Project Development Phase
Model Performance Test

Date	9 November 2023
Team ID	593076
Project Name	Project - Disease Prediction using Machine Learning
Maximum Marks	10 Marks

Model Performance Testing:

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

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Classification Model:** Accuray Score - 0.9523 & Classification Report	

```
[ ]  y_pred = knn_new.predict(X1_val)
     yt_pred = knn_new.predict(X1_train)
     y_pred1 = knn_new.predict(X1_test)
     print('The Training Accuracy of the algorithm is ', accuracy_score(y1_train, yt_pred))
     print('The Validation Accuracy of the algorithm is ', accuracy_score(y1_val, y_pred))
     print('The Testing Accuracy of the algorithm is', accuracy_score(y1_test, y_pred1))

     The Training Accuracy of the algorithm is  0.9639227642276422
     The Validation Accuracy of the algorithm is  0.9613821138211383
     The Testing Accuracy of the algorithm is 0.9523809523809523
```

	precision	recall	f1-score	support
(vertigo) Paroymsal Positional Vertigo	1.00	1.00	1.00	1
AIDS	1.00	1.00	1.00	1
Acne	0.50	1.00	0.67	1
Alcoholic hepatitis	1.00	1.00	1.00	1
Allergy	0.00	0.00	0.00	1
Arthritis	1.00	1.00	1.00	1
Bronchial Asthma	1.00	1.00	1.00	1
Cervical spondylosis	1.00	1.00	1.00	1
Chicken pox	1.00	1.00	1.00	1
Chronic cholestasis	1.00	1.00	1.00	1
Common Cold	1.00	1.00	1.00	1
Dengue	1.00	1.00	1.00	1
Diabetes	1.00	1.00	1.00	1
Dimorphic hemmorhoids(piles)	1.00	1.00	1.00	1
Drug Reaction	1.00	1.00	1.00	1
Fungal infection	1.00	1.00	1.00	2
GERD	1.00	1.00	1.00	1
Gastroenteritis	1.00	1.00	1.00	1
Heart attack	1.00	1.00	1.00	1
Hepatitis B	1.00	1.00	1.00	1
Hepatitis C	1.00	1.00	1.00	1
Hepatitis D	1.00	1.00	1.00	1
Hepatitis E	1.00	1.00	1.00	1
Hypertension	1.00	1.00	1.00	1
Hyperthyroidism	1.00	1.00	1.00	1
Hypoglycemia	1.00	1.00	1.00	1
Hypothyroidism	1.00	1.00	1.00	1
Impetigo	1.00	1.00	1.00	1
Jaundice	1.00	1.00	1.00	1
Malaria	1.00	1.00	1.00	1
Migraine	1.00	1.00	1.00	1
Osteoarthristis	1.00	1.00	1.00	1
Paralysis (brain hemorrhage)	1.00	1.00	1.00	1
Peptic ulcer diseae	1.00	1.00	1.00	1
Pneumonia	1.00	1.00	1.00	1
Psoriasis	1.00	1.00	1.00	1
Tuberculosis	1.00	1.00	1.00	1
Typhoid	1.00	1.00	1.00	1
Urinary tract infection	1.00	1.00	1.00	1
Varicose veins	1.00	1.00	1.00	1
hepatitis A	1.00	1.00	1.00	1
accuracy			0.98	42
macro avg	0.96	0.98	0.97	42
weighted avg	0.96	0.98	0.97	42

2.	Tune the Model	Hyperparameter Tuning - As the accuracies are already so high, we need not do hyperparameter tuning for the models. Validation Method	

```
rfc_results = []
knn_results = []
```

```
for main in [0.020,0.018,0.016,0.014,0.012,0.01,0.008]:
    to_drop = []
    for i,j in zip(feat_imp.keys(),feat_imp.values()):
        if j < main:
            to_drop.append(i)

    X_new = X.drop(to_drop,axis = 1)
    y_new = y
    X1_train, X1_val, y1_train, y1_val = train_test_split(X_new, y_new, test_size=0.2)
    X1_test = X_test.drop(to_drop,axis = 1)
    y1_test = y_test
    rfc_new = RandomForestClassifier()
    rfc_new.fit(X1_train, y1_train)
    temp1 = model_evaluation1(X1_train.shape[1], rfc_new)
    rfc_results.append(temp1)
    knn_new = KNeighborsClassifier()
    knn_new.fit(X1_train, y1_train)
    temp2 = model_evaluation1(X1_train.shape[1],knn_new)
    knn_results.append(temp2)
```

```
randomf = pd.DataFrame(data = rfc_results,columns=['Number of features','Training Accuracy','Testing Accuracy'])
```

```
randomf = pd.DataFrame(data = rfc_results,columns=['Number of features','Training Accuracy','Testing Accuracy'])
```

```
randomf
```

	Number of features	Training Accuracy	Testing Accuracy
0	5	0.168699	0.166667
1	10	0.352642	0.357143
2	21	0.650152	0.642857
3	32	0.884909	0.880952
4	39	0.909299	0.904762
5	46	0.957317	0.952381
6	64	0.997459	1.000000

```
knn_table = pd.DataFrame(data = knn_results,columns=['Number of features','Training Accuracy','Testing Accuracy'])
```

```
knn_table
```

```
to_drop = []
for i,j in zip(feat_imp.keys(),feat_imp.values()):
    if j < 0.01:
        to_drop.append(i)
```

```
len(to_drop)
```

```
43
```

```
X_new = X.drop(to_drop,axis = 1)
y_new = y
```