Project #1: Customer Churn Detection

Introduction

Customer churn prediction is the process of identifying customers who are likely to cancel their subscription to a service based on their usage patterns and behaviors. It is a critical task for businesses, as retaining existing customers is often more cost-effective than acquiring new ones.

Project Overview

The goal of this project is to build a predictive model to determine whether a customer is likely to exit (churn) based on historical data about their behavior and interactions with the service. The insights generated by the model will enable businesses to take proactive measures to retain customers and reduce churn rates.

Dataset

The dataset used in this project is **Churn_Modelling.csv**, which includes various features related to customer demographics, financial behavior, and service engagement.

Features:

1. **RowNumber**: Row index of the data (used for reference only).
2. **CustomerId**: Unique identifier for each customer.
3. **Surname**: Customer's last name.
4. **CreditScore**: Customer's credit score.
5. **Geography**: Customer's geographic location.
6. **Gender**: Customer's gender.
7. **Age**: Customer's age.
8. **Tenure**: Number of years the customer has been with the company.
9. **Balance**: Customer's account balance.
10. **NumOfProducts**: Number of products the customer uses.
11. **HasCrCard**: Whether the customer has a credit card (1 = Yes, 0 = No).
12. **IsActiveMember**: Whether the customer is an active member (1 = Yes, 0 = No).
13. **EstimatedSalary**: Customer's estimated annual salary.

Label (Target):

- **Exited**: Indicates whether the customer has exited the service (1 = Yes, 0 = No).

Steps for the Project: Customer Churn Detection

Workflow:

1. **Reading the Dataset:**
    - Load the `Churn_Modelling.csv` file using a library like pandas.
    - Perform data exploration to understand the dataset, check for missing or inconsistent values, and get a general sense of the features.
2. **Data Visualization:**
    - Create visualizations to better understand the dataset, including:
        - Distribution of numerical features (e.g., `Age`, `CreditScore`, `Balance`, `EstimatedSalary`).
        - Categorical feature distributions (e.g., `Gender`, `Geography`).
        - Correlation heatmap to identify relationships between features.
        - Distribution of the target variable (`Exited`) to check for class imbalance.
3. **Feature Engineering:**
    - Process non-numerical columns such as `Geography` and `Gender` by encoding them into numerical values (e.g., using One-Hot Encoding or Label Encoding).
    - Ensure all features used in the model are clean, relevant, and properly formatted.
4. **Splitting the Dataset:**
    - Split the dataset into training and testing sets (e.g., 80% for training and 20% for testing) using tools like `train_test_split` from sklearn.
5. **Oversampling (if needed):**
    - If the dataset is imbalanced (e.g., significantly fewer customers who exited), use techniques such as **SMOTE (Synthetic Minority Oversampling Technique)** to balance the target classes.
6. **Feature Standardization:**
    - Standardize the features so that they are on the same scale (e.g., mean = 0, standard deviation = 1). This step is particularly important for algorithms like **SVM** and **KNN**.
7. **Model Evaluation:**
    - Train and evaluate the following algorithms:
        - **K-Nearest Neighbors (KNN)**
        - **Naive Bayes**
        - **Support Vector Machine (SVM)**
        - **Decision Tree (DT)**
    - Use the training set to fit the models and the testing set to evaluate their performance.
8. **Comparing Algorithms:**
    - Compare the models using evaluation metrics such as:
        - **Accuracy**
        - **Precision and Recall**
        - **F1-Score**
        - **ROC-AUC Score**
    - Identify the best-performing model based on the overall evaluation.

Deliverables:

- A **trained model** capable of predicting whether a customer will churn or not.
- A **detailed comparison** of the performance of the selected algorithms (KNN, Naive Bayes, SVM, DT) with relevant evaluation metrics.
- A **GitHub repository** containing:
    - **Complete project code**, including:
        - Dataset loading and preprocessing.
        - Data visualization.
        - Feature engineering and standardization.
        - Oversampling (if applied).
        - Model training, evaluation, and comparison.
    - **Project documentation** with:
        - An explanation of the steps.
        - Results and findings.
    - **Visualization of results**, such as performance metrics and comparison charts.
    - **README file** explaining the project and how to run it.