1. **Import Necessary Libraries:** Import libraries for data processing, visualization, modeling, and evaluation.

2. **Load Data**

3. **Exploratory Data Analysis (EDA) Before Feature Engineering:** Display basic statistics (mean, median, mode) for numerical features, Plot distributions for numerical variables (e.g., histograms, boxplots), Analyze relationships between features and the target variable using scatter plots and correlation heatmaps, Explore the distribution of categorical variables (e.g., bar charts), Check for missing values and outliers.

4. **Feature Engineering:** Create new features based on domain knowledge or data insights, for timeseries data, create lagged variables or rolling averages, transform variables to reduce skewness, if necessary (e.g., log transformation), bin or categorize continuous features if relevant, drop or adjust features based on their correlation or importance.

5. **EDA after Feature Engineering:** Reexamine the distribution of newly created features, compare distributions of original and transformed features, analyze the correlations between engineered features and the target variable, visualize trends or patterns in new features.

6. **Data Preprocessing:** Identify categorical and numerical features, handle missing values using `SimpleImputer`, encode categorical variables using `OneHotEncoder`, scale numerical variables using `StandardScaler`.

7. **Split Data:** Use `train_test_split` to divide the dataset into training and testing sets.

8. **Define Models:** Random Forest, Logistic Regression, Gradient Boosting Classifier, Decision Tree, KNearest Neighbors

9. **Set Up Pipelines:** Create a pipeline for each classifier that includes: Preprocessing steps for numerical and categorical features.

10. **Model Training and Evaluation**: Loop through each model and fit the model using the training data. Predict on the testing set. Evaluate model performance using: Accuracy, Precision, Recall, F1 Score, Confusion Matrix, ROC Curve and AUC Score.

11. **CrossValidation:** Perform crossvalidation using `cross_val_score` for each classifier. Store crossvalidation scores for comparison.

12. **Hyperparameter Tuning:** Define parameter grids for `GridSearchCV`. Perform hyperparameter tuning to find the best parameters for each model. Fit the tuned model on the training data.

13. **Voting Classifier:** Combine different classifiers into a `VotingClassifier`. Fit the ensemble model on the training set.Predict on the testing set using the `VotingClassifier`.Generate a classification report for the `VotingClassifier` including: Precision,Recall, F1 Score. Evaluate the ensemble model's performance using: Accuracy Score, Confusion Matrix

14. **Plot ROC Curve for Voting Classifier:** Generate ROC curve data (True Positive Rate and False Positive Rate) for the `VotingClassifier`.

Calculate the AUC score.

Plot the ROC curve for the `VotingClassifier` using `matplotlib` or `RocCurveDisplay`.

Display the AUC score on the plot.