ML in DH – Assignment 01

Overview:

Crude Oil Prices:

The problem at hand is the volatility and unpredictability of crude oil prices, which has significant economic, social, and environmental impacts on a global scale. Crude oil prices are often influenced by geopolitical tensions, market speculation, production decisions by major oil-producing countries, and global supply and demand factors. The fluctuations in these prices can create instability in both producing and consuming nations, with consequences for economic growth, energy costs, and environmental sustainability.

I have taken a dataset from ***Github.io*** *and the dataset contains 5 attributes:*

1) *Open Crude*
2) *Close Crude*
3) *High Crude*
4) *Low Crude*
5) *Volume Crude*

These attributes is related to crude oil data which includes information like prices, production rates, or other related statistics over a period of time. The source could range from public databases, government agencies, financial reports, or industry publications, and the collection process typically involves gathering raw data, cleaning, and structuring it into a CSV format for analysis.

Link to the dataset: ***https://github.com/swoyam2609/Oil-and-Gas-Price-Analysis-and-Prediction/blob/main/Datasets/crude.csv***

Implemented Code: (Environment – Google Colab)

1) ***Importing libraries of Python*** –
   There are several libraries are used in implementation of python architecture such as NumPy, pandas, seaborn, sci-kit learn, StandardScaler etc which is serving as a domain for finding the accuracy and prediction results to our problem statement.

2) ***Loading the dataset -*** *We have to provide the link to the dataset by uploading the .csv file in the following path for representation of the dataset in the code.*

3) *df.head(), df.drop(['Date','VolumeCrude'],axis=1,inplace=True) -* • `df.drop([...])`*: Removes specified rows or columns.*

   • `['Date', 'VolumeCrude']`*: Specifies the columns to be dropped.*

- `axis=1`: *Indicates that columns (not rows) should be dropped.* `inplace=True`: *Modifies the DataFrame directly instead of returning a new one.*

4) ***df.corr()*** - *.corr() calculates the correlation coefficients between all numeric columns.*
   *The correlation values range from **-1 to 1**:*
   ***1** → Perfect positive correlation (as one variable increases, the other increases).*
   ***0** → No correlation.*
   ***-1** → Perfect negative correlation (as one variable increases, the other decreases).*

5) ***Data Preprocessing** : Data preprocessing is a crucial step to clean, transform, and prepare raw data for analysis or machine learning models. It includes multiple steps:*
   I) *Handling missing values*
   II) *Data cleaning*
   III) *Handling Outliers (Using IQR method)*

6) ***Encoding Categorical Variables :***
   i) *Selecting Categorical Columns with datetime Type*
   ii) *Applying Label Encoding to Datetime Columns*
      a. `label_encoder.fit_transform(data[col])`
         i. *Converts the datetime values into numerical labels.*
      b. *Assigns the transformed values back to the original column.*

7) ***Training, Testing and Splitting the data :***
   - *Convert Date to Ordinal Format (continuous variables coversion)*
   - *Splitting Features (X) and Target (y)*
   - *Splitting X and y into training and testing sets.*
   test_size=0.2: 20% of data is used for testing, 80% for training
   - *Train a Linear Regression Model*

8) ***Predicting the model : (Dummy Variables)***
   - *Making Predictions*
   - *Creating a scatter plot :* ***X-axis (predictions)*** *→ Model's predicted crude oil prices.*
   Y-axis (actual_values) *→ True crude oil prices from y_test.*
   - *Labeling the Plot*
   - *Displaying the Plot*

   Function dummies(x, df) - Converts a categorical variable (x) into dummy variables (one-hot encoding).
   - Removes the original column x
   *- Copying the Original DataFrame : **Each categorical column** is replaced with **dummy variables**.*

9) *Evaluating the model :*
   - The R^2 score (R-squared) measures how well the model's predictions match the actual values.
   - Tests the model on X_test, y_test

10) *Applying Random Forest Regressor ML Algorithm :*
   - *Creating and Training Models*
   - Making Predictions

- Evaluating Model Performance (Calculates RMSE (Root Mean Squared Error) for both model – Lower MSE gives better model.
- Training a New Random Forest Model with 100 Trees

Therefore, the random forest model using Mean squared error and R² score achieves the best MSE= 0.8363049294631263 and R² = **0.998554827111594,** *making it the top performer of Crude Oil price prediction.*

Q) Why I am using Random Forest algorithm?

-> i) Crude oil prices don't follow a simple linear pattern - they fluctuate based on multiple unpredictable factors.
Linear Regression assumes a straight-line relationship between features and price, which may not capture the true complexity.
Random Forest, being a tree-based model, captures non-linear relationships and interactions between variables better.

iii) *Works Well with Large Datasets & Many Features*
iv) *Less Affected by Outliers & Noise :*
   - Oil prices can suddenly spike due to wars, OPEC decisions, or financial crises.
   - Random Forest reduces their impact by averaging predictions from multiple decision trees.
v) *Reduces Overfitting Compared to Decision Tree*
   - Random Forest reduces overfitting by averaging multiple trees, leading to more robust and stable predictions.

Q) Why Random forest and not other algorithms for articulating the problem statement on crude oil price prediction? Mention atleast 5 algorithms which cannot be used (or can be used but has some limitations) for solving the selected problem.

-> Crude oil price prediction is a complex, non-linear problem influenced by various economic, geopolitical, and market factors. Random Forest (RF) is a great choice because it handles non-linearity, is resistant to outliers, and works well with high-dimensional data.

However, other algorithms may not be suitable due to specific limitations.

Here are 5 algorithms that are not ideal (or have significant limitations) for crude oil price prediction:

i) *k-Nearest Neighbors (KNN) (Not Ideal Due to High Dimensionality)*
   - Can be used, but performs poorly with large datasets.
ii) *Support Vector Regression (SVR) (Not Suitable for Large Datasets)*
   - Can be used, but computationally expensive.
   - SVR is computationally slow for large datasets, which is a problem when dealing with financial time series data.

iii) ***Decision Tree Regression (Overfits Too Easily)***

- Can be used, but has serious overfitting issues.

- *Prone to overfitting because a single decision tree memorizes patterns in training data instead of generalizing.*
- *Doesn't handle volatility well, making poor predictions when the market changes*

iv) Naive Bayes Regression (Not Suitable for Continuous Data)

- Completely unsuitable for crude oil price prediction.

Naive Bayes assumes independence between features*, which is unrealistic for crude oil prices (many factors interact).*

Works well for ***classification problems****, not continuous numerical predictions.*