in other rare event case scenarios. In general, regarding vehicular accident prediction, the future work possibilities of the negative sampling process tie into the limitations of the project that it is being applied to. In the case of APP, additional information pertaining to driver specific and vehicle specific data would yield additional potential for negative sample generation and data manipulation methods, along with providing the predictive model the additional information it needs to enhance its performance. Furthermore, the addition of more explanatory variables for vehicular accidents would overall provide a much needed boost

to the project's insight ability. At the time of writing this work, most variables pertaining to human data, vehicle specific data, and other commonly used roadway data (e.g., traffic volume) is not viable for use. The data itself is either not available or it is highly infrequent and incomplete, negating any potential analytical power. A potential remedy for this issue would be the utilization of Open Street Map to add in additional roadway information for Chattanooga, TN. Lastly, the process of creating code to generate negative samples, while not overly complex, is a lengthy process to get correct, and involves a significant amount of code. Future endeavors could seek to simplify this process and even streamline the code itself to an easily replicable algorithm which would need only minute edits from project to project.

REFERENCES

[1] S. Remanan, "Logistic regression: A simplified approach using python," Sep 2018. [Online]. Available: https://towardsdatascience.com/logistic-regression-a-simplified-approach-using-python-c4bc81a87c31

[2] A. Lazarevic, J. Srivastava, and V. Kumar, "Data mining for analysis of rare events: A case study in security, financial and medical applications," *Slides of PAKDD tutorial*, vol. 72, 2004.

[3] M. Maalouf, "Rare events and imbalanced datasets: an overview," 10 2015.

[4] G. King and L. Zeng, "Logistic regression in rare events data," *Political analysis*, vol. 9, no. 2, pp. 137–163, 2001.

[5] Y. Goldberg and O. Levy, "word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.

[6] K. Xu, Y. Feng, S. Huang, and D. Zhao, "Semantic relation classification via convolutional neural networks with simple negative sampling," *arXiv preprint arXiv:1506.07650*, 2015.

[7] C. Dyer, "Notes on noise contrastive estimation and negative sampling," *arXiv preprint arXiv:1410.8251*, 2014.

[8] M. Sama, M. Saeidi, T. Togia, and R. Kulkarni, "The effect of negative sampling strategy on the performance of the deep structured semantic model," 01 2017.

[9] Z. Yuan, X. Zhou, T. Yang, and J. Tamerius, "Predicting traffic accidents through heterogeneous urban data: A case study," 2017.

[10] A. Hebert, T. Guedon, T. Glatard, and B. Jaumard, "High-resolution road vehicle collision prediction for the city of montreal," *2019 IEEE International Conference on Big Data (Big Data)*, Dec 2019. [Online]. Available: http://dx.doi.org/10.1109/BigData47090.2019.9006009

[11] H. Meng, X. Wang, and X. Wang, "Expressway crash prediction based on traffic big data," in *Proceedings of the 2018 International Conference on Signal Processing and Machine Learning*. Association for Computing Machinery, 2018, p. 11 to 16.

[12] M. M. Ahmed and M. A. Abdel-Aty, "The viability of using automatic vehicle identification data for real-time crash prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 459–468, 2012.

[13] A. Theofilatos, G. Yannis, P. Kopelias, and F. Papadimitriou, "Predicting road accidents: a rare-events modeling approach," *Transportation research procedia*, vol. 14, pp. 3399–3405, 2016.

[14] S.-h. Park, S.-m. Kim, and Y.-g. Ha, "Highway traffic accident prediction using vds big data analysis," *The Journal of Supercomputing*, vol. 72, no. 7, pp. 2815–2831, 2016.

[15] F. Bouchet, J. Rolland, and J. Wouters, "Rare event sampling methods," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 29, no. 8, p. 080402, 2019. [Online]. Available: https://doi.org/10.1063/1.5120509

[16] "Generating negative samples." [Online]. Available: http://deepdive.stanford.edu/generating_negative_examples

[17] T. Au, M.-L. I. Chin, and G. Ma, "Mining rare events data by sampling and boosting: a case study," in *International Conference on Information Systems, Technology and Management*. Springer, 2010, pp. 373–379.

[18] S. Glen, "Stratified random sample: Definition, examples," Dec 2013. [Online]. Available: https://www.statisticshowto.com/stratified-random-sample/

[19] J. Brownlee, "Logistic regression for machine learning," Apr 2016. [Online]. Available: https://machinelearningmastery.com/logistic-regression-for-machine-learning/

[20] D. Graupe, *Principles of artificial neural networks*. World Scientific, 2013, vol. 7.

[21] J. Brownlee, "Crash course on multi-layer perceptron neural networks," 2016. [Online]. Available: https://machinelearningmastery.com/neural-networks-crash-course

[22] "Neural network bias: Bias neuron, overfitting, and underfitting." [Online]. Available: https://missinglink.ai/guides/neural-network-concepts/neural-network-bias-bias-neuron-overfitting-underfitting

[23] J. Roland, P. Way, and M. Sartipi, "Studying the effects of weather and roadway geometrics on daily accidents." Proceedings of Cyber-Physical Systems and Internet-of-Things, 2019.

[24] "tf.keras.optimizers.nadam — tensorflow core v2.3.0," 2015. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Nadam

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[26] C. Ranjan, "Extreme rare event classification using autoencoders in keras," 2019. [Online]. Available: https://towardsdatascience.com/extreme-rare-event-classification-using-autoencoders-in-keras-a565b386f098

[27] D. Wilson, "Using machine learning to predict car accident risk," 2019. [Online]. Available: https://medium.com/geoai/using-machine-learning-to-predict-car-accident-risk-4d92c91a7d57

[28] G. VanBelle, L. Fisher, P. Heagerty, and T. Lumley, *Biostatistics*, 2nd ed. Hoboken, New Jersey: John Wiley and Sons, Inc., 2004.

VITA

Jeremy Roland graduated from Signal Mountain High School in 2014, studying under the International Baccalaureate (IB) program offered there. He graduated from the University of Tennessee at Chattanooga in December 2018 with a Bachelor's Degree in Software Systems and a Minor in Business Administration. During his senior year, he began working as a student assistant researcher at the Smart Communications and Analysis Lab (SCAL), later rebranded as the Center for Urban Informatics and Progress (CUIP), at UTC. This research experience led him to pursue his Master's degree at UTC for Computer Science. Through working with CUIP, he gained invaluable experience, met some amazing people, and had some amazing experiences. He graduated from UTC with a Master's Degree in Computer Science in December 2020.