Group 2
Poorvi Rajendra Raut
Anushree Rawat
Amna Shamshad
Priyanka De

Project Overview

1) INTRODUCTION
2) DATA SET ANALYSIS
3) PREPROCESSING DATA
4) EDA (EXPLORATORY DATA ANALYSIS)
5) FEATURE ENCODING AND CORRELATION
6) BALANCING DATA
7) MACHINE LEARNING MODELS
8) CONCLUSION
9) FUTURE WORK

INTRODUCTION

- Credit card scam is a type of financial crime that involves the unauthorized use of a credit or debit card to make purchases or withdraw funds. It is a significant problem for banks, credit card companies, merchants, and consumers, with billions of dollars lost annually to fraudsters.
- One of the most effective ways to detect credit card scam is through the use of machine learning algorithms.
- The ultimate goal is to create a system for detecting credit card scam that can precisely identify fraudulent transactions.

DATA SET ANALYSIS

- Source of Dataset: https://www.kaggle.com/datasets/kartik2112/fraud-detection?resource=download
- Our dataset provides all information containing all transaction details to determine a transaction is fraud or not fraud
- Here our dataset already been divided into training and testing sets, but we have merged it together and now entire dataset contains 1,852,394 rows and 22 columns.
- Our goal is to determine our target variable : is_fraud as fraud transaction (1) or not fraud transaction (0)

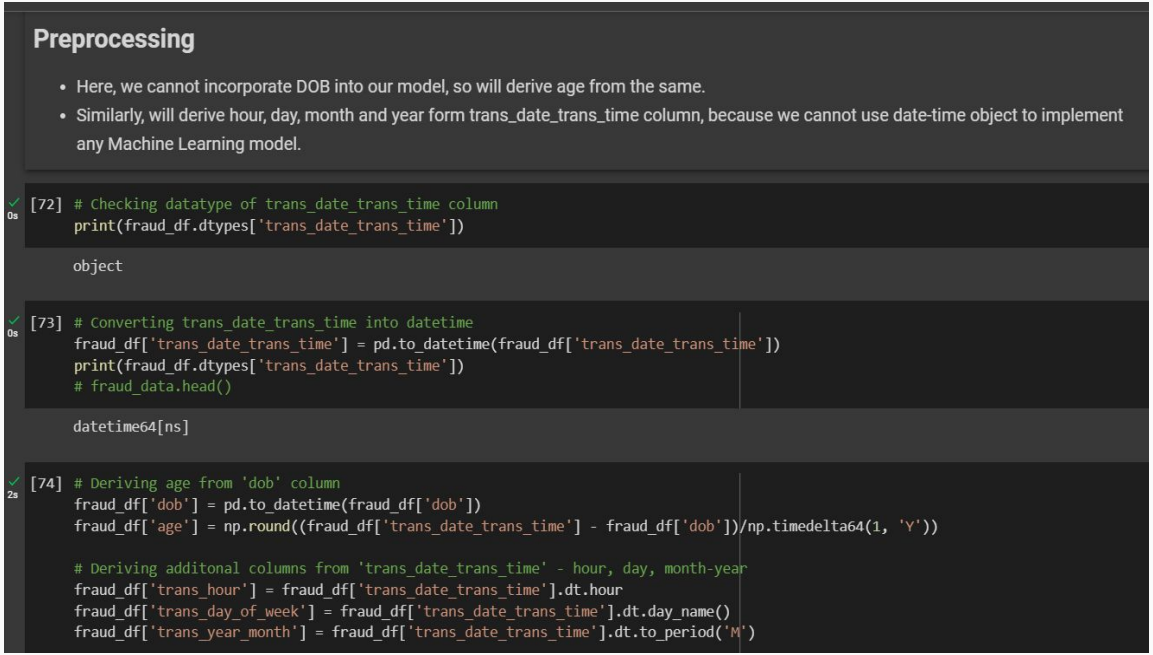
DATA SET OVERVIEW

Variable	Description
trans_date_trans_time	Transaction timestamp
cc_num	Unique credit card number
merchant	Merchant name
category	Transaction category
amt	Transaction amount
merch_long	Longitude of merchant
merch_lat	Latitude of merchant
is_fraud	Nature of transaction
trans_num	Transaction number
unix_time	Time in unix format

Variable	Description
first	First name of cardholder
last	First name of cardholder
gender	Sex of cardholder
street	Transaction address
city	Transaction city
state	Transaction state
zip	Transaction zip code
job	Job of the cardholder
dob	Date of birth of cardholder
city_pop	Population of the city

PRE-PROCESSING DATA

- In dataset we have DOB column which is of object type that cannot be incorporated directly into our model, so will derive age from the same.
- Similarly, we have derived hour, day and month-year from trans_date_trans_time column, because we cannot use date-time object to implement any Machine Learning model.
- After checking missing values, scaling has been done on the entire dataset too.



Preprocessing

- Here, we cannot incorporate DOB into our model, so will derive age from the same.
- Similarly, will derive hour, day, month and year form trans_date_trans_time column, because we cannot use date-time object to implement any Machine Learning model.

```
[72] # Checking datatype of trans_date_trans_time column
     print(fraud_df.dtypes['trans_date_trans_time'])

     object
```

```
[73] # Converting trans_date_trans_time into datetime
     fraud_df['trans_date_trans_time'] = pd.to_datetime(fraud_df['trans_date_trans_time'])
     print(fraud_df.dtypes['trans_date_trans_time'])
     # fraud_data.head()

     datetime64[ns]
```

```
[74] # Deriving age from 'dob' column
     fraud_df['dob'] = pd.to_datetime(fraud_df['dob'])
     fraud_df['age'] = np.round((fraud_df['trans_date_trans_time'] - fraud_df['dob'])/np.timedelta64(1, 'Y'))

     # Deriving additonal columns from 'trans_date_trans_time' - hour, day, month-year
     fraud_df['trans_hour'] = fraud_df['trans_date_trans_time'].dt.hour
     fraud_df['trans_day_of_week'] = fraud_df['trans_date_trans_time'].dt.day_name()
     fraud_df['trans_year_month'] = fraud_df['trans_date_trans_time'].dt.to_period('M')
```

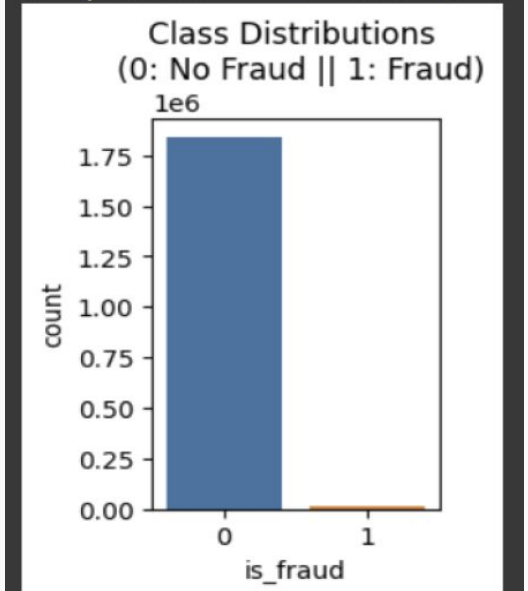
EDA (Exploratory Data Analysis)

- Starting EDA from target variable, which is is_fraud column according to our dataset
- Here, we can see that dataset is highly skewed, which needs to be fixed. Otherwise while implementing ML models it will have biases for high volume data

```
0    1842743
1       9651
Name: is_fraud, dtype: int64
```

```
No Fraud 99.48 % of the dataset
Fraud 0.52 % of the dataset

Text(0.5, 1.0, 'Class Distributions
```



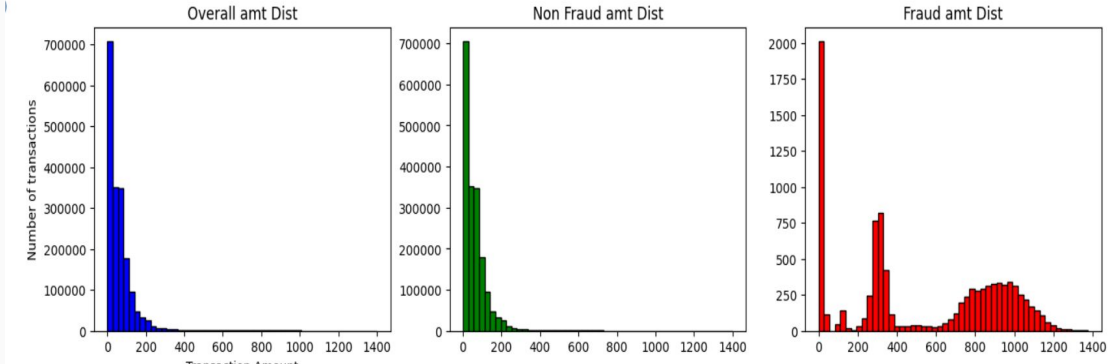
EDA

- **Data Summarization:**
  Here, the mean/average of the 'fraud distribution' is way higher than the 'non-fraud distribution'.

  That means the amount lost in a fraud transaction is very high. Which is a usual behaviour looking at the real-life scenarios.

- **Data Visualization:**
  The histogram plot of the fraud and non-fraud distribution shows that the Transaction amount is very high, for fraud distributions

	Distribution	Overall Distribution	Non-Fraud Distribution	Fraud Distribution
0	count	1.852394e+06	1.842743e+06	9651.000000
1	mean	7.006357e+01	6.765128e+01	530.661412
2	std	1.592540e+02	1.535481e+02	391.028873
3	min	1.000000e+00	1.000000e+00	1.060000
4	25%	9.640000e+00	9.610000e+00	240.075000
5	50%	4.745000e+01	4.724000e+01	390.000000
6	75%	8.310000e+01	8.256000e+01	902.365000
7	max	2.894890e+04	2.894890e+04	1376.040000

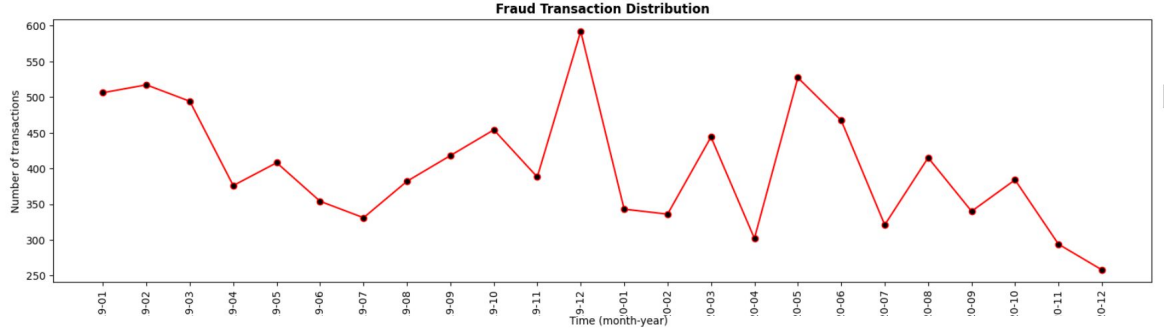


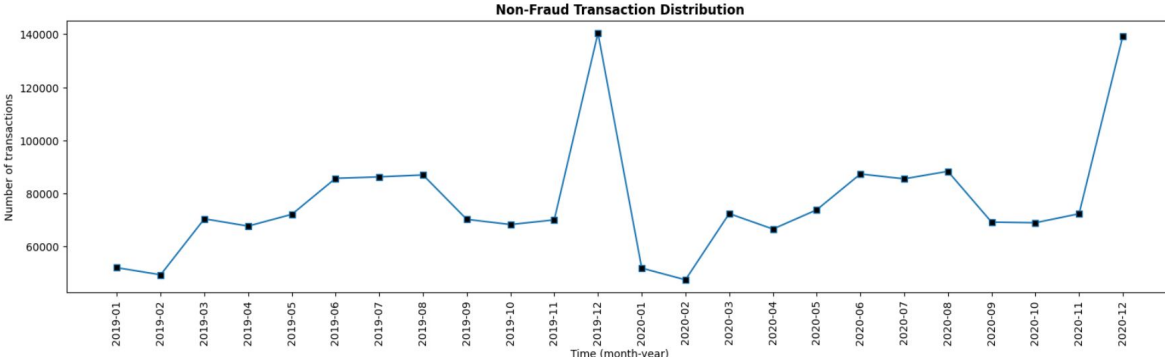
EDA

- **Correlation Analysis:**

  Correlation between time(year_month) vs number of transactions examined. Red for fraud and blue for non-fraud.

  Shows high fraud transactions in 2019-12 and reduced fraud transactions in 2020-12





EDA

- **Correlation Analysis:**

  Gender vs target variable.

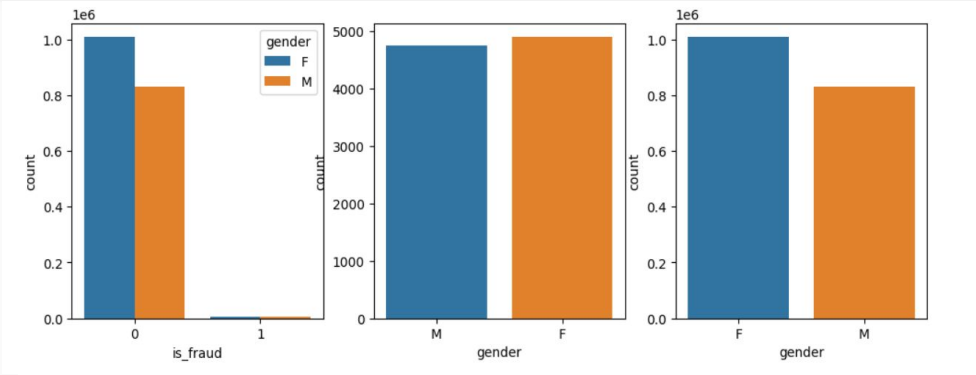
  Shows high fraud transactions for Female, compared to males.

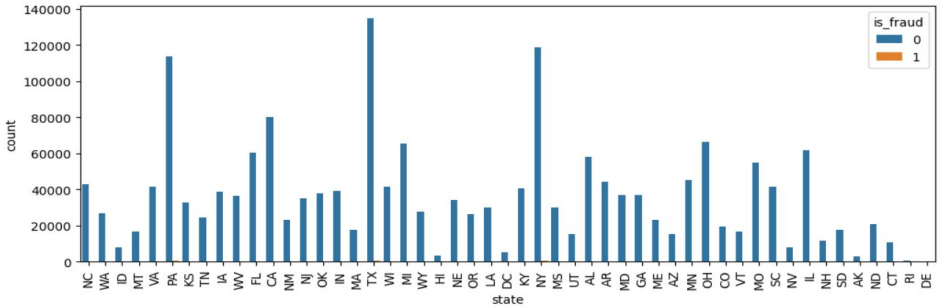
- **Correlation Analysis:**

  State vs target variable.

  Looking at the graph most number of transactions are noted in 3 states (PA, TX and NY).

  *similarly we have explored all the features.





Feature Encoding and Correlation

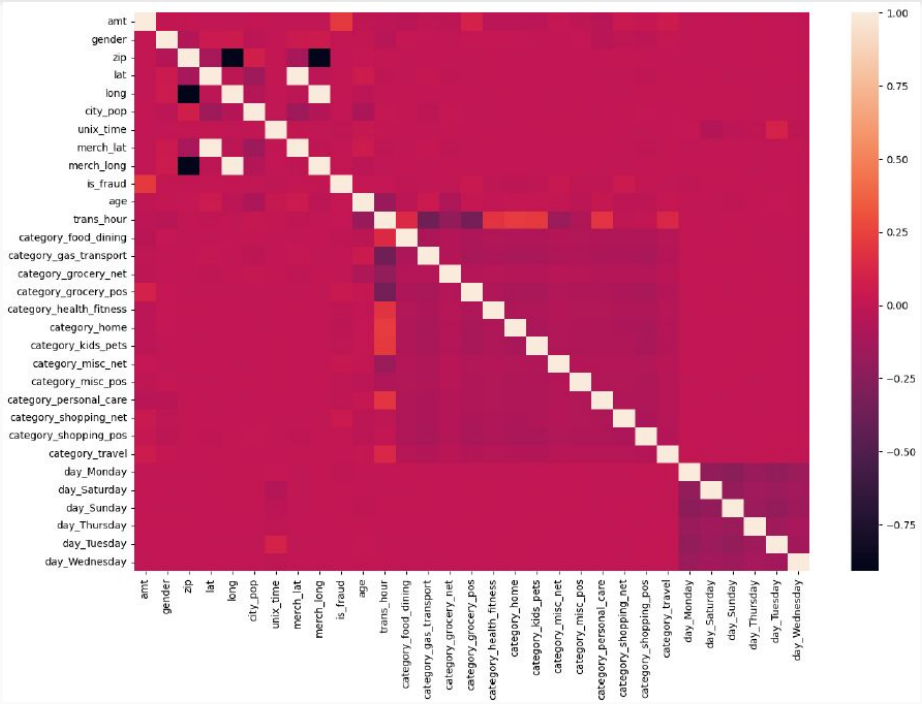
- One hot encoding

One hot encoding is a technique that we use to represent categorical variables as numerical values in a machine learning model.

- Direct mapping

In this method, each unique value or category in a feature is assigned a unique numerical code, also known as an encoding.

- Here, we have dropped some of the unnecessary and encoded columns.



Balancing Data

We have implemented Logistic Regression model without balancing the skewness in the data and we got following result:

	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
Logistic Regression - Without Balancing		0.994476	0.994398	0.994398	0.991983	0.089202	0.006467

We got least precision and recall, which shows less precision and completeness to predict true positive of the model

Balancing Data

We have tried three different sampling method:

- Random Under Sampling
- Random Over Sampling
- SMOTE method

And finalized SMOTE (Synthetic Minority Oversampling Technique) method to balance our dataset

```
# Checking Target/Class variable frequency distribution
print(y_sm.value_counts())
```

```
0    1842743
1    1842743
Name: is_fraud, dtype: int64
```

Machine Learning Models

Logistic Regression without balancing

- In Logistic Regression, we model the probability of an event occurring given a set of independent input variables.
- When training a logistic regression model without balancing the classes, it means that we are not adjusting the training data to address any class imbalance that may exist. This can lead to biased results, where the model may over-predict the majority class and under-predict the minority class.
- With an unbalanced dataset which has bias, we get an accuracy of 99.4%

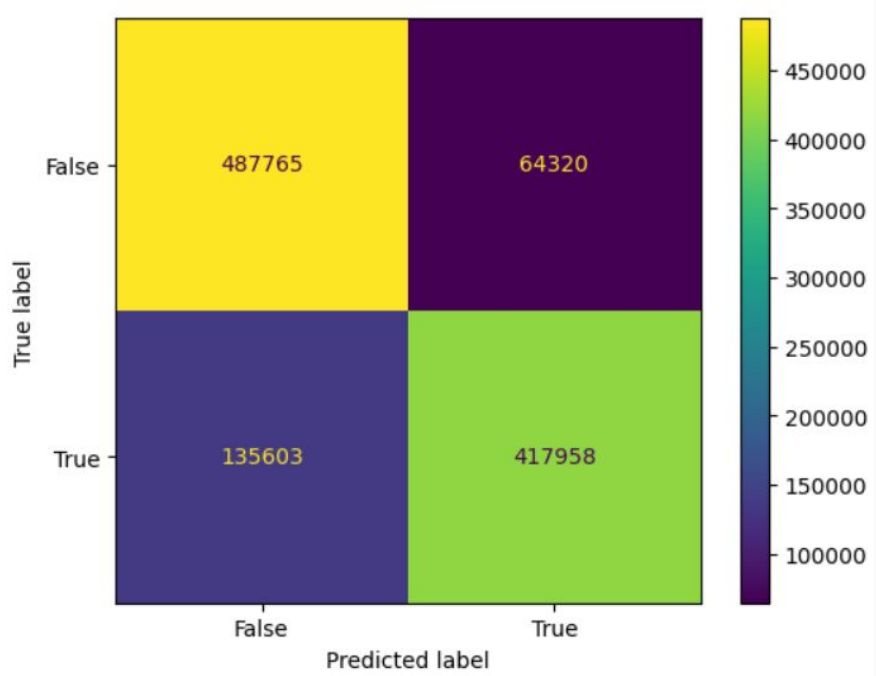
	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
0	Logistic Regression - Without Balancing	0.994466	0.994378	0.994378	0.991936	0.039604	0.002723

Logistic Regression with balancing using SMOTE

- When training a logistic regression model, it is important to address any class imbalance that may exist to prevent biased results. SMOTE or Synthetic Minority Oversampling Technique is an oversampling technique.

- In oversampling technique, the minority data is duplicated from the minority data. While it increases the number of data, it does not give any new information or variation to the machine learning model.

- With sampling strategy as “minority”, we get 81.9% accuracy



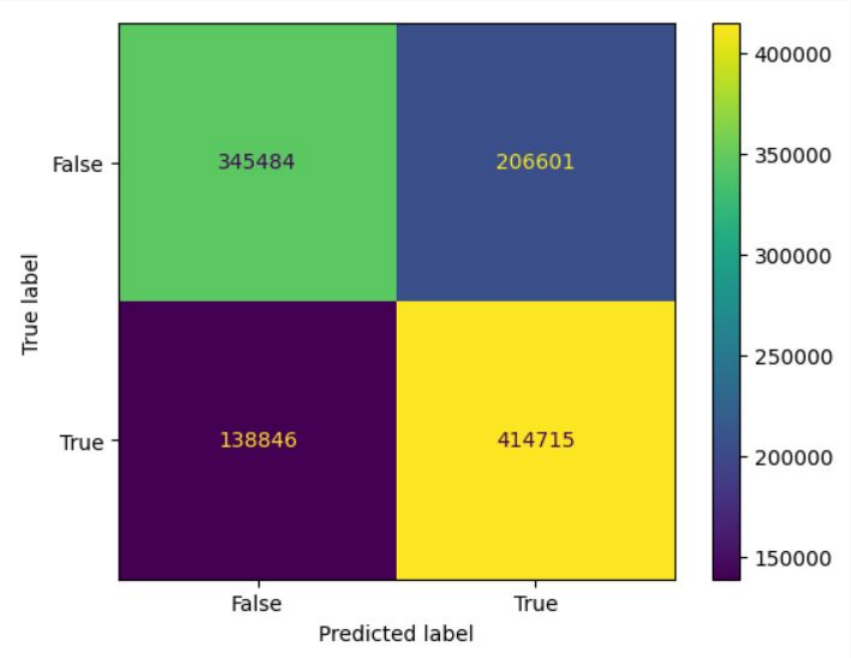
	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
0	Logistic Regression - SMOTE	0.81937	0.81918	0.81918	0.818441	0.866633	0.755035

Gaussian Naive Bayes

- Naive Bayes is a supervised machine learning algorithm based on Bayes' theorem with “naive” assumption that the features are independent of each other given the class label.

- The Naive Bayes algorithm assumes that the features are continuous and follow a Gaussian (normal) distribution. . In other words, it computes the probability of each class given the features using the distribution.

- we get an accuracy of 68.7%



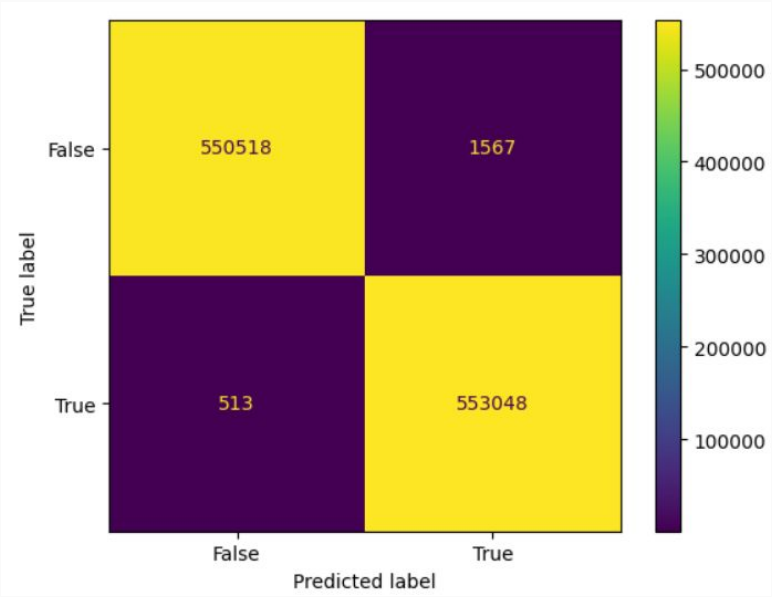
	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
0	Logistic Regression - SMOTE	0.819370	0.819180	0.819180	0.818441	0.866633	0.755035
1	Gaussian Naive Bayes - SMOTE	0.688274	0.687561	0.687561	0.686357	0.667478	0.749177

DECISION TREE: CART (CLASSIFICATION AND REGRESSION TREE)

- CART is a decision tree algorithm that can be used for both classification and regression tasks. The algorithm works by recursively splitting the data into subsets based on the value of a chosen feature, with the goal of maximizing the purity of the resulting subsets.

- In CART, each internal node of the tree represents a feature, and each branch represents a possible value of that feature. The leaves of the tree represent the final decision or prediction.

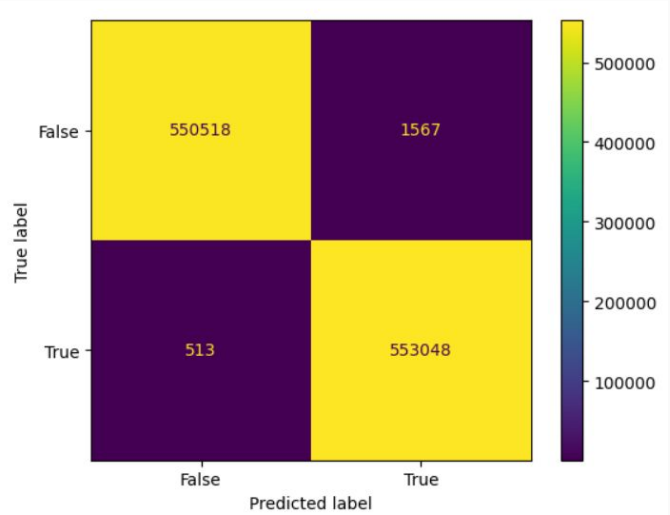
- We get an accuracy of 99.7%



	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
0	Logistic Regression - SMOTE	0.819370	0.819180	0.819180	0.818441	0.866633	0.755035
1	Gaussian Naive Bayes - SMOTE	0.688274	0.687561	0.687561	0.686357	0.667478	0.749177
2	Decision Tree Classifier - SMOTE	1.000000	0.997172	0.997172	0.997172	0.996773	0.997581

RANDOM FOREST

- Random forest consists of a large number of individual decision trees that operate as an ensemble.
- Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes the model's prediction
- With n_estimators as 100 and minimum leaf size as 1, we get 99.8% accuracy.



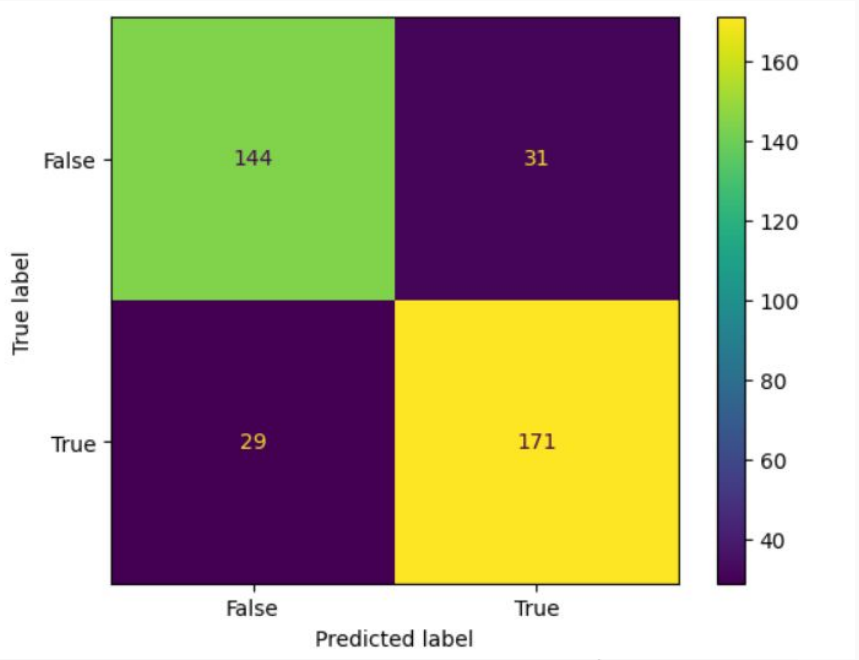
	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
0	Logistic Regression - SMOTE	0.819370	0.819180	0.819180	0.818441	0.866633	0.755035
1	Gaussian Naive Bayes - SMOTE	0.688274	0.687561	0.687561	0.686357	0.667478	0.749177
2	Decision Tree Classifier - SMOTE	1.000000	0.997172	0.997172	0.997172	0.996773	0.997581
3	Random Forest Classifier - SMOTE	0.999999	0.998119	0.998119	0.998119	0.997175	0.999073

K-NEAREST NEIGHBOR (KNN)

- The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems

- The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

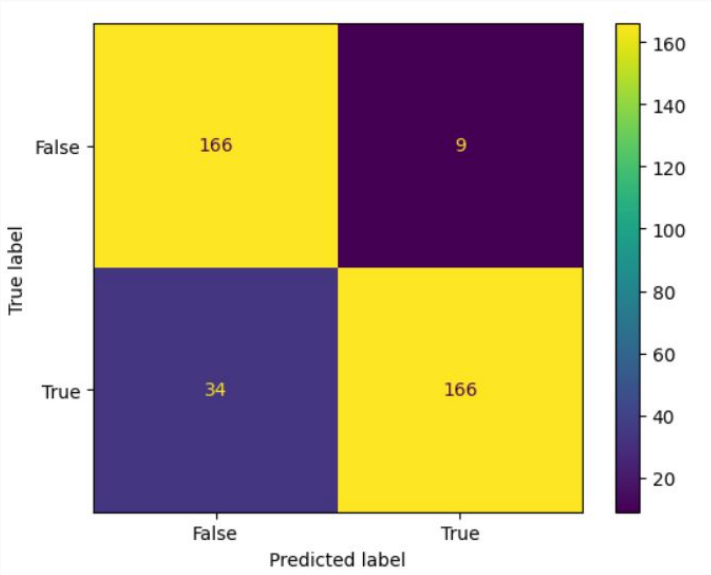
- With number of neighbours as 3 and uniform weights, we get an accuracy of 84.6%



	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
0	KNN - RUS	0.930286	0.84	0.84	0.839938	0.846535	0.855

SUPPORT VECTOR MACHINE

- SVM works by mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable.
- A separator between the categories is found, then the data are transformed in such a way that the separator can be drawn as a hyperplane.
- With trade off value as 1 and degree of polynomial 3, we get 88.6% accuracy



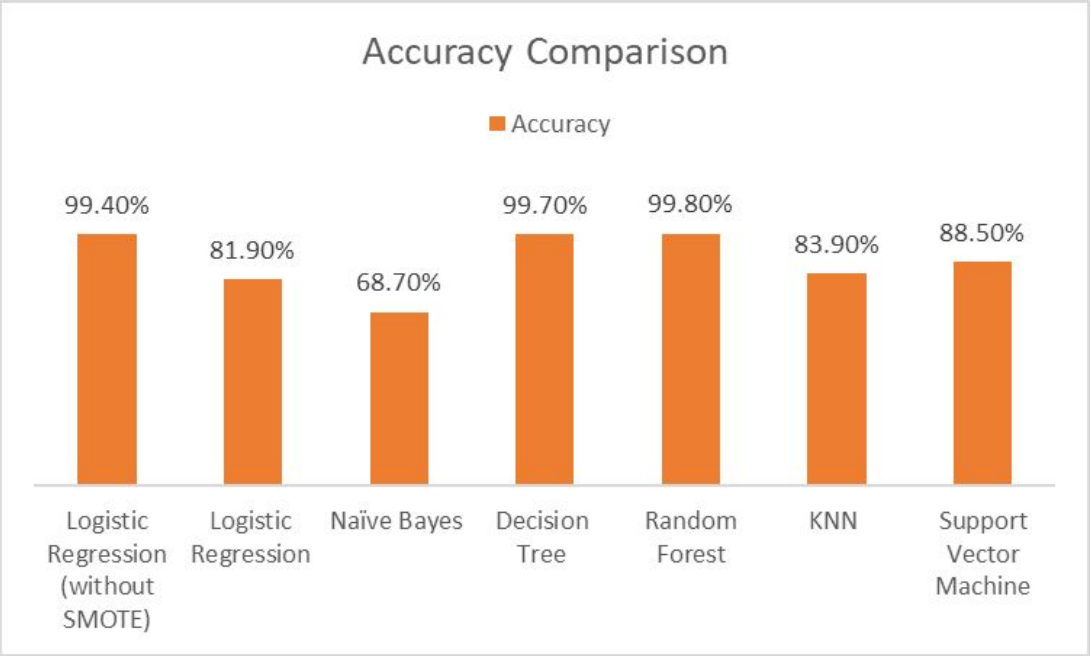
	Model Name	Training Score	Testing Score	Accuracy	F1 Score	Precision	Recall
0	KNN - RUS	0.930286	0.840000	0.840000	0.839938	0.846535	0.855
1	SVM - RUS	0.894857	0.885333	0.885333	0.885333	0.948571	0.830

CONCLUSION

Here, every model is being evaluated using many factors such as accuracy, precision, recall, F1 score etc.

Looking at the figures it can be seen that top most efficient models for our data and distribution are:

- **Decision Tree**
- **Random Forest**



FUTURE WORK

- Looking at the data veracity and its volume, which model/ml algorithm is to select is a very crucial decision. Moreover, cleaning and processing data according to the algorithm is more important and effort taking task then implementing a Machine Learning Model.
- Working on this project, we gain knowledge and some chief insights of the definition: credit card scam detection.
- This project can be incorporated into online business like e-commerce, online banking where chances of fraudulent transaction is very high. And in the further extension we can try to fit more complex data within our model and to predict it accurately.

THANK YOU