Statistical Analyses on S&P 500 Index

Presented by GROUP G

YE, Yifan 20551471

LI, Jiahao 20568357

HUANG, Zhenzhen 20568058

XIE, Hao 20568644

CHENG, Qing 20567494

CONTENT

01

Introduction

02

Linear Regression Model Construction

03

Linear Regression Model Diagnostics

04

ANOVA

05

Machine Learning

01

Introduction

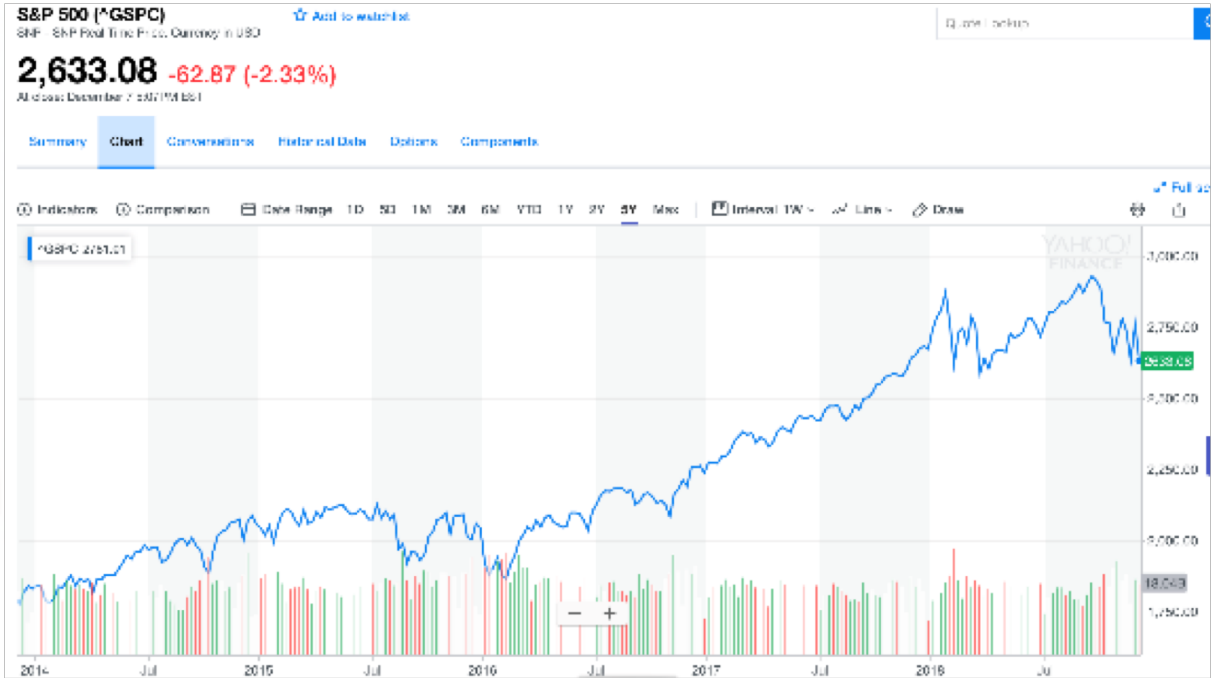
- Background
- Data Exploration
- Data Preprocessing

Background

S&P 500 Index

A free float-adjusted market capitalization-weighted stock market index in the United States.

Record and monitor daily changes of the largest companies of the American stock market.



The movement of S&P 500 in five years

Background

Potential candidate financial indexes

‘FTSE100’: A stock index for London Stock Exchange with 100 companies’ stocks
'NIKKEI225’: A stock index for the Tokyo Stock Exchange
'HSI’: Index of Hangseng
'NASDAQ’: A stock index including common stocks and similar securities listed on the NASDAQ stock market
'SSECOPOSITE(IN USD)‘: A stock market index of all stocks (A shares and B shares) that are traded at the Shanghai Stock Exchange
'DJIA’ : Dow Jones Industrial Average
'USD-GBP’: British Pound-US Dollar exchange rate
'USD-JPY’: Japanese Yen-US Dollar exchange rate
'GOLD PRICE’: Commodity price of gold
'crude oil’’: Commodity price of crude oil

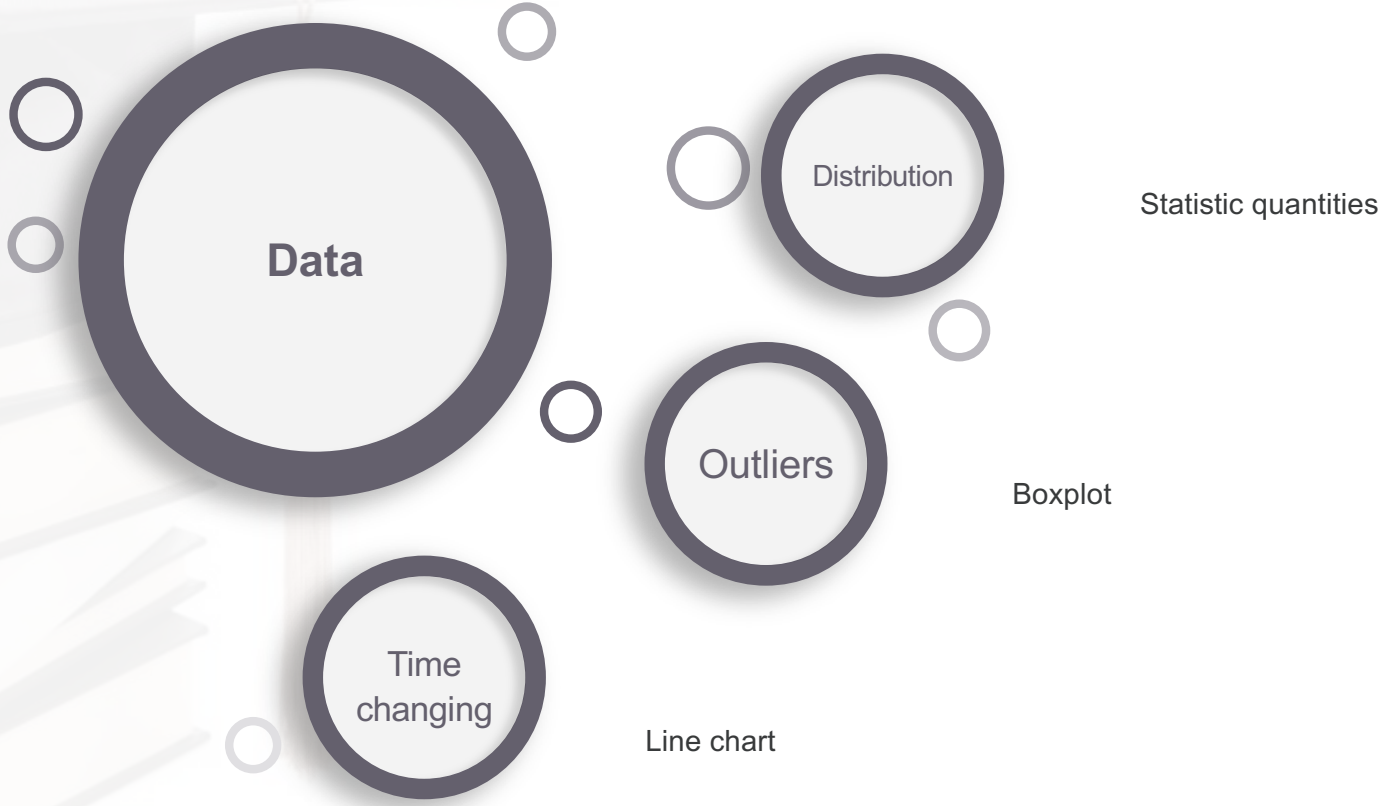
Background

Data: 2010-2018 daily closing value of S&P 500 and popential financial indexes

Purpose:

- To find out relationships among the daily close value of S&P 500 index and other kinds of financial data
- To explore the difference between mean return values from different years
- To predict the up and downs of the index by different statistical methods

Data Exploration



Data Exploration

Financial Indexes

01

The Return of S&P500

02

Relationship Between Financial Indexes and S&P500

03

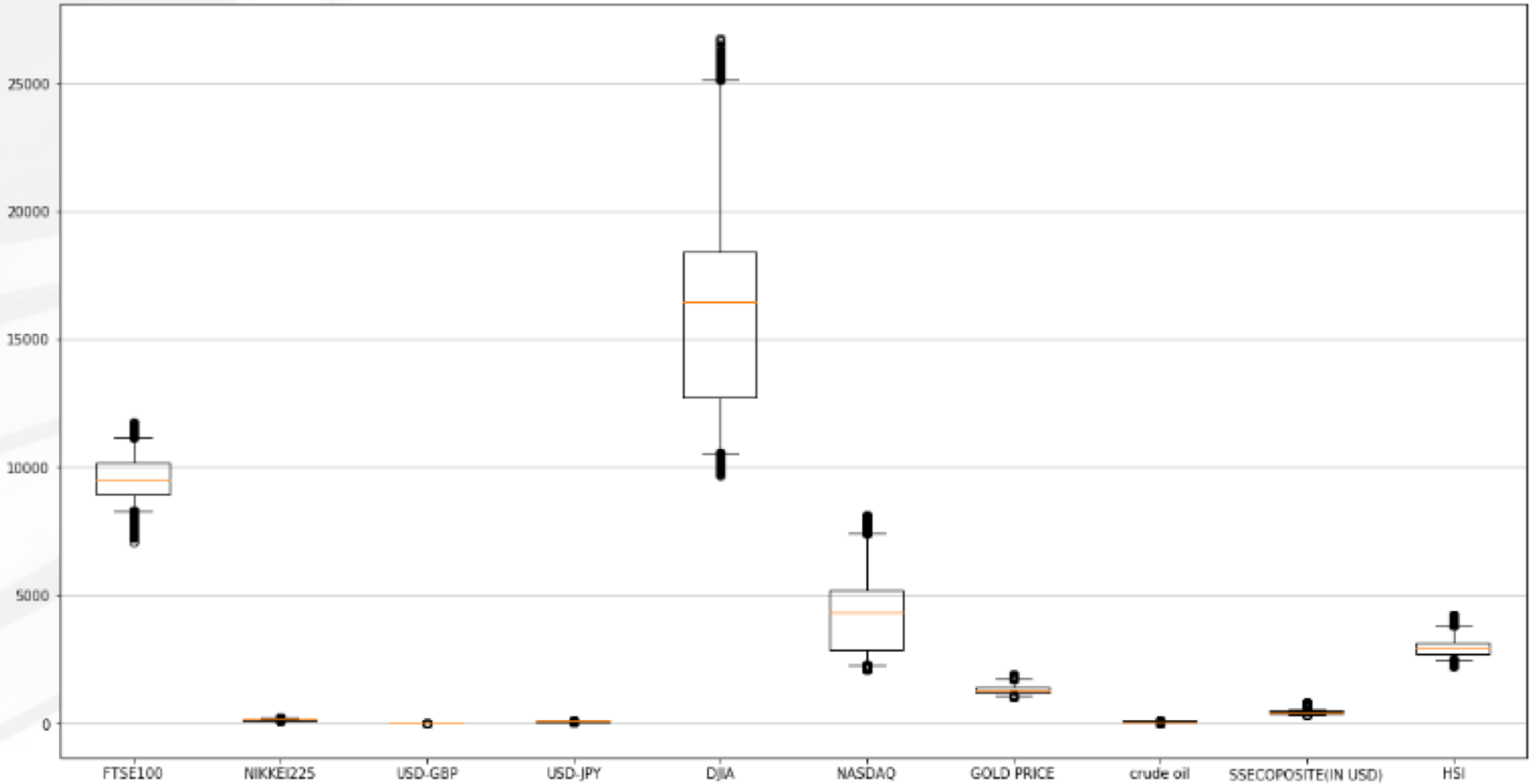
Data Exploration

Data Information of Financial Indexes

	FTSE100	NIKKEI225	USD-GBP	USD-JPY	DJIA	NASDAQ	GOLD PRICE	crude oil	SSECOPOSITE(IN USD)	HSI
mean	9567.89	147.30	0.67	100.29	16581.41	4373.00	1342.74	74.35	431.64	3000.24
std	885.87	29.53	0.06	14.59	4413.28	1615.18	186.13	22.37	85.11	387.04
min	7093.83	103.23	0.58	75.82	9686.48	2091.79	1051.10	26.55	314.68	2231.71
25%	8915.31	118.58	0.63	84.20	12727.38	2893.38	1219.27	51.54	366.72	2735.81
50%	9539.30	146.98	0.65	102.36	16458.11	4351.16	1286.74	77.18	426.34	2950.59
75%	10153.27	165.43	0.72	112.18	18403.60	5218.22	1407.00	94.89	471.24	3157.98
max	11772.46	219.22	0.82	125.63	26743.50	8109.69	1897.46	112.86	832.07	4241.01

Data Exploration

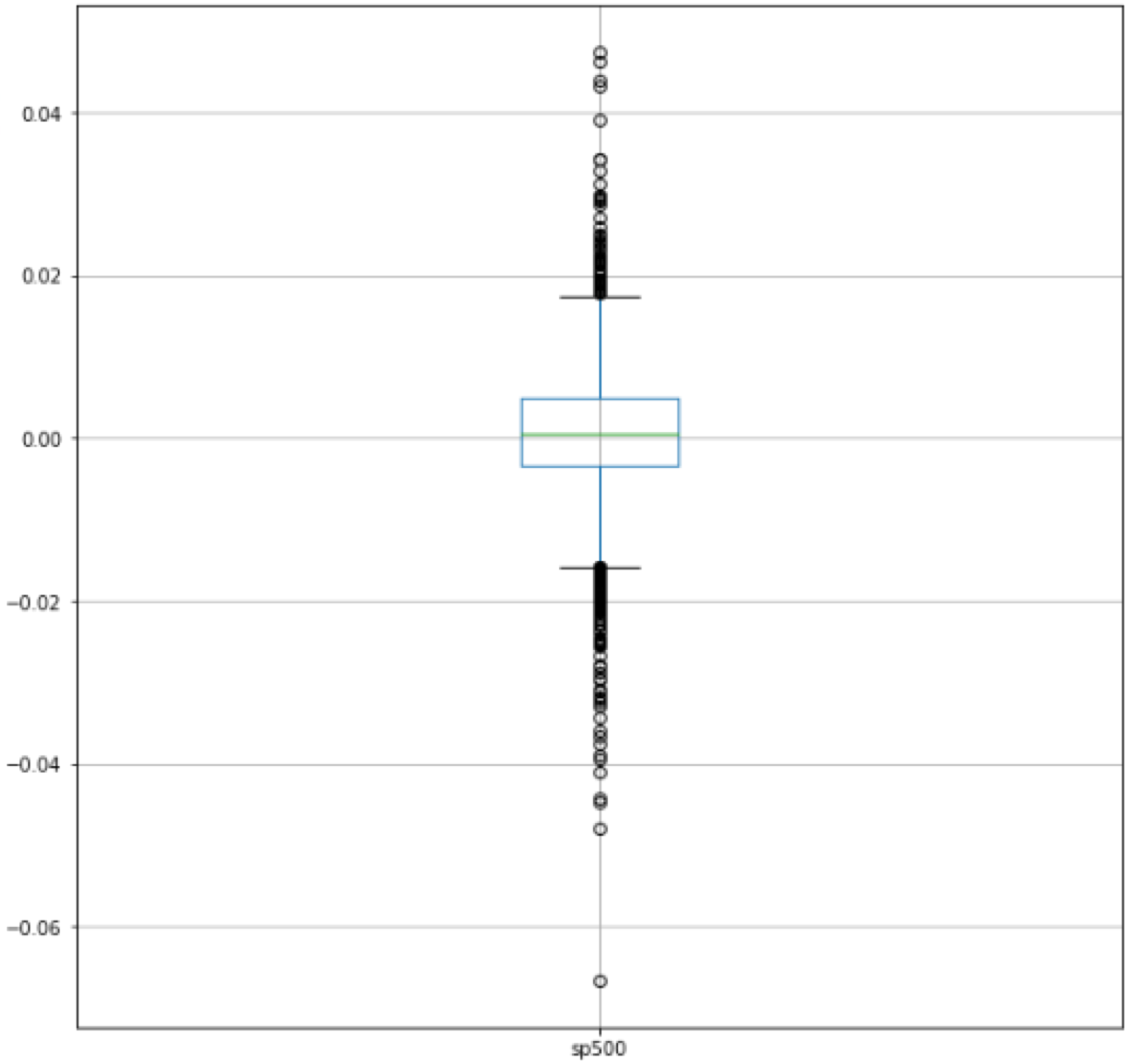
Boxplot of Financial Indexes



Data Exploration

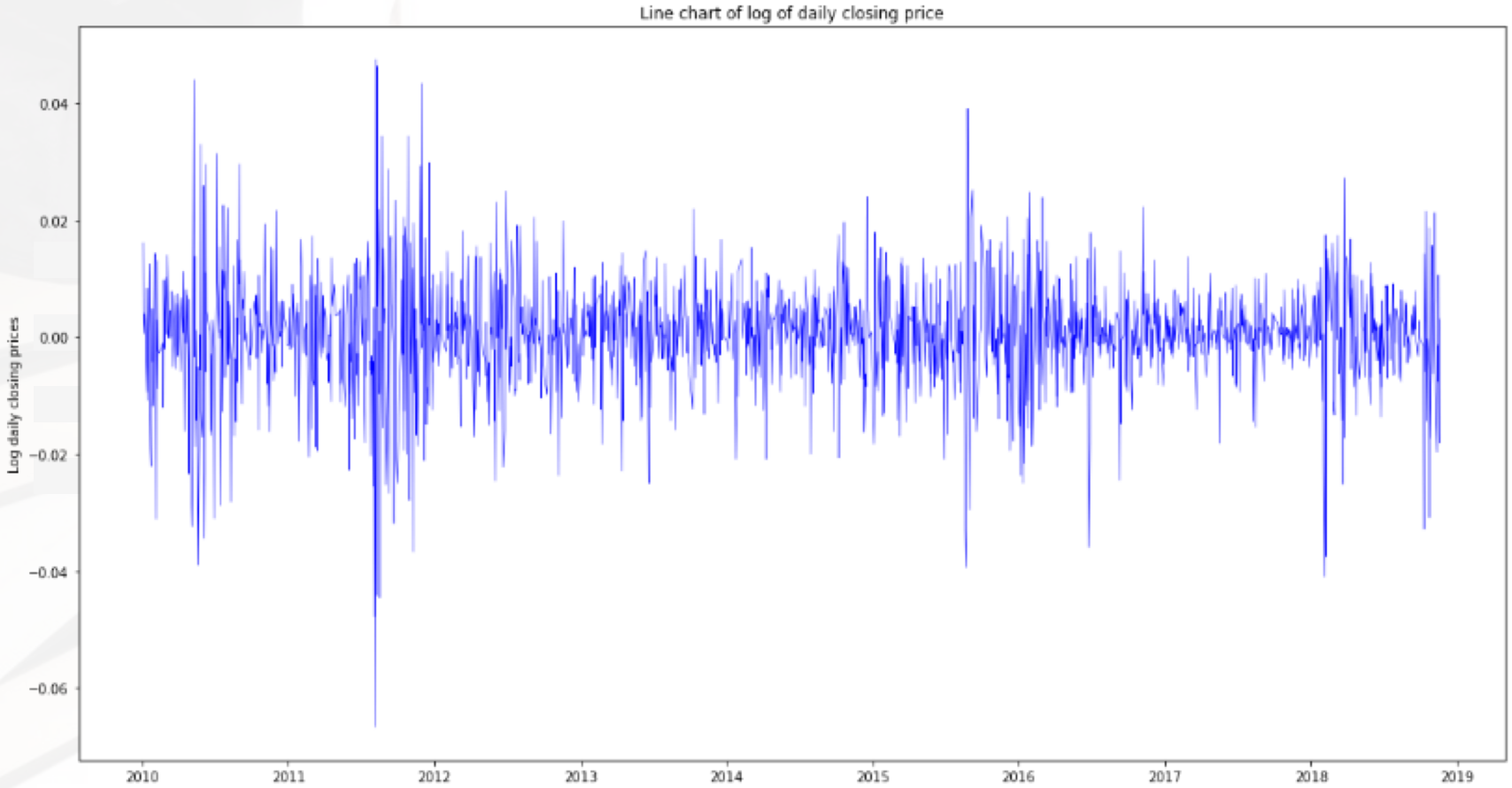
The return of S&P 500

	sp500
mean	0.00
std	0.01
min	-0.07
25%	-0.00
50%	0.00
75%	0.01
max	0.05



Data Exploration

Plot of daily return value of S&P 500 closing price

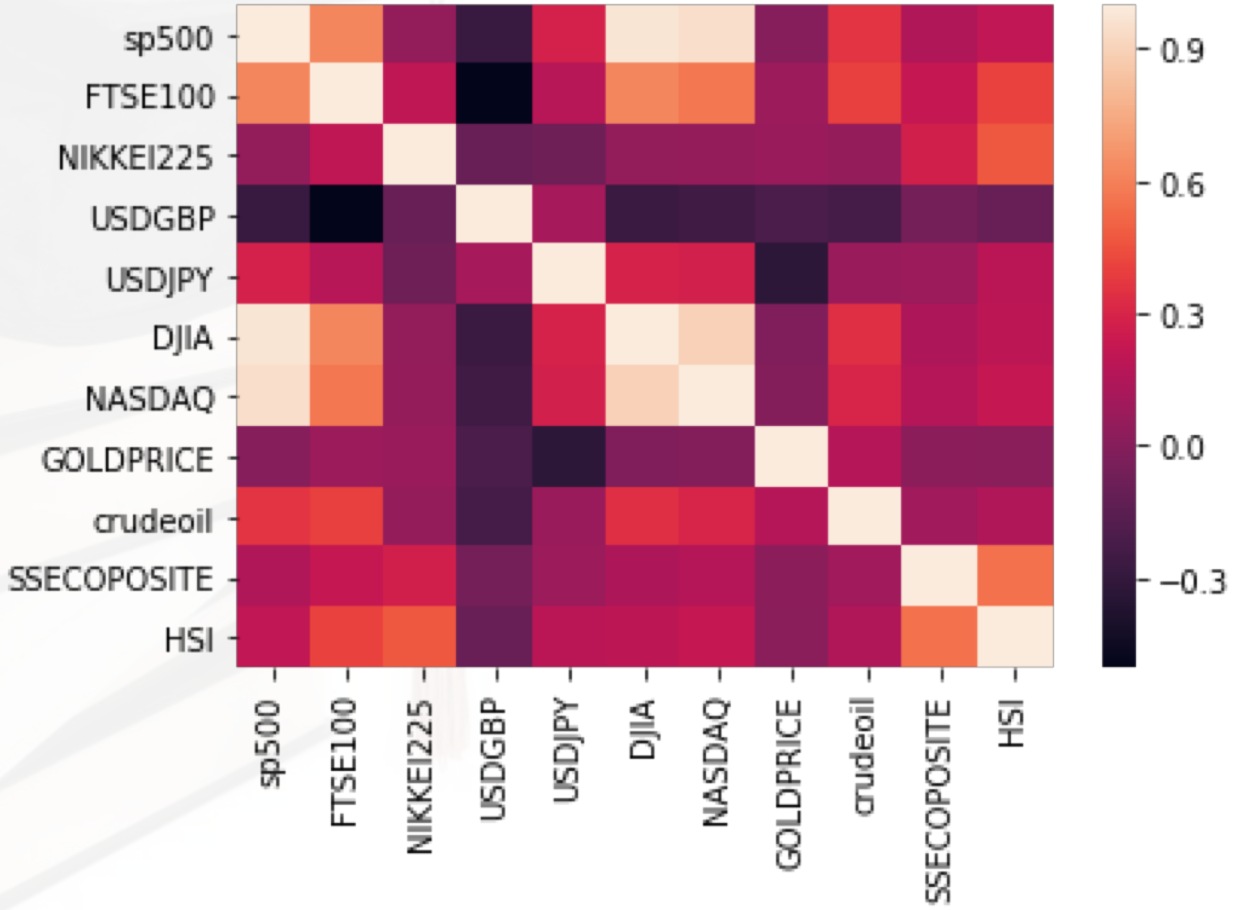


Data Exploration

Correlation between Financial Indexes and S&P500

	sp500	FTSE100	NIKKEI225	USDGBP	USDJPY	DJIA	NASDAQ	GOLDPRICE	crudeoil	SSECOPOSITE	HSI
sp500	1.000000	0.622389	0.053860	-0.278938	0.287327	0.972714	0.951640	0.002853	0.362548	0.143730	0.209138
FTSE100	0.622389	1.000000	0.201320	-0.503433	0.173049	0.619389	0.571856	0.082637	0.403513	0.226160	0.410482
NIKKEI225	0.053860	0.201320	1.000000	-0.101947	-0.082606	0.049759	0.057463	0.075006	0.058270	0.277371	0.475025
USDGBP	-0.278938	-0.503433	-0.101947	1.000000	0.115947	-0.273960	-0.246273	-0.207499	-0.231286	-0.055074	-0.099395
USDJPY	0.287327	0.173049	-0.082606	0.115947	1.000000	0.291179	0.279466	-0.326821	0.077894	0.079287	0.186918
DJIA	0.972714	0.619389	0.049759	-0.273960	0.291179	1.000000	0.896667	-0.016986	0.346903	0.137925	0.198343
NASDAQ	0.951640	0.571856	0.057463	-0.246273	0.279466	0.896667	1.000000	-0.004042	0.305553	0.161116	0.228454
GOLDPRICE	0.002853	0.082637	0.075006	-0.207499	-0.326821	-0.016986	-0.004042	1.000000	0.170512	0.030287	0.014785
crudeoil	0.362548	0.403513	0.058270	-0.231286	0.077894	0.346903	0.305553	0.170512	1.000000	0.098702	0.147558
SSECOPOSITE	0.143730	0.226160	0.277371	-0.055074	0.079287	0.137925	0.161116	0.030287	0.098702	1.000000	0.552965
HSI	0.209138	0.410482	0.475025	-0.099395	0.186918	0.198343	0.228454	0.014785	0.147558	0.552965	1.000000

Data Exploration

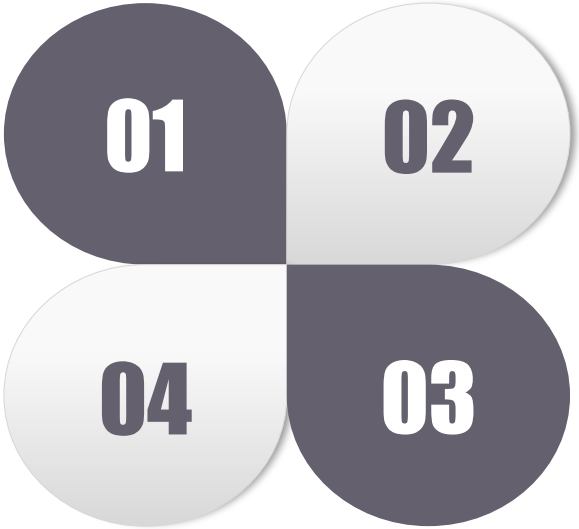


Data Preprocessing

Z-scores normalization

The range of different factors are not in the same magnitude.

Here using z-scores method to normalize the data.



Outlier samples

Feature Selection

Missing Value

Method1: Delete all the missing values.
Method2: Using the mean value to fill the missing values.

02

Linear Regression Model Construction

- Model assumption
- Model construction
- Model selection

Multi-linear regression models

Multi-linear regression model

$$sp500_i = \alpha_1 FTSE100_i + \alpha_2 NIKKEI225_i + \alpha_3 USDGBP_i + \alpha_4 USDJPY_i + \alpha_5 DJIA_i + \alpha_6 NASDAQ_i + \alpha_7 GOLDPRICE_i + \alpha_8 crudeoil_i + \alpha_9 SSECOPOSITE_i + \alpha_{10} HSI_i + \varepsilon_i$$

$sp500_i$: response variable

$FTSE100_i, \dots, HSI_i$: independent variables

$\alpha_1, \dots, \alpha_{10}$: partial regression coefficients

ε: random error.

Multi-linear regression models

- **Model Assumption**

A1: ε have zero mean
A2: ε are independent
A3: ε have a common unknown variance σ^2
A4: ε are normally distributed
A5: There is no linear relationship among the explanatory variables
A6: Linearity: The true relationship between the mean of the response variable and the explanatory variables is linear

Multi-linear regression

Model Construction

```
lreg =
smf.ols("sp500~FTSE100+NI
KKEI225+USDGBP+USDJPY
+DJIA+NASDAQ+GOLDPRIC
E+crudeoil+SSECOPOSITE+
HSI",return_df).fit()
print(lreg.summary())
```

R^2 is 0.998, this Multi-linear model can highly explain SP500. Some explanatory variables are not significant.

OLS Regression Results

Dep. Variable:	sp500	R-squared:	0.998
Model:	OLS	Adj. R-squared:	0.998
Method:	Least Squares	F-statistic:	1.083e+05
Date:	Sun, 09 Dec 2018	Prob (F-statistic):	0.00
Time:	18:16:20	Log-Likelihood:	-8648.1
No. Observations:	1922	AIC:	1.732e+04
Df Residuals:	1911	BIC:	1.738e+04
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-500.8315	24.207	-20.689	0.000	-548.307	-453.356
FTSE100	0.0849	0.002	52.268	0.000	0.082	0.088
NIKKEI225	-0.4654	0.111	-4.211	0.000	-0.682	-0.249
USDGBP	591.6695	21.245	27.849	0.000	550.003	633.336
USDJPY	1.0012	0.146	6.877	0.000	0.716	1.287
DJIA	0.0584	0.001	41.717	0.000	0.056	0.061
NASDAQ	0.1332	0.004	37.316	0.000	0.126	0.140
GOLDPRICE	-0.0065	0.006	-1.107	0.268	-0.018	0.005
crudeoil	-1.7471	0.066	-26.514	0.000	-1.876	-1.618
SSECOPOSITE	0.0199	0.013	1.574	0.116	-0.005	0.045
HSI	-0.1034	0.004	-26.978	0.000	-0.111	-0.096

Omnibus:	171.328	Durbin-Watson:	0.096
Prob(Omnibus):	0.000	Jarque-Bera (JB):	223.815
Skew:	0.752	Prob(JB):	2.51e-49
Kurtosis:	3.729	Cond. No.	1.23e+06

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.23e+06. This might indicate that there are strong multicollinearity or other numerical problems.

Multi-linear regression models

Model selection

method	Best score	variable
AdjR2	0.998229	All Variables
AIC	17316.9	FTSE100, NIKKEI225, DJIA, USDJPY, USDGBP, NASDAQ, crudeoil, HSI
BIC	17366.9	FTSE100, NIKKEI225, DJIA, USDJPY, USDGBP, NASDAQ, crudeoil, HSI

From the best subset selection methods above, we can choose FTSE100, NIKKEI225, DJIA, USDJPY, USDGBP, NASDAQ, crudeoil, HIS as explanatory variables.

Multi-linear regression models

By using ***forward selection****:*

AIC best score: 17316.9
Variables: *FTSE100, NIKKEI225, DJIA, USDJPY, USDGBP, NASDAQ, crudeoil, HSI*

By using ***backward selection****:*

AIC best score: 17316.9
Variables: *FTSE100, NIKKEI225, DJIA, USDJPY, USDGBP, NASDAQ, crudeoil, HSI*

Summary:

Different methods on AIC score are consistent with each other, the final model selected include *FTSE100, NIKKEI225, DJIA, USDJPY, USDGBP, NASDAQ, crudeoil, HSI.*

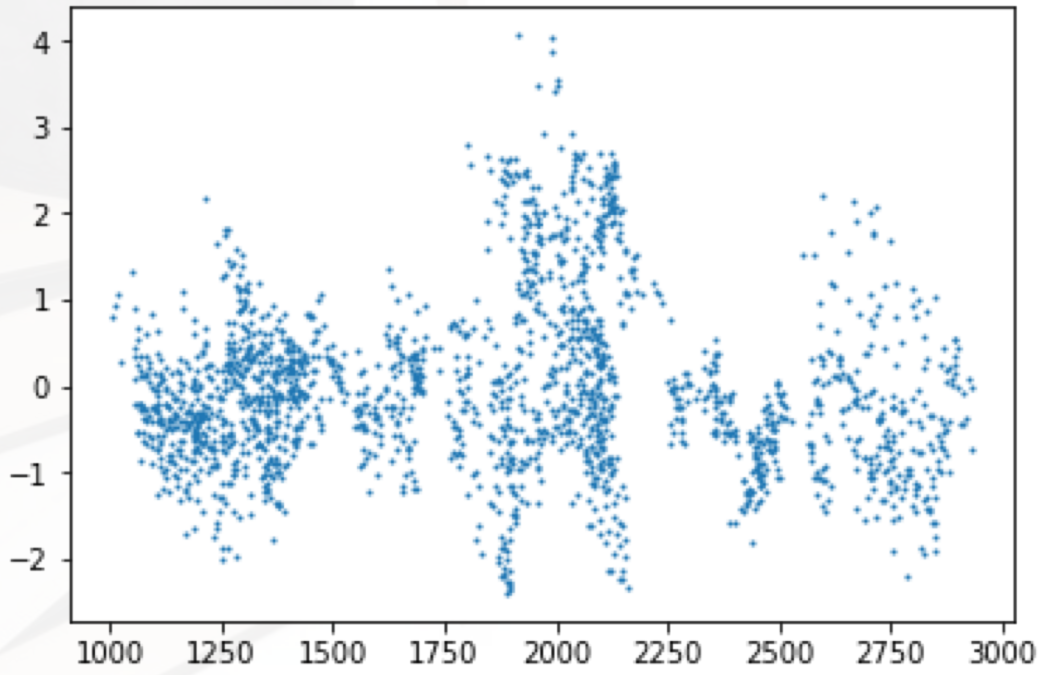
03

Linear Regression Model Diagnostics

- Model diagnostics
- Model remedies

Multi-linear regression models

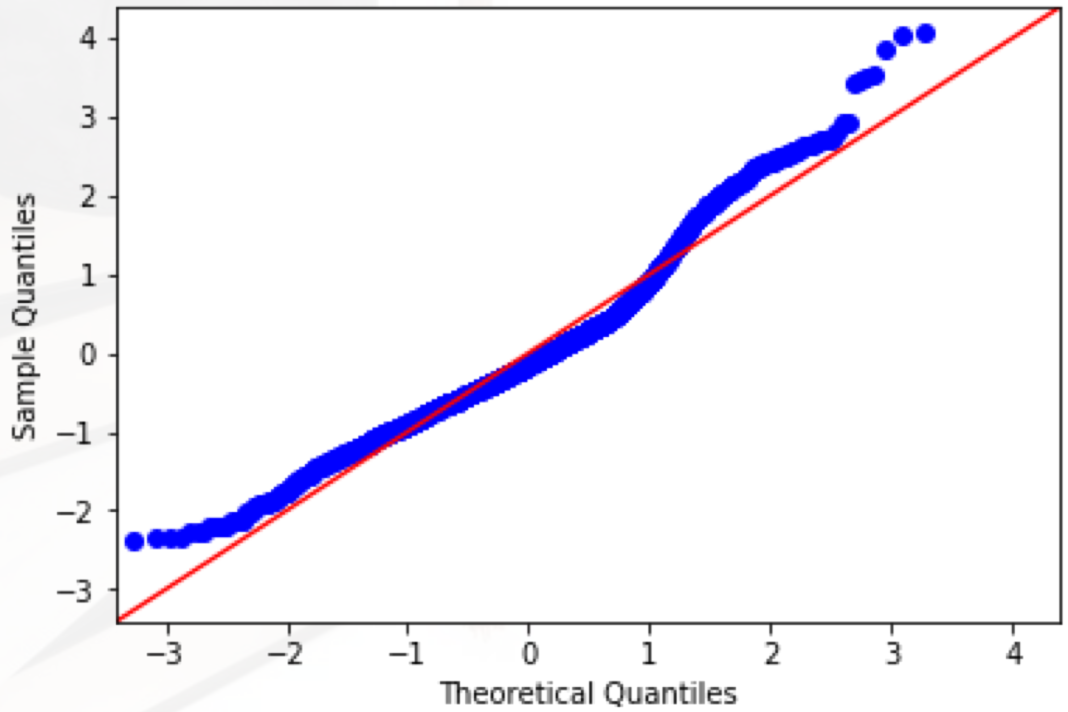
Externally studentized residual plot (A3, assumption of constant variances)



Conclusion: There is some patterns of the externally studentized residual. So the assumption of constant variance should be suspected.

Multi-linear regression models

Q-Q plot: (A4, assumption of normal distribution)



Conclusion:
From the qq plot, we suspect the assumption of a normal distribution.
Then we do normality test of extresid.

Multi-linear regression models

Normality test of externally studentized residual

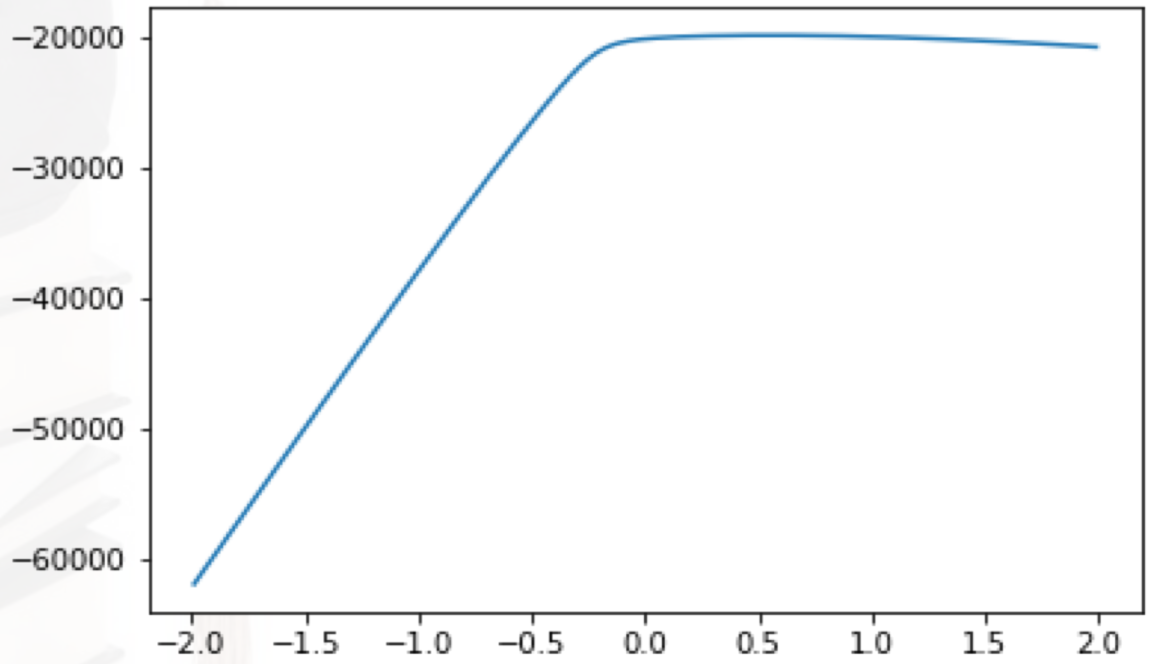
Test method	Statistic	p-value / Critical Value
Shapiro-Wilk test	0.9604	1.7757e-22
K-S test	0.0845	2.1014e-12
Anderson-Darling test	25.0873	[0.575, 0.655, 0.785, 0.916, 1.09] [15. , 10. , 5. , 2.5, 1.]

For Shapiro-Wilk test and K-S test, p-values are smaller than corresponding significant level (0.15, 0.1, 0.05, 0.025, 0.01), so we conclude that the H0 should be rejected.

For the Anderson-Darling test, the statistic is larger than corresponding critical values at different significant level as above. So we draw the same conclusion as previous tests.

Multi-linear regression models

Box-Cox transformation



We choose lambda to be 0.55.

Multi-linear regression models

We fit a new OLS model after Box-Cox transformation

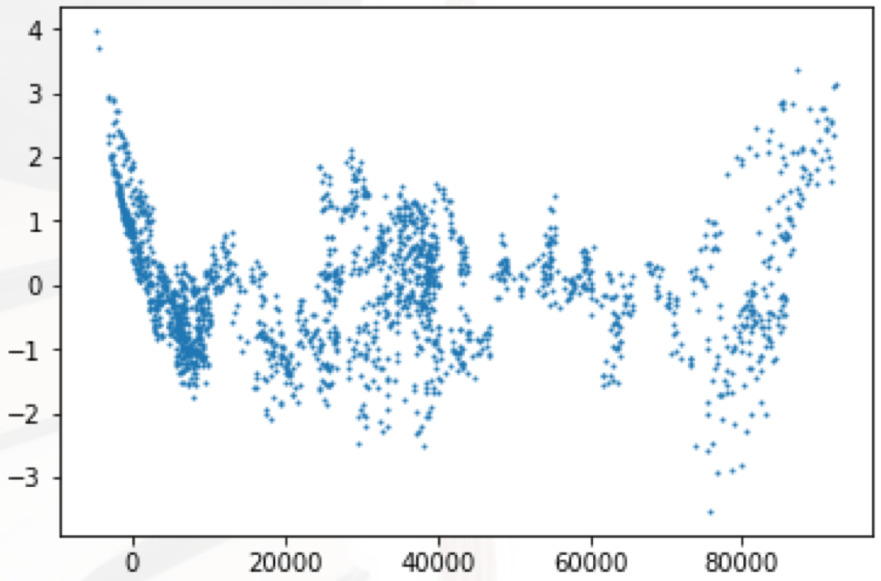
Dep. Variable:	y	**R-squared:**	0.989
Model:	OLS	**Adj. R-squared:**	0.989
Method:	Least Squares	**F-statistic:**	1.754e+04
Date:	Sun, 09 Dec 2018	**Prob (F-statistic):**	0.00
Time:	18:53:08	**Log-Likelihood:**	-4740.0
No. Observations:	1922	**AIC:**	9502.
Df Residuals:	1911	**BIC:**	9563.
Df Model:	10		
Covariance Type:	nonrobust		

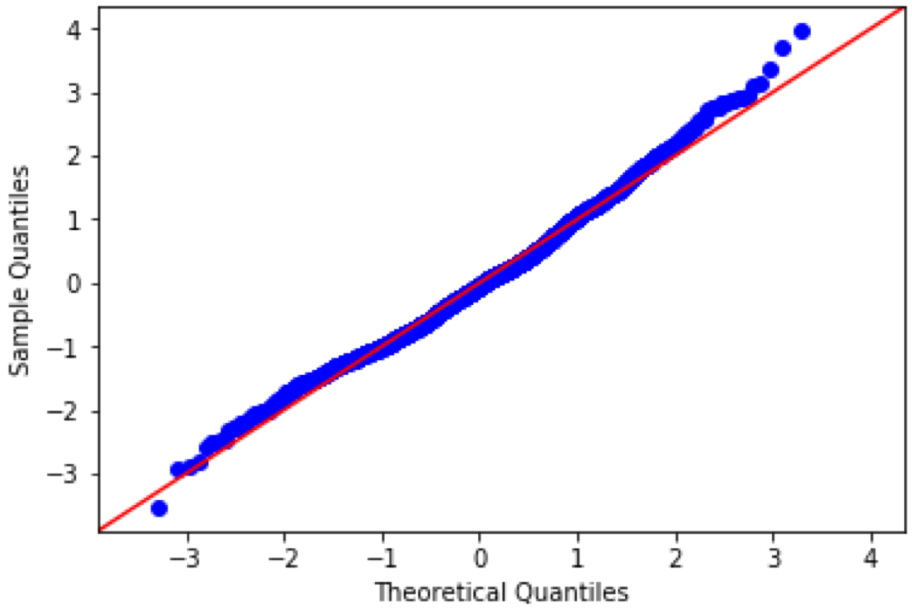
	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-135.6707	3.169	-42.816	0.000	-141.885	-129.456
FTSE100	0.0108	0.000	50.729	0.000	0.010	0.011
NIKKEI225	-0.0184	0.014	-1.271	0.204	-0.047	0.010
USDGBP	71.3208	2.781	25.645	0.000	65.867	76.775
USDJPY	0.1811	0.019	9.505	0.000	0.144	0.219
DJIA	0.0035	0.000	18.907	0.000	0.003	0.004
NASDAQ	0.0042	0.000	9.034	0.000	0.003	0.005
GOLDPRICE	0.0118	0.001	15.233	0.000	0.010	0.013
crudeoil	-0.1504	0.009	-17.435	0.000	-0.167	-0.133
SSECOPOSITE	0.0097	0.002	5.876	0.000	0.006	0.013
HSI	-0.0162	0.001	-32.359	0.000	-0.017	-0.015

Omnibus:	12.847	**Durbin-Watson:**	0.110
Prob(Omnibus):	0.002	**Jarque-Bera (JB):**	15.168
Skew:	-0.121	**Prob(JB):**	0.000509
Kurtosis:	3.362	**Cond. No.**	1.23e+06

Multi-linear regression models

Externally studentized residual plot and Q-Q plot after box_cox transform:





Multi-linear regression models

Normality test of extresid after B-C transformation

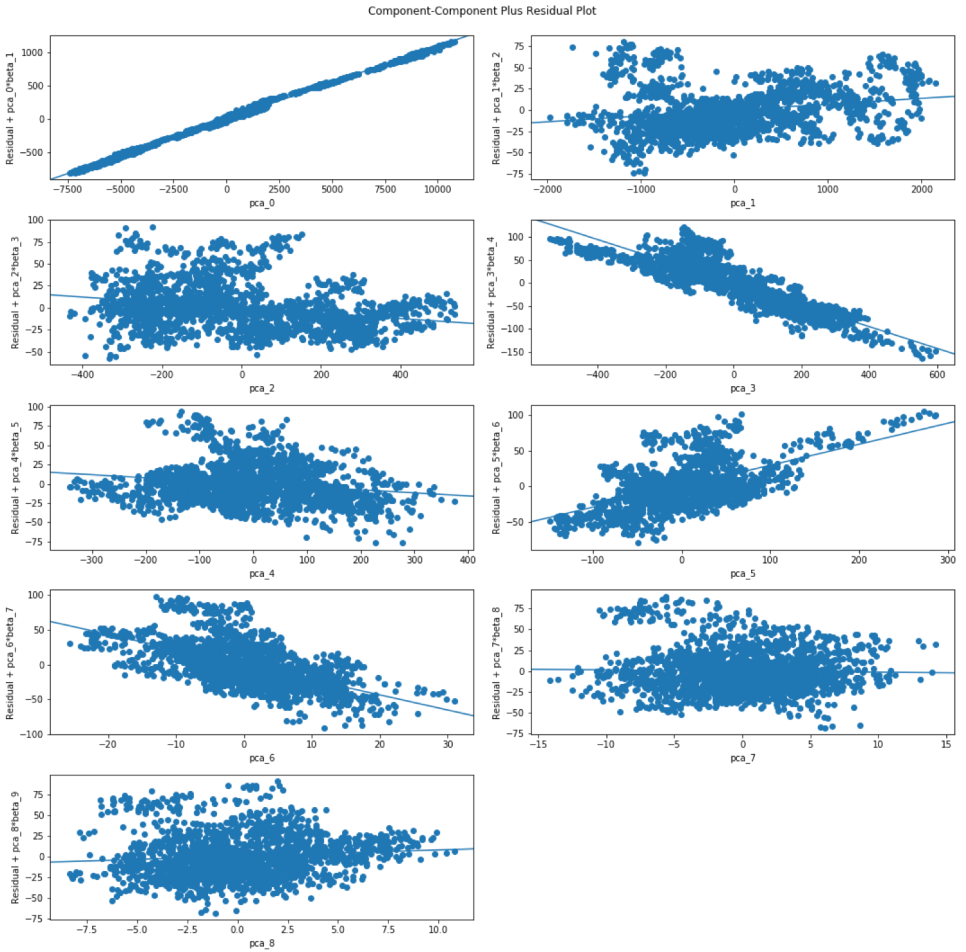
Test method	statistic	p-value
Shapiro-Wilk test	0.9914	3.1467e-9
K-S test	0.0397	0.0045
Anderson-Darling test	4.5914	[0.575, 0.655, 0.786, 0.916, 1.09] [15. , 10. , 5. , 2.5, 1.]

It shows that the transformation doesn’t work.

Multi-linear regression models

Assessing nonlinearity (A6)

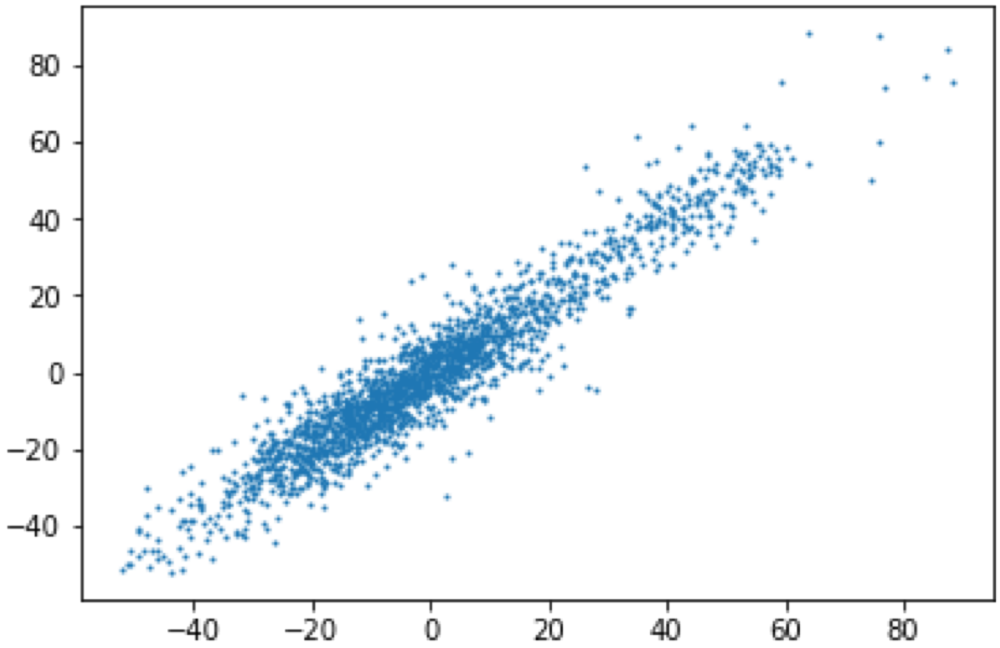
The relationships between some variables and the response are not linear.



Multi-linear regression models

Diagnostic independence (A1)

Durbin-Watson Statistic: 0.0958



Multi-linear regression models

Remedy – Time series model

- We can check if the process we are dealing with is a stationary process by checking the existence of unit roots.
- Here we use Augmented Dickey-Fuller (ADF) test for this purpose.
- We can also check if the process is a white noise. Because white noise will not provide any useful message.
- We will use Ljung-Box test to achieve this goal.

Multi-linear regression models

We make use of the package “arch”. The result below shows that the return of S&P 500 is (weak) stationary.

In [13]:
```
# unit root test
# for non-stationary series, we can difference many times to get a staionary series
# A time series is stationary or not can be checked by lag operator Lx_t=x_{t-1}
# for instance, consider 1st order: (1-L)x_t=\epsilon_t, the root of 1-L is 1
# the norm of 1 is not > 1, so it is not stationary
# if all roots are > 1 (doesn't exist unit root), then stationary
# use ADF test to check this phenomenon
from arch.unitroot import ADF
adfRet=ADF(Ret)
# return a class
print(adfRet.summary().as_text())
# the statistic is smaller than critical value, then reject null hypothesis (it is stationary)
# it is stationary, as the conclusion before
```

```
True
   Augmented Dickey-Fuller Results
=====================================
Test Statistic                -11.256
P-value                         0.000
Lags                               24
-------------------------------------

Trend: Constant
Critical Values: -3.43 (1%), -2.86 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

Multi-linear regression models

```
adfClose=ADF(Close)
print(adfClose.summary().as_text())
# not stationary
```

```
True
   Augmented Dickey-Fuller Results
=====================================
Test Statistic                 -0.484
P-value                         0.895
Lags                               19
-------------------------------------

Trend: Constant
Critical Values: -3.43 (1%), -2.86 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

```
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

- The close price as a process is not stationary (p-value = 0.895).

- We consider the first order difference and then it becomes stationary. (p-value is almost zero)

```
Closediff = np.diff(np.array(sp500['Close']))
Closediff = pd.DataFrame(Closediff).dropna()
Closediff = np.array(Closediff)
adfClosediff=ADF(Closediff)
print(adfClosediff.summary().as_text())
# it is stationary
```

```
True
   Augmented Dickey-Fuller Results
=====================================
Test Statistic                -12.735
P-value                         0.000
Lags                               18
-------------------------------------

Trend: Constant
Critical Values: -3.43 (1%), -2.86 (5%), -2.57 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

Multi-linear regression models

We make use of the package “statsmodels”. The Ljung-Box test shows that the return process is not a pure random process. Because of a very low p-value 3.2374496435720 33e-05. Besides, the first order difference process is also not a pure random process under 0.1 or 0.05 significance level.

```
# next we check if it is pure random
# use LB test
LjungBox1=stattools.q_stat(stattools.acf(Ret)[1:13],len(Ret))
LjungBox1[1][-1]
# second row is p-values, which should be considered
# very low p-value, so it is not a white noise series
```

Out[26]: 3.237449643572033e-05

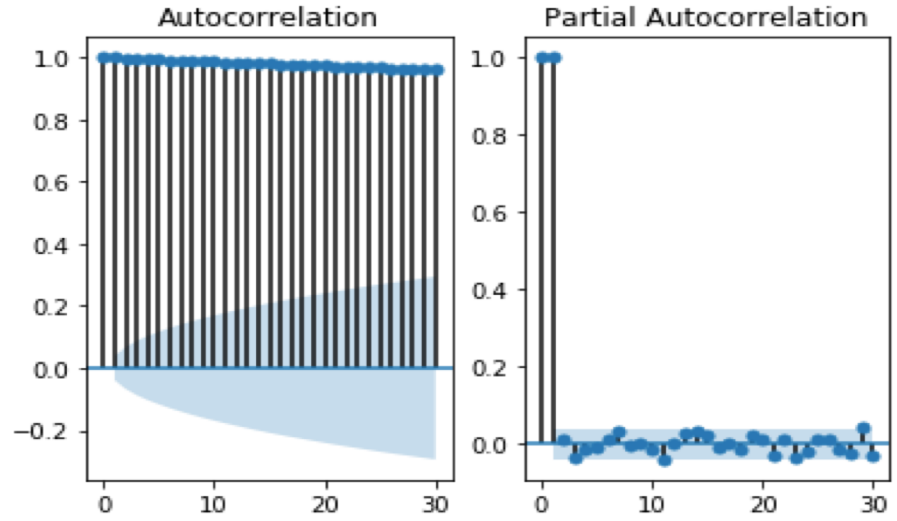
```
In [33]: LjungBox2=stattools.q_stat(stattools.acf(Closediff)[1:13],len(Closediff))
         LjungBox2[1][-1]
         # p-value small, so not pure random, which means maybe autocorrelated
```

Out[33]: 0.015481117618448956

Multi-linear regression models

- We give the ACF and PACF plots of the close price process below.

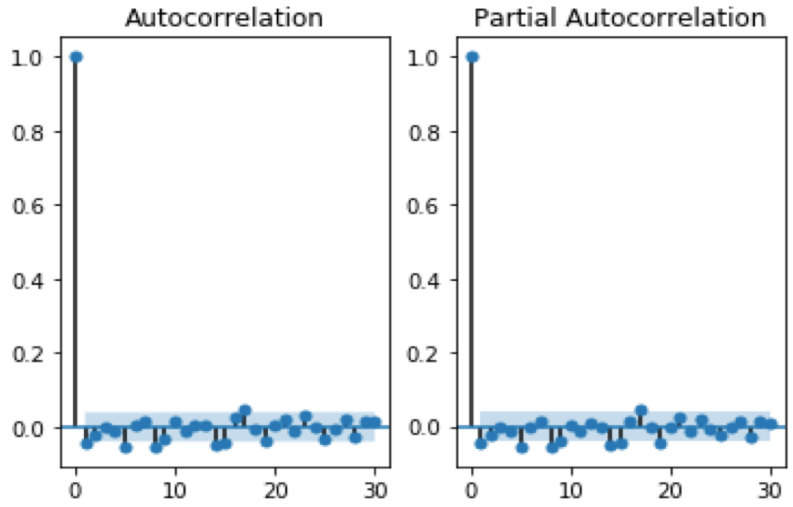
```
In [29]: # modelling
from statsmodels.graphics.tsaplots import *
axe1=plt.subplot(121)
axe2=plt.subplot(122)
plot1=plot_acf(Close,lags=30,ax=axe1)
plot2=plot_pacf(Close,lags=30,ax=axe2)
```



Multi-linear regression models

- We give the ACF and PACF plots of the first order difference of the close price process below.

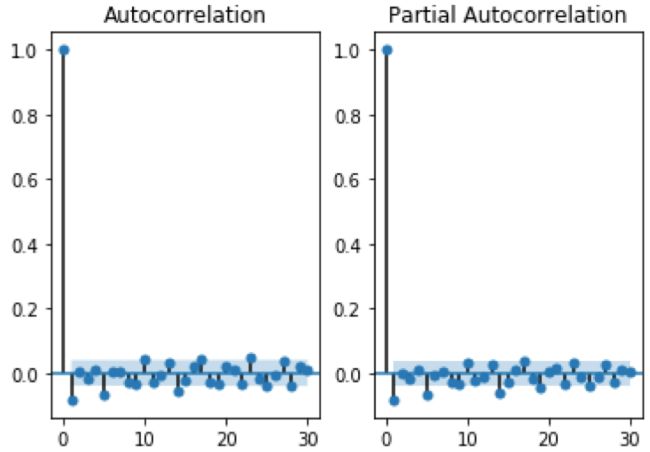
```
In [31]: # plotting above results
         axe1=plt.subplot(121)
         axe2=plt.subplot(122)
         plot1=plot_acf(Closediff,lags=30,ax=axe1)
         plot2=plot_pacf(Closediff,lags=30,ax=axe2)
```



Multi-linear regression models

- We give the ACF and PACF plots of the return process below.

```
]: # plotting above results
   axe1=plt.subplot(121)
   axe2=plt.subplot(122)
   plot1=plot_acf(Ret,lags=30,ax=axe1)
   plot2=plot_pacf(Ret,lags=30,ax=axe2)
```



Multi-linear regression models

- We can try a lot of time series methods
- For example, we can try ARIMA(1,1,0)

ARIMA Model Results

Dep. Variable:	D.Close	**No. Observations:**	2515
Model:	ARIMA(1, 1, 0)	**Log Likelihood**	-10557.138
Method:	css-mle	**S.D. of innovations**	16.099
Date:	Sun, 09 Dec 2018	**AIC**	21120.276
Time:	22:12:25	**BIC**	21137.766
Sample:	1	**HQIC**	21126.624

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.6939	0.308	2.253	0.024	0.090	1.298
ar.L1.D.Close	-0.0421	0.020	-2.110	0.035	-0.081	-0.003

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	-23.7810	+0.0000j	23.7810	0.5000

Multi-linear regression models

Multicollinearity (A5)

Checking VIFs : 2362.6357, 8.3445, 42.9418, 7.2746, 18.1869,
153.8309, 133.9914, 4.8760, 8.7599, 4.6698, 8.8738

There are many variables that have a VIF exceeding 10,
so we drop the one with the largest VIF and fit a new model.

After dropping the variable with the largest VIF and get new VIFs:
2243.0302, 42.4006, 4.4940, 11.0908, 153.8268,
133.9413, 4.3872, 6.9141, 4.4210, 7.4071

After dropping the variable with the largest VIF several times,
the situation doesn't get better, so we turn to other methods.

Multi-linear regression models

Principal component regression:

Accumulate variance ratios are:
[0.9656883 0.99527633 0.9973852
 0.99917563 0.99984036 0.99999518
 0.99999862 0.99999952 1.   ]
for all the principal components.

Check VIFs after PCA:
[1.0, 0.9999999999999998,
 1.0, 1.0, 0.9999999999999998,
 1.0000000000000002, 1.0, 1.0, 1.0, 1.0]

No multicollinearity after PCA (obvious)

doing linear regression on pca component:

OLS Regression Results

Dep. Variable:	sp500	R-squared:	0.998
Model:	OLS	Adj. R-squared:	0.998
Method:	Least Squares	F-statistic:	8.554e+04
Date:	Sun, 09 Dec 2018	Prob (F-statistic):	0.00
Time:	20:25:20	Log-Likelihood:	-8975.4
No. Observations:	1922	AIC:	1.797e+04
Df Residuals:	1912	BIC:	1.803e+04
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1852.5071	0.590	3138.039	0.000	1851.349	1853.665
pca_0	0.1091	0.000	872.317	0.000	0.109	0.109
pca_1	0.0069	0.001	9.650	0.000	0.005	0.008
pca_2	-0.0307	0.003	-11.456	0.000	-0.036	-0.025
pca_3	-0.2372	0.003	-81.621	0.000	-0.243	-0.231
pca_4	-0.0393	0.005	-8.250	0.000	-0.049	-0.030
pca_5	0.2955	0.010	29.899	0.000	0.276	0.315
pca_6	-2.1682	0.066	-32.665	0.000	-2.298	-2.038
pca_7	-0.1388	0.129	-1.073	0.284	-0.393	0.115
pca_8	0.7608	0.177	4.290	0.000	0.413	1.109

Omnibus:	181.822	Durbin-Watson:	0.049
Prob(Omnibus):	0.000	Jarque-Bera (JB):	241.145
Skew:	0.780	Prob(JB):	4.33e-53
Kurtosis:	3.761	Cond. No.	4.72e+03

Multi-linear regression models

Ridge regression (penalty for coefficient)

In order to deal with multicollinearity,

we use ridge regression, which will

shrink some coefficients towards zero.

(coefficients are shown in the right)

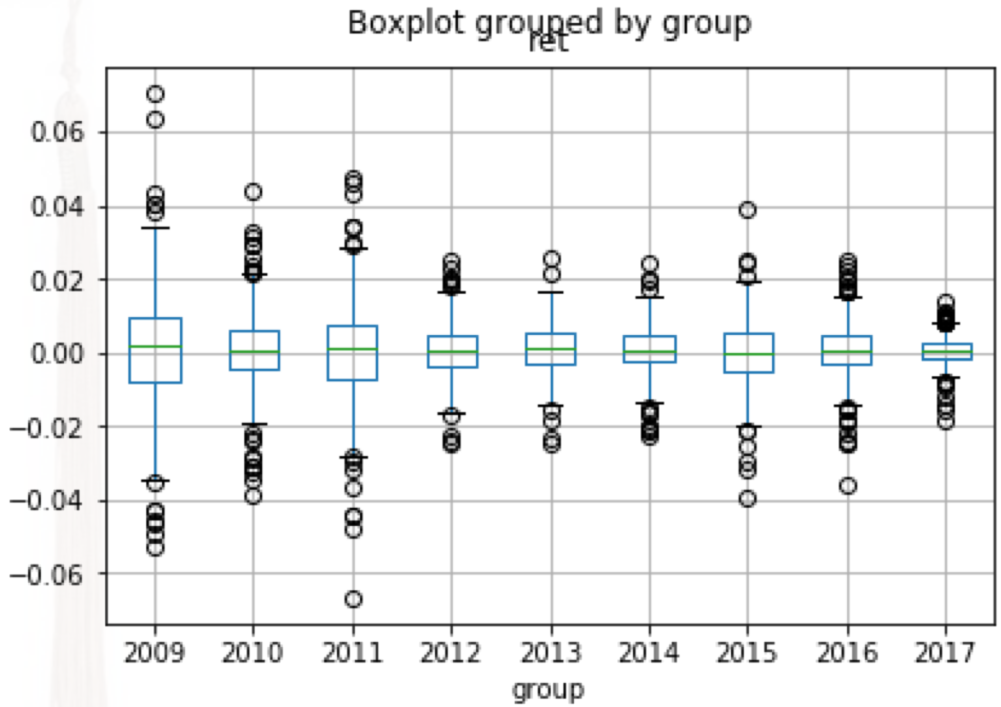
```
array([[-4.98517850e+02],
       [ 8.48273970e-02],
       [-4.65192060e-01],
       [ 5.89627062e+02],
       [ 9.97873628e-01],
       [ 5.84420487e-02],
       [ 1.33185113e-01],
       [-6.71973876e-03],
       [-1.74856783e+00],
       [ 1.98315533e-02],
       [-1.03443505e-01]])
```

04

ANOVA

- Boxplot by year
- Hypothesis test of group means
- Model diagnostics

ANOVA



Graphically we observe the same means for the 9 groups, although many outliers exist.

ANOVA

Source	d.f.	Sum of Squares (SS)	Mean sum of squares (MS)	F-value
Treatment / Between	8	0.000222	0.00002775	0.254139456
Error / Within	2255	0.246228	0.000109192	
Total	2263	0.24645		

According to Python output, P ($F_{8,2255} > F$) = 0.979876 > α = 0.05, so we do not reject H_0 at the 95% confidence level, and conclude that the group means are equal.

ANOVA

Sample Means for Each Group

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.0010	0.001	1.492	0.136	-0.000	0.002
C(group)[T.2010]	-0.0004	0.001	-0.474	0.636	-0.002	0.001
C(group)[T.2011]	-0.0009	0.001	-0.939	0.348	-0.003	0.001
C(group)[T.2012]	-0.0004	0.001	-0.479	0.632	-0.002	0.001
C(group)[T.2013]	7.022e-05	0.001	0.075	0.940	-0.002	0.002
C(group)[T.2014]	-0.0005	0.001	-0.568	0.570	-0.002	0.001
C(group)[T.2015]	-0.0010	0.001	-1.035	0.301	-0.003	0.001
C(group)[T.2016]	-0.0006	0.001	-0.631	0.528	-0.002	0.001
C(group)[T.2017]	-0.0003	0.001	-0.286	0.775	-0.002	0.002

ANOVA

Calculation of Sample Means

$\widehat{\mu_1}$ = 0.0010

$\widehat{\mu_2}$ = 0.0010 -0.0004 = 0.0006

$\widehat{\mu_3}$ = 0.0010 -0.0009 = 0.0001

$\widehat{\mu_4}$ = 0.0010 -0.0004 = 0.0006

$\widehat{\mu_5}$ = 0.0010 +7.022e-05 = 0.00107022

$\widehat{\mu_6}$ = 0.0010 -0.0005 = 0.0005

$\widehat{\mu_7}$ = 0.0010 -0.0010 = 0.0000

$\widehat{\mu_8}$ = 0.0010 -0.0006 = 0.0004

$\widehat{\mu_9}$ = 0.0010 -0.0003 = 0.0007

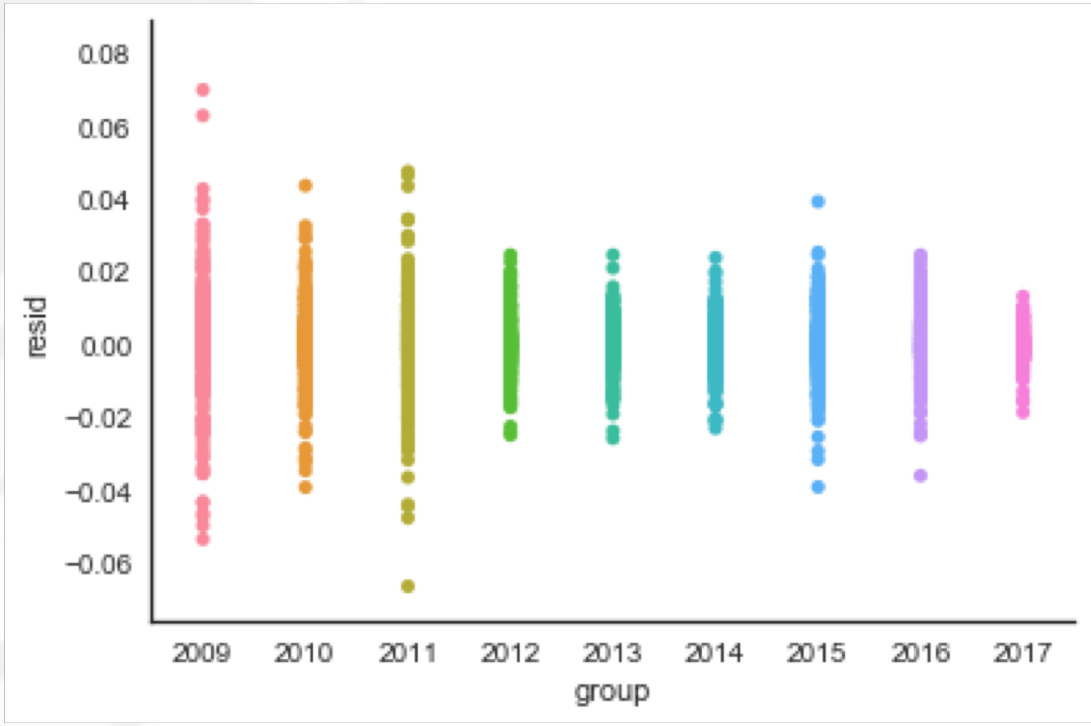
ANOVA

Confidence Intervals for Model Parameters (alpha =0.01)

	[0.005	0.995]
Intercept	-0.001	0.003
C(group)[T.2010]	-0.003	0.002
C(group)[T.2011]	-0.003	0.002
C(group)[T.2012]	-0.003	0.002
C(group)[T.2013]	-0.002	0.002
C(group)[T.2014]	-0.003	0.002
C(group)[T.2015]	-0.003	0.001
C(group)[T.2016]	-0.003	0.002
C(group)[T.2017]	-0.003	0.002

ANOVA

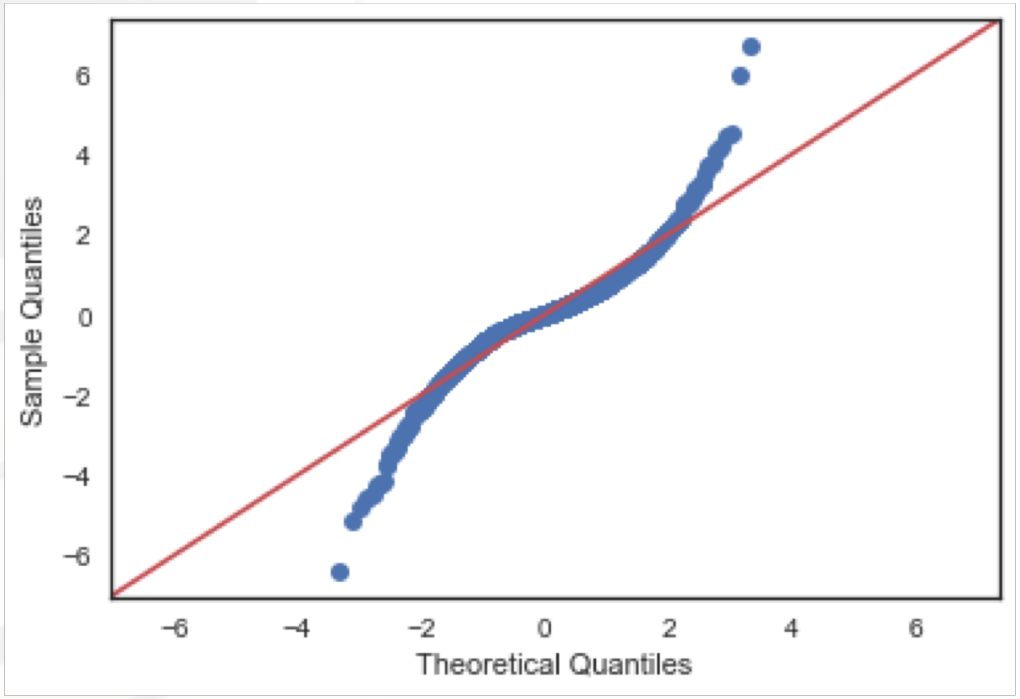
Diagnostics -- Residual Plot



The plot shows that the residuals are distributed around zero line, implying that they have the same zero mean. However, the dispersion pattern indicates unequal variance.

ANOVA

Diagnostics -- Q-Q plot



Normality assumption is seriously violated in this case. The result will be verified by some numerical tests later.

ANOVA

Diagnostics -- Homogeneity of Variances Tests

According to Python output, we have the following results:

```
Levene test (mean)
38.510021041269006 8.3255471777776147e-58
Levene test (median)
37.94558405086633 5.84889194331322e-57
```

→ Levene test of mean and median strong indicates unequal variance among groups. The p-value is so small that even at 99% significant level we would reject the homogeneous variance assumption.

ANOVA

Diagnostics -- Normality Tests

```
# Fligner-Killeen test
Hf, pf = stats.fligner(*groups)
print("Fligner-Killeen test")
print(Hf, pf)
```

```
Fligner-Killeen test
265.88355284643035 7.358511558939317e-53
```

```
# (ii) For the assumption of normal distribution
# Normality test of res
# Shapiro-Wilk test
shapiro(res)
```

```
(0.9269900321960449, 7.302852694031527e-32)
```

```
# K-S test
z = (res - np.mean(res))/np.std(res)
kstest(z, cdf = 'norm')
```

```
KstestResult(statistic=0.9999999999879949, pvalue=0.0)
```

Normality fails in both tests.

ANOVA

Remedy -- Kruskal test

```
# Normality fails --> use Kruskal-Wallis test
# Kruskal-Wallis test to compare medians
# Find unique group labels and their corresponding indices
label, idx = np.unique(i1['group'], return_inverse=True)

# Make a list of arrays containing the y-values corresponding to each unique label
group_new = [i1['ret'][idx == j] for j, l in enumerate(label)]
print(friedmanchisquare(*group_new))
```

```
FriedmanchisquareResult(statistic=5.191466666667111, pvalue=0.7369296460182552)
```

The p-value is 0.7369, so we do not reject the null hypothesis that normality holds.

05

Machine Learning

- feature select
- train_test_split
- different classfiers
- results comparasion
- further work

features select

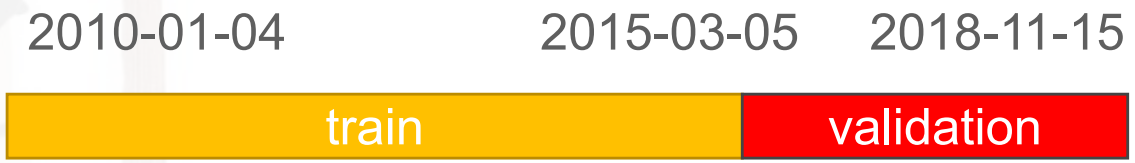
index	day 1	...	day t	day t+1	...	day n
SP500	R_{SP500}	...	R_{SP500}	R_{SP500}	...	R_{SP500}
FTSE100	$R_{FTSE100}$	...	$R_{FTSE100}$	$R_{FTSE100}$	...	$R_{FTSE100}$
NIKKEI225	$R_{NIKKEI225}$	...	$R_{NIKKEI225}$	$R_{NIKKEI225}$	...	$R_{NIKKEI225}$
USDGBP	R_{USDGBP}	...	R_{USDGBP}	R_{USDGBP}	...	R_{USDGBP}
USDJPY	R_{USDJPY}	...	R_{USDJPY}	R_{USDJPY}	...	R_{USDJPY}
DJIA	R_{DJIA}	...	R_{DJIA}	R_{DJIA}	...	R_{DJIA}
NASDAQ	R_{NASDAQ}	...	R_{NASDAQ}	R_{NASDAQ}	...	R_{NASDAQ}
GOLD	R_{GOLD}	...	R_{GOLD}	R_{GOLD}	...	R_{GOLD}
Crude Oil	$R_{Crudeoil}$	...	$R_{Crudeoil}$	$R_{Crudeoil}$	...	$R_{Crudeoil}$
SSE	R_{SSE}	...	R_{SSE}	R_{SSE}	...	R_{SSE}
HSI	R_{HSI}	...	R_{HSI}	R_{HSI}	...	R_{HSI}

features

label

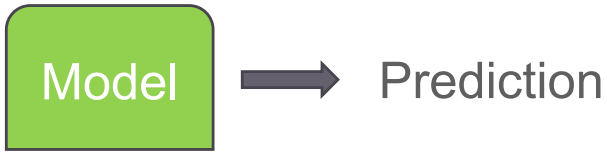
day t+1	label
$R_{SP500}>0$	1
$R_{SP500}<0$	0

train_test_split



sequential

sliding window



Logistic Regression

Accuracy score	55.13%

predict all the label to be 1.
55.13% is just the ratio of rise in validation data
Logistic Regression might not be suitable in this problem.

SVM

kernel	accuracy
rbf	55.13%
linear	55.13%
poly	55.13%
sigmoid	55.13%

svm might not be suitable

K-means

Assumption : n_clusters=2 in Kmeans
one for rise, one for fall

cluster	ratio of rise
1	54.27%
2	54.11%

insignificant difference, asumption might fail

Random Forest

Choosing n_estimators = 100

depth	accuracy
2	54.78%
3	55.65%
4	55.83%
5	54.09%
6	55.30%

choosing depth = 4, accuracy = 55.83%

XGBoost

choosing n_estimators = 100

depth	accuracy
2	54.78%
3	55.65%
4	54.61%
5	54.43%
6	54.96%

choosing depth = 3, accuracy=55.65%

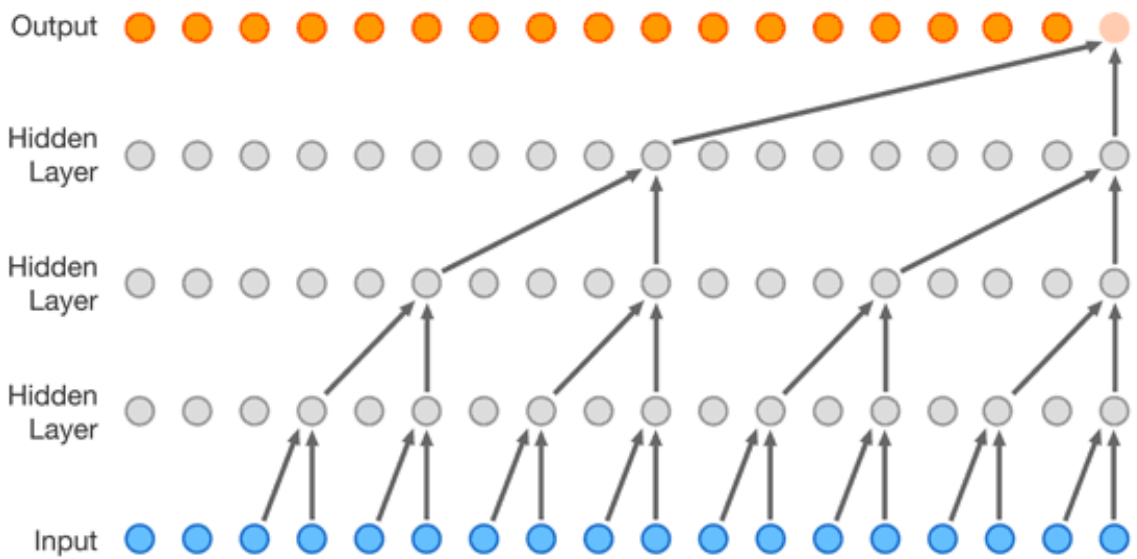
no much difference from random forest

Multi-Layer Perception

solver	accuracy
lbfgs	55.30%
sgd	55.13%
adam	55.13%

choosing solver = lbfgs,accuracy=55.30%
other solver fails

CNN



Convolution

index	feature	day1	...	dayt	...	day t+ window	...	dayn
SP500	open							
	high							
	low							
	close							
FTSE100	open							
	high							
	low							
	close							

embeded length = 8

filter_length = 16

Accuracy

Accuracy score	54.51%

Confusion Matrix

confusion matrix	predict fall	predict rise
Actual fall	0.172	0.706
Actual rise	0.226	0.808

Different Classifier

Type	model	result
Linear	Logistic Regression	fail
	SVM	all kernel fail
Non-Linear	K-means	fail
	Random Forest	accuracy: 55.83%
	XGBoost	accuracy: 55.65%
	Multi-Layer Perception	lbfgs accuracy: 55.30%
	CNN	accuracy:54.51%

Conclusion: Non-Linear model might be more suitable

further work

1、increase time window length

2、add some indicator, like william percentage

3、increase sample volume

4、downsample

Thanks!