STAT 101C – Final Report

Basketball Game

101CMonsters: Tuyen Duong, Dara Hashemi, Ki Hyun Park.

1. Abstract:

In this assignment we used data studiously collected by a basketball fan and a former analyst for the Dodgers (2000 - 2012) to build a model to predict the outcome: winning or losing for home team after each basketball game from 2013 to 2014. The training data provides different metrics of both opposing teams along with date, league, id, gameID, team name in each game. The testing data includes the same variables except for the results. We used different models and classification methods to predict outcomes for the testing data and submitted each attempt on Kaggle to check the accuracy. Eventually, the method which gave us the best score was Principal Component Regression with about 68% accuracy rate.

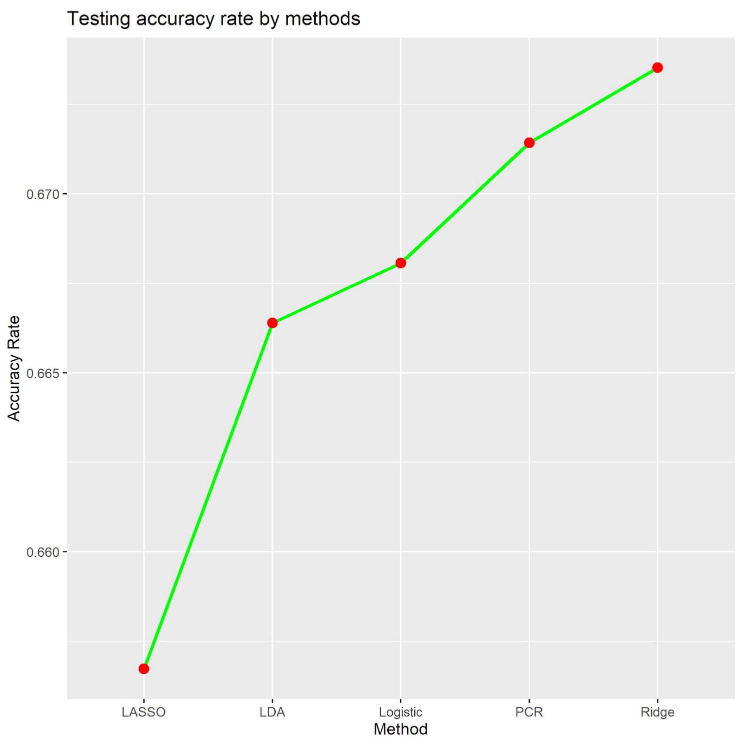
2. Data Preparation:

First, we checked if there was any missing value in our data. There were no NAs or missing values presented in the training and testing data. Next, we summarized each variable to check if they carried any invalid value under their context. All variables contained values within an appropriate range. There were no outstanding outliers or suspected values.

Before the analysis, we removed variables which did not play a big role in our data. They were id, gameID, HT, VT, HT league, VT league, and date. Because PCR only accepts the response variable as binary, we created a new column called HTWins01 from HTWins that coded "Yes" as 1 and "No" as 0. All other metrics would be used in the model as covariates.

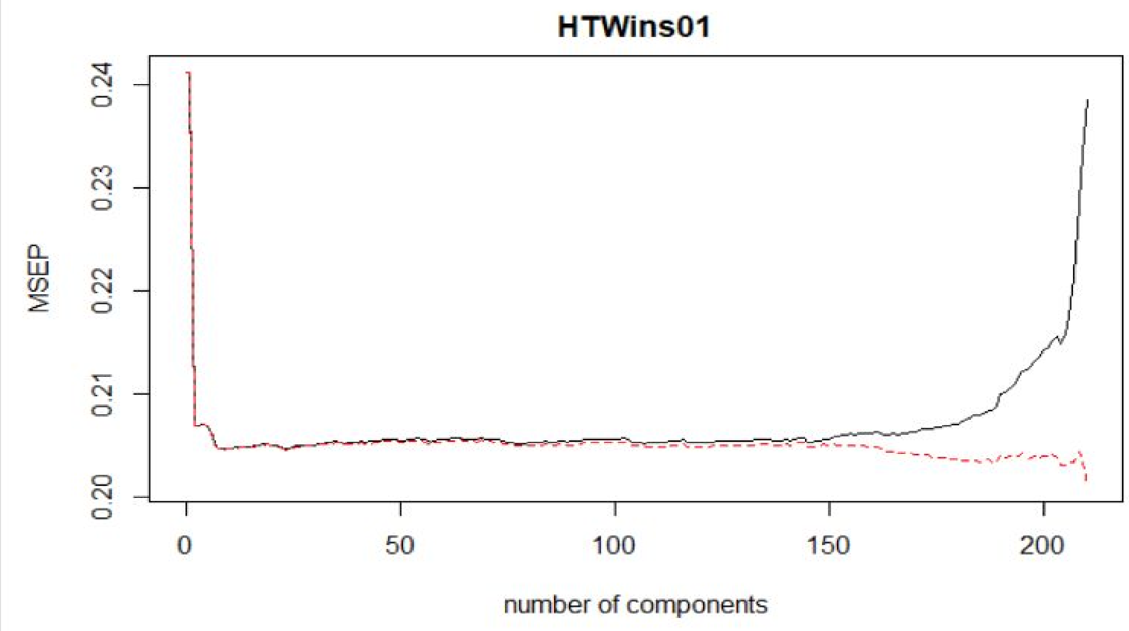
3. Description of Model / Best Model

After creating a model using each method, we compared the accuracy rates found using a testing data set made from our actual training data. The results are shown below.



The two methods which performed best on our testing data in terms of accuracy rate are Principal Component Regression and Ridge Regression. Although the Ridge accuracy rate was higher on our testing data, the best score on Kaggle was given when we used the PCR method.

Our best model on Kaggle utilized Principal Component Regression accompanied by cross validation. To minimize overfitting, we first randomly split the full training data into training and testing with a ratio of 75:25. We used the function "pcr()" available in R to fit the response variable which had been coded as the binary outcome against all other metrics.



According to the validation plot, Mean Squared Error drastically decreases at around 13 components and does not change much with a higher number of components. At 150 components, the MSE increases again. This indicates that there is a high collinearity among many variables in our model, as more variables being used will not help improve the testing MSE. The best accuracy rate we got when applying this model to the testing set (set aside from the full training set) was 67.14% and 68% for the testing data from 2013-2014.

4. Why does Principal Component Work Well?

The full data set consists of 217 variables in total, which raised issues of collinearity and variable selection. Because some variables are already explained by others, PCR helps to reduce the complexity while still being able to capture the variance explained by these variables. In our specific case, information is given for both the home team and visiting team, some variables overlap others. Also, measurements like "fgm" and "fga" could be highly correlated, and PCR plays a role in solving these problems.

Although correlation issues were mostly solved by PCR, the accuracy rate was not so high. There were some problems coming from the data which may have affected our prediction.

i. In this analysis, we did not split the data by time, but by randomness. The result of each game should be predicted by those games which have already been played. The performance of a team or its players could also be affected by the preceding games.
ii. Each team will have different characteristics that might not be captured with one generalized model. An approach for this is ideally fitting a model for each home team versus all other teams.
iii. Current variables may not provide enough information to fully predict the result of each game. Variable transformation is necessary to improve the predictive power but due to time

limit we could only use all raw data. Some approaches which may have been helpful is moving average and cumulative score of each metric of each team.

In conclusion, PCR is helpful when dealing with large data sets that contains multicollinearity. The accuracy rate can be improved by learning more about the data and the methods used to transform the variables.

5. Codes:

```
train <- fread("train.csv", stringsAsFactors = TRUE)

train2 <- train[, -c(1, 2, 4, 5, 6, 7, 8)]

## Split train.csv into training and testing

set.seed(123)

train.i <- sample(1:nrow(train2), 0.75 * nrow(train2), replace = FALSE)

train.tr <- train2[train.i, ]

train.te <- train2[-train.i, ]


train.tr01 <- train.tr

train.tr01$HTWins01 <- as.numeric(train.tr$HTWins == "Yes")

train.te01 <- train.te

train.te01$HTWins01 <- as.numeric(train.te$HTWins == "Yes")


# PC regression

library(pls)

set.seed(123)

train.pcr <- pcr(HTWins01 ~. , data = train.tr01[, -1], scale = TRUE, validation = "CV")

train.pcr$coefficients[,,13]

validationplot(train.pcr, val.type = "MSEP")

train.pcr$loadings

train.pcr.pred <- predict(train.pcr,         # predict testing

                newdata = train.te01,

                ncomp = 13, type = "response")

train.pcr.pred01 <- as.numeric(train.pcr.pred > 0.5)


## Apply PCA to test.csv

test <- fread("test.csv", stringsAsFactors = TRUE)

test2 <- train[, -c(1,2,4:8)]
```

```
train.pcr.pred <- predict(train.pcr,         # predict testing
                 newdata = test2,
                 ncomp = 13, type = "response")
train.pcr.pred01 <- ifelse(train.pcr.pred > 0.5, "Yes", "No")
## Export the results
library(dplyr)
test.predict <- bind_cols(id = test2$id, HTWins = train.pcr.pred01)
write.csv(test.predict, "test_predict3.csv", row.names = FALSE)
```