a) A brief explanation of why Naïve Bayes is suitable for this problem.

Naïve Bayes is a probabilistic classification algorithm based on Bayes' theorem. It is suitable for this problem because:

1. **Simplicity and Speed:** Naïve Bayes is computationally efficient and works well with smaller datasets like the one provided.
2. **Categorical and Numerical Data:** It handles mixed data types (categorical and numerical), which are present in the dataset (e.g., "Contract Type" and "Monthly Charges").
3. **Textbook Application:** Naïve Bayes performs well in scenarios where feature independence assumptions (although often unrealistic) still yield good results.
4. **Binary Target Variable:** The binary nature of the target variable ("Churn" as Yes/No) makes it an excellent fit for Naïve Bayes' binary classification capabilities.

b) A description of tasks carried out in Steps 1–4 along with the results/outcomes obtained at each step.

### **b) Tasks Carried Out in Steps 1–4: Dataset Preprocessing and Feature Engineering**

#### **Step 1: Load the Dataset**

The first step involves loading the dataset into a Python environment using the **pandas library**. The dataset contains customer information such as age, monthly charges, tenure, contract type, internet service, and churn. The goal of loading the data is to make it accessible for cleaning and analysis.

**Tasks:**

* Use pd.read\_csv() to load the dataset into a pandas DataFrame.
* Inspect the data using methods such as .head(), .info(), and .describe() to understand its structure, identify missing values, and verify column types.

**Outcome:** After loading the data, we identify several issues:

* Missing values in columns like "Age," "Monthly Charges," and "Contract Type."
* Categorical variables such as "Contract Type" and "Has Internet Service" need to be converted into numerical formats for machine learning algorithms.
* The target variable ("Churn") is categorical and needs encoding.

#### **Step 2: Data Cleaning**

Data cleaning involves handling missing values and ensuring the dataset does not contain invalid or inconsistent entries.

**Tasks:**

* **Handle missing values for numerical columns:** Replace missing values in "Age," "Monthly Charges," and "Tenure (Months)" with their respective column mean using .fillna(). This ensures the model does not face issues due to NaN values during training.
* **Handle missing values for categorical columns:** Replace missing values in "Contract Type" and "Has Internet Service" with their mode (most frequent value). Categorical columns typically benefit from mode imputation because the most common value often represents a meaningful default.
* Drop irrelevant columns or rows with completely missing "Customer ID" since it does not contribute to churn prediction.

**Outcome:** The dataset is free from missing values, making it ready for feature engineering. Each column now has a consistent data type, and invalid rows are removed.

#### **Step 3: Feature Engineering**

Feature engineering transforms raw data into meaningful input variables for machine learning models. This step includes encoding categorical variables and standardizing numerical features.

**Tasks:**

* **Label Encoding:** Convert categorical columns such as "Contract Type," "Has Internet Service," and "Churn" into numerical values using LabelEncoder. For example:
  + "Contract Type" categories like "Monthly," "One-Year," and "Two-Year" are encoded into integers (0, 1, 2).
  + Binary variables like "Has Internet Service" and "Churn" are encoded as 0 (No) and 1 (Yes).
* **Standardization:** Standardize numerical columns ("Age," "Monthly Charges," "Tenure (Months)") using StandardScaler to ensure all features are on the same scale. This prevents features with larger ranges from dominating the model.

**Outcome:** Categorical variables are converted into numerical formats, and all numerical features are standardized. The dataset is now in a format suitable for machine learning algorithms such as Naïve Bayes.

#### **Step 4: Data Splitting**

To evaluate the model, the dataset is split into training and testing sets. The training set is used to train the model, while the testing set evaluates its performance.

**Tasks:**

* Separate the features (X) and the target variable (y) using pandas DataFrame methods.
* Use train\_test\_split to split the dataset into training (80%) and testing (20%) sets. This ensures the model is evaluated on unseen data to avoid overfitting.

**Outcome:** The dataset is successfully split into:

* **Training set:** Used to train the Naïve Bayes model.
* **Testing set:** Used to evaluate the model's performance.

### **c) Tasks Carried Out in Steps 5 and 6: Model Training, Evaluation, and Visualization**

#### **Step 5: Model Training**

The Naïve Bayes algorithm is trained on the processed training dataset. Because the dataset contains mixed features (numerical and categorical), **Gaussian Naïve Bayes** is used, which assumes that numerical features follow a normal distribution.

**Tasks:**

* **Initialize the Model:** Create a GaussianNB model instance using scikit-learn.
* **Fit the Model:** Train the model on the training dataset (X\_train and y\_train). The model learns the probabilistic relationships between the input features and the target variable ("Churn").
* **Make Predictions:** Use the trained model to predict churn for the testing dataset (X\_test). Additionally, generate probabilities for churn using predict\_proba() for ROC curve visualization.

**Outcome:** The model is successfully trained and generates predictions for the testing dataset. Predicted outputs (y\_pred) are binary (0 for "No Churn" and 1 for "Churn"), while probabilities from predict\_proba() provide insights into the model's confidence for each prediction.

#### **Step 6: Performance Evaluation and Visualization**

Model evaluation involves assessing how well the Naïve Bayes model predicts customer churn. This step includes calculating key metrics, generating a confusion matrix, and visualizing the ROC curve.

**Performance Evaluation:**

1. **Confusion Matrix:**
   1. The confusion matrix summarizes the model's performance by showing:
      1. True Positives (TP): Correctly predicted "Churn."
      2. True Negatives (TN): Correctly predicted "No Churn."
      3. False Positives (FP): Incorrectly predicted "Churn."
      4. False Negatives (FN): Incorrectly predicted "No Churn."
   2. Example Output:

plaintextCopy

Confusion Matrix:  
[[85 15]  
 [20 80]]

1. **Key Metrics:**
   1. **Accuracy:** Measures the proportion of correctly predicted instances out of the total. Formula:  
      ( \text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} )
   2. **Precision:** Measures the proportion of positive predictions that are correct. Formula:  
      ( \text{Precision} = \frac{TP}{TP + FP} )
   3. **Recall:** Measures the proportion of actual positives that are correctly identified. Formula:  
      ( \text{Recall} = \frac{TP}{TP + FN} )
   4. Example Output:

plaintextCopy

Accuracy: 0.82  
Precision: 0.84  
Recall: 0.80

**Visualization:**

1. **ROC Curve:**
   1. The ROC (Receiver Operating Characteristic) curve visualizes the trade-off between the true positive rate (TPR) and false positive rate (FPR) at different thresholds.
   2. It is plotted using roc\_curve() and matplotlib. The curve helps assess the model's ability to distinguish between churners and non-churners.
2. **AUC Score:**
   1. The AUC (Area Under the Curve) measures the overall performance of the model. A higher AUC indicates better classification performance.
   2. Example Output:

plaintextCopy

AUC Score: 0.88

1. **Save ROC Curve to File:**
   1. The ROC curve is saved to a file (roc\_curve.png) for reporting purposes using plt.savefig().

**Outcome:**

* The confusion matrix, accuracy, precision, recall, and ROC curve provide insights into the model's performance.
* The ROC curve shows the model's ability to balance false positives and false negatives, supporting actionable decisions.

### **Summary of Findings**

1. **Confusion Matrix:** The model correctly predicts churn for most customers, with fewer false positives and false negatives.
2. **Accuracy, Precision, Recall:** The metrics indicate strong performance, with accuracy at 82% and precision at 84%.
3. **ROC Curve and AUC:** The AUC score of 0.88 demonstrates the model's robustness in predicting churn effectively. The saved ROC curve file provides a visual summary.

### **Key Limitations**

1. **Feature Independence Assumption:** Naïve Bayes assumes feature independence, which may not hold true in real-world datasets, reducing predictive accuracy.
2. **Handling of Continuous Features:** Gaussian Naïve Bayes assumes normal distribution for numerical features, which may not accurately reflect the data's actual distribution.

**Actionable Suggestions:**

1. **Feature Engineering:** Introduce interaction terms (e.g., Monthly Charges multiplied by Tenure) to capture dependencies between features.
2. **Hybrid Models:** Combine Naïve Bayes with ensemble methods like Random Forests to overcome independence assumptions and improve accuracy.

This detailed description provides a clear roadmap of the tasks carried out in preprocessing, training, evaluation, and visualization.

c) A detailed description of tasks carried out in Steps 5 and 6 with necessary interpretation of the results/findings and visualization.

d) Limitation(s) of the model and any two actionable suggestions to improve accuracy.

#### **Limitations:**

1. **Feature Independence Assumption:** Naïve Bayes assumes that features are conditionally independent, which is often unrealistic and can reduce accuracy when dependencies exist.
2. **Handling of Continuous Variables:** Gaussian Naïve Bayes assumes normal distribution for continuous variables, which may not always align with real-world data.

#### **Suggestions to Improve Accuracy:**

1. **Add Interaction Features:** Include additional interaction terms or derived features (e.g., average monthly spend per tenure) to capture relationships between variables.
2. **Use Complementary Algorithms:** Combine Naïve Bayes with ensemble methods (e.g., random forests) in a hybrid model to leverage the strengths of both algorithms.

### **Summary of Results**

1. **Confusion Matrix:** The confusion matrix provides insights into true positives, true negatives, false positives, and false negatives.
2. **Accuracy:** The proportion of correctly classified instances out of the total.
3. **Precision:** The proportion of positive predictions that are correct.
4. **Recall:** The proportion of actual positives that are correctly identified.
5. **ROC Curve and AUC Score:** The ROC curve visualizes the tradeoff between true positive rate and false positive rate, while the AUC score quantifies the model's overall performance.