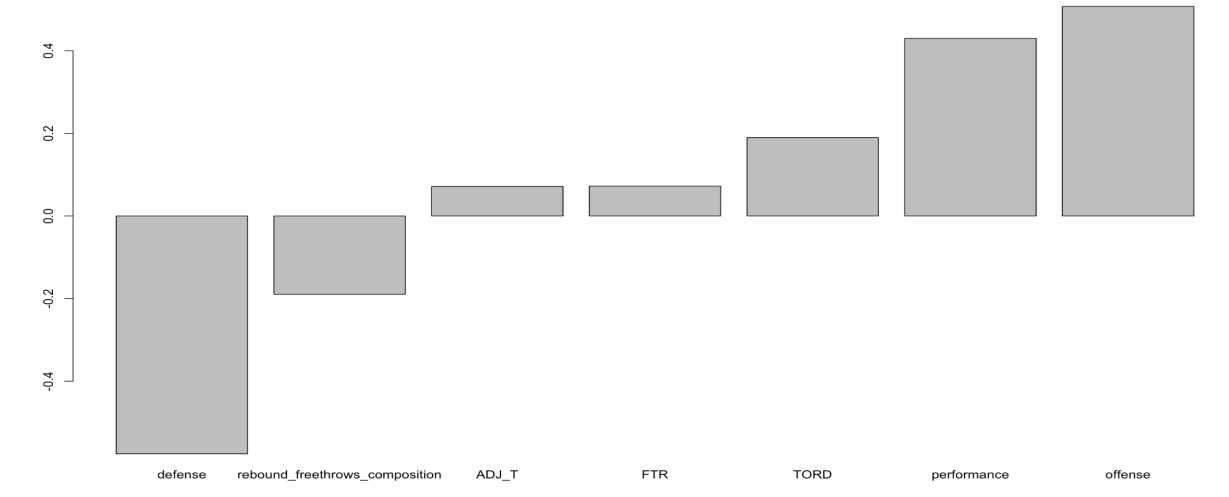
Final Report

By Yeshwanth Devabhaktuni and Geoffrey Myers

Non-Technical Summary

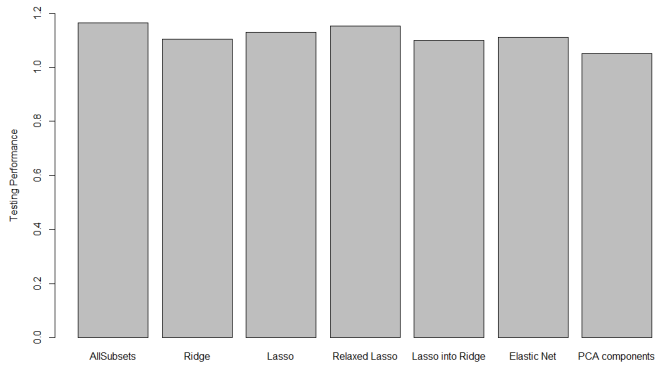
College basketball is an event in which college basketball teams compete against each other in order to be able to get into march madness which is the knockout tournament. In order to predict a team's ability to win games and get into the postseason in the process while at the same time, understanding the underlying reasons and factors that make these predictions in context is very important. For this reason, we chose to focus on predicting 2 different variables - the ability of the team to win games based on different statistics and a team's ability to venture deeper into the postseason, based on a few more statistics like the seed of the team and the conference it is based out of, considering that power 5 conferences are given more weight to get into the postseason as opposed to non power 5 conferences.

For the prediction of the regular game wins of each of the teams without the involvement of the conferences, the most important aspects of the game and season statistics were first, the defense - which showed that is the defense of a team is not good and specifically average shots (not points) allowed per 100 possessions is high or percentage of shots allowed is high, the team might not win as many games while on offense the most important is shots scored every 100 possessions. Maximizing these would improve the winnability of a game. The composition of offensive, defensive rebounds and free throws defense together does not impact the game as much, though as a combination they contribute negatively so improving offensive rebounds would help boost this combination to contribute positively. Finally, adjusted tempo, free throws and steal rate (TORD) contribute to improving a team's ability to win games but not nearly as much as the offense does so, optimizing these can help teams that already have good offenses to play lights out games as well.



For predicting the number of postseason wins, the variables that appeared in most models were adjusted offensive and defensive efficiency, wins above bubble, which are the number of wins above what is considered to be the minimum number to get in, if a team is a top 4 seed, and if the team is in a power conference. These variables showing up in various models

indicate that they have some greater importance than the other variables that did not appear in predicting the number of postseason wins.

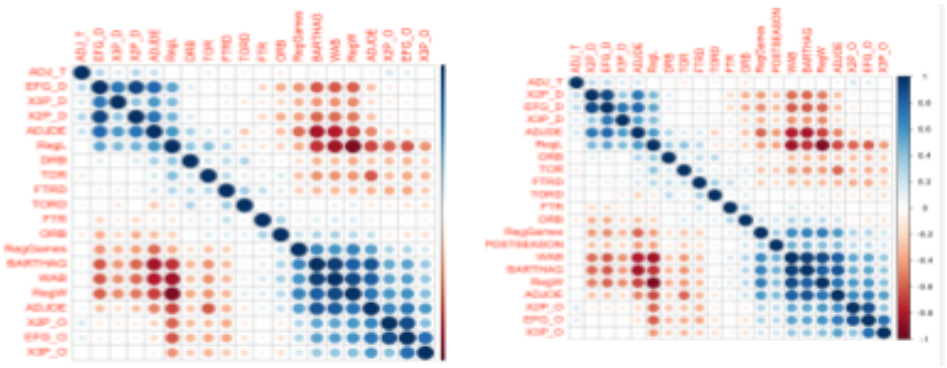


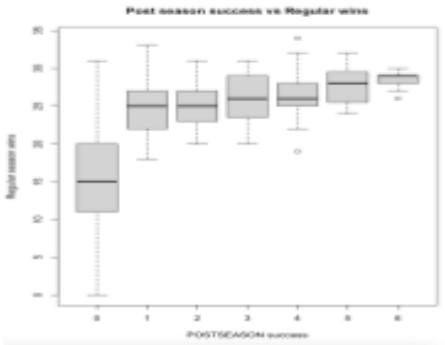
This chart above shows the testing performance of each model. The lower the bar, the better the model performed. The names of the models are the techniques used, though we can evaluate the performance without needing to understand what the techniques are doing. The model on the far right named PCA components performed the best in testing. This model is the one to use for practical applications, such as sport gambling or for filling out a NCAA tournament bracket.

Technical Summary
Exploratory analysis:

The first step in our exploration was to transform variables. The first variable to be transformed was the postseason variable. That variable is one we are interested in investigating and possibly trying to predict, as it represents how far a team makes it in the NCAA tournament. In the original dataset, it was a categorical variable, representing the round the team made it to, so to facilitate regression it was converted into the number of wins. There is a play-in part of the tournament called the first four, where technically someone could have one win the tournament and still lose in the round of 64, versus any other team will have zero wins if they lose in the round of 64. We made the decision to not count that as a tournament win as it is technically a play-in game to get into the tournament. Another variable that needed to be modified was the number of games variable along with the wins variable. Both variables represented the season long total, so it included tournament games and wins. To make them usable, they were both converted to the regular season totals by subtracting the tournament games and wins from the respective totals. I also added the variable regular season losses by subtracting wins from games.

The main attempt of the exploratory analysis was to identify whether there was multicollinearity, understanding the underlying data and its distribution. The below correlation plot shows only the continuous variables using Pearson correlation to understand multicollinearity within the dataset and the one below that uses Spearman correlation to understand even the correlation between ordinal variables and ordinal as well as continuous variables.





In both visualizations, it is very clear that the defensive statistics are clustered together and the offensive ones are clustered together, but the conclusion of the plots above are that regular wins and postseason are highly correlated to one another while each is positively correlated with offensive stats and negatively with defensive stats. Because of this, PCA can be used for dimensionality reduction and factor analysis for interpreting the variables. Once this is done, we can run ordinary least squares, ridge and lasso regressions. The boxplot shows that regular game wins and postseason success are highly correlated so our analysis will first focus on understanding the success of teams during the regular season by predicting regular game wins, followed by understanding the variables that help teams succeed in the postseason and tie both these predictions together to understand whether variables contributing to both predictions are very similar or different.

```
Call:
lm(formula = POSTSEASON ~ . - X1 - CONF - TEAM - RegGames - YEAR,
    data = cbbupdated)

Residuals:
    Min      1Q  Median      3Q     Max
-2.7792 -0.4929 -0.0952  0.3397  4.2031

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.044684   2.906511   2.424  0.01576 *
RegW         0.030019   0.034404   0.873  0.38338
RegL        -0.050394   0.040293  -1.251  0.21171
ADJOE        0.271350   0.036633   7.407 6.61e-13 ***
ADJDE       -0.324162   0.048609  -6.669 7.74e-11 ***
BARTHAG     -9.273021   1.784758  -5.196 3.12e-07 ***
EFG_O       -0.043790   0.205416  -0.213  0.83129
EFG_D       -0.295650   0.229680  -1.287  0.19869
TOR          0.060557   0.043719   1.385  0.16671
TORD        -0.070630   0.045569  -1.550  0.12186
ORB         -0.027105   0.020395  -1.329  0.18454
DRB          0.085140   0.027086   3.143  0.00178 **
FTR         -0.014706   0.009887  -1.487  0.13762
FTRD        -0.001778   0.010339  -0.172  0.86353
`2P_O`       0.005153   0.128866   0.040  0.96812
`2P_D`       0.270281   0.144870   1.866  0.06275 .
`3P_O`      -0.042760   0.105434  -0.406  0.68526
`3P_D`       0.220108   0.116912   1.883  0.06040 .
ADJ_T       -0.009101   0.015227  -0.598  0.55036
WAB         -0.056869   0.044904  -1.266  0.20602
SEED2       -0.268039   0.263676  -1.017  0.30993
SEED3       -0.130516   0.290027  -0.450  0.65292
SEED4       -0.148974   0.307392  -0.485  0.62817
SEED5       -0.668381   0.332856  -2.008  0.04525 *
SEED6       -0.611251   0.347492  -1.759  0.07926 .
SEED7       -0.021185   0.365654  -0.058  0.95382
SEED8       -0.313123   0.381083  -0.822  0.41171
SEED9       -0.239677   0.392950  -0.610  0.54221
SEED10      -0.513779   0.393599  -1.305  0.19246
SEED11      -0.219629   0.398068  -0.552  0.58141
SEED12      -0.046296   0.437560  -0.106  0.91579
SEED13      -0.062583   0.467871  -0.134  0.89365
SEED14      -0.020386   0.494737  -0.041  0.96715
SEED15      -0.046452   0.540313  -0.086  0.93153
SEED16      -0.043815   0.623384  -0.070  0.94400
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9251 on 441 degrees of freedom
  (1979 observations deleted due to missingness)
Multiple R-squared:  0.5455,    Adjusted R-squared:  0.5105
F-statistic: 15.57 on 34 and 441 DF,  p-value: < 2.2e-16
```

```
> vif(lm1)
             GVIF  Df GVIF^(1/(2*Df))
RegW     8.136027   1        2.852372
RegL    10.333010   1        3.214500
ADJOE   30.106511   1        5.486940
ADJDE   38.311557   1        6.189633
BARTHAG 51.035638   1        7.143923
EFG_O  173.309680   1       13.164713
EFG_D  173.115811   1       13.157348
TOR      3.621893   1        1.903127
TORD     6.603138   1        2.569657
ORB      4.095664   1        2.023775
DRB      3.388305   1        1.840735
FTR      1.572315   1        1.253920
FTRD     2.050397   1        1.431921
`2P_O`  80.760039   1        8.986659
`2P_D` 103.354885   1       10.166360
`3P_O`  39.760074   1        6.305559
`3P_D`  32.194290   1        5.674001
ADJ_T    1.331733   1        1.154007
WAB     28.056032   1        5.296795
SEED   122.083180  15        1.173695
```

In further exploration, we looked at an initial linear regression model to see how it performed. Examining the model, it passes the f-test, meaning that at least one of the predictors is different from zero. The r^2 of the model is .55, or 55% of the variability in post season wins is covered by the model. Only adjusted offensive efficiency, adjusted defensive efficiency, power rating, offensive rebound rate allowed, 2-point defense, 3-point defense, and two of the levels of the dummy variable for seed had a significant p-value, though there might be multicollinearity confounding the p-values. There is a significant amount of multicollinearity in the model, as many variables have vifs over 10. With that, one of our approaches will focus on model building, which will look at methods such as regularized regression and PCA to handle that multicollinearity.

Approach 1

In the first approach regular game wins during the season is the variable of interest and use only the game based statistics to help predict the wins. In this approach we will be leaving out the use of the ordinal variables called POSTSEASON - since these are regular season game wins and have nothing to do with the post season and SEED - since seed has nothing to do with a team's ability to win a game. YEAR, CONF and TEAM were also removed since they were categorical variables as they had no information to understand game wins.The next part was to apply the corr test on the dataset with a 95% confidence level to determine the most correlated variables and which ones to leave out of the factor analysis. In the below image, it can be seen that TORD, FTR, ADJ_T can be removed since they are correlated with very few other variables

```
> colSums(Mtest) - 1
      RegW       RegL      ADJOE      ADJDE    BARTHAG      EFG_O      EFG_D        TOR       TORD        ORB        DRB
        16         16         15         16         16         15         16         14          0         11         15
       FTR       FTRD       X2P_O      X2P_D      X3P_O      X3P_D      ADJ_T        WAB POSTSEASON
         4         12         15         16         12         14          3         16         16
```

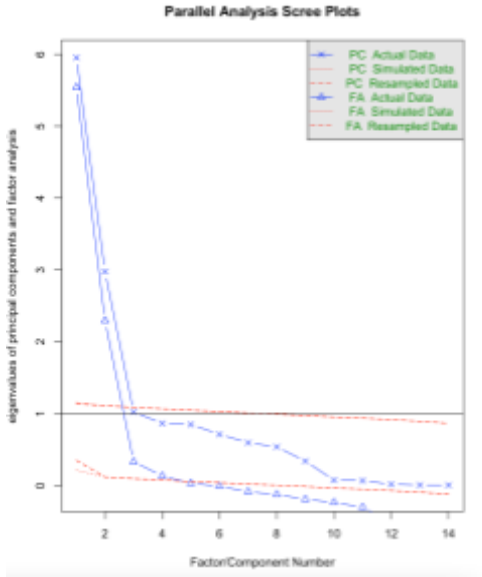
Once the above feature selection was done I split the dataset into a test train split with 80% of the samples being in the train group and 20% being in the test group. Using the 80% and 20% split, I applied an initial regression fit to the data and got an RMSE value of 1.792147 on the training set and 1.675166 on the test set, but the vif scores were high as below.

```
> car::vif(fit)
      TORD      ADJ_T        FTR      ADJOE      ADJDE      EFG_O      EFG_D        TOR        ORB        DRB       FTRD
  3.114119   1.198091   1.392955  29.200209  23.224181 144.610010 223.248933   3.267945   3.053923   2.248467   1.701040
     X2P_O      X2P_D      X3P_O      X3P_D    BARTHAG        WAB
 67.745071 121.637375  31.258051  41.931359  37.949484  12.779472
```

Once this was done, I applied the prcomp function for PCA and the scree plot to determine the appropriate number of factors that are needed to interpret the data. Next, the principal components below show that based on the scree plot and cumulative variance, most of the data is being captured by 2 components, as they account for almost 64% of the variance. Though the component visuals show that they initially knee out around the 4th component but it also does seem like around the 9th and 10th components there is a significant change between the bars which may also suggest important information being lost. According to the cumulative variance next to the scree plot below, it can be seen again that PC1 and PC2 account for most of the variance while until the 7th and 8th components,there isn't much data being captured in

each component and until the 8th component, more than 90% is not captured.

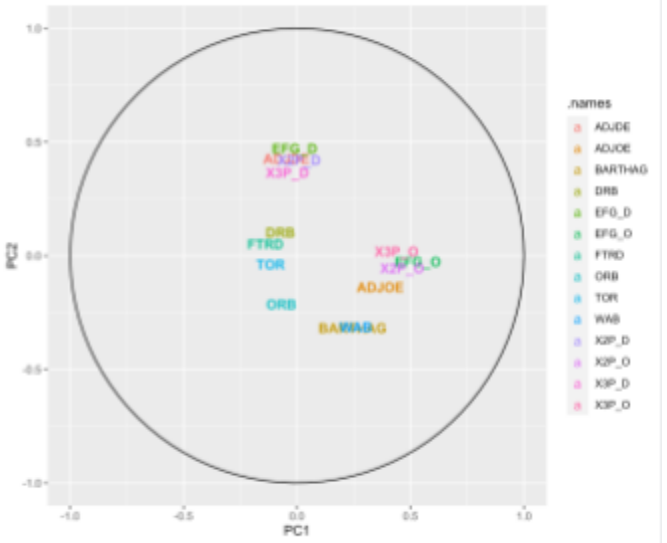
In the parallel analysis plot below again, it can be seen that there are 4 factors that may be most relevant for analysis but further experimentation is required since as mentioned above and based on the scree plot and cumulative proportion,to determine the most appropriate number of factors. After the parallel analysis, I used the principal function for factor analysis starting initially with 3 components. All techniques used the varimax rotation and the loadings were interpreted. 3 components did not separate the loadings well enough, but 4 did better.



Now based on the 4 factors on the image below, it can be seen that the 4 components capturef most variance showing that the defensive statistics are clustered together, the offensive statistics are clustered together, the turnover percentage, free throw defense and offensive rebound rate are clustered closer together, while power rating (barthag) and wins above bubble are closer - which also makes sense because they are associated with team performance up until that point in the regular season. In the loadings of the factors themselves, it is clear that RC2 has effective field goal defense percentage as the major correlated variable followed by adjusted defensive efficiency, which is the points allowed per 100 possessions, then by x2p_d which is the 2

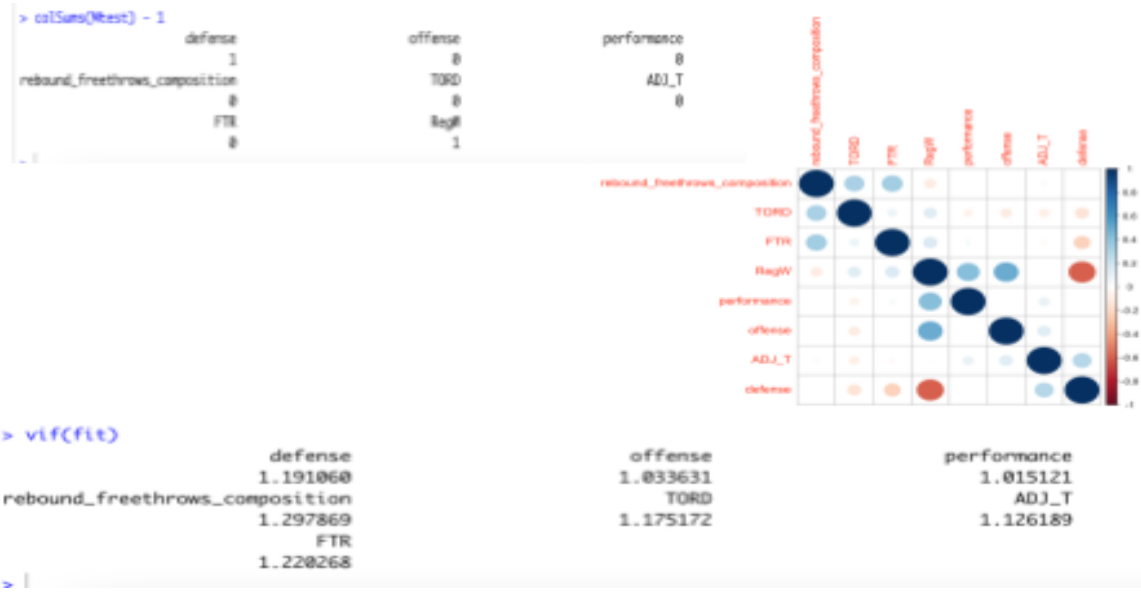
point shots allowed, followed by 3 pointers allowed which shows all the defensive statistics belong together but there is also WAB and BARTHAG which are performance indicators that contribute negatively which gives an idea that the defensive statistics contribute negatively to overall performance, while offensive statistics contribute positively, though not as much as defensive statistics, we will name rc2 as defense and rc1 as offense. The component RC4 seems to just have turnover rate, WAB and BARTHAG along with ADJOE which shows that teams with lower turnover rates are more relevant to this component, makes sense considering that giving up the ball would be bad for the team, we will call this component performance since it measures performance up until that point in the season. The final component shows the composition of rebounds and free throw defense which are correlated to the factor RC3, we can call this rebounds and free throws allowed. The p-value of the hypothesis test shows less than 0 according to the output of the principal function with 4 factors along with root mean square of the residuals value of 0.07 which shows that 4 factors are not sufficient to distinctly explain the variance, but in this case they enough to interpret the variance well. The corr plot below shows the relationships after the factor analysis followed by a corr test and getting vif values of the regression using the 4 factors along with factors TORD, ADJ_T and FTR, hence making the total number of variables **8 variables** along with the dependent variable called RegW.

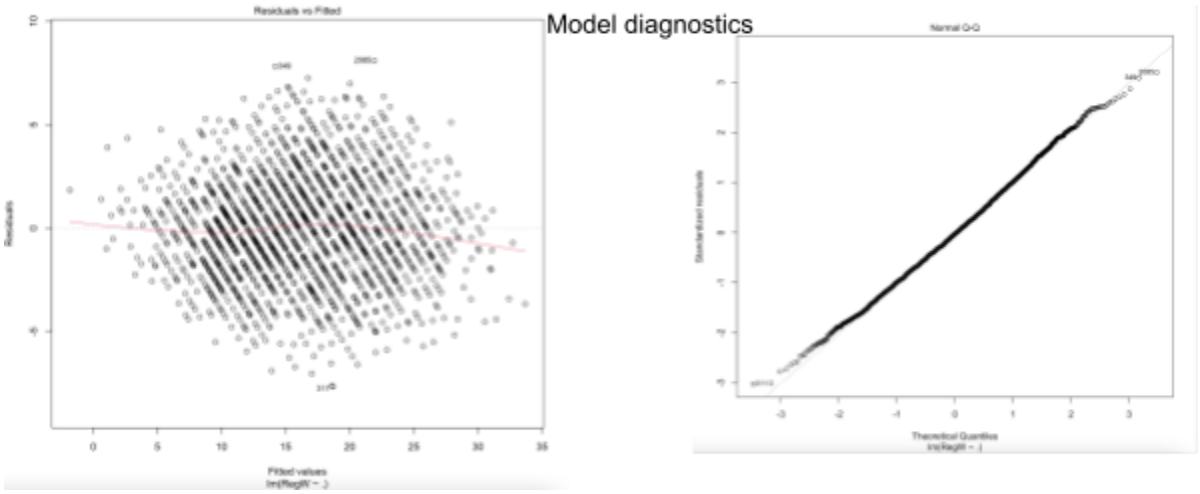
```
> print(p2$loadings, cutoff=0.4, sort=T)

Loadings:
        RC2    RC1    RC4    RC3
ADJDE    0.874
BARTHAG -0.650  0.437  0.567
EFG_D    0.966
X2P_D    0.859
X3P_D    0.749
WAB     -0.641  0.464  0.546
EFG_O           0.953
X2P_O           0.829
X3P_O           0.785
ADJOE           0.656  0.671
TOR                   -0.783
ORB     -0.434                0.642
DRB                           0.585
FTRD                          0.653

                 RC2   RC1   RC4   RC3
SS loadings    4.175 3.198 2.041 1.400
Proportion Var 0.298 0.228 0.146 0.100
Cumulative Var 0.298 0.527 0.672 0.772
```



After running the ordinary least squares regression after scaling the independent variables / factors, the vif scores were also reasonable and below 2 for all, so I ran the regression and got a r-square value of 0.8378 and a RMSE value of 2.529457 on the training set and a RMSE value of 2.68738 on the test set (the test set is transformed to 8 variables using the factor model), which is reasonable but may be improved with lasso regression as the most relevant variable would be chosen with it. So after running lasso regression on the factors with the left out variables from factor analysis, we get an RMSE value of 2.560426 on training data and 2.720346 for testing which shows that it is comparable to the ordinary least squares as well. Upon doing the same with the individual initial variables without the factors with lasso regression, I got an RMSE value of 1.852109 for training and 1.917816 testing RMSE values using 10 fold cross validation for both. The image below shows model diagnostics for the ols method described above and the residuals

seem to be randomly scattered and the q-q plot is normal which means that the model may be a good fit on the combination of factors and variables





In the image below (independent variables scaled), it can be seen that running lasso regression on variables instead of factors and using lambda.1se gets the most important variables for the model. In the model below,2 point shots taken (X2P_O), 2 points allowed (X2P_D), 3 point shots (X3P_O) and power rating (BARTHAG) are deemed irrelevant, possibly because thay have or contain info explained by other variables.

Conclusion for approach 1

The statistics containing RMSE values being higher for factors as opposed to when regular variables are used shows that some information is being lost, though most is being

Lasso full all variables vs factors using lambda.1se from cv.glmnet

```
18 x 1 sparse Matrix of class "dgCMatrix"
                      1
(Intercept) 16.2540733
ADJOE       -1.3955962
ADJDE        2.0751448
BARTHAG      .
EFG_O        1.6827932
EFG_D       -1.8137383
TOR         -0.9456113
TORD         1.4922987
ORB          0.7717461
DRB         -1.2475055
FTR          0.1879881
FTRD        -0.5613618
X2P_O        .
X2P_D        .
X3P_O        .
X3P_D       -0.2950165
ADJ_T        0.2519260
WAB          4.8786674
```

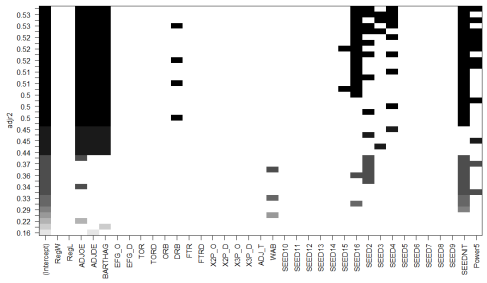
```
8 x 1 sparse Matrix of class "dgCMatrix"
                                      1
(Intercept)                  16.2540733
defense                      -3.5182277
offense                       3.0517317
performance                   2.5720046
rebound_freethrows_composition -0.9233710
TORD                          0.9658580
ADJ_T                         0.2992748
FTR                           0.2660522
>
```

captured by the 8 variables with the 4 factors, as the difference in the ols and lasso RMSE methods for both is very low. Though when lasso is run on the initial variables, it shows that some defensive and offensive scoring variables can be removed with almost negligible loss to the RMSE value with defensive and offensive statistics still contributing the most towards the prediction of regular game wins, though wins above bubble seems to be highly relevant in these predictions as well.

The practical significance is that defense is the most important aspect of the game followed by the offense, with 2 point defense being the most important. Within the offense, based on the factor analysis, the most important aspects are field goals shot and made with 2 pointers being slightly more important than 3. Using this information, teams can understand where they are lacking as compared to most teams and focus efforts on improving in specific areas within defense and offense or optimize their free throws or rebound rates as they are a little less relevant than overall offense and defense

Approach 2

For approach two, we explored building predictive models for the variable postseason, a variable that represents the number of wins in the NCAA basketball tournament. Unlike the first approach, we will leave in the SEED variable as it pertains to the tournament. The conference variable was also used, and was transformed into the variable Power5, which represents if a team is in a power conference (ACC, Big 10, Big 12, PAC 12, and SEC). This was done as there is a large gap in historical performance in the power conferences and the mid-major conferences and the difference between the individual conferences in those groups is very minor.



```
> vif(bestAllSub)
                GVIF Df GVIF^(1/(2*Df))
ADJOE    24.853088  1        4.985287
ADJDE    20.603023  1        4.539055
BARTHAG 57.641417  1        7.592194
SEED      3.262244 16        1.037642
Power5    1.623192  1        1.274045
DRB       1.230482  1        1.109271
```

Model: POSTSEASON ~ ADJOE + ADJDE + BARTHAG + SEED + Power5 + DRB

The first model building technique that was tried was all subsets selection. Using the graph shown above, the model above was created. Though, this model had significant multicollinearity, which was resolved by removing barthag, which only changed the R^2 by .1. Though, other regression methods explored later will better resolve this multicollinearity. In terms of performance, the model had an R^2 of .59 and an RMSE ratio (Test/Train) of 1.6.

Lasso	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 + SEED.NIT + Power5
Lasso into Ridge	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 + SEED.NIT + Power5
Elastic Net	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 + SEED.7 + SEED.16 SEED.NIT + Power5
Relaxed Lasso	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 SEED.NIT + Power 5

The second model building technique that was tried was regularized regression with ridge, lasso, and elastic net. Since these techniques need only numeric variables, the SEED variable was transformed into dummies of the different factor levels. The cross-validated versions of all of these techniques were used. Ridge regression was the first technique that was tried, which led to a model with all of the predictors as ridge does not perform feature selection. That model had an R^2 of .54 and had an RMSE ratio of 1.10. The next technique was lasso regression. One of the main uses of lasso regression is feature selection, which is helpful here as we have large feature space. Lasso ended up zeroing out all up 8 features. It had an R^2 of .57 and had an RMSE ratio of 1.13. Ridge regression was also tried with leftover variables from Lasso, and that model had an R^2 of .57 with an RMSE ratio of 1.10. Relaxed Lasso regression was tried next, which runs lasso twice in order to relax the regularization on the betas. The model had an R^2 of .58 and an RMSE ratio of 1.15. Elastic net was also tried using the alphas of .25, .5, and . 75, with .25 ultimately performing the best of three. That model had an R^2 of .55 and an RMSE ratio of 1.11. Of these models, Relaxed lasso probably is the best, as it had the best R^2 and had a similar RMSE ratio as the rest, while also having a small number of variables. Those variables were probably the most important in the dataset, as they were not zeroed out.

Model: POSTSEASON ~ PC1 + PC3 + PC4 + PC7 + PC9 + PC13 + PC16 + PC18 + PC20 + PC21 + PC23 + PC25 + PC26 + PC28 + PC33

```
> round(pca1$rotation[,25], 2)[which(abs(pca1$rotation[,25]) > .25)]
    FTRD    X3P_O    X3P_D   SEED.1 SEED.NIT
    0.37     0.31    -0.25     0.27    -0.28
> round(pca1$rotation[,33], 2)[which(abs(pca1$rotation[,33]) > .25)]
RegL  WAB
0.39 0.81
> round(pca1$rotation[,20], 2)[which(abs(pca1$rotation[,20]) > .25)]
 SEED.1 SEED.10 SEED.14 SEED.15
   0.41   -0.28    0.29    0.29
```

Next regression with PCA was tried due to PCA inert dimensionality reduction along with it being a good way to handle multicollinearity. PCA was performed with scaled data due to the varying sizes of the variables. A linear model was then built using all of the PCs. After an initial run, the insignificant PCs were then removed, creating the model that is shown above. Looking at the output above which shows the three most important PCs according to Lm.beta, PC 25 seems to be offensive 3 pt performance and free throws allowed versus 3 point defense. PC 33 is just regular season losses and wins above the bubble, which that PC could just represent regular season performance. PC 20 seems to capture some differences in seeding. While those PCs are the most important, there are still many PCs which make it nearly impossible to understand what is going on in the model, though a lot PCs are needed to capture a lot of the smaller differences, which in this data set represent the upsets in the tournament. The upsets are the most interesting points in the dataset, so if a lesser number of PCs were used, that information would be lost. The model had an R^2 of .56 and had an RMSE ratio 1.05.

All Results

Type	r.squared	RMSE.Ratio
AllSubsets	0.5900000	1.163416
Ridge	0.5428017	1.103220
Lasso	0.5703210	1.130053
Relaxed Lasso	0.5795024	1.151878
Lasso into Ridge	0.5703210	1.098859
Elastic Net	0.5547114	1.112250
PCA components	0.5600000	1.050042

Conclusion for Approach 2

Overall, there were two models that seemed to perform better than the rest. The first was the relaxed lasso model. While it did perform at a similar level to the other regularized regression methods, it did have the advantage of being the most explainable model. The other model that performed very well was the model using PCA. This model had an R^2 comparable to all of the other models but had a better RMSE ratio. Though, this model struggles with interpretability as it has too many PCs to truly understand which individual variables are making an impact on the model. Some variables that appeared across many of the models were adjusted offensive and defensive efficiency, wins above bubble, Seed 1, 2, 3, and 4 and if the team was in a power 5 conference. Some of these variables, such as adjusted offensive and defensive efficiency, appeared as useful variables in both of the approaches.This highlights that teams that are strong in these metrics have both good regular season and postseason success and these metrics could be used to measure the quality of a team.

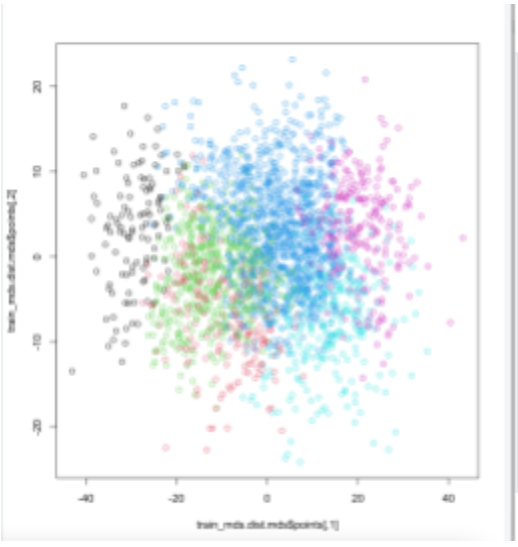
In terms of a practical application for this analysis, the two models have very different use cases. Since the PCA model does not have a high level of interpretability, it would be most useful for sports betting. It would be useful for that as one only care about figuring out how far a team went and betting on that. While the relaxed lasso model does not have as directly of an

application, looking at the weighted coefficients could be useful for coaching to see which variables matter the most. Looking at the weighted coefficients could also be useful in tandem with looking at variables that appeared in most of the models. That could be useful for trying to predict the outcome of an individual game by looking at which team performs better in those variables.

Individual Reports

Yeshwanth Devabhaktuni

I was responsible for the first approach completely, I used the data to critically analyze which variables would best explain the ability to perform regression on them to predict game wins during the regular season of the ncaa basketball tournament. I came up with a couple of ideas on how to split up the project with me working on the regular games as the variable of interest and my partner using ordinal variables for his analysis which is significantly more focussed on regression. Initially I came up with the Pearson correlation visualization in the milestone for exploratory analysis as well as understanding the distribution of the data as a whole to get an idea of whether transformations would be necessary or not and made the boxplot that is in the exploratory analysis to understand the relationship between postseason and regular games. Once this was done, I decided to explore both factor analysis as well as clustering for the dataset. In factor analysis, regular game wins was the variable of interest while in cluster analysis, I wanted to see whether clustering happened around either specific conferences or around the POSTSEASON variable. In the first approach, I experimentally generated visualizations, including the corr test results above, the scree plot to understand the components, the PCA plot on the factor analysis to understand how the first couple of factors would interact with each other and how much variance they would account for. I used these factors in the regression and tried to understand the best variables. I also performed lasso regression on the factors and on the initial variables. In cluster analysis I ran the multidimensional scaling which resulted in a stress factor of 0.1234496 which was not ideal since it is high and clean clustering may or may not be possible. I ran a hierarchical clustering model on the data anyway and split the data up into 7 categories to see whether they would separate into each of the categories and describe the postseason variable as a result but got the matrix shown below, which showed me that clustering may not be the best technique to move forward with and hence I moved further into factor analysis and regression. Once the clustering experimental results were generated, it was clear that my teammate could work on identifying the best features that contributed most to the team's success in the postseason.



```
> table(cbb[,24],clusters)
   clusters
       1    2    3    4    5    6
  0   17  166  448 1054  299  247
  1   27   23   54    8    0    0
  2   29    9   17    1    0    0
  3   20    3    5    0    0    0
  4    8    1    4    1    0    0
  5    6    1    0    0    0    0
  6    6    0    1    0    0    0
>
```

Finally, we got together to identify the best variables contributing to both and whether both have the same contributing variables and / or components with the same variables contributing the most to them.
Administratively, I was responsive to my teammate's emails and resolved issues as and when they came up with my teammate and had regular meetings, this was a joint effort.

What I learnt

From a knowledge standpoint, it became clear to me that data might not be sufficiently separated among factors sometimes but it may still be interpreted clearly which means the p-value for the chi square test though relevant may not give us all the necessary or useful information about interpretability. When regression is run on factors that are formed during factor analysis, there is a chance that some important information may be lost even when a majority of the data is captured but it can be a good tradeoff if results are more interpretable and the final model does not have the problem of multicollinearity, in which case the predictions may not be reliable as it undermines the statistical significance of the final model. I also learnt that using too many classes with hierarchical cluster analysis may not be the best way to understand the data. I also was able to learn that scaling can help get the right interpretable beta values for the model and that scaling the variable of interest, though appropriate, may not be the best idea since it would not gives us the difference in game wins being predicted in this case. It was also interesting to use the training set to get the factors and use the factors model to predict / compress the test data and use the test data for predictions. With regards to this specific dataset, I learnt that sports statistics always have a set of positive variables and negative variables that may contribute equally and the main aspect to look for is a pattern within these sets of variables and reduce their multicollinearity to get interpretable results.

Geoffrey Myers

Firstly, I handled most of the initial data transformations along with exploring the initial regression results. Those transformations included creating the regular season wins and loses variables along with transforming the postseason variable from a categorical variable to a numeric one. After that initial exploration, I was responsible for all of approach 2. Approach 2 centered around model building around trying to predict the number of postseason wins. This approach required more variable transformation, as the conference variable needed to be transformed in order to be usable in a regression method. That variable, which previously was a categorical variable representing the conference that a team belonged to, was transformed into a dummy variable. This dummy variable was called Power5, meaning that if this variable was a 1, then the team was in a power 5 conference (ACC, Big Ten, Big 12, PAC 12, and SEC), and if it was a 0, then it was considered to be a mid-major conference. This decision was made as the gap between mid-major and power 5 conferences is larger, whereas the difference between individual conferences in those tiers is rather small.

Various model building techniques were tried. The first technique that was used to get a baseline model was all subsets selection. Regularized regression methods, including Ridge, Lasso (both normal and relaxed) and elastic net, were tried next. Regularized regression was used due to some overfitting and multicollinearity in the all subsets model. That multicollinearity also led to the next approach, which was using PCA for regression.

Type	r.squared	RMSE.Ratio
AllSubsets	0.5900000	1.163416
Ridge	0.5428017	1.103220
Lasso	0.5703210	1.130053
Relaxed Lasso	0.5795024	1.151878
Lasso into Ridge	0.5703210	1.098859
Elastic Net	0.5547114	1.112250
PCA components	0.5600000	1.050042

All Subsets	POSTSEASON ~ ADJOE + ADJDE + BARTHAG + SEED + Power5 + DRB
Lasso	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 + SEED.NIT + Power5
Lasso into Ridge	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 + SEED.NIT + Power5
Elastic Net	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 + SEED.7 + SEED.16 SEED.NIT + Power5
Relaxed Lasso	POSTSEASON ~ ADJOE + ADJDE + WAB + SEED.1 + SEED.2 + SEED.3 + SEED.4 SEED.NIT + Power 5
PCA	POSTSEASON ~ PC1 + PC3 + PC4 + PC7 + PC9 + PC13 + PC16 + PC18 + PC20 + PC21 + PC23 + PC25 + PC26 + PC28 + PC33

Looking at the results, there were two models that seemed to perform better than the rest. The first was the relaxed lasso model. While all of the regularized regression models performed at similar levels, it did have the advantage of being the most explainable model. The other model that performed very well was the model using PCA. This model had an R^2 comparable to all of the other models but had a better RMSE ratio. Though, this model struggles with interpretability as it has too many PCs to truly understand which individual variables are making an impact on the model. Some variables that appeared across many of the models were adjusted offensive and defensive efficiency, wins above bubble, Seed 1, 2, 3, and 4 and if the team was in a power 5 conference.

What I learnt

Through this process of performing this analysis, I learned a lot about analysis and this data. The main takeaway is know your data. For this data, I think an outlier analysis might have been appropriate and something to explore, as the upsets in the tournament are probably the most interesting data points, at least from a domain perspective. With that, I think I learned that the data should really drive what kind of analysis you should perform. While there were some interesting results, I felt that there could have been more interesting routes of inquiry that were better suited for the data, such as treating postseason wins as a categorical variable and performing correspondence analysis to see which features were closely associated with a certain number of wins.

Appendix

Variable definitions

RegW: Number of regular season wins

RegL: Number of regular season loses

ADJOE: Adjusted Offensive Efficiency (An estimate of the offensive efficiency (points scored per 100 possessions) a team would have against the average Division I defense)

ADJDE: Adjusted Defensive Efficiency (An estimate of the defensive efficiency (points allowed per 100 possessions) a team would have against the average Division I offense)

BARTHAG: Power Rating (Chance of beating an average Division I team)

EFG_O: Effective Field Goal Percentage Shot

EFG_D: Effective Field Goal Percentage Allowed

TOR: Turnover Percentage Allowed (Turnover Rate)

TORD: Turnover Percentage Committed (Steal Rate)

ORB: Offensive Rebound Rate

DRB: Offensive Rebound Rate Allowed

FTR : Free Throw Rate (How often the given team shoots Free Throws)

FTRD: Free Throw Rate Allowed

2P_O: Two-Point Shooting Percentage

2P_D: Two-Point Shooting Percentage Allowed

3P_O: Three-Point Shooting Percentage

3P_D: Three-Point Shooting Percentage Allowed

ADJ_T: Adjusted Tempo (An estimate of the tempo (possessions per 40 minutes) a team would have against the team that wants to play at an average Division I tempo)

WAB: Wins Above Bubble (The bubble refers to the cut off between making the NCAA March Madness Tournament and not making it)

Postseason: The number of games a team wins the NCAA Tournament

Seed: Seed in NCAA Tournament with NIT representing not making the tournament

Power5: Whether a team is in a Power 5 conference (Acc, Big10, Big12, PAC12 and SEC)

Code for approach 1

```
library(DescTools)

library(Hmisc) #Describe Function

library(psych) #Multiple Functions for Statistics and Multivariate Analysis

library(GGally) #ggpairs Function

library(ggplot2) #ggplot2 Functions


library(corrplot) #Plot Correlations

library(REdaS) #Bartlett's Test of Sphericity

library(psych) #PCA/FA functions

library(factoextra) #PCA Visualizations

library("FactoMineR") #PCA functions

library(ade4) #PCA Visualizations

library(rcompanion)

library(Metrics)

setwd("~/Downloads")

cbb <- read.csv("cbbUpdated.csv");


nums <- unlist(lapply(cbb, is.numeric))


cbb_numeric_init <- cbb[ , nums]
```

```
cbb_numeric = subset(cbb_numeric_init, select = -c(SEED,X,YEAR,POSTSEASON) )

cbb_numeric_final = subset(cbb_numeric_init, select = -c(X,YEAR,SEED) )

cbb_ordinal = subset(cbb_numeric_init, select = -c(X,YEAR,SEED))

M <- cor(cbb_numeric)
corrplot(M, method = "circle", order="AOE")

M_ord <- cor(cbb_ordinal, method="spearman")
corrplot(M_ord, method = "circle", order="AOE")

boxplot(RegW ~ POSTSEASON, data=cbb, xlab = "POSTSEASON success",
        ylab = "Regular season wins", main = "Post season success vs Regular wins")

hist(cbb_numeric[])

cbb_numeric_trans = subset(cbb_numeric_final, select = -c(RegGames))
head(cbb_numeric_trans)

cor_cbb_numeric_trans = cor(cbb_numeric_trans)

#applying corrtest with a 95% confidence interval here#
res1 <- cor.mtest(cor_cbb_numeric_trans, conf.level = .95)
```

```
corrplot(cor_cbb_numeric_trans, p.mat = res1$p, sig.level = .05)


McorrTest <- corr.test(cor_cbb_numeric_trans, adjust="none")
round(McorrTest$p, 2)
Mtest <- ifelse(McorrTest$p < 0.05,T,F)
Mtest


colSums(Mtest) - 1


#splitting test and train here#
## 80% of the sample size
smp_size <- floor(0.80 * nrow(cbb_numeric_trans))


## set the seed to make your partition reproducible
set.seed(123)
train_ind <- sample(seq_len(nrow(cbb_numeric_trans)), size = smp_size)


train <- cbb_numeric_trans[train_ind, ]
test <- cbb_numeric_trans[-train_ind, ]


#Removing TORD and all ordinal values#
cbb_numeric_train <- subset(train, select = -c(RegW,RegL,POSTSEASON,TORD,ADJ_T,FTR))
cbb_numeric_test <- subset(test, select = -c(RegW,RegL,POSTSEASON,TORD,ADJ_T,FTR))
```

```
fit = lm(RegW ~
TORD+ADJ_T+FTR+ADJOE+ADJDE+EFG_O+EFG_D+TOR+ORB+DRB+FTRD+X2P_O+X2P
_D+X3P_O+X3P_D+BARTHAG+WAB, data=train)

summary(fit)

sqrt(mean(fit$residuals^2))



pred = predict(fit,test)

rmse(test$RegW,pred)



car::vif(fit)



#do parallel analysis here as well#

cbb_cor = cor(cbb_numeric_train)

p <- prcomp(cbb_numeric_train, scale=T)

summary(p)

plot(p)

abline(1,0, col="red")



fa.parallel(cbb_numeric_train)



p$sdev^2



source("PCA_plot.R")



KMO(cbb_numeric_train)
```

```
p1 = principal(cbb_numeric_train,  rotate="varimax", nfactors=3);
summary(p1)
print(p1)
print(p1$loadings, cutoff=0.4,sort=T)
PCA_Plot_Psyc(p1)

p2 = principal(cbb_numeric_train,  rotate="varimax", nfactors=4);
summary(p2)
print(p2$loadings, cutoff=0.4, sort=T)
PCA_Plot_Psyc(p2)

p4 = principal(cbb_numeric_train,  rotate="varimax", nfactors=6);
summary(p4)
print(p4$loadings, cutoff=0.4, sort=T)
PCA_Plot_Psyc(p4)

p3 = principal(cbb_numeric_train,  rotate="varimax", nfactors=9);
summary(p3)
print(p3$loadings, cutoff=0.4)
PCA_Plot_Psyc(p3)

print(p3$communality,2)
print(p3$uniquenesses,2)
```

```
#Factors for regression# #Explain using domain knowledge here as well#

#I found that for interpretability 4 factors and for p-value significance, 9 factors need to be used#

p2 = principal(cbb_numeric_train, rotate="varimax", nfactors=4);

summary(p2)

print(p2$loadings, cutoff=0.4, sort=T)

print(p2)


print(p2$communality,2)

print(p2$uniquenesses,2)


PCA_Plot_Psyc(p2)


p3 = principal(cbb_numeric_train,  rotate="varimax", nfactors=9);

summary(p3)

print(p3$loadings, cutoff=0.4)

PCA_Plot_Psyc(p3)


print(p3$communality,2)

print(p3$uniquenesses,2)


#goodness of fit chi square testing here#

factanal_fit = factanal(cbb_numeric_train, 4, n.obs=1964, scores="regression")

print(factanal_fit$loadings, cutoff=0.4, sort=T)
```

```
print(factanal_fit)

factanal_fit$correlation

factanal_fit$scores


scores = as.data.frame(p2$scores)

names(scores) = c("defense","offense","performance","rebound_freethrows_composition");

scores$TORD = train$TORD

scores$ADJ_T = train$ADJ_T

scores$FTR = train$FTR


scores = scale(scores)

scores=as.data.frame(scores)

scores$RegW = train$RegW


scores_correlation <- cor(scores)

corrplot(scores_correlation, method = "circle", order="AOE")


res1 <- cor.mtest(scores_correlation, conf.level = .95)

corrplot(scores_correlation, p.mat = res1$p, sig.level = .05)


McorrTest <- corr.test(scores_correlation, adjust="none")

round(McorrTest$p, 2)

Mtest <- ifelse(McorrTest$p < 0.05,T,F)

Mtest
```

```
colSums(Mtest) - 1


library(car)

library("QuantPsyc")

fit = lm(RegW ~ ., data=scores)

summary(fit)

vif(fit)


sqrt(mean(fit$residuals^2))


library(MASS)


b = lm.beta(fit)

barplot(sort(b))


#testing#

test_pred_factors = as.data.frame(predict(p2,cbb_numeric_test))

names(test_pred_factors) =
c("defense","offense","performance","rebound_freethrows_composition");

test_pred_factors$TORD = test$TORD

test_pred_factors$ADJ_T = test$ADJ_T

test_pred_factors$FTR = test$FTR

test_pred_factors = as.data.frame(scale(test_pred_factors))
```

```
library(Metrics)
test_pred = predict(fit,test_pred_factors)
rmse_test = rmse(test$RegW,test_pred)
rmse_test

summary(train$RegW)
summary(test$RegW)
dim(scores)


anova(fit,bestfitAdjR2)

#Quality of model fit analysis#

#residuals#
#q-q plot#
#histogram of normalized residuals#
#fitted vs residuals#
plot(fit)


library(glmnet)
```

```
xTrain = as.matrix(scale(scores[-8]))

yTrain = as.matrix(scores[8])


xTest = as.matrix(scale(test_pred_factors))

yTest = as.matrix(test$RegW)

#try lasso as well for the final paper#

fitLasso_cv = cv.glmnet(xTrain,yTrain,alpha=1,nfolds=10)

plot(fitLasso_cv)

fitLasso_cv$lambda.min

fitLasso_cv$lambda.1se


lassoPredTrain = predict(fitLasso_cv, xTrain, s="lambda.1se")

rmseLassoTrain = sqrt(mean((lassoPredTrain - yTrain)^2))

rmseLassoTrain


lassoPred = predict(fitLasso_cv, xTest, s="lambda.1se")

lassoRidge = sqrt(mean((lassoPred - yTest)^2))

lassoRidge


preds_factors = predict(fitLasso_cv,type="coef", s = 0.1322782)

preds_factors


xTrain_init = as.matrix(scale(subset(train, select=
-c(RegW,RegL,POSTSEASON,WAB,BARTHAG))))

yTrain_init = as.matrix(subset(train, select=c(RegW)))
```

```
xTest_init = as.matrix(scale(subset(test, select=
-c(RegW,RegL,POSTSEASON,WAB,BARTHAG))))

yTest_init = as.matrix(subset(test, select=c(RegW)))


#try lasso for initial variables and compare both#

fitLasso_cv = cv.glmnet(xTrain_init,yTrain_init,alpha=1,nfolds=10)

plot(fitLasso_cv)

fitLasso_cv$lambda.min

fitLasso_cv$lambda.1se


lassoPredTrain = predict(fitLasso_cv, xTrain_init, s="lambda.1se")

rmseLassoTrain = sqrt(mean((lassoPredTrain - yTrain_init)^2))

rmseLassoTrain


lassoPred = predict(fitLasso_cv, xTest_init, s="lambda.1se")

rmseLasso = sqrt(mean((lassoPred - yTest_init)^2))

rmseLasso


preds_all = predict(fitLasso_cv,type="coef", s = 0.1234496)

preds_all


lasso_fit =


# Compute the distance function (since all are numeric)
```

```
train_mds = subset(cbb_numeric_trans, select=-c(POSTSEASON))
train_mds.dist = dist(train_mds)
train_mds.dist
train_mds.dist.mds = isoMDS(train_mds.dist)  # cmdscale doesn't have a nice Shepard diagram
train_mds.dist.mds$stress   # Note ... this is in %, so is very low = .023

# Now plot the MDS
plot(train_mds.dist.mds$points)
text(x,y, labels = row.names(cbb_numeric_train), cex = 0.7)

#Shepard#
p_shepard = Shepard(train_mds.dist, train_mds.dist.mds$points)
plot(p_shepard, pch=".")

require(fastcluster)

cluster1 = hclust(train_mds.dist)
plot(cluster1, cex=0.6, hang = -1)
clusters = cutree(cluster1, k = 6)

plot(train_mds.dist.mds$points, col=clusters)
text(x,y, labels = row.names(cbb), cex = 0.5)

table(cbb[,24],clusters)
```

Code for Approach 2

```
library(tidyverse)
library(car)
library(caret)
library(glmnet)

cbb <- cbbUpdated %>% select(-X)
row.names(cbb) <- paste(cbb$TEAM,cbb$YEAR, sep = "_")
cbb <- cbb %>% select(-c("TEAM", "YEAR", "RegGames"))

cbb$Power5 <- ifelse(cbb$CONF %in% c("ACC", "B10", "B12", "SEC", "P12"), 1, 0)

cbb <- cbb %>% select(-"CONF")

cbb$SEED <- replace_na(cbb$SEED, "NIT")
cbb$SEED <- as.factor(cbb$SEED)


cbbDummies <- dummyVars(~ ., data=cbb)
c <- as.data.frame(predict(cbbDummies, newdata = cbb))
summary(c)


set.seed(489)
s <- sample(nrow(c), nrow(c)*.8)
train <- c[s,]
test <- c[-s,]


# All subsets ----------------------------------------------------------------------
s1 <- sample(nrow(cbb), nrow(cbb)*.8)
trainNonDummy <- cbb[s1,]
testNonDummy <- cbb[-s1,]


library(leaps)
allSub <- regsubsets(POSTSEASON ~ . , data=trainNonDummy, nbest=5)
plot(allSub, scale="adjr2")
bestAllSub <- lm(POSTSEASON ~ ADJOE + ADJDE + SEED + Power5 + DRB, data =
trainNonDummy)
summary(bestAllSub)
```

```
vif(bestAllSub)
plot(bestAllSub)

rmseAlltr <- sqrt(mean((bestAllSub$residuals)^2))
rmseAlltr

allPred <- predict(bestAllSub, testNonDummy)
rmseAlltest <- sqrt(mean((testNonDummy$POSTSEASON - allPred)^2))
rmseAlltest
rmseAlltest/rmseAlltr
lm.beta(bestAllSub)
# Setup for regularized regression ----------
library(glmnet)
xTrain <- as.matrix(train[,-20])
yTrain <- as.matrix(train[,20])

xTest <- as.matrix(test[,-20])
yTest <- as.matrix(test[,20])
# Ridge ----------
cbbRidge <- cv.glmnet(xTrain, yTrain, alpha=0)
plot(cbbRidge)
cbbRidge$glmnet.fit
lambda1se <- cbbRidge$lambda.1se
ridgePredTest <- predict(cbbRidge, xTest, s="lambda.1se")
rmseRidgeTest<- sqrt(mean((ridgePredTest - yTest)^2))
rmseRidgeTest

ridgePredTrain <- predict(cbbRidge, xTrain, s="lambda.1se")
rmseRidgeTrain<- sqrt(mean((ridgePredTrain - yTrain)^2))
rmseRidgeTrain

rsq <- glmnet(xTrain, yTrain, alpha=0, lambda= lambda1se)
rsq$dev.ratio

rmseRidgeTest/rmseRidgeTrain

# Lasso ----------
cbbLasso <- cv.glmnet(xTrain, yTrain, alpha=1)
plot(cbbLasso)
lambda1seR <- cbbLasso$lambda.1se
lassoPredTest <- predict(cbbLasso, xTest, s="lambda.1se")
rmseLassoTest <- sqrt(mean((lassoPredTest - yTest)^2))

lassoPredTrain <- predict(cbbLasso, xTrain, s="lambda.1se")
```

```
rmseLassoTrain <- sqrt(mean((lassoPredTrain - yTrain)^2))

rsqL <- glmnet(xTrain, yTrain, alpha = 1, lambda = lambda1seR)
rsqL
coef(cbbLasso)
rmseLassoTest/rmseLassoTrain

# Relaxed Lasso ----------------------------------------------------------------
cbbLassoR <- cv.glmnet(xTrain, yTrain, alpha=1, relax = T)
plot(cbbLassoR)
lambda1seR <- cbbLassoR$lambda.1se
lassoPredTestR <- predict(cbbLassoR, xTest, s="lambda.1se")
rmseLassoTestR <- sqrt(mean((lassoPredTestR - yTest)^2))
rmseLassoTestR
lassoPredTrainR <- predict(cbbLassoR, xTrain, s="lambda.1se")
rmseLassoTrainR <- sqrt(mean((lassoPredTrainR - yTrain)^2))
rmseLassoTrainR
rmseLassoTestR/rmseLassoTrainR
rsqLR <- glmnet(xTrain, yTrain, alpha = 1, lambda = lambda1seR, relax=T)
rsqLR$dev.ratio


# Lasso into Ridge ----------------------------------------------------------
xTrainL <- as.matrix(train[,c(3, 4, 19, 21, 29, 30, 31, 37, 38)])
yTrainL <- as.matrix(train[,20])

xTestL <- as.matrix(test[,c(3, 4, 19, 21, 29, 30, 31, 37, 38)])
yTestL <- as.matrix(test[,20])


cbbRidgeL <- cv.glmnet(xTrainL, yTrainL, alpha=0)
lambda1seL <- cbbRidgeL$lambda.1se
ridgePredTestL <- predict(cbbRidgeL, xTestL, s="lambda.1se")
rmseRidgeTestL<- sqrt(mean((ridgePredTestL - yTestL)^2))
rmseRidgeTestL

ridgePredTrainL <- predict(cbbRidgeL, xTrainL, s="lambda.1se")
rmseRidgeTrainL<- sqrt(mean((ridgePredTrainL - yTrainL)^2))
rmseRidgeTrainL

rmseRidgeTestL/rmseRidgeTrainL

rsqLr <- glmnet(xTrainL, yTrainL, alpha=0, lambda= lambda1se)
rsqLr$dev.ratio
```

```
coef(cbbRidgeL)


summary(cbbRidge)
# Elastic Net ----------------------------------------------------------------
elastic <- cv.glmnet(xTrain, yTrain, alpha = .25)
trainPE <- predict(elastic, xTrain, s="lambda.1se")
rmseTrainE <- sqrt(mean((yTrain - trainPE)^2))
rmseTrainE
testPE <- predict(elastic, xTest, s="lambda.1se")
rmseTestE <- sqrt(mean((yTest - testPE)^2))
rmseTestE
rmseTestE/rmseTrainE
rsqEL <- glmnet(xTrain, yTrain, alpha = .25, lambda = elastic$lambda.1se)
rsqEL$dev.ratio

elastic <- cv.glmnet(xTrain, yTrain, alpha = .5)
trainPE <- predict(elastic, xTrain, s="lambda.1se")
rmseTrainE <- sqrt(mean((yTrain - trainPE)^2))
rmseTrainE
testPE <- predict(elastic, xTest, s="lambda.1se")
rmseTestE <- sqrt(mean((yTest - testPE)^2))
rmseTestE

elastic <- cv.glmnet(xTrain, yTrain, alpha = .75)
trainPE <- predict(elastic, xTrain, s="lambda.1se")
rmseTrainE <- sqrt(mean((yTrain - trainPE)^2))
rmseTrainE
testPE <- predict(elastic, xTest, s="lambda.1se")
rmseTestE <- sqrt(mean((yTest - testPE)^2))
rmseTestE

coef(elastic)

# PCA ------------------------------------------------------------------------
library(car)
library(lm.beta)
library(corrplot)
library(psych)
corr.test(c, adjust = "none")
pca1 <- prcomp(c[-c(20)], scale = T)
plot(pca1)
summary(pca1)
```

```
screeplot(pca1)


pcData <- as.data.frame(pca1$x)
pcData$POSTSEASON <- c$POSTSEASON
print(pca1$rotation, cutoff = .4)

round(pca1$rotation[,25], 2)[which(abs(pca1$rotation[,25]) > .25)]
round(pca1$rotation[,33], 2)[which(abs(pca1$rotation[,33]) > .25)]
round(pca1$rotation[,20], 2)[which(abs(pca1$rotation[,20]) > .25)]

s2 <- sample(nrow(pcData), nrow(pcData)*.8)
trainPC <- pcData[s2,]
testPC <- pcData[-s2,]

pcLm <- lm(POSTSEASON ~ ., data = trainPC)
summary(pcLm)
pcLm <- lm(POSTSEASON ~ PC1 + PC3 + PC4 + PC7 + PC9 + PC13 + PC16 + PC18 +  PC20
+ PC21 + PC23 + PC25 + PC26 + PC28 + PC33, data = pcData)
summary(pcLm)

rmsePCtr <- sqrt(mean((pcLm$residuals)^2))
rmsePCtr

pcPred <- predict(pcLm, testPC)
rmsePCtest <- sqrt(mean((testPC$POSTSEASON - pcPred)^2))
rmsePCtest
rmsePCtest/rmsePCtr

stdCoef <- lm.beta(pcLm)
sort(abs(stdCoef$coefficients))

# Table ----------------------------------------------------------------------
tbl <- data.frame("Type" = "AllSubsets",
                  "r squared" = .59, "RMSE Ratio" = rmseAlltest/rmseAlltr)
tbl <- tbl %>% rbind(list("Ridge", rsq$dev.ratio, rmseRidgeTest/rmseRidgeTrain))
tbl <- tbl %>% rbind(list("Lasso", rsqL$dev.ratio, rmseLassoTest/rmseLassoTrain))
tbl <- tbl %>% rbind(list("Relaxed Lasso", rsqLR$dev.ratio, rmseLassoTestR/rmseLassoTrainR))
tbl <- tbl %>% rbind(list("Lasso into Ridge", rsqL$dev.ratio, rmseRidgeTestL/rmseRidgeTrainL))
tbl <- tbl %>% rbind(list("Elastic Net", rsqEL$dev.ratio,rmseTestE/rmseTrainE))
tbl <- tbl %>% rbind(list("PCA components", .56, rmsePCtest/rmsePCtr))
tbl

plot(tbl$r.squared, col = tbl$Type)
```

```
barplot(tbl$r.squared, names.arg = tbl$Type, ylab = "R Squared")
barplot(tbl$RMSE.Ratio, names.arg = tbl$Type, ylab = "Testing Performance", ylim = c(0,1.2))
```