multiple_linear_regression

January 27, 2022

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
#Import Data
df = pd.read_csv("ml_data_salary.csv")
df.head(2)
```

[ ]:

	age	distance	YearsExperience	Salary
0	31.1	77.75	1.1	39343
1	31.3	78.25	1.3	46205

[ ]:
```
X = df[['age','distance','YearsExperience']]
y=df['Salary']
```

0.1 Creating a model and Data fitting

[ ]:
```
model= LinearRegression().fit(X,y)
model
```

[ ]:
```
LinearRegression()
```

0.1.1 Checking coefficients of Input and Slope

[ ]:
```
model.coef_
```

[ ]:
```
array([-3.00216193e+15,  1.18788781e+15,  3.24424072e+13])
```

[ ]:
```
model.intercept_
```

[ ]:
```
973272214586587.5
```

[ ]:
```
model.predict([[31.1,80,1.1]])
```

```
C:\Users\dell7450\AppData\Local\Programs\Python\Python310\lib\site-
packages\sklearn\base.py:450: UserWarning: X does not have valid feature names,
but LinearRegression was fitted with feature names
  warnings.warn(
```

[ ]:
```
array([2.67274757e+15])
```

0.1.2 Splitting and Training (80-20 Data)

[ ]:
```
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=0)

model = LinearRegression().fit(X_train, y_train)
model
```

[ ]:
```
LinearRegression()
```

[ ]:
```
# Assignment is how to plot multiple linear regression model
# How to test efficacy of the model? ( split train / test)
```

0.1.3 Regression Score (Accuracy Measurement)

https://scikit-learn.org/stable/modules/model_evaluation.html#mean-absolute-percentage-error

[ ]:
```
#reg= LinearRegression().fit(X_test, y_test)
print("Regression score without splitting =",model.score(X,y))
print("After splitting my train score =",model.score(X_train,y_train))
# this test will not work on numeric It will only work on classification data␣
 ↪like in decision tree algorithm
print(" After splitting test score =",model.score(X_test,y_test))
```

```
Regression score without splitting = 0.9565684395539251
After splitting my train score = 0.9409532368371482
 After splitting test score = 0.988401541985491
```

0.1.4 Score Checking

[ ]:
```
from sklearn.metrics import accuracy_score
y_pred = model.predict(X_test)
y_pred
# Compare with side p rakhi we test vs predicted test
score = accuracy_score(y_test,y_pred,normalize=False)
print("The accuracy score of model when compared with two test values is",score)
```

```
The accuracy score of model when compared with two test values is 0
```

Explained Variance Score

[ ]:
```
from sklearn.metrics import explained_variance_score
explained_variance_score(y_test, y_pred)
```

[ ]:
```
0.9896930311538696
```

Max Error

[ ]:
```
from sklearn.metrics import max_error
max_error(y_test, y_pred)
```

[ ]:
```
7751.0
```

Mean Absolute Error

[ ]:
```
from sklearn.metrics import mean_absolute_error
mean_absolute_error(y_test, y_pred)
```

[ ]:
```
2469.1666666666665
```

Mean Squared Error

[ ]:
```
 from sklearn.metrics import mean_squared_error
 mean_squared_error(y_test, y_pred)
```

[ ]:
```
12571912.166666666
```

Mean Absolute Percentage Error

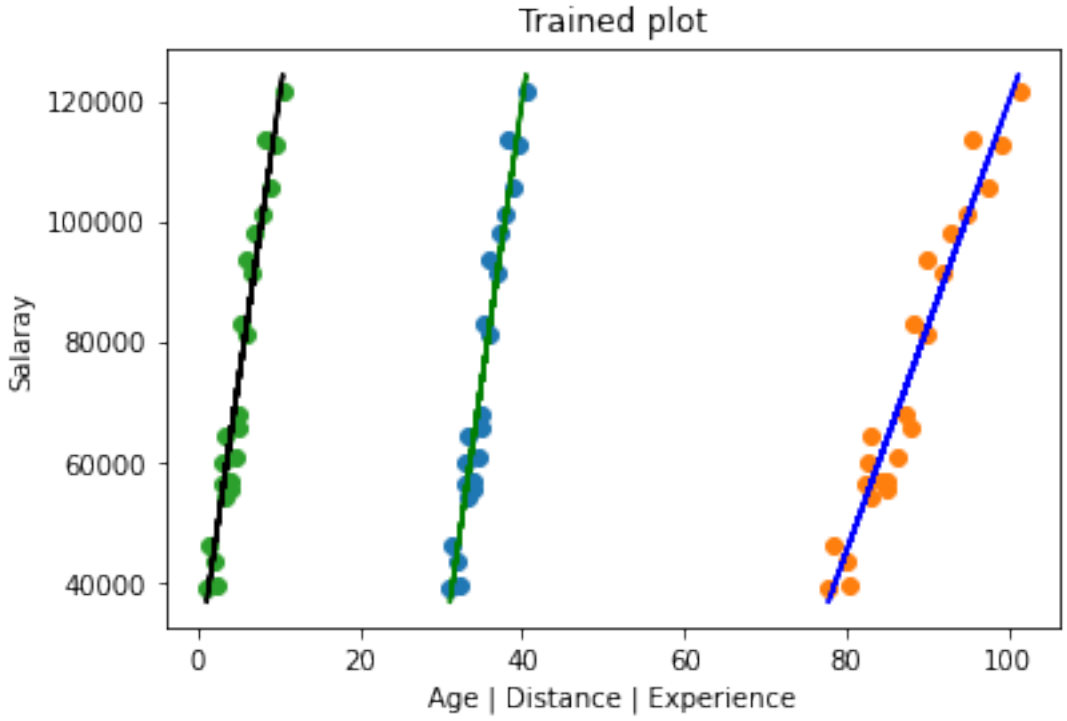
[ ]:
```
from sklearn.metrics import mean_absolute_percentage_error
mean_absolute_percentage_error(y_test, y_pred)
```

[ ]:
```
0.041779872719803136
```

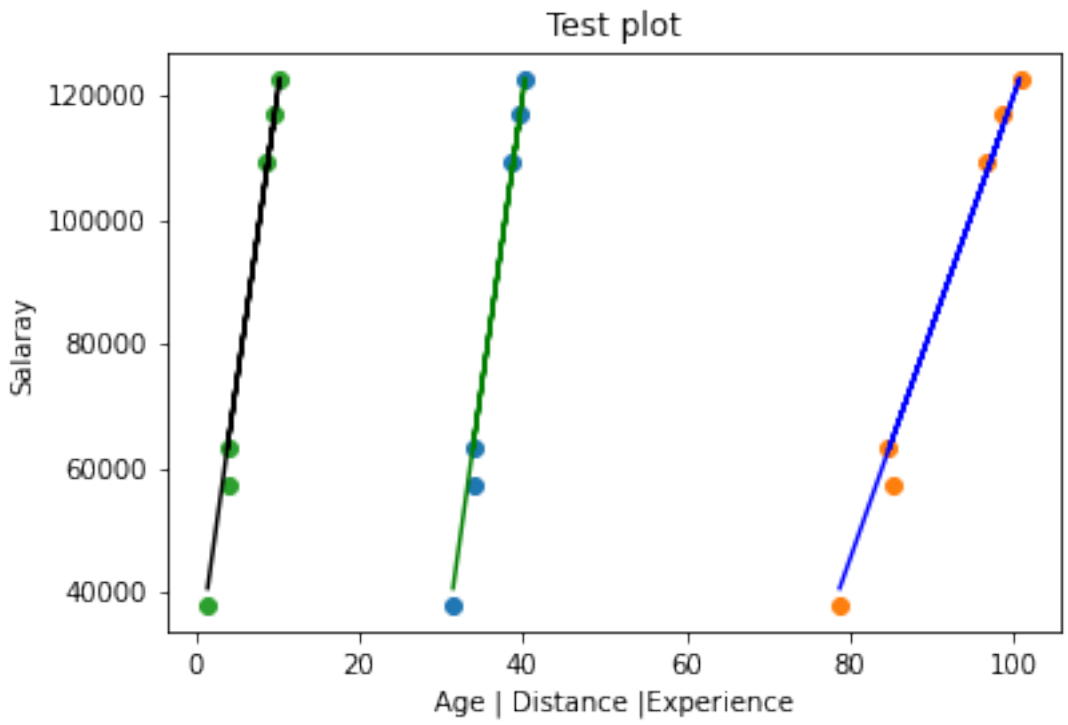
0.1.5 Plotting multiple Linear Regression Model

[ ]:
```
import matplotlib.pyplot as plt
plt.scatter(X_train.age , y_train)
plt.plot(X_train.age, model.predict(X_train), color='green')

plt.scatter(X_train.distance , y_train)
plt.plot(X_train.distance, model.predict(X_train), color='blue')

plt.scatter(X_train.YearsExperience , y_train)
plt.plot(X_train.YearsExperience, model.predict(X_train), color='black')

plt.xlabel("Age | Distance | Experience")
plt.ylabel("Salaray")
plt.title("Trained plot")
plt.show()


plt.scatter(X_test.age, y_test)
plt.plot(X_test.age, model.predict(X_test), color='green')

plt.scatter(X_test.distance, y_test)
plt.plot(X_test.distance, model.predict(X_test), color='blue')
```

```
plt.scatter(X_test.YearsExperience , y_test)
plt.plot(X_test.YearsExperience, model.predict(X_test), color='black')

plt.xlabel("Age | Distance |Experience")
plt.ylabel("Salaray")
plt.title("Test plot")
```



[ ]: Text(0.5, 1.0, 'Test plot')



0.1.6 Prediction of future and Test Values

[ ]:
```
# Predicting the Test set results
y_pred = model.predict(X_test)
y_pred
```

[ ]:
```
array([ 40640., 122688.,  64832.,  63040., 115136., 107584.])
```

[ ]:
```
# Predicitng fixed values
X_testin = [[28,45,1.1],[22,23,3,]]
y_pred = model.predict(X_testin)
y_pred
```

```
C:\Users\dell7450\AppData\Local\Programs\Python\Python310\lib\site-
packages\sklearn\base.py:450: UserWarning: X does not have valid feature names,
but LinearRegression was fitted with feature names
  warnings.warn(
```

[ ]:
```
array([4.70711563e+16, 1.52251782e+17])
```

1 Doctor Sahab recommended in plots to use 3D plots with multiple lines

Must explore this because you used only 2d plot for this

Out of sample accuracy increase by splitting

[ ]: