NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

NANYANG BUSINESS SCHOOL

BC2407 Analytics II

Seminar Group 4 Team 4

Prepared By:
ERNEST ANG CHENG HAN (U1921310H)
JAYLENE KOH (U1921460F)
ONG QING QING (U1910396H)
SIAH WEN ZHAO (U1810947F)
LOW QI HUI (U1910298D)

Table of Contents

1. Executive Summary

1.1 Business Statement

Electric vehicles (EVs) have been gaining traction in recent years due to the shift in consumers' mindset and emphasis on leaving a smaller carbon footprint in the wake of global warming. Tesla is an American company focused on accelerating the transition to sustainable energy, with EVs as its main product offering (Tesla, n.d.). However, we see many instances of Tesla's EVs getting into autonomous car crashes in the media due to AI and software malfunctions. This phenomenon has negatively affected Tesla's reputation and sales, especially in a crucial time where it has to consolidate market share while its competitors are trying to catch up in terms of technology and introduction of EV models.

Hence, this report aims to push forth a machine learning solution, SAFE-T. SAFE-T is a machine learning solution that will generate relevant and useful insights on malware detection in its Tesla car computer chips by leveraging on past malware detection data and data from other departments to allow its engineers and client relations to take a detective rather than reactive approach to possible malware and malfunctions. This would further decrease accidents on the road attributed to computer chips in EVs, and **boost revenue for Tesla by enhancing its safety reputation**.

1.2 Analytics Process

For the purpose of proof of concept, SAFE-T took in a dataset which focused on predicting presence of malware depending on properties of a computer chip. Firstly, data preparation is carried out to make this dataset more relevant for Tesla. We did not remove any values to ensure integrity and distribution of the data is not compromised. Both univariate and bivariate analysis is carried out to see the impact of the independent variables on the dependent variable under the past malware detection dataset to retrieve insights and solidify possible correlations. In addition to data visualization, we performed association rules mining through apriori algorithm to gain deeper insights on the associations between features in the dataset and the common trends for when the chip has a detection.

Prior to constructing our prediction model, we performed Synthesized Minority Oversampling Technique (SMOTE) to resolve the issue of imbalance data to complement the stream processing method of data collection. We also performed an automated feature selection method, Boruta, to reduce our model complexity to better Tesla's needs. For the construction of the model, we examined the applicability of multiple models, namely neural networks, random forest, logistic regression. We then further enhanced the performance of these models using methods such as hyperparameter tuning, k-fold cross validation and bootstrapping. The criterion in choosing the best mode is that the best model should ideally have highest model accuracy, higher false positive than false negative and fastest time taken for prediction. **We found random forest to be the most appropriate data for use in our context.**

1.3 Insights and Recommendations

Overall, SAFE-T will be a proprietary form of safety technology that could help Tesla consolidate market share as the safest EV on the road by providing latest insights of malware detection in regard to its own computer chip. By consolidating historical and real-time data regarding malware detection through **stream processing**, Tesla would be able to better prevent possible malware attacks on its cars, which is important as failure to do so would result in fatal car accidents. However, just like all business analytics solutions, SAFE-T should only serve as a recommendation that safety engineers and clients can capitalize during their journey in a Tesla.

2. Introduction of Business Problem

2.1 Macroanalysis of the Electric Vehicle Industry

Increasing demands for environmentally friendly technology, coupled with the ability to finally do so with feasibility and profitability in mind has catalyzed the rise of EV players in the car industry. According to a study by Forbes last October 2020, the global EV sales rose 65% from 2017 to 2018 for a total of 2.1 million vehicles (Cohen, 2020). To perform a macroanalysis, we first have to ask ourselves the following questions:

1. Are the pull factors towards EVs by key stakeholders effective? Why or why not?
2. Are the push factors given to consumers convincing enough? Why or why not?
3. What can be done to further push for faster uptake of EVs?

A deep dive into reasons behind this rise reveal the following reasons that will ensure the enduring presence of EV makers:

2.1.1 Global Warming

A report has found that the US transportation sector produces nearly thirty percent of all US global warming emissions, more than almost any other sector, and this has led to unpredictable and sometimes, damaging weather behavior, as seen by the increase in severity of natural disasters such as wildfires in California (EPA, 2020). This alarming phenomenon has led to consumer shift in behaviour to products which leave a smaller carbon footprint in their daily travels, such as the use of EVs. According to Ergon Energy, EVs have a plethora of benefits even outside the context of building an environmentally friendly future: from reducing pollution from toxic vehicle-related emissions, to reducing reliance on petroleum, EVs are also cheaper to run and maintain in the long run (Ergon, n.d.).

2.1.2 Strong Regulatory Support

This shift towards EVs is strongly supported, and increasingly enforced in powerful countries around the world. In 2020, California Governor Newsom announced that he would join 15 countries in phasing out gas powered cars by 2035 (S&P Global, 2021). This is also seen even in Singapore, where the country-state recently announced in its Budget 2021 that $30 million was planned to be invested over the next five years on EVs and other EV-related initiatives. These plans would lower the cost of owning an EV and encourage the shift towards purchasing one (Singapore Budget 2021, n.d.).

These are very strong pull factors that would attract consumers to purchase. However, there are still challenges and obstacles that are can be addressed to solidify the push factors for consumers themselves to adopt the change. Through our research on the business problem which will be further elaborated on in Section 2.2, we have arrived at the main issue that is deterring people from taking the leap of faith to switch over to EVs: Safety.

2.2 Analysis of the Business Problem

We have identified two root causes linked to our business problem:

2.2.1 The Danger of an Autonomous Vehicle

Picture 1.1: Snapshot of Deadly Autonomous Driving Accident (BBC, 2020)

Picture 1.1 illustrates what consumers see increasingly on the news: the dangers of owning a self-driving vehicle. Tesla's strongest differentiating factor lies in it being one of the pioneers in having autonomous driving cars on the road, and these accidents directly impact the safety reputation of Tesla since it has the largest market share in the EV industry. Electric and autonomous vehicles are usually synonymous as technology-focused early adopters such as Tesla want both innovations in the same car. It is also easier to implement autonomous features on EVs due to its original emphasis on technology to increase driving range and battery efficiency. While autonomous vehicles statistically have a lower accident death rate, individual accidents are getting higher attention and bad publicity due to its novel nature and it being the talk of the town with many reputable car brands shifting R&D in this direction. This creates an impression of Tesla being unsafe.

2.2.2 Susceptibility to Malware Attacks

EVs rely heavily on its computer chips to increase the efficiency and usage of electricity. It is also being used for all safety and autonomous functions. Tesla cars receive software updates over the air on a need basis determined by its software engineers. The car's connectivity also possesses as a threat as it would be susceptible to external hacking over the internal. As such, this advancement of technology has become a double-edged sword as consumers are able to enjoy a better riding experience, but at the same time would be exposed to the dangers of introducing too much technology into a vehicle. For example, software and AI malfunctions have been the main cause of EV accidents. In fact, Tesla recently had to recall 135,000 vehicles due its in-car system bricking which compromised advanced assistant driver and safety features (Bell, 2021).

2.2.3 Increased Competition

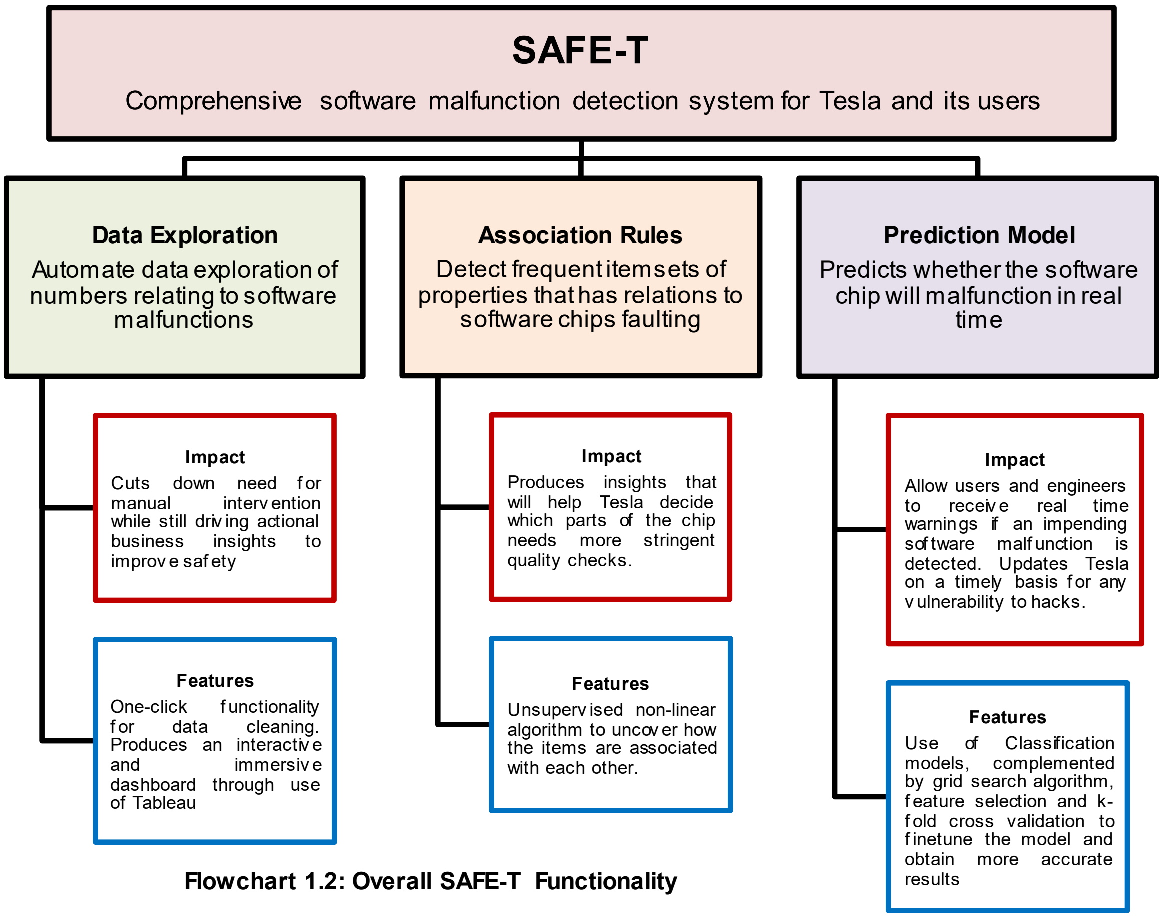
Tesla is at risk of being overtaken in terms of market share with more reputable brands stepping into the EV industry. A prime example would be the Ford Mustang Mach-E, which accounted for almost all of Tesla's share loss from of 12% in the US (Business Insider, 2021). As such, Tesla is under pressure to solidify its first mover advantage and market share and should seek to differentiate from its competitors as much as it can.

2.3 Overall Business Challenge

Hence, the overall business challenge identified for Tesla is how it can effectively address the concerns of its target market in terms of safety, and generate higher revenue with more cars sold, while increasing its competitive advantage through an advanced suite of safety detection solutions aimed at detecting possible malware in its computer chips.

2.4 Proposed Analytics Solution

As such, this projects seeks to address the problem statement and propose SAFE-T, a software malfunction detection system tailored to Tesla's needs, as seen in **Flowchart 1.2**.


SAFE-T
Comprehensive software malfunction detection system for Tesla and its users
Data Exploration
Automate data exploration of numbers relating to software malfunctions
Association Rules
Detect frequent itemsets of properties that has relations to software chips faulting
Prediction Model
Predicts whether the software chip will malfunction in real time
Impact
Cuts down need for manual intervention while still driving actional business insights to improve safety
Features
One-click functionality for data cleaning. Produces an interactive and immersive dashboard through use of Tableau
Impact
Produces insights that will help Tesla decide which parts of the chip needs more stringent quality checks.
Features
Unsupervised non-linear algorithm to uncover how the items are associated with each other.
Impact
Allow users and engineers to receive real time warnings if an impending software malfunction is detected. Updates Tesla on a timely basis for any vulnerability to hacks.
Features
Use of Classification models, complemented by grid search algorithm, feature selection and k-fold cross validation to finetune the model and obtain more accurate results

Flowchart 1.2: Overall SAFE-T Functionality

To better explain how SAFE-T can address the business problems above, we have classified the benefits of having SAFE-T into two pillars:

2.4.1 Safer Cars: Higher Revenue

With Tesla being at risk of being overtaken by competitors, SAFE-T provides an additional layer of security to consumers in terms of safety through two main tenets: empowering engineers and client relations department with the latest data and predictions. For the engineering department, it means being able to leverage on big data to improve its computer chips in resistance to malware. For the client relations, they are better able to inform customer of possible malfunctions and portray a strong emphasis on safety. Having safer cars after implementing SAFE-T, consumers will be more than likely to purchase cars from Tesla. This will certainly increase the revenue from Tesla Cars.

2.4.2 Consolidate Market Share

While other EV companies are still at the testing and initial delivery phase, safety of cars might not be the number one priority. Tesla will have an edge over their competitors by providing an additional form of insurance to their consumers by delivering safer cars, allowing it to be ahead in this technological arms race.

3. Data Cleaning and Exploration

Our group has chosen a potential application of SAFE-T on a past malware detection dataset, where we used past data regarding properties of a chip to see what permutations of properties would lead to a higher probability of malware in a chip. By running the dataset through SAFE-T, we can detect the effect each property has on a chip and convert past data into useful insights for Tesla to leverage on to enhance safety. The procedures and steps outlined are similar across all our models.

3.1 Initial Dataset

The following datasets were used as part of our Proof of Concept for SAFE-T:

Classification of Malwares (CLaMP)
Link to dataset: https://www.kaggle.com/saurabhshahane/classification-of-malwares
Data dictionary: Table 1.1 in Appendix
The dataset consists of

Using the Classification of Malwares (CLaMP) data set as a proof of concept for our solution SAFET, the output variable would be a categorical (1/0) on whether the chip has a malware detection, 'MalwareDetection'.

3.2 Forged Variables

Our dataset is meant to be a proof of concept, and in attempting to prepare the dataset to align more to the business objectives of Tesla being able to monitor its fleet of cars, we first changed some columns to include possible data that Tesla would include to enhance safety monitoring across all cars. The ability to do so is due to the fact that our solution is data agnostic. SAFE-T is able to pull data from multiple departments, such as the sales department which will record the location and model of the car, together with the computer chip production information, and allow us to draw insights that we would not expect. Hence, as seen in **Table 1.3**, we changed some of the columns to simulate this and allow us to draw business insights that would go beyond the computer chip, such as enabling the client relations department to also understand possible safety concerns in real time. Equipped with this knowledge across departments, they are better able to put forth a strong emphasis on safety. By consolidating more data points, SAFE-T will also be increasing its model prediction capability.

Original Column	Changed Column	Rationale
Machine	Models	To simulate the three main models that Tesla has. This will be the data pulled in from the sales department.
ImageBase	ChargeCycles	Both are numerical variables. Data pulled from car diagnostic data already regularly sent to

		Tesla.
SizeOfImage	CarMileage	Both are numerical variables. Data pulled from car diagnostic data already regularly sent to Tesla.
MajorSubsystemVersion	SoftwareVersion	To simulate the four chips software versions created by Tesla since its inception. Data pulled from existing software logging system in Tesla.
E-lfanew	Country	E-lfanew had the same number of levels as the number of countries Tesla is in. Countries were populated in accordance with Tesla's revenue. Aka, factor level with highest proportion in the dataset was renamed to the United States, which has raked in the highest revenue for Tesla yearly. Data will be pulled from the sales department.

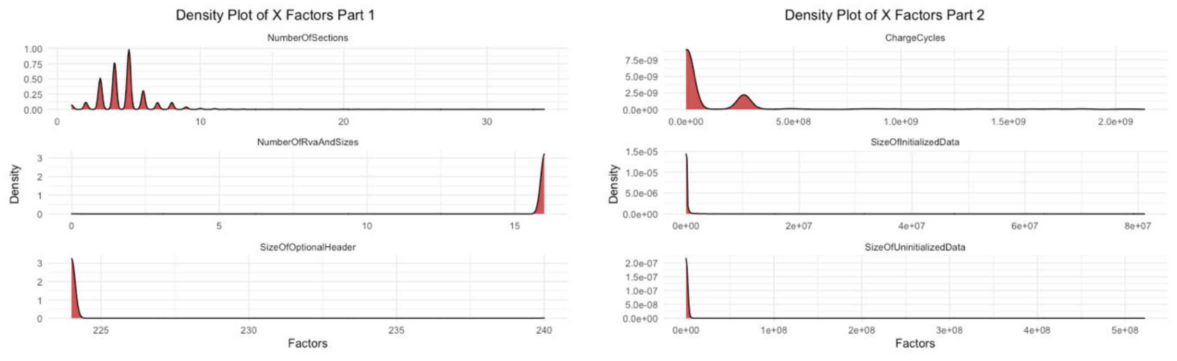
Table 1.3: Change in Columns to Suit Business Objectives

3.3 Treatment of Redundant Data

We focused on removing columns with more than 90% NA values as well as columns with single factor as they were likely to have no impact on the model and would not provide much useful insights. These columns are 'E_res', 'E_res2', 'E_Magic' and 'E_crlc'. Thereafter, we converted the columns to their appropriate data types before proceeding with exploratory data analysis (EDA). Each column's data type can be found in the **Appendix A: Data Dictionary**.

3.4 Numeric Variables: Univariate Exploration and Cleaning

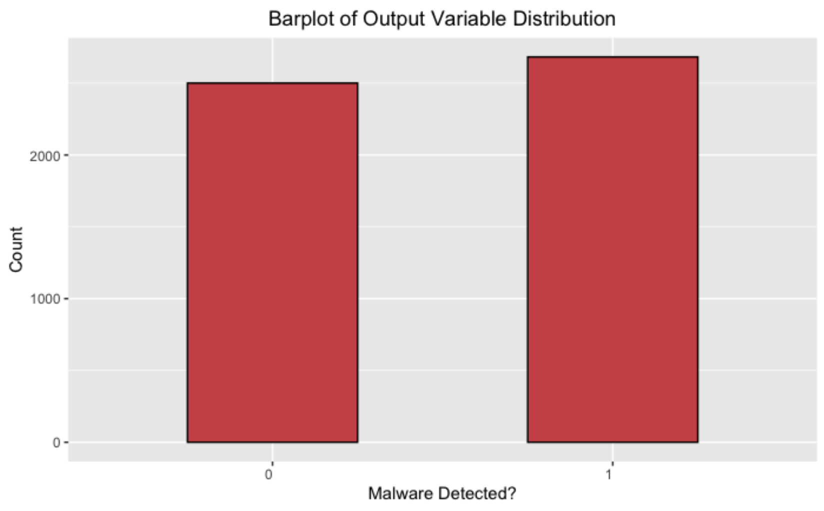
Analysing the numerical variables, we observed many extreme outliers as seen in the univariate kernel density plots as seen in **Plots 1.4**, resulting in a skewed distribution for multiple columns. Checking the outliers, there is no obvious indications that the outlier is due to incorrectly entered or measured data so we cannot confidently say that those values do not truly belong in the data set. We also hypothesised that the computer chip properties, such as 'SizeofInitializedData', would not differ too much from chip to chip, hence the distribution of the values for computer chip parts should be relatively skewed and we should not try to balance out these distributions.



Plots 1.4: Kernel density plot of computer chip properties

Furthermore, we do not have an abundance of data. Our data set is relatively small at 5,184 rows and removing the outliers might affect the distribution of our data which in turn would affect our models later on. Hence, we decided to not attempt to balance the data.

3.5 Categorical Variables: Univariate Exploration and Cleaning



Plot 1.5: Barplot of Output Variable, MalwareDetected

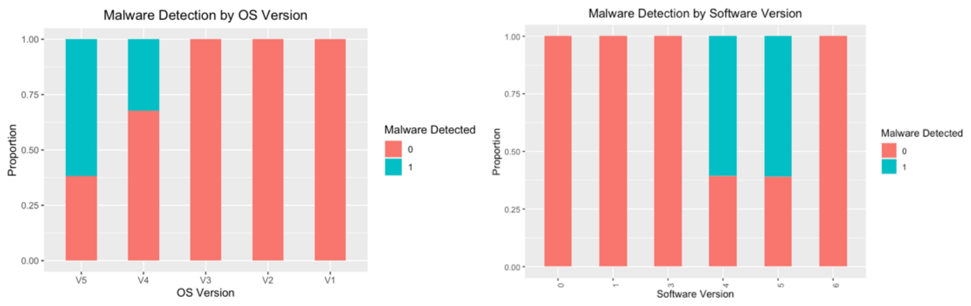
The distribution **(Plot 1.5)** shows the frequency of the Output variable. We see that our dataset is relatively balanced, which will allow us to ensure that our Linear Regression model used later will not be affected in terms of its model intercept. However, as Tesla gathers more data, we are likely to see more '0's than '1's as the output variable as it is unlikely for almost half of Tesla's cars to be hit with malware. The usefulness of our solution will remains as car manufacturers should aim for zero accidents, rather than keeping it below a certain threshold. Hence, in this case, we will then have to apply SMOTE to balance the data, which will be elaborated further on in the report.

4. Data Visualisation

4.1 Categorical Variables: Bivariate Analysis and Visualisation

By using bivariate plots of Malware Detection by categorical variables, we can observe and derive important business relations. Note that these are possible insights that could easily be observed with visualisation.

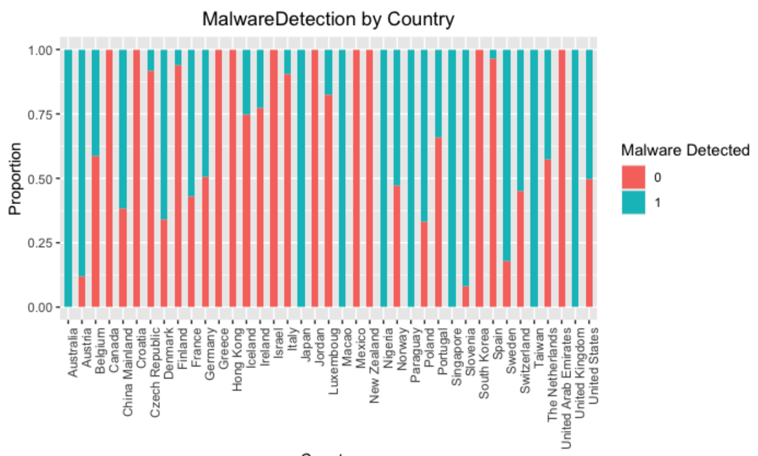
4.1.1 Malware Detection by OS Version and Software Version



Plot 1.6: Bivariate barplot of OS Version and Software Version against MalwareDetection

Practical business insight can be drawn from bivariate categorical barplots. From **Plot 1.6** later software versions and OS versions are more likely to have malware detected in the computer chips. This suggests that the latest software developed by Tesla could likely have compromised the safety aspect of the chip for more functionality. However, given that this data was edited by us, it may not reflect the actual distribution of Tesla's. This is the beauty of SAFE-T, as it is able to automate the churning out of such insights. Tesla could then draw from these data and focus on fixing its software and OS versions, for example.

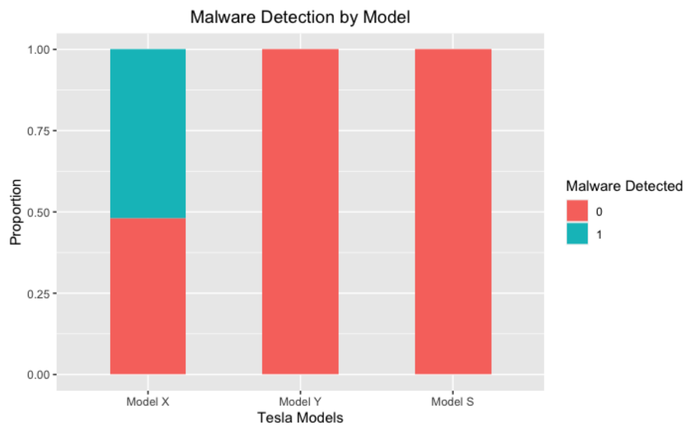
4.1.2 Malware Detection by Country



Plot 1.7: Malware Detection by Country

In **Plot 1.7**, we can observe that Countries such as Australia, Japan, Macao, Nigeria, Paraguay, Singapore, Taiwan and United Kingdom all showed a high number and proportion of Malware detected while countries such as Canada, Croatia, Greece, Hong Kong, Israel, Jordan, Mexico, New Zealand, South Korea and United Arab Emirates showed the lowest proportion of Malware Detected. Tesla could then act on this information by proactively checking the possible malware issues by notifying its local engineers, ensuring that this solution is tailored to each country Tesla expands to.

4.1.3 Malware Detection by Model



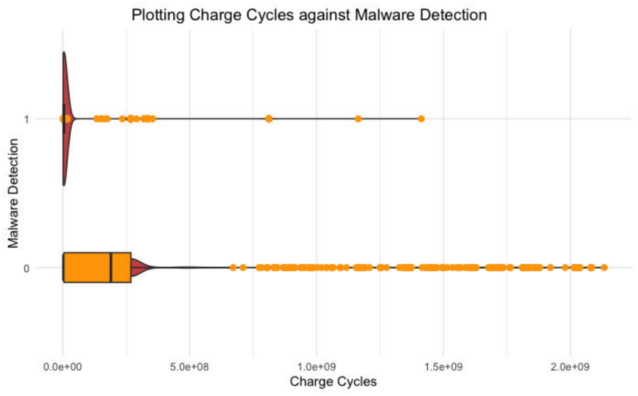
Plot 1.8: Malware Detection by Model

When plotting Malware Detection by Model **(Plot 1.8),** we can conclude that Model X is more likely to cause problems due to the larger proportion of Malware Detected as compared to Model Y and Model S. Although Model S being the earliest model released, followed by Model X and Y, it can be seen that the newer the model of Tesla Cars does not mean that it will cause lesser problems. This corroborates with the fact that newer cars, along with newer computer chips and software versions might lead to more safety concerns. Hence, SAFE-T allows any user to draw such useful insights just by looking at these graphs.

4.2 Numeric Variables: Bivariate Analysis and Visualization

Next, using bivariate plots of malware detection by the numeric variables, we can observe and derive important business relationships and insights.

4.2.1 Malware Detection by Charge Cycles

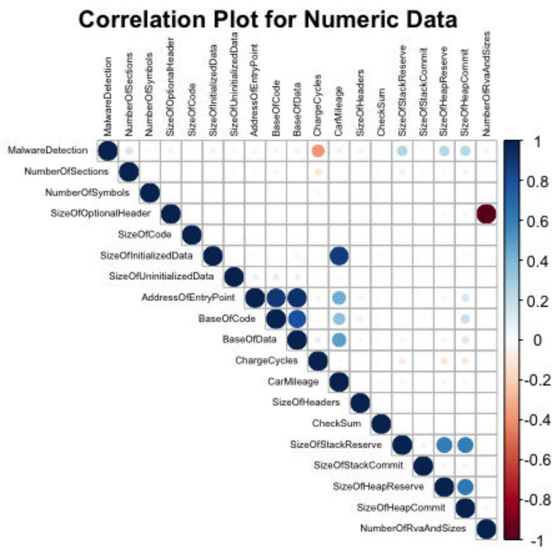


Plot 1.9: Charge Cycle by Malware Detection

In **Plot 1.9**, we can see that the distribution of chips with malware detected are mostly concentrated in the lower range of charge cycles, while it is more evenly spread out for chips with no malware detected. In fact, there is a strong presence of outliers when there is no malware detected. This could suggest that newer cars with lesser charge cycles are more susceptible to malware detection, and this agrees with our discovery in the previous section that cars with newer OS and software version are more susceptible to malware.

4.2.2 Correlation Plot

We first check for possible multicollinearity between the predictor variables using a correlation matrix. Generally, we observe slight correlations between our variables, however majority are in acceptable ranges about 0.



Plot 2.1: Correlation Plot for Numerical Data

From the correlation matrix, 'SizeOfOptionalHeader' and 'NumberOfAndSizes' are highly negatively correlated variables, while 'AddressOfEntryPoint', 'BaseOfData', 'BaseOfCode' are highly correlated variables, ‘SizeOfInitializedData’ and ‘CarMileage’ are also highly correlated variables. Intuitive insights include high correlation of ‘SizeOfInitializedData’ and ‘CarMileage’ as a car which has been used longer would also have more data stored in its chip due to more historical travel data.

We place an emphasis on using variables with lesser correlation, to prevent indirect biases when training our model, which is why we will check and may remove such columns later during the modeling process. Reducing the impact of multicollinearity is important as we intend to make use of regression analysis. Independent variables help isolate the factors from each other, ensuring that the relationships derived from our analysis are accurate and replicable, enabling us to build a statistically sound and robust predictive model.

5. Association Rules

Association rule mining is an unsupervised machine learning technique to find potentially useful rules automatically from the data. Rule mining is used for uncovering associations between features in datasets and common trends in the transactions. By implementing the ***apriori algorithm***, a list of rules is generated as an output to function as if then-scenarios regarding the objects in the set. We want to gain deeper insights on which set of features have a strong relation with malware, in addition to the insights gained from data visualization.

5.1 Data preparation for Association Rules

As our data is in a data frame, it is required of us to transform our data into transactional data. Given that that there can be large ranges of numbers in each numerical column, our aim in association rule mining is thus to identify a specific range in each numerical column to establish a stronger relation in regard to the response variable, ‘MalwareDetection. Hence, we chose to break the continuous range for each numeric column into discrete range instead by splitting the continuous range into intervals before transforming into transactional data. This would allow us to draw understandable and actionable insights.

5.2 Findings from Association Rules

Using the arules package in R, we found rules in the form ‘if a then b’ where a and b are two events. We want to find strong rules that shows” if having these properties in the chip then malware detected”. Before we look into our findings, we first define some terms in the results. Support tells us how often we will be able to apply the association rule, whereas confidence tells us the strength of the rule in situations where it is applicable. To reduce the possible number of rules we have to analyse as it would be computationally heavy and inefficient to calculate the statistics for all rules possible, we apply the apriori algorithm. This algorithm makes use of the apriori principle which states that ‘If an itemset is frequent, then all of its subsets must also be frequent”. To use this algorithm, we specify the minimum support and confidence to be 0.45 and 0.8. A relatively high support and confidence is set as from the data visualization above, we can see that the properties are frequently and strongly related to the detection of malware.

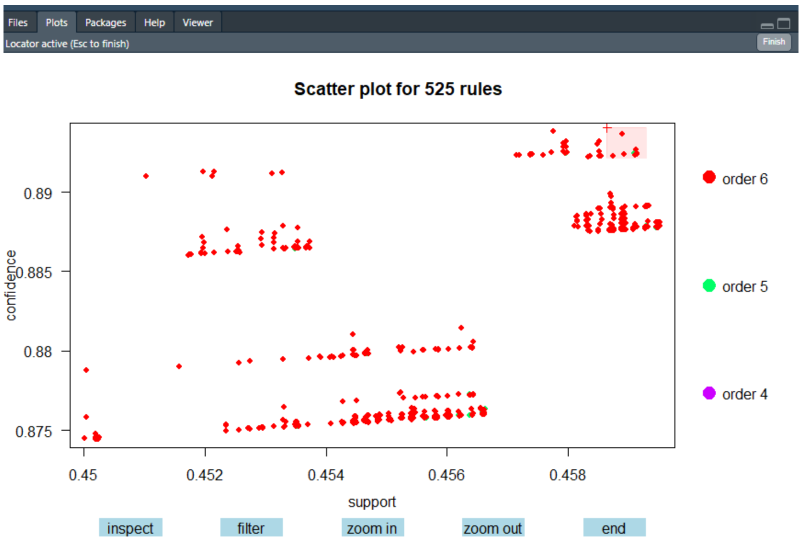
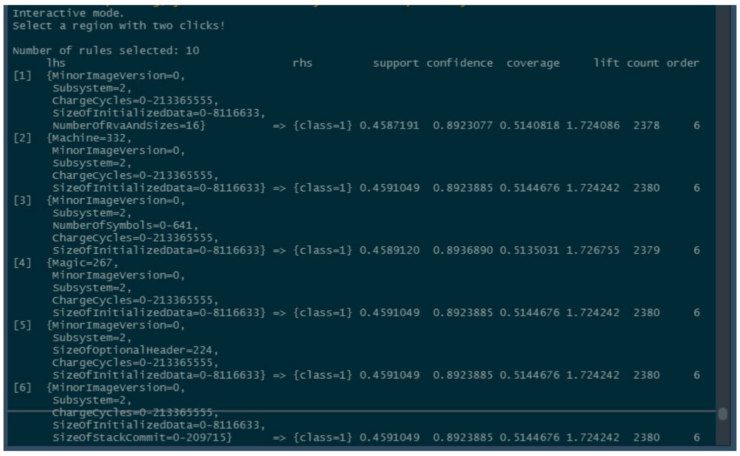


Figure 2.2: TwoKey Interactive Association Rules Plot

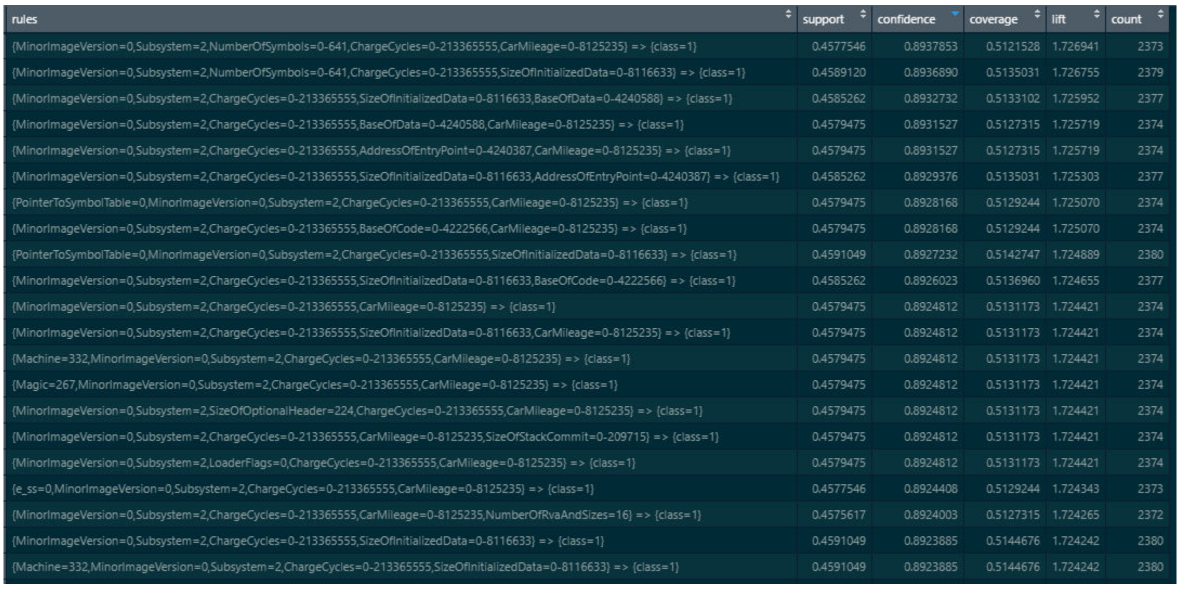
Order refers to the number of items in the dataset. The TwoKey plot is a way of displaying all discovered association rules at once, while also providing the means to review and manage them **(Figure 2.2).** It is a powerful tool in order to get a first overview of the distribution of confidence and support. Features such as separate groups of rules or outliers are detected immediately. From the two-key plot graph generated, we can see that the rules obtained mostly have 5 properties on the left-hand side. The detection of malware frequently appears, and the relationship is highly true with 5 properties. We can see that support and confidence are positively correlated. For the set of properties that appears more frequently, there is higher confidence to say that the relationship is true, and we can conclude that set of properties does lead to malware detection.



```
Interactive mode.
Select a region with two clicks!

Number of rules selected: 10
     lhs                                 rhs         support confidence  coverage     lift count order
[1]  {MinorImageVersion=0,
      Subsystem=2,
      ChargeCycles=0-213365555,
      SizeOfInitializedData=0-8116633,
      NumberOfRvaAndSizes=16}          => {class=1} 0.4587191  0.8923077 0.5140818 1.724086  2378     6
[2]  {Machine=332,
      MinorImageVersion=0,
      Subsystem=2,
      ChargeCycles=0-213365555,
      SizeOfInitializedData=0-8116633} => {class=1} 0.4591049  0.8923885 0.5144676 1.724242  2380     6
[3]  {MinorImageVersion=0,
      Subsystem=2,
      NumberOfSymbols=0-641,
      ChargeCycles=0-213365555,
      SizeOfInitializedData=0-8116633} => {class=1} 0.4589120  0.8936890 0.5135031 1.726755  2379     6
[4]  {Magic=267,
      MinorImageVersion=0,
      Subsystem=2,
      ChargeCycles=0-213365555,
      SizeOfInitializedData=0-8116633} => {class=1} 0.4591049  0.8923885 0.5144676 1.724242  2380     6
[5]  {MinorImageVersion=0,
      Subsystem=2,
      SizeOfOptionalHeader=224,
      ChargeCycles=0-213365555,
      SizeOfInitializedData=0-8116633} => {class=1} 0.4591049  0.8923885 0.5144676 1.724242  2380     6
[6]  {MinorImageVersion=0,
      Subsystem=2,
      ChargeCycles=0-213365555,
      SizeOfInitializedData=0-8116633,
      SizeOfStackCommit=0-209715}      => {class=1} 0.4591049  0.8923885 0.5144676 1.724242  2380     6
```

Figure 2.3: View rules individually

The interactive feature of this graph allows us to individual rules by selecting them and clicking the inspect button or inspect sets of rules by selecting a rectangular region of the plot and clicking the inspect button. We can see that generally, the set of properties that most frequently appears and we have the highest confidence of saying that these properties will give a malware detection is true, are shown on the right **(Figure 2.3).**



rules	support	confidence	coverage	lift	count
{MinorImageVersion=0,Subsystem=2,NumberOfSymbols=0-641,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4577546	0.8937853	0.5121528	1.726941	2373
{MinorImageVersion=0,Subsystem=2,NumberOfSymbols=0-641,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633} => {class=1}	0.4589120	0.8936890	0.5135031	1.726755	2379
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633,BaseOfData=0-4240588} => {class=1}	0.4585262	0.8932732	0.5133102	1.725952	2377
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,BaseOfData=0-4240588,CarMileage=0-8125235} => {class=1}	0.4579475	0.8931527	0.5127315	1.725719	2374
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,AddressOfEntryPoint=0-4240387,CarMileage=0-8125235} => {class=1}	0.4579475	0.8931527	0.5127315	1.725719	2374
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633,AddressOfEntryPoint=0-4240387} => {class=1}	0.4585262	0.8929376	0.5135031	1.725303	2377
{PointerToSymbolTable=0,MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4579475	0.8928168	0.5129244	1.725070	2374
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,BaseOfCode=0-4222566,CarMileage=0-8125235} => {class=1}	0.4579475	0.8928168	0.5129244	1.725070	2374
{PointerToSymbolTable=0,MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633} => {class=1}	0.4591049	0.8927232	0.5142747	1.724889	2380
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633,BaseOfCode=0-4222566} => {class=1}	0.4585262	0.8926023	0.5136960	1.724655	2377
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4579475	0.8924812	0.5131173	1.724421	2374
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633,CarMileage=0-8125235} => {class=1}	0.4579475	0.8924812	0.5131173	1.724421	2374
{Machine=332,MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4579475	0.8924812	0.5131173	1.724421	2374
{Magic=267,MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4579475	0.8924812	0.5131173	1.724421	2374
{MinorImageVersion=0,Subsystem=2,SizeOfOptionalHeader=224,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4579475	0.8924812	0.5131173	1.724421	2374
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,CarMileage=0-8125235,SizeOfStackCommit=0-209715} => {class=1}	0.4579475	0.8924812	0.5131173	1.724421	2374
{MinorImageVersion=0,Subsystem=2,LoaderFlags=0,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4579475	0.8924812	0.5131173	1.724421	2374
{e_ss=0,MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,CarMileage=0-8125235} => {class=1}	0.4577546	0.8924408	0.5129244	1.724343	2373
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,CarMileage=0-8125235,NumberOfRvaAndSizes=16} => {class=1}	0.4575617	0.8924003	0.5127315	1.724265	2372
{MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633} => {class=1}	0.4591049	0.8923885	0.5144676	1.724242	2380
{Machine=332,MinorImageVersion=0,Subsystem=2,ChargeCycles=0-213365555,SizeOfInitializedData=0-8116633} => {class=1}	0.4591049	0.8923885	0.5144676	1.724242	2380

Figure 2.4: Rules in Table Form

We can also view the rules in table form **(Figure 2.4).** As useful rules are generally rules with high confidence, sorted in descending order of confidence, we can see that {MinorImageVersion=0, Subsystem=2, NumberOfSymbols=0-641, ChargeCycles=0-213365555, CarMileage=0-8125235} => {MalwareDetection=1} appears very frequently in the database at 0.4578. We are the most confident at 89.39% in saying that {MinorImageVersion=0, Subsystem=2, NumberOfSymbols=0-641, ChargeCycles=0-213365555, CarMileage=0-8125235} will give a malware detection is true. It tells us that the rule can be applied rather often as the support is high, and the high confidence tells us that the rule is strong in situations where it is applicable.

Association rule is especially helpful in helping us look out for the frequent patterns of properties that results in faulting. We can reduce the production of chips with these properties or alert the sales department to not engage in the sales of these chips. Vice versa, we can also set the right-hand side to show no detection to see which set of properties is least likely to give detection is true. And with that, we can gain insights on what kind of chips should be manufactured and used for electric vehicles that can significantly reduce malfunction.

6. Data preparation for models

6.1 SMOTE to balance imbalance output classes

Due to a very high volume of data we are expecting the model to receive, it may be unpredictable as to what kind of data we are conducting machine learning for data streams **(refer to Section 8.3)** on. It would not be efficient to manually check and balance the proportion of the MalwareDetection output each time the data set is changed as we expect new data to be continuously added to the data set. The challenge of working with imbalanced datasets is that most machine learning techniques will ignore, and in turn have poor performance on, the minority class, although typically it is performance on the minority class that is most important. Hence, we would automate the checking of the minority class and utilise Synthetic Minority Oversampling Technique (SMOTE) to synthesize new examples from the minority class prior to fitting the model.

6.2 Reducing Model Complexity Using Feature Selection

We implemented Boruta, a feature selection library in R which uses a variation of the Random Forest model to select the most significant variables with the highest predictive power. We chose Boruta as unlike most of the traditional feature selection algorithms which follow a minimal optimal method where they rely on a small subset of features which yields a minimal error on a chosen classifier (which may only apply to the training dataset), Boruta follows an all-relevant feature selection method where it captures all features which are in some circumstances relevant to the outcome variable.

Boruta helps reduce the number of explanatory variables in our model to mitigate the effects of overfitting by building predictive models free from correlated variables, biases and unwanted noise. The Feature Selection step also ensures that computational cost of modelling and model complexity is reduced. Model complexity refers to the number of features included in a given predictive model and the structure of the model. It also refers to the computational complexity. Since the trained model will be loaded in the chip of the vehicle, we want to reduce the complexity as our prediction model cannot be too big and complex due to the limited storage capacity and computational power of the chip. The figure below **(Figure 2.5)** shows a plot of the importance of all the selected attributes in ascending order relevant to the outcome.

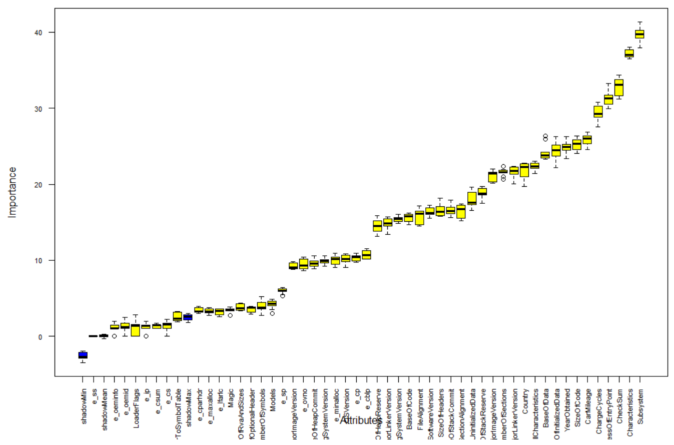
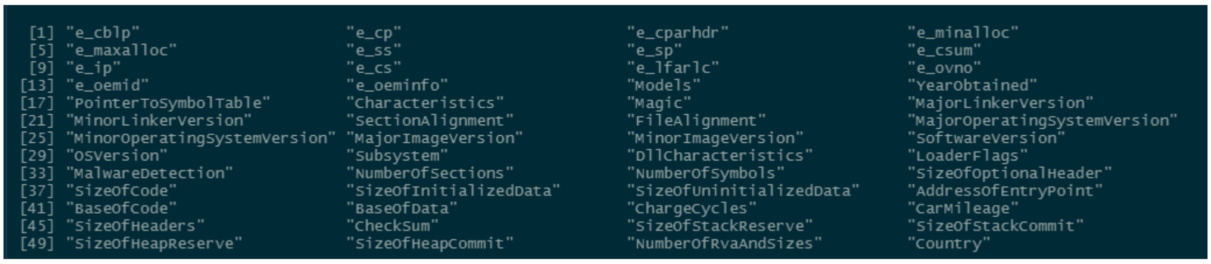


Figure 2.5: Importance of Selected Attributes

6.3 Final Dataset

To sum up, we did not remove the outliers, renamed some columns for relevancy, convert categorical data from strings to numeric form, reduced number of features using feature selection, and then used SMOTE to balance the output classes.

Hence, our final dataset will consist of these variables, where MalwareDetection is our output variable y, as seen in **Figure 2.6** below.



```
 [1] "e_cblp"                     "e_cp"                    "e_cparhdr"              "e_minalloc"
 [5] "e_maxalloc"                 "e_ss"                    "e_sp"                   "e_csum"
 [9] "e_ip"                       "e_cs"                    "e_lfarlc"               "e_ovno"
[13] "e_oemid"                    "e_oeminfo"               "Models"                 "YearObtained"
[17] "PointerToSymbolTable"       "Characteristics"         "Magic"                  "MajorLinkerVersion"
[21] "MinorLinkerVersion"         "SectionAlignment"        "FileAlignment"          "MajorOperatingSystemVersion"
[25] "MinorOperatingSystemVersion" "MajorImageVersion"      "MinorImageVersion"      "SoftwareVersion"
[29] "OSVersion"                  "Subsystem"               "DllCharacteristics"     "LoaderFlags"
[33] "MalwareDetection"           "NumberOfSections"        "NumberOfSymbols"        "SizeOfOptionalHeader"
[37] "SizeOfCode"                 "SizeOfInitializedData"   "SizeOfUninitializedData" "AddressOfEntryPoint"
[41] "BaseOfCode"                 "BaseOfData"              "ChargeCycles"           "CarMileage"
[45] "SizeOfHeaders"              "CheckSum"                "SizeOfStackReserve"     "SizeOfStackCommit"
[49] "SizeOfHeapReserve"          "SizeOfHeapCommit"        "NumberOfRvaAndSizes"    "Country"
```

Figure 2.6: Summary of Variables

The correlation plot of the final values **(Figure 2.7)** is as shown:

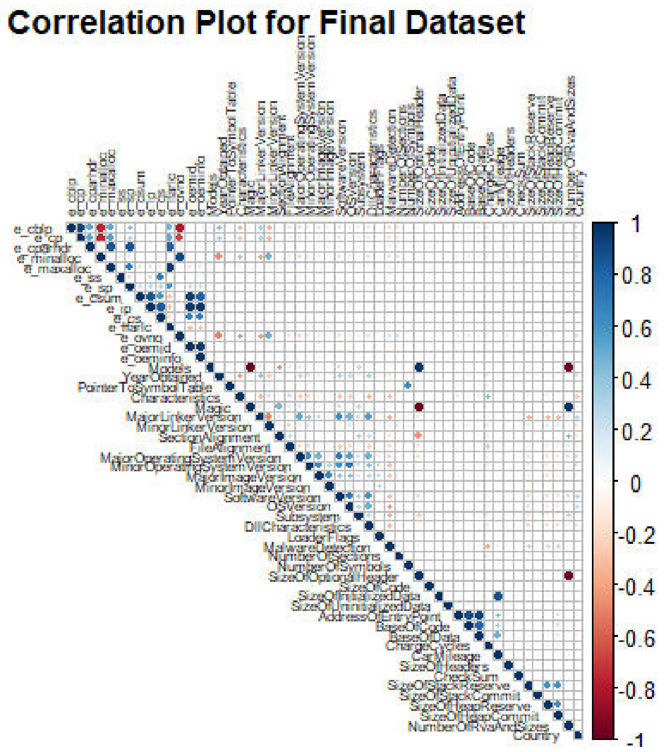


Figure 2.7: Correlation Plot for Final Dataset

We also confirm that our input variables are generally uncorrelated and independent from each other.

7. Proposed Predictive Models

SAFET leverages upon the state-of-the-art predictive techniques to classify and segregate the input variables to quickly and accurately predict the probability of chip malfunction in Tesla's electric vehicles. In order to determine which model provides an accurate representation of the data sets, all the three models that are Logistic Regression, Random Forest and Neural Network would be performed on the data.

7.1 Evaluating Model Performance

7.1.1 Confusion Matrix Comparisons

Confusion matrix allows us to derive a multitude of statistical measurements such as classification accuracy, type I error, type II error, and f-score which can be used to compare and evaluate the performances of the different models. Given that we are implementing multiple models, evaluation of confusion matrix is quintessential in primarily deriving the most efficient models to be used.

7.1.2 Time Taken for Prediction

Apart from the results of the confusion matrices, it is important to note that different predictive models require different quantities of processing power and time. This is another necessary evaluation factor in bringing balance and a more well-rounded conclusion to our selecting the best model for organizations to leverage on, given that a model which brings about the best accuracy and the lowest error rates may be too expensive and time consuming for firms to utilize, rendering it useless and unable to be leveraged on.

7.2 Model Pre-processing Techniques

7.2.1 Hyperparameter Tuning

Hyperparameter tuning, which chooses a set of optimal hyperparameters for a learning algorithm to control the learning process, is used on all 3 of our models for performance improvement. We made use of the Grid Search algorithm to achieve this, improving our general model performance while reducing the impact of overfitting. Grid search algorithm select the best parameters for our optimization problem from a list of parameter options that we provide, hence automating the 'trial-and-error' method.

7.2.2 K-fold cross validation

Here, we implemented a stratified 10-fold cross-validation on all our models with a fixed seed to determine the average classification accuracies across different folds. This cross-validation process gives us a general understanding of which of the 3 models may be more suitable for our problem statement, as all our data is used for both training and testing. This cross validation is also expected to result in a less biased or less optimistic estimate of the model skill than a traditional train test split. Furthermore, being able to achieve similar accuracies for k number of results would mean that our algorithm is consistent, and we can be confident that by training it on all the data set and deploying it in production will lead to similar performance.

7.3 Logistic Regression

Logistic regression was used as the one of the models to predict the binary classification of the "MalwareDetection" dependent variable. Additionally, we built this and every model that we are proposing twice: once after implementing feature selection and once without. For logistic regression, there was a slight improvement when implementing the variables identified with feature selection as seen below. This is unsurprising given that feature selection ensures the best subset of variables are used amongst the 50 given, as well as prevents overfitting the models with too many variables which in the process reduces model complexity.

Logistic Regression without feature selection:

	Actual		
Prediction		0	1
	0	720	55
	1	30	750

Accuracy: 0.9453, Type I: 0.0400 (False Positive), Type II: 0.0683 (False Negative)

Logistic Regression with feature selection:

	Actual		
Prediction		0	1
	0	701	38
	1	49	767

Accuracy: 0.9441, Type I: 0.0653 (False Positive), Type II: 0.0472 (False Negative)

Take note that for each of the 2 logistic regression models, the optimal train-test split of dataset was achieved via k-fold cross validation and the hyperparameters has been tuned optimally via Grid Search Algorithm. Also, during the above correlation plot, we found that "NumberOfRvaAndSizes" has a high correlation with another feature. Running **variance inflation factor** to check for multicollinearity, we found that there is indeed perfect multicollinearity. This issue is resolved when "NumberOfRvaAndSizes" is rejected during the feature selection process where it is removed from the model.

7.4 Random Forest

Random forest is a supervised learning algorithm which essentially builds a collection of classification and regression trees to predict the values of variables we desire. Each decision tree created in the process is used in the prediction process and the overall prediction of the forest of decision trees is determined either via a majority vote for classifying categorical variables or averaging the predicted continuous variables from each decision tree. Additionally, each sample of the dataset which was used to build each decision tree was obtained via the resampling technique – bootstrapping.

7.4.1 Bootstrapping

Bootstrapping is essential a random sampling technique with replacement which enables a fair estimation of the sample's distribution, along with its standard error and confidence interval given that the same sample of the dataset is reused and resampled. It is also a convenient method to lower the processing power and costs involved with repeating the sampling procedure to get other groups of sample data

Random Forest without feature selection:

	Actual		
Prediction		0	1
	0	729	5
	1	21	800

Accuracy: 0.9833, Type I: 0.0280 (False Positive), Type II: 0.0062 (False Negative)

Random Forest with feature selection:

	Actual		
Prediction		0	1
	0	728	3
	1	22	802

Accuracy: 0.9839, Type I: 0.0293 (False Positive), Type II: 0.0037 (False Negative)

For random forest, given that an ensemble or "forest" of decision trees are built in the process and these very decision trees utilize k-fold cross validation, we can be assured that the train-test split of the dataset is optimal. Again, to further optimize the performance of the random forest, its hyperparameters were tuned also with Grid Search Algorithm.

7.5 Neural Network

An artificial neural network is a type of deep learning machine learning algorithm which consists of a collection of artificial neural nodes which are used in predicting variables. These nodes are organized into different types of layers: input layer, hidden layers and output later. The input layer consists of the nodes which serves as the entry point for the input data which are processed and given an output value via activation functions, weights, and biases. Each of these output values are assigned to each node of the next layer, the hidden layer, which repeats the process of generating an output to the nodes of the next layer till the final layer of the neural network is reached, the output layer. Performing the activation function on the final sum of output values within the output layer gives the predicted value of the neural network. If the cost function of the neural network is too great, a built-in algorithm known as back propagation is invoked which basically helps the neural network to fine tune the weights, and biases of the neural network, to a point whereby its prediction performance is more accurate

7.5.1 Normalizing numeric columns

Before implementing this impressive algorithm, we need to normalize some of the numerical and continuous variables within the dataset. The purpose of this is to allow these types of variables to be on a somewhat common scale without distorting differences in the range of values. It is essential to the efficient running and execution of neural networks. In our project, we filtered the appropriate variables and applied Min-max normalization before implementing the neural network.

7.5.2 Finding optimal hidden nodes to reduce complexity of neural network

Increasing hidden layers can help improve the accuracy of the neural network but at the same time require more processing power and time along with resulting in a highly complexed and overfitted model to be used for prediction. As such, it is necessary to ensure that we account for this balance in adding more hidden layers and arriving at an accurate prediction which may not even be useful in the first place due to overfitting.

Neural Network without feature selection:

	Actual		
Prediction		0	1
	0	708	15
	1	42	790

Accuracy: 0.9633, Type I: 0.0560, Type II: 0.0186

Neural Network with feature selection:

	Actual		
Prediction		0	1
	0	718	17
	1	32	788

Accuracy: 0.9685, Type I: 0.0427, Type II: 0.0211

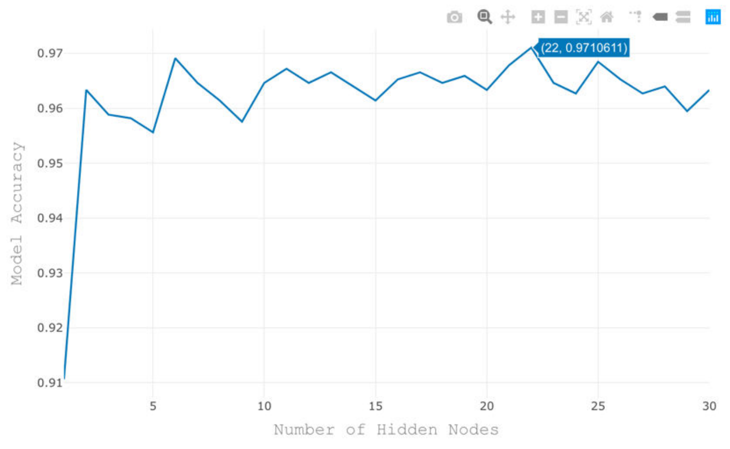


Figure 2.8: Neural Network Accuracy Graph

For both the neural networks, the number of hidden layers used were 22, given that it gave us the highest model accuracy as previously seen under 7.5.2 **(Figure 2.8).**

7.6 Model Evaluation

Feature selection is mostly useful in raising accuracy levels and reducing the complexity of our model. It generally increases type I but decreases type II errors as seen in most of the models. Detecting errors in general is crucial to prevent faults and disastrous accidents in electric vehicles but in our context, type II errors are more deadly than type I errors given that falsely determining that the electronic chip will not malfunction when it actually will be compared to falsely determining that the electronic chip will malfunction when it actually will not. The former would more likely contribute to explosions and accidents with electric vehicles while the latter would just result in a waste of resources but not injury and a potential loss of human lives. It would be more favorable for us to train a model with high accuracy and false positive, to which feature selection achieves in both areas.

	Logistic Regression	Random Forest	Neural Network
Accuracy	94.41%	98.39%	96.85%
Type I Error	6.53%	2.93%	4.27%
Type II Error	4.72%	0.37%	2.11%

Table 2.9: Overview of Models with Feature Selection

Referencing to the summarized table of model results in **Table 2.9**, Random Forest seems to be best model that has the highest accuracy and can be created in a relatively fair amount of time and computing power, with an impressive 98% classification accuracy, given that the Neural Network we propose requires 22 hidden nodes and yields a lower prediction accuracy and that logistic regression may be a simpler and easier model to build but yet again yield lower prediction accuracies.

Another point of comparisons between the models would be the time taken by each model to return the results **(Table 3.1).** This is especially important because in real life situation where the time taken to predict a malfunction can be the turning point as to whether a potentially fatal accident is impending or not. We want the minimum delay or lag as by the time the malfunction is detected, the accident would have happened or there may not be enough time to stop the accident. We want to detect a malfunction as fast as possible for more time allowance in preventing an accident. The table below shows a summary of the average time taken by each model to return a predicted result.

Model	Time Taken
Logistic Regression	6.163035 mins
Random Forest	20.45394 secs
Neural Network (22 hidden nodes)	1.615653 mins

Table 3.1: Model time taken

Random forest takes the shortest amount of time, compared to Neural Network and our Logistic Regression Model. Ideally, neural network would be the best model in keeping the lag time to the minimal but based on the other criteria mentioned above along with the fact that 22 hidden nodes is req, it loses to random forest based on complexity and accuracy. **Hence, random forest would still be the best model based on both criteria.**

8. Suggested Solutions and Implications

8.1 Assumptions

We made several assumptions while developing our solution:

1. Outliers represent an under sampled part of the data
2. Agnostic data from other departments can help SAFE-T derive new insights for safety insights
3. Safety is the top priority of Tesla

Given that this is a proof of concept, we did not have access to actual internal data from Tesla. With such a small dataset, we can only assume that the presence of multiple outliers is because our data represents an under-sampled part of the data, and that the accuracy of our model will increase with more data fitted into it. Furthermore, since Tesla is a relatively new company, we would assume that our dataset would be proportionately smaller as compared to other car manufacturers that have been in the game for much longer. To prevent our models from rejecting the actual dataset, we rigorously tested our models with multiple metrics and cross-validations to predict only significant features. As such, SAFE-T would comfortably adapt to Tesla’s unseen data.

Secondly, we utilised sales department data to locate and provide further information regarding which car and where the car is around the world. With Tesla’s sales achieving strong year-on-year growth, we believe the sales department will have the largest amount of available data besides the engineering and testing teams. SAFE-T would be able to make use of this large data as it can leverage on Big Data to create new insights and establish relationships that would not be easily uncovered without it.

Lastly, the assumption of safety being the utmost concern was made after intensive research of the EV industry at large. We ensured that our solution from beginning-to-end was tailored to the EV industry, as seen by how a large part of the model is based on a sophisticated computer chip which is mainly only used in EVs for advanced driving features such as autonomous driving. After research, we established that EV manufacturers placed safety as one of their top priorities as it was one of the biggest obstacles to mainstreaming their product.

8.2 Implications

As such, we believe that Tesla has not tapped into the full potential in terms of assuring clients of their vehicle's safety, resulting in underperforming revenue numbers and also having to spend time fighting lawsuits when accidents occur as a result of its driving features due to software malfunction.

Hence, SAFE-T aims to give Tesla an edge against other EV manufacturers:

1. Tesla would have the capability to tie in agnostic data to derive insights by pulling data from multiple departments to draw new insights which would not have been found manually through association rules and the machine learning model.
2. SAFE-T provides meaningful and actionable insights that would enable staff from other departments such as customer service to act on, such as tending to customer questions regarding safety of their purchased car, cutting down possible legal fees while boosting its safety reputation.
3. SAFE-T can eventually be transformed into a structured data processing system that could go beyond providing safety assurance, if it proven to be effective. For example, it could even be used for Financial Planning and Analysis if the right data is used.

8.3 Future Implementations: Stream Processing

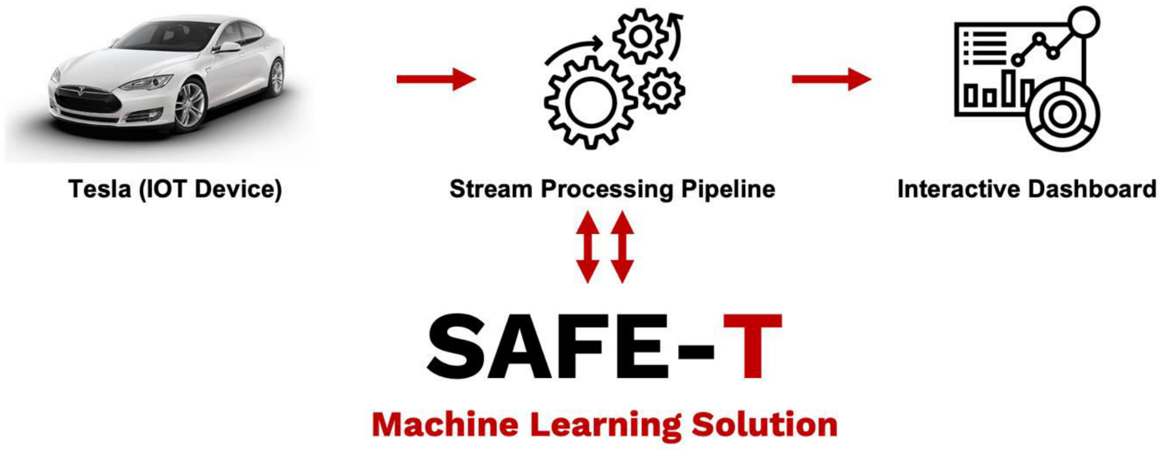


Figure 3.2: Overview of SAFE-T's Future Implementations

SAFE-T can be further enhanced by collecting the abovementioned data in real time through stream processing of data **(Figure 3.2)**. Stream processing is the processing of data in motion, and computing as it is produced or received in real time. This is especially applicable to safety as users and Tesla alike should be aware of any possible safety issues in real time, even a delay of 5 minutes could cause the unthinkable. This would allow Tesla to quickly react and push any remedies over the air into its cars so that any accidents due to malware could be provided. The models would be running in real time in both Tesla cars and Tesla headquarters, where data is being transferred between each other in real time. For now, however, our solution aims to provide an interactive dashboard that would perform data visualizations and monitor model accuracy for software engineers to act on (Appendix 11.6).

8.4: Business Performance Measures

Despite model accuracy being a strong factor for strategy success, we should also take note of other possible business measures that can be used to evaluate and improve the model functionality as seen in **Table 3.3**:

KPIs	Objectives	Metrics
Financial	Increase revenue with stronger reputation and maximise shareholder's wealth	1. Year on year revenue growth 2. Margin expanstion 3. Tesla share price performance
Customer statistics	Grow customer base by ensuring current ones are satisfied through detective rather than reactive customer care, and new ones are acquired	1. Customer satisfaction/online reviews 2. Text mining of market sentiment
Environmental, social and governance measures	Drive sustainable energy, and show care for the human life through enhanced safety and prediction measures.	1. Tangible ESG targets (E.g. Carbon Emissions saved, deaths avoided)

Table 3.3: Business Performance Measures

We believe that as long as the KPIs are hit, Tesla would definitely be able to remain as the leader in the EV industry. In the unlikely case that it does not, Tesla management can then step in to look at other factors besides safety and also finetune SAFE-T to possibly identify and target other facets of the business in order to gain market share.

9. Conclusion

SAFE-T is an intelligent and adaptable solution capable of providing valuable insights. With safety in mind, we set out to ensure that SAFE-T would be able to pull in more data from other departments in order to accurately identify which cars on the road are at risk of malware in real time, allowing us to provide Tesla with an intangible asset to its suite of safety tools.

In the future, we hope to integrate stream processing as the form of data collection and be able to process data coming in from Tesla cars all around the world in order to further boost the efficiency of SAFE-T's performance. Our goal is to provide Tesla with an edge against its increasingly strong competitors, with a solution that is robust, easy to integrate and most of all, reliable in the eyes of its drivers on the road.

10. Bibliography

British Broadcasting Corporation (2020, 16 September). *Uber Self-Driving Operator Charged over Fatal Crash.* Retrieved 2 April 2021*, from* https://www.bbc.com/news/technology-54175359

Business Insider (2021, 3 March). Tesla may be losing its electric-vehicle crown as Ford's Mustang Mach-E sales heat up. Retrieved 3 April 2021, from https://www.businessinsider.com/tesla-losing-electric-car-lead-ford-mustang-mach-e-sales-2021-3

Ergon Energy (n.d.). Benefits of Electric Vehicles. Retrieved 1 April 2021, from https://www.ergon.com.au/network/smarter-energy/electric-vehicles/benefits-of-electric-vehicles

Cohen, A. (2020, October 26). 'Plugging Into The Future: The EV Market Outlook'. Retrieved on 14 February, 2020, from https://www.forbes.com/sites/arielcohen/2020/10/26/plugging-into-the-future-the-electric-vehicle-market-outlook/?sh=90f27ca98121

KIA Corporation (2021). 'Are Electric Cars Dangerous?'. Retrieved on 20 February, 2020, from https://www.kia.com/dm/discover-kia/ask/are-electric-cars-dangerous.html

Singapore Government (2021, January 19). 'Budget 2021'. Retrieved on 20 February, 2021 from https://www.gov.sg/features/budget2021?utm_source=sustainability&utm_medium=amobee&utm_campaign=budget2021

S&P Global (2021, 7 January). Calif. Governor pitches $1.5b for hydrogen and electric vehicles infrastructure. Retrieved 1 April 2021, from https://www.spglobal.com/marketintelligence/en/news-insights/latest-news-headlines/calif-governor-pitches-1-5b-for-hydrogen-and-electric-vehicles-infrastructure-62002444

United States Environmental Protection Agency (n.d.). Sources of Greenhouse Gas Emissions. Retrieved 30 March 2021, from https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions

Trivedi, A. (2020, November 16). 'Is Everyone Back To Buying Cars? Think Again'. Retrieved on 22 February, 2021, from https://www.bloomberg.com/opinion/articles/2020-11-15/china-s-electric-vehicle-surge-excites-investors-but-look-again

K. G. (2019, August 19). Infographic: Hopes and fears around self-driving cars. Retrieved February 21, 2021, from https://www.roboticsbusinessreview.com/unmanned/unmanned- ground/infographic-hopes-and-fears-around.com

Bell, S. (2021, February 02). Tesla recalls 135,000 vehicles over Faulty chip that can Brick Touchscreen. Retrieved February 21, 2021, from https://www.carscoops.com/2021/02/tesla-recalls 135000-vehicles-over-faulty-chip-that-can-brick-touchscreen/

11. Appendix

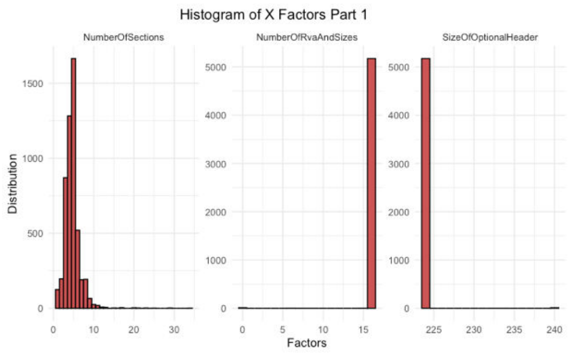
11.1 Data Dictionaries

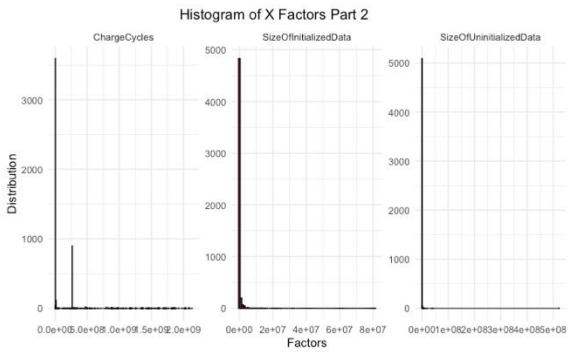
Table 1.1: Malware Classification Dictionary (Grouped into understandable terms)

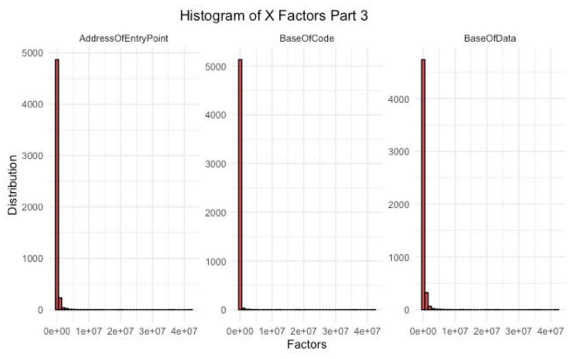
Column	No. of Vars	Meaning
EX-DOS File Variables	19	Various properties which create an image backup of data
File Headers	7	Various properties regarding structure of Portable Executable format
Optional Headers	29	Various properties which record the operating system the computer chip is operating in
Total Variables	55	

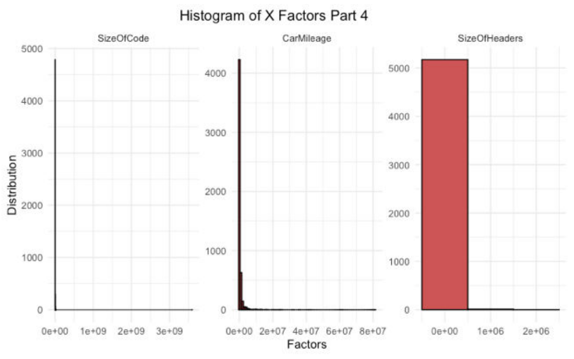
11.2 Data Visualizations for Unused Graphs

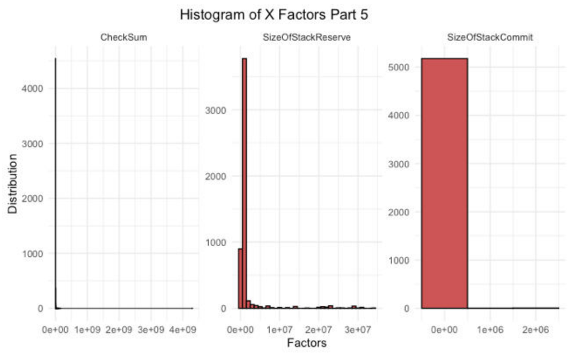
Univariate Histogram Analysis

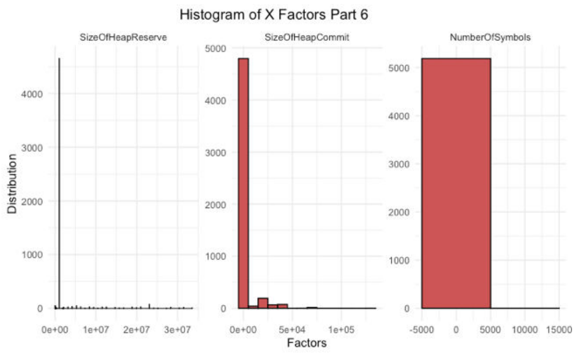




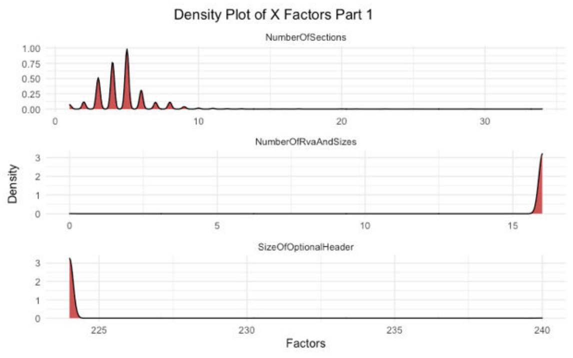


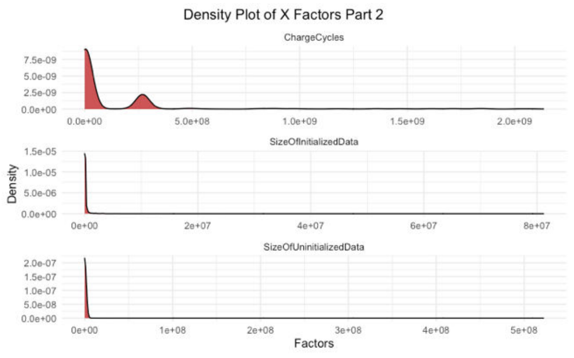


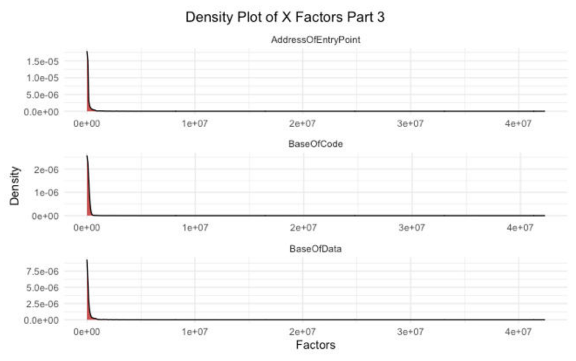


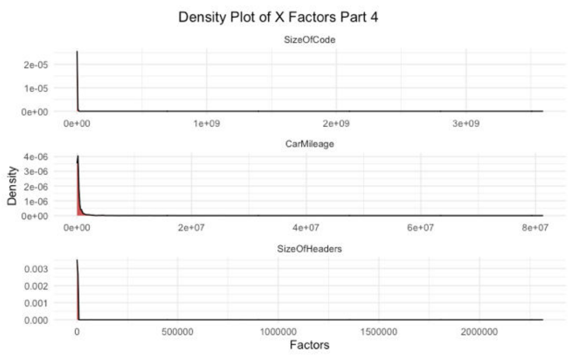


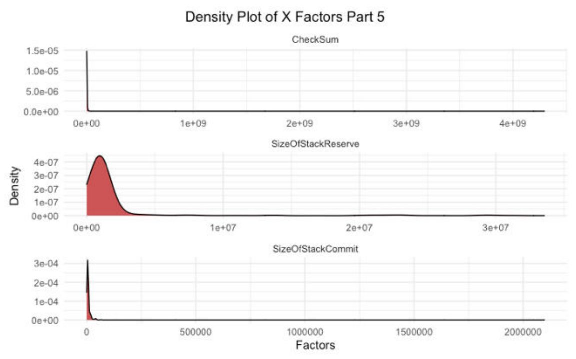
Univariate Kernel Density Analysis

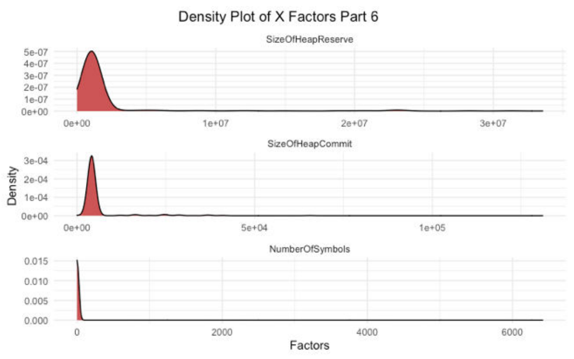




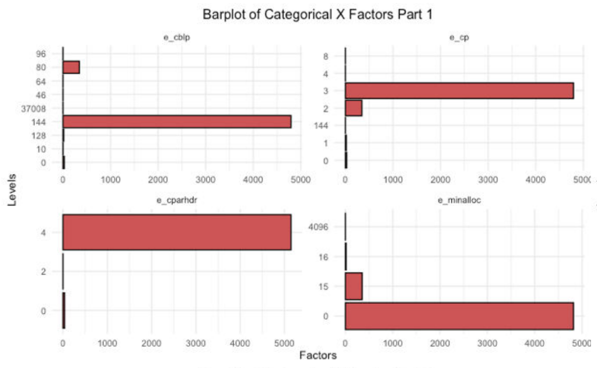

Density Plot of X Factors Part 3
AddressOfEntryPoint
BaseOfCode
BaseOfData
Density
Factors

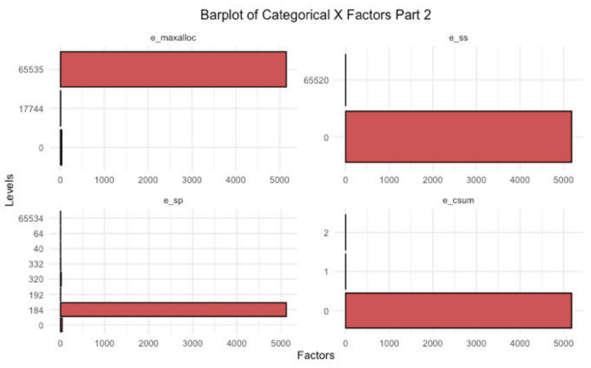

Density Plot of X Factors Part 4
SizeOfCode
CarMileage
SizeOfHeaders
Density
Factors

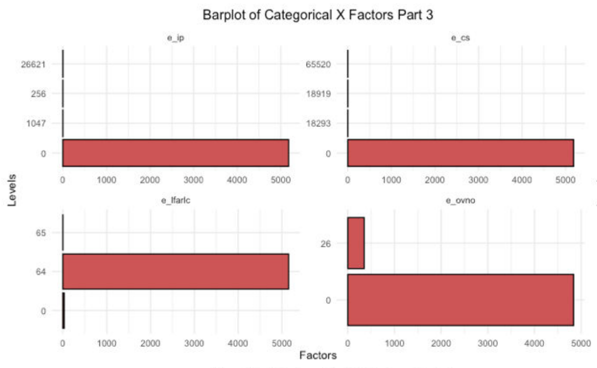

Density Plot of X Factors Part 5
CheckSum
SizeOfStackReserve
SizeOfStackCommit
Density
Factors

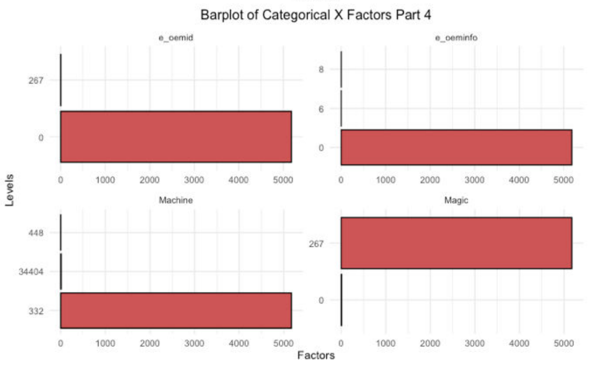

Density Plot of X Factors Part 6
SizeOfHeapReserve
SizeOfHeapCommit
NumberOfSymbols
Density
Factors

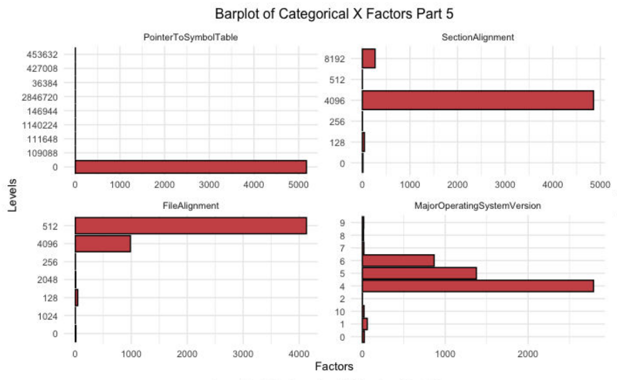
Univariate Barplot Analysis

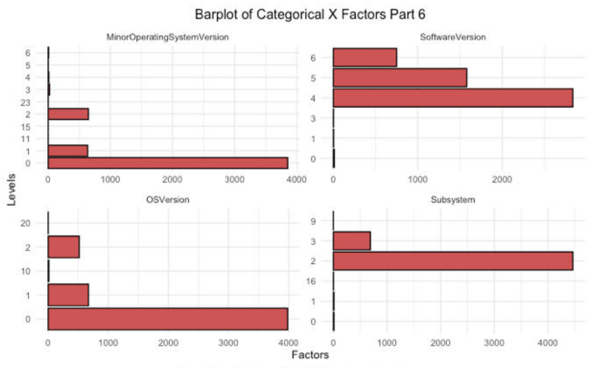


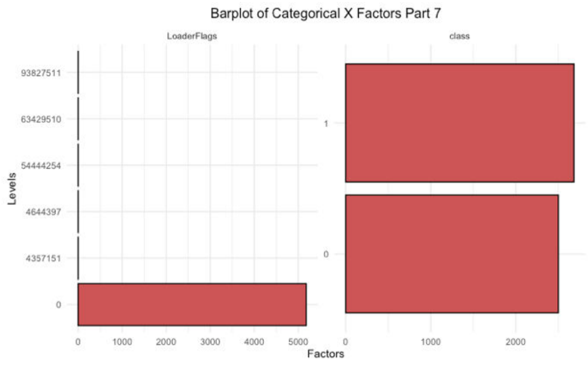


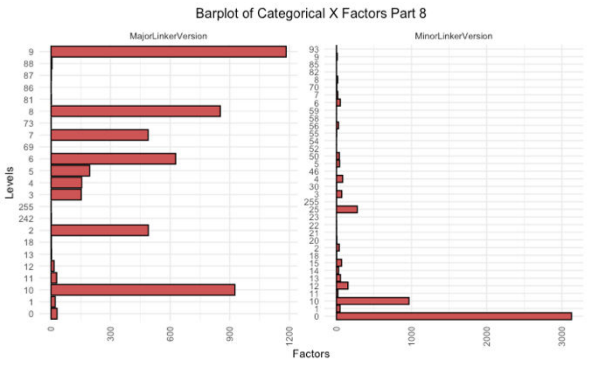






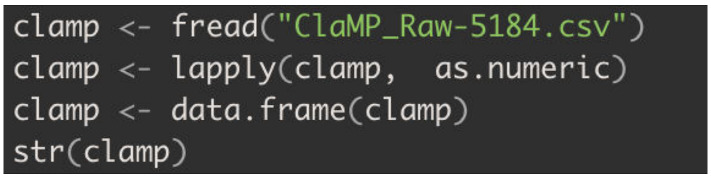






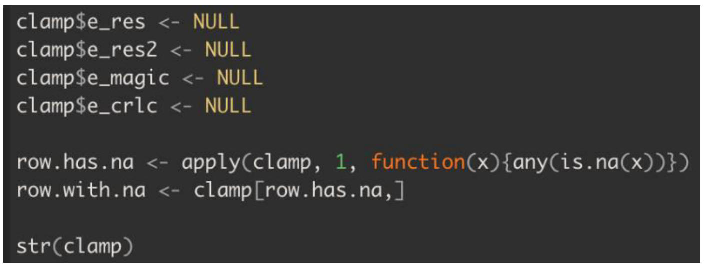
11.2 Data Preparation

Step 1: Reading in the dataset and converting all columns into numerical variables



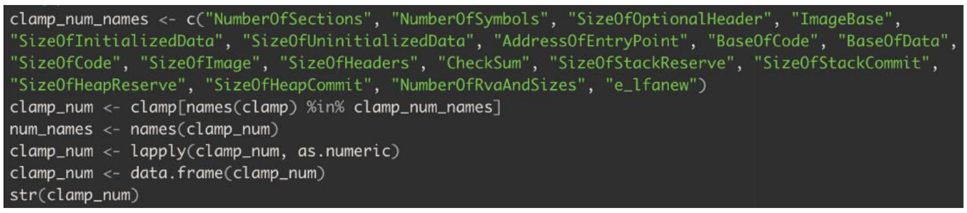
```
clamp <- fread("ClaMP_Raw-5184.csv")
clamp <- lapply(clamp,  as.numeric)
clamp <- data.frame(clamp)
str(clamp)
```

Step 2: Identify the columns with NA and remove them



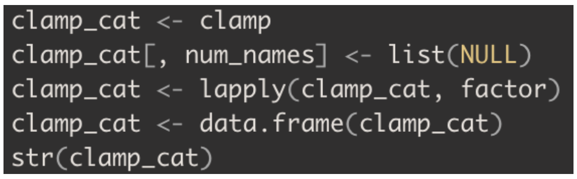
```
clamp$e_res <- NULL
clamp$e_res2 <- NULL
clamp$e_magic <- NULL
clamp$e_crlc <- NULL

row.has.na <- apply(clamp, 1, function(x){any(is.na(x))})
row.with.na <- clamp[row.has.na,]

str(clamp)
```

Step 3: Extracting the numerical variables



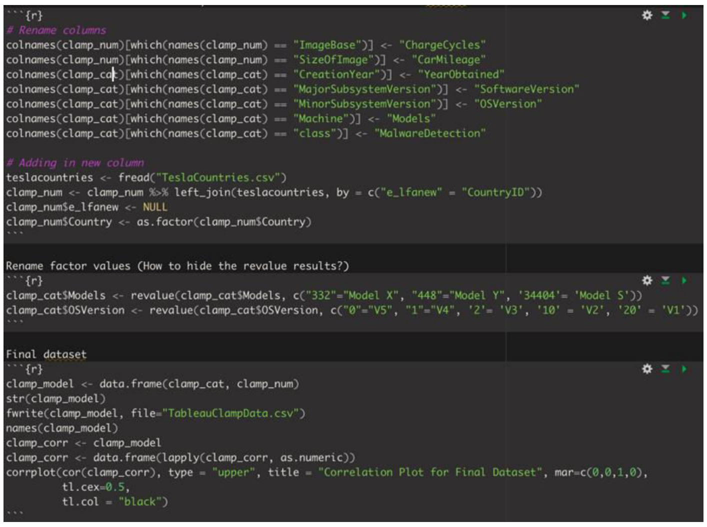
```
clamp_num_names <- c("NumberOfSections", "NumberOfSymbols", "SizeOfOptionalHeader", "ImageBase",
"SizeOfInitializedData", "SizeOfUninitializedData", "AddressOfEntryPoint", "BaseOfCode", "BaseOfData",
"SizeOfCode", "SizeOfImage", "SizeOfHeaders", "CheckSum", "SizeOfStackReserve", "SizeOfStackCommit",
"SizeOfHeapReserve", "SizeOfHeapCommit", "NumberOfRvaAndSizes", "e_lfanew")
clamp_num <- clamp[names(clamp) %in% clamp_num_names]
num_names <- names(clamp_num)
clamp_num <- lapply(clamp_num, as.numeric)
clamp_num <- data.frame(clamp_num)
str(clamp_num)
```

Step 4: Extracting the categorical variables and converting them into factors



```
clamp_cat <- clamp
clamp_cat[, num_names] <- list(NULL)
clamp_cat <- lapply(clamp_cat, factor)
clamp_cat <- data.frame(clamp_cat)
str(clamp_cat)
```

Step 5: Renaming of columns



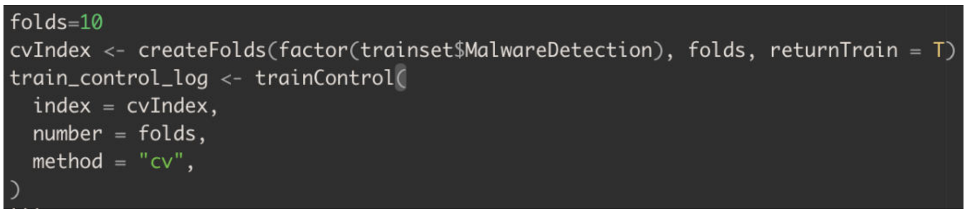
~~~
```{r}
# Rename columns
colnames(clamp_num)[which(names(clamp_num) == "ImageBase")] <- "ChargeCycles"
colnames(clamp_num)[which(names(clamp_num) == "SizeOfImage")] <- "CarMileage"
colnames(clamp_cat)[which(names(clamp_cat) == "CreationYear")] <- "YearObtained"
colnames(clamp_cat)[which(names(clamp_cat) == "MajorSubsystemVersion")] <- "SoftwareVersion"
colnames(clamp_cat)[which(names(clamp_cat) == "MinorSubsystemVersion")] <- "OSVersion"
colnames(clamp_cat)[which(names(clamp_cat) == "Machine")] <- "Models"
colnames(clamp_cat)[which(names(clamp_cat) == "class")] <- "MalwareDetection"

# Adding in new column
teslacountries <- fread("TeslaCountries.csv")
clamp_num <- clamp_num %>% left_join(teslacountries, by = c("e_lfanew" = "CountryID"))
clamp_num$e_lfanew <- NULL
clamp_num$Country <- as.factor(clamp_num$Country)
~~~

Rename factor values (How to hide the revalue results?)
```{r}
clamp_cat$Models <- revalue(clamp_cat$Models, c("332"="Model X", "448"="Model Y", '34404'= 'Model S'))
clamp_cat$OSVersion <- revalue(clamp_cat$OSVersion, c("0"="V5", "1"="V4", '2'= 'V3', '10' = 'V2', '20' = 'V1'))
```

Final dataset
```{r}
clamp_model <- data.frame(clamp_cat, clamp_num)
str(clamp_model)
fwrite(clamp_model, file="TableauClampData.csv")
names(clamp_model)
clamp_corr <- clamp_model
clamp_corr <- data.frame(lapply(clamp_corr, as.numeric))
corrplot(cor(clamp_corr), type = "upper", title = "Correlation Plot for Final Dataset", mar=c(0,0,1,0),
         tl.cex=0.5,
         tl.col = "black")
```
```
```

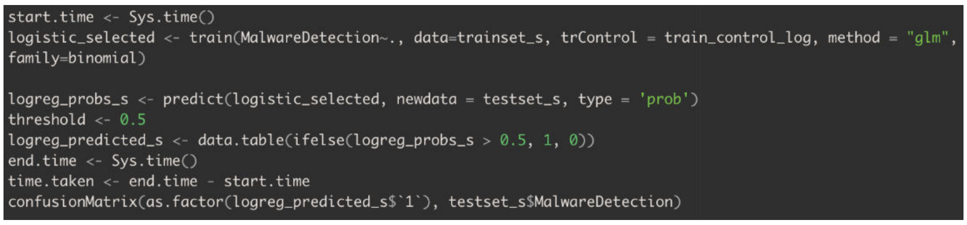
11.3 Logistic Regression

Step 1: K-Fold Cross Validation



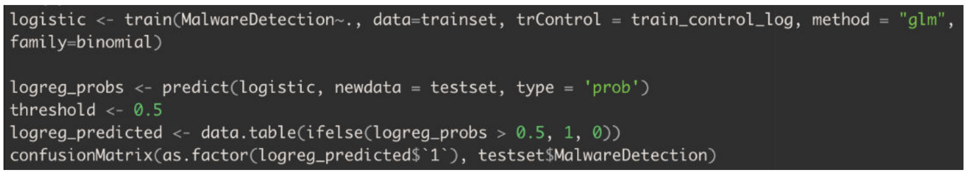
```r
folds=10
cvIndex <- createFolds(factor(trainset$MalwareDetection), folds, returnTrain = T)
train_control_log <- trainControl(
  index = cvIndex,
  number = folds,
  method = "cv",
)
```

Step 2.1: Perform Logistic Regression with Feature Selection



```r
start.time <- Sys.time()
logistic_selected <- train(MalwareDetection~., data=trainset_s, trControl = train_control_log, method = "glm",
family=binomial)

logreg_probs_s <- predict(logistic_selected, newdata = testset_s, type = 'prob')
threshold <- 0.5
logreg_predicted_s <- data.table(ifelse(logreg_probs_s > 0.5, 1, 0))
end.time <- Sys.time()
time.taken <- end.time - start.time
confusionMatrix(as.factor(logreg_predicted_s$`1`), testset_s$MalwareDetection)
```

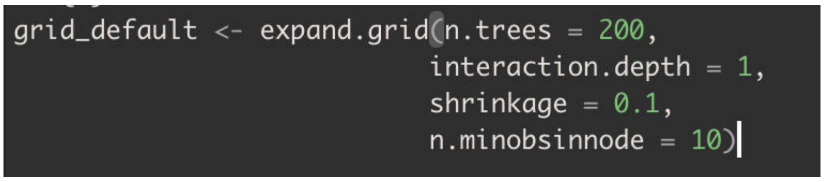
Step 2.2: Perform Logistic Regression with Feature Selection



```r
logistic <- train(MalwareDetection~., data=trainset, trControl = train_control_log, method = "glm",
family=binomial)

logreg_probs <- predict(logistic, newdata = testset, type = 'prob')
threshold <- 0.5
logreg_predicted <- data.table(ifelse(logreg_probs > 0.5, 1, 0))
confusionMatrix(as.factor(logreg_predicted$`1`), testset$MalwareDetection)
```

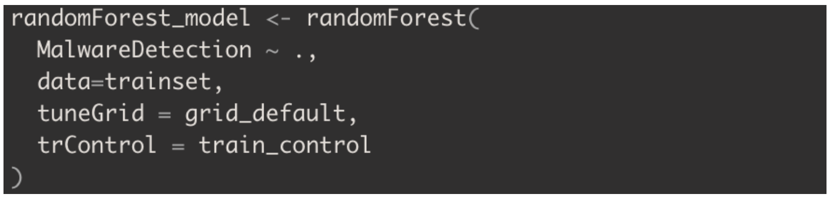
11.4 Random Forest

Step 1: Grid Search Algorithm



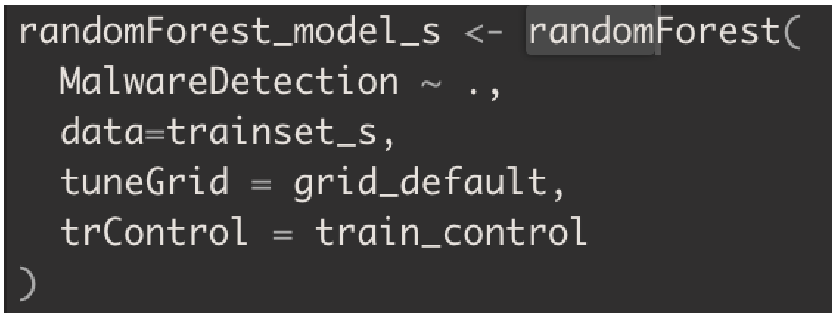
```r
grid_default <- expand.grid(n.trees = 200,
                            interaction.depth = 1,
                            shrinkage = 0.1,
                            n.minobsinnode = 10)
```

Step 2.1: Random Forest (without Feature Selection)



```r
randomForest_model <- randomForest(
  MalwareDetection ~ .,
  data=trainset,
  tuneGrid = grid_default,
  trControl = train_control
)
```

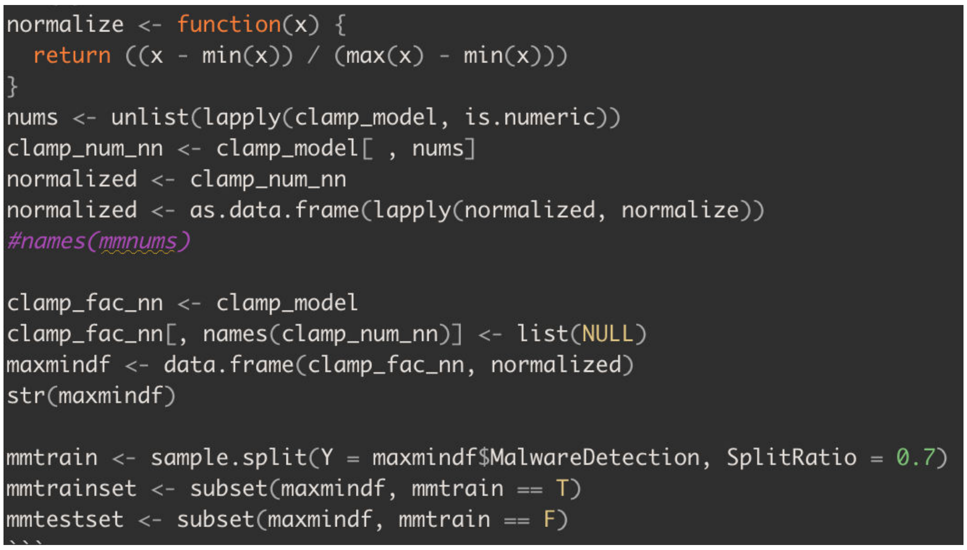
Step 2.2: Random Forest (with Feature Selection)



```r
randomForest_model_s <- randomForest(
  MalwareDetection ~ .,
  data=trainset_s,
  tuneGrid = grid_default,
  trControl = train_control
)
```

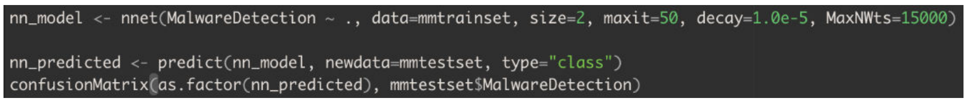
11.5 Neural Network

Step 1: Normalizing Dataset via Min-Max Normalization



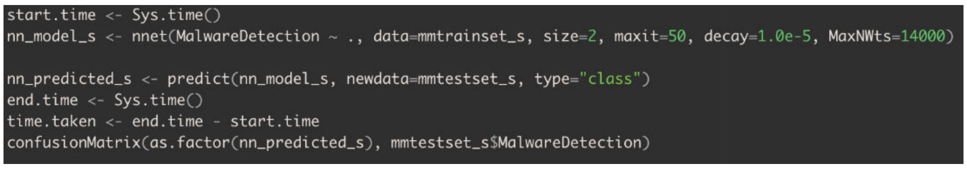
```
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x)))
}
nums <- unlist(lapply(clamp_model, is.numeric))
clamp_num_nn <- clamp_model[ , nums]
normalized <- clamp_num_nn
normalized <- as.data.frame(lapply(normalized, normalize))
#names(mmnums)

clamp_fac_nn <- clamp_model
clamp_fac_nn[, names(clamp_num_nn)] <- list(NULL)
maxmindf <- data.frame(clamp_fac_nn, normalized)
str(maxmindf)

mmtrain <- sample.split(Y = maxmindf$MalwareDetection, SplitRatio = 0.7)
mmtrainset <- subset(maxmindf, mmtrain == T)
mmtestset <- subset(maxmindf, mmtrain == F)
```

Step 2.1: Neural Network (without Feature Selection)



```
nn_model <- nnet(MalwareDetection ~ ., data=mmtrainset, size=2, maxit=50, decay=1.0e-5, MaxNWts=15000)

nn_predicted <- predict(nn_model, newdata=mmtestset, type="class")
confusionMatrix(as.factor(nn_predicted), mmtestset$MalwareDetection)
```

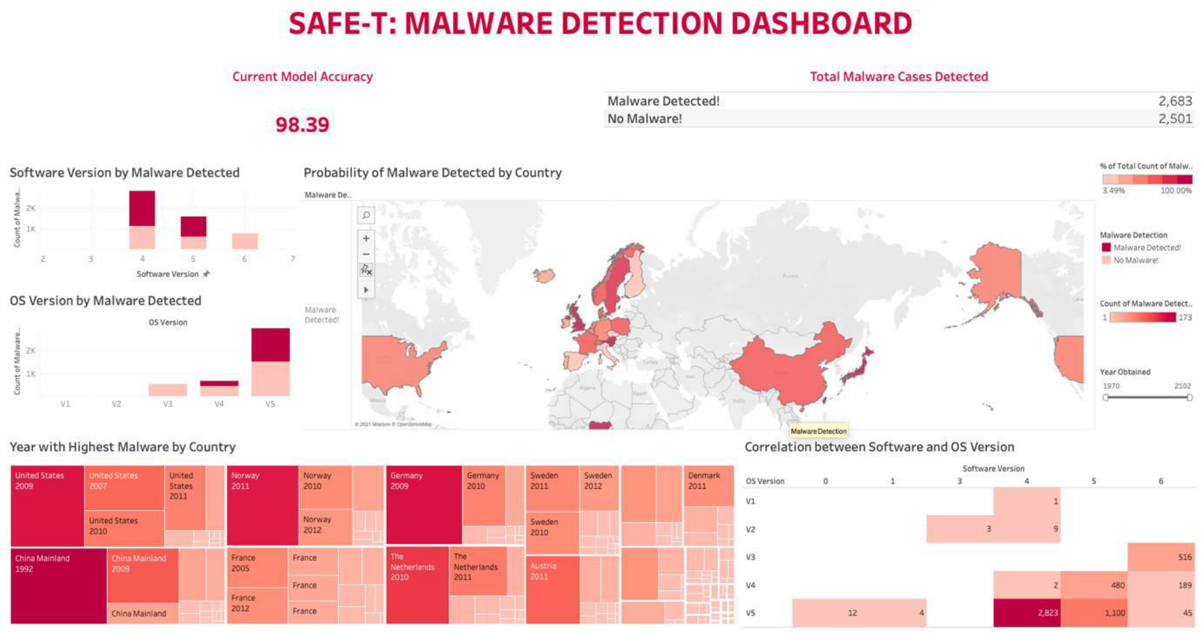
Step 2.2: Neural Network (with Feature Selection)



```
start.time <- Sys.time()
nn_model_s <- nnet(MalwareDetection ~ ., data=mmtrainset_s, size=2, maxit=50, decay=1.0e-5, MaxNWts=14000)

nn_predicted_s <- predict(nn_model_s, newdata=mmtestset_s, type="class")
end.time <- Sys.time()
time.taken <- end.time - start.time
confusionMatrix(as.factor(nn_predicted_s), mmtestset_s$MalwareDetection)
```

11.6 Interactive Dashboard

This interactive dashboard will provide software engineers and the client relations department the ability to detect rather than react to malware, improving operational efficiency while decreasing the time it takes for a customer to be informed of possible malware, improving its safety culture. This will be further explored during the presentation.



SAFE-T: MALWARE DETECTION DASHBOARD

Current Model Accuracy

98.39

Total Malware Cases Detected

Malware Detected!	2,683
No Malware!	2,501

Correlation between Software and OS Version

	Software Version					
OS Version	0	1	3	4	5	6
V1				1		
V2			3	9		
V3						516
V4				2	480	189
V5	12	4		2,823	1,100	45