Model Optimization and Tuning Phase Template

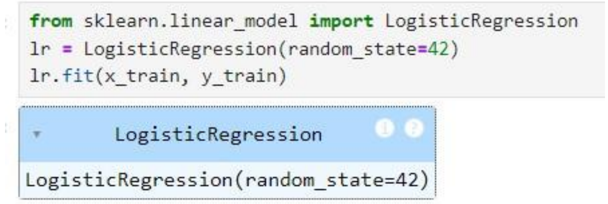
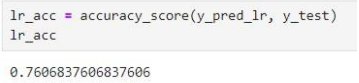
Date	13 july 2024
Team ID	739805
Project Title	Prediction and Analysis of Liver Patient Data Using Machine Learning
Maximum Marks	10 Marks

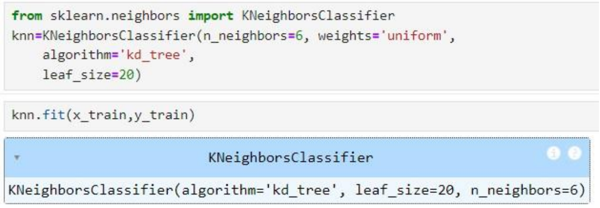
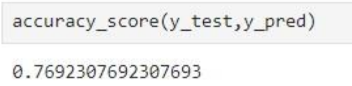
Model Optimization and Tuning Phase

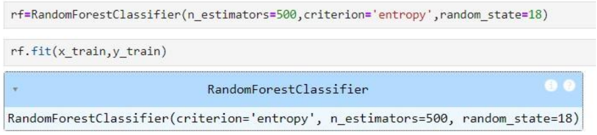
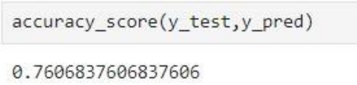
The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

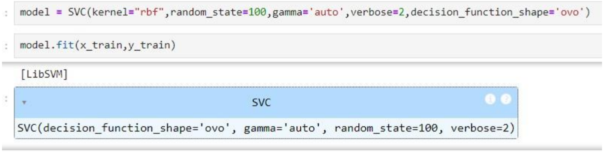
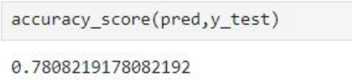
Hyperparameter Tuning Documentation (6 Marks):

Model	Tuned Hyperparameters	Optimal Values

Logistic Regression	
	from sklearn.linear_model import LogisticRegression
	lr = LogisticRegression(random_state=42)
	lr.fit(x_train, y_train)
	LogisticRegression
	LogisticRegression(random_state=42)
		
		lr_acc = accuracy_score(y_pred_lr, y_test)
		lr_acc
		0.7606837606837606

K neighbors Classifier	
	`from sklearn.neighbors import KNeighborsClassifier`
	`knn=KNeighborsClassifier(n_neighbors=6, weights='uniform',`
	`    algorithm='kd_tree',`
	`    leaf_size=20)`
	`knn.fit(x_train,y_train)`
	`KNeighborsClassifier`
	`KNeighborsClassifier(algorithm='kd_tree', leaf_size=20, n_neighbors=6)`
		
		`accuracy_score(y_test,y_pred)`
		`0.7692307692307693`

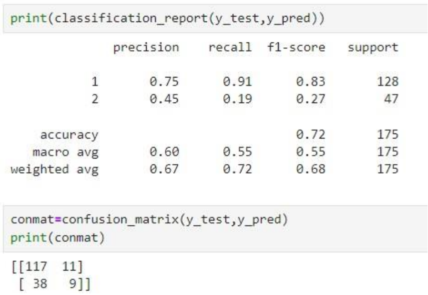
RandomForest Classifier	
	`rf=RandomForestClassifier(n_estimators=500,criterion='entropy',random_state=18)`
	`rf.fit(x_train,y_train)`
	`RandomForestClassifier`
	`RandomForestClassifier(criterion='entropy', n_estimators=500, random_state=18)`
		
		`accuracy_score(y_test,y_pred)`
		`0.7606837606837606`

SVC	
	`model = SVC(kernel="rbf",random_state=100,gamma='auto',verbose=2,decision_function_shape='ovo')`
	`model.fit(x_train,y_train)`
	`[LibSVM]`
	`SVC`
	`SVC(decision_function_shape='ovo', gamma='auto', random_state=100, verbose=2)`
		
		`accuracy_score(pred,y_test)`
		`0.7808219178082192`

Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric	Optimized Metric

Logistic Regression

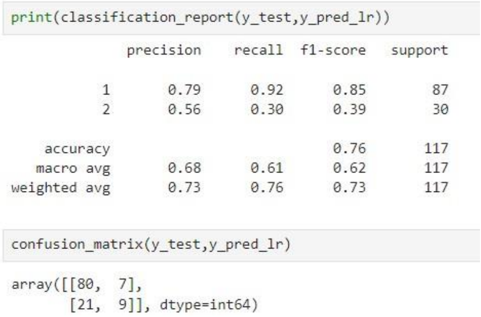


```
print(classification_report(y_test,y_pred))
              precision    recall  f1-score   support

           1       0.75      0.91      0.83       128
           2       0.45      0.19      0.27        47

    accuracy                           0.72       175
   macro avg       0.60      0.55      0.55       175
weighted avg       0.67      0.72      0.68       175

conmat=confusion_matrix(y_test,y_pred)
print(conmat)
[[117  11]
 [ 38   9]]
```



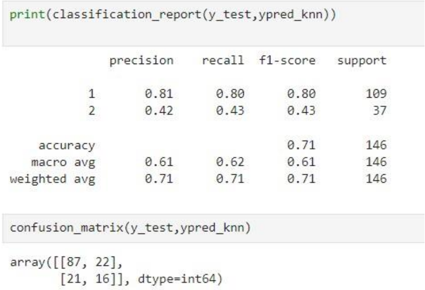
```
print(classification_report(y_test,y_pred_lr))
              precision    recall  f1-score   support

           1       0.79      0.92      0.85        87
           2       0.56      0.30      0.39        30

    accuracy                           0.76       117
   macro avg       0.68      0.61      0.62       117
weighted avg       0.73      0.76      0.73       117

confusion_matrix(y_test,y_pred_lr)
array([[80,  7],
       [21,  9]], dtype=int64)
```

K neighbors Classifier



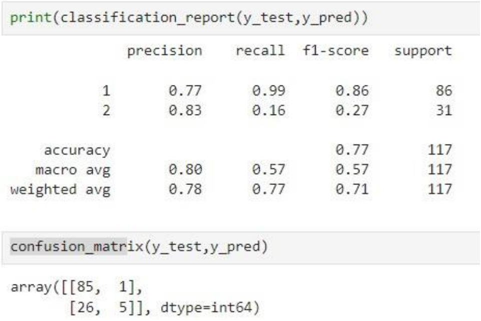
```
print(classification_report(y_test,ypred_knn))

              precision    recall  f1-score   support

           1       0.81      0.80      0.80       109
           2       0.42      0.43      0.43        37

    accuracy                           0.71       146
   macro avg       0.61      0.62      0.61       146
weighted avg       0.71      0.71      0.71       146

confusion_matrix(y_test,ypred_knn)
array([[87, 22],
       [21, 16]], dtype=int64)
```



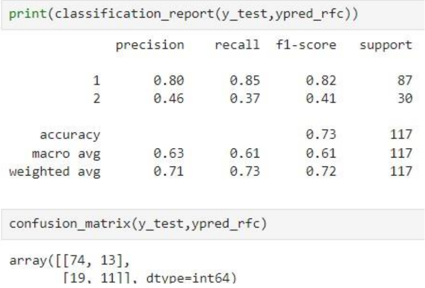
```
print(classification_report(y_test,y_pred))
              precision    recall  f1-score   support

           1       0.77      0.99      0.86        86
           2       0.83      0.16      0.27        31

    accuracy                           0.77       117
   macro avg       0.80      0.57      0.57       117
weighted avg       0.78      0.77      0.71       117

confusion_matrix(y_test,y_pred)
array([[85,  1],
       [26,  5]], dtype=int64)
```

RandomForest Classifier

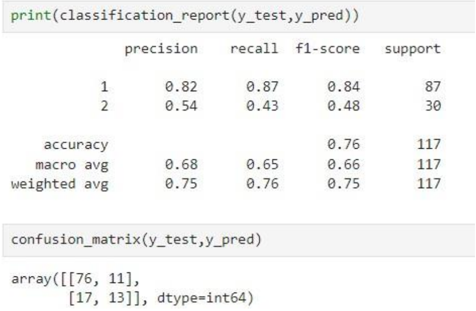


```
print(classification_report(y_test,ypred_rfc))
              precision    recall  f1-score   support

           1       0.80      0.85      0.82        87
           2       0.46      0.37      0.41        30

    accuracy                           0.73       117
   macro avg       0.63      0.61      0.61       117
weighted avg       0.71      0.73      0.72       117

confusion_matrix(y_test,ypred_rfc)
array([[74, 13],
       [19, 11]], dtype=int64)
```

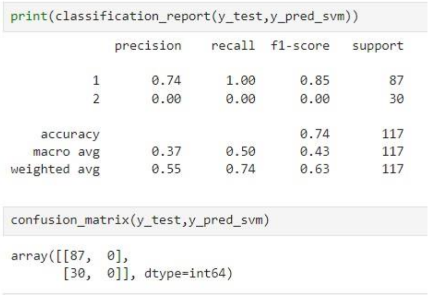
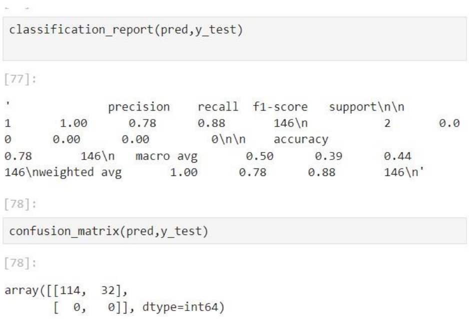


```
print(classification_report(y_test,y_pred))
              precision    recall  f1-score   support

           1       0.82      0.87      0.84        87
           2       0.54      0.43      0.48        30

    accuracy                           0.76       117
   macro avg       0.68      0.65      0.66       117
weighted avg       0.75      0.76      0.75       117

confusion_matrix(y_test,y_pred)
array([[76, 11],
       [17, 13]], dtype=int64)
```

SVC	 print(classification_report(y_test,y_pred_svm)) precision recall f1-score support 1 0.74 1.00 0.85 87 2 0.00 0.00 0.00 30 accuracy 0.74 117 macro avg 0.37 0.50 0.43 117 weighted avg 0.55 0.74 0.63 117 confusion_matrix(y_test,y_pred_svm) array([[87, 0], [30, 0]], dtype=int64)	 classification_report(pred,y_test) [77]: ' precision recall f1-score support\n\n 1 1.00 0.78 0.88 146\n 2 0.0 0 0.00 0.00 0\n\n accuracy 0.78 146\n macro avg 0.50 0.39 0.44 146\nweighted avg 1.00 0.78 0.88 146\n' [78]: confusion_matrix(pred,y_test) [78]: array([[114, 32], [0, 0]], dtype=int64)

Final Model Selection Justification (2 Marks):

Final Model	**Reasoning**
SVC	SVC is selected as for its Effective in High-Dimensional Spaces, Robust to Overfitting handle both linear and non-linear classification problems by employing kernel functions, making it a versatile and powerful tool for a wide range of applications