Predicting Oil Spot price using Time Series Data

Richie Zhang RLZ6628@NYU.EDU

Uday Sapra US2055@NYU.EDU

Rohan Sanghrajka RJS9346@NYU.EDU

Milestone 3

https://github.com/RichieZrq/Deep-Learning-Oil-Price-Prediction-Project

1. Research Question

What are the factors that influence oil spot prices, and can we predict these prices using time series analysis combined with external asset classes and macroeconomic indicators?

2. Methodology

2.1 Headlines Sentiment Scores

We wanted to incorporate sentiment within our models. Hence, we decided to combine web-scraping with LLMs. We listed a total of 20 potential news sources to scrape based on their coverage of oil. After experimenting with all 20 sources, we scraped all oil news headlines from Jan. 4th 2021 to Oct. 10th 2024 for 6 sources: Arab News, Financial Times, CNBC, OilPrice.com, Middle East Economic Survey (MEES), and Energy Intelligence. East Economic Survey (MEES), Arab News, and Energy Intelligence. Using the headlines data, we generated sentiment scores for each day for each source using both Mistral NeMo 12B prompting and a pretrained BERT model. The Mistral model is a general-purpose LLM while the BERT was specifically trained on X data for sentiment analysis. As each model outputs a different set of sentiment scores, we analyzed scores and validated them by comparing the original text to their sentiment scores.

2.2 Modeling using PatchTST and Temporal Fusion Transformer

2.2.1 Hypothesis

We chose PatchTST and TFT since both of them are transformer based models but have different statistical natures. PatchTST is an autoregressive model and uses the past of the target to predict its future values. TFT is a multivariate model that incorporates exogenous features (covariates) to model the target. We wanted to compare whether economic and market factors enable us to predict the target of oil stock price better. Our dataset for TFT includes 30 total variables. These include a diverse array of variables across multiple categories:

1. **Economic Variables:** S&P 500, Interest Rates, Oil Futures, VIX, OVX, USO, DXY, and Crude Oil spot prices.
2. **Sentiments:** Extracted using Mistral and BERT to gauge market sentiment and its influence on oil price dynamics.

3. **Engineered Future Features for Decoder:** Includes date variables modeled as functions of Sin and Cos for better seasonality interpretation, along with lagged target variables such as TargetLag(7), TargetLag(30), and TargetLag(365), which help capture temporal dependencies and patterns in the target variable.

2.2.2 Modeling

PatchTST was modelled using FastAI and TSAI Libraries. Temporal Fusion Transformer was modeled using PyTorch-Lightning and Pytorch-Forecasting libraries. Hyperparameter tuning was done on both models using Optuna. For hyperparameter tuning, we fixed the forecast horizon for models at [16, 48, 96] to capture both short-term and long-term forecasting performance and tuned all other hyperparameters to best fit these horizons. Below are all the hyperparameters we tuned. The range was chosen based on the size of our dataset and kept to a minimum to keep the model complexity low.

PatchTST Hyperparameters:

1. **fcst_history**: Length of historical data used for forecasting. Range: 16 to 128
2. **n_heads**: Number of attention heads in the model. Range: 1 to 4
3. **d_model**: Dimensionality of the model's embedding space. Range: 16 to 128
4. **patch_len**: Length of patches created from time-series data. Range: 0 to fcst_history
5. **batch_size**: Number of samples per batch during training. Range: 8 to 128
6. **epochs**: Number of training iterations over the dataset. Range: 5 to 20
7. **n_layers**: Number of encoder layers in the model. Range: 1 to 4
8. **d_ff**: Dimensionality of the feedforward network in the model. Range: 16 to 256
9. **attn_dropout**: Dropout rate applied to attention weights. Range: 0.0 to 0.5
10. **dropout**: Dropout rate applied to linear layers in the model. Range: 0.0 to 0.5
11. **stride**: Step size used when creating patches from the data. Range: 1 to 5
12. **Learning_rate**: Was discovered for each model using a cyclical policy between lr_min and lr_max.

TFT Hyperparameters:

1. **epochs**: Number of training epochs for the model. Range: 5 to 15
2. **learning_rate**: Step size for updating model weights during optimization. Range: Log-uniform between 10^{-4} and 10^{-1}.
3. **hidden_size**: Dimensionality of the mode's embedding. Range: 8 to 16
4. **lstm_layers**: Number of LSTM layers in the model architecture. Range: 1 to 4
5. **attention_head_size**: Number of attention heads in the model. Range: 1 to 4
6. **dropout**: Dropout rate to prevent overfitting by randomly zeroing out neurons. Range: 0.1 to 0.5
7. **loss**: Loss function used to evaluate model performance. Options: ['SMAPE', 'MAPE', 'MAE', 'RMSE', 'MSE'].
8. **gradient_clip_val**: Maximum gradient value to clip during backpropagation to avoid exploding gradients. Range: 0.1 to 1
9. **batch_size**: Number of samples processed in a single batch. Range: 32 to 128
10. **max_encoder_length**: Forecast Horizon Range: 32 to 128

2.2.3 Validation

Validation was done on a hold-out test-set of proportion 0.2. on. Each model needs a minimum forecasting history (hyperparameter) to predict the fixed horizon [16, 48, 96]. Thus, the test-set was converted into windows of the forecasting history (x_test) and subsequent horizon (y_test). Then, the model was inference on each window to predict the horizon. We took the mean of MAE and SMAPE loss across these windows. MAE and SMAPE were chosen due to interpretability. Since we are dealing with oil price and absolute measure of loss (MAE) and a percentage point measure (SMAPE) were important. The total number of windows inference on is given by:

len(test) - (forecasting history + horizon) - 1

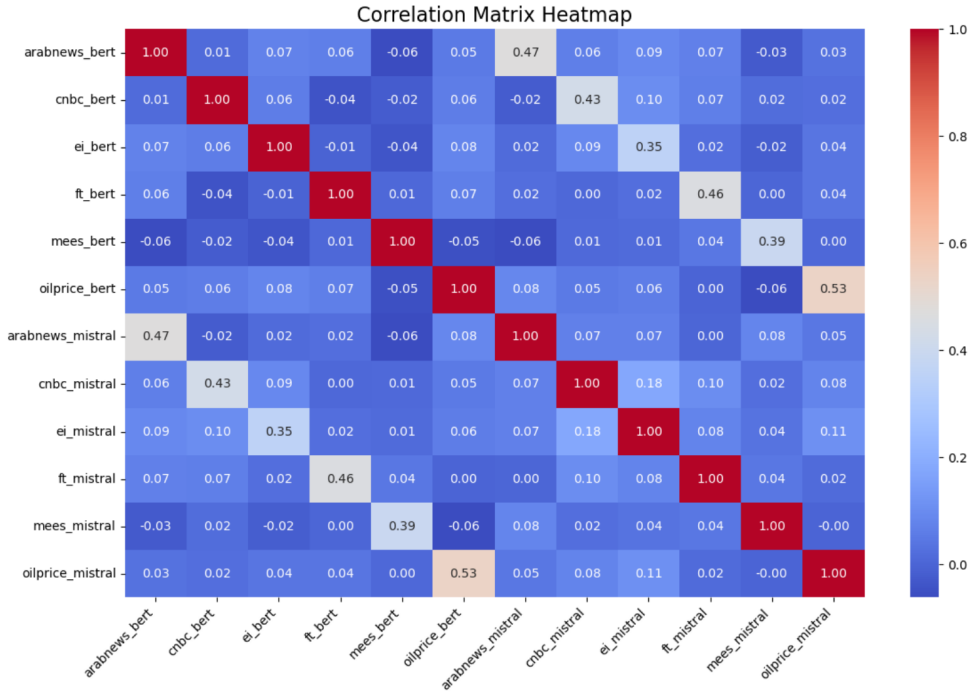
2.3 Benchmarking

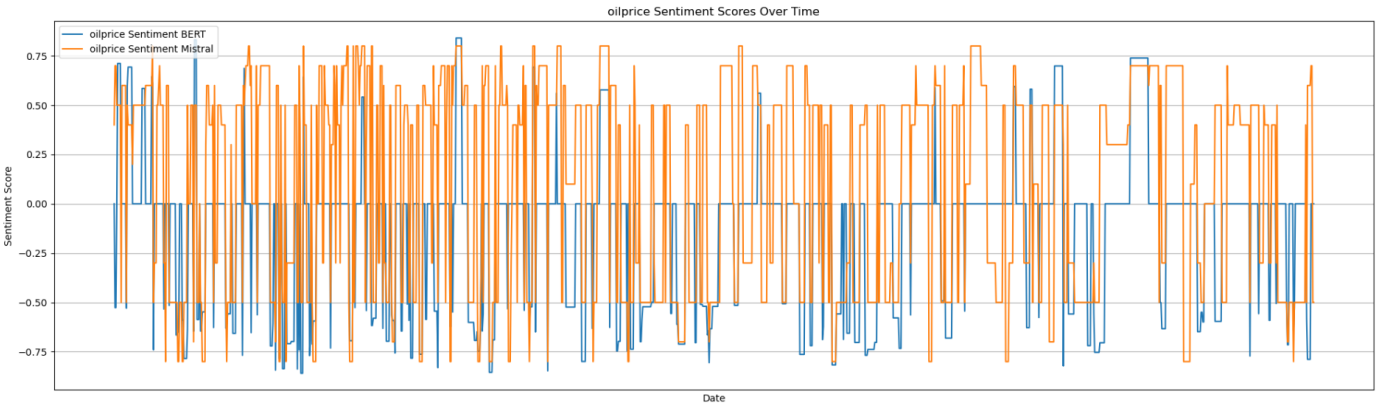
ARIMA and SARIMAX were used as statistical benchmarks. ARIMA is an autoregressive model while SARIMAX is a multivariate model that allows for exogenous variables with known future values. Hence, both statistical models could not incorporate our economic or sentiment variables but SARIMAX was able to use our engineered lags and sin/cos seasonality features. ARIMA was modeled using Nixtla's AutoArima model and SARIMAX was modeled using StatsForecast from StatsModels. They were benchmarked on the same forecast horizons and validation metrics as the transformer models.

3. Results

3.1 Sentiment Analysis

After computing sentiment scores using Mistral and BERT, we found the correlation between sentiment scores of the two methods and created a visualization of all sentiment scores to understand them:





3.2 Modeling

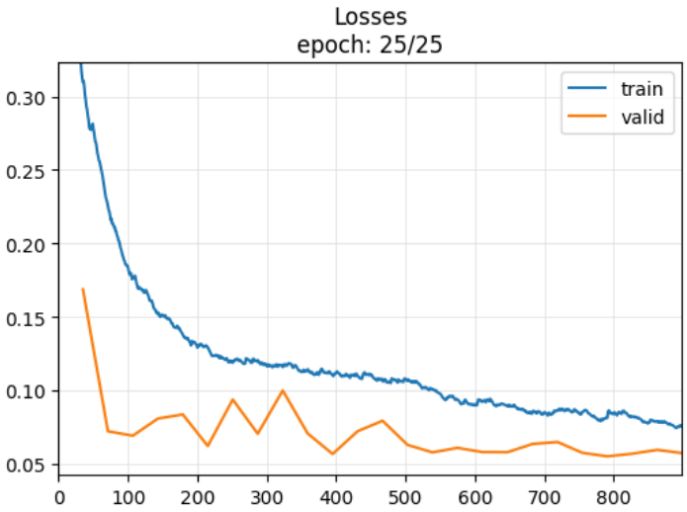
Below is a snapshot of the full model performance.

Horizon	Model	SMAPE (%)	MAE (USD)
4*16 Steps	ARIMA	8.99	6.90
	SARIMAX	6.82	5.30
	PatchTST	**3.15**	**2.47**
	TFT	3.56	2.80
4*48 Steps	ARIMA	9.72	7.48
	SARIMAX	6.722	5.26
	PatchTST	**4.86**	**3.84**
	TFT	7.36	6.96
4*96 Steps	ARIMA	10.45	8.07
	SARIMAX	**6.4**	**5.23**
	PatchTST	6.65	5.24
	TFT	10.37	8.55

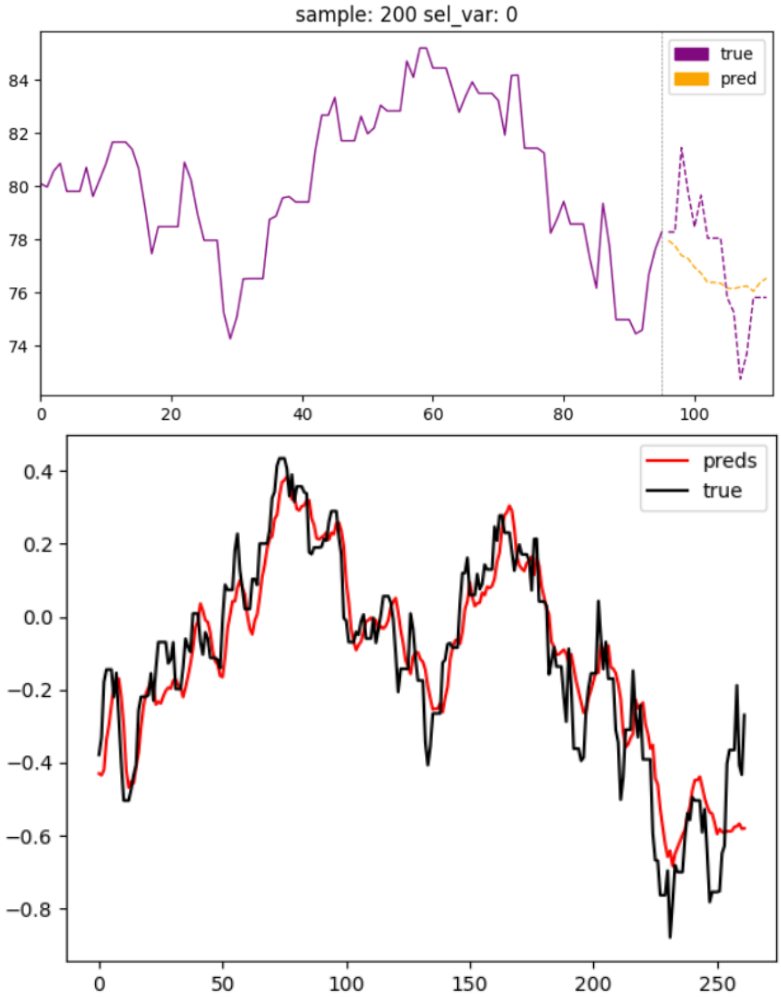
As can be seen, for short to medium term forecasts PatchTST was the best model across all metrics. However, for longer-term forecasts, SARIMAX was the best model. ARIMA did not fit, and the TFT performed poorly compared to both PatchTST and SARIMAX in the medium to long-term forecasts.

3.3 Training

Below is a snapshot of the overall best model's training curves - PatchTST at prediction length 16. The training was very stable and we used a cyclical learning rate policy for training



The first figure below presents the prediction of the model in the last window in our test set.



The second plot above captures the first prediction of the model for every window. We inference the model on window t, take the first prediction (n) from its forecast of the length horizon. Then we inference the model on window t+1 (which contains the true value for n), and we take the first prediction of the model (n+1). We did this for all windows. We want to note that the plot is not representative of the true performance of the model. Instead, it conveys the model's ability to predict from its last input time-step. As can be seen, our model is very sensitive to the last-time step in its input or each successive new piece of information, which exhibits a good fit.

4. Analysis

4.1 Sentiment Analysis

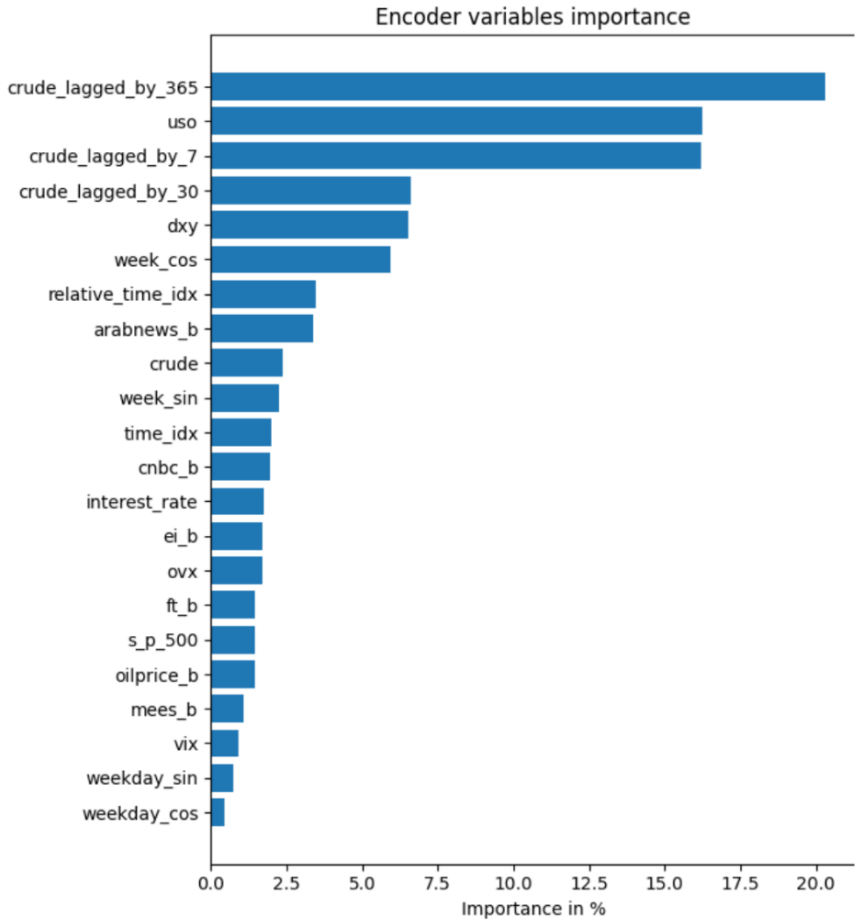
After comparing the sentiment scores from BERT and Mistral, we found that different news sources have low correlation in their sentiment scores. The sentiment scores of the same website from the two methods have a correlation of around 0.45. Mistral generally produced more positive sentiment scores compared to BERT. This is especially clear in the oilprice.com scores shown above. When comparing the scores to the actual text, we found that BERT generally outperformed Mistral in terms of generating accurate sentiment scores that capture the overall sentiment of the headlines. Therefore, we only used BERT sentiment scores in our modeling, as it represents news headlines' sentiments more accurately.

4.2 Modeling

The impressive performance of PatchTST and SARIMAX overarchingly tells us that for oil, autoregressive features are the most important. The addition of economic indicators and sentiments leading to worse performance indicates that economic variables are not very important in predicting oil price. In addition, the quality of the sentiments could be poor leading to inconclusive results for the TFT.

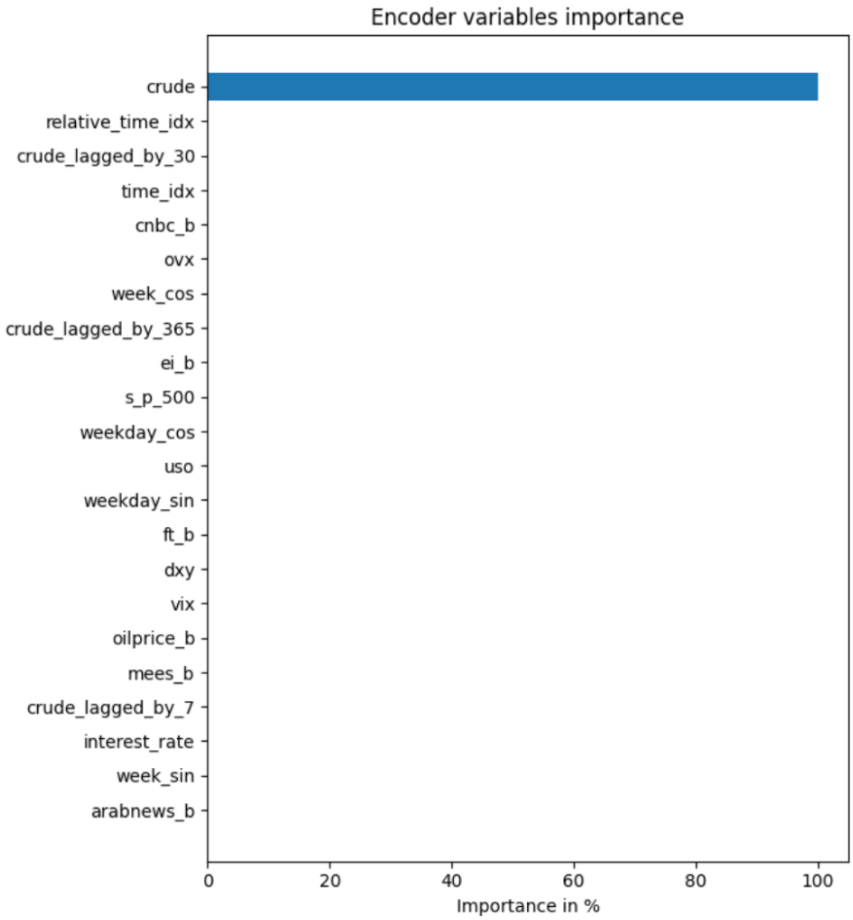
We also studied the feature importance of the TFT model at all horizon lengths to discover the findings below. These feature importances are the attention weights from the encoder part of the transformer across each feature.

1. Short-Term Model (16 Future Days)



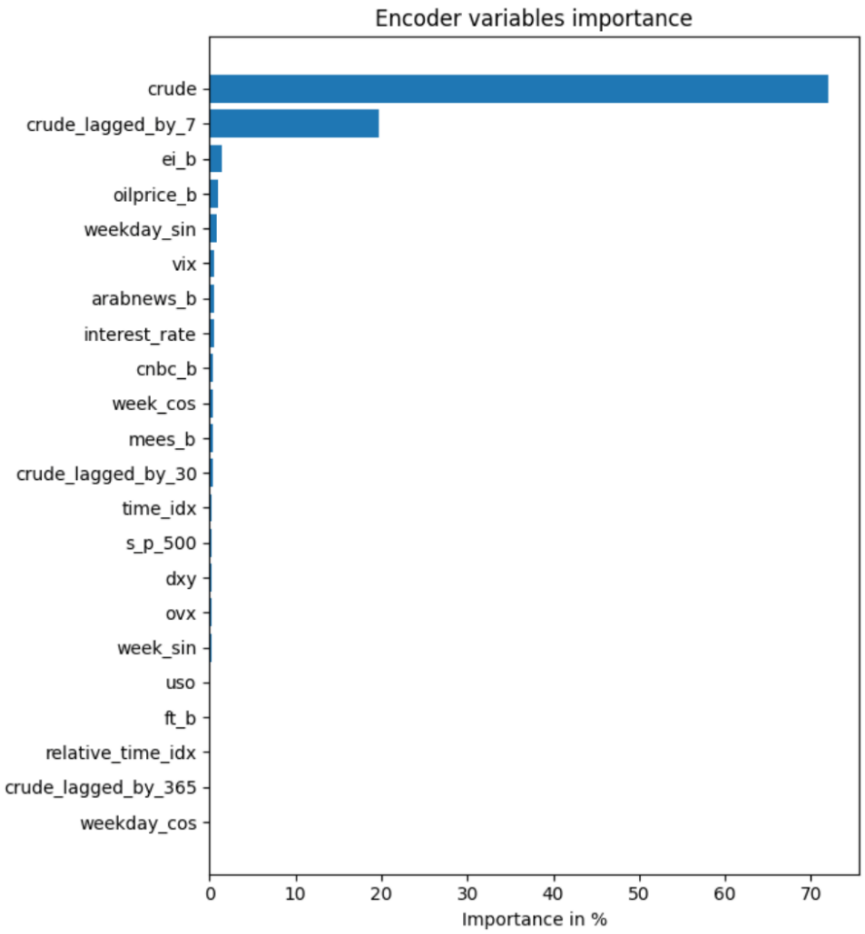
As can be seen, the short-term models give the most importance to oil lagged by 365 followed by US oil prices and then other news and economic features. This suggests that for short-term oil price modeling - economic features and sentiment do hold relevance. However, this relevance declines as we predict oil price for longer terms.

2. Medium Term Model (48 Future Days)



As can be seen, for medium-term oil prediction, the model disregarded all features but the autoregressive oil feature in the encoder itself.

3. Long Term Model (96 Future Days)



Similarly, for the long-term TFT model, historical crude price was the most important predictor of future crude price and the model shrank all other features.

Conclusively, the model weights indicate that for long-term oil price modeling, autoregressive features are more important, while for short-term, exogenous features like economic indicators assist modeling. This also explains why the TFT performed the best for short-term forecasts where sentiment and economic factors could be utilized for modeling. But as we increased the forecast horizon, the TFT struggled as the exogenous features could not give additional predictive power. This was combined with additional model complexity that led to high-variance in model performance - putting it below the other two models.

5. Conclusions

5.1 Sentiment Analysis

We could only scrape 6 out of the 20 sources we selected due to scraping difficulties and time limitations. This could potentially cause selection bias in news headlines. Another consideration is that when concatenating all headlines from the same day, the overall sentiment of each headline is mixed with others, potentially confusing the model that we use. Instead of generating scores for a long text with all headlines concatenated together, a next step could be to generate sentiment scores for each headline, then aggregating all headlines for a day by finding the average of the sentiment scores.

5.2 Modeling

Oil seems to be a very complex variable to predict. Even with high-model complexity and feature-sets, our multivariate transformer could not outperform autoregressive models. This could be due to poor sentiment scores or a very complex feature set, hence, the next steps would be to work on better sentiment modeling, and reduce the feature-set through better feature engineering, The TFT also shower a very high degree of variance, every model, even with similar hyperparameters, converged to different solutions. The problem persisted even when we reduced the model complexity and feature set. This could also indicate a lack of data, which was 800 samples for training, Hence, in the future, we also want to expand on data.

We were impressed by PatchTST's ability to outperform SARIMAX with a reduced feature set. While SARIMAX used exogenous variables of lags and seasonality, PatchTST was solely autoregressive. Hence, we plan to further analyse model performance to understand what contributes to better performance.

The conclusion, thus for modeling, is that oil is best predicted autoregressively with low model complexity. Economic factors and sentiments add further noise to the modeling process, leading to models with extremely high variance.

6. Work Flow

October-November

1. **Data Collection and Preprocessing**: Rohan
2. **Feature Engineering**
   (a) Hypothesis (Finance & Commodities Domain Knowledge), Correlation Testing, and Feature Selection - Rohan
   (b) Data Mining (news headlines scraping) - Richie
   (c) News Sentiment Scores Generation - Richie and Uday
   (d) Implementation - Uday
3. **Model Prototyping, Development, and Validation (TFT and PatchTST)**: Uday

December

4. **Benchmarking Model Prototyping, Development, and Testing (ARIMA and Sarima)**: Rohan and Uday
5. **Sentiment Scores Analysis**: Richie
6. **Model Prototyping, Development, and Validation (TFT and PatchTST)**: Uday
7. **Results and Conclusions (Visualizations)** Rohan, Uday, and Richie
8. **Presentation and Report** - Rohan, Uday, and Richie

References

1. Cardiff NLP. twitter-roberta-base-sentiment-latest. Hugging Face, https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest.
2. Lim, Bryan, et al. "Temporal Fusion Transformers for Interpretable Multi-Horizon Time Series Forecasting." *arXiv.Org*, 27 Sept. 2020, arxiv.org/abs/1912.09363.
3. Nie, Yuqi, et al. "A Time Series Is Worth 64 Words: Long-Term Forecasting with Transformers." *arXiv.Org*, 5 Mar. 2023, arxiv.org/abs/2211.14730.
4. Nixtla. "AutoARIMA." *StatsForecast Documentation*, Nixtla, https://nixtlaverse.nixtla.io/statsforecast/src/core/models.html#autoarima.
5. StatsForecast: Nixtla. "SARIMAX." *StatsForecast Documentation*, Nixtla, https://nixtlaverse.nixtla.io/statsforecast/src/core/models.html#sarimax.