PREDICTION OF GRADUATE ADMISSION APPLICATION OUTCOME

TEAM 8

Rushika Bejjanki

Danying He

Giri Nanduru

Jiyad Ur Rehman

May 14, 2019

Introduction

In the life of a student approaching their graduation or has already acquired bachelor's degree and exploring options to continue higher education into graduate school, he/she is confronted with a variety of choices to pick from. In preparation to apply to graduate schools that spark interest in the potential admission seeker, the prospective student must fit the profile typical of incoming student population for that university. The profile is closely related to the reputation of the academic department as well as the academic institution, built upon the faculty eminence, facilities, research opportunities etc. Similarly, the student candidature is assessed using benchmark testing, performance in the undergraduate curriculum, letters of recommendation, clearly stated academic goals and objectives, the reputation of the college/university awarding the bachelor's degree etc. This project seeks to statistically analyze a Kaggle dataset consisting of quantified prospective student data elements and developing a model that predicts the admission outcome (Mohan S Acharya, 2019).

The prospective student might be prompted to ponder upon the question "can I get into a university of my dreams"? This could be a hypothetical question akin to knowing the result prior to beginning a game. One way to treat the question as a probability problem solved using data analytics using a set of predictors that yield a decimal probability of acceptance to a generic graduate school whose applicants & decision data is available. The probability could then be translated into binary yes/no response using a conversion logic. A data analytics expert is able to support the solution by quantifying the accuracy of outcome from the estimated model for prediction.

A student may also begin with the question "what scores or documents are required for consideration of my application to be complete?" in order to preselect school(s). This relates to the data analytics question "what are all the predictors and what is our response variable in my dataset?" Inspection of dataset elements is helpful in this regard. The predictors are the student performance metrics GRE Score, TOEFL Score, SOP, LOR, CGPA, and Research. The response variable is the Chance of Admit value. The column Serial No. is not relevant as a model predictor.

A follow-up to the initial question maybe to ask, "What factors can bolster my chances of admission to the graduate school I'm pursuing?" From data analytics perspective, it may reframe

as “which set of predictors are most effective in predicting the response?” Initial step is to understand the nature of the distribution of data. Normal data distribution indicates a linear relationship between predictor(s) and response. The degree of multicollinearity between predictors can be established using a combination of correlation plots. The model output parameters such as confidence level (p-values), degree of explained (regression sum of squares) and unexplained variation (residual sum of squares) in the error output, accuracy from confusion matrix can provide scientific basis to the solution.

Which data analytic techniques are suited for the graduate admission prediction? The team will apply multiple regression methods to compare the results to determine the best model and present the output to support the analysis. In addition, team will employ the data visualization concepts and the best practices learned from the Applied Statistics and Visualization course to graphically present the solution for ease of comprehension for the target audience whose focus is on seeking answers to the subject questions rather than get immersed with the complexity of data analytics.

Model Parameters and Terminology

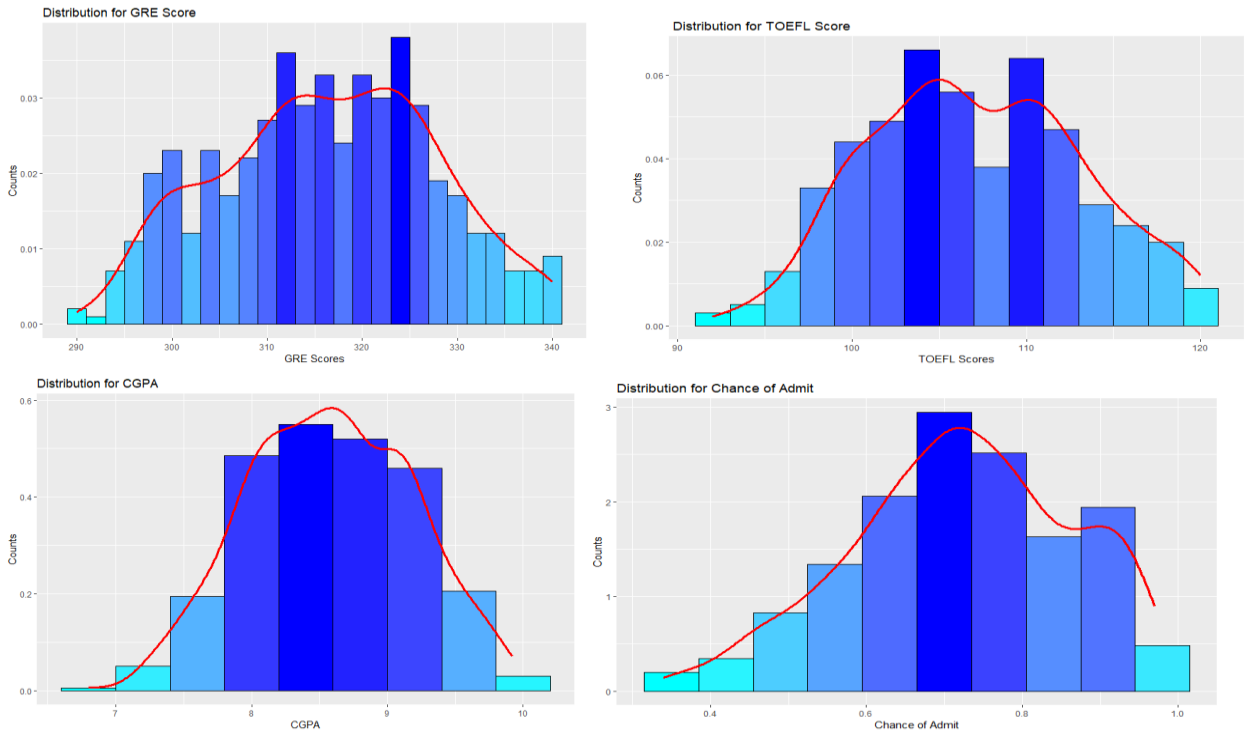
The admission dataset consists of 500 records formatted in a comma delimited file downloaded from Kaggle (Mohan S Acharya, 2019). Using a combination of information gleaned from the dataset author and interpretation, we present the context of the variables included in the dataset below.

Table 1: Admission Dataset Variables

Attribute	R Dataframe Type	Description
Serial No.	Numeric Integer	Row Number in the Kaggle dataset – unused
GRE Score	Numeric Integer	Standardized test, continuous variable, out of 340
TOEFL Score	Numeric Integer	Standardized test, continuous variable, out of 120
University Rating	Numeric	Hypothetical Rating of Institution awarding applicant’s bachelor’s degree on a scale of 1-5
SOP	Numeric	Strength of applicant’s statement of purpose (goals statement) on a scale of 1-5, step 0.5

LOR	Numeric	Strength of letter(s) of recommendation, expert opinion on applicant's academic preparation on a scale of 1-5, step 0.5
CGPA	Numeric	Cumulative GPA, continuous variable, out of 10
Research	Numerical => Logical	Binary value, if the applicant has prior research experience. Numerical value converted into logical value (True/False)
Chance of Admit	Numeric	Response variable for acceptance, decimal value, range 0 to 1

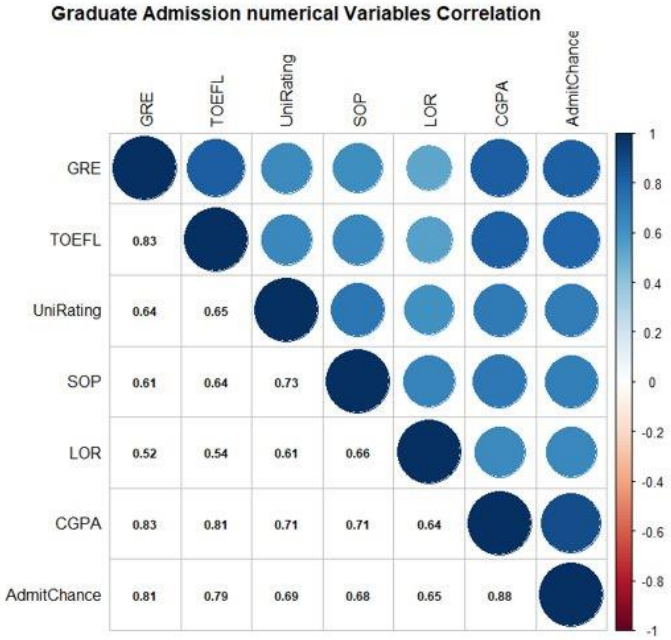
Data Analysis



The team analyzed the GRE Scores, TOEFL Scores, CGPA and the Chance of Admit using histogram and density plots. The GRE plot is a slightly left skewed, normally distributed curve having 2 spikes one near 312 and the other near 324 and gradually decreases after the 326th frequency. The TOEFL Scores plot is a slightly left skewed, having spikes near 105 and 111. The CGPA plot is also slightly left skewed having a spike near 7. Similarly, the chance of admit plot is a left skewed having spikes at 0.7 and 0.9.

Correlation Analysis

In statistics, the correlation is statistical association, which refers to the degree to which pairs of variables are linearly related. And the correlation coefficient is a numerical measure between two variables. The value is in the range of -1 to +1, while the coefficient is close to -1 or +1 represents a strong agreement of linear relationship and 0 represents a strong disagreement.



In predicting the outcome of the response, all variables are positively influencing each other and there are no inverse relationships. The correlation between CGPA and admit chances stand at 0.88, followed by GRE & TOEFL scores which show an 80% correlation against the same. In terms of correlations between the predictors, there's a strong correlation between TOEFL, GRE, & CGPA. An SME may argue that it is a fair expectation for a high achieving student's record also carry into standardized testing. The application is incomplete without all three parameters and the dataset is small enough, multicollinearity is not a factor for further consideration.

Linear Regression Model Analysis

We used simple linear regression model to gauge the significance of the predictors and the fitness of the model to solve the given problem. The cumulative adjusted R^2 value is about 0.82 which tells us that model is a reasonable fit. A scatter plot of admission probability against the most significant variable CGPA in Appendix A provides additional insight into the relationships. While higher Grade Point Average (CGPA) is associated with increased probability of admissions. The points are color coded points by the GRE score (below 25 percentile, 25-75%, and above

75%). Graph indicates that students with lower CGPA also perform weaker on the standardized testing in general. There is some evidence of heteroscedasticity although it is not pronounced as the points are more scattered at the lower range and dense at the upper range of predictors. However, the regression line shows evenly distributed data points and portends to a linear relationship.

Table 2: Linear Regression Results

	Intercept	GRE	TOEFL	Univ Rating	LOR	CGPA	Research
Coefficient	-1.28	0.002	0.003	0.006	0.017	0.119	0.024
p Value	< 2e-16	0.0002	0.0013	0.0694	1.45E-05	< 2e-16	0.0002
Adjusted R2	0.8197						
p Value	< 2.2e-16						

In addition to the R function lm() for simple linear regression, the team explored the Leaps package function regsubsets() using the exhaustive option to determine the best model fit for the admission dataset. Going from inclusion of one variable to all available variables, we can see the variables included for best for the specific configuration in the table below. CGPA emerges as the most is valuable predictor which must be included no matter how many variables are included in the model, corroborating the earlier finding of the importance of this predictor.

The last four columns present the standard metrics to yield a comparison between one to seven variable models. Adjusted R^2 provides the degree of how the model is able to fit the data in a multiple predictor model that we have. Higher this index better out regression model is. The Bayesian Information Criteria index (BIC) and CP are variant of Akaike's information metric that penalize inclusion of additional variables. Lower the BIC and CP, better the model is (Taylor, 2019).

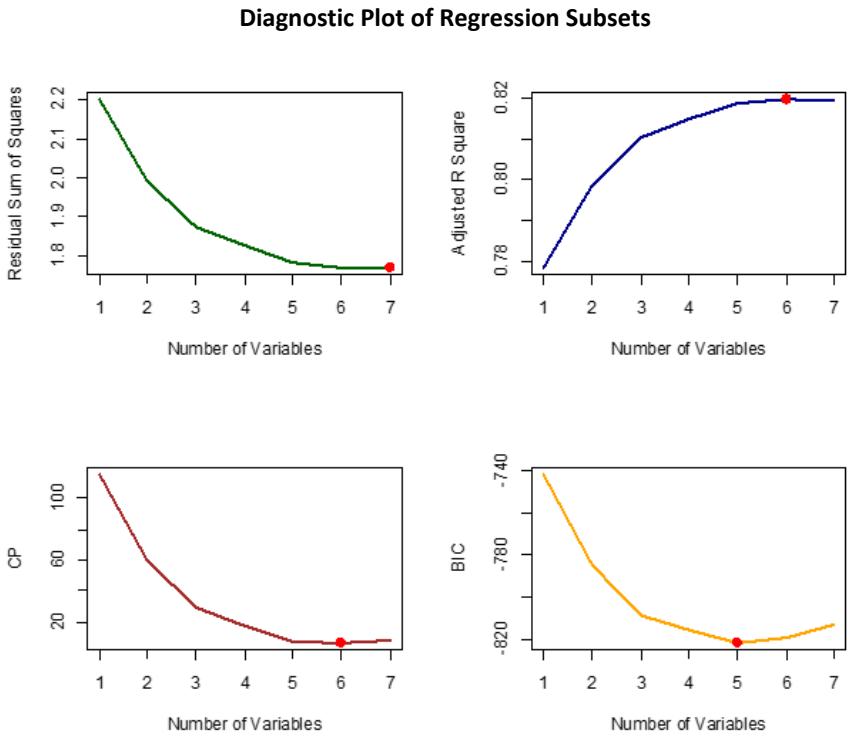
The figures in red indicate the best metrics determined by the regression model. BIC is the lowest when the variables GRE, TOEFL, LOR, CGPA, and Research are included and we note that cumulative improvements in regression metrics plateau around five/six variable models.

Table 3: Linear Regression Model Variable Metrics

Count	GRE	TOEFL	Univ Rating	SOP	LOR	CGPA	Research	Adjusted R2	CP	RSS	BIC
1						YES		0.778	115.5	2.2	-742.58
2	YES					YES		0.799	59.6	1.99	-785.47
3	YES				YES	YES		0.81	29.34	1.88	-808.89
4	YES				YES	YES	YES	0.815	17.72	1.83	-815.91
5	YES	YES			YES	YES	YES	0.819	7.43	1.78	-821.95
6	YES	YES	YES		YES	YES	YES	0.819	**6.12**	1.77	-819.08
7	YES	YES	YES	YES	YES	YES	YES	0.819	8	1.77	-812.99

Below are the diagnostic plots from the seven variable model matrix output from Leaps package subsets function discussed above. Depending on the metric chosen, we visually corroborate the data evidence where the trend line indicate that five and seven variable models are the strongest. The six variable model is consistent strong across all metrics of evaluation providing us with an inflection point.

The team also explored the Leaps library function "regsubsets" with training and cross validation as shown in the R code in Appendix B consisting of 75% training data and 25% test data with replacement. This method yielded a further refined model, reducing the mean square error from 0.0036 to 0.0032. When a variation of the regsubsets using multiple data buckets (10) with cross validation is implemented using a specialized predict routine, the results are similar to



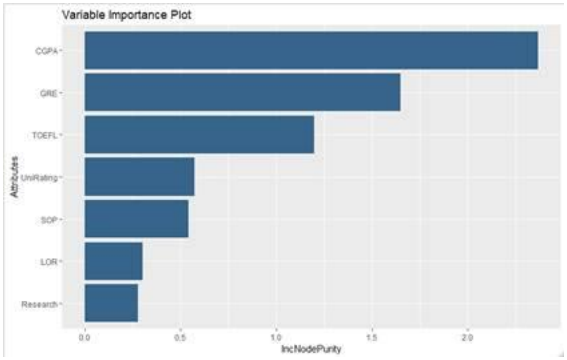
linear regression without regsubsets. Here's the RMSE table for linear regression.

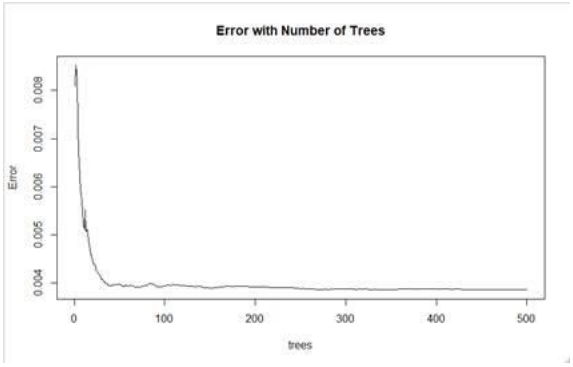
Model	Train RMSE	Test RMSE
Linear Regression (CV)	0.06244	0.05744

Regression Analysis using Random Forest Trees

Random forest is an algorithm that can be used for both classification and regression problems. The idea of this method is to combine several learning models to increase the prediction result. To explain it in simple words, Donges demonstrates that "random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction" (Donges, 2019).

For the Random Forest the data was divided into train and test set. 75% was kept in training set and rest was used as a validation set. While dividing the data set the seed was set to '123.' The model was developed with the default parameters that is ntree=500 and mtry was also kept as default. "AdmitChance" was kept as response variable and all other were kept as predictors. The following plot shows the results of the random forest. According to our model CGPA was found to be the most important variable. The plot shows the decrease in error with number of trees.





The above model achieved the following result:

Model	Train RMSE	Test RMSE
Random Forest	0.06188599	0.06331481

The results above show that the model did performed well in a sense as both train and test RMSE are close to each other which suggest that the model performed similarly on the unseen

data as well. We also tried by removing less important variables from the model however it didn't improve the performance.

Random Forest with Cross Validation

After the simple random forest, we tried implementing it with cross validation, the value k was equals to 10. The RMSE results show a little improvement however it is not much significant. This could be because the size of data was too small to make any significant impact on the RMSE values. But with these results we can conclude that Random Forest with cross validation performed better among the two.

Model	Train RMSE	Test RMSE
Random Forest (CV)	0.06181928	0.06251908

Conclusion

The final project allowed team 8 an opportunity to pose a set of relevant domain specific questions, translate them into technical questions and explore statistical analysis methods learned from the STAT 515 course in search of fitness of methods for resolution. As a result, team was able to determine the degree to which each variable correlates with the response variable and determine the right combination of variables to predict the outcome. The team leveraged the relevant R libraries and functions to fit models that are particularly suited to the admission dataset. The project helped us explore data (descriptive), build models (predictive), and aid decision making (prescriptive) simultaneously.

The analysis was somewhat limited by the source dataset in terms of attributes included and data collected by its authors. It nevertheless afforded an excellent opportunity multiple regression models and identify the best fit for a problem that should resonate with majority of the current graduate student community. In comparing the model results from multiple independent methods, we found convergence to ensure high confidence in the information shared, rooted in the most suitable statistical methods. The project symbolized a positive reinforcement that power of data analytic tools and teamwork is valuable in augmenting productivity of the domain experts and the organization as a whole, which is essential to the success of Data Analytics practitioners.

References

Donges, N. (2019, May 02). *The Random Forest Algorithm.* Retrieved from Machinelearning-Blog.Com: https://machinelearning-blog.com/2018/02/06/the-random-forest-algorithm/

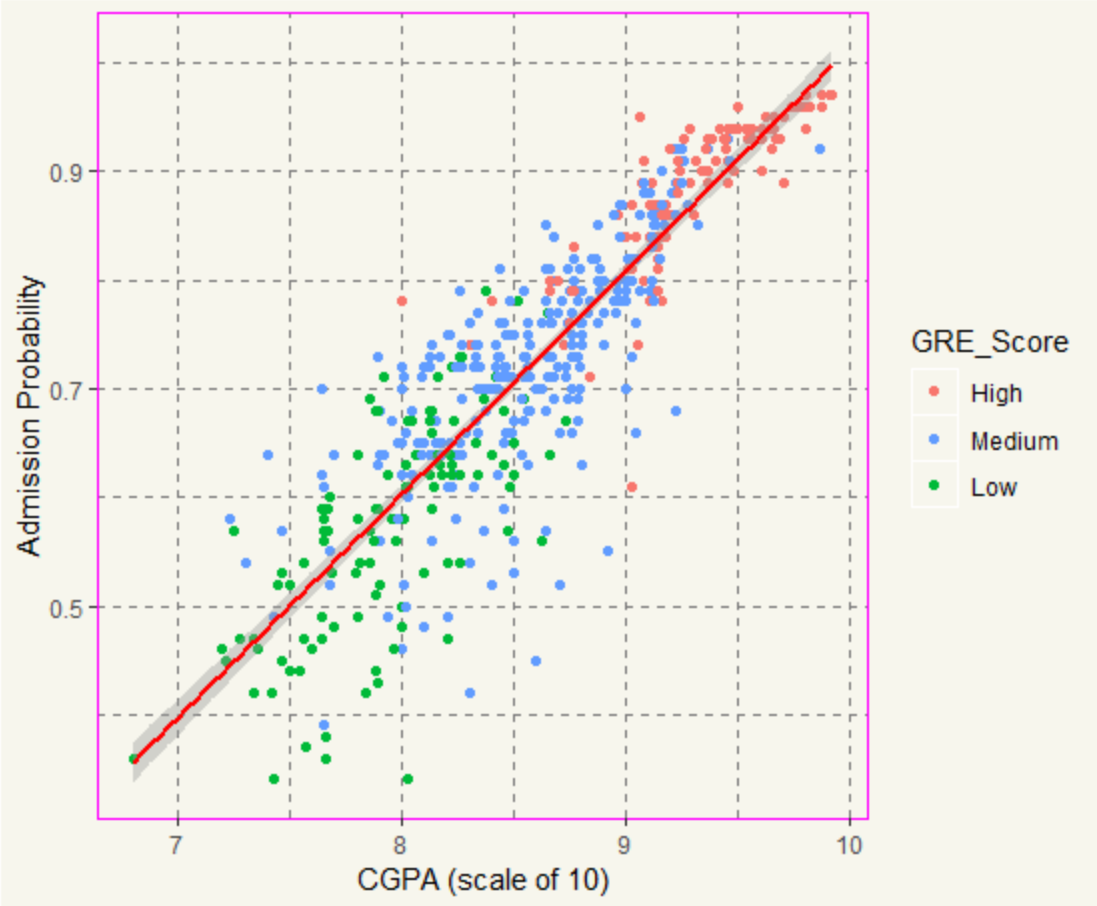
Lander, J. P. (2017). *R for Everyone: Advanced Analytics and Graphics, (2nd ed.).* Addison Wesley Data & Analytics Series.

Mohan S Acharya, A. A. (2019). A Comparison of Regression Models for Prediction of Graduate Admissions. *IEEE International Conference on Computational Intelligence in Data Science*.

Pickle, Carr, D. B., & L.W. (2010). *Visualizing Data Patterns with Micromaps, CRC Press.*

Taylor, J. (2019). *Statistics 203: Introduction to Regression and Analysis of Variance Model Selection: General Techniques.* Retrieved from Stanford: http://statweb.stanford.edu/~jtaylo/courses/stats203/notes/selection.pdf

APPENDIX A – CGPA vs Admission Outcome Scatter Plot & Regression Line



APPENDIX B – R CODE

```
# Library------------------------------------------------

library(readr) #Load csv data files
library(corrplot) #Run correlation plots
library(leaps) # Regression Subsets
library(randomForest) # Regression with Random Forest Trees
library(ggplot2) #Plotting output from Random Forest
library(tidyverse) #mutate function


# Data Loading and preparing -----------------------------

Admission_Predict_Ver1_1 <- read_csv("Admission_Predict_Ver1.1.csv")

data <- Admission_Predict_Ver1_1
originaldata <- data

rm(Admission_Predict_Ver1_1)


colnames(data)[colnames(data)=="GRE Score"] <- "GRE"
colnames(data)[colnames(data)=="TOEFL Score"] <- "TOEFL"
colnames(data)[colnames(data)=="University Rating"] <- "UniRating"
colnames(data)[colnames(data)=="Chance of Admit"] <- "AdmitChance"

data$`Serial No.` <- NULL

# Data Loading and preparing -----------------------------

# Plot Theme
plot_theme <- function(base_size = 12, base_family = "Helvetica") {
  theme(
    plot.background = element_rect(fill = "#F7F6ED"),
    legend.key = element_rect(fill = "#F7F6ED"),
    legend.background = element_rect(fill = "#F7F6ED"),
    panel.background = element_rect(fill = "#F7F6ED"),
    panel.border = element_rect(colour = "magenta", fill = NA, linetype = "solid"),
    panel.grid.minor = element_line(colour = "#7F7F7F", linetype = "dashed"),
    panel.grid.major = element_line(colour = "#7F7F7F", linetype = "dashed")
  )
}
```

```
#Histogram and Density plots

#GRE Scores
a <- ggplot(data,aes(x=GRE,..density..)) +
  geom_histogram(binwidth=2, color="black", aes(fill=..count..)) + scale_fill_gradient("Count",
low="cyan", high="blue") +
  labs(x="GRE Scores",y="Counts",title="Distribution for GRE Score", fill= FALSE) +
geom_line(stat="density", color="red",size=1.2)
a + guides(fill=FALSE)

#TOEFL Scores
a <- ggplot(data,aes(x=TOEFL,..density..)) +
  geom_histogram(binwidth=2, color="black", aes(fill=..count..)) + scale_fill_gradient("Count",
low="cyan", high="blue") +
  labs(x="TOEFL Scores",y="Counts",title=" Distribution for TOEFL Score") +
geom_line(stat="density", color="red",size=1.2)
a + guides(fill=FALSE)

#CGPA
a <- ggplot(data,aes(x=CGPA,..density..)) +
  geom_histogram(binwidth=0.4, color="black", aes(fill=..count..)) +
scale_fill_gradient("Count", low="cyan", high="blue") +
  labs(x="CGPA",y="Counts",title="Distribution for CGPA ",fill=FALSE) +
geom_line(stat="density", color="red",size=1.2)
a + guides(fill=FALSE)

#Chance of Admit
a <- ggplot(data,aes(x=AdmitChance,..density..)) +
  geom_histogram(binwidth=0.07, color="black", aes(fill=..count..)) +
scale_fill_gradient("Count", low="cyan", high="blue") +
  labs(x="Chance of Admit",y="Counts",title="Distribution for Chance of Admit ", fill= FALSE)
+ geom_line(stat="density", color="red",size=1.2)
a + guides(fill=FALSE)

# CORRELATION PLOT ----------------------------------------

#form the model matrix and correlation variable
varMat <- model.matrix(~.-Research,data=data)[,-1]     #the first column is all 1's, remove this.
varCor <- cor(varMat)

#plot the correlation
corrplot(varCor,method = "circle",
        tl.col = "black", mar = c(0,0,2,0),
        title = "Graduate Admission numerical Variables Correlation")
```

```
corrplot(varCor,add = TRUE,                    # add the above plot
        type = "lower", method = "number",number.font = 2,
        number.cex = .75,col = "black",
        diag = FALSE,tl.pos = "n", cl.pos = "n")

rm(varMat, varCor)


########## SCATTER PLOTS for best indicators in Correlation ####

scatterdata <- tbl_df(data)
scatterdata <- mutate(scatterdata, GRE_Score = ifelse(GRE>325, "High", ifelse(GRE<308,
"Low", "Medium")))

ggplot(scatterdata, aes(x = CGPA, y = AdmitChance, color = GRE_Score)) +
  geom_point(size = 2) + labs(x="CGPA (scale of 10)", y = "Admission Probability") +
  geom_smooth(method = "lm", color = "red") +
scale_color_discrete(breaks=c("High","Medium","Low"))

rm(scatterdata)

#Clear linear relationship visible between CGPA and admission probability

###################### LINEAR REGRESSION ########################

str(data)

#Check if there are any rows with missing values
anyNA(data)

#scaling function
#normalize <- function(x) {
#  return ((x - min(x)) / (max(x) - min(x)))
#}

head(data)
regdata <- as.data.frame(data)
#Scaling - not yielding the results - stick with as is
#regdata <- as.data.frame( normalize(data[1:2] ))
#regdata <- cbind(regdata, data[3:7])

regdata$GRE <- as.integer(regdata$GRE)
regdata$TOEFL <- as.integer(regdata$TOEFL)
#regdata$UniRating <- as.factor(regdata$UniRating)
regdata$Research <- as.logical(regdata$Research)
```

```
head(regdata)

summary(regdata)


#contrasts(regdata$UniRating)

#Fit a model using all predictors
linear.fit=lm(AdmitChance ~.,data=regdata)
summary(linear.fit)

# SOP is not a reliable indicator with a p value of 0.73, Unirating is 0.12

#Fit a model using all predictors but SOP
linear.fit=lm(AdmitChance ~ .-SOP,data=regdata)
summary(linear.fit)
# All predictors are significant now exception of UniRating at 0.06

#linear.fit=lm(AdmitChance ~ .-SOP -UniRating,data=regdata)
#summary(linear.fit)

# How do Residuals look?
ggplot(linear.fit, aes(x = .fitted, y = .resid)) +
  geom_point(color = "red", size = 1) + geom_hline(yintercept = 0) +
  plot_theme() +
  ylab("Residuals") +
  xlab("Fitted Values")

#Use subsets and run the model
regfit.full = regsubsets(AdmitChance ~., regdata)
reg.summary <- summary(regfit.full)
reg.summary

cor(reg.summary$outmat)
round(coef(regfit.full,loc), 3)
names(reg.summary)

adjr2 <- reg.summary$adjr2 #best with all indicators in place except SOP, 5 var => 0.819, 6 var
=> 0.820 marginal diff
cp <- reg.summary$cp #lowest with all predictors in place except SOP
rss <- reg.summary$rss #lowest with all predictors except SOP in place
bic <- reg.summary$bic #lowest with 5 predictors, SOP and UniRating not imporving BIC

#Diagnostic Plots
```

```
subsets_data <- data.frame(adjr2, cp, rss, bic, vars)
colnames(subsets_data) <- c("Adjusted R Square", "CP", "Residual Sum of Squares", "BIC")

ggplot(subsets_data, aes(x = vars, y = adjr2)) +
  geom_line(color = "blue") +
  geom_point(color = "red", size = 3) +
  plot_theme() +
  scale_x_discrete(limits = c(1:7)) +
  ylab("Adjusted R Square") +
  xlab("Number of Variables")

ggplot(subsets_data, aes(x = vars, y = rss)) +
  geom_line(color = "blue") +
  geom_point(color = "red", size = 3) +
  plot_theme() +
  scale_x_discrete(limits = c(1:7)) +
  ylab("Residual Sum of Squares") +
  xlab("Number of Variables")

ggplot(subsets_data, aes(x = vars, y = bic)) +
  geom_line(color = "blue") +
  geom_point(color = "red", size = 3) +
  plot_theme() +
  scale_x_discrete(limits = c(1:7)) +
  ylab("BIC") +
  xlab("Number of Variables")

ggplot(subsets_data, aes(x = vars, y = cp)) +
  geom_line(color = "blue") +
  geom_point(color = "red", size = 3) +
  plot_theme() +
  scale_x_discrete(limits = c(1:7)) +
  ylab("CP") +
  xlab("Number of Variables")


# ggplot yields better plots than Plot function
# windows()
# par(mfrow = c(2,2))
#
# xlab = "Number of Variables"
# # 1st row 1st  column
# plot(reg.summary$rss,xlab = xlab, ylab = "Residual Sum of Squares", col= "dark green", lwd =
2, type = "l") +
# loc <- which.min(reg.summary$rss)
```

```
# loc
# points(loc,reg.summary$rss[loc], col = "red",cex = 2,pch = 20)
#
#
# # 1st row 2nd  column
# plot(reg.summary$adjr2,xlab = xlab, ylab = "Adjusted R Square", col= "dark blue", lwd = 2,
type = "l")
# loc <- which.max(reg.summary$adjr2)
# loc
# points(loc,reg.summary$adjr2[loc], col = "red",cex = 2,pch = 20)
#
# # 2nd row 1st column
# plot(reg.summary$cp,xlab = xlab, ylab = "CP", col= "brown", lwd = 2, type = 'l')
# loc <- which.min(reg.summary$cp)
# loc
# points(loc,reg.summary$cp[loc], col = "red",cex = 2,pch = 20)
#
# # 2nd row 2nd column
# plot(reg.summary$bic,xlab = xlab, ylab = "BIC", col= "orange", lwd = 2, type = 'l')
# loc <-  which.min(reg.summary$bic)
# loc
# points(loc,reg.summary$bic[loc], col = "red",cex = 2,pch = 20)
#
#
# dev.off()

## What is the mean square error (base case)?
mse <- round(reg.summary$rss[6]/nrow(regdata), 4)
mse #0.0035 for 5, 6, 7 variable model, lets see if train/test technique improves the result
substantially
rmse <- round(sqrt(mse), 2)
rmse #0.06


######## Linear Regression with subset selection methods ########
### Base case no folds ###
?regsubsets

set.seed(123)
train = sample(c(TRUE,FALSE), nrow(regdata)*0.75, replace = TRUE)

test = !train

#regdata[train, ]
```

```
# Find the best training set models
regfit.best = regsubsets(AdmitChance~., data = regdata[train,], nvmax = 7)
coef(regfit.best, 5)

# Obtain the test set design matrix
test.mat = model.matrix(AdmitChance~., data = regdata[test,])

train.mat = model.matrix(AdmitChance~., data = regdata[train,])
mean(regfit.best$rss -regfit.best$ress)

regdata[test.mat]

head(regdata[train, 8])

#regfit.best$adjr2 #best with all indicators in place except SOP, 5 var => 0.819, 6 var => 0.820 marginal diff
#regfit.best$cp #lowest with all predictors in place except SOP
#regfit.best$rss #lowest with all predictors except SOP in place
#regfit.best$bic #lowest with 5 predictors, SOP and UniRating not imporving BIC

# Vector for errors
val.errors = rep(NA,7)
training.errors = rep(NA, 7)

# Run for each number of variables in the model
for (i in 1:7) {
  # obtain the training set coefficients
  coefi = coef(regfit.best,id = i)

  # predict test set values
  pred = test.mat[,names(coefi)] %*% coefi

  pred2 = train.mat[,names(coefi)] %*% coefi

  # Obtain the MSE
  val.errors[i] = mean((regdata$AdmitChance[test] - pred)^2)
  training.errors[i] = mean((regdata$AdmitChance[train] - pred2)^2)
}

round(val.errors, 4)
round(training.errors, 4)

## test and training mse 0.0033 vs 0.0039 for 6 var model

### Regsubsets with 10 Folds ###
```

```
k <- 10
set.seed(123)

#define the predict function to work on each fold

predict.regsubsets =
  function(object,newdata,id,...){
    form = as.formula(object$call[[2]])
    mat = model.matrix(form,newdata)
    coefi = coef(object,id = id)
    xvars = names(coefi)
    mat[,xvars] %*% coefi
  }

folds <- sample(1:k,nrow(regdata),  replace = TRUE)

cv.errors = matrix(NA,nrow = k,ncol = 7, dimnames = list(NULL, paste(1:7)))

# The fold and number of variables loops
for (j in 1:k) { # fold loop

  # The 7 best models with jth fold omitted
  bestfit.fold = regsubsets(AdmitChance ~., data = regdata[folds != j,],nvmax = 7)

  # The MSE for the fold prediction error
  for (i in 1:7) {# number of variable loop
    pred = predict.regsubsets(bestfit.fold, regdata[folds == j,],id = i)
    cv.errors[j,i] = mean((regdata$AdmitChance[folds == j] - pred)^2)
  }
}

# Find the mean across the fold MSE for each model
mean.cv.errors = apply(cv.errors,2,mean)
round(mean.cv.errors, 4) # 0.0037 for 5 to 7 var models
#rmse = 0.06 again
# base case performs better

par(mfrow = c(1,1))
plot(mean.cv.errors,type = 'b')
which.min(mean.cv.errors)

##############  Random Forest  ###############
#  Define Train and Test

# Train and Test ----------------------------------------
```

```
set.seed(123)
num <- sample(1:500, nrow(data)*0.75, replace = FALSE)

train <- data[num,]
test <- data[-num,]

rm(num)

# Train and Test ----------------------------------------

temp <- test$AdmitChance
temp <- as.data.frame(temp)

# Random Forest -----------------------------------------

rf <- randomForest(AdmitChance ~., data = train) # with all the variables

varImpPlot(rf, main = 'Model Importance Plot')

impplot <- rf$importance


impplot <- as.data.frame(impplot)

impplot$Attribute <- rownames(impplot)

#mesh2$cat2 <- order(mesh2$Category, mesh2$Count, decreasing=TRUE)

p <- ggplot(data = impplot, aes(reorder(Attribute, IncNodePurity), IncNodePurity)) +
geom_col(mapping = NULL, data = NULL, position = "stack",
                                          width = NULL, na.rm = FALSE, show.legend = NA,
                                          inherit.aes = TRUE,fill = 'steelblue4') + plot_theme()
p + coord_flip() + xlab("Attributes") + labs(title = "Variable Importance Plot")

rm(p,impplot)

plot(rf, main = "Error with Number of Trees")

pretest <- predict(rf,test[-8])

temp$rf <- pretest
```

```
rf2 <- randomForest(AdmitChance ~ CGPA+UniRating+SOP+LOR+Research, data = train) # with few variables

pretest <- predict(rf2,test[c(-1,-2,-8)])

temp$rf2 <- pretest


rf3 <- randomForest(AdmitChance ~GRE+TOEFL+UniRating+SOP+LOR+CGPA, data = train) # without research based imp plot

pretest <- predict(rf3,test[c(-7,-8)])

temp$rf3 <- pretest

rm(pretest)


# RMSE ----------------------------------------------------

library(ModelMetrics)

rmse(temp$temp,temp$rf) # with all variable = 0.03700746


rmse(temp$temp,temp$rf2)# with all variable = 0.05727096


rmse(temp$temp,temp$rf3) # without Research  = 0.03746171

# Based on the RMSE value model with all the predictors and without Research performed best

# RMSE ----------------------------------------------------


# RANDOM FOREST WITH CV ------------------------------------------------


k = 10

fold = 1:10

datacv <- data

datacv$kfold <- sample(1:k, nrow(datacv), replace = TRUE)
```

```
length(which(datacv$kfold == 9))

prediction <- data.frame()
test_sets <- data.frame()

train_sets <- data.frame()
train_pred <- data.frame()

for(n in 1:k){
  ###Grab all the rows with the id 'n', aggregate them into a test set
  test_set = datacv[which(datacv$kfold %in% n), -ncol(datacv)]

  ###All the other rows (the other 9 parts) go in the training set
  train_set = datacv[which(datacv$kfold %in% fold[-c(n)]), -ncol(datacv)]

  forest = randomForest(AdmitChance ~., data = train_set, importance = TRUE,   ntree = 500)

  ###Run the model on the test set, save the prediction in a dataframe, the test set in another.
Then you can compare them to get your performance measure.
  n_predict = data.frame(predict(forest, test_set))
  prediction = rbind(prediction, n_predict)
  test_sets = rbind(test_sets, as.data.frame(test_set))

  train_sets = rbind(train_sets, train_set)
  train_pred = rbind(train_pred, as.data.frame(forest$predicted))
}

test_sets$

library(ModelMetrics)


a <- rmse(prediction$predict.forest..test_set., test_sets$AdmitChance) # = 0.062

varImpPlot(forest)

b <- rmse(train_pred$`forest$predicted`, train_sets$AdmitChance) # = 0.062

rm(a,b)



# RANDOM FOREST WITH CV ----------------------------------------------
```