**Carolina Carvalho Manhães Leite (leite2)**

**IE598 MLF F19 – 09/29/2019**

**Module 5 Homework (Dimensionality Reduction) - Treasury Yield Curve dataset**

---

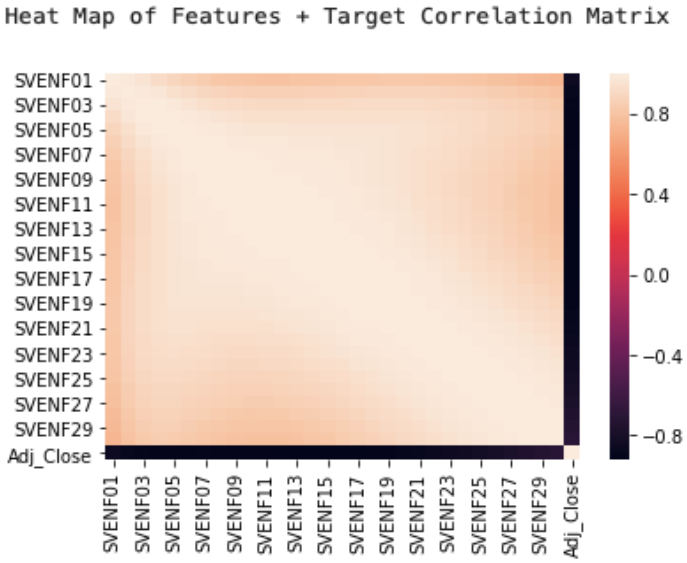
**Part 1: Exploratory Data Analysis**

Describe the data set sufficiently using the methods and visualizations that we used previously. Include any output, graphs, tables, that you think is necessary to represent the data. Label your figures and axes. DO NOT INCLUDE CODE, only output figures! Split data into training and test sets. Use random_state = 42. Use 85% of the data for the training set. Use the same split for all experiments.

--

By performing the basic descriptive statistics for all the columns in the original dataset I noticed that there were some missing values in the target variable ("Adj_Close"). I deleted those values, reducing the number of observations from 8,635 to 8,071. Even though this part (that computes the DS and respective boxplots for every variable) is still in the code, it's commented and I won't display those results here (for the sake of comprehension). I didn't find any anomalous behavior besides those missing values, and we can just move on to the next analysis.

An interesting aspect of the data is its correlation matrix:




We don't need the actual correlation values (we have the color scale) to come to the conclusion all features are correlated both within themselves and also with the target variable (with opposite correlation signs but still highly correlated). I plotted a chart with the index of the variable (i.e., their maturity dates) and the correlation of this variable with the target. It's no surprise that all correlations are high (above

0.6, roughly) and, interestingly, the correlation gets higher as the maturity gets closer to 15 (actually that's the peak), with a low point in the 30th year:


We can confirm that by accessing the table with each respective correlation (it will be printed when the code runs).

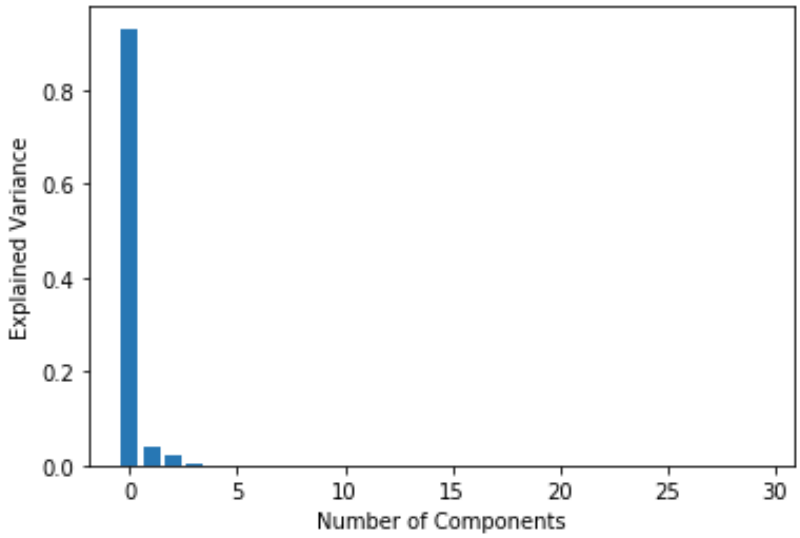
**Part 2: Perform a PCA on the Treasury Yield dataset**

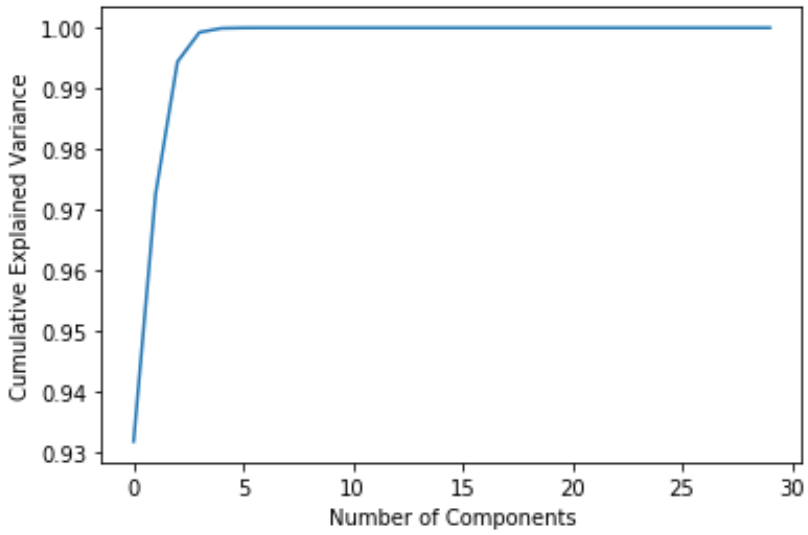
Compute and display the explained variance ratio for all components, then recalculate and display on n_components=3. What is the cumulative explained variance of the 3-component version?

--

I performed the PCA twice: one without setting a specific number of components (which means: it'll consider all variables and the number of components will be the same as the number of features) and then considering 3 components.

With the first PCA analysis we're already able to build two interesting charts, both with the explained variance ratio (the first one per component and the second one being a cumulative curve):




From both charts we can see that the variance is “rapidly” explained. Precisely, we can see that 0.99999999 is explained by the 9th component, just as the following vector of cumulative explained variance ratio shows us:

```
[0.93179697 0.97256205 0.99440592 0.99925725 0.99992059 0.99998933
 0.99999881 0.99999991 0.99999999 1.         1.         1.
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.        ]
```

We might not need to go this far to sufficiently explain the variance in our dataset. With 3 components we’re already able to explain 0.99440592 of it:

```
The cumulative explained variance for those 3 components is (in order):

[0.93179697 0.97256205 0.99440592]

It means that, together, they explain 0.994 of all variance.
```

**Part 3: Logistic regression classifier v. SVM classifier - baseline**

Fit a linear classifier model to both datasets (the original dataset with 30 attributes and the PCA transformed dataset with 3 PCs.) using SKlearn. Calculate its accuracy R2 score and RMSE for both in sample and out of sample (train and test sets). (You may use CV accuracy score if you wish).

Fit a SVM regressor model to both datasets using SKlearn. Calculate its accuracy R2 score and RMSE for both in sample and out of sample (train and test sets). (You may use CV accuracy score if you wish).

--

I understood the term “logistic regression” as actually meaning “linear regression”, since our target variable is a continuous one (our problem is not a classification one, not even a multiclassification). Thus, for this part, I built four models:

1. Linear Regression with all attributes (using SGDRegressor);
2. Linear Regression with 3 first PCA components (using SGDRegressor);
3. SVM with all attributes (using SVR);
4. SVM with 3 first PCA components (using SVR).

For all of them I calculated some basic metric values, such as the MSE and the R2 (actually, I noticed they both sum up to 1, so I only displayed the R2 on the table) and also the processing time for each of these models. The results were:

```
1st Model: LINEAR REGRESSION with all attributes
Metrics:

MSE train: 0.115
MSE test: 0.120
R2 train: 0.885
R2 test: 0.880

The processing time was: 0.044

------------

2nd Model: LINEAR REGRESSION with 3 principal components
Metrics:

MSE train: 0.142
MSE test: 0.146
R2 train: 0.858
R2 test: 0.854

The processing time was: 0.029
```

```
3rd Model: SVR with all attributes
Metrics:

MSE train: 0.106
MSE test: 0.107
R2 train: 0.894
R2 test: 0.893

The processing time was: 25.638

------------

4th Model: SVR with 3 principal components
Metrics:

MSE train: 0.137
MSE test: 0.140
R2 train: 0.863
R2 test: 0.860

The processing time was: 21.454
```

**Part 4: Conclusions**

Write a short paragraph summarizing your findings. Which model performs best on the untransformed data? Which transformation leads to the best performance increases? How does training time change for the two models. Report your results using the Results worksheet format. Embed the completed table in your report.

--

The compiled results can be seen in the following table:

| | **Experiment 1 (Treasury Yields)** | | | |
|---|---|---|---|---|
| | **Linear Regression** | | **SVR** | |
| **Baseline (all attributes)** | Train R2 | 0.885 | Train R2 | 0.894 |
| | Test R2 | 0.880 | Test R2 | 0.893 |
| | Proc. time | 0.044 | Proc. time | 25.638 |
| **PCA transform (3 PCs)** | Train R2 | 0.858 | Train R2 | 0.863 |
| | Test R2 | 0.854 | Test R2 | 0.860 |
| | Proc. time | 0.029 | Proc. time | 21.454 |

What we can see from the results is that every behavior we were “expecting” was actually found:

1. High performance metrics for both the training and the testing sets (expected: the features and the target variable are actually highly correlated);
2. Test performance lower than the train one;
3. For both the untransformed data and the transformed one the SVR performs better than the Linear Regression (SVR is a more complex model);
4. We reduced the performance of the models when using just the 3 first principal components (but not drastically, since those components capture more than 99% of the variance);
5. The processing time is way higher for the SCR than for the linear regression, and was more drastically reduced for the SVR methods when comparing the model with all features and the model with just the 3 components. The percentual difference may be higher for the Linear regression but, in terms of "absolute values of time", the reduction in the SVR was more significative.

**Part 5: Appendix**

https://github.com/leiteccml/Carolina_2019