**Insurance Claims- Fraud Detection**

This repo contains the Customer Churn Analysis project as part of my data science portfolio. This project is completed as part of the Datatrained Evaluation Project. Evaluation metric of the project is accuracy i.e. percentage of loan approval that is correctly predicted. After trying and testing 4 different algorithms, the best accuracy on the leaderboard is achieved by Logistic Regression (0.8142),Followed by Random Forest (0.7771). followed by SVC (0.7714) and Decision Tree (0.7628).

This project covers the whole process from problem statement to model development and evaluation:

1. Problem Definition.

2. Data Analysis.

3. EDA Concluding Remark.

4. Pre-Processing Pipeline.

5. Building Machine Learning Models.

6. Concluding Remarks.

**Problem Definition**

**Problem Statement:**

**Business case:**  
Insurance fraud is a huge problem in the industry. It's difficult to identify fraud claims. Machine Learning is in a unique position to help the Auto Insurance industry with this problem.

In this project, you are provided a dataset which has the details of the insurance policy along with the customer details. It also has the details of the accident on the basis of which the claims have been made.

In this example, you will be working with some auto insurance data to demonstrate how you can create a predictive model that predicts if an insurance claim is fraudulent or not.

**Problem Statement is Translated into Data Science Problem:**

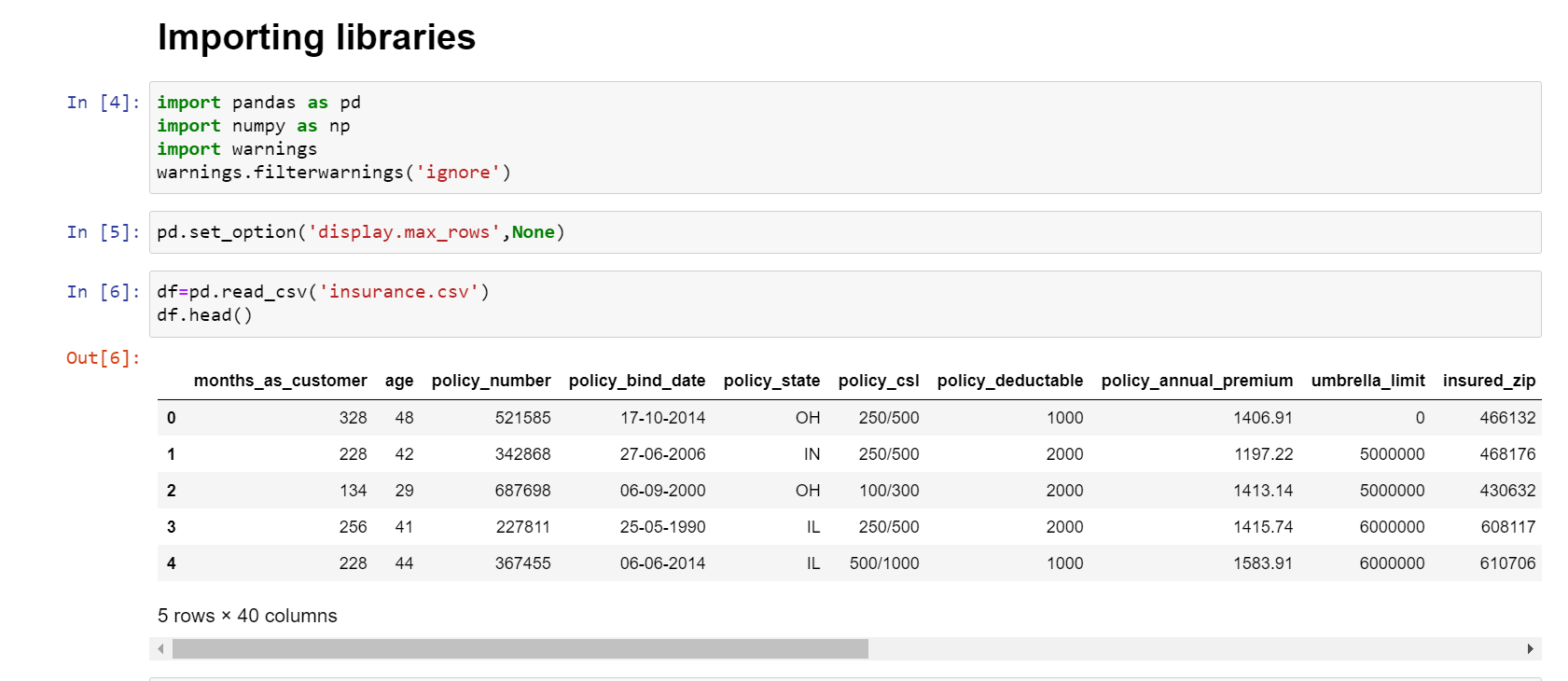
This is a classification problem where we have to predict whether a fraud\_reported will be approved or not. Specifically, it is a binary classification problem where we have to predict either one of the two classes given i.e. approved (Y) or not approved (N). Another way to frame the problem is to predict whether the fraud\_reported will likely to default or not, if it is likely to default, then the fraud\_reported would not be approved, and vice versa. The dependent variable or target variable is the fraud\_reported, while the rest are independent variable or features. We need to develop a model using the features to predict the target variable.

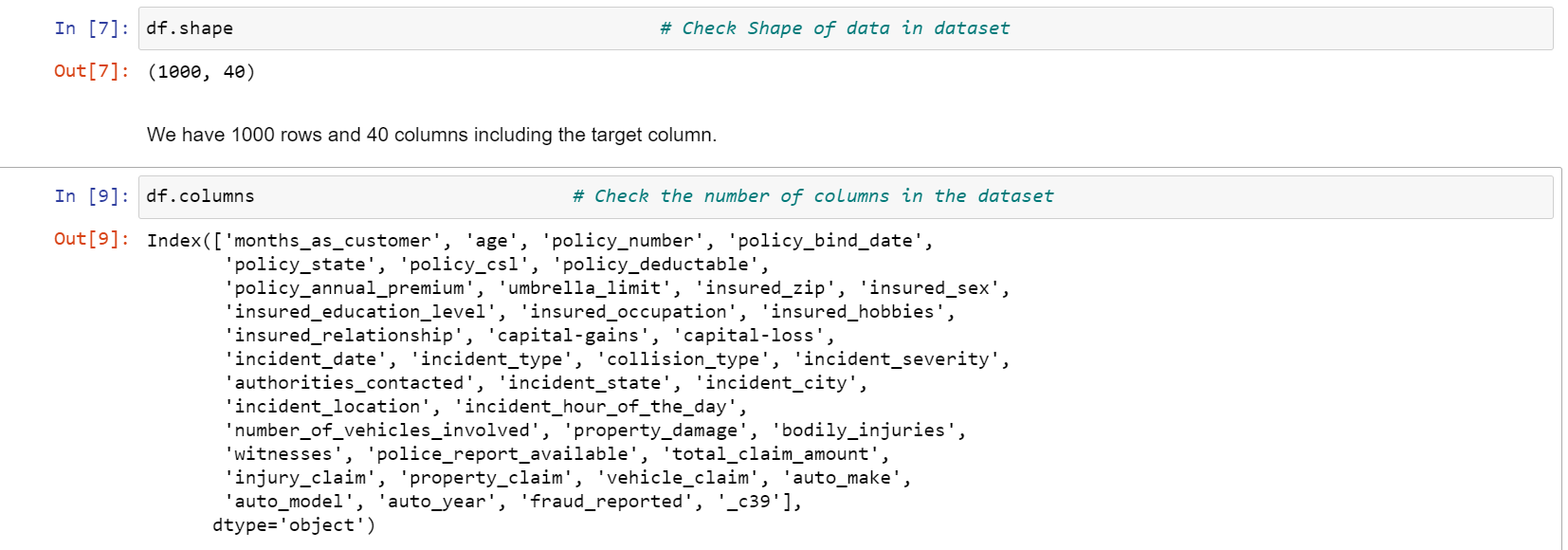
**Data Analysis**

The data have already been provided by Datatrained Institute. The training set will be used for training the model, i.e. our model will learn from this data. It contains all the independent variables and the target variable. The test set contains all the independent variables, but not the target variable. We will apply the model to predict the target variable for the dataset. There are 40 columns of features and 1000 rows of records in the Dataset.

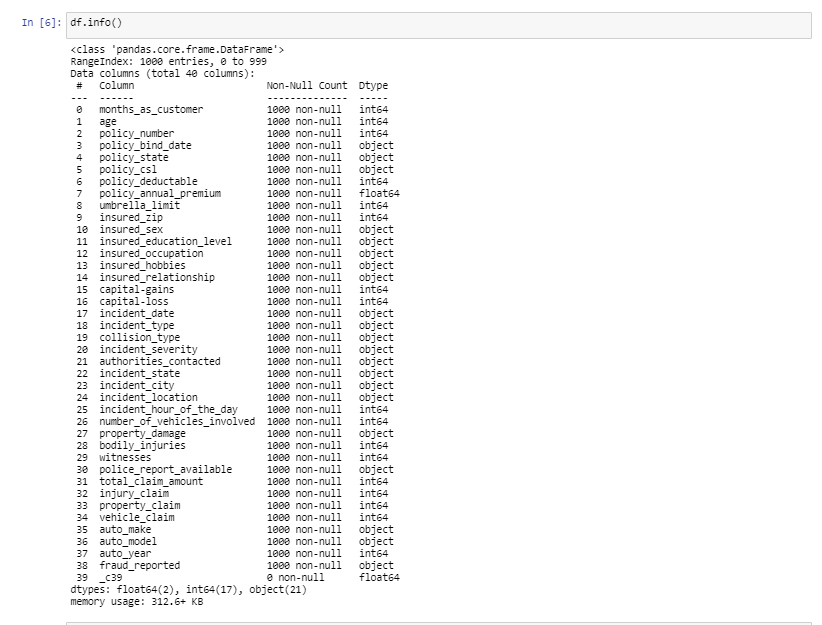
# Exploratory Data Analysis (EDA) Concluding Remark

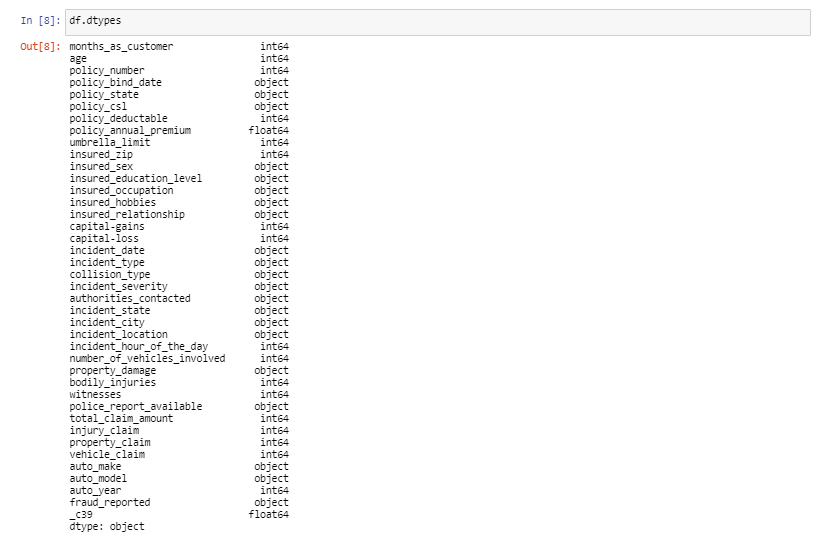
# We will use Python to explore the data in order to gain a better understanding of the features and target variable. We will also analyze the data to summarize their main characteristics, using various visualization techniques.





**Note:**There are 40 columns of features and 1000 rows of records in the Dataset. The features are similar in the dataset except the fraud\_reported. We will predict the fraud\_reeported using the model built using the dataset.





**Note:**There are 3 data types in the data

* object: Object format means variables are categorical. Categorical variables in our dataset are: policy\_bind\_date, policy\_state, policy\_csl, insured\_sex,insured\_education\_level,insured\_occupation,insured\_hobbies,insured\_relationship,incident\_date,incident\_date,incident\_type,collision\_type,incident\_severity,authorities\_contacted,incident\_state,incident\_city,incident\_location,property\_damage,police\_report\_available,auto\_make,auto\_model,fraud\_reported.
* int64: It represents the integer variables. Months\_as\_customer,age,policy\_number,policy\_deductable,umbrella\_limit,insured\_zip,capital-gains,capital-loss,bodily\_injuries,witnesses,total\_claim\_amount,injury\_claim,property\_claim,vehicle\_claim,autp\_year is of this format.
* float64: It represents the variable which have some decimal values involved. They are also numerical variables. Numerical variables in our dataset are: policy\_annual\_premium,\_c39.

**Why Data Types are important?**

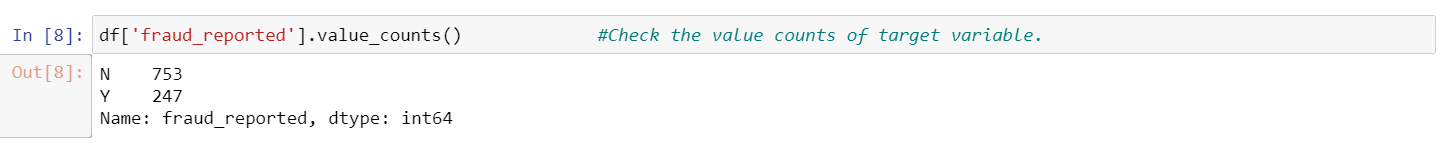
Datatypes are an important concept because statistical methods can only be used with certain data types. You have to analyze continuous data differently than categorical data otherwise it would result in a wrong analysis. Therefore knowing the types of data you are dealing with, enables you to choose the correct method of analysis.

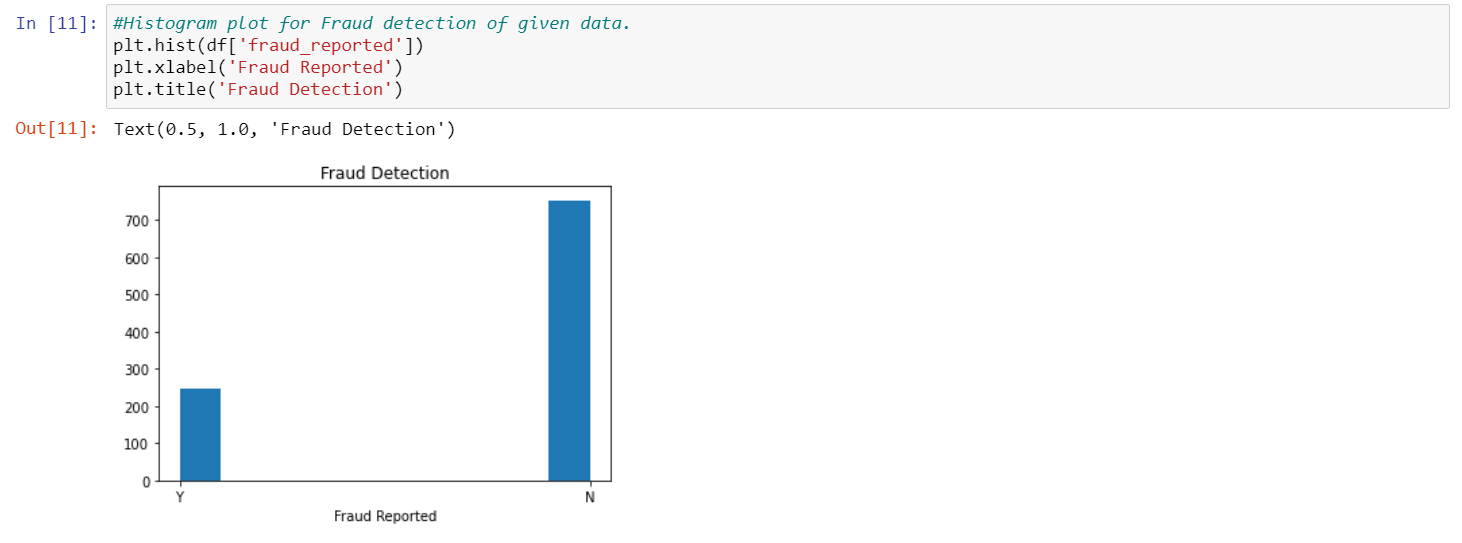
## Univariate analysis

Univariate analysis is when we analyze each variable individually. For categorical features we can use frequency table or bar plots which will calculate the number of each category in a particular variable. For numerical features, a histogram or a box-plot can be used to look at the distribution of the variable. With a histogram, you can check the central tendency, variability, modality, and kurtosis of a distribution. Note that a histogram can’t show you if you have any outliers. This is why we also use box-plots.

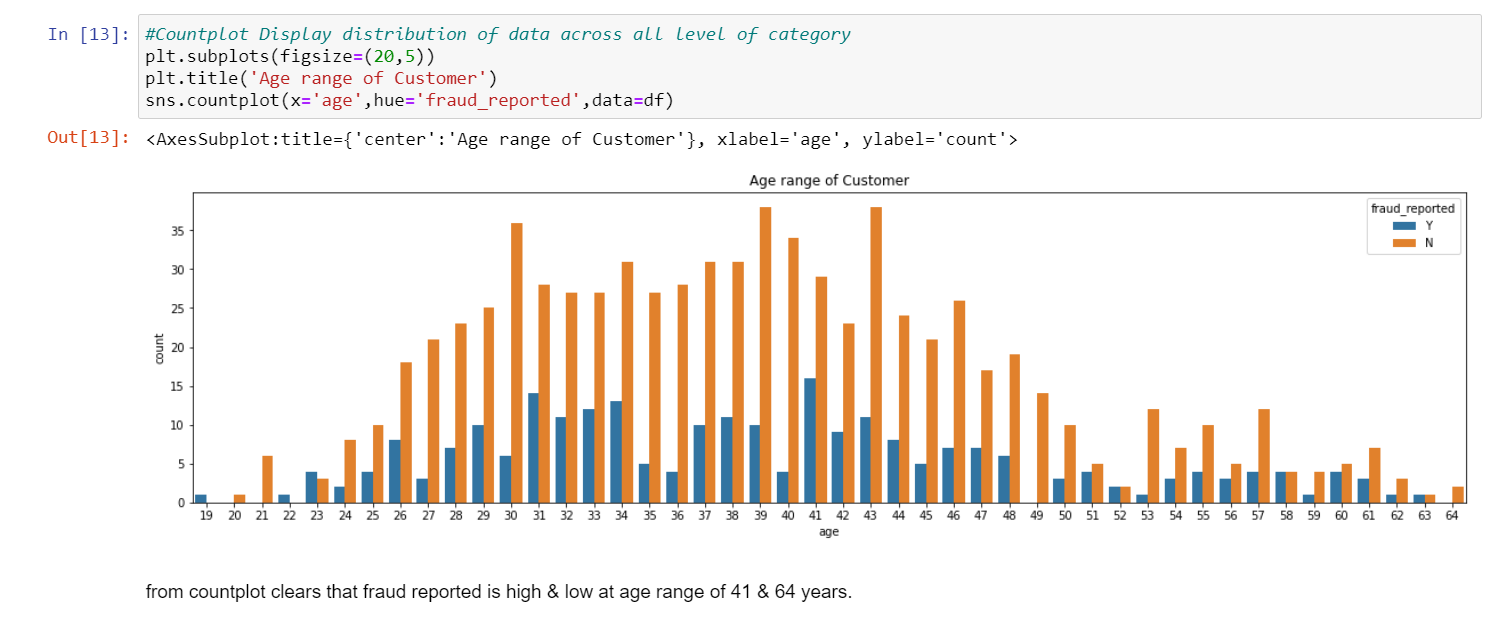
### Target Variable (Categorical)

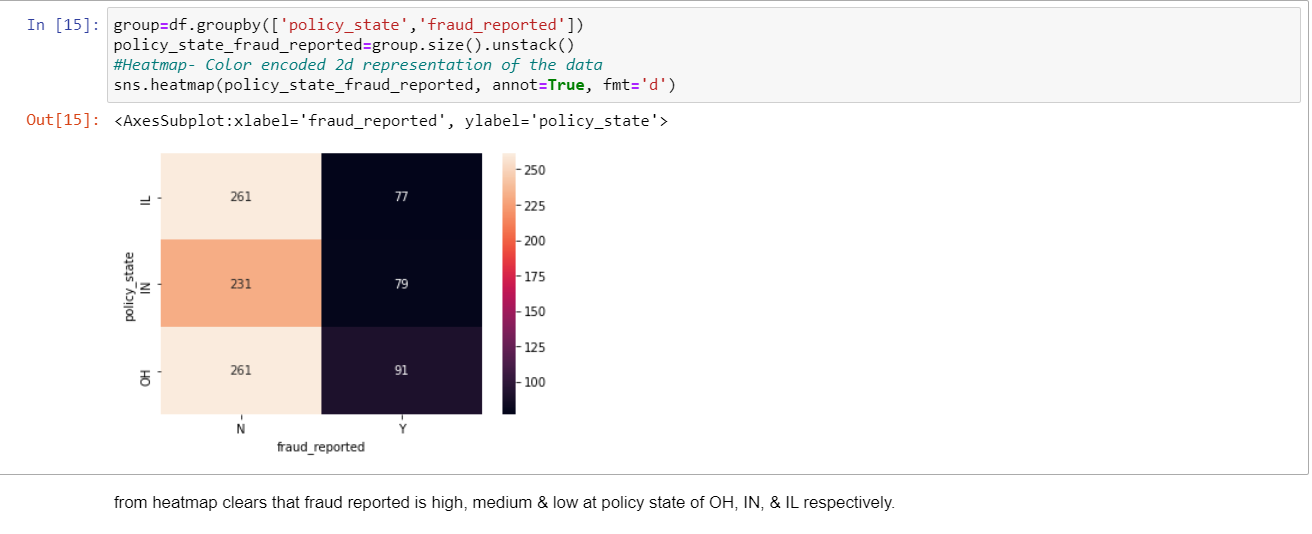
We will first look at the target variable, i.e., fraud\_reported. As it is a categorical variable, let us look at its frequency table, percentage distribution and bar plot.

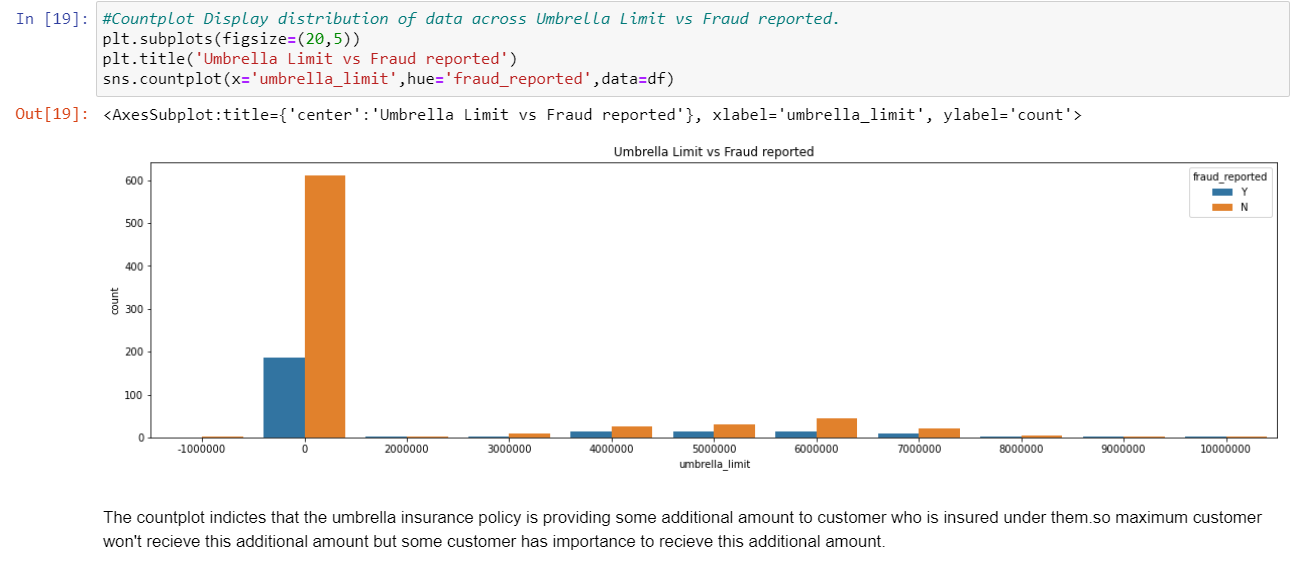


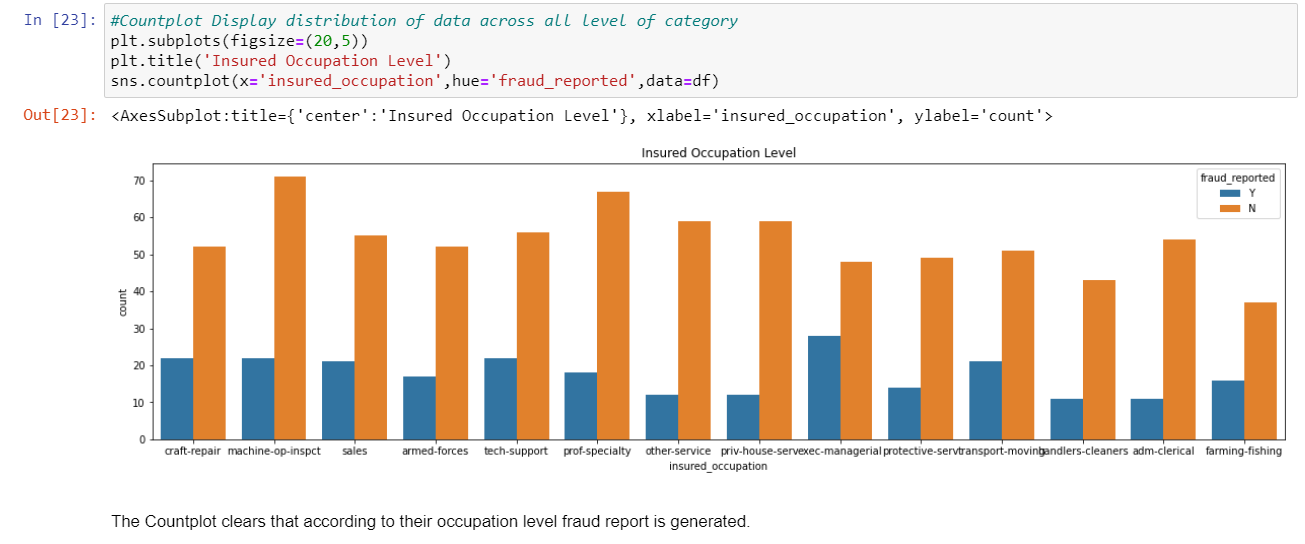


**Note:**The fraud\_reported of 753 (around 75%) NO of 247 was approved. There is no imbalanced classes issue in this dataset, thus accuracy as an evaluation metric should be appropriate. On the other hand, if there are imbalanced or skewed classes, then we might need to use precision and recall as evaluation metrics.

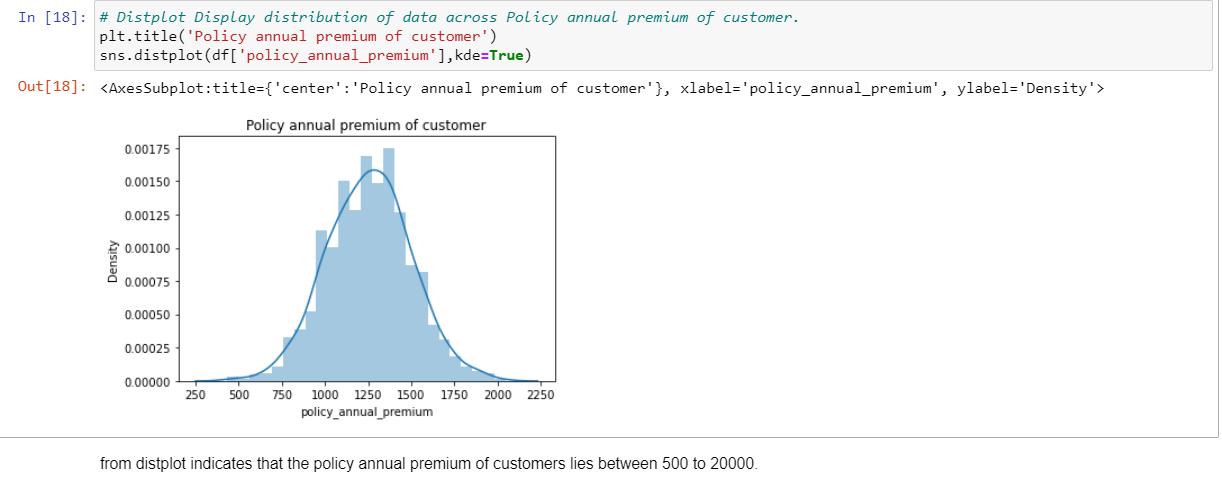


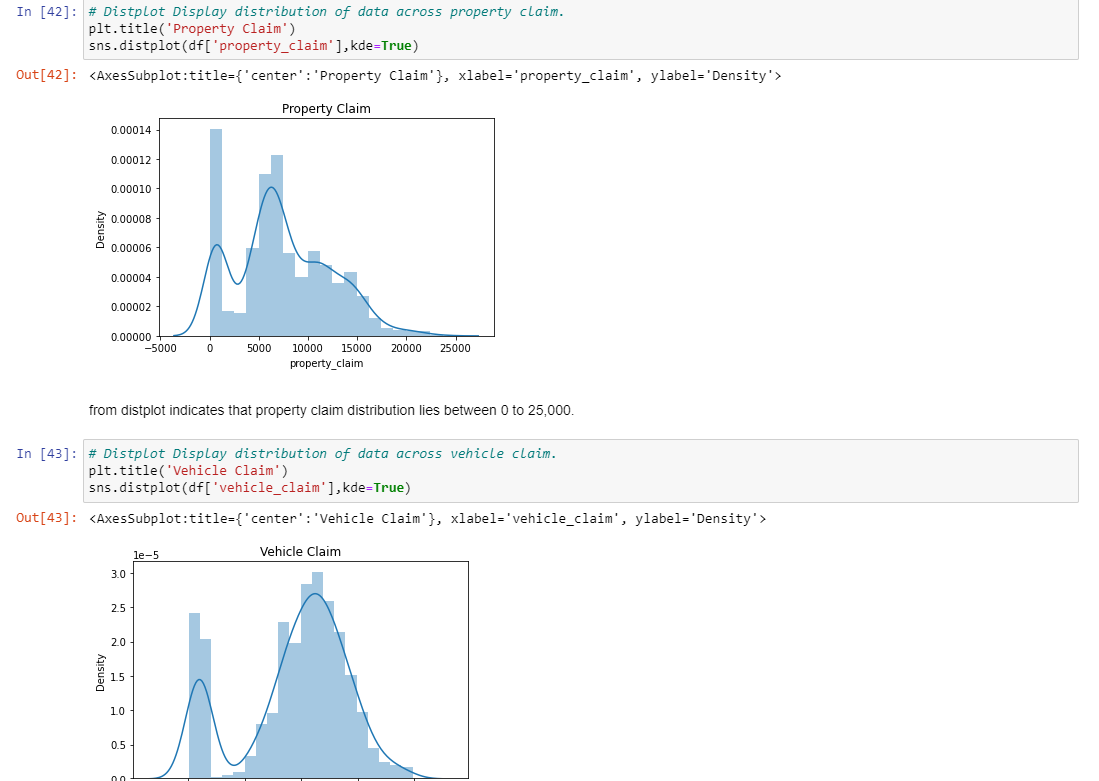








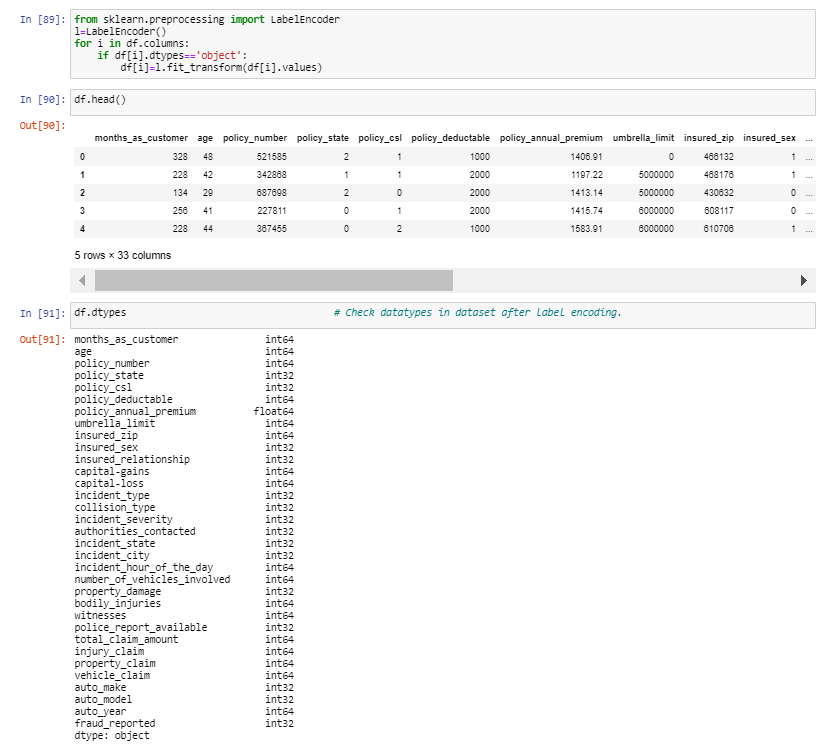




Graphical Represention Conclusion:

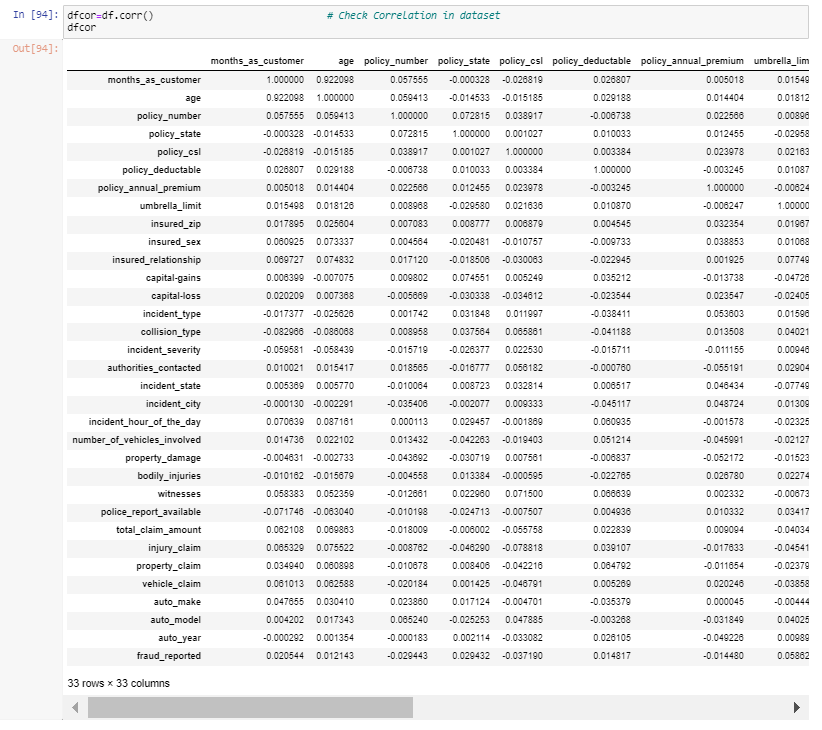
The columns that are going to be drop are policy bind date,incident date, incident location,insured education level,insured occupation,insured hobbies & c39.They are strings,cannot be categorized and don't contribute much to the outcome.

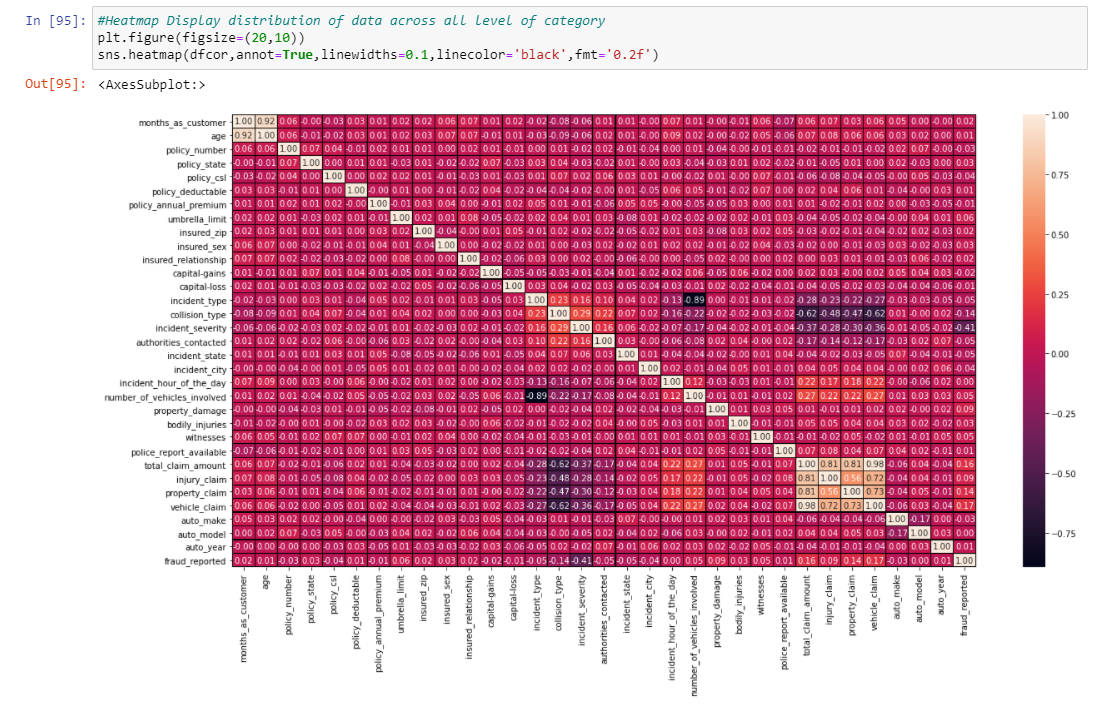
Converting all object Attributes into Numerical Attributes by using LabelEncoder



# Correlation Matrix

# Now lets look at the correlation between all the numerical variables. We can use the corr() to compute pairwise correlation of columns, excluding NA/null values using pearson correlation coefficient. Then we will use the heat map to visualize the correlation. Heatmaps visualize data through variations in coloring. The variables with darker color means their correlation is more.





**Pre-Processing Pipeline**

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data pre-processing is a method of resolving such issues.

## Missing value and outlier treatment

After exploring all the variables in our data, we can now impute the missing values and treat the outliers because missing data and outliers can have adverse effect on the model performance.

### Missing value imputation

Let’s list out feature-wise count of missing values.

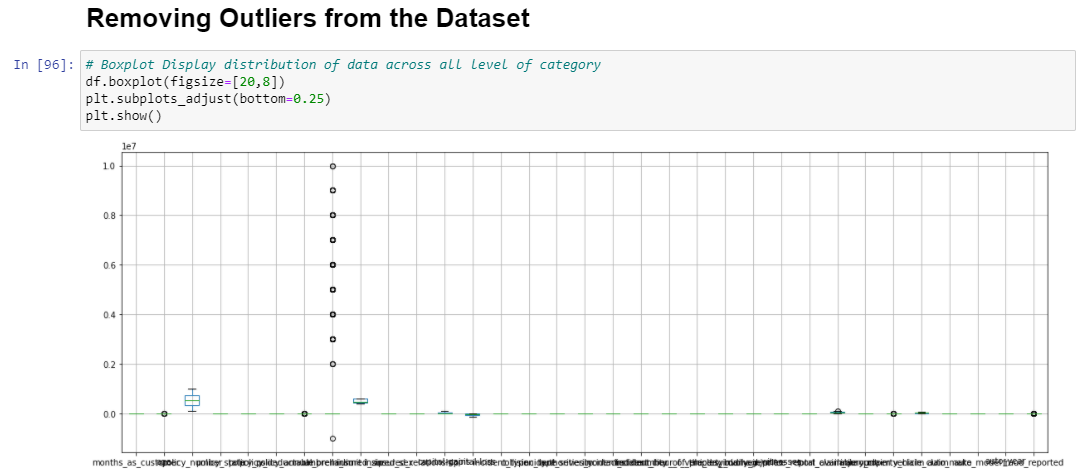


Since we didn’t find any null value in the Dataset .so they is no need of filling missing values.

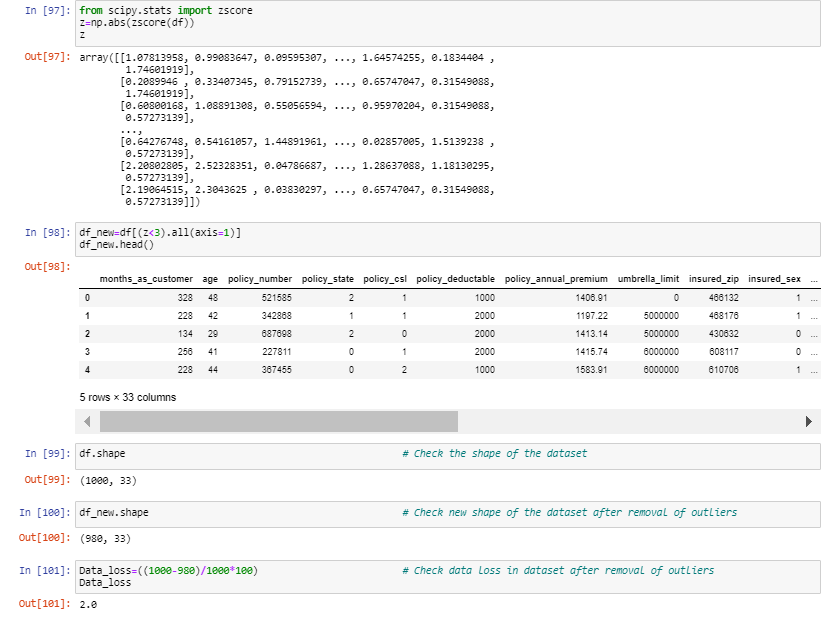
### Outlier Treatment

As we saw huge difference in the mean and standard deviation in the dataset,umbrella\_limit contains outliers so we have to treat them as the presence of outliers affects the distribution of the data. Having outliers in the dataset often has a significant effect on the mean and standard deviation and hence affecting the distribution. We must take steps to remove outliers from our data sets.

Due to these outliers bulk of the data in the loan amount is at the left and the right tail is longer. This is called right skewness (or positive skewness). One way to remove the skewness is by doing the power transformation. As we take the power transformation, it does not affect the smaller values much, but reduces the larger values. So, we get a distribution similar to normal distribution

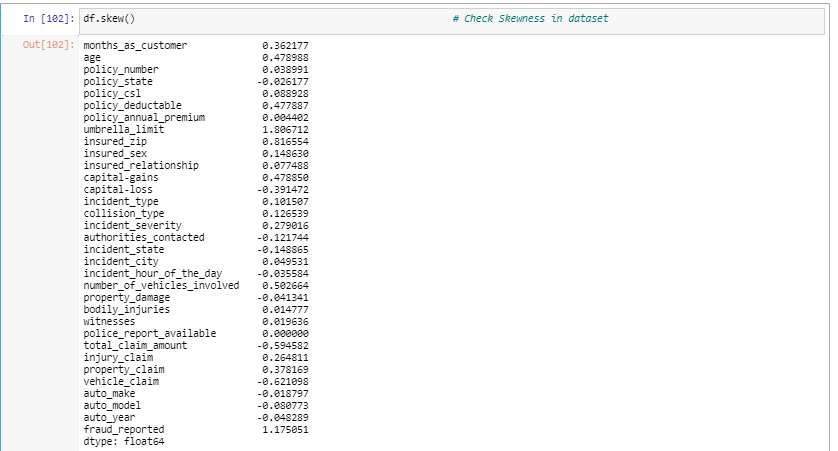


Now by seeing the above visualization we came across the outliers are in few attributes.So were are removing the ouliers by using **Z-Score method**.

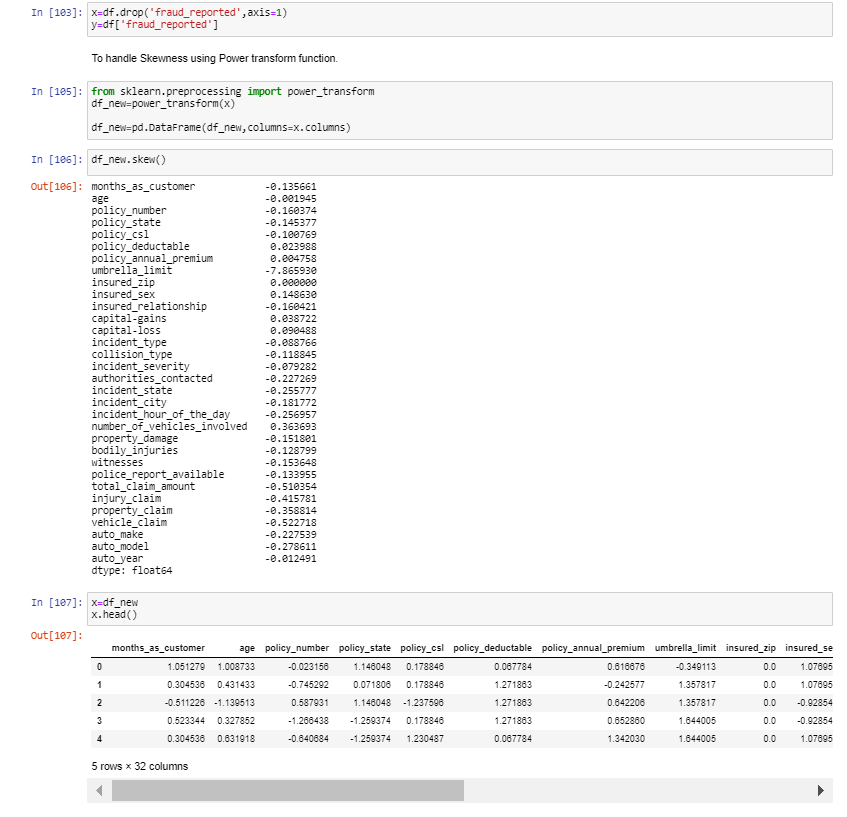


Above we can see the Data loss of the Dataset its around 2.00%.Hence outliers are removed from the Dataset.

# To check distribution of Skewness



# From above visualization we can see skewness of the dataset were few attributes found with skewness. Hence Removing Skewness is Done by using Power Transform Method.



# Building Machine Learning Models

## Evaluation Metrics for Classification Problems

The process of model building is not complete without evaluation of model’s performance. Suppose we have the predictions from the model, how can we decide whether the predictions are accurate? We can plot the results and compare them with the actual values, i.e. calculate the distance between the predictions and actual values. Lesser this distance more accurate will be the predictions. Since this is a classification problem, we can evaluate our models using any one of the following evaluation metrics:

**Accuracy**: Let us understand it using the confusion matrix which is a tabular representation of Actual vs Predicted values. This is how a confusion matrix looks like:

|  |
| --- |
|  |

True Positive - Targets which are actually true(Y) and we have predicted them true(Y)

True Negative - Targets which are actually false(N) and we have predicted them false(N)

False Positive - Targets which are actually false(N) but we have predicted them true(T)

False Negative - Targets which are actually true(T) but we have predicted them false(N)

Using these values, we can calculate the accuracy of the model. The accuracy is given by:

*Accuracy = (TP+TN) / (TP+TN+FP+FN)*

**Precision**: It is a measure of correctness achieved in true prediction i.e. of observations labeled as true, how many are actually labeled true.

*Precision = TP / (TP + FP)*

**Recall (Sensitivity)** - It is a measure of actual observations which are predicted correctly i.e. how many observations of true class are labeled correctly. It is also known as ‘Sensitivity’. E.g. Proportion of patients with a disease who test positive.

*Recall = TP / (TP + FN)*

**Specificity** - It is a measure of how many observations of false class are labeled correctly. E.g. Proportion of patients without the disease who test negative.

*Specificity = TN / (TN + FP)*

Specificity and Sensitivity plays a crucial role in deriving ROC curve.

**ROC curve**

* Receiver Operating Characteristic(ROC) summarizes the model’s performance by evaluating the trade offs between true positive rate (Sensitivity) and false positive rate (1- Specificity).
* The area under curve (AUC), referred to as index of accuracy(A) or concordance index, is a perfect performance metric for ROC curve. Higher the area under curve, better the prediction power of the model.
* The area of this curve measures the ability of the model to correctly classify true positives and true negatives. We want our model to predict the true classes as true and false classes as false.
* So it can be said that we want the true positive rate to be 1. But we are not concerned with the true positive rate only but the false positive rate too. For example in our problem, we are not only concerned about predicting the Y classes as Y but we also want N classes to be predicted as N.
* We want to increase the area of the curve which will be maximum for class 2,3,4 and 5 in the above example.
* For class 1 when the false positive rate is 0.2, the true positive rate is around 0.6. But for class 2 the true positive rate is 1 at the same false positive rate. So, the AUC for class 2 will be much more as compared to the AUC for class 1. So, the model for class 2 will be better.
* The class 2,3,4 and 5 model will predict more accurately as compared to the class 0 and 1 model as the AUC is more for those classes.

This is how a ROC curve looks like:

|  |
| --- |
|  |

# At the competition’s page, it has been mentioned that our submission data would be evaluated based on the accuracy. Hence, we will use accuracy as our evaluation metric.

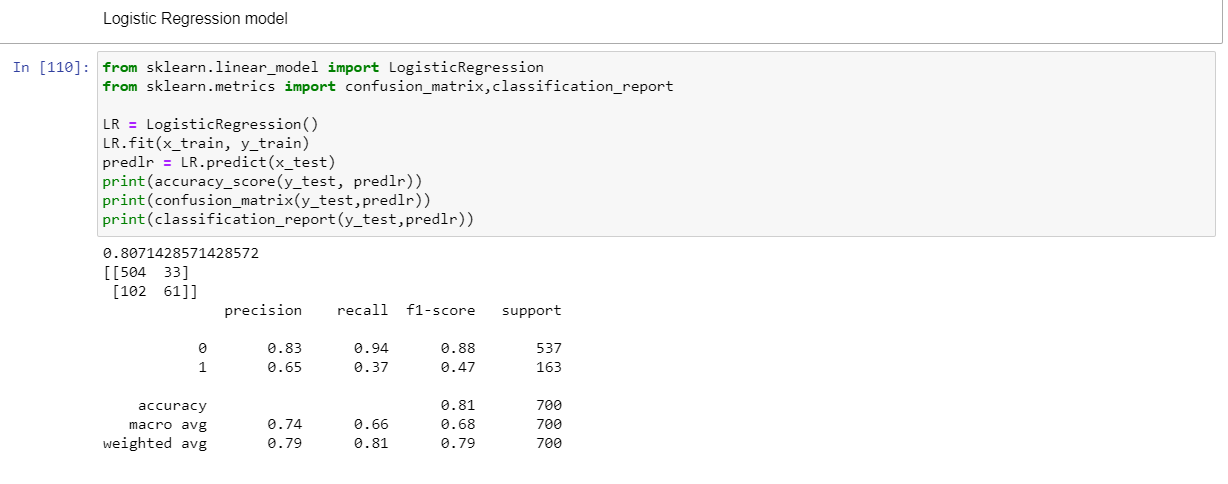
# Classification Model



## Model Building : Logistic Regression

Let us make our first model to predict the target variable. We will start with Logistic Regression which is used for predicting binary outcome.

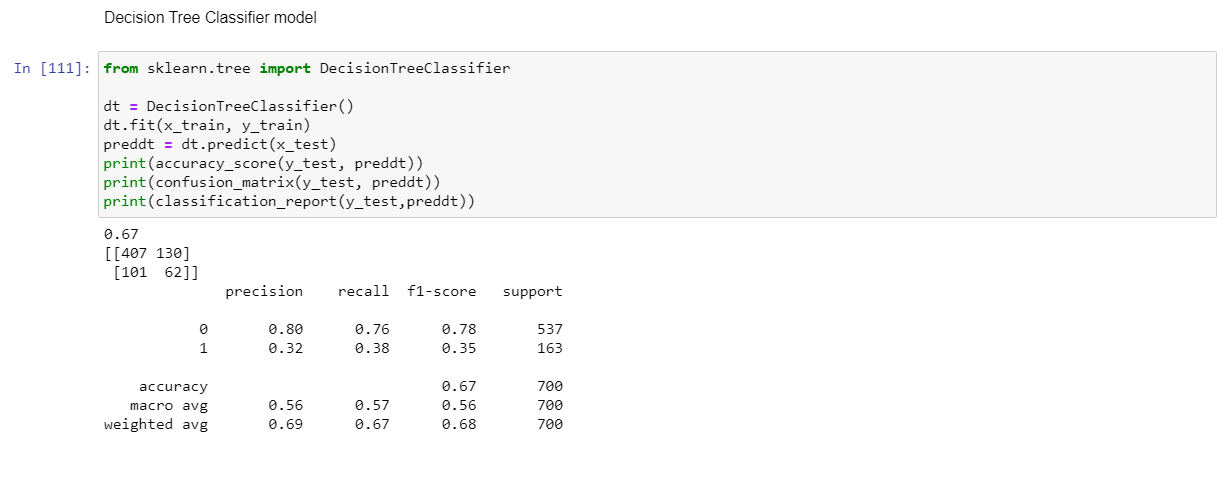
* Logistic Regression is a classification algorithm. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables.
* Logistic regression is an estimation of Logit function. Logit function is simply a log of odds in favor of the event.
* This function creates a s-shaped curve with the probability estimate, which is very similar to the required step wise function



### Decision Tree

Decision tree is a type of supervised learning algorithm(having a pre-defined target variable) that is mostly used in classification problems. In this technique, we split the population or sample into two or more homogeneous sets(or sub-populations) based on most significant splitter / differentiator in input variables.

Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable.



### Random Forest

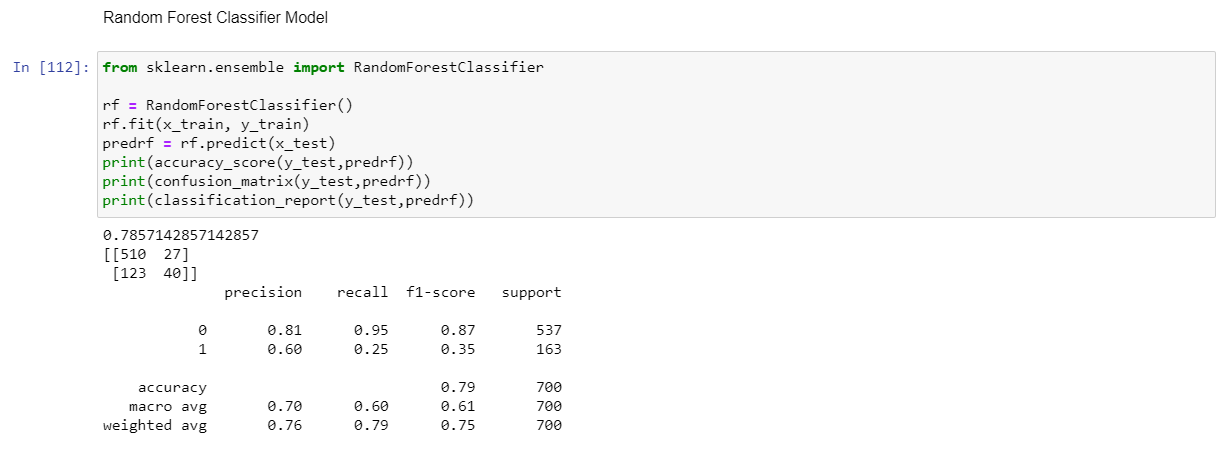
* RandomForest is a tree based bootstrapping algorithm wherein a certain no. of weak learners (decision trees) are combined to make a powerful prediction model.
* For every individual learner, a random sample of rows and a few randomly chosen variables are used to build a decision tree model.
* Final prediction can be a function of all the predictions made by the individual learners.
* In case of regression problem, the final prediction can be mean of all the predictions.

There are some parameters worth exploring with the sklearn RandomForestClassifier:

* n\_estimators
* max\_features

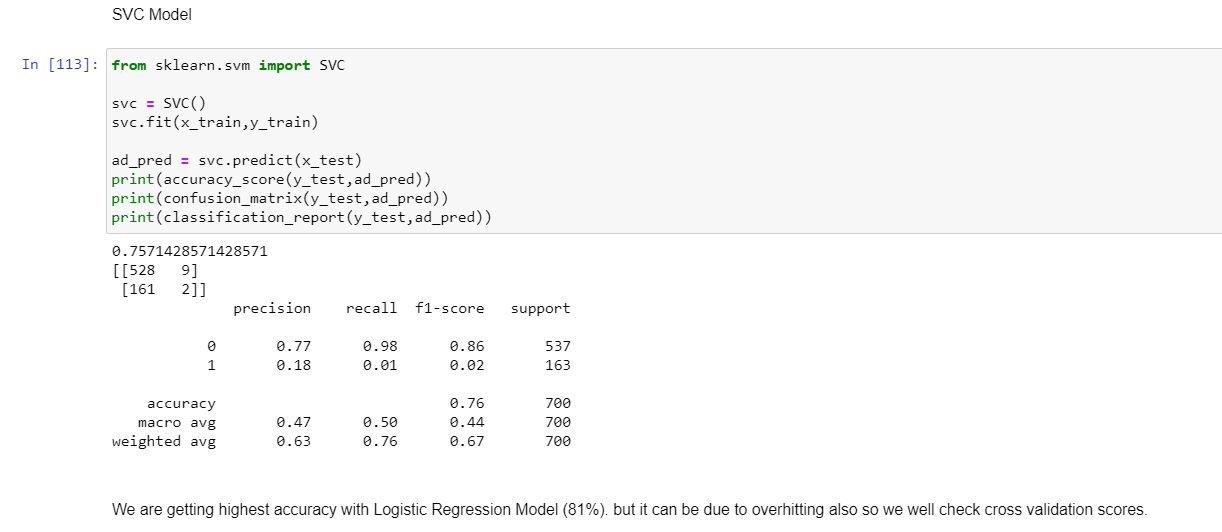
n\_estimators = ususally bigger the forest the better, there is small chance of overfitting here. The more estimators you give it, the better it will do. We will use the default value of 10.

max depth of each tree (default none, leading to full tree) - reduction of the maximum depth helps fighting with overfitting. We will limit at 10.



# SVC Model

# The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is.



We are getting highest accuracy with Logistic Regression Model (81%). but it can be due to overhitting also so we well check cross validation scores.

# Cross Validation Scores

# Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.



Min difference in accuracy and cross validation score is for SVC model . so this is our best model.

### GridSearchCV

We will try to improve the accuracy by tuning the hyperparameters for this model. We will use grid search to get the optimized values of hyper parameters. GridSearch is a way to select the best of a family of hyper parameters, parametrized by a grid of parameters.

We will use GridSearchCV in sklearn.model\_selection for an exhaustive search over specified parameter values for an estimator. GridSearchCV implements a “fit” and a “score” method. It also implements “predict”, “predict\_proba”, “decision\_function”, “transform” and “inverse\_transform” if they are implemented in the estimator used.

The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid, hence GridSearchCV. More info at <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>

We will tune the max\_depth and n\_estimators parameters. max\_depth decides the maximum depth of the tree and n\_estimators decides the number of trees that will be used in random forest model.

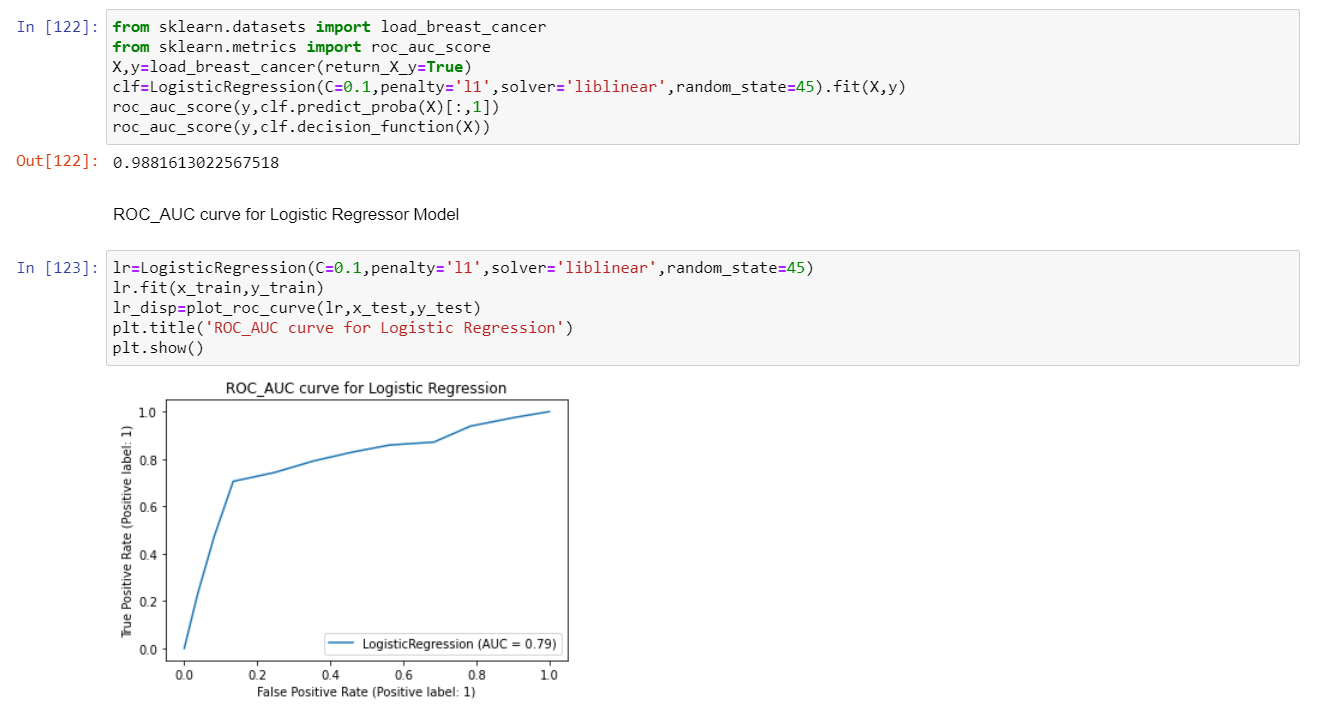
# Hyper parameter Tuning



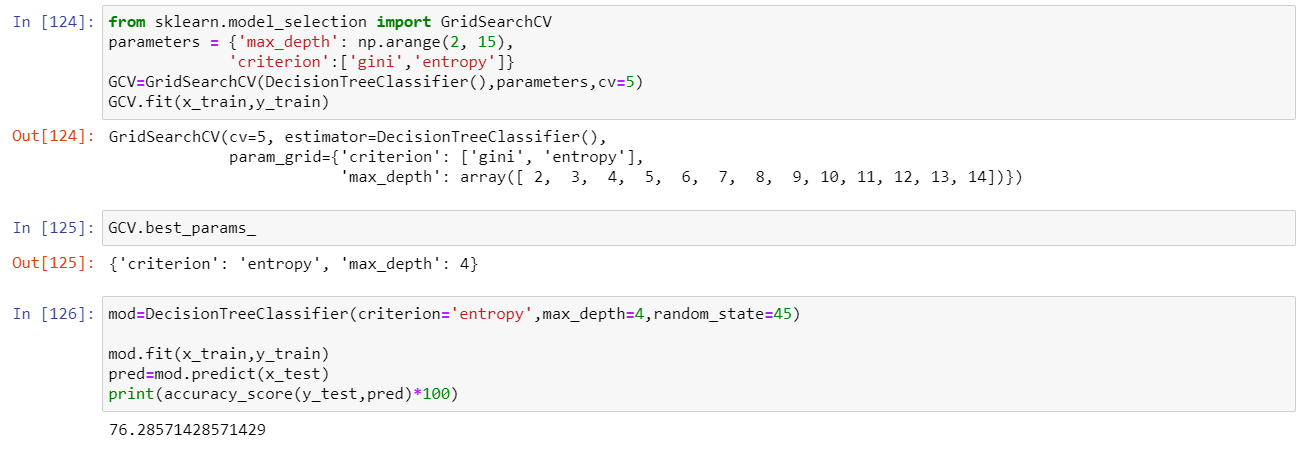
**Hyperparameter tuning for Logistic Regression Model**



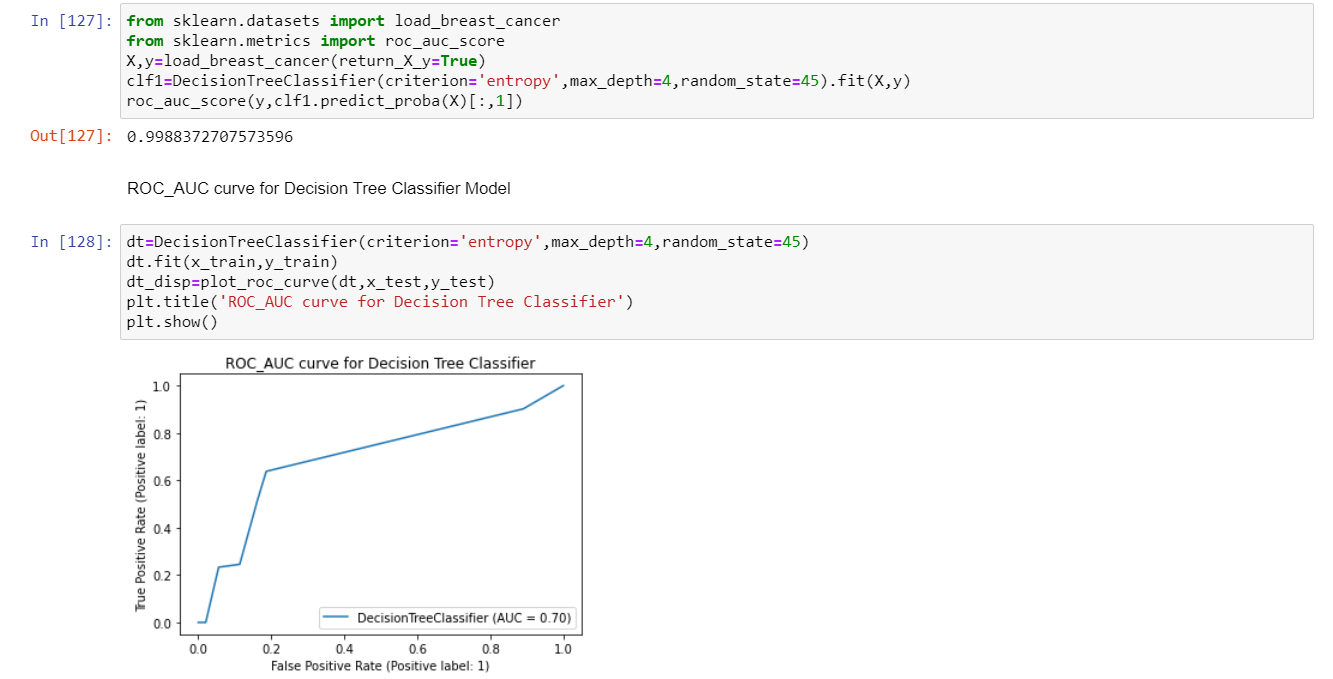
**ROC\_AUC Score for Logistic Regressor Model**



**Hyperparameter tuning for Decision Tree Classifier Model**



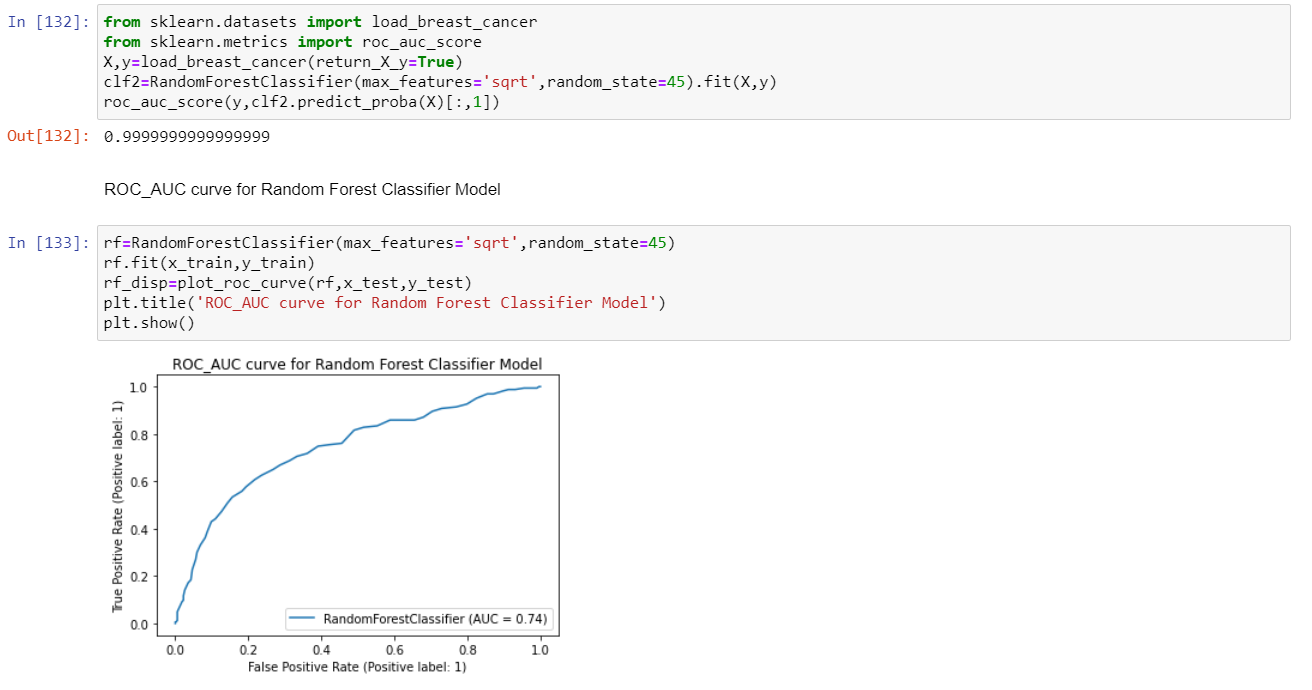
**ROC\_AUC Score for Decision Tree Classifier Model**



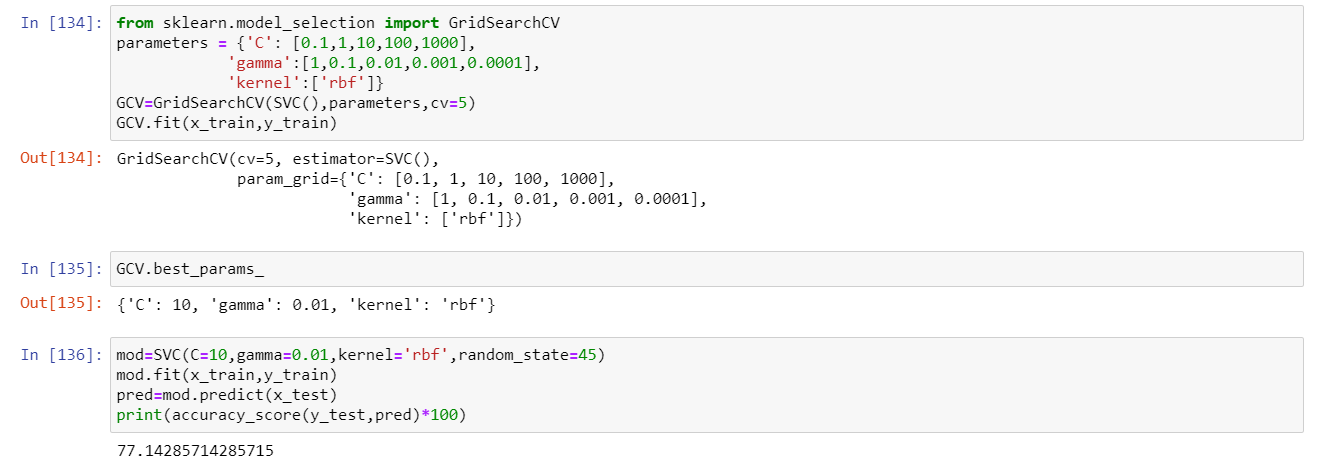
**Hyperparameter tuning for Random Forest Classifier Model**



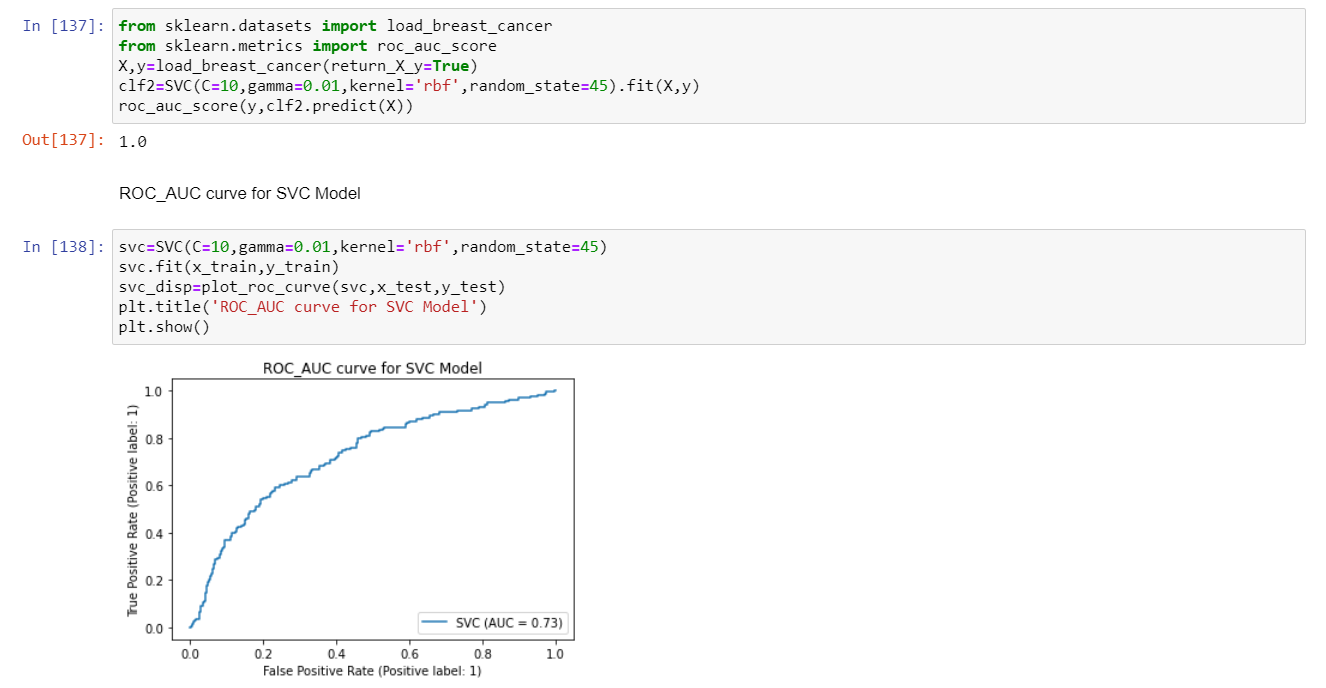
**ROC\_AUC Score for Random Forest Classifier Model**



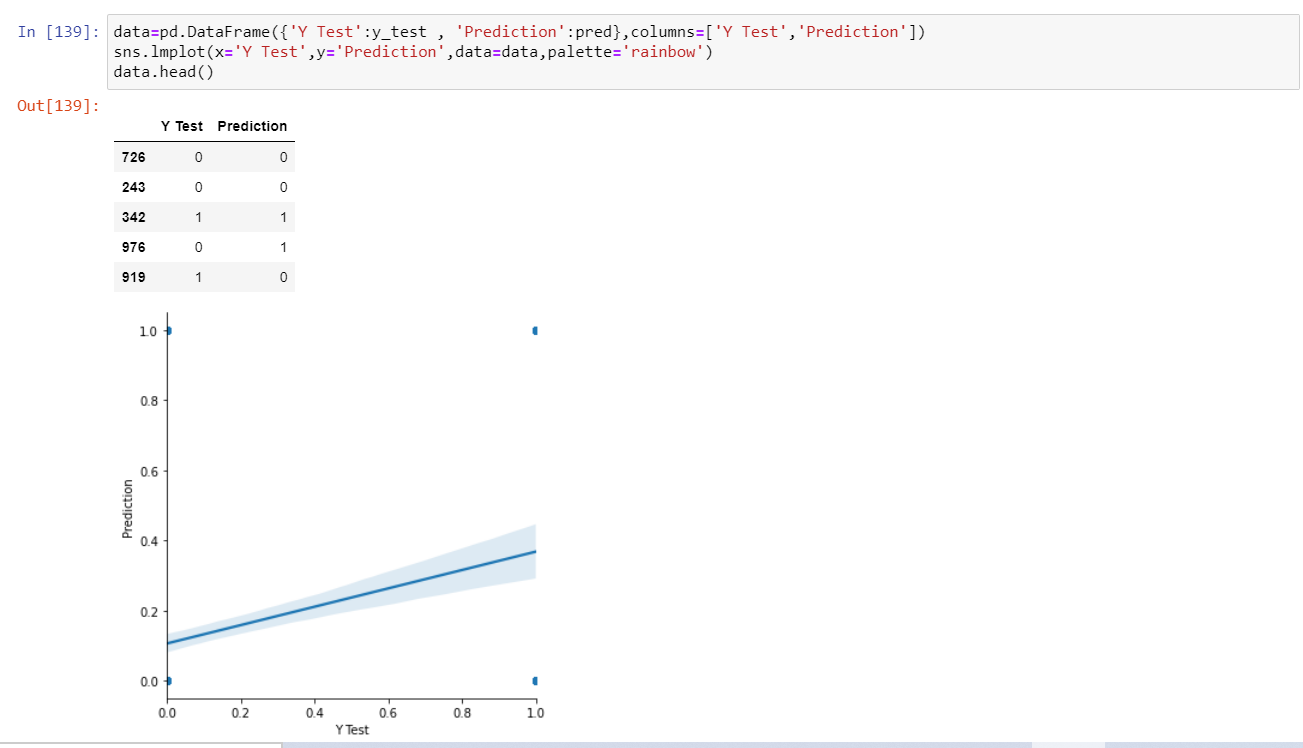
**Hyperparameter tuning for SVC model**

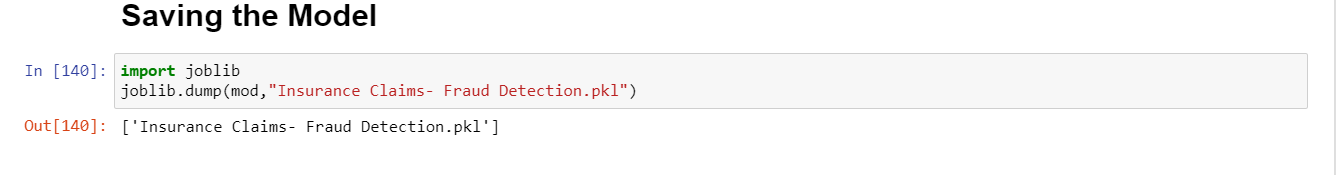


**ROC\_AUC Score for SVC Model**



Compare with Actual Final Vs Sample Prediction





# Conclusion

After trying and testing 4 different algorithms, the best accuracy on the leaderboard is achieved by by Logistic Regression (0.8142),Followed by Random Forest (0.7771). followed by SVC (0.7714) and Decision Tree (0.7628). While new features created via feature engineering helped in predicting the target variable, it did not improve the overall model accuracy much. Compared to using default parameter values, GridSearchCV helped improved the model's mean validation accuracy by providing the optimized values for the model's hyperparameters. On the whole, a logistic regression classifier provides the best result in terms of accuracy for the given dataset, without any feature engineering needed. Because of its simplicity and the fact that it can be implemented relatively easy and quick, Logistic Regression is often a good baseline that data scientists can use to measure the performance of other more complex algorithms. In this case, however, a basic Logistic Regression has already outperformed other more complex algorithms like Random Forest and SVC, for the given dataset.

**Suggestions for Improvement**. There are many things that can be tried to improve the models’ predictions. We can create and add more variables, try different models with different subset of features and/or rows, etc. Some of the ideas are listed below:

* Combine the applicants with 1,2,3 or more dependents and make a new feature as discussed in the EDA part.
* Make independent vs independent variable visualizations to discover some more patterns.
* Arrive at the EMI using a better formula which may include interest rates as well.
* Try ensemble modeling (combination of different models). More about ensemble techniques can be found at the references.
* Try neural network using Tensorflow or PyTorch

