CASE STUDY: Credit Card Fraud Detection

- Submitted By : Subrahmanyeswarao Karri

AGENDA

1. Introduction
   - Problem statement
   - Goals
   - Objective
2. Overview of data
3. Insights
4. Model Reflection
5. Conclusion
6. Business Recommendation

- Credit card fraud is an inclusive term for fraud committed using a payment card, such as a credit card or debit card. The purpose may be to obtain goods or services, or to make payment to another account which is controlled by a criminal. In recent times, the number of fraud transactions has increased drastically due to which credit card companies are facing a lot of challenges. For many banks, retaining high profitable customers is the most important business goal. Banking fraud, however, poses a significant threat to this goal.Apart from this, other ways of making fraudulent transactions are as follows:
  - Manipulation or alteration of genuine cards
  - Creation of counterfeit cards
  - Stolen or lost credit cards
  - Fraudulent telemarketing

Problem statement

- Detecting credit card fraud using machine learning is a must in banking industry.
- They need to put proactive monitoring and fraud prevention mechanisms in place.
- Machine learning fraud detection algorithms are way more effective than humans.
- The concept behind using machine learning for fraud detection is that fraudulent transactions have specific features that legitimate transactions.

  - Machine learning helps these institutions-
  - To reduce time-consuming manual reviews.
  - Costly chargebacks and fees.
  - Denial of legitimate transactions.

GOALS

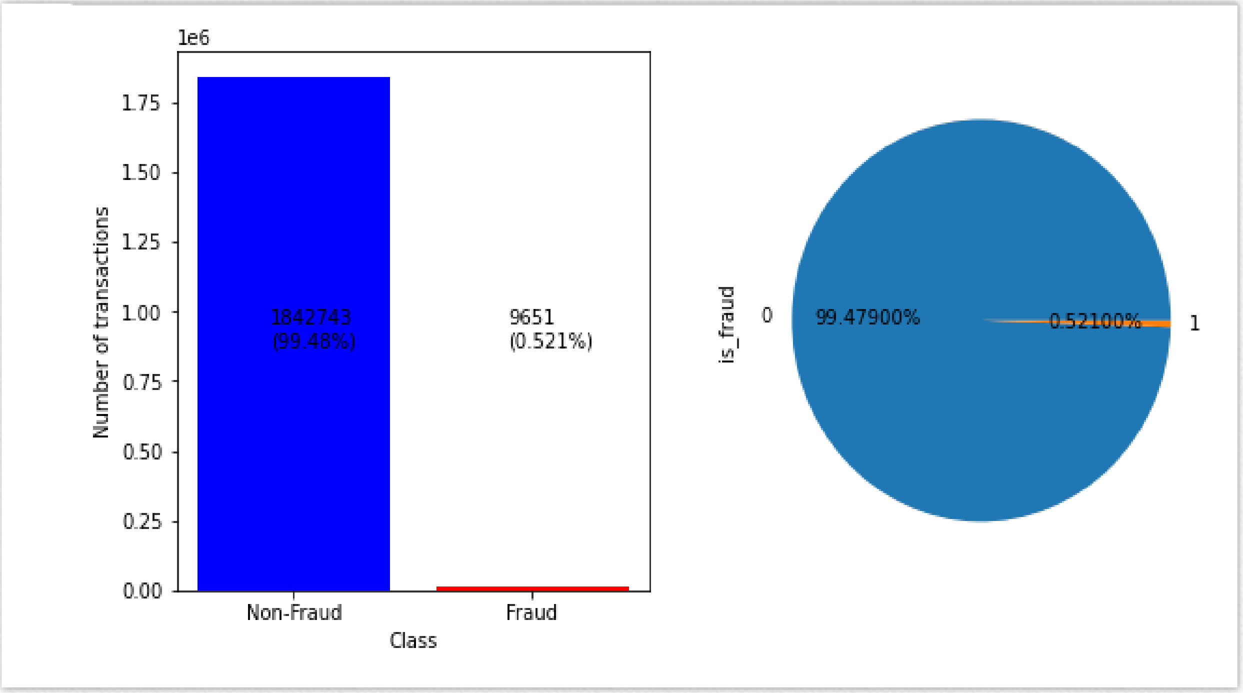
Objective : Analyze the business impact of these fraudulent transactions and recommend the optimal ways that the bank can adopt to mitigate the fraud risks

1. Work as a part of the analytics team working on a fraud detection model and its cost-benefit analysis.

2. Build a machine learning model to detect fraudulent transactions based on the historical transactional data of customers with a pool of merchants

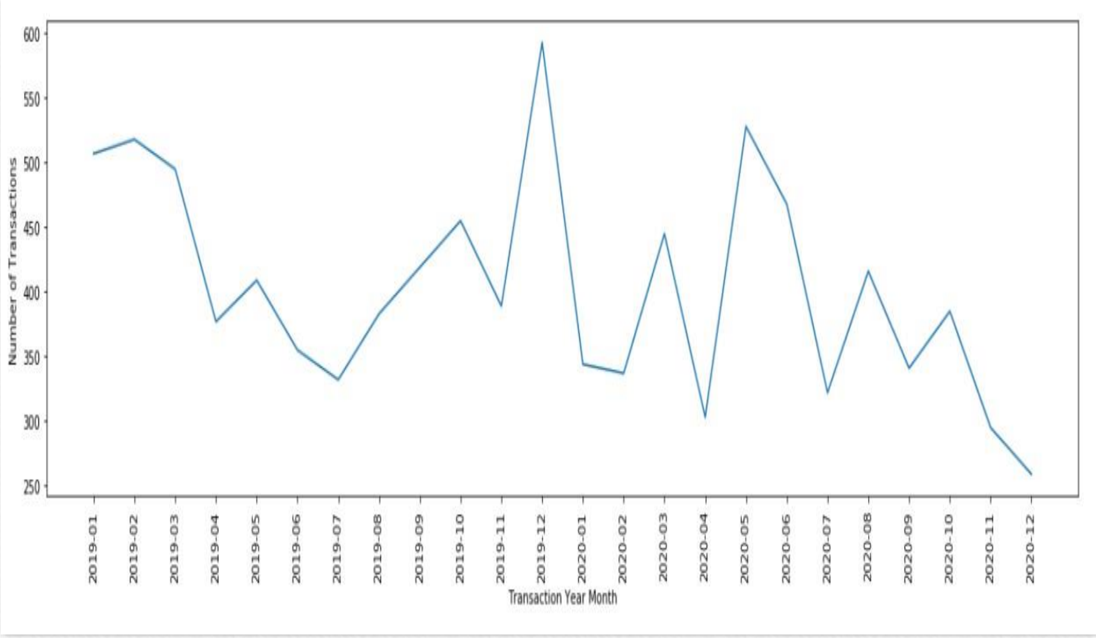
OVERVIEW OF DATA

1. Fraud_data = 9651 transaction (0.521%)
2. 2. Non-Fraud_data= 1842743 transactions(99.48%)

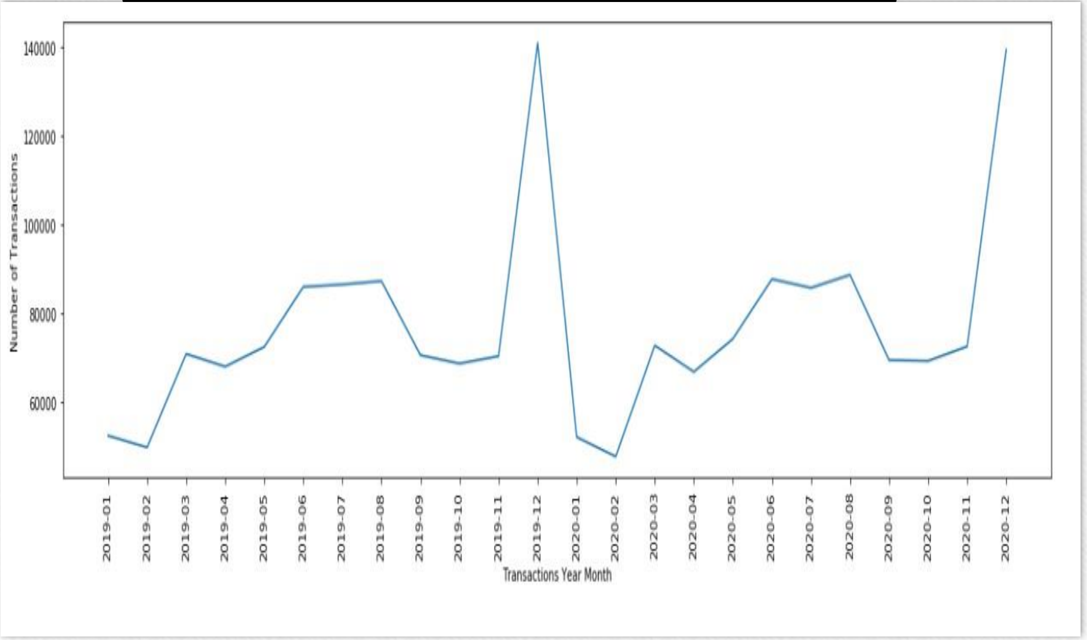


Visual representation of frequency of transactions and fraud_transactions monthly wise

No. of fraud transactions done month wise



No of transactions done month wise

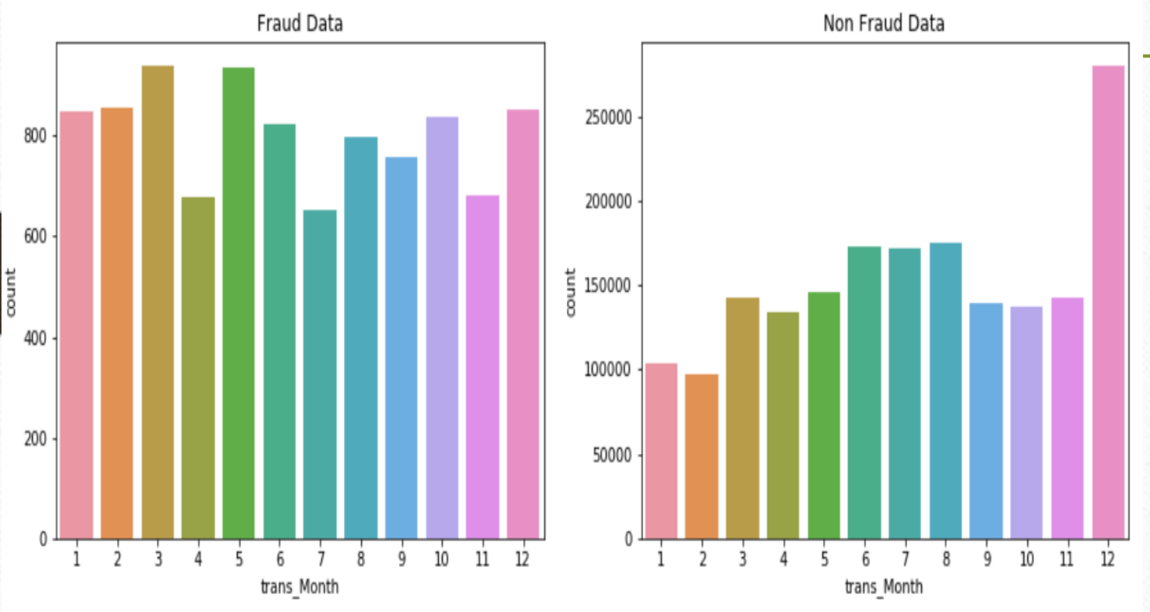


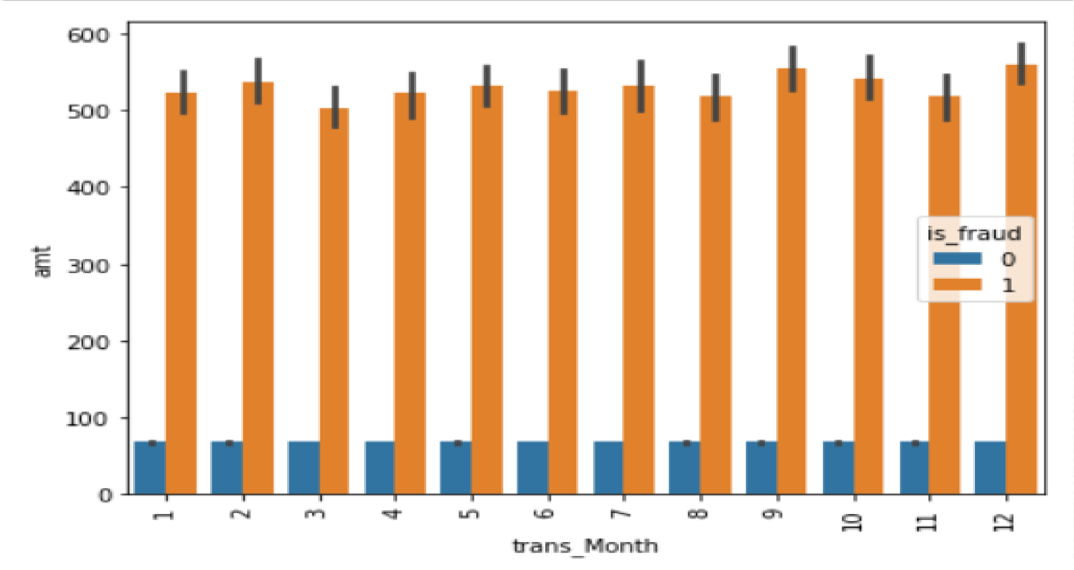
Fraud transactions increased on new year eve and specifically on dec month , later on may, august , october months

INSIGHTS

- Visual Analysis for better understanding based on fraud data(1) and non fraud data(0)
- Plots will make you understand Frauds happening on the credit cards which are issued to the customers of the bank.

1.Analysis based on Fraud transactions in different Months

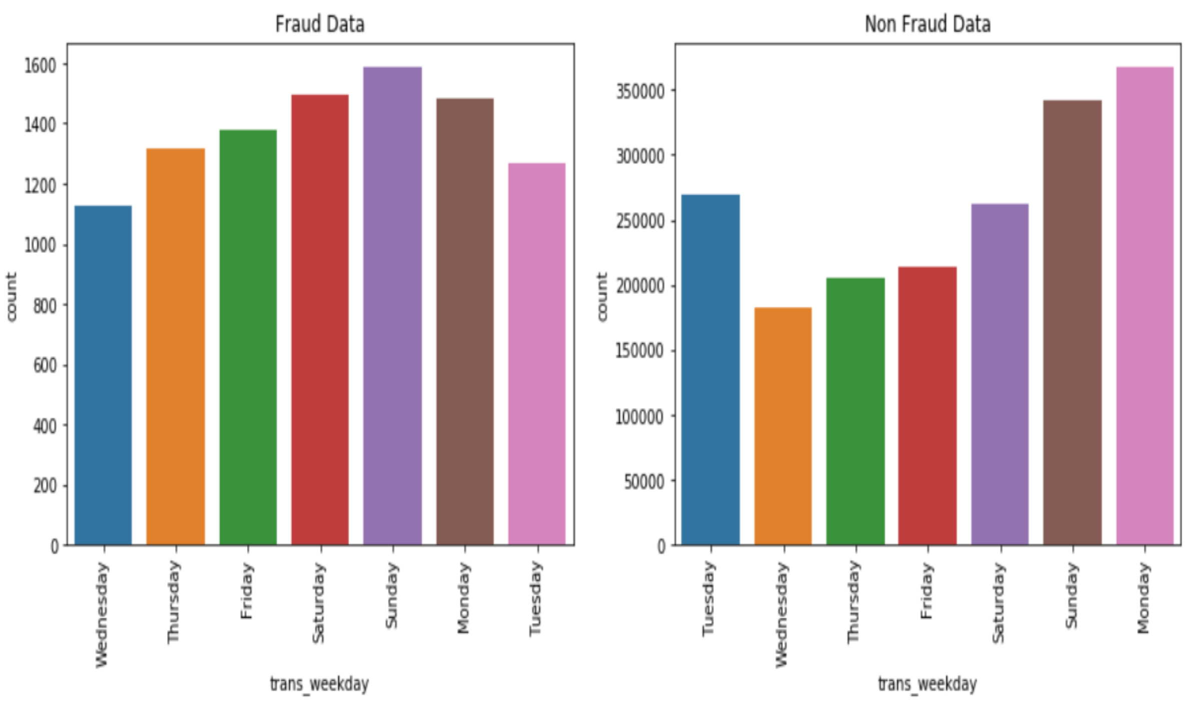


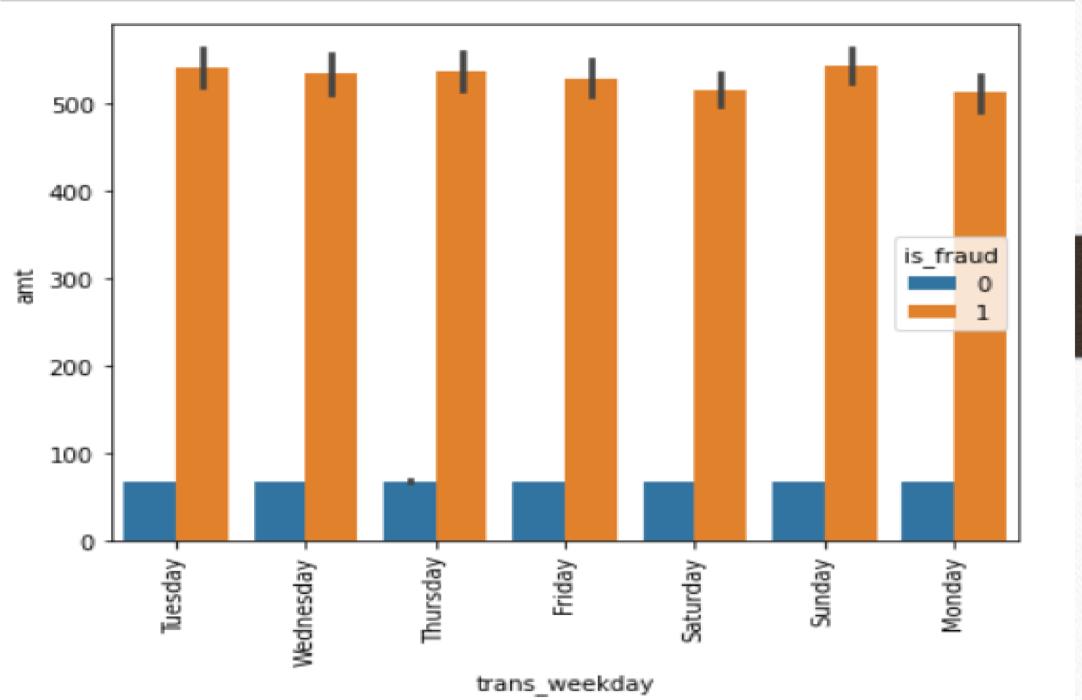


Count of frauds transactions are more in 3rd and 5th month where count of normal transaction is less.

Nearly same amount spend for fraud transactions done through out the Month.

2. Analysis based on Fraud transactions in Weekdays

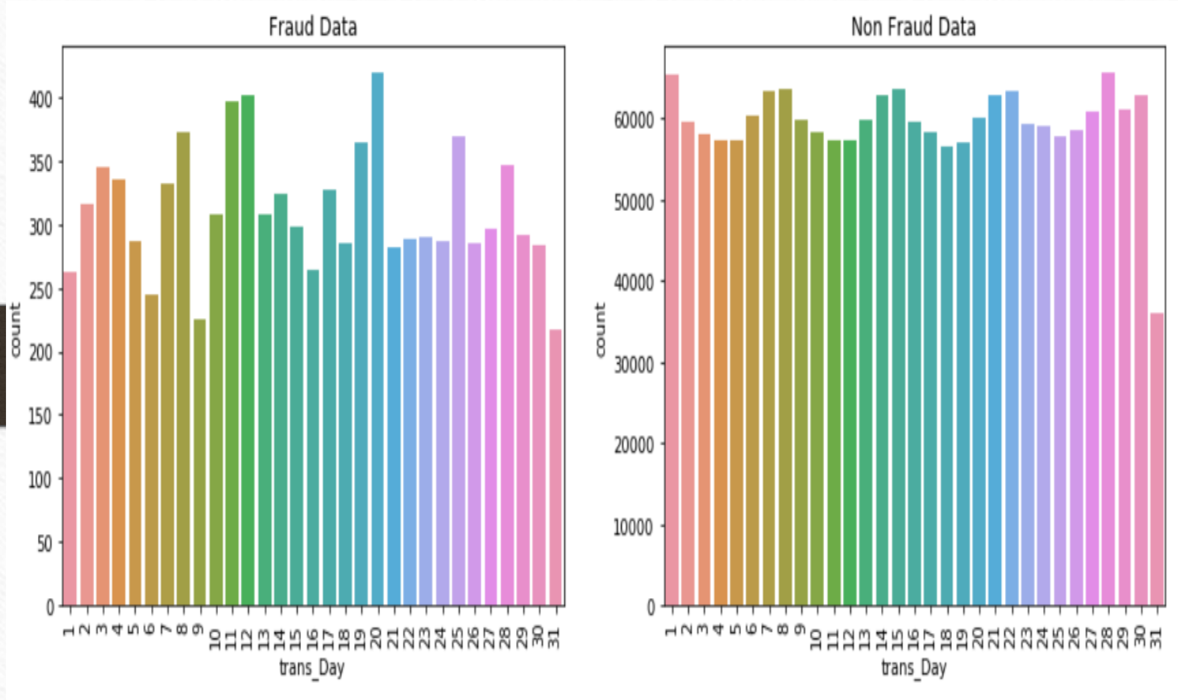


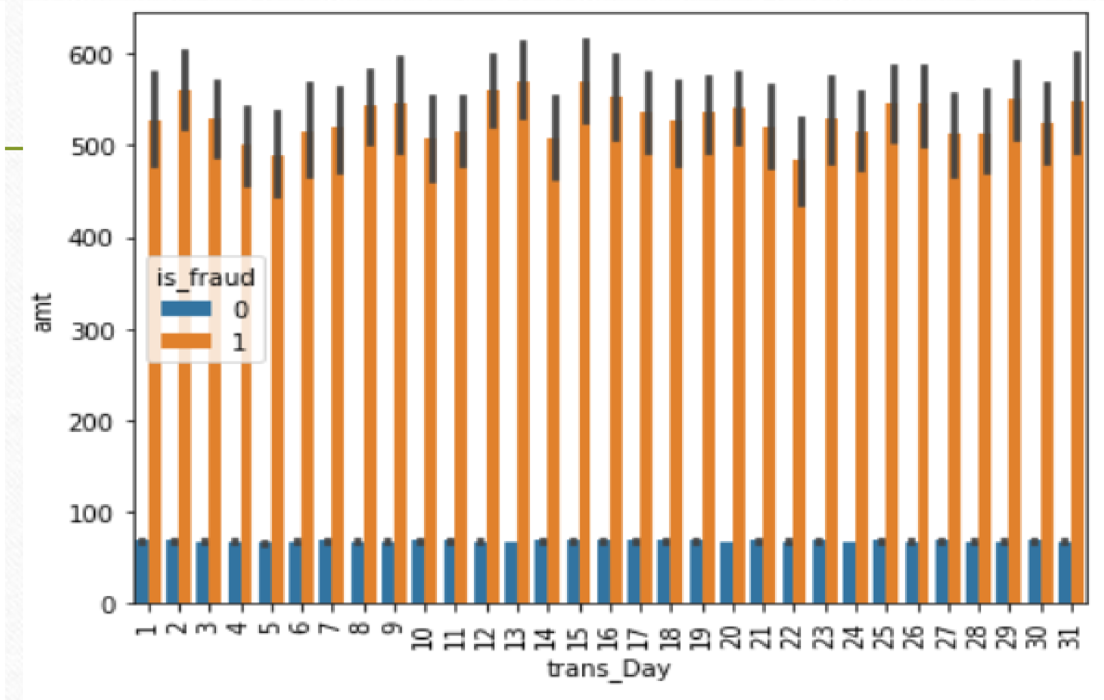


Count of frauds transactions are more on Sunday, Saturday and Monday as compared to the count of normal transaction

Nearly same amount spend for fraud transactions were done through out the Weekday.

3. Analysis based on Fraud transactions in Days

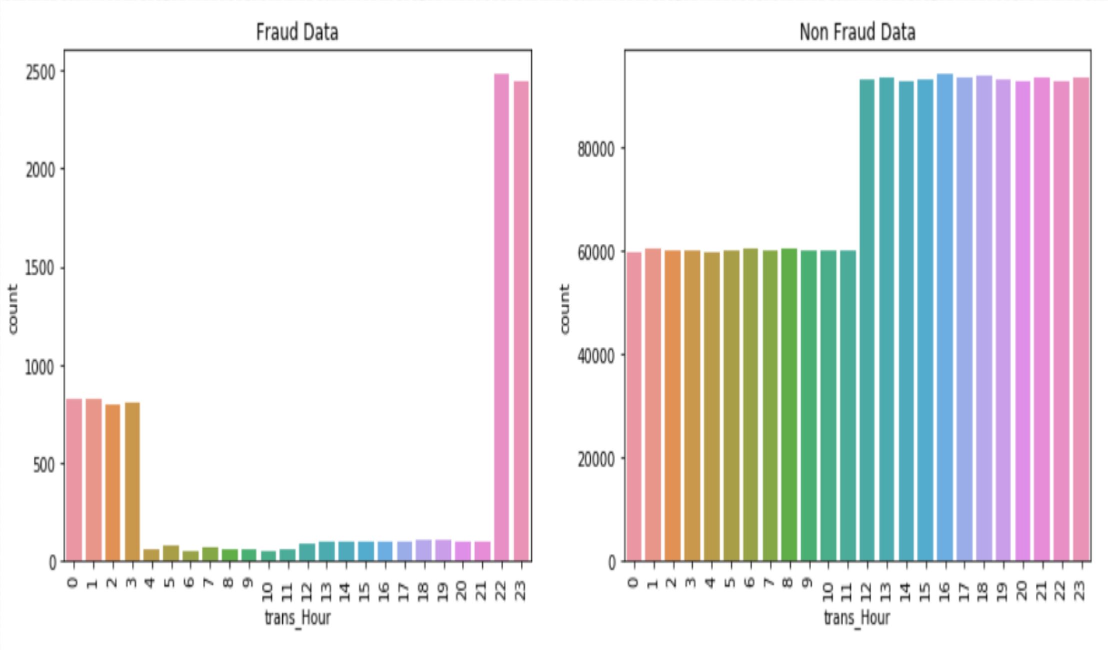


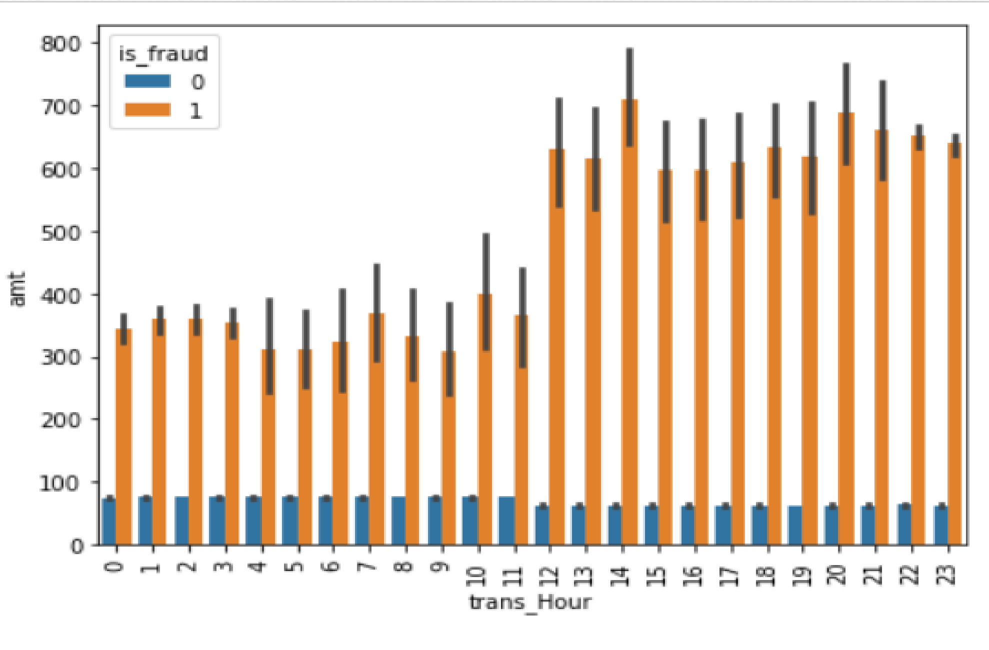


Count of Frauds transactions are more on 20th, 12th and 11th days of month.

The maximum amount spend for fraud transactions are same through out the days of month.

4. Analysis based on Fraud transactions in Hours

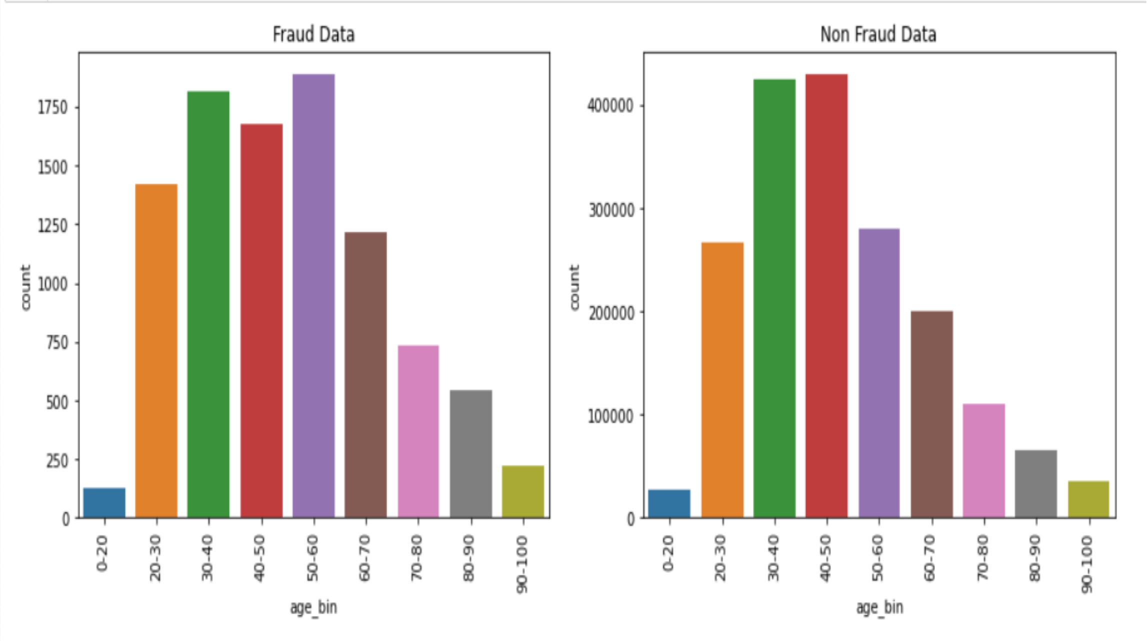




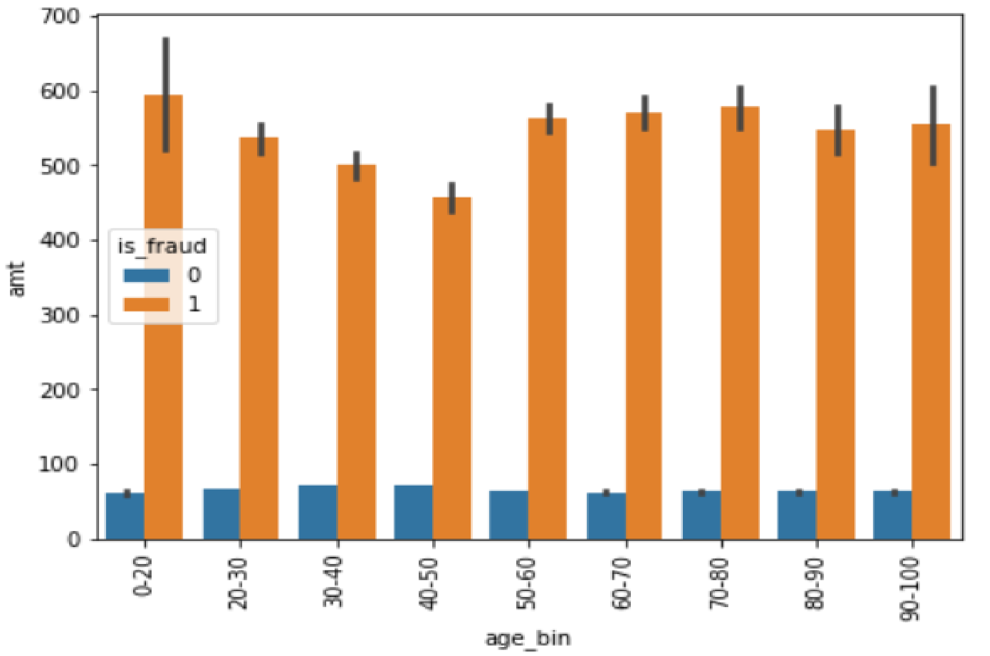
Frauds transactions are done at odd hours of the day i.e. between 22 - 3 Hr

The maximum amount spend for fraud transactions were done mostly between 12 to 23 Hr

5. Analysis based on Fraud transactions in different Age_bins

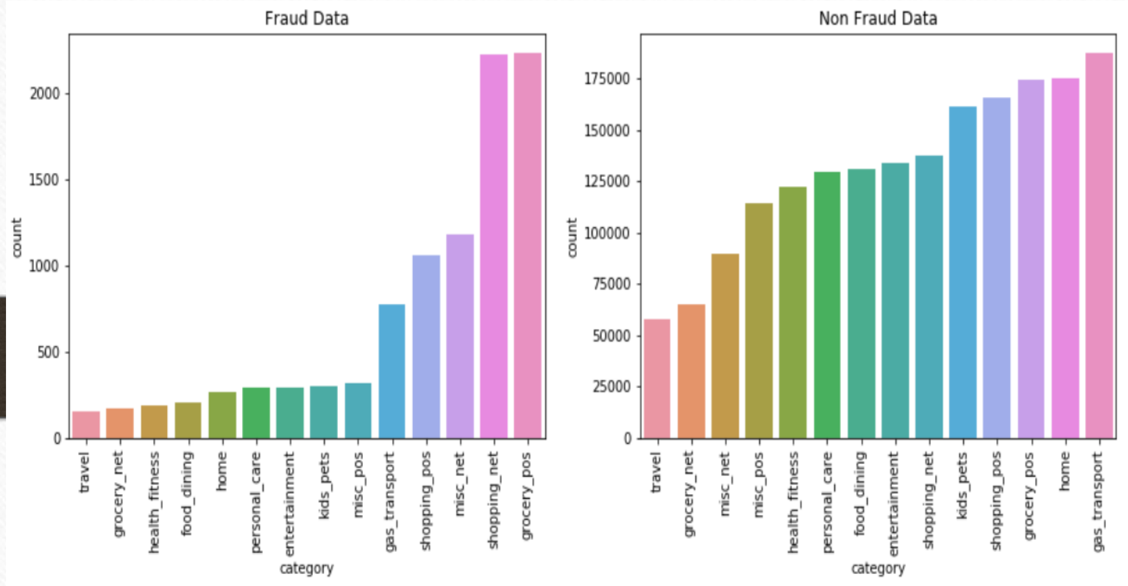


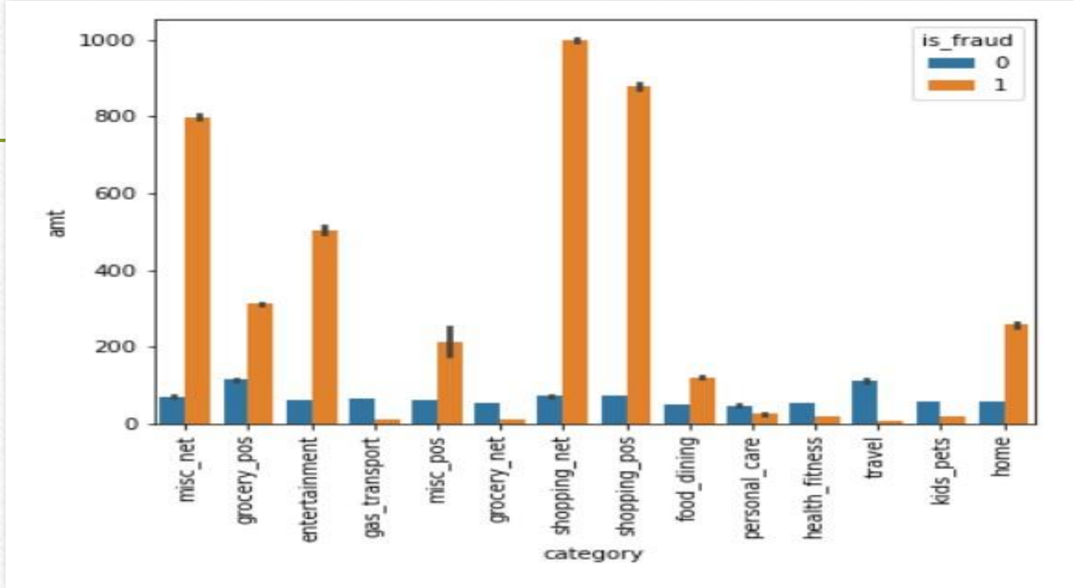
The increased count of fraud transactions are noticed in the age group of 50-60 , age group of 30-40, age group of 20 to 60



The maximum amount spend for fraud transactions belongs to credit card holders "0-20", "60-70" and "70-80" age bin.

6. Analysis based on Fraud transactions in different Categories



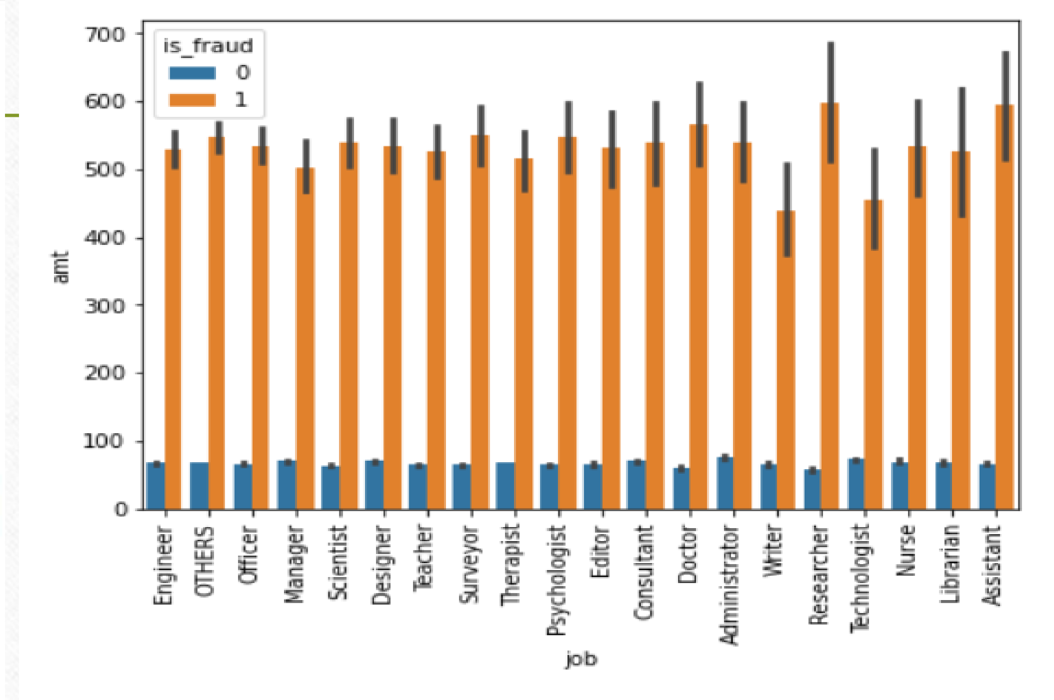


Count of Frauds transactions are done more at grocery_pos, shopping_net, misc_net, shopping_pos, gas_transport Categories

The maximum amount spend were on shopping_net, shopping_pos, misc_net category and entertainment for fraud transactions.

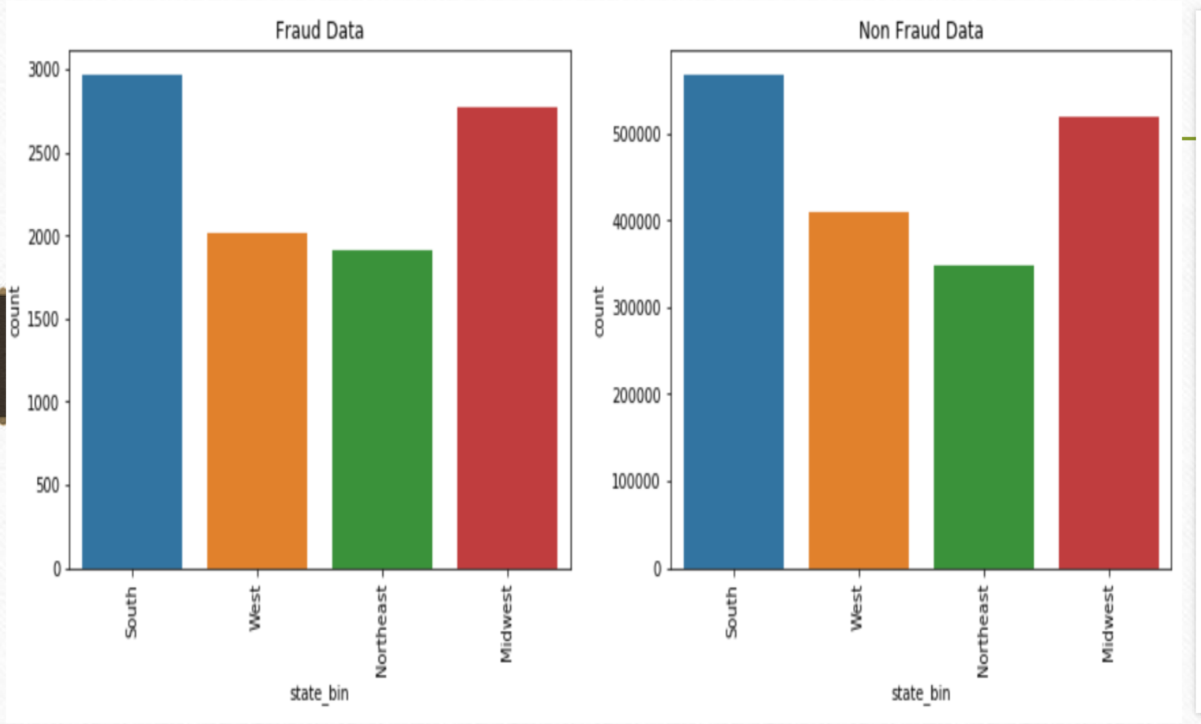
7. Analysis based on Fraud transactions in different Jobs

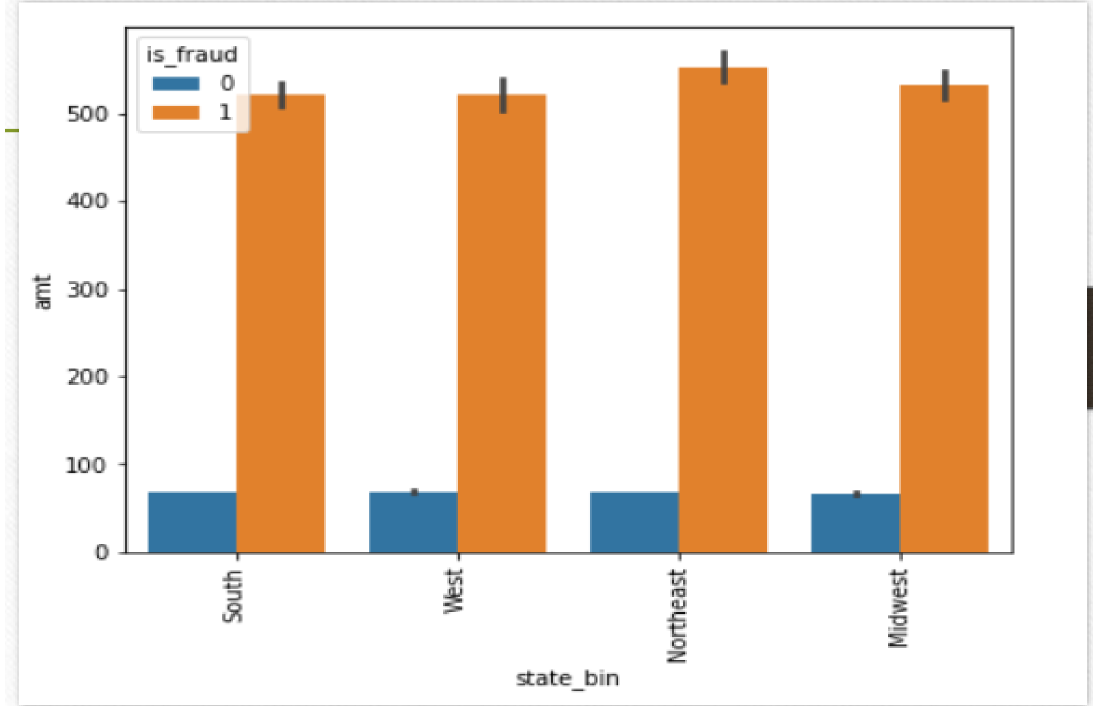
Count of Frauds transactions are done more Engineer, officer, others, scientist Categories



The maximum amount spend for fraud transactions were under the job of credit card holders of Researcher, Assistant.

8. Analysis based on Fraud transactions in different



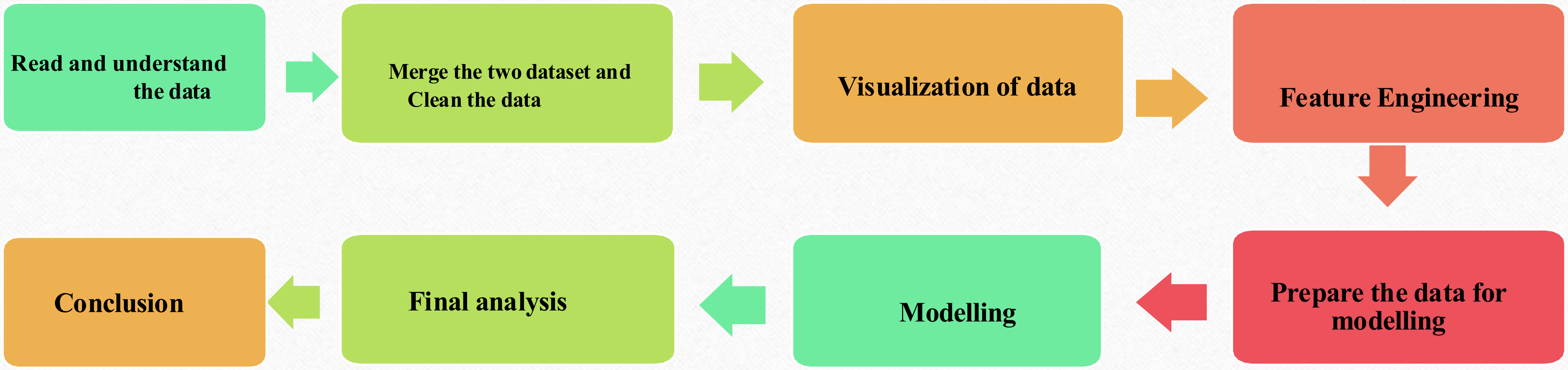


Count of Frauds transactions are more in South and Midwest region.

The amount spend in all the regions are same but in Northeast region is slightly more than others for fraud transactions.

MODELIING

Flow chart of modelling



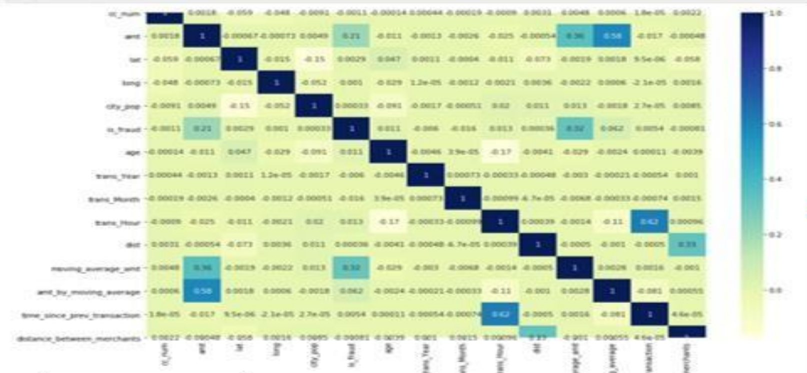
PREPARE DATA FOR MODELLING

Drop high correlated variables

Label encoding(variable on large categories)

Dummy creations(variable on counted categori

Data ready for the splitting into X,y (train,test)



correlation

Dummy creation

```python
# Creating a dummy variable for 'Category' variable and dropping the first one.
Catg_dummy = pd.get_dummies(fraud_data['category'], drop_first=True, prefix='catg')

# Creating a dummy variable for 'Gender' variable and dropping the first one.
gender_dummy = pd.get_dummies(fraud_data['gender'], drop_first=True, prefix='gender')

# Creating a dummy variable for 'Weekday' variable and dropping the first one.
week_dummy = pd.get_dummies(fraud_data['trans_weekday'], drop_first=True, prefix='weekday')

# Creating a dummy variable for 'Month' variable and dropping the first one.
month_dummy = pd.get_dummies(fraud_data['trans_Month'], drop_first=True, prefix='month')

# Creating a dummy variable for 'Year' variable and dropping the first one.
year_dummy = pd.get_dummies(fraud_data['trans_Year'], drop_first=True, prefix='year')
```

Label encoding

```python
# Import label encoder
from sklearn import preprocessing

# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()

# Encoding labels of following columns.
fraud_data['state']= label_encoder.fit_transform(fraud_data['state'])
fraud_data['job']= label_encoder.fit_transform(fraud_data['job'])
fraud_data['merchant']= label_encoder.fit_transform(fraud_data['merchant'])
fraud_data['city']= label_encoder.fit_transform(fraud_data['city'])
fraud_data['street']= label_encoder.fit_transform(fraud_data['street'])
```

Splitting the Data into X & y

```python
from sklearn.model_selection import train_test_split

# Putting feature variable to X
X = fraud_data.drop(['is_fraud','cc_num', 'trans_num'],axis=1)

# Putting response variable to y
y = fraud_data['is_fraud']
```

Checking for Class imbalance in Train & Test

```
y_train.value_counts(normalize=True)
0    0.99479
1    0.00521
Name: is_fraud, dtype: float64

y_test.value_counts(normalize=True)
0    0.994791
1    0.005209
Name: is_fraud, dtype: float64
```

Treating the class imbalance (using SMOTE / ADASYN)

MODELLING

MODELLING STATS

- Both the models of XGBoost i.e modelling done using default parameter using SMOTE and Hyper-tuned model using SMOTE data of XGBoost are giving same results .
- Hence one of then can be used for further analysis

ALGORITHM	Accuracy	Recall	F1-Score	Precision	Specificity
Decision Tree (default SMOTE)	99.6	84.7	66.5	54.8	99.6
Decision Tree (default ADASYN)	99.6	84.1	67.4	56.3	99.7
Decision Tree SMOTE Tuned	99.6	88.8	56.5	41.4	99.3
Random Forests (default SMOTE)	99.8	80.0	84.3	89.1	99.9
Random Forests (default ADASYN)	99.8	77.6	83.6	90.6	100
Random Forests SMOTE Tuned	99.8	80.0	84.3	41.1	99.3
XGBoost (default SMOTE)	**99.9**	**89.6**	**87.4**	**85.2**	**99.9**
XGBoost (default ADASYN)	99.9	89.1	86.7	84.4	99.9
XGBoost SMOTE Tuned	**99.9**	**89.6**	**87.4**	**85.2**	**99.9**

Model Reflection

Based on the accuracy, ROC, precision and recall of different models, we will consider XGBOOST (Hyper-parameter Tuning) for SMOTE data as our final model.

The test accuracy is 99.9%, recall is 89.6% and ROC is 99.8% .

The recall for fraudulent transaction is 89.6%, which is highest among all other models. Since our buisness objective is more important to identify fraudulent transaction than the non- fraudulent transaction accurately. High recall means model will correctly identify almost all fraudulent transaction.

Hence XGBOOST (Hyperparameter Tuning) model for SMOTE data is chosen based on its performance on Recall metric.

Compilation of models For Test data (Target 1)

Clasification Report for Decision Tree on Test data on default Hyperparameter

	precision(%) Positive predictive value	recall(%) (sensitivity)	f1-score(%)	Accuracy(%)	ROC(%)	Specificity(%)	False postive rate (%)	Negative (%) predictive value
SMOTE data	54.8	84.7	66.5	99.6	92.2	99.6	0.4	99.9
ADASYN data	56.3	84.1	67.4	99.6	91.9	99.7	0.3	99.9

Clasification Report for Decision Tree on Test data on SMOTE Hyperparameter Tunning

	precision(%) Positive predictive value	recall(%) (sensitivity)	f1-score(%)	Accuracy(%)	ROC(%)	Specificity(%)	False postive rate (%)	Negative (%) predictive value
SMOTE data	41.4	88.8	56.5	99.3	95.5	99.3	0.7	99.9

---

2. Clasification Report for Random forest on Test data on default Hyperparameter

	precision(%) (Positive predictive value)	recall(%) (sensitivity)	f1-score(%)	Accuracy(%)	ROC(%)	Specificity(%)	False postive rate (%)	Negative (%) predictive value
SMOTE data	89.1	80.0	84.3	99.8	99.6	99.9	0.1	99.9
ADASYN data	90.6	77.6	83.6	99.8	99.5	100	0.0	99.9

Clasification Report for Random Forest on Test data on SMOTE Hyperparameter Tunning

	precision(%) Positive predictive value	recall(%) (sensitivity)	f1-score(%)	Accuracy(%)	ROC(%)	Specificity(%)	False postive rate (%)	Negative (%) predictive value
SMOTE data	41.1	88.8	56.5	99.3	95.5	99.3	0.7	99.9

---

3. Clasification Report for XGBoost on Test data on default Hyperparameter

	precision(%) (pp value)	recall(%) (sensitivity)	f1-score(%)	Accuracy(%)	ROC(%)	Specificity(%)	False postive rate (%)	Negative (%) predictive V
SMOTE data	85.2	89.6	87.4	99.9	99.8	99.9	0.1	99.9
ADASYN data	84.4	89.1	86.7	99.9	99.7	99.9	0.1	99.9

Clasification Report for XGBoost on Test data on SMOTE Hyperparameter Tunning

	precision(%) +ve predictive value	recall(%) (sensitivity)	f1-score(%)	Accuracy(%)	ROC(%)	Specificity(%)	False postive rate (%)	-ve (%) predictive
SMOTE data	85.2	89.6	87.4	99.9	99.8	99.9	0.1	99.9

Cost Benefit Analysis

Part 1(on Whole data)

Cost Benefit Analysis(Part 1)

Questions	Answer
1. Average number of transactions per month	77183.0833333333
2. Average number of fraudulent transaction per month	402.125
3. Average amount per fraud transaction	530.6614122888819

Part 2(After Modelling)

Cost Benefit Analysis(Part 2)

Questions	Answer
1. Cost incurred per month before the model was deployed (2*3 of part1)----------------------------------	213392.2204
2. Average number of transactions per month detected as fraudulent by the model (TF)--------------------	405.125
3. Cost of providing customer executive support per fraudulent transaction detected by the model---------	1.5
4. Total cost of providing customer support per month for fraudulent transactions detected by the model(TF*$1.5) ---	607.6875
5. Average number of transactions per month that are fraudulent but not detected by the model (FN)-------	16.25
6. Cost incurred due to fraudulent transactions left undetected by the model (FN*3rd of part 1)----------	8623.25
7. Cost incurred per month after the model is built and deployed (4+6)----------------------------------	9230.94
8. Final savings = Cost incurred before - Cost incurred after(1-7)--------------------------------------	204161.28

Conclusion :Profit predicted after using ML models

Cost Incurred Before Using ML Model

=

$ 213392.220.4

Total cost For customer support per month for fraudulent transactions detected

=

$ 607.6875

Cost incurred per month after the model is built and deployed

=

$ 9230.94

Savings = Cost Incurred Before ML Model – Cost Incurred After ML model is Deployed

Savings

= $ 204161.28

Hence 95.67% of drastic decrease in amount paid by the bank to the customer for their loss by fraud transactions using this Model.

Business Recommendation

1. The fraudulent transaction probability of a transaction increases with increase in `hist_trans_avg_amt_24h` values. Based upon past spending pattern we have derived`hist_trans_avg_amt_24h` which is actually average amount spent through transactions in last 24 hours by the credit card holder's. So if comparable amount spent in last 24hrs v/s past spent data gets increased then its ideal for Bank to sent an SMS ALERT! to customer confirming about the transaction

2. The fraudulent transaction probability of a transaction increases with increase in weekday Thursday, Saturday and Monday values. As per the pattern model shows that major fraud transactions are noticed in weekday Thursday, Saturday and Monday. So banks need to be extra cautious and high alert on this specific days to avoid fraudulent transactions on these categories.

3. The fraudulent transaction probability of a customer increases with increase in `amt` values. At any point in time if bank notices the nature of amount spent is higher then regular spending pattern in such cases bank should noticed the same at early stage by sending neccessary alerts to customers.

4. The fraudulent transaction probability of a customer increases with increase in categories home, shopping_pos, grocery_pos, health_fitness, gas_transport values. Model predicted that major fraud transactions are occurred in the categories as these are the platform where any customer would spend large transactional amount so as fraudsters also follows the same trend. In such case its always recommended to bank to keep an eye on the track record of spend amount through FLASH SMS ALERT mentioning the detailed transaction history to respective credit card holders.

5. The fraud transactions are majorly done during odd hours of the day i.e. between 22 - 3 Hr so banks needs to ensure to send an SMS ALERT during such odd hours.

Thank you