A logo for college computing

Description automatically generated**Assessment Cover Page**

**Declaration**

By submitting this assessment, I confirm that I have read the CCT policy on academic misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source.

I declare it to be my own work and that all material from third parties has been appropriately referenced.

I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.

|  |  |
| --- | --- |
| *Student Full Name: Caroline Floss Bresolin* |  |
| *Student Number: 2023190* |  |
| *Module Title: Strategic Thinking* |  |
| *Assessment Title: CA3 Project* |  |
| *Assessment Due Date: 11/05/2024* |  |
| *Date of Submission: 11/05/2024*  **Use of AI Tools**  I have not used any AI tools or technologies in the preparation of this assessment. |  |

[Introduction 3](#_Toc166318015)

[Business description 4](#_Toc166318016)

[Success criteria/indicators 5](#_Toc166318017)

[Technologies used 5](#_Toc166318018)

[Business Understanding 5](#_Toc166318019)

[Data Understanding 6](#_Toc166318020)

[Heatmap 17](#_Toc166318021)

[Outliers 19](#_Toc166318022)

[Target distribution 20](#_Toc166318023)

[Data Preparation 21](#_Toc166318024)

[Null values 21](#_Toc166318025)

[Date type 22](#_Toc166318026)

[Columns to drop 23](#_Toc166318027)

[Transforming categorical into numerical 23](#_Toc166318028)

[Deciding the model 25](#_Toc166318029)

[Metrics 26](#_Toc166318030)

[Cross Validation 28](#_Toc166318031)

[SMOTE 29](#_Toc166318032)

[PCA (Principal component analysis) 30](#_Toc166318033)

[Standardization of the data 31](#_Toc166318034)

[Modelling 32](#_Toc166318035)

[LogisticRegression 32](#_Toc166318036)

[Hyperparameters tuned 33](#_Toc166318037)

[Metrics 33](#_Toc166318038)

[Evaluation results using PyCaret 34](#_Toc166318039)

[LIME 38](#_Toc166318040)

[SHAP 39](#_Toc166318041)

[Linear Discriminant Analysis (LDA) 43](#_Toc166318042)

[Hyperparameters tuned 43](#_Toc166318043)

[Metrics 44](#_Toc166318044)

[Evaluation results using PyCaret 45](#_Toc166318045)

[SHAP (LDA) 47](#_Toc166318046)

[SVC (Support Vector Classification) 51](#_Toc166318047)

[Hyperparameters tuning 51](#_Toc166318048)

[Metrics 52](#_Toc166318049)

[LIME SVC 53](#_Toc166318050)

[Decision Three (CART) 54](#_Toc166318051)

[Hyperparameters Tuning: 54](#_Toc166318052)

[Metrics 56](#_Toc166318053)

[Evaluating results using PyCaret 56](#_Toc166318054)

[SHAP CART 58](#_Toc166318055)

[Evaluation 60](#_Toc166318056)

[Deployment 61](#_Toc166318057)

[Conclusion 61](#_Toc166318058)

[Reference List 62](#_Toc166318059)

# Introduction

Auto insurance companies lose billions of dollars per year due to insurance fraud. Applicants, policyholders, mechanics and accident victims all participate in car insurance fraud. Car insurance companies **lose $29 billion per year** because of this, according to a 2017 study by Verisk. Lost premiums affect car insurance customers, too. Up to 14% of what you pay for car insurance premiums goes toward covering your insurance company's lost premiums (valuepenguin, 2024).

The insurance industry faces significant challenges in detecting and preventing fraudulent claims, which can lead to substantial financial losses. In response to this challenge, my study aims to leverage machine learning techniques to develop predictive models capable of identifying potential instances of insurance fraud. This study leverages machine learning models and interpretability tools like LIME and SHAP to identify potential instances of insurance fraud.

# Business description

The dataset at hand contains data from 01/01/2015 to 01/03/2015 within three states in the US: Ohio, Indiana, and Illinois. It includes details from insured individuals, such as their age, duration as customers, hobbies, and occupation, as well as information regarding law enforcement and claims. The claim-related data includes the amount, location of occurrence, whether authorities were notified, and the severity of the accident. In the classification framework established for this case study, the response (or target) variable is labelled "fraud\_reported". This column contains "Y" values to indicate fraudulent cases and "N" values for non-fraudulent cases.

The primary objective of this study is to develop predictive models that can accurately classify insurance claims as fraudulent or legitimate. Automating the fraud detection process aims to minimize financial losses associated with fraudulent claims while enhancing operational efficiency and customer satisfaction.

The detection of insurance fraud poses several challenges, including:

* Imbalanced Data: The distribution of fraudulent claims within the dataset is often highly skewed, with a relatively small proportion of claims being fraudulent.
* Complex Patterns: Fraudulent activities may exhibit intricate patterns and behaviours that are not easily discernible using traditional rule-based approaches.
* Adaptive Fraud Tactics: Fraudsters continuously adapt their tactics to evade detection, making it challenging to develop static fraud detection rules.

## Success criteria/indicators

Despite encountering several challenges, the use of SMOTE to handle the imbalanced dataset along with techniques for hyperparameter tuning enabled four models to achieve satisfactory outcomes, with metrics exceeding 80%. This suggests potential cost savings of around 50M.

## Technologies used

CRISP-DM is employed as a methodology to guide the process. Exploratory Data Analyses (EDA) is a crucial step in getting a better understanding of the dataset which was employed using libraries such as Seaborn, Pyplot and Matplotlib. Data Preparation is essential before employing any machine learning model. The SMOTE technique was used to resample the data in order to address the imbalanced target. This study is focused on three machine learning models that were the ones which achieved the highest metrics, Logistic Regression, Linear Discriminant Analyses and Support Vetorial Categorical. Additionally, libraries such as scikit-learn, pandas, and NumPy were utilized for data manipulation and model building.

# Business Understanding

There is an estimate that between 10% and 20% of insurance claims are fraudulent. American families pay an additional $400 to $700 per year in insurance premiums to help cover the cost of insurance fraud, according to the FBI (valuepenguin, 2024).

Fraudulent bodily injury claims cost car insurance companies between $6.8 and $9.3 billion per year. In some cases, drivers involved in an accident commit fraud by inflating the severity of an otherwise legitimate claim. Known as "buildup," this type of fraud takes place in 21% of bodily injury claims, which pay for injuries to the other driver or passenger in an accident you cause, and 18% of personal injury protection claims, which pay for your medical bills after an accident, regardless of fault. Some of the most common types of auto insurance fraud are (valuepenguin, 2024):

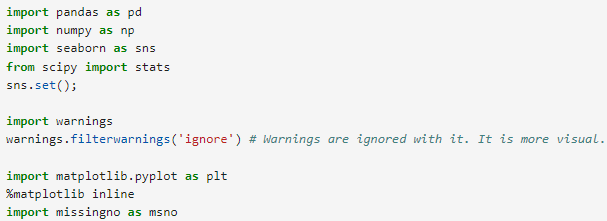
* Lying on your insurance application to get a lower rate
* Faking or exaggerating an injury
* Filing multiple claims for the same accident
* Filing a claim for an accident that never happened
* Causing an accident on purpose to file a claim with another driver's insurance
* Filing a claim for a stolen vehicle that doesn't exist or is still in your possession
* Inflating the cost or extent of repairs needed to get more money from the insurance company

21% of insurance plans to invest in AI in the next two years (InsuranceFraud.org, n.d.).

# Data Understanding

The data understanding phase of CRISP-DM involves taking a closer look at the data available for mining. This step is critical in avoiding unexpected problems during the next phase--data preparation--which is typically the longest part of a project.

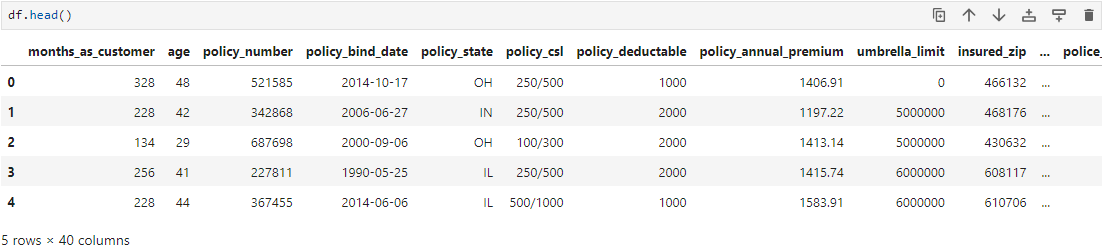
As the first step, essential libraries are imported:



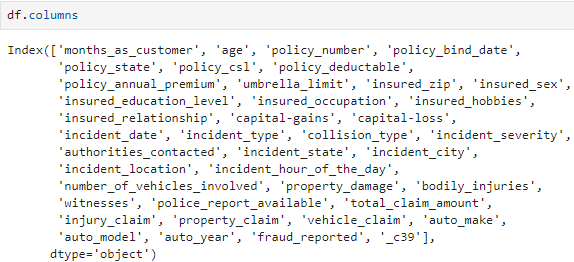
The dataset was sourced by Kaggle (www.kaggle.com, n.d.) and is in CSV format. While reading it, it assigns the possible null values to the proper format.



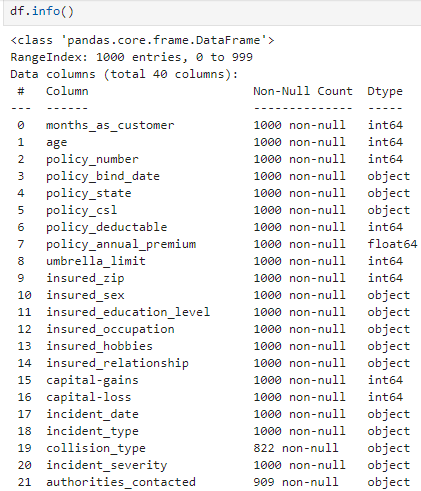
The “.head” function is used to have a brief view of the dataset. With it, we can notice that the dataset has 40 columns:



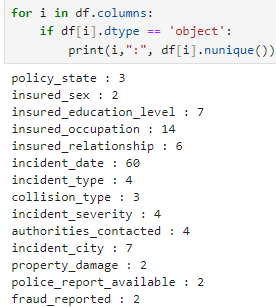
As all the columns don’t fit on the screen, running the “.columns” function displays the names of all of them:



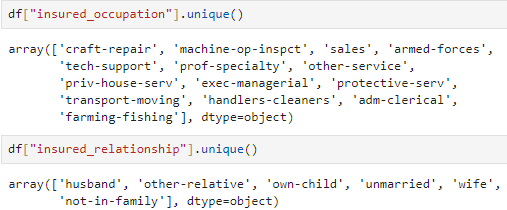
The .info() function provides comprehensive information about a DataFrame, including the data types of variables in each column, the number of non-null values, and the dimensions of the DataFrame in terms of rows and columns:



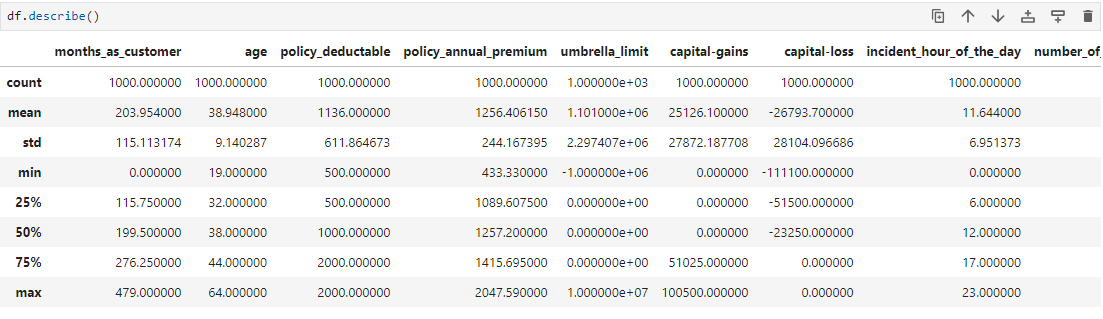
The code below provides how many unique variables are in each categorical feature:



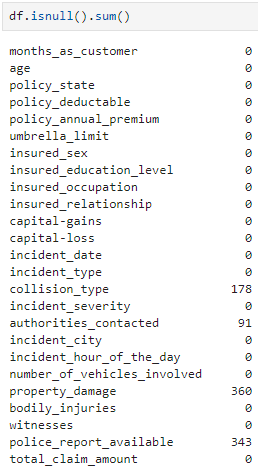
In addition, to display the names of those variables we can use the “.unique” followed by the column’s name. It is important to understand each feature.



The describe function offers a statistical summary of the dataset, providing key metrics such as mean, standard deviation, minimum, maximum, and quartile values.

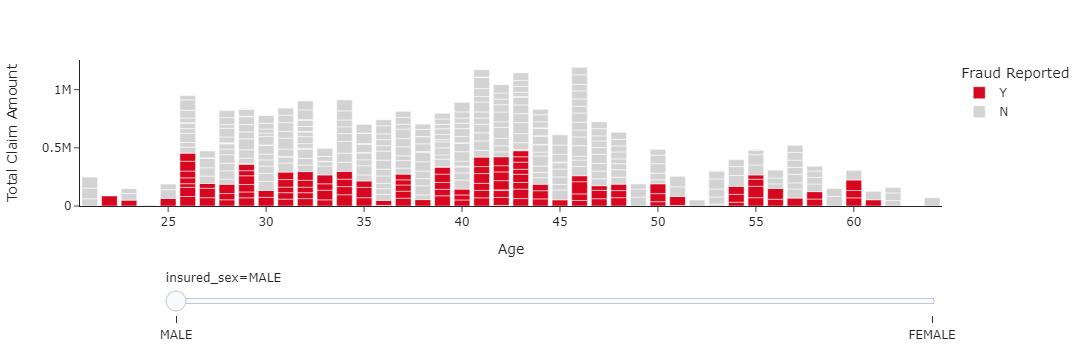


The “.isnull” function proves to be invaluable as it provides a comprehensive overview of null values in the dataset:

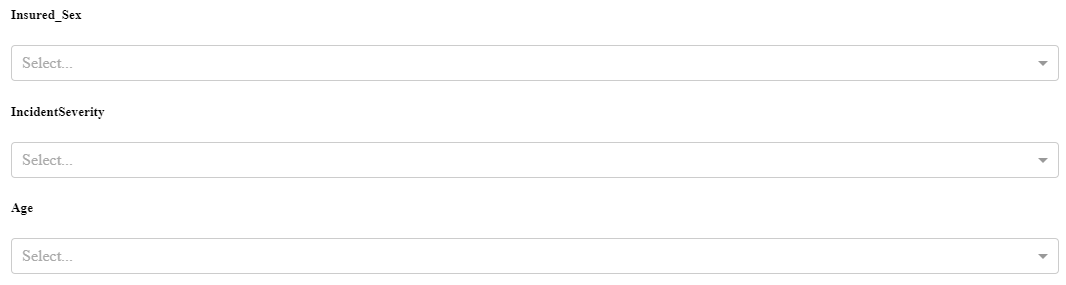


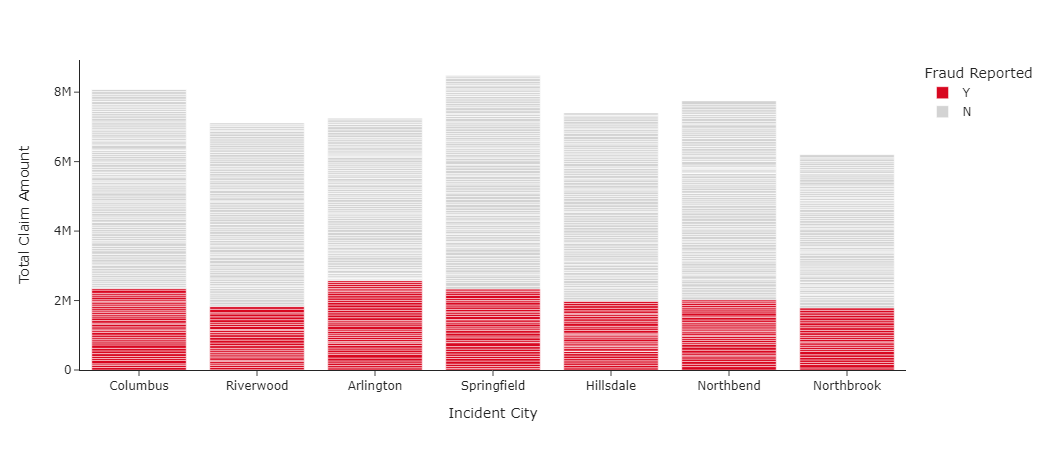
Pyplot is a plotting library for creating static, animated, and interactive visualizations in Python.

The visualization below enables the user to analyse the total claim amount by age and sex and select whether it was reported as fraud or not. We can notice that at some age there is a higher probability of being reported as fraudulent, for instance, those males in their 20s.



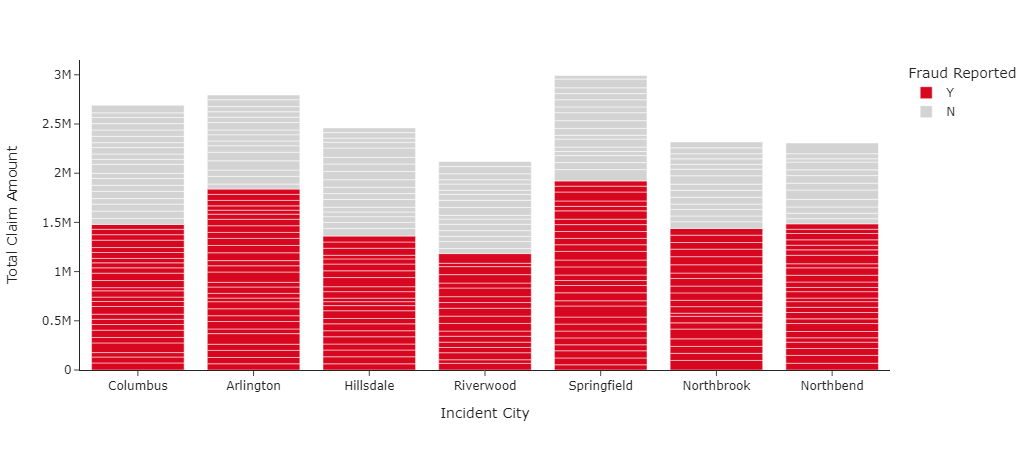
Utilizing a dashboard that showcases the claimed amount by 'incident city', users can delve deeper into the analysis based on 'insured sex', 'incident severity', and 'age'.



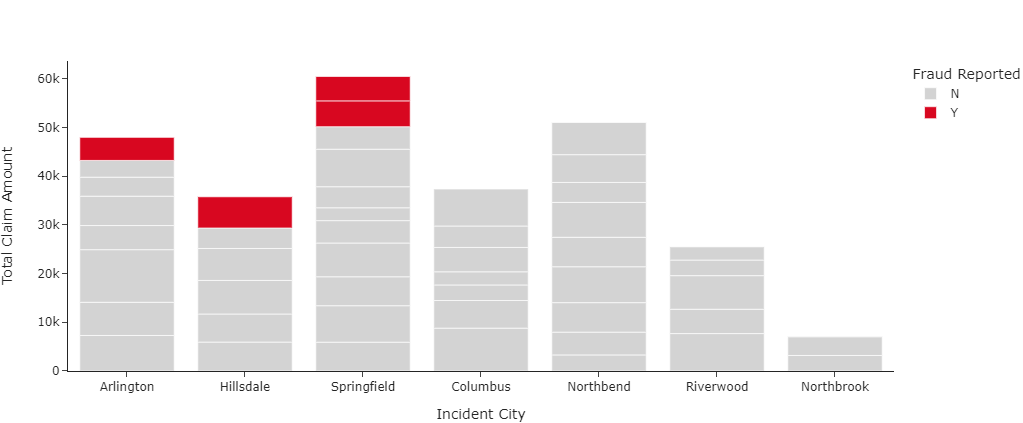


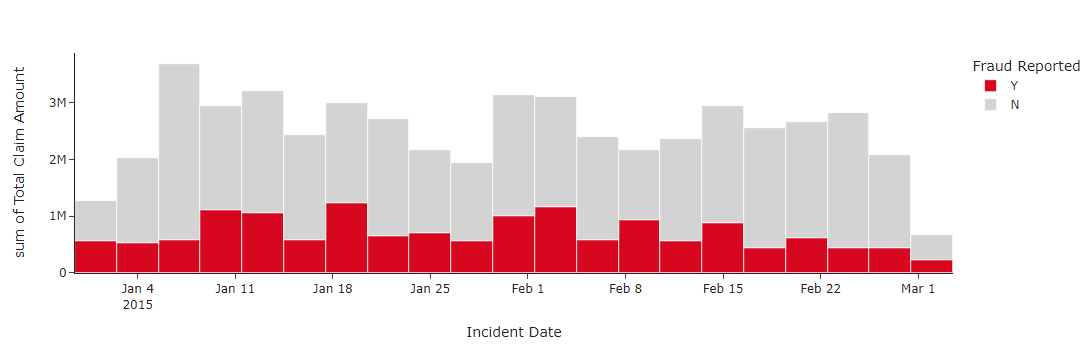
For example, when filtering only for 'Major Damage', it becomes apparent that in certain cities, there is a higher probability of being reported as fraudulent:



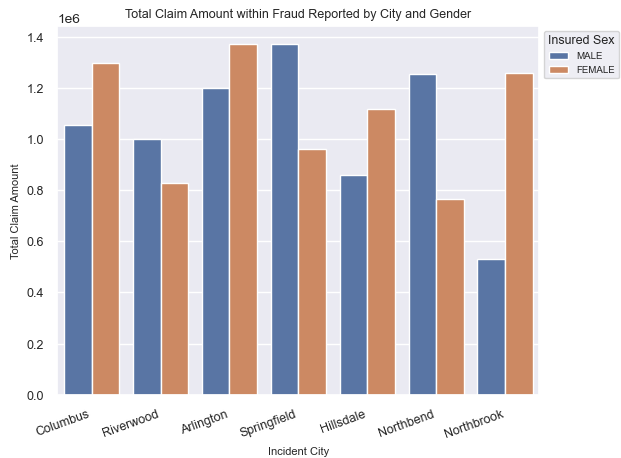


While in ‘Trivial Damage” most of them are not identified as fraudulent.

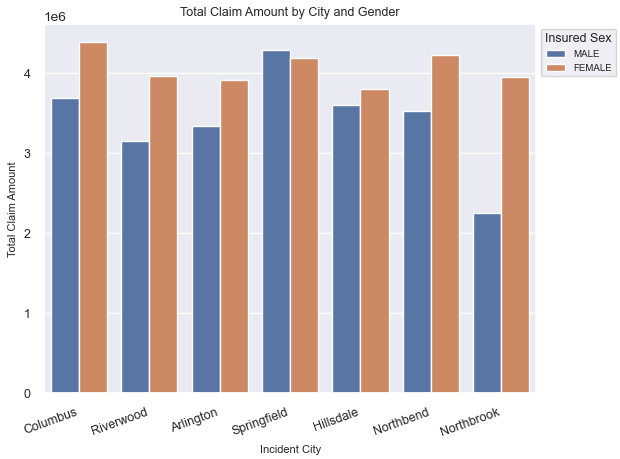


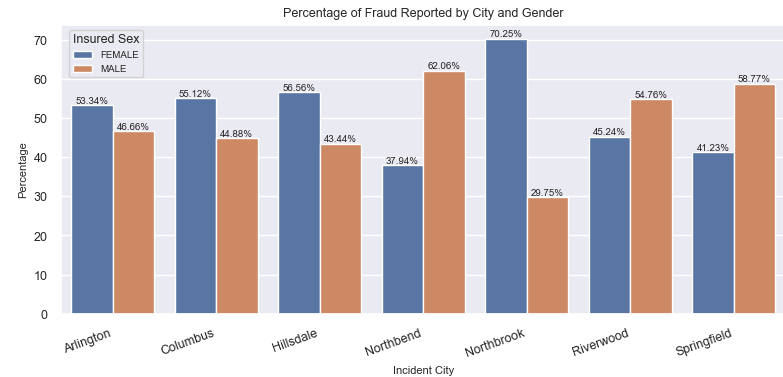
Analyzing by 'incident date' suggests that the number of reported fraudulent cases has decreased over time. Moreover, there seems to be no correlation between the total claimed amount and fraudulence. For instance, the day with the highest claimed amount does not necessarily coincide with the highest reported fraudulent cases.

The image below illustrates the distribution of claimed amounts by city and gender. It's evident that in all cities, one gender has a higher representation.

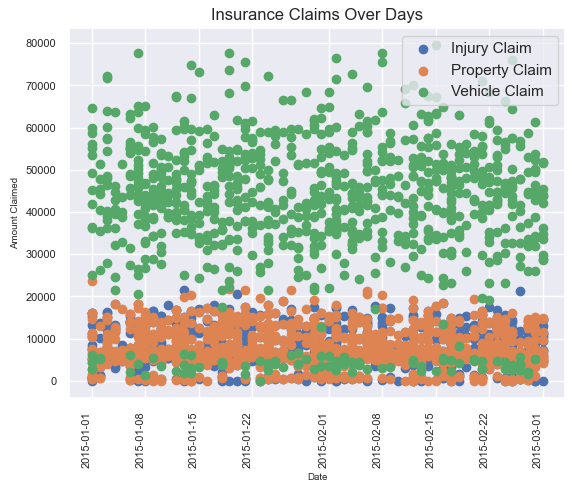


However, as we've learned, correlation doesn’t necessarily imply causation. An image is displayed to investigate whether the higher frequency of fraudulent claims in certain genders is due to their prevalence. For example, in *Riverwood*, there are more males reported for fraudulence, while the total number of claims shows more females. This suggests a higher risk for males to make fraudulent claims in this city.



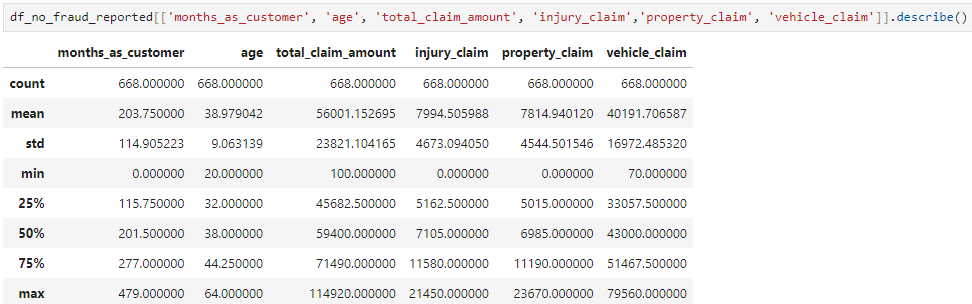


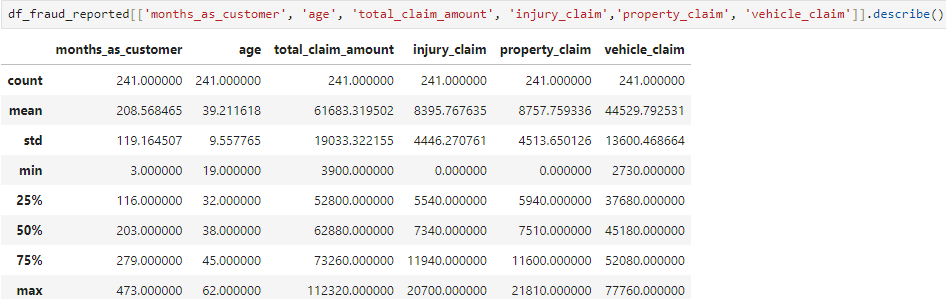
The scatter plot below suggests a normal distribution in 'Property Claim' and 'Injury Claim', whereas 'Vehicle Claim' stands out as the costliest, with a wider range of claimed amounts.



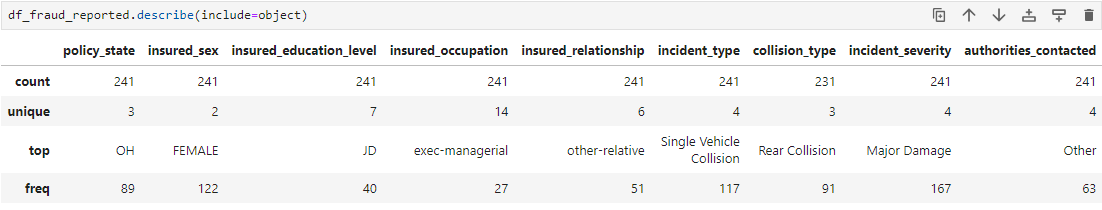
The '.describe' function showcases the statistical variables of six selected features.

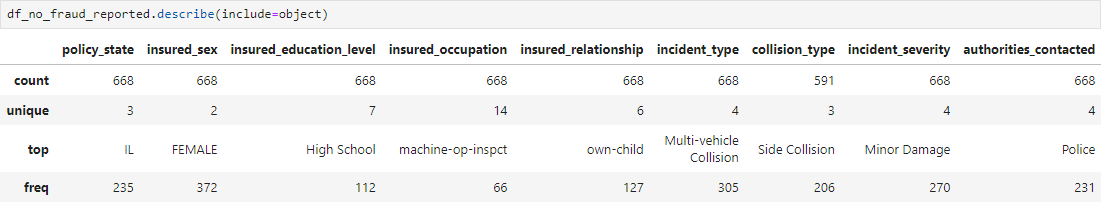
The images below suggest that the mean of the “total claim amount” is around 10% higher when reported as fraudulent. Similarly, it occurs in injury claims, property claims and vehicle claims.



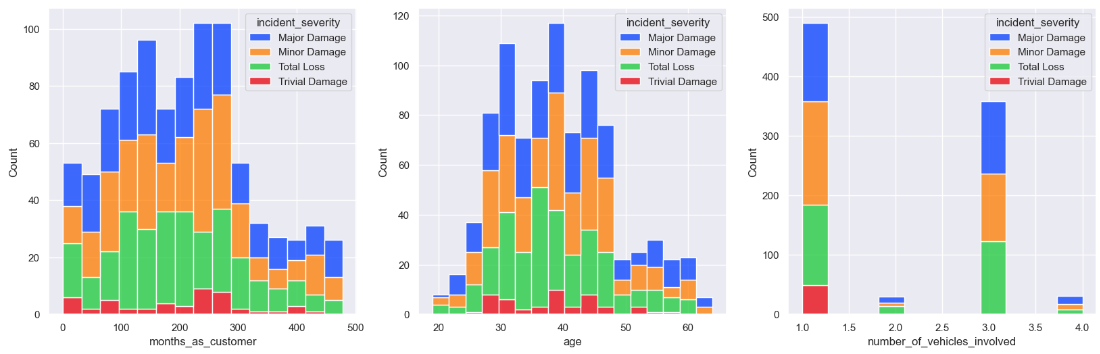


When ‘object’ is included in the code, it shows that in fraudulent claims the most common “incident\_severy” is Major Damage, while in not fraudulent claims is Minor Damage.



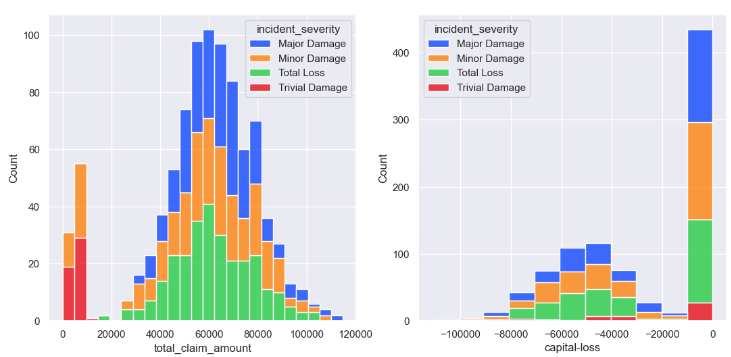


Since "incident\_severity" appears to be correlated with fraud detection, histograms are displayed to thoroughly examine five features.



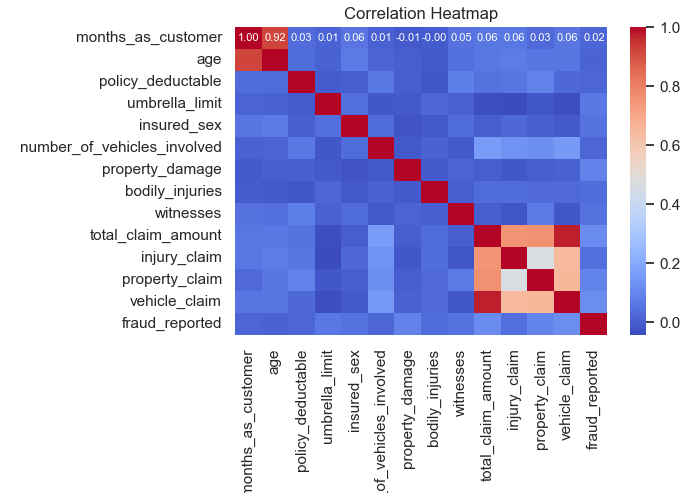
Insights gained:

* It suggests that the majority of the oldest customers typically do not file claims for trivial damage. Customers between 200 and 300 months are more likely to open a claim.
* The highest number of claims is among customers in their 40s, which could be attributed to the likelihood of car ownership at this age. However, it does not necessarily indicate that individuals in this age group are more inclined to file a claim.

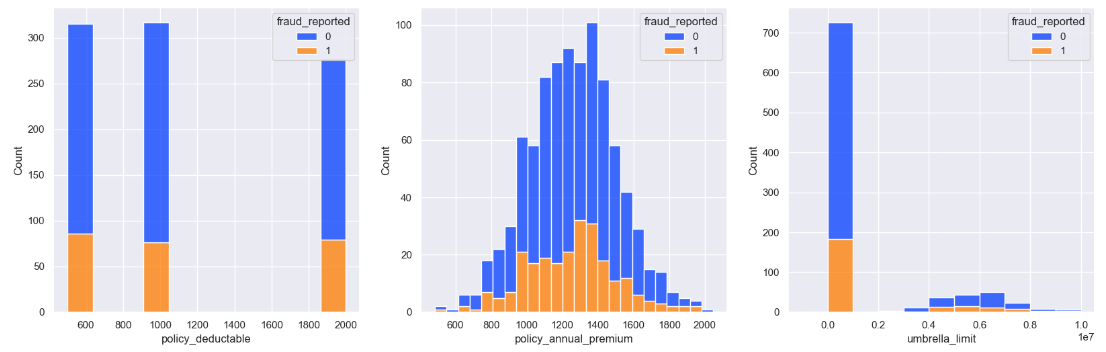


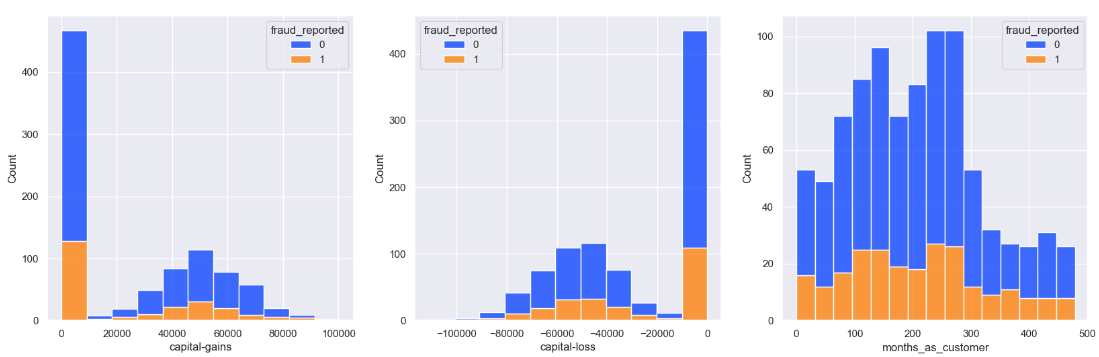
* Lower amounts claimed are attributed to “Trivial Damage” and “Minor Damage”. However, Minor Damage sill appears close to the highest amount claimed.
* “Capital loss” is more likely to be 0.

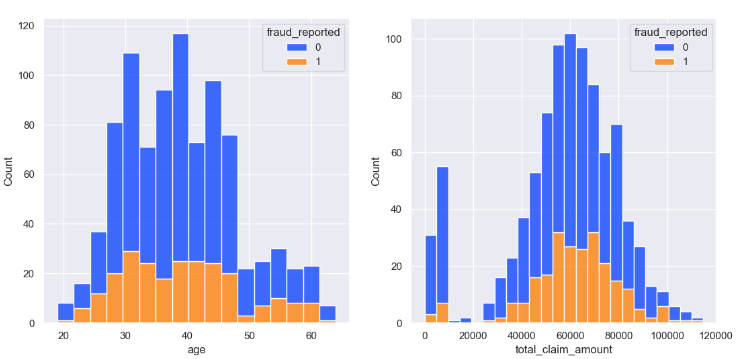
### Heatmap

Numerical columns are separated to create a correlation heatmap. It indicates a correlation between 'months\_as\_customer' and 'age', which is logical since older individuals are more likely to have been customers for a longer period. Additionally, correlations are observed within the claimed amounts, suggesting some association between the total amount and the types of claims.



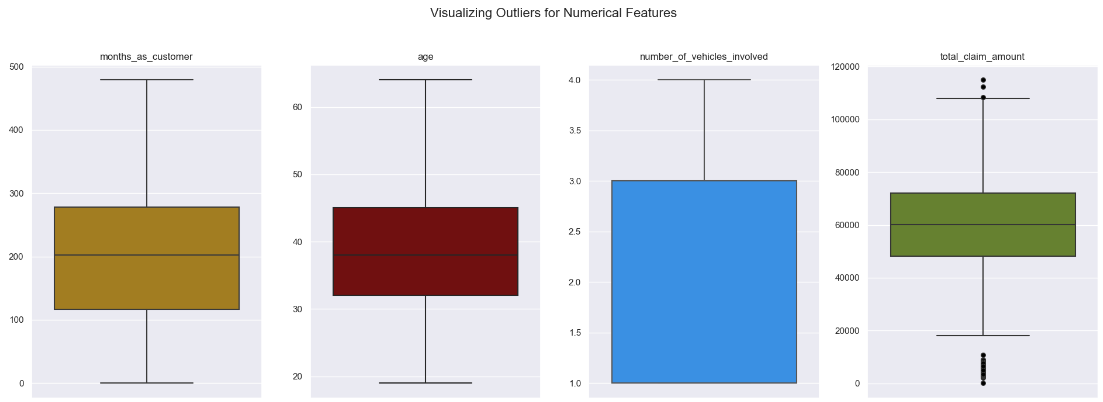






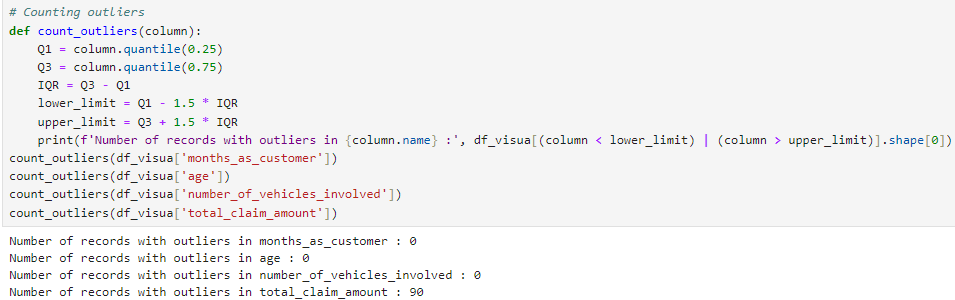
### Outliers

Identifying outliers is an important step as some machine learning models are sensitive to them.

Four features, 'months\_as\_customer', 'age', 'number\_of\_vehicles\_involved' and 'total\_claim\_amount' are displayed to check its distribution:

As is shown in the images above, only 'total\_claim\_amount' contains outliers.

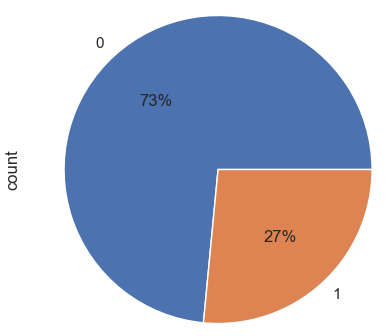
A code based on the quantiles is used to count the total of outliers in each feature:



There are 90 rows identified as outliers in the 'total\_claim\_amount', which is considered normal due to the fewer claims with extreme amounts. Since altering them wouldn't result in any significant improvement to the model, I've opted to retain them.

### Target distribution

The feature “fraud\_reported” will be used as a target in order to make predictions about whether a claim is identified as fraudulent or not. Displaying a pie chart we can see that the class is highly imbalanced with 73% of non-fraudulent and 27% being reported as fraudulent. This is an issue that will be handled later.



**Conclusions**: The data analysis process involved several steps, including dataset exploration, visualization, outlier detection, and understanding target distribution. Insights gained include through EDA:

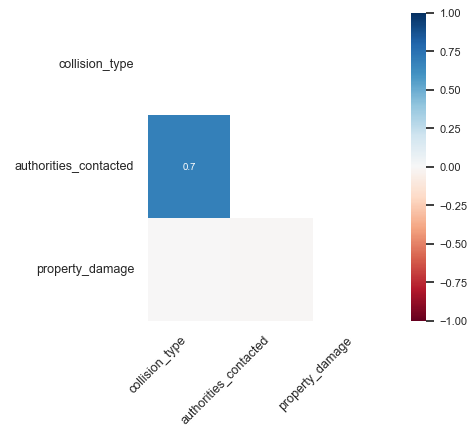
* The majority of oldest customers typically do not file claims for trivial damage, while customers between 200 and 300 months are more likely to open a claim.
* The highest number of claims is among customers in their 40s, possibly due to car ownership at this age, but not necessarily indicating a higher inclination to file a claim.
* Lower claim amounts are associated with "Trivial Damage" and "Minor Damage," although "Minor Damage" appears close to the highest amount claimed.
* "Capital loss" is more likely to be 0.

# Data Preparation

### Null values

The initial step in Data Preparation involves handling null values, as they can potentially impact the analyses and any techniques applied to the dataset.

A heatmap is displayed within the columns containing null values to identify potential correlations among them. Notably, the null values in the "collision type" column exhibit a strong correlation with the "authorities contacted" column.



Given that 'authorities\_contacted' contains 91 rows with null values and exhibits a strong correlation with null values in 'collision\_type', it suggests that these rows may not provide meaningful insights. Therefore, they are dropped from the dataset.

Null values within the features 'collision\_type', 'property\_damage', and 'police\_report\_available' are filled with 'Unknown'. Given that the missing values are random and represent more than 15% of the data, filling them in with another variable may not be beneficial.



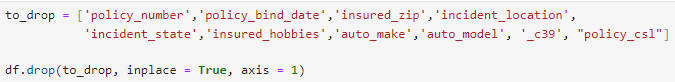
### Date type

The feature “incident\_date” is transformed into a date type.

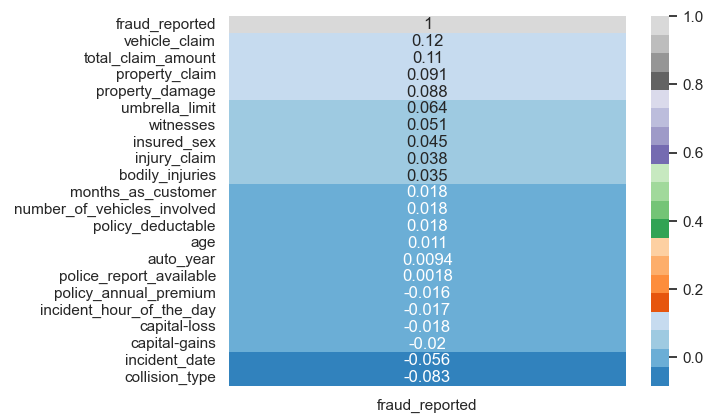


### Columns to drop

Columns that are not useful for this study are dropped from the dataset:



In addition, a heatmap is displayed to show the correlation of the columns with the feature “fraud\_reported”, where those with low correlation will be dropped.



Dropped columns:

****

### Transforming categorical into numerical

Categorical features with up to four unique variables are encoded using *LabelEncoder:*





Categorical features that have more than three unique variables are encoded employing *OneHotEncoder.* It transforms each categorical feature into a new set of binary features (0s and 1s). Each binary feature corresponds to one of the unique categories in the original feature. OneHotEncoder is useful because many machine learning algorithms require numerical input, and encoding categorical features into a numerical format allows these algorithms to effectively learn from categorical data.



So far the pre-processed dataset has 909 rows and 36 columns:



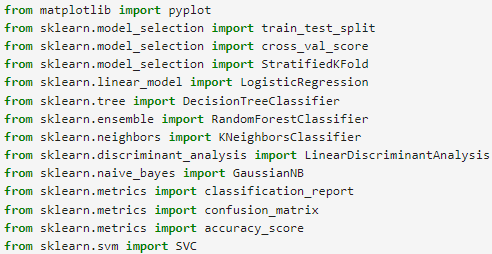
**Summary**: The data preparation process included:

* Transforming tree categorical features into numerical employing One Hot Encoder: ‘insured\_occupation’, ‘incident\_severity’, ‘incident\_city’, which resulted in 22 columns.
* Categorical features with up to four unique variables are encoded using LabelEncoder. Columns: "insured\_sex", "collision\_type", "property\_damage", "police\_report\_available", "fraud\_reported"
* The feature “incident\_date” is transformed into a date type
* Columns identified as not necessary to machine learning models were dropped: '\_c39', 'authorities\_contacted', 'auto\_make', 'auto\_model', 'auto\_year', 'capital-gains', 'capital-loss', 'collision\_type', 'incident\_date', 'incident\_hour\_of\_the\_day', 'incident\_location', 'incident\_state', 'incident\_type', 'insured\_education\_level', 'insured\_hobbies', 'insured\_relationship', 'insured\_zip', 'police\_report\_available', 'policy\_annual\_premium', 'policy\_bind\_date', 'policy\_csl', 'policy\_number', 'policy\_state'
* Null values present in the feature "authorities\_contacted" were dropped. Null values in the columns"collision\_type", "property\_damage", "police\_report\_available" were filled with "Unknown".

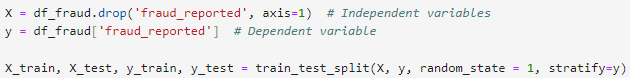
# Deciding the model

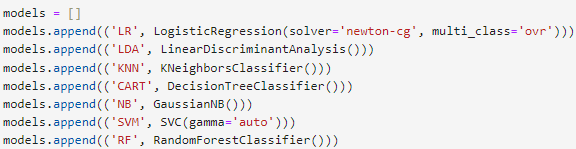
With a pre-processed dataset in hand, I'll test seven models and choose the top four based on their metrics for further analysis and improvement. These models are selected for their interpretability and efficiency, ensuring they are easy to interpret and not overly time-consuming.

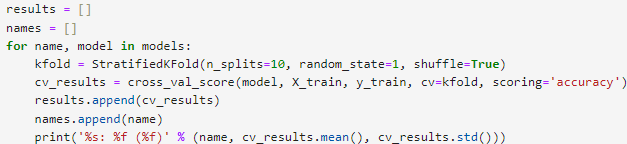
Libraries used:

****

The target variable 'y' is set to "fraud\_reported", while the remaining columns are assigned to 'X'. The dataset is split into training and testing sets to ensure the model's efficiency. Given the imbalanced class, the 'stratify' parameter is set to 'y', ensuring that class distribution is maintained in both the training and testing sets. The test set serves as a sample of the data reserved for final model evaluation, providing confidence in its accuracy on unseen data. I've allocated 80% of the data for model training and reserved 20% for testing purposes.

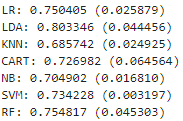
****

****

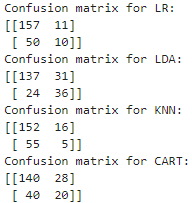
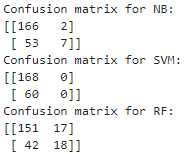
****

### Metrics

Initially, when analyzing only accuracy, the metrics might seem satisfactory.

****

However, upon examining the confusion matrix, particularly in cases where fraudulence is reported, we observe poor performance:

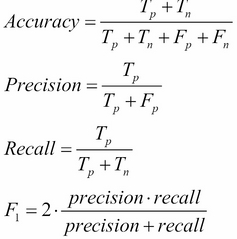
For this study the false negative rate is substantial. The main purpose of a fraud detection model is to minimize these false negatives. To do so, the first step would be to choose the right evaluation metric. In a classification task, there are four important metrics: Accuracy, Precision, Recall and F1 Score.

Accuracy provides the proportion of correctly classified instances.

Precision focuses on the accuracy of positive predictions.

Recall (Sensitivity or True Positive Rate): Recall measures the proportion of correctly predicted positive instances among all actual positive instances.

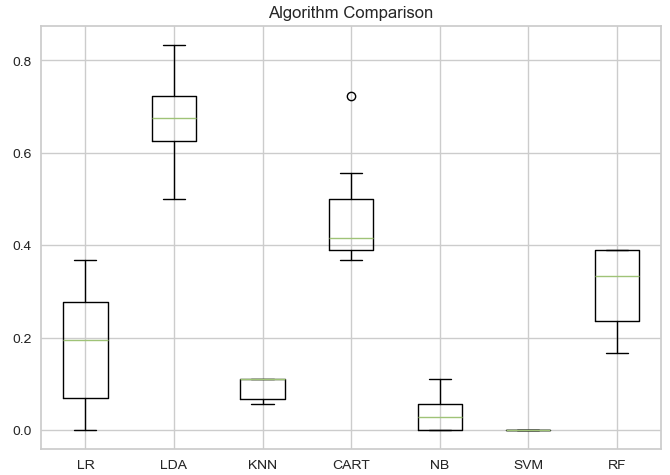
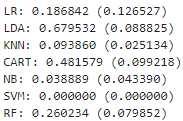
The F1 Score is the harmonic mean of precision and recall.



Selecting the right evaluation metric:

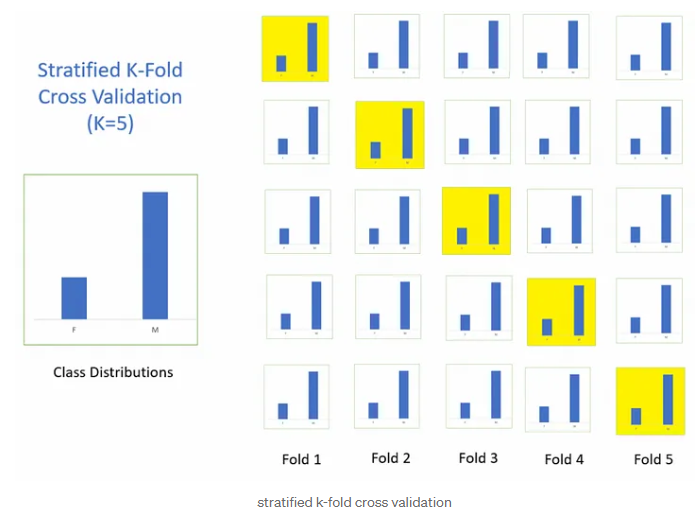
In scenarios where the priority is to minimize false negatives while accepting the possibility of false positives, we opt for decision thresholds characterized by either **low Precision** or **high Recall** values. For instance, in fraud detection systems, the primary concern is to ensure that no fraudulent activity goes undetected, even if it means tolerating some instances of misclassifying legitimate transactions as fraudulent. This approach is justified by the fact that the absence of fraud can be confirmed through additional investigations, whereas failing to identify fraudulent activity can have significant consequences.

Chosen *Recall* as the main metric, it is notable how it decreased when compared with *Accuracy*:

**** 

### Cross Validation

The models are employed using Stratified K-Fold Cross Validation. Predictive model building prompts the question: How accurate are the predictions? Quantifying expected model performance is vital in machine learning projects. Cross-validation methods assess prediction effectiveness by dividing data into training and test sets. The model is trained on one set and tested on another, allowing estimation of performance on unseen data. Stratified K-Fold Cross Validation is similar to K-Fold Cross Validation, but with an additional step to ensure that each fold has approximately the same proportion of target classes as the whole dataset. This is particularly useful when dealing with imbalanced datasets, which is the case of this dataset. The image below illustrates how it works:



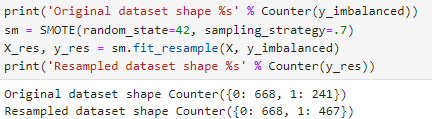
Based on those metrics, four models are chosen, three of the highest and the one with the lowest: *Logistic Regression*, *Linear Discriminant Analysis*, *Decision Three Classifier* and *SVM* (Support Vector Classifier).

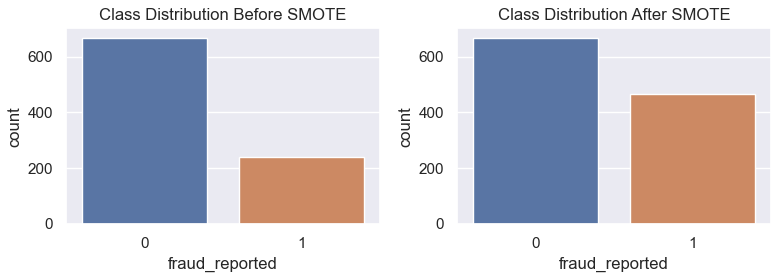
The next phase is to handle the unbalanced class, in order to improve model metrics.

# SMOTE

Given its unbalanced distribution, the SMOTE technique is employed to mitigate it. Addressing imbalance in datasets is crucial in machine learning to ensure more reliable results. Without addressing the imbalance, models may overfit by memorizing and predicting all instances as non-fraudulent.

SMOTE (Synthetic Minority Over-sampling Technique) is a method employed to balance the distribution of classes within a dataset by oversampling or undersampling, resulting in a more evenly distributed dataset. In this project, sampling strategy 0.7 was used.

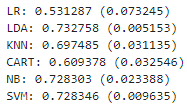




# PCA (Principal component analysis)

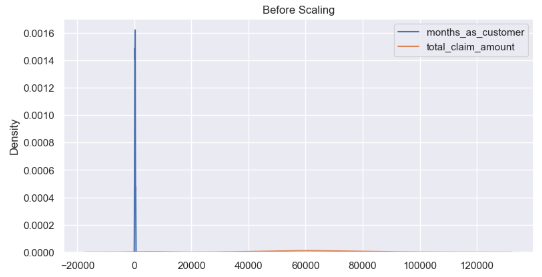
Given that the actual dataset contains 53 columns, I'll explore whether reducing its dimensionality would lead to more reasonable metrics. The dataset used is without SMOTE.

Principal component analysis, or PCA, is a statistical procedure that allows you to summarize the information content in large data tables utilizing a smaller set of “summary indices” that can be more easily visualised and analyzed (Sartorius, 2020). The efficacy of PCA (Principal Component Analysis) is contingent upon the data distribution. In the context of this dataset, the metrics indicate that employing PCA may not be meaningful.



# Standardization of the data

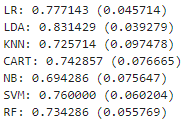
Standardization is a scaling technique wherein it makes the data scale-free by converting the statistical distribution of the data into the format mean = 0 (zero) and standard deviation = 1. In the images below we can compare how the distribution of columns “months\_as\_customer” and “total\_claim\_amount” changes:

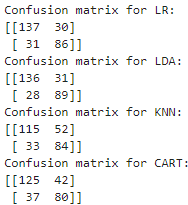
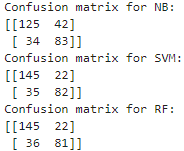




Using standardised techniques makes the dataset more amenable to various machine-learning algorithms, and may be favourable in some scenarios.

Looking at the recall metrics of the models, it is notable the improvement after SMOTE and standardizing:



# Modelling

At this stage, the selected models undergo parameter tuning, delving deeper into the models to comprehend and interpret the results.

## LogisticRegression

Logistic regression is a supervised machine learning algorithm used for classification tasks where the goal is to predict the probability that an instance belongs to a given class or not. It is a statistical algorithm which analyzes the relationship between two data factors. In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tend to be 1, and those below the threshold value tend to be 0 (GeeksforGeeks, 2017).

Logistic Regression has three important parameters and choosing the right ones can potentially increase metrics. They are solvers, penalty and C.



### Hyperparameters tuned

Optimising the values of the hyperparameters, for a specific task, will often lead to superior model performance and thereby help to prevent overfitting.

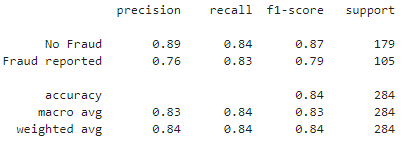
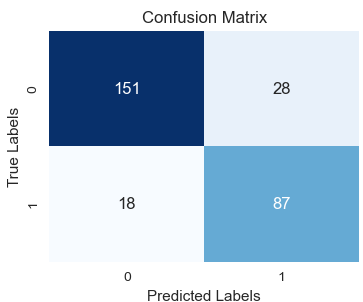
Grid Search is one of the most popular choices for hyperparameter tuning and involves an exhaustive search through a list of all possible hyperparameter values to try. Every possible combination is tried, and then the best-performing model can be selected. While this option tends to perform well, it is computationally expensive in larger datasets (Anon, 2023).

*GridSearchCV* is used coupled with *RepeatedStratifiedKFold* which is a cross-validation method.

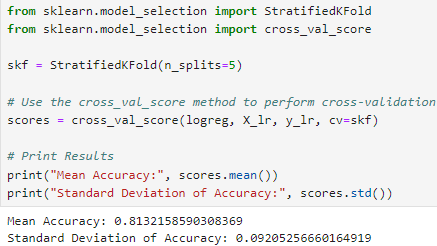


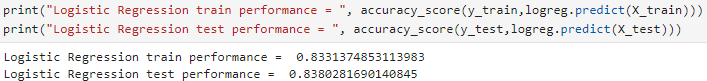
### Metrics

As it is shown above, the best score is achieved with 'C': 10, 'penalty': 'l2' and 'solver': ' newton-cg'.

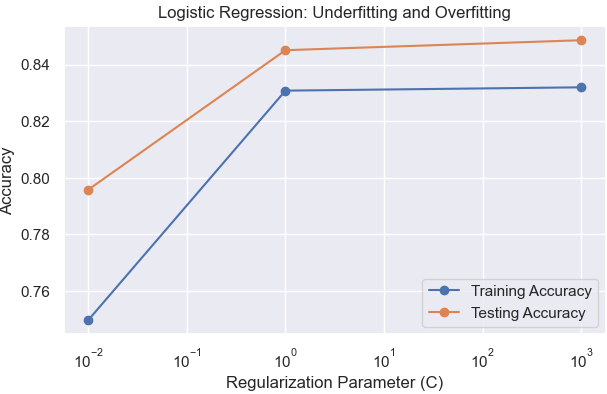
Cross-validation is particularly crucial in scenarios involving imbalanced datasets to ensure reliable accuracy estimates.





Train and test are both aligned in their accuracy score which suggests a good model fit.

The image below illustrates the change in accuracy across different selected parameters for C. In case of underfitting or overfitting, it could be identified by it. A lower C could potentially cause underfitting.



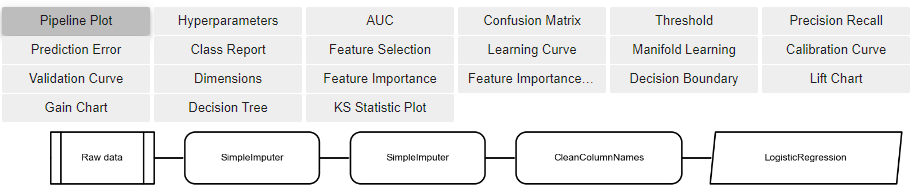
The Logistic Regression model is applied without any scaling model, due to its metrics being higher and it makes it easier to results interpretability.

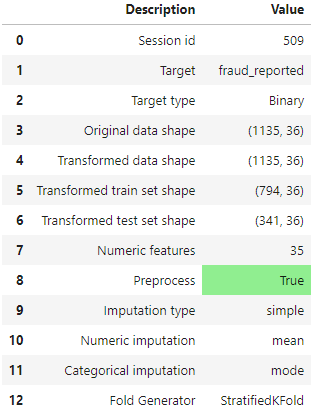
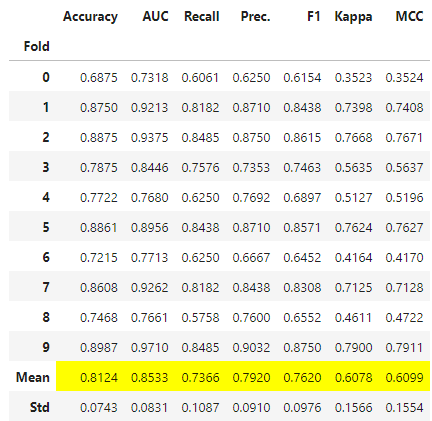
### Evaluation results using PyCaret

PyCaret is a library that provides analysis and model explainability functions.

When using the function “evaluate\_model” it provides a brief overview of the model and several options to plot the model outcomes.

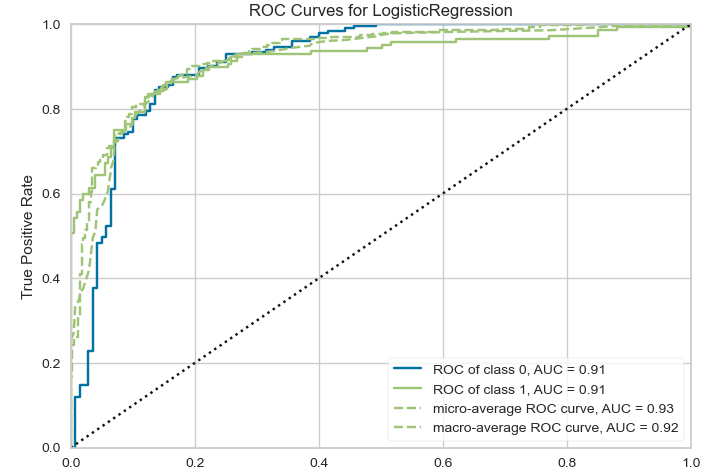
The image below illustrates the available options. These evaluation metrics and visualizations provide a comprehensive understanding of the logistic regression model's performance and can help to make informed decisions about model deployment or further model refinement:

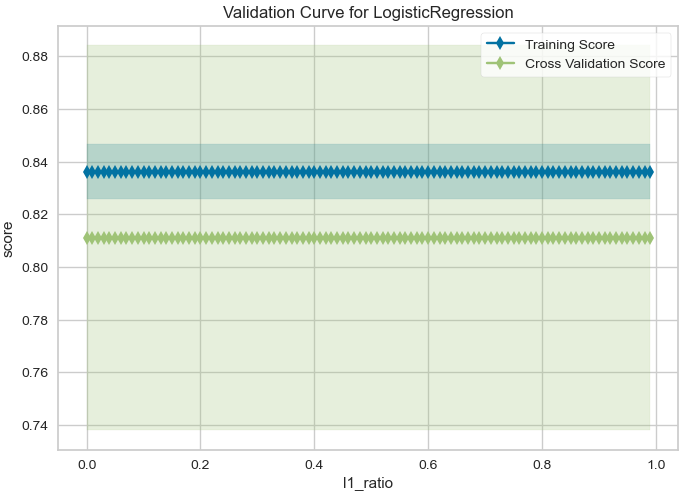


The image above on the right illustrates the model metrics for each fold during cross-validation.

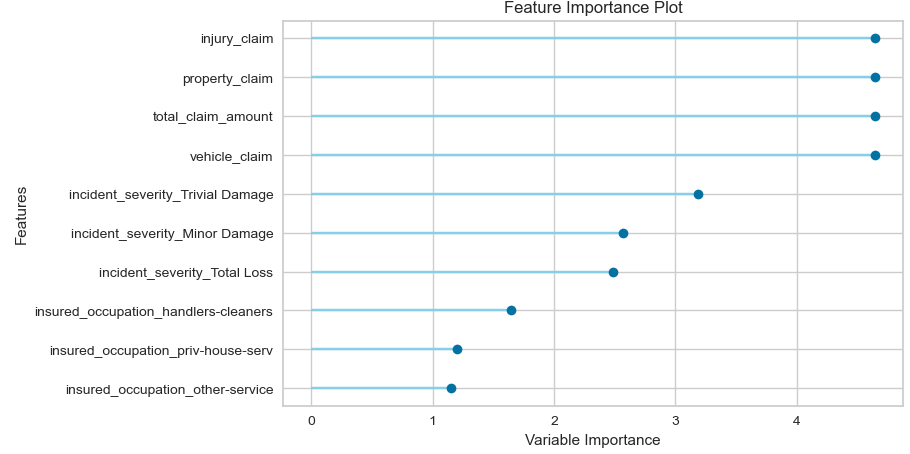
Area Under the Receiver Operating Characteristic Curve (AUC-ROC) serves as a pivotal metric in assessing the performance of classification models, offering insights into their discriminatory power and predictive accuracy. By plotting the true positive rate against the false positive rate across varying classification thresholds, the ROC curve offers a comprehensive visualization of a model's ability to distinguish between different classes. A higher AUC-ROC score indicates superior discriminative ability. In this case, the logistic regression (LR) model achieved an ROC score of 0.91, which approaches 1, indicating near-perfect classification performance.



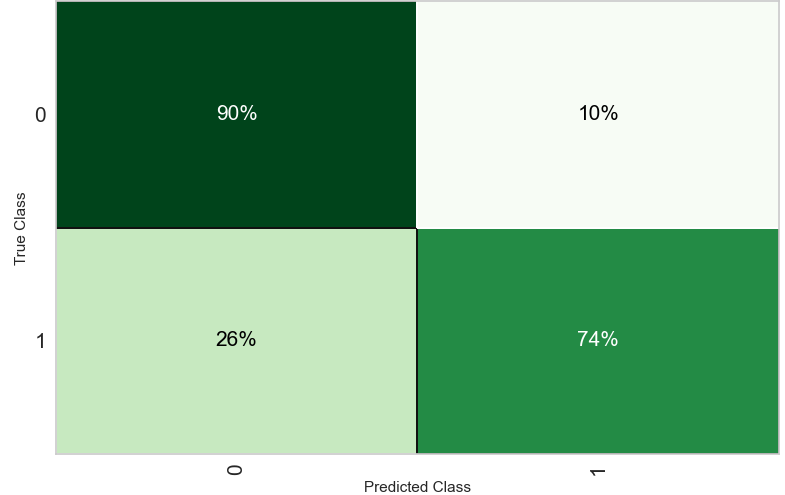


By the Feature Importance, it suggests that ‘*injury claim’, ‘property claim’ and ‘total claim’* have a strong influence on the model outcome, which means that according to those amounts, it may have a higher probability in a claim to be reported as fraudulent or not.

Followed by ‘*incident\_severity\_Trivial Damage*’ and ‘*incident\_severity\_Minor Damage*’ which makes sense if we consider that the earlier analyses, showed that most of the Trivial and Minor Damage are represented by those reported as non-fraudulent claims.



The Confusion Matrix illustrate that only 26% of the fraudulent claims might not be captured.

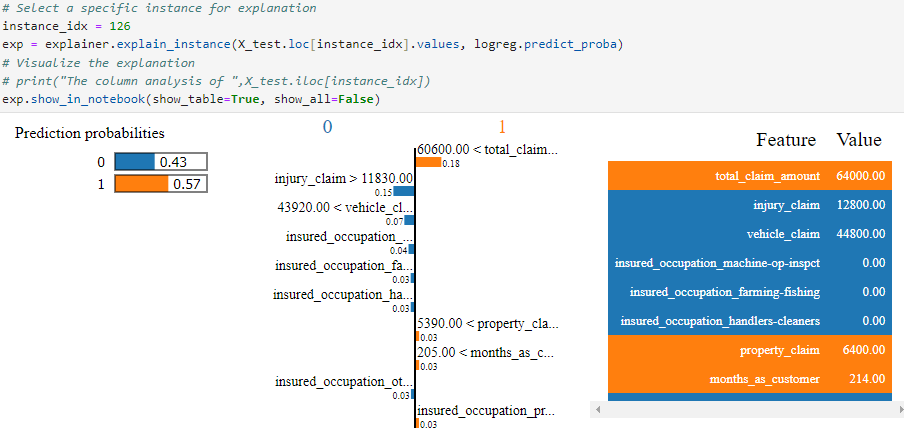


### LIME

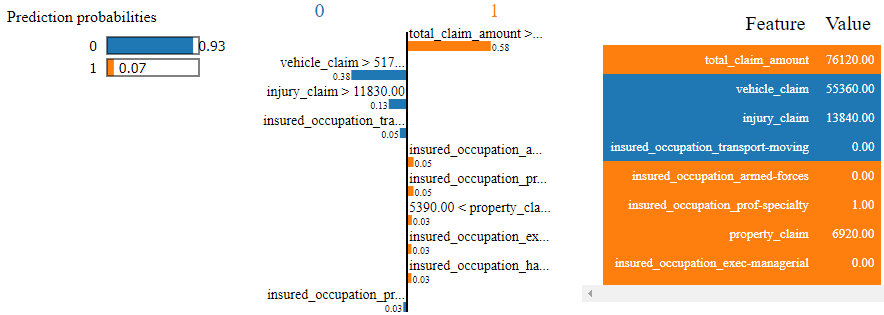
In order to build trust in the model, we run multiple cross-validations and perform hold-out set validation. These simulations give an aggregated view of model performance over unknown data. This does not help in understanding why some of our predictions are correct while others are wrong nor can we trace our model’s decision path. In other words, we cannot understand its learning or figure out its spurious conclusions.

LIME ( Local Interpretable Model-agnostic Explanations )is a novel explanation technique that explains the prediction of any classifier in an interpretable and faithful manner by learning an interpretable model locally around the prediction (Sharma, 2018).

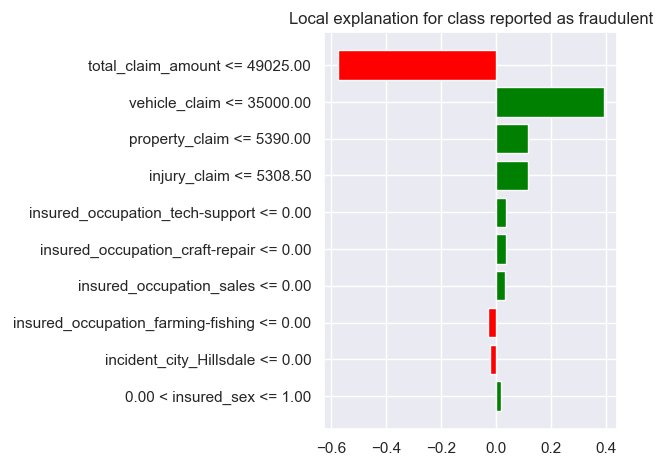
Using LIME I selected one instance to analyse the model explainer. The image below shows the features that contributed to being classified as fraudulent. It suggests that the ”total\_claim amount” is the most relevant in this instance.



On the other hand, when analysing one instance classified as non-fraudulent, the most important features are “vehicle claim” and “injury claim”:



The image below explains the predicted claim at index 126. It illustrates that for each feature, there is a range of *Lime* values, which cumulatively contribute to determining the predictive class.



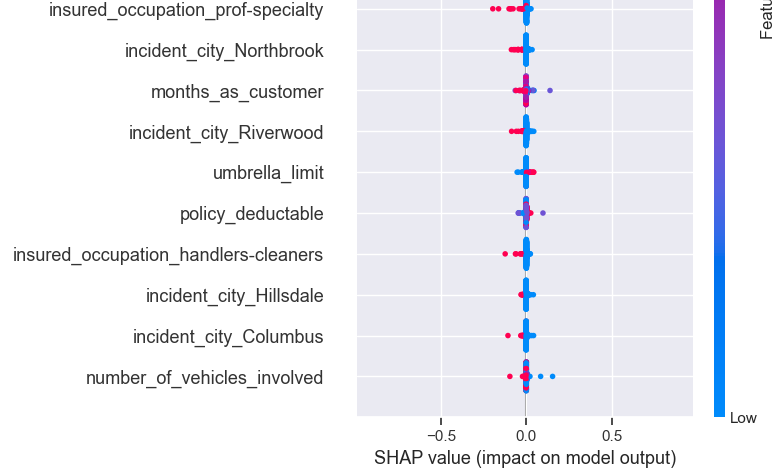
### SHAP

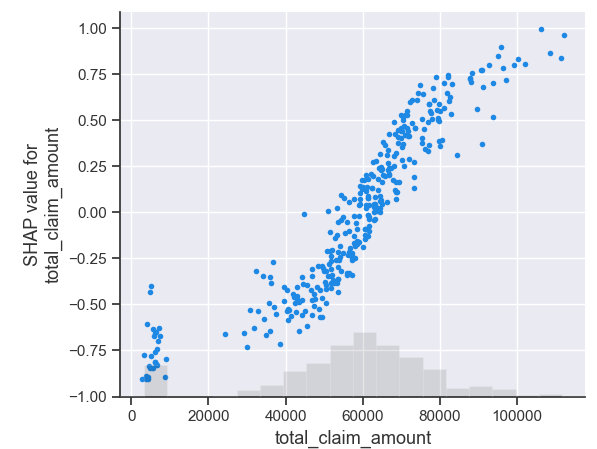
SHAP (SHapley Additive exPlanations) values explain machine learning model outputs by employing a game theoretic approach. Each feature is assigned an importance value, representing its contribution to the final prediction. These values illustrate how each feature influences predictions, their significance relative to others, and the interplay between features. Positive SHAP values indicate a positive impact on predictions, while negative values suggest a negative impact, with magnitude indicating the strength of the effect. They provide a consistent and objective means of understanding feature impact in machine learning models.

Understanding a linear model typically involves examining the coefficients assigned to each feature. These coefficients indicate how the model's output changes with variations in input features. However, relying solely on coefficients to gauge feature importance has limitations. Coefficients are influenced by the scale of input features, meaning a larger coefficient doesn't necessarily signify greater importance. For instance, if we measure a home's age in minutes rather than years, the coefficient for the age feature would be minuscule despite its significance. Thus, the magnitude of a coefficient alone isn't a reliable indicator of feature importance in a linear model. In the dataset, the "total claim amount" might appear as the most important feature due to its significantly higher numerical value compared to other features. However, this doesn't necessarily mean it's the most important feature in influencing the model's output. Similar to the issue with coefficients, the scale of the feature can skew its perceived importance. Therefore, solely considering the numerical magnitude of the "total claim amount" could lead to misleading conclusions about its actual significance in the model.

SHAP Values explainer for *X\_test:*

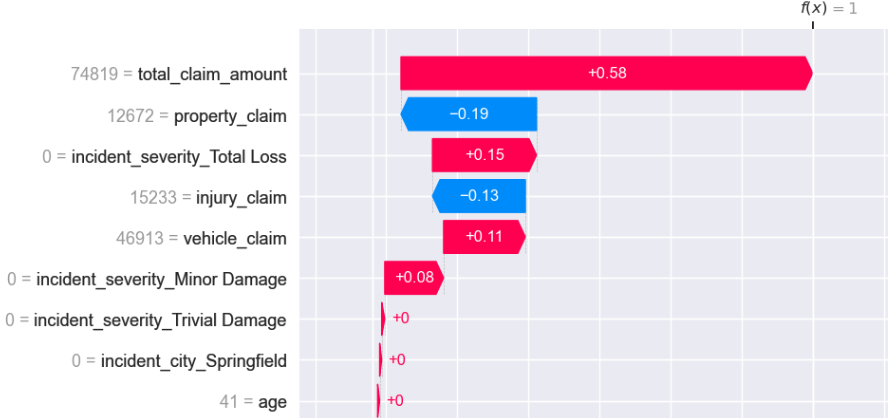


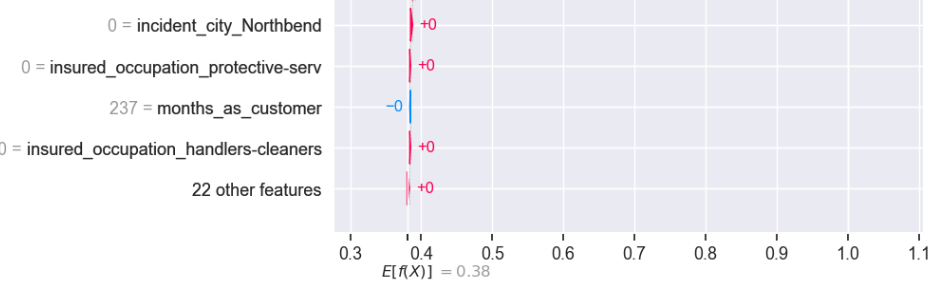




Analysing the *waterfall* function in SHAP for the instance in index 20:

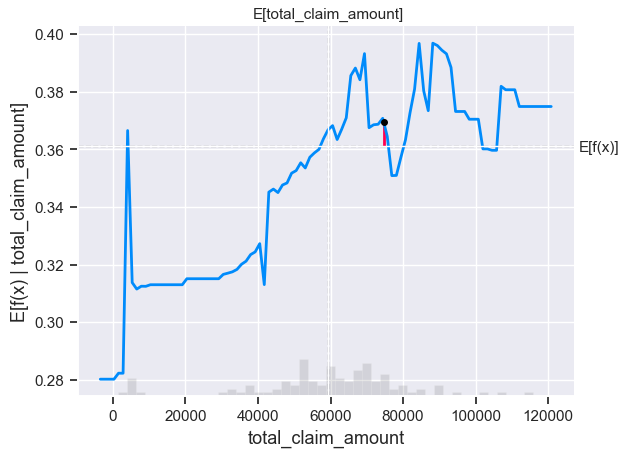


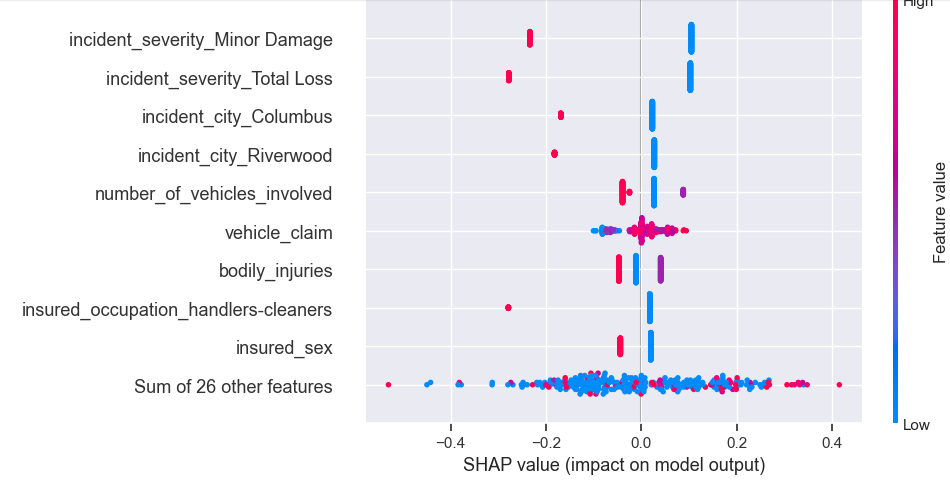




In the context of SHAP, the values displayed in the waterfall plot typically represent the contributions of individual features to the difference between the model output and a reference prediction. These contributions are calculated based on the SHAP values generated for each sample by the model.

Shap Explainer of X\_test (model ebm):



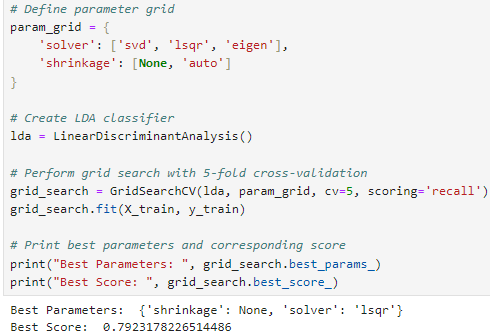


## Linear Discriminant Analysis (LDA)

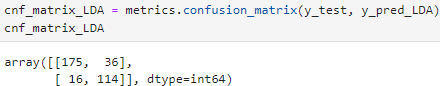
Linear Discriminant Analysis (LDA) is a statistical technique for categorizing data into groups. It identifies patterns in features to distinguish between different classes. For instance, it may analyze characteristics like size and colour to classify fruits as apples or oranges. LDA aims to find a straight line or plane that best separates these groups while minimizing overlap within each class. By maximizing the separation between classes, it enables accurate classification of new data points. In simpler terms, LDA helps make sense of data by finding the most effective way to separate different categories, aiding tasks like pattern recognition and classification (Dash, 2021).

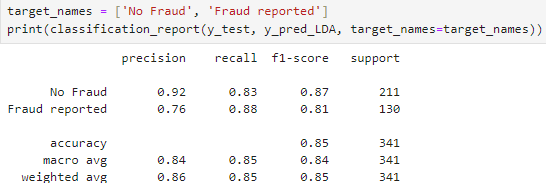
### Hyperparameters tuned

To obtain the optimal parameters, GridSearchCV is employed. When configured to prioritize the 'recall' scoring metric, it identifies 'lsqr' as the best solver. However, experimentation with 'eigen' reveals even higher performance scores.

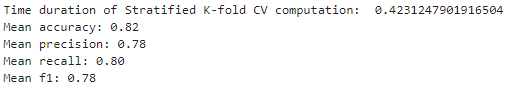


### Metrics

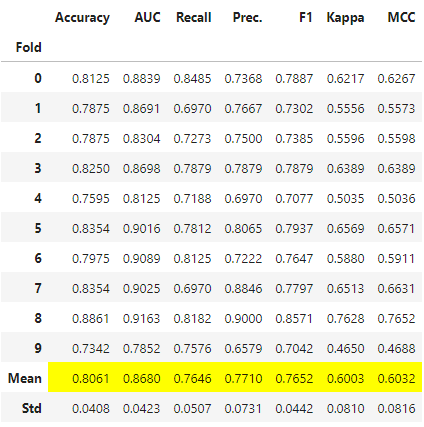


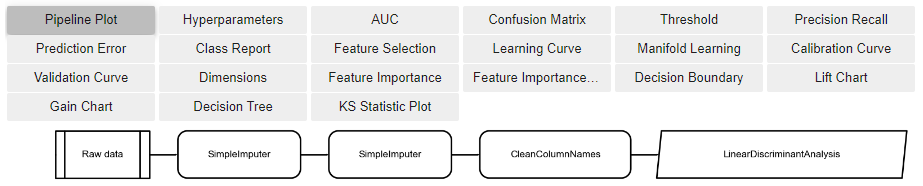


Upon examining the cross-validation metrics, it shows reasonable results:

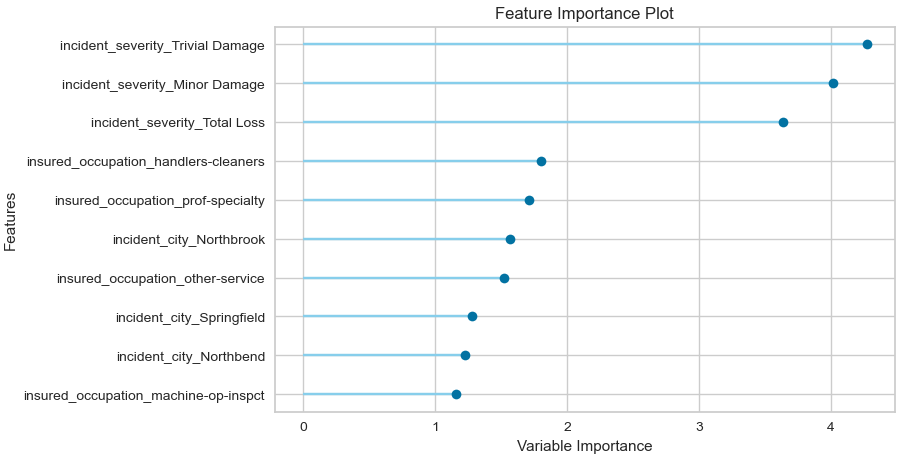


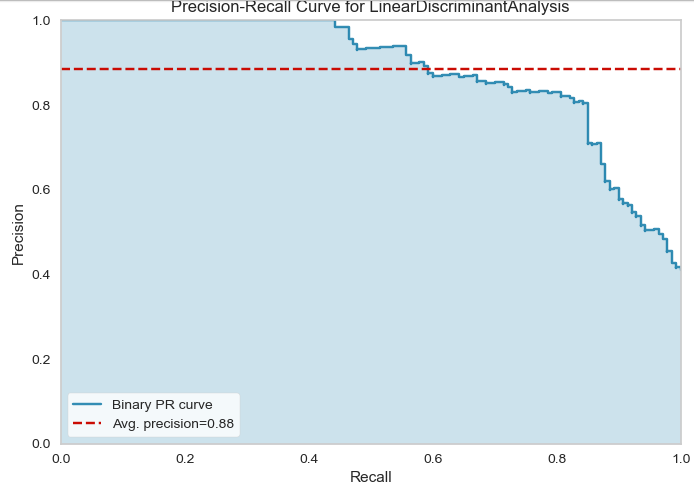
### Evaluation results using PyCaret



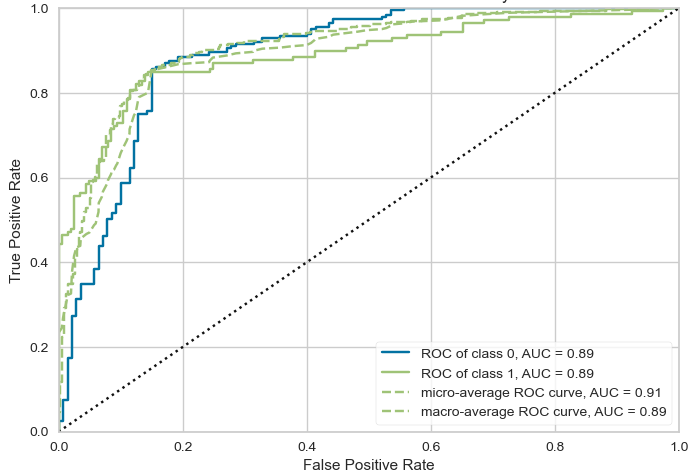


Like logistic regression, Linear Discriminant Analysis (LDA) also reveals that incident severity significantly influences the model outcome. Conversely, the magnitudes of injury, property, or vehicle claims do not exhibit a strong influence on the model's predictions.



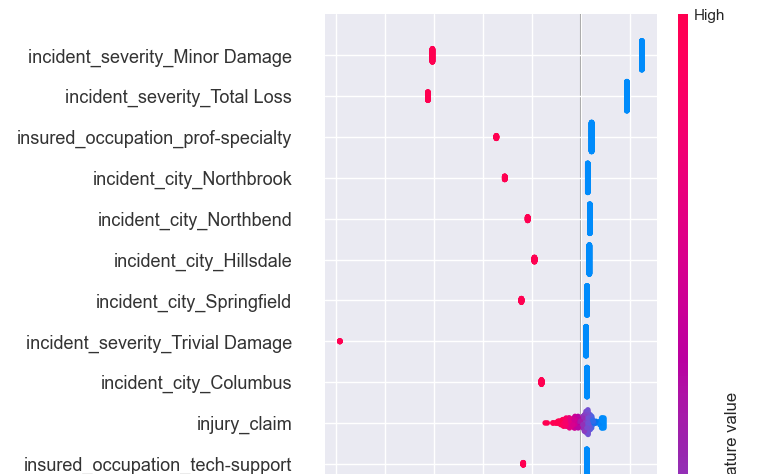


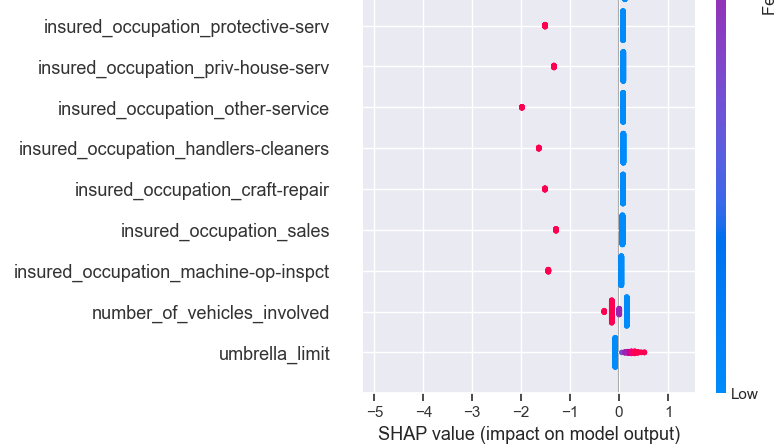
ROIC Curves for LinearDiscriminantAnalyses

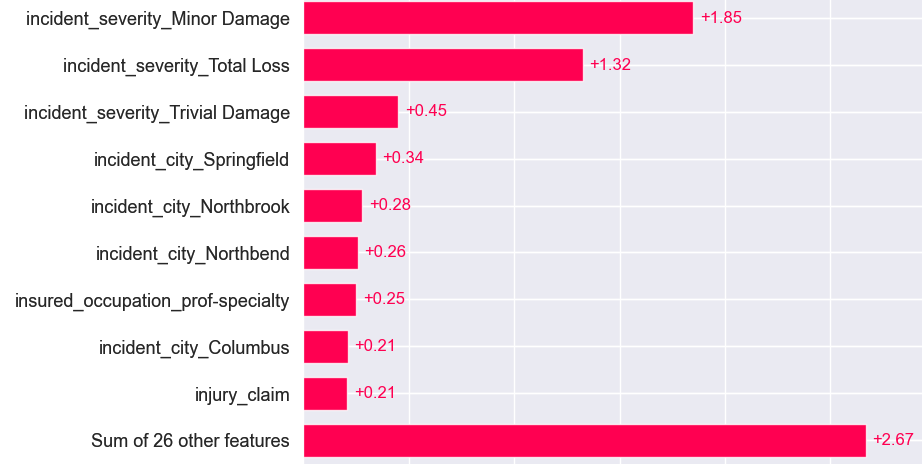


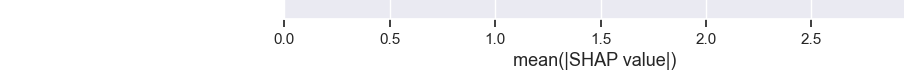
### SHAP (LDA)

When comparing SHAP values between LR and LDA, it becomes apparent that due to their distinct processes, they tend to exhibit varying degrees of feature importance.

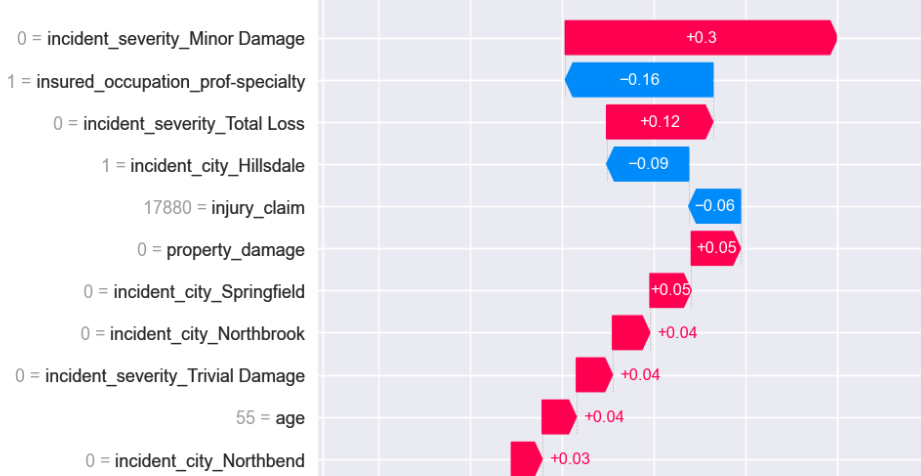








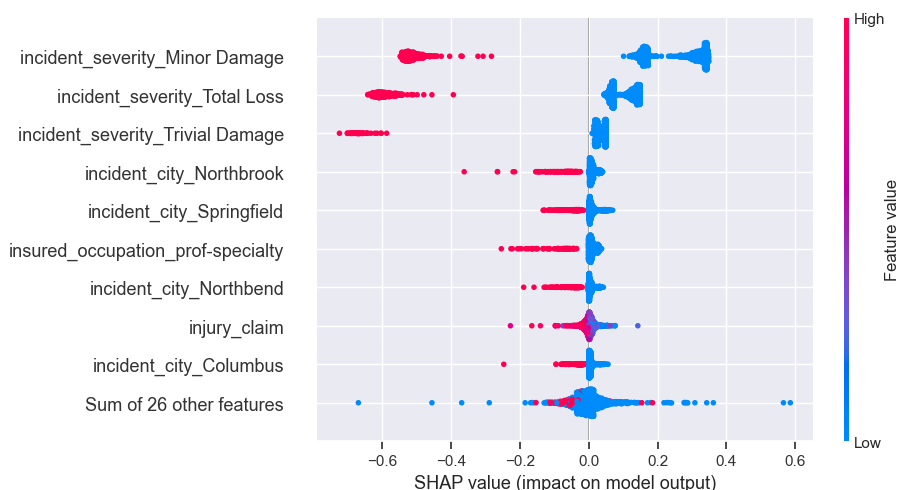
SHAP Waterfall for the instance in the index 20:





Beeswarm

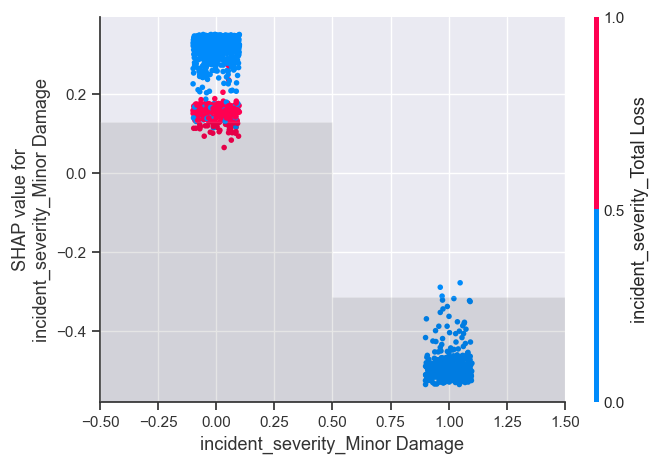
The image below demonstrates that 'incident severity Minor Damage' carries significant weight in non-fraudulent predictions, but its influence diminishes considerably in fraudulent cases.

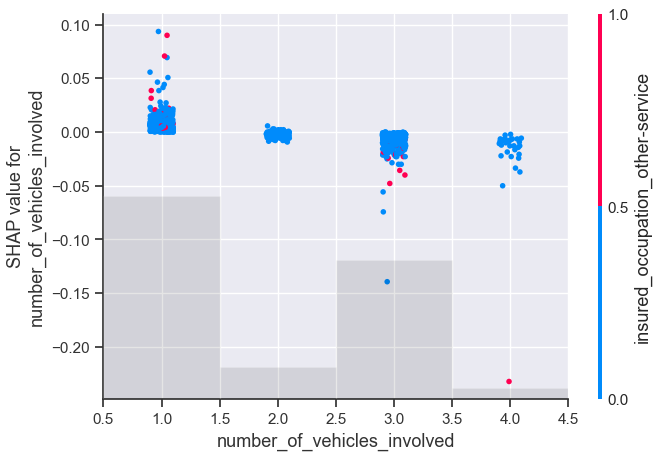


A SHAP heatmap visualizes the impact of features on model predictions. The colour intensity at each intersection indicates the feature's contribution to the prediction.



Since SHAP values represent a feature's responsibility for a change in the model output, the plot below represents the change in predicted fraudulent claims as the 'incident severity Minor Damage' changes. Vertical dispersion at a single value of 'incident severity' represents interaction effects with other features. To help reveal these interactions we can color by another feature. If we pass the whole explanation tensor to the colour argument the scatter plot will pick the best feature to colour by. In this case, it picks 'incident severity total loss'.





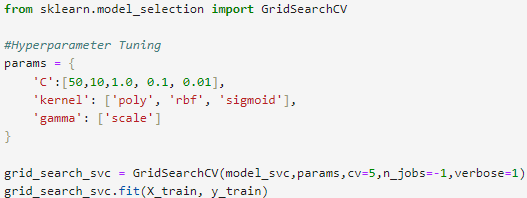
## SVC (Support Vector Classification)

In the mathematical context, an SVM refers to a set of ML algorithms that use kernel methods to transform data features by employing kernel functions. Kernel functions rely on the process of mapping complex datasets to higher dimensions in a manner that makes data point separation easier. The function simplifies the data boundaries for non-linear problems by adding higher dimensions to map complex data points (Spiceworks, n.d.).

Since SVM is sensitive to outliers, the scaled dataset is used.

### Hyperparameters tuning

Grid Search CV is used to hyperparameters tuning:

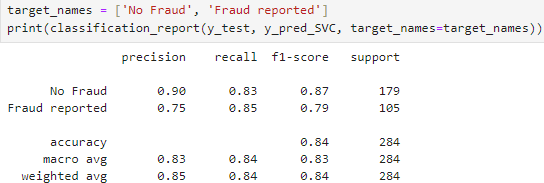




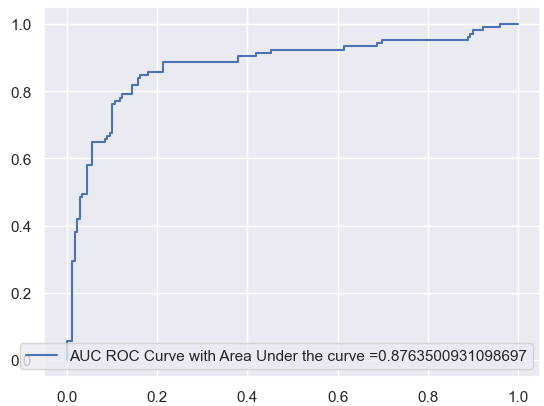
Once found the best parameters for the model, it is time to check its metrics.

### Metrics





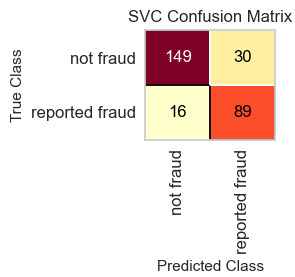
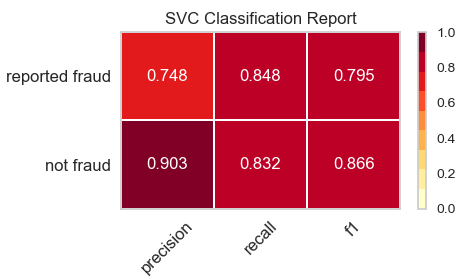
The AUC ROC shows 88% of confidence:



Cross Validation

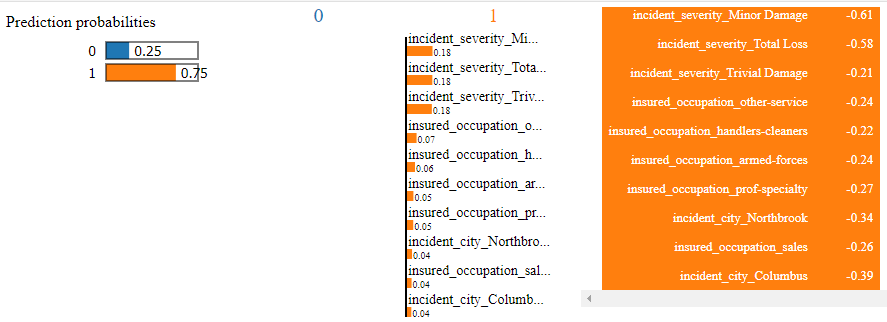


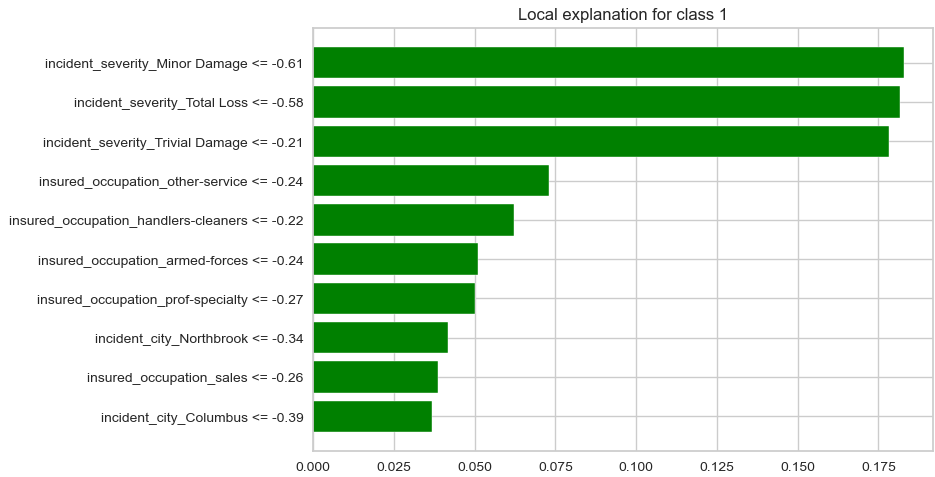
At the beginner, before scaling and handling the imbalanced class SVC wasn’t able to identify any fraudulent claim and after these techniques coupled with hyperparameter tuning, it can deliver high results:

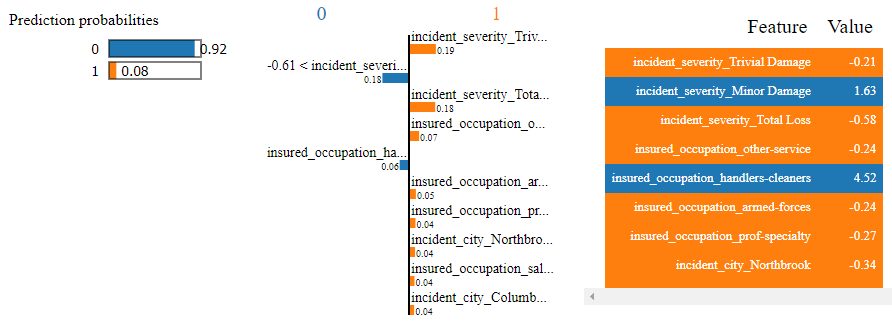
 

### LIME SVC

Instance at index 126:







With the Lime explainer, we can observe similarities in feature importance between the two instances above, even when one is predicted as fraudulent and another one as non-fraudulent.

## Decision Three (CART)

CART stands for Classification And Regression Tree. It is a type of decision tree which can be used for both classification and regression tasks based on a non-parametric supervised learning method.

### Hyperparameters Tuning:

In Decision Trees, the parameters consist of the selected features 𝑓, and their associated split points 𝑠, which define how data propagate through the nodes in a tree. Some of the most common hyperparameters include:

*Choice of splitting loss function*, used to determine (𝑓,s) at a given node

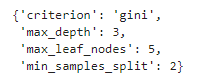
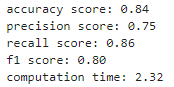
*Maximum depth of the tree*, which sets a hard limit on how much branching can occur

*Minimum number of samples for a split*, which places a lower bound on how much data must enter a node for it not to be considered a leaf node during training

*Maximum number of leaf nodes*, a value used to set an upper bound on the number of terminal points in the tree.

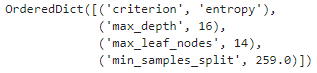
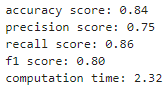
To tune the hyperparameters, two methods are employed: Grid Search, as previously described, and Bayesian Search.

Let’s start with Grid Search:

Bayes Search CV

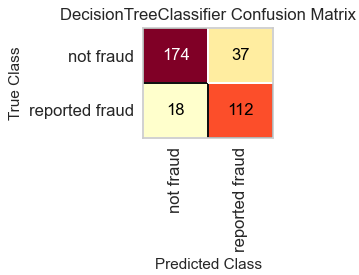
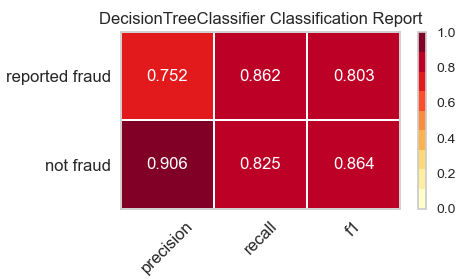
The hyperparameter space is defined by statistical distributions. We can further influence how the tuning performs through a careful selection of prior distributions. This method is also computationally efficient, but it is more complex to use or explain when compared with Grid Search (Anon, 2023).

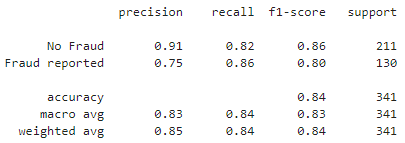
Given that both yield satisfactory metrics, I chose to utilize Bayes Search CV parameters due to their lower computational cost.

Running the model with hyperparameter-tuned, higher metrics are achieved. According to the confusion matrix, this model improved especially concerning non-fraudulent predictions.

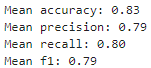
### Metrics



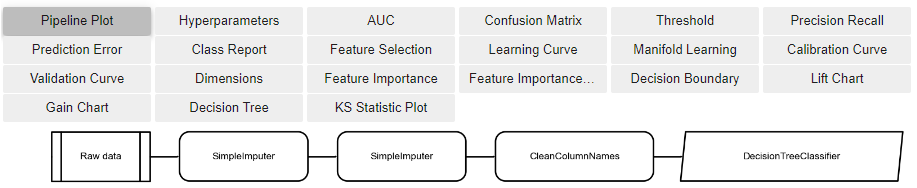
Analysing the report of tests and predictions also shows improvement:

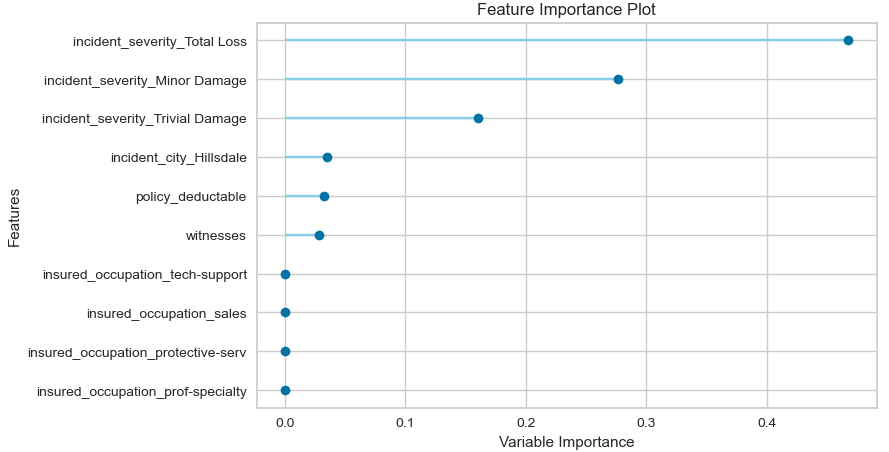


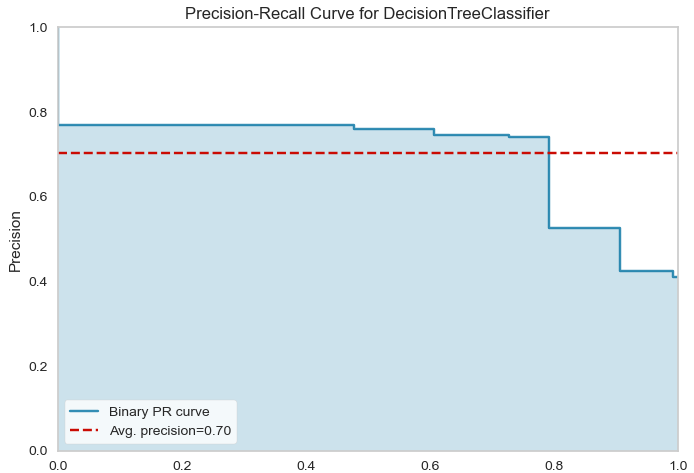
Cross-validation report results (hyperparameters tuned):

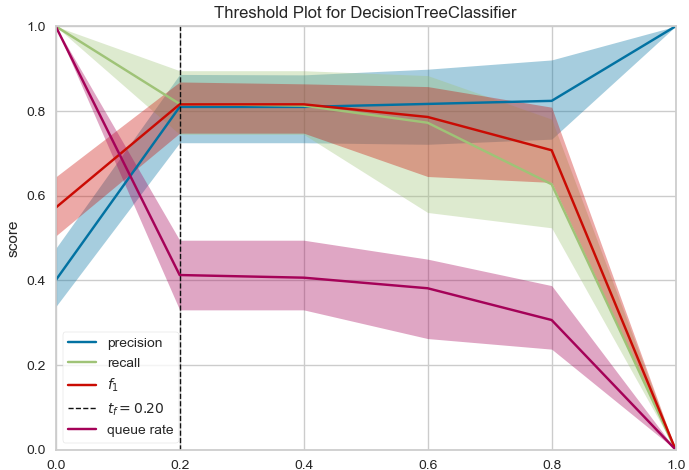


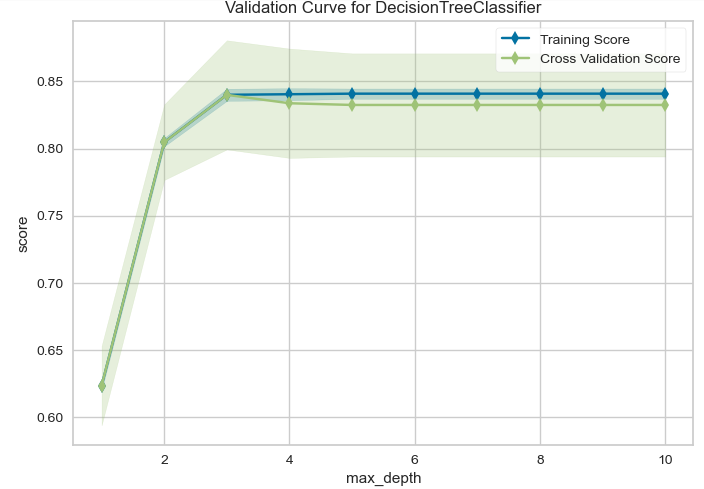
### Evaluating results using PyCaret





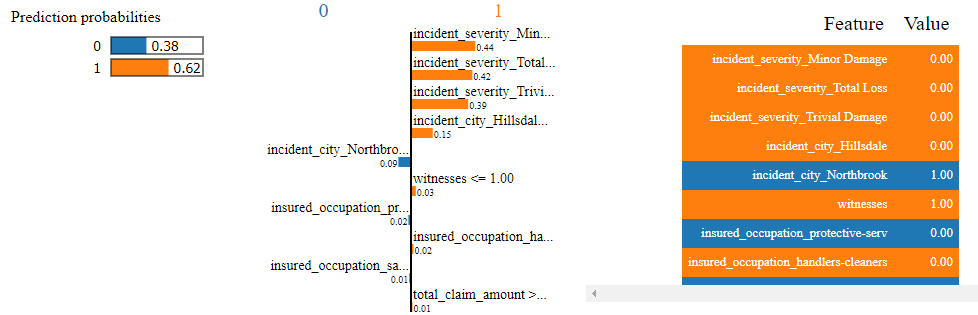




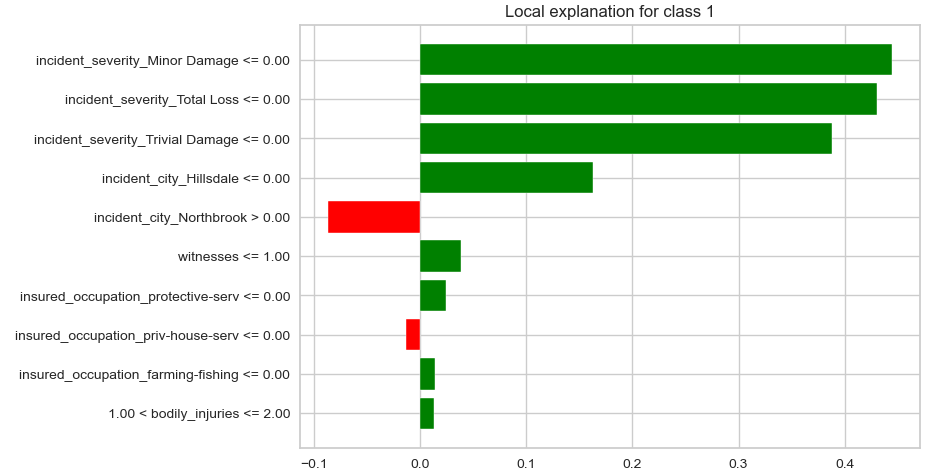


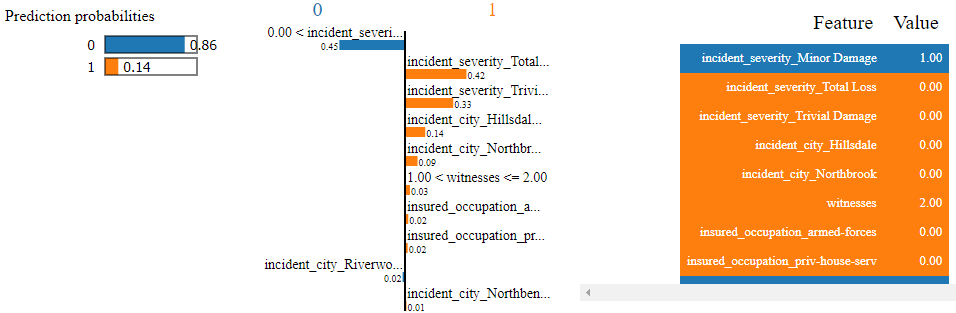
### SHAP CART

Analysing model interpretability of a fraudulent claim:

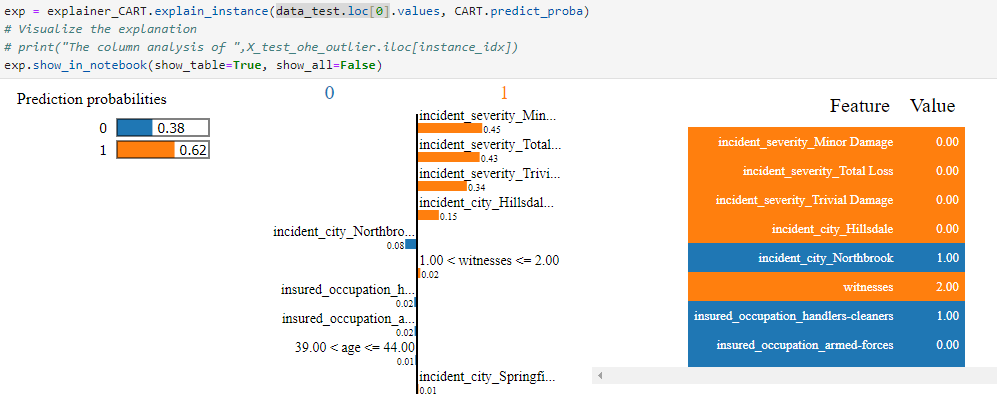


Below it illustrates the weighted features contributing to the prediction:

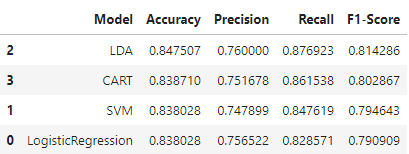
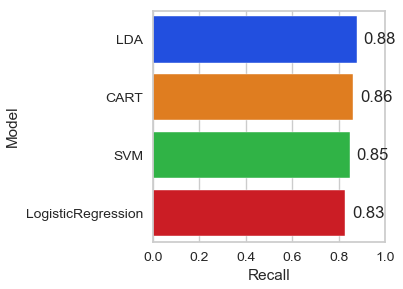




Above is notable that “Incident severity Minor Damage” has a strong weight regards the predicted class. Therefore as an example, I created a data frame with the same variables, except for “Incident severity Minor Damage”, which was changed to “0” and interestingly it is predicted as fraudulent:



# Evaluation

The following image illustrates the performance metrics on the test set for each trained model, it indicates that all the trained models yield satisfactory results. Among them, LDA demonstrates the highest scores both in Accuracy and Recall. Considering its ease of evaluation, minimal time consumption, and straightforward interpretability, LDA emerges as a promising choice for identifying fraudulent claims. 

What is interesting is the evolution of metrics in the SVM model when addressing issues such as imbalanced target and distribution using scaling methods. This demonstrates the importance of understanding the dataset at hand and the model being employed. Additionally, hyperparameter tuning was applied to find the best fit, leading to an improvement in metrics. It is also crucial to consider the risk of overfitting; however, analysing the results suggests a well-functioning model, yielding reasonable outcomes. Several metrics were considered and thoroughly analysed. Cross-validation was also employed to ensure better efficiency.

A brief overview of the trained models:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Claim | Accuracy | Precision | Recall | F1 Score | Cross Validation | Scaled | AUC |
| Logistic Regression | *Non Fraudulent* | 0.84 | 0.89 | 0.84 | 0.87 | 0.81 | No | 0.90 |
| *Reported as fraud* |  | 0.76 | 0.83 | 0.79 |  |  |
| LDA | *Non Fraudulent* | 0.85 | 0.92 | 0.83 | 0.87 | 0.82 | No | 0.85 |
| *Reported as fraud* |  | 0.76 | 0.88 | 0.81 |  |  |
| SVM | *Non Fraudulent* | 0.84 | 0.90 | 0.83 | 0.87 | 0.82 | Yes | 0.88 |
| *Reported as fraud* |  | 0.75 | 0.85 | 0.79 |  |  |
| CART | *Non Fraudulent* | 0.84 | 0.91 | 0.82 | 0.86 | 0.83 | No | 0.85 |
| *Reported as fraud* |  | 0.75 | 0.86 | 0.80 |  |  |

# Deployment

Continuous monitoring of deployed models is essential to ensure their performance and accuracy over time. Monitoring involves tracking key performance metrics, detecting drifts in data distributions, and identifying model degradation.

Establishing a feedback loop enables gathering insights from model predictions and user interactions. Feedback helps improve model performance by incorporating new data and refining model algorithms.

By following these deployment steps, organizations can effectively integrate predictive models for insurance fraud detection into their operational workflows, improving fraud detection capabilities and reducing financial losses.

# Conclusion

The dataset initially displayed a challenge due to an imbalance in the target variable, common in fraud detection tasks. Employing Synthetic Minority Over-sampling Technique (SMOTE) addressed this imbalance, ensuring a representative dataset for model training.

Through meticulous hyperparameter tuning, four models emerged, each with commendable performance metrics. This underscores the efficacy of chosen methodologies and guarantees confidence in predictive capabilities. High-scoring metrics reaffirm the suitability of machine learning techniques for identifying fraudulent claims in insurance.

Cross-validation techniques bolstered model robustness, ensuring generalizability to unseen data. This iterative validation enhances the findings' reliability and real-world applicability.

Utilizing PyCaret, SHAP and Lime for interpretability provided insights into predictive model mechanics, enhancing transparency and trustworthiness.

This study contributes significantly to fraud detection in insurance by showcasing machine learning techniques' efficacy and emphasizing interpretability's importance. Stakeholders can make informed decisions and combat fraudulent activities effectively.

The results are intended for educational purposes, establishing a foundation for future research and practical implementation. Continuous refinement and validation of predictive models are crucial to staying ahead of emerging threats in fraud detection.

GitHub link:

<https://github.com/CCT-Dublin/capstone-project-sep-2023-ft-Caroline2023190>

# Reference List

www.kaggle.com. (n.d.). *Auto Insurance Claims Fraud Detection*. [online] Available at: https://www.kaggle.com/datasets/antopravinjohnbosco/auto-insurance-claims-fraud-detection.

Anon, (2023). *3 Methods to Tune Hyperparameters in Decision Trees - Inside Learning Machines*. [online] Available at: https://insidelearningmachines.com/tune\_hyperparameters\_in\_decision\_trees/#:~:text=We%20can%20tune%20hyperparameters%20in.

Dash, S.K. (2021). *Linear Discriminant Analysis | What is Linear Discriminant Analysis*. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/08/a-brief-introduction-to-linear-discriminant-analysis/.

dash-example-index.herokuapp.com. (n.d.). *Dynamic-callback*. [online] Available at: https://dash-example-index.herokuapp.com/dynamic-callback [Accessed 20 Apr. 2024].

developer.mozilla.org. (n.d.). *Color picker tool - CSS: Cascading Style Sheets | MDN*. [online] Available at: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS\_Colors/Color\_picker\_tool.

GeeksforGeeks. (2017). *Understanding Logistic Regression*. [online] Available at: https://www.geeksforgeeks.org/understanding-logistic-regression/.

InsuranceFraud.org. (n.d.). *Fraud Stats*. [online] Available at: https://insurancefraud.org/fraud-stats/.

Navlani, A. (2019). *Python Logistic Regression Tutorial with Sklearn & Scikit*. [online] www.datacamp.com. Available at: https://www.datacamp.com/tutorial/understanding-logistic-regression-python.

plotly.com. (n.d.). *Theming*. [online] Available at: https://plotly.com/python/templates/.

Sartorius. (2020). *What Is Principal Component Analysis (PCA) and How It Is Used?* [online] Available at: https://www.sartorius.com/en/knowledge/science-snippets/what-is-principal-component-analysis-pca-and-how-it-is-used-507186#:~:text=Principal%20component%20analysis%2C%20or%20PCA.

Sharma, A. (2018). *Decrypting your Machine Learning model using LIME*. [online] Medium. Available at: https://towardsdatascience.com/decrypting-your-machine-learning-model-using-lime-5adc035109b5.

Spiceworks. (n.d.). *What Is a Support Vector Machine? Working, Types, and Examples*. [online] Available at: https://www.spiceworks.com/tech/big-data/articles/what-is-support-vector-machine/#:~:text=A%20support%20vector%20machine%20(SVM)%20is%20a%20machine%20learning%20algorithm.

valuepenguin. (2024). *Insurance Fraud Statistics*. [online] Available at: https://www.valuepenguin.com/auto-home-insurance-fraud.

www.forbes.com. (n.d.). *Insurance Fraud Statistics 2024 – Forbes Advisor*. [online] Available at: <https://www.forbes.com/advisor/insurance/fraud-statistics/>.

Gitbook.io. (2024). *Analyze | Docs*. [online] Available at: https://pycaret.gitbook.io/docs/get-started/functions/analyze [Accessed 11 May 2024].