Assignment 6

Introduction:

The purpose of this assignment is to understand how to use the Principal Components Analysis as a method of dimension reduction and as a remedial measure for multicollinearity in Ordinary Least Squares regression. Our data set consists of daily closing stock prices for twenty stocks and a large-cap index fun from Vanguard (VV). In order to evaluate the relationships between variables, we'll begin by taking the log-return of each variable which will allow us to compare each of the stock/index variables by a comparable measure. This will help us in understanding the correlation of each variable, enabling us to fit a proper regression model that will accurately predict the future values of the VV fund.

Results:

In utilizing the PearsonCorr procedure, we're able to understand the correlation of the log-return of each stock to the Vanguard Index which is shown in Table 1, below.

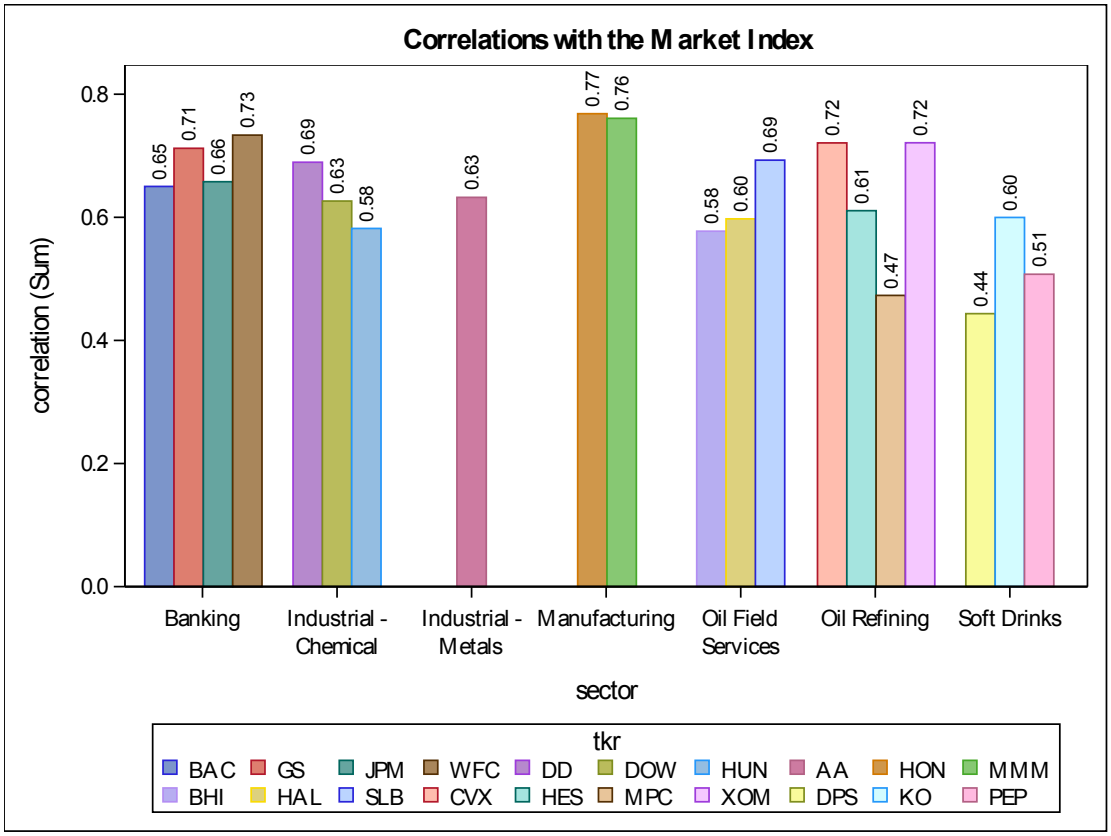
Table 1: Stock Correlation to Vanguard Index Fund

Observation	Correlation	Ticker	Sector
1	0.63241	AA	Industrial – Metals
2	0.65019	BAC	Banking
3	0.57750	BHI	Oil Field Services
4	0.72090	CVX	Oil Refining
5	0.68952	DD	Industrial – Chemical
6	0.62645	DOW	Industrial – Chemical
7	0.44350	DPS	Soft Drinks
8	0.71216	GS	Banking
9	0.59750	HAL	Oil Field Services
10	0.61080	HES	Oil Refining
11	0.76838	HON	Manufacturing
12	0.58194	HUN	Industrial – Chemical
13	0.65785	JPM	Banking

14	0.59980	KO	Soft Drinks
15	0.76085	MMM	Manufacturing
16	0.47312	MPC	Oil Refining
17	0.50753	PEP	Soft Drinks
18	0.69285	SLB	Oil Field Services
19	0.73357	WFC	Banking
20	0.72111	XOM	Oil Refining

In order to visualize these correlations, Figure 1 displays a plot of the ticker correlation to the Vanguard Market index, grouped by sector. What's interesting here is that the Banking and Manufacturing sectors appear to have similar correlations, whereas the Oil Refining and Soft Drinks sectors don't consist of a similar set of correlations.

Figure 1: Ticker Correlation to Vanguard Market Index by Sector



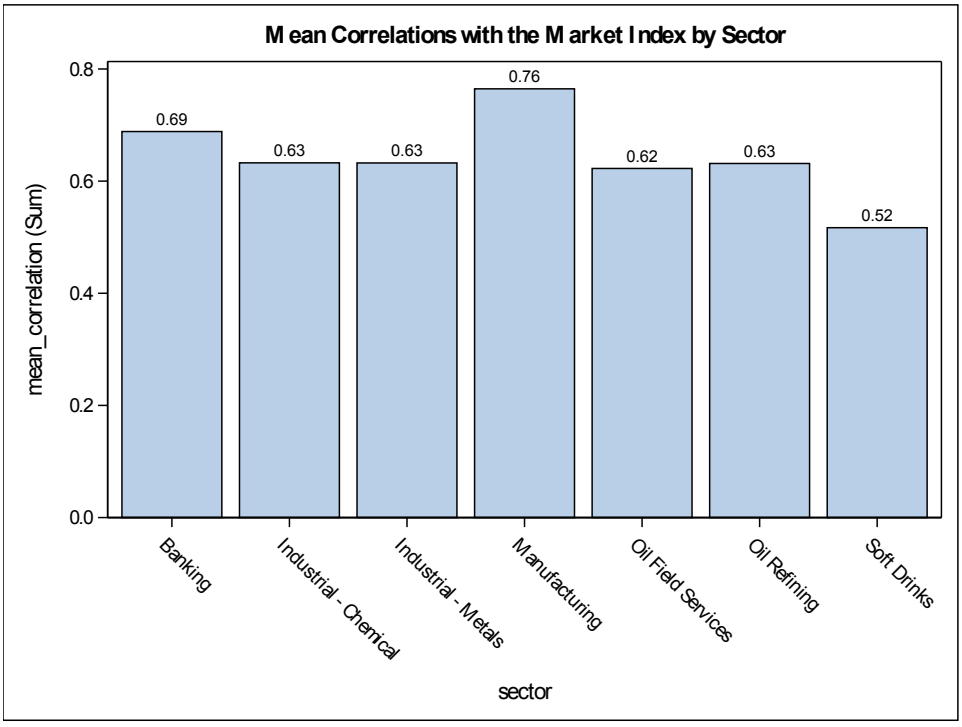
Due to the fact that we're not seeing consistent correlations across each of the sectors, we'll explore the mean correlation by sector, presented in Table 2, below.

Table 2: Mean Sector Correlation to Vanguard Index Fund

Observation	Sector	Type	Frequency	Mean Correlation
1	Banking	1	4	0.68844
2	Industrial – Chemical	1	3	0.63264
3	Industrial – Metals	1	1	0.63241
4	Manufacturing	1	2	0.76461
5	Oil Field Services	1	3	0.62262
6	Oil Refining	1	4	0.63148
7	Soft Drinks	1	3	0.51694

Figure 2, presents a graphical display of the results from Table 2. We can see here that the manufacturing sector appears to have the highest correlation, while the soft drinks sector has the lowest. What's interesting here is that all of the remaining sectors appear to have a similar correlation.

Figure 2: Mean Correlation with Market Index Vanguard by Sector



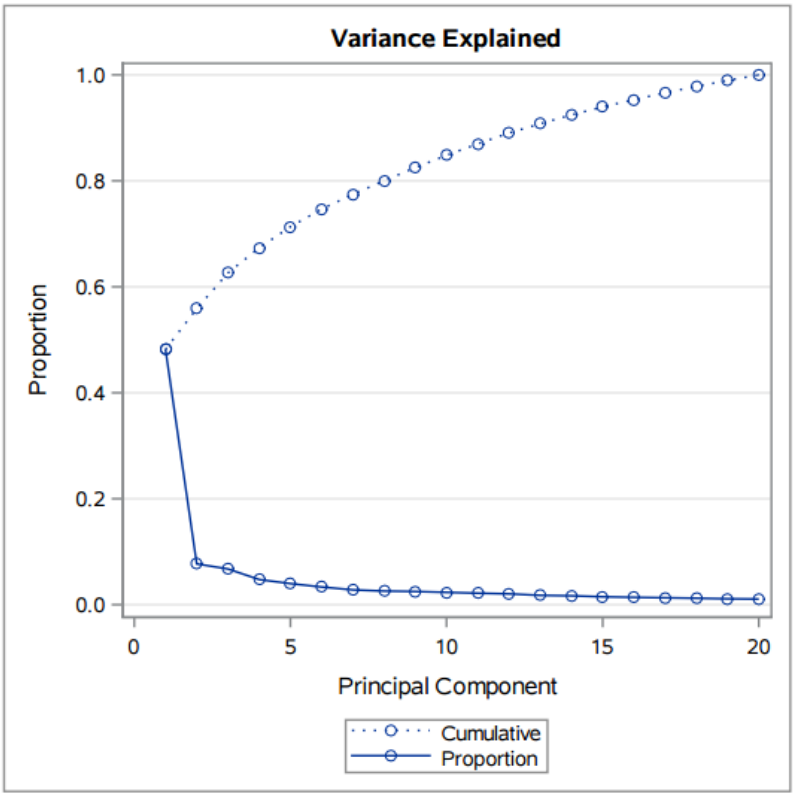
Our next step is creating a new data set which removes the Market Index, the dependent variable, and allows us to begin utilizing the principal component methods. We've utilized SAS to calculate the eigenvalues of our variables, which is shown below in Table 3. Additionally, we've chosen visually examine these results through a scree plot, presented in Figure-3. This will help us determine which of our principal components we should keep moving forward into our regression model.

Table 3: Eigenvalues of the Correlation Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	9.63645075	8.09792128	0.4818	0.4818
2	1.53852947	0.19109235	0.0769	0.5587
3	1.34743712	0.39975791	0.0674	0.6261
4	0.94767921	0.15217268	0.0474	0.6735
5	0.79550653	0.12909860	0.0398	0.7133
6	0.66640793	0.10798740	0.0333	0.7466
7	0.55842052	0.04567198	0.0279	0.7745
8	0.51274854	0.01590728	0.0256	0.8002
9	0.49684126	0.03250822	0.0248	0.8250
10	0.46433304	0.03089374	0.0232	0.8482
11	0.43343929	0.02568332	0.0217	0.8699
12	0.40775598	0.05667006	0.0204	0.8903
13	0.35108592	0.01597897	0.0176	0.9078
14	0.33510695	0.03813712	0.0168	0.9246
15	0.29696984	0.02068234	0.0148	0.9394
16	0.27628750	0.01692712	0.0138	0.9532
17	0.25936037	0.01730228	0.0130	0.9662
18	0.24205809	0.02020002	0.0121	0.9783
19	0.22185807	0.01013445	0.0111	0.9894

20	0.21172363	-	0.0106	1.0000

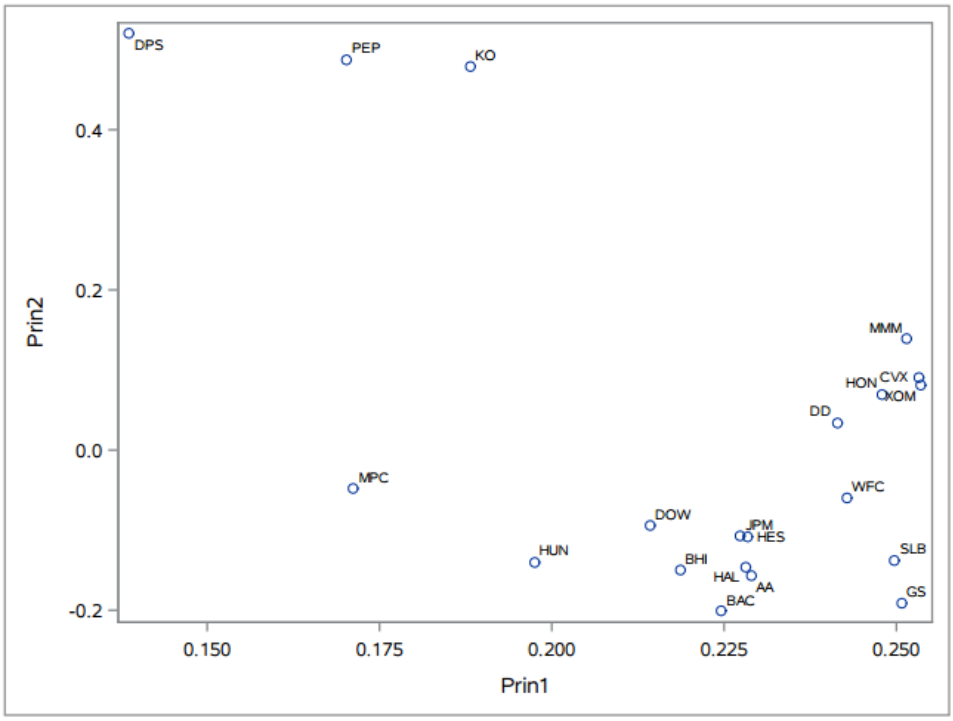
Figure 3: Scree Plot with Cumulative Variance Explained



The Scree Plot above allows us to quickly evaluate our calculated principal components and understand how each contributes to the overall variability of the data. We'll utilize the results from the table above, along with the scree plot, in order to determine which principal components we'll keep in our regression model. If our threshold criteria was 80% of the explained variability, we could select the first eight principal components.

Figure 4, presented below, allows us to observe the first two principal component vectors to uncover any hidden relationships in our data.

Figure 4: Principal Component 1 and Principal Component 2



We'll now split the original data set into a training and test group. 70% of the data will be selected at random within the training group, and we'll merge the original log-return calculations with our selected principal components. Next, we fit a regression model using all of the raw predictor variables. The diagnostic output is shown in Table 4 and Table 5.

Table 4: Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr>F
Model	20	0.01790	0.00089510	140.04	<.0001
Error	317	0.00203	0.00000639		
Corrected Total	337	0.01993			

Table 5: Model Performance

Source	
Root MSE	0.00253
Dependent Mean	0.00061635

Coeff Var	410.18453
R-Square	0.8983
Adjusted R-Square	0.8919

It appears this model has a relatively high R-Square and Adjusted R-Square value. Our F-Value is also high, and appears to be significant.

Next, we'll fit a regression model using our eight selected principal components. The diagnostic output is shown in Table 6 and Table 7, below. Our R-Square and Adjusted R-Square values practically remain unchanged, while our F-Value has increased dramatically. This leads us to believe that our PCA regression model has improved our level of predictability.

Table 6: PCA Model Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr>F
Model	8	0.01776	0.00222	337.13	<.0001
Error	329	0.00217	0.00000659		
Corrected Total	337	0.01993			

Table 7: PCA Model Performance

Source	
Root MSE	0.00257
Dependent Mean	0.00061635
Coeff Var	416.36522
R-Square	0.8913
Adjusted R-Square	0.8886

In order to test the assumption that our PCA model is a better predictor than our initial model, which utilized all of the log-return variables in our data set, we'll compute the Mean Square Error and Mean Absolute Error for each. These calculations were performed on both the training and test data, and are presented in Table 8.

Table 8: Mean Square Error and Mean Absolute Error

Observation	Training	MSE_1	MAE_1	MSE_2	MAE_2
1	0	.000009306	.002144904	.000009677	.002179249
2	1	.000005994	.001902032	.000006410	.001975239

The results above lead us to conclude that the PCA model is a better indicator of predictability. Using the PCA method allows us to work with fewer parameters, while also addressing multicollinearity, and provides us with a simpler interpretation of the model.

Code:

```
libname mydata "/scs/wtm926/" access=readonly;

data temp;
	set mydata.stock_portfolio_data;
run;
proc print data=temp(obs=10); run; quit;
proc sort data=temp; by date; run; quit;
data temp;
	set temp;
	return_AA  = log(AA/lag1(AA));
	return_BAC = log(BAC/lag1(BAC));
	return_BHI = log(BHI/lag1(BHI));
	return_CVX = log(CVX/lag1(CVX));
	return_DD  = log(DD/lag1(DD));
	return_DOW = log(DOW/lag1(DOW));
	return_DPS = log(DPS/lag1(DPS));
	return_GS  = log(GS/lag1(GS));
	return_HAL = log(HAL/lag1(HAL));
	return_HES = log(HES/lag1(HES));
	return_HON = log(HON/lag1(HON));
	return_HUN = log(HUN/lag1(HUN));
	return_JPM = log(JPM/lag1(JPM));
	return_KO  = log(KO/lag1(KO));
	return_MMM = log(MMM/lag1(MMM));
	return_MPC = log(MPC/lag1(MPC));
	return_PEP = log(PEP/lag1(PEP));
	return_SLB = log(SLB/lag1(SLB));
	return_WFC = log(WFC/lag1(WFC));
	return_XOM = log(XOM/lag1(XOM));
	*return_VV  = log(VV/lag1(VV));
	response_VV = log(VV/lag1(VV));
run;
proc print data=temp(obs=10); run; quit;

*ods trace on;
ods output PearsonCorr=portfolio_correlations;
proc corr data=temp;
*var return: with response_VV;
var return_:;
with response_VV;
run; quit;
*ods trace off;

proc print data=portfolio_correlations; run; quit;
```

```
data wide_correlations;
	set portfolio_correlations (keep=return_:);
run;

proc transpose data=wide_correlations out=long_correlations;
run; quit;

data long_correlations;
	set long_correlations;
	tkr = substr(_NAME_,8,3);
	drop _NAME_;
	rename COL1=correlation;
run;

proc print data=long_correlations; run; quit;

* Merge on sector id and make a colored bar plot;
data sector;
input tkr $ 1-3 sector $ 4-35;
datalines;
AA  Industrial - Metals
BAC Banking
BHI Oil Field Services
CVX Oil Refining
DD  Industrial - Chemical
DOW Industrial - Chemical
DPS Soft Drinks
GS  Banking
HAL Oil Field Services
HES Oil Refining
HON Manufacturing
HUN Industrial - Chemical
JPM Banking
KO  Soft Drinks
MMM Manufacturing
MPC Oil Refining
PEP Soft Drinks
SLB Oil Field Services
WFC Banking
XOM Oil Refining
VV  Market Index;
run;

proc print data=sector; run; quit;

proc sort data=sector; by tkr; run;
```

```
proc sort data=long_correlations; by tkr; run;

data long_correlations;
	merge long_correlations (in=a) sector (in=b);
	by tkr;
	if (a=1) and (b=1);
run;

proc print data=long_correlations; run; quit;


* Make Grouped Bar Plot;
* p. 48 Statistical Graphics Procedures By Example;
ods graphics on;
title 'Correlations with the Market Index';
proc sgplot data=long_correlations;
	format correlation 3.2;
	vbar tkr / response=correlation group=sector groupdisplay=cluster datalabel;
run; quit;
ods graphics off;

* Still not the correct graphic.  We want the tickers grouped and color coded by sector;
* We want ticker labels directly under the x-axis and sector labels under the ticker
	labels denoting each group.  Looks like we have an open SAS graphics project.;
ods graphics on;
title 'Correlations with the Market Index';
proc sgplot data=long_correlations;
	format correlation 3.2;
	vbar sector / response=correlation group=tkr groupdisplay=cluster datalabel;
run; quit;
ods graphics off;

* SAS can produce bar plots by sector of the mean correlation;
proc means data=long_correlations nway noprint;
class sector;
var correlation;
output out=mean_correlation mean(correlation)=mean_correlation;
run; quit;

proc print data=mean_correlation; run;

ods graphics on;
title 'Mean Correlations with the Market Index by Sector';
proc sgplot data=mean_correlation;
	format mean_correlation 3.2;
	vbar sector / response=mean_correlation datalabel;
```

```
run; quit;
ods graphics off;

ods graphics on;
title 'Mean Correlations with the Market Index by Sector - SGPLOT COMPUTES MEANS';
proc sgplot data=long_correlations;
	format correlation 3.2;
	vbar sector / response=correlation stat=mean datalabel;
run; quit;
ods graphics off;

* Reset title statement to blank;
title '';


data return_data;
	set temp (keep= return_:);
	* What happens when I put this keep statement in the set statement?;
	* Look it up in The Little SAS Book;
run;

proc print data=return_data(obs=10); run;

ods graphics on;
proc princomp data=return_data out=pca_output outstat=eigenvectors plots=scree(unpackpanel);
run; quit;
ods graphics off;

proc print data=pca_output(obs=10); run;

proc print data=eigenvectors(where=(_TYPE_='SCORE')); run;
* Display the two plots and the Eigenvalue table from the output;

* Plot the first two eigenvectors;
data pca2;
	set eigenvectors(where=(_NAME_ in ('Prin1','Prin2')));
	drop _TYPE_ ;
run;

proc print data=pca2; run;

proc transpose data=pca2 out=long_pca; run; quit;
proc print data=long_pca; run;

data long_pca;
	set long_pca;
```

```
        format tkr $3.;
        tkr = substr(_NAME_,8,3);
        drop _NAME_;
run;

proc print data=long_pca; run;

* Plot the first two principal components;
ods graphics on;
proc sgplot data=long_pca;
scatter x=Prin1 y=Prin2 / datalabel=tkr;
run; quit;
ods graphics off;

data cv_data;
        merge pca_output temp(keep=response_VV);
        * No BY statement needed here.  We are going to append a column in its current order;
        * generate a uniform(0,1) random variable with seed set to 123;
        u = uniform(123);
        if (u < 0.70) then train = 1;
        else train = 0;

        if (train=1) then train_response=response_VV;
        else train_response=.;

run;

proc print data=cv_data(obs=10); run;

* You can double check this merge by printing out the original data;
proc print data=temp(keep=response_VV obs=10); run; quit;

ods graphics on;
proc reg data=cv_data;
model train_response = return_: / vif ;
output out=model1_output predicted=Yhat ;
run; quit;
ods graphics off;

ods graphics on;
proc reg data=cv_data;
model train_response = prin1-prin8 / vif ;
output out=model2_output predicted=Yhat  ;
run; quit;
ods graphics off;
```

```
proc print data=model1_output(obs=10); run;

* Model 1;
data model1_output;
	set model1_output;
	square_error = (response_VV - Yhat)**2;
	absolute_error = abs(response_VV - Yhat);
run;

proc means data=model1_output nway noprint;
class train;
var square_error absolute_error;
output out=model1_error
	mean(square_error)=MSE_1
	mean(absolute_error)=MAE_1;
run; quit;

proc print data=model1_error; run;

* Model 2;
data model2_output;
	set model2_output;
	square_error = (response_VV - Yhat)**2;
	absolute_error = abs(response_VV - Yhat);
run;

proc means data=model2_output nway noprint;
class train;
var square_error absolute_error;
output out=model2_error
	mean(square_error)=MSE_2
	mean(absolute_error)=MAE_2;
run; quit;

proc print data=model2_error; run;

* Merge them together in one table;
data error_table;
	merge model1_error(drop= _TYPE_ _FREQ_) model2_error(drop= _TYPE_ _FREQ_);
	by train;
run;

proc print data=error_table; run;
```