ML HACKATHON

**CONVOLOVE 3.0**

CREDIT CARD BEHAVIOUR SCORE

horizontal line

#### 

### 1. Approach Details

The approach to developing the Behaviour Score involved systematic steps to prepare the data and build a predictive model. These steps are detailed as follows:

#### **Data Preprocessing**

* **Handling Missing Values**:
  + Columns with more than 30% missing values were removed to enhance data quality and reduce noise.
  + For the remaining missing values, imputation techniques were applied:
    - Bureau-related columns: Imputed using the median to ensure consistency.
    - Bureau enquiry columns: Filled with zero, assuming no enquiries in cases of missing data.
    - On-us attributes and transaction attributes: Imputed using the median to handle numerical gaps.
* **Feature Selection**:
  + **Low Variance Features**: Features with variance below 0.01 were dropped, as they added little to no predictive value.
  + **Correlation Analysis**: Highly correlated features (with a correlation threshold > 0.9) were identified and removed to reduce multicollinearity and improve model stability.
* **Dimensionality Reduction with PCA**:  
  PCA (Principal Component Analysis) was employed to reduce the dataset to 10 components while retaining most of the variance. This step reduced dimensionality and enhanced computational efficiency.

#### **Model Development**

* **Data Splitting**:  
  The processed data was split into training (70%) and testing (30%) sets using stratified sampling to maintain the balance between default (bad\_flag = 1) and non-default (bad\_flag = 0) cases.
* **Handling Class Imbalance**:  
  To address the imbalance in the bad\_flag target variable, SMOTE (Synthetic Minority Oversampling Technique) was used to oversample the minority class in the training data, creating a balanced dataset for model training.

### Algorithm Used

The following describes the neural network model used for predicting credit card fraud:

1. **Model Type**A **Deep Neural Network** (DNN) was employed, leveraging **Keras** to build the architecture. It is a **feedforward neural network** designed to capture complex patterns in data through multiple layers of processing.
2. **Architecture**The neural network consists of the following layers:
   * **Input Layer**: The input layer consists of 64 neurons, corresponding to the number of features in the data after pre-processing. The activation function used is **ReLU** (Rectified Linear Unit), which helps in introducing non-linearity into the model and allows for faster training.
   * **Hidden Layers**: Two hidden layers follow the input layer:
     + The first hidden layer contains 32 neurons with **ReLU** activation.
     + The second hidden layer contains 16 neurons, again with **ReLU** activation.
     + **Dropout regularization** (0.3 rate) is applied after each hidden layer to reduce overfitting by randomly dropping 30% of neurons during training.
   * **Output Layer**: The output layer consists of a single neuron with **sigmoid activation**, which outputs a probability between 0 and 1, representing the likelihood of the transaction being fraudulent.
3. **Compilation**The model was compiled using:
   * **Optimizer**: **Adam optimizer** was used for training, with a learning rate of 0.001. Adam combines the advantages of both **RMSprop** and **Momentum**, making it well-suited for complex models.
   * **Loss Function**: **Binary cross-entropy** loss was used as this is a binary classification task (fraud or no fraud).
   * **Metrics**: **Accuracy** was chosen as the primary metric for evaluating the model's performance.
4. **Training Process**The model was trained using the **fit()** method with:
   * **Epochs**: 20 epochs (iterations over the entire training dataset).
   * **Batch Size**: 32, which is a reasonable choice for efficient learning.
   * **Validation Split**: 0.2 of the training data was reserved for validation during training to monitor the model's performance and avoid overfitting.

This neural network model was designed to learn complex relationships within the data, accounting for class imbalance using dropout and a balanced training set with SMOTE.

### 

### 

### 

### 3. Steps Followed

This section provides a detailed breakdown of the steps taken during the development of the Behaviour Score model. Each step was designed to systematically process the data, address data quality issues, and build predictive models.

#### **Step 1: Data Preprocessing**

Data preprocessing was a crucial step to prepare the dataset for modeling. The following actions were taken:

1. **Handling Missing Values**:
   * Columns with more than 30% missing values were identified and dropped, as they were deemed unlikely to contribute valuable information.
   * The remaining missing values were handled using targeted imputation strategies:
     + **Bureau Attributes**: Median imputation was applied to replace missing values while preserving the central tendency of the data.
     + **Bureau Enquiry Attributes**: Missing values were filled with zeros, under the assumption that no recorded enquiries implied no activity.
     + **On-us and Transaction Attributes**: Median imputation was also applied to ensure consistency and avoid data loss.
2. **Feature Selection**:
   * **Low Variance Filtering**: Features with variance below a threshold of 0.01 were removed to eliminate attributes that provided minimal information.
   * **Correlation Analysis**: A correlation matrix was computed to identify highly correlated features (absolute correlation > 0.9). These features were removed to reduce multicollinearity, which can negatively impact model performance.
3. **Dimensionality Reduction using PCA**:
   * Principal Component Analysis (PCA) was applied to the remaining features to reduce the dataset to 10 components, while retaining most of the variance. This helped in simplifying the model without losing significant information and improved computational efficiency.

#### **Step 2: Class Balancing**

Class imbalance in the target variable (bad\_flag) was a significant challenge, as the proportion of defaults (bad\_flag = 1) was much smaller than non-defaults (bad\_flag = 0). To address this:

* **SMOTE (Synthetic Minority Oversampling Technique)** was used to generate synthetic samples for the minority class, creating a balanced training dataset. This step ensured that models were not biased towards the majority class and improved their ability to detect defaults.

### Step 3: Model Building and Testing

The following model was built and tested to determine the most effective approach for predicting credit card defaults:

1. **Keras Deep Neural Network:**
   * A custom deep learning model was built using TensorFlow/Keras. The architecture included:
     + **Input Layer:** The input layer consists of 64 neurons, corresponding to the features in the dataset, with ReLU activation.
     + **Hidden Layers:** Two hidden layers were used, one with 64 neurons and another with 32 neurons. Both layers utilized ReLU activation functions to capture complex relationships in the data.
     + **Dropout Layers:** Dropout regularization with a rate of 0.3 was added between the layers to prevent overfitting and improve generalization by randomly dropping units during training.
     + **Output Layer:** A single output neuron with a sigmoid activation function for binary classification (predicting fraud or non-fraud).
   * The model was compiled using the Adam optimizer with a learning rate of 0.001 and binary cross-entropy as the loss function. The accuracy metric was used to evaluate model performance during training.
   * SMOTE (Synthetic Minority Over-sampling Technique) was applied to the training data to address the class imbalance. The training data was resampled to balance the number of fraud (positive) and non-fraud (negative) examples.

#### **Step 5: Model Evaluation**

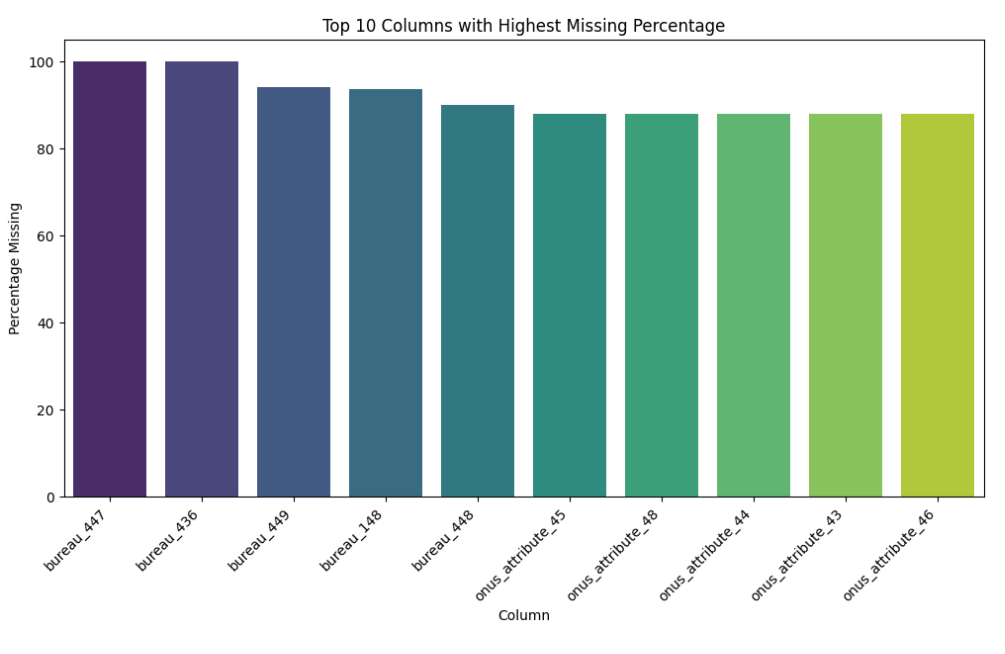
* The model's performance was evaluated on the test set, and **AUC-ROC** and **classification report** metrics (precision, recall, F1-score) were used to assess its predictive accuracy for both classes (fraud and non-fraud).

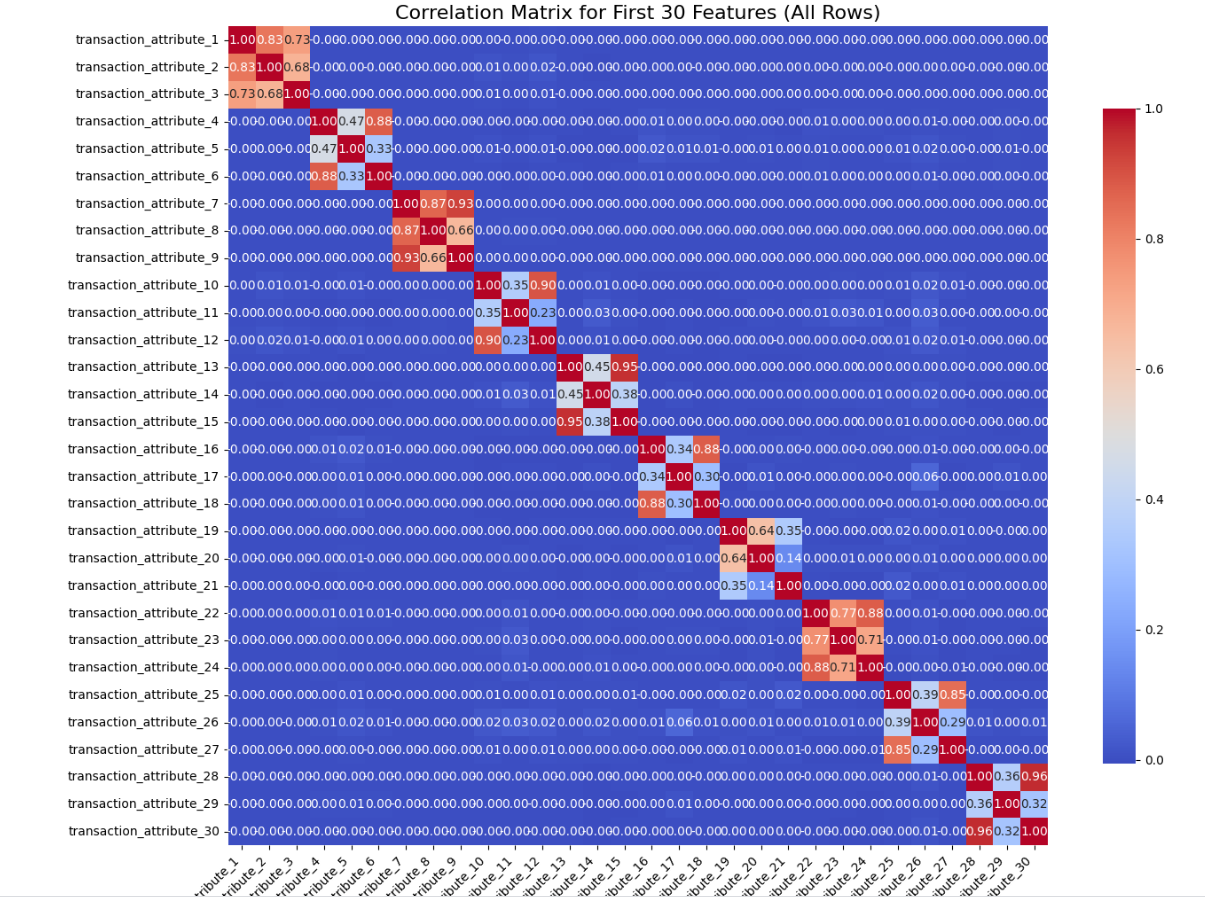
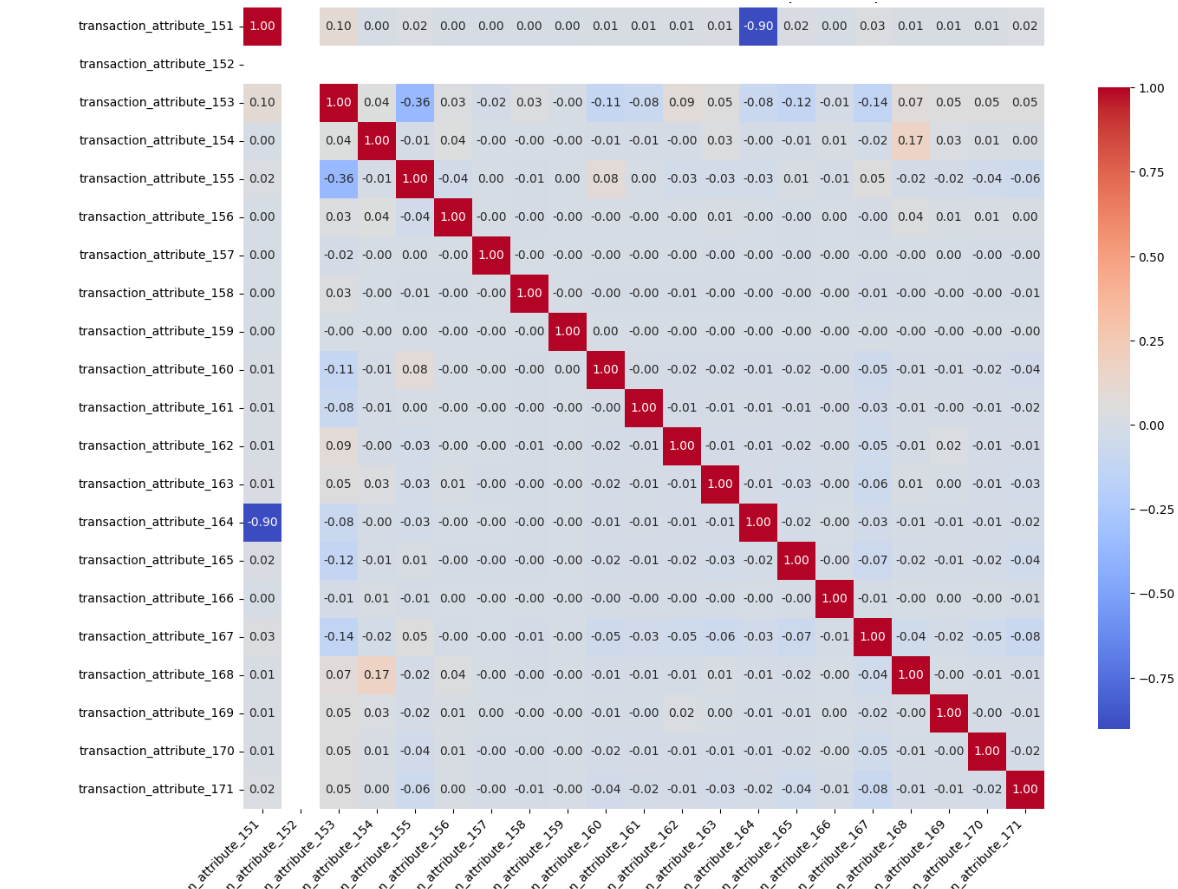


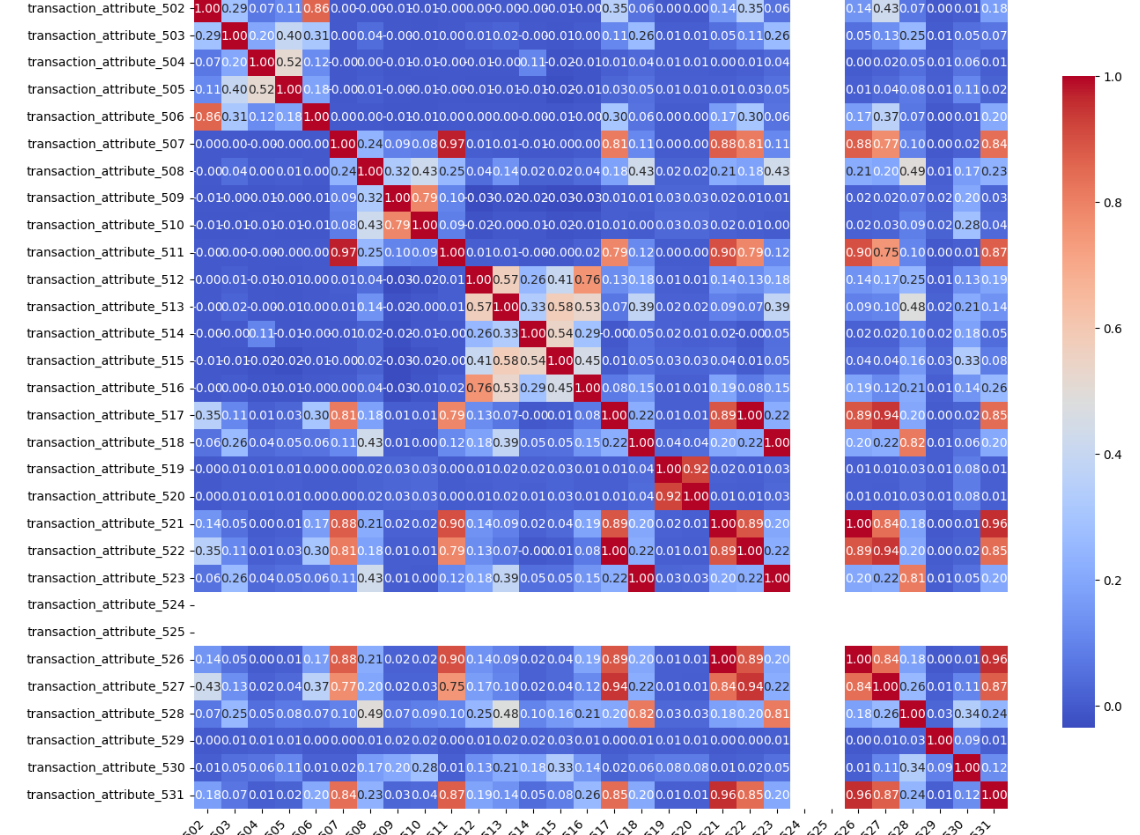
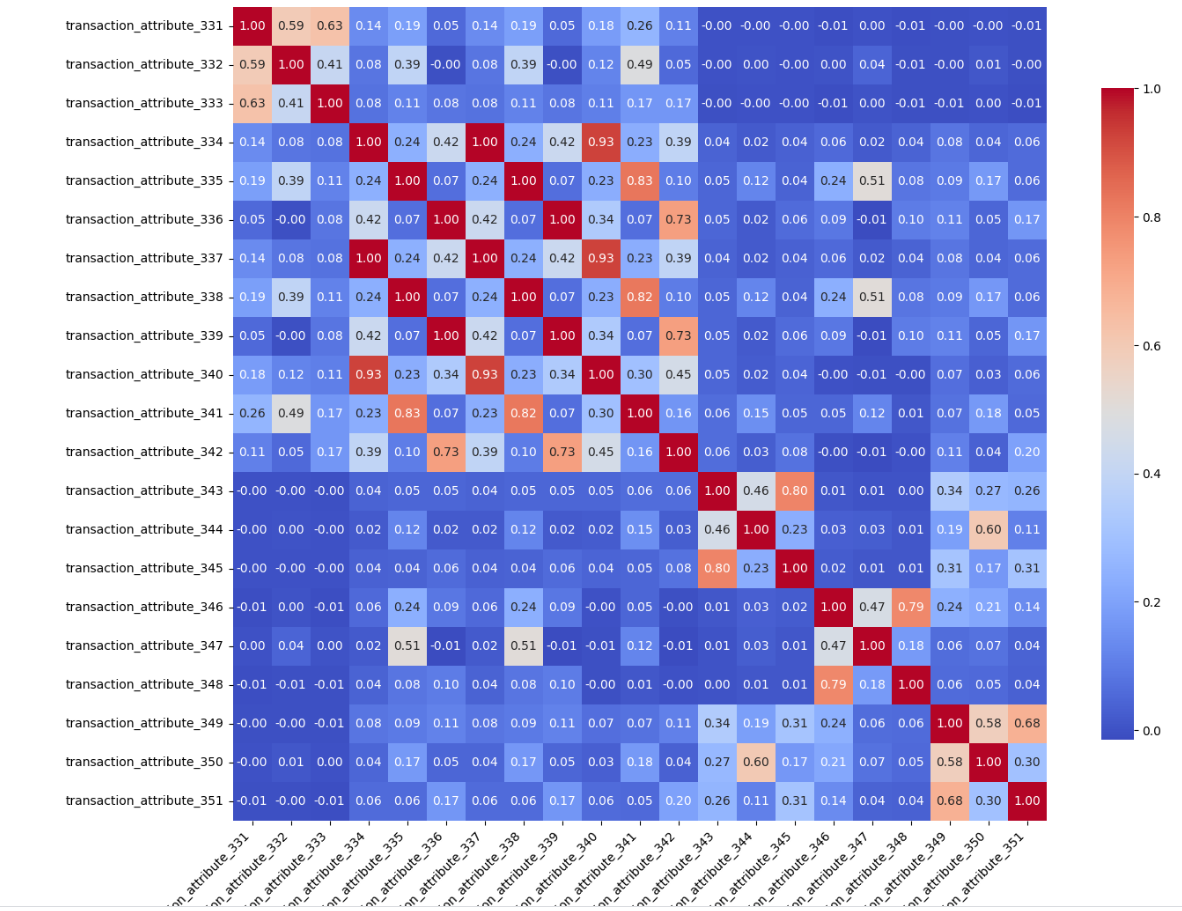


## Observations:

Missing value percentage:

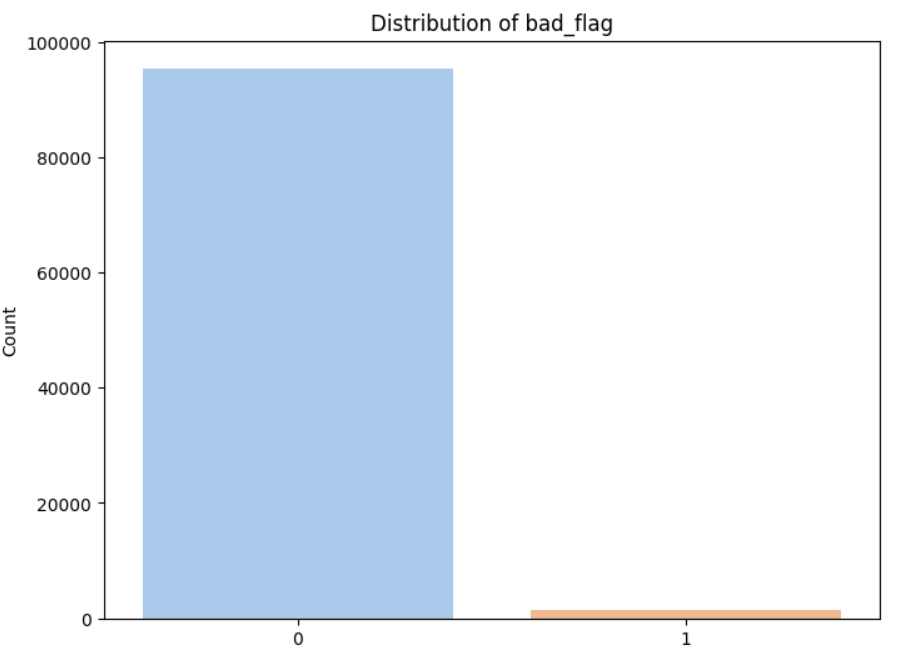


**Trends and correlations observed in Transaction attribute:**  
 



Many columns were related to each other

**Imbalance:**



**Correlation matrix among columns:**

