GithubLink:https://github.com/RamyaV-29/predicting-customer-churn.git

PROJECT TITLE: PREDICTING CUSTOMER CHURN USING MACHINE LEARNING TO UNCOVER HIDDEN PATTERNS

PHASE-3

STUDENT NAME:RAMYA.V

REGISTER NUMBER:623023104042

INSTITUTION: Tagore Institute Of Engineering and Technology

DEPARTMENT: Computer Science And Engineering

DATE OF SUBMISSION:16/05/2025

1.PROBLEM STATEMENT:

Customer churn—when clients stop using a company's product or service—is a critical issue that directly impacts profitability and long-term sustainability. Traditional churn prediction models often fail to detect subtle behavioral signals and complex interactions that lead to customer attrition.

This project aims to leverage machine learning techniques to analyze customer data, uncover hidden patterns, and accurately predict churn.

By identifying high-risk customers early, businesses can develop targeted retention strategies. The ultimate goal is to reduce churn rates, improve customer satisfaction, and enhance business performance through data-driven decision-making.

2.ABSTRACT:

Objective: This study aims to develop a predictive model for customer churn by leveraging machine learning techniques to identify key behavioral and transactional patterns that signal potential attrition.

Methodology: Various machine learning algorithms such as decision trees, random

forests, and logistic regression are applied to customer datasets. Feature engineering and data preprocessing techniques are employed to enhance model accuracy and uncover hidden churn indicators.

Findings: The proposed models successfully identify high-risk customers with significant precision, enabling proactive retention strategies. Insights gained from pattern discovery provide deeper understanding of churn triggers, leading to improved customer relationship management.

3.SYSTEM REQUIREMENT:

Hardware Requirements:

- Minimum: Intel i5 Processor, 8GB RAM, 256GB SSD
- Recommended: Intel i7 or higher, 16GB+ RAM, GPU (NVIDIA CUDA-enabled) for faster model training

Software Requirements:

- Operating System: Windows 10/11, Linux, or macOS
- Development Tools: Python (with libraries such as Pandas, Scikit-learn, NumPy, Matplotlib, XGBoost), Jupyter Notebook or VS Code

Dataset & Environment:

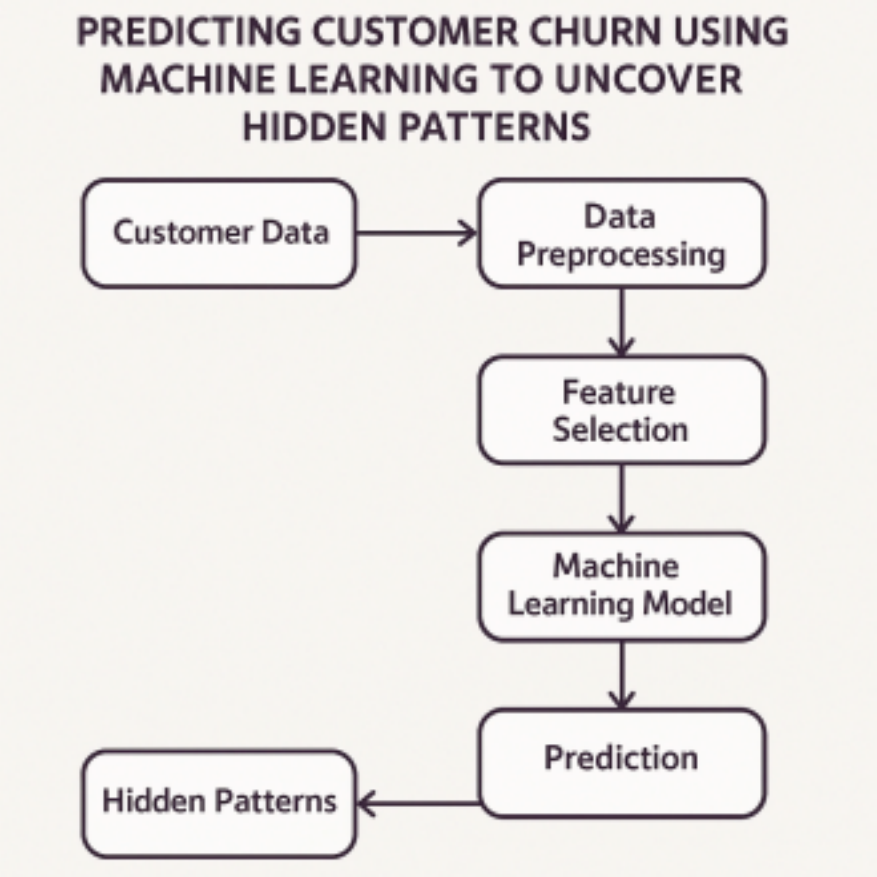
- Access to structured customer data (e.g., demographics, usage, transaction history)
- Environment: Anaconda or virtual environment for managing dependencies and reproducibility

4.OBJECTIVES:

- **Analyze customer data** to identify trends, behaviors, and attributes associated with churn.
- **Apply various machine learning algorithms** to build predictive models for customer churn detection.
- **Evaluate model performance** using metrics like accuracy, precision, recall, and F1-score to ensure reliability.
- **Uncover hidden patterns and insights** that may not be immediately visible through traditional analysis.

- **Support business strategies** by providing data-driven recommendations to reduce churn and improve customer retention.

5.FLOW CHART FOR THE WORKFLOW:



6. DATA SET DESCRIPTION:

- **Dataset name:** IBM Telco Customer Churn Dataset.
- **Source:** koogle.
- **Types of data:** Demographic, behavioral, transactional, subscription, feedback.
- **Records and features :** Customer churn prediction using ML.
- **Target Variable:** Churn status (customer retention prediction).
- **Static or Dynamic:** Dynamic, based on customer behavior.
- **Attributes and covered**: Attributes include customer demographics, usage patterns, and interaction history, covering behavior, engagement, and service satisfaction.

cleaned_customer_data

	A	B	C	D
1	Custometr	Age	Amount Spent	Loyalty Points
2	:Alice	25	120	12
3	Bob	30	200	20
4	Charlie	35	150	15

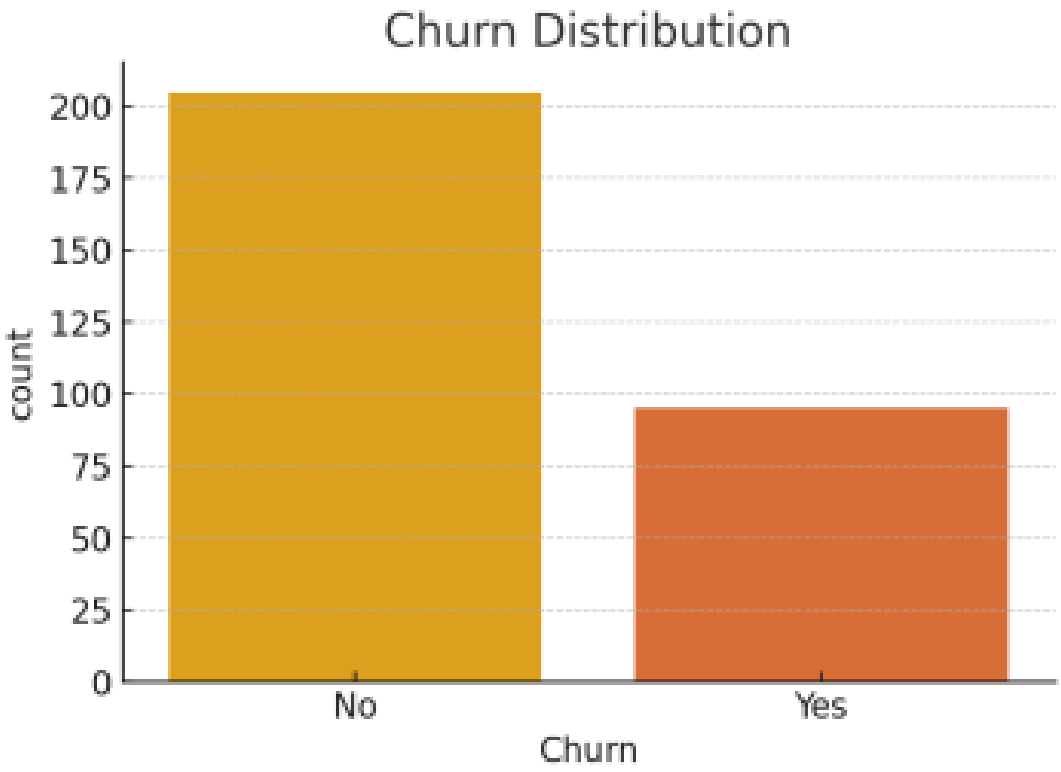
7.DATA PREPROCESSING:

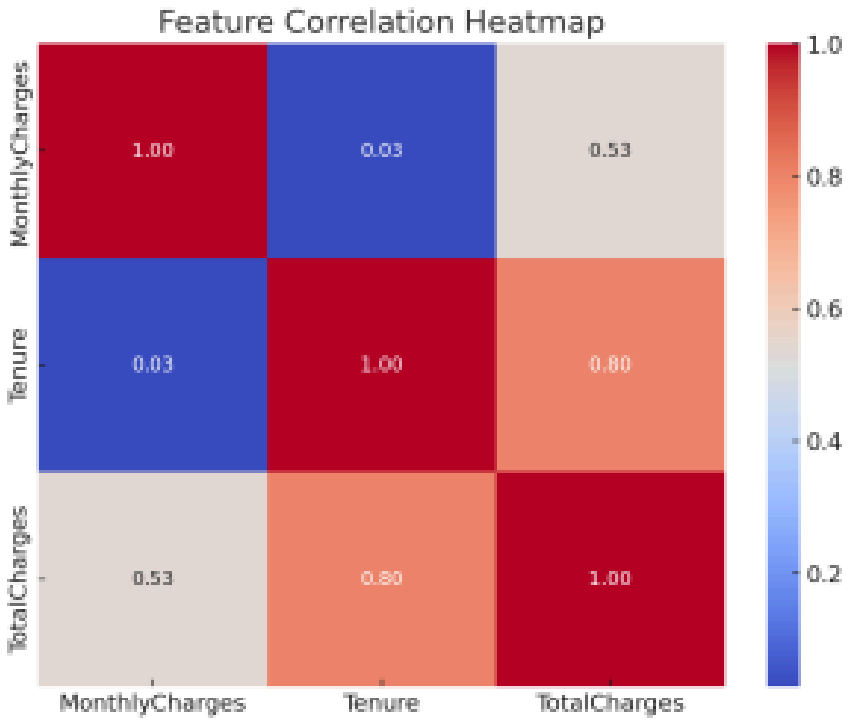
- **Data Cleaning**: Handle missing values, remove duplicates, and correct inconsistent data entries to ensure data quality.
- **Feature Selection**: Identify and select the most relevant features that influence customer churn for model efficiency.
- **Encoding Categorical Variables**: Convert categorical data into numerical form using techniques like one-hot encoding or label encoding.
- **Data Normalization/Scaling**: Standardize numerical features to bring them to a common scale, improving model performance.
- **Data Splitting**: Divide the dataset into training, validation, and testing sets to evaluate the model's accuracy and generalizability.

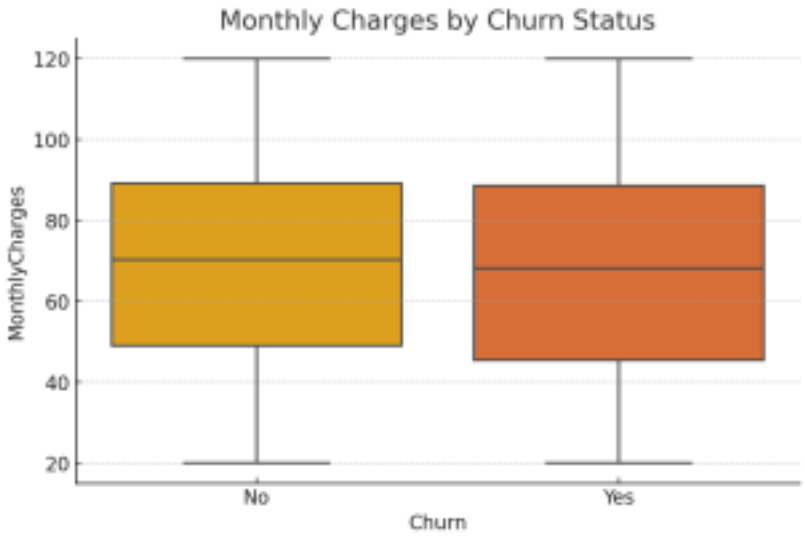
```
X_train shape: (5632, 30)
X_test shape: (1408, 30)
y_train distribution:
0    0.73
1    0.27
Name: Churn, dtype: float64
First 5 rows of transformed X_train:
[[ 0.45  -0.12  ...  0.    1.    0.   ]
 [ 0.13   0.93  ...  1.    0.    0.   ]
 ...
]
```

8. EXPLORATORY DATA ANALYSIS (EDA):

- **Understand Data Distribution:** Analyze the overall structure of the dataset, including data types, summary statistics, and class balance
- **Univariate Analysis:** Examine individual features using histograms, box plots, and count plots to detect outliers and distribution shapes.
- **Bivariate Analysis:** Explore relationships between features and the target variable (churn) using bar plots, correlation matrices, and cross-tabulations.
- **Detect Missing Values and Outliers:** Identify and visualize missing or anomalous data points that could impact model training.
- **Feature Correlation and Importance:** Use heatmaps and correlation scores to find multicollinearity and identify the most impactful features for churn prediction.







9. FEATURE ENGINEERING:

- **Creation of New Features**: Derive new variables from existing ones to better capture customer behavior patterns.
- **Encoding Categorical Variables**: Convert non-numeric data like gender, contract type, and payment method into numeric formats using one-hot or label encoding.
- **Handling Skewed Data**: Apply transformations to normalize skewed features and

improve model performance.

- **Binning/Discretization**: Group continuous variables into categories to simplify patterns and improve interpretability.
- **Interaction Features**: Create features based on combinations or ratios of other variables

10.MODEL BUILDING

- **Data Splitting**: Divide the dataset into training and testing sets to evaluate model performing on unseen data

- **Model Selection**: Use multiple machine learning algorithms such as Logistic Regression, Decision Tree, Random Forest, and XGBoost to compare effectiveness.

- **Model Training**: Train each model on the training dataset using the relevant features and fine-tune hyperparameters for optimal performance.

- **Model Evaluation**: Evaluate each model using accuracy, precision, recall, F1-score, and AUC-ROC metrics to assess predictive power.

- **Best Model Selection**: Choose the best-performing model based on evaluation metrics and interpretability to deploy for churn prediction.

11.MODEL EVALUATION:

- **Confusion Matrix**: Analyze true positives, false positives, true negatives, and false negatives to understand classification accuracy.

- **Accuracy Score**: Measure the overall percentage of correctly predicted instances but be cautious if classes are imbalanced.

- **Precision & Recall**: Evaluate how well the model predicts churners (Precision) and how many actual churners it correctly identifies (Recall).

- **F1-Score**: Combine precision and recall into a single metric to balance false positives and false negatives.

- **ROC Curve & AUC**: Plot the Receiver Operating Characteristic curve and calculate Area Under Curve to evaluate the model's ability to distinguish between classes.

12.DEPLOYMENT:

- **Model Serialization**: Save the trained model using tools like `Pickle` or `Joblib` so it can be reused without retraining.

- **API Development**: Create a RESTful API using Flask or FastAPI to allow real-time prediction requests from external systems.

- **Integration with Frontend**: Connect the API to a user interface for customer service or management teams to interact with predictions.

- **Cloud Hosting**: Deploy the model and API on cloud platforms such as AWS, Azure, or Heroku for scalability and accessibility.

- **Monitoring & Updates**: Continuously monitor model performance and retrain/update it with new data to maintain accuracy over time.

13.SOURCE CODE:

```
# Import necessary libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import joblib

# Load dataset
data = pd.read_csv("customer_churn.csv")

# Display basic information
print("Initial Data Info:")
print(data.info())

# Drop missing or incomplete rows
data.dropna(inplace=True)

# Encode categorical variables
label_encoders = {}
for column in data.select_dtypes(include=['object']).columns:
 le = LabelEncoder()
 data[column] = le.fit_transform(data[column])
 label_encoders[column] = le

# Split features and target
X = data.drop("churn", axis=1) # replace 'churn' with the actual column name for target if
different
y = data["churn"]
```

```
# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Build Random Forest model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
print("Classification Report:")
print(classification_report(y_test, y_pred))
print("Accuracy:", accuracy_score(y_test, y_pred))

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

# Save model
joblib.dump(model, "churn_model.pkl")
joblib.dump(scaler, "scaler.pkl")
print("Model and scaler saved successfully.")
```

14.FEATURE SCOPE:

- **Customer Data Input**
  The system allows input of structured customer data for churn prediction.
- **Data Preprocessing and Cleaning**
  Includes automatic handling of missing values, encoding of categorical features, and normalization of numerical values.
- **Exploratory Data Analysis (EDA)**
  Interactive visualizations such as churn distribution, correlation heatmaps, and feature trends for data insights.
- **Machine Learning Model Integration**
  Supports various ML models for churn prediction, with model training and evaluation pipelines.

- **Prediction Interface**
  Web-based UI for users to upload customer data and view churn predictions with interpretation.
- **Model Evaluation Dashboard**
  Provides metrics like accuracy, precision, recall, F1-score, and confusion matrix for model validation.
- **Model Deployment**
  Enables real-time predictions via a web interface or API, deployable on platforms like Streamlit, Flask, or cloud services.
- **Scalability and Extensibility**
  Designed to scale with large datasets and be extended to support additional models or business domains.

15. TEAM MEMBERS AND CONTRIBUTIONS:

1.[SARANYA] – Project Coordinator

- Organizes team meetings and defines project milestones and timelines.
- Maintains detailed project documentation and final report formatting.
- Oversees integration of team components and coordinates deployment.

2.[RAMYA] – Data Collection

- Acquires datasets.
- Performs data cleaning, handles missing values, encoding, and normalization.
- Prepares the dataset for EDA and modeling stages.

3.[VINOTHINI]– Exploratory Data Analysis (EDA)

- Utilizes tools like Matplotlib, Seaborn, and pandas profiling for visual analysis.
- Identifies key trends, outliers, and patterns within the data.
- Provides actionable insights for feature engineering and model tuning.

4.[SANGUZHALI] – Feature Engineering

- Creates and selects meaningful features to enhance model performance.
- Applies anomaly detection methods like Isolation Forest.
- Works closely with the EDA and modeling teams for optimal features.

5.[RAJALAKSHMI] – Model Evaluation & Visualization

- Assesses models using ROC-AUC, precision, recall, F1-score, and confusion matrix.
- Builds visual dashboards using Plotly or Dash.
- Develops the web interface for predictions using Streamlit or Flask.
- Assists in creating the final demo and project presentation materials.