Assignment #6: Principal Components in Predictive Modeling (100 points)

Data: The data for this assignment is the stock portfolio data set. This data will be made available by your instructor.

As we transition from linear regression analysis to multivariate analysis the methods and their implementations become more complex to understand and use. In order to help you better understand the material we will still perform some hands on learning using the methods with data, but we will do so in more of a tutorial manner in which we walk you through the analysis step-by-step while providing you with a majority of the code needed to perform the analysis. Students are expected to understand the analysis using the code provided, and then properly communicate the analysis and results through a properly written report.

Assignment Instructions:

In this assignment we will use Principal Components Analysis as a method of dimension reduction and as a remedial measure for multicollinearity in linear regression. Along the way we will also learn how to manipulate data in R and make some R graphics useful for these topics.

(1) Let's begin our analysis with some data prep. Our raw data consists of daily closing stock prices for twenty stocks and a large-cap index fund from Vanguard (VV). We will use the log-returns of the individual stocks to explain the variation in the log-returns of the market index. We will explore this concept using both linear regression and principal components analysis.

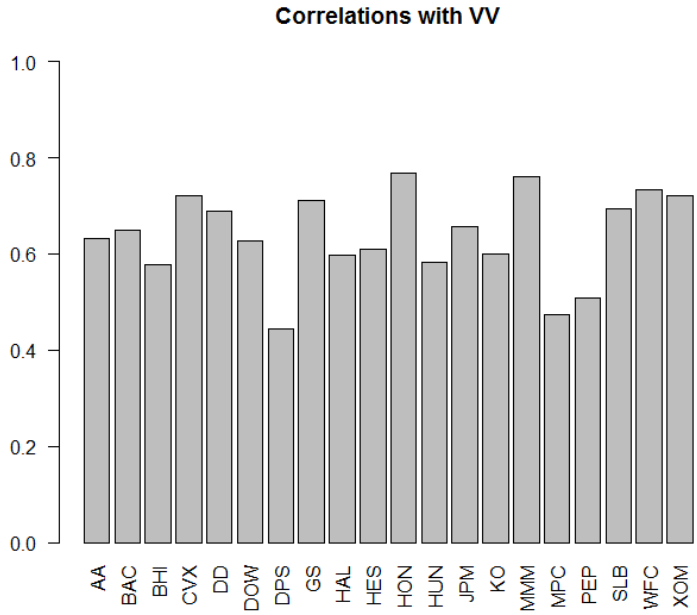
The following code will get your started. Note that you need to define the variable `my.path`.

```
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]]);
```

(2) Now that we have our data defined, let's begin our exploratory data analysis by examining the correlations. We will want to compute the complete correlation matrix, and then consider a visualizing a subset of those correlations for quick and easy comparison. Which row or column of the correlation matrix are we most interesting in?

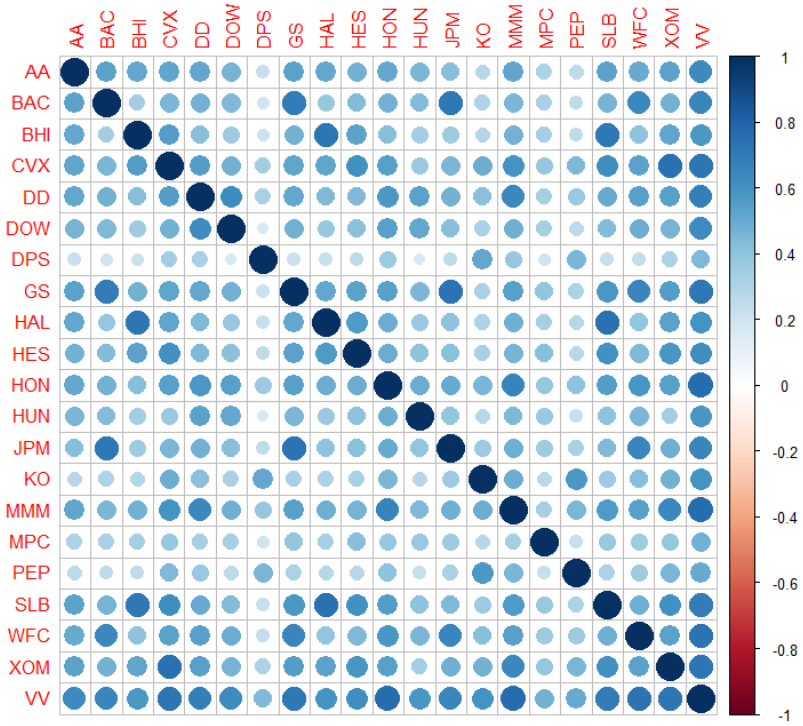
```
# Compute correlation matrix for returns;
returns.cor <- cor(returns.df)
returns.cor[,c('VV')]

# 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')
```



(3) How about we make an even fancier data visualization of the correlations? The corrplot will allow us to visualize all pairwise correlations in the data. Is the corrplot more useful or insightful than our simple barplot? Do we know the difference between a 'statistical graphic' and a 'data visualization'?

Note: You might need to install the `corrplot` package.

```
# Make correlation plot for returns;
# If you need to install corrplot package;  Note how many dependencies this package has;
install.packages('corrplot', dependencies=TRUE)

library(corrplot)
corrplot(returns.cor)
```



With respect to the concept of multicollinearity can we look at the corrplot and pick three stocks that should have low VIF values? Can we pick three stocks that should have high VIF values?

(4) In addition to statistical graphics and data visualizations we can use models as tools for exploratory data analysis. Modeling is an inherently iterative process, and we typically begin the modeling process by fitting some 'naïve models' that are nothing more than models that we believe are good starting points for the modeling process. Typically we might start with a small model and the full model as our two initial naïve models. The full model also allows us to compute the VIF value for every predictor variable.

Note: You will need the `car` package in order to use the `vif()` function.

```
# load car package
library(car)

# 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)

# Fit the full model
model.2 <- lm(VV ~
AA+BAC+GS+JPM+WFC+BHI+CVX+DD+DOW+DPS+HAL+HES+HON+HUN+KO+MMM+MPC+PEP+SLB+XOM,
data=returns.df)
summary(model.2)
vif(model.2)
```

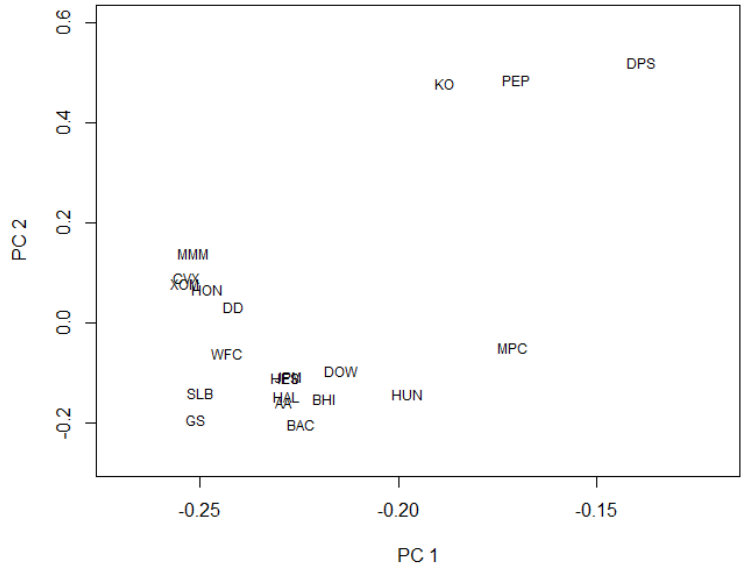
Should we have any multicollinearity concerns for either of these models? For what value of VIF should we be concerned to about multicollinearity?

(5) One remedy for multicollinearity is to transform the predictor variables using principal component analysis. Let's compute the principal components for the return data. We will use the `princomp()` function that is available in Base R.

Note that we are not going to explicitly standardize the data to mean zero and unit variance. The log-return transformation that we computed for each security is our standardization.

Let's plot the loadings for first two principal components from the principal components analysis. (What are the loadings?) When we plot the loadings, we can see relationships in the data. Do we see any groupings (or clusters) in the plot of the first two principal components? Any surprises?

```
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];
names(pc.1)

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)
```



Extra R Practice: Can you color code this plot by industry? Here are the ticker symbols and their industry. You do not need to use the colors that used in the table.

Ticker		Name	Industry	
1	AA	Alcoa Aluminum	Industrial - Metals	7
2	BAC	Bank of America	Banking	2
3	BHI	Baker Hughes Incorprated	Oil Field Services	3
4	CVX	Chevron	Oil Refining	4
5	DD	Dupont	Industrial - Chemical	5
6	DOW	Dow Chemical	Industrial - Chemical	5
7	DPS	DrPepper Snapple	Soft Drinks	6
8	GS	Goldman Sachs	Banking	2
9	HAL	Halliburton	Oil Field Services	3
10	HES	Hess Energy	Oil Refining	4
11	HON	Honeywell International	Manufacturing	1
12	HUN	Huntsman Corporation	Industrial - Chemical	5
13	JPM	JPMorgan Chase	Banking	2
14	KO	The Coca-Cola Company	Soft Drinks	6
15	MMM	3M Company	Manufacturing	1
16	MPC	Marathon Petroleum Corp	Oil Refining	4
17	PEP	Pepsi Company	Soft Drinks	6
18	SLB	Schlumberger	Oil Field Services	3
19	WFC	Wells Fargo	Banking	2
20	XOM	Exxon-Mobile	Oil Refining	4
21	VV	Vanguard Large Cap Index	Market Index	9

(6) If we want to use PCA for dimension reduction, then after we compute the principal components we need to decide how many principal components to keep. How many principal components do you think that we should keep? Why? (Hint: What decision rules should we use to determine the number of principal components to keep?) Later in the assignment we will use the first eight principal components. Why eight?

One tool for deciding the number of principal components to keep is to make a scree plot. Note that R will make a scree plot by default when plotting a PCA object.

```
# Plot the default scree plot;
plot(returns.pca)
```

The default scree plot is a scree plot, but it is not that great. Maybe we should know how to make our own plots and make plots that are better than the default scree plot. Let's make two common plots associated with PCA, and let's make them look nicer than the default scree plot.

Note that we need to understand the PCA output and how to make plots in R in order to make these plots.

```
# Make Scree Plot
scree.values <- (returns.pca$sdev^2)/sum(returns.pca$sdev^2);

plot(scree.values,xlab='Number of Components',ylab='',type='l',lwd=2)
points(scree.values,lwd=2,cex=1.5)
title('Scree Plot')



# 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')
```

Use these two plots to decide how many principal components to keep.

(7) Now let's use principal components in predictive modeling. The predictor variables for our predictive model will be the PCA scores. (What are the scores?) In order for us to be building predictive models, we need to have a train data set and a test data set so let's split our data into train and test data sets manually by generating a uniform(0,1) random variable and attaching it to the data frame.

In order to simplify this assignment we will compute and score the principal components on the whole data set, and then we will split it into training and test data sets. In a production environment we would have estimated our principal components on our training data set, stored our eigenvectors, and then scored them on each new data set (out-of-sample) as needed. However, the mechanics of those computations would distract us from the important parts of

this assignment. If you understand the difference between what we are doing in this assignment, and what we would do in a production environment, then you are welcome to try the latter as a separate practice exercise for your own personal development.

```
# Create the data frame of PCA predictor variables;
return.scores <- as.data.frame(returns.pca$scores);
return.scores$VV <- returns.df$VV;
return.scores$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)

# 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)

# Compute the Mean Absolute Error on the training sample;
pca1.mae.train <- mean(abs(train.scores$VV-pca1.lm$fitted.values));
vif(pca1.lm)

# 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));
```

Using our train and test data sets, let's fit a linear regression model using the first eight principal components and compute the Mean Absolute Error (MAE) for that model. We will call this model pca1.lm. Let's also score that model out-of-sample on our test data set and compute the out-of-sample MAE.

Note: The VIF values associated with every predictor variable in any principal components regression model should all be one. Why?

(8) Is our principal components regression model a 'better' model or the 'best' model? What does it mean for Model A to be better than Model B? How about in predictive modeling? When we compare models in predictive modeling we need to compare the models on an objective performance criterion.

Let's compute and compare the in-sample and out-of-sample predictive accuracy of our principal components regression model to Model #1 and Model #2 from earlier. We will create the matching train/test split of the raw log returns data, fit and score Model #1 and Model #2, and compute the appropriate MAE values to compare with pca1.lm.

```
# 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 so that they are easily compared and discussed in your paper. Discuss these results. Which model is the best model? Why?

(9) Principal components is an example of a class of Statistical Learning methods called 'Unsupervised Learning'. They are called unsupervised learning methods because they have no response variable. Statistical Learning methods that have a response variable are called 'Supervised Learning'. In this assignment we have looked at PCA from the traditional unsupervised learning perspective. Now we want to look at PCA in a supervised learning perspective, which will be more useful when using principal components in predictive modeling.

In predictive modeling we frequently wrap unsupervised learning methods into a supervised learning problem as a means to improve our predictive models. One example where we wrap an unsupervised learning method into a supervised learning problem is using PCA to reduce the

dimension of predictor variables in a linear regression model. Earlier in the assignment we selected eight principal components as the 'correct' or 'best' number of principal components to keep, and we fit the regression model pca1.lm using the first eight principal components. The decision to keep eight principal components was made using a decision rule from the standard unsupervised learning problem. Does our unsupervised decision rule translate to producing the best predictive model? More directly, is the eight principal components model the best predictive model? How else might we select the 'best' number of principal components to keep in our predictive modeling problem? Why don't we consider a supervised approach of using variable selection to select the number of principal components to keep, and which principal components to keep?

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

library(MASS)
backward.lm <- stepAIC(full.lm,direction=c('backward'))
summary(backward.lm)
backward.mae.train <- mean(abs(train.scores$VV-backward.lm$fitted.values));
vif(backward.lm)

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

How many principal components does the variable selection approach suggest that we keep? Which ones? How does this differ from our previous discussions about the number of principal components to keep? Create a new table where you add these new MAE results to the previous MAE results. How does our backward.lm model compare to all of the other models that we have fit in this assignment? Which model is best and why?

Assignment Document:

All assignment reports should conform to the standards and style of the report template provided to you. Results should be presented and discussed in an organized manner with the discussion in close proximity of the results. The report should not contain unnecessary results or information. The document should be submitted in pdf format. Name your file Assignment6_LastName.pdf.