Kent State University
Ambassador Crawford College of Business and Entrepreneurship
Spring 2024

Predictive Analysis for Accident Frequency and Insurance Risk Assessment

(BA-64060-001) - Group 10

Names	Contribution
SHARANYA DOMAKONDA	Developed Data Exploration, predictive models and assessed their performance on the train and test data set. Summarized project findings in the report and analyzed the hyperparameters for feature selection
JASWANTH BUDIGI	Implementation of Classification model and performed Accuracy analysis and Data Analysis, Project report, Power point presentation

Project Objective

The project intends to develop a predictive model that can depict the probabilities of a person getting into some levels of accident occurences, such as no accidents, very few accidents or many accidents. We will address this by using demographic data like driving experience, vehicle type, past accidents, and other variables in our analysis. The main aim of the classification is to classify people into two risk groups: high risk and low risk, considering technological advancement and background. The main goal of this analysis is to conduct insurance risk assessment with a view to developing specific intervention programs, and to promote safer driving.

Overview of Data

- Number of rows (Sample size) – 10,001 rows.
- How many columns (variables/features) – 19 columns
- The target variable is Outcome. The outcome column records whether the customer claimed insurance in the previous year or not.
- Categorical variables: 11
- Numerical variables: 8
- Missing values: 1939

The dataset contains information about various individuals.

- Id
- Age
- Gender
- Race
- Driving experience
- Education
- Income
- Credit score
- Vehicle ownership
- Vehicle year
- Married
- Children
- Postal code
- Annual mileage
- Vehicle type
- Speeding violations
- DUIs

- Past accidents
- Outcome

```
##        id             age              gender             race          
##  Min.   :   101   Length:10000       Length:10000       Length:10000  
##  1st Qu.:249639   Class :character   Class :character   Class :character  
##  Median :501777   Mode  :character   Mode  :character   Mode  :character  
##  Mean   :500522                                                          
##  3rd Qu.:753975                                                          
##  Max.   :999976                                                          
##                                                                          
##  driving_experience  education            income           credit_score    
##  Length:10000       Length:10000       Length:10000       Min.   :0.0534  
##  Class :character   Class :character   Class :character   1st Qu.:0.4172  
##  Mode  :character   Mode  :character   Mode  :character   Median :0.5250  
##                                                           Mean   :0.5158  
##                                                           3rd Qu.:0.6183  
##                                                           Max.   :0.9608  
##                                                           NA's   :982     
##  vehicle_ownership  vehicle_year         married            children        
##  Length:10000       Length:10000       Length:10000       Length:10000  
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                           
##                                                                           
##                                                                           
##                                                                           
##   postal_code    annual_mileage  vehicle_type       speeding_violations
##  Min.   :10238   Min.   : 2000   Length:10000       Min.   : 0.000     
##  1st Qu.:10238   1st Qu.:10000   Class :character   1st Qu.: 0.000     
##  Median :10238   Median :12000   Mode  :character   Median : 0.000     
##  Mean   :19865   Mean   :11697                      Mean   : 1.483     
##  3rd Qu.:32765   3rd Qu.:14000                      3rd Qu.: 2.000     
##  Max.   :92101   Max.   :22000                      Max.   :22.000     
##                  NA's   :957                                           
##       DUIs        past_accidents     outcome         
##  Min.   :0.0000   Min.   : 0.000   Length:10000       
##  1st Qu.:0.0000   1st Qu.: 0.000   Class :character   
##  Median :0.0000   Median : 0.000   Mode  :character   
##  Mean   :0.2392   Mean   : 1.056                      
##  3rd Qu.:0.0000   3rd Qu.: 2.000                      
##  Max.   :6.0000   Max.   :15.000                      
##
```

The dataset encompasses details regarding different individuals, covering a range of attributes such as age, gender, race, driving experience, education, income, credit score, vehicle ownership, marital status, children, postal code, annual mileage, vehicle type, speeding violations, DUIs, past accidents, and outcome. It is collected for analytical purposes, aiming to evaluate driving patterns, potential risk factors for accidents or insurance claims, and demographic profiles of the individuals involved. This dataset is versatile, suitable for diverse analytical endeavors like predictive modeling and risk assessment.

Descriptive Statistics:

The code performs a thorough exploration and descriptive analysis of the dataset, including steps such as data cleaning, transformation, imputation, and visualization. Here's a detailed review of each section:

1. **Data Cleaning and Transformation:**
    - To begin with, we used the subset () function to remove unnecessary columns from the dataset, including 'id', 'postal_code', 'age', and 'driving_experience'.
    - We, then used the summary () function to generate a summary of the cleaned dataset (Customer) with basic statistics and insights into remaining variables.

2. **Categorical Variable Encoding:**
    - Categorical variables in the dataset are converted into dummy variables using one-hot encoding. This step ensures that categorical variables are represented in a form that machine learning algorithms can understand. The dummy_cols() function from a package is used for this task.

3. **Imputation of Missing Values:**
    - The numerical variables 'credit_score' and 'annual_mileage' are imputed with the k-nearest neighbors (kNN) method. The kNN() function is used to replace missing values with estimates derived from adjacent data points.
4. **Selection of Numerical Variables:**
    - For additional analysis, the script chooses a subset of numerical variables from the dataset ('credit_score', 'annual_mileage', 'speeding_violations', 'DUIs', 'past_accidents'). It also includes the target variable 'outcome_True' in the selection.

5. **Descriptive Statistics:**

    - The summary () function calculates and summarizes basic statistics (e.g. mean, median, quartiles) for selected numerical variables. This provides information about the central tendency, spread, and distribution of the variables.
    - Histograms and boxplots are generated to show numerical variable distribution. Histograms show the frequency distribution of individual variables, whereas boxplots show a summary of the distribution, including outliers, quartiles, and the median.
    - The relationships between numerical variables are quantified using a correlation matrix. The cor () function is used to compute Pearson correlation coefficients, which measures the strength and direction of linear relationships between two variables.
    - To visualize the correlation matrix, we used the corrplot () function to generate a color-coded matrix with text labels for correlation coefficients. Hierarchical clustering is used to reorganize variables and highlight patterns in the correlation structure.

A summary of the dataset is given to highlight how the categorical target variable 'outcome' was transformed into numerical format for predictive modelling.

The data exploration and descriptive analysis provide a comprehensive understanding of the dataset, including structure, summary statistics, distributional characteristics, and variable relationships. This lays the groundwork for further analysis and modelling.

Histograms:

Histograms show the distributions of various driving-related variables.The credit score histogram displays a normal distribution, with most scores centered around 0.6, indicating that the sampled population is moderately reliable. The majority of people drive 10,000 to 15,000 miles per year. Both speeding tickets and DUIs are right-skewed, indicating that while the majority of people have few or no incidents, a small percentage engage in these risky behaviors consistently.. Previous accidents have followed a similar pattern, with the majority having none, and the frequency is rapidly decreasing as the number grows. Finally, the histogram labeled "Outcome_True" most likely represents a binary event, with the majority displaying '0' (False), indicating that the condition or event is uncommon among the participants. When the histogram for the outcome variable is converted to binary format, it shows that more people are classified as low risk (no accident) than high risk (accident). These visualizations can aid in understanding risk profiles and driving habits for applications such as insurance risk assessment.

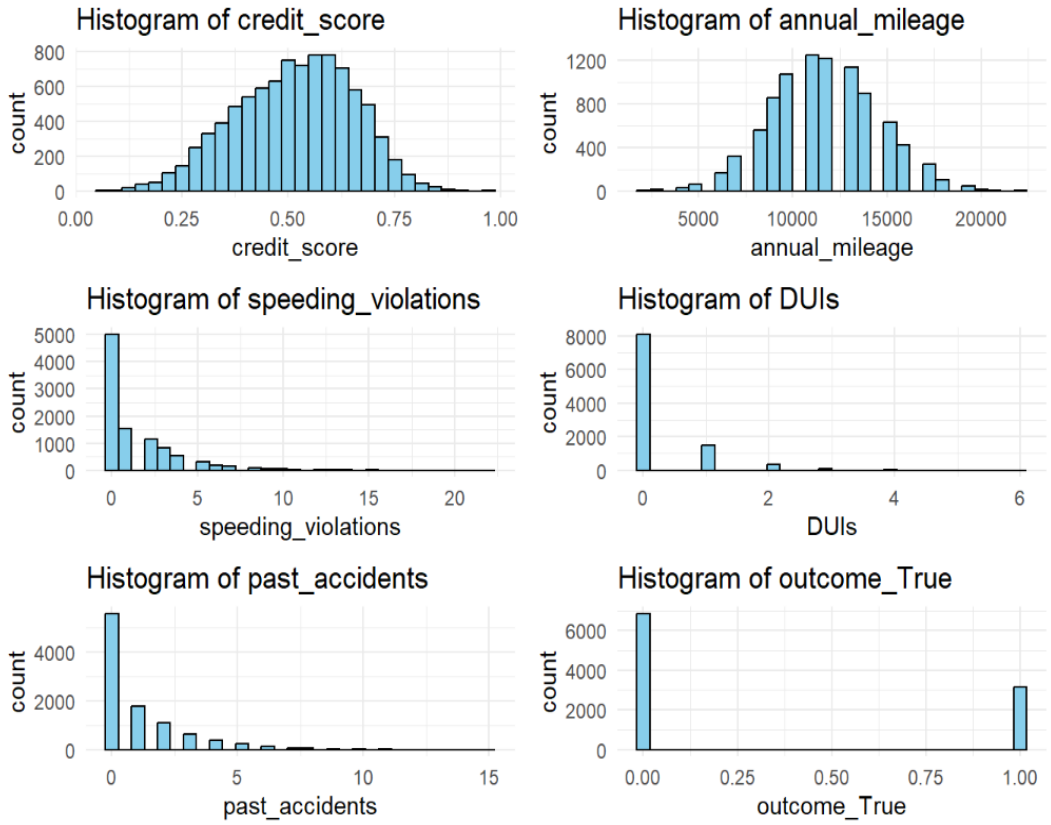


Fig 1.1

Box plots:

The boxplots show the distribution of various driving metrics. Credit scores are generally evenly distributed, with a median of 0.5-0.6 and no extreme outliers, indicating moderate creditworthiness throughout the population. Annual mileage averages around 12,000 miles, with outliers ranging from very low to very high, indicating that driving habits vary. Speeding violations and past accidents are concentrated at lower frequencies for most people, but there are outliers with significantly higher counts, indicating a small number of frequent violators or accident-prone drivers. DUIs are almost non-existent in the sample, with a few outliers totaling up to six incidents, indicating rare severe cases. The 'Outcome True' boxplot shows no variation, implying a uniformly negative outcome for the measured condition in the sample. These visualizations aid in understanding the central tendencies, spread, and presence of outliers in driving behaviors, all of which are required for risk assessment.

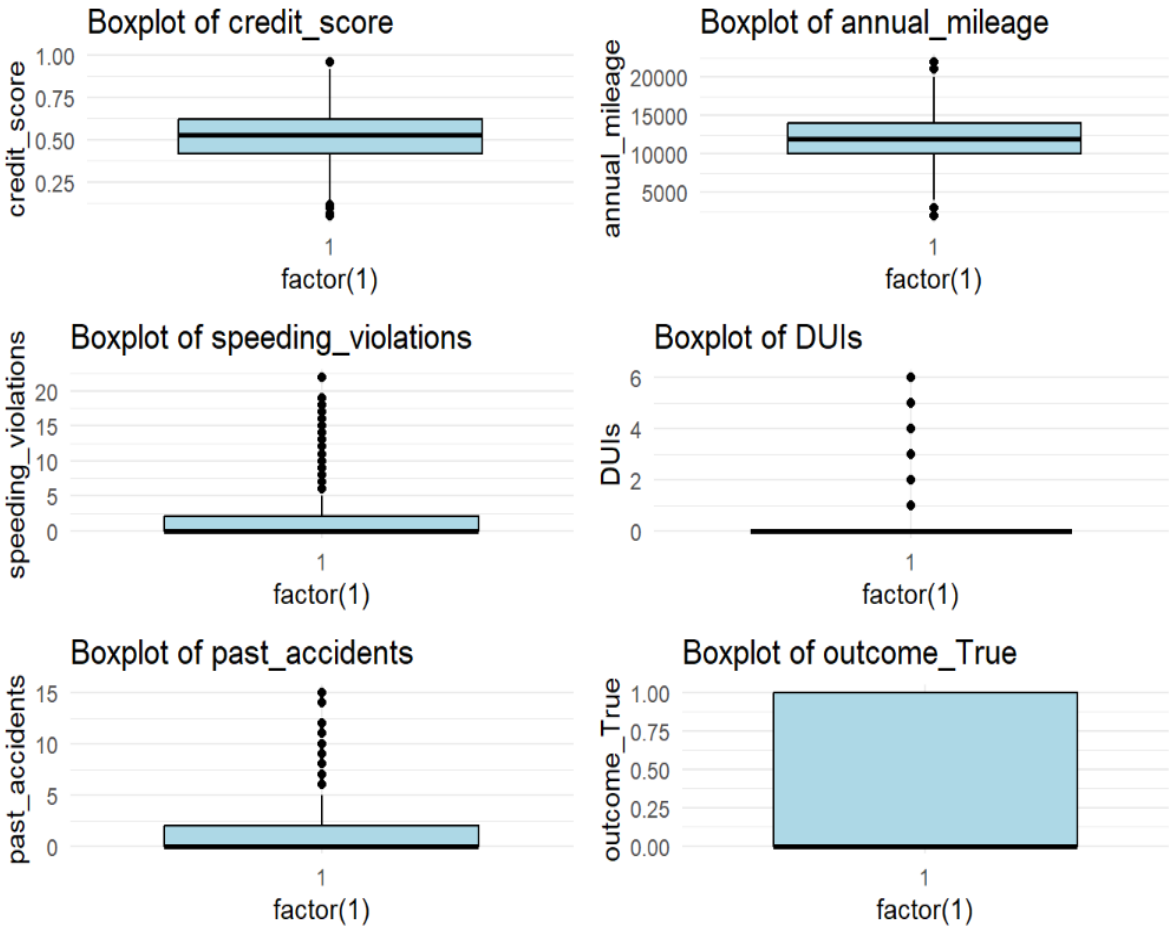


Fig 1.2

Correlation Matrix:

The correlation matrix reveals some intriguing relationships between driving-related variables. Annual mileage has a mildly positive correlation with the "Outcome_True" event, implying that driving more may increase the likelihood of this event. Credit score, on the other hand, has a moderately negative correlation with "Outcome_True," implying that higher credit scores may result in fewer occurrences of this event, possibly indicating more responsible behavior. Interestingly, there are slight positive correlations between credit score and negative driving behaviors such as DUIs, speeding violations, and past accidents, which is counterintuitive. DUIs and speeding violations both have positive correlations with one another and with past accidents, indicating that risky behaviors are likely to be associated. However, there are minor negative correlations between DUIs, speeding violations, and the "Outcome_True" event, which may indicate data irregularities or that these behaviors do not directly translate to the more serious outcomes covered by "Outcome_True." The strongest correlation observed is between speeding violations and past accidents, reinforcing the dangers of speeding. These findings imply complex interdependencies between driving behaviors, outcomes, and socioeconomic factors such as credit score.

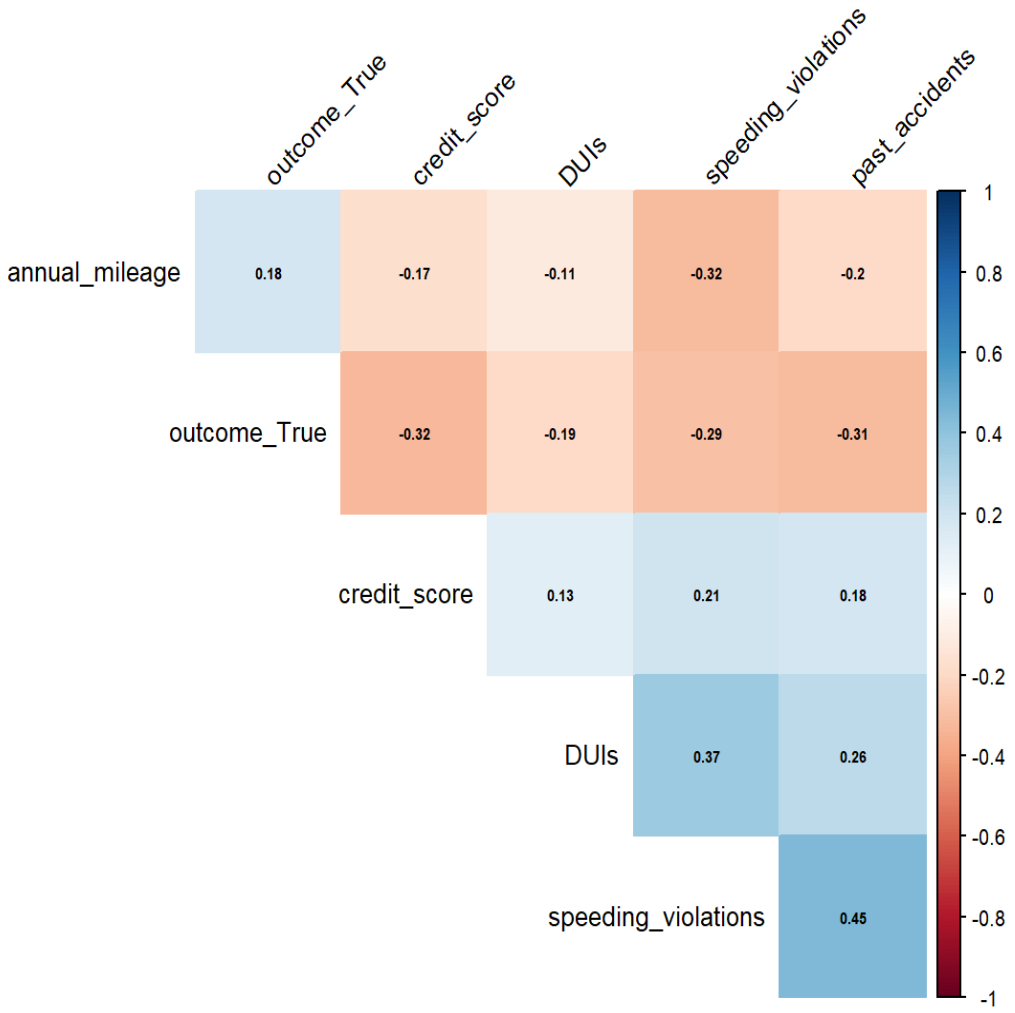


FIG 1.3

Feature Selection:

Methodology and Modeling strategy

During our data preprocessing phase, we used a variety of techniques to identify the most relevant features for our analysis. The process included the following steps:

- **Remove Constant or Zero Variance Features:** We started by making sure that our dataset, Customer_new, was ready for analysis by removing features with constant or zero variance. This step was critical for streamlining our dataset and focusing on variables that yielded useful insights.
- **Principal Component Analysis (PCA):** The task of determining the most significant features among the list of features responsible for the variability in our dataset was achieved by means of de-correlation technique known as PCA. Via this approach, we managed to make the space of features less dimensional by keeping all the important information that is included in the set of features.
- **Absolute Loadings and High Loading Threshold:** Absolute loadings from PCA were examined to identify variables that made significant contributions to the principal components. A 0.4 threshold was used to identify features with high loadings, indicating their importance in explaining the variance in the data.
- **Correlation Analysis:** To identify highly correlated variables, we created a correlation matrix using numerical features. Features with a correlation greater than the specified threshold (0.75) were deemed redundant and removed from further analysis.
- **Final Feature Selection:** The features chosen based on PCA loadings and correlation analysis were combined to create a set of variables deemed most relevant for our predictive modelling task. Credit score, annual mileage, speeding violations, DUIs, and past accidents were among these features.
- **Handling Missing Values:** To ensure data integrity and completeness, we addressed missing values in the numerical dataset (Customer_numerical) using the na.omit () function before proceeding with modelling.
- **PCA on Reduced Dataset**: Finally, we used PCA on the reduced dataset containing the selected features to capture the underlying structure and relationships between variables. Understanding the distribution of data points and how each variable contributed to the principal components was made possible by this step.

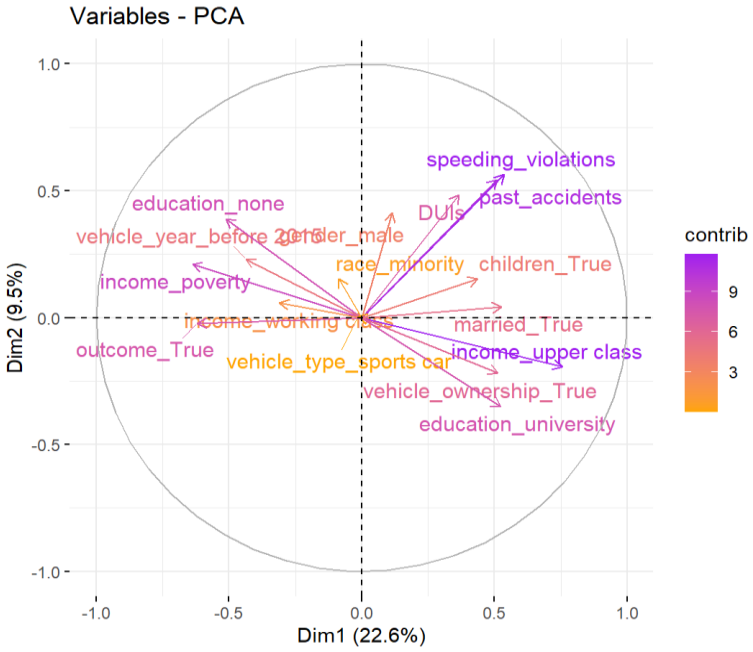
Through the application of these feature selection techniques, we hope to improve the predictive modeling process's quality by concentrating on the most informative variables and reducing noise and redundancy in the data.

Feature Selection process

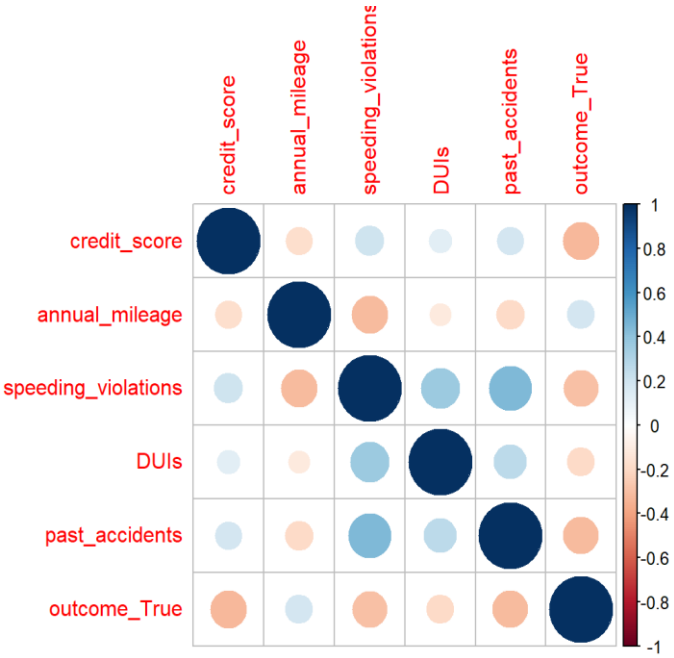
During the feature selection process, several steps are taken to identify and reduce the number of features in the dataset.

1. **Initial Data Cleaning:** Constant or zero variance features are removed from the dataset so that only informative variables remain. Checks are performed to ensure that the dataset has adequate variability and does not contain any missing values required for Principal Component Analysis (PCA).

2. **Principal component analysis (PCA):** PCA is applied to the cleaned dataset to identify underlying patterns and reduce the data's dimensionality. The PCA results are visualized using a variance plot, which depicts each feature's contribution to the principal components.



3. **Correlation Analysis:** A correlation matrix is generated to identify relationships between numerical features in the dataset. Multicollinearity issues are addressed by identifying and potentially removing features with high correlations.

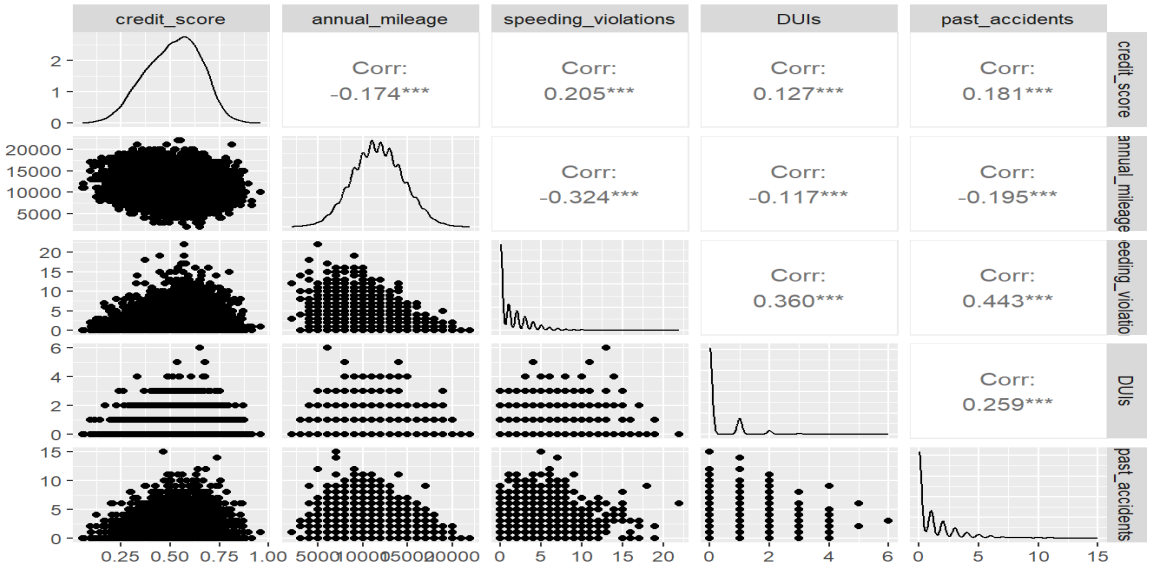


4. **Feature Selection based on Loadings**: Absolute loadings from PCA are computed to identify features that make significant contributions to the principal components.
A threshold is set to select features with loadings greater than a certain value, indicating their significance in explaining the data's variance.

```r
# Compute absolute loadings from PCA
# Correct access to the loadings (rotations)
loadings <- abs(pca_results$var$coord)

# Identify variables with high loadings (significant contribution to the principal components)
high_loading_threshold <- 0.4# Adjust this threshold as needed
high_loadings <- apply(loadings, 2, max) > high_loading_threshold # apply over columns
selected_features_by_loadings <- rownames(loadings)[which(high_loadings)]

# Compute correlation matrix of the numerical dataset
cor_matrix <- cor(Customer_numerical, use = "complete.obs")

# Find features with high correlation to potentially remove redundant variables
cor_threshold <- 0.75 # Adjust the correlation threshold as needed
features_to_remove_due_to_correlation <- findCorrelation(cor_matrix, cutoff = cor_threshold, names = TRUE)

# Set difference to exclude highly correlated features found
selected_features <- setdiff(selected_features_by_loadings, features_to_remove_due_to_correlation)

# Check and ensure selected features are in the dataframe
selected_features <- selected_features[selected_features %in% names(Customer_numerical)]

# Create a reduced dataset with selected features
if(length(selected_features) > 0) {
  Xnew <- Customer_numerical[, selected_features, drop = FALSE]

  # Plot the correlation matrix of the reduced dataset
  corrplot(cor(Xnew), method = "circle")

  # Plot the pair matrix using GGally package
  ggpairs(Xnew)
} else {
  print("No features were selected or they do not exist in the dataset.")
}
```

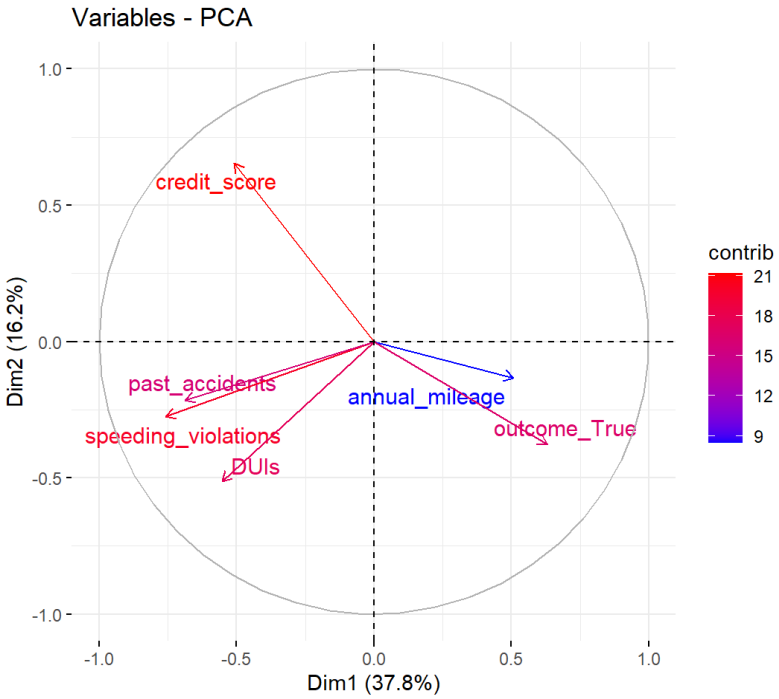
5. **Further feature selection:** To avoid dataset redundancy, features that are strongly correlated with others are identified and potentially removed. The final set of selected features is determined by subtracting the features chosen based on loadings from those eliminated due to high correlation.

6. **Visualization:** The reduced dataset's correlation matrix is plotted to show the relationships between the selected features. A pair matrix plot is created to depict the pairwise relationships between the selected features.



7. **Summary:** A summary of the selected features is provided, as well as a list of the variables that were included in the reduced dataset.

8. **PCA with Reduced Dataset:** Following feature selection, PCA is performed again on the reduced dataset to investigate the underlying patterns.



How/ Why do you choose specific features?

During the feature selection process, specific features were chosen based on their contribution to the dataset's variance and relationship with other variables. Here's a breakdown of how and why specific features were selected:

- **Initial Filtering:** Features with zero or constant variance were removed to ensure that they had no significant impact on the analysis.

- **Principle Component Analysis (PCA):** PCA was used to better understand the underlying structure of the data and to identify the variables that contribute the most to its variance. Variables with high loadings, indicating a significant contribution to principal components, were selected for further investigation.

- **Correlation analysis:** A correlation matrix was created to identify pairs of variables that had high correlation coefficients. Features with high correlations (above a predetermined threshold) were flagged for possible removal to reduce dataset redundancy.

- **Final selection:** Features were chosen based on both their individual contribution to variance (from PCA) and their relationship to other variables. Highly correlated features were removed to avoid multicollinearity issues in subsequent analyses.

- **Specific Features Selected:** The final dataset included the following features: credit_score, annual_mileage, speeding_violations, DUIs, and past_accidents. These features were chosen because they contribute significantly to the dataset's variance and are relevant to the project's objectives.

- **Final PCA Visualization:** PCA was performed on the reduced dataset, which included numerical and converted categorical variables. A correlation circle plot was used to visualize PCA results and show the contribution of variables to principal components. Overall, the feature selection process sought to retain the most informative variables while minimizing redundancy and multicollinearity, thereby improving the quality and interpretability of subsequent analyses.

Optimal Hyperparameters:

The optimal hyperparameters were determined through iterative refinement and experimentation. We fine-tuned thresholds for loadings and correlation coefficients using empirical testing, balancing feature relevance with model complexity. We also made sure that the PCA parameters, such as scaling and centering, were set correctly to allow for accurate representation of feature contributions. The threshold for high loadings and correlation was empirically determined and may need to be adjusted depending on the dataset and project requirements. For example, the thresholds for high loadings (high_loading_threshold) and correlation (cor_threshold) are predefined values that can be fine-tuned to achieve the desired level of feature selection stringency.

- **Threshold Tuning for Loadings and Correlation:** The thresholds for high loadings (high_loading_threshold) and correlation (cor_threshold) are initially fixed values. These values are determined using initial analysis and domain knowledge, but they are subject to change based on empirical testing. These thresholds are fine-tuned iteratively and experimentally to achieve a balance between feature relevance and model complexity. For example, in the code provided, a 0.4 threshold is set for high loadings, indicating that features with loadings greater than this value are significant contributors to principal components. Similarly, a correlation threshold of 0.75 is used to identify highly correlated features, which may indicate data redundancy.

- **Appropriate Parameter Settings for PCA:** In addition to threshold tuning, appropriate PCA parameter settings are critical for accurately representing feature contributions. Scaling and centering parameters are adjusted to ensure that the PCA analysis accurately represents the underlying structure of the data. Scaling ensures that all variables are on the same scale, preventing larger features from dominating the analysis. Centering the data means shifting it so that the mean of each variable is zero, ensuring that the principal components capture the relative relationships between variables. By adjusting these parameters, the PCA analysis can effectively capture the variance in the dataset and identify the most influential features.

- **Flexibility and Adjustability:** It is important to note that the thresholds and parameter settings are not fixed and can be changed to meet the specific needs of the dataset and project objectives. The thresholds for high loadings and correlation, as well as the PCA parameters, can be adjusted based on the desired level of feature selection stringency and data characteristics. These hyperparameters can be refined through empirical testing and validation to improve model performance and result interpretability.

Machine Learning:

K-Nearest Neighbors (KNN) Model

For classification, we also apply another simple yet powerful machine learning method called the K-Nearest Neighbors (KNN) algorithm.. Data quality was ensured by performing sufficient data cleaning before adding the data to the Customer numerical dataset. Additionally, the outcome variable was transformed into a factor variable that could be used with R regression algorithms.
To evaluate the model's performance effectively, we divide the dataset into training and testing subsets. Using a seed in the random number generator ensures that the data split is consistent. We're allocating 80% of the data to the training set and 20% to the testing set.

In the KNN model we use $k = 5$ nearest neighbors. This parameter is important because it specifies how many neighboring points influence a new point's classification. Choosing the appropriate k can significantly impact the model's accuracy, sensitivity, and specificity.

The model correctly predicted 944 true negatives and 258 true positives. There were 178 cases where class 0 was incorrectly classified as class 1 (false positives), and 249 cases where class 1 was incorrectly classified as class 0.

The model's overall accuracy is 73.79%. This metric indicates that the model's predictions are approximately three out of four times correct. However, the accuracy should be interpreted considering the no information rate (68.88%), which represents the accuracy that could be obtained by always predicting the most common class.

Additional metrics from the confusion matrix provide further insights:

The Kappa statistic (0.3642) indicates that there is moderate agreement between the predicted and actual classifications after accounting for chance.
Mcnemar's Test assesses the marginal homogeneity of two categorical variables. A significant result ($p = 0.0007052$) suggests a disagreement between the KNN-predicted labels and the actual labels.

The sensitivity or recall for class 0 is 84.14%, implying that the model is reasonably good at detecting true negatives. Class 0 has a lower specificity of 50.89%, indicating that it is less effective at detecting true positives. The positive predictive value (or precision) is 79.13%, and the negative predictive value is 59.17%, indicating the accuracy of positive and negative predictions.

In conclusion, while the KNN model has acceptable accuracy and sensitivity, its low specificity and predictive values indicate room for improvement. These metrics suggest ways to improve the model's performance, such as adjusting k values or exploring alternative modeling approaches or features.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 944 249
##          1 178 258
##
##                Accuracy : 0.7379
##                  95% CI : (0.7158, 0.7591)
##     No Information Rate : 0.6888
##     P-Value [Acc > NIR] : 7.865e-06
##
##                   Kappa : 0.3642
##
##  Mcnemar's Test P-Value : 0.0007052
##
##             Sensitivity : 0.8414
##             Specificity : 0.5089
##          Pos Pred Value : 0.7913
##          Neg Pred Value : 0.5917
##              Prevalence : 0.6888
##          Detection Rate : 0.5795
##    Detection Prevalence : 0.7324
##       Balanced Accuracy : 0.6751
##
##        'Positive' Class : 0
##
```

Naive Bayes Model

We used the Naive Bayes algorithm in R by calling its relevant function from the class e1071 package. Naive Bayes is a popular statistical classification and machine learning method (statistical learning) based on the Bayes' theorem of assumption that yields predictors are independent. This becomes especially suitable for large datasets, and applying it in practice can be very powerful, especially handled by the algorithm Naive Bayes can perform remarkably well.

For our analysis we used the trained Naive bayes model of this training dataset with the class attribute outcome_True as the target variable. Later the model was assessed on confusion matrix, which helps in visualizing the performance of the model against actual outcomes.

The Naive Bayes model results are summarized in a confusion matrix and it provides necessary details and clear images of the predictive capability of the model.

The model achieved an error-free recognition for 437 true positives (class 1) and 633 true negatives (class 0).

Nevertheless, 489 of them were initially suspected to be false negatives (class 0) while 70 cases were initially suspected to be false positives (class 1).

This model has an overall accuracy 65.68%, being the lowest measurement among all forms of intelligence. The no-information rate is 68.88%. The no-information rate stands for the accuracy of our opinion that this instant is unlikely to happen. The probability corresponding to an accuracy

greater than a base rate of 50.5% is 0.9973, which means that the model does not outperform a random prediction model.

Kappa value of 34.74% evaluates the agreement of prediction with the truth, adjusted for the agreement that could happen by chance. A Kappa value closer to 0 suggests less agreement beyond chance, whereas a value closer to 1 indicates strong agreement.
Mcnemar's Test indicates whether there is a significant difference between the classifier's performance on two different classes. A p-value < 2e-16 suggests a significant difference.

Additional metrics from the confusion matrix provide further insights:

The sensitivity or recall for class 0 is 56.42%, which means that the model correctly identifies slightly more than half of the actual negatives.
The specificity for class 0 is significantly higher, at 86.19%, indicating that the model is very good at detecting true positives.
The positive predictive value (precision) for class 0 is 90.04%, implying that the model's prediction of class 0 is very likely correct.
In contrast, the negative predictive value is lower, at 47.19%, indicating that nearly half of the class 1 predictions are incorrect.
The balanced accuracy, which is the average of sensitivity and specificity, is 71.31%, indicating moderate effectiveness in balancing recognition of both classes.

Though in theory the Naive Bayes model has the highest predictive accuracy and specificity, these indices can be further improved. This, however, provokes the question of achieving the balance between sensitivity and specificity, with some false positives being incorporated into the model under the assumption of the random nature of features and disregarding their interdependence that may be applicable to reality. Thus, it gives us ground to try new classifier modifications or to make changes in this modelling method to enhance it.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 633  70
##          1 489 437
##
##                Accuracy : 0.6568
##                  95% CI : (0.6332, 0.6799)
##     No Information Rate : 0.6888
##     P-Value [Acc > NIR] : 0.9973
##
##                   Kappa : 0.3474
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.5642
##             Specificity : 0.8619
##          Pos Pred Value : 0.9004
##          Neg Pred Value : 0.4719
##              Prevalence : 0.6888
##          Detection Rate : 0.3886
##    Detection Prevalence : 0.4316
##       Balanced Accuracy : 0.7131
##
##        'Positive' Class : 0
##
```

Insights and Conclusion:

Insights:

1. **Model Performance Evaluation:** The project examined the effectiveness of two classical machine learning algorithms, KNN and Naive Bayes, for binary classification of customers by a set of predefined attributes. Confusion matrices were employed to assess model performance of which included accuracy, sensitivity, specificity, and positive predictive value.

2. **Comparative Analysis:** The KNN model demonstrated superior overall accuracy compared to Naive Bayes, achieving approximately 73.79% accuracy versus 65.68%. This suggests that KNN is more effective in handling the balance between false positives and false negatives for the given dataset.

3. **Strengths and Weaknesses:** KNN showed higher sensitivity, indicating better performance in identifying true negatives, but at the expense of lower specificity. On the other hand, Naive Bayes demonstrated high specificity and positive predictive value but struggled with sensitivity, indicating a miss in identifying positive outcomes.

4. **Consideration of Consequences:** The specific requirements and consequences of false positives and false negatives in the context of the application should influence the model selection. KNN may be preferred in scenarios where failure to detect a negative outcome has serious consequences, whereas Naive Bayes may be appropriate in situations where the cost of false positives is higher.

5. **Room for Improvement:** Both models could be improved for better predictability. This could include adjusting model parameters, incorporating new features, or investigating alternative modeling techniques that better address the sensitivity-specificity balance.

Conclusion:

The project's conclusions highlight how crucial it is to select the appropriate machine learning model depending on the needs of the application and the effects of false positives and false negatives. Naive Bayes performed better in specificity and positive predictive value than KNN, which had higher overall accuracy and sensitivity.

In practice, the decision between KNN and Naive Bayes should be based on the nature of the problem and the risks of misclassification. Additional research could focus on improving predictive performance by refining model parameters, investigating new features, or experimenting with different modeling techniques.

Overall, the project provides useful insights into the comparative performance of KNN and Naive Bayes in a binary classification task, laying the groundwork for future improvements and advances in predictive modeling for similar applications.

References:

Here are some reference links:

- **Basic Statistics**
  - http://www.sthda.com/english/wiki/scatter-plot-matrices-r-base-graphs
  - https://seaborn.pydata.org/tutorial/categorical.html
  - https://cran.r-project.org/web/packages/corrplot/vignettes/corrplot-intro.html
- **Principal Component Analysis (PCA)**
  - https://www.keboola.com/blog/pca-machine-learning
  - https://www.r-bloggers.com/2021/05/principal-component-analysis-pca-in-r/
- **Machine Learning Performance Evaluation**
  - https://developers.google.com/machine-learning/testing-debugging/metrics/metrics
  - https://machinelearningmastery.com/
- **K-Nearest Neighbors (KNN) Model**
  - https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/
  - https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/]
- **Naive Bayes Model**
  - https://machinelearningmastery.com/
  - https://www.analyticsvidhya.com/blog/2021/09/naive-bayes-algorithm-a-complete-guide-for-data-science-enthusiasts/
- **Machine Learning Performance Evaluation**
  - https://developers.google.com/machine-learning/testing-debugging/metrics/metrics
  - https://machinelearningmastery.com/