```
Saranpat Prasertthum (655667271)
IE598 MLF F18
Module 5 Homework (Dimensionality Reduction)
Use the Treasury Yield Curve dataset
```

Out[1]:

Click here to toggle on/off the raw code.

Out[3]:

Date	SVENF01	SVENF02	SVENF03	SVENF04	SVENF05	SVENF06	SVENF07	SVENF08	S
2019-05-17	2.1224	2.0266	2.1023	2.2377	2.3790	2.5042	2.6069	2.6885	
2019-05-16	2.1239	2.0317	2.1096	2.2468	2.3901	2.5171	2.6217	2.7049	
2019-05-15	2.0874	1.9956	2.0844	2.2289	2.3736	2.4980	2.5984	2.6779	
2019-05-14	2.1319	2.0559	2.1451	2.2856	2.4257	2.5461	2.6428	2.7188	
2019-05-13	2.1051	2.0234	2.1180	2.2632	2.4051	2.5248	2.6198	2.6940	

5 rows × 31 columns

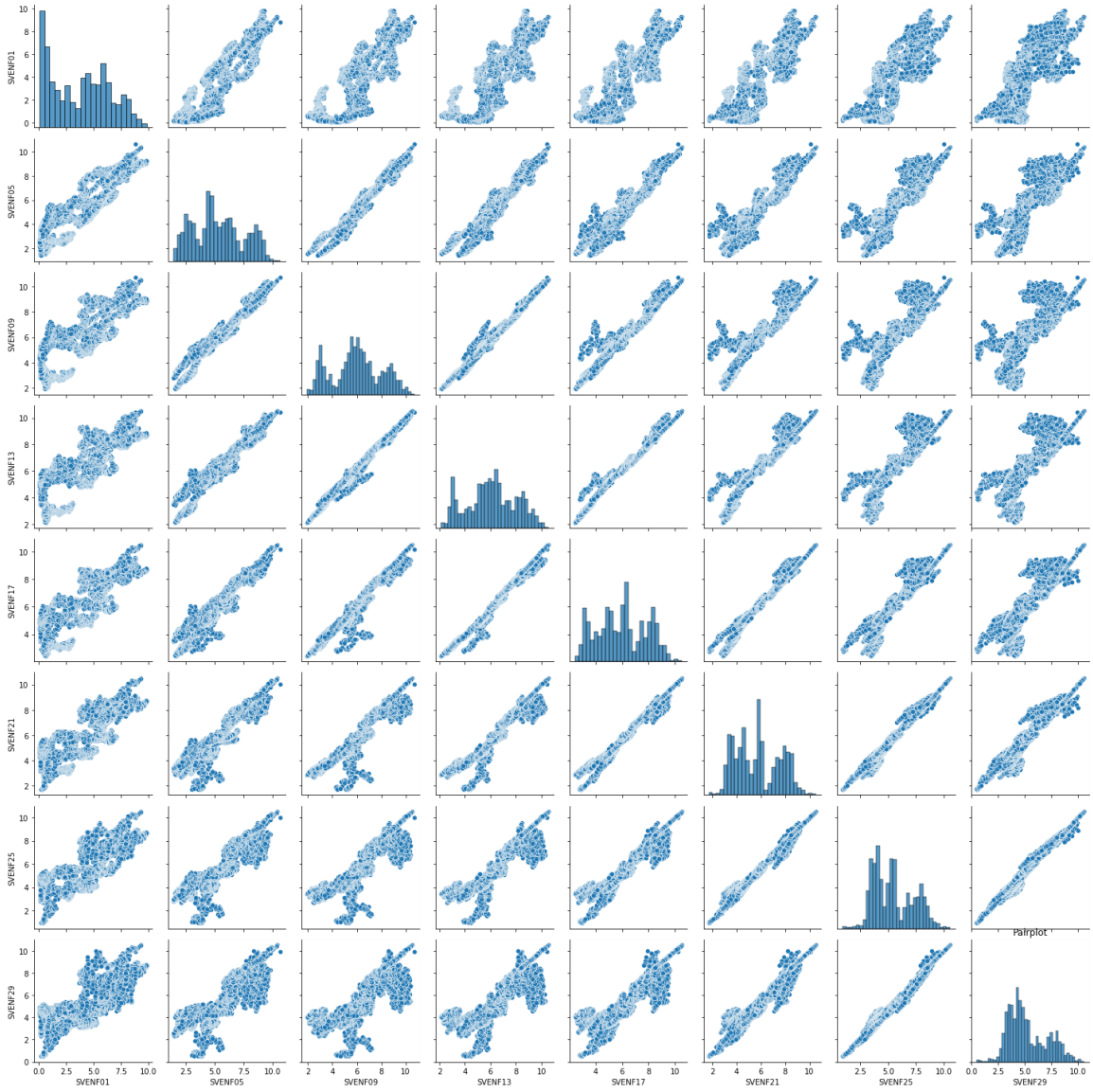
Part 1: Exploratory Data Analysis

Explore datatype of each cols

Out[5]:

	float
SVENF01	8071
SVENF02	8071
SVENF03	8071
SVENF04	8071
SVENF05	8071
SVENF06	8071
SVENF07	8071
SVENF08	8071
SVENF09	8071
SVENF10	8071
SVENF11	8071
SVENF12	8071
SVENF13	8071
SVENF14	8071
SVENF15	8071
SVENF16	8071
SVENF17	8071
SVENF18	8071
SVENF19	8071
SVENF20	8071
SVENF21	8071
SVENF22	8071
SVENF23	8071
SVENF24	8071
SVENF25	8071
SVENF26	8071
SVENF27	8071
SVENF28	8071
SVENF29	8071
SVENF30	8071
Adj_Close	8071

Scatter plot: Pair plot



Data frame shape

```
Number of Columns : 31
Number of Rows : 8071
```

Statistical Summaries

Out[8]:

	SVENF01	SVENF02	SVENF03	SVENF04	SVENF05	SVENF06	SVE
count	8071.000000	8071.000000	8071.000000	8071.000000	8071.000000	8071.000000	8071.0
mean	3.785311	4.258972	4.669363	5.022430	5.318493	5.559644	5.7
std	2.648060	2.498137	2.341348	2.221632	2.137801	2.080405	2.0
min	0.072700	0.327300	0.630300	1.013000	1.424500	1.698200	1.8
25%	1.144050	1.865600	2.536550	3.023050	3.544700	4.063300	4.4
50%	3.986500	4.393300	4.505500	4.718900	5.051300	5.394600	5.6
75%	5.901500	6.221250	6.461300	6.626600	6.779550	6.908050	7.0
max	9.813800	9.887800	10.145600	10.459900	10.649900	10.741400	10.7

8 rows × 31 columns

More Statistical Summaries

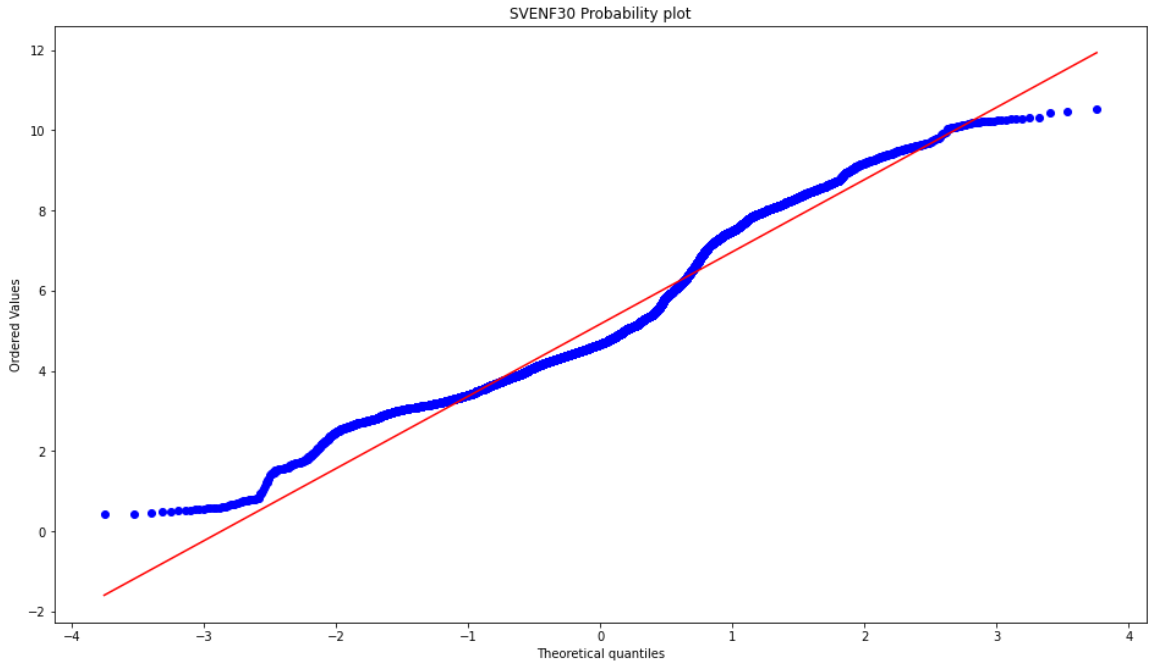
Out[9]:

	SVENF01	SVENF02	SVENF03	SVENF04	SVENF05	SVENF06	SVENF07	SVENF0
Boundary for 10 percentile	0.3326	1.0690	1.7507	2.2505	2.5029	2.7015	2.8794	2.981
Boundary for 20 percentile	0.8002	1.5791	2.2666	2.7630	3.0774	3.2498	3.4406	3.697
Boundary for 30 percentile	1.5379	2.2918	2.9355	3.6452	4.1877	4.5532	4.8027	4.995
Boundary for 40 percentile	2.6503	3.1340	3.7994	4.2409	4.6082	4.9288	5.2072	5.423
Boundary for 50 percentile	3.9865	4.3933	4.5055	4.7189	5.0513	5.3946	5.6637	5.870
Boundary for 60 percentile	4.7050	5.0117	5.4079	5.6254	5.8145	5.9761	6.1366	6.276
Boundary for 70 percentile	5.6197	5.8859	6.0442	6.2370	6.4225	6.5767	6.7085	6.842
Boundary for 80 percentile	6.2687	6.6430	6.9640	7.3057	7.5645	7.7646	7.8642	7.926
Boundary for 90 percentile	7.5553	7.8737	8.1212	8.3588	8.5194	8.6279	8.7289	8.791
Boundary for 100 percentile	9.8138	9.8878	10.1456	10.4599	10.6499	10.7414	10.7663	10.747

10 rows × 31 columns

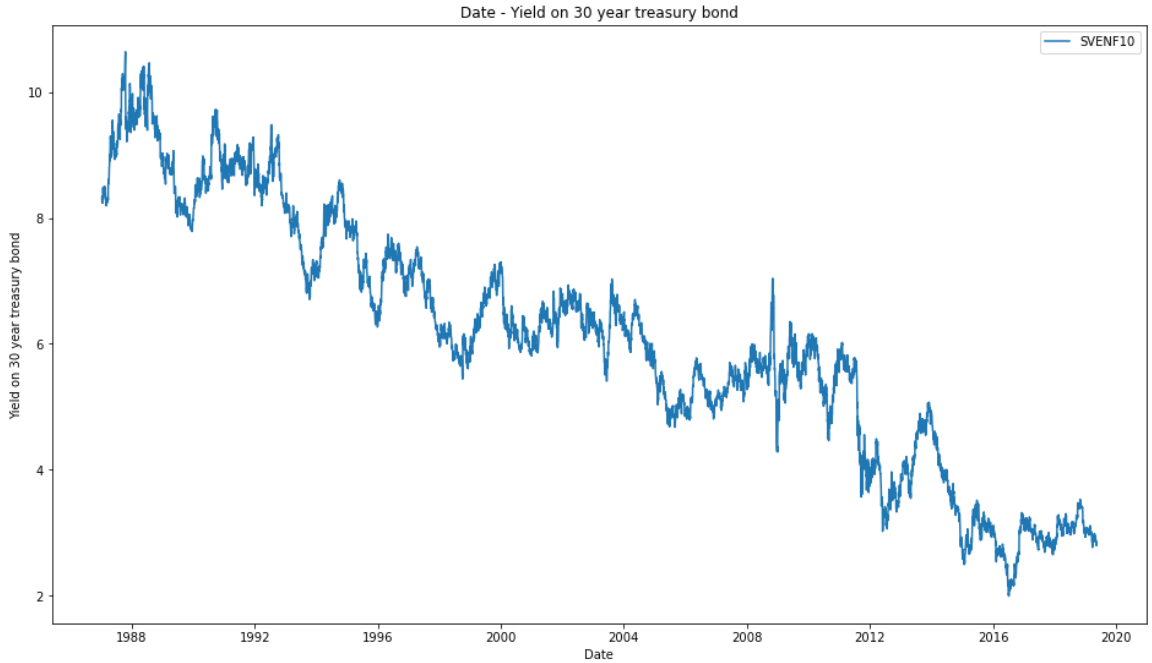
Q-Q plot

showing if the data is gaussian

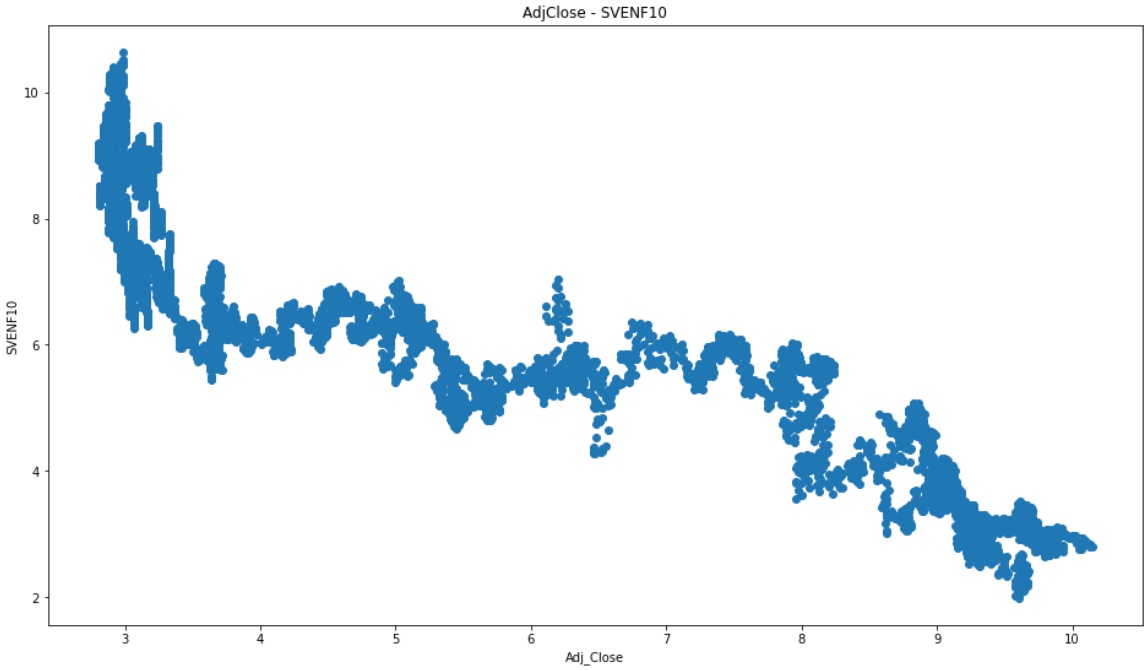



```
SVENF30 does not look Gaussian (reject H0)

C:\Users\SARAN\Anaconda3\lib\site-packages\scipy\stats\morestats.py:1760:
UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
```

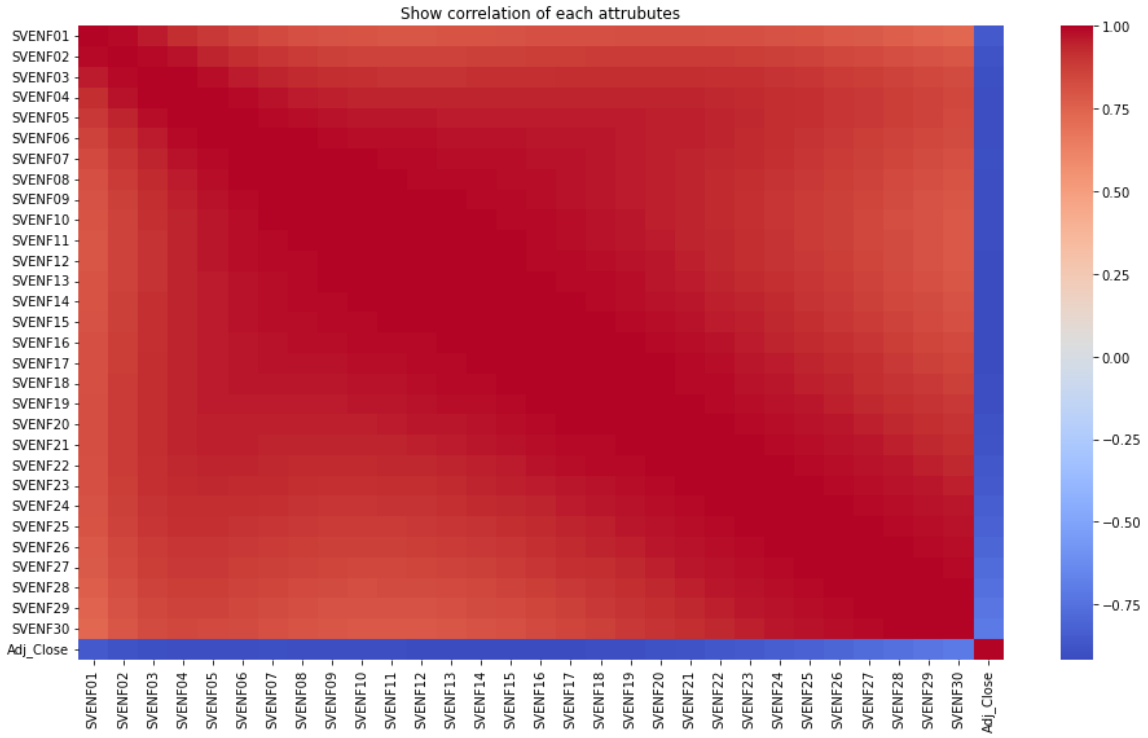
Ploting SVENF10



Cross ploting SVENF10 - AdjClose



Heat map plot



Create train test set

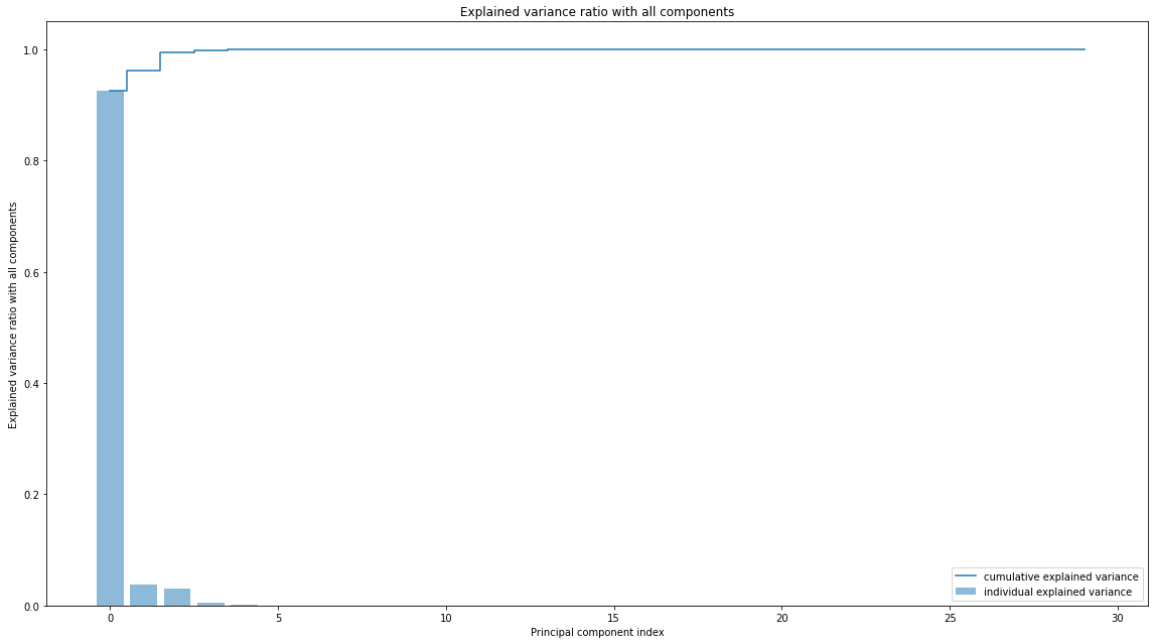
Part 2: Perform a PCA on the Treasury Yield dataset

Expained variences on all components

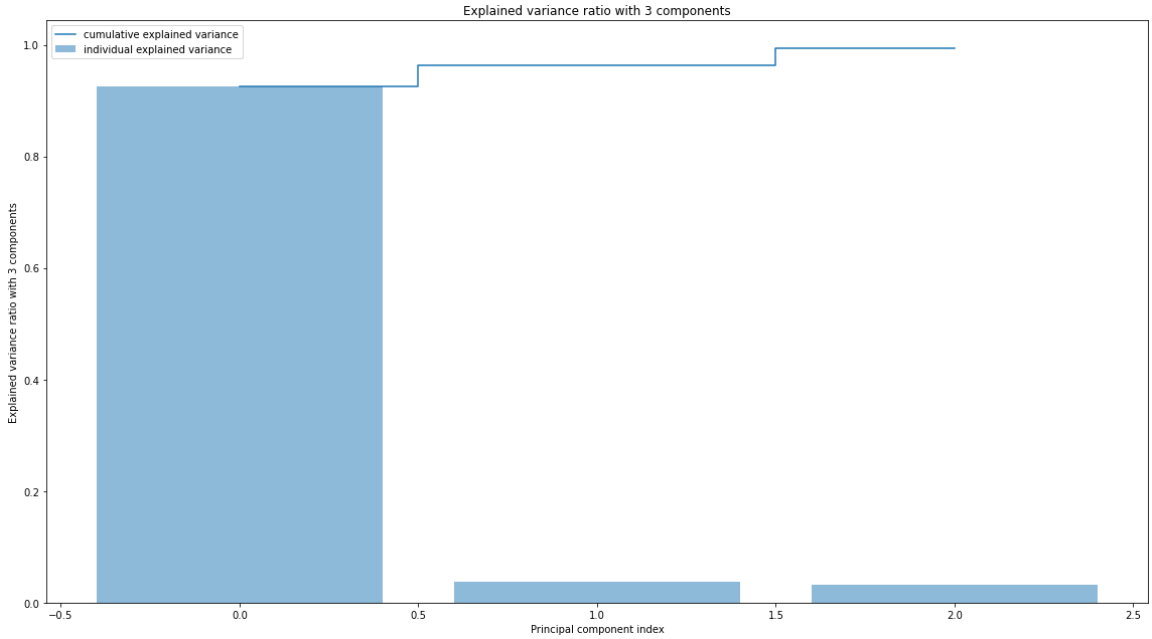
```
Explained Var Ratio with all component:
 [9.25027254e-01 3.77198563e-02 3.11962115e-02 5.11829721e-03
 8.45006479e-04 8.14071111e-05 1.06386900e-05 1.23073879e-06
 8.99497477e-08 7.14094977e-09 4.89071592e-10 3.83422436e-11
 8.63162713e-12 7.54060102e-12 7.44722038e-12 7.41409677e-12
 7.37633844e-12 7.36922042e-12 7.21033060e-12 7.16011018e-12
 7.08499808e-12 7.01615861e-12 6.97953948e-12 6.83297854e-12
 6.78790385e-12 6.76011093e-12 6.68796631e-12 6.63106214e-12
 6.57322725e-12 6.42225375e-12]

Culmulative Var Ratio with all component:
 [0.92502725 0.96274711 0.99394332 0.99906162 0.99990663 0.99998803
 0.99999867 0.9999999  0.99999999 1.         1.         1.
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.        ]

----------------------------------------------------------------------
Explained Var Ratio with 3 component:
 [0.92502725 0.03771986 0.03119621]

Culmulative Var Ratio with 3 component:
 [0.92502725 0.96274711 0.99394332]
```

Plot Explained - Principle

PCA on :All components



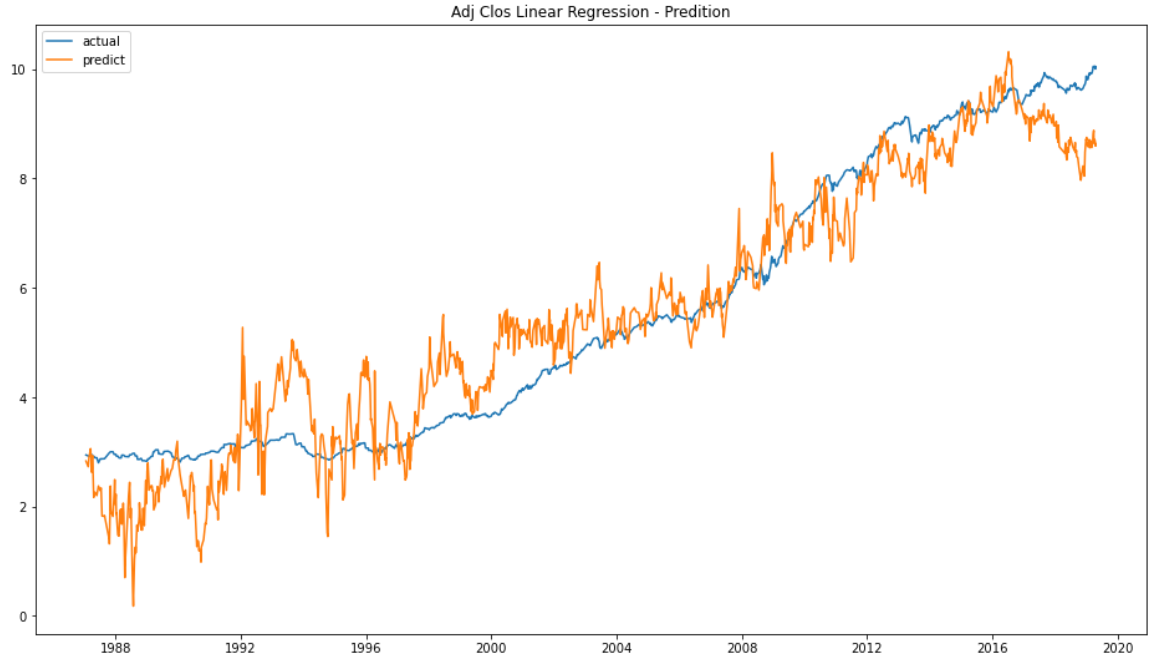
PCA on :3 components

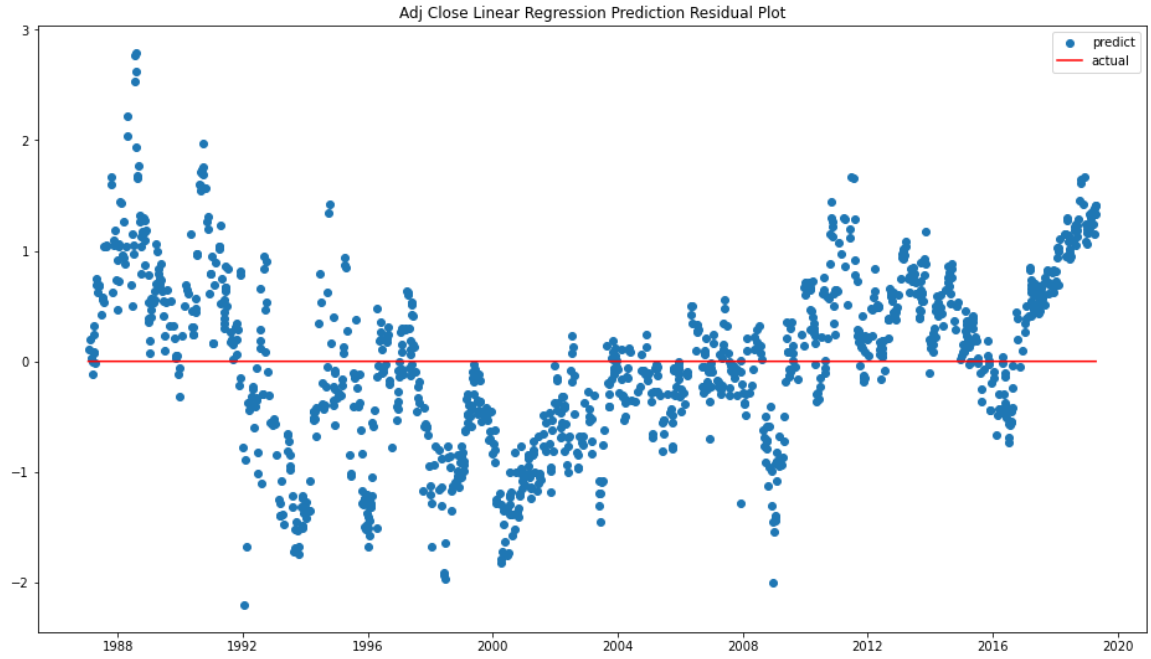


Part 3: Linear regression v. SVM regressor - baseline

create function to train

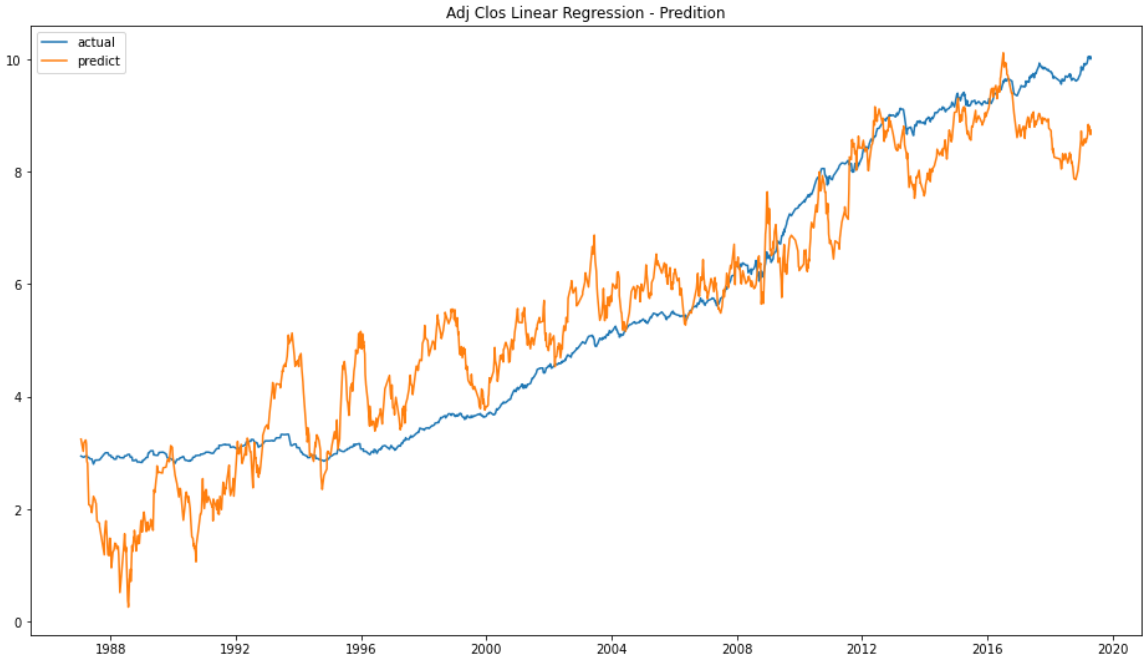
Linear Regression w/ PCA

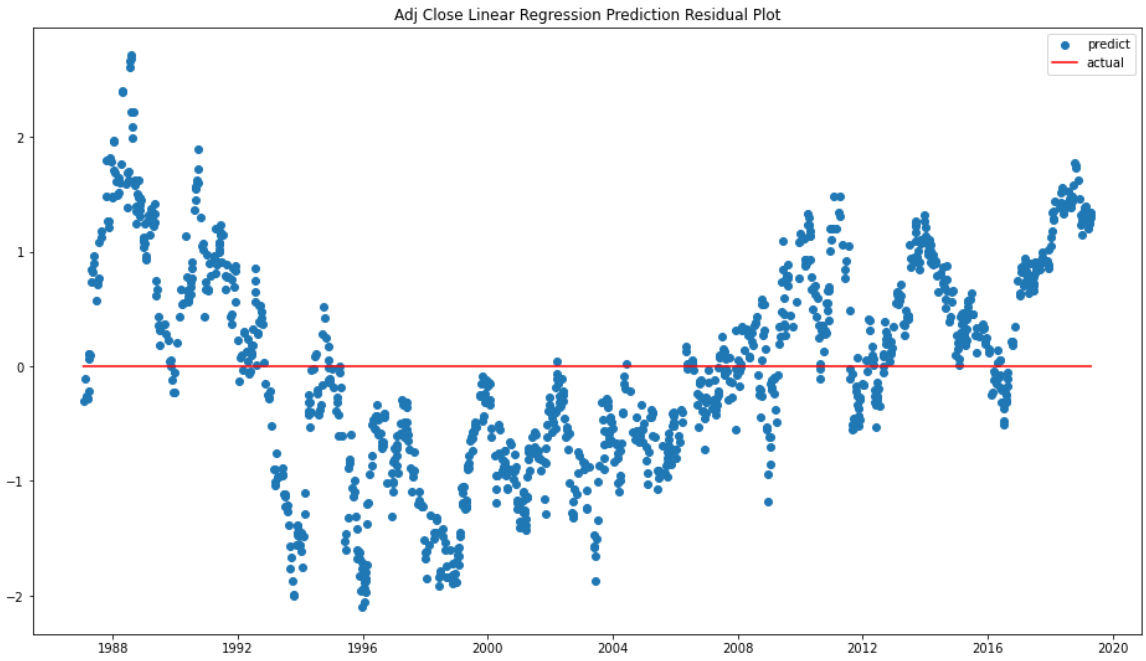





```
RootMean Squared Error: 0.7823695855057545
R-score:  0.9041309535337267
```

Linear Regression w PCA

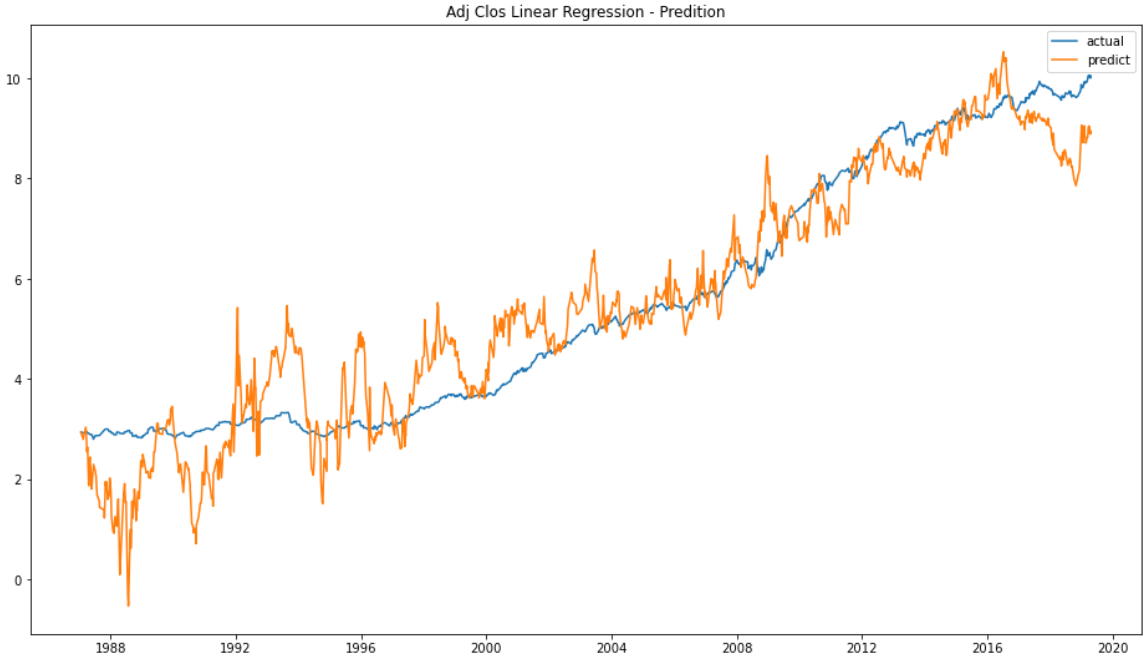


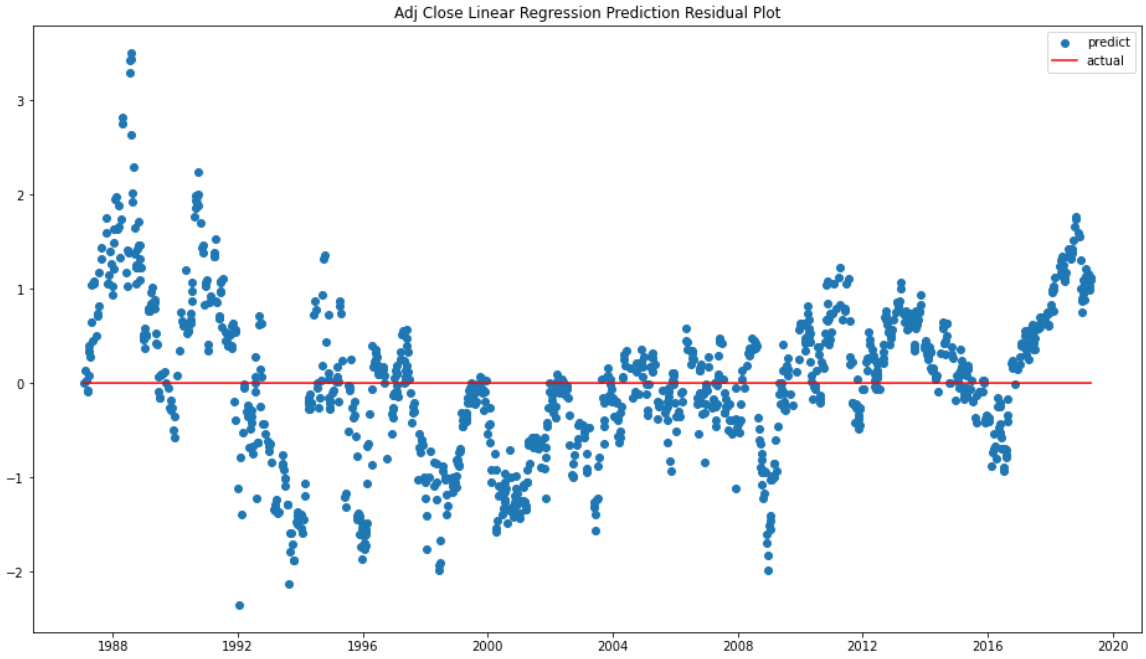



```
RootMean Squared Error: 0.9236577822901372
R-score:  0.8663783970490899
```

Using SVM

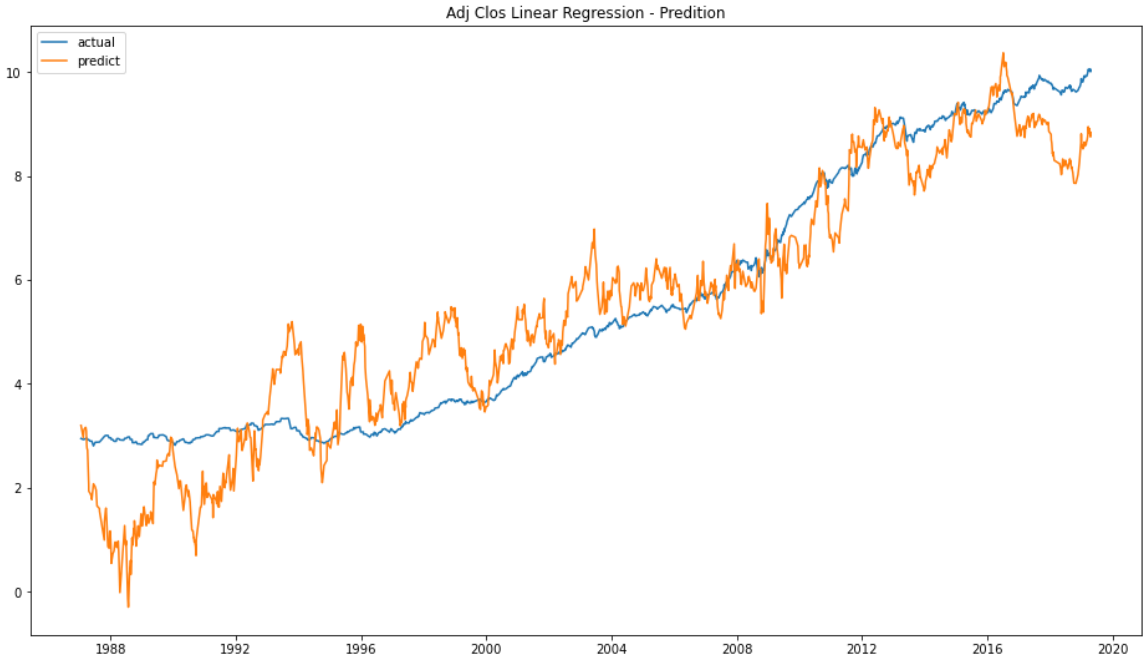
SVM w/ PCA

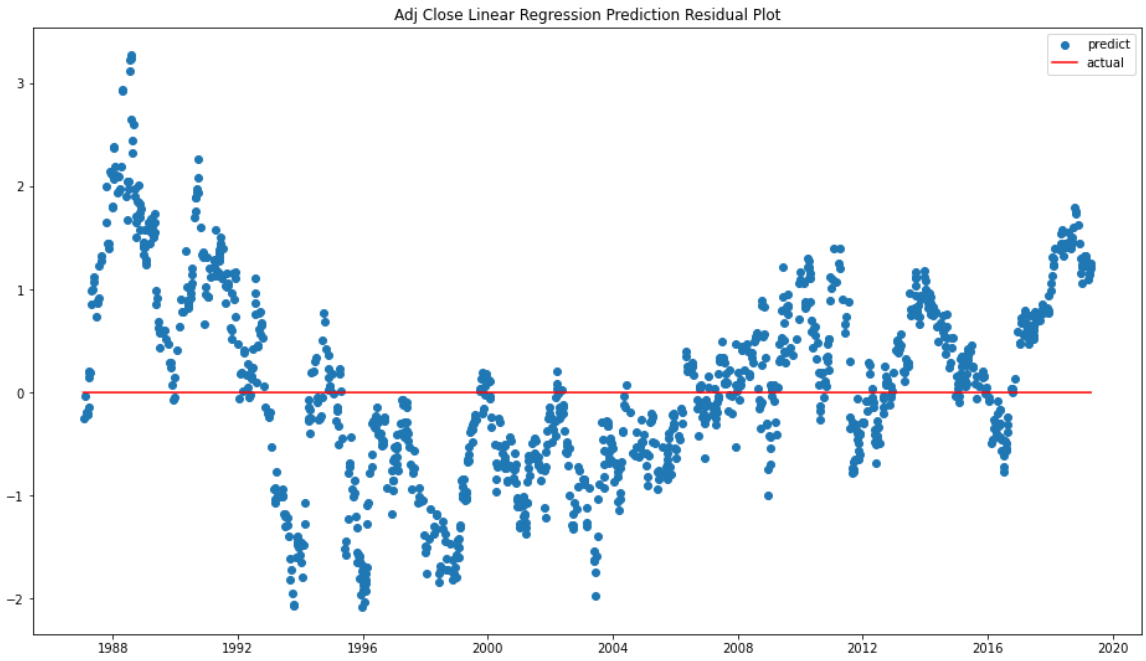




RootMean Squared Error: 0.8209562317057244
R-score: 0.8944411871024072

SVM w PCA





RootMean Squared Error: 0.9413851824569281
R-score: 0.8612000830758056

Part 4: Conclusions

From our findings, we discovered that using principal component analysis (PCA) can actually decrease the accuracy of a model. This is because PCA reduces the number of dimensions that contain crucial information, leading to information loss. On the other hand, reducing the dimensionality of the data through PCA can result in faster model execution with fewer features or components.

Part 5: Appendix

My name is Saranpat Prasertthum
My NetID is: 655667271
I hereby certify that I have read the University policy on Academic Integr
ity and that I am not in violation.