**Homework Assignment 3**

*(due through Blackboard before end-of-day on 4/5/2020)*

Things you should please do, lest you should lose my tender affection:

* Incorporate your last name into the name of the solution file you upload.
* Make sure your name is inside the document itself on the Name: line below.
* All code must be text inside of the document that I can copy and paste – do not include screenshots of code.
* Make sure that your work is original – **do not copy your code from or share your code with anybody else in class.**

**Name: Indraja Nutalapati**

Directions: **Show *any code required for all the following steps*** and include graphics or other output where specifically requested. I won’t repeatedly ask for the code for each step – just understand that you are always supposed to supply the code unless the step doesn’t require any code (e.g., because it is asking you to interpret some previously supplied output).

Also, if the step asks a question, you must **directly answer the question** – don’t just paste the output from R. You should include the output, but also tell me what it means.

**1. Predicting Social Media Engagement (60 points)**

In this exercise, you will try to predict the number of lifetime impressions a Facebook post will get.

a. You will be working with the facebook\_posts.csv file in Blackboard. Download that file to your local filesystem. Load it into a data frame called *posts* using the read.csv2 function, which is meant for files that use a semicolon as a column separator.

**posts <- read.csv2(file = 'D:/Data Mining & Data Base Systems/R/facebook\_posts.csv')**

b. Remove any rows that do not have “Photo” in the *Type* column.

**posts <- subset(posts, Type == 'Photo')**

c. Now, remove the *Type* column (it can no longer have any predictive value as it is the same for all rows), the *Total.Interactions* column, and any column with *Lifetime* in its title, except for *Lifetime.Post.Total.Impressions*, which we are going to use as our target variable (the variable to be predicted). This means you will be removing 9 columns. You can achieve this using whatever code you like. We saw one way to do this in class, but there are many ways to do most tasks in R.

**posts <- subset( posts, select = -c(Type, Total.Interactions, Lifetime.Post.Total.Reach, Lifetime.Engaged.Users,Lifetime.Post.Consumers, Lifetime.Post.Consumptions,Lifetime.Post.Impressions.by.people.who.have.liked.your.Page, Lifetime.Post.reach.by.people.who.like.your.Page, Lifetime.People.who.have.liked.your.Page.and.engaged.with.your.post) )**

d. Practice saying the word “okurrr” in the voice of Cardi B. Keep going until you have it just right. Once all this COVID-19 madness calms down, we will have a special competition in which everyone will try to talk like Cardi B and the winner will get a miniature ostrich. I am lying. None of this will happen. Ignore this step.

e. Use the summary() function to investigate your *posts* data frame. Paste the output of the function here.

**summary(posts)**

**Page.total.likes Category Post.Month**

**Min. : 81370 Min. :1.000 Min. : 1.000**

**1st Qu.:109670 1st Qu.:1.000 1st Qu.: 4.000**

**Median :128032 Median :2.000 Median : 7.000**

**Mean :122354 Mean :1.918 Mean : 6.805**

**3rd Qu.:136013 3rd Qu.:3.000 3rd Qu.:10.000**

**Max. :139441 Max. :3.000 Max. :12.000**

**Post.Weekday Post.Hour Paid**

**Min. :1.000 Min. : 1.000 Min. :0.00**

**1st Qu.:2.000 1st Qu.: 3.000 1st Qu.:0.00**

**Median :4.000 Median : 9.000 Median :0.00**

**Mean :4.108 Mean : 7.998 Mean :0.28**

**3rd Qu.:6.000 3rd Qu.:11.000 3rd Qu.:1.00**

**Max. :7.000 Max. :23.000 Max. :1.00**

**NA's :1**

**Lifetime.Post.Total.Impressions comment**

**Min. : 570 Min. : 0.000**

**1st Qu.: 5390 1st Qu.: 1.000**

**Median : 8118 Median : 3.000**

**Mean : 28995 Mean : 7.493**

**3rd Qu.: 17014 3rd Qu.: 7.000**

**Max. :1110282 Max. :372.000**

**like share**

**Min. : 0.0 Min. : 0.00**

**1st Qu.: 57.0 1st Qu.: 10.00**

**Median : 100.0 Median : 19.00**

**Mean : 182.6 Mean : 27.16**

**3rd Qu.: 186.0 3rd Qu.: 32.00**

**Max. :5172.0 Max. :790.00**

**NA's :1 NA's :4**

f. Fit a linear model to your data. The linear model should try to predict *Lifetime.Post.Total.Impressions* (our target variable) based on all the other columns in the data frame. Name your model *lm.all\_predictors*.

**lm.all\_predictors = lm(Lifetime.Post.Total.Impressions~., data=posts)**

g. Use the summary() function to see the characteristics of your *lm.all\_predictors* model. Paste the output of the function here.

**summary(lm.all\_predictors)**

**Call:**

**lm(formula = Lifetime.Post.Total.Impressions ~ ., data = posts)**

**Residuals:**

**Min 1Q Median 3Q Max**

**-113420 -18903 -8995 1668 1039786**

**Coefficients:**

**Estimate Std. Error t value Pr(>|t|)**

**(Intercept) 1.634e+05 6.311e+04 2.589 0.00997**

**Page.total.likes -9.758e-01 6.817e-01 -1.431 0.15311**

**Category -1.133e+04 4.406e+03 -2.571 0.01048**

**Post.Month 1.430e+02 3.449e+03 0.041 0.96695**

**Post.Weekday 1.746e+02 1.785e+03 0.098 0.92215**

**Post.Hour -7.046e+02 8.579e+02 -0.821 0.41194**

**Paid 2.616e+03 8.164e+03 0.320 0.74885**

**comment 7.068e+02 3.801e+02 1.859 0.06369**

**like 1.285e+02 2.642e+01 4.864 1.64e-06**

**share -6.884e+02 2.459e+02 -2.800 0.00535**

**(Intercept) \*\***

**Page.total.likes**

**Category \***

**Post.Month**

**Post.Weekday**

**Post.Hour**

**Paid**

**comment .**

**like \*\*\***

**share \*\***

**---**

**Signif. codes:**

**0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**Residual standard error: 74550 on 411 degrees of freedom**

**(5 observations deleted due to missingness)**

**Multiple R-squared: 0.1789, Adjusted R-squared: 0.1609**

**F-statistic: 9.951 on 9 and 411 DF, p-value: 7.437e-14**

h. Based on the p-values and significance codes you see in your *lm.all\_predictors* model, which predictor variables have such a strong apparent relationship with the target variable that the chances of these variables having no real connection to the target variable are less than 5%? (In short, we are looking for p-values of less than 0.05.) List those variables here.

**Category**

**like**

**share**

i. Based on your *lm.all\_predictors* model, how much would you expect each *like* to impact the value of *Lifetime.Post.Total.Impressions*? Will each *like* increase or decrease the target value? By how much? Make sure you express the “how much” as a decimal value – convert it from scientific notation if that’s how R is reporting it. (Google will do this right in the search box if you search “<value> to decimal” and replace <value> with the value in scientific notation.)

**Each like increases the target value by 128.5**

j. How much of the variation in *Lifetime.Post.Total.Impressions* does your *lm.all\_predictors* model explain? (Hint: Look at R-squared for the model.)

**The variation in *Lifetime.Post.Total.Impressions*** **by the model lm.all\_predictors is 0.1609**

k. Now, fit a new linear model to your *posts* data set. The new model should try to predict *Lifetime.Post.Total.Impressions* (our target variable) based on just the *like* column. Name your model *lm.like*.

**lm.like = lm(Lifetime.Post.Total.Impressions~like, data=posts)**

l. Use the summary() function to see the characteristics of your *lm.like* model. Paste the output of the function here.

**summary(lm.like)**

**Call:**

**lm(formula = Lifetime.Post.Total.Impressions ~ like, data = posts)**

**Residuals:**

**Min 1Q Median 3Q Max**

**-116066 -17061 -13797 -9300 1077611**

**Coefficients:**

**Estimate Std. Error t value Pr(>|t|)**

**(Intercept) 14192.85 4177.25 3.398 0.000744 \*\*\***

**like 81.40 10.71 7.604 1.88e-13 \*\*\***

**---**

**Signif. codes:**

**0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**Residual standard error: 76100 on 423 degrees of freedom**

**(1 observation deleted due to missingness)**

**Multiple R-squared: 0.1203, Adjusted R-squared: 0.1182**

**F-statistic: 57.82 on 1 and 423 DF, p-value: 1.876e-13**

m. Based on your *lm.like* model, how much would you expect each *like* to impact the value of *Lifetime.Post.Total.Impressions*? Will each *like* increase or decrease the target value? By how much? Again, make sure you express the “how much” as a decimal value.

**Each like increases the target value by 81.40**

n. How much of the variation in *Lifetime.Post.Total.Impressions* does your *lm.like* model explain? Does it explain more or less than *lm.all\_predictors*?

**The variation in *Lifetime.Post.Total.Impressions*** **by the model lm.like is 0.1182 and it is less than lm.all\_predictors**

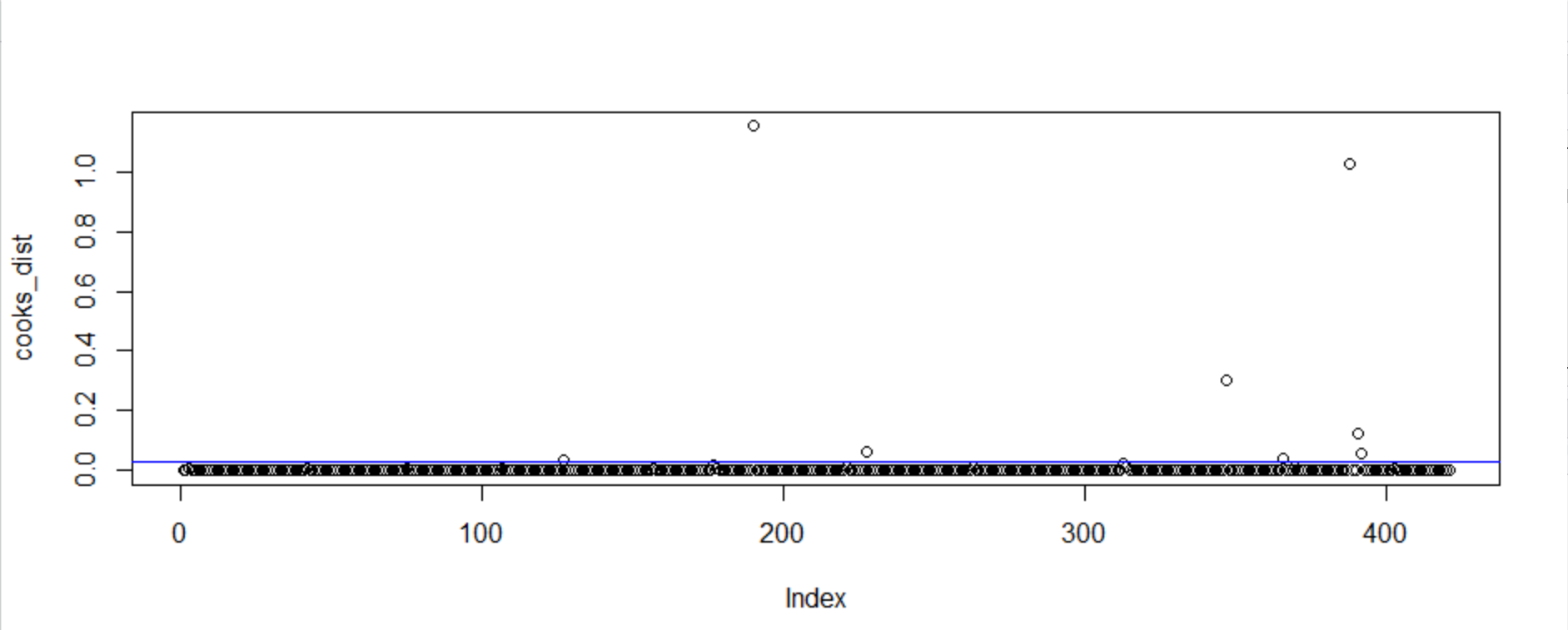
o. Calculate Cook’s Distance for all the observations in your *lm.all\_predictors* model and save those distances in a vector called *cooks\_dist*.

**cooks\_dist <- cooks.distance(lm.all\_predictors)**

p. Plot those distances on a graph and draw a horizontal cutoff line at 4 times the average Cook’s Distance for these observations to show which ones we will consider outliers. Label the outlier points with their row names. Show your completed plot.

**plot(cooks\_dist)**

**abline(h=4 \* mean(cooks\_dist),col="blue")**



q. Now, save the row names of these outlier observations into a new vector called *outliers*.

**outliers <- as.numeric(row.names(as.data.frame(cooks\_dist[cooks\_dist >= 4 \* mean(cooks\_dist)])))**

r. Create a new data frame called *posts\_rm\_out* that contains the data from posts but with the outlier observations removed.

**posts\_rm\_out <- posts[!row.names(posts) %in% outliers, ]**

s. Create a new linear model based on *posts\_rm\_out*. The new model should try to predict *Lifetime.Post.Total.Impressions* (our target variable) based on all the other columns in the data frame. Name your new model *lm.all\_predictors\_rm\_out*.

**lm.all\_predictors\_rm\_out = lm(Lifetime.Post.Total.Impressions~., data = posts\_rm\_out)**

t. Use the summary() function to see the characteristics of your *lm.all\_predictors\_rm\_out* model. Paste the output of the function here.

**summary(lm.all\_predictors\_rm\_out)**

**Call:**

**lm(formula = Lifetime.Post.Total.Impressions ~ ., data = posts\_rm\_out)**

**Residuals:**

**Min 1Q Median 3Q Max**

**-62173 -12336 -6056 4175 220899**

**Coefficients:**

**Estimate Std. Error t value Pr(>|t|)**

**(Intercept) 1.065e+04 2.532e+04 0.421 0.67428**

**Page.total.likes 2.824e-01 2.721e-01 1.038 0.29993**

**Category -1.077e+04 1.783e+03 -6.038 3.55e-09**

**Post.Month -2.189e+03 1.366e+03 -1.603 0.10969**

**Post.Weekday -7.569e+02 7.056e+02 -1.073 0.28406**

**Post.Hour -1.375e+02 3.389e+02 -0.406 0.68507**

**Paid 8.777e+03 3.239e+03 2.710 0.00703**

**comment -8.732e+01 2.231e+02 -0.391 0.69571**

**like 8.359e+01 1.101e+01 7.590 2.24e-13**

**share 5.481e+00 1.187e+02 0.046 0.96320**

**(Intercept)**

**Page.total.likes**

**Category \*\*\***

**Post.Month**

**Post.Weekday**

**Post.Hour**

**Paid \*\***

**comment**

**like \*\*\***

**share**

**---**

**Signif. codes:**

**0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**Residual standard error: 29280 on 403 degrees of freedom**

**(5 observations deleted due to missingness)**

**Multiple R-squared: 0.326, Adjusted R-squared: 0.311**

**F-statistic: 21.66 on 9 and 403 DF, p-value: < 2.2e-16**

u. Compare your new *lm.all\_predictors\_rm\_out* model to your previous *lm.all\_predictors* model. Is there any difference in the variables considered significant (having a p-value less than 0.05)? Is the effect of each *like* on your target variable stronger or weaker in the new model?

**Yes, there is a difference in the variables considered to be significant. The variables considered significant in the new model are also Category, paid and like.**

**The effect of each like on the target variable is stronger in the new model.**

v. Now, fit one last linear model to your *posts\_rm\_out* data set. The new model should try to predict *Lifetime.Post.Total.Impressions* based on just the *like* column. Name your model *lm.like\_rm\_out*.

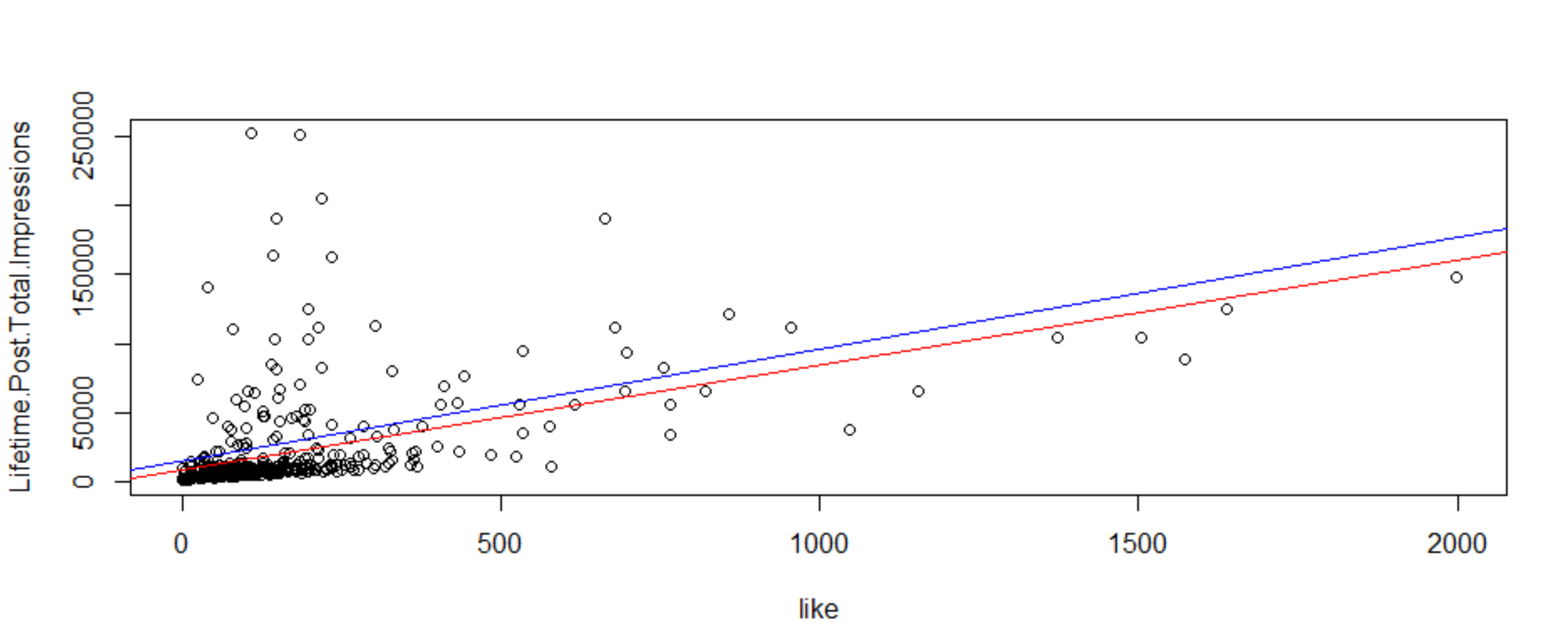
**lm.like\_rm\_out <- lm(Lifetime.Post.Total.Impressions~like, data=posts\_rm\_out)**

w. Create a scatterplot of *posts\_rm\_out* data that shows *Lifetime.Post.Total.Impressions* and *like* on the same graph. Plot the lines for both your *lm.like* and *lm.like\_rm\_out* models on that one graph. Paste the graph here.

**plot(posts\_rm\_out$like, posts\_rm\_out$Lifetime.Post.Total.Impressions, ylab = "Lifetime.Post.Total.Impressions", xlab = "like" )**

**abline(lm.like, col = "blue")**

**abline(lm.like\_rm\_out, col = "red")**



x. Now, considering posts with the following characteristics:

Page.total.likes:100000

Category:1

Post.Month:10

Post.Weekday:7

Post.Hour:13

Paid:1

comment:48

like:2019

share:144

Using your *lm.all\_predictors\_rm\_out* model, show the interval in which the mean *Lifetime.Post.Total.Impressions* for all such posts will fall with 95% confidence.

**new\_data <- data.frame(Page.total.likes=c(100000),Category=c(1),Post.Month=c(10),Post.Weekday=c(7),Post.Hour=c(13),Paid=c(1),comment=c(48),like=c(2019),share=c(144),Lifetime.Post.Total.Impressions=c(mean(posts\_rm\_out$Lifetime.Post.Total.Impressions)))**

**predict(lm.all\_predictors\_rm\_out, newdata=new\_data, interval = "confidence")**

**fit lwr upr**

**173277.6 140026.7 206528.5**

y. Now assume a single post having the characteristics mentioned in the previous step. Using your *lm.all\_predictors\_rm\_out* model, predict the number of *Lifetime.Post.Total.Impressions* this post will get (i.e., fit), and determine the minimum number of *Lifetime.Post.Total.Impressions* you would expect this post to get (with 95% confidence).

**new\_data <- data.frame(Page.total.likes=100000,Category=1,Post.Month=10,Post.Weekday=7,Post.Hour=13,Paid=1,comment=48,like=2019,share=144)**

**predict(lm.all\_predictors\_rm\_out, newdata=new\_data,interval="predict")**

**fit lwr upr**

**173277.6 106794.9 239760.3**

**The number of *Lifetime.Post.Total.Impressions* this post will get is “173277.6”**

**The minimum number of *Lifetime.Post.Total.Impressions* this post would get (with 95% confidence)is “106794.9”**

z. Again, assume a single post with all the same characteristics listed in step x, except that it has a *like* value of 2020. Using your *lm.all\_predictors\_rm\_out* model, show your best prediction (fit) for its *Lifetime.Post.Total.Impressions*. How does this value compare to the value you predicted in the previous step? Does the difference in the prediction make sense based on what you know about the *lm.all\_predictors\_rm\_out* model? Explain your answer.

**new\_data <- data.frame(Page.total.likes=100000,Category=1,Post.Month=10,Post.Weekday=7,Post.Hour=13,Paid=1,comment=48,like=2020,share=144)**

**predict(lm.all\_predictors\_rm\_out, newdata=new\_data,interval="predict")**

**fit lwr upr**

**173361.2 106872.2 239850.2**

**The best prediction value for *Lifetime.Post.Total.Impressions* has increased from “173277.6” to “173361.2”. The difference in the prediction makes sense that change in one like would impact the prediction of *Lifetime.Post.Total.Impressions* and the effect is stronger on the variable in the new model**

**2. IMDB Score Predictions (40 points)**

You have been asked to predict whether or not a movie is likely to receive a high score on IMDB.

a. You will be working with the imdb.csv file in Blackboard. Download that file to your local filesystem. Load it into a data frame called *movies* using the read.csv function. You will be building a model to predicting the values in the *imdb\_score\_high* column. A “1” designates a high-scoring movie and a “0” designates a movie that does not have a high score.

**movies <- read.csv("D:/Data Mining & Data Base Systems/R/imdb.csv")**

b. Break the *movies* data into two sets, a training data set called *movies\_train* and a test data set called *movies\_test*. The training data set should contain 80% of the records from *movies*, and the test data set should contain the remaining 20%. You should split the two data sets using a random sample, and you should use your own unique Villanova Id number to set the seed before taking your sample.

**train\_pct <- 0.8**

**movies\_train\_rows <- sample(1:nrow(movies\_clean),**

**train\_pct \* nrow(movies\_clean))**

**movies\_train <- movies\_clean[movies\_train\_rows,]**

**movies\_test <- movies\_clean[-movies\_train\_rows,]**

c. Build a logistic regression model from your training data set. Your model should predict the value of *imdb\_score\_high* based on all the other variables. Call your model whatever you like.

**glm\_fit <- glm(imdb\_score\_high~., data=movies\_train, family=binomial)**

d. Run the summary() function on your model and show the output here. Are there any variables that are not significant at the 0.05 level? If so, list those explicitly. Otherwise, state that there are none.

**summary(glm\_fit)**

**Call:**

**glm(formula = imdb\_score\_high ~ ., family = binomial, data = movies\_train)**

**Deviance Residuals:**

**Min 1Q Median 3Q Max**

**-3.1428 -0.8446 -0.5165 0.9220 3.5272**

**Coefficients:**

**Estimate Std. Error z value Pr(>|z|)**

**(Intercept) 7.231e+01 1.378e+01 5.249 1.53e-07 \*\*\***

**num\_critic\_for\_reviews 2.192e-03 8.412e-04 2.606 0.009166 \*\***

**duration 2.383e-02 3.041e-03 7.835 4.69e-15 \*\*\***

**director\_facebook\_likes 3.204e-05 2.089e-05 1.534 0.125120**

**lead\_actor\_facebook\_likes 1.525e-05 1.132e-05 1.347 0.177955**

**gross -1.035e-08 1.388e-09 -7.457 8.85e-14 \*\*\***

**num\_voted\_users 1.250e-05 1.152e-06 10.851 < 2e-16 \*\*\***

**cast\_total\_facebook\_likes -1.556e-05 1.027e-05 -1.516 0.129574**

**facenumber\_in\_poster -9.318e-02 2.619e-02 -3.558 0.000374 \*\*\***

**num\_user\_for\_reviews -7.918e-04 2.332e-04 -3.395 0.000686 \*\*\***

**budget -1.449e-09 1.596e-09 -0.908 0.363981**

**title\_year -3.679e-02 6.905e-03 -5.329 9.90e-08 \*\*\***

**aspect\_ratio -1.012e+00 2.195e-01 -4.610 4.02e-06 \*\*\***

**movie\_facebook\_likes 7.189e-06 4.769e-06 1.507 0.131694**

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**(Dispersion parameter for binomial family taken to be 1)**

**Null deviance: 2749.2 on 1987 degrees of freedom**

**Residual deviance: 2097.2 on 1974 degrees of freedom**

**AIC: 2125.2**

**Number of Fisher Scoring iterations: 6**

**Yes, there are variables that are not significant at 0.05 level**

**lead\_actor\_facebook\_likes**

**cast\_total\_facebook\_likes**

**budget**

e. Based on your results, which variables are positively associated with a high IMDB score? (In other words, as the value of those variables increases, a high IMDB score becomes more likely.)

**Following variables are positively associated with a high IMDB score**

**num\_critic\_for\_reviews**

**duration**

**director\_facebook\_likes**

**lead\_actor\_facebook\_likes**

**num\_voted\_users**

**movie\_facebook\_likes**

f. Based on your results, which variables are negatively associated with a high IMDB score? (In other words, as the value of those variables increases, a high IMDB score becomes less likely.)

**Following variables are negatively associated with a high IMDB score**

**Gross**

**cast\_total\_facebook\_likes**

**facenumber\_in\_poster**

**num\_user\_for\_reviews**

**budget**

**title\_year**

**aspect\_ratio**

g. Now, generate a set of predictions for *imdb\_score\_high* on your test data set, and create a confusion matrix to compare these predictions to the real values (ground truth) from the test data set. Show the confusion matrix output.

**target\_threshold <- .3**

**glm\_fit\_probs <- predict(glm\_fit,type="response",**

**newdata = movies\_test)**

**glm\_fit\_preds <- rep("No", nrow(movies\_test))**

**glm\_fit\_preds[glm\_fit\_probs > target\_threshold]="Yes"**

**confusionMatrix(as.factor(glm\_fit\_preds),movies\_test$imdb\_score\_high, positive = "Yes", mode = "prec\_recall")**

**Confusion Matrix and Statistics**

**Reference**

**Prediction No Yes**

**No 136 29**

**Yes 129 203**

**Accuracy : 0.6821**

**95% CI : (0.6392, 0.7228)**

**No Information Rate : 0.5332**

**P-Value [Acc > NIR] : 1.024e-11**

**Kappa : 0.3781**

**Mcnemar's Test P-Value : 3.380e-15**

**Precision : 0.6114**

**Recall : 0.8750**

**F1 : 0.7199**

**Prevalence : 0.4668**

**Detection Rate : 0.4085**

**Detection Prevalence : 0.6680**

**Balanced Accuracy : 0.6941**

**'Positive' Class : Yes**

h. What was the overall classification accuracy of your model?

**Overall classification accuracy of the model is 68%**

i. Did your model get a higher score on precision or recall?

**The model got a higher score on recall**

j. Let’s assume that you want to improve whichever score was lower (either precision or recall). Adjust the threshold value for your predictions in a way that will improve this score and then regenerate your predictions. State the new threshold value that you used.

**New Threshold used is 0.45**

k. Create a new confusion matrix for your new predictions. Show the output.

**target\_threshold <- .45**

**glm\_fit\_probs <- predict(glm\_fit,type="response",**

**newdata = movies\_test)**

**glm\_fit\_preds <- rep("No", nrow(movies\_test))**

**glm\_fit\_preds[glm\_fit\_probs > target\_threshold]="Yes"**

**confusionMatrix(as.factor(glm\_fit\_preds),movies\_test$imdb\_score\_high, positive = "Yes", mode = "prec\_recall")**

**Confusion Matrix and Statistics**

**Reference**

**Prediction No Yes**

**No 209 76**

**Yes 56 156**

**Accuracy : 0.7344**

**95% CI : (0.6933, 0.7727)**

**No Information Rate : 0.5332**

**P-Value [Acc > NIR] : < 2e-16**

**Kappa : 0.4636**

**Mcnemar's Test P-Value : 0.09818**

**Precision : 0.7358**

**Recall : 0.6724**

**F1 : 0.7027**

**Prevalence : 0.4668**

**Detection Rate : 0.3139**

**Detection Prevalence : 0.4266**

**Balanced Accuracy : 0.7305**

**'Positive' Class : Yes**

l. Did you improve the score (either precision or recall) you had hoped to improve? How did the new set of predictions affect the other score (the one that had been higher after your first set of predictions)?

**Yes, improved precision**. **Both accuracy value got increased and recall value got decreased.**

m. Draw a ROC curve for your model. Show the ROC curve below and state the AUC your model achieved on the test data set.

**roc\_obj <- roc(movies\_test$imdb\_score\_high, glm\_fit\_probs)**

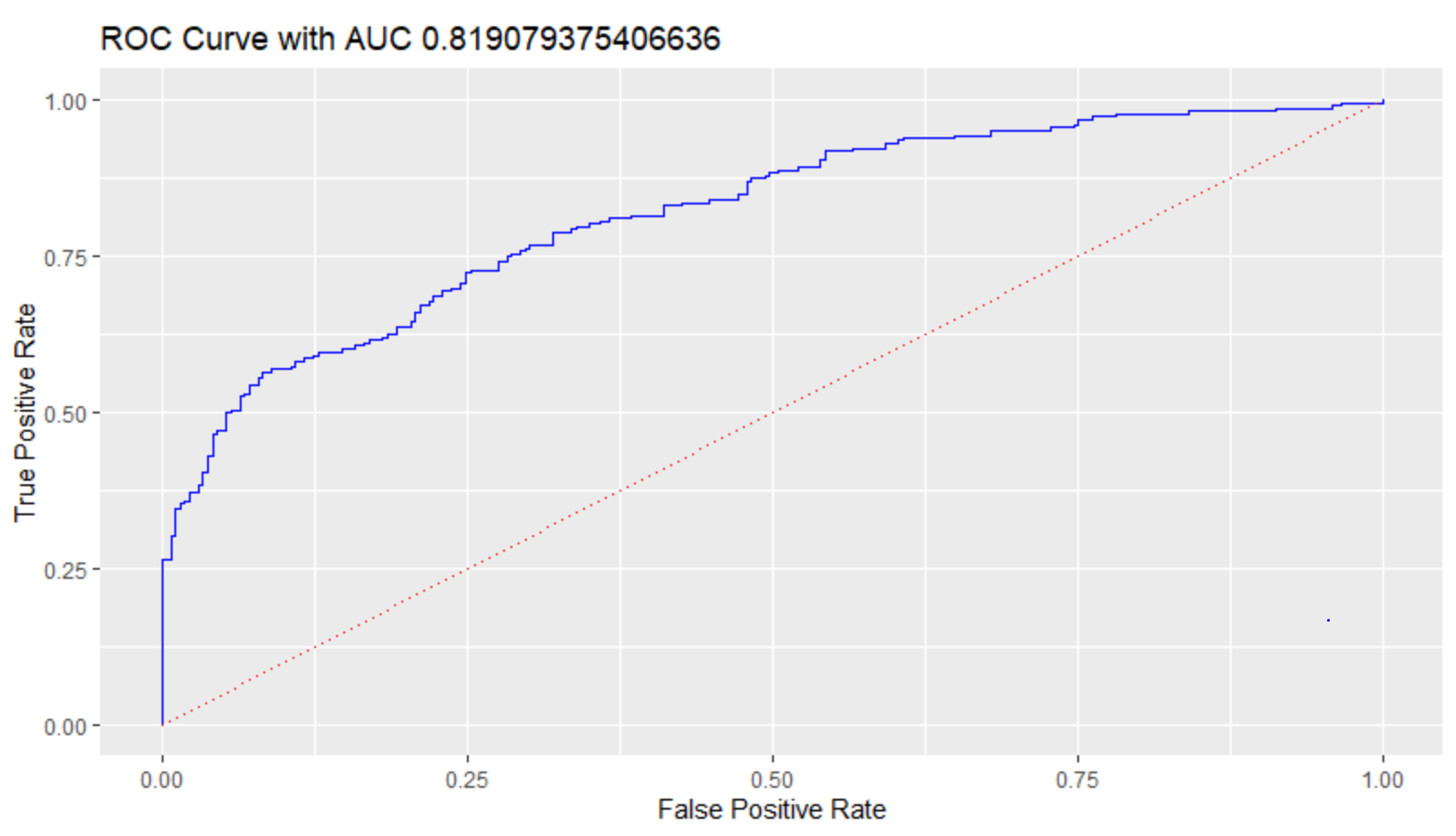
**roc\_obj$auc**

**roc\_plot <- ggroc(roc\_obj, legacy.axes = TRUE, color="blue")**

**roc\_plot + xlab("False Positive Rate") + ylab("True Positive Rate") +**

**geom\_segment(aes(x = 0, xend = 1, y = 0, yend = 1),**

**color="red", linetype="dotted") +ggtitle(paste("ROC Curve with AUC",roc\_obj$auc))**



**Area under the curve: 0.8191**

n. Imagine a movie with the following characteristics:

|  |  |
| --- | --- |
| num\_critic\_for\_reviews | 97 |
| duration | 110 |
| director\_facebook\_likes | 54 |
| lead\_actor\_facebook\_likes | 1137 |
| gross | 14681875 |
| num\_voted\_users | 12281 |
| cast\_total\_facebook\_likes | 3165 |
| facenumber\_in\_poster | 1 |
| num\_user\_for\_reviews | 37 |
| budget | 25000000 |
| title\_year | 2014 |
| aspect\_ratio | 2.35 |
| movie\_facebook\_likes | 19000 |
| **new\_data <- data.frame(num\_critic\_for\_reviews=97, duration=110,director\_facebook\_likes=54,**  **lead\_actor\_facebook\_likes=1137,**  **gross=14681875,**  **num\_voted\_users=12281,**  **cast\_total\_facebook\_likes=3165,**  **facenumber\_in\_poster=1,**  **num\_user\_for\_reviews=37,**  **budget=25000000,**  **title\_year=2014,**  **aspect\_ratio=2.35,**  **movie\_facebook\_likes=19000**  **)**  **glm\_fit\_probs <- predict(glm\_fit,type="response",**  **newdata = new\_data)**  **glm\_fit\_probs** |  |

Based on your model, what is the probability that this move will get a high rating on IMDB?

**The probability that this movie will get a high rating on IMDB is 0.2005401**

o. Now, pick one of the variables above and modify it in a way that will make a high rating on IMDB more likely. Describe which variable you modified and how you modified it. Then, show the new probability of a high score as predicted by your model.

**new\_data <- data.frame(num\_critic\_for\_reviews=97, duration=110,director\_facebook\_likes=54,**

**lead\_actor\_facebook\_likes=1137,**

**gross=14681875,**

**num\_voted\_users=15000,**

**cast\_total\_facebook\_likes=3165,**

**facenumber\_in\_poster=1,**

**num\_user\_for\_reviews=37,**

**budget=25000000,**

**title\_year=2014,**

**aspect\_ratio=2.35,**

**movie\_facebook\_likes=19000**

**)**

**glm\_fit\_probs <- predict(glm\_fit,type="response",**

**newdata = new\_data)**

**glm\_fit\_probs**

**I have picked num\_voted\_users and increased the value from 12281 to 17000**

**New probability of a high score as predicted by the model is 0.2101628.**