**Predictive Analysis**

**Lab-6**

**Isha Yadav**

**500107798**

**Q. Customer Churn Prediction (regression)**

**Dataset: Telecom customer data (e.g., from Kaggle): Suggested link - https://www.kaggle.com/datasets/abhinav89/telecom-customer**

**Data preprocessing (handling missing values, outliers, feature scaling)**

**Exploratory data analysis (EDA) to identify potential predictors**

**Building a simple linear regression model**

**Evaluating model performance using accuracy, precision, recall, F1-score.**

**Code:**

# Import necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, confusion\_matrix

from scipy import stats

# Load the dataset

df = pd.read\_csv('/content/Telecom\_customer\_churn.csv')

# Clean column names

df.columns = df.columns.str.strip().str.lower()

# Data Preprocessing

# Handling missing values

df.fillna(method='ffill', inplace=True)

# Handle non-numeric categorical variables

categorical\_cols = df.select\_dtypes(include=['object']).columns

df = pd.get\_dummies(df, columns=categorical\_cols, drop\_first=True) # One-hot encoding

# Handle outliers using Z-score

df = df[(np.abs(stats.zscore(df.select\_dtypes(include=[np.number]))) < 3).all(axis=1)]

# Feature Scaling using StandardScaler (excluding the 'churn' column)

scaler = StandardScaler()

numeric\_features = df.drop(columns=['churn']).select\_dtypes(include=[np.number]).columns # Exclude 'churn'

df[numeric\_features] = scaler.fit\_transform(df[numeric\_features])

# Exploratory Data Analysis (EDA)

# Visualizing Churn distribution

sns.countplot(x='churn', data=df)

plt.title('Churn Distribution')

plt.show()

# Building the Linear Regression Model

# Splitting the dataset into features and target

X = df.drop(columns=['churn']) # Features

y = df['churn'] # Target

# Splitting into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Initialize and train the linear regression model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Predict on test data

y\_pred = model.predict(X\_test)

y\_pred = [1 if x > 0.5 else 0 for x in y\_pred] # Binarizing the predictions

# Evaluating the Model

accuracy = accuracy\_score(y\_test, y\_pred)

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

f1 = f1\_score(y\_test, y\_pred)

# Print the evaluation metrics

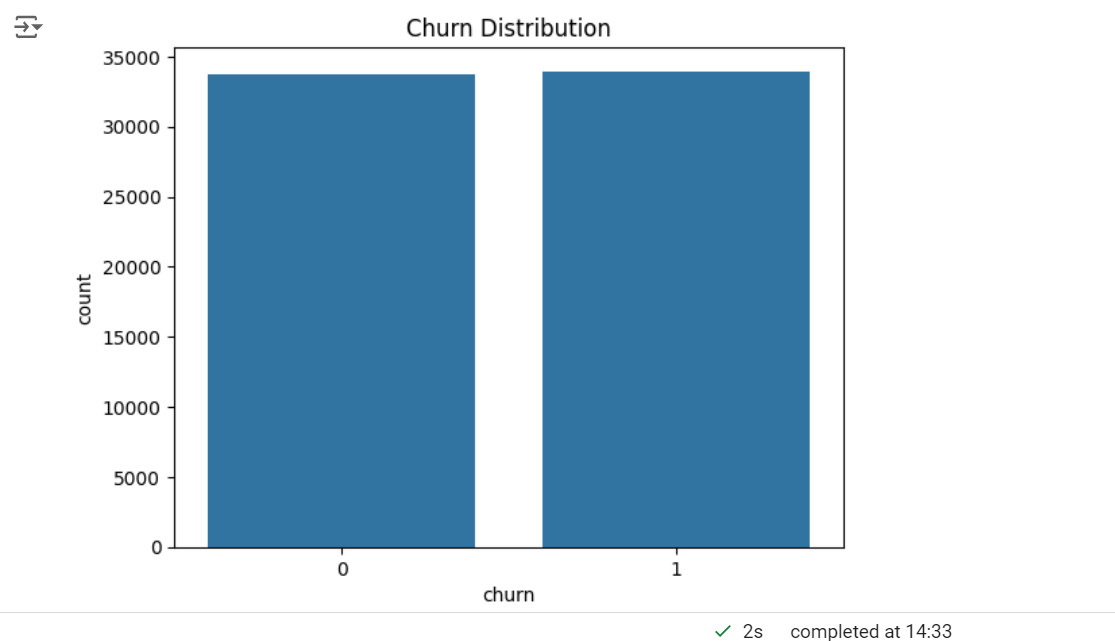
print(f"Accuracy: {accuracy:.2f}")

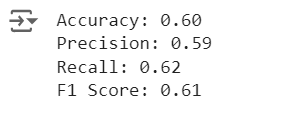
print(f"Precision: {precision:.2f}")

print(f"Recall: {recall:.2f}")

print(f"F1 Score: {f1:.2f}")

**OUTPUT:**

****

****