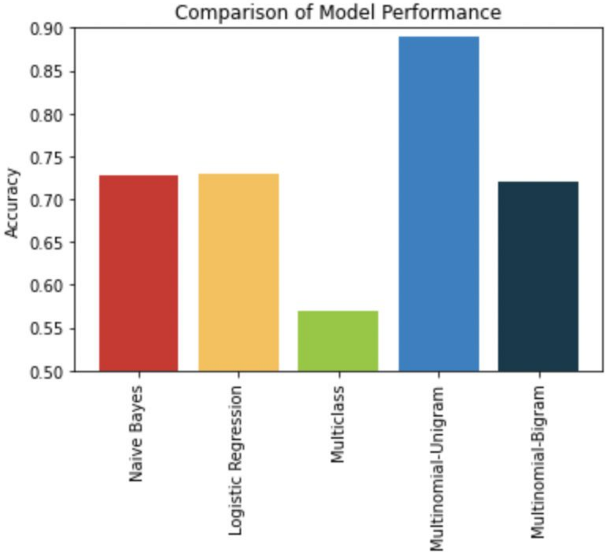
Project 1 Summary

- The project involves **text mining of Reddit comments on ChatGPT**, utilizing **topic modeling** to uncover themes and **VADER SentimentIntensityAnalyzer** to generate sentiment scores. The sentiment scores are then used as ground truth to test various text classification methods such as **Logistic regression and Naive Bayes.**
- The dataset contains **52,414 rows and 4 columns**, and it includes comments from four different subreddits:
  - r/ChatGPT
  - r/technology
  - r/Futurology
  - r/dataisbeautiful.
- We ran Topic Modeling to uncover hidden topics : **QnA, Use case, Human like, Advanced Search Engine**
- We concluded that the **Multinomial Logistic model** produced the **best results for sentiment analysis** on the ChatGPT subreddit.



Gold Price Prediction

Mehta,
Sakshi

Padam,
Simran

Shah,
Jenika

Shankar,
Vani

TABLE OF CONTENTS

01

INTRODUCTION

02

TIME SERIES DATA

03

DATA CLEANING, PREPROCESSING & FEATURE SELECTION

04

REGRESSION

05

TIME SERIES

06

CONCLUSION & FUTURE EXTENSIONS

Introduction

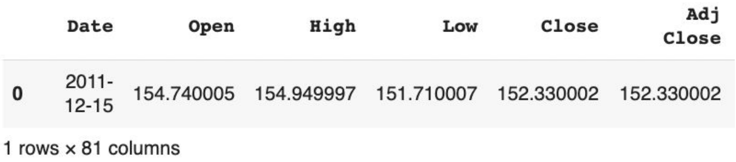
- **Objective:** predict gold prices using machine learning techniques
- **Context:**
  - Gold is an important asset that is widely traded across the world
  - Powerful asset that is seen as an hedge against inflation, “safety asset”
- **Models**
  - Naive Approach: Regression
    - Basic Linear Regression, LASSO (Dimensionality Reduction), XG Boost Regression
    - Evaluated model performance using root mean squared error (RMSE), and R^2
  - Time Series: Analysis and Models
    - Performed time series analysis
    - Implemented auto-regressive models, such as ARIMA
- **Goal:** Allow investors to be able model future gold prices to provide valuable insights and better portfolios

Github link: https://github.com/simran-padam/GoldPriceForecasting.git

Time Series Data

- Data was collected from November 18th, 2011 to December 31st, 2018 and included daily trading asset prices (**1718 x 80**)
- Data for attributes such as Oil Price, Standard and Poor's (S&P) 500 index, Dow Jones Index US Bond rates (10 years), Euro USD exchange rates, prices of precious metals Silver and Platinum and other metals such as Palladium and Rhodium, prices of US Dollar Index, Eldorado Gold Corporation and Gold Miners ETF were gathered.
- The historical data of Gold ETF fetched from Yahoo finance has 7 columns: Date, Open, High, Low, Close, Adjusted Close, and Volume.
- **Open** is the **outcome variable** which is the value you have to predict.

Sample Output:



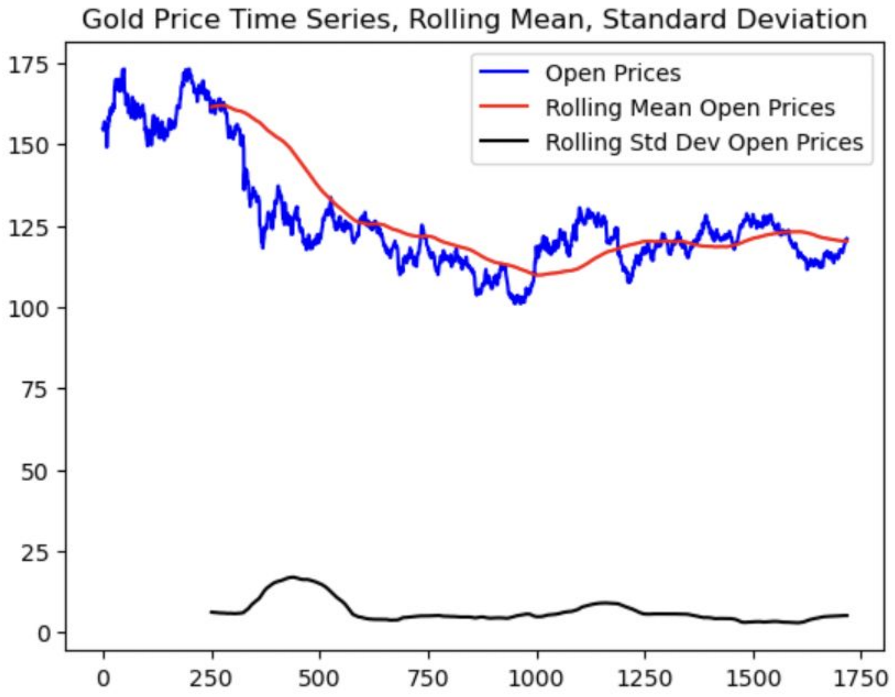
	Date	Open	High	Low	Close	Adj Close
0	2011-12-15	154.740005	154.949997	151.710007	152.330002	152.330002

1 rows × 81 columns

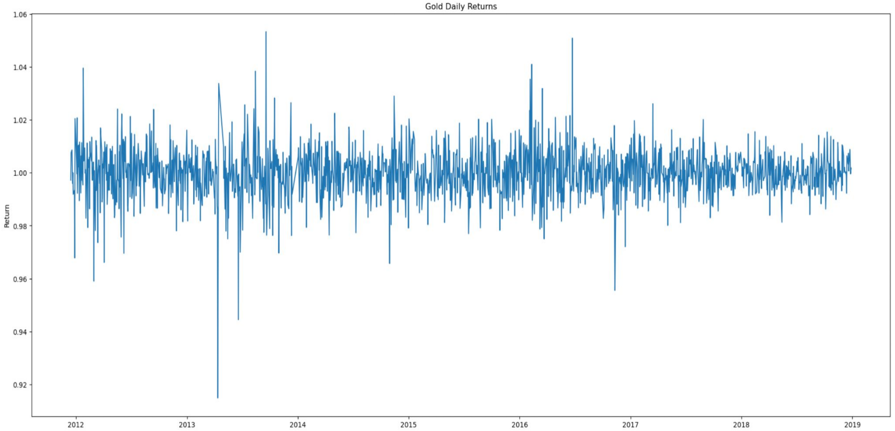
Data Cleaning

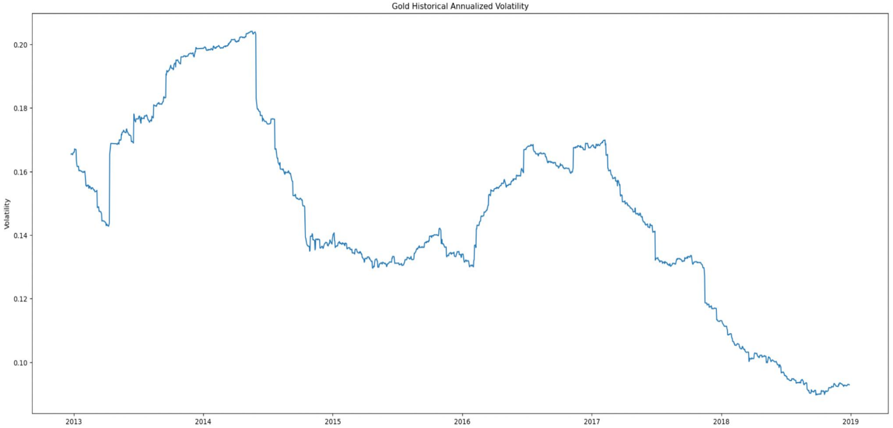
Before fitting models, we chose to exclude data from end 2011-mid 2013

- Gold saw sharp drop in prices during this time period, marking the end of a 12 year bull run. Due to strong economic data and mitigated fears of inflation, investors began to invest in riskier assets, such as stocks. Dropping this section of data allows us to fit + predict more accurately the prices of gold during "normal" times



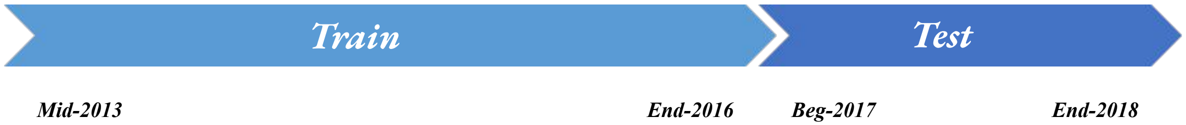
Gold Daily Returns & Volatility





Data Preprocessing

- There were inconsistencies in variable names, variable renaming was performed by lower casing and by adding underscores. For eg: “ad close” as “adj_close”
- Convert data column to date time format
- Our data, unlike traditional daily time series data, only has data on active trading days, which excludes weekends and holidays. Because we only had 9 holiday days in a year, we left the data as is
- For outlier treatment, variables were floored at 5% and capped at 95%
- Time series dataset was split into train and test sets in the following way:



Train

Test

Mid-2013

End-2016

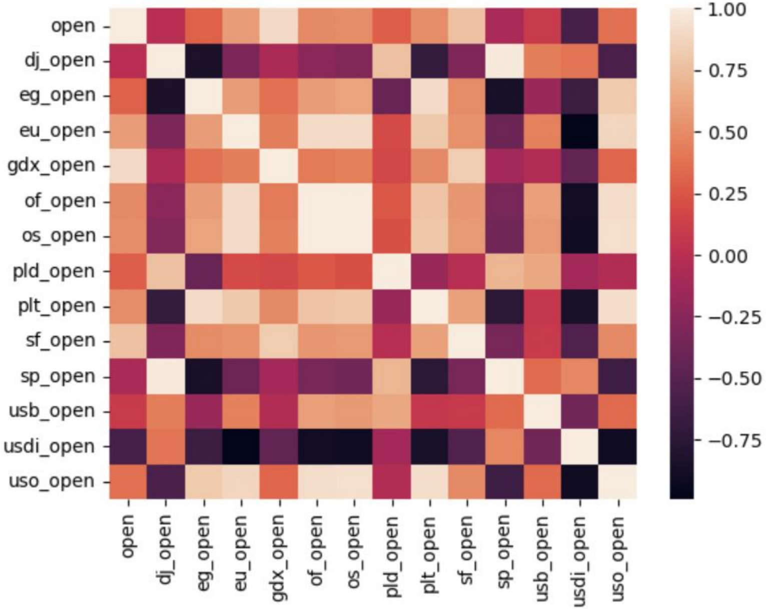
Beg-2017

End-2018

- After splitting the data, variables were standardised using sklearn *StandardScaler()*

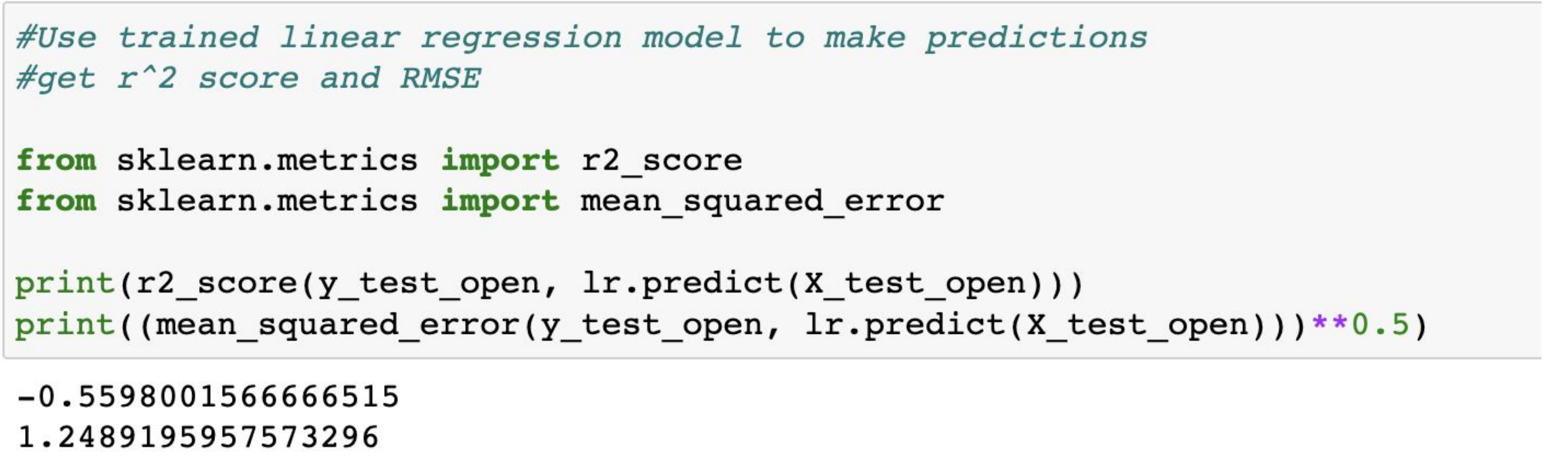
Feature Selection

- **Redundancy** : For each index such as S&P 500, DJI, we have Open, Close, High, Low, Adjusted close. For brevity and to remove high collinearity, we have used open variables for each index. Rho_price was also removed due to missing values
- **Correlation Matrix**: Generated a correlation matrix heat map to visualize the pairwise correlations between the *open* variables in our dataset.
- **LASSO**: Using LASSO, we performed feature selection and identified the following variables as the most important predictors of the outcome variable: *dj_open, gdx_open, plt_open, sf_open*
- **XGBoost*:*** alpha parameter was tuned in the Grid Search to perform L1 regularisation



Base Model : Regression (w/o dimensionality reduction)

R^2 and MSE of test set:

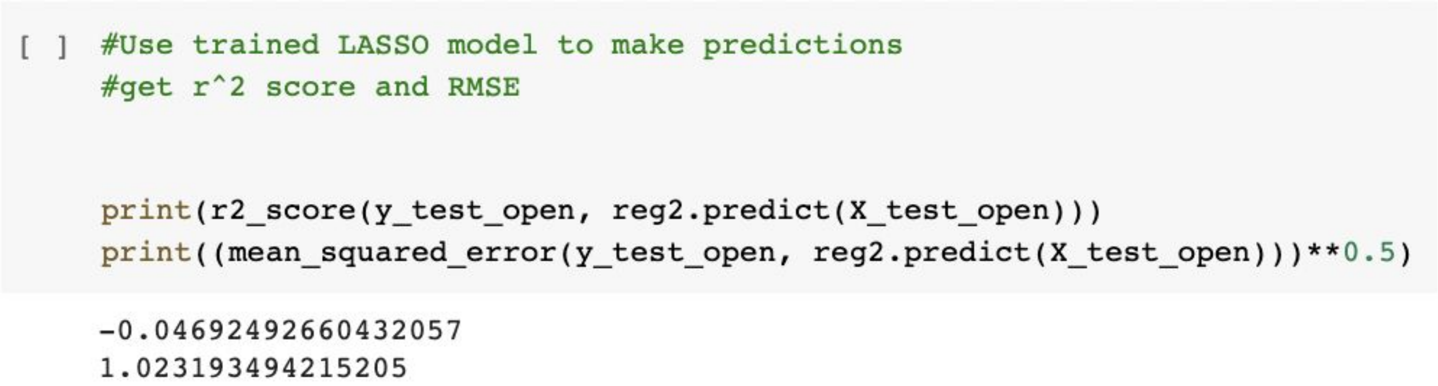


```
#Use trained linear regression model to make predictions
#get r^2 score and RMSE

from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error

print(r2_score(y_test_open, lr.predict(X_test_open)))
print((mean_squared_error(y_test_open, lr.predict(X_test_open)))**0.5)
```

```
-0.5598001566666515
1.2489195957573296
```

Regression with Dimensionality reduction (Lasso)

R^2 and MSE of test set after dimensionality reduction: *dj_open, gdx_open, plt_open, sf_open*

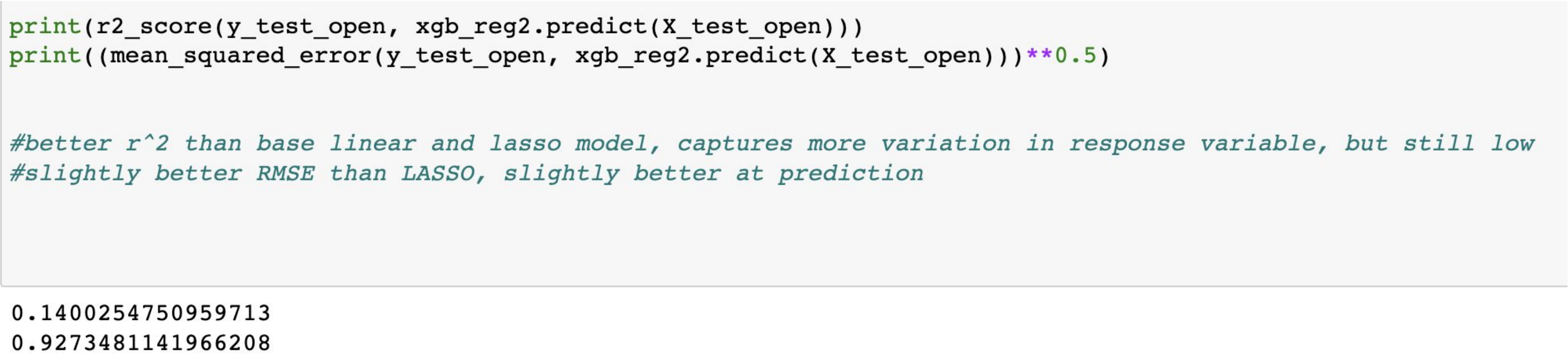


```
[ ]  #Use trained LASSO model to make predictions
     #get r^2 score and RMSE


     print(r2_score(y_test_open, reg2.predict(X_test_open)))
     print((mean_squared_error(y_test_open, reg2.predict(X_test_open)))**0.5)
```

```
-0.04692492660432057
1.023193494215205
```

XGBoost Regression

R^2 and MSE of test set in XGB Regression:



```
print(r2_score(y_test_open, xgb_reg2.predict(X_test_open)))
print((mean_squared_error(y_test_open, xgb_reg2.predict(X_test_open)))**0.5)


#better r^2 than base linear and lasso model, captures more variation in response variable, but still low
#slightly better RMSE than LASSO, slightly better at prediction
```

```
0.1400254750959713
0.9273481141966208
```

TIME SERIES: Basic Auto Regressive Model (ARIMA)

- ARIMA - Autoregressive Integrated Moving Average Model
  - Assumes that previous values of target variable can be used to predict future values
- Hyperparameters (p,d,q)
  - p: is the order of AR term, which depends on past values/lags to predict future values
    - Determines the amount of past values to consider to predict single future value
  - d: number of differencing, taking difference between current and previous values
    - Method to make data stationary, important assumption in many time series models
  - q: the size of the moving average window
    - Accounts for past forecast errors to predict future values (MA term)
- Four variables were used in the analysis : *dj_open, gdx_open, plt_open, sf_open*

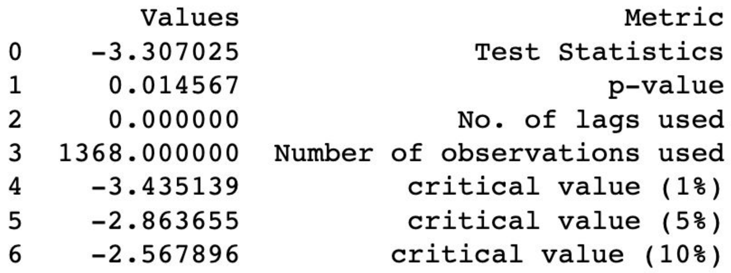
TIME SERIES: Analysis

Stationarity

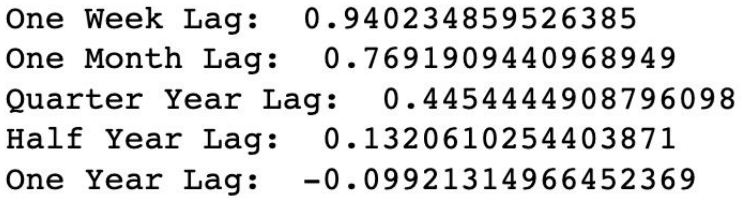
- Presence of means the way a time series changes is constant (no trends)
- Dickey Fuller Hypothesis Test Output →
    - P-value > 0.01 fail to reject null hypothesis → non - stationationary data

Autocorrelation

- Measures how correlated the times series data at a given point with past values, basis for autoregressive time series models
- Lag correlations →
    - Strong correlation for more frequent time windows



```
        Values                       Metric
0    -3.307025              Test Statistics
1     0.014567                      p-value
2     0.000000              No. of lags used
3  1368.000000  Number of observations used
4    -3.435139           critical value (1%)
5    -2.863655           critical value (5%)
6    -2.567896          critical value (10%)
```

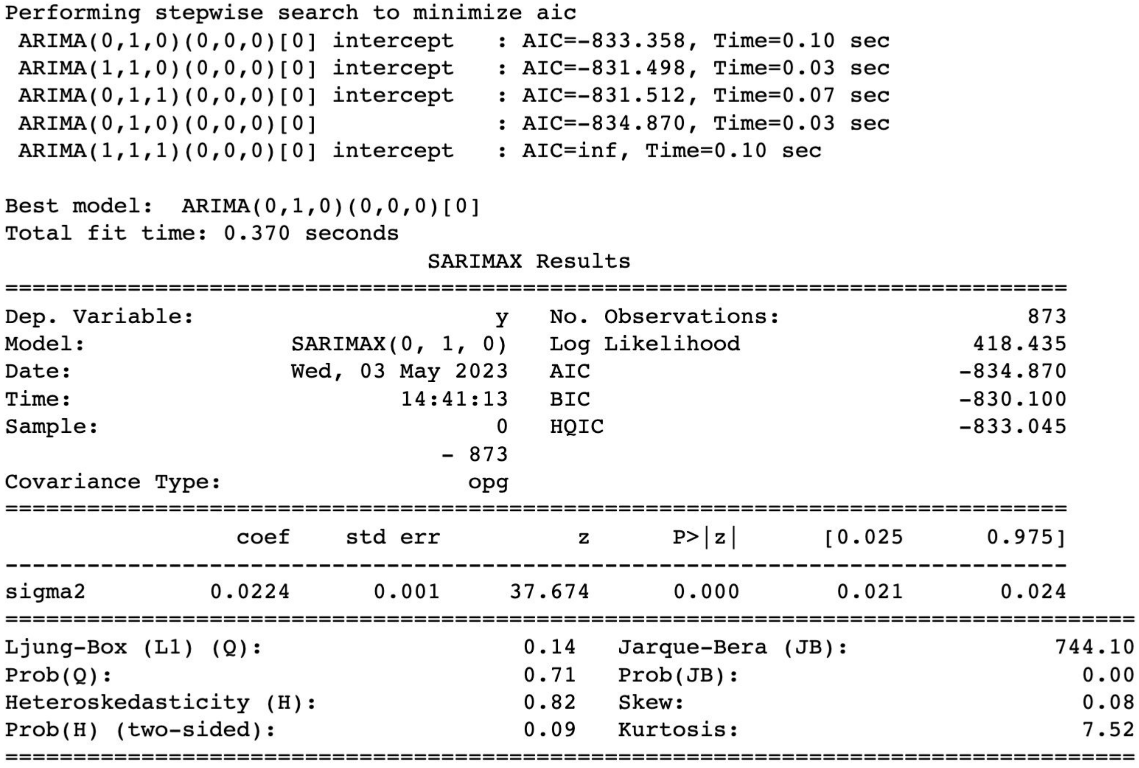


```
One Week Lag:  0.940234859526385
One Month Lag:  0.7691909440968949
Quarter Year Lag:  0.4454444908796098
Half Year Lag:  0.1320610254403871
One Year Lag:  -0.09921314966452369
```

TIME SERIES: ARIMA Hyperparameter Tuning

Auto.ARIMA output

- Performed optimization by customizing auto.arima algorithm →
- Look for smallest AIC
  - Estimation of prediction error
  - Want smallest AIC for best model
- Chose
  - ARIMA(0,1,0) due to smallest AIC



```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=-833.358, Time=0.10 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=-831.498, Time=0.03 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=-831.512, Time=0.07 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=-834.870, Time=0.03 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.10 sec

Best model:  ARIMA(0,1,0)(0,0,0)[0]
Total fit time: 0.370 seconds
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  873
Model:               SARIMAX(0, 1, 0)   Log Likelihood                 418.435
Date:                Wed, 03 May 2023   AIC                           -834.870
Time:                        14:41:13   BIC                           -830.100
Sample:                             0   HQIC                          -833.045
                                - 873
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2         0.0224      0.001     37.674      0.000       0.021       0.024
===================================================================================
Ljung-Box (L1) (Q):                   0.14   Jarque-Bera (JB):               744.10
Prob(Q):                              0.71   Prob(JB):                         0.00
Heteroskedasticity (H):               0.82   Skew:                             0.08
Prob(H) (two-sided):                  0.09   Kurtosis:                         7.52
===================================================================================
```

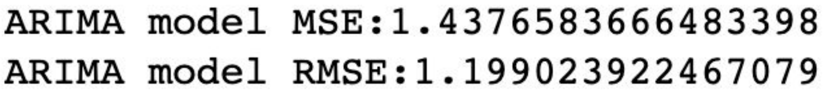
TIME SERIES: SARIMA

- SARIMA - Seasonal Autoregressive Integrated Moving Average Model
    - Assumes that previous values of target variable can be used to predict future values and incorporates potential effects of seasonality
    - Seasonality: When data experiences regular and predictable changes during certain times of the year (i.e. ice cream sales)
- Hyperparameters (p,d,q) + (P,D,Q,M)
    - Besides ARIMA parameters, SARIMA has additional parameters (P,D,Q,M)
        - Set as (0,0,0,0) for the ARIMA model
        - For SARIMA Model, set (0,1,0,5)
            - Strongest autocorrelation was seen in weekly lags (5 trading days in a week), thus we let M = 5
- Four variables were used in the analysis : *dj_open, gdx_open, plt_open, sf_open*

TIME SERIES: Model Results

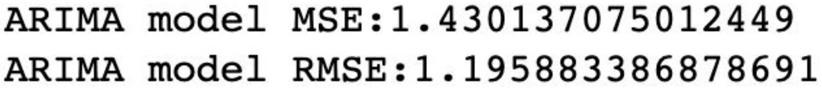
ARIMA

Uni-variate Regression



```
ARIMA model MSE:1.4376583666483398
ARIMA model RMSE:1.199023922467079
```

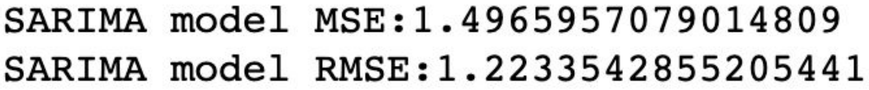
Multivariate Regression



```
ARIMA model MSE:1.430137075012449
ARIMA model RMSE:1.195883386878691
```

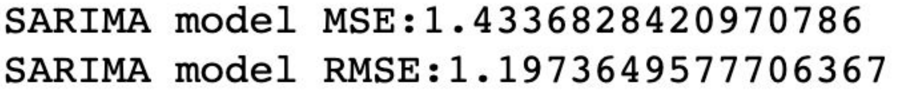
SARIMA

Uni-variate Regression



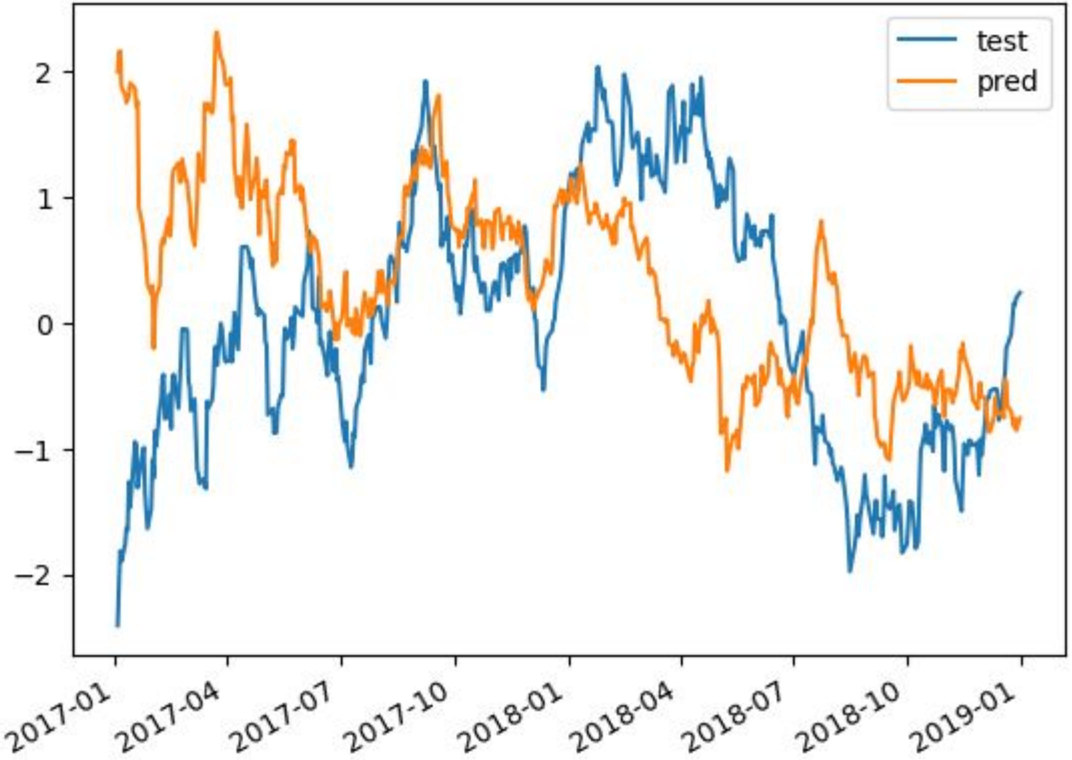
```
SARIMA model MSE:1.4965957079014809
SARIMA model RMSE:1.2233542855205441
```

Multivariate Regression



```
SARIMA model MSE:1.4336828420970786
SARIMA model RMSE:1.1973649577706367
```

TIME SERIES: Best Model is Multivariate ARIMA



Conclusion + Future Extensions

Conclusion

- In conclusion, our time series model will enable investors to gain valuable insights into the future gold prices and make better investment decisions during “normal” times.
- By using our model, investors can now have a greater understanding of the gold market, allowing them to make informed decisions and take advantage of potential opportunities in the market.
- Important to recognize that in present times, where we see market volatility and significant increase in demand for gold due to factors like high inflation, our model might not be the best. For this, we can incorporate sensitivity analysis

Future Extensions

- Incorporate more data
    - Feed the model more data
    - Better be able to understand possible seasonality effect (for SARIMA)
- Exogenous variables
    - Removing some variables and adding better ones
- Other methodology
    - LSTM (best model for stock price prediction) or Time Series Clustering Models