Model Development Phase Template

Date	15 March 2024
Team ID	738303
Project Title	Machine Learning Approach For Employee Performance Prediction
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

Random Forest Regression model is the best fit for the employee performance prediction model.

Initial Model Training Code:

```
#splitting data into features and target column
x = data.drop(columns=['actual_productivity','wip'], axis=1)
y = data['actual_productivity']

print(x)

print(y)

#splitting data into train test split
#import train_test_split dependency
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)

x.shape,x_train.shape,x_test.shape
```

Model Validation and Evaluation Report:

Note: Machine Learning Approach For Emplyee Performance Prediction is a regression model . It is not possible to create or fit a confusion matrix for regression models.

So we cannot create confusion matrix for this project.

Model	Regression Report	MSE	Confusion Matrix

<table>
<tr>
<td>Linear regression model</td>
<td>

```
#model building
#importing linear regression dependency
from sklearn.linear_model import LinearRegression
linear=LinearRegression()
linear.fit(x_train,y_train)
#linear model mean squared error
score_train=linear.predict(x_train)
mse_train=mean_squared_error(y_train,score_train)
print("mean squared error in training data in linear regression is:",mse_train)

score_test=linear.predict(x_test)
mse_test=mean_squared_error(y_test,score_test)
print("mean squared error in testing data in linear regression is:",mse_test)

#linear model r2_score
score_train=linear.predict(x_train)
mse_train=r2_score(y_train,score_train)
print("r2_score in training data in linear regression is:",mse_train)

score_test=linear.predict(x_test)
mse_test=r2_score(y_test,score_test)
print("r2_score in test data in linear regression is:",mse_test)

#linear model mean_absolute_error
score_train=linear.predict(x_train)
mse_train=mean_absolute_error(y_train,score_train)
print("mean_absolute_error in training data in linear regression is:",mse_train)

score_test=linear.predict(x_test)
mse_test=mean_absolute_error(y_test,score_test)
print("mean_absolute_error in testing data in linear regression is:",mse_test)
```

(variable) mse_train: Float

</td>
<td>0.021</td>
<td>

	Actual Value	predicted_value
921	0.268214	0.432858
321	0.800359	0.799398
101	0.681061	0.671121
920	0.325000	0.591028
58	0.667604	0.593638
790	0.800980	0.735931
948	0.768847	0.549655
969	0.768847	0.526311
410	0.650417	0.631047
1079	0.750396	0.750391

</td>
</tr>
<tr>
<td>Random forest model</td>
<td>

```
#Random Forest Regressor
from sklearn.ensemble import RandomForestRegressor
RandomForest = RandomForestRegressor()
RandomForest.fit(x_train, y_train)

#Random Forest Regressor mean squared error
score_train=RandomForest.predict(x_train)
mse_train=mean_squared_error(y_train,score_train)
print("mean squared error in training data in Random Forest Regressor is:",mse_train)

score_test=RandomForest.predict(x_test)
mse_test=mean_squared_error(y_test,score_test)
print("mean squared error in testing data in Random Forest Regressor is:",mse_test)

#Random Forest Regressor r2_score
score_train=RandomForest.predict(x_train)
mse_train=r2_score(y_train,score_train)
print("r2_score in training data in Random Forest Regressor is:",mse_train)

score_test=RandomForest.predict(x_test)
mse_test=r2_score(y_test,score_test)
print("r2_score in test data in Random Forest Regressor is:",mse_test)

#Random Forest Regressor mean_absolute_error
score_train=linear.predict(x_train)
mse_train=mean_absolute_error(y_train,score_train)
print("mean_absolute_error in training data in Random Forest Regressor is:",mse_train)

score_test=linear.predict(x_test)
mse_test=mean_absolute_error(y_test,score_test)
print("mean_absolute_error in testing data in Random Forest Regressor is:",mse_test)
```

</td>
<td>0.0120</td>
<td>

	Actual Value	predicted_value
921	0.268214	0.432858
321	0.800359	0.799398
101	0.681061	0.671121
920	0.325000	0.591028
58	0.667604	0.593638
790	0.800980	0.735931
948	0.768847	0.549655
969	0.768847	0.526311
410	0.650417	0.631047
1079	0.750396	0.750391

</td>
</tr>
</table>

Xgboost model

```
#Xgboost regression
import xgboost as xgb
model_xgb=xgb.XGBRegressor(n_estimators=200,max_depth=5,learning_rate=0.1)
model_xgb.fit(x_train,y_train)


#Xgboost mean squared error
score_train=model_xgb.predict(x_train)
mse_train=mean_squared_error(y_train,score_train)
print("mean squared error in training data in Xgboost regression is:",mse_train)

score_test=model_xgb.predict(x_test)
mse_test=mean_squared_error(y_test,score_test)
print("mean squared error in testing data in Xgboost regressionr is:",mse_test)

#Xgboost Regressor r2_score
score_train=model_xgb.predict(x_train)
mse_train=r2_score(y_train,score_train)
print("r2_score in training data in Xgboost regression is:",mse_train)

score_test=model_xgb.predict(x_test)
mse_test=r2_score(y_test,score_test)
print("r2_score in test data in Random Xgboost regressionr is:",mse_test)

#Xgboost regression mean_absolute_error
score_train=linear.predict(x_train)
mse_train=mean_absolute_error(y_train,score_train)
print("mean_absolute_error in training data in Xgboost regression is:",mse_train)

score_test=linear.predict(x_test)
mse_test=mean_absolute_error(y_test,score_test)
print("mean_absolute_error in testing data in Xgboost regression is:",mse_test)
```

0.0133

	Actual Value	predicted_value
921	0.268214	0.432858
321	0.800359	0.799398
101	0.681061	0.671121
920	0.325000	0.591028
58	0.667604	0.593638
790	0.800980	0.735931
948	0.768847	0.549655
969	0.768847	0.526311
410	0.650417	0.631047
1079	0.750396	0.750391