BDS-Assignment2

Author: Jingyi Wu(andrew id: jingyiw2)

Used Libraries:
Numpy
Pandas
Matplotlib
Seaborn
xgboost.sklearn
sklearn
plotly

1. Load the original and updated datasets as “data” and “updated” using pandas.

The loading results are as below.

```
data[(data['yyyymm']>=196501) & (data['yyyymm']<=200001)]
```

	yyyymm	Index	D12	E12	b/m	tbl	AAA	BAA	lty	ntis	Rfree	infl	ltr	corpr	svar	csp	CRSP_SPvw	CRSP_SPvwx
1128	196501	87.56	2.516667	4.593333	0.471723	0.0381	0.0443	0.0480	0.0422	0.017886	0.003170	0.000000	0.0040	0.0081	0.000166	-0.002201	0.0348	0.0338
1129	196502	87.43	2.533333	4.636667	0.471399	0.0393	0.0441	0.0478	0.0424	0.014813	0.003270	0.000000	0.0014	0.0009	0.000393	-0.002202	0.0037	-0.0009
1130	196503	86.16	2.550000	4.680000	0.469490	0.0393	0.0442	0.0478	0.0422	0.019147	0.003270	0.001068	0.0054	0.0012	0.000145	-0.002137	-0.0122	-0.0137
1131	196504	89.11	2.570000	4.733333	0.452559	0.0393	0.0443	0.0480	0.0422	0.014713	0.003270	0.003197	0.0036	0.0021	0.000163	-0.002143	0.0356	0.0348
1132	196505	88.42	2.590000	4.786667	0.454664	0.0389	0.0444	0.0481	0.0423	0.016319	0.003236	0.002125	0.0018	-0.0008	0.000334	-0.002566	-0.0031	-0.0084

```
print(updated.shape)
updated.head()
```

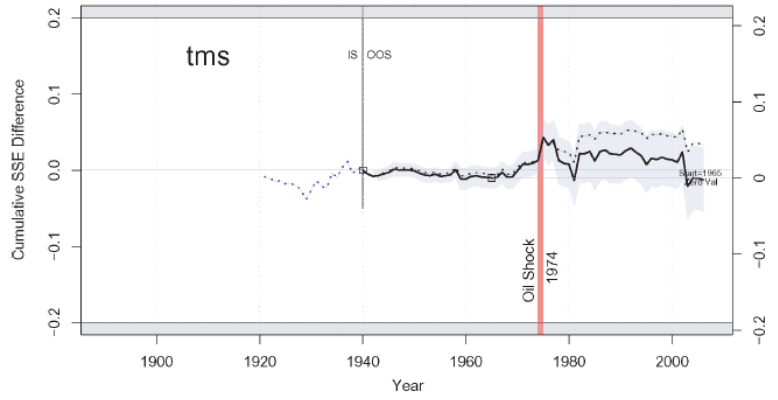
```
(1812, 18)
```

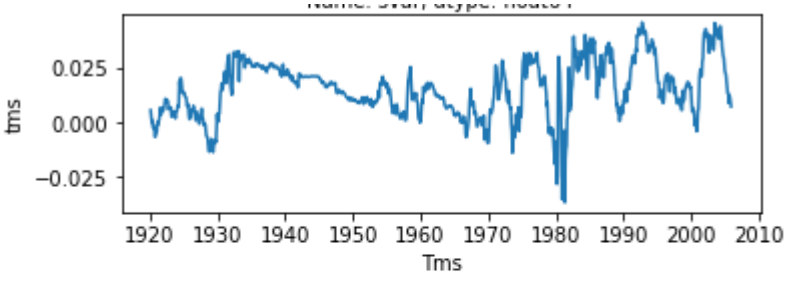
	yyyymm	Index	D12	E12	b/m	tbl	AAA	BAA	lty	ntis	Rfree	infl	ltr	corpr	svar	csp	CRSP_SPvw	CRSP_SPvwx
0	187101	4.44	0.26	0.4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	187102	4.50	0.26	0.4	NaN	NaN	NaN	NaN	NaN	NaN	0.004967	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	187103	4.61	0.26	0.4	NaN	NaN	NaN	NaN	NaN	NaN	0.004525	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	187104	4.74	0.26	0.4	NaN	NaN	NaN	NaN	NaN	NaN	0.004252	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	187105	4.86	0.26	0.4	NaN	NaN	NaN	NaN	NaN	NaN	0.004643	NaN	NaN	NaN	NaN	NaN	NaN	NaN

2. Create the predictive variables in the same way as the paper, and generate a new data frame containing these features called ”df”.

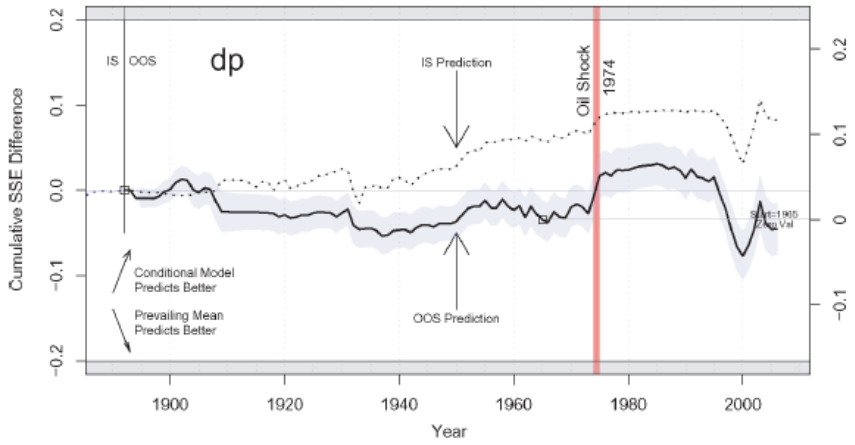
   Plot each variable’s monthly data and check with the annual performance of IS insignificant predictors in the paper. Here we use parts of the features to make comparison.

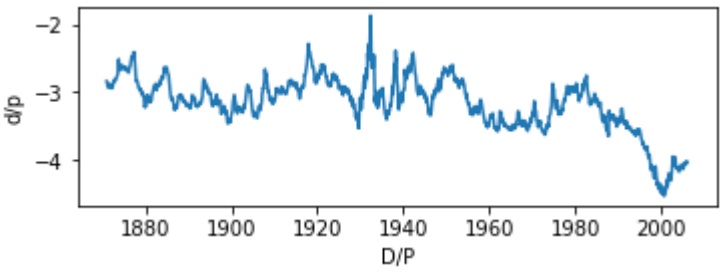
   For tms, we can see an increasing trend starting from 1970 and a sharp drop in 1980. For annual data, the line is similar and more smooth.






```
count    1031.000000
mean        0.015226
std         0.012665
min        -0.036500
25%         0.006650
50%         0.015400
75%         0.024250
max         0.045500
Name: tms, dtype: float64
```

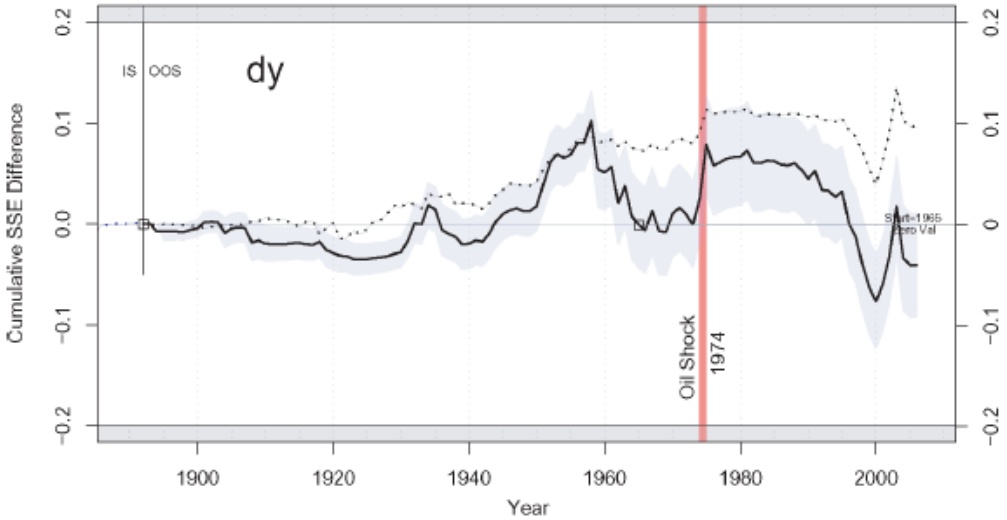


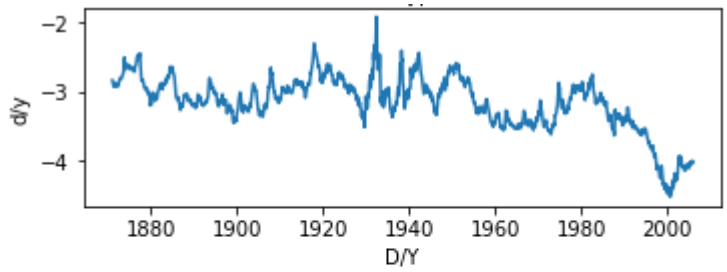


count	1620.000000
mean	-3.158223
std	0.401189
min	-4.524005
25%	-3.358060
50%	-3.111874
75%	-2.900097
max	-1.873246

Name: d/p, dtype: float64

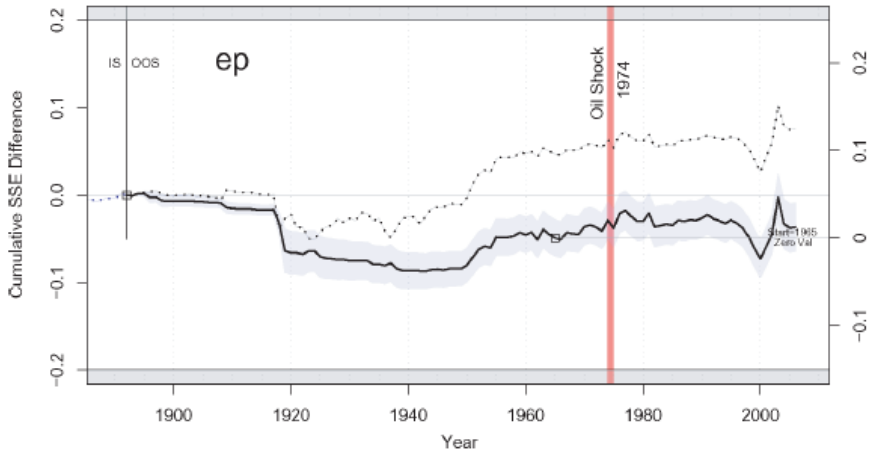
For dividend-price ratio, similar drop in 2000 and increase in 1974 for actual values and cumulative squared prediction errors.

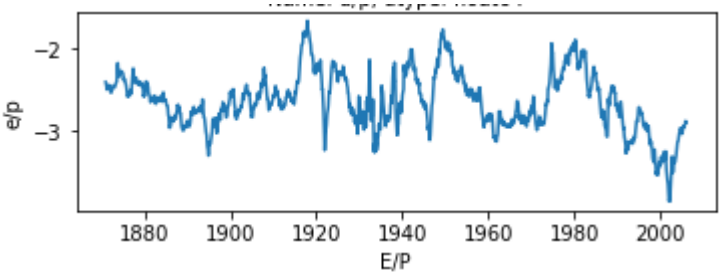




count	1619.000000
mean	-3.154938
std	0.398263
min	-4.531322
25%	-3.349773
50%	-3.106398
75%	-2.898996
max	-1.912904

Name: d/y, dtype: float64





count	1620.000000
mean	-2.637163
std	0.347272
min	-3.839448
25%	-2.871866
50%	-2.645911
75%	-2.409252
max	-1.670063

Name: e/p, dtype: float64

For dy and ep, we can find similar pattern, especially after the key event in 1974 happened. In summary, the variables are correctly interpreted.

3. Use rolling multiple linear regression for three time periods data listed and calculate key metrics.
   For three time periods data, we generate three data frames, df1, df2 and df3.
   To implement rolling regression, here we wrote a function called "rolling_regression" and for each iteration, we build model and generate predicted fi and mean_y. After all iteration, we get three arrays, true y values, mean y values, and preficted y values fi.

   For MSEa, $\frac{1}{n}\sum(yi - fi)^2$. For MSEn, $\frac{1}{n}\sum(yi - \bar{y})^2$. We get R2, and get MAE using sklearn

metric function. The results are as below.

Rolling Linear Regression Results

	R2	RMSE	MAE
1965.01-2008.12	-0.402222	0.0513366	0.037625
1976.01-2008.12	-0.587618	0.0542258	0.038781
2000.01-2008.12	-2.937812	0.0842482	0.054425

Note that here the R2 value are negative, indicating the model does not have predictive ability at all.

4. To improve the performance, since this is time-series financial data and multi-collinearity between variables exists, we use Lasso regression to see if it generates a better prediction.

Rolling Lasso Regression Results

	R2	RMSE	MAE
1965.01-2008.12	-0.0160496	0.0436995	0.033077
1976.01-2008.12	-0.0398146	0.0438845	0.032872
2000.01-2008.12	-0.1570206	0.0456671	0.034215

By changing into lasso regression, we can see that the performance improved and error decreased. Lasso regression proves to be a better fit here.

Here I also tried xgb regressor, and the performance is as below.

Rolling XGB Regression Results

	R2	RMSE	MAE
1965.01-2008.12	-0.8392916	0.0587955	0.039552
1976.01-2008.12	-0.9246126	0.0597042	0.040805
2000.01-2008.12	-3.4790942	0.0898521	0.047352

Before applying the model, I assumed that tree model will not work well in this dataset and the truth is as what I expected. Since tree models are quite sensitive to the number of instances and here we use rolling method which means for some iterations, the train set is quite small and the model has great errors, the XGB performs unsatisfying.