Assignment 6: Principle Components in Predictive Modeling

Sri Seshadri

7/27/2017

1. Introduction

This report discusses the analysis of stock portfolio data to predict log returns of Vanguard large cap index fund using stock indices of 20 companies from various industries. The choice of Principle Component Analysis (PCA) for selecting predictors for predictive modeling is compared with full model (all twenty indices as predictors), arbitary model (random selection of predictors) and backward elimination methods of selection. The Mean Absolute Error (MAE) is used a criteria for evaluating predictive performance. It is seen that the PCA based approach and backward elimination methods perform well and PCA is a good data reduction technique.

2. Exploratory analysis

Since there are number of variables in the data, we start by quantifying the associations amongst them by plotting the correlations. The index VV (Vanguard Index) is of interest and it would be useful to plot the correlation of VV with other variables. Figure 1 shows the correlations of VV with other stock indices.

2.1 Statistical graphic Vs Data visualization

While figure 1 is useful for understanding the VV's correlations with other stock indices, it would be useful to gain insights into how other stock indicies are correlated amongst themselves. From figure 2, it can be noted that higher correlations are on the base of the right triangles formed by the matrx diagonal i.e. indices SLB, WFC, XOM and VV are relatively highly correlated with other indices. Such insights are possible only with visualizations such as this.

It can be seen that indices DPS, MPC and PEP are not significantly correlated with other indices. They are likely to have low Variance Inflation Factors (VIF), as mentioned above GS, XOM and SLB are likely to have higher VIF due to their high correlations with other indices.

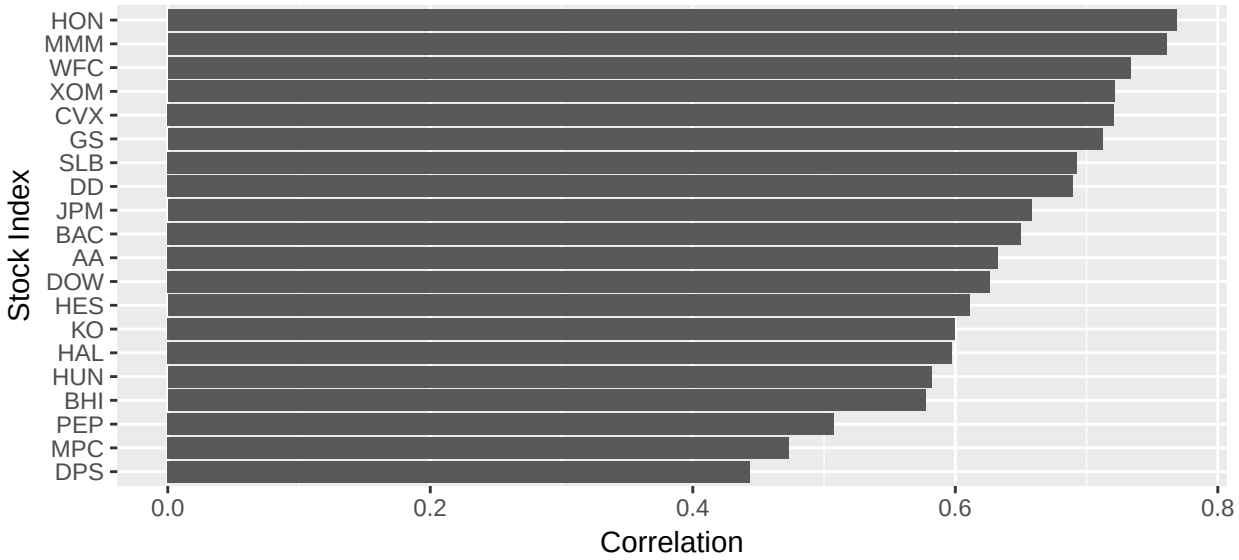

Figure 1: Correlations with VV

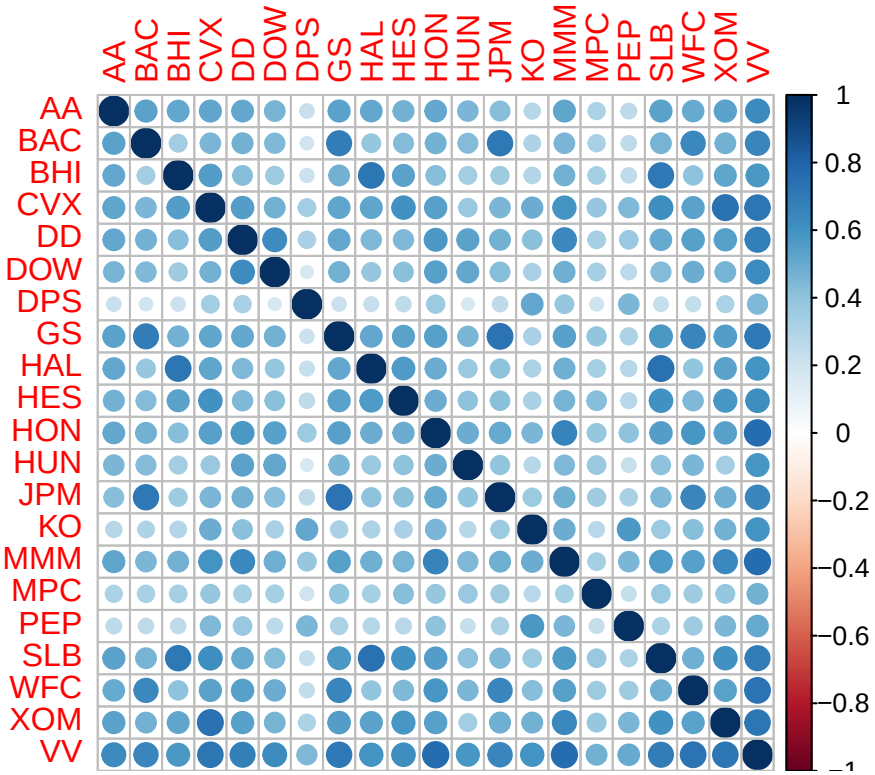

Figure 2: Correlation plot

2.2 Variance Inflation Factor (VIF)

In the previous section, it was seen that GS, XOM, and SLB were highly correlated with other predictors and they were suspected to have relatively high VIF. One way of assessing the VIF is to fit regression models and assessing VIF. Two models; an arbitary model and a full modelare fit. The VIFs are assesed for the predictor variables in the model and is shown below in table 1. Table 1 shows the top 5 VIFs for each model. It is seen that variance inflation factors are less than the thumb rule of 10, therefore there isn't strong evidence of concern due to multicolinearity. It would be interesting to see how Principle Component Analysis (PCA) would work on this data.

```
##
## Call:
## lm(formula = VV ~ GS + DD + DOW + HON + HUN + JPM + KO + MMM +
##     XOM, data = returns)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0139179 -0.0016005 -0.0000926  0.0016690  0.0172703
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0001008  0.0001331   0.757 0.449290
## GS          0.0784765  0.0138277   5.675 2.37e-08 ***
## DD          0.0354057  0.0177154   1.999 0.046204 *
## DOW         0.0406763  0.0116993   3.477 0.000552 ***
## HON         0.1449817  0.0170837   8.487 2.53e-16 ***
## HUN         0.0385118  0.0077371   4.978 8.93e-07 ***
## JPM         0.0505123  0.0132262   3.819 0.000151 ***
## KO          0.1419686  0.0176282   8.054 6.14e-15 ***
## MMM         0.1336002  0.0239378   5.581 3.96e-08 ***
## XOM         0.1480728  0.0213601   6.932 1.31e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002951 on 491 degrees of freedom
## Multiple R-squared:  0.8518, Adjusted R-squared:  0.849
## F-statistic: 313.5 on 9 and 491 DF,  p-value: < 2.2e-16
```

```
##
## Call:
## lm(formula = VV ~ ., data = returns)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0139266 -0.0015542  0.0000313  0.0015042  0.0140759
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.953e-05  1.213e-04   0.820 0.412433
## AA           1.538e-02  1.040e-02   1.479 0.139894
## BAC          2.723e-02  9.697e-03   2.808 0.005188 **
## BHI          1.604e-02  1.161e-02   1.382 0.167752
## CVX          5.742e-02  2.068e-02   2.776 0.005720 **
## DD           1.003e-02  1.625e-02   0.618 0.537144
## DOW          3.600e-02  1.069e-02   3.366 0.000824 ***
```

```
## DPS          5.659e-02  1.493e-02   3.790 0.000170 ***
## GS           3.434e-02  1.358e-02   2.529 0.011766 *
## HAL         -1.976e-03  1.210e-02  -0.163 0.870344
## HES          4.393e-03  9.688e-03   0.453 0.650432
## HON          1.071e-01  1.608e-02   6.658 7.62e-11 ***
## HUN          2.867e-02  7.222e-03   3.969 8.31e-05 ***
## JPM          2.224e-02  1.329e-02   1.674 0.094849 .
## KO           9.425e-02  1.847e-02   5.103 4.82e-07 ***
## MMM          1.093e-01  2.202e-02   4.964 9.63e-07 ***
## MPC          1.079e-02  7.024e-03   1.536 0.125134
## PEP          2.092e-02  2.034e-02   1.028 0.304293
## SLB          4.851e-02  1.453e-02   3.339 0.000905 ***
## WFC          7.738e-02  1.580e-02   4.897 1.33e-06 ***
## XOM          5.797e-02  2.301e-02   2.519 0.012094 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002666 on 480 degrees of freedom
## Multiple R-squared:  0.8818, Adjusted R-squared:  0.8768
## F-statistic:   179 on 20 and 480 DF,  p-value: < 2.2e-16
```

Table 1: Top 5 VIF by model

Model	Predictors	VIF
Arbitary model	GS	2.705795
	MMM	2.590177
	DD	2.368257
	JPM	2.324600
	HON	2.261397
Full Model	SLB	3.258982
	GS	3.197811
	XOM	2.949826
	CVX	2.920187
	HAL	2.919924

3. Principle Component Analysis (PCA)

The tickers from AA through XOM (not including VV) are transformed using PCA. Note that the tickers need not be standardized as we are going to be using the log returns of the tickers. The scatter plot of loadings (loadings are the weights of the variables that contribute to the component - the columns of the loading matrix correspond to eigen vectors) of the first two principle components is shown in Figure 3. It is seen that the tickers belonging to soft drinks industry are grouped together. They also have a higher weightage in explaining the variation in the data.

It will be interesting to see how PCA reduced the data in terms of variation.

3.1 Choosing number of principle components

The choice of number of principle components are to be based on how much of the variance in the variables do the components cumulatively explain.

- Rule of thumb is to choose components that explain 70% to 90% of the variation in the variables.
- Another option is to choose components that have eigenvalues greater than the average of eigen values. If eigenvalues are extracted from correlation matrix, components with eigenvalues less than 1 are excluded.
- Use Scree diagram, a plot of eigenvalues or component variance againt the component number. The number of components selected the is the value corresponding to the location where, the Scree plot forms an elbow.

3.1.1 Scree plot

Scree plots are useful in identifying the principle components that contribute to explaining the variation in the variables. Figures 4, 5 and 6 show different variations of the scree plot. Figure 4 is the default version of scree plot in R. The scree plot looks to be truncated to 10 components.It is seen that component 1 explains the majority of the variance.

It would be useful to see the proportion of variance explained by the components. Figure 5 shows the proportion of the variance that the principle components are able to explain. It is seen that the contribution of principle components to explanation of variance plateaus at component number 8. Additional components beyond 8 are not a significant addition to explaining the variance. As a rule of thumb, principle components that explain 80% (or 70% to 90%) of the total variance is chosen for data reduction. It will be useful to plot the cumulative variance of the principle components. Figure 6 shows the first 8 components explain 80% of the total variance in the data.

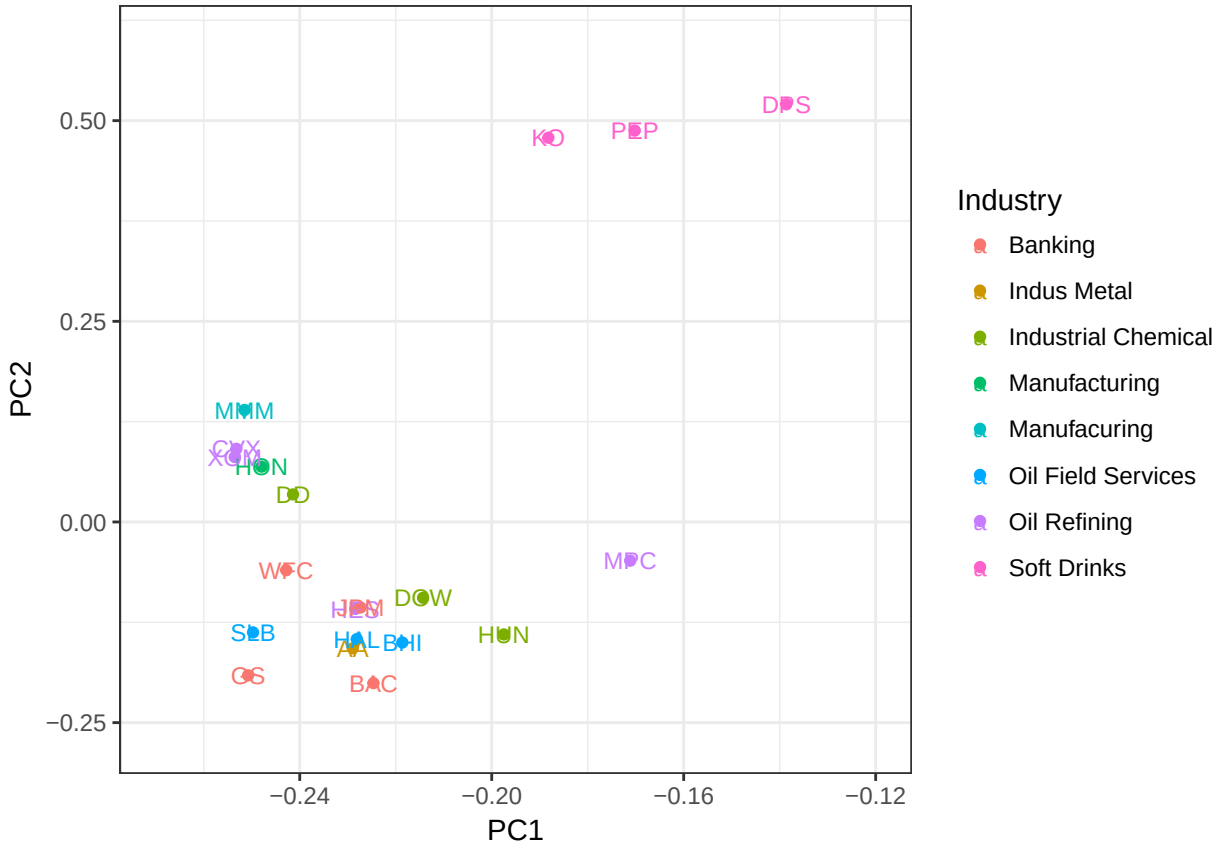

Figure 3: Variable loadings of first 2 principle components

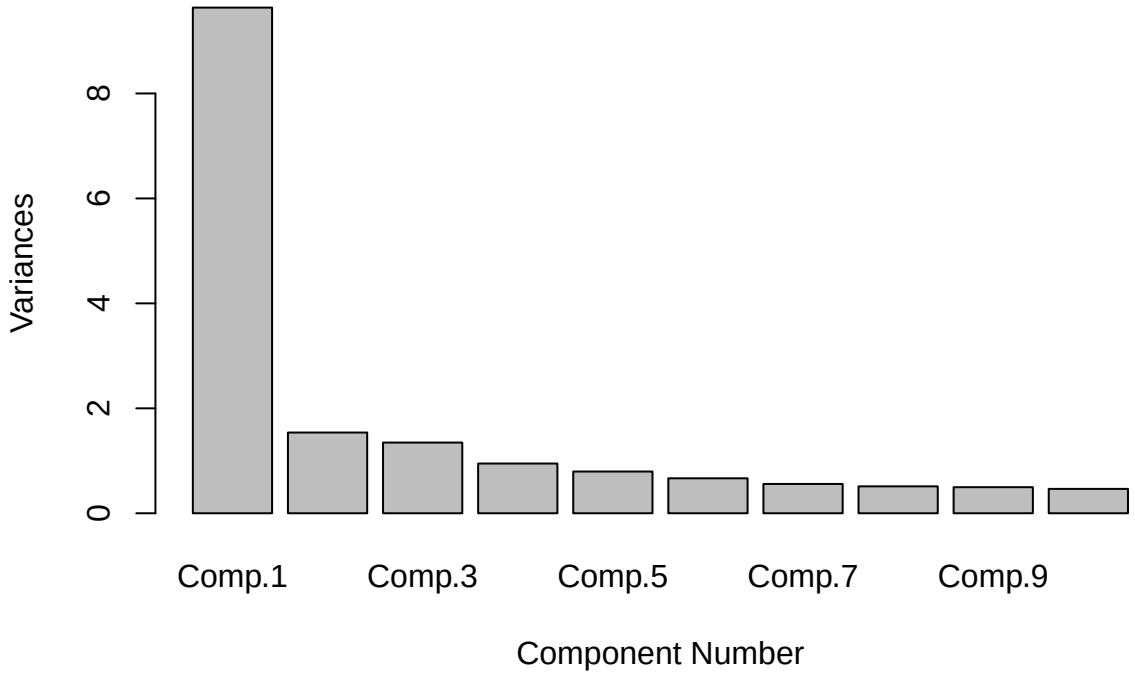

Figure 4: Scree Plot - Not so good looking

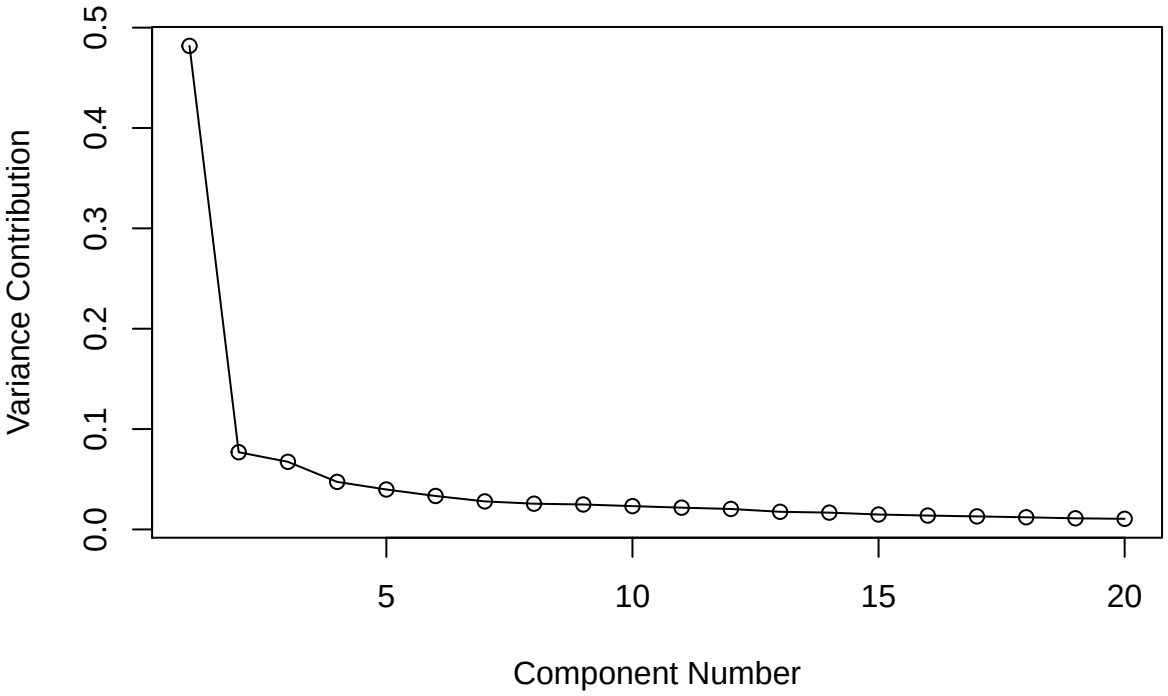

Figure 5: Scree plot - Variance explained

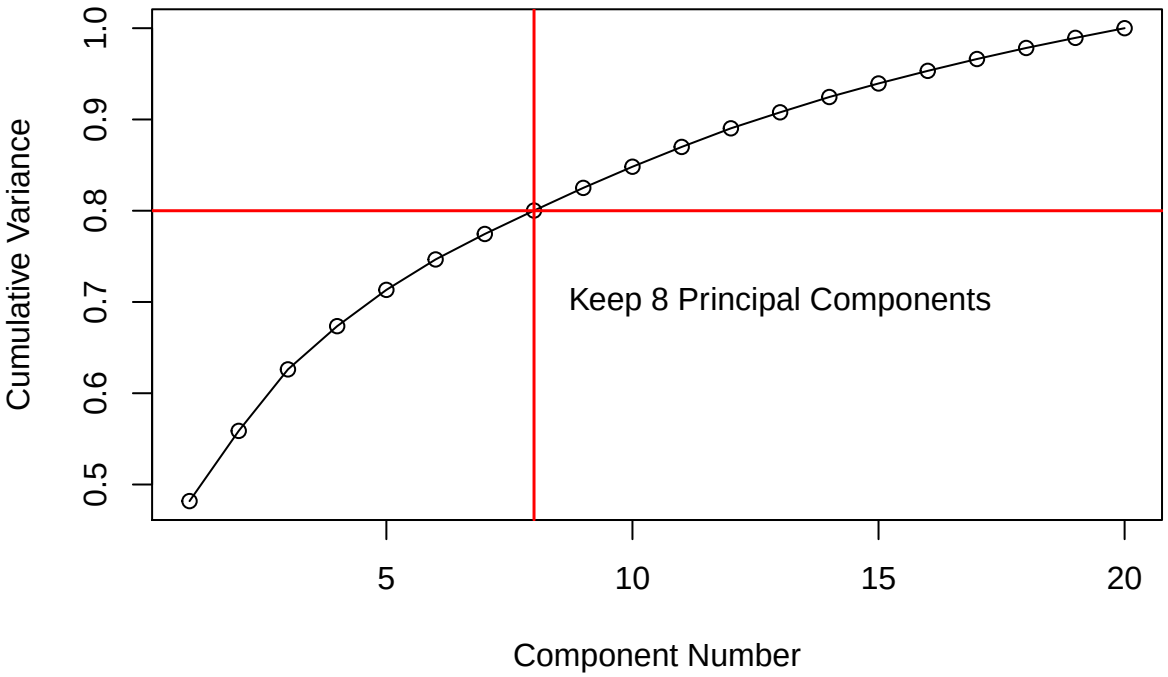

Figure 6: Scree cumulative variance explained

With the data reduced to eight principle components, the eight components are to be used as predictors for modelling. The next section discusses the model.

$$Score_i = a_{i1}(x_1 - \bar{x_1}) + a_{i2}(x_2 - \bar{x_2}) + ...$$

3.2. Predictive model with Principle Components as predictors

The scores of the principle components are used as predictors. Score is calculated by multiplying the variable values minus the mean by the "loading"" of the variable.

We will split the scores into training and test sets for modelling. 70% of the data is chosen in random as the training set and the remaining of data is used as validation set. Ideally, the principle components' eigenvectors are computed based on the training sample and is used to compute the principle components of the validation or test sample. For simplicity, in this assignment the priciple components are calculated using the entire data set.

Table 2: Training and Validation sampling

Data	Samples
Training set	367
Validation set	134
Total	501

3.2.1 Linear Regression with principle components as predictors

A linear regression model with the 8 principle component scores as perdictors is fit. The model and the fit is summarized below. Let us call the model, "PCA model".

VV = 0.00075 - 0.00227 * Comp.1 + 0.00046 * Comp.2 + 5e-04 * Comp.3 + 0.00016 * Comp.4 - 0.00018 * Comp.5 - 8e-05 * Comp.6 + 4e-05 * Comp.7 - 0.00035 * Comp.8

```
##
## Call:
## lm(formula = VV ~ Comp.1 + Comp.2 + Comp.3 + Comp.4 + Comp.5 +
##     Comp.6 + Comp.7 + Comp.8, data = returns.scores)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0132078 -0.0016047  0.0000119  0.0016485  0.0147418
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.520e-04  1.222e-04   6.155 1.56e-09 ***
## Comp.1      -2.265e-03  3.936e-05 -57.548  < 2e-16 ***
## Comp.2       4.582e-04  9.850e-05   4.651 4.25e-06 ***
## Comp.3       5.032e-04  1.053e-04   4.781 2.31e-06 ***
## Comp.4       1.551e-04  1.255e-04   1.236   0.2171
## Comp.5      -1.816e-04  1.370e-04  -1.325   0.1856
## Comp.6      -8.223e-05  1.497e-04  -0.549   0.5830
## Comp.7       3.958e-05  1.635e-04   0.242   0.8088
## Comp.8      -3.544e-04  1.706e-04  -2.077   0.0383 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.002735 on 492 degrees of freedom
## Multiple R-squared:  0.8724, Adjusted R-squared:  0.8703
## F-statistic: 420.5 on 8 and 492 DF,  p-value: < 2.2e-16
```

Table 3: VIF of Principle components

PC	VIF
Comp.1	1
Comp.2	1
Comp.3	1
Comp.4	1
Comp.5	1
Comp.6	1
Comp.7	1
Comp.8	1

It is seen that VIF for the components are 1. This is not surprising because the principle components are by design are orthogonal to each other. The eigenvectors are constrained to be uncorrelated to each other.

3.2.2. Predictive metric (MAE) for PCA model

The table below shows the predictive performance of the PCA model. The training and test sample's mean absolute error (MAE) are close.

Table 4: MAE of PCA model

Model	Train.MAE	Test.MAE
PCA model	0.0020201	0.0019084

3.3. Linear regression models with tickers as predictors

It will be interesting to compare the performance of the "PCA model" with linear regression models using raw tickers as the predictors. We will compare the PCA models by fitting an arbitarty model and a full model like the models in section 2.2. To compare the arbitary and full models with PCA model, it would be useful to split the data set into training and validation samples.

3.3.1. Arbitary model

The results for the arbitary model is shown below.

VV = 4e-05 + 0.11552 * GS - 0.00219 * DD + 0.03199 * DOW + 0.20029 * HON + 0.02764 * HUN + 0.02047 * JPM + 0.1659 * KO + 0.1465 * MMM + 0.13541 * XOM

```
##
## Call:
## lm(formula = VV ~ GS + DD + DOW + HON + HUN + JPM + KO + MMM +
##     XOM, data = returns.train)
##
## Residuals:
```

```
##        Min         1Q     Median         3Q        Max
## -0.0064406 -0.0015686 -0.0001513  0.0016071  0.0072019
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.983e-05  2.238e-04   0.178  0.85903
## GS           1.155e-01  2.375e-02   4.865 3.41e-06 ***
## DD          -2.185e-03  3.425e-02  -0.064  0.94923
## DOW          3.199e-02  2.146e-02   1.491  0.13862
## HON          2.003e-01  3.579e-02   5.596 1.34e-07 ***
## HUN          2.764e-02  1.448e-02   1.909  0.05857 .
## JPM          2.047e-02  1.986e-02   1.030  0.30487
## KO           1.659e-01  2.799e-02   5.928 2.84e-08 ***
## MMM          1.465e-01  5.231e-02   2.800  0.00592 **
## XOM          1.354e-01  3.324e-02   4.074 8.18e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002534 on 124 degrees of freedom
## Multiple R-squared:  0.891,  Adjusted R-squared:  0.8831
## F-statistic: 112.7 on 9 and 124 DF,  p-value: < 2.2e-16
```

Table 5: MAE of Arbitary model

Model	Train.MAE	Test.MAE
Arbitary model	0.0019327	0.0023605

3.3.2 Full model

A full model with all tickers as predictors is fit. The model and the results are shown below.

VV = 4e-05 + 0.02005 * AA + 0.01476 * BAC - 0.03591 * BHI + 0.05339 * CVX - 0.03999 * DD + 0.03 * DOW + 0.05556 * DPS + 0.06427 * GS + 0.0127 * HAL + 0.02601 * HES + 0.15656 * HON + 0.01143 * HUN + 0.01154 * JPM + 0.10486 * KO + 0.12352 * MMM + 0.01511 * MPC + 0.01481 * PEP + 0.0548 * SLB + 0.07723 * WFC + 0.06282 * XOM

```
##
## Call:
## lm(formula = VV ~ ., data = returns.train[, -22])
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0068499 -0.0012635 -0.0000868  0.0015194  0.0066229
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.648e-05  2.173e-04   0.168  0.86699
## AA           2.005e-02  1.947e-02   1.030  0.30536
## BAC          1.476e-02  1.886e-02   0.783  0.43549
## BHI         -3.591e-02  1.828e-02  -1.964  0.05198 .
## CVX          5.339e-02  4.060e-02   1.315  0.19115
## DD          -3.999e-02  3.324e-02  -1.203  0.23150
## DOW          3.000e-02  2.204e-02   1.361  0.17624
```

```
## DPS          5.556e-02  2.869e-02   1.936  0.05532 .
## GS           6.427e-02  2.498e-02   2.573  0.01139 *
## HAL          1.270e-02  2.222e-02   0.571  0.56881
## HES          2.601e-02  2.022e-02   1.286  0.20092
## HON          1.566e-01  3.525e-02   4.442 2.09e-05 ***
## HUN          1.143e-02  1.453e-02   0.786  0.43337
## JPM          1.154e-02  1.941e-02   0.595  0.55322
## KO           1.049e-01  3.373e-02   3.109  0.00238 **
## MMM          1.235e-01  5.079e-02   2.432  0.01660 *
## MPC          1.511e-02  1.237e-02   1.221  0.22449
## PEP          1.481e-02  4.099e-02   0.361  0.71859
## SLB          5.480e-02  2.914e-02   1.880  0.06263 .
## WFC          7.723e-02  3.102e-02   2.490  0.01423 *
## XOM          6.282e-02  3.685e-02   1.705  0.09098 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002349 on 113 degrees of freedom
## Multiple R-squared:  0.9147, Adjusted R-squared:  0.8996
## F-statistic: 60.57 on 20 and 113 DF,  p-value: < 2.2e-16
```

Table 6: MAE of Full model

Model	Train.MAE	Test.MAE
Full model	0.0016402	0.0021866

3.3.3 Model comparison

THe models that have been fit so far is shown in the table below.

```
##
## ---------------------------------------
## Models   Equation
## -------- ------------------------------
## PCA      VV = 0.00075 - 0.00227 *
##          Comp.1 + 0.00046 * Comp.2 +
##          5e-04 * Comp.3 + 0.00016 *
##          Comp.4 - 0.00018 * Comp.5 -
##          8e-05 * Comp.6 + 4e-05 *
##          Comp.7 - 0.00035 * Comp.8
##
## Arbitary VV = 4e-05 + 0.11552 * GS -
##          0.00219 * DD + 0.03199 * DOW +
##          0.20029 * HON + 0.02764 * HUN
##          + 0.02047 * JPM + 0.1659 * KO
##          + 0.1465 * MMM + 0.13541 * XOM
##
## Full     VV = 4e-05 + 0.02005 * AA +
##          0.01476 * BAC - 0.03591 * BHI
##          + 0.05339 * CVX - 0.03999 * DD
##          + 0.03 * DOW + 0.05556 * DPS +
##          0.06427 * GS + 0.0127 * HAL +
##          0.02601 * HES + 0.15656 * HON
```

```
##           + 0.01143 * HUN + 0.01154 *
##           JPM + 0.10486 * KO + 0.12352 *
##           MMM + 0.01511 * MPC + 0.01481
##           * PEP + 0.0548 * SLB + 0.07723
##           * WFC + 0.06282 * XOM
## ---------------------------------------
##
## Table: Model equations
```

The predicitve perfromance of the models are compared in table 8. It is seen that from a predictive performance stand point the PCA model is the best model; it has the lowest MAE. Also, from a VIF perspective the PCA model has the lowest VIF possible, which makes the regression coefficients stable. This reflects in the predictive performance, especially when there test sample has minor extrapolation in the predicitve space compared to the training sample.

It is seen that not all principle components' coefficiennts where statistically signifiant, it'll be useful to see how automated variable selection methods choose principle components as predictors. PCA being an unsupervised variable reduction technique, it'll be useful to wrap this technique with supervised (variable selection with response variable) automated variable selection technique.

Table 7: Predictive performance of regression models

Model	Train.MAE	Test.MAE
PCA model	0.0020201	0.0019084
Arbitary model	0.0019327	0.0023605
Full model	0.0016402	0.0021866

4. Automated variable selection

The backward elimination variable selection method is employed to choose predictors from the 20 priciple components. The fit for the backward elimination is shown below.

VV = 0.00075 - 0.00226 * Comp.1 + 0.00041 * Comp.2 + 0.00049 * Comp.3 + 0.00022 * Comp.4 - 0.00028 * Comp.8 - 0.00045 * Comp.9 + 0.00032 * Comp.10 + 0.00072 * Comp.11 + 0.00046 * Comp.14 - 0.00043 * Comp.16

The backward elimination method picked 10 of the 20 principle components compared to the 8 that was picked in the PCA analysis in section 3.1. The backward elimination chose components 9, 10,11,14 and 16 in place of Components 5,6, and 7 in section 3.1. The variance inflation factors reamin at 1 which is not surprising, for reasons mentioned in section 3.2.1.

```
##
## Call:
## lm(formula = VV ~ Comp.1 + Comp.2 + Comp.3 + Comp.4 + Comp.8 +
##     Comp.9 + Comp.10 + Comp.11 + Comp.14 + Comp.16, data = train.scores)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.013595 -0.001649  0.000062  0.001465  0.014160
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.521e-04  1.437e-04   5.232 2.87e-07 ***
## Comp.1      -2.261e-03  4.598e-05 -49.182  < 2e-16 ***
## Comp.2       4.058e-04  1.150e-04   3.530 0.000470 ***
```

```
## Comp.3       4.941e-04  1.245e-04   3.970 8.71e-05 ***
## Comp.4       2.163e-04  1.423e-04   1.520 0.129485
## Comp.8      -2.792e-04  1.971e-04  -1.417 0.157442
## Comp.9      -4.515e-04  2.009e-04  -2.247 0.025246 *
## Comp.10      3.242e-04  2.067e-04   1.568 0.117664
## Comp.11      7.228e-04  2.134e-04   3.387 0.000784 ***
## Comp.14      4.591e-04  2.428e-04   1.890 0.059507 .
## Comp.16     -4.311e-04  2.741e-04  -1.573 0.116554
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002751 on 356 degrees of freedom
## Multiple R-squared:  0.8749, Adjusted R-squared:  0.8714
## F-statistic: 248.9 on 10 and 356 DF,  p-value: < 2.2e-16
```

```
##   Comp.1   Comp.2   Comp.3   Comp.4   Comp.8   Comp.9  Comp.10  Comp.11
## 1.004340 1.009935 1.009405 1.008870 1.004940 1.008710 1.005426 1.007111
##  Comp.14  Comp.16
## 1.006297 1.005230
```

4.1. Model comparison

The predictive performance of the models fitted are compared below. Though the backward elimination's test MAE is better than the rest, its not by a significant margin. The PCA and the backward elimination models are performing alike.

Table 8: Comparison of regression models

Model	Train.MAE	Test.MAE
PCA model	0.0020201	0.0019084
Arbitary model	0.0019327	0.0023605
Full model	0.0016402	0.0021866
Backward	0.0020258	0.0018424

5. Conclusion

The stock portfolio data was analyzed and predictive models for predicting Vanguard Large Cap index with stocks from twenty companies belonging to various industries as predictors were modelled. Several predictor selection approaches like PCA, full model, arbitary model and backward elimination method were used. The predictive performance of the models were compared. It was found that PCA was a good data reduction technique that reduces the number of variables to key principle components for analysis and modeling. The PCA based model and backward elimination model had comparable predictive performance. They performed better than the arbitary and full model.

APPENDIX

A.1 R code

```
knitr::opts_chunk$set(echo = F, tidy.opts = list(width.cutoff = 70), tidy = TRUE)
library(magrittr)
library(ggplot2)
# read data in
stocks <- read.csv(file = "stock_portfolio.csv", header = T)
# format date & sort chronologically
stocks$Date <- as.Date(stocks$Date, "%d -%b-%y")
stocks <- stocks %>% dplyr::arrange(Date)
# get log-returns of the data
logreturns <- function(x) {
    log(x[-1]/x[-length(x)])
}
returns <- purrr::map_df(.x = stocks[, -1], .f = logreturns)
# compute correlations table

correlations <- as.data.frame(cor(returns))

# get only VV
library(ggplot2)
library(forcats)
VV_cor <- as.data.frame(correlations$VV[-nrow(correlations)])
colnames(VV_cor) <- "VV"
rownames(VV_cor) <- rownames(correlations)[-21]
ggplot(data = VV_cor, mapping = aes(x = fct_reorder(rownames(VV_cor), VV),
    y = VV)) + geom_col() + coord_flip() + xlab("Stock Index") + ylab("Correlation")
#+ ggtitle ('                                        Correlations with VV')

corrplot::corrplot(corr = as.matrix(correlations))
library(car)
# somemodel
RandomModel <- lm(VV ~ GS + DD + DOW + HON + HUN + JPM + KO + MMM + XOM,
    data = returns)
summary(RandomModel)

vifrandom <- as.data.frame(sort(car::vif(RandomModel), decreasing = T))
vifrandom$Model <- "Arbitary model"
vifrandom$Predictors <- row.names(vifrandom)
colnames(vifrandom) <- c("VIF", "Model", "Predictors")

# fullmodel
form <- paste0(colnames(correlations)[-21], collapse = "+")
fullmodel <- lm(VV ~ ., data = returns)
summary(fullmodel)
viffm <- as.data.frame(sort(car::vif(fullmodel), decreasing = T))
viffm$Model <- "Full Model"
viffm$Predictors <- row.names(viffm)
colnames(viffm) <- c("VIF", "Model", "Predictors")
VIF <- rbind(vifrandom, viffm, row.names = F)
VIFReport <- VIF[1:nrow(VIF) - 1, ] %>% dplyr::group_by(Model) %>% dplyr::top_n(n = 5,
```

```
    wt = VIF)

VIFReport <- VIFReport[, c("Model", "Predictors", "VIF")]
knitr::kable(VIFReport, format = "latex", caption = "Top 5 VIF by model") %>%
    kableExtra::kable_styling(latex_options = "striped") %>% kableExtra::collapse_rows(columns = 1)
returns.pca <- princomp(returns[, -21], cor = T)
pc.1 <- returns.pca$loadings[, 1]
pc.2 <- returns.pca$loadings[, 2]
pcs <- data.frame(pc.1, pc.2, names(pc.1))
colnames(pcs) <- c("PC1", "PC2", "Index")
pcs$Industry <- c("Indus Metal", "Banking", "Oil Field Services", "Oil Refining",
    "Industrial Chemical", "Industrial Chemical", "Soft Drinks", "Banking",
    "Oil Field Services", "Oil Refining", "Manufacturing", "Industrial Chemical",
    "Banking", "Soft Drinks", "Manufacuring", "Oil Refining", "Soft Drinks",
    "Oil Field Services", "Banking", "Oil Refining")
ggplot(data = pcs, mapping = aes(x = PC1, y = PC2, label = Index, color = Industry)) +
    geom_point() + geom_text(size = 3) + theme_bw() + xlim(c(-0.27, -0.12)) +
    ylim(c(-0.27, 0.6))
plot(returns.pca, xlab = "Component Number", main = " ")
# screeplot(returns.pca)
varcontribution <- returns.pca$sdev^2/sum(returns.pca$sdev^2)
plot(varcontribution, xlab = "Component Number", ylab = "Variance Contribution",
    type = "l")
points(varcontribution)
cumvariance <- cumsum(returns.pca$sdev^2)/sum(returns.pca$sdev^2)
plot(cumvariance, xlab = "Component Number", ylab = "Cumulative Variance",
    type = "l")
points(cumvariance)
abline(h = 0.8, lwd = 1.5, col = "red")
abline(v = 8, lwd = 1.5, col = "red")
text(13, 0.7, "Keep 8 Principal Components")
set.seed(200)
returns.scores <- as.data.frame(returns.pca$scores)
returns.scores$VV <- returns$VV
returns.scores$u <- runif(n = nrow(returns.scores), min = 0, max = 1)

train.scores <- subset(returns.scores, u < 0.7)
test.scores <- subset(returns.scores, u >= 0.7)

df <- cbind(Data = c("Training set", "Validation set", "Total"), Samples = c(nrow(train.scores),
    nrow(test.scores), nrow(train.scores) + nrow(test.scores)))
knitr::kable(df, align = c("l", "r"), caption = "Training and Validation sampling")

# Linear Regression with PCA
pca.lm <- lm(VV ~ Comp.1 + Comp.2 + Comp.3 + Comp.4 + Comp.5 + Comp.6 +
    Comp.7 + Comp.8, data = returns.scores)
coef_pca <- round(coef(pca.lm), 5)
signs_pca <- ifelse(sign(coef_pca) == 1, "+", "-")
formula_pca <- as.character(formula(pca.lm))[3]
predictors_pca <- unlist(strsplit(formula_pca, split = "+", fixed = T))
Betas_pca <- paste(abs(coef_pca[2:length(coef_pca)]), "*", predictors_pca)
pcaeqn <- paste("VV = ", paste(coef_pca[1], paste(paste(signs_pca[2:length(signs_pca)],
    Betas_pca), collapse = " ")))
```

```r
summary(pca.lm)
vifpc <- as.data.frame(vif(pca.lm))
vifpc$PC <- row.names(vifpc)
colnames(vifpc) <- c("VIF", "PC")
row.names(vifpc) <- NULL
knitr::kable(vifpc[, c("PC", "VIF")], caption = "VIF of Principle components")
pca.lm.MAE <- mean(abs(pca.lm$residuals))
pca.test <- predict(pca.lm, newdata = test.scores)
pca.test.MAE <- mean(abs(test.scores$VV - pca.test))

MAEdf <- data.frame(Model = "PCA model", Train.MAE = pca.lm.MAE, Test.MAE = pca.test.MAE)
knitr::kable(MAEdf, caption = "MAE of PCA model")
set.seed(200)
returns$u <- returns.scores$u
returns.train <- subset(returns, u >= 0.7)
returns.test <- subset(returns, u < 0.7)
# df <- cbind(Data = c('Training set', 'Validation set','Total'),
# Samples =
# c(nrow(returns.train),nrow(returns.test),nrow(returns.train) +
# nrow(returns.test))) knitr::kable(df,align = c('l','r'),caption =
# 'Training and Validation sampling of Raw returns')

# Arbitary model
model.1 <- lm(VV ~ GS + DD + DOW + HON + HUN + JPM + KO + MMM + XOM, data = returns.train)
coef_model.1 <- round(coef(model.1), 5)
signs_model.1 <- ifelse(sign(coef_model.1) == 1, "+", "-")
formula_model.1 <- as.character(formula(model.1))[3]
predictors_model.1 <- unlist(strsplit(formula_model.1, split = "+", fixed = T))
Betas_model.1 <- paste(abs(coef_model.1[2:length(coef_model.1)]), "*",
    predictors_model.1)
model.1eqn <- paste("VV = ", paste(coef_model.1[1], paste(paste(signs_model.1[2:length(signs_model.1)],
    Betas_model.1), collapse = " ")))
# model.1 <- lm(VV~ GS+DD+DOW+HON+HUN +JPM + KO + MMM+ XOM, data =
# returns.train)
summary(model.1)
model1.train.MAE <- mean(abs(model.1$residuals))
model1.test <- predict(model.1, newdata = returns.test)
model1.test.MAE <- mean(abs(model1.test - returns.test$VV))

ArbMAEdf <- data.frame(Model = "Arbitary model", Train.MAE = model1.train.MAE,
    Test.MAE = model1.test.MAE)
knitr::kable(ArbMAEdf, caption = "MAE of Arbitary model")

# full model
model.2 <- lm(VV ~ ., data = returns.train[, -22])
coef_model.2 <- round(coef(model.2), 5)
signs_model.2 <- ifelse(sign(coef_model.2) == 1, "+", "-")
formula_model.2 <- as.character(formula(model.2))[3]
predictors_model.2 <- unlist(strsplit(formula_model.2, split = "+", fixed = T))
Betas_model.2 <- paste(abs(coef_model.2[2:length(coef_model.2)]), "*",
    predictors_model.2)
model.2eqn <- paste("VV = ", paste(coef_model.2[1], paste(paste(signs_model.2[2:length(signs_model.2)],
    Betas_model.2), collapse = " ")))
```

```r
summary(model.2)

model2.train.MAE <- mean(abs(model.2$residuals))
model2.test <- predict(model.2, newdata = returns.test)
model2.test.MAE <- mean(abs(model2.test - returns.test$VV))

FullMAEdf <- data.frame(Model = "Full model", Train.MAE = model2.train.MAE,
    Test.MAE = model2.test.MAE)
knitr::kable(FullMAEdf, caption = "MAE of Full model")
models <- data.frame(Models = c("PCA", "Arbitary", "Full"), Equation = c(pcaeqn,
    model.1eqn, model.2eqn))
pander::pandoc.table(models, "Model equations", justify = "left", emphasixe.cols = 2)

ModelComp <- rbind(MAEdf, ArbMAEdf, FullMAEdf)
knitr::kable(ModelComp, caption = "Predictive performance of regression models")

# backward elimination
full.lm <- lm(VV ~ ., data = train.scores)
# summary(full.lm)

library(MASS)
backward.lm <- stepAIC(full.lm, direction = c("backward"), trace = F)


coef_backward.lm <- round(coef(backward.lm), 5)
signs_backward.lm <- ifelse(sign(coef_backward.lm) == 1, "+", "-")
formula_backward.lm <- as.character(formula(backward.lm))[3]
predictors_backward.lm <- unlist(strsplit(formula_backward.lm, split = "+",
    fixed = T))
Betas_backward.lm <- paste(abs(coef_backward.lm[2:length(coef_backward.lm)]),
    "*", predictors_backward.lm)
backward.lmeqn <- paste("VV = ", paste(coef_backward.lm[1], paste(paste(signs_backward.lm[2:length(signs
    Betas_backward.lm), collapse = " ")))


summary(backward.lm)
vif(backward.lm)


bkward.MAE.train <- mean(abs(backward.lm$residuals))
bkward.test <- predict(backward.lm, newdata = test.scores)
bkward.MAE.test <- mean(abs(bkward.test - test.scores$VV))

BkwardMAEdf <- data.frame(Model = "Backward", Train.MAE = bkward.MAE.train,
    Test.MAE = bkward.MAE.test)
ModelComp <- rbind(ModelComp, BkwardMAEdf)
knitr::kable(ModelComp, caption = "Comparison of regression models")
```