PGPDSE FT- Pune (Nov'22)
Capstone Project – Final Report

Batch details	Post Graduate Program – Data Science & Engineering (Pune Nov'22)
Team members	1. Jash Gaddam 2. Shweta Suryavanshi 3. Harshada Karande 4. Jitendra Dixit 5. Parul Bhaiya
Domain of Project	Finance and Risk Analysis
Proposed project title	Online Bank Account Fraud Detection: Machine Learning Model a way forward.
Group Number	07
Team Leader	Jash Gaddam
Mentor Name	Mr. Jatinder Bedi

Overview

Banking industry has been undergoing major digital switch overs to provide enhanced services to the consumers, application for opening a new account is one of them. However, due to a fast-paced advancement in technologies that come with their loopholes, fraudsters have been using sophisticated techniques to commit fraud making it difficult for traditional rule-based systems to detect and prevent fraudulent activities.

Banking establishments are highly affected as this type of activities directly harm the bank consumer relationship and come with negative consequences for both the bank as well as the individuals involved. The banks may have to face situations such as loss of potential new consumers, loss of trust, reputation damages, and any substantial monetary loss for reparations of these situations. The consumers on the other hand may suffer from identity theft, monetary losses in some cases, service dissatisfaction, and sentiment affliction in some cases.

Problem Statement, Data and Summary-

Dataset :

The Bank Account Fraud Dataset (BAF) is a data suite published by NeurIPS (neural information processing systems foundation) 2022 with an aim to provide of synthetic accounts replicating real-world online bank account application fraud activity records.

In this step we will try to ,

- Understand the essence of the data.
- finding the range, categories and unique values present.

A. Understanding the dataset, it's dimensions, datatype, variable names.

The dataset contains of 10 lakh rows and 32 columns. There are 5 categorical columns and 27 numerical columns. Few categorical features possess anonymized data, and few features have unbalanced distribution of classes like source.

The table below describes the kind of data that is available in each of the column.

Name	**Description**	Data Type
Fraud_bool	Represented in binary values of 0 and 1. 0 displaying a negative outcome showcasing there is no fraudulent activities in the application. 1 showcases a affirmation of fraudulent activities being present.	**int64**
Income	The annual income of applicants is shown in a range of 0.1 to. 0.9. 0.1 being the lowest and 0.9 being the highest class of income.	**float64**
Name_email_similarity	Displays the similarity between the name of the applicants and their email address. The values are displayed in numeric	**float64**

	format ranging between 0 to 1. Higher values indicating higher similarity.	
Prev_address_months_count	The values in this column represent the count of months of previous residency of the applicant from the day of application. The range goes from -1 to 383.	**int64**
Current_address_month_count	Shows the count of months for current address residency of the applicant. The range goes from -1 to 428.	**int64**
Customer_age	Ages of the customers in decades form ranging from 10 to 90.	**int64**
Days_since_request	number of days since the application request. Ranges from 4 to 79.	**float64**
Intended_balcon_amount	Initial amount transferred for application of accounts. The range is between -15 to 113.	**float64**
Payment_type	The type of credit payment plan utilized by the applicant. There are 5 anonymized types of values.	**object**
Zip_count_4w	The count of applications made in the same zip code. Ranges from 1 to 6700.	**int64**
Velocity_6h	average number of applications made per hour in last 6 hours. Ranges between -17 to 16716.	**float64**
Velocity_24h	average number of applications made per hour	**float64**

	in 24 hrs. ranges between 1300 to 9507.	
Velocity_4w	average number of applications made per hour in the last 4 weeks. The range lies between 2825 to 6995.	**float64**
Bank_branch_count_8w	count of applications in selected branch of the bank in 8 weeks. Ranges between 0 to 2385.	**int64**
Date_of_birth_distinct_emails_4w	count of applicants with the same date of birth who have applied in the last 4 weeks. The range goes from 0 to 39.	**int64**
Employment_status	Employment status of the applicants represented in 7 different pseudonymous values.	**object**
Credit_risk_score	The internal score of credit risk during application that is set up with respect to the standards and regulations of the bank. Ranges between -170 to 389.	**int64**
Email_is_free	showcased in Boolean values of 0 and 1. 0 being free domain of application email and 1 being paid domain of application email.	**int64**
Housing_status	The housing status of applicants, for privacy reasons have been anonymized. There are 7 categories.	**float64**
Phone_home_valid	validity of the phone number being provided by the applicant in binary values 0 and 1. 0	**int64**

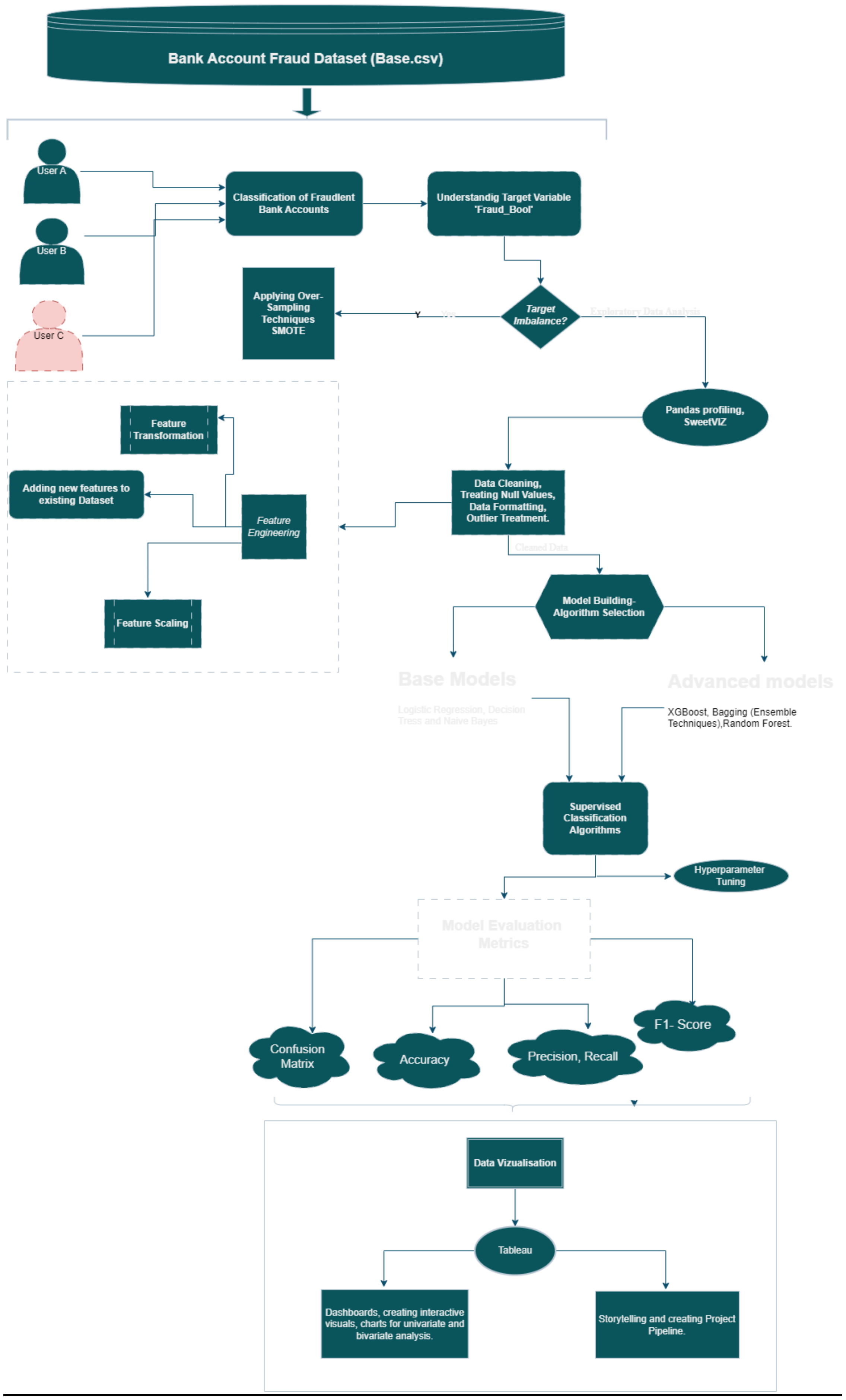
	representing invalid and 1 representing valid.	
Phone_mobile_valid	validity of the mobile number provided by the applicant in binary values 0 and 1. 0 being invalid and 1 being valid.	**int64**
Bank_months_count	The count of months of the previous account held by the applicant if any.	**int64**
Has_other_cards	Holding card(s) from the same company, represented in binary values 0 and 1. 0 showcasing applicant not holding any other card, 1 being positive affirmation of a card being held by the user.	**int64**
Proposed_credit_limit	applicant's proposed credit limit. Ranges between 190 to 2100.	**float64**
Foreign_request	represents whether the application request is made from a different country, showcased in binary values 0 and 1.	**int64**
Source	the online source of application. Has two categories internet and teleapp.	**object**
Session_length_in_minutes	length of the user's session / usage on the website of the bank. Ranges from -1 to 86 minutes.	**float64**
Device_os	The user's operation system utilised during the application request session. 5 categories being windows, macintosh, x11, linux, others.	**object**

Keep_alive_session	The log out option being used by the user represented in binary values 0 and 1.	**int64**
Device_distinct_emails_8w	number of distinct emails used by the user from the same device. Ranges between -1 to 2.	**int64**
Device_fraud_count	Count of fraudulent activities from used device.	**int64**
Month	The month of application.	**int64**

a.2 Checking for wrongly identified datatypes

All the variables datatypes are correctly identified and assigned .

Step-by-step Walkthrough of Solution –


Bank Account Fraud Dataset (Base.csv)
User A
User B
User C
Classification of Fraudlent Bank Accounts
Understandig Target Variable 'Fraud_Bool'
Target Imbalance?
Y
N
Exploratory Data Analysis
Applying Over-Sampling Techniques SMOTE
Pandas profiling, SweetVIZ
Feature Transformation
Adding new features to existing Dataset
Feature Engineering
Feature Scaling
Data Cleaning, Treating Null Values, Data Formatting, Outlier Treatment.
Cleaned Data
Model Building- Algorithm Selection
Base Models
Logistic Regression, Decision Trees and Naive Bayes
Advanced models
XGBoost, Bagging (Ensemble Techniques),Random Forest.
Supervised Classification Algorithms
Hyperparameter Tuning
Model Evaluation Metrics
Confusion Matrix
Accuracy
Precision, Recall
F1- Score
Data Vizualisation
Tableau
Dashboards, creating interactive visuals, charts for univariate and bivariate analysis.
Storytelling and creating Project Pipeline.

B. Data Exploration (EDA) :

Relationship between variables

Python provides with wide range of exploratory data analysis libraries via- pandas profiling and sweetviz, which is very useful for univariate and bivariate analysis, feature selections, presence of collinearity, null values, heatmaps and much more.

In this stage, the collected data is cleaned, transformed, and prepared for further analysis. This includes tasks such as handling missing values, there are no missing values in the dataset.

We have done outlier detection and removal, feature scaling, and feature engineering. Also have done the pandas profiling/sweetviz report. We also checked skewness and kurtosis of the data, and have used statistical test for checking the significance of the variables.

INSIGHTS FROM SWEETVIZ REPORT/ PANDAS PROFILLING:

Target variable (fraud_bool) has highly imbalanced data distrbution in the classes

- – class 0 (not fraudulent) having 99% data.
- – class 1 (presence of fraudulent) having only 1% weightage in data.

– Target variable shares a strong correlation with credit_risk_score

- Name_email_similarity shares a high correlation with categorical columns that have anonymised data, it also has alot of unique values.
- Income has been binned into 10 numerical categories.

– 0.9 have a higher weightage of data, this implies that income of the applicants in this data is comparatively higher.

- Current_address_month count holds alot of unique values as well, they have higher association ratio with housing_status(anonymised column)
- Customer_age is more dense in the age group of 30's.
- days_since_request show a density of datapoints in the range of 0 to 10 days. This indicates that most ofthe applicants can expect a approval for their account in within 10 days.
- payment_status,housing_status, being an anonymised data column, it's difficult to determine which category in the column is showing higher or lower weightage and understanding the behaviour of the data overall.
- 78% of the applicants do not have other cards whereas the other 22% do own other cards.
- foreign_request have a binary classification of 0 and 1. The data iis imbalanced in the classes with their beeing only 3% foreign requests being made for the above dataset.

Statistical significance of variables

Statistical significance is a term used in statistics to indicate whether an observed difference between two groups or the relationship between two variables is unlikely to have occurred by chance. In other words, it is a measure of the degree to which an observed effect is likely to be a true effect, as opposed to a random variation or coincidence.

To determine the statistical significance of variables, we performed various hypothesis tests. A hypothesis test involves stating a null hypothesis, which is the assumption that there is no relationship between the variables or no difference between the groups. Then, we collected the data and use statistical methods to calculate a p-value, which is the probability of observing the data you collected, or more extreme data, if the null hypothesis is true.

We performed chi2_contingency, f_oneway(ANOVA), ttest etc to derive and differentiate between significant and insignificant features and accordingly add to our base model the most appropriate significant features with respect to our target feature – 'fraud_bool'

- According to the anova test conducted, for **device_os** column has a significance as pvalue is less than alpha.
- **source** column seems to show an insignificance with relation to target variable based on the anova test as their p-value is more than alpha.

The p-value list for the numerical columns after performing ttest is showcasing an array where except for the first value all the other numerical values have pvalue less than the alpha, hence it can be concluded that these columns are significant.

C. Feature Engineering

Scaling: Scaling features to a common range can improve the performance of some machine learning algorithms. We have performed standard scaling on numeric data to prevent scale sparsity between the features.

Encoding categorical variables: Many machine learning algorithms cannot handle categorical variables directly. Therefore, categorical variables must be transformed into numeric variables using techniques such as one-hot encoding , ordinal encoding or Label encoding. We will perform label encoding on few columns to derive more information out of it for model building and extract important information out of them.

Feature extraction: It was necessary to extract new features from the raw data in order to capture important patterns. We will use feature extraction techniques like Principal Component Analysis (PCA) and LDA in order to reduce dimensionality and also gain maximum information from our data without losing many important descriptions.
For our following Data, we have performed feature engineering for one column so far.

Binning the required columns

The credit risk score column showcases the risk factor of the applicant in numerical form.

- It is also a column that shares a collinearity with the target variable and can be considered as a significant variable based on the standard deviation, pvalue(with reference to the stats test).
- However it also holds a large number of unique values which can be somewhat complicated for the model to learn the patterns from.
- Hence we will be binning the credit score column and change it into a categorical column

Encoding the data

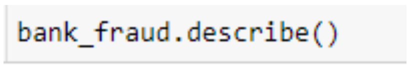
There were five categorical columns that will be required to be encoded in order to move onto model building for the data. Encoding would be performed using the get_dummies function of pandas library.
Identifying the dependent variables and assigning them as X where as assigning the dependent variable (bank_fraud) as Y for splitting into train and test set.

The split will be done by using the train_test_split functionality from the sklearn.model_selection library package.

Changing 'Months in current address" to "Years in current Address"

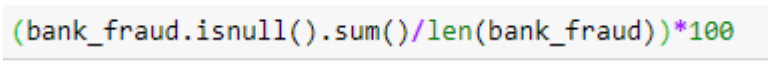
To get inferences more related to the customers stay details in the current address, we converted an existing feature to a new feature and converted the months data into years and included it to the model bulding.

Appendix:



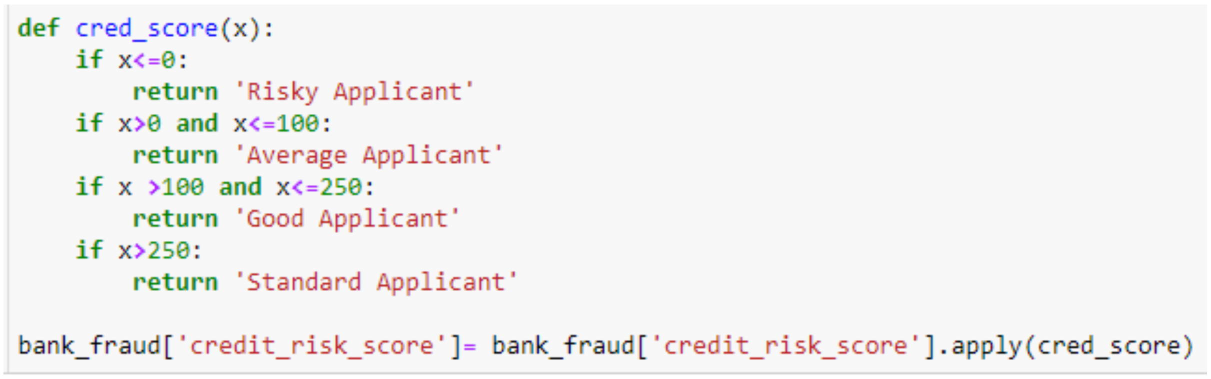
```
bank_fraud.describe()
```

The `describe()` function that is used to generate a descriptive statistical summary of a DataFrame . It calculates various statistics like count,mean,sd, minimum and maximum values as well as quartiles for each numerical column in the dataframe.



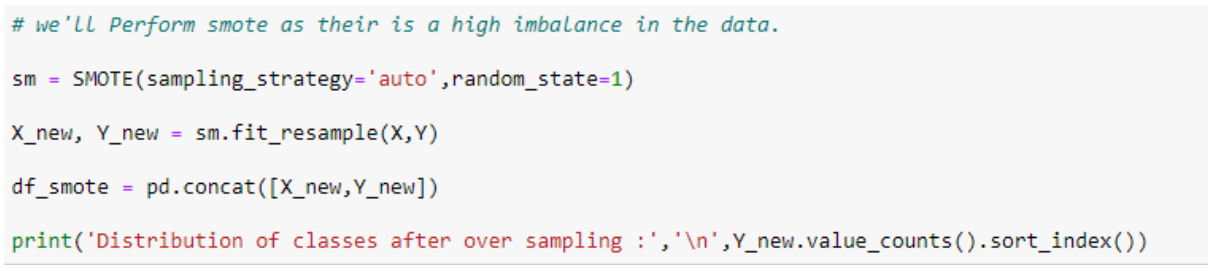
```
(bank_fraud.isnull().sum()/len(bank_fraud))*100
```

This will show us the percentage of the null values in the dataset



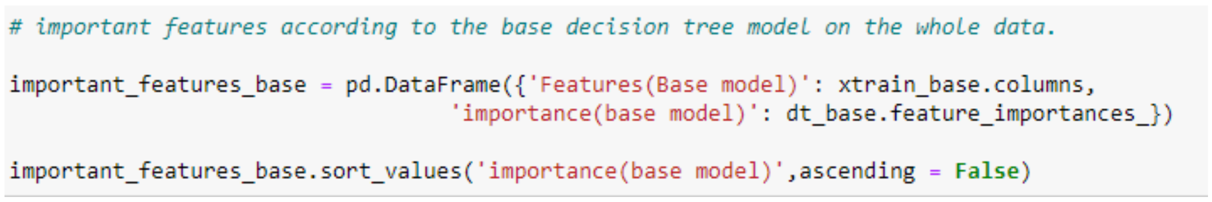
```
def cred_score(x):
    if x<=0:
        return 'Risky Applicant'
    if x>0 and x<=100:
        return 'Average Applicant'
    if x >100 and x<=250:
        return 'Good Applicant'
    if x>250:
        return 'Standard Applicant'

bank_fraud['credit_risk_score']= bank_fraud['credit_risk_score'].apply(cred_score)
```

Here we have defined a function for credit score. Values less than 0 will be showed as risky applicant. Values greater than 0 and less than 100 showed as average applcant,values greater than 100 and less than 250 showed as good applicant and greater than 250 showed as standard applicant.



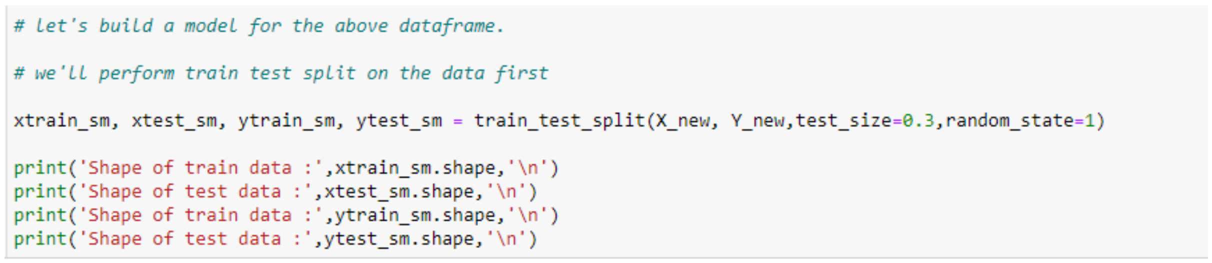
```
# we'll Perform smote as their is a high imbalance in the data.

sm = SMOTE(sampling_strategy='auto',random_state=1)

X_new, Y_new = sm.fit_resample(X,Y)

df_smote = pd.concat([X_new,Y_new])

print('Distribution of classes after over sampling :','\n',Y_new.value_counts().sort_index())
```

We have used smote analysis (over sampling), as our target data is very imbalance so we have to use smote for our data.



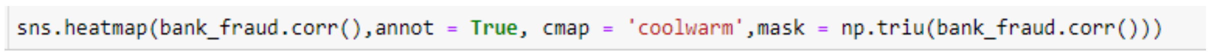
```
# important features according to the base decision tree model on the whole data.

important_features_base = pd.DataFrame({'Features(Base model)': xtrain_base.columns,
                                        'importance(base model)': dt_base.feature_importances_})

important_features_base.sort_values('importance(base model)',ascending = False)
```

On the base model we found the important features and sorted them in descending order.



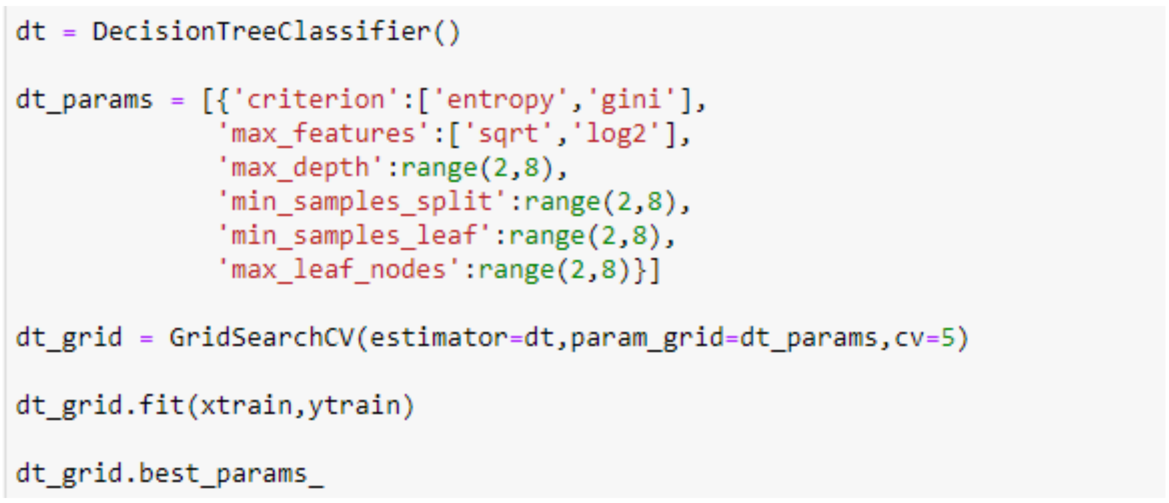
```
# let's build a model for the above dataframe.

# we'll perform train test split on the data first

xtrain_sm, xtest_sm, ytrain_sm, ytest_sm = train_test_split(X_new, Y_new,test_size=0.3,random_state=1)

print('Shape of train data :',xtrain_sm.shape,'\n')
print('Shape of test data :',xtest_sm.shape,'\n')
print('Shape of train data :',ytrain_sm.shape,'\n')
print('Shape of test data :',ytest_sm.shape,'\n')
```

Here we have splitted the variables as dependent and independent variables and we have done the train test split on the data with 70:30 ratio.



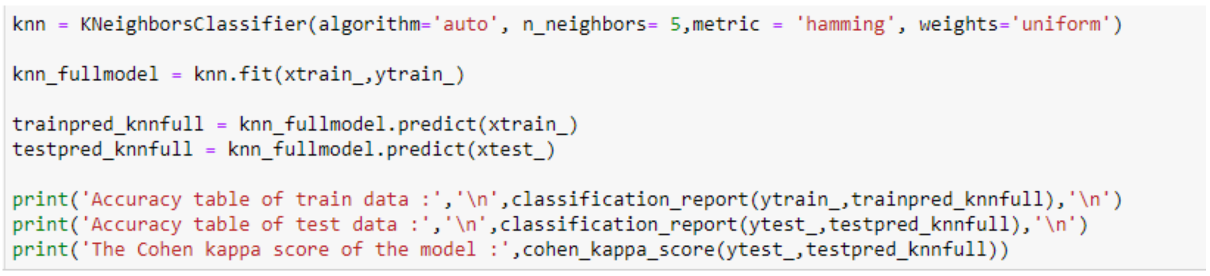
```
sns.heatmap(bank_fraud.corr(),annot = True, cmap = 'coolwarm',mask = np.triu(bank_fraud.corr()))
```

A heatmap is a graphical representation of data using color-coding to represent various values. Heatmaps are mainly used for the representation of data. Where colour of the cell is represented by the value it represents. Heatmaps are commonly used in data visualization to show the distribution and intensity of the variables.



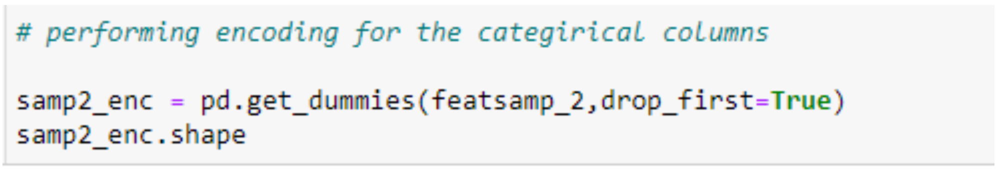
```python
dt = DecisionTreeClassifier()

dt_params = [{'criterion':['entropy','gini'],
             'max_features':['sqrt','log2'],
             'max_depth':range(2,8),
             'min_samples_split':range(2,8),
             'min_samples_leaf':range(2,8),
             'max_leaf_nodes':range(2,8)}]

dt_grid = GridSearchCV(estimator=dt,param_grid=dt_params,cv=5)

dt_grid.fit(xtrain,ytrain)

dt_grid.best_params_
```

Base model, and performing grid search cross validation with value as 5 folds which returns the best performing parameters , fit for our model.



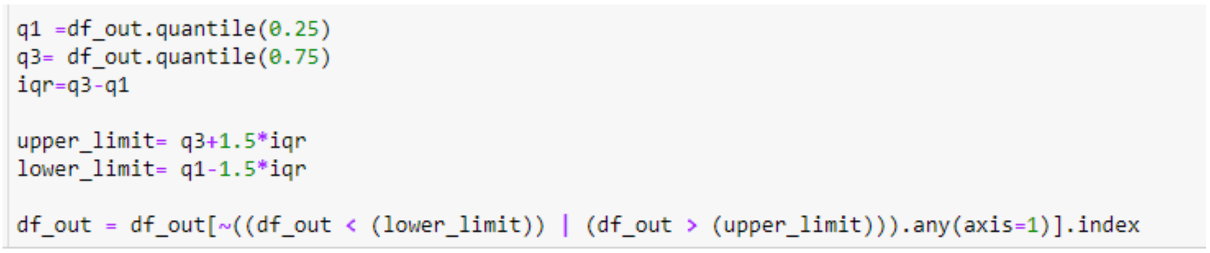
```python
knn = KNeighborsClassifier(algorithm='auto', n_neighbors= 5,metric = 'hamming', weights='uniform')

knn_fullmodel = knn.fit(xtrain_,ytrain_)

trainpred_knnfull = knn_fullmodel.predict(xtrain_)
testpred_knnfull = knn_fullmodel.predict(xtest_)

print('Accuracy table of train data :','\n',classification_report(ytrain_,trainpred_knnfull),'\n')
print('Accuracy table of test data :','\n',classification_report(ytest_,testpred_knnfull),'\n')
print('The Cohen kappa score of the model :',cohen_kappa_score(ytest_,testpred_knnfull))
```

KNN-Model initialization and fitting the model with input train parameters of target and independent variables. Printing the accuracy and kappa score for the model performance.



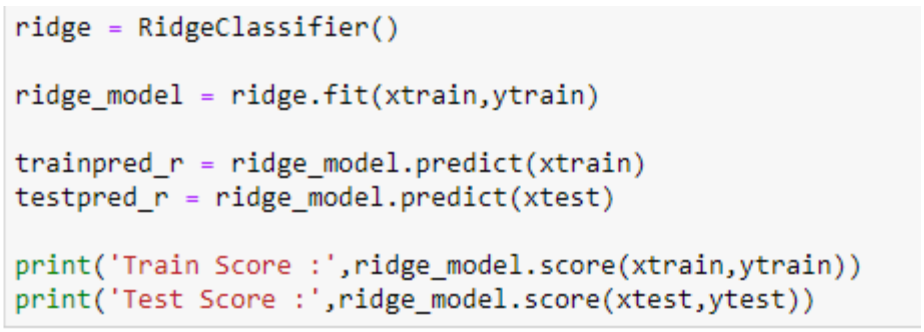
```python
# performing encoding for the categirical columns

samp2_enc = pd.get_dummies(featsamp_2,drop_first=True)
samp2_enc.shape
```

Performing dummy encoding on categorical columns and converting them into sets of zeroes and ones to include them into our model building.



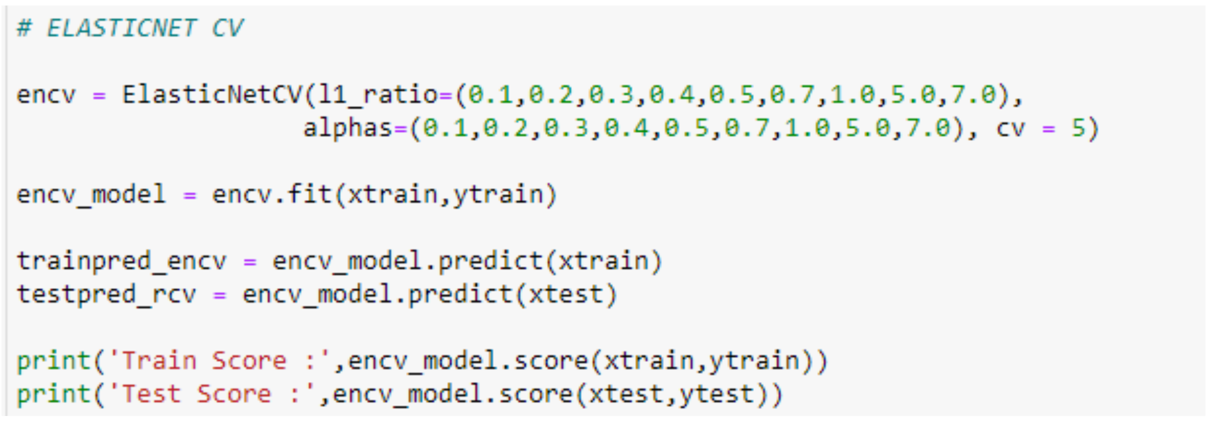
```python
q1 =df_out.quantile(0.25)
q3= df_out.quantile(0.75)
iqr=q3-q1

upper_limit= q3+1.5*iqr
lower_limit= q1-1.5*iqr

df_out = df_out[~((df_out < (lower_limit)) | (df_out > (upper_limit))).any(axis=1)].index
```

One of the widely used method to remove outliers using the Inter quartile range method and removing values more or less than, lying under that span.



```
ridge = RidgeClassifier()

ridge_model = ridge.fit(xtrain,ytrain)

trainpred_r = ridge_model.predict(xtrain)
testpred_r = ridge_model.predict(xtest)

print('Train Score :',ridge_model.score(xtrain,ytrain))
print('Test Score :',ridge_model.score(xtest,ytest))
```

Ridge Regularization



```
# ELASTICNET CV

encv = ElasticNetCV(l1_ratio=(0.1,0.2,0.3,0.4,0.5,0.7,1.0,5.0,7.0),
                    alphas=(0.1,0.2,0.3,0.4,0.5,0.7,1.0,5.0,7.0), cv = 5)

encv_model = encv.fit(xtrain,ytrain)

trainpred_encv = encv_model.predict(xtrain)
testpred_rcv = encv_model.predict(xtest)

print('Train Score :',encv_model.score(xtrain,ytrain))
print('Test Score :',encv_model.score(xtest,ytest))
```

ElasticNet Regularization

D. <u>Model Evaluation (Base and Advance Models)</u> :

A decision Tree model was chosen as the base model for the raw data above.

The accuracy score for the train and test data derived was 1.00 and 0.98 respectively.

Looking at the accuracy score we can say that the model is going into overfit condition. since there is an imbalance in data it is obvious that the model will only learn the behaviour and pattern of one class at a stretch. this can be a big problem as the model should be able to understand the patterns for both the classes to properly identify the presence of fraudulent activities.

A model can go into overfitting condition for several reasons :

- If there is presence of imbalance in the variables data –
- It was observed that the target variable consists of highly imbalanced data.
- An imbalance in the classes of variables makes the model highly trained in just one class where as the model fails to understand and learn the patterns in the minority classes that can help in proper identification of the classes for prediction.
- If there is alot of noise present in the data - The data consists of extreme outliers in multiple columns

• If there is any irrelevant features/information present in the data.

– Through the heatmap, statistical hypothesis tests conducted earlier and feature importance values derived from the base model it can be determined that there are a few columns that can be considered as irrelevant or of insignificance as they share a very low relation with the target variable.

Model evaluation

Description of the final model in detail-

KNN Model (K-Nearest Neighbours)

1. Model performance -

- The KNN Algorithm uses feature similarity to identify and determine the prediction of the datapoints.

- The algorithm identifes and understands the patterns in the training set of the data based on which the test datapoints get predicted for the classes.

- For the following model 4 knn parameters algorithm, n_neighbours, metric, weights were taken into consideration

- n_neighbours determine the number of nearest neighbours to be considered during class identification

- metrics - Metrics determine the distance to be calculated between two datapoints of training set for to determine the value of that dataoint which can be later predicted to the test datapoint.

- Weights is a parameter that assigns weightage to each nearest neighbour based on the parameter passed.

For our model the parameters were :

- The algorithm was set as auto to find the most optimal point for the value passed.

- n_neighbours was taken as 5

- Hamming distance was used as our data is categorically distributed in binary columns (0 & 1). It determines the distance based on the values. If the values of x and y are same it will pass the distance as 0 or else 1.

- weights were given as uniform in order to ensure equal weightage is given to the datapoints.

Model Evaluation :

- The. model was evaluated by creating classification report deriving the precision, recall, f1-score, accuracy of the training and test data.

- Looking at the accuracy score it can be said that the model is a goodfit with not much difference for train and test data accuracy at 0.91 and 0.86 respectively.

1. Precision : determines the number of correct outputs are provided by the model. Out of all the predicted values it gives us the ratio for how many of them were actually true.

   - In this case the precision of the model is shows higher precision for class 1 than class 0.

2. Recall : gives us the ratio that out of all the positive classes how many of them were correctly predicted by our model.

   - For our above model, the recall is higher for class 0 in train and test set, for class 1 it is higher for train set but comparatively lower for training set. This indicates that the actual positive values for class 0 is being more accurately predicted by the model.

3. F-1 score : f1 score gives us the score by evaluating the precision and recall measures at the same time.

   - The f1 score here is pretty high for train and test set as for both the classes.

4. Cohen kappa score measures the agreement between the two raters (test set, predicted set)

   - The cohen kappa score determines the overall accuracy of the model.

   - For the above model the cohen kappa score is 0.715.

   - It can be said that there is substantial agreement for the overall model for the prediction of values of classes.

- Looking at the precision values, it was observed that the possibility of the value predicted as 0 (absence of fraudulent activities) is lesser than that of the class 1 (presence of fraudulent activities). We can say that the model is predicting fraudulent activities well.

- However if we look at the recall values. it can be determined that the possibility of the predicted value not being a fraudulent activity(0) has a higher score. However the possibilty of fraudulent activity being showcased during application is being predicted well in train set over test set.

- Since the recall values show a better result for class 0 and precision values show a better result for class 1 we'll consider the f1 score as a measure for evaluating the accuracy. Here the f1 score as well shows a good measure for training and test set of class 0 and training set of class 1. But a comaparatively lower score for test set of class 1.

- The cohen kappa score shows that the there is a substantial agrement bvetween the test set and the predicted set, which means that the accuracy of the predictions done by the models is overall solid.

This can be interpreted as -

1. The model is giving a good performance for the prediction of presence or absence of fraudulent activites.

2. Although the evaluations showcase that there is a lower recall value for fraudulent activities being actually true, it is still high considering that the happenings of fraudulent activites are also lower, and the data itself had a lesser number of active fraudulent activites datapoints present.

3. Overall it can be said that the model is good fit model.

5. Comparison to benchmark How does your final solution compare to the benchmark you laid out at the outset? Did you improve on the benchmark? Why or why not?

The minimal accuracy score required for the model was initially set at 0.85. The finished model met the bar established at the beginning with an accuracy score of 0.86.

On additional evaluation measures, such as precision, recall, and F1-score, the final model outperforms the benchmark. The model's greater precision rating for class 1 reflects how well it can anticipate fraudulent behaviour.

Additionally, the recall value for class 0 is greater in the train and test sets, demonstrating that the model is effective at predicting legitimate activities. Furthermore, the F1-score demonstrates a strong measure for the training and test sets for class 0 and class 1, demonstrating the model's overall good fit. The Cohen kappa score demonstrates that there is strong agreement between the test set and the predicted set, showing that the models' predictions were generally accurate**.**

Because the final model outperforms the benchmark on some assessment measures while still meeting the benchmark on others, the final solution represents an upgrade over the benchmark**.**

6. Visualization(s) In addition to quantifying your model and the solution, please include all relevant visualizations that support the ideas/insights that you gleaned from the data.

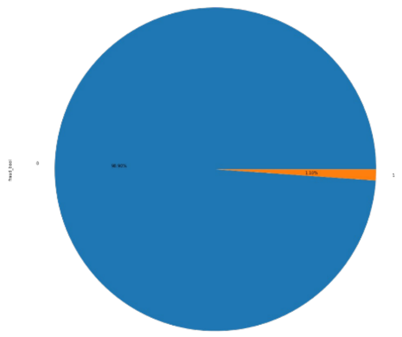


Fig: Target Variable Distribution

The data has highly imbalanced weightage for each class.

- **Unbalanced** refers to a classification data set with unbalanced class proportions. Majority classes are those that represent a sizable share of the data set. Minority classes are those that make up a smaller percentage.

Our target variable 'fraud_bool', is a highly unbalanced feature with classes [0,1]
and percentage share of **98.90%** and **1.10% of 0 and 1 respectively.**

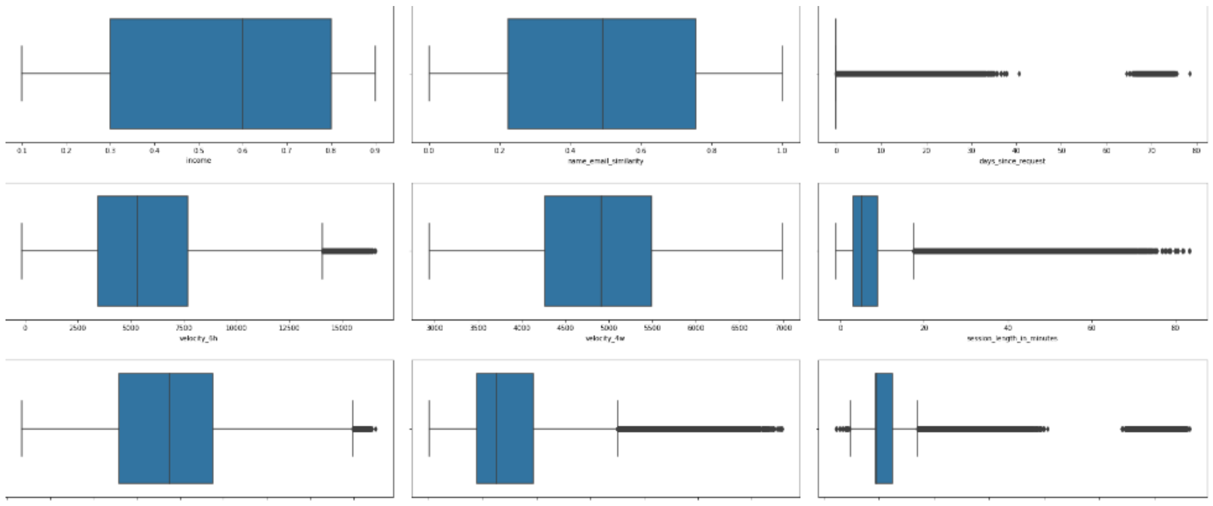


Fig : Boxplots showing presence of outliers in numerical features

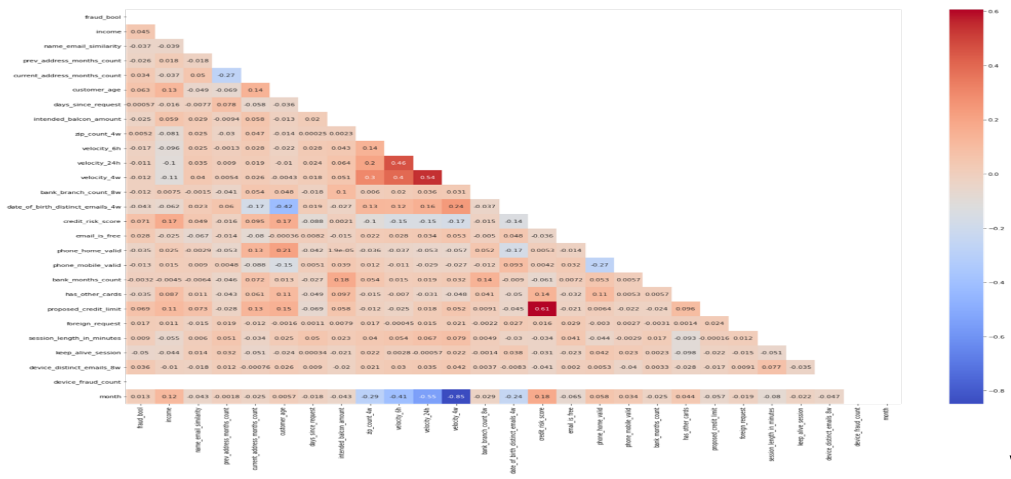


Fig: Heatmap showing correlation between features

Multi-collinearity-

High correlations between two or more independent variables are known as multicollinearity (predictors). This phenomena fundamentally involves the correlation of independent variables. Based on their correlation, it assesses how closely two variables are related.

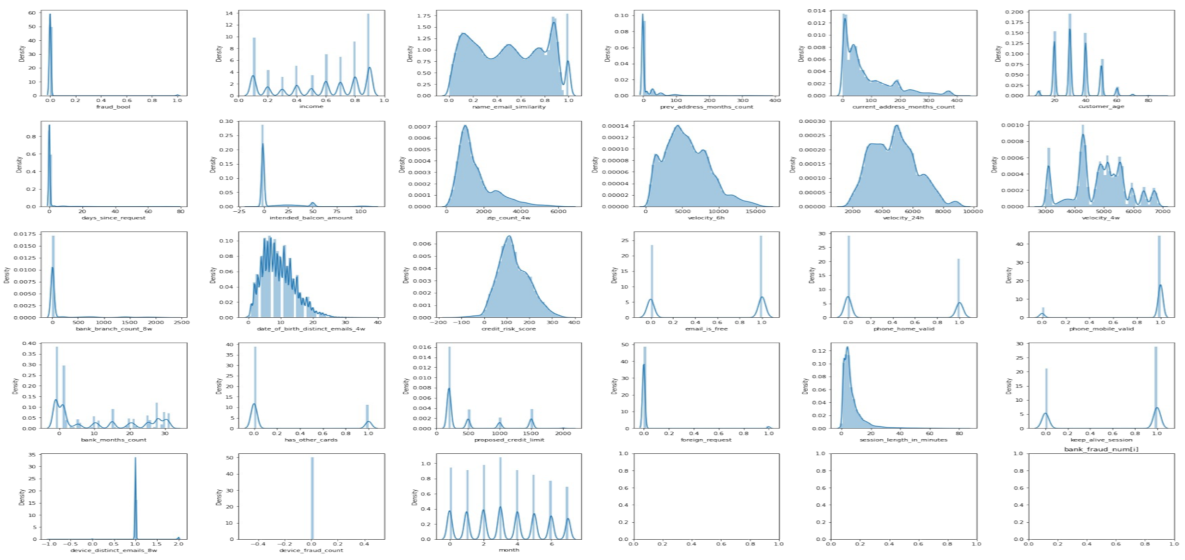


Fig : Distribution of Continous numerical features

- Columns, Fraud_bool, email_is_free, phone_number_valid, phone_home_valid, has_other_cards, foreing_request, keep_alive_sessions have binary classification(0 and 1).
  - – fraud_bool, phone_mobile_valid, foreign_request, has_other_cards have a very high imbalanced distribution of datapoints.
  - – All the other columns have a moderately decent distribution of datapoints within the classes.
- Columns income, customer_age have binned data in multiple numerical classes. It can be observed that for each class the distribution.

  – income shows a marginally moderate difference of distribution among the classes.

  – customer_age displays a high distribution between the range 20 to 40.

- All the other columns have a highly uneven distribution of data.

The skewness of the data gave us the following observations –

- Columns prev_address_months_count, days_since_request, foreign_request, session_length_in_minutes have a high positive right skewness in the distribution of the data.
- Columns intended_balcon_amount, current_address_months_count, zip_count_4w, bank_branch_count_8w, has_other_cards, proposed_credit_limit, device_distinct_emails_8w have a moderately low positive right skewd data distribution.
- Columns income, velocity_4w, email_is_free, phone_mobile_valid, keep_alive_session have a low negatively left skewed data distribution.

After checking the kurtosis values it was observed –

Kurtosis helps us understand how much data values are present in the tails of the data. It helps us identify any extreme fluctuations between values of features. Kurtosis can be of three types let's see which columns the data for the above dataset falls under :

– Columns income, name_email_similarity, current_address_months_count, customer_age, zip_count_4w, velocity_6h, velocity_24h, velocity_4w, date_of_birth_distinct_emails_4w, credit_risk_score, email_is_free, phone_home_valid, bank_months_count, has_other_cards, proposed_credit_limit, keep_alive_session, device_fraud_count, month have a leptokurtic distribution with kurtosis less than 3.

– Columns fraud_bool, prev_address_months_count, days_since_request, intended_balcon_amount, bank_branch_count_8w, phone_mobile_valid, foreign_request, session_length_in_minutes are platykurtic in nature with their kurtosis being greater than 3.

Taking into consideration that some of these columns have binary classification, and binned into certain numerical categories, the distribution, skewness and kurtosis for those columns can be subject to their individual unique classes.

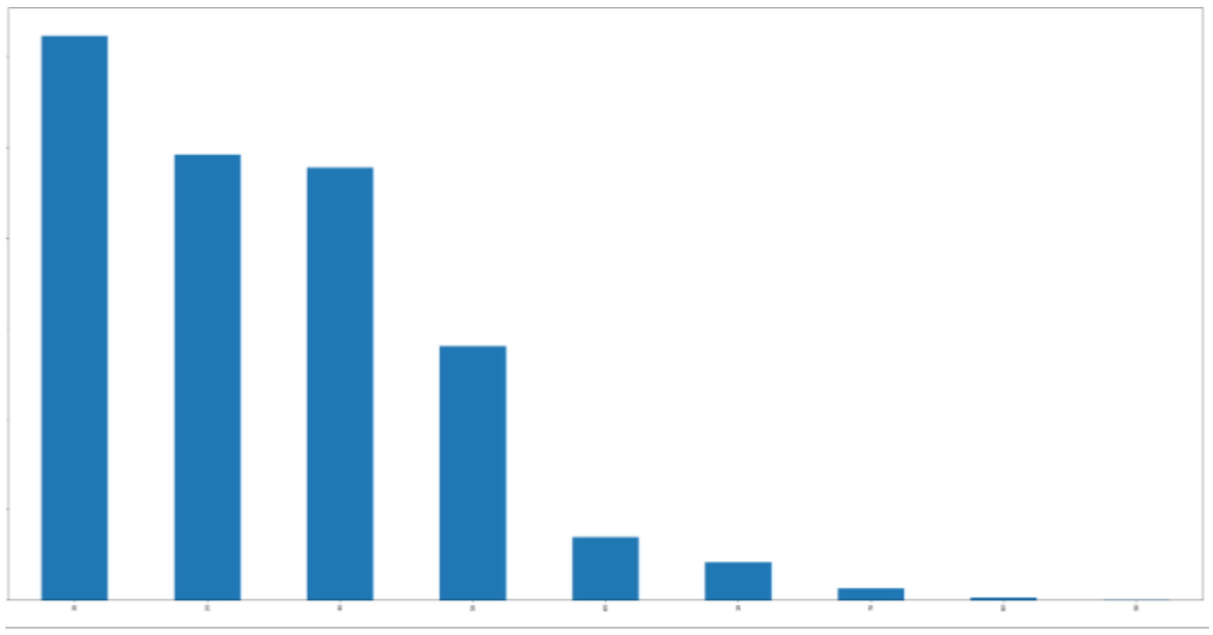


Fig: Distribution of customer age

7. How does your solution affect the problem in the domain or business? What recommendations would you make, and with what level of confidence?

Our problem statement signifies to a classification type of problem, where our sole purpose is that the model should be able to correctly classify a particular customer in a positive or negative classes respectively. If our model is deployed for production, the solution built by us would return concise and precise results with 95% confidence or accuracy, and correctly predict customers into respectice classes. The recall value suggests that how well our model is able to classify class one as positive and negative as class zero , and hence our recall score should be higher while False positive rate should be lesser as misidentified classes can result into the bank losing it's potential customers as well as maybe a person or fraudulent consumer getting access to Bank's details if it is still identified as positive class.

Also, the trained model is evaluated on the test data to measure its performance and identify areas of improvement. After performing smote analysis we started with random forest model, again we have selected some features from the original data and created the dataframe and from that dataframe we have taken the sample.

In return, the solution created for this project has the potential to significantly affect the financial industry's fraud detection field. Financial companies can avoid substantial losses, safeguard consumer data, and preserve their reputation by correctly anticipating whether a transaction is fraudulent or not.

The following suggestions are made to enhance model performance and lower the incidence of false positives:

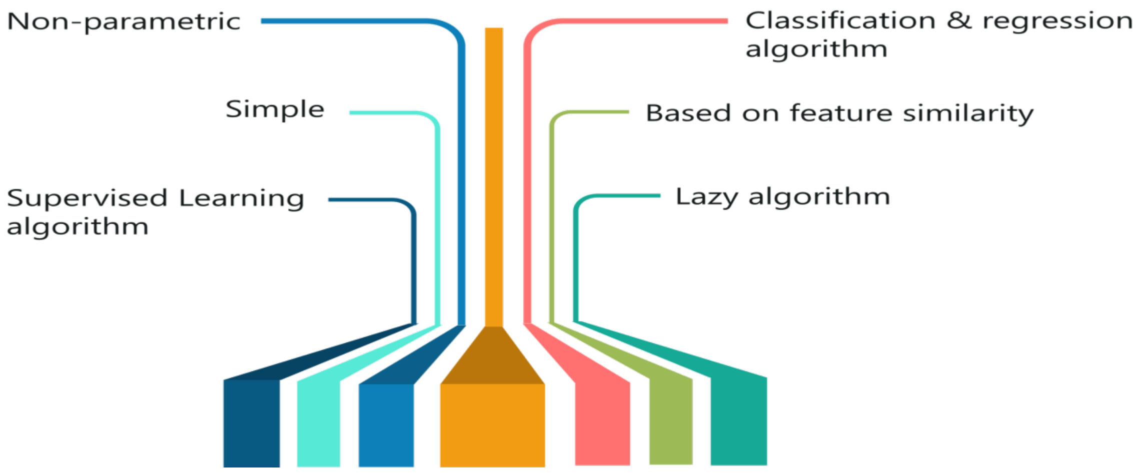
1. Gathering more information: Because the dataset utilised for this research is unbalanced, gathering more information, particularly information on fraudulent transactions, could improve the model's performance.

2.Using an ensemble of models: By merging the results of various models, an ensemble of models can increase the accuracy and resilience of the final forecast.

3.Tuning hyperparameters: By fine-tuning the hyperparameters using methods like grid search, random search, or Bayesian optimization, the performance of the model can be further enhanced.

We can recommend using this model in the financial industry's fraud detection sector with a high degree of confidence. To create a thorough fraud detection approach, it is crucial to keep in mind that the model shouldn't be utilised in isolation and should be supported by additional measures including human review, process improvement, and compliance with rules.

8. Limitations What are the limitations of your solution? Where does your model fall short in the real world? What can you do to enhance the solution?


Non-parametric
Classification & regression algorithm
Simple
Based on feature similarity
Supervised Learning algorithm
Lazy algorithm

The following are some drawbacks of applying the k-nearest neighbours algorithm, as the figure implies-

1. Given that it stores all of the training data, associated computation costs are substantial and hence in turn a large memory storage is necessary, failing to adhere this would lead into increased system complexity.

2. Need to calculate K's value is a important task before fitting the training values into the model. It is a crucial step that has to be kept into consideration before model building.

3. In the case of a high value of N, prediction is slow and is sensitive to little details. In large datasets, the cost of calculating the distance between the new point and each existing points is huge which degrades the performance of the algorithm.

4. Predictions are comparatively slower and sensitive to minute features when N is high. In large datasets, the algorithm's speed suffers due to the high cost of computing the distance between each new point and each current point.

5. Because it becomes challenging for the algorithm to determine the distance in each dimension when there are many dimensions, the KNN algorithm does not perform well with high dimensional data.

6. Need for feature scaling: Prior to using the KNN method on any dataset, feature scaling (also known as standardisation and normalisation or min-max scaling) is required. If we don't, KNN might produce inaccurate forecasts which might result in producing incorrectly classified evaluation metrics.

7. KNN is susceptible to noise in the dataset. It is also sensitive to missing values and outliers. We should always make it a point to impute or treat missing values and outliers before actually building and fitting the model with the data points, respectively.

9. What have we learned from the process? What would we do differently next time?

Machine learning includes creating algorithms and statistical models that let computers learn from experience and become better at completing tasks. These models and algorithms are built to learn from data and forecast or decide without being given specific instructions.

Machine learning comes in a variety of forms, such as reinforcement learning, unsupervised learning, and supervised learning. In contrast to unsupervised learning, which uses unlabeled data, supervised learning involves training a model using labelled data.

We come to the conclusion that the dataset was in a complete state, but with few limitations on categorical data and still lacks some feature vectors. Since the remaining dimensions of the subspace of the input space that we were trying to generalize are unknown, none of the classifiers could do better than 95% (KNN). More feature vectors must be produced in the future if similar research are carried out to produce the dataset utilised in this study so that the classifiers can have a deeper understanding of the issue at hand with an optimum value of 'K'.

A lot many classifiers can be included to segregate between two classes with a good number of accuracy and recall values. Support vector machines, Catboost models are powerful algorithms but due to their sensitivity towards large datasets they were ruled out from final consideration.\

References-

1. https://www.kaggle.com/datasets/sgpjesus/bank-account-fraud-dataset-neurips-2022/code?select=Base.csv

2. https://datagy.io/python-support-vector-machines/

3. https://www.educba.com/nearest-neighbors-algorithm/

4. https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html

5. https://www.netguardians.ch/banking-fraud/

6. http://psychologyandeducation.net/pae/index.php/pae/article/view/2518

7. https://towardsdatascience.com/5-anomaly-detection-algorithms-every-data-scientist-should-know-b36c3605ea16

8. https://serokell.io/blog/anomaly-detection-in-machine-learning

------------------------------- **Final Report Presentation Checkpoint**---