PROJECT REPORT

TOPIC: Stock exchange prediction

By Jayanti Bhattacharya

ISTANBUL STOCK EXCHANGE Data Set

Download: Data Folder, Data Set Description

Abstract: Data sets includes returns of Istanbul Stock Exchange with seven other international index; SP, DAX, FTSE, NIKKEI, BOVESPA, MSCE_EU, MSCI_EM from Jun 5, 2009 to Feb 22, 2011.

Data Set Characteristics:	Multivariate, Univariate, Time-Series	**Number of Instances:**	536	**Area:**	Business
Attribute Characteristics:	Real	**Number of Attributes:**	8	**Date Donated**	2013-06-01
Associated Tasks:	Classification, Regression	**Missing Values?**	N/A	**Number of Web Hits:**	150855

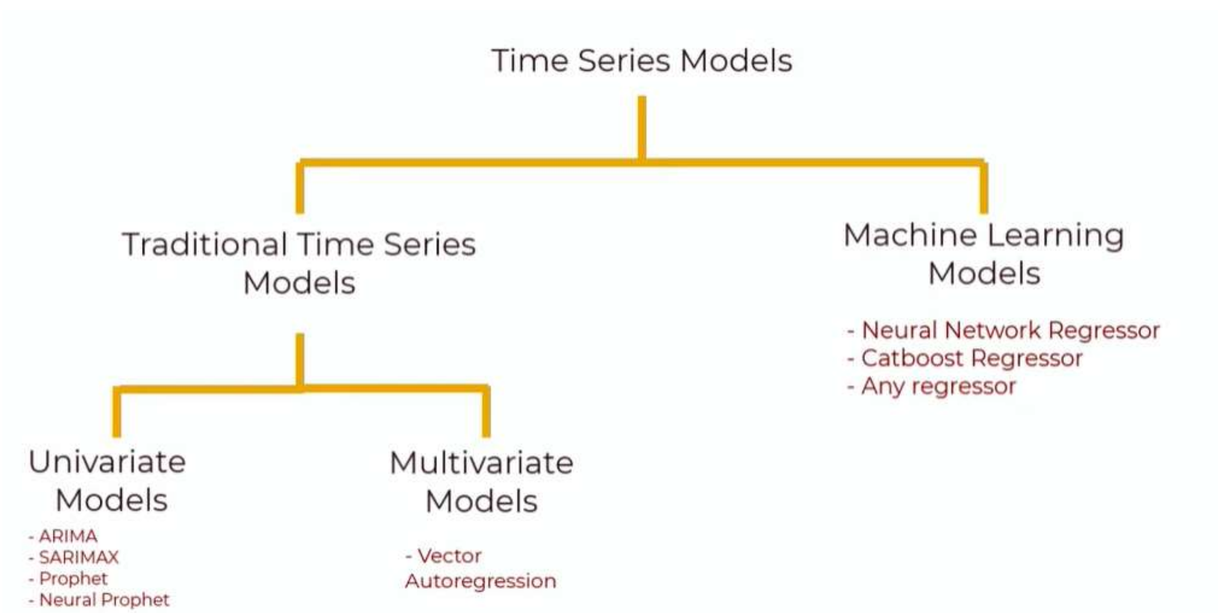
About the dataset:

The dataset has 10 columns: DATE, ISE, ISE.1, SP, DAX, FTSE, NIKKEI, BOVESPA, EM and EU. The starting date is 5th January 2009 and the ending date is 22nd February 2011 and has 536 rows.

Problem statement:

The objective of this project is to predict the value of a stock exchange, when the values of the other stock exchanges are given. Using statistical models like ARIMA, VAR and machine learning models like XGBoost, trying to find the model for best prediction.

As given in the graph below, we will try to predict the value of the stock exchange using both traditional time series models and machine learning models.

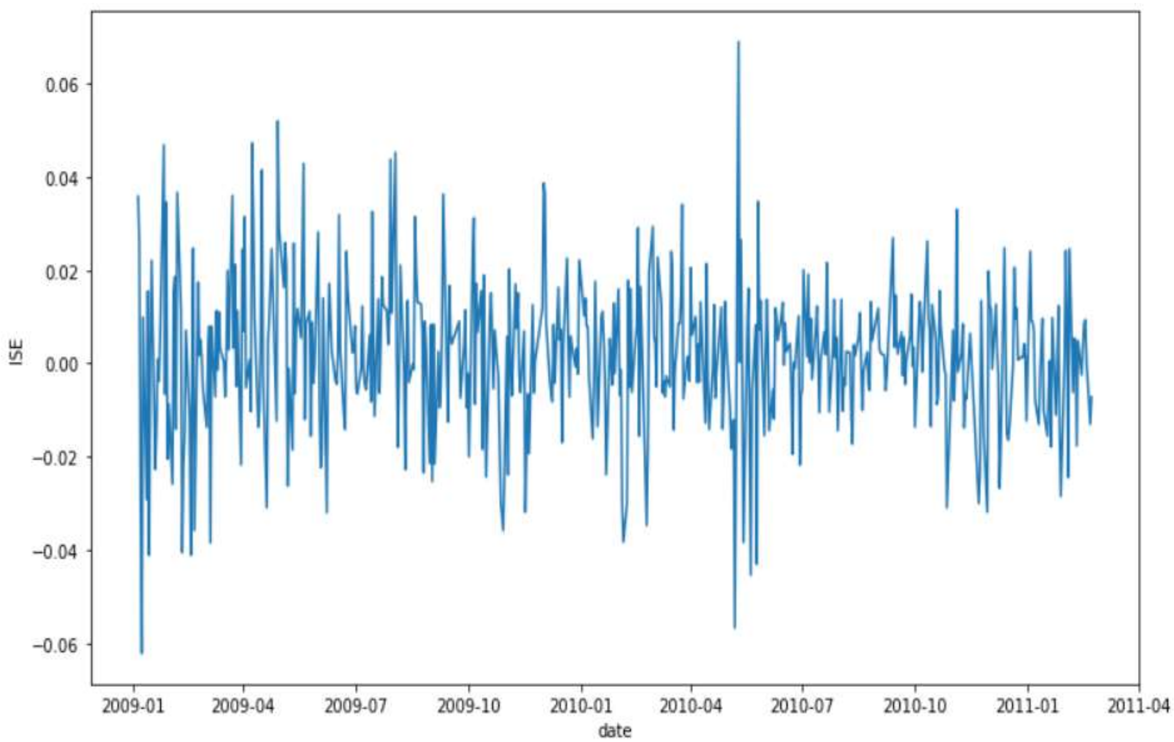


*** This report is based on the stock exchange ISE (1st column)*

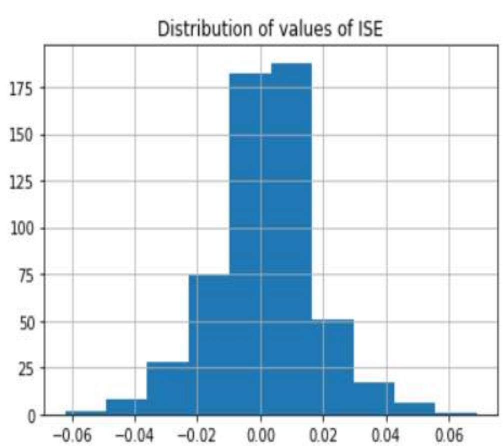
Methodology:

1) Analysis of the time-series data

The relevant python libraries and the dataset is imported. The plot of date vs ISE is:

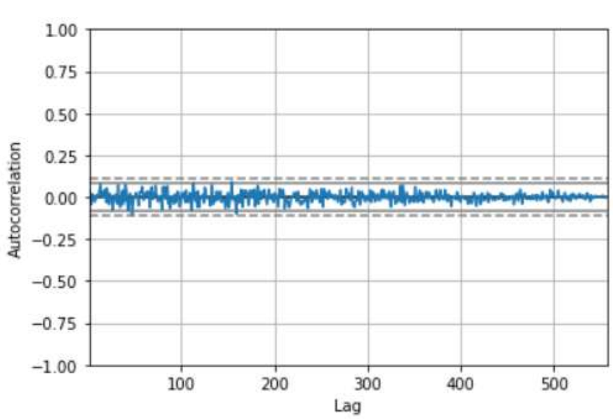


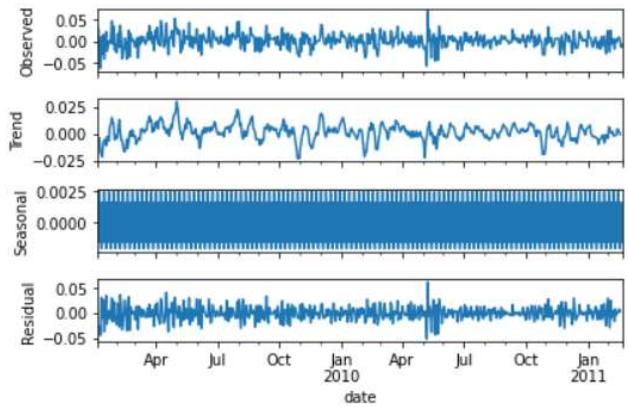
The data is then transformed by dropping all the columns and the frequency according to every business day within the period. The mean is close to zero and variance is constant. The distribution of values follows Gaussian distribution which lead to the assumption of the data being white-noise.



	ISE
count	557.000000
mean	0.001611
std	0.016220
min	-0.062208
25%	-0.006520
50%	0.002217
75%	0.010203
max	0.068952

From the autocorrelation plot, we see that the values of the data are not correlated with each other. Also, from the seasonal decompose graphs we see that the trend resembles the observed graph and has no clear positive or negative incline. The seasonal sequence has no clear pattern, hence there is no pattern in graph, Residual also has no fixed pattern as well. Hence there is no seasonality in the above time series.





The Ad-Fuller test for stationarity when tested on this data, shows that the test statistic is less than 1%, 5%, 10% critical values, p-value is 0 and number of lags used is also 0, which proves that the autocorrelation among values is 0.

The data has a mean 0, constant variance and no autocorrelation among values, hence it is white noise and cannot be forecasted by definition (as its values at different time points are statistically independent).

The above statement is proven by fitting to an ARIMA model.

2) ARIMA model: (Univariate analysis)

The best ARIMA model order (p,d,q) is determined by the auto_arima function which comes out to be (0,0,0) (as the data is white noise). The summary of the model after fitting training data to the model is given below:

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=-3002.646, Time=0.69 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=-3007.686, Time=0.14 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=-3007.120, Time=0.06 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=-3007.098, Time=0.13 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=-3004.210, Time=0.06 sec
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=-3005.122, Time=0.14 sec

Best model:  ARIMA(0,0,0)(0,0,0)[0] intercept
Total fit time: 1.240 seconds
```

ARMA Model Results

Dep. Variable:	ISE	**No. Observations:**	446
Model:	ARMA(0, 0)	**Log Likelihood**	1189.795
Method:	css	**S.D. of innovations**	0.017
Date:	Thu, 08 Jul 2021	**AIC**	-2375.591
Time:	13:13:22	**BIC**	-2367.390
Sample:	01-05-2009 - 09-20-2010	**HQIC**	-2372.357

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.0021	0.001	2.616	0.009	0.001	0.004

The test data is then predicted using the model. The actual and predicted values come out to be:

The graph cannot at all predict the results properly, proving our assumption that the time-series is white noise is true.

3) VAR model: (Multivariate analysis)

Here, all the columns are tested for stationarity using ad-fuller test. Here, all the columns of the dataset are stationary.

Then, we check for check for causality using Granger Causality test, hence checking if other columns cause ISE. In the Granger Causality test, we see:

1. *The data for test whether the time series in the second column Granger causes the time series in the first column.*
2. *If p<0.05 for all the 4 tests we can say that the 2nd column specified in the causality test causes ISE for that lag onwards.*

In the picture, we see Granger Causality test of ISE with respect to ISE.1 for 3 lags and the same is repeated for all the other columns.

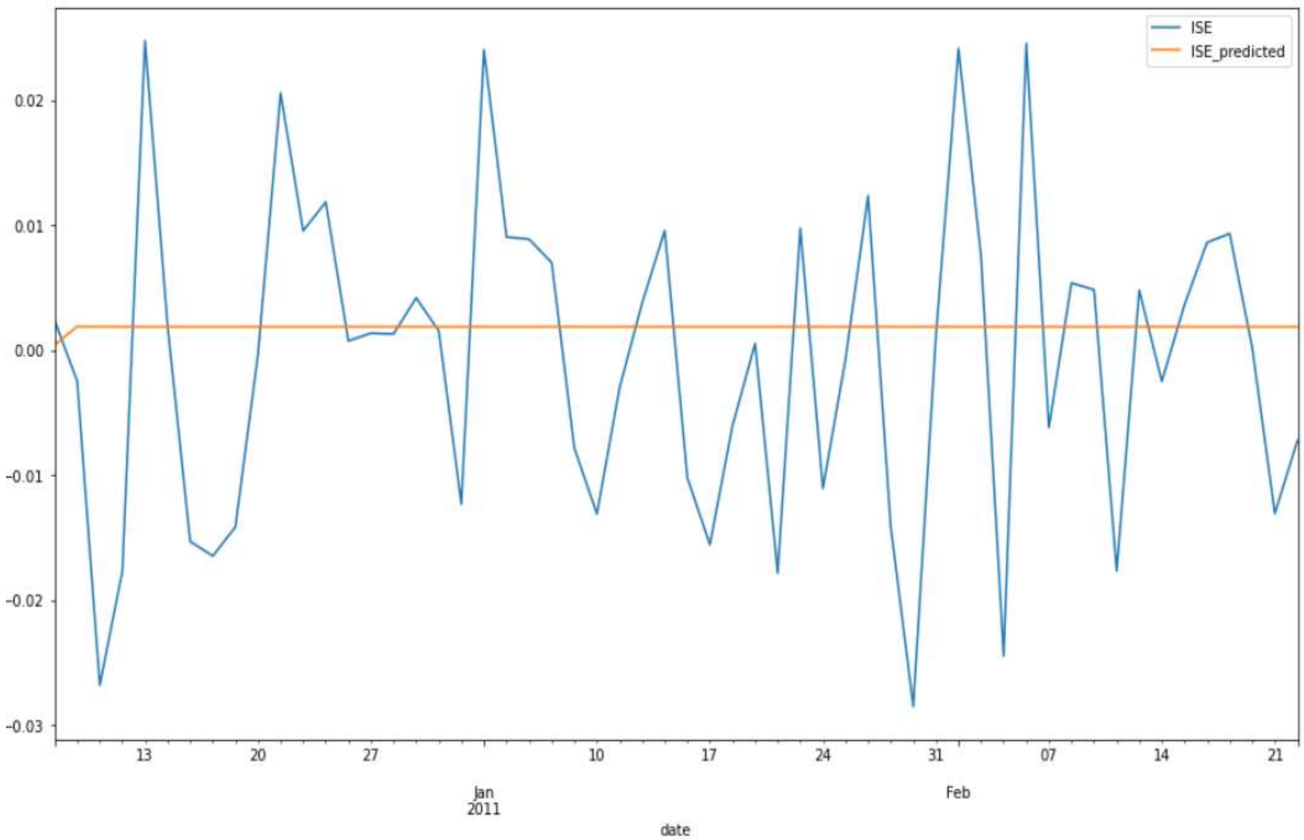
```
Does ISE.1 causes ISE?
------------------------------------------------------------

Granger Causality
number of lags (no zero) 1
ssr based F test:         F=0.1324  , p=0.7161  , df_denom=553, df_num=1
ssr based chi2 test:   chi2=0.1331  , p=0.7153  , df=1
likelihood ratio test: chi2=0.1331  , p=0.7153  , df=1
parameter F test:         F=0.1324  , p=0.7161  , df_denom=553, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=0.0673  , p=0.9349  , df_denom=550, df_num=2
ssr based chi2 test:   chi2=0.1359  , p=0.9343  , df=2
likelihood ratio test: chi2=0.1359  , p=0.9343  , df=2
parameter F test:         F=0.0673  , p=0.9349  , df_denom=550, df_num=2

Granger Causality
number of lags (no zero) 3
ssr based F test:         F=0.6261  , p=0.5984  , df_denom=547, df_num=3
ssr based chi2 test:   chi2=1.9024  , p=0.5929  , df=3
likelihood ratio test: chi2=1.8992  , p=0.5936  , df=3
parameter F test:         F=0.6261  , p=0.5984  , df_denom=547, df_num=3
```

After doing the Granger Causality tests, we see that two other columns SP and BOVESPA cause ISE, so we make a dataframe using these 3time series. The data is then divided into training and test sets and the relevant libraries for vector auto regressions are imported.

We selected the order for the VAR model using select_order to compute lag order selections based on each of the available information criteria. Here both AIC and FPE have shown the order 1 as minimum so we select the order as 1. After using the model and fitting the training data to it, we get the following summary:

```
                           Statespace Model Results
==================================================================================
Dep. Variable:     ['ISE', 'SP', 'BOVESPA']   No. Observations:               501
Model:                               VAR(1)   Log Likelihood             4373.581
                                + intercept   AIC                       -8711.162
Date:                      Tue, 03 Aug 2021   BIC                       -8635.263
Time:                              12:32:21   HQIC                      -8681.382
Sample:                          01-05-2009
                               - 12-06-2010
Covariance Type:                        opg
===================================================================================
Ljung-Box (Q):          33.12, 49.59, 40.55   Jarque-Bera (JB):   33.00, 59.15, 55.19
Prob(Q):                   0.77, 0.14, 0.45   Prob(JB):              0.00, 0.00, 0.00
Heteroskedasticity (H):    0.56, 0.42, 0.51   Skew:                -0.03, -0.02, 0.18
Prob(H) (two-sided):       0.00, 0.00, 0.00   Kurtosis:              4.26, 4.68, 4.58
                           Results for equation ISE
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
intercept      0.0017      0.001      2.315      0.021       0.000       0.003
L1.ISE        -0.0563      0.043     -1.300      0.194      -0.141       0.029
L1.SP          0.0860      0.075      1.142      0.253      -0.062       0.233
L1.BOVESPA     0.1649      0.062      2.675      0.007       0.044       0.286
```

Then we get the model's predictions for the testing data The actual and predicted values come out to be:

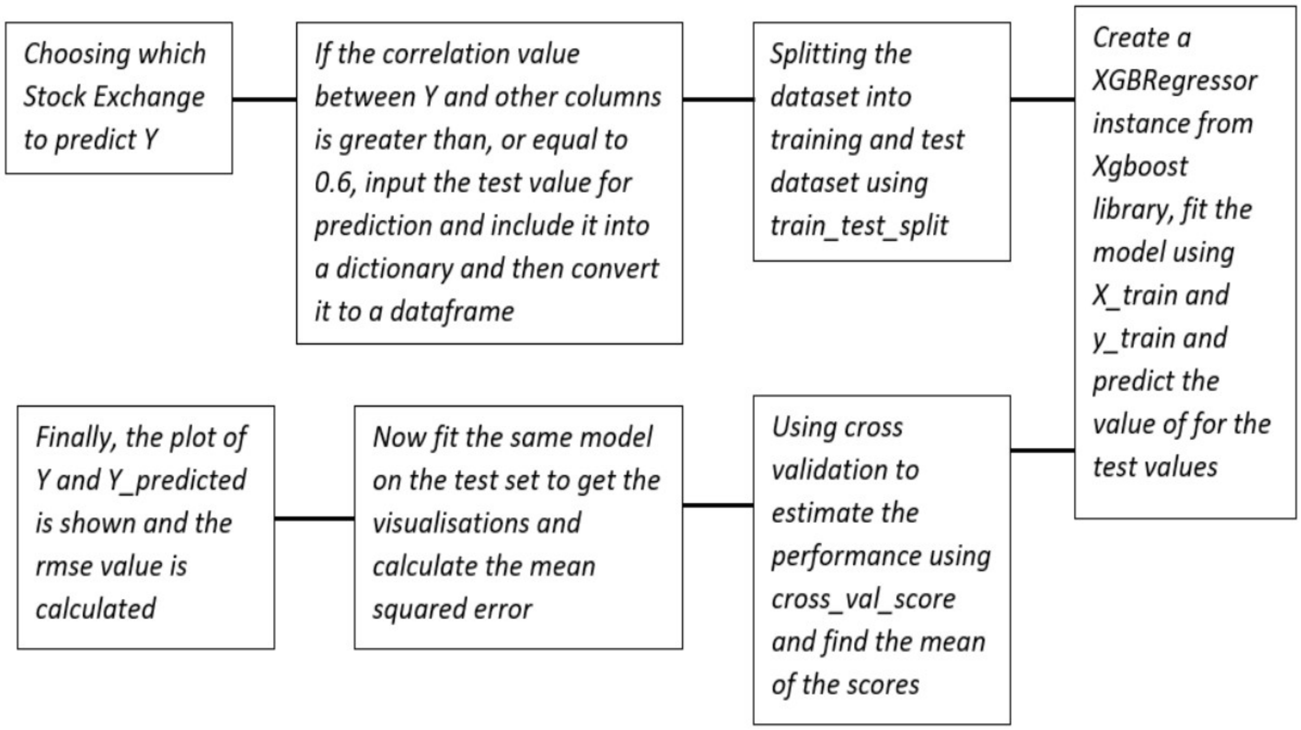


The graph cannot at all predict the results properly, the predictions shown by the plot above are correct as the time series data is white noise and cannot be predicted.

4) XGBoost Model

We can see from the above 2 models: ARIMA and VAR, that traditional time series data is not being able to predict the test values at all, since the data is white noise time-series data. Hence, a machine learning technique is being used to get better predictions for the model and here XGBoost is used.

XGBoost is an efficient implementation of gradient boosting that can be used for regression predictive modelling. Here, all the analysis, model fitting and results are done through one function. The entire working of the model is done below:



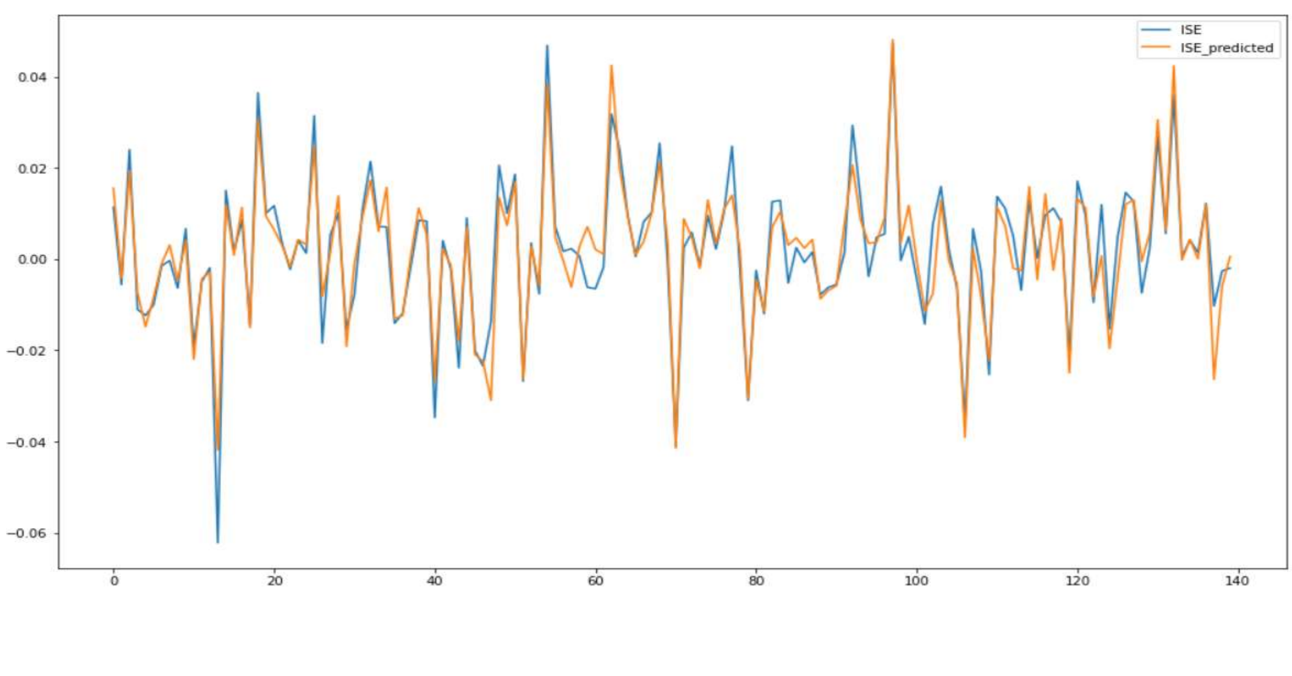
Snippet of the code:

```
def predict_function():
  str1 = str(input('Which stock exchange do you want to predict: '))
  print('---------------------------------------------------------------')
  if str1 == 'NIKKEI':
    print('There is no good correlation among columns and NIKKEI so cannot be predicted')
    return
  y = df[str1]
  dict1 = {}
  value = 0
  lst = []
  for i in df.columns:
    if i!=str1:
      j = y.corr(df[i])
      if j>=0.6:
        lst.append(i)
        value = float(input('Enter the test value for column {}:   '.format(i)))
        dict1[i]=value
  test_values = pd.DataFrame(dict1,index=[0])
  X = df[lst]
  X_train, X_test, y_train, y_test = train_test_split(X, y)
  my_model = XGBRegressor(n_estimators=1000, early_stopping_rounds=5,verbosity=0)
  my_model.fit(X_train, y_train,verbose=False)
  print('---------------------------------------------------------------')
  scores = -1 * cross_val_score(my_model, X_test, y_test,
                                cv=5,
                                scoring='neg_mean_absolute_error')
```

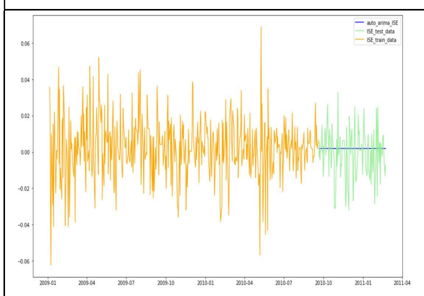
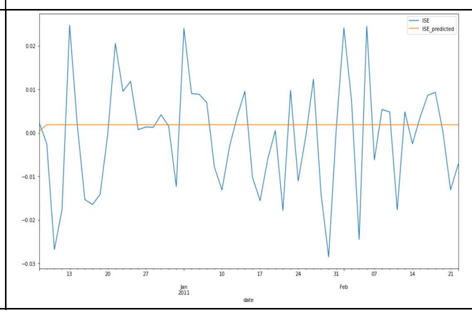
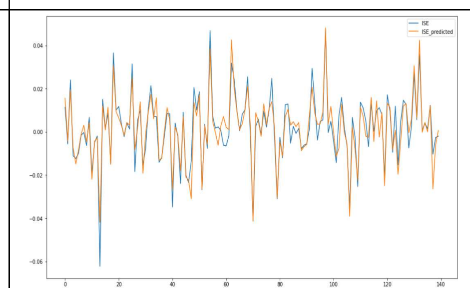
```python
print("Mean absolute error scores for cross validation:\n", scores)
print('--------------------------------------------------------------')
print("Average mean absolute error score (across experiments):",scores.mean())
pred_test_values = my_model.predict(test_values)
print('--------------------------------------------------------------')
print('The predicted stock exchange value is: ',pred_test_values)
print('--------------------------------------------------------------')
predictions = my_model.predict(X_test)
predictions = pd.DataFrame(predictions)
predictions.columns = ['predictions']
y_test = y_test.to_frame(name=str1).reset_index()
y_test.drop(columns='index',inplace=True)
test_pred = pd.concat([y_test,predictions],axis=1,ignore_index=True)
test_pred.columns = [str1,str1+'_predicted']
mse = math.sqrt(mean_squared_error(test_pred[str1],test_pred[str1+'_predicted']))
print('Root mean square error between the predictions and the test values are', mse)
print('--------------------------------------------------------------')
test_pred.plot(figsize=(16,9))
```

Here, the user inputs the stock exchange they want to predict, after inputting the relevant values, the user gets back the predicted value of the stock exchange as well as other analysis results.

```
Which stock exchange do you want to predict: ISE
--------------------------------------------------------------
Enter the test value for column ISE.1:   0.038376
Enter the test value for column DAX:   0.002193
Enter the test value for column FTSE:   0.003894
Enter the test value for column EU:   0.012698
Enter the test value for column EM:   0.028524
--------------------------------------------------------------
Mean absolute error scores for cross validation:
 [0.00425394 0.00456587 0.00475533 0.00392497 0.00468747]
--------------------------------------------------------------
Average mean absolute error score (across experiments): 0.004437516757469941
--------------------------------------------------------------
The predicted stock exchange value is:  [0.03489619]
--------------------------------------------------------------
Root mean square error between the predictions and the test values are 0.005299169059987376
--------------------------------------------------------------
```



Difference between ARIMA, VAR and XGBoost models:

ARIMA	***VAR***	***XGBoost***
It stands for Autoregressive Integrated Moving Average	*It stands for Vector Autoregression*	*It stands for eXtreme Gradient Boosting*
ARIMA models are used for univariate time series. The structure is that the variable is a linear function of past lags of itself and past shocks	*VAR models are used for multivariate time series. The structure is that each variable is a linear function of past lags of itself and past lags of the other variables*	*XGBoost is a powerful machine learning approach for building supervised regression models. XGBoost is an efficient implementation of gradient boosting that can be used for regression predictive modelling*
White noise time series data cannot be predicted by Arima model due to its statistical properties like mean=0, constant variance and no autocorrelation	*White noise time series data cannot be predicted by var model due to its statistical properties like mean=0, constant variance and no autocorrelation*	*White noise can be predicted by XGBoost as it doesn't take the statistical properties of the data into account as it exclusively cares about quality of prediction*
		

Conclusion:

After doing the data analysis we see that the data is white noise data as it has statistical properties like mean=0, constant variance and no autocorrelation among values. Due to this traditional time series models like ARIMA, VAR are not able to predict the data at all. But efficient machine learning approaches like XGBoost is able to predict the data as it doesn't take the statistical properties of the data into account as it exclusively cares about quality of prediction. Hence, after fitting the stock exchange data to several models, we see that the XGBoost model worked the best out of the three models and gives out accurate predictions.

Thank You!