Project Development Phase
Model Performance Test

Date	18 November 2022
Team ID	PNT2022TMID20879
Project Name	Project - Statistical Machine learning Approaches to Liver Disease Prediction
Maximum Marks	10 Marks

Model Performance Testing:

The project team shall fill in the following information in the model performance testing template.

- **Metrics:**
  - **Regression Model:**

- ***Random Forest:***

Values:

```
r2_score: -0.38965517241379266
MSE: 0.26495726495726496
MAE: 0.26495726495726496
RMSE: 0.5147399974329419
```

Screenshot:

Performance Metrics

```
In [15]: from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
```

Random Forest

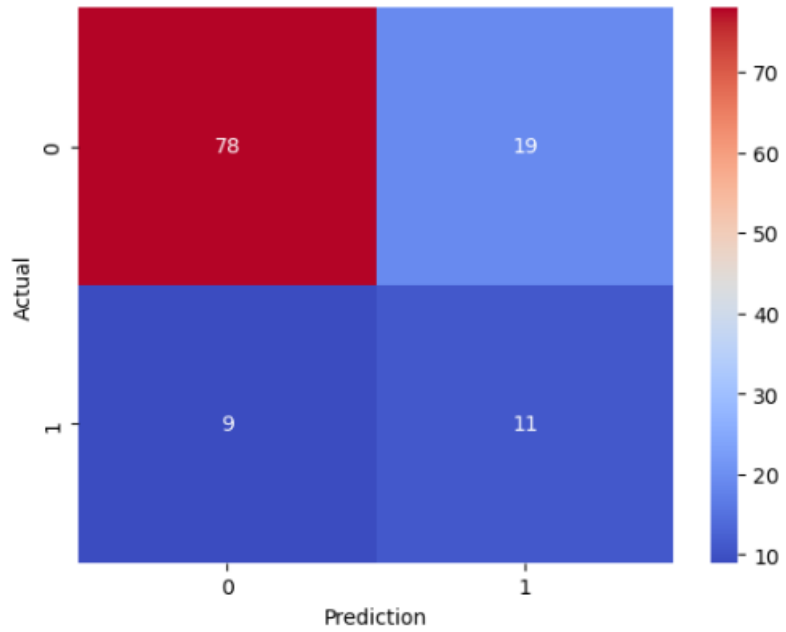
```
In [18]: RFmodel_pred = RFmodel.predict(X_test)
```

```
In [19]: print("r2_score :",r2_score(Y_test,RFmodel_pred))
         print("MSE :",mean_squared_error(Y_test,RFmodel_pred))
         print("MAE :",mean_absolute_error(Y_test,RFmodel_pred))
         print("RMSE :",np.sqrt(mean_squared_error(Y_test,RFmodel_pred)))

         r2_score : -0.38965517241379266
         MSE : 0.26495726495726496
         MAE : 0.26495726495726496
         RMSE : 0.5147399974329419
```

Confusion Matrix:

```
In [39]: cf=confusion_matrix(RFmodel_pred,Y_test)
         sb.heatmap(cf,cmap='coolwarm',annot=True)
         plt.xlabel("Prediction")
         plt.ylabel("Actual")
         plt.show()
```



Accuracy Score:

```
In [35]: RFmodel_pred = RFmodel.predict(X_test)

In [36]: RFmodel_accuracy = accuracy_score(RFmodel_pred,Y_test)

In [37]: print("RF Accuracy:",RFmodel_accuracy)
         RF Accuracy: 0.7606837606837606
```

❖ Classification Model:

- ***Support Vector Machine Model:***

Values:

r2_score: -0.34482758620689613
MSE: 0.2564102564102564
MAE: 0.2564102564102564
RMSE: 0.5063696835418333

Screenshot:

Performance Metrics

```
In [15]: from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
```

SVM Model

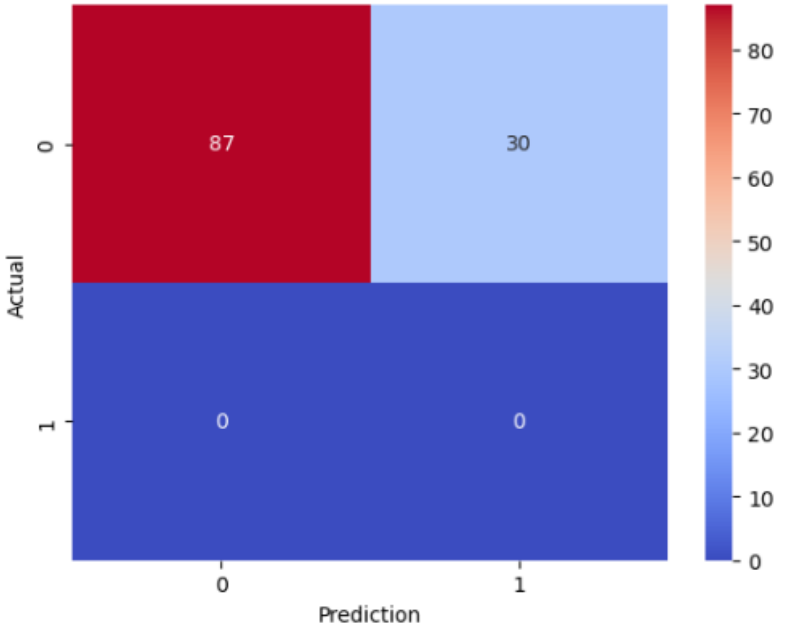
```
In [16]: svm_pred = svm.predict(X_test)

In [17]: print("r2_score :",r2_score(Y_test,svm_pred))
         print("MSE :",mean_squared_error(Y_test,svm_pred))
         print("MAE :",mean_absolute_error(Y_test,svm_pred))
         print("RMSE :",np.sqrt(mean_squared_error(Y_test,svm_pred)))

         r2_score : -0.34482758620689613
         MSE : 0.2564102564102564
         MAE : 0.2564102564102564
         RMSE : 0.5063696835418333
```

Confusion Matrix:

```
In [34]: cf=confusion_matrix(svm_pred,Y_test)
         sb.heatmap(cf,cmap='coolwarm',annot=True)
         plt.xlabel("Prediction")
         plt.ylabel("Actual")
         plt.show()
```



Accuracy Score:

```
In [30]: svm_pred = svm.predict(X_test)

In [31]: svm_accuracy = accuracy_score(svm_pred,Y_test)

In [32]: print("SVM Accuracy:",svm_accuracy)
         SVM Accuracy: 0.7435897435897436
```

- ***KNN Model:***

  Values:

  r2_score: -0.6586206896551721
  MSE: 0.3162393162393162
  MAE: 0.3162393162393162
  RMSE: 0.5623515948579823

Screenshot:

Performance Metrics

```
In [15]: from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
```

KNN Model

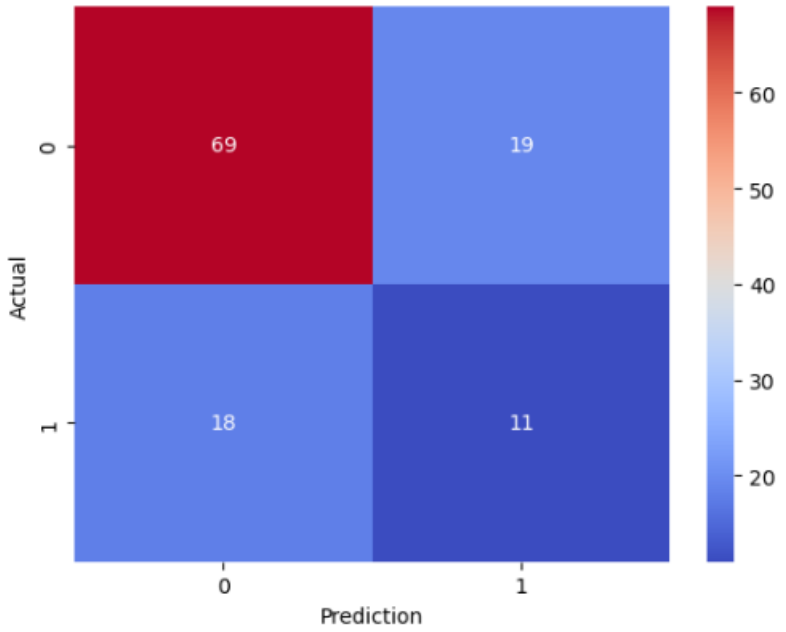
```
In [20]: KNN_pred = KNN.predict(X_test)
```

```
In [21]: print("r2_score :",r2_score(Y_test,KNN_pred))
         print("MSE :",mean_squared_error(Y_test,KNN_pred))
         print("MAE :",mean_absolute_error(Y_test,KNN_pred))
         print("RMSE :",np.sqrt(mean_squared_error(Y_test,KNN_pred)))

         r2_score : -0.6586206896551721
         MSE : 0.3162393162393162
         MAE : 0.3162393162393162
         RMSE : 0.5623515948579823
```

Confusion Matrix:

```
In [44]: cf=confusion_matrix(KNN_pred,Y_test)
         sb.heatmap(cf,cmap='coolwarm',annot=True)
         plt.xlabel("Prediction")
         plt.ylabel("Actual")
         plt.show()
```



Accuracy Score:

```
In [41]: KNN_accuracy = accuracy_score(KNN_pred,Y_test)
```

```
In [42]: print("KNN Accuracy:",KNN_accuracy)

         KNN Accuracy: 0.6837606837606838
```

Report:

Best Accuracy Model:

Compare to the three models RF has more accuracy compared to other models. So, RF accuracy is 76% while testing in Jupiter but the actual accuracy of RF is 81% deployed in IBM Cloud Watson Studio.

- Tune the Model:
  - Hyperparameter Tuning:

Hyperparameter Tuning

In [22]:
```
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
```

In [23]:
```
forest = RandomForestClassifier(random_state = 42)
```

In [24]:
```
n_estimators = [10, 15, 5, 8, 20, 50]
max_features = ['auto', 'sqrt', 'log2']
max_depth = [10, 25, 35, 55]
max_depth.append(None)
min_samples_split = [2, 5, 10, 15, 20]
min_samples_leaf = [1, 2, 5, 10, 15]

hyperF = dict(n_estimators = n_estimators, max_features=max_features, max_depth = max_depth,
              min_samples_split = min_samples_split,
             min_samples_leaf = min_samples_leaf)

gridF = RandomizedSearchCV(forest, hyperF, cv = 5, verbose = 2,
                           n_jobs = -1, n_iter=50)
bestF = gridF.fit(X_train, Y_train)
```

```
Fitting 5 folds for each of 50 candidates, totalling 250 fits
```

```
C:\Users\Nithiya Devi S\anaconda3\lib\site-packages\sklearn\ensemble\_forest.py:427: FutureWarning: `max_features='auto'` has b
een deprecated in 1.1 and will be removed in 1.3. To keep the past behaviour, explicitly set `max_features='sqrt'` or remove th
is parameter as it is also the default value for RandomForestClassifiers and ExtraTreesClassifiers.
  warn(
```

In [25]:
```
print(bestF.best_params_)
```

```
{'n_estimators': 20, 'min_samples_split': 10, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 55}
```

In [26]:
```
rfe_trial = bestF.predict(X_train)
rfe_pred = bestF.predict(X_test)
```

In [27]:
```
print('Random Forest Hyperparameter:')
print('\n')
print(Y_test, rfe_pred)
```

```
Random Forest Hyperparameter:


355    2
407    1
90     1
402    1
268    1
      ..
516    1
305    2
167    1
312    2
329    2
Name: Dataset, Length: 117, dtype: int64 [1 1 1 1 1 1 2 1 2 1 2 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 2 1 1 1
 1 1 1 1 2 1 1 1 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2
 1 1 2 1 2 1 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1 2
 1 2 1 1 1 1]
```

  - Validation Method:

Validation testing

In [28]:
```
from sklearn.model_selection import cross_val_score, cross_val_predict
```

In [30]:
```
RFmodel = RandomForestClassifier(random_state = 42)
```

In [31]:
```
accuracy = cross_val_score(RFmodel,X_train,Y_train,cv=5,scoring="accuracy")
```

In [32]:
```
np.mean(accuracy)
```

Out[32]:
```
0.6931137039579044
```

In [34]:
```
y_pred = cross_val_predict(RFmodel,X_train,Y_train,cv=5)
y_pred
```

Out[34]:
```
array([1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2,
       1, 2, 2, 1, 1, 2, 1, 1, 2, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 2, 1, 1, 1, 1, 2, 1, 2, 2, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1,
       2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1,
       1, 1, 1, 2, 1, 1, 1, 2, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 2, 1, 1, 2, 1, 2,
       1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 1, 1, 2, 1, 1,
       1, 2, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 2,
       2, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1,
```

In [39]:
```
sb.distplot(y_pred)
```

```
C:\Users\Nithiya Devi S\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with si
milar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[39]:
```
<AxesSubplot:ylabel='Density'>
```