

**BUDT 758T**

**assignment #5: 100 PTS**

The goal of this homework is to introduce you to working with subset selection, LASSO, and Ridge regression. As with Assignment 4, you will need to create random partitions of a data set, build your model on the training data set, and then compute prediction errors using the test data set. You are required to complete this assignment in R—be sure to include the code you used and output any results you use!

**The Data (Same data as Assignment 4!)**

Average theater movie revenues have been in decline in recent years, even as major blockbusters and superhero franchises soar. To investigate this phenomenon, major motion picture studios in the United States want to identify factors that lead to a movie being financially successful (where “successful” is defined as a movie making at least twice in revenue than it had for a budget). To this end, studio executives have collected information on thousands of movies from 2000 to 2017, with information collected from the popular movie database site TMDb (The Movie Database).

This data is collected in the CSV file *movies\_data.csv* on ELMS. It is based on a popular Kaggle database that can be found at <https://www.kaggle.com/tmdb/tmdb-movie-metadata/version/2>.

The variables in *movie\_data.csv* are defined as follows:

|  |  |
| --- | --- |
| **Variable** | **Description** |
| *id* | An ID number describing the location of the movie in the data set |
| *title* | The title of the movie (when released) |
| *genre* | A primary genre of the movie |
| *budget* | The budget of the movie |
| *revenue* | The total revenue of the movie |
| *production\_companies* | The number of production companies involved in making the movie |
| *united\_states* | Whether any part of the film was produced in the United States ("Yes") or not ("No") |
| *english* | Whether the film's dialogue is in English ("Yes") or not ("No") |
| *title\_change* | Whether the film was released under a different title than it was originally produced ("Yes") or not ("No") |
| *popularity* | A TMDb value for how popular the movie is on their site (where a larger number means the movie is more popular) |
| *vote\_average* | The average voter rating for the movie |
| *vote\_count* | The total number of votes the movie has received |
| *month* | The month in which the movie was released (1 = January through 12 = December) |
| *year* | The year in which the movie was released |
| *runtime* | The total runtime of the movie (in minutes) |
| *successful* | Was the movie financially successful (1) or not (0) |

**Assignment**

Please answer all questions in the dedicated space and upload on Canvas. Please ensure that your numbering of questions matches those below. You must include your R code, but because this assignment has significantly more R code than previous assignments, you are welcome to include your full code at the end of the assignment file rather than including it with the appropriate question. However, you should make sure any output that is requested or necessary to answer the question is including with the question. Any additional output you wish to provide may be included at the end of your assignment in an appendix, if you wish.

Remember: you are allowed to consult with others in the class on this assignment, but all submitted work must be your own (and don’t forget to include the names of anyone you consulted in the last question!).

1. **15 points: Data Preparation**
   1. Read the data set into R. Remember to use *read.csv()* rather than importing it from the folder directly (which uses *read\_csv()*) to make sure you do not encounter NA values!

movies <- read.csv("movies\_data.csv")

* 1. Change the *month* variable to a factor variable.

movies$month <- as.factor(movies$month)

* 1. Set the seed in R to 6022.

set.seed(6022)

* 1. Randomly partition the data set in the following order (note that if you do *not* follow this order, many of the questions in this assignment will not make sense to you!):
     1. Split 30% of the observations in *movies\_data* to use as testing data. Using these observations, create a testing data set called *movies\_test.*

num\_obs <- nrow(movies)

test\_obs <- sample(num\_obs, 0.30\*num\_obs)

movies\_test <- movies[test\_obs, ]

* + 1. Save the remaining 70% of the data as *movies\_rest*.

movies\_rest <- movies[-test\_obs, ]

* + 1. Split 20% of the observations in *movies\_rest* to use as validation data. Using these observations, create a validation data set called *movies\_valid.*

num\_obs <- nrow(movies\_rest)

valid\_obs <- sample(num\_obs, 0.20\*num\_obs)

movies\_valid <- movies\_rest[valid\_obs, ]

* + 1. Save the remaining data as *movies\_train*.

movies\_train <- movies\_rest[-valid\_obs, ]

* 1. Compared to Assignment 4, we have more testing data but less validation data. Given the methods we want to consider in this assignment (stepwise regression, LASSO, and Ridge regression) compared to the methods used in the previous assignment (trees and KNN), why might we have made this change?

Step, LASSO, and Ridge are all regression procedures; Step does not use validation data to choose a model, and LASSO and Ridge use cross-validation. This means we likely want to have a smaller validation data set, because these models do not rely on validation. KNN and Trees, on the other hand, use validation data to pick model parameters (the value of k and the size of the tree, respectively). We need to make sure we have a large validation data set to get representative data to choose these parameters. Since validation data is not so necessary, we instead shift that data to the testing data set. That should give us a better idea of how our models will perform in practice.

1. **20 points: Run a regression procedure to predict *revenue* from all other variables except *id, title,* and *successful* on the training data using:**

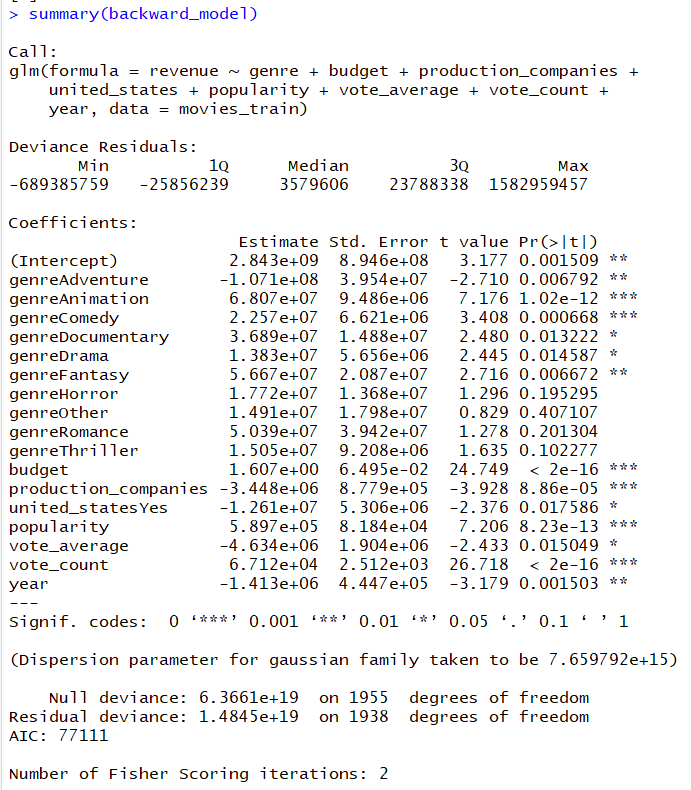
* **Backward elimination**
* **Forward selection**

movies\_all <- glm(revenue~.-id-title-successful, data=movies\_train)

movies\_null <- glm(revenue~1, data=movies\_train)

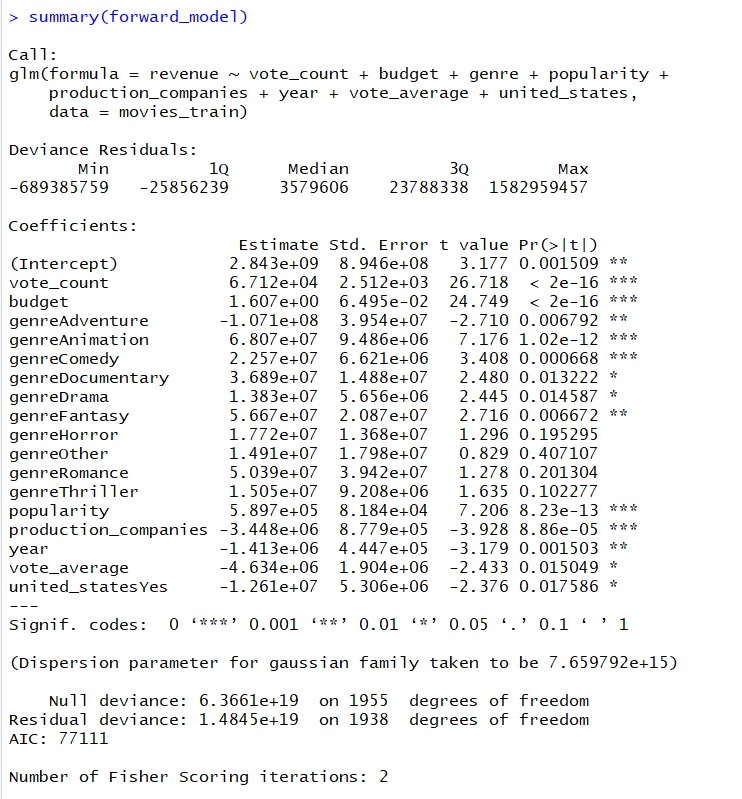
backward\_model = step(movies\_all, direction="backward")

summary(backward\_model)



forward\_model = step(movies\_null, scope=list(upper=movies\_all), direction="forward")

summary(forward\_model)



* 1. Which variable did backward elimination eliminate first? Which variable did forward selection select first?

Backward elimination eliminated ‘month’ first.

Forward selection selected ‘vote\_count’ first.

* 1. Consider the final model for both step procedures.
     1. What type of model did you get?

Linear Regression

* + 1. Based on these results, do you think we should also consider running a stepwise forward or stepwise backward procedure? Explain your answer.

Based on these results, I think we need not consider running a stepwise forward or stepwise backward because we got the same models finally for both the methods.

But it’s always better to run stepwise procedures to see if the AIC decreases because stepwise models run some extra models including/excluding some variables.

* 1. Choose the best of the two models based on prediction.
     1. Which model did you choose?

Both the models are exactly the same, so we could choose either of them.

* + 1. What measure did you use for prediction?

**RMSE**

backward\_pred\_val <- predict(backward\_model, newdata=movies\_valid)

RMSE\_back\_val <- sqrt(mean((backward\_pred\_val-movies\_valid$revenue)^2))

forward\_pred\_val <- predict(forward\_model, newdata=movies\_valid)

RMSE\_forw\_val <- sqrt(mean((forward\_pred\_val-movies\_valid$revenue)^2))

* + 1. What was the value of that measure for this model?

74620182 was the RMSE for both the models.

1. **20 points: Run a regression procedure to predict *successful* from all other variables except *id, title,* and *revenue* on the training data using:**

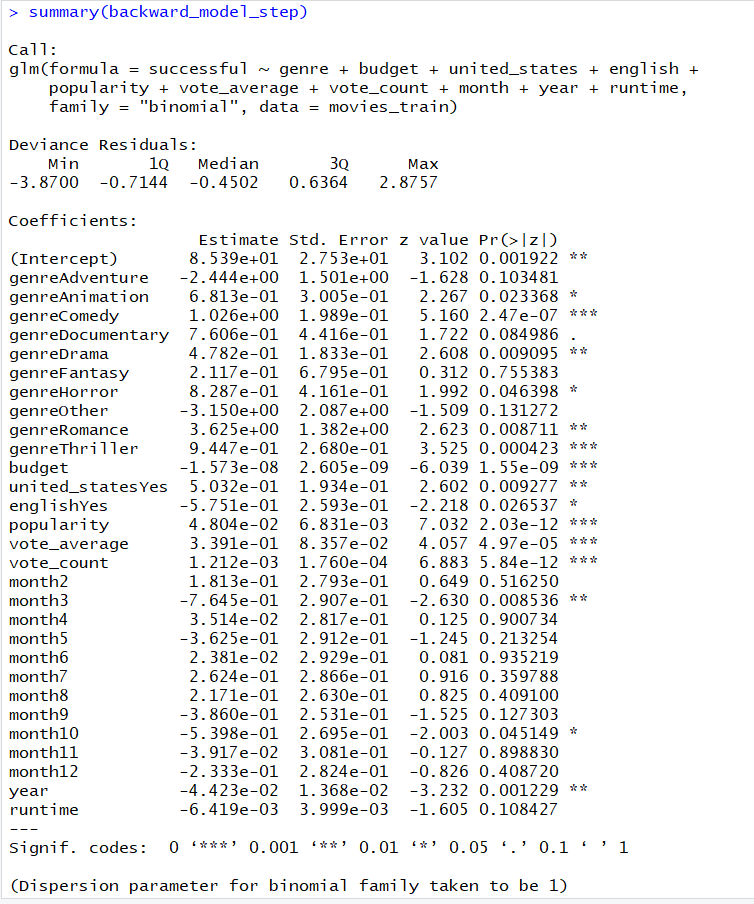
* **Backward stepwise**
* **Forward stepwise**

successful\_all <- glm(successful~.-id-title-revenue, family="binomial", data=movies\_train)

successful\_null <- glm(successful~1, family="binomial", data=movies\_train)

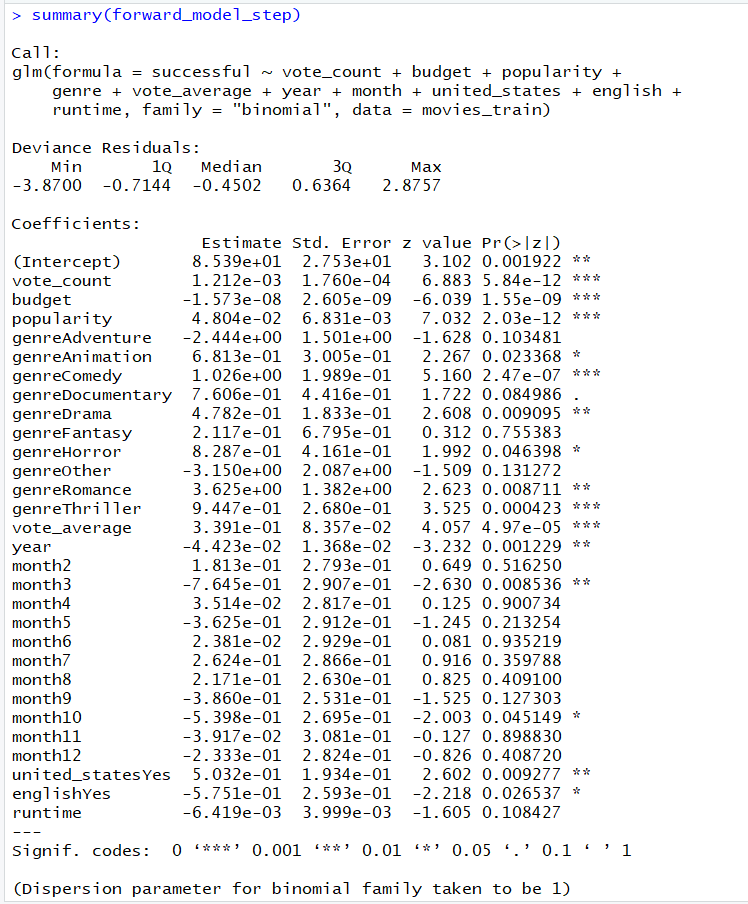
backward\_model\_step = step(successful\_all, direction="both")

summary(backward\_model\_step)



forward\_model\_step = step(successful\_null, scope=list(upper=movies\_all), direction="both")

summary(forward\_model\_step)



* 1. *Without running any additional models,* were these procedures ultimately the same as running the standard backward elimination and forward selection for this problem, or did they take additional steps? Support your answer.

Yes, for this problem these procedures were ultimately the same as running the standard backward elimination and forward selection when we see the final results.

But in the background these models tried extra combinations compared to the standard elimination methods.

* 1. Calculate and report model accuracy for these models with cutoffs of 0.2, 0.4, 0.5, 0.6, 0.7, and 0.8.
     1. Which data set did you use for this?

Validation data (movies\_valid)

* + 1. Which cutoff achieves the highest accuracy (and on which model)?

**Backward Stepwise Model :**

for(i in c(0.2, 0.4, 0.5, 0.6, 0.7, 0.8)) {

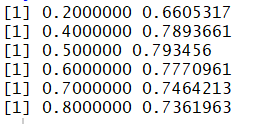
back\_probs = predict(backward\_model\_step,newdata = movies\_valid,type="response")

back\_class = ifelse(back\_probs>i,1,0)

acc = sum(ifelse(back\_class==movies\_valid$successful,1,0))/nrow(movies\_valid)

print(c(i, acc))

}



**Forward Stepwise Model :**

for(i in c(0.2, 0.4, 0.5, 0.6, 0.7, 0.8)) {

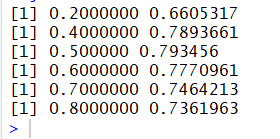
forward\_probs = predict(forward\_model\_step,newdata = movies\_valid,type="response")

forward\_class = ifelse(forward\_probs>i,1,0)

acc = sum(ifelse(forward\_class==movies\_valid$successful,1,0))/nrow(movies\_valid)

print(c(i, acc))

}



As we can see, the highest accuracy is 0.793456 for a cutoff of 0.5 for both the models.

1. **10 points: Comparison to Assignment 4**
   1. In both Question (3) above and in Assignment 4, we used accuracy to measure the results. Do you think that is a good prediction measure to use for this problem? Support your answer.

While accuracy may seem to be good enough, True positive rate would have made more sense here as there are only 38% movies that are successful. And also because we are more concerned if the movie is successful.

* 1. Could we have used the methods of Assignment 4 (trees and kNN) to solve the same problem as in Question (2) (in Assignment 5)? Support your answer.

Yes, we could have used Regression Trees to predict revenue and we have the option of pruning the tree which is essentially similar to that of elimination procedures.

But we can’t use kNN to solve a regression problem.

1. **25 points: LASSO and Ridge**
   1. We often use the *model.matrix()* function on our data before we enter it into LASSO or Ridge calculation.
      1. Why do we do this?

Because these models require numerical values to do the calculations in the background and model.matrix() converts factors into numerical dummy variables.

* + 1. What other method have we seen so far that might benefit from using *model.matrix()* before we ran it?

Knn because it requires only numeric variables.

* 1. Use *model.matrix()* on the same X variables in Question (3) to get new data to run LASSO and Ridge.

movies\_rest\_X <- model.matrix( ~ .-1, movies\_rest[,c(3,4,6:15)])

movies\_test\_X <- model.matrix( ~ .-1, movies\_test[,c(3,4,6:15)])

* 1. Use the new matrix from part (b) to run both LASSO and Ridge (logistic) procedures to predict *successful*. Use cross-validation to choose the best value and report the values you find. Which data set did you use to run these procedures?

**Ridge :**

glmnet\_ridge=glmnet(movies\_rest\_X,movies\_rest$successful,family="binomial",alpha=0)

glmnet\_ridge.cv=cv.glmnet(movies\_rest\_X,movies\_rest$successful,family="binomial",alpha=0)

plot(glmnet\_ridge.cv)

best.lambda\_r=glmnet\_ridge.cv$lambda.min

best.lambda\_r



**LASSO :**

glmnet\_lasso=glmnet(movies\_rest\_X,movies\_rest$successful,family="binomial",alpha=1)

glmnet\_lasso.cv=cv.glmnet(movies\_rest\_X,movies\_rest$successful,family="binomial",alpha=1)

plot(glmnet\_lasso.cv)

best.lambda\_l=glmnet\_lasso.cv$lambda.min

best.lambda\_l



I used the movies\_rest data set for the above procedures.

* 1. How many variables does your best Ridge model use? How many variables does your best LASSO model use?

predict(glmnet\_ridge,s=best.lambda\_r,type="coefficients")

predict(glmnet\_lasso,s=best.lambda\_l,type="coefficients")

Best Ridge model uses 32 variables.

Best LASSO model uses variables 29 variables.

1. **10 points: Choosing a logistic-based model**
   1. Using a cutoff of 0.5, which model would you suggest we use in practice: your stepwise logistic model, your LASSO model, or your Ridge model? Support your answer.

**Logistic model (Backward step) :**

successful\_retrained = glm(successful ~ genre + budget + united\_states + english

+ popularity + vote\_average + vote\_count + month + year + runtime,

family = "binomial", data = movies\_rest)

backward\_model\_step\_retrained = step(successful\_retrained, direction="both")

summary(backward\_model\_step\_retrained)

back\_probs = predict(backward\_model\_step\_retrained,newdata = movies\_test,type="response")

back\_class = ifelse(back\_probs>0.5,1,0)

acc = sum(ifelse(back\_class==movies\_test$successful,1,0))/nrow(movies\_test)



**Logistic model (Forward step) :**

forward\_model\_step\_retrained = step(successful\_retrained, direction="both")

summary(forward\_model\_step\_retrained)

forward\_probs = predict(forward\_model\_step\_retrained,newdata = movies\_test,type="response")

forward\_class = ifelse(forward\_probs>0.5,1,0)

acc = sum(ifelse(forward\_class==movies\_test$successful,1,0))/nrow(movies\_test)

acc



**LASSO model :**

lasso\_probs = predict(glmnet\_lasso,s=best.lambda\_l,newx=movies\_test\_X,type="response")

lasso\_class = ifelse(lasso\_probs>0.5,1,0)

acc = sum(ifelse(lasso\_class==movies\_test$successful,1,0))/nrow(movies\_test)



**Ridge model :**

ridge\_probs = predict(glmnet\_ridge,s=best.lambda\_r,newx=movies\_test\_X,type="response")

ridge\_class = ifelse(ridge\_probs>0.5,1,0)

acc = sum(ifelse(ridge\_class==movies\_test$successful,1,0))/nrow(movies\_test)



As we can see above , we will choose LASSO model because it has got the highest testing accuracy among the three models.

* 1. Based on accuracy and your results from Assignment 4 (or from the answer key for Assignment 4 if you prefer), are logistic regression-based methods like these three better for this problem than trees or kNN? Support your answer.

Based on our results from Assignment 4 we can’t comment anything here because the testing data sets are different in both the assignments and hence it’s not practical to compare these testing results either.