Mehrdad Baradaran

99222020

Second Assignment

Machine Learning

vehicle Insurance Claim Fraud Detection dataset

Classification problems

Outline

- **Introducing Dataset**
- **Data analysis**
- **Using different models**
    - **Logistic Regression**
    - **SVM**
    - **Decision-Trees**
    - **Random Forest**
    - **KNN**
    - **Naive Bayes**
- **Use stratified cross-validation**
- **Check whether this dataset is imbalanced or not**
- **Try to boost the performance of the SVM and Random Forest**
- **Use XGBoost model**

What is vehicle Insurance Claim Fraud Detection dataset?

Vehicle insurance fraud involves conspiring to make false or exaggerated claims involving property damage or personal injuries following an accident. Some common examples include staged accidents where fraudsters deliberately “arrange” for accidents to occur; the use of phantom passengers where people who were not even at the scene of the accident claim to have suffered grievous injury, and make false personal injury claims where personal injuries are grossly exaggerated.

Abstract:

Here we are dealing with a classification problem that we have to work with Vehicle Insurance Claim Fraud Detection dataset.

At first, we talk about the data itself, and by doing a series of analyzes and drawing graphs, we try to find generalities about the variables and their relationship, so that if we need to change the data, add or even delete them. Next, we will talk about encoding and converting categorical features to numeric.

Now we can apply different classification models on it and compare the results according to accuracy and Roc curve.

Then we discuss how we can improve the model and balance the data.

Finally, we will talk about how the XGBoost model has performed compared to other models.

Introduction:

In this task, the most important question is, which models and features will give me the best predictions?

We used different methods to answer this question.

Both in preprocessing and in model_selection and even in feature_selection.

Well, at first, we load the data set and see its specifications, how many features it has or how many null data it has, and the type of each feature and...

Next, we analyzed the data using the following methods:

- Histogram plot
- barplot
- boxplot
- countplot
- pieplot
- scatterplot

In addition to graphs, we also used statistical methods to see what effect each feature has on the label or even on other features.

Is its distribution a normal distribution or does it have outliers?

The presence or absence of features has a positive effect. What is the correlation between the variables and...

The story does not end there.

Now we need to do things so that the model can start training with these data.

Because a series of features are CATEGORICAL and models can only work with NUMERIC data, so we have to expand these features using the encoding method or any other method. Next, we divide the data into two groups, train and test.

Now we can define the models we want to use to fit the data.

Here we used logistic regression models and SVM etc.

After fitting the data, the models should be compared, but unfortunately there is a problem, and that is that our data is very unbalanced, because 95% of the labels are no and the rest are yes. As a result, we use ROC curve to check tpr and fpr together in each model.

Of course, precision_recall curve can be used for further checks.

Next, by using cross validation, we will try to increase the performance of the models and check whether the imbalance of the data can be solved or not.

After checking the charts and results, we notice a little positive effect in the process.

So, in the following, we use two methods of undersampling and oversampling.

To check the effect of each.

After the implementation and analysis of the graphs, it can be concluded that it has a greater impact, although it can be logical because the data with the yes label constitutes only 5% of the total data.

Of course, we do these methods in two ways

- Before splitting the data
- After splitting the data

And we will check the more detailed results in the methods section.

Now we are trying to boost the SVM and Random forest models by doing things like hyperparameters tuning and different preprocessing and even feature engineering. For the preprocessing part, we used scaler methods and reported the results with scalers such as minmaxscaler and robustscaler.

For tuning, we used a ready-made method and library that outputs the best model and parameters on all model states with different parameters. You thought right! It is very time consuming algorithm but it is very valuable in some cases.

But we thought that maybe by doing feature selection, we could get a better result, since in the correlation matrix there were fewer features related to each other.

And for this we use SelectKBest, f_classif. And finally, we combine all these together to see the result.

To finish the work, we will also try the xgboost model to see to what extent the work we did can compete with the powerful xgboost model.

Methods :

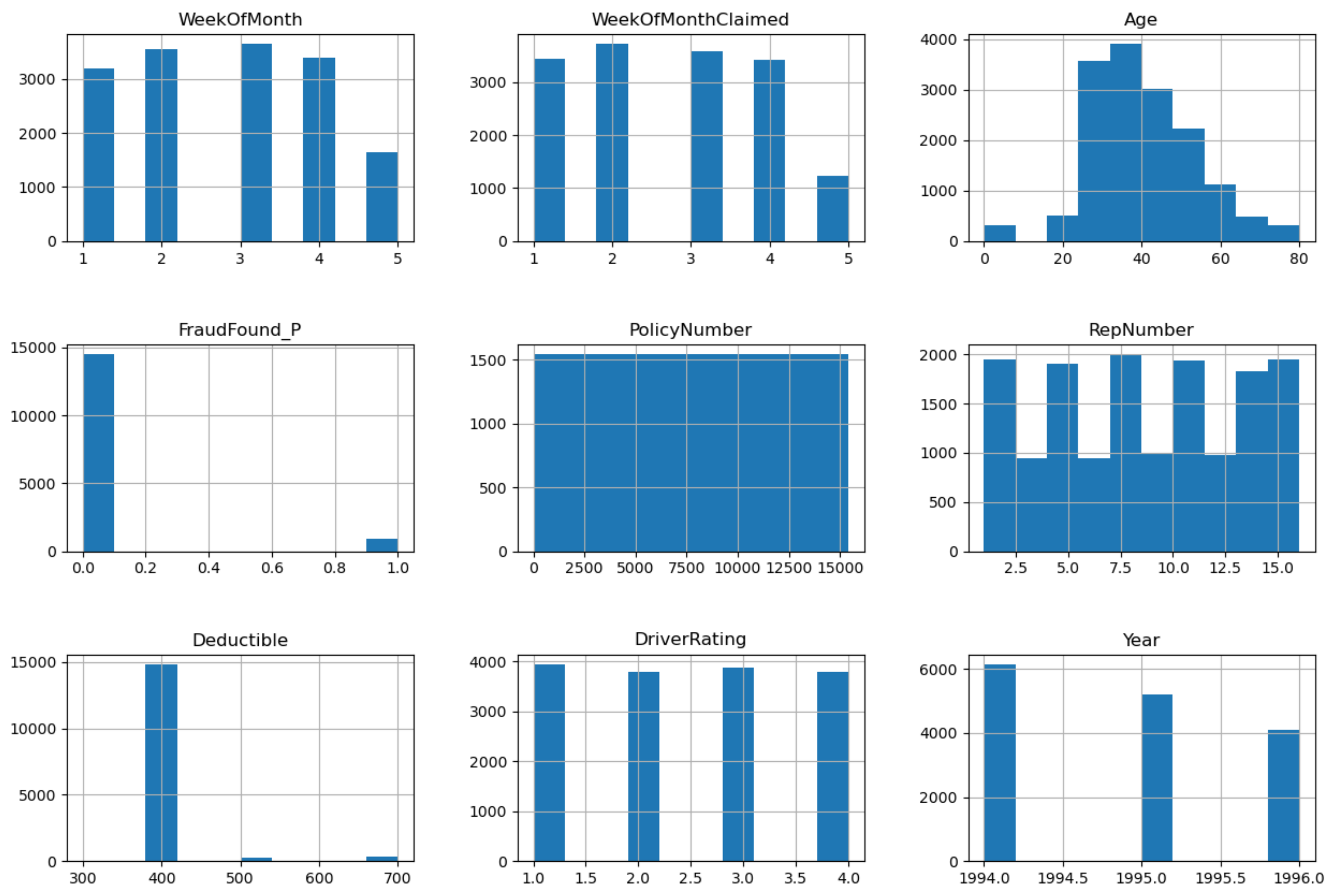
Well, in this part, we will report each part of the implementation along with the results.

First of all, we must import the required libraries:

- Pandas
- Numpy
- Matplot
- Sklearn
- Seaborn, etc.

Download the required dataset from the site and open it in dataframe format. We check the dimensions and number of samples and their features.

We also get the number of null data. And we count how many categorical and numerical features we have and which features they are. Now we do some EDA on the data.

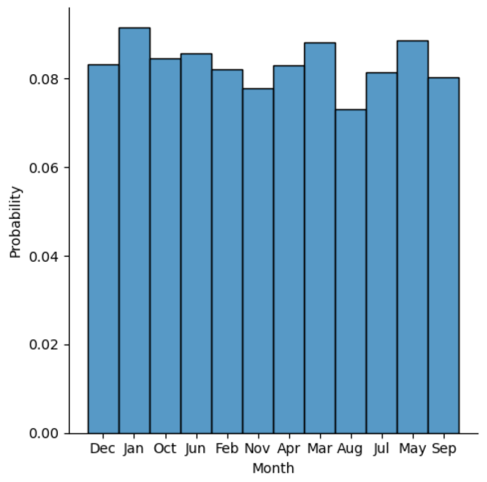


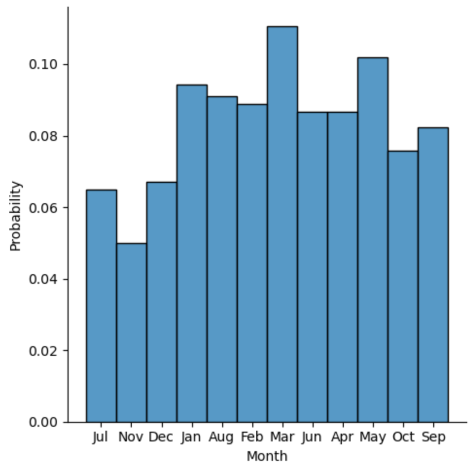
The following can be seen from the following distribution charts:

1. Policy numbers looks randomly distributed, may be it is just a serial number
2. Most of the deductibles are in 3 categories. Most commonly involved in claims is 400
3. Driver Rating are somewhat equally distributed
4. Fraudulent claims have very small proportion
5. Only 3 years of cars showing in the data and year 1994 (older cars) have more involvement in accidents.

Since this section is very large and long, we will check some features and data and the rest are the same.

For the first feature, which is month, we first display its distribution and calculate the average number of accidents in each month, and finally find out which months have the most fake insurance requests.

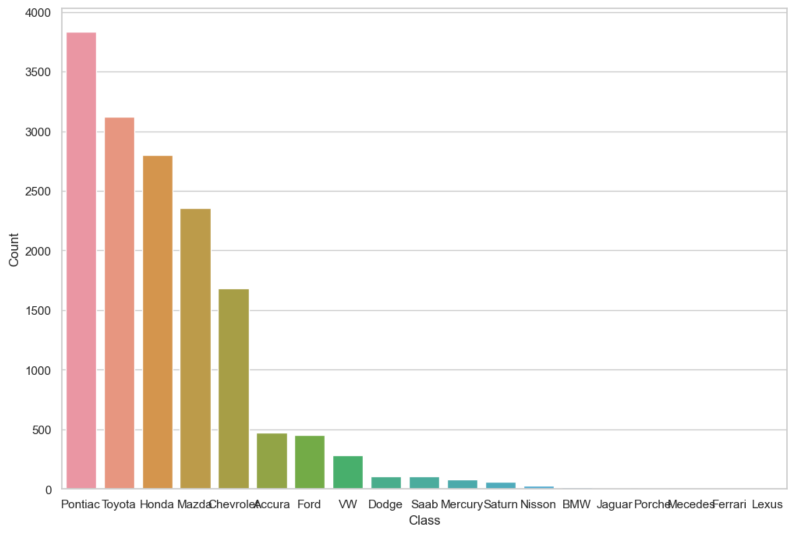




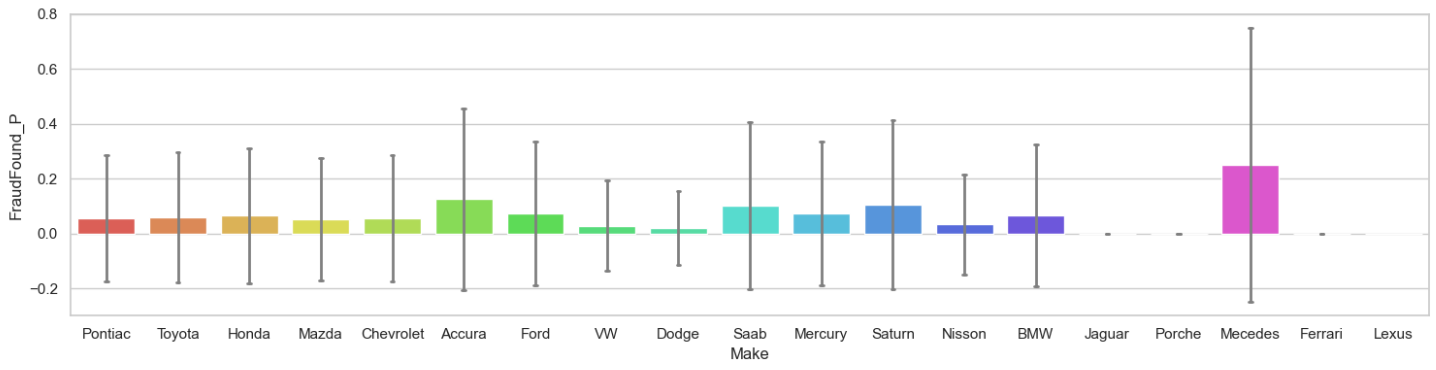
And we conclude that Amongst fraudulent cases months of March and May has relatively higher probability.

The next feature that we check is the car model. First, we calculate which cars have the most requests.

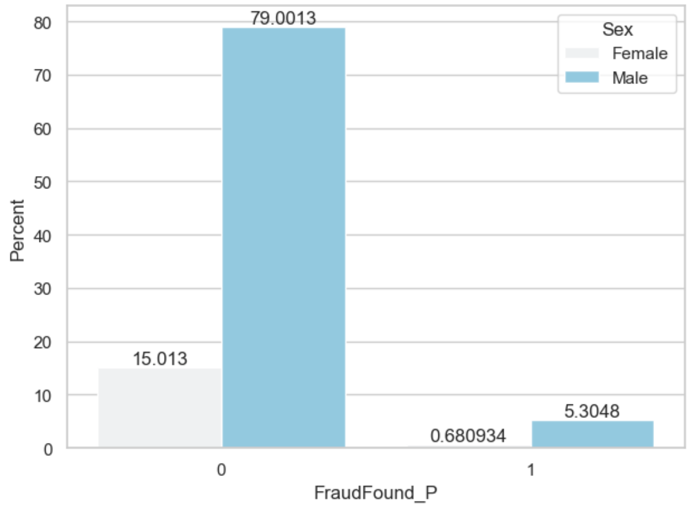
And in the following, we show the distribution of each model in fraud claims.



Surprisingly the highest probablity of fraudulent transactions is in high end cars like Accura and Mercedes, most likely due to higher incentive for frauds as the cars being costlier.

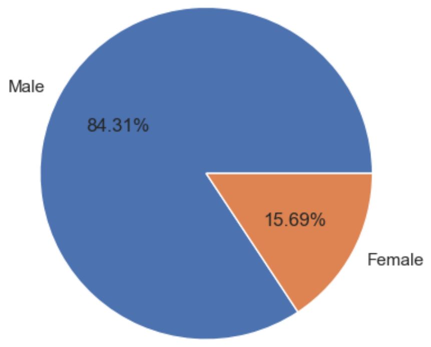


Contribution of sex type is the next feature that is also very important. For each gender, we show the percentage of men and women who applied for fraud and how many did not. In the following, we displayed the total number of fraud and non-fraud requests.



And we also displayed this issue in the form of a circular diagram for visual representation and understanding, and after analysis, we concluded that:

Amongst the total claims males contribute 84.03% for non-fraudulent transactions but contribute 88.62% of fraudulent transactions. Males are more likely to submit fraudulent claims

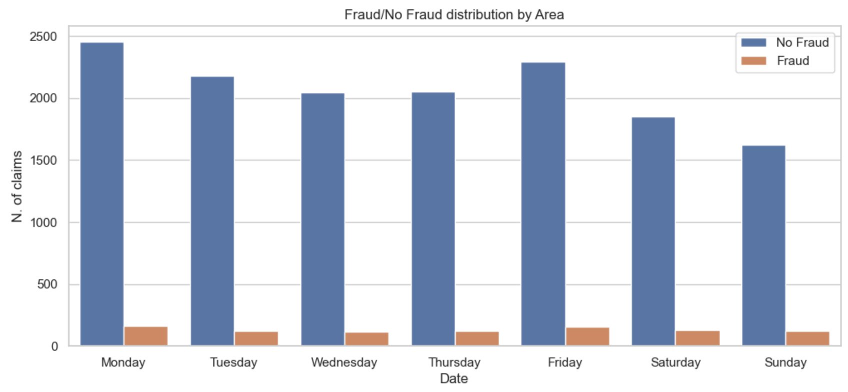


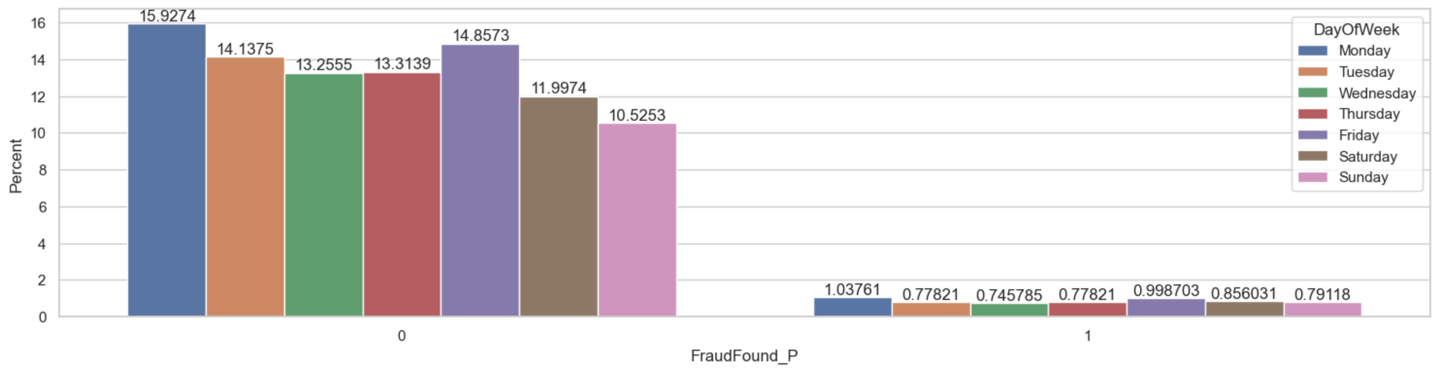
The last feature we check is day of week.

We examined many features in the code file.

For each day of the week, we counted how many of the requests that existed on that day were fraud or not.

And we showed the ratio of these in the form of different plots.

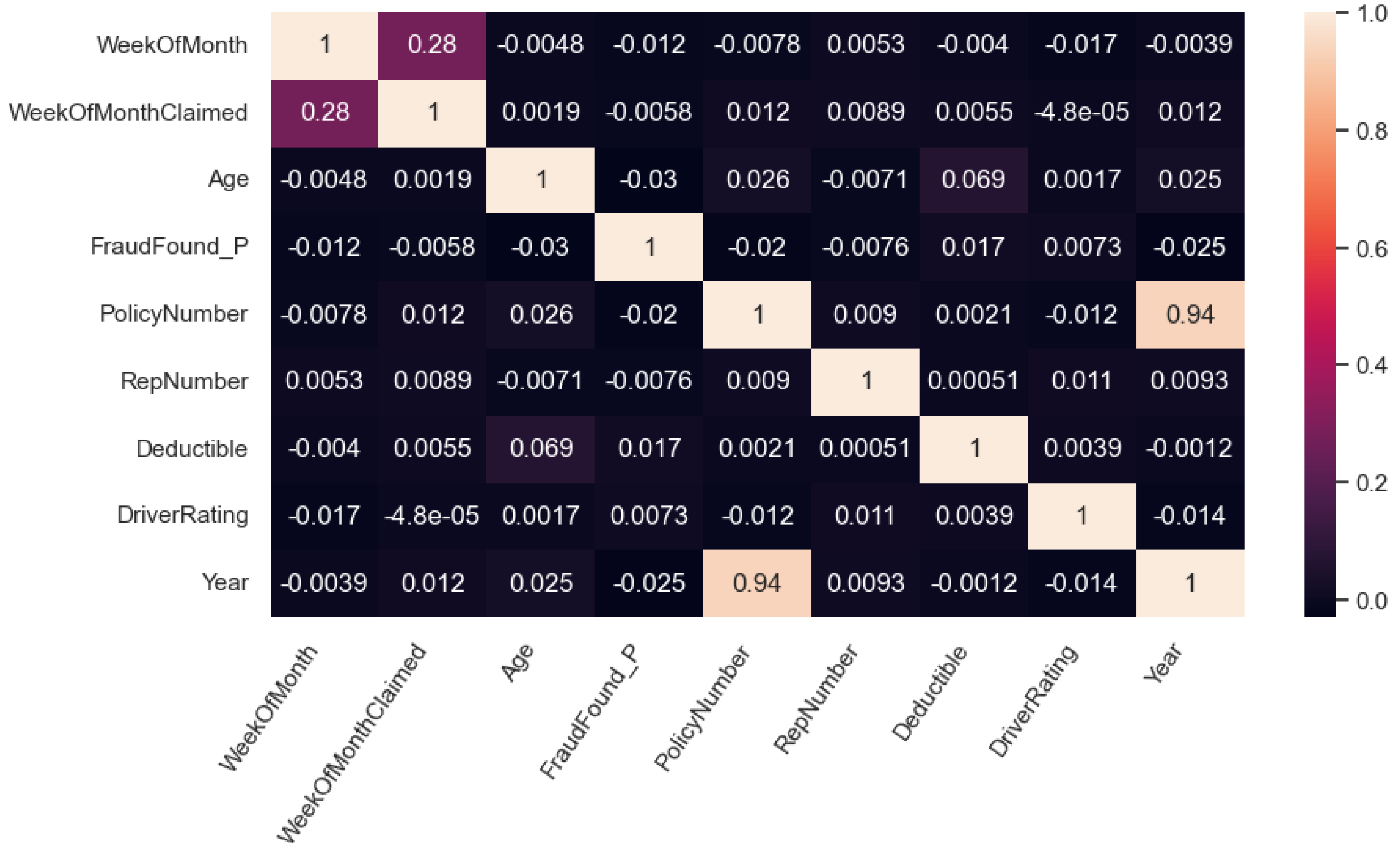




And finally we concluded that:

Percentage of fraudulent transactions is higher on Monday and Friday. Similarly, the number of claims is also higher for Monday and Friday. Probably due to rushing to go to work on Monday or coming back Friday evening. But we do not have any time data.

At the end of this section, we draw the correlation matrix of the features.

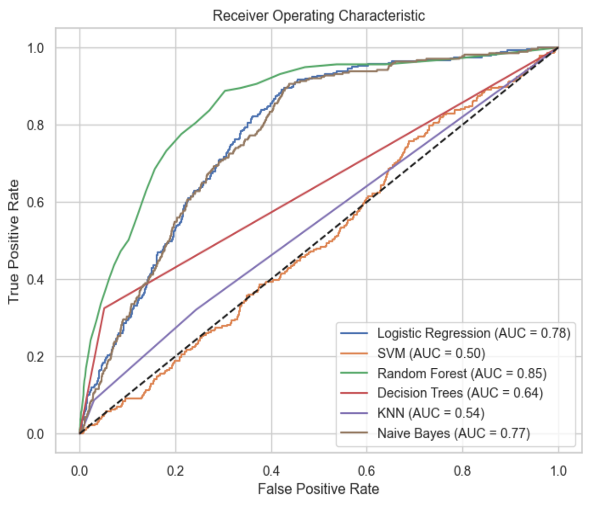


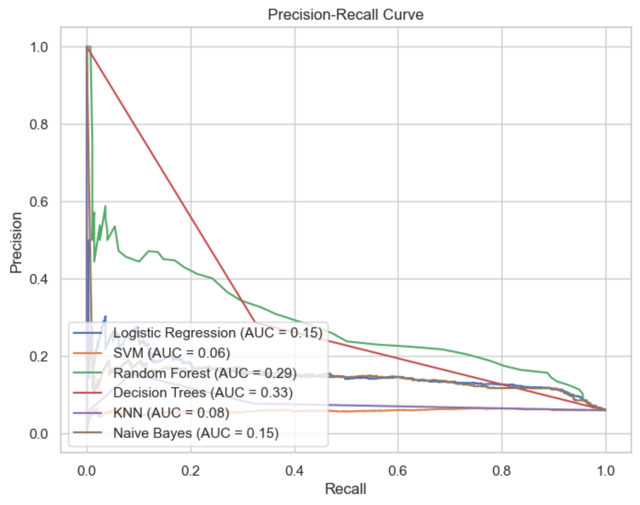
Now it's time to make our dataset ready to use so that our models can use them.

First of all, we need to see if we need to remove the feature. We removed the policy type after analyzing the graphs and converted the rest of the qualitative features into quantitative features using the OrdinalEncoder and OneHotEncoder functions. This can be done in different ways. so that 117 features can be separated. We converted to 57 separate features. And we used the following models:

- Logistic Regression
- SVM
- Decision-Trees
- Random Forest
- KNN
- Naive Bayes

After fitting the data, we display the results in ROC Curve format.

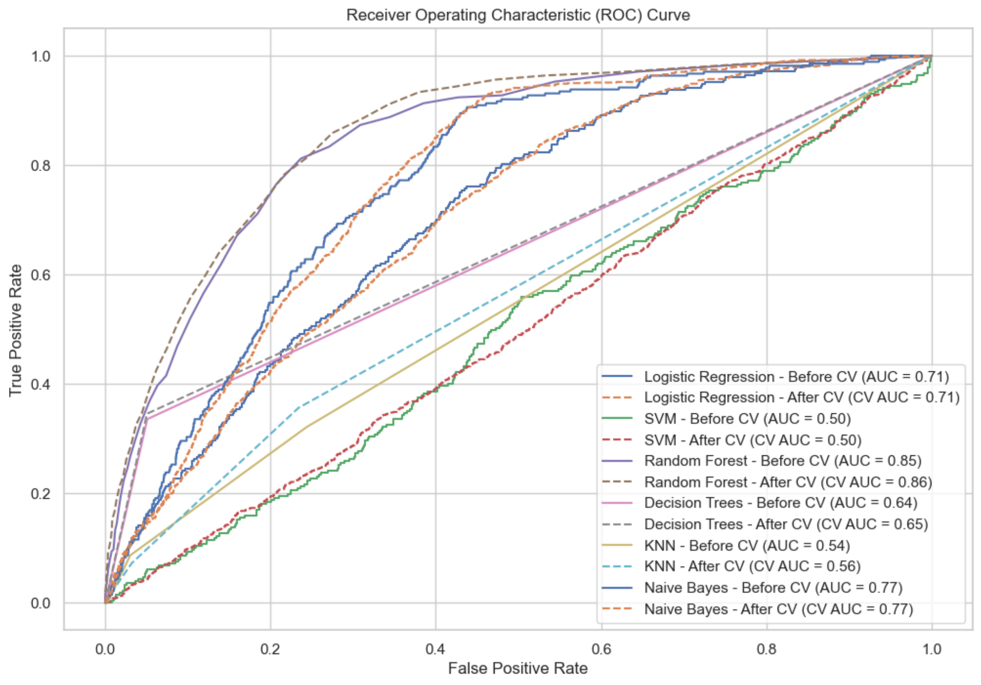




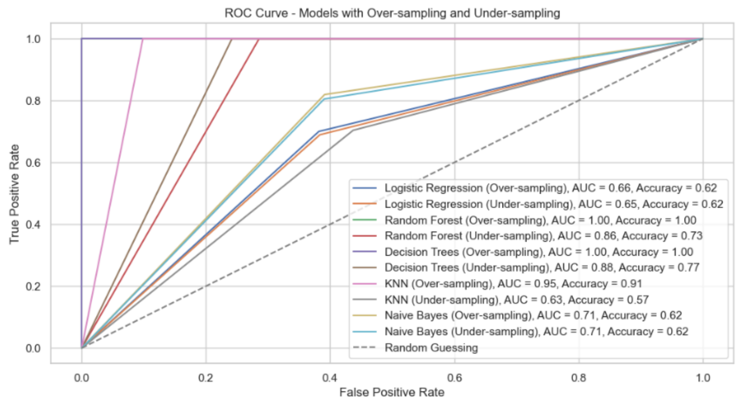
Among the following models, it can be seen that the Random Forest model performed better than the others and the SVM model was the worst model. Maybe this is because the number of our features is too high and the SVM model does not perform well on high-dimensional data.

Next, we use stratified cross-validation so that we can get better results.

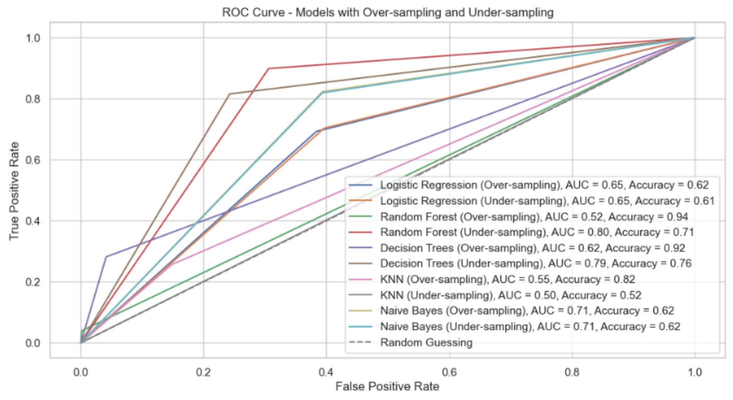
And we can see the results in the graph.



Since our data is very unbalanced, we use undersampling and oversampling methods, but the first thing that is important is that if we fit the models before splitting the data, it will be overfit, and this is because the model All the data is noble and he sees them, in fact he keeps the data until he learns them.

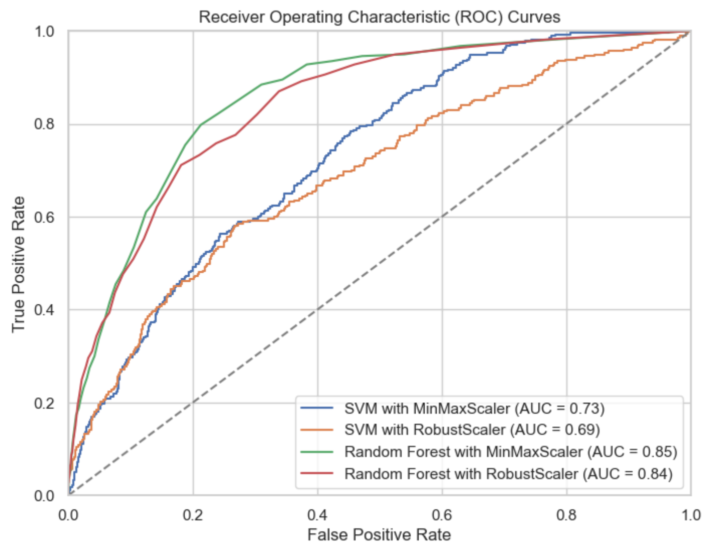


As it is clear from the diagram, if we split the data before sampling, this problem will not arise.



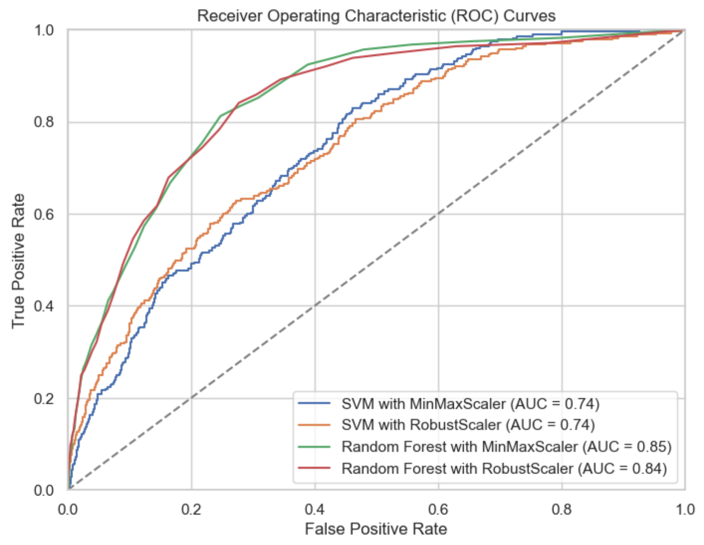
But what is clear from the graph is that cross validation works better than sampling.

In the future, we will try to make our models better and better. To begin with, we use scaler functions such as robust scaler and minmaxscaler, and it can be well understood from the results that both models show better performance.



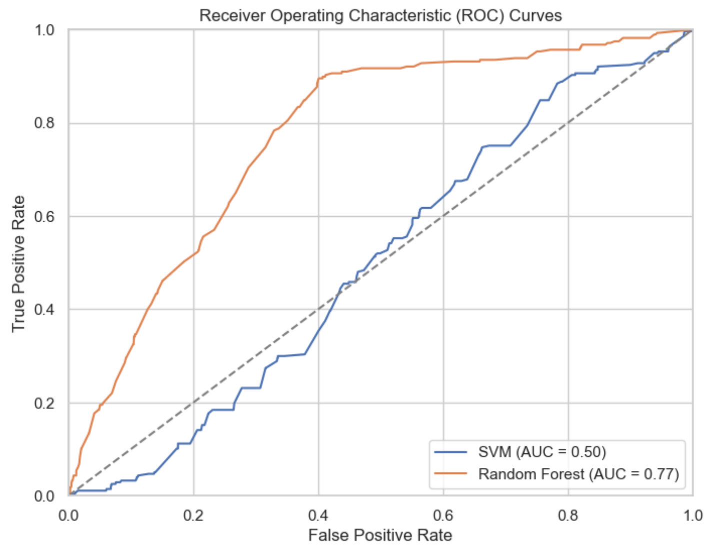
We try to use feature engineering, for example, we add the square root of the variables.

And in the rest of the work, we will use the same scaler functions to see the result of combining these two.



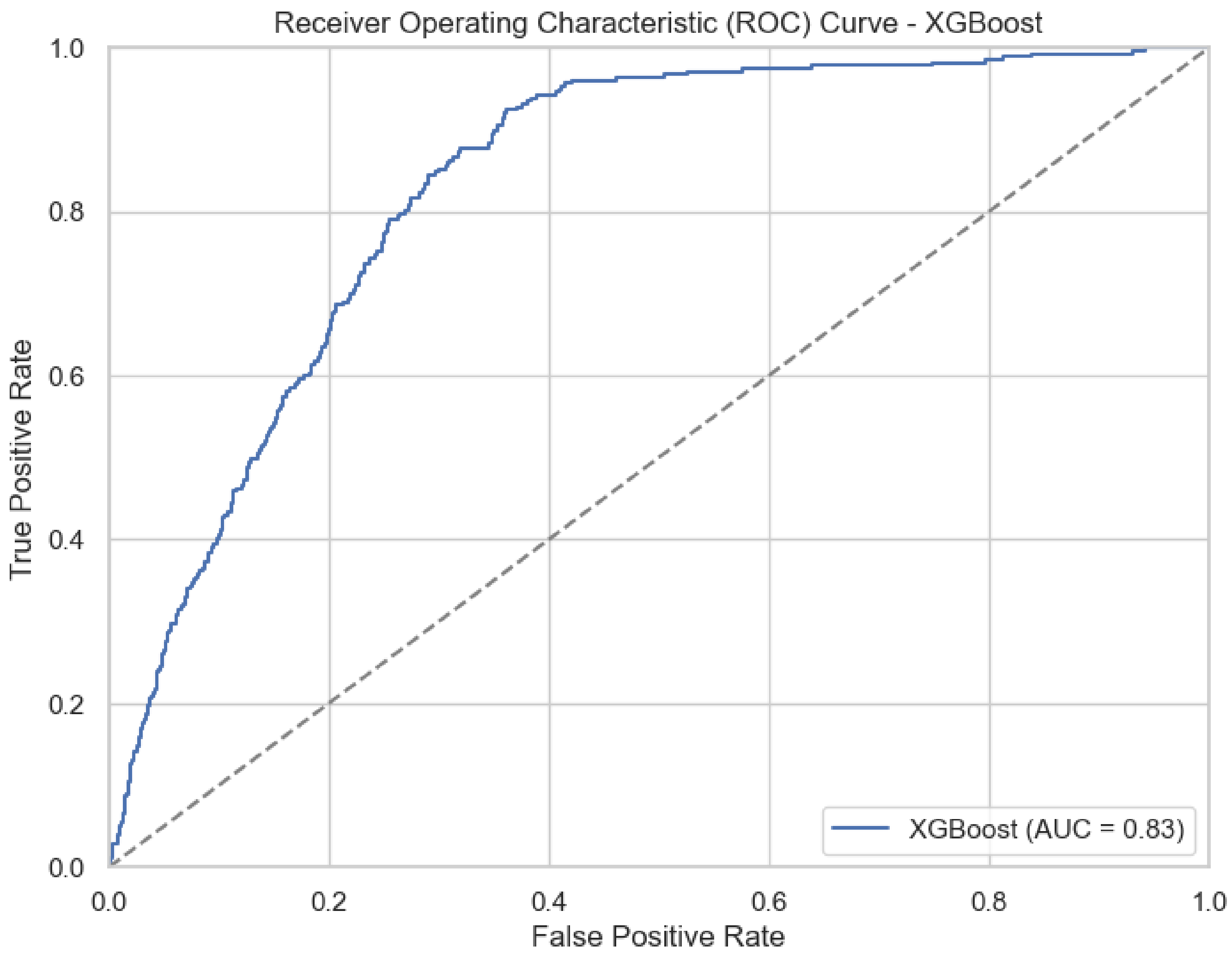
It is clearly visible that the result has improved.

We also did several other things, such as using gridsearch and feature selection algorithms, and we checked their results, which obviously did not show us better results than the previous methods. So we decided to combine all these to see the final result.



The SVM model is very close to the random mode, which indicates that its result is bad, but Random forest still has a good result.

Finally, we tried to use the strong xgboost model to see how the methods and work we did are comparable to this model. As it is clear from the results, we were able to obtain a model during this programming and classification process that It works better than the xgboost model.


Receiver Operating Characteristic (ROC) Curve - XGBoost
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
XGBoost (AUC = 0.83)

Conclusion:

In the process of solving the Vehicle Insurance Claim Fraud Detection classification problem, we applied several methods and techniques to improve the performance of our models. Here is a summary of the approaches we used:

- Exploratory Data Analysis (EDA): We performed an initial analysis of the dataset to understand its structure, identify any missing values or outliers, and gain insights into the distribution of features.

- Preprocessing: We handled missing values by either removing rows with missing data or imputing the missing values using appropriate strategies. We also encoded categorical features using techniques such as one-hot encoding or label encoding to convert them into numerical representations.

- Model Selection: We trained several classification models using Scikit-Learn, including Logistic Regression, Support Vector Machines (SVM), Random Forest, Decision Trees, K-Nearest Neighbors (KNN), Naive Bayes, and Gradient Boosting Classifier. These models provided a range of algorithms to compare and evaluate their performance.

- Model Evaluation: We evaluated the models using various metrics such as accuracy, ROC curves, and AUC scores. These metrics helped us assess the models' performance in distinguishing between fraudulent and non-fraudulent vehicle insurance claim applications.

- Cross-Validation: We utilized stratified cross-validation to ensure robust evaluation of the models and mitigate the risk of overfitting. This technique partitions the data into multiple subsets, ensuring that each subset maintains the same class distribution as the original dataset.

- Imbalanced Data Handling: Since the dataset had an imbalanced class distribution, we employed techniques like oversampling and undersampling to address this issue. Oversampling involved replicating instances of the minority class to balance the class distribution, while undersampling involved removing instances from the majority class.

- Hyperparameter Tuning: We performed hyperparameter tuning using techniques like GridSearchCV to search for the best combination of hyperparameters for the models. This helped optimize the models' performance and improve their ability to generalize to new data.

- Feature Engineering: We explored feature engineering techniques such as creating new features, calculating ratios or indices, and transforming existing features to enhance the predictive power of the models. These engineered features provided additional information and potential insights for better classification.

- Model Boosting: We applied boosting techniques such as XGBoost and Gradient Boosting Classifier to further enhance the performance of the models. These algorithms focus on sequentially improving the weak models by giving more weight to the misclassified instances.

Overall, through a combination of exploratory data analysis, preprocessing, model selection, evaluation, cross-validation, handling imbalanced data, hyperparameter tuning, feature engineering, and model boosting, we aimed to build robust classification models for detecting fraudulent vehicle insurance claims. The comprehensive approach allowed us to improve the models' accuracy, ROC curves, and overall performance, enabling more effective identification of potential fraud instances.