Taiyo.ai Assignment

...

By Amitava Roy
3rd Year UG IIT Kharagpur

Email ID: roy.rishi2002@gmail.com

Problem Statement 1

Time-Series Analysis:

Use any daily time series from Investing.com or similar source with a strong sample of covariates. Target commodities price like: Oil, Natural Gas, Resin, or Metal Prices. ○ Please make sure to get an extensive list of feature space, think through structural other external factors.

○ Option 1. Feature Importance. Dynamic Time Warping and/or XGBoost/Shapley Value hybrid model approach to quantify which factors influence the target positively or negative ■ Keep the analysis focused on the feature selection and feature importance aspects

○ Option 2. LSTM derivatives on day ahead prediction with confidence bounds ● How would you improve and present your results with more time and resources ● Related thinking and planning in a short report

Time Series Forecasting on Uranium Prices

Dataset Downloaded from here

Have a look at Python Notebook here

Our Dataset contains prices of various commodities and indexes across a time period 1980-2016.

We have extracted the data related to Uranium, All Commodity Price Index and Metals Price Index for the Assignment.

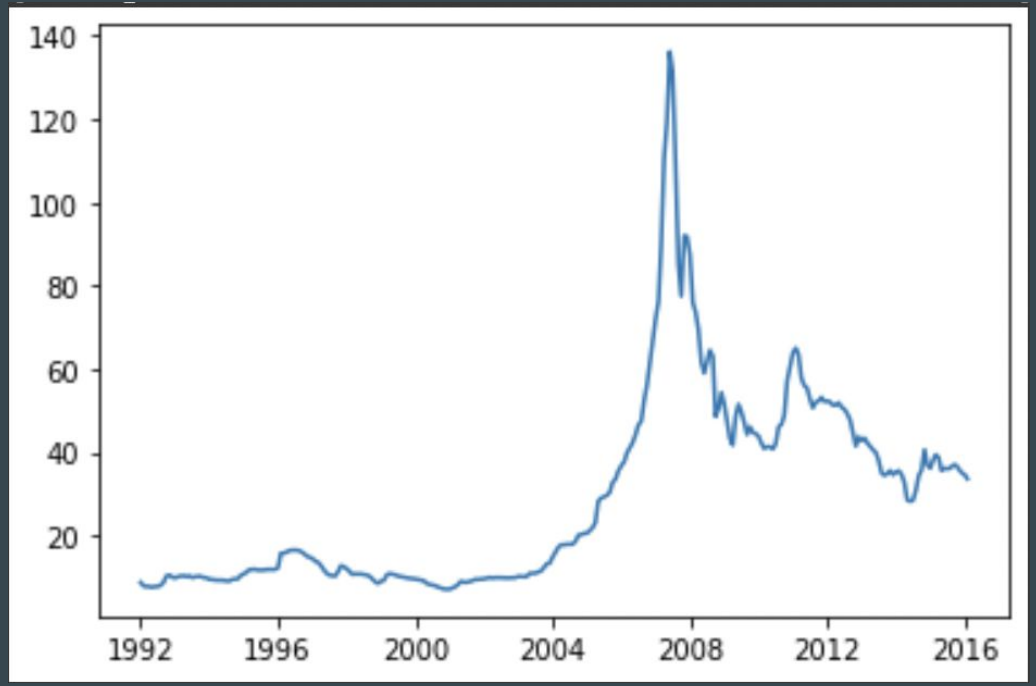
Having a Look at Our Extracted DataSet's First Few Rows

Date	All Commodity Price Index	Metals Price Index	Uranium
1992-01-01	52.876718	58.451712	8.75
1992-02-01	53.020784	60.607861	7.90
1992-03-01	52.736939	61.039659	7.75
1992-04-01	54.270214	62.008882	7.75
1992-05-01	55.577815	62.094826	7.55

Some Statistics on our Data

	All Commodity Price Index	Metals Price Index	Uranium
count	290.000000	290.000000	290.000000
mean	100.438645	110.578716	28.765148
std	51.429999	58.826096	23.668199
min	41.971597	48.018317	7.100000
25%	58.176006	60.206977	10.100000
50%	72.484482	79.264397	16.500000
75%	140.798155	169.616004	42.632812
max	219.899500	256.236049	136.222222

Variation of Uranium Prices over the Time-Period



Now, we take a window_size of 3 ie. We split the data into three columns. The columns contain prices of Uranium on (t-3)th month, (t-2)th month and (t-1)th month if we want to estimate the price of Uranium on t-th month. Therefore we get our independent variable matrix with 3 columns and n rows. Our dependent variable matrix has only 1 column ie. prices of Uranium on t-th day.

Now, we design our LSTM Model.

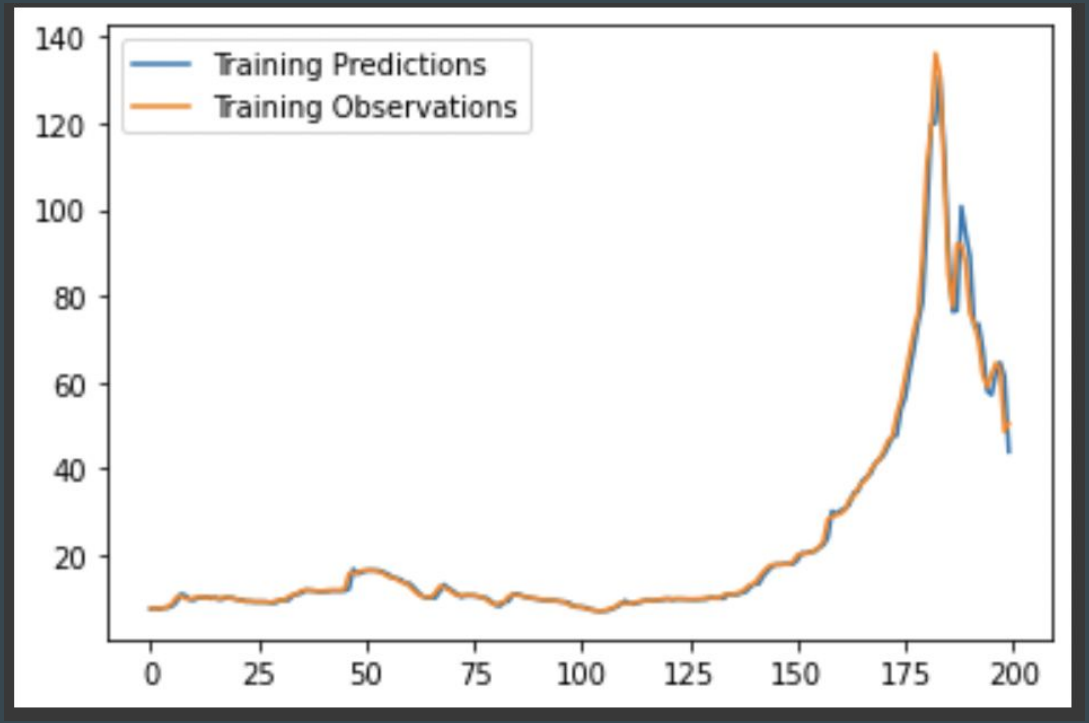
Univariate LSTM Model Designed

```
model = Sequential([layers.Input((3, 1)),
                    layers.LSTM(64),
                    layers.Dense(32, activation='relu'),
                    layers.Dense(32, activation='relu'),
                    layers.Dense(1)])

model.compile(loss=MeanSquaredError(),
              optimizer=Adam(learning_rate=0.001),
              metrics=['mean_absolute_error'])

cp1 = ModelCheckpoint('model1/', save_best_only=True)

model.fit(X_train1, y_train1, validation_data=(X_val1, y_val1), epochs=100, callbacks=[cp1])
```

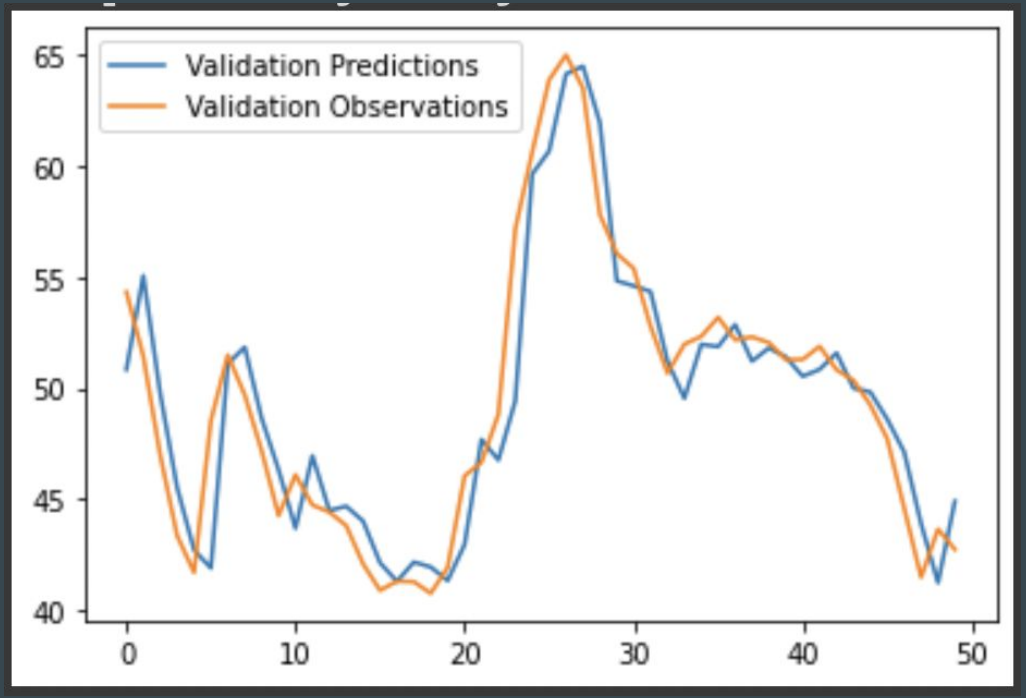
Summary of Model Trained

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 lstm_6 (LSTM)                (None, 64)                16896

 dense_24 (Dense)             (None, 32)                2080

 dense_25 (Dense)             (None, 32)                1056

 dense_26 (Dense)             (None, 1)                 33

=================================================================
Total params: 20,065
Trainable params: 20,065
Non-trainable params: 0

_________________________________________________________________
None
```

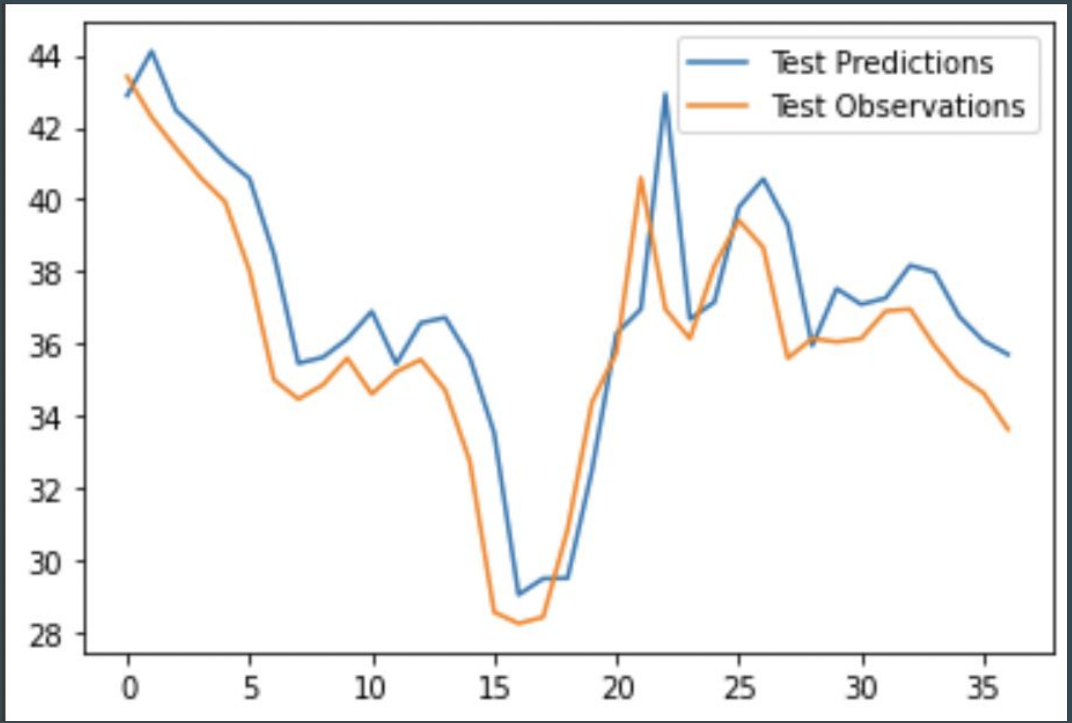
Our Model's Predictions on Training Data v Actual Training Observations



Our Model's Predictions on Validation Data v Actual Validation Observations



Our Model's Predictions on Test Data v Actual Test Observations



Evaluation of Our Model

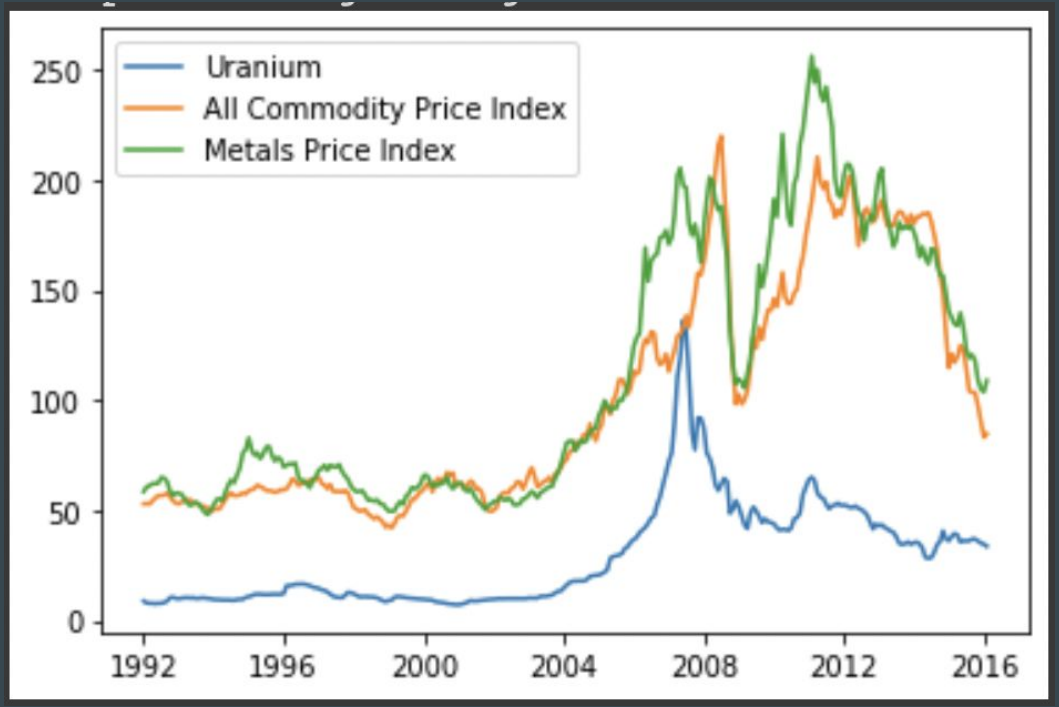
```
[134] scores = model1.evaluate(X_test1, y_test1)
```

```
2/2 [==============================] - 1s 8ms/step - loss: 4.4566 - mean_absolute_error: 1.6692
```

Conclusion

Therefore, we have observed that our Univariate LSTM Model has been quite successful in predicting the Uranium Prices to a good extent. Now, we would try to fit in 2 more features, ie. All Commodity Price Index and Metals Price Index over the same time-period. Let us see how the model changes after the inclusion of these factors and whether Uranium Prices is somewhat correlated to them.

Variation of all 3 Features over the Time-Period



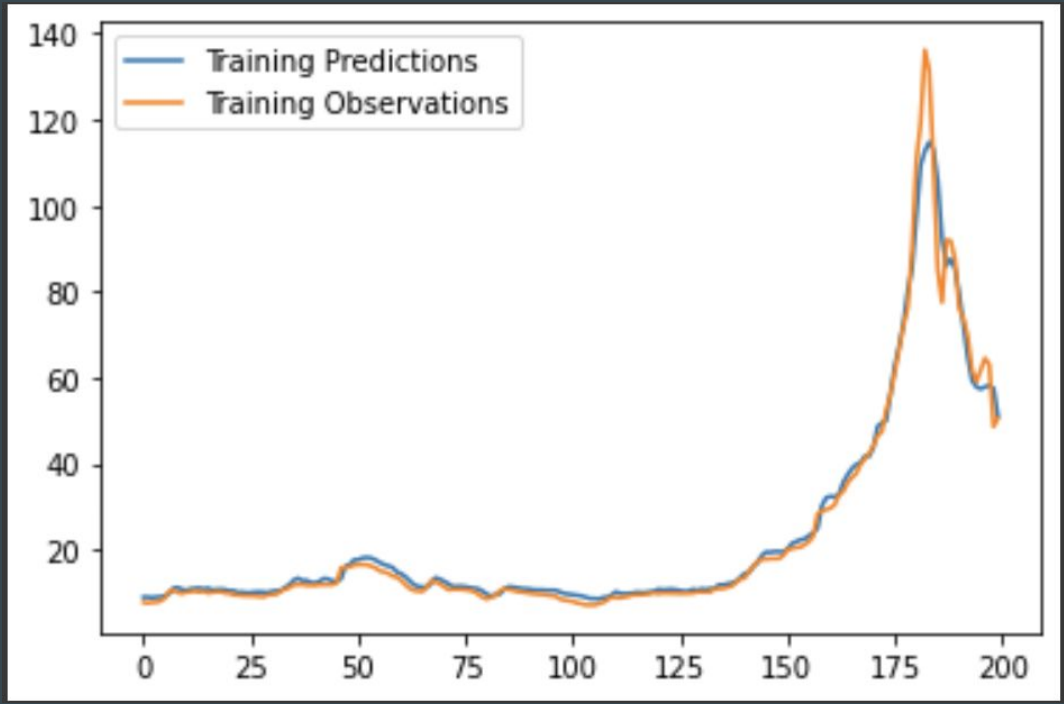
Multivariate LSTM Model Designed

```
model2 = Sequential([layers.Input((3, 3)),
                     layers.LSTM(64),
                     layers.Dense(32, activation='relu'),
                     layers.Dense(32, activation='relu'),
                     layers.Dense(1)])

model2.compile(loss=MeanSquaredError(),
               optimizer=Adam(learning_rate=0.001),
               metrics=['mean_absolute_error'])

cp2 = ModelCheckpoint('model2/', save_best_only=True)

model2.fit(X_train2, y_train2, validation_data=(X_val2, y_val2), epochs=100, callbacks=[cp2])
```

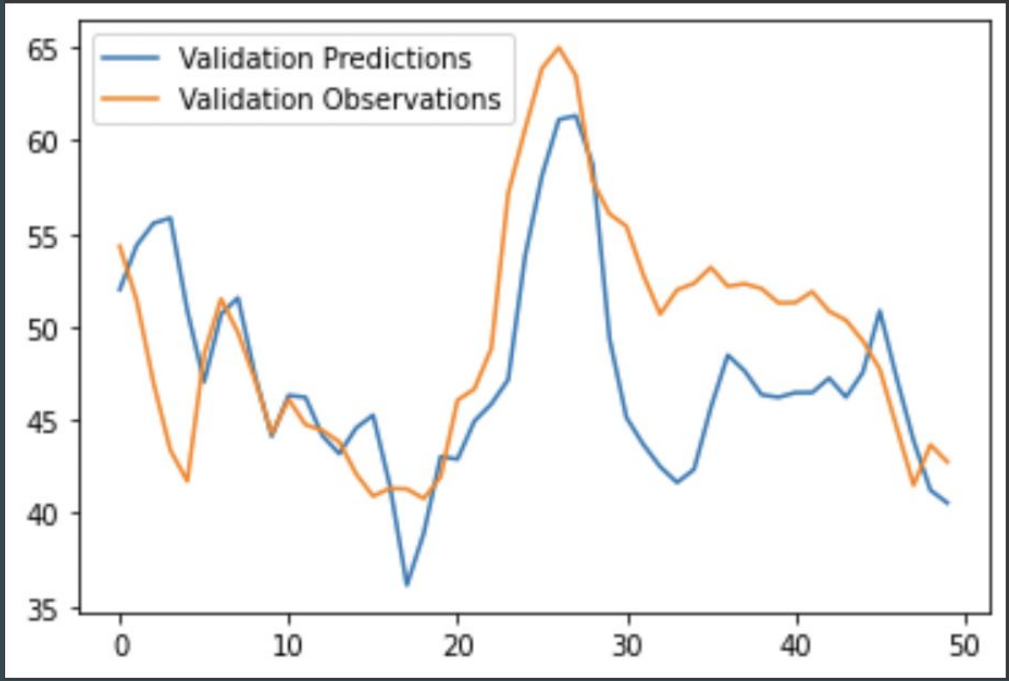
Summary of Model Trained

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 lstm_7 (LSTM)                (None, 64)                17408

 dense_27 (Dense)             (None, 32)                2080

 dense_28 (Dense)             (None, 32)                1056

 dense_29 (Dense)             (None, 1)                 33

=================================================================
Total params: 20,577
Trainable params: 20,577
Non-trainable params: 0
_________________________________________________________________
None
```

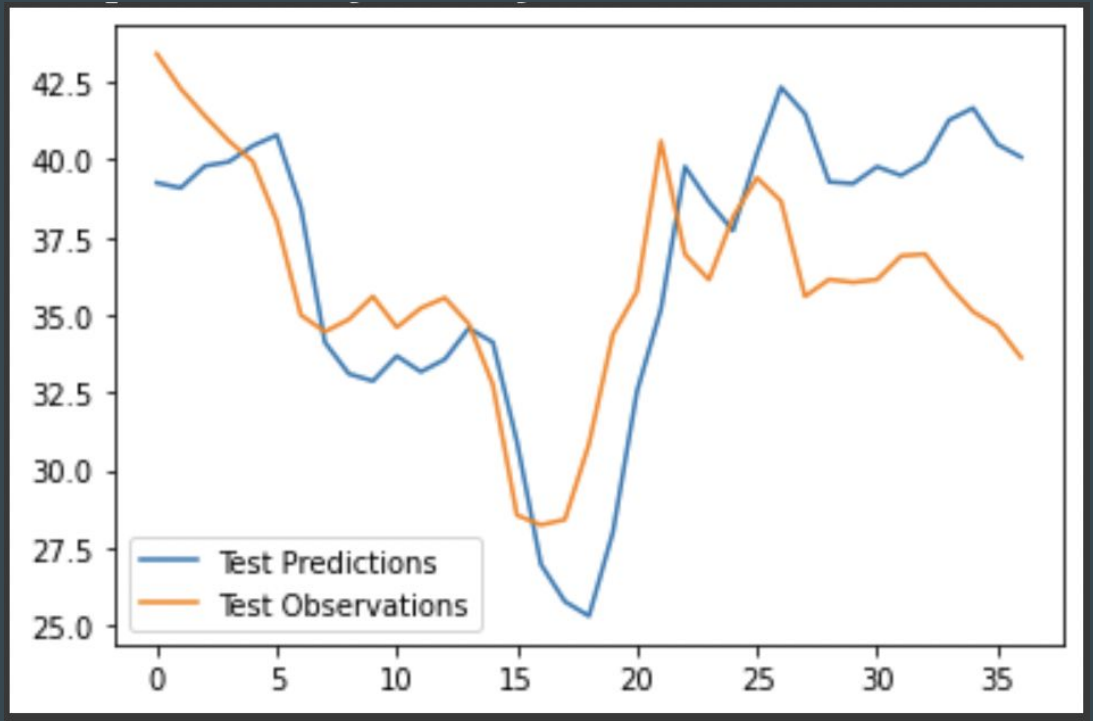
Model's Predictions on Training Data v Actual Training Observations



Model's Predictions on Validation Data v Actual Validation Observations



Model's Prediction on Test Data v Actual Test Observations



Evaluation of Our Model

```
[149] scores = model2.evaluate(X_test2,y_test2)

      2/2 [==============================] - 1s 8ms/step - loss: 12.3549 - mean_absolute_error: 2.9834
```

Conclusion

We therefore observed that our multivariate model also estimates the prices of Uranium to a good extent. However, it is not as good as the univariate model which can be observed by the mean absolute error which is higher for the multivariate case.

I believe if we include more data and add some more features to the multivariate model that are relevant to Uranium Prices, we can generate a better model that will estimate better.

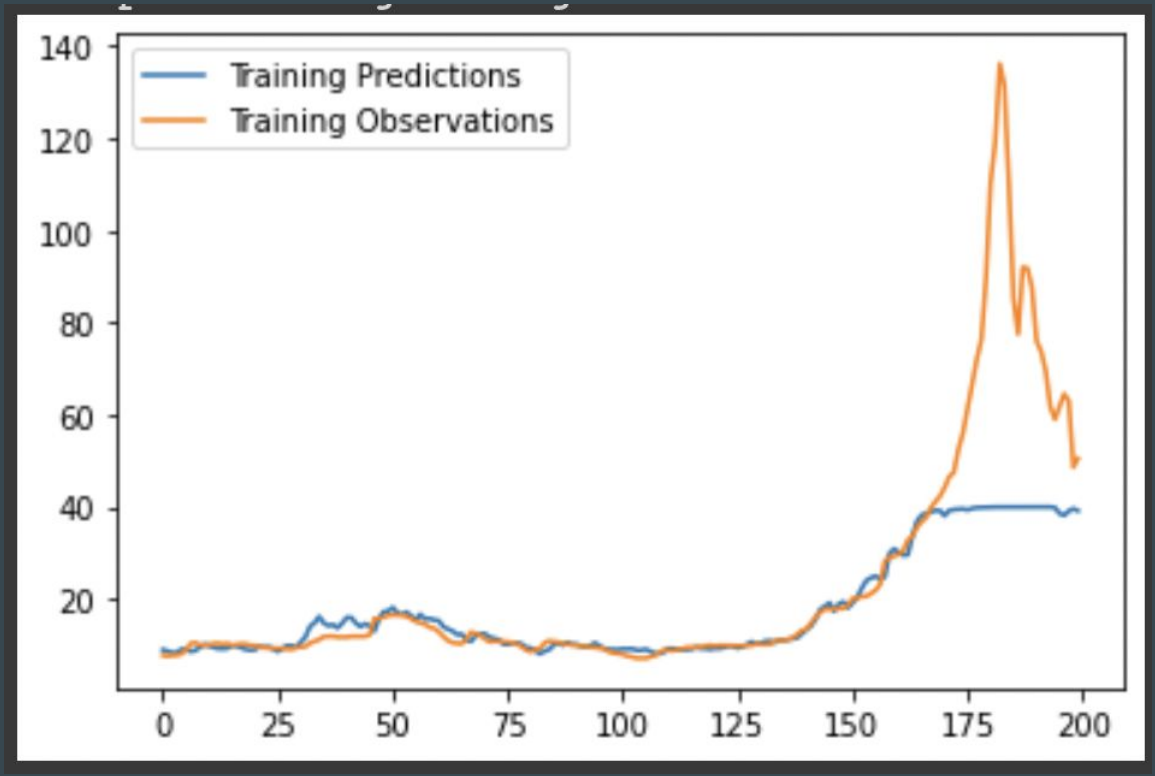
Designing a Multivariate GRU Model

```
model3 = Sequential([layers.Input((3, 3)),
                     layers.GRU(64),
                     layers.Dense(32, activation='relu'),
                     layers.Dense(32, activation='relu'),
                     layers.Dense(1)])
X_train3, y_train3, X_val3, y_val3, X_test3, y_test3 = X_train2, y_train2, X_val2, y_val2, X_test2, y_test2
cp3 = ModelCheckpoint('model3/', save_best_only=True)
model3.compile(loss=MeanSquaredError(), optimizer=Adam(learning_rate=0.0001), metrics=[RootMeanSquaredError()])


model3.fit(X_train3, y_train3, validation_data=(X_val3, y_val3), epochs=100, callbacks=[cp3])
```

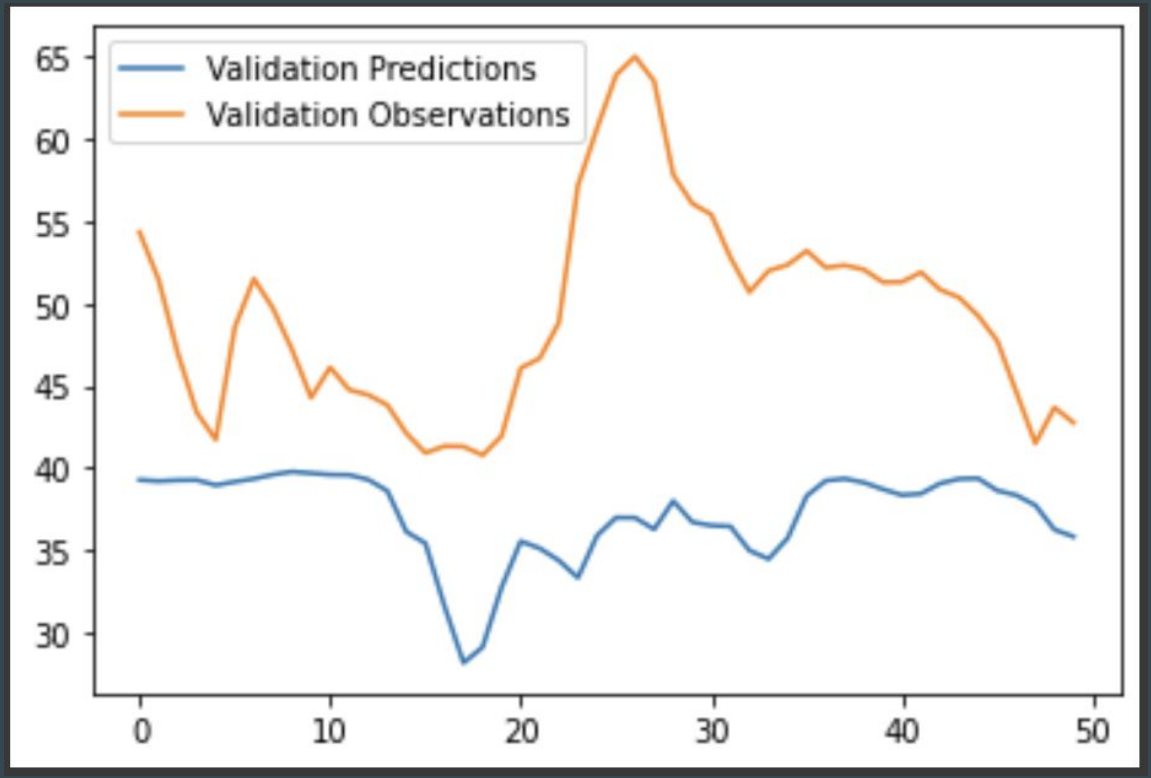
Summary of Model Trained

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 gru_2 (GRU)                  (None, 64)                13248

 dense_30 (Dense)             (None, 32)                2080

 dense_31 (Dense)             (None, 32)                1056

 dense_32 (Dense)             (None, 1)                 33

=================================================================
Total params: 16,417
Trainable params: 16,417
Non-trainable params: 0
_________________________________________________________________
None
```

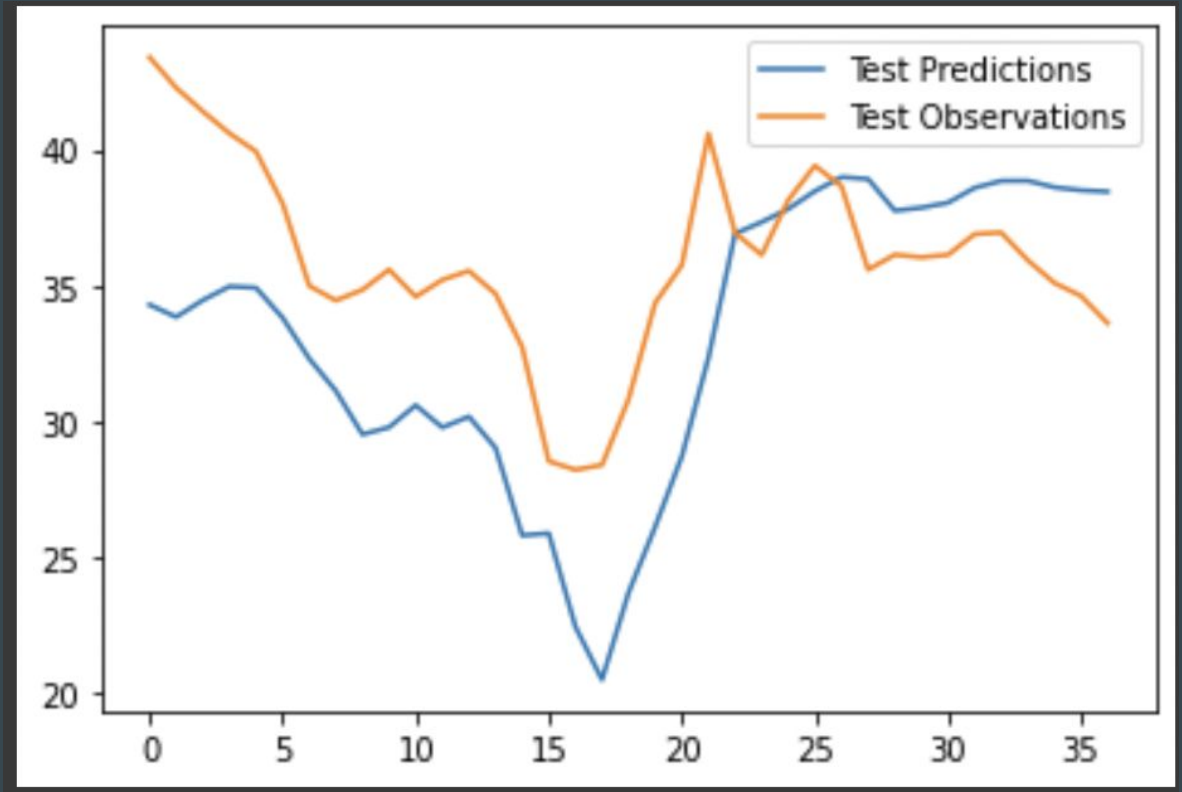
Model's Prediction on Training Data v Actual Training Observations



Model's Prediction on Validation data v Actual Validation Observations



Model's Prediction on Test Data v Actual Test Observations



Conclusion

From the graphs itself, we can observe that the GRU model does not fit as good as the LSTM models. The GRU model however is faster compared to the LSTM Models since the number of parameters to train is lesser. However, accuracy wise it is far inferior.

The best model as per our observations is the Univariate LSTM Model. However, I believe the multivariate model can surpass this if we feed in more data and few more features relevant to Uranium prices like Country data, Mining Data, etc.

Problem Statement 2

Natural Language Understanding

Use World Bank Projects dataset

Option 1. Tagging for Keyword Extraction or Named Entity Recognition (NER) using models such as YAKE, BERT-derived models, spaCy, or Google NLP API ■ Specifically identify either “sector” or “sub-sector” or entities like Government Agency, Company Name, Contractors, Investor, or unit measurements such as cost per square kilometer.

Option 2. Binary Classifier ○ Using the status variable build a binary classifier to predict the probability whether a project will be “closed” or “canceled/distressed”

Binary Classifier using NLP and Logistic Regression

The dataset has been downloaded from here.

Have a look at Python Notebook here.

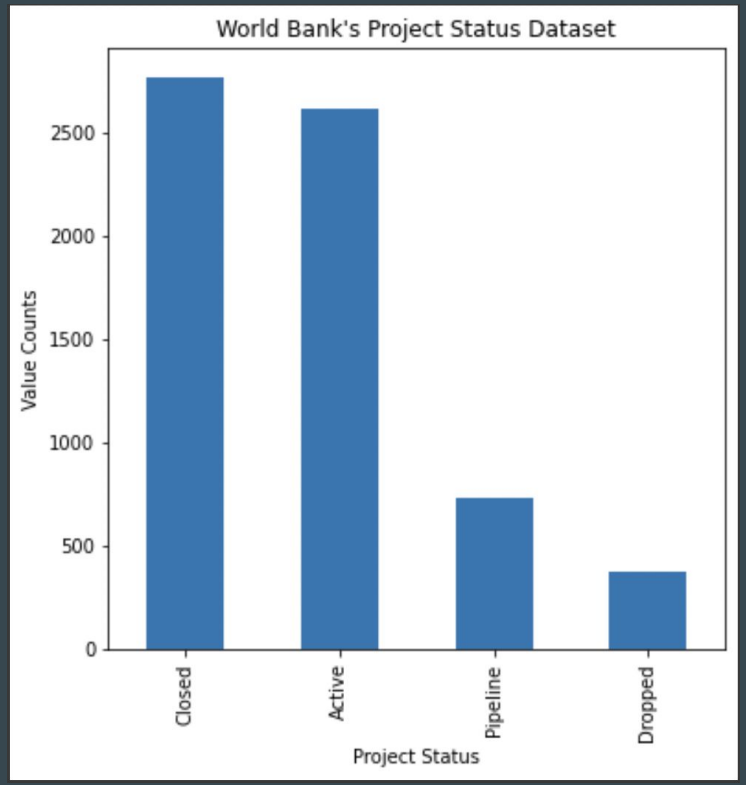
We have used the World Bank Dataset that contains various information related to projects that have been implemented across countries over a large time period.

We have tried to build a classifier that predicts whether a project will have a status “Closed” or not based upon its Project Development Objective data.

Dataset after Extracting a Few Features

	Region	Country	Project Status	Project Name	Project Development Objective	Implementing Agency
2	Africa East	Republic of Angola	Active	Third Angola Growth and Inclusion Development ...	The development objective is to support the Go...	Ministry of Finance of the Republic of Angola
3	Africa East	Republic of Madagascar	Active	Connecting Madagascar for Inclusive Growth	The Project Development Objective is to improv...	Road Agency
4	Middle East and North Africa	Kingdom of Morocco	Active	Resilient and Sustainable Water in Agriculture	The project development objectives (PDO) are t...	Ministry of Agriculture - Directorate of Irrig...

Displaying Counts of each Project Status Category



Fraction of Projects that have Status = ‘Closed’

```
#Fraction of projects that have "Closed" status ie. Label 1

len(df[df['Label'] == 1])/len(df)
```

```
0.42637091805298827
```

Therefore, even if we make a prediction without a model and say that a project has a status-“Closed”, we would have a 42.6% chance of being correct!

Preprocessing

After this, we have extracted the Project Development Objective Column and processed the text data.

We have removed any stopwords, unwanted characters and converted words having conjunctions into their corresponding phrase without conjunctions.

We have then tokenized words and also used the WordNetLemmatizer.

We have then splitted the data into 70% training data and 30% testing data.

After this, we have applied the Bag of Words model and TF-IDF model on our training and testing data.

Designing Logistic Regression Model

```
def simple_logistic_classify(X_tr, y_tr, X_test, y_test, description, _C=1.0):
    model = LogisticRegression(C=_C).fit(X_tr, y_tr)
    score = model.score(X_test, y_test)
    print('Test Score with', description, 'features', score)
    return model
```

Comparing Bag of Words and TF-IDF

Following output was obtained:

```
Test Score with bow features 0.6935318275154004

Test Score with tf-idf features 0.6981519507186859
```

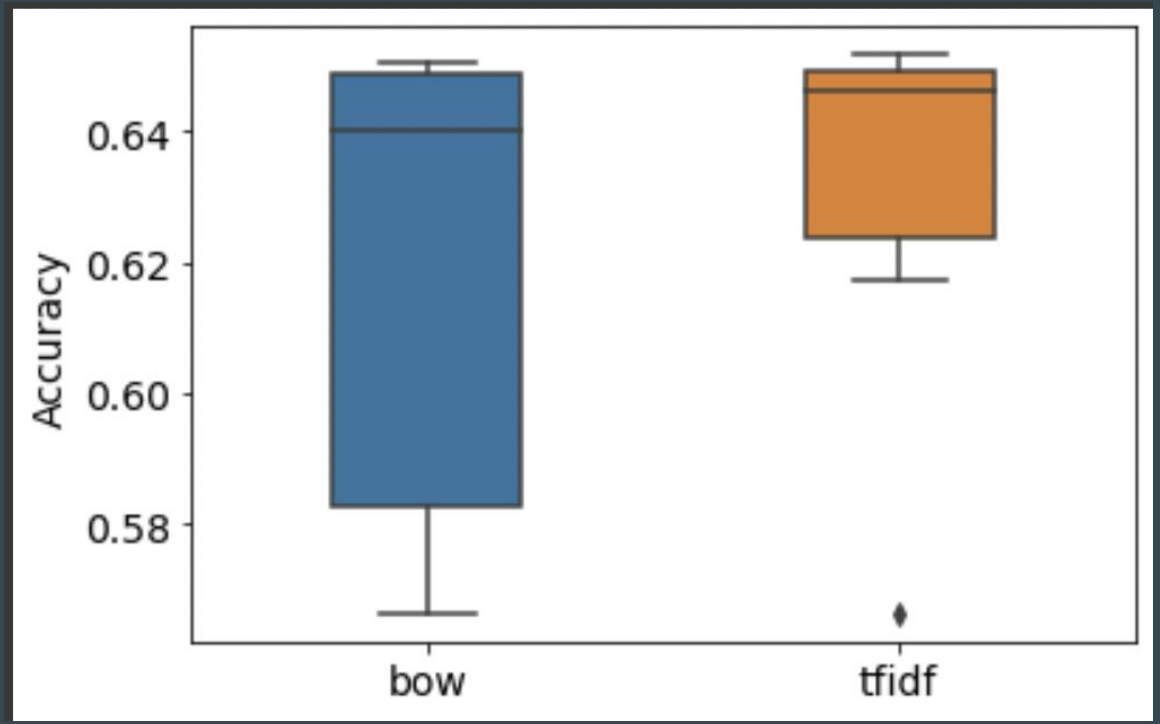
This was followed by Tuning the Parameters to find the model with best accuracy.

After that, again a comparison was made between Bag of Words and TF-IDF.

Following output was obtained:

```
Test Score with bow features 0.7027720739219713
Test Score with tf-idf features 0.715605749486653
```

Accuracy Comparison Graph



Conclusion

Therefore, we have observed that the TF-IDF model is slightly superior than the Bag of Words Model. However both these models provide a far superior accuracy rate compared to the 42.6% probability initially.

I believe the model can be improved if we add more features like Country Data, Cost of Project Data and Data on Money Grants and Implementing Agencies. Due to the time constraint of the assignment, I could not incorporate these features.

THANK YOU