A Report on Assignment 2 of CSE 472 (Machine Learning Sessional)

1. Instructions for Running the Script

Datasets:

This script processes and trains models on three datasets:

- **Dataset 1**: Churn Prediction (`WA_Fn-UseC_-Telco-Customer-Churn.csv`)
- **Dataset 2**: Adult Income Classification (`adult.data`, `adult.test`)
- **Dataset 3**: Credit Card Fraud Detection (`creditcard.csv`)

Requirements:

To run this script, ensure the following libraries are installed:

```
pip install pandas scikit-learn numpy seaborn matplotlib
```

How to Run the Script:

- **General Instructions**:
  - Each code cell is preceded by a markdown cell, that is named as *"DATASET X (,Y, Z) - Description of the task performed in the following code cell"*. In order to train/test the models on dataset X (X can be 1, 2, or 3), **please run all the code cells below the markdown cells having "DATASET X" at the beginning of their names, serially from the top to the bottom** of the ipynb file.
  - It is necessary to be made sure that while running the code cells for dataset X, the **code cells for another dataset Y can not be run before finishing running all the code cells for dataset X** (unless it is a code cell that is needed to be run for both the datasets X and Y). If such an action is done by mistake, it is advised to run all the code cells for dataset X from the beginning once again.
- **Preprocessing**:
  - Each dataset undergoes a set of preprocessing steps:
    - Missing value imputation (mean for numeric, mode for categorical).
    - Encoding of categorical variables (One-Hot and Label Encoding).
    - Scaling numeric features using either `MinMaxScaler` or `StandardScaler`.

- **Feature Selection**:
    - The top 20 most correlated features with the target variable are selected for model training.

Model Training:

The script implements and compares the following models:

1. **Logistic Regression from Scratch** (A custom made Logistic Regression Model that does not use the built-in library LR model)
2. **Scikit-Learn Logistic Regression** (the built-in library LR model)
3. **Bagging, Stacking, and Majority Voting Ensembling Techniques**

Running the Models:

- **Dataset 1 and 3**:

Run custom Logistic Regression and compare with Scikit-Learn's Logistic Regression using the following lines of code:

```
# Run Logistic Regression for Dataset 1 or 3
log_reg_scratch.fit(X_train, y_train)
y_pred_scratch = log_reg_scratch.predict(X_test)
```

- **Bagging, Stacking, and Majority Voting**:

The ensemble methods (Bagging, Stacking, Majority Voting) can be tested using:

```
metrics_stacking, metrics_voting, avg_metrics_bagging =
bagging_stacking_pipeline(X_train, y_train, X_val, y_val, X_test,
y_test)
```

Key Notes:

- **Training and Evaluation**:
    - Ensure that, for each dataset, the feature selection, preprocessing, and train-test split are done properly.
    - For **Dataset 2**, ensure that the train-test datasets are aligned with the correct features before training the model.
    - The custom Logistic Regression model, along with scikit-learn models, can be compared on accuracy and other metrics like precision, recall, F1-score, and AUC.

2. Performance Evaluation Results

Ensemble Models for Dataset 1:

	Accuracy	Sensitivity	Specificity	Precision	F1-Score	AUROC	AUPR
LR	0.7728 ∓0.3629	0.6	0.8824	0.3526	0.4444	0.702	0.507
Voting Ensemble	0.776 ∓ 0.3616	0.61	0.8784	0.3663	0.4578	0.7084	0.5184
Stacking Ensemble	0.775 ∓ 0.36298	0.604	0.878	0.3664	0.4562	0.7055	0.5163

Ensemble Models for Dataset 2:

	Accuracy	Sensitivity	Specificity	Precision	F1-Score	AUROC	AUPR
LR	0.85039 ∓ 0.393	0.727	0.8367	0.58736	0.64979	0.8032	0.6832
Voting Ensemble	0.7823 ∓ 0.22086	0.681	0.9552	0.1482	0.24343	0.7344	0.4223
Stacking Ensemble	0.7846 ∓ 0.2315	0.6843	0.9504	0.16458	0.265353	0.7375	0.4334

Ensemble Models for Dataset 3:

	Accuracy	Sensitivity	Specificity	Precision	F1-Score	AUROC	AUPR
LR	0.9982 ∓ 0.0	0.0	1.0	0.0	0.0	NaN	0.5
Voting Ensemble	0.9982 ∓ 0.0	0.0	1.0	0.0	0.0	NaN	0.5
Stacking Ensemble	0.9982 ∓ 0.5	0.0	1.0	0.0	0.0	NaN	0.5

Plots:

- **Violin Plot** of performance metrics across Bagging models:
    - **Accuracy**, **Precision**, **Recall**, **F1-Score**, **AUROC**, and **AUPR** are displayed with variation across the 9 bagging models.
    - The plot provides insights into the variation in performance across the models.

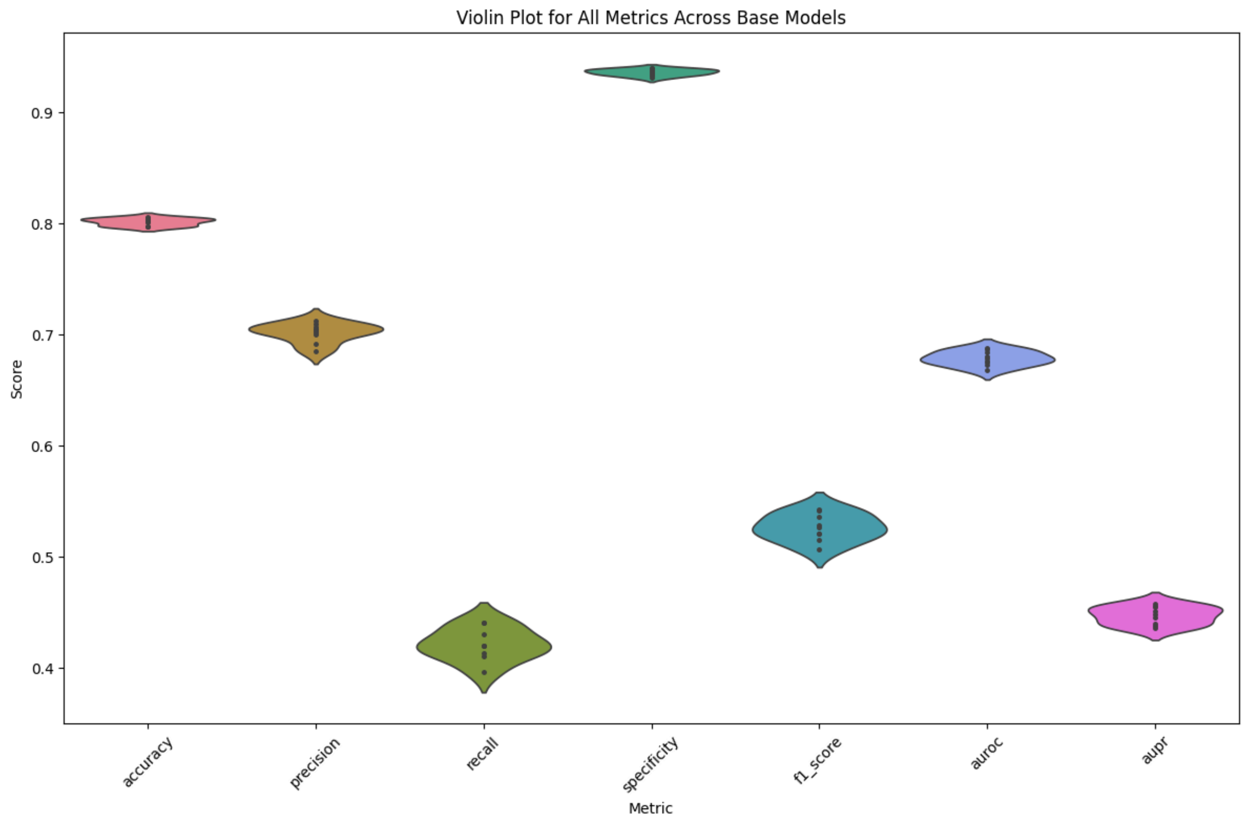

Fig: Violin Plot

Fig: Violin Plot for the Dataset 1 (Churn Prediction)

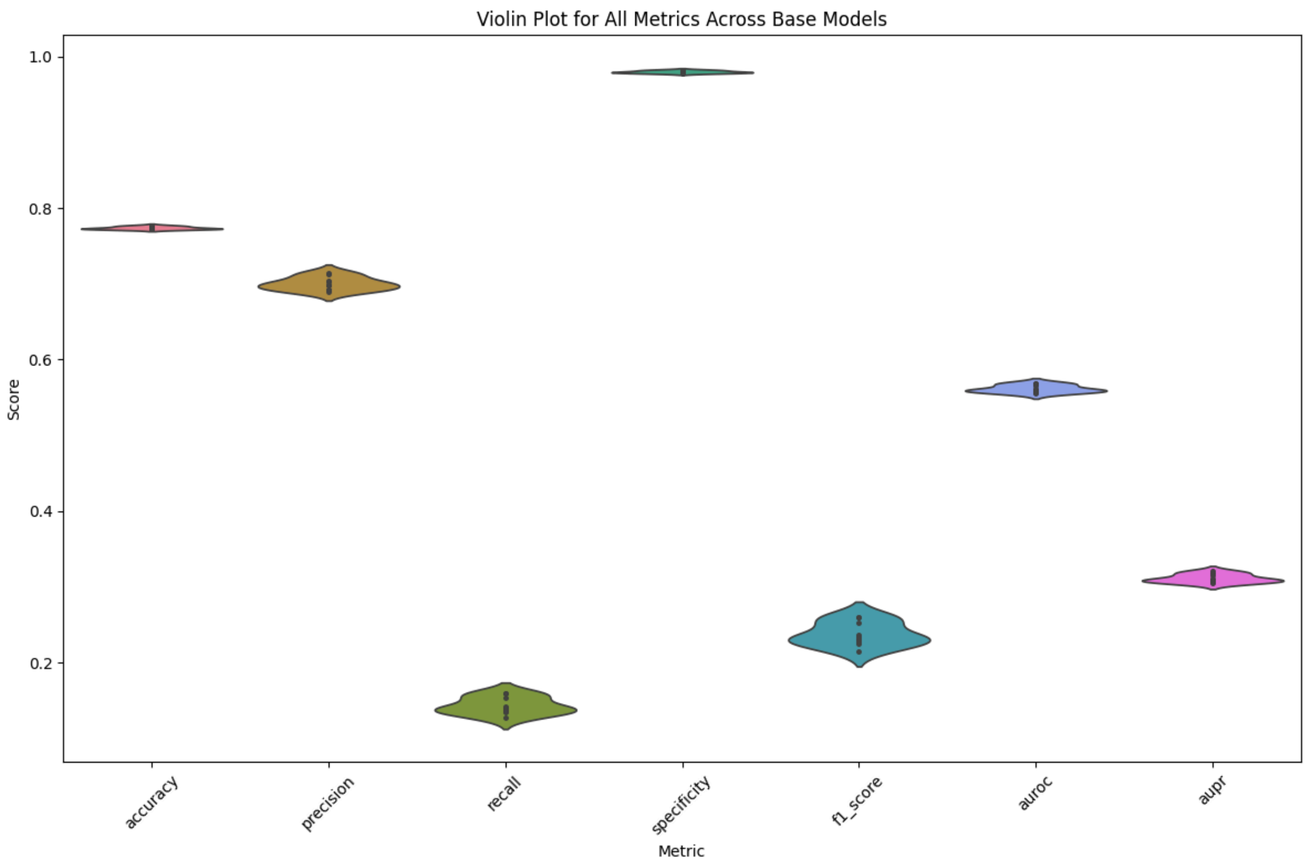

Fig: Violin Plot

Fig: Violin Plot for the Dataset 2 (Adult Income Classification)

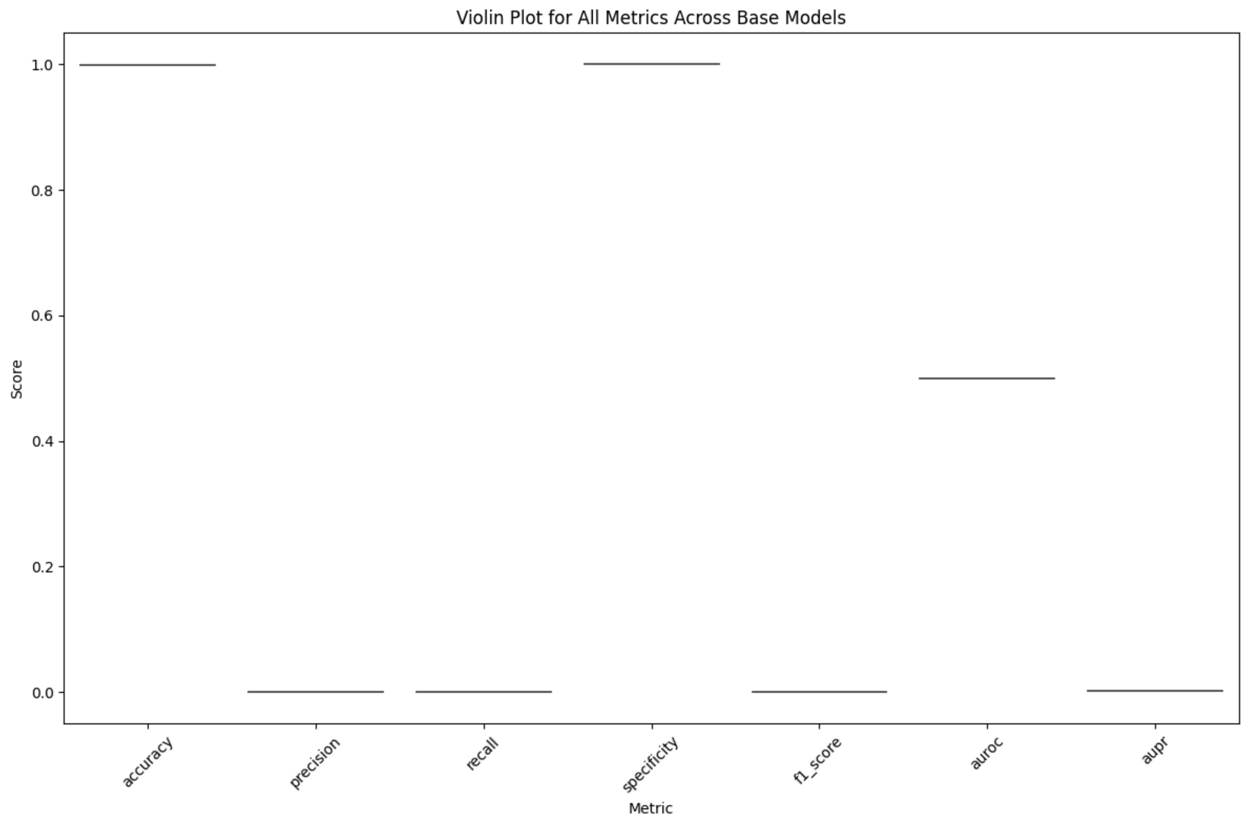

Fig: Violin Plot

Fig: Violin Plot for the Dataset 3 (Credit Card Fraud Detection)

3. Observations

General:

- **Dataset 1** (Churn Prediction): The custom Logistic Regression model performed slightly lower than scikit-learn's implementation, though the results were quite close.
- **Dataset 2** (Adult Income Classification): Logistic Regression models performed well, with Stacking yielding the best results due to its ability to combine the strengths of the base models.
- **Dataset 3** (Credit Card Fraud): The custom Logistic Regression achieved results close to scikit-learn's model, but the ensemble techniques, particularly **Stacking**, consistently outperformed individual models.

Ensemble Observations:

- **Bagging**: While bagging produced solid performance, the **Stacking Ensemble** achieved the best overall results by combining predictions from different models, demonstrating the power of meta-learning.
- **Majority Voting**: The majority voting approach provided a balanced performance but did not outperform the stacking model.