6BUIS017W.1

Customer Relationship and Change Management

CW - 2

Module Leader: Mr. Trevor Soris

Degree: BSc (Hons) in Business Data Analytics

Name: Ahmed Ammar Ismeth Mohideen

UOW ID - w1869538

IIT ID – 2021053

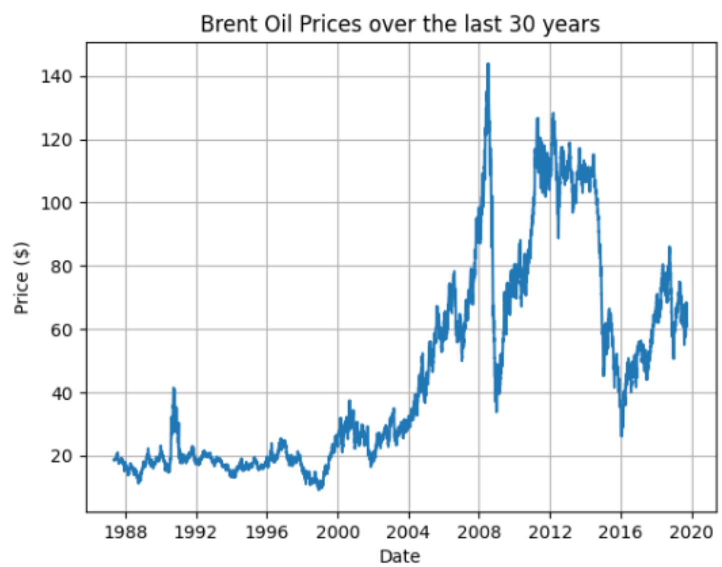
Table of Contents

TASK A

A. Data Visualization

Plotting and comment on the simple line chart using Python.

```
[ ] # Plot Brent oil prices
    plt.plot(data['Date'], data['Price'])
    plt.title('Brent Oil Prices over the last 30 years')
    plt.xlabel('Date')
    plt.ylabel('Price ($)')
    plt.grid()
    plt.show()
```



2. Comment on line chart behavior :

To relate these observations to specific highs and lows in the prices of Brent crude, the following can be pointed out:

Highs:

1. 1990–1991 Gulf War: Brent crude prices reached an all-time high because of Iraq's invasion of Kuwait, raising fears of supply disruption. The chart shows a distinct peak in the diagram during this time.

2.2008 Financial Crisis (Pre-Collapse): Oil prices reached an all-time high of about $145 per barrel in mid-2008, reflecting strong demand during global economic expansion.

3. 2011–2014 Stability: High prices prevailed within the band of $80 to $120, supported by high demand and limited supply.

Lows:

1.1998 Asian Financial Crisis:
This should reflect a marked low as lower demand from the region saw the price of Brent crude fall.

2.2008 Financial Crisis (Collapse):
From $145, this plummeted down to about $40 per barrel as economies of the world contracted.

3.2014–2016 Oil Price Collapse:
The prices witnessed a decline of 44% because of over-supply led mainly by US shale oil and weakening demand worldwide.

B) Building explanatory variables.

1.Building the explanatory variables.

Two moving averages are calculated:

- **MA3**: This moving average calculates the average price over the last three days.
- **MA9**: This moving average calculates the average price over the last nine days.

Question B

```
[ ] data['MA3'] = data['Price'].shift(1).rolling(window=3).mean()
    data['MA9']= data['Price'].shift(1).rolling(window=9).mean()
```

```
data
```

	Date	Price	MA3	MA9
0	1987-05-20	18.63	NaN	NaN
1	1987-05-21	18.45	NaN	NaN
2	1987-05-22	18.55	NaN	NaN
3	1987-05-25	18.60	18.543333	NaN
4	1987-05-26	18.63	18.533333	NaN
...	...	...	...	...
8211	2019-09-24	64.13	64.713333	64.163333
8212	2019-09-25	62.41	64.673333	64.286667
8213	2019-09-26	62.08	63.733333	64.470000
8214	2019-09-27	62.48	62.873333	64.562222
8215	2019-09-30	60.99	62.323333	63.902222

8216 rows × 4 columns

Purpose of using Moving Averages :

1.Feature Engineering for Prediction:

Moving averages transform the raw, past price data into simplified metrics—MA3 and MA9—that capture recent trends. These are then used as input features for the linear regression model in predicting future prices.

2.Smoothing Price Fluctuations:

They smooth out short-term volatility, filtering out noise from daily oil price fluctuations. This will make the model focus on meaningful trends rather than erratic movements.

3.Identifying Short- and Long-Term Trends:

MA3: Captures short-term trends over the past 3 days, reflecting immediate price momentum.

MA9: Represents broader, long-term trends over the past 9 days, providing insights into the general direction of price movements.

4.Lagging Indicators:

Since moving averages are calculated using past data, they act as lagging indicators summarizing historical price actions. This ensures that information available before the prediction point is used, thereby avoiding data leakage.

5.Enhancing Predictive Robustness:

By combining short- and long-term averages, the model can better understand the relationship between past trends and future prices, enhancing its predictive power and robustness.

In financial time-series data, such as Brent oil prices, moving averages are critical because:

They highlight trends without being overly influenced by short-term anomalies.
They are very important inputs to regression models, which depend on meaningful patterns for accurate predictions.

C) Define the Train and Test Data

Question C

```
[ ] # Initialising X and assigning the two feature variables
    X = data[['MA3','MA9']]
```

```
[ ] # Setting-up the dependent variable
    Y = data['Price']
```

```
[ ] X,Y
```

```
(            MA3        MA9
 0           NaN        NaN
 1           NaN        NaN
 2           NaN        NaN
 3     18.543333        NaN
 4     18.533333        NaN
 ...         ...        ...
 8211  64.713333  64.163333
 8212  64.673333  64.286667
 8213  63.733333  64.470000
 8214  62.873333  64.562222
 8215  62.323333  63.902222

 [8216 rows x 2 columns],
 0        18.63
 1        18.45
 2        18.55
 3        18.60
 4        18.63
          ...
 8211     64.13
 8212     62.41
 8213     62.08
 8214     62.48
 8215     60.99
 Name: Price, Length: 8216, dtype: float64)
```

Linear regression models require independent variables X that explain the dependent variable Y. Since the moving averages are constructed to capture short- and long-run trends of oil prices, they are appropriate features for the model.

In this case, the relationship between the moving averages and the actual price is what the model is trying to capture.

```
from sklearn.model_selection import train_test_split
import numpy as np
```

```
[ ] # Define Train and Test Data
    X = data[['MA3', 'MA9']]
    y = data['Price']
```

```
[ ] # Fill NaN values with 0 in X
    X = X.fillna(0)
```

```
[ ] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Define Reproducible Seed
    np.random.seed(42)
```

Why set NaN with 0 Specifically?

1.Neutral Impact on Model:

At least setting NaNs to 0 let the model work with the data, 0 is usually a neutral reference level.

2.Avoid Overfitting:

Replacing NaNs with more complex imputation methods, like mean or median, may introduce bias; 0 keeps things simple and avoids unintended patterns.

Training-Testing Split:

Ensures that the model is trained on one subset of data and validated on another, preventing "data leakage" where the model might learn from the test data and inflate performance metrics.

Reproducibility(Seed):

Critical for iterative development, where you want to compare models or configurations under identical conditions.

D) Building a Linear Regression Model (LR) using Moving Averages (MA3) and (MA9) as inputs :

Question D

```
[ ] from sklearn.linear_model import LinearRegression
```

```
[ ] #Build and Train Linear Regression Model
    model = LinearRegression()
    model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

The fit method enables the model to learn the relationship between the explanatory variables (MA3, MA9) and the target variable (Price). Once the model is trained, it can be used to make predictions on new, unseen data (X_test) using the learned coefficients and intercept. Linear regression provides coefficients ($\beta 1,\beta 2$) that quantify the impact of each feature.

E) Prediction Function and Result

Question E

```
[ ] y_pred_train = model.predict(X_train)
```

```
from sklearn.metrics import mean_squared_error, r2_score
mse_train = mean_squared_error(y_train, y_pred_train)
r2_train = r2_score(y_train, y_pred_train)
print(f'Mean Squared Error: {mse_train}')
print(f'R-squared: {r2_train}')
```

```
Mean Squared Error: 1.9136573555218528
R-squared: 0.9982049052244079
```

```
[ ] y_pred_test = model.predict(X_test)
```

```
[ ] mse_test = mean_squared_error(y_test, y_pred_test)
    r2_test = r2_score(y_test, y_pred_test)
    print(f'Mean Squared Error: {mse_test}')
    print(f'R-squared: {r2_test}')
```

```
Mean Squared Error: 1.894185677511491
R-squared: 0.9982508614615273
```

What Similar MSE and R^2 Scores Indicate ?

1. **Model Generalization**:

- The similar performance of the model on both seen and unseen data points to the fact that it has learned underlying patterns present in the data on which it was trained.
- This would mean that the model would generalize better to new unseen data, which is usually what most predictive models are after.

Evaluation of Metrics

1. **Mean Squared Error (MSE):**

   Training MSE: 1.91

   Testing MSE: 1.89
   The MSE for the training and testing datasets is nearly identical, showing that the model generalizes well to unseen data. This means the model is neither overfitting nor underfitting.

2. **R-Squared (R^2):**

   Training R^2: 0.9982

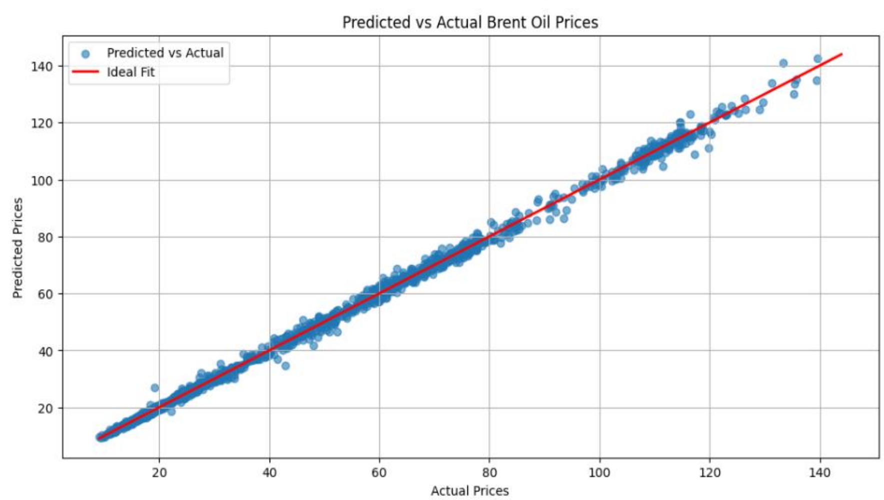
   Testing R^2: 0.9983
   The R^2 values are extremely high and consistent between training and testing datasets, indicating the model explains nearly all the variance in the target variable.

Key Insights

1. **Generalization**:
    - The small difference between training and testing metrics confirms that the model has learned the underlying patterns in the data and is not overfitting.
2. **High Predictive Power**:
    - An R^2 value of ~0.998 shows that the model accounts for 99.82% and 99.83% of the variance in the training and testing data, respectively. This suggests the model is highly accurate.
3. **Low Error**:
    - The low MSE values for both training and testing datasets indicate that the model's predictions are very close to the actual values.

Visualization of Linear Regression Model

```
plt.figure(figsize=(12, 6))
plt.scatter(y_test, y_pred_test, alpha=0.6, label='Predicted vs Actual')
plt.plot([y.min(), y.max()], [y.min(), y.max()], color='red', lw=2, label='Ideal Fit')
plt.title('Predicted vs Actual Brent Oil Prices')
plt.xlabel('Actual Prices')
plt.ylabel('Predicted Prices')
plt.legend()
plt.grid(True)
plt.show()
```



- The points are tightly clustered around the red line, the model performs well, indicating accurate predictions.
- The above scatter suggests that the points lie quite close to the red line, so, generally, the model predicts quite decent results for the Brent oil price. The residuals from the line consist of the errors in the prediction.

F) Calculate the Alpha and Beta values in Python

Question F

```
[ ] #Extract Alpha and Beta Values
    alpha = model.intercept_
    betas = model.coef_
    print(f'Alpha (Intercept): {alpha}')
    print(f'Betas (Coefficients): {betas}')
```

```
Alpha (Intercept): 0.07239408806389491
Betas (Coefficients): [ 1.19505257 -0.19640733]
```

```
[ ] # Formulate the Linear Regression Model
    def regression_equation():
        equation = f"Price = {alpha:.2f} + {betas[0]:.2f} * MA3 + {betas[1]:.2f} * MA9"
        print(f"Linear Regression Equation: {equation}")
        return equation

    regression_equation()
```

```
Linear Regression Equation: Price = 0.07 + 1.20 * MA3 + -0.20 * MA9
'Price = 0.07 + 1.20 * MA3 + -0.20 * MA9'
```

Alpha (α): Baseline price when MA3 and MA9 are 0.

Beta (β1,β2): Measure the contribution of MA3 and MA9 to the Price. They provide insights into the impact and direction of each independent variable on the dependent variable.

- **Dominant Feature**: MA3 has a stronger impact on the price compared to MA9, as its coefficient (1.201.201.20) is much larger in magnitude than −0.20-0.20−0.20.
- **Relationship Direction**:
  - MA3 has a **positive** relationship with price (higher MA3 leads to a higher predicted price).
  - MA9 has a **negative** relationship with price (higher MA9 slightly reduces the predicted price).
- **Baseline Influence**: The intercept (0.070.070.07) is small, indicating the moving averages largely drive price predictions.

REFERENCES

1.Archer, L., Barnes, P., Caffarra, C., Dargay, J., Horsnell, P., Van Der Linde, C., Skeet, I. and Mabro, R. (1990). *OXFORD INSTKIJTE E N E R G Y STUDIES = FOR - The First Oil War: Implications of the Gulf Crisis in the Oil Market*. [online] Available at: https://oxfordenergy.org/wpcms/wp-content/uploads/2010/11/SP1-TheFirstOil-WarImplicationsoftheGulfCrisisintheOilMarket-LArcheretal-1990.pdf.

2.The development of prices and costs during the THE DEVELOPMENT OF PRICES AND COSTS DURING THE 2008-09 RECESSION. (2012). Available at: https://www.ecb.europa.eu/pub/pdf/other/art2_mb201204en_pp71-85en.pdf.

3.Stocker, M., Baffes, J. and Vorisek, D. (2018). *What triggered the oil price plunge of 2014-2016 and why it failed to deliver an economic impetus in eight charts*. [online] World Bank Blogs. Available at: https://blogs.worldbank.org/en/developmenttalk/what-triggered-oil-price-plunge-2014-2016-and-why-it-failed-deliver-economic-impetus-eight-charts.

4.Mabro, R. (1998). *The Oil Price Crisis of 1998*. [online] Available at: https://www.oxfordenergy.org/wpcms/wp-content/uploads/2010/11/SP10-TheOilPriceCrisisof1998-RMabro-1998.pdf.

5.The development of prices and costs during the THE DEVELOPMENT OF PRICES AND COSTS DURING THE 2008-09 RECESSION. (2012). Available at: https://www.ecb.europa.eu/pub/pdf/other/art2_mb201204en_pp71-85en.pdf.

6.Stocker, M., Baffes, J. and Vorisek, D. (2018). *What triggered the oil price plunge of 2014-2016 and why it failed to deliver an economic impetus in eight charts*. [online] World Bank Blogs. Available at: https://blogs.worldbank.org/en/developmenttalk/what-triggered-oil-price-plunge-2014-2016-and-why-it-failed-deliver-economic-impetus-eight-charts.