Ex. No: 13 DATE:	MINIPROJECT – CREDIT CARD FRAUD DATA ANALYSIS AND VISUALIZATION

Introduction:

With the rapid growth of digital transactions, financial fraud has become a major concern for banks, businesses, and consumers. Traditional methods often fail to detect fraudulent activity effectively, especially when such transactions make up a very small fraction of the overall data.

This project applies data analytics to identify patterns in transaction behavior that are associated with fraud. Using a real-world dataset, the project explores demographic and merchant-level trends, applies statistical analysis to detect significant associations, and uses visual tools to present insights clearly. The goal is to demonstrate how AI techniques can support early fraud detection and informed decision-making.

Problem Statement:

The presence of highly imbalanced fraud data and complex behavioral patterns makes manual fraud detection ineffective. The key challenges addressed include:

- Extremely low fraud occurrence (0.57%)
- Difficulty in correlating user attributes (e.g., gender, merchant) with fraud likelihood
- Need for automated detection and pattern discovery

Objectives:

1. Analyze and preprocess financial transaction data
2. Extract temporal and behavioral patterns associated with fraud
3. Perform statistical correlation tests to identify fraud-related factors
4. Visualize the entire transaction landscape using Power BI
5. Quantify fraud risk across attributes like merchant, category, gender

Data Collection:

The dataset was sourced from Kaggle's "Credit Card Fraud Detection" repository by Kartik2112. It is a simulated credit card transaction dataset containing legitimate and fraud transactions from the duration 1st Jan 2019 - 31st Dec 2020. It covers credit cards of 1000 customers doing transactions with a pool of 800 merchants. This simulation was run for the duration - 1 Jan 2019 to 31 Dec 2020.

No of rows:10,48,575

Dataset Structure

The dataset consists of multiple weather attributes, including:

1. **Transaction date and time**– Timestamp for each observation.
2. **Merchant** – The merchant the payment has been made to.
3. **Category** – Category of the purchase made.
4. **Amount** – Amount of money spent (Dollars).
5. **Names of the card holders** – First and last names of the card holders.
6. **Gender** – Gender of the card holder.
7. **Addresses-** Addresses of the card holders, including the street, city , latitudes and longitudes.
8. **Job-** The job of the registered card holder.
9. **State –** The state the card holder is residing in.
10. **DoB-** Date of Birth of the card holder.
11. **Is Fraud –** Is the transaction fraudulent (Yes- 1, No- 0).

Data Preprocessing & Cleaning:

1. Handling Missing Values

Although the dataset was complete as it was a simulation, null values have been checked for to ensure data quality. It has been handled using techniques such as:

- Forward Fill (FFill): Propagates the last valid value forward.
- Mean/Median Imputation**:** Replaces missing numeric entries with the average or median.
- Zero Fill: Used when missing values imply absence.

```
df.fillna(method='ffill', inplace=True)
```

2. Removing Duplicates

Duplicate rows can distort analysis, especially in transaction-level data. All duplicates were removed to ensure one-to-one record mapping per transaction.

```
df.drop_duplicates(inplace=True)
```

3. Date- Time Feature Extraction

The original transaction time was converted to a datetime object to extract meaningful features. Year, Month, Day, Hour were derived for time-based trend analysis.

```
df['trans_date_trans_time'] = pd.to_datetime(df['trans_date_trans_time'], format='%d-%m-%Y %H:%M')

df['year'] = df['trans_date_trans_time'].dt.year

df['month'] = df['trans_date_trans_time'].dt.month

df['day'] = df['trans_date_trans_time'].dt.day

df['hour'] = df['trans_date_trans_time'].dt.hour
```

4. Encoding Categorical Data

Merchant names contained a "fraud_" prefix which was removed for clarity. Other categorical fields (e.g., gender, category) can be label-encoded if used for modelling.

```
df['merchant'] = df['merchant'].str.replace('^fraud_', '', regex=True)
```

5. Dropping irrelevant Columns

Columns such as credit card number, transaction number, and UNIX timestamp were removed as they do not contribute to fraud detection or analysis.

```
df.drop(columns=['cc_num', 'trans_num', 'unix_time', 'trans_date_trans_time'], inplace=True)
```

Exploratory Data Analysis:

EDA helps identify patterns, trends, and relationships among features.

1. Summary Statistics

Summary statistics are numerical values that describe and summarize the main characteristics of a dataset. They provide a quick overview of the distribution, central tendency, and variability of the data.

```
df=pd.read_csv(r'C:\Users\vigne\Downloads\archive(11)\fraudtrain.csv')
print(df.shape)
print(df.head())
print(df.info())
print(df.describe())
```

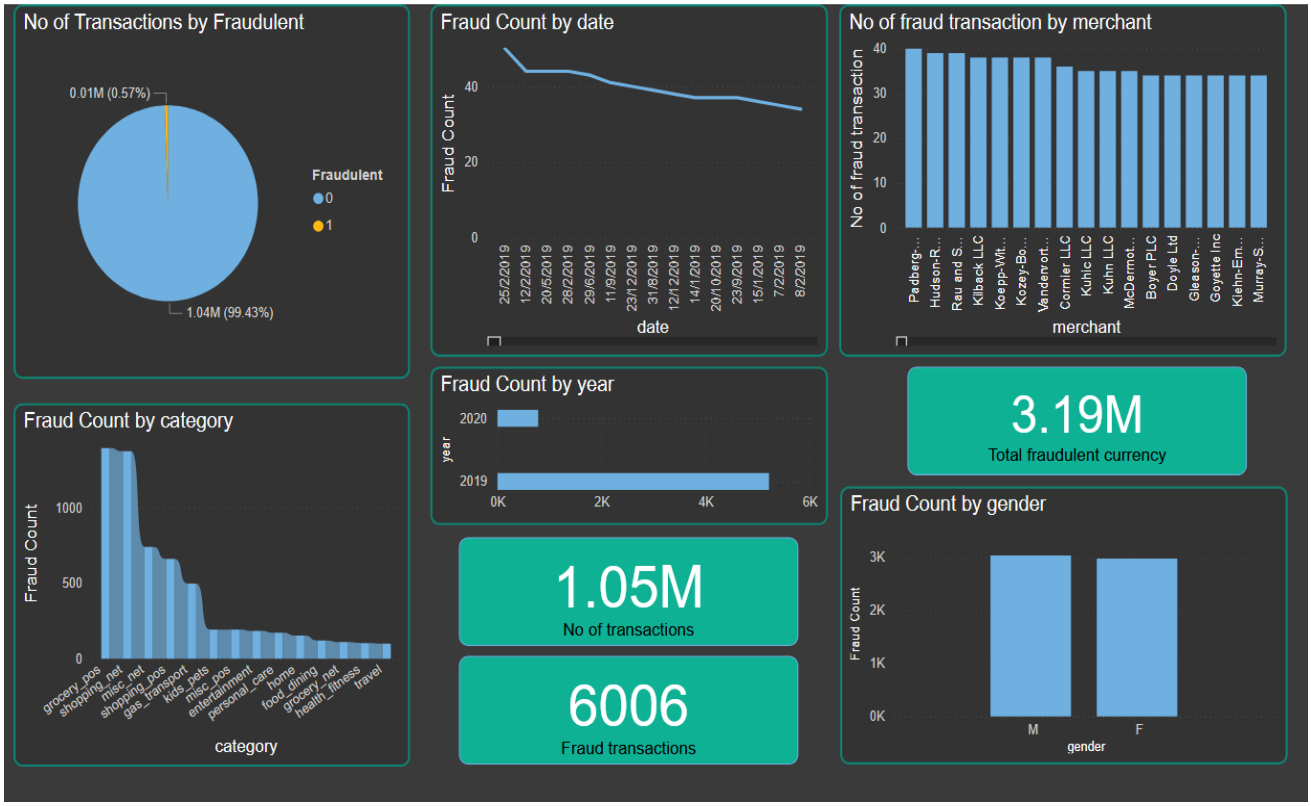
```
(1048575, 23)
   S.no trans_date_trans_time        cc_num                            merchant       category     amt      first     last gender  ...      long city_pop
0     0      01-01-2019 00:00  2.703190e+15          fraud_Rippin, Kub and Mann       misc_net    4.97   Jennifer    Banks      F  ...  -81.1781     3495
1     1      01-01-2019 00:00  6.304230e+11      fraud_Heller, Gutmann and Zieme    grocery_pos  107.23  Stephanie     Gill      F  ... -118.2105      149  Sp
2     2      01-01-2019 00:00  3.885950e+13                fraud_Lind-Buckridge  entertainment  220.11     Edward  Sanchez      M  ... -112.2620     4154
3     3      01-01-2019 00:01  3.534090e+15  fraud_Kutch, Hermiston and Farrell  gas_transport   45.00     Jeremy    White      M  ... -112.1138     1939
4     4      01-01-2019 00:03  3.755340e+14                 fraud_Keeling-Crist       misc_pos   41.96      Tyler   Garcia      M  ...  -79.4629       99

[5 rows x 23 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 23 columns):
 #   Column                 Non-Null Count    Dtype
---  ------                 --------------    -----
 0   S.no                   1048575 non-null  int64
 1   trans_date_trans_time  1048575 non-null  object
 2   cc_num                 1048575 non-null  float64
 3   merchant               1048575 non-null  object
 4   category               1048575 non-null  object
 5   amt                    1048575 non-null  float64
 6   first                  1048575 non-null  object
 7   last                   1048575 non-null  object
 8   gender                 1048575 non-null  object
 9   street                 1048575 non-null  object
 10  city                   1048575 non-null  object
 11  state                  1048575 non-null  object
 12  zip                    1048575 non-null  int64
 13  lat                    1048575 non-null  float64
 14  long                   1048575 non-null  float64
 15  city_pop               1048575 non-null  int64
 16  job                    1048575 non-null  object
 17  dob                    1048575 non-null  object
 18  trans_num              1048575 non-null  object
 19  unix_time              1048575 non-null  int64
 20  merch_lat              1048575 non-null  float64
 21  merch_long             1048575 non-null  float64
 22  is_fraud               1048575 non-null  int64
dtypes: float64(6), int64(5), object(12)
memory usage: 184.0+ MB
```

2. Data Visualization

Visualizing data provides insights into seasonal variations, correlations, and anomalies.

We have used PowerBI for data visualization purposes with various charts like Pie chart, Line chart, Stacked column chart, Cards, Ribbon charts and Clustered Column charts for visualizing the data in different formats . The dashboard presents insights into the data and how each of the features affects the results.

PowerBI Dashboard:



No of fraudulent transactions:

The Pie chart Displays the percentage of fraudulent transactions out of the total number of transactions.

Number of fraudulent transactions: 6006

Fraud count time series:

The line chart time series displays the Number of fraudulent transactions over the dates from 1st January 2019 to 31st December 2020.

No of Fraud transactions to merchants:

The stacked column chart displays the number of fraud transactions made to the merchants. The Highest number of fraud transactions has been made to Padberg-Welch

Fraud transaction in Spending category:

The ribbon chart displays the number of fraudulent transactions in the each of the category the transaction is classified under (groceries, fuel, online shopping, etc). The highest number of fraudulent transactions is under grocery shopping.

Fraud count over the years:

The bar chart displays the fraud counts in the years 2019 and 2020. The highest number of frauds occurred in 2019 with a total count of 5220 and 786 in 2020.

Fraud count by gender:
The bar chart displays the amount of frauds committed by Male and Female individuals. Males have committed 3301 counts of fraud and Females 2975 counts of fraud.

Cards:

The cards display the aggregation of numerical data such as the total number of fraudulent transactions (6006), Total number of transactions (1.05 million) and the total amount of fraudulent currency (3.1 million).

Feature Engineering Techniques Used

1.Extracted year, month, day, and hour from trans_date_trans_time to enable temporal pattern analysis.

```
df['trans_date_trans_time']=pd.to_datetime(df['trans_date_trans_time'],format='%d-%m-%Y %H:%M')
df['year']=df['trans_date_trans_time'].dt.year
df['month']=df['trans_date_trans_time'].dt.month
df['day']=df['trans_date_trans_time'].dt.day
df['hour']=df['trans_date_trans_time'].dt.hour
```

2.Text Cleaning:
Removed the "fraud_" prefix from the merchant field to simplify grouping and interpretation.

```
df['merchant']=df['merchant'].str.replace('^fraud_', '', regex=True)
```

3.Irrelevant feature removal:
Dropped fields like cc_num, unix_time, trans_num that add noise or leak sensitive info without analytical value.

```
df.drop(columns=['cc_num','trans_num','unix_time','trans_date_trans_time'],axis=1,inplace=True)
```

CORRELATION FINDINGS:

The correlation between gender and fraudulent transactions has been calculated by using the Chi-square test as they are both categorical data.

```
contingency_table = pd.crosstab(df['gender'], df['is_fraud']==1)
chi2, p, dof, expected = chi2_contingency(contingency_table)
print("Chi-square Statistic:{0:.2f}".format(chi2))
print(f"Degrees of Freedom: {dof}")
print("p-value:{0:.8f}".format(p))
```

```
Degrees of Freedom: 1
p-value: 4.940014157810641e-16
There is a significant association between gender and being a fraudster.
```

Key Findings:

- Fraudulent transactions occurred across multiple merchants, with certain vendors showing >30 fraud cases.
- Fraud was more prevalent in grocery_pos and shopping_net categories.
- Statistically significant gender disparity in fraud involvement.
- Temporal peaks were observed around specific dates and hours.

Future Enhancements:

- Integrate a machine learning model for real-time fraud prediction
- Use geolocation clustering to detect location-based fraud rings
- Create a web dashboard for financial institutions

CONCLUSION:

The project "Credit Card Fraud Analysis and Visualization" aimed to explore transaction data to identify patterns associated with fraudulent activity. Through data preprocessing, feature extraction, and statistical analysis, key insights were uncovered regarding the distribution of fraud across categories, merchants, and user demographics.

Power BI visualizations further highlighted trends in fraud frequency over time, high-risk merchant profiles, and fraud concentration across transaction categories. Despite a low overall fraud rate, the project demonstrated that meaningful anomalies can be detected through careful data examinations.