Assignment #6
Jennifer M. Wanat

Introduction:

In this report our objective is to use both linear regression and Principal Components Analysis to create models using the log-returns of individual stocks to explain the variation in the log-returns of the market index.

After an initial data preparation, an exploratory data analysis of the data set was completed by examining the correlation of individual stocks against the market index. Two linear regression models were creating using different numbers of predictors. Principal component analysis was used to reduce dimensionality and remediate multicollinearity, and the analysis was used to select principal components for a third linear regression model. Finally, an automated variable selection technique was utilized the training data to select principal components for a fourth linear regression model. The generated model summaries and parameters were provided, and assessed for predictive accuracy by examining the mean absolute error.

The data set was randomly split into a training set and test set to allow for cross-validation of the predictive models. The predictive accuracy of each model was also assessed with the test data set. The analysis was conducted with the R programming language.

Data:

The stock portfolio data set contains 23 variables with 502 observations.

Section 1: Data Preparation

In order to prepare the data set for analysis, the date entries were converted from a string to dd-Mon-yy date format. The data set was then sorted by the date and placed into a new data frame called sorted. The sort will allow for the calculation of the return, which is the natural logarithm of the daily closing price to the next daily closing price. This was conducted for all 21 stocks and saved into a new data frame called returns.

Section 2: Exploratory Data Analysis of Correlations

The correlation matrix was computed for the returns data frame. A bar plot was constructed to visual the correlation between all of the stocks against the Vanguard Large Cap Index (ticker VV). As VV is a large cap index fund there is positive correlation with all stocks in the portfolio. Only two stocks, DPS at 0.4435 and MPC at 0.47312, had a correlation less than 0.5.

Figure 2.1: Bar plot of stock correlation against Vanguard Large Cap Index (VV).

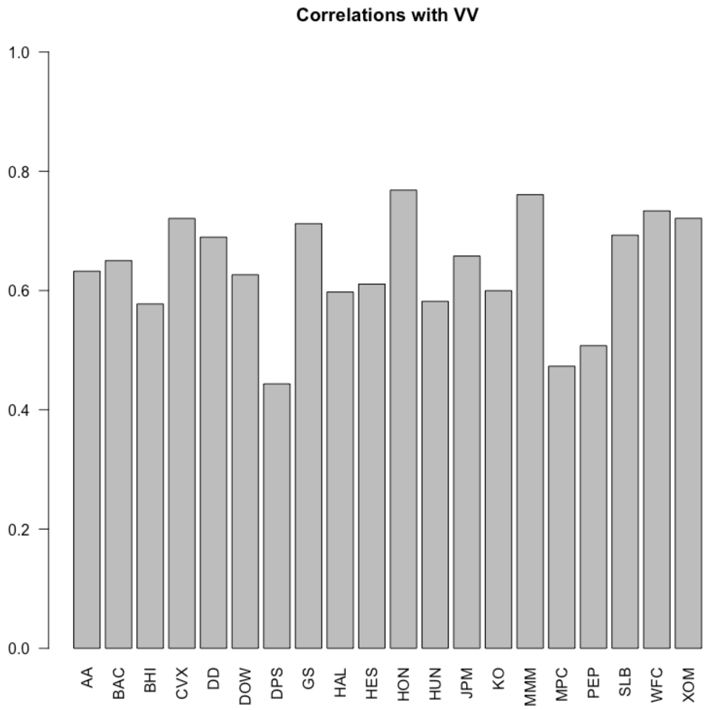
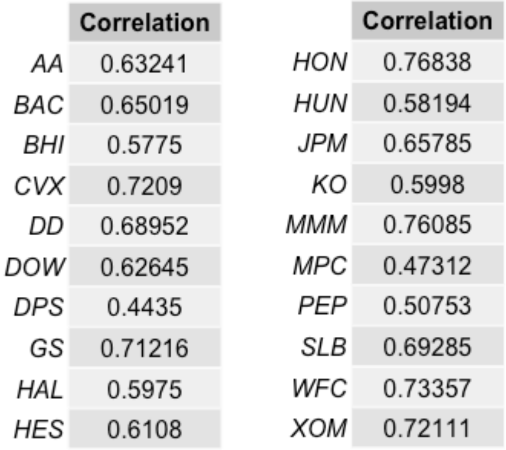


Table 2.1: Correlation values of stock against VV.

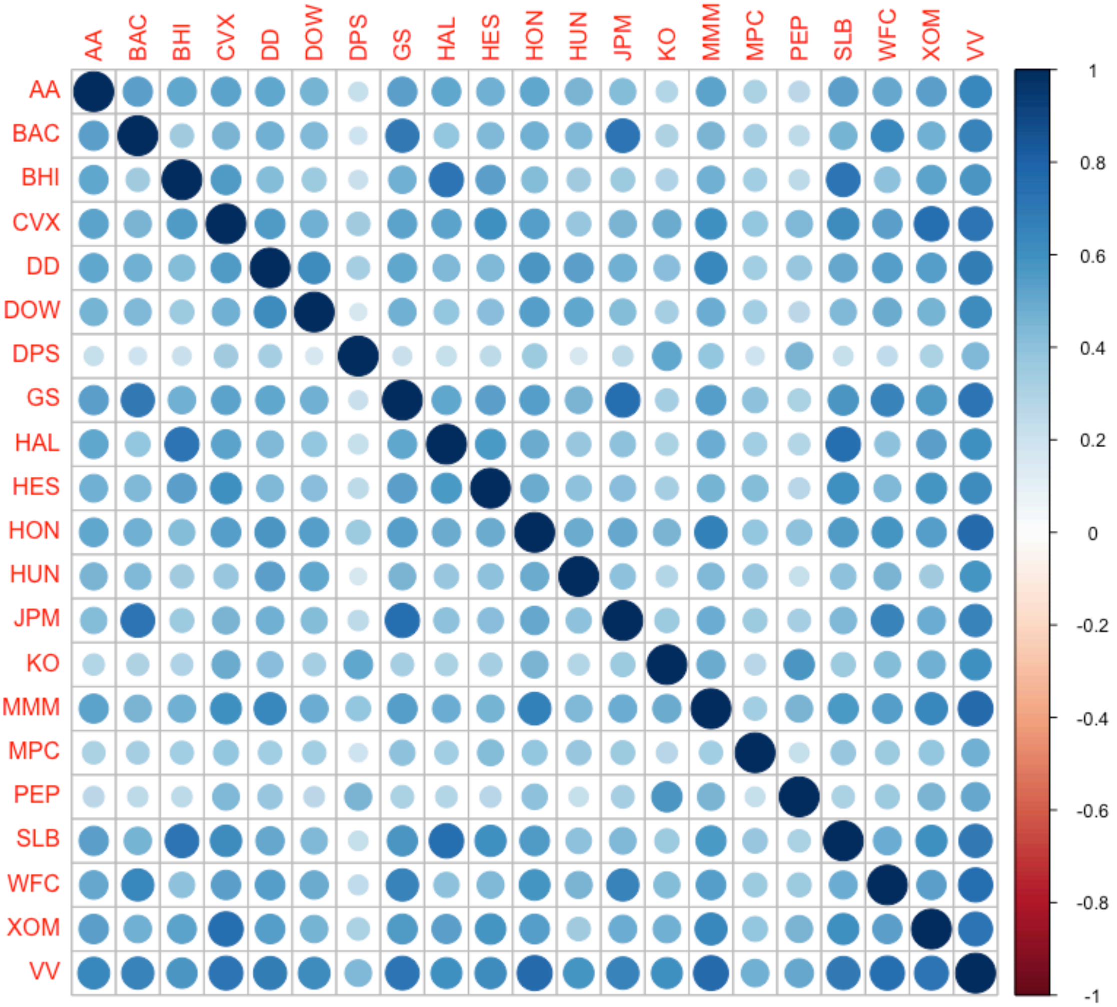


	Correlation		Correlation
AA	0.63241	HON	0.76838
BAC	0.65019	HUN	0.58194
BHI	0.5775	JPM	0.65785
CVX	0.7209	KO	0.5998
DD	0.68952	MMM	0.76085
DOW	0.62645	MPC	0.47312
DPS	0.4435	PEP	0.50753
GS	0.71216	SLB	0.69285
HAL	0.5975	WFC	0.73357
HES	0.6108	XOM	0.72111

Section 3: Data Visualization of Correlations

The correlation matrix was constructed into a graphical display. The correlation coefficient is proportional to the color and size of the circle. Positive correlation is blue and negative correlation is red. The large dark blue circle visualized diagonally across the plot represents a correlation of 1, and it is the correlation of the stock against itself.

Table 3.1: Graphical representation of stock correlation.



The variance inflation factor (VIF) estimates how much the variance of a predictor is inflated and a high value reflects multicollinearity. A predictor with high correlation to many predictors will have a high VIF value and conversely, a predictor with low correlation to many predictors will have a low VIF value. Stocks DPS, MPC and PEP should have low VIF values and stocks GS, SLB, and XOM should have high VIF values.

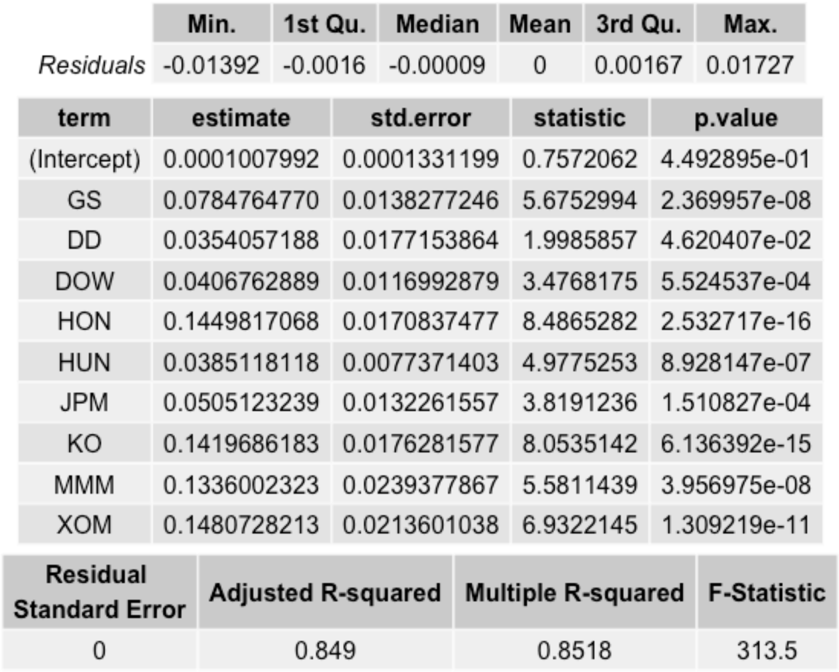
Section 4: Modeling Exploratory Data Analysis

Two naïve models, a small model with nine predictors and a full model with all 20 predictors, were created as part of the EDA. The returns data set was used. The model summaries are:

$$\text{Small Model: } E(VV) = b_0 + b_1GS + b_2DD + b_3DOW + b_4HON + b_5HUN + b_6JPM + b_7KO + b_8MMM + b_9XOM$$

$$E(VV) = 0.0001008 + 0.0784765\ GS + 0.0354057\ DD + 0.0406763\ DOW + 0.1449817\ HON + 0.0385118\ HUN + 0.0505123\ JPM + 0.1419686\ KO + 0.1336002\ MMM + 0.1480728\ XOM$$

Table 4.1: Small model.



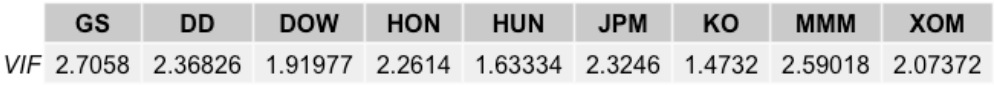
	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Residuals	-0.01392	-0.0016	-0.00009	0	0.00167	0.01727

term	estimate	std.error	statistic	p.value
(Intercept)	0.0001007992	0.0001331199	0.7572062	4.492895e-01
GS	0.0784764770	0.0138277246	5.6752994	2.369957e-08
DD	0.0354057188	0.0177153864	1.9985857	4.620407e-02
DOW	0.0406762889	0.0116992879	3.4768175	5.524537e-04
HON	0.1449817068	0.0170837477	8.4865282	2.532717e-16
HUN	0.0385118118	0.0077371403	4.9775253	8.928147e-07
JPM	0.0505123239	0.0132261557	3.8191236	1.510827e-04
KO	0.1419686183	0.0176281577	8.0535142	6.136392e-15
MMM	0.1336002323	0.0239377867	5.5811439	3.956975e-08
XOM	0.1480728213	0.0213601038	6.9322145	1.309219e-11

Residual Standard Error	Adjusted R-squared	Multiple R-squared	F-Statistic
0	0.849	0.8518	313.5

All components have a low standard error. All predictors have a significant p-value less than 0.01, except for DD which is non-significant. The small model has an adjusted R-squared of 0.849, which means that 84.9% of the variation of the response variable is explained by the predictor variables. The VIF values are low and less than three.

Table 4.2: Small model VIF values.

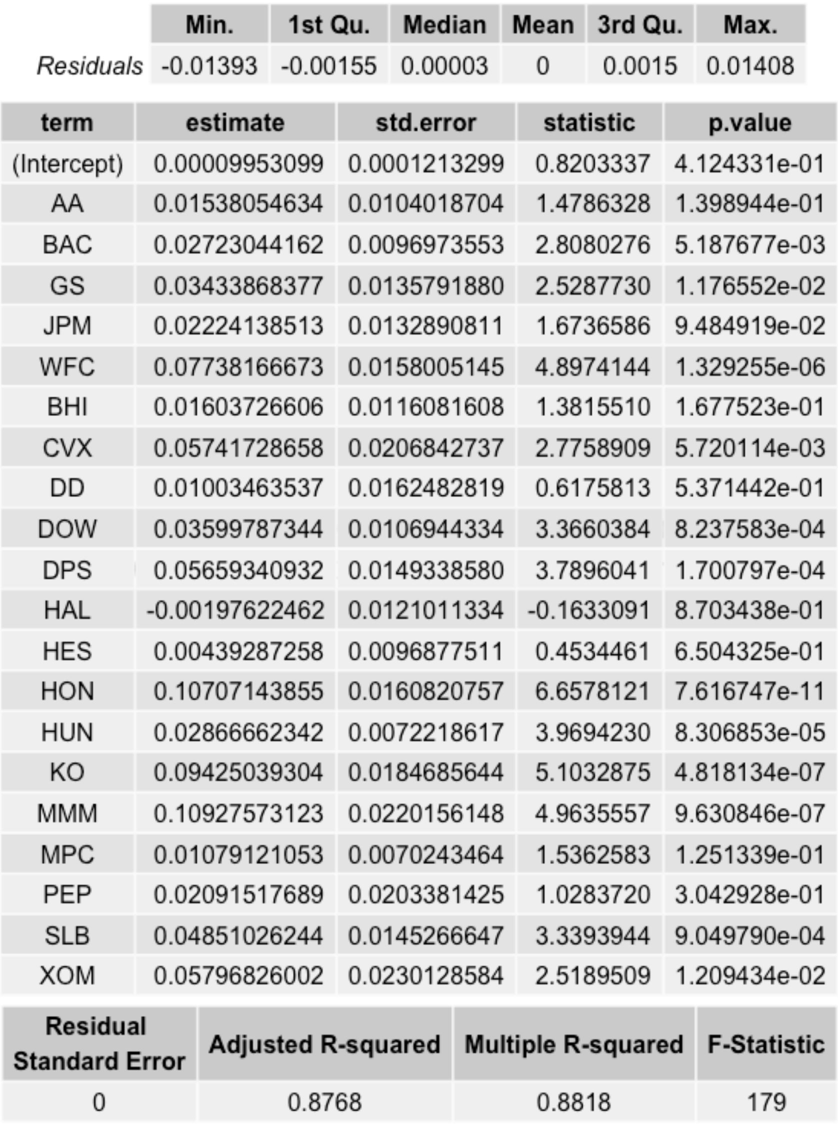


	GS	DD	DOW	HON	HUN	JPM	KO	MMM	XOM
VIF	2.7058	2.36826	1.91977	2.2614	1.63334	2.3246	1.4732	2.59018	2.07372

$$\text{Full Model:} \quad E(VV) = b_0 + b_1AA + b_2BAC + b_3GS + b_4JPM + b_5WFC + b_6BHI + b_7CVX + b_8DD + b_9DOW + b_{10}DPS + b_{11}HAL + b_{12}HES + b_{13}HON + b_{14}HUN + b_{15}KO + b_{16}MMM + b_{17}MPC + b_{18}PEP + b_{19}SLB + b_{20}XOM$$

$$E(VV) = 0.00009953 + 0.01538055AA + 0.02723044\ BAC + 0.03433868\ GS + 0.02224139\ JPM + 0.07738167\ WFC + 0.01603727\ BHI + 0.05741729\ CVX + 0.01003464\ DD + 0.03599787\ DOW + 0.05659341\ DPS - 0.00197622\ HAL + 0.00439287\ HES + 0.10707144\ HON + 0.02866662\ HUN + 0.09425039\ KO + 0.10927573\ MMM + 0.01079121\ MPC + 0.02091518\ PEP + 0.04851026\ SLB + 0.05796826\ XOM$$

Table 4.2: Full model.



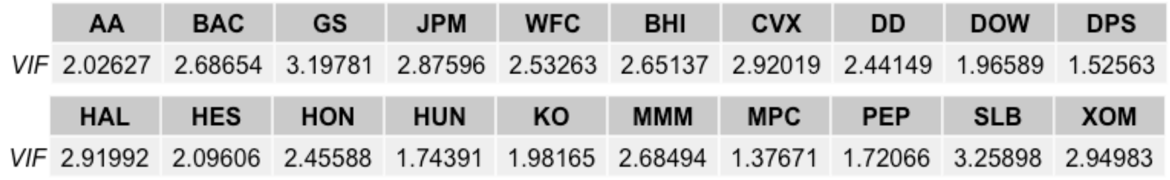
	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Residuals	-0.01393	-0.00155	0.00003	0	0.0015	0.01408

term	estimate	std.error	statistic	p.value
(Intercept)	0.00009953099	0.0001213299	0.8203337	4.124331e-01
AA	0.01538054634	0.0104018704	1.4786328	1.398944e-01
BAC	0.02723044162	0.0096973553	2.8080276	5.187677e-03
GS	0.03433868377	0.0135791880	2.5287730	1.176552e-02
JPM	0.02224138513	0.0132890811	1.6736586	9.484919e-02
WFC	0.07738166673	0.0158005145	4.8974144	1.329255e-06
BHI	0.01603726606	0.0116081608	1.3815510	1.677523e-01
CVX	0.05741728658	0.0206842737	2.7758909	5.720114e-03
DD	0.01003463537	0.0162482819	0.6175813	5.371442e-01
DOW	0.03599787344	0.0106944334	3.3660384	8.237583e-04
DPS	0.05659340932	0.0149338580	3.7896041	1.700797e-04
HAL	-0.00197622462	0.0121011334	-0.1633091	8.703438e-01
HES	0.00439287258	0.0096877511	0.4534461	6.504325e-01
HON	0.10707143855	0.0160820757	6.6578121	7.616747e-11
HUN	0.02866662342	0.0072218617	3.9694230	8.306853e-05
KO	0.09425039304	0.0184685644	5.1032875	4.818134e-07
MMM	0.10927573123	0.0220156148	4.9635557	9.630846e-07
MPC	0.01079121053	0.0070243464	1.5362583	1.251339e-01
PEP	0.02091517689	0.0203381425	1.0283720	3.042928e-01
SLB	0.04851026244	0.0145266647	3.3393944	9.049790e-04
XOM	0.05796826002	0.0230128584	2.5189509	1.209434e-02

Residual Standard Error	Adjusted R-squared	Multiple R-squared	F-Statistic
0	0.8768	0.8818	179

All components have a low standard error. Thirteen of the predictors have a significant p-value less than 0.1, except for JPM, DD, HAL, HES, MPC, PEP, and XOM which are non-significant. The small model has an adjusted R-squared of 0.8768. All stocks have a VIF less than 3, with the exception of GS (3.19781) and SLB (3.25898) in the full model. These two stocks may have multicollinearity issues.

Table 4.3: Full model VIF values.



	AA	BAC	GS	JPM	WFC	BHI	CVX	DD	DOW	DPS
VIF	2.02627	2.68654	3.19781	2.87596	2.53263	2.65137	2.92019	2.44149	1.96589	1.52563

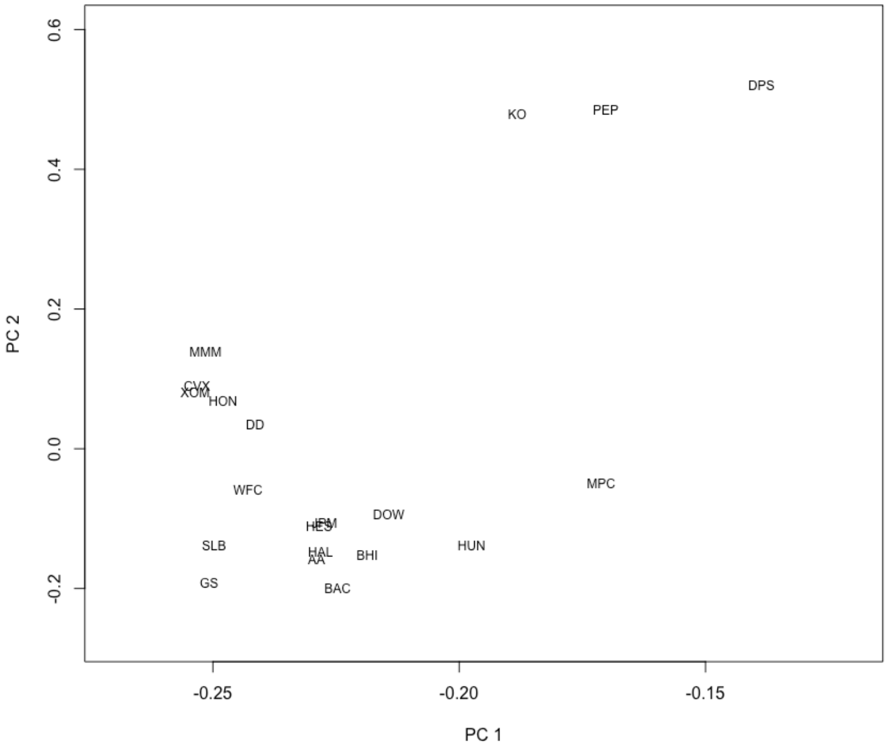
	HAL	HES	HON	HUN	KO	MMM	MPC	PEP	SLB	XOM
VIF	2.91992	2.09606	2.45588	1.74391	1.98165	2.68494	1.37671	1.72066	3.25898	2.94983

Section 5: Principal Component Analysis

Principal component analysis (PCA) is a technique used to reduce dimensionality of the predictor variables and remedy multicollinearity. PCA will be used to transform the predictors into a new set of orthogonal predictors, thereby reducing collinearity. PCA will also order, from highest to lowest, the predictors according to the proportion of the variance explained.

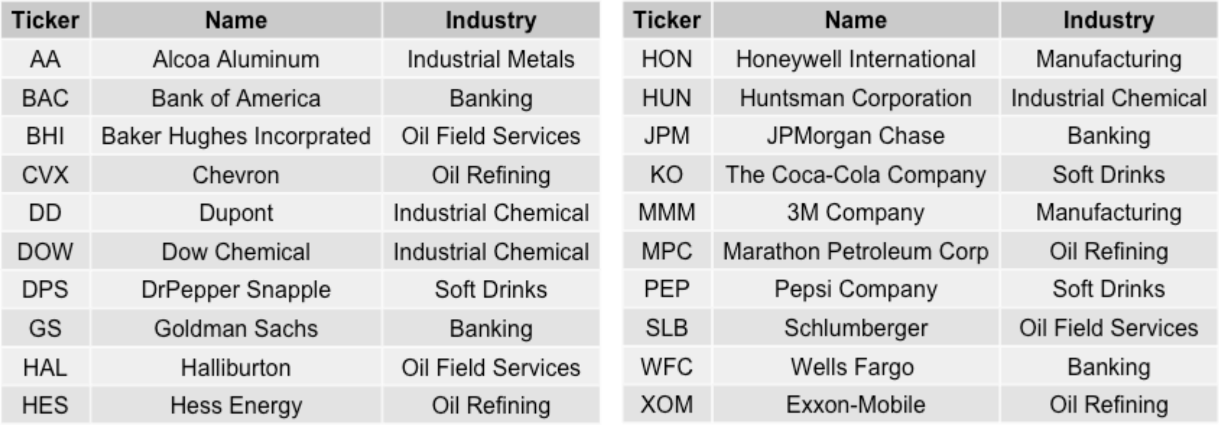
The loadings of the PCA analysis are the matrix of variable loadings (the columns contain the eigenvectors). The loadings for the first two principal components were examined in a plot.

Figure 5.1: Plot of loadings for the first two principal components.



Overall, the stocks are grouped by industry. Refer to Table 5.2 for industry classification.

Table 5.2: Table of stocks.

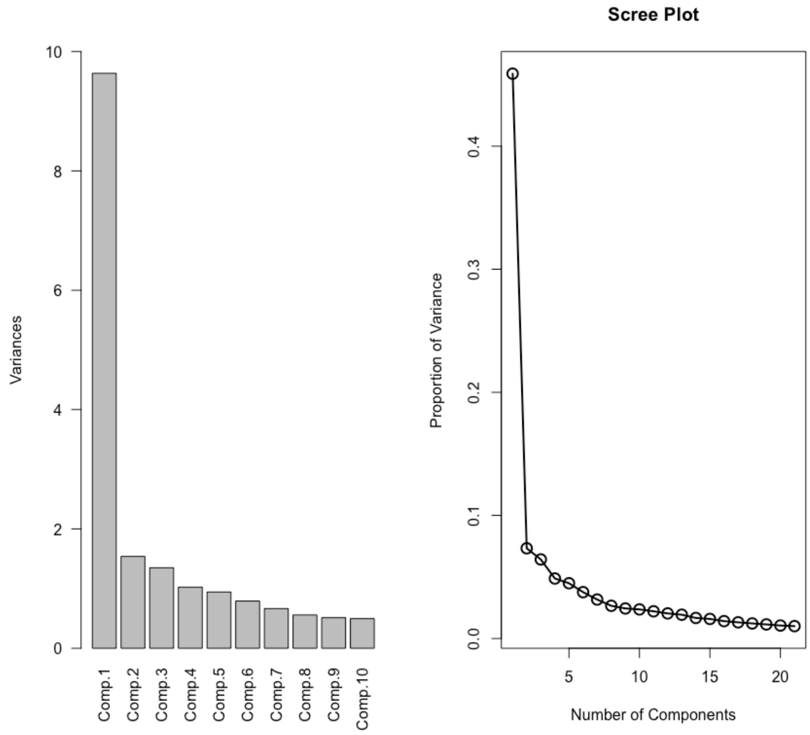


Ticker	Name	Industry	Ticker	Name	Industry
AA	Alcoa Aluminum	Industrial Metals	HON	Honeywell International	Manufacturing
BAC	Bank of America	Banking	HUN	Huntsman Corporation	Industrial Chemical
BHI	Baker Hughes Incorprated	Oil Field Services	JPM	JPMorgan Chase	Banking
CVX	Chevron	Oil Refining	KO	The Coca-Cola Company	Soft Drinks
DD	Dupont	Industrial Chemical	MMM	3M Company	Manufacturing
DOW	Dow Chemical	Industrial Chemical	MPC	Marathon Petroleum Corp	Oil Refining
DPS	DrPepper Snapple	Soft Drinks	PEP	Pepsi Company	Soft Drinks
GS	Goldman Sachs	Banking	SLB	Schlumberger	Oil Field Services
HAL	Halliburton	Oil Field Services	WFC	Wells Fargo	Banking
HES	Hess Energy	Oil Refining	XOM	Exxon-Mobile	Oil Refining

Section 6: Principal Component Analysis Visualization

After PCA is completed, the analysis can be visualized to help decide the number of principal components that should be kept.

Figure 6.1: Scree plots of PCA component variance.



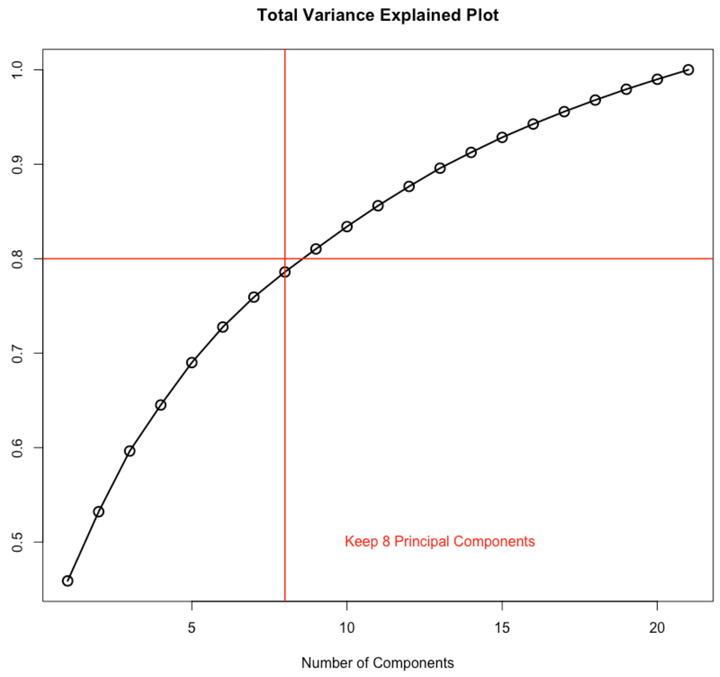
a)

b)

The plot of the left in Figure 6.1 is a bar plot of the variance for the first ten principal components. The scree plot on the right is the proportion of the variance explained by each principal component. The number of principal components to keep is located at the "elbow" of the plot where the line flattens out.

The cumulative proportion of variance by each principal component was plotted as a visual summary in Figure 6.2. A red horizontal line indicates 80% of the total variance. A red vertical line indicates which component was selected to explain approximately 80% of the total variation in the data. The first eight principal components will be kept.

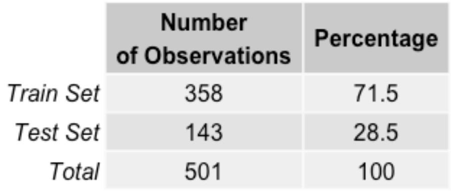
Figure 6.2: Total variance explained by component.



Section 7: Predictive Modeling with PCA

A linear regression model was made utilizing the first eight principal components from the PCA analysis. In order to assess model performance, the data set was split into a 70/30 train/test split. This would allow for cross-validation after a predictive model has been developed. The train data set would be used for in-sample model development and the test data set would be used for out-of-sample model assessment of predictive accuracy.

Table 7.1: Observation counts and percentage of train and test data sets.

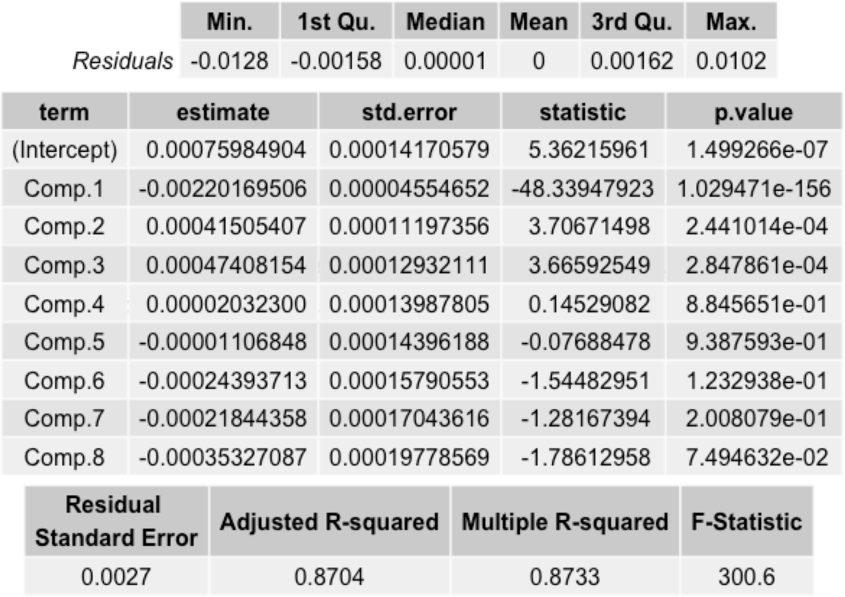


	Number of Observations	Percentage
Train Set	358	71.5
Test Set	143	28.5
Total	501	100

PCA1 Model: E(VV) = $b_0 + b_1$ Comp.1 + b_2 Comp.2 + b_3 Comp.3 + b_4 Comp.4 + b_5 Comp.5 + b_6 Comp.6 + b_7 Comp.7 + b_8 Comp.8

E(VV) = 0.00075985 - 0.00220170 Comp.1 + 0.00041505 Comp.2 + 0.00047408 Comp.3 + 0.00002032 Comp.4 - 0.00001107 Comp.5 - 0.00024394 Comp.6 - 0.00021844 Comp.7 - 0.00035327 Comp.8

Table 7.2: PCA1 model.



	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Residuals	-0.0128	-0.00158	0.00001	0	0.00162	0.0102

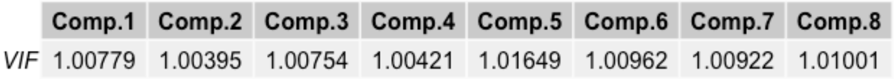
term	estimate	std.error	statistic	p.value
(Intercept)	0.00075984904	0.00014170579	5.36215961	1.499266e-07
Comp.1	-0.00220169506	0.00004554652	-48.33947923	1.029471e-156
Comp.2	0.00041505407	0.00011197356	3.70671498	2.441014e-04
Comp.3	0.00047408154	0.00012932111	3.66592549	2.847861e-04
Comp.4	0.00002032300	0.00013987805	0.14529082	8.845651e-01
Comp.5	-0.00001106848	0.00014396188	-0.07688478	9.387593e-01
Comp.6	-0.00024393713	0.00015790553	-1.54482951	1.232938e-01
Comp.7	-0.00021844358	0.00017043616	-1.28167394	2.008079e-01
Comp.8	-0.00035327087	0.00019778569	-1.78612958	7.494632e-02

Residual Standard Error	Adjusted R-squared	Multiple R-squared	F-Statistic
0.0027	0.8704	0.8733	300.6

All components have a low standard error. The first three components have a significant p-value less than 0.001, and component 8 has a p-value significance less than 0.1. Principal components 3 through 7 had non-significant t-statistics. The PCA 1 model has an adjusted R-squared of 0.8704.

The VIF values are low and approximately one, as the components have undergone PCA and the predictors have been orthogonally transformed to remediate multicollinearity.

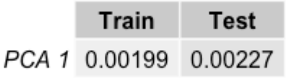
Table 7.3: PCA1 model VIF values.



	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8
VIF	1.00779	1.00395	1.00754	1.00421	1.01649	1.00962	1.00922	1.01001

The mean absolute error (MAE) is the average of all absolute errors. The MAE was lower with the train data set.

Table 7.4: PCA1 model MAE values.

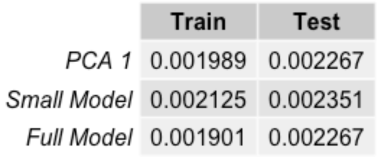


	Train	Test
PCA 1	0.00199	0.00227

Section 8: Predictive Modeling Comparison

The same random train/test split was applied to the raw log returns data set. The predictive accuracy of the PCA1 model was compared to the in-sample (train) and out-of-sample (test) predictive accuracy of small and full models using MAE.

Table 8.1: Model MAE values.



	Train	Test
PCA 1	0.001989	0.002267
Small Model	0.002125	0.002351
Full Model	0.001901	0.002267

The lowest MAE values are observed with the full model using in-sample data. The PCA 1 and full models were tied with the lowest MAE using out-of-sample data. The full model contains two predictors with VIF values greater than three, which indicates collinearity. The PCA 1 model utilized orthogonally transformed variables from PCA, therefore collinearity was remediated. The full model used 20 predictors and had an adjusted R-squared of 0.8768, while the PCA 1 model used eight principal components and had an adjusted R-squared of 0.8704.

Section 9: Predictive Modeling – Supervised Learning

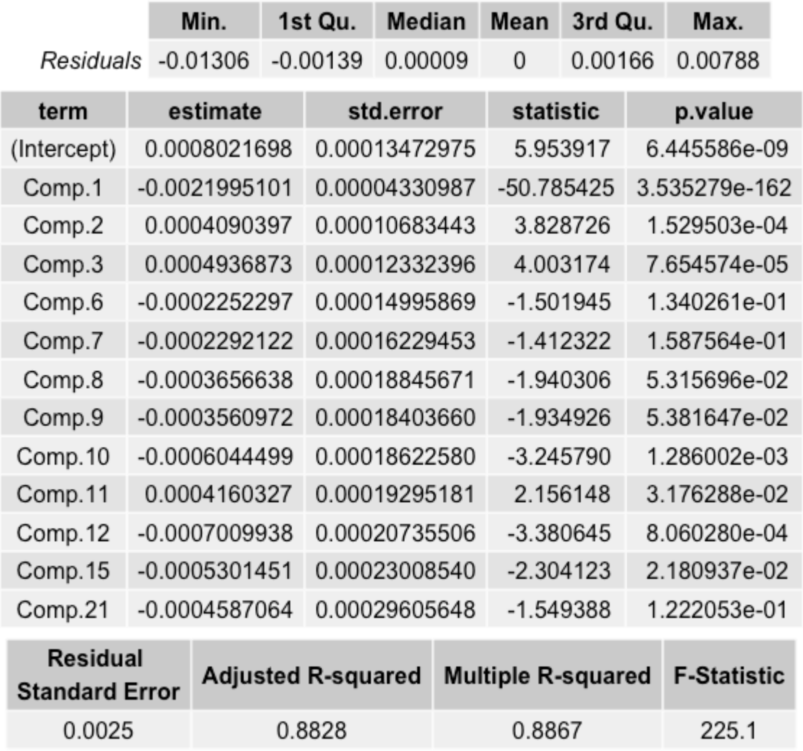
The PCA generated model, PCA 1, has no response variable. Models generated without a response variable are from a class of statistical learning methods called unsupervised learning. A model created with a response variable would be from a class of statistical learning methods called supervised learning. An example of a supervised learning method would be backward elimination. The MAE values from the backward elimination model will be compared to prior models for assessing predictive accuracy.

Backward elimination begins with linear regression containing all predictors. Predictor variables are removed until a model with the lowest AIC has been achieved. The following model was generated with backward elimination:

Backward Model: E(VV) = b_0 + b_1Comp.1 + b_2 Comp.2 + b_3 Comp.3 + b_4 Comp.6 + b_5 Comp.7 + b_6 Comp.8 + b_7 Comp.9 + b_8 Comp.10 + b_9 Comp.11 + b_{10} Comp.12 + b_{11} Comp.15 + b_{12} Comp.21

Backward Model: E(VV) = 0.00080217 - 0.00219951Comp.1 + 0.00040904 Comp.2 + 0.00049369 Comp.3 - 0.00022523 Comp.6 - 0.00022921 Comp.7 - 0.00036566 Comp.8 - 0.00035610 Comp.9 - 0.00060445 Comp.10 + 0.00041603 Comp.11 - 0.00070099 Comp.12 - 0.00053015 Comp.15 - 0.00045871 Comp.21

Table 9.1: Backward elimination model.



	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Residuals	-0.01306	-0.00139	0.00009	0	0.00166	0.00788

term	estimate	std.error	statistic	p.value
(Intercept)	0.0008021698	0.00013472975	5.953917	6.445586e-09
Comp.1	-0.0021995101	0.00004330987	-50.785425	3.535279e-162
Comp.2	0.0004090397	0.00010683443	3.828726	1.529503e-04
Comp.3	0.0004936873	0.00012332396	4.003174	7.654574e-05
Comp.6	-0.0002252297	0.00014995869	-1.501945	1.340261e-01
Comp.7	-0.0002292122	0.00016229453	-1.412322	1.587564e-01
Comp.8	-0.0003656638	0.00018845671	-1.940306	5.315696e-02
Comp.9	-0.0003560972	0.00018403660	-1.934926	5.381647e-02
Comp.10	-0.0006044499	0.00018622580	-3.245790	1.286002e-03
Comp.11	0.0004160327	0.00019295181	2.156148	3.176288e-02
Comp.12	-0.0007009938	0.00020735506	-3.380645	8.060280e-04
Comp.15	-0.0005301451	0.00023008540	-2.304123	2.180937e-02
Comp.21	-0.0004587064	0.00029605648	-1.549388	1.222053e-01

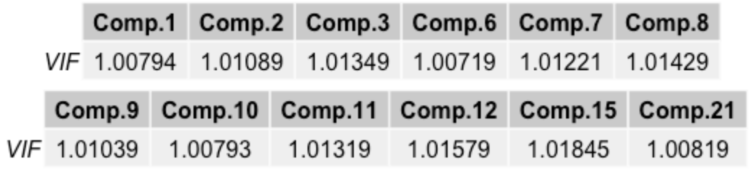
Residual Standard Error	Adjusted R-squared	Multiple R-squared	F-Statistic
0.0025	0.8828	0.8867	225.1

The backward elimination (BE) model selected 12 principal components to keep. The BE model used the same principal components as the PCA 1 model, but removed components 4 and 5, and added components 9, 10, 11, 12, 15, and 21.

All components have a low standard error. The first three components and component 12 have a significant p-value less than 0.001, and components 8 through 12 and 15 have a p-value significance between 0.1 and 0.001. Principal components 6, 7 and 21 had non-significant t-statistics. The BE model has an adjusted R-squared of 0.8828.

The VIF values are low and approximately one, as the components have undergone PCA and the predictors have been orthogonally transformed to remediate multicollinearity.

Table 9.2: Backward elimination model VIF values.



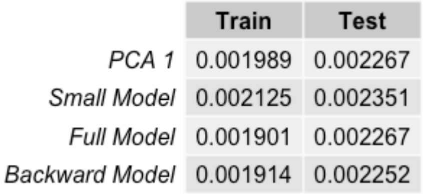
	Comp.1	Comp.2	Comp.3	Comp.6	Comp.7	Comp.8
VIF	1.00794	1.01089	1.01349	1.00719	1.01221	1.01429

	Comp.9	Comp.10	Comp.11	Comp.12	Comp.15	Comp.21
VIF	1.01039	1.00793	1.01319	1.01579	1.01845	1.00819

The MAE values from all model were placed into Table 9.3 for comparison purposes. The lowest MAE values are observed with the full model using in-sample data and with the BE model using out-of-sample data. The full model contains two predictors with VIF values greater than three, which indicates collinearity. The BE model utilized orthogonally transformed variables from

PCA, therefore collinearity was remediated. The full model used 20 predictors and had an adjusted R-squared of 0.8768, while the BE model used 12 principal components and had an adjusted R-squared of 0.8828.

Table 9.3: Model MAE values.



	Train	Test
PCA 1	0.001989	0.002267
Small Model	0.002125	0.002351
Full Model	0.001901	0.002267
Backward Model	0.001914	0.002252

Conclusion:

Principal component analysis was utilized to remediate multicollinearity within the stock portfolio data set. A backward elimination model with the generated principal components selected 12 predictors with reduced VIF values. The model had an adjusted R-squared of 0.8828, the lowest out-of-sample MAE, and second lowest in-sample MAE. The backward elimination model is recommended for use to explain the variation in the log-returns of the market index as a function of the log-returns of individual stocks.

References:

Everitt, B. (2010). Multivariable Modeling and Multivariate Analysis for the Behavioral Sciences. Boca Raton, FL: CRC Press, Taylor & Francis Group.

Fox, J. and Weisberg, S. (2011). An R Companion to Applied Regression, Second Edition. Thousand Oaks, CA: Sage Publications, Inc.

Lander, J. (2014). R for Everyone. Upper Saddle River, NJ: Addison-Wesley.

Code:

```
# Jennifer Wanat
# Fall 2017
# Assignment6.R

install.packages('corrplot', dependencies=TRUE)
install.packages('factoextra', dependencies=TRUE)
library(corrplot)
library(car)
library(RColorBrewer)
library(gridExtra)
library(broom)
library(ggplot2)
library(factoextra)
# library(MASS) added in section 9

###################### Section 1 ################################
my.path <- "~/Desktop/R/"
my.data <- read.csv(paste(my.path,'stock_portfolio.csv',sep=''),header=TRUE);
head(my.data)
str(my.data)

# Note Date is a string of dd-Mon-yy in R this is '%d-%B-%y';
my.data$RDate <- as.Date(my.data$Date,'%d-%B-%y');
sorted.df <- my.data[order(my.data$RDate),];
head(sorted.df)
AA <- log(sorted.df$AA[-1]/sorted.df$AA[-dim(sorted.df)[1]]);
# Manually check the first entry: log(9.45/9.23)

# Type cast the array as a data frame;
returns.df <- as.data.frame(AA);
returns.df$BAC <- log(sorted.df$BAC[-1]/sorted.df$BAC[-dim(sorted.df)[1]]);
returns.df$BHI <- log(sorted.df$BHI[-1]/sorted.df$BHI[-dim(sorted.df)[1]]);
returns.df$CVX <- log(sorted.df$CVX[-1]/sorted.df$CVX[-dim(sorted.df)[1]]);
returns.df$DD <- log(sorted.df$DD[-1]/sorted.df$DD[-dim(sorted.df)[1]]);
returns.df$DOW <- log(sorted.df$DOW[-1]/sorted.df$DOW[-dim(sorted.df)[1]]);
returns.df$DPS <- log(sorted.df$DPS[-1]/sorted.df$DPS[-dim(sorted.df)[1]]);
returns.df$GS <- log(sorted.df$GS[-1]/sorted.df$GS[-dim(sorted.df)[1]]);
returns.df$HAL <- log(sorted.df$HAL[-1]/sorted.df$HAL[-dim(sorted.df)[1]]);
returns.df$HES <- log(sorted.df$HES[-1]/sorted.df$HES[-dim(sorted.df)[1]]);
returns.df$HON <- log(sorted.df$HON[-1]/sorted.df$HON[-dim(sorted.df)[1]]);
returns.df$HUN <- log(sorted.df$HUN[-1]/sorted.df$HUN[-dim(sorted.df)[1]]);
returns.df$JPM <- log(sorted.df$JPM[-1]/sorted.df$JPM[-dim(sorted.df)[1]]);
returns.df$KO <- log(sorted.df$KO[-1]/sorted.df$KO[-dim(sorted.df)[1]]);
returns.df$MMM <- log(sorted.df$MMM[-1]/sorted.df$MMM[-dim(sorted.df)[1]]);
returns.df$MPC <- log(sorted.df$MPC[-1]/sorted.df$MPC[-dim(sorted.df)[1]]);
returns.df$PEP <- log(sorted.df$PEP[-1]/sorted.df$PEP[-dim(sorted.df)[1]]);
returns.df$SLB <- log(sorted.df$SLB[-1]/sorted.df$SLB[-dim(sorted.df)[1]]);
returns.df$WFC <- log(sorted.df$WFC[-1]/sorted.df$WFC[-dim(sorted.df)[1]]);
returns.df$XOM <- log(sorted.df$XOM[-1]/sorted.df$XOM[-dim(sorted.df)[1]]);
returns.df$VV <- log(sorted.df$VV[-1]/sorted.df$VV[-dim(sorted.df)[1]]);

#################### Section 2 ######################################
# Compute correlation matrix for returns;
returns.cor <- cor(returns.df)
returns.cor[,c('VV')]
grid.table(round(returns.cor[,c('VV')], 5), rows = names(returns.cor[,c('VV')]), cols
=c('Correlation'))
#same table, but transposed
grid.table(t(round(returns.cor[,c('VV')], 5)), cols = names(returns.cor[,c('VV')]), rows
=c('Correlation'))

#This one is not working right now.
grid.table(t(round(returns.cor[1:10,c('VV')], 5)), rows = names(returns.cor[1:10,c('VV')]),
cols =c('Correlation'))
grid.table(t(round(returns.cor[11:20,c('VV')], 5)), rows =
names(returns.cor[11:20,c('VV')]), cols =c('Correlation'))

#this one is working right now
grid.table(round(returns.cor[1:10,c('VV')], 5), rows = names(returns.cor[1:10,c('VV')]),
cols =c('Correlation'))
grid.table(round(returns.cor[11:20,c('VV')], 5), rows = names(returns.cor[11:20,c('VV')]),
cols =c('Correlation'))

# Barplot the last column to visualize magnitude of correlations;
barplot(returns.cor[1:20,c('VV')],las=2,ylim=c(0,1.0))
title('Correlations with VV')

################### Section 3 #####################################
# Make correlation plot for returns;
# If you need to install corrplot package; Note how many dependencies this package has;
corrplot(returns.cor)

################# Section 4 ######################################
# load car package
#This code is located at the beginning of the file
# Fit some model
model.1 <- lm(VV ~ GS+DD+DOW+HON+HUN+JPM+KO+MMM+XOM,
data=returns.df)
summary(model.1)
vif(model.1)
```

```
#create table for model 1 VIF
vif.model.1 <- round(vif(model.1),5)
#grid.table(vif.model.1, rows = c('GS', 'DD', 'DOW', 'HON', 'HUN', 'JPM', 'KO', 'MMM',
'XOM'),
#          cols = c('VIF'))
#This also works
grid.table(t(vif.model.1), cols = names(vif.model.1), rows =c('VIF'))


#Function for model summary statistics
model.summary <- function(model){
  residualse.summary <- round(summary(lm(model))$sigma,4)
  adjrs.summary <- round(summary(lm(model))$adj.r.squared,4)
  multrs.summary <- round(summary(lm(model))$r.squared,4)
  fstat.summary <- round(unname(summary(lm(model))$fstatistic)[1], 1)
  return(c(residualse.summary, adjrs.summary, multrs.summary, fstat.summary))
}


#Model residual summary table
grid.table(t(round(summary(model.1$residuals), 5)), rows = c('Residuals'))
#Model summary table of coefficient
m1.coefficients <-  tidy(model.1)
grid.table(m1.coefficients, row = NULL)
#Table of model ANOVA
model1.summary <- t(model.summary(model.1))
colnames(model1.summary) <- c('Residual \nStandard Error', 'Adjusted R-
squared','Multiple R-squared','F-Statistic')
grid.table(model1.summary)


# Fit the full model
model.2 <- lm(VV ~

AA+BAC+GS+JPM+WFC+BHI+CVX+DD+DOW+DPS+HAL+HES+HON+HUN+KO+M
MM+MPC+PEP+SLB+XOM,
             data=returns.df)
summary(model.2)
vif(model.2)
#create table for model 1 VIF
vif.model.2 <- round(vif(model.2),5)

grid.table(t(vif.model.2[1:10]), cols = names(vif.model.2[1:10]), rows =c('VIF'))
grid.table(t(vif.model.2[11:20]), cols = names(vif.model.2[11:20]), rows =c('VIF'))
```

```
#Model residual summary table
grid.table(t(round(summary(model.2$residuals), 5)), rows = c('Residuals'))
#Model summary table of coefficient
m2.coefficients <-  tidy(model.2)
grid.table(m2.coefficients, row = NULL)
#Table of model ANOVA
model2.summary <- t(model.summary(model.2))
colnames(model2.summary) <- c('Residual \nStandard Error', 'Adjusted R-
squared','Multiple R-squared','F-Statistic')
grid.table(model2.summary)

################ Section 5 #####################################
returns.pca <- princomp(x=returns.df[,-21],cor=TRUE)
# See the output components returned by princomp();
names(returns.pca)
pc.1 <- returns.pca$loadings[,1];
pc.2 <- returns.pca$loadings[,2];

ticker <- c(names(sorted.df)[2:22])

industry <- c('Industrial Metals',
              'Banking',
              'Oil Field Services',
              'Oil Refining',
              'Industrial Chemical',
              'Industrial Chemical',
              'Soft Drinks',
              'Banking',
              'Oil Field Services',
              'Oil Refining',
              'Manufacturing',
              'Industrial Chemical',
              'Banking',
              'Soft Drinks',
              'Manufacturing',
              'Oil Refining',
              'Soft Drinks',
              'Oil Field Services',
              'Banking',
              'Oil Refining',
              'Market Index')

name <- c('Alcoa Aluminum',
      'Bank of America',
      'Baker Hughes Incorprated',
      'Chevron',
      'Dupont',
      'Dow Chemical',
      'DrPepper Snapple',
      'Goldman Sachs',
      'Halliburton',
      'Hess Energy',
      'Honeywell International',
      'Huntsman Corporation',
      'JPMorgan Chase',
      'The Coca-Cola Company',
      '3M Company',
      'Marathon Petroleum Corp',
      'Pepsi Company',
      'Schlumberger',
      'Wells Fargo ',
      'Exxon-Mobile',
      'Vanguard Large Cap Index')

color <- c('yellow',
       'darkblue',
       'green',
       'purple',
       'lightblue',
       'lightblue',
       'red',
       'darkblue',
       'green',
       'purple',
       'orange',
       'lightblue',
       'darkblue',
       'red',
       'orange',
       'purple',
       'red',
       'green',
       'darkblue',
       'purple',
       'lightgrey')

#This plot works but not color coded by industry
plot(-10,10,type='p',xlim=c(-0.27,-0.12),ylim=c(-0.27,0.6),xlab='PC 1',ylab='PC 2')
text(pc.1,pc.2,labels=names(pc.1),cex=0.75)

#table of ticker, names and industry
```

```
industry.table <- cbind(ticker, name, industry)
colnames(industry.table) <- c("Ticker", "Name", "Industry")
grid.table(industry.table[1:10,])
grid.table(industry.table[11:20,])


################### Section 6 #####################################
par(mfrow = c(1,2))
# Plot the default scree plot;
plot(returns.pca, las=2, main= ' ', ylim=c(0,10))
#title('Variance of Components')

# Make Scree Plot
scree.values <- (returns.pca$sdev^2)/sum(returns.pca$sdev^2);
plot(scree.values,xlab='Number of Components',ylab='Proportion of
Variance',type='l',lwd=2)
points(scree.values,lwd=2,cex=1.5)
title('Scree Plot')
par(mfrow = c(1,1))

# Make Proportion of Variance Explained
variance.values <- cumsum(returns.pca$sdev^2)/sum(returns.pca$sdev^2);
plot(variance.values,xlab='Number of Components',ylab='',type='l',lwd=2)
points(variance.values,lwd=2,cex=1.5)
abline(h=0.8,lwd=1.5,col='red')
abline(v=8,lwd=1.5,col='red')
text(13,0.5,'Keep 8 Principal Components',col='red')
title('Total Variance Explained Plot')

#This looks nice, returns overlay of the first two graphs below (variance and scree)
#need ggplot2 and factoextra libraries
#but not used
#fviz_eig(returns.pca)

############### Section 7 #########################################
# Create the data frame of PCA predictor variables;
return.scores <- as.data.frame(returns.pca$scores);
return.scores$VV <- returns.df$VV;
set.seed(123)
return.scores$u <- runif(n=dim(return.scores)[1],min=0,max=1);
head(return.scores)

# Split the data set into train and test data sets;
train.scores <- subset(return.scores,u<0.70);
test.scores <- subset(return.scores,u>=0.70);
dim(train.scores)
dim(test.scores)
dim(train.scores)+dim(test.scores)
dim(return.scores)

#Create table of observations in each data frame and calculate percentage split
total <- dim(return.scores)[1]
total.train <- dim(train.scores)[1]
total.test <- dim(test.scores)[1]

percent.total <- round((total/total)*100, 1)
percent.train <- round((total.train/total)*100, 1)
percent.test <- signif((total.test/total)*100, 3)

total.all <- c(total.train, total.test, total)
percent.all <- c(percent.train, percent.test, percent.total)

overview.split <- cbind(total.all, percent.all)
colnames(overview.split) <- c("Number\nof Observations", "Percentage")
rownames(overview.split) <- c("Train Set", "Test Set", "Total")
grid.table(overview.split)

# Fit a linear regression model using the first 8 principal components;
pca1.lm <- lm(VV ~
Comp.1+Comp.2+Comp.3+Comp.4+Comp.5+Comp.6+Comp.7+Comp.8,
            data=train.scores);
summary(pca1.lm)

#Model residual summary table
grid.table(t(round(summary(pca1.lm$residuals), 5)), rows = c('Residuals'))
#Model summary table of coefficient
pca1.coefficients <-  tidy(pca1.lm)
grid.table(pca1.coefficients, row = NULL)
#Table of model ANOVA
pca1.summary <- t(model.summary(pca1.lm))
colnames(pca1.summary) <- c('Residual \nStandard Error', 'Adjusted R-
squared','Multiple R-squared','F-Statistic')
grid.table(pca1.summary)

# Compute the Mean Absolute Error on the training sample;
pca1.mae.train <- mean(abs(train.scores$VV-pca1.lm$fitted.values));
pca1.vif <- round(vif(pca1.lm), 5)
#create a table of VIF
grid.table(t(pca1.vif), cols = names(pca1.vif), rows =c('VIF'))


# Score the model out-of-sample and compute MAE;
```

```
pca1.test <- predict(pca1.lm,newdata=test.scores);
pca1.mae.test <- mean(abs(test.scores$VV-pca1.test));

#create table of MAE values for pca1 model 1
pca1.mae.table <- cbind(round(pca1.mae.train, 5), round(pca1.mae.test, 5))
colnames(pca1.mae.table) <- c('Train', 'Test')
rownames(pca1.mae.table) <- c('PCA 1')
grid.table(pca1.mae.table)



################# Section 8 #######################################
# Let's compare the PCA regression model with a 'raw' regression model;
# Create a train/test split of the returns data set to match the scores data set;
returns.df$u <- return.scores$u;
train.returns <- subset(returns.df,u<0.70);
test.returns <- subset(returns.df,u>=0.70);
dim(train.returns)
dim(test.returns)
dim(train.returns)+dim(test.returns)
dim(returns.df)


# Fit model.1 on train data set and score on test data;
model.1 <- lm(VV ~ GS+DD+DOW+HON+HUN+JPM+KO+MMM+XOM,
data=train.returns)
model1.mae.train <- mean(abs(train.returns$VV-model.1$fitted.values));
model1.test <- predict(model.1,newdata=test.returns);
model1.mae.test <- mean(abs(test.returns$VV-model1.test));

# Fit model.1 on train data set and score on test data;
model.2 <- lm(VV ~

BAC+GS+JPM+WFC+BHI+CVX+DD+DOW+DPS+HAL+HES+HON+HUN+KO+MMM
+MPC+PEP+SLB+XOM,
        data=train.returns)
model2.mae.train <- mean(abs(train.returns$VV-model.2$fitted.values));
model2.test <- predict(model.2,newdata=test.returns);
model2.mae.test <- mean(abs(test.returns$VV-model2.test));

#Present the MAE values from models pca1.lm, model.1, and model.2 in a table
pca1.mae <- cbind(round(pca1.mae.train, 6), round(pca1.mae.test, 6))
model1.mae <- cbind(round(model1.mae.train, 6), round(model1.mae.test, 6))
model2.mae <- cbind(round(model2.mae.train, 6), round(model2.mae.test, 6))
mae.table <- rbind(pca1.mae, model1.mae, model2.mae)
colnames(mae.table) <- c('Train', 'Test')
```

```
rownames(mae.table) <- c('PCA 1', 'Small Model', 'Full Model')
grid.table(mae.table)


############### Section 9 ########################################
full.lm <- lm(VV ~ ., data=train.scores);
summary(full.lm)

library(MASS)

backward.lm <- stepAIC(full.lm,direction=c('backward'))
summary(backward.lm)

#Model residual summary table
grid.table(t(round(summary(backward.lm$residuals), 5)), rows = c('Residuals'))
#Model summary table of coefficient
backward.coefficients <-  tidy(backward.lm)
grid.table(backward.coefficients, row = NULL)
#Table of model ANOVA
backward.summary <- t(model.summary(backward.lm))
colnames(backward.summary) <- c('Residual \nStandard Error', 'Adjusted R-squared','Multiple R-squared','F-Statistic')
grid.table(backward.summary)

backward.mae.train <- mean(abs(train.scores$VV-backward.lm$fitted.values));
backward.vif <- round(vif(backward.lm), 5)

#create a table of VIF
grid.table(t(backward.vif[1:6]), cols = names(backward.vif[1:6]), rows =c('VIF'))
grid.table(t(backward.vif[7:12]), cols = names(backward.vif[7:12]), rows =c('VIF'))

backward.test <- predict(backward.lm,newdata=test.scores);
backward.mae.test <- mean(abs(test.scores$VV-backward.test));

#Present the MAE values from models pca1.lm, model.1, model.2, and backward.lm in a table
backward.mae <- cbind(round(backward.mae.train, 6), round(backward.mae.test, 6))
mae.table2 <- rbind(pca1.mae, model1.mae, model2.mae, backward.mae)
colnames(mae.table2) <- c('Train', 'Test')
rownames(mae.table2) <- c('PCA 1', 'Small Model', 'Full Model', 'Backward Model')
grid.table(mae.table2)
```