SC1015 Mini Project

Machine Learning & Statistical Modelling of S&P 500

B135_Team_3

Brandon Jang Jin Tian
U2220936G

Chung Zhi Xuan
U2220300H

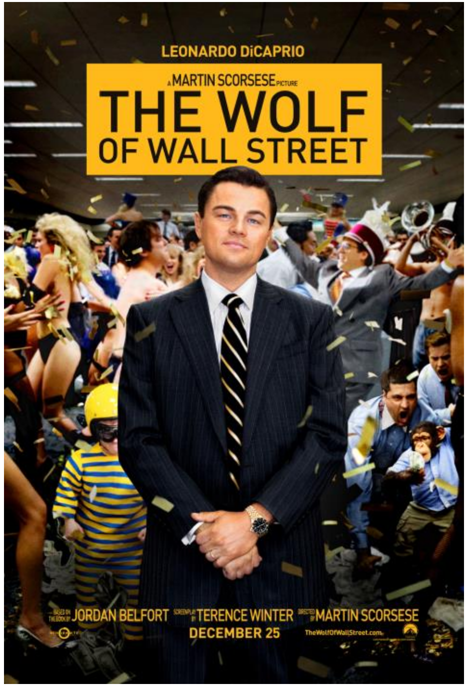
Tee Qin Tong Bettina
U2221901F

TABLE OF CONTENTS

01

Motivation and Problem Statement

Practical Motivation



Sample Collection

kaggle

S&P 500 Historical Data

Historical Data of S&P 500 Index From 1927 to 2020

Sample Collection

Import Economic Data from Federal Reserve Economic Data (FRED) API

In [7]:
```
fred = Fred(api_key='94f47cb371e5afc0e7b4f1ade6f0d840')
```

In [8]:
```
df = {}

df['gdp'] = fred.get_series('GDP') #quarterly
df['gnp'] = fred.get_series('GNP') #quarterly
df['real_gdp'] = fred.get_series('GDPC1') #quarterly
df['real_gdp_per_capita'] = fred.get_series('A939RX0Q048SBEA') #quarterly
df['net_exports'] = fred.get_series('NETEXP') #quarterly
#gross national income
df['gni'] = fred.get_series('A023RC1Q027SBEA') #quarterly
df['govt_spending'] = fred.get_series('GCEC1') #quarterly
df['consumer_spending'] = fred.get_series('PCEC') #quarterly
df['private_domestic_investment'] = fred.get_series('Y006RC1Q027SBEA') #quarterly
#Consumer Price Index for All Urban Consumers: All Items in U.S. City Average
df['cpi'] = fred.get_series('CPIAUCSL', frequency='q', aggregation_method='avg') #monthly, change to quarterly
#Consumer Price Index for All Urban Consumers: Fuel Oil and Other Fuels in U.S. City Average
df['cpi_oil'] = fred.get_series('CUSR0000SEHE', frequency='q', aggregation_method='avg') #monthly, change to quarterly
#interest rates, discount rates per annum
df['ir'] = fred.get_series('INTDSRUSM193N', frequency='q', aggregation_method='avg') #monthly, change to quarterly
df['unemployment_rate'] = fred.get_series('UNRATE', frequency='q', aggregation_method='avg')

df = pd.DataFrame(df)
df
```

Problem Formulation

Will a **Machine Learning model** or a **Statistical model** be better in predicting S&P 500 index?

Are we able to prove S&P 500 to be a **reliable** index to buy?

Will predicting models **justify** the reliability of S&P 500?

Data Preparation

- Adjust the values from daily to quarterly

In [11]:
```
SPX_median = SPX.resample("Q", convention='start', origin='start').median()
SPX_median.index = SPX_median.index + pd.offsets.MonthBegin(-3)
SPX_median
```

Out[11]:

Date	Open	High	Low	Close	Adj Close	Volume
1950-01-01	17.195001	17.195001	17.195001	17.195001	17.195001	1.620000e+06
1950-04-01	18.270000	18.270000	18.270000	18.270000	18.270000	1.850000e+06
1950-07-01	18.459999	18.459999	18.459999	18.459999	18.459999	1.860000e+06
1950-10-01	19.850000	19.850000	19.850000	19.850000	19.850000	2.140000e+06
1951-01-01	21.639999	21.639999	21.639999	21.639999	21.639999	2.070000e+06
...	...	...	...	...	...	...
2019-10-01	3088.344971	3098.130005	3081.744995	3093.619995	3093.619995	3.472745e+09
2020-01-01	3243.265014	3259.330079	3233.464966	3244.954956	3244.954956	4.001320e+09
2020-04-01	2930.909912	2954.860107	2912.159912	2930.189941	2930.189941	5.567400e+09
2020-07-01	3336.974976	3355.229981	3317.755005	3332.765014	3332.765014	4.254600e+09
2020-10-01	3434.280029	3447.280029	3405.169922	3426.919922	3426.919922	3.988080e+09

284 rows × 6 columns

Data Preparation

S&P 500 Historical Data

+

Federal Reserve Economic Data (FRED)

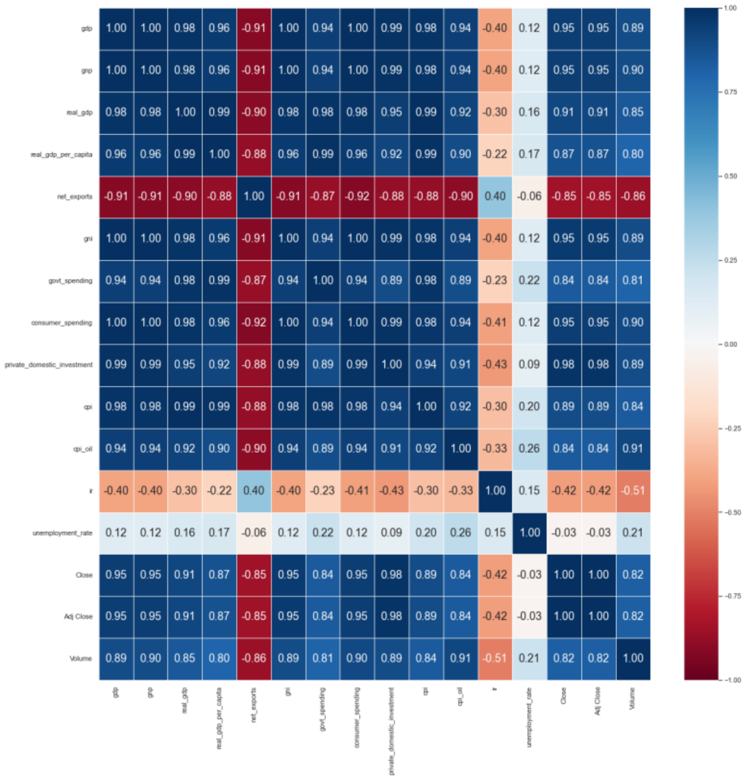
In [12]:
```
df_concat = pd.concat([df,SPX_median], axis=1)
df_concat = df_concat.dropna()
df_concat = df_concat.drop(['Open','High','Low'], axis=1)
display(df_concat)
df_concat.info()
```

	gdp	gnp	real_gdp	real_gdp_per_capita	net_exports	gni	govt_spending	consumer_spending	private_domestic_investment	cpi
1950-01-01	280.828	282.056	2186.365	14500.0	2.203	278.734	599.569	182.920	1.071	23.587
1950-04-01	290.383	291.699	2253.045	14889.0	1.643	291.250	610.519	186.806	1.164	23.767
1950-07-01	308.153	309.760	2340.112	15398.0	-0.740	309.130	600.663	200.505	1.247	24.203
1950-10-01	319.945	321.554	2384.920	15623.0	-0.154	320.905	643.100	197.946	1.289	24.693
1951-01-01	336.000	337.537	2417.311	15769.0	0.177	334.071	711.537	209.207	1.296	25.697
...	...	...	...	...	...	...	...	...	...	...
2019-10-01	21706.532	21961.573	19215.691	58017.0	-514.683	22066.642	3360.850	14619.042	530.682	257.888
2020-01-01	21538.032	21794.019	18989.877	57279.0	-522.725	22136.020	3387.944	14440.160	542.381	258.803
2020-04-01	19636.731	19805.951	17378.712	52393.0	-526.273	20061.715	3448.043	13049.766	537.107	256.315
2020-07-01	21362.428	21562.441	18743.720	56479.0	-692.412	21365.888	3395.877	14388.701	561.469	259.239
2020-10-01	21704.706	21867.367	18924.262	56993.0	-768.588	22325.818	3394.795	14586.036	586.224	261.045

284 rows × 16 columns

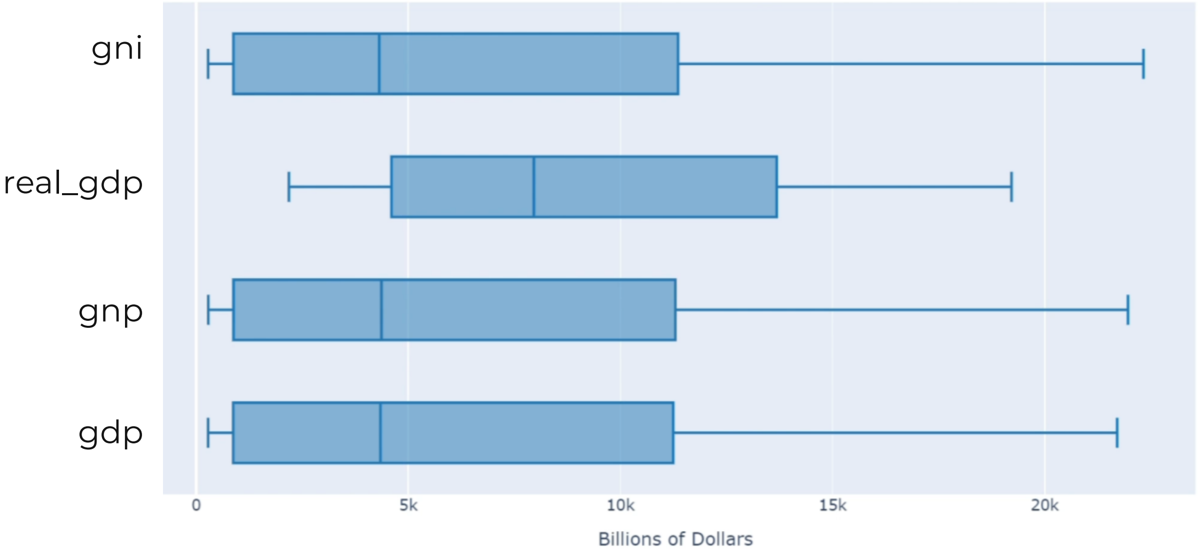
Data Preparation - Correlation

- Clean data based on the correlation values against Close
- Remove variables that have correlation values < 0.9
- GDP, GNP, Real GDP, GNI, Consumer Spending and Private Domestic Investment



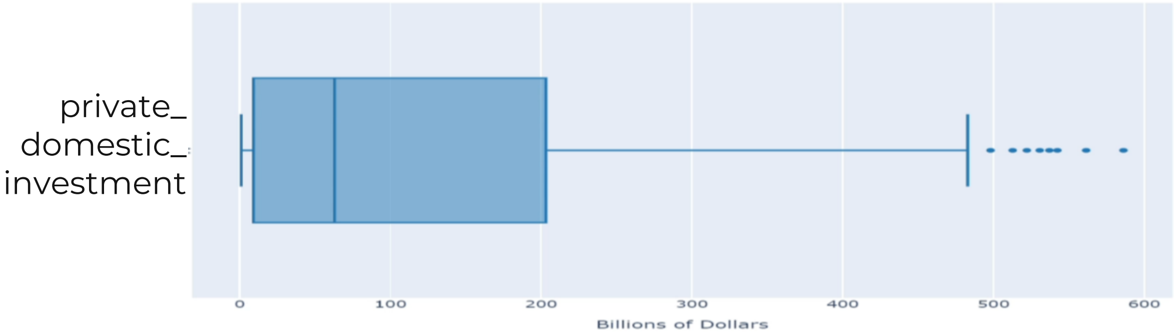
Data Preparation - Outliers

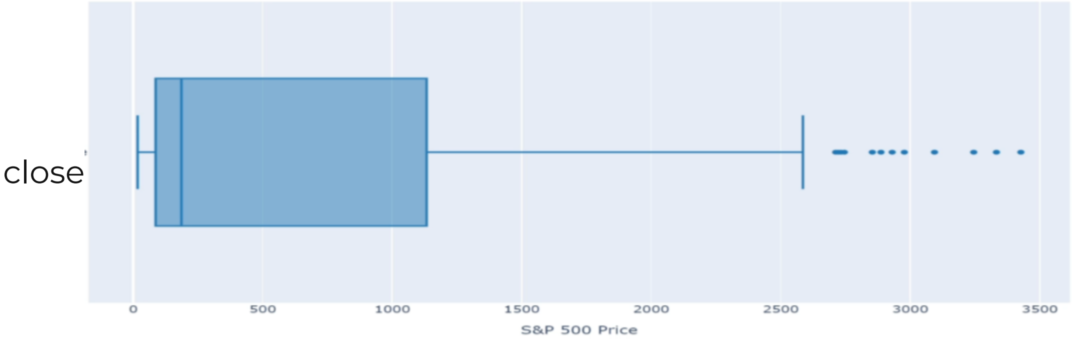
- Identifying outliers in the data
- Plot boxplot to identify outliers present
- GNI, Real GDP, GNP, GDP have no outliers in the data set



Data Preparation - Outliers

- Private Domestic Investment and Close variables contain outliers



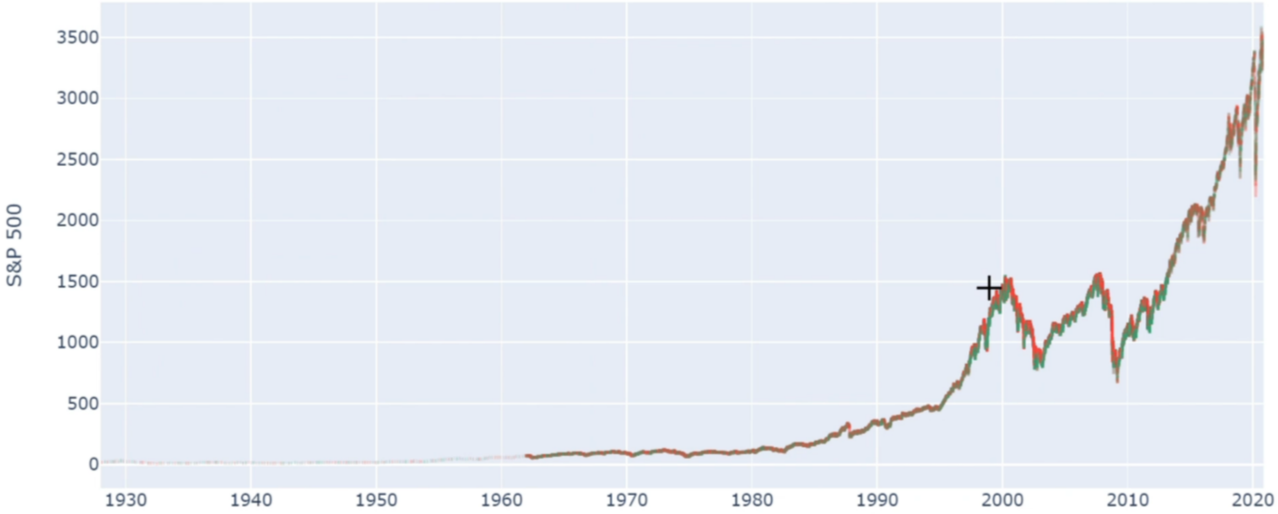


02

Exploratory Data Analysis

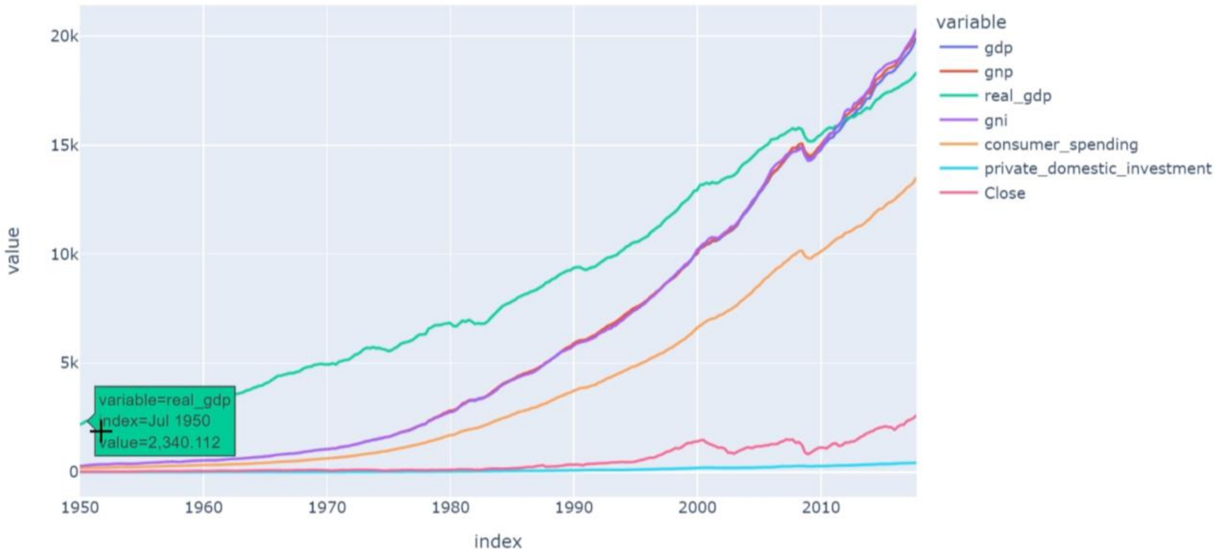
Exploratory Analysis & Analytic Visualisation

- **Candlestick Plot** to visualize the price of S&P 500



Exploratory Analysis & Analytic Visualisation

- **Time Series** to visualize the changes in index with respect to each variable



03

Predictive Models

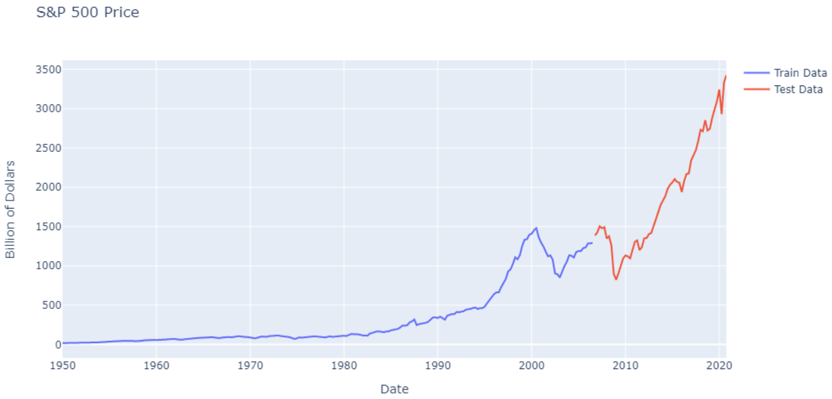
Predictive Models

Machine Learning	Statistical	
LSTM Long Short-Term Memory	**ARIMA** Autoregressive integrated moving average	**SARIMA** Seasonal-ARIMA
An artificial recurrent neural network used in deep learning to predict future prices	A statistical analysis model that uses time series data to either better understand the data set or to predict future trends	Takes into account of seasonality when predicting the future trends

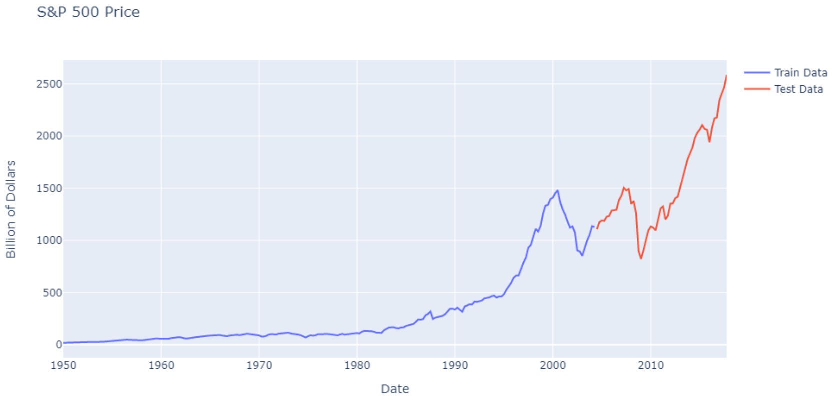
Train-Test Split

Train Data: 80%
Test Data: 20%

With Outliers

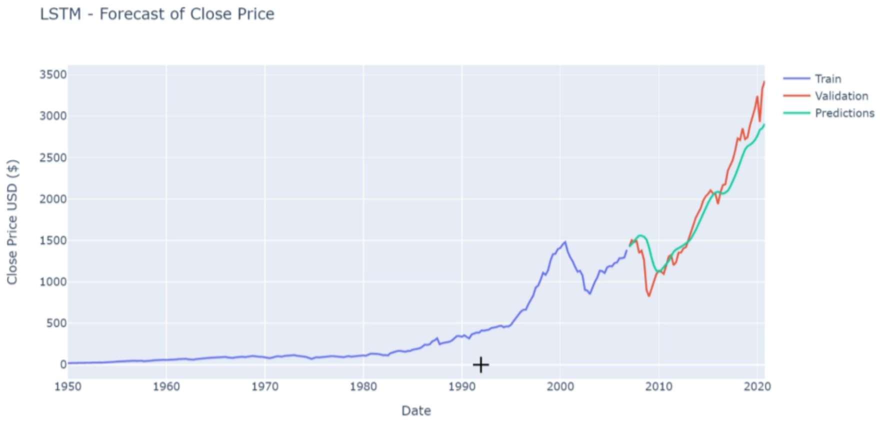


Without Outliers

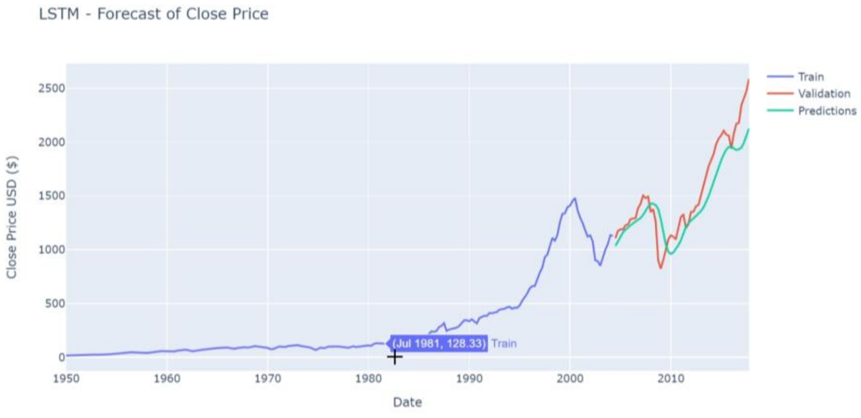


Machine Learning - LSTM

With Outliers



Without Outliers



Generating the ARIMA Model

```python
model_autoARIMA = auto_arima(x_train, start_p=0, start_q=0,
                      test='adf',       # use adftest to find optimal 'd'
                      max_p=4, max_q=4, # maximum p and q
                      m=1,              # frequency of series
                      d=None,           # let model determine 'd'
                      seasonal=False,   # No Seasonality
                      start_P=0,
                      D=0,
                      trace=True,
                      error_action='ignore',
                      suppress_warnings=True,
                      stepwise=True)
print(model_autoARIMA.summary())
model_autoARIMA.plot_diagnostics(figsize=(15,8))
plt.show()
```

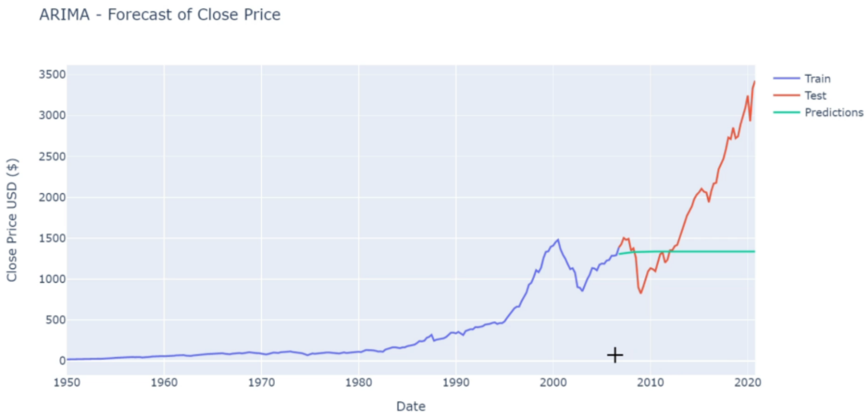
```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=2150.365, Time=0.03 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=2111.710, Time=0.11 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=2120.424, Time=0.21 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=2157.415, Time=0.03 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=2108.431, Time=0.17 sec
 ARIMA(3,1,0)(0,0,0)[0] intercept   : AIC=2099.457, Time=0.20 sec
 ARIMA(4,1,0)(0,0,0)[0] intercept   : AIC=2100.068, Time=0.29 sec
 ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=2099.669, Time=0.53 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=2105.216, Time=0.34 sec
 ARIMA(4,1,1)(0,0,0)[0] intercept   : AIC=2101.668, Time=0.80 sec
 ARIMA(3,1,0)(0,0,0)[0]             : AIC=2099.276, Time=0.15 sec
 ARIMA(2,1,0)(0,0,0)[0]             : AIC=2109.159, Time=0.12 sec
 ARIMA(4,1,0)(0,0,0)[0]             : AIC=2100.148, Time=0.18 sec
 ARIMA(3,1,1)(0,0,0)[0]             : AIC=2099.699, Time=0.25 sec
 ARIMA(2,1,1)(0,0,0)[0]             : AIC=2105.008, Time=0.21 sec
 ARIMA(4,1,1)(0,0,0)[0]             : AIC=2101.690, Time=0.50 sec

Best model:  ARIMA(3,1,0)(0,0,0)[0]
Total fit time: 4.157 seconds
                               SARIMAX Results
==============================================================================
Dep. Variable:                      y   No. Observations:                  227
Model:               SARIMAX(3, 1, 0)   Log Likelihood               -1045.638
Date:                Tue, 04 Apr 2023   AIC                           2099.276
Time:                        21:09:53   BIC                           2112.959
Sample:                    01-01-1950   HQIC                          2104.798
                         - 07-01-2006
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.3192      0.039      8.139      0.000       0.242       0.396
ar.L2          0.0833      0.048      1.730      0.084      -0.011       0.178
ar.L3          0.2267      0.025      9.067      0.000       0.178       0.276
sigma2       610.1641     23.859     25.574      0.000     563.402     656.926
===================================================================================
Ljung-Box (L1) (Q):                   0.02   Jarque-Bera (JB):              1583.86
Prob(Q):                              0.88   Prob(JB):                         0.00
Heteroskedasticity (H):             182.88   Skew:                            -0.78
Prob(H) (two-sided):                  0.00   Kurtosis:                        15.87
===================================================================================
```

Statistical - ARIMA

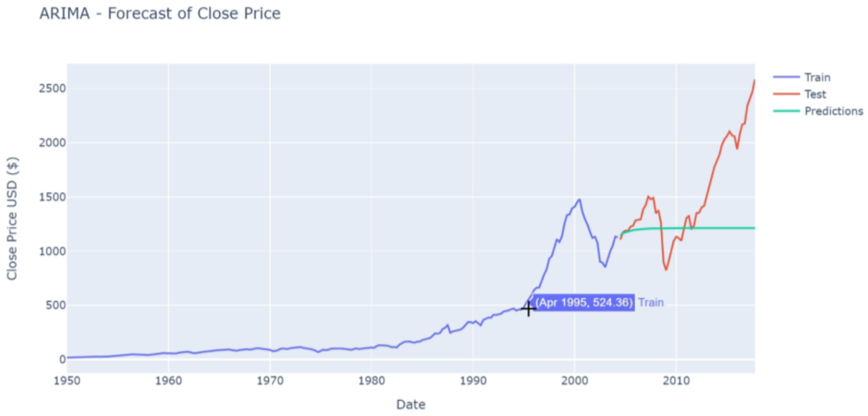
With Outliers

Optimal ARIMA model is (3,1,0)



Without Outliers

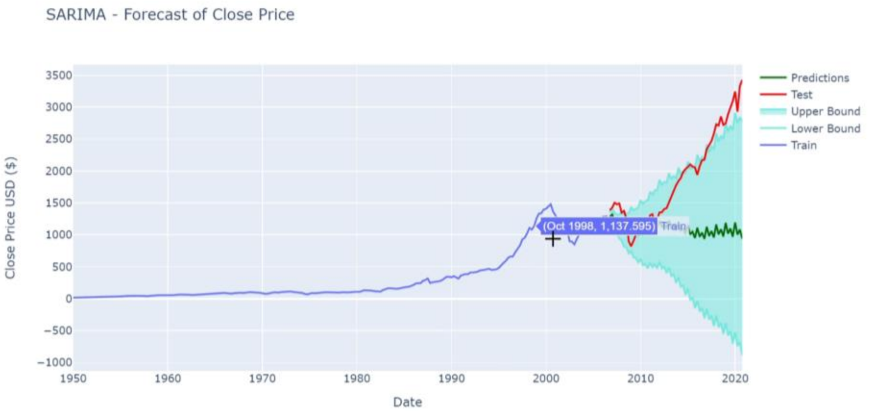
Optimal ARIMA model is (3,1,1)



Statistical - SARIMA

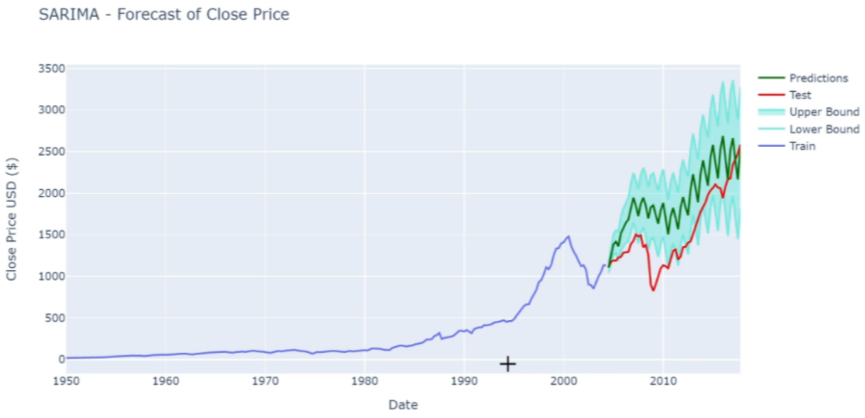
With Outliers

Optimal SARIMA model is (2, 0, 1)x(2, 2, [], 4)



Without Outliers

Optimal SARIMA model is (3, 0, 1)x(2, 2, [], 4)



04

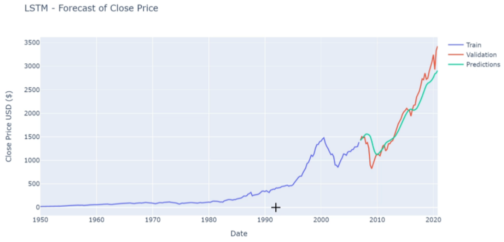
Statistical Inference & Information Presentation

Statistical Inference

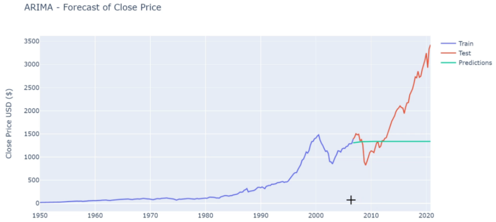
With Outliers

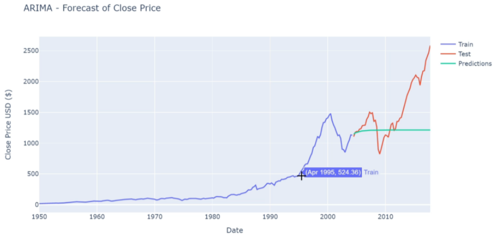
Without Outliers

LSTM

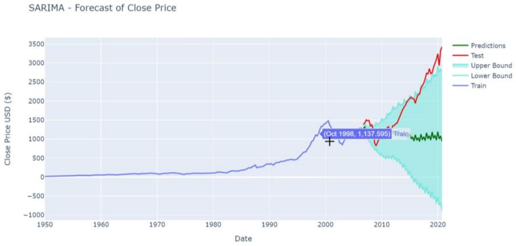


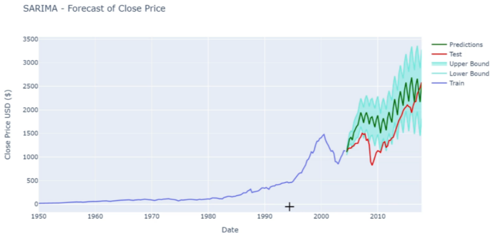
ARIMA





SARIMA





Statistical Inference & Information Presentation

With Outliers

```
{'LSTM_Test': 51.597890569614925,
 'Linear Regression Test': 272.8623891859237,
 'ARIMA': 890.3537042779377,
 'SARIMA': 1082.2973032709044}
```

Without Outliers

```
{'LSTM_Test': 108.67615652437789,
 'Linear Regression Test': 253.55492493307582,
 'ARIMA': 546.1224083484476,
 'SARIMA': 472.58294620426545}
```

Statistical Inference & Information Presentation

	gdp	gnp	real_gdp	gni	consumer_spending	private_domestic_investment	Close
2018-01-01	20155.486	20467.925	18437.127	20574.837	13677.349	447.647	2732.219971
2018-04-01	20470.197	20772.518	18565.697	20787.673	13850.838	463.342	2712.209961
2018-07-01	20687.278	20958.320	18699.748	21091.158	13988.794	468.156	2853.580078
2018-10-01	20819.269	21094.695	18733.741	21295.748	14102.937	482.915	2722.179932
2019-01-01	21013.085	21299.154	18835.411	21490.817	14145.897	498.098	2745.729980
2019-04-01	21272.448	21562.375	18962.175	21677.974	14323.749	512.729	2886.979980
2019-07-01	21531.839	21813.003	19130.932	21822.711	14482.196	522.137	2977.180054
2019-10-01	21706.532	21961.573	19215.691	22066.642	14619.042	530.682	3093.619995
2020-01-01	21538.032	21794.019	18989.877	22136.020	14440.160	542.381	3244.954956
2020-04-01	19636.731	19805.951	17378.712	20061.715	13049.766	537.107	2930.189941
2020-07-01	21362.428	21562.441	18743.720	21365.888	14388.701	561.469	3332.765014
2020-10-01	21704.706	21867.367	18924.262	22325.818	14586.036	586.224	3426.919922

Statistical Inference & Information Presentation

With Outliers

```
{'LSTM_Test': 51.597890569614925,
 'Linear Regression Test': 272.8623891859237,
 'ARIMA': 890.3537042779377,
 'SARIMA': 1082.2973032709044}
```

Without Outliers

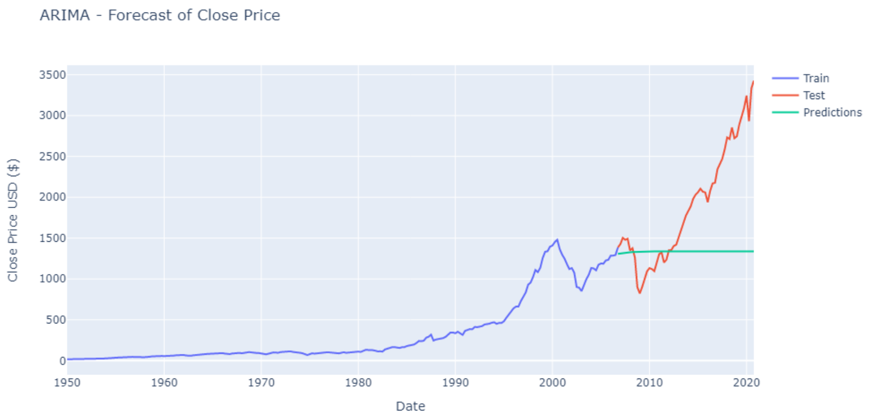
```
{'LSTM_Test': 108.67615652437789,
 'Linear Regression Test': 253.55492493307582,
 'ARIMA': 546.1224083484476,
 'SARIMA': 472.58294620426545}
```

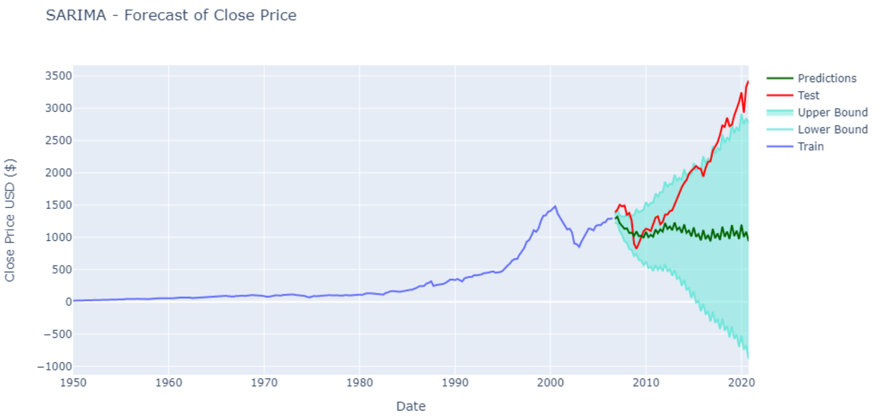
Statistical Inference & Information Presentation

With Outliers

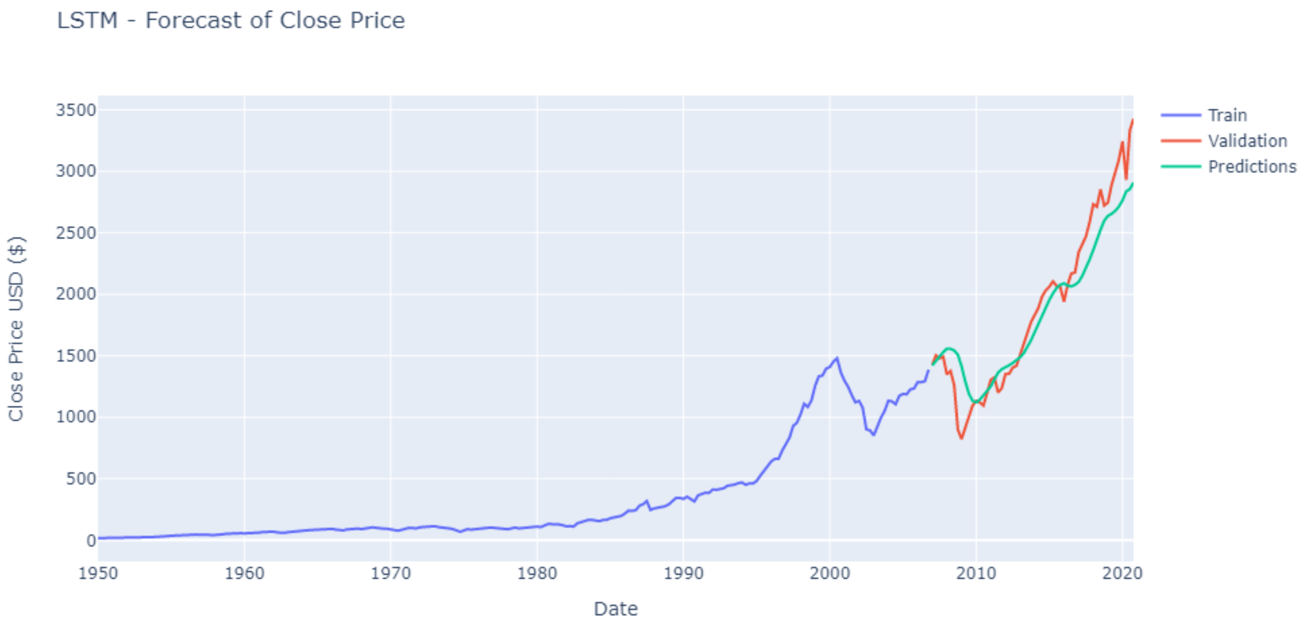
```
{'LSTM_Test': 51.597890569614925,
 'Linear Regression Test': 272.8623891859237,
 'ARIMA': 890.3537042779377,
 'SARIMA': 1082.2973032709044}
```

Statistical Inference & Information Presentation





Statistical Inference & Information Presentation

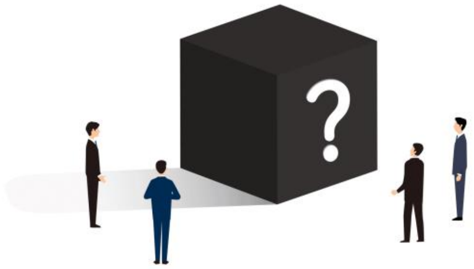


05

Insights and Conclusion

Ethical Considerations

- **Possibility of Reinforced Human Bias**
  - Data Used in Algorithm could possess biases
- **Lack of Transparency**
  - External parties can trust our model and make informed decisions
- **Over Reliance on Model**
  - Not used as the sole basis for investment decisions



Conclusion

- Dataset **with outliers** is more accurate and realistic to predict real-index prices
- ARIMA Predictive Model is not realistic due to predicted prices not being seasonal
- **LSTM** Predictive Model is the most accurate model out of the 3 models to predict index prices

LSTM Model predicts a steady upward trend for the S&P 500 index prices. Hence, the LSTM predicting model can justify that S&P 500 index is a **reliable index** to purchase.

THANKS!

Credits: This presentation template was created by **Slidesgo**, including icons by **Flaticon** and infographics & images by **Freepik**