**Part 1**

1. Linear regression:

Linear Regression is about quantifying the relationship between

two numerical variables, as well as modeling numerical

response variables using a numerical or categorical explanatory

variable.

1. Correlation:

Correlation describes the strength of the \*linear \* association between two

variables. It takes values between -1 (perfect negative) and +1 (perfect positive).

A value of 0 indicates no linear association.

cor(smdetails$Metropolitan.Residence, smdetails$SingleFemaleHousehold)

1. Function of regression line:

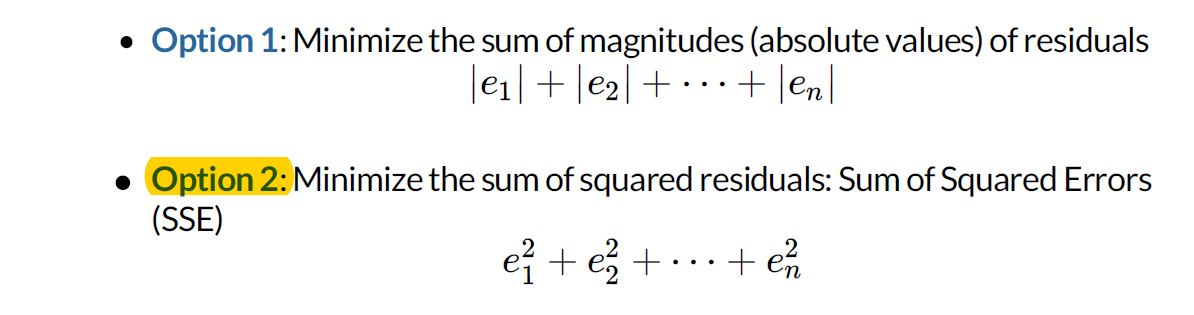
Y = mx+c

1. Residuals:

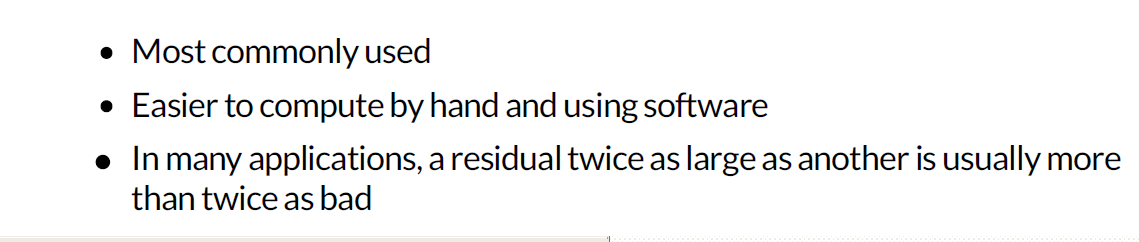
Residual is the difference between the observed ( ) and predicted.

1. A measure for the best line:

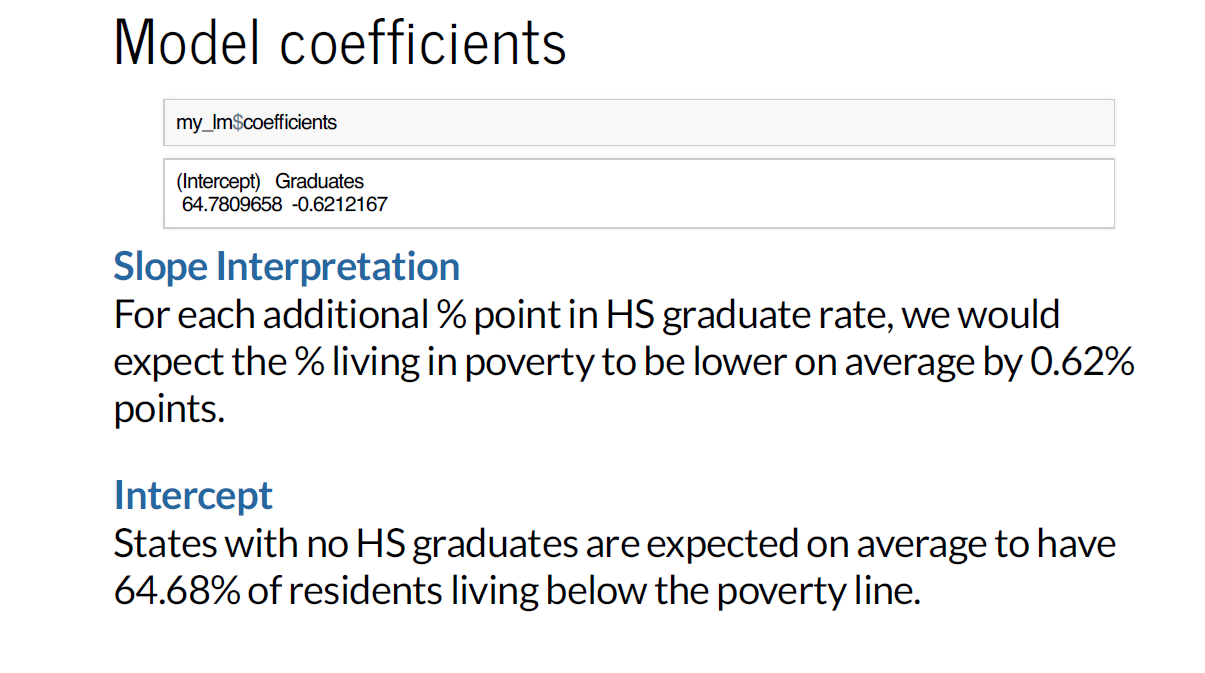
We want a line that has small residuals.



1. Why least squares?



1. Slope and intercept interpretation:



1. Prediction:

Using the linear model to predict the value of the response variable for a given value of the explanatory variable is called prediction, simply by plugging in the value of in the linear

model equation.

1. R-squared:

It tells us what percent of variability in the response variable is explained by the model. It is calculated as the square of the correlation coefficient.

**Assignment 6**

1. Load the file:

smdetails = read.csv('data\_poverty.txt', sep = '\t')

smdetails

1. Pick two variables that we didn't study:

I am picking the variables Metropolitan.Residence and SingleFemaleHousehold.

1. Check correlation:

cor(smdetails$Metropolitan.Residence, smdetails$SingleFemaleHousehold)

# since correlation is not zero, there is linear association between 2 variables Metropolitan.Residence and SingleFemaleHousehold.

1. Plot as scatter-plot for the two variables:

require(lattice)

xyplot(SingleFemaleHousehold~Metropolitan.Residence, data = smdetails)

1. Create the regression model for the variables:

smmodel = lm(SingleFemaleHousehold~Metropolitan.Residence, data = smdetails)

smmodel

1. Write the formula of the model and interpret the model:

SingleFemaleHousehold = 8.28954 + 0.04628\*Metropolitan.Residence

m = slope = 0.04628,

y = SingleFemaleHousehold,

c = 8.28954,

x = Metropolitan.Residence,

y = mx+c

# For each additional % point in Metropolitan.Residence rate, we would expect the % SingleFemaleHousehold to be higher # on average by 0.62%. States with no Metropolitan.Residence are expected on average to have 8.28954% of SingleFemaleHousehold.

1. Perform one prediction with the model:

Prediction for 65% Metropolitan.Residence

8.28954 + 0.04628\*65

# If 65% population of a state live in metropolitan then single females would be 11.29% of that state pouplation.

1. Check the significance of the dependent variable (as in how confident we are that the relationship that shows in the data is not from sampling error):

anova(smmodel)

since the p value is below 5%, therefore, we are confident that the prediction will have very less error

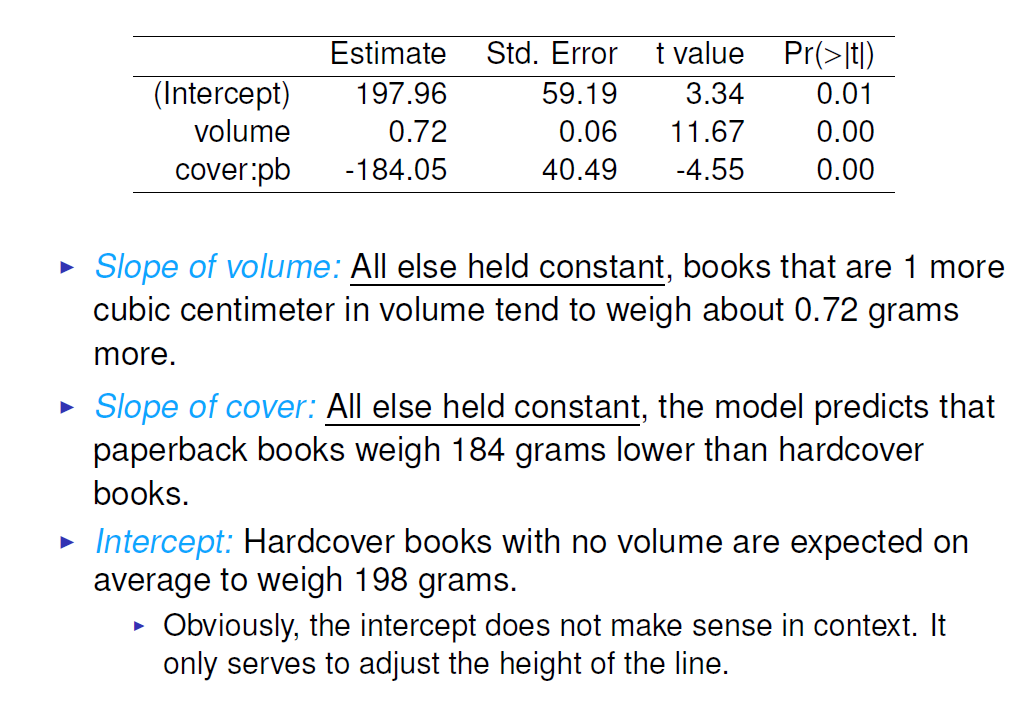
**Part 2**

1. Multiple regression:

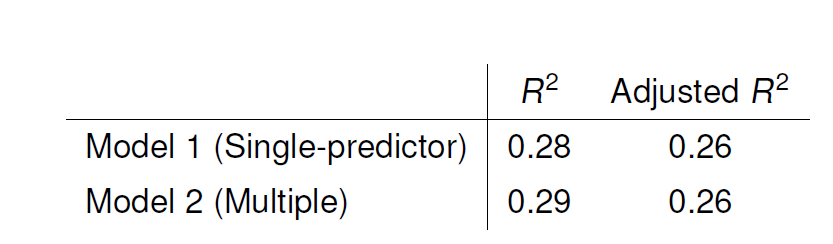
my\_lm=lm(Poverty˜Graduates+Metropolitan.Residence, data=poverty)

1. Interpretation of the regression coefficients in Multiple regression:

weight = 197.96 + 0.72 volume − 184.05 cover : pb



1. R-squared for multiple regression:
2. R-squared vs adjusted R-squared:



When any variable is added to the model R2 increases.

But if the added variable doesn’t really provide any new

information, or is completely unrelated, adjusted R2 does not

increase.

**Assignment 7 – Linear regression**

1. Plot a diagram to show the relationship between ‘runs’ and ‘at\_bats’. Consider ‘at\_bats’ as the explanatory variable.

bbteamdetails = read.delim('baseball.CSV', sep = ',')

bbteamdetails

plot(runs~at\_bats, data = bbteamdetails)

or

require(lattice)

xyplot(runs~at\_bats, data= baseball)

1. Can you quantify this relationship ?

cor(bbteamdetails$runs,bbteamdetails$at\_bats)

# We can quantify the relation between these 2 variables with correlation.since correlation is not zero, there is linear association between 2 variables.

1. A more efficient way to find the min Sum of Squares is to use the lm f unction in R to fit the linear model (a.k.a. regression line).

m1 <- lm ( runs ~ at\_bats , data = baseball )

Here it can be read that we want to make a linear model of runs as a function of at\_bats. The second argument specifies that R should look in the baseball data frame to find the runs and at\_bats variables.

summary(m1) contains all of the information we need about the linear model that was just fit

What is the linear function that describes the relationship (Hint: look at the coefficients in the summary)?

runs = at\_bats\*0.6305-2789.2429

1. Fit another model that uses homeruns to predict runs . Using the estimates from the R output, write the equation of the regression line.

plot(runs~homeruns, data = bbteamdetails)

runswhenhomeruns = lm( runs~homeruns , data = bbteamdetails )

runswhenhomeruns

summary(runswhenhomeruns)

# runs = 1.835\*homeruns + 415.239

# SLope interpretation - For each additional homerun, we would expect the win to be higher on average by 48.81 times. With no homeruns also, wins are expected to be 48.81 times.

1. Create a scatter plot using the following commands:

plot(bbteamdetails$runs ~ bbteamdetails$at\_bats )

abline(runswhenatbats)

If the prediction is done for a value of x that is outside the range of the dataset,the process is called extrapolation. The problem is that the prediction requires linear association and we can not be certain that the relationship between X and Y will be linear outside of the range that is observered..

# The abline can help you predict y at any value x.

# Alteranate for abline(runswhenatbats) is abline(-2789.2429,0.6305) or abline(lm(bbteamdetails$runs~bbteamdetails$at\_bats))

1. If a team manager saw the least squares regression line and not the actual data, how many runs would he or she predict for a team with 5,579 at\_bats ? Is this an overestimate or an underestimate, and by how much? In other words, what is the residual for this prediction?

runs = at\_bats\* 0.6305 - -2789.2429

(5579\*0.6305) - 2789.2429

residuals(runswhenatbats)

# This is an over estimate by 15.59

bbteamdetails$predicted = predict(runswhenatbats) # Saves the predicted values

bbteamdetails$residuals = residuals(runswhenatbats) # Saves the residual values

require(dplyr)

bbteamdetails %>% select(at\_bats, runs, predicted, residuals)

1. Choose another traditional variable from baseball daga that you think might be a good predictor of runs . Produce a scatterplot of the two variables and fit a linear model. At a glance, does there seem to be a linear relationship?

plot(runs~hits, data = bbteamdetails)

cor(bbteamdetails$runs,bbteamdetails$hits)

abline(lm( runs~hits, data= bbteamdetails))

# since correlation is not zero, there is linear association between 2 variables.

runswhenhits = lm( runs ~ hits , data = bbteamdetails )

runswhenhits

# runs = 0.7589\*hits - 375.5600

1. How does this relationship compare to the relationship between runs and at\_bats ? Use the R2 values from the two model summaries to compare. Does your variable seem to predict runs better than at\_bats ? How can you tell?

summary(runswhenhits)$r.squared

summary(runswhenatbats)$r.squared

yes the hits variable predicts runs better than at\_bats because 64.1 percent of variability in the response

variable is explained by runs~hits model. we analysed using R squared method.

**Assignment 7 – Multiple regression**

1. Describe the distribution of score . Is the distribution skewed? What does that tell you about how students rate courses? Is this what you expected to see? Why, or why not?

studenteval = read.delim('evals.CSV', sep = ',')

studenteval

hist(studenteval$score)

mean.displ=mean(studenteval$score)

median.displ=median(studenteval$score)

skewness\_displ=3\*(mean.displ-median.displ)/sd(studenteval$score)

skewness\_displ

# It is negatively skewed.Majority of the students rated morethan 4 for the professors.

1. Excluding score , select two other variables and describe their relationship using an appropriate

visualization (scatterplot or side-by-side boxplots).

plot(cls\_did\_eval~cls\_students, data = studenteval)

cor(studenteval$cls\_did\_eval,studenteval$cls\_students)

# since correlation is not zero, there is linear association between 2 variables.

1. The data set contains several variables on the beauty score of the professor: individual ratings from each of the six students who were asked to score the physical appearance of the professors and the average of these six scores. Let’s take a quick look at the relationship between one of these scores and the average beauty score.

plot( studenteval$bty\_avg ~ studenteval$bty\_f1lower )

cor ( studenteval$bty\_avg,studenteval$bty\_f1lower )

As expected the relationship is quite strong – after all, the average score is calculated using the individual scores. We can actually take a look at the relationships between all beauty variables (columns 13 through 19) using the following command:

plot(studenteval[, 13 : 19 ]).

These variables are collinear (correlated), and adding more than one of these variables to the model would not add much value to the model. In this application and with these highly-correlated predictors, it is reasonable to use the average beauty score as a single representative of these variables. In order to see if beauty is still a significant predictor of professor score after we’ve accounted for the gender of the professor, we can add the gender term into the model.

m\_bty\_gen = lm ( score ~ bty\_avg + gender , data = studenteval )

summary ( m\_bty\_gen )

Is bty\_avg still a significant predictor of score ? Has the addition of gender to the model changed the parameter estimate for bty\_avg ?

yes bty\_avg is a significant predictor since it has 3 stars. Addition of gender has changed the paramter for #estimate as we need to add 0.17239 for the score if the gender of the professor is male. Also gender predictor has 3 #starts and it is significant.

1. What is the equation of the line corresponding to males? (Hint: For males, the parameter estimate is multiplied by 1.) For two professors who received the same beauty rating, which gender tends to have the higher course evaluation score?

For two professors who received the same beauty rating, male tend to have higher score as the equation for male is #score = 3.74 + bty\_avg\*0.07 + 0.17239. But for female, the eqaution for score = 3.74 + bty\_avg\*0.07

1. Create a new model called m\_bty\_rank with gender removed and rank added in. How does R appear to handle categorical variables that have more than two levels? Note that the rank variable has three levels: teaching, tenure track, tenured.

m\_bty\_rank = lm ( score ~ bty\_avg + rank , data = studenteval )

summary ( m\_bty\_rank )

If the professor rank is teaching then the equation is score = 3.98 + bty\_avg\*0.067. However, if the professor is #tenure track then score = 3.98 + bty\_avg\*0.067 - 0.16 and for tenured professor, the score = 3.98 + bty\_avg\*0.067 - #0.12.

linear modelling(score ~ bty\_avg + rank) shows how much higher a professor can score if they have a beauty #rating #that is one point higher while holding all other variables constant.

1. We will start with a full model that predicts professor score based on rank, ethnicity, gender, language of the university where they got their degree, age, proportion of students that filled out evaluations, class size, course level, number of professors, number of credits, average beauty rating, outfit, and picture color. Which variable would you expect to have the highest p-value in this model? Why? Hint: Think about which variable would you expect to not have any association with the professor score. Let’s run the model:

m\_full = lm ( score ~ rank + ethnicity + gender + language + age + cls\_perc\_eval + cls\_students + cls\_level + cls\_profs + cls\_credits + bty\_avg + pic\_outfit + pic\_color , data = studenteval )

summary (m\_full )

I expect color of outfit of professor in picture(i.e, pic\_outfitnot) variable to not have any association with the #professor score.The reason being no student evaluates professor based on the outfit in the picture.

1. Check your suspicions from the previous exercise. Include the model output in your response.

I was wrong. In reality, cls\_profs has the least association to "scores" as it has highest p-value.

1. Drop the variable with the highest p-value and refit the model. Did the coefficients and significance of the other explanatory variables change? (One of the things that makes multiple regression interesting is that coefficient estimates depend on the other variables that are included in the model.) If not, what does this say about whether or not the dropped variable was collinear with the other explanatory variables?

Since cls\_profs has highest p-value, i am dropping that.

m\_full1 = lm ( score ~ rank + ethnicity + gender + language + age + cls\_perc\_eval + cls\_students + cls\_level + cls\_credits + bty\_avg + pic\_outfit + pic\_color , data = studenteval )

summary (m\_full1 )

# There was minute change in the coefficients and significance of the other explanatory variables after dropping #cls\_profs.

1. Using backward-selection and p-value as the selection criterion, determine the best model. You do not need to show all steps in your answer, just the output for the final model. Also, write out the linear model for predicting score based on the final model you settle on.?

Dropping the variable with the highest p-value and refit the model is backward selection using p-value.

m\_full2 <- lm(score ~ ethnicity + gender + language + age + cls\_perc\_eval +

cls\_credits + bty\_avg + pic\_color, data = studenteval)

summary(m\_full2)

#Score = 3.771922 + ethnicitynot minority\* 0.167872 + gendermale\*0.207112 - languagenon-english\*0.206178 - #age\*0.006046 + cls\_perc\_eval\* 0.004656 + cls\_creditsone credit\*0.505306 + bty\_avg\*0.051069 - pic\_colorcolor\*0.190579

1. Verify that the conditions for this model are reasonable using diagnostic plots.

plot(m\_full2)

# A - Residuals vs Fitted

# The dotted line at y=0 indicates our fit line.Any point on fit line obviously has zero residual. Points above have positive residuals and points below have negative residuals.

# B - Normal Q-Q Plot

# The Normal Q-Q plot is used to check if our residuals follow Normal distribution or not.

#The residuals are normally distributed if the points follow the dotted line closely

# Since the residual points follow the dotted line closely so our model residuals have passed the test of Normality.

# C- Scale - Location Plot

# A horizontal red line is ideal and would indicate that residuals have uniform variance across the range. Since ours #is close to horizontal, i think its ideal.

**Part 3**

1. k-nearest neighbor classifer:

There are many methods to perform classification prediction.

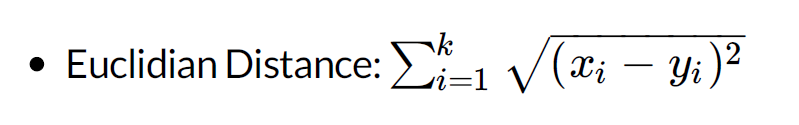
One of them is called k-nearest neighbor classifier.

1. knn: how it works:

knn trains all samples and classifies new instances based on a

similarity (distance) measure. For example, the similarity

measure can be formulated as follows:



1. Install classification package for knn:

install.packages('class')

require("class")

1. Advantages of Knn:

It is nonparametric, which means that you do not have to

make the assumption of data distribution (still have to scale

the data).

1. Disadvantages of Knn:

--It is hard to interpret the classified result.

--It is an expensive computation for a large dataset.

--The performance relies on the number of dimensions.

Therefore, for a high dimension problem, you should reduce

the dimension to increase the process performance.

1. Steps in Knn:

Step1 : select your variables and normalize the data

require(class)

Loandata = read.csv('ClassifyRisk\_historical.csv')

require(ggplot2)

normalize<-function(x){

(x-min(x))/(max(x)-min(x))

}

input=Loandata[,c('age','income','loans')]

input.norm<-sapply(input, normalize)

Step 2: Split the data for test

label<-Loandata$risk

set.seed(1234)

indicies=sample(1:2,length(Loandata$risk), replace = T, prob=c(.8,.2))

indicies

training.input=input.norm[indicies==1, ]

training.input

training.label=label[indicies==1]

training.label

testing.input=input.norm[indicies==2,]

testing.input

testing.label=label[indicies==2]

testing.label

Step 3: Knn application

require(class)

set.seed(1234)

predicted.label<-knn(train=training.input, cl=training.label, test=testing.input, k=4)

predicted.label

data.frame(predicted.label, testing.label)

sum(predicted.label==testing.label)/length(testing.label)

prop.table(table(Loandata$risk))

table(predicted.label,testing.label)

accuracy=sum(predicted.label==testing.label)/length(predicted.label)

accuracy

**Assignment 8**

1. Using the historical data (ClassifyRisk\_historical.csv) create a knn classification algorithm, calculate and report the accurarcy of the values. Only use the numerical values in your model first.

require(class)

Loandata = read.csv('ClassifyRisk\_historical.csv')

require(ggplot2)

normalize<-function(x){

(x-min(x))/(max(x)-min(x))

}

input=Loandata[,c('age','income','loans')]

input.norm<-sapply(input, normalize)

label<-Loandata$risk

set.seed(1234)

indicies=sample(1:2,length(Loandata$risk), replace = T, prob=c(.8,.2))

indicies

training.input=input.norm[indicies==1, ]

training.input

training.label=label[indicies==1]

training.label

testing.input=input.norm[indicies==2,]

testing.input

testing.label=label[indicies==2]

testing.label

require(class)

set.seed(1234)

predicted.label<-knn(train=training.input, cl=training.label, test=testing.input, k=4)

predicted.label

data.frame(predicted.label, testing.label)

sum(predicted.label==testing.label)/length(testing.label)

prop.table(table(Loandata$risk))

table(predicted.label,testing.label)

accuracy=sum(predicted.label==testing.label)/length(predicted.label)

accuracy

1. Update your model to include categorical values.

require(class)

require(ggplot2)

normalize<-function(x){

(x-min(x))/(max(x)-min(x))

}

table(Loandata$mortgage)

class\_dummies=model.matrix( ~mortgage - 1, data=Loandata)

head(class\_dummies)

table(Loandata$marital\_status)

class\_dummies1=model.matrix( ~marital\_status -1, data=Loandata)

head(class\_dummies1)

input= subset(Loandata, select=c(age,income,loans))

input = data.frame(input,class\_dummies)

input = data.frame(input,class\_dummies1)

input

input.norm<-sapply(input, normalize)

input.norm

label<-Loandata$risk

set.seed(1234)

indicies=sample(1:2,length(Loandata$risk), replace = T, prob=c(.8,.2))

training.input=input.norm[indicies==1, ]

training.input

training.label=label[indicies==1]

training.label

testing.input=input.norm[indicies==2,]

testing.input

testing.label=label[indicies==2]

testing.label

require(class)

set.seed(1234)

predicted.label<-knn(train=training.input, cl=training.label, test=testing.input, k=1)

predicted.label

data.frame(predicted.label, testing.label)

sum(predicted.label==testing.label)/length(testing.label)

prop.table(table(Loandata$risk))

table(predicted.label,testing.label)

accuracy=sum(predicted.label==testing.label)/length(predicted.label)

accuracy

1. Using your model predict the risk type for the applications in ‘classifyrisk.csv’. Note: Keep in mind that classifyrisk.csv does not have ‘labels’. You will have to train/test the model on the historical data. You can select the optimal k value given the historical data. Then you can treat the entire historical data as your training data nd the classifyrisk data as your test dataset. You will upload your .rmd file and the updated classifyrisk.csv file with risk column populated with your predictions.

# The accuracy is 95% with numeric variables and 87% with both numeric and categorical variables. Therefore, we choose only numeric variables to predict the risk type for the applications in 'classifyrisk.csv'.

require(class)

require(ggplot2)

normalize<-function(x){

(x-min(x))/(max(x)-min(x))

}

input=Loandata[,c('age','income','loans')]

input.norm<-sapply(input, normalize)

label<-Loandata$risk

Loandata1 = read.csv('classifyrisk.csv')

testinput=Loandata1[,c('age','income','loans')]

testinput.norm<-sapply(testinput, normalize)

label1=Loandata1$risk

#set.seed(1234)

#indicies=sample(1:2,length(Loandata$risk), replace = T, prob=c(.8,.2))

training.input=input.norm

training.label=label

testing.input=testinput.norm

testing.label=label1

require(class)

predicted.label<-knn(train=training.input, cl=training.label, test=testing.input, k=4)

predicted.label

risk1 = predicted.label

classifyrisk=subset(Loandata1,select = -risk)

classifyrisk = cbind(classifyrisk, risk1)

classifyrisk

sum(is.na(classifyrisk))

write.csv(classifyrisk, 'C:/Users/chinn/Desktop/Spring 18/Data mining/Week 13/HW/newoutput/classifyrisk.csv',row.names = F)

**Part 4**

1. Decision Tree:

One attractive classification method involves the construction

of a decision tree, a collection of decision nodes, connected by

branches, extending downward from the root node until

terminating in leaf nodes.

1. Install classification package for decision tree:

Library(tree)

1. Steps for decision tree:

Step 1: normalise, split training and test data (same as knn)

#Load the package tree

library(tree)

#Load input

Loandata = read.csv('ClassifyRisk\_historical.csv')

library(ggplot2)

#Generic function created to reuse for normalizing the variables

normalize<-function(x){

(x-min(x))/(max(x)-min(x))

}

#Assign input and label

input=Loandata[,c('age','income','loans')]

input.norm<-sapply(input, normalize) # Normalize the input

label<-Loandata$risk

# Split training and test data

set.seed(1234)

indicies=sample(1:2,length(Loandata$risk), replace = T, prob=c(.8,.2))

indicies

## Data Split

train\_data=input.norm[indicies==1, ]

train\_labels=label[indicies==1]

test\_data=input.norm[indicies==2,]

test\_labels=label[indicies==2]

test\_data<-data.frame(test\_data, test\_labels)

head(test\_data)

train\_data<-data.frame(train\_data, train\_labels)

head(train\_data)

Step 2: Application of decision tree:

require('tree')

# Implementing the model decision tree

my.model<-tree(train\_labels~age+income+loans, data=train\_data)

summary(my.model)

#Examine the model

plot(my.model) # both plot and text should run together

text(my.model)

Step 3: Prediction:

my.predictions=predict(my.model, test\_data, type='class')

my.predictions

test\_data$test\_labels

results=data.frame(my.predictions,test\_data$test\_labels )

results

table(results)

accuracy=sum(my.predictions==test\_labels)/length(my.predictions)

accuracy

Step 4: Cross validation:

require(tree)

cv\_tree=cv.tree(my.model, FUN=prune.misclass)

names(cv\_tree)

plot( cv\_tree$size, cv\_tree$dev , type='b')

# based on above plot choose the number. Here I chose 2.

pruned.model<-prune.misclass(my.model, best=2)

plot(pruned.model)

text(pruned.model)

Step 5: Update predicition

pruned.predictions<-predict(pruned.model, test\_data, type='class')

table(pruned.predictions, test\_data$test\_labels)

accuracy=sum(pruned.predictions==test\_labels)/length(pruned.predictions)

accuracy