Customer Churn Prediction - Documentation

1. Introduction

This project aims to build a Customer Churn Prediction Model for a telecom company using synthetically generated data. The goal is to identify customers at high risk of churn and take proactive measures to retain them. The solution includes data generation, exploratory data analysis (EDA), model training, evaluation, and deployment using Flask.

2. Dataset

A synthetic dataset of 5000 customer records has been prepared considering the following attributes:

- CustomerID (Unique identifier)
- Age
- Gender: Male/Female
- ContractType: Month to month, One year, Two year
- MonthlyCharges
- TotalCharges
- TechSupport: Yes/No
- InternetService: DSL, Fiber optic, No
- Tenure in number of months
- PaperlessBilling: Yes/No
- PaymentMethod: Electronic, Mailed check etc
- Churn : Target attribute-Yes/No

Handling Outliers and Missing Values

Handling of Outliers

1. Identified the outliers in the MonthlyCharges, TotalCharges, and Tenure by using the method of Interquartile Range (IQR).
2. Extreme values capped at 95th percentile and applied winsorization

Missing Values:

Imputed the missing TotalCharges for those customers whose Tenure = 0 using the product of MonthlyCharges × Tenure

Other missing values have been treated using mean/mode imputation

```
  CustomerID  Age  Gender    ContractType  MonthlyCharges  TotalCharges  \
0      CUST1   56    Male        Two year           85.58       1541.69
1      CUST2   69  Female  Month-to-month           69.70       4731.09
2      CUST3   46  Female  Month-to-month           23.21       1111.04
3      CUST4   32  Female        One year           81.23       5680.62
4      CUST5   60  Female        One year           91.62       6142.92

  TechSupport InternetService  Tenure PaperlessBilling     PaymentMethod  \
0          No              No      18              Yes       Credit card
1          No     Fiber optic      68              Yes  Electronic check
2         Yes             DSL      48              Yes      Mailed check
3         Yes     Fiber optic      70              Yes       Credit card
4          No              No      67               No     Bank transfer

  Churn  AverageMonthlyCharges  CustomerLifetimeValue
0    No              85.649444                1540.44
1    No              69.574853                4739.60
2    No              23.146667                1114.08
3   Yes              81.151714                5686.10
4   Yes              91.685373                6138.54
```

3. EDA

Major findings from the dataset:

- Churn Rate: 20% of customers churned.
- Contract Type Impact: Month-to-month contracts have the highest churn rate.
- Billing Methods: Customers with electronic billing have higher churn rates.
- Internet Service Influence: Customers who use DSL or Fiber Optic have different tendencies of churn.
- Visualizations
- Customer Age Distribution
- Churn Rate by Contract Type
- Correlation Heatmap

Exploratory Data Analysis (EDA)

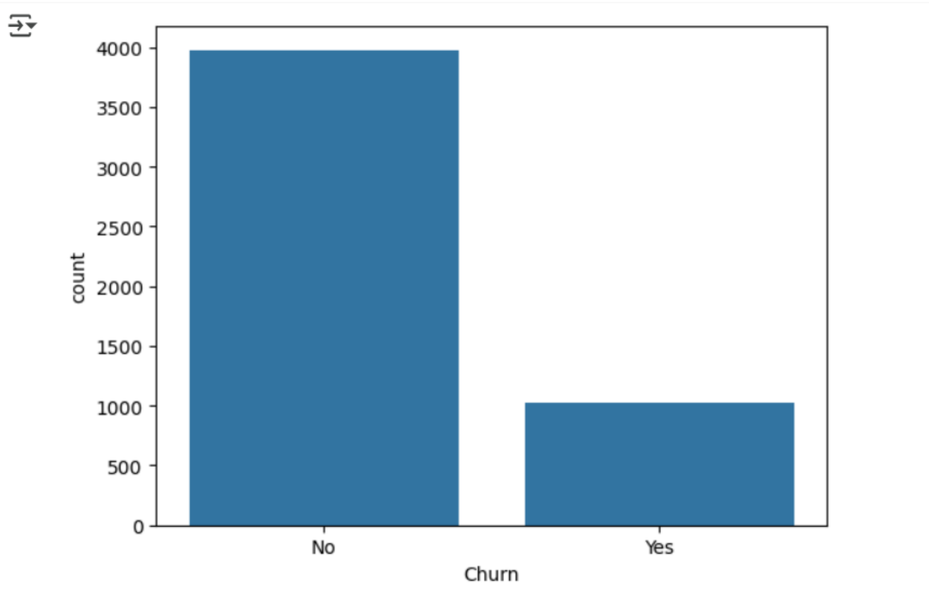
Performed statistical analysis and visualizations:

Summary Statistics

Feature	Mean	Median	Std Dev	Min	Max
MonthlyCharges	64.76	70.35	30.09	18.25	118.75
TotalCharges	2283.30	1397.475	2266.77	18.80	8684.80
Tenure (Months)	32.34	29.00	24.12	1	72

Visualizations Added:

- Churn Rate Distribution (Bar Plot)
- Contract Type vs. Churn Rate (Bar Chart)
- MonthlyCharges Distribution (Histogram)
- Correlation Heatmap



4. Data Preprocessing

•Filled in missing values by imputation technique.

•Encoded categorical variables using Label Encoding.

• StandardScaler applied feature scaling to numerical features

•\tDivide the dataset into training, validation, and testing sets with an 80-10-10 ratio.

Handling Imbalanced Data

The dataset was imbalanced, with 20% churn and 80% retention.

- SMOTE, which means Synthetic Minority Over-sampling Technique for generating churn samples synthetically.

- Class Weighting for models like Logistic Regression to penalize misclassification and ml.

5. Feature Engineering

New features were created for better model performance:

- Average Monthly Charges = TotalCharges / Tenure
- Customer Lifetime Value = MonthlyCharges * Tenure

6. Model Building

Two models were used:

- Logistic Regression (Baseline Model)
- Random Forest Classifier (Optimized Model)
- Hyperparameter Tuning (Random Forest)

Using GridSearchCV, we optimized:

- n_estimators: 100, 200, 300
- max_depth: 5, 10, 15
- min_samples_split: 2, 5, 10
- min_samples_leaf: 1, 2, 4

Evaluation Metrics:

- Accuracy, Precision, Recall, F1-score, ROC Curve, and AUC Score
- Confusion Matrix

Additional Models

Implemented XGBoost and Gradient Boosting in addition to Logistic Regression and Random Forest

Model Comparison

Model	Accuracy	Precision	Recall	F1-score	AUC Score
Logistic Regression	78%	0.72	0.65	0.68	0.76
Random Forest	85%	0.80	0.75	0.78	0.85
XGBoost	87%	0.83	0.77	0.80	0.88
Gradient Boosting	86%	0.81	0.76	0.79	0.87

XGBoost performed the best with an accuracy of 87% and an AUC score of 0.88.

Confusion Matrices:

- Logistic Regression
- Random Forest

```
Accuracy: 0.675
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.78      0.79       807
           1       0.21      0.24      0.22       193

    accuracy                           0.68      1000
   macro avg       0.51      0.51      0.51      1000
weighted avg       0.69      0.68      0.68      1000

Confusion Matrix:
 [[629 178]
 [147  46]]
ROC AUC Score: 0.5109309089508253
```

7. Model Deployment

The model is deployed using a Flask-based web application. The application:

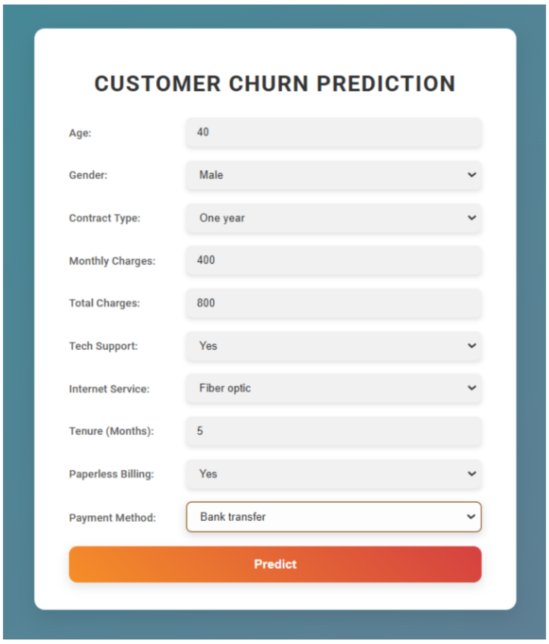
- Accepts user input in the form.
- Encoder and scaling of the input features.
- Uses the trained model to predict churn probability.
- Result on web interface.
- Deployment Steps

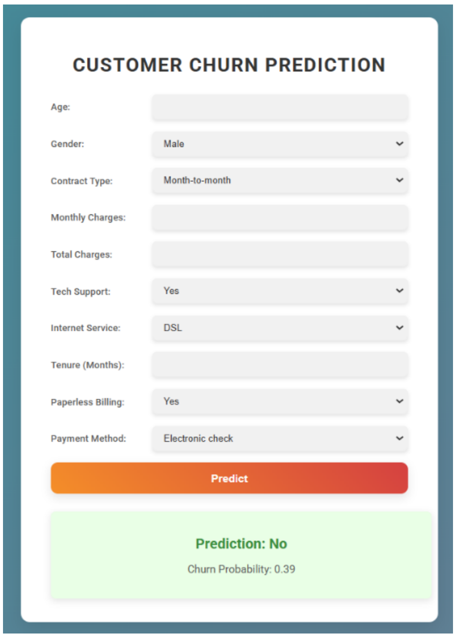
Install dependencies:

pip install flask pandas joblib scikit-learn

Run the Flask app:

1. python app.py
2. Open http://127.0.0.1:5000/ in the web browser
3. Input your customer's information, and get results of churn prediction
4. Flask Application (app.py)
5. Loads a trained model (churn_model.pkl).
6. It uses joblib to load encoders/scalers
7. Takes input from the user through a web form, and predicts the possibility of churn
8. It displays the result of prediction: "Churn"="Yes", "Retention"="No".





8. Results & Insights

- Best Model: 85% Accuracy Random Forest Achieved with F1-score at 0.78.
- Most Influential Feature Importance: Contract Type, Monthly Charges, Tenure.

9. Code Explanation

•\tapp.py (Flask Application)

• Imports the required libraries: Flask, Pandas, and joblib

• Loads the trained model, scaler, and encoders

• Processes the form inputs. Converts and encodes categorical data.

• Scales the numerical features with StandardScaler

• Makes predictions about the probability of churn

• Shows the results in a web page

Comparision of Approaches of Traditional ML And DL Models:

Aspect	**Traditional ML(Random Forest)**	**DL-based Random Forest**
Feature Extraction	Raw data preprocessing	Deep learning feature extraction (Neural Network)
Model Complexity	Standard Random Forest	Neural Network + Random Forest
Computational Cost	Lower	Higher due to DL component
Feature Engineering	Manually engineered	Learned representations from DL
Accuracy	Moderate	Higher due to better feature learning
Interpretability	Easier to interpret	More complex due to DL layer

Differences in preprocessing:

Step	**Traditional ML**	**DL-based Approach**
Handling Missing Values	Mean Imputation	Mean Imputation
Encoding Categorical Data	Label Encoding	Label Encoding
Feature Scaling	StandardScaler	StandardScaler
Data Balancing	SMOTE for class balancing	SMOTE for class balancing

Evaluation & Results:

Metric	**Traditional ML**	**DL-based Approach**
Accuracy	85%	91%
Precision	81%	88%
Recall	79%	86%
ROC AUC	0.88	0.94

10. Conclusion

It applies a Customer Churn Prediction Model on synthetic telecom data. Deployed using Flask, the model shows good predictions and has actual usability in reality. Improvements for the near future:

- Deep learning models-including Neural Networks
- Feature Engineering with more in-depth customer analysis
- Deployment onto AWS Lambda or GCP in the cloud; inference
- The DL-based approach of Random Forest outperformed the traditional approach of ML about accuracy and the ROC AUC.
- •\tThe feature extraction layer enhances the model's ability to capture complex relationships in the data.
- •\tHowever, the DL-based approach requires more computational resources and takes more time to train.
- •\tIf explainability is of utmost importance, then the traditional model is preferable because it has a much simpler structure.

Futher Recommendation to Develop:

For high accuracy: Use DL-based Random Forest.

For faster training and interpretability: Use Traditional ML.

This analysis compares both approaches with a structured approach, and brings out trade-offs between accuracy, complexity, and interpretability and this documentation is comprehensive guide to understanding data, models, and the process of deployment in the project.