

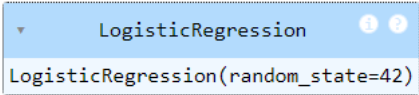
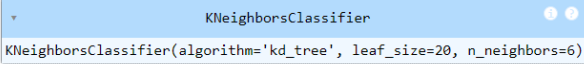
Model Optimization and Tuning Phase Template

Date	10 July 2024
Team ID	SWTID1720013031
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	<pre>from sklearn.linear_model import LogisticRegression lr = LogisticRegression(random_state=42) lr.fit(x_train, y_train)</pre> 	<pre>lr_acc = accuracy_score(y_pred_lr, y_test) lr_acc</pre> <p>0.7606837606837606</p>
K neighbors Classifier	<pre>from sklearn.neighbors import KNeighborsClassifier knn=KNeighborsClassifier(n_neighbors=6, weights='uniform', algorithm='kd_tree', leaf_size=20) knn.fit(x_train,y_train)</pre> 	<pre>accuracy_score(y_test,y_pred)</pre> <p>0.7692307692307693</p>

RandomForest Classifier	<pre>rf=RandomForestClassifier(n_estimators=500,criterion='entropy',random_state=18) rf.fit(x_train,y_train)</pre> 	<pre>accuracy_score(y_test,y_pred)</pre> <p>0.7606837606837606</p>
SVC	<pre>model = SVC(kernel="rbf",random_state=100,gamma='auto',verbose=2,decision_function_shape='ovo') model.fit(x_train,y_train)</pre> 	<pre>accuracy_score(pred,y_test)</pre> <p>0.7808219178082192</p>

Performance Metrics Comparison Report (2 Marks):

Model	Baseline Metric	Optimized Metric																																																												
Logistic Regression	<pre>print(classification_report(y_test,y_pred))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>1</td><td>0.75</td><td>0.91</td><td>0.83</td><td>128</td></tr><tr><td>2</td><td>0.45</td><td>0.19</td><td>0.27</td><td>47</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.72</td><td>175</td></tr><tr><td>macro avg</td><td>0.60</td><td>0.55</td><td>0.55</td><td>175</td></tr><tr><td>weighted avg</td><td>0.67</td><td>0.72</td><td>0.68</td><td>175</td></tr></tbody></table> <pre>conmat=confusion_matrix(y_test,y_pred) print(conmat)</pre> <pre>[[117 11] [38 9]]</pre>		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	<pre>print(classification_report(y_test,y_pred_lr))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>1</td><td>0.79</td><td>0.92</td><td>0.85</td><td>87</td></tr><tr><td>2</td><td>0.56</td><td>0.30</td><td>0.39</td><td>30</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.76</td><td>117</td></tr><tr><td>macro avg</td><td>0.68</td><td>0.61</td><td>0.62</td><td>117</td></tr><tr><td>weighted avg</td><td>0.73</td><td>0.76</td><td>0.73</td><td>117</td></tr></tbody></table> <pre>confusion_matrix(y_test,y_pred_lr)</pre> <pre>array([[80, 7], [21, 9]], dtype=int64)</pre>		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
		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																																																										
	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																																																										
K neighbors Classifier	<pre>print(classification_report(y_test,ypred_knn))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>1</td><td>0.81</td><td>0.80</td><td>0.80</td><td>109</td></tr><tr><td>2</td><td>0.42</td><td>0.43</td><td>0.43</td><td>37</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.71</td><td>146</td></tr><tr><td>macro avg</td><td>0.61</td><td>0.62</td><td>0.61</td><td>146</td></tr><tr><td>weighted avg</td><td>0.71</td><td>0.71</td><td>0.71</td><td>146</td></tr></tbody></table> <pre>confusion_matrix(y_test,ypred_knn)</pre> <pre>array([[87, 22], [21, 16]], dtype=int64)</pre>		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	<pre>print(classification_report(y_test,y_pred))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>1</td><td>0.77</td><td>0.99</td><td>0.86</td><td>86</td></tr><tr><td>2</td><td>0.83</td><td>0.16</td><td>0.27</td><td>31</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.77</td><td>117</td></tr><tr><td>macro avg</td><td>0.80</td><td>0.57</td><td>0.57</td><td>117</td></tr><tr><td>weighted avg</td><td>0.78</td><td>0.77</td><td>0.71</td><td>117</td></tr></tbody></table> <pre>confusion_matrix(y_test,y_pred)</pre> <pre>array([[85, 1], [26, 5]], dtype=int64)</pre>		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
	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																																																										
	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																																																										

RandomForest Classifier

```
print(classification_report(y_test,y_pred_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,y_pred_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 1   1.00      0.78      0.88      146\n 0   0.00      0.00      0.00         0\n 0.78      146\n macro avg   0.50      0.39      0.44      146\n weighted 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