# Customer Churn Prediction using Machine Learning

## Objective

* Create a predictive model to determine which customers are likely to cancel a subscription.

## Dataset: Customer Churn

* **Source**: Provided CSV file (Customer-Churn.csv)
* **Target Column**: Churn (Yes/No)
* **Features**: Customer demographics, account info, services subscribed (e.g., InternetService, MonthlyCharges)

## Step-by-Step Process

### 1. Importing Libraries

* import pandas as pd
* import numpy as np
* These are the core Python libraries used for data manipulation and numerical calculations

### 2. Load Dataset

* df = pd.read\_csv('Telco-Customer-Churn.csv')
* Loads the data into a DataFrame for analysis.

### Data Preprocessing

### Handle Missing Values

* df['TotalCharges'] = pd.to\_numeric(df['TotalCharges'], errors='coerce')

The TotalCharges column has empty strings.

Converted it to numeric using pd.to\_numeric() with errors='coerce', which transforms invalid entries into NaN.

**Imputation**: Used SimpleImputer(strategy='mean') to replace NaN with the column's mean value.

### Encode Categorical Variables

* Applied **pd.get\_dummies()** to convert categorical columns to numeric dummy variables.

### Feature Scaling

* Standardized numerical features using **StandardScaler from sklearn**.
* Scaling ensures all features are on the same scale (mean=0, std=1), which is critical for algorithms like **PCA** and **Logistic Regression.**

### Dimensionality Reduction (PCA)

* Applied **Principal Component Analysis (PCA)** to reduce feature dimensions to 10.
* **PCA** transforms features into new components that retain most of the original variance.
* Calculated **explained\_variance\_ratio\_** to analyze how much variance each component explains.
* Used **np.cumsum()** to compute cumulative variance.

### Data Balancing using SMOTE

* Churn datasets are often imbalanced: more '**No**' than '**Yes'** churns.
* Used **SMOTE (Synthetic Minority Oversampling Technique)** to create synthetic samples of the minority class.
* Ensures equal representation of both classes for better model performance

### Model Training

### Train-Test Split

* Used **train\_test\_split()** to divide the data into 80% training and 20% testing subsets.
* Ensures fair evaluation of model on unseen data.

### Logistic Regression

* Selected Logistic Regression for binary classification due to its simplicity and interpretability.
* Trained on the **PCA-transformed, SMOTE-balanced training data**.
* **from sklearn.linear\_model import LogisticRegression**
* **model = LogisticRegression()**
* **model.fit(X\_train\_balanced, y\_train\_balanced)**
* Trains a logistic regression classifier on the balanced dataset.

### 5.Model Evaluation

### Confusion Matrix

* Shows True Positives, True Negatives, False Positives, and False Negatives.
* Helps identify where the model makes errors.

**Confusion Matrix**: **TP, TN, FP, FN**

**Classification Report**:

* **Precision**: How many predicted churns were actual churns.
* **Recall**: How many actual churns were correctly predicted.
* **F1-Score**: Balance between precision and recall.

**ROC AUC**:

**ROC (Receiver Operating Characteristic) Curve**

**AUC (Area Under the Curve)**

* Measures the model's ability to distinguish between classes.
* AUC = 0.85 means the model is very effective.
* Model's ability to distinguish between churned and not churned customers.

### Interpretation

* Accuracy: 75%
* Precision (Churn): 52%
* Recall (Churn): 81%
* F1 Score: 63%
* ROC AUC Score: 0.85
* High recall means the model is excellent at capturing actual churns. Slightly lower precision implies some false positives.

## ✅ Final Results Example

* Confusion Matrix:
* [[758 278]
* [ 71 302]]

**Classification Report:**

* precision recall f1-score support
* False 0.91 0.73 0.81 1036
* True 0.52 0.81 0.63 373
* accuracy 0.75 1409
* macro avg 0.72 0.77 0.72 1409
* weighted avg 0.81 0.75 0.77 1409

**ROC AUC Score: 0.85**

## Key Learnings **:**

* Data preprocessing (type conversion, missing value handling, encoding) is crucial.
* Feature scaling and dimensionality reduction simplify modeling.
* Class balancing using SMOTE significantly improves recall.
* Logistic regression is a good starting model for churn prediction.