

Model Development Phase Template

Date	12 JULY 2024
ID	740022
Project Title	Lymphography Classification Using ML
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code:

```
[ ] 1 clf.fit(x_train, y_train)
    2
    3 # Make predictions on the testing data
    4 y_pred = clf.predict(x_test)
    5
    6 # Evaluate the classifier
    7 report = classification_report(y_test, y_pred)
    8 print("Classification Report:\n", report)
```

```
↔ Classification Report:
              precision    recall  f1-score   support

         2.0         0.74         1.00         0.85         14
         3.0         1.00         0.64         0.78         14
         4.0         1.00         1.00         1.00          2

 accuracy                   0.83         30
 macro avg              0.91         0.88         0.88         30
 weighted avg           0.88         0.83         0.83         30
```

```
[ ] 1 from sklearn.linear_model import LogisticRegression
    2 from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
1 model_lr = LogisticRegression()
2 model_lr.fit(x_train, y_train)
3 lr_pred_test = model_lr.predict(x_test)
4 lr_pred_train = model_lr.predict(x_train)
5 test_acc_lr = accuracy_score(y_test, lr_pred_test)
6 train_acc_lr = accuracy_score(y_train, lr_pred_train)
7 print('Logistic Regression test accuracy: ', test_acc_lr)
8 print(classification_report(y_test, lr_pred_test))
```

```
Logistic Regression test accuracy: 0.8
precision    recall  f1-score   support

2.0         0.71     1.00     0.83        12
3.0         1.00     0.71     0.83        17
4.0         0.00     0.00     0.00         1

accuracy          0.80        30
macro avg         0.57     0.57     0.55        30
weighted avg      0.85     0.80     0.80        30
```

KNN

```
1 knn_score1 = metrics.accuracy_score(y_train, knn_pred1) * 100
2 print(knn_score1)
3 knn_score = metrics.accuracy_score(y_test, knn_pred) * 100
4
5 ### Print classification report for regular
6 print(" ----- Regular Training Set Used ----- ")
7 print("Classification report for {}:\n{}".format(knn, metrics.classification_report(y_test, knn_pred)))
8 print("Accuracy score:", knn_score)
```

```
81.35593220338984
----- Regular Training Set Used -----
Classification report for KNeighborsClassifier():
precision    recall  f1-score   support

2.0         0.87     0.93     0.90        14
3.0         0.80     0.86     0.83        14
4.0         0.00     0.00     0.00         2

accuracy          0.83        30
macro avg         0.56     0.60     0.57        30
weighted avg      0.78     0.83     0.80        30

Accuracy score: 83.33333333333334
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix

Decision tree classifier	<pre> classification report: precision recall f1-score support 0.0: 0.75 1.00 0.80 10 1.0: 1.00 0.64 0.79 14 2.0: 1.00 1.00 1.00 1 accuracy: 0.83 macro avg: 0.88 0.88 0.89 25 weighted avg: 0.86 0.83 0.83 25 </pre>	0.8	<pre> [] 1 confusion_matrix(y_test,y_pred) => array([[14, 0, 0], [5, 9, 0], [0, 0, 2]]) </pre>
Logistic Regression	<pre> Logistic Regression test accuracy: 0.8 precision recall f1-score support 0.0: 0.75 1.00 0.81 12 1.0: 1.00 0.71 0.83 17 2.0: 0.00 0.00 0.00 1 accuracy: 0.73 0.72 0.80 30 macro avg: 0.57 0.57 0.55 30 weighted avg: 0.57 0.60 0.60 30 </pre>	0.8	<pre> [] 1 confusion_matrix(y_test,lr_pred_test) => array([[12, 0, 0], [4, 12, 1], [1, 0, 8]]) </pre>
K-Nearest Neighbors	<pre> KNN test accuracy is: 0.8333333333333333 precision recall f1-score support 0.0: 0.80 0.94 0.89 12 1.0: 0.80 0.71 0.77 17 2.0: 0.80 0.00 0.00 1 accuracy: 0.77 macro avg: 0.53 0.54 0.53 30 weighted avg: 0.76 0.73 0.75 30 </pre>	0.83	<pre> [] 1 confusion_matrix(y_test,knn_pred_test) => array([[11, 1, 0], [5, 12, 0], [0, 1, 0]]) </pre>
Gradient Boosting	<pre> GB accuracy is: 0.8333333333333333 precision recall f1-score support 0.0: 0.75 0.92 0.80 12 1.0: 0.95 0.82 0.89 17 2.0: 0.90 0.00 0.00 1 accuracy: 0.83 0.84 0.83 30 macro avg: 0.54 0.54 0.54 30 weighted avg: 0.83 0.84 0.83 30 </pre>	0.83	<pre> [] 1 confusion_matrix(y_test,gb_pred_test) => array([[11, 1, 0], [5, 12, 0], [0, 1, 0]]) </pre>