PREDICTIVE ANALYSIS

LAB-6

NAME : SHALU PRIYA

ROLL : R2142220839

SAP I'D : 500107625

BATCH : 4(NH)

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.

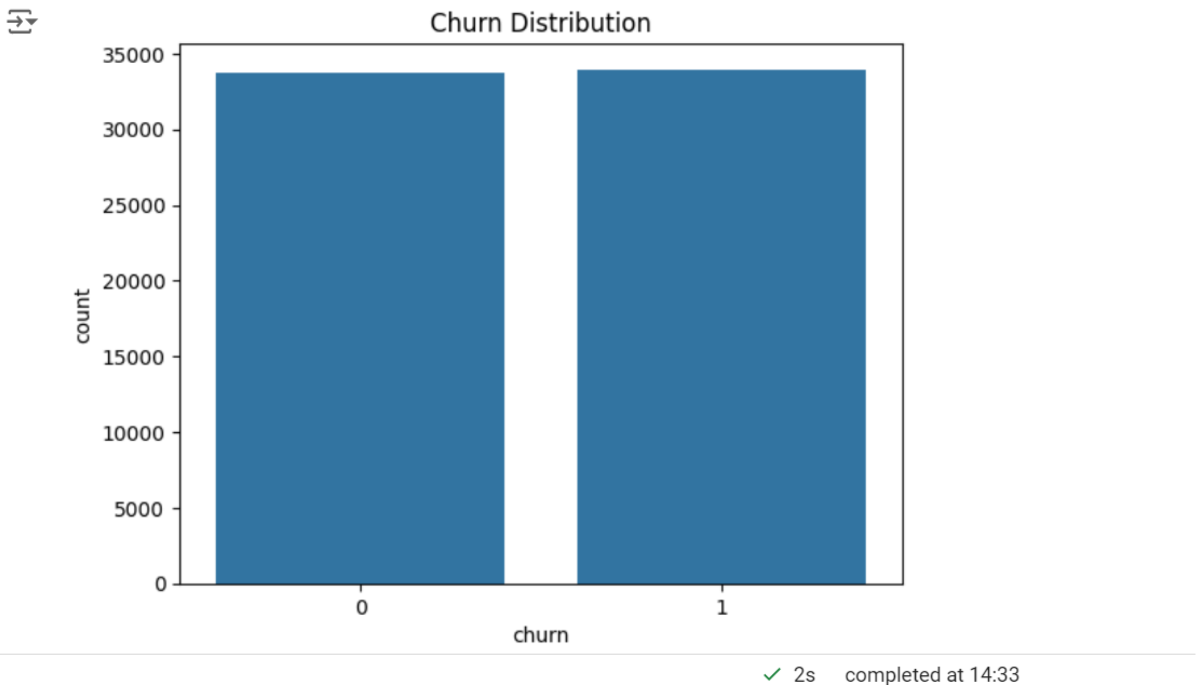
Code:

```
# 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)]
```

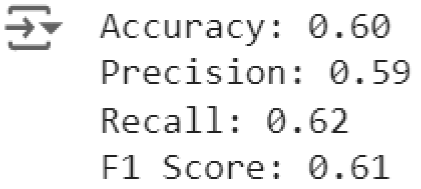
```
# Feature Scaling using StandardScaler (excluding the 'churn' column)
scaler = StandardScaler()
numeric_features = df.drop(columns=['churn']).select_dtypes(include=[np.number]).columns  # Exclude 'churn'
df[numeric_features] = scaler.fit_transform(df[numeric_features])

# 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:



✓ 2s completed at 14:33



```
Accuracy: 0.60
Precision: 0.59
Recall: 0.62
F1 Score: 0.61
```