Assignment 3: Individual Assignment

Remmy Bisimbeko - B26099 - J24M19/011

My GitHub - https://github.com/RemmyBisimbeko/Data-Science

Instructions:

The National Coffee Research Institute generated disease resistant coffee varieties which they analysed for the quality using a cupping assessment.

You will use the provided Excel workbook labelled "Uganda Coffee_Cupping.xlsx" Uganda Coffee_Cupping.xlsx containing three sheets:

a) Sheet 1; Training Data b) Sheet 2: Test Data 1 c) Sheet 3: Test Data 2

Transform the datasets and make predictions on the "OVERALL SCORE" of coffee in two different areas of ORIGIN;

1. Kayunga (10 MARKS)
2. Rwenzori (10 MARKS)

In [ ]:
```
# Import Libs
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

In [ ]:
```
# loding the data into a DataFrame
df = pd.read_excel('Data Sets/Uganda Coffee_Cupping.xlsx', sheet_name='Tr
```

In [ ]:
```
# Preprocess the data
# Drop any unnecessary columns - or delete manually from excel
# df = df.drop(['Unnamed: 0'], axis=1)
```

In [ ]:
```
# I now convert categorical variables to dummy variables
df = pd.get_dummies(df, columns=['ORIGIN', 'VARIETY'])
```

In [ ]:
```
# Spliting the data into features (X) and target (y)
X = df.drop(['OVERALL SCORE'], axis=1)
y = df['OVERALL SCORE']
```

In [ ]:
```
# Exclude the columns that are not present in the test data
X = X.drop(['ORIGIN_Ibanda', 'ORIGIN_Mityana', 'ORIGIN_Mukono'], axis=1)
```

In [ ]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
```

In [ ]:
```
# Train machine learning model
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[ ]:
```
▾ LinearRegression
LinearRegression()
```

In [ ]:
```
# generate predictions for the Kayunga region

kayunga_df = pd.read_excel('Data Sets/Uganda Coffee_Cupping.xlsx', sheet_
# kayunga_df = kayunga_df.drop(['Unnamed: 0'], axis=1)
```

In [ ]:
```
# Convert categorical variables to dummy variables with the same prefix a
kayunga_df = pd.get_dummies(kayunga_df, columns=['ORIGIN', 'VARIETY'], pr
```

In [ ]:
```
# Then re-order the columns to match the training data
kayunga_df = kayunga_df[X_train.columns]
```

In [ ]:
```
print(df.columns)
df.dtypes
```

```
Index(['FRAGRANCE/AROMA', 'FLAVOR', 'SALT/ ACID', 'BITTER/ SWEET',
       'AFTERTASTE', 'MOUTH FEEL', 'BALANCE', 'UNIFORMITY', 'CLEAN CUPS',
       'OVERALL SCORE', 'ORIGIN_Ibanda', 'ORIGIN_Mityana', 'ORIGIN_Mukon
o',
       'VARIETY_KR3', 'VARIETY_KR4', 'VARIETY_KR5', 'VARIETY_KR6',
       'VARIETY_KR7'],
      dtype='object')
```

Out[ ]:
```
FRAGRANCE/AROMA    float64
FLAVOR             float64
SALT/ ACID         float64
BITTER/ SWEET      float64
AFTERTASTE         float64
MOUTH FEEL         float64
BALANCE            float64
UNIFORMITY           int64
CLEAN CUPS           int64
OVERALL SCORE      float64
ORIGIN_Ibanda         bool
ORIGIN_Mityana        bool
ORIGIN_Mukono         bool
VARIETY_KR3           bool
VARIETY_KR4           bool
VARIETY_KR5           bool
VARIETY_KR6           bool
VARIETY_KR7           bool
dtype: object
```

In [ ]:
```
kayunga_df['OVERALL SCORE'] = np.nan

for index, row in kayunga_df.iterrows():
    # Calculate the overall score for this row based on the other columns
    kayunga_df.loc[index, 'OVERALL SCORE'] = row[['FRAGRANCE/AROMA', 'FLA

print(kayunga_df)
```

	FRAGRANCE/AROMA	FLAVOR	SALT/ ACID	BITTER/ SWEET	AFTERTASTE	\
0	7.00	6.00	6.50	6.75	6.50	
1	8.00	7.75	7.00	7.00	7.25	
2	8.25	7.25	7.00	7.00	6.75	
3	7.50	7.75	7.50	7.50	7.25	
4	7.00	7.25	8.00	8.00	7.75	
5	8.00	7.00	7.00	7.00	7.50	
6	8.00	8.00	7.00	7.00	7.25	
7	8.00	7.75	7.50	7.50	7.75	
8	7.50	7.75	7.00	7.25	7.25	
9	7.00	8.00	7.50	8.00	7.00	
10	7.25	7.00	7.00	7.00	6.50	
11	6.50	7.00	6.75	7.00	6.00	
12	7.75	7.00	7.00	7.00	7.00	
13	7.00	7.00	7.00	7.00	7.00	
14	7.00	7.25	7.00	7.25	7.25	
15	7.75	7.00	6.75	6.50	7.25	
16	8.25	7.75	7.00	7.00	7.25	
17	7.00	7.50	7.50	7.25	7.50	
18	7.25	7.50	7.00	7.25	7.00	
19	7.25	8.00	7.75	8.00	8.00	
20	7.50	7.00	7.00	7.00	7.00	
21	8.00	7.75	7.00	7.25	7.75	
22	8.00	7.50	7.25	7.50	7.00	
23	7.25	7.50	7.00	7.25	7.50	
24	7.00	7.00	6.75	6.75	7.00	

	MOUTH FEEL	BALANCE	UNIFORMITY	CLEAN CUPS	VARIETY_KR3	VARIETY_KR4	\
0	6.75	8.00	10	10	True	False	
1	7.00	7.00	10	10	True	False	
2	6.75	7.00	10	10	True	False	
3	7.75	7.50	10	10	True	False	
4	7.75	7.00	10	10	True	False	
5	7.00	7.75	10	10	False	True	
6	7.00	7.00	10	10	False	True	
7	7.75	7.25	10	10	False	True	
8	7.25	7.25	10	10	False	True	
9	7.75	7.50	10	10	False	True	
10	6.75	7.00	8	8	False	False	
11	6.50	7.00	8	8	False	False	
12	7.00	7.00	8	8	False	False	
13	7.00	8.00	8	8	False	False	
14	7.25	7.00	8	8	False	False	
15	6.50	7.00	10	10	False	False	
16	7.00	7.00	10	10	False	False	
17	7.50	7.25	10	10	False	False	
18	7.50	7.50	10	10	False	False	
19	7.75	7.75	10	10	False	False	
20	6.75	8.00	10	10	False	False	
21	7.00	7.00	10	10	False	False	
22	7.00	7.00	10	10	False	False	
23	7.50	7.25	10	10	False	False	
24	7.00	7.00	10	10	False	False	

	VARIETY_KR5	VARIETY_KR6	VARIETY_KR7	OVERALL SCORE
0	False	False	False	7.500000
1	False	False	False	7.888889
2	False	False	False	7.777778
3	False	False	False	8.083333

```
4         False        False        False       8.083333
5         False        False        False       7.916667
6         False        False        False       7.916667
7         False        False        False       8.166667
8         False        False        False       7.916667
9         False        False        False       8.083333
10         True        False        False       7.166667
11         True        False        False       6.972222
12         True        False        False       7.305556
13         True        False        False       7.333333
14         True        False        False       7.333333
15        False         True        False       7.638889
16        False         True        False       7.916667
17        False         True        False       7.944444
18        False         True        False       7.888889
19        False         True        False       8.277778
20        False        False         True       7.805556
21        False        False         True       7.972222
22        False        False         True       7.916667
23        False        False         True       7.916667
24        False        False         True       7.611111
```

In [ ]:
```
kayunga_predictions = model.predict(kayunga_df.drop(['OVERALL SCORE'], ax
# Final predictions Displayed at the bottom
```

In [ ]:
```
# NOw I generate predictions for the Rwenzori region
rwenzori_df = pd.read_excel('Data Sets/Uganda Coffee_Cupping.xlsx', sheet
# rwenzori_df = rwenzori_df.drop(['Unnamed: 0'], axis=1)
```

In [ ]:
```
# Convert categorical variables to dummy variables with the same prefix a
rwenzori_df = pd.get_dummies(rwenzori_df, columns=['ORIGIN', 'VARIETY'],
```

In [ ]:
```
# I did the reorder the columns to match the training data
rwenzori_df = rwenzori_df[X_train.columns]
```

In [ ]:
```
# How's data lookin like?
print(df.columns)
df.dtypes
```

```
Index(['FRAGRANCE/AROMA', 'FLAVOR', 'SALT/ ACID', 'BITTER/ SWEET',
       'AFTERTASTE', 'MOUTH FEEL', 'BALANCE', 'UNIFORMITY', 'CLEAN CUPS',
       'OVERALL SCORE', 'ORIGIN_Ibanda', 'ORIGIN_Mityana', 'ORIGIN_Mukon
o',
       'VARIETY_KR3', 'VARIETY_KR4', 'VARIETY_KR5', 'VARIETY_KR6',
       'VARIETY_KR7'],
      dtype='object')
```

Out[ ]:
```
FRAGRANCE/AROMA    float64
FLAVOR             float64
SALT/ ACID         float64
BITTER/ SWEET      float64
AFTERTASTE         float64
MOUTH FEEL         float64
BALANCE            float64
UNIFORMITY           int64
CLEAN CUPS           int64
OVERALL SCORE      float64
ORIGIN_Ibanda         bool
ORIGIN_Mityana        bool
ORIGIN_Mukono         bool
VARIETY_KR3           bool
VARIETY_KR4           bool
VARIETY_KR5           bool
VARIETY_KR6           bool
VARIETY_KR7           bool
dtype: object
```

In [ ]:
```
# Fix for Overal Score error
rwenzori_df['OVERALL SCORE'] = np.nan

for index, row in rwenzori_df.iterrows():
    # Calculate the overall score for this row based on the other columns
    rwenzori_df.loc[index, 'OVERALL SCORE'] = row[['FRAGRANCE/AROMA', 'FL

print(rwenzori_df)
```

	FRAGRANCE/AROMA	FLAVOR	SALT/ ACID	BITTER/ SWEET	AFTERTASTE	\
0	7.25	6.75	6.75	7.50	6.75	
1	7.50	7.75	7.50	7.75	7.50	
2	7.55	8.00	7.00	7.75	8.00	
3	8.00	7.00	6.00	7.00	7.00	
4	6.50	7.00	6.75	7.00	7.00	
5	7.00	7.00	7.00	7.25	7.00	
6	7.00	7.75	7.75	7.75	7.50	
7	7.25	8.00	7.00	7.25	8.25	
8	8.00	7.00	7.00	6.00	7.00	
9	7.00	7.00	7.00	7.00	7.00	
10	6.75	7.00	6.75	7.25	7.00	
11	7.00	7.50	7.00	7.50	7.25	
12	6.75	7.75	8.00	7.75	8.00	
13	8.00	6.50	7.00	7.00	6.50	
14	7.25	7.00	7.50	7.50	7.25	
15	7.75	7.75	7.25	7.50	7.50	
16	7.25	7.50	7.75	7.25	7.25	
17	7.25	7.00	7.00	7.50	7.00	
18	7.00	8.00	6.75	7.00	7.50	
19	8.00	7.75	7.00	7.00	7.00	
20	7.25	7.00	7.25	7.00	7.00	
21	7.00	7.50	7.75	7.50	7.50	
22	7.60	7.60	7.00	6.75	7.25	
23	7.00	6.50	6.75	7.00	7.00	
24	7.50	7.50	7.00	7.00	7.50	

	MOUTH FEEL	BALANCE	UNIFORMITY	CLEAN CUPS	VARIETY_KR3	VARIETY_KR4	\
0	6.75	6.75	10	10	True	False	
1	7.50	7.50	10	10	True	False	
2	8.00	7.75	10	10	True	False	
3	7.00	10.00	10	10	True	False	
4	7.00	7.00	10	10	True	False	
5	7.25	7.50	10	10	False	True	
6	7.25	7.75	10	10	False	True	
7	8.00	7.50	10	10	False	True	
8	7.00	10.00	10	10	False	True	
9	7.00	6.75	10	10	False	True	
10	7.25	6.75	10	10	False	False	
11	7.25	7.50	10	10	False	False	
12	7.75	7.75	10	10	False	False	
13	6.75	7.50	10	10	False	False	
14	7.25	7.50	10	10	False	False	
15	7.00	7.75	10	10	False	False	
16	8.00	7.50	10	10	False	False	
17	7.00	7.00	10	10	False	False	
18	7.00	7.75	10	10	False	False	
19	7.00	7.50	10	10	False	False	
20	7.00	7.50	10	10	False	False	
21	7.50	7.25	10	10	False	False	
22	7.00	7.00	10	10	False	False	
23	7.00	10.00	10	10	False	False	
24	7.50	7.50	10	10	False	False	

	VARIETY_KR5	VARIETY_KR6	VARIETY_KR7	OVERALL SCORE
0	False	False	False	7.611111
1	False	False	False	8.111111
2	False	False	False	8.227778
3	False	False	False	8.000000

```
4         False        False        False       7.583333
5         False        False        False       7.777778
6         False        False        False       8.083333
7         False        False        False       8.138889
8         False        False        False       8.000000
9         False        False        False       7.638889
10         True        False        False       7.638889
11         True        False        False       7.888889
12         True        False        False       8.194444
13         True        False        False       7.694444
14         True        False        False       7.916667
15        False         True        False       8.055556
16        False         True        False       8.055556
17        False         True        False       7.750000
18        False         True        False       7.888889
19        False         True        False       7.916667
20        False        False         True       7.777778
21        False        False         True       8.000000
22        False        False         True       7.800000
23        False        False         True       7.916667
24        False        False         True       7.944444
```

In [ ]:
```
rwenzori_predictions = model.predict(rwenzori_df.drop(['OVERALL SCORE'],
# Final predictions all displayed at the bottom
```

In [ ]:
```
# Finally, display the final result
print("Predictions for Kayunga:")
print(kayunga_predictions)
print("\nPredictions for Rwenzori:")
print(rwenzori_predictions)
```

```
Predictions for Kayunga:
[7.0234375  7.7734375  7.5234375  7.6484375  7.7734375  7.7734375
 7.7734375  8.0234375  7.6484375  7.2734375  6.8984375  6.6484375
 7.3984375  7.5234375  7.3984375  7.56054688 7.68554688 7.43554688
 7.18554688 8.06054688 7.43945312 7.81445312 7.31445312 7.43945312
 7.06445312]

Predictions for Rwenzori:
[7.0234375  7.8984375  8.1484375  7.8984375  7.1484375  7.2734375
 7.7734375  8.1484375  8.0234375  7.1484375  6.8984375  7.2734375
 7.8984375  6.8984375  7.2734375  7.81054688 7.31054688 7.06054688
 7.68554688 7.43554688 7.18945312 7.43945312 7.56445312 7.56445312
 7.56445312]
```