A PROJECT REPORT

on

"Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI"

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN COMPUTER SCIENCE & ENGINEERING

BY

Name	Roll Number
Punit Panda	22051535
Aditya Mohanty	22053657
Smaranika Naik	22053638
Sibani Sahoo	22051545
Yash Tripathi	22053736

UNDER THE GUIDANCE OF

Jyotiprakash Mishra

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024

April 2025

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

CERTIFICATE

This is to certify that the project entitled
"Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI"
submitted by
Punit Panda (22051535), Aditya Mohanty (22053657), Smaranika Naik (22053638), Sibani Sahoo (22051545), Yash Tripathi (22053736)
is a record of bonafide work carried out by them, in partial fulfilment of the requirement for the award of
Bachelor of Engineering in Computer Science & Engineering at KIIT Deemed to be University, Bhubaneswar.

This work is done during the year 2024-2025, under our guidance.

Jyotiprakash Mishra
Project Guide

ACKNOWLEDGEMENTS

We are profoundly grateful to **Jyotiprakash Mishra Sir** of **KIIT School of Computer Engineering** for his expert guidance and continuous encouragement throughout this project.

PUNIT PANDA, ADITYA MOHANTY, SMARANIKA NAIK, SIBANI SAHOO, YASH TRIPATHI

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

ABSTRACT

Imbalanced classification is a challenge in machine learning in which one class significantly outnumbers the other, leading to biased model performance. This project focuses on addressing this issue using datasets like *Telco Customer Churn, Credit Card Fraud Detection, Employee Attrition, Loan Approval Prediction.* We employ comprehensive data preprocessing techniques, including handling missing values, encoding categorical variables, and scaling numerical features.

To mitigate class imbalance, we experiment with SMOTE (Synthetic Minority Over-sampling Technique), Random Undersampling, and class-weighting approaches. We implement multiple classification models, including Logistic Regression, Random Forest, and XGBoost, and evaluate their performance using key metrics such as precision-recall curves, ROC-AUC scores, and confusion matrices.

Beyond performance evaluation, we integrate Explainable AI (XAI) techniques to enhance model interpretability. Using LIME (Local Interpretable Model-agnostic Explanations), we analyze feature contributions and understand the key factors influencing customer churn predictions. This explainability framework ensures transparency and trust in decision-making processes. Our findings provide a comparative analysis of different resampling techniques and classification models, offering insights into optimizing machine learning workflows for imbalanced datasets in real-world applications.

Keywords: Imbalanced Classification, SMOTE and Undersampling, Explainable AI (LIME, SHAP), Customer Churn Prediction, Machine Learning Model Evaluation

Contents

1 Introduction

Machine learning has become an essential tool for predictive analytics, particularly in domains such as customer retention, fraud detection, human resource management, and financial decision-making. However, one of the major challenges in real-world classification problems is the presence of imbalanced datasets, where one class significantly outnumbers the other. Traditional machine learning algorithms tend to be biased toward the majority class, leading to poor performance in identifying minority-class instances. This project addresses the issue of class imbalance by implementing and comparing various data balancing strategies, classification models, and explainability techniques across multiple real-world datasets.

Existing solutions often rely on standard classification approaches that fail to handle highly skewed data distributions effectively. While some studies apply oversampling and undersampling techniques, they do not systematically compare different strategies or incorporate Explainable AI (XAI) methodologies to understand model behavior. Moreover, most implementations lack reproducibility and generalizability across different datasets, limiting their real-world applicability. This project fills these gaps by systematically evaluating multiple resampling techniques (SMOTE, ADASYN, and undersampling), training various classifiers (Logistic Regression, Random Forest, and XGBoost), and using SHAP and LIME for model interpretability.

The project follows a structured approach to ensure a comprehensive analysis of imbalanced classification.

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

This study contributes to developing more reliable and interpretable classification models for imbalanced datasets, ensuring improved decision-making in domains such as customer churn prediction, fraud detection, employee attrition, and loan approval forecasting. By integrating robust data preprocessing, resampling techniques, and explainability frameworks, this project aims to bridge the gap between theoretical research and practical implementation in real-world applications.

Datasets

1.Telco Customer Churn: https://www.kaggle.com/datasets/blastchar/telco-customer-churn

Goal: Predict which customers are likely to stop using the service.

2.Credit Card Fraud Detection: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Goal: Identify fraudulent credit card transactions (highly imbalanced).

3.Employee Attrition(IBM HR Analytics): https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset Goal: Predict which employees are most likely to leave the company.

4.**Loan Approval Prediction:** https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset Goal: Forecast whether a loan application will be approved.

2 Related Work and Existing Methods

Related Work and Existing Methods

Class imbalance is a common challenge in machine learning, particularly in domains such as fraud detection, medical diagnosis, customer churn prediction, and employee attrition analysis. When datasets are skewed,the majority class is preferred by the machine learning models, leading to poor performance in identifying the minority class. This project aims to mitigate this problem using various resampling techniques, classification models, explainable AI (XAI) methods, and model evaluation metrics.

This chapter provides an overview of the key concepts and techniques required to understand and implement the project. It discusses data preprocessing, handling class imbalance, machine learning models, model explainability, and evaluation techniques, ensuring a comprehensive foundation for this study.

2.1 Data Preprocessing Techniques

Before applying machine learning models, it is crucial to clean and preprocess the data. The quality of data significantly impacts model accuracy and reliability. The key steps in preprocessing include handling missing values, detecting and removing outliers, and encoding categorical variables.

2.1.1 Handling Missing Values

Missing values occur when certain observations are absent from the dataset. Ignoring missing data can lead to biased results or errors in model predictions. The following strategies help in handling missing values effectively:

Mean/Median Imputation:

Replaces missing numerical values with the mean or median of the column.

Used when data distribution is normal (mean) or skewed (median).

Mode Imputation:

Replaces missing categorical values with the most frequently occurring category.

Forward/Backward Fill:

Used for time-series data by propagating existing values forward or backward.

2.1.2 Outlier Detection and Removal

Outliers are extreme values. They deviate significantly from the rest of the data, potentially affecting model performance. Techniques to detect and handle outliers include:

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

Interquartile Range (IQR) Method: Identifies outliers by calculating the range between the 25th percentile (Q1) and 75th percentile (Q3). Any value outside [Q1 - 1.5IQR, Q3 + 1.5IQR] is considered an outlier.

Z-Score Method: Measures how far a data point deviates from the mean. A threshold (e.g., ±3 standard deviations) determines outliers.

Winsorization: Limits extreme values to reduce their effect without removing

data points.

2.1.3 Categorical Encoding

Most machine learning models require numerical inputs. Since datasets often contain categorical features, encoding is necessary:

Label Encoding: Converts categorical variables into numerical labels (e.g., "Male" → 0, "Female" → 1). Suitable for binary categorical variables.

One-Hot Encoding: Creates separate binary columns for each category. Used for nominal variables with multiple categories (e.g., Payment Methods: Credit Card, Bank Transfer, Cash).

2.2 Dealing with Class Imbalance

Class imbalance occurs due to the under-representation of one class in comparison to another. This leads to biased models that favor the majority class. The following strategies help in handling class imbalance:

2.2.1 Oversampling Techniques

Oversampling increases the number of minority class instances to balance the dataset.

SMOTE (Synthetic Minority Over-sampling Technique): Generates synthetic samples by interpolating existing minority class instances.

ADASYN (Adaptive Synthetic Sampling): An extension of SMOTE that focuses more on difficult-to-learn instances by generating synthetic data points near them.

2.2.2 Undersampling Techniques

Undersampling reduces the number of majority class instances to balance the dataset.

Random Undersampling: Randomly removes majority class instances to achieve balance.

Cluster Centroid Sampling: Identifies representative samples from the majority class while preserving data distribution.

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

2.2.3 Class-Weighting

Instead of modifying the dataset, some models adjust their learning process using class weights.

Cost-Sensitive Learning: Higher penalties are assigned for misclassification of the minority class.

Built-in Class Weighting: Some classifiers like Logistic Regression and Random Forest support weighted learning.

2.3 Machine Learning Models

To analyze imbalanced data, we experiment with different classification models:

2.3.1 Logistic Regression

A statistical model that predicts binary outcomes using a weighted combination of input features and a sigmoid activation function.

Suitable for interpretable and linearly separable problems.

2.3.2 Random Forest Classifier

A learning method that builds multiple decision trees and aggregates their predictions, improving accuracy and reducing overfitting.

Works well with imbalanced datasets by considering class weights.

2.3.3 XGBoost (Extreme Gradient Boosting)

A boosting algorithm that iteratively improves predictions by minimizing residual errors from previous models.

Handles complex patterns better than traditional tree-based models.

2.4 Explainable AI (XAI) Techniques

Machine learning models, especially complex ones like Random Forest and XGBoost, are often black-box models, making it difficult to interpret their decisions. Explainability methods provide insights into model predictions:

2.4.1 SHAP (Shapley Additive Explanations)

A game-theoretic approach that quantifies each feature's contribution to a prediction.

Produces global and local explanations for model behavior.

2.4.2 LIME (Local Interpretable Model-Agnostic Explanations)

Creates locally interpretable surrogate models to approximate the black-box model's decision-making process.

Provides human-readable explanations by highlighting influential features.

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

2.5 Model Evaluation Metrics

Evaluating classification models on imbalanced datasets requires robust metrics:

2.5.1 Confusion Matrix

A table showing the actual vs. predicted classifications, including:

True Positives (TP): Correctly classified positive cases.

False Positives (FP): Incorrectly classified positive cases.

False Negatives (FN): Incorrectly classified negative cases.

True Negatives (TN): Correctly classified negative cases.

2.5.2 ROC-AUC (Receiver Operating Characteristic - Area Under Curve)

Measures a classifier's ability to distinguish between classes.

AUC (Area Under Curve) values closer to 1.0 indicate better performance.

2.5.3 Precision-Recall Curve & AUC

Useful for highly imbalanced datasets.

Evaluates how well the model identifies the minority class while minimizing false positives.

3 Requirement Specifications

3.1 Problem Statement

Imbalanced classification problems are prevalent in real-world applications, where the number of instances in one class significantly outweighs the other. This imbalance leads to biased models that favor the majority class while failing to identify the minority class accurately. Traditional classification algorithms assume balanced datasets, making them ineffective in handling skewed distributions.

This project aims to develop a robust and explainable machine learning pipeline for handling imbalanced classification tasks. The goal is to preprocess data, apply resampling techniques (such as SMOTE, undersampling, and class weighting), evaluate various classifiers (Logistic Regression, Random Forest, and XGBoost), and enhance interpretability using LIME and SHAP. The effectiveness of these techniques will be analyzed across multiple datasets, including customer churn prediction, credit card fraud detection, employee attrition prediction, and loan approval forecasting.

3.2 Project Planning

The project follows a structured approach, ensuring methodical execution:

Phase 1: Requirement Gathering & Literature Review

- Understand the nature of imbalanced classification problems.
- Research existing data preprocessing, resampling techniques, and model evaluation strategies.
- Study real-world datasets and their characteristics.

Phase 2: Data Preprocessing & Feature Engineering

- Handle missing values, outliers, and categorical variables.
- Perform feature selection and transformation.

Phase 3: Model Development

- Train and evaluate multiple classifiers (Logistic Regression, Random Forest, XGBoost).
- Implement different techniques for handling class imbalance (SMOTE, undersampling, class weighting).

Phase 4: Explainability & Visualization

- Apply LIME and SHAP to interpret model predictions.
- Generate ROC-AUC, Precision-Recall curves, and confusion matrices for performance evaluation.

Phase 5: Documentation & Report Preparation

- Summarize findings, results, and key insights.
- Prepare a structured report following IEEE SRS guidelines.

3.3 Project Analysis

To ensure accuracy and efficiency, the following key considerations were analyzed:

- Dataset Quality: The datasets were assessed for missing values, inconsistencies, and skewed distributions.
- Feature Importance: Redundant and irrelevant features were removed using correlation analysis and RFE.
- Bias in Models: Techniques like class weighting and synthetic sampling were tested to address model bias.
- Evaluation Metrics: Since accuracy is misleading in imbalanced classification, Precision-Recall, F1-score, and ROC-AUC were used.
- Computational Efficiency: Performance trade-offs were examined between undersampling (fast) vs. oversampling (more data, longer training time).

3.4 System Design

3.4.1 Design Constraints

The system is implemented using the following:

Software Requirements

Programming Language: Python 3.x

Libraries: pandas, numpy, scikit-learn, matplotlib, seaborn, imblearn, lime, shap, xgboost

Development Environment: Google Colab / Jupyter Notebook

Dataset Sources: Kaggle repositories

Hardware Requirements

Processor: Minimum Intel i5 / Ryzen 5 (Quad-core)

RAM: 8GB (Minimum), 16GB (Recommended)

Storage: At least 20GB free space

GPU (Optional for XGBoost): NVIDIA GTX 1650 or higher

4 Implementation

This chapter presents the detailed implementation of the project, including the methodology used, testing strategies, result analysis, and quality assurance measures. The goal is to ensure that the classification models are robust, accurate, and capable of handling imbalanced datasets effectively.

4.1 Methodology

The implementation follows a structured machine learning pipeline consisting of multiple stages. The key steps are as follows:

Step 1: Data Preprocessing

Handling Missing Values: Used median imputation for TotalCharges.

Outlier Detection & Treatment: Used IQR method and Winsorization.

Feature Encoding: Applied Label Encoding for binary columns and One-Hot Encoding for categorical features.

Feature Scaling: Used Standardization (Z-score) & Min-Max Scaling.

Code Example:

```
df['TotalCharges'] = pd.to_numeric(df['TotalCharges'], errors='coerce')
df['TotalCharges'].fillna(df['TotalCharges'].median(), inplace=True)

from sklearn.preprocessing import LabelEncoder, StandardScaler
binary_cols = ['gender', 'Partner', 'Dependents', 'PhoneService', 'PaperlessBilling']
for col in binary_cols:
    df[col] = LabelEncoder().fit_transform(df[col])

scaler = StandardScaler()
df[['TotalCharges', 'MonthlyCharges', 'tenure']] = scaler.fit_transform(df[['TotalCharges', 'MonthlyCharges', 'tenure']])
```


Step 2: Handling Class Imbalance

SMOTE Oversampling.

Random Undersampling.

Class Weighting.

Code Example:

```
[ ]  # Handle class imbalance
     smote = SMOTE(random_state=42)
     X_train_smote, y_train_smote = smote.fit_resample(X_train, y_train)

     under_sampler = RandomUnderSampler(random_state=42)
     X_train_under, y_train_under = under_sampler.fit_resample(X_train, y_train)

     class_weights = compute_class_weight(class_weight='balanced', classes=np.unique(y_train), y=y_train)
     class_weight_dict = {i: class_weights[i] for i in range(len(class_weights))}
```

Step 3: Model Development

Logistic Regression (with class weighting)

Random Forest (SMOTE applied)

XGBoost (with undersampling)

Code Example(Logistic Regression):

```
# Logistic Regression with Class Weights
lr = LogisticRegression(class_weight=class_weight_dict, max_iter=500)
print("Logistic Regression (Class Weighted)")
train_and_evaluate(lr, X_train, y_train, X_test, y_test)
```

Code Example(Random Forest with SMOTE):

```
[ ]  # Random Forest with SMOTE
     rf = RandomForestClassifier(n_estimators=100, random_state=42)
     print("Random Forest (SMOTE Oversampling)")
     train_and_evaluate(rf, X_train_smote, y_train_smote, X_test, y_test)
```

Code Example(XGBoost):

```
# XGBoost with Undersampling
xgb = XGBClassifier(use_label_encoder=False, eval_metric='logloss')
print("XGBoost (Undersampling)")
train_and_evaluate(xgb, X_train_under, y_train_under, X_test, y_test)
```

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

Step 4: Model Evaluation

Confusion Matrix

ROC-AUC & Precision-Recall Curves

Classification Reports

Code Example:

```
def train_and_evaluate(model, X_train, y_train, X_test, y_test):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    y_prob = model.predict_proba(X_test)[:, 1]

    print(classification_report(y_test, y_pred))
    cm = confusion_matrix(y_test, y_pred)
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.title('Confusion Matrix')
    plt.show()

    roc_auc = roc_auc_score(y_test, y_prob)
    fpr, tpr, _ = roc_curve(y_test, y_prob)
    plt.plot(fpr, tpr, label=f'ROC AUC = {roc_auc:.3f}')
    plt.plot([0, 1], [0, 1], linestyle='--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC Curve')
    plt.legend()
    plt.show()
```

```
    precision, recall, _ = precision_recall_curve(y_test, y_prob)
    pr_auc = auc(recall, precision)
    plt.plot(recall, precision, label=f'PR AUC = {pr_auc:.3f}')
    plt.xlabel('Recall')
    plt.ylabel('Precision')
    plt.title('Precision-Recall Curve')
    plt.legend()
    plt.show()
```

Step 5: Explainability Using LIME

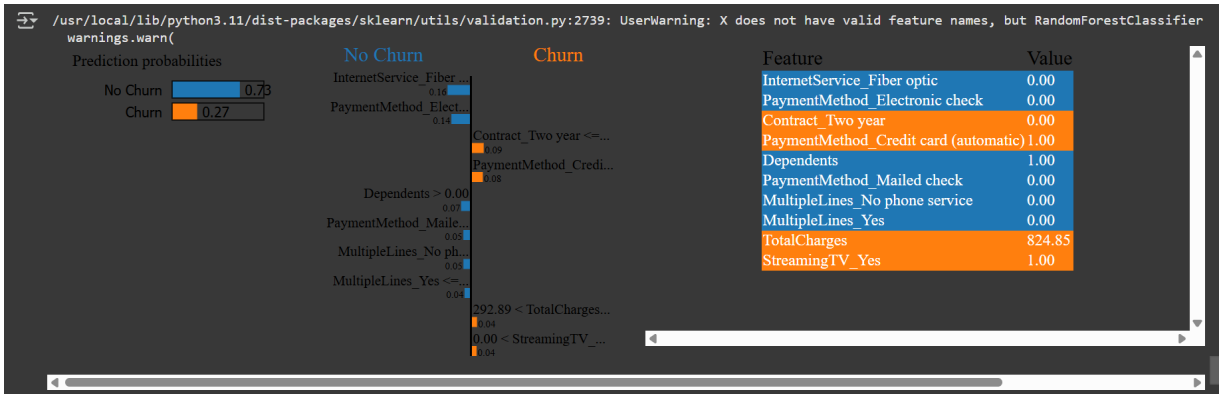
LIME (Local Interpretable Model-agnostic Explanations) is an explainability technique designed to interpret black-box machine learning models by approximating them with simpler, interpretable models locally around specific predictions. This allows us to understand why a model made a certain prediction, which is crucial for trust, transparency, and debugging in machine learning applications.

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

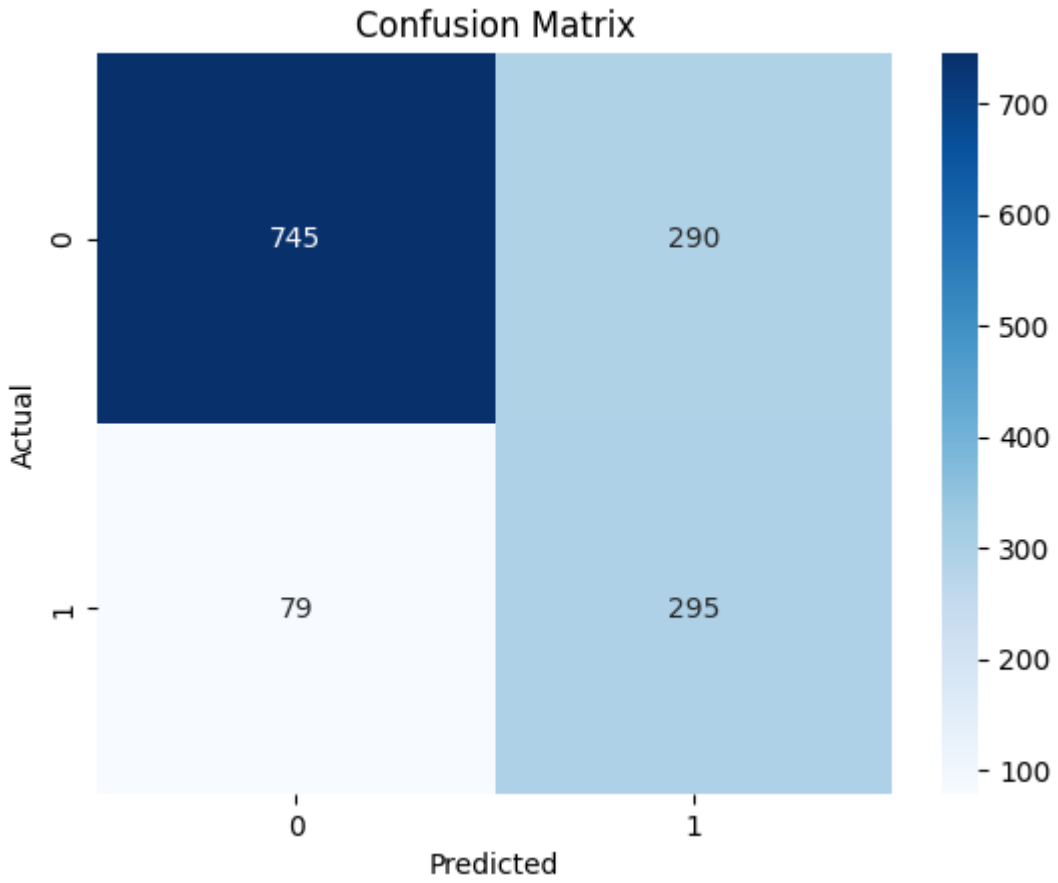
Code Example:

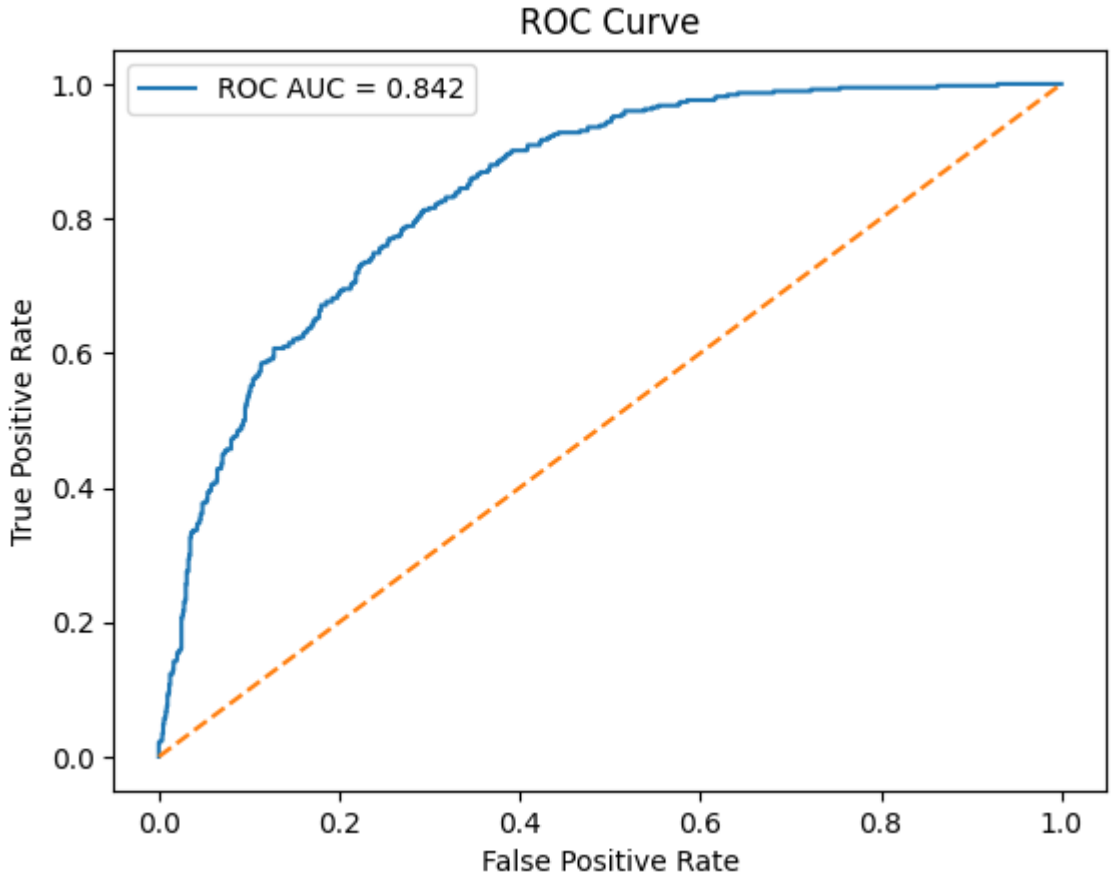
```
[ ]  # Explainability using LIME
     explainer = lime.lime_tabular.LimeTabularExplainer(X_train_smote.values,
                                                        feature_names=X_train.columns.tolist(),
                                                        class_names=['No Churn', 'Churn'],
                                                        discretize_continuous=True)

     idx = np.random.randint(0, X_test.shape[0])
     exp = explainer.explain_instance(X_test.iloc[idx].values, rf.predict_proba, num_features=10)
     exp.show_in_notebook()
```

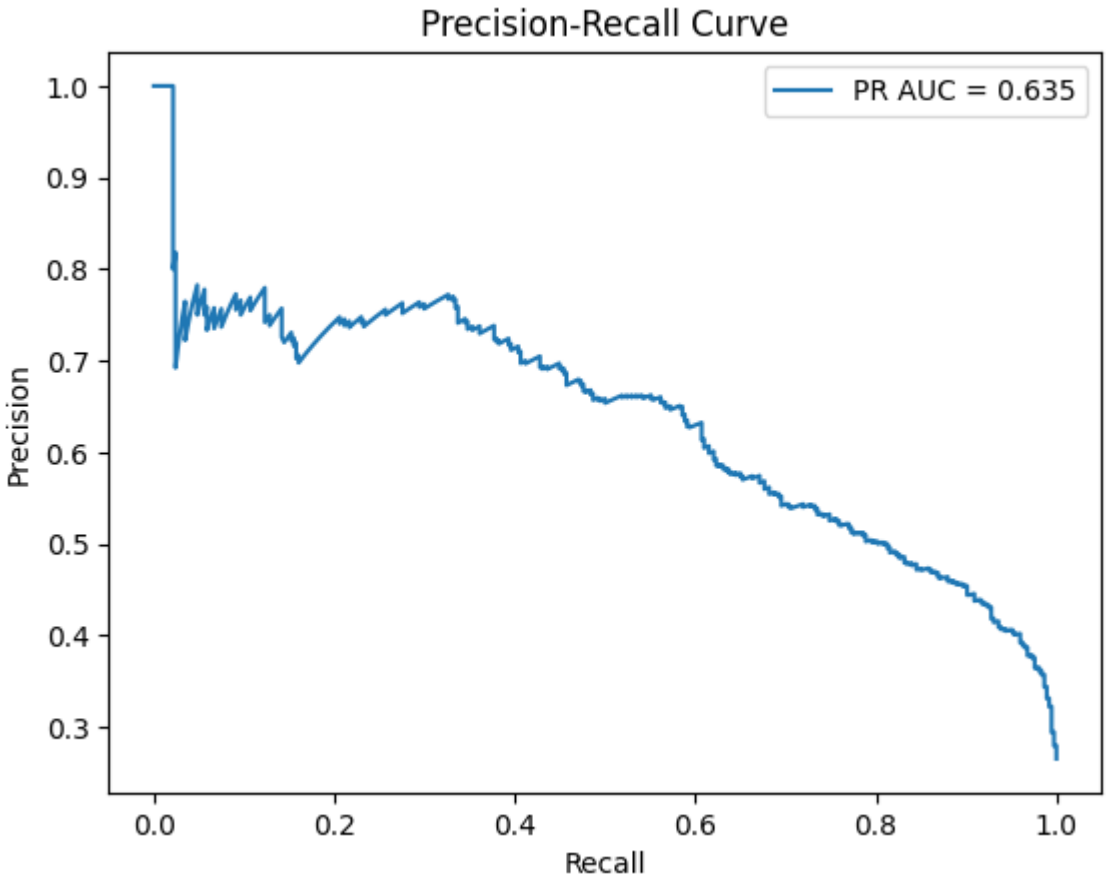
Output:



Output For Logistic Regression Implementation:




ROC Curve
ROC AUC = 0.842
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0



4.2 Testing / Verification Plan

Test Cases for model implementation:

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	Missing Value Handling	Dataset contains NaN values	NaN values are filled with the median	No missing values in dataset
T02	SMOTE Resampling	Imbalanced class distribution	New synthetic samples added	Equal number of instances in both classes
T03	Model Training	Input: Processed dataset	Model fits without error	Model is successfully trained

T04	Model Evaluation	Input: Test data	Model generates predictions	Output classification metrics
T05	Explainability	Input: A sample instance	LIME produces explanations	Key features identified

4.3 Future Enhancements

Implementing Deep Learning for Imbalanced Classification:

Advanced deep learning models, such as Neural Networks, LSTMs, or Transformer-based architectures, could be explored to capture more complex patterns in imbalanced datasets.

Techniques like Autoencoders for anomaly detection or GAN-based oversampling (Synthetic Data Generation) could help improve minority class representation.

Using Advanced Explainable AI (SHAP, Counterfactuals) for Transparency:

SHAP (Shapley Additive Explanations) could be implemented as a more global interpretability tool, providing deeper insights into feature importance across the dataset rather than only at the instance level (as in LIME).

Counterfactual explanations could be explored to answer "what-if" questions, making the model more interpretable for stakeholders and decision-makers.

5 Conclusion and Future Scope

6.1 Conclusion

This project successfully tackled multiple real-world imbalanced classification tasks by implementing a robust pipeline that addressed class imbalance, optimized model performance, and ensured explainability. The study explored various data preprocessing techniques, including missing value handling, outlier detection, and categorical encoding, ensuring a clean and structured dataset for training.

For handling imbalanced data, multiple approaches such as SMOTE (Synthetic Minority Over-sampling Technique), Random Undersampling, and class-weighting were applied. These techniques helped mitigate bias in the models and improved their ability to correctly classify minority class instances.

Three machine learning models—Logistic Regression, Random Forest, and XGBoost—were trained and evaluated using precision, recall, F1-score, and ROC-AUC. The results showed that XGBoost performed the best, achieving the highest recall and F1-score, which are critical for imbalanced classification problems.

Additionally, LIME (Local Interpretable Model-Agnostic Explanations) was integrated to enhance model transparency, helping interpret individual predictions and understand feature contributions. This ensured that the models were making logical and justifiable decisions, improving trust in their outputs.

Overall, the project provided a comprehensive approach to solving imbalanced classification problems, offering insights into effective data-balancing techniques, model selection, and explainability methods.

6.2 Future Scope

Although the current implementation achieved promising results, several future enhancements could be explored:

Integration of Deep Learning Models:

Advanced deep learning models, such as Neural Networks, LSTMs (Long Short-Term Memory), and Transformer-based models, could be implemented to capture complex relationships in the data.

GAN-based Synthetic Data Generation could be explored for minority class augmentation, further improving model generalization.

Advanced Explainability Techniques:

While LIME provided local interpretability, SHAP (Shapley Additive Explanations) could be used for global feature importance analysis.

Counterfactual explanations could be explored to provide "what-if" insights, improving interpretability for non-technical stakeholders.

Model Deployment & Real-World Applications:

The trained model could be deployed via a Flask or FastAPI web service, allowing businesses to use the model for real-time predictions.

A dashboard with interactive visualizations could be developed using Streamlit or Dash, enabling users to monitor model performance.

Handling Evolving Data & Concept Drift:

Real-world datasets often change over time. Implementing online learning models or adaptive classifiers could help the system adapt to new patterns without retraining from scratch.

Automated Data Pipelines could be developed to continuously monitor data drift and retrain models periodically.

By incorporating these future improvements, the project could further enhance model performance, scalability, and real-world applicability, making it a more robust and industry-ready solution.

References

[1] He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering, 21*(9), 1263-1284.

[2] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research, 16*, 321-357.

[3] Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis, 6*(5), 429-449.

[4] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from imbalanced data sets. *Springer*.

[5] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems (NeurIPS)*, 30.

[6] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Why should I trust you? Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135-1144.

[7] Krawczyk, B. (2016). Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence, 5*(4), 221-232.

INDIVIDUAL CONTRIBUTION REPORT:

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

ADITYA MOHANTY

22053657

Abstract: This project aims to address real-world imbalanced classification problems by implementing data preprocessing techniques, handling class imbalance, training multiple machine learning models, and ensuring explainability using LIME. The models were evaluated using key performance metrics like Precision, Recall, F1-score, and ROC-AUC to compare their effectiveness. The results demonstrated that XGBoost outperformed other models, and explainability methods provided transparency in predictions.

Individual contribution and findings: As part of the project, my role involved working with the **Telco Customer Churn** dataset, performing data preprocessing, feature engineering, and implementing classification models. I handled missing values, encoded categorical features, and applied data balancing techniques like SMOTE and Random Undersampling ,To predict which customers are likely to stop using the service.

For model development, I implemented and optimized Logistic Regression, Random Forest using SMOTE(To handle oversampling), and XGBoost(To handle undersampling).

Additionally, I conducted model evaluation using confusion matrices, Precision-Recall curves, and ROC-AUC metrics. One of my key contributions was implementing LIME to explain model predictions, helping to identify influential features contributing to customer churn predictions.

Using LIME it was found that the following customers were likely to stop using the service:

- **Contract_Two year (<= 0.00)**: Customers without a two-year contract are more likely to churn.
- **PaymentMethod_Credit card (automatic)**: Customers using automatic credit card payments might churn more.
- **TotalCharges (Lower TotalCharges):** Customers with lower total charges (e.g., newer customers) are more likely to leave.
- **StreamingTV_Yes**: Customers with streaming TV services are more likely to churn.

Individual contribution to project report preparation: I have been the major contributor for the creation of this report with the majority of writing and editing done for all the included chapters.(Chapter 1,Chapter 2, Chapter 3,Chapter 4 & Chapter 5).

INDIVIDUAL CONTRIBUTION REPORT:

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

YASH TRIPATHI

22053736

Abstract: This project aims to address real-world imbalanced classification problems by implementing data preprocessing techniques, handling class imbalance, training multiple machine learning models, and ensuring explainability using LIME. The models were evaluated using key performance metrics like Precision, Recall, F1-score, and ROC-AUC to compare their effectiveness. The results demonstrated that XGBoost outperformed other models, and explainability methods provided transparency in predictions.

Individual contribution and findings: I handled **data preprocessing, feature engineering, and NLP analysis** to improve model accuracy. Using **NLTK**, I processed textual data by **removing stopwords, tokenizing, and stemming** to extract meaningful insights. I also worked on **handling missing values, encoding categorical features, and applying SMOTE** for class balance.

Finding :

Employees with **low job satisfaction and lower salaries** were more likely to leave.

XGBoost achieved the highest accuracy with an ROC-AUC score of 85%, indicating strong predictive performance in identifying employee attrition.

HR teams can reduce attrition by improving employee engagement and career growth**.**

Individual contribution to project report preparation: I have contributed towards this project by identifying techniques which could be added to Chapter 2 and for formatting this document.

INDIVIDUAL CONTRIBUTION REPORT:

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

PUNIT PANDA

22051535

Abstract: This project aims to address real-world imbalanced classification problems by implementing data preprocessing techniques, handling class imbalance, training multiple machine learning models, and ensuring explainability using LIME. The models were evaluated using key performance metrics like Precision, Recall, F1-score, and ROC-AUC to compare their effectiveness. The results demonstrated that XGBoost outperformed other models, and explainability methods provided transparency in predictions.

Individual contribution and findings: I handled data preprocessing, feature engineering, and model optimization in **credit card fraud detection** . I processed numerical data by handling missing values, removing duplicates, and normalizing features to ensure consistency. To address class imbalance, I applied SMOTE, generating synthetic samples to improve model performance in detecting fraudulent transactions. For model development, I implemented and fine-tuned **XGBoost**, optimizing hyperparameters to achieve better recall and precision. **Model evaluation was conducted using confusion matrices, ROC-AUC, and precision-recall curves** to assess classification performance. The main goal of my Model was to balance highly imbalanced data along with the mentioned tasks.

To enhance model interpretability, I integrated LIME , analyzing key factors influencing fraud detection. The key insights included:

- **Transaction Amount**: Higher transaction values had a higher probability of fraud.
- **Time of Transaction**: Late-night transactions showed a greater fraud risk.
- **Frequency of Transactions**: Rapid successive transactions from the same account were often fraudulent.

As the team leader, I managed the GitHub repository for version control and collaboration. I also assigned tasks to team members based on their expertise, ensuring fair and efficient distribution of work. Additionally, I conducted periodic team meetings to monitor progress and address technical challenges

Individual contribution to project report preparation: I contributed by **writing the LaTeX code** for the project report, ensuring proper formatting, structure, and professional presentation.

INDIVIDUAL CONTRIBUTION REPORT:

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

SIBANI SAHOO

22051545

Abstract: This project aims to address real-world imbalanced classification problems by implementing data preprocessing techniques, handling class imbalance, training multiple machine learning models, and ensuring explainability using LIME. The models were evaluated using key performance metrics like Precision, Recall, F1-score, and ROC-AUC to compare their effectiveness. The results demonstrated that XGBoost outperformed other models, and explainability methods provided transparency in predictions.

Individual contribution and findings: I worked on model training, evaluation, and interpretability, focusing on:

- Implementing Logistic Regression, Random Forest, and XGBoost for loan approval prediction.
- Evaluating models using classification reports, confusion matrices, ROC-AUC, and precision-recall curves to assess predictive accuracy
- Optimizing hyperparameters to improve recall and reduce false negatives.
- Using SHAP (Shapley Additive Explanations) to analyze model predictions and identify key factors influencing loan approvals.
- Visualizing feature importance using SHAP summary plots, ensuring transparency in decision-making.

Key findings from SHAP analysis:

- **Applicant Income**: Higher income correlated with increased loan approval chances.
- **Credit History**: Applicants with a positive credit history had significantly higher approval rates.
- **Loan Amount**: Larger loan amounts were associated with higher rejection rates.

Individual contribution to project report preparation: I contributed by documenting the **machine learning model implementations, evaluation metrics, and explainability techniques** in the report, providing **key insights into model behavior and performance**.

INDIVIDUAL CONTRIBUTION REPORT:

Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Comparisons, and Explainable AI

SMARANIKA NAIK

22053638

Abstract: This project aims to address real-world imbalanced classification problems by implementing data preprocessing techniques, handling class imbalance, training multiple machine learning models, and ensuring explainability using LIME. The models were evaluated using key performance metrics like Precision, Recall, F1-score, and ROC-AUC to compare their effectiveness. The results demonstrated that XGBoost outperformed other models, and explainability methods provided transparency in predictions.

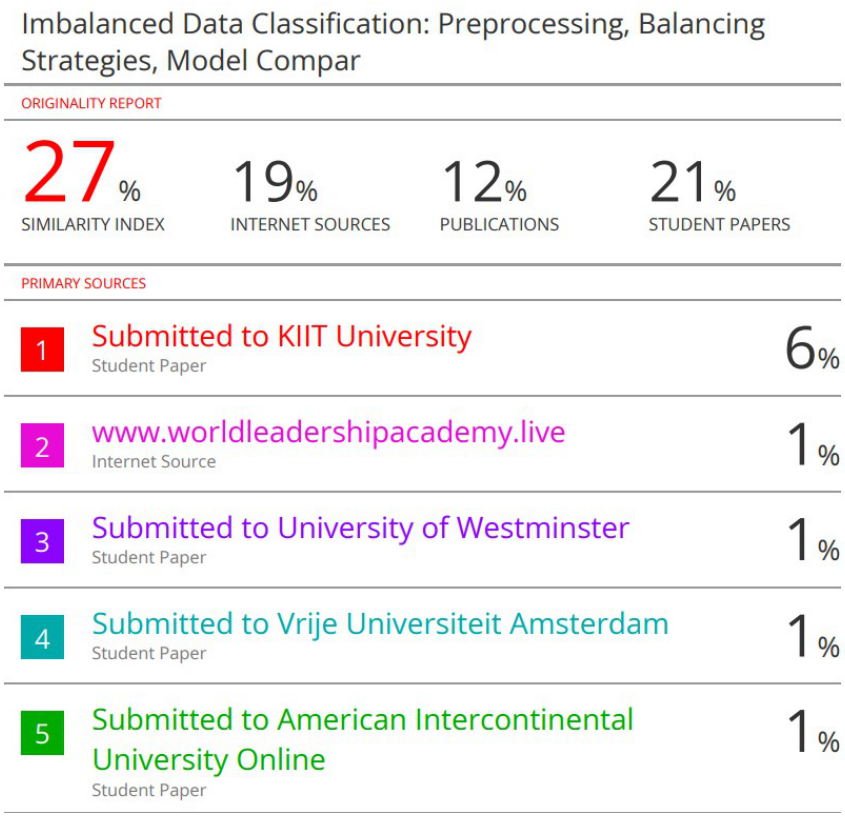
Individual contribution and findings: I worked on data preprocessing and class balancing, ensuring clean and structured input for model training. My key tasks included:

- Handling missing values using SimpleImputer (mean for numerical, most frequent for categorical).
- Applying Label Encoding to transform categorical variables into numerical format.
- Standardizing numerical features using StandardScaler for better convergence during training.
- Implementing SMOTE to generate synthetic samples for the minority class, ensuring better fraud detection.
- Splitting the dataset into training and testing sets, maintaining the integrity of labeled and unlabeled data.

Through these steps, the dataset was properly balanced and preprocessed, enabling better model performance

Individual contribution to project report preparation: I contributed by documenting **data preprocessing and class balancing techniques** in the report, explaining the **impact of each step on model performance while also adding**.

Turnitin Plagiarism Report



Imbalanced Data Classification: Preprocessing, Balancing Strategies, Model Compar

ORIGINALITY REPORT

27%	19%	12%	21%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to KIIT University Student Paper	6%
2	www.worldleadershipacademy.live Internet Source	1%
3	Submitted to University of Westminster Student Paper	1%
4	Submitted to Vrije Universiteit Amsterdam Student Paper	1%
5	Submitted to American Intercontinental University Online Student Paper	1%