

Project Development Phase Model Performance Test

Date	17-11-2022
Team ID	PNT2022TMID05406
Project Name	Project – Exploratory Analysis of Rainfall Prediction
Maximum Marks	10 Marks

Model Performance Testing:

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

S.No.	Parameter	Values	Screenshot
1.	Metrics	Regression Model: MAE - , MSE - , RMSE - , R2 score - Classification Model: Confusion Matrix - , Accuracy Score- & Classification Report -	See Below
2.	Tune the Model	Hyperparameter Tuning - Validation Method -	See Below

1. Metrics

Model: Random Forest Classifier

```

Out[72]: RandomForestRegressor(max_depth=100, max_features='sqrt', min_samples_leaf=4,
                               min_samples_split=10, n_estimators=800)

In [73]: y_train_predict=random_forest_model.predict(X_train)
         y_test_predict=random_forest_model.predict(X_test)

In [74]: print("-----Test Data-----")
         print('MAE:', metrics.mean_absolute_error(y_test, y_test_predict))
         print('MSE:', metrics.mean_squared_error(y_test, y_test_predict))
         print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_predict)))

         print("\n-----Train Data-----")
         print('MAE:', metrics.mean_absolute_error(y_train,y_train_predict))
         print('MSE:', metrics.mean_squared_error(y_train, y_train_predict))
         print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_predict)))

-----Test Data-----
MAE: 34.0736351101093
MSE: 2344.903660733718
RMSE: 48.424205318556524

-----Train Data-----
MAE: 25.959317224155633
MSE: 1459.517765598739
RMSE: 38.203635502380386

```

2. Tune the Model

Hyperparameter Tuning:

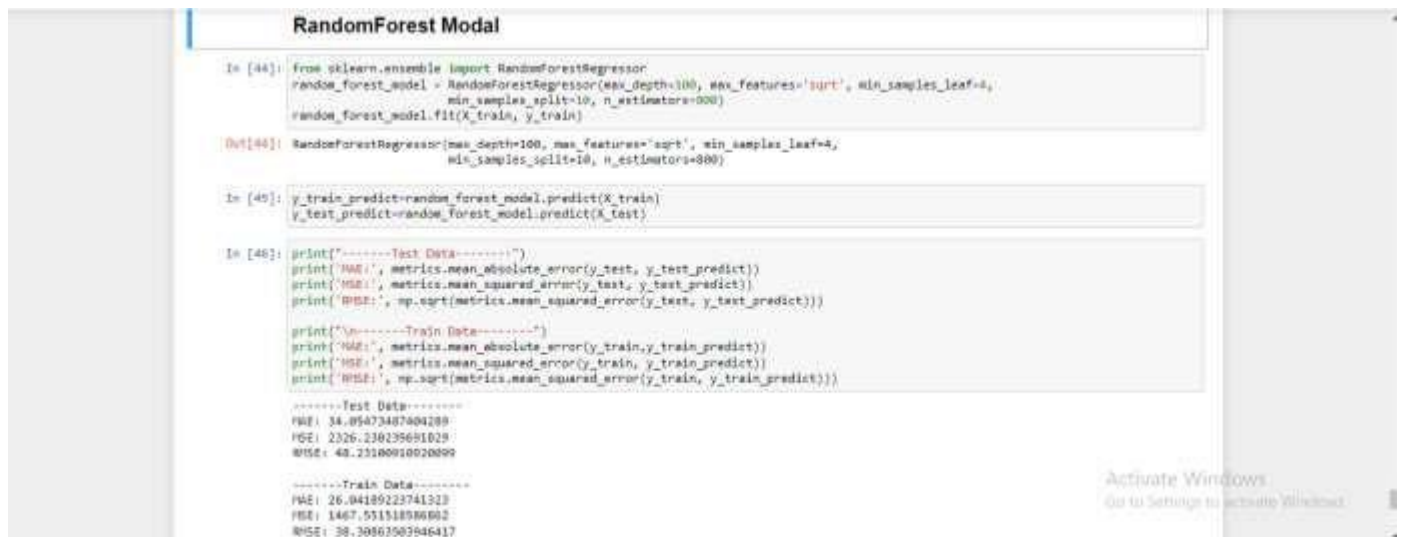
- The number of features is important and should be tuned in random forest classification.
- Initially all parameters in the dataset are taken as independent values to arrive at the dependent decision of Exploratory Analysis of Rainfall Prediction
- But the result was not accurate so used only 8 more correlated values as independent values to arrive at the dependent decision of Exploratory Analysis of Rainfall Prediction

Validation Method:

It involves **partitioning the training data set into subsets, where one subset is held out to test the performance of the model**. This data set is called the validation data set.

Cross validation is to use different models and identify the best:

Linear Regression Model performance values:



```
RandomForest Modal

In [44]: from sklearn.ensemble import RandomForestRegressor
random_forest_model = RandomForestRegressor(max_depth=100, max_features='sqrt', min_samples_leaf=4,
min_samples_split=10, n_estimators=800)
random_forest_model.fit(X_train, y_train)

Out[44]: RandomForestRegressor(max_depth=100, max_features='sqrt', min_samples_leaf=4,
min_samples_split=10, n_estimators=800)

In [45]: y_train_predict=random_forest_model.predict(X_train)
y_test_predict=random_forest_model.predict(X_test)

In [46]: print("-----Test Data-----")
print("MAE:", metrics.mean_absolute_error(y_test, y_test_predict))
print("MSE:", metrics.mean_squared_error(y_test, y_test_predict))
print("RMSE:", np.sqrt(metrics.mean_squared_error(y_test, y_test_predict)))

print("\n-----Train Data-----")
print("MAE:", metrics.mean_absolute_error(y_train, y_train_predict))
print("MSE:", metrics.mean_squared_error(y_train, y_train_predict))
print("RMSE:", np.sqrt(metrics.mean_squared_error(y_train, y_train_predict)))

-----Test Data-----
MAE: 34.85473487404289
MSE: 2326.230235691029
RMSE: 48.231800910020899

-----Train Data-----
MAE: 26.04189223741323
MSE: 1467.551518986692
RMSE: 38.30863083946417
```

Hence we tested with Logistic regression and Random Forest Classification wherein the accuracy of Random Forest classification is 95% compared with Logistic Regression.

Metric	Linear Regression	Random Forest Classification
Accuracy	Testing accuracy : 41.699999999999996 Training accuracy : 33.1	Testing accuracy : 42.5 Training accuracy : 72.5
Other metrics	<pre> from sklearn.linear_model import LogisticRegression model=LogisticRegression(solver='lbfgs',max_iter=500) print('LogisticRegression\n') model.fit(x_train.values,y_train.values.ravel()) prediction = model.predict(x_test) from sklearn.metrics import confusion_matrix print('confusion_matrix') print(confusion_matrix(prediction,y_test)) print('\n') print('accuracy_score') print(accuracy_score(prediction,y_test)) print('\n') </pre> <p>LogisticRegression</p> <p>confusion_matrix</p> <pre>[[49 0] [5 26]]</pre> <p>accuracy_score</p> <p>0.9375</p>	<pre> from sklearn.ensemble import RandomForestClassifier model = RandomForestClassifier() model.fit(x_train , y_train) prediction = model.predict(x_test) print(prediction) from sklearn.metrics import confusion_matrix print('RandomForest\n') print('confusion_matrix') print(confusion_matrix(prediction,y_test)) print('\n') print('accuracy_score') print(accuracy_score(prediction,y_test)) print('\n') </pre> <p>RandomForest</p> <p>confusion_matrix</p> <pre>[[52 1] [2 29]]</pre> <p>accuracy_score</p> <p>0.9425</p>

The above table shows that Random Forest Classification gives better results over Logistic Regression.