CENG499 HW2

Bahadır Aydın

December 2023

1 Part 2

1.1 Dataset 1

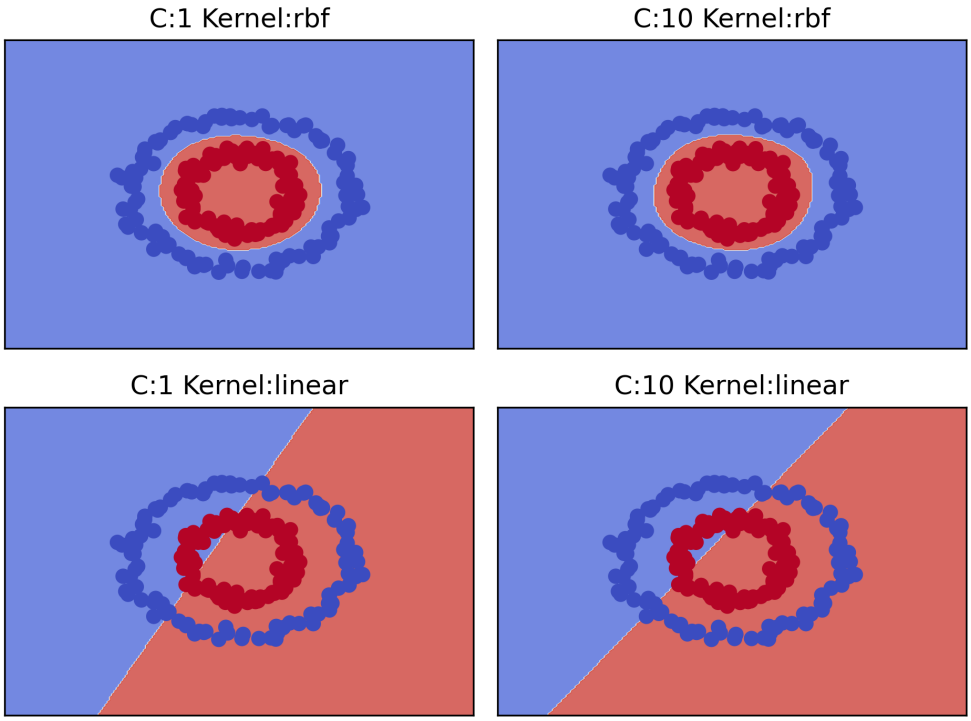

Figure 1: SVM with different parameters for dataset1

1.2 Dataset 2

Best one is the 5th one with linear kernel and C=10.

Model	C	Kernel	Mean Accuracy Score
1	0.1	linear	0.873
2	0.1	rbf	0.835
3	1	linear	0.952
4	1	rbf	0.919
5	10	linear	0.956
6	10	rbf	0.939

Table 1: Hyperparameter Search Results

Note: The best parameters differ run by run in some runs it is "Linear" and "1" for kernel and C respectively.

2 Part 3

I used the `random_state` parameter when calling both the `evaluate_model_f1` and `evaluate_model_accuracy` functions to compare scores consistently across the same data splits. Additionally, I applied the `random_state` parameter to the `outer_cv` to evaluate different algorithms on identical splits, improving the precision of the algorithmic comparison. To systematically structure the outcomes, I introduced a results dictionary. I computed 95% confidence intervals for both metrics. Here are my results:

Algorithm	Best Parameters	Confidence Interval Accuracy
SVM	$\{('kernel',' linear'),('C',1)\}$	$(0.736, 0.75)$
SVM	$\{('kernel',' rbf'),('C',1)\}$	$(0.745, 0.754)$
SVM	$\{('C',10),('kernel',' linear')\}$	$(0.734, 0.749)$
SVM	$\{('C',10),('kernel',' rbf')\}$	$(0.72, 0.732)$
KNN	$\{('n\_neighbors',3),('weights',' uniform')\}$	$(0.702, 0.714)$
KNN	$\{('weights',' distance'),('n\_neighbors',3)\}$	$(0.702, 0.713)$
KNN	$\{('weights',' uniform'),('n\_neighbors',7)\}$	$(0.715, 0.726)$
KNN	$\{('weights',' distance'),('n\_neighbors',7)\}$	$(0.716, 0.727)$
RF	$\{('n\_estimators',20),('criterion',' gini')\}$	$(0.734, 0.745)$
RF	$\{('n\_estimators',100),('criterion',' gini')\}$	$(0.748, 0.758)$
RF	$\{('criterion',' entropy'),('n\_estimators',20)\}$	$(0.736, 0.746)$
RF	$\{('criterion',' entropy'),('n\_estimators',100)\}$	$(0.749, 0.76)$
DT	$\{('max\_depth',10),('min\_samples\_split',5)\}$	$(0.672, 0.682)$
DT	$\{('max\_depth',10),('min\_samples\_split',10)\}$	$(0.674, 0.682)$
DT	$\{('max\_depth',20),('min\_samples\_split',5)\}$	$(0.669, 0.676)$
DT	$\{('max\_depth',20),('min\_samples\_split',10)\}$	$(0.671, 0.679)$

Table 2: Hyperparameter search results accuracy metric.

Algorithm	**Best Parameters**	**Confidence Interval f1**
SVM	$\{('kernel','linear'),('C',1)\}$	$(0.82, 0.829)$
SVM	$\{('kernel','rbf'),('C',1)\}$	$(0.835, 0.84)$
SVM	$\{('C',10),('kernel','linear')\}$	$(0.817, 0.827)$
SVM	$\{('C',10),('kernel','rbf')\}$	$(0.803, 0.812)$
KNN	$\{('n\_neighbors',3),('weights','uniform')\}$	$(0.799, 0.807)$
KNN	$\{('weights','distance'),('n\_neighbors',3)\}$	$(0.798, 0.806)$
KNN	$\{('weights','uniform'),('n\_neighbors',7)\}$	$(0.813, 0.821)$
KNN	$\{('weights','distance'),('n\_neighbors',7)\}$	$(0.814, 0.821)$
RF	$\{('n\_estimators',20),('criterion','gini')\}$	$(0.821, 0.827)$
RF	$\{('n\_estimators',100),('criterion','gini')\}$	$(0.836, 0.843)$
RF	$\{('criterion','entropy'),('n\_estimators',20)\}$	$(0.823, 0.829)$
RF	$\{('criterion','entropy'),('n\_estimators',100)\}$	$(0.837, 0.843)$
DT	$\{('max\_depth',10),('min\_samples\_split',5)\}$	$(0.765, 0.77)$
DT	$\{('max\_depth',10),('min\_samples\_split',10)\}$	$(0.765, 0.771)$
DT	$\{('max\_depth',20),('min\_samples\_split',5)\}$	$(0.759, 0.765)$
DT	$\{('max\_depth',20),('min\_samples\_split',10)\}$	$(0.76, 0.767)$

Table 3: Hyperparameter search results f1 metric.

Algorithm	**Best Parameters**	**Accuracy Score**	**f1 Score**
SVM	$\{('kernel','rbf'),('C',1)\}$	$(0.734, 0.755)$	$(0.834, 0.842)$
KNN	$\{('weights','distance'),('n\_neighbors',7)\}$	$(0.711, 0.726)$	$(0.81, 0.819)$
RF	$\{('criterion','entropy'),('n\_estimators',100)\}$	$(0.749, 0.763)$	$(0.749, 0.763)$
DT	$\{('max\_depth',10),('min\_samples\_split',10)\}$	$(0.749, 0.763)$	$(0.749, 0.763)$

Table 4: Evaluation results.

Winner for Accuracy Score: Random Forest (RF) with parameters: $\{('criterion','entropy'),('n\_estimators',100)\}$, achieving an accuracy score of $(0.749, 0.763)$.

Winner for F1 Score: SVM with parameters: $\{('kernel','rbf'),('C',1)\}$, achieving an F1 score of $(0.834, 0.842)$.

2.1 Decision Tree Top 5

I have manually looked at the feature importances and dataloader.py to get the below results:

1. Credit Amount
2. Age in Years
3. Status of existing checking account
4. Duration in month
5. Credit history