Capstone Project Documentation

Project Name: Find Default (Prediction of Credit Card Fraud) using Machine Learning

Submitted by : Vicky Suryabhan Dighore

INDEX

Problem Statement

As the reliance on credit cards for online transactions continues to grow, the incidence of credit card fraud poses a significant threat to both consumers and financial institutions. Credit card fraud, defined as the unauthorized use of someone else's credit card information to make purchases or withdraw funds, leads to substantial financial losses and undermines consumer trust in electronic payment systems.

Despite advancements in security measures, detecting fraudulent transactions in real-time remains a challenge. This problem necessitates the development of effective detection systems that can accurately identify and mitigate fraudulent activities while minimizing false positives, thereby ensuring that customers are protected from unauthorized charges and credit card companies maintain their reputation for security.

Dataset Understanding

The dataset comprises credit card transactions made by European cardholders in September 2013. It includes data from two days, featuring a total of **284,807 transactions**, out of which **492 are classified as fraudulent**.

This dataset exhibits a significant imbalance, as the positive class (fraudulent transactions) accounts for only **0.172%** of all transactions. This high level of imbalance presents challenges for detecting fraudulent activity, as traditional machine learning models may struggle to accurately identify the minority class without appropriate handling of the imbalance.

EDA & DATAPREPROCESSING STEPS

Insights:

1.all fraud transactions occur for an amount less than 2500

2.It can also be observed that the fraud transactions are evenly distributed about time.

Exploratory Data Analysis (EDA)

Exploratory Data Analysis is a critical step in understanding the dataset and uncovering underlying patterns that can inform modeling strategies. For the credit card fraud detection dataset, the following EDA steps are conducted:

1. **Dataset Overview:**

- Load the dataset and display the first few rows to gain an initial understanding of its structure and content.
- Check the total number of transactions and the distribution of the target variable (fraud vs. non-fraud).

2. **Summary Statistics:**
   - Generate summary statistics (mean, median, standard deviation, etc.) for numerical features to understand their distributions.
   - Assess the presence of any outliers that may influence the model.
3. **Class Distribution:**
   - Visualize the distribution of the target variable (fraudulent vs. non-fraudulent transactions) using bar charts or pie charts to highlight the class imbalance.
   - In this dataset, fraudulent transactions account for only **0.172%**, indicating a significant class imbalance.
4. **Correlation Analysis:**
   - Use a correlation matrix to examine relationships between numerical features. This can help identify highly correlated features that may be redundant.
   - Visualize the correlation matrix with a heatmap to make it easier to spot strong correlations.
5. **Feature Distribution Visualization:**
   - Create histograms or box plots for key numerical features to visualize their distributions and identify potential skewness or outliers.
   - Analyze categorical features (if any) using count plots to understand their distribution.
6. **Time-based Analysis:**
   - If time-related features are present, explore trends over time to identify patterns or anomalies in transactions.
7. **Fraudulent Transaction Analysis:**
   - Analyze the characteristics of fraudulent transactions, such as average transaction amounts, geographical distribution, and frequency of occurrences.

Data Preprocessing

Once EDA is completed, data preprocessing is essential for preparing the dataset for modeling. The following steps are undertaken:

1. **Data Inspection:**
   - Load the dataset and check for missing values and duplicates. Assess the data types of each feature to ensure they are appropriate for analysis.
2. **Handling Missing Values:**
   - Identify any missing values in the dataset. Depending on the context, missing values can be:
     - Removed if they constitute a small percentage of the dataset.
     - Imputed using methods like mean or median for numerical features or mode for categorical features.
3. **Data Transformation:**
   - Normalize or standardize numerical features to ensure they are on a similar scale. This is particularly important for algorithms sensitive to feature scaling, such as logistic regression and neural networks.
   - Convert categorical variables to numerical format using one-hot encoding or label encoding where necessary.
4. **Feature Engineering:**
   - Create new features that may enhance model performance, such as:
     - Temporal features (e.g., time of transaction, day of the week).
     - Aggregated statistics (e.g., average transaction amount per user).
   - Drop irrelevant or redundant features identified during EDA.
5. **Handling Class Imbalance:**
   - Address the class imbalance using techniques such as:
     - **Oversampling:** Utilize methods like SMOTE (Synthetic Minority Over-sampling Technique) to generate synthetic examples of the minority class.
     - **Undersampling:** Reduce the number of majority class instances to balance the dataset.
     - **Class Weights:** Assign higher weights to the minority class during model training to improve detection.
6. **Data Splitting:**

- Split the dataset into training and testing sets, typically in an 80/20 or 70/30 ratio. Use stratified sampling to ensure that both sets maintain the same class distribution.

7. **Feature Selection:**
   - Utilize feature selection techniques to retain only the most relevant features, reducing dimensionality and improving model efficiency. This can be done using correlation analysis, recursive feature elimination, or tree-based feature importance.
8. **Final Data Review:**
   - Conduct a final review of the preprocessed dataset to ensure it is clean, balanced, and ready for modeling.

FEATURE ENGINEENERING

Feature engineering is the process of using domain knowledge to extract features (input variables) from raw data that make machine learning algorithms work more effectively. It plays a critical role in building predictive models and can significantly impact their performance.

- **Synthetic Minority Oversampling Technique (SMOTE)**: To address the class imbalance, SMOTE was applied to generate synthetic samples of the minority class (fraudulent transactions).
- **Feature Selection**: Mutual information was calculated to identify the most relevant features for predicting fraud, with features like V14, V10, and Amount showing high relevance.

Importance of Feature Engineering

1. **Improves Model Accuracy**:
   - Well-engineered features can lead to more accurate models. Effective features help the algorithm capture the underlying patterns in the data better.
2. **Reduces Overfitting**:
   - By creating relevant features and eliminating redundant or irrelevant ones, feature engineering can help prevent models from overfitting to noise in the training data.

3. **Enhances Interpretability**:
    - Thoughtfully engineered features can make the model more interpretable, enabling better understanding and communication of how predictions are made.
4. **Facilitates Better Insights**:
    - Feature engineering allows for the exploration of data and the extraction of meaningful insights that can inform business decisions or scientific inquiries.

Model Selection (Description of Design Choices)

Design Choices

In this project, various classifiers will be explored to effectively detect fraudulent transactions from a highly imbalanced dataset. Given the challenges of fraud detection, selecting appropriate models and tuning their parameters is crucial to achieving optimal performance.

1. **Classifier Choice:**
    - The **Stochastic Gradient Descent (SGD) Classifier** is chosen for its flexibility and efficiency, especially with large datasets. It supports various loss functions and penalties, allowing it to adapt well to the specific characteristics of the data.
    - The model is capable of handling both binary and multi-class classification problems and can be regularized to prevent overfitting, making it suitable for our imbalanced dataset.
2. **Hyperparameter Tuning:**
    - A **Grid Search** approach is employed to systematically explore combinations of hyperparameters for the SGDClassifier. This includes variations in the loss function (e.g., 'log' for logistic regression and 'hinge' for SVM), penalty types (L1 and L2), and the learning rate (alpha).
    - A custom scoring metric, **Matthews Correlation Coefficient (MCC)**, is used to evaluate the models, as it provides a balanced measure that is particularly useful in imbalanced classification scenarios.
3. **Pipeline Creation:**
    - A **Pipeline** is established to streamline the preprocessing and modeling steps, ensuring that feature scaling (via StandardScaler) is applied consistently before model fitting.
4. **Cross-Validation:**

- A 5-fold cross-validation approach is used to ensure that the model evaluation is robust and less susceptible to variance due to data splitting.

  Multiple classifiers were evaluated to determine the best model for fraud detection:

- **Logistic Regression**: A baseline model to understand the linear relationships.
- **Random Forest Classifier**: Chosen for its robustness and ability to handle imbalanced datasets.
- **K-Nearest Neighbors (KNN)**: Selected for its simplicity and effectiveness in classification tasks.
- **Support Vector Classifier (SVC)**: Evaluated for its performance in high-dimensional spaces.
- **Stochastic Gradient Descent (SGD)**: Used for its efficiency in large datasets.

  GridSearchCV was employed for hyperparameter tuning to optimize model performance.

Performance Evaluation

To evaluate the effectiveness of the selected model, several performance metrics are computed:

1. **Classification Report:**
   - The classification report provides a detailed breakdown of the model's performance, including precision, recall, and F1-score for each class. This is essential for understanding how well the model distinguishes between fraudulent and non-fraudulent transactions.
2. **AUC-ROC Score:**
   - The Area Under the Receiver Operating Characteristic Curve (AUC-ROC) score is calculated to assess the model's ability to discriminate between the positive (fraudulent) and negative (non-fraudulent) classes. A higher AUC score indicates better model performance.
3. **F1-Score:**
   - The F1-score is computed as the harmonic mean of precision and recall, providing a single metric that balances both false positives and false negatives. This is particularly important in fraud detection, where missing a fraudulent transaction (false negative) can be costly.
4. **Accuracy:**

- Accuracy is measured to determine the overall performance of the model. However, due to the imbalanced nature of the dataset, accuracy alone may not be a sufficient indicator of model performance.

  The models were evaluated using the following metrics:

- **Accuracy**: The proportion of true results among the total number of cases examined.
- **F1-Score**: The harmonic mean of precision and recall, providing a balance between the two.
- **AUC-ROC**: The area under the receiver operating characteristic curve, indicating the model's ability to distinguish between classes.

Example Results

- **Random Forest Classifier**: Achieved an accuracy of 99.96% and an F1-Score of 0.88.
- **K-Nearest Neighbors**: Achieved an accuracy of 99.83% with an F1-Score of 0.63.

Benefits of Model to technical user or company and potential users

The fraud detection model offers substantial benefits for both technical users within the company and potential external stakeholders:

1. **Enhanced Fraud Detection Capabilities:**
   - The model improves the accuracy and speed of fraud detection, enabling the company to flag suspicious transactions more effectively. This proactive approach helps reduce financial losses due to fraud, safeguarding both the company and its customers.
2. **Operational Efficiency:**
   - Automating the fraud detection process with machine learning significantly reduces the manual effort needed to review transactions. This allows the company's resources to focus on high-priority tasks and ensures that the team is only alerted for high-risk cases, improving overall efficiency.
3. **Scalability:**
   - The model is designed to handle large datasets, making it scalable as transaction volumes grow. For a company experiencing growth in transaction frequency, this model provides a solution that can adapt to increasing data without compromising performance.
4. **Real-Time Decision Making:**
   - Integrating this model into real-time transaction systems enables immediate fraud detection, which can prevent fraudulent transactions before they

complete. This real-time capability is a significant advantage in mitigating potential damage caused by fraud.

5. **Data-Driven Insights:**
   - The model provides insights into transaction patterns and fraud trends. Technical teams can analyze these insights to understand new fraud tactics, helping them continually refine fraud detection strategies and update the model with changing fraud patterns.
6. **Improved Customer Trust and Retention:**
   - By ensuring safe transactions and protecting customers from fraudulent activities, the company can enhance customer trust and loyalty, ultimately improving retention rates. Customers are more likely to continue using services that proactively guard their financial safety.

Potential Users

- **Financial Institutions:**
  - Banks, credit card companies, and payment processors can use the model to secure their payment networks, protect their customers, and reduce fraudulent transactions.
- **eCommerce Platforms:**
  - Online merchants and eCommerce platforms dealing with high transaction volumes can incorporate the model to identify and prevent fraudulent purchases, reducing chargebacks and protecting revenues.
- **Payment Gateways:**
  - Payment processing companies that facilitate transactions between customers and merchants can benefit from detecting fraud early in the transaction flow.
- **Insurance Companies**:
  - Firms offering insurance against fraud could use this model to identify fraudulent claims or transactions within their portfolios, reducing payout costs.

Future Work

The future development of the fraud detection model could focus on enhancing accuracy, interpretability, and flexibility. Below are several areas for potential improvement and further work:

1. **Integration with Advanced Deep Learning Models**:

- Future iterations could explore using deep learning techniques like recurrent neural networks (RNNs) or convolutional neural networks (CNNs) to improve performance on high-dimensional and sequential data, potentially capturing even subtler patterns of fraudulent behavior.

2. **Real-Time Detection Enhancements**:
   - Future work can improve the model's performance for real-time detection, optimizing response times to ensure that transactions are flagged almost instantly.
3. **Incremental Learning and Adaptation**:
   - As fraud patterns continuously evolve, integrating incremental learning techniques would allow the model to adapt to new patterns without requiring full retraining. This capability would ensure that the model remains effective against emerging fraud tactics.
4. **Explainability and Interpretability**:
   - Adding explainable AI (XAI) components would make the model more transparent, allowing technical teams to understand why a transaction was flagged. This interpretability is essential for compliance, and it helps build trust among stakeholders who rely on the model's decisions.
5. **User-Friendly Interface for Monitoring and Analysis**:
   - Building a user-friendly dashboard for monitoring model performance, flagged transactions, and false positives would empower non-technical teams to analyze trends and make data-driven decisions.
6. **Enhanced Model Robustness**:
   - Using techniques like ensemble learning and adversarial training could make the model more resilient against attempts to evade detection by sophisticated fraudsters.
7. **Improved Data Privacy and Security**:
   - Data security is critical for fraud detection systems. Future development could focus on techniques like federated learning to enable model updates without exposing sensitive transaction data, thus maintaining data privacy.
8. **Cross-Platform Deployment**:
   - Extending the model for seamless integration across mobile and web platforms would ensure robust fraud detection in all environments where transactions occur, protecting a broader range of customers.
9. **Monitoring and Handling Bias**:

  - To ensure fair decision-making, future work should monitor the model for biases and investigate any disparities in model performance across different demographics or transaction types.