1. Given that the dataset for both the Logistic Regression and CART model will be the same, i.e., the Heart dataset. The Y (dependent variable) for both the analyses is “AHD” from the original dataset. In my research, I have renamed it to “Diagnosis\_Heart\_Disease.” Furthermore, I changed it to a binary (0-1) factor variable, “Heart\_Disease\_Binary,” for the remainder of my analysis.

The X (independent variables) will be all the remaining fields/variables comprising the overall dataset structure.

Here’s a detailed description of all the columns ->

Age: Age of subject

Sex: Gender of subject:

0 = female 1 = male

Chest-pain type: Type of chest-pain experienced by the individual:

1 = typical angina

2 = atypical angina

3 = non-angina pain

4 = asymptomatic angina

Resting Blood Pressure: Resting blood pressure in mm Hg

Serum Cholesterol: Serum cholesterol in mg/dl

Fasting Blood Sugar: Fasting blood sugar level relative to 120 mg/dl: 0 = fasting blood sugar <= 120 mg/dl

1 = fasting blood sugar > 120 mg/dl

Resting ECG: Resting electrocardiographic results

0 = normal

1 = ST-T wave abnormality

2 = left ventricle hyperthrophy

Max Heart Rate Achieved: Max heart rate of the subject.

Exercise Induced Angina:

0 = no 1 = yes

ST Depression Induced by Exercise Relative to Rest: ST Depression of the subject.

Peak Exercise ST Segment:

1 = Up-sloping

2 = Flat

3 = Down-sloping

Number of Major Vessels (0-3) Visible on Fluoroscopy: Number of visible vessels under fluoroscopy

Thal: Form of thalassemia: 3

3 = normal

6 = fixed defect

7 = reversible defect

Diagnosis of Heart Disease: Indicates whether the subject is suffering from heart disease or not:

0 = absence

1, 2, 3, 4 = heart disease present

Let’s conduct a few data preparation and preliminary EDL steps, which will serve as a basis for both the models ->

names(heart) # See the current fields and field names

heart\_lr <- heart[,-1] # Remove the first column as it serves no purpose

names <- c("Age", # Create a vector containing the desired column names

"Sex",

"Chest\_Pain\_Type",

"Resting\_Blood\_Pressure",

"Serum\_Cholesterol",

"Fasting\_Blood\_Sugar",

"Resting\_ECG",

"Max\_Heart\_Rate\_Achieved",

"Exercise\_Induced\_Angina",

"ST\_Depression\_Exercise",

"Peak\_Exercise\_ST\_Segment",

"Num\_Major\_Vessels\_Flouro",

"Thalassemia",

"Diagnosis\_Heart\_Disease")

colnames(heart\_lr) <- names # Change the existing column names

names(heart\_lr) # Check to ensure that the changes have taken place

head(heart\_lr) # Check the first six records of the dataset

sum(is.na(heart\_lr)) # Check the total number of NA's present

str(heart\_lr) # Check the metadata of the dataset

attach(heart\_lr) # Optional step -> doing this makes accessing the columns easier

############# Convert a few required categorical variables from integers to factors ###############

heart\_lr$Sex <- factor(heart\_lr$Sex)

heart\_lr$Fasting\_Blood\_Sugar <- factor(heart\_lr$Fasting\_Blood\_Sugar)

heart\_lr$Resting\_ECG <- factor(heart\_lr$Resting\_ECG)

heart\_lr$Exercise\_Induced\_Angina <- factor(heart\_lr$Exercise\_Induced\_Angina)

heart\_lr$Peak\_Exercise\_ST\_Segment <- factor(heart\_lr$Peak\_Exercise\_ST\_Segment)

glimpse(heart\_lr) # a dplyr equivalent of str()

summary(heart\_lr) # Get a basic data profile

# table(Num\_Major\_Vessels\_Flouro)

# dplyr provides a lot of commands for data wrangling. One of the most powerful of them

# being mutate() which can be used to create new fields or edit the existing fields similar

# to the CASE statement on SQL.

heart\_lr <- heart\_lr %>% mutate(Chest\_Pain\_Type\_Numeric = case\_when(Chest\_Pain\_Type == "typical" ~ 1,

Chest\_Pain\_Type == "nontypical" ~ 2,

Chest\_Pain\_Type == "nonanginal" ~ 3,

Chest\_Pain\_Type == "asymptomatic" ~ 4

))

heart\_lr$Chest\_Pain\_Type\_Numeric <- factor(heart\_lr$Chest\_Pain\_Type\_Numeric)

heart\_lr <- heart\_lr %>% mutate(Resting\_ECG\_Results = case\_when(Resting\_ECG == 0 ~ "Normal",

Resting\_ECG == 1 ~ "ST-T Wave Abnormality",

Resting\_ECG == 2 ~ "Left Ventricle Hyperthropy"

))

heart\_lr <- heart\_lr %>% mutate(Heart\_Disease\_Binary = case\_when(Diagnosis\_Heart\_Disease == "No" ~ 0,

Diagnosis\_Heart\_Disease == "Yes" ~ 1

))

heart\_lr$Heart\_Disease\_Binary <- factor(heart\_lr$Heart\_Disease\_Binary)

heart\_lr <- heart\_lr %>% mutate(Thalassemia\_Levels = case\_when(Thalassemia == "normal" ~ 3,

Thalassemia == "fixed" ~ 6,

Thalassemia == "reversable" ~ 7

))

heart\_lr$Thalassemia\_Levels <- factor(heart\_lr$Thalassemia\_Levels)

glimpse(heart\_lr)

which(is.na(heart\_lr),arr.ind = TRUE) # Figure out which rows have blanks or NA's present in them

heart\_clean <- heart\_lr %>% drop\_na() %>% filter(Thalassemia != "?") # dplyr provides the piping

# capability that can be used to chain multiple commands together

# View(heart\_clean)

glimpse(heart\_clean)

# heart\_clean <- heart\_clean[,-c(3,13,14,16)]

heart\_clean <- heart\_clean[,-c(3,13,16)] # Subset out a few columns which we don't need

glimpse(heart\_clean)

heart\_tbl <- heart\_clean %>% select(Sex,Fasting\_Blood\_Sugar,Resting\_ECG,

Exercise\_Induced\_Angina,Peak\_Exercise\_ST\_Segment,

Thalassemia\_Levels,Heart\_Disease\_Binary,Chest\_Pain\_Type\_Numeric) %>%

gather(key = "key", value = "value", -Heart\_Disease\_Binary)

heart\_tbl %>% ggplot(aes(value)) +

geom\_bar(aes(x = value,

fill = Heart\_Disease\_Binary),

alpha = .6,

position = "dodge",

color = "black",

width = .8

) +

labs(x = "", # Create basic barplots to understand the prevalence and causes of heart attacks

y = "",

title = "Scaled Effect of Categorical Variables") +

theme(

axis.text.y = element\_blank(),

axis.ticks.y = element\_blank()) +

facet\_wrap(~ key, scales = "free", nrow = 4) +

scale\_fill\_manual(

values = c("#fde725ff", "#20a486ff"),

name = "Heart\nDisease",

labels = c("No HD", "Yes HD"))

heart\_tbl\_cont <- heart\_clean %>% select(Age, Resting\_Blood\_Pressure,

Serum\_Cholesterol,

Max\_Heart\_Rate\_Achieved,

ST\_Depression\_Exercise,

Num\_Major\_Vessels\_Flouro,

Heart\_Disease\_Binary) %>%

gather(key = "key", value = "value", -Heart\_Disease\_Binary)

heart\_tbl\_cont %>% ggplot(aes(y = value)) +

geom\_boxplot(aes(fill = Heart\_Disease\_Binary),

alpha = .6,

fatten = .7) +

labs(x = "",

y = "",

title = "Boxplots for Numeric Variables") + # Create boxplots to see the outliers in the continuous fields

scale\_fill\_manual(

values = c("#fde725ff", "#20a486ff"),

name = "Heart\nDisease",

labels = c("No HD", "Yes HD")) +

theme(

axis.text.x = element\_blank(),

axis.ticks.x = element\_blank()) +

facet\_wrap(~ key,

scales = "free",

ncol = 2)

heart\_tbl\_corr <- heart\_clean %>% ggcorr(high = "#20a486fe",

low = "#fde725ee",

label = TRUE,)

heart\_tbl\_corr <- heart\_clean %>% ggcorr(high = "#20a486fe",

low = "#fde725ee",

label = TRUE,

hjust = 0.80,

size = 3,

label\_size = 3,

nbreaks = 5) +

labs(title = "Correlation Matrix",

subtitle = "Pearson Method for pair-wise correlations")

heart\_tbl\_corr # The above lines plot the one -to-one correlation between the fields

Diagram

Description automatically generated

Bar chart

Description automatically generated with medium confidence

Chart

Description automatically generated

Basis the above plots, a few conclusions can be drawn ->

1. Presence of exercise-induced angina seems to be moderately associated with increased incidence of heart attacks.
2. Males tend to have a higher incidence of heart attacks.
3. Higher Thalassemia score may indicate a higher chance of heart attack.
4. Higher cholesterol levels correspond to higher levels of heart attacks.

It should be noted that the above observations do not correspond to causality.

1. **Logistic Regression** -> The original document undergoes a 75/25 train/test split for the logistic regression model. Concurrently, a few factor character variables have either been re-coded or dropped.

The regression has been performed once, each using the training and full datasets, respectively. The outputs of the same can be found below:

logit\_train <- glm(Heart\_Disease\_Binary ~ .,data = heart\_train, family = "binomial")

summary(logit\_train)

logit\_full <- glm(Heart\_Disease\_Binary ~ .,data = heart\_clean, family = "binomial")

summary(logit\_full)

logLik(logit\_train)

logLik(logit\_full)

with(logit\_full, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE))

with(logit\_train, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE))

Text

Description automatically generated

Text

Description automatically generated

The dataset “heart\_clean” is the entire dataset post the initial data cleaning phase. The log-likelihood and the p-values of both the models are given below:

Text

Description automatically generated

The regression was re-run to basis the significant variables observed to yield a more parsimonious model as shown below:

logit\_train\_parsimonious <- glm(Heart\_Disease\_Binary ~ Resting\_Blood\_Pressure +

Exercise\_Induced\_Angina +

Thalassemia\_Levels +

Num\_Major\_Vessels\_Flouro,data = heart\_train, family = "binomial")

summary(logit\_train\_parsimonious)

logLik(logit\_train\_parsimonious)

logit\_full\_parsimonious <- glm(Heart\_Disease\_Binary ~ Sex + Resting\_Blood\_Pressure +

Chest\_Pain\_Type\_Numeric +

Thalassemia\_Levels +

Num\_Major\_Vessels\_Flouro,data = heart\_clean, family = "binomial")

summary(logit\_full\_parsimonious)

logLik(logit\_full)

with(logit\_full\_parsimonious, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE))

with(logit\_train\_parsimonious, pchisq(null.deviance - deviance, df.null - df.residual, lower.tail = FALSE))

Text

Description automatically generated

Text

Description automatically generated

Similarly, the log-likelihood and the p-values of the above two models are also shown below:



Text

Description automatically generated

We’ll also do regression by setting the dependent variable against no independent variable, i.e., regressing it against the intercept. The results of the above are given below:

Text

Description automatically generated

**Model Interpretations**: The benefit of doing the regression each on the training and the entire dataset alongside their parsimonious models respectively allows us to see which independent variables are significant in predicting the chance of inducing a heart attack. Let us go through both the parsimonious models and interpret what they mean ->

1. Train -> Mathematically,

**Heart\_Disease\_Binary = -4.829 + 0.0135 \* (Resting\_Blood\_Pressure) + 1.9877 \* (Exercise Induced Angina – Lv1) + 1.3903 \* (Thalassemia – Level 6) + 2.358 \* (Thalassemia – Level 7) + 1.317 \* (Num\_Major\_Vessels\_Fluoro)**

The above equation yields the following interpretations ->

1. For a unit increase in Resting BP, the odds of having a heart attack increase by a factor of 0.0135.

ii.) Having an Exercise-Induced Angina – Lv1 is associated with a 1.9877 odds increase for a heart attack.

iii.) Having a Fixed Defect Thalassemia, i.e., Thalassemia – Lv6, is associated with a 1.3903 odds increase for a heart attack.

iv.) Having a Reversible Defect Thalassemia, i.e., Thalassemia – Lv7, is associated with a 2.358 odds increase for a heart attack.

v.) For a unit increase in the Number of Major vessels visible under Fluoroscopy, the odds of having a heart attack increase by a factor of 1.317.

b.) Full data -> Mathematically,

**Heart\_Disease\_Binary = -6.052 + 0.951 \* (Sex-Male) + 0.02 \* (Resting\_Blood\_Pressure) + 0.174 \* (Chest\_Pain\_Type\_Numeric – Level 2) + (-0.11) \* (Chest\_Pain\_Type\_Numeric – Level 3) + 2.04 \* (Chest\_Pain\_Type\_Numeric – Level 4) + 0.892 \* (Thalassemia – Level 6) + 1.835 \* (Thalassemia – Level 7) + 1.174 \* (Num\_Major\_Vessels\_Fluoro)**

The above equation yields the following interpretations ->

1. For males, the odds of having a heart attack increase by a factor of 0.951.

ii.) For a unit increase in Resting BP, the odds of incurring a heart attack increase by a factor of 0.02

iii.) Having an Atypical Angina, i.e., Chest\_Pain\_Type\_Numeric – Level 2, is associated with a 0.174 odds increase of a heart attack.

iv.) Having Non-Angina pain, i.e., Chest\_Pain\_Type\_Numeric – Level 3 is associated with a 0.11 odds decrease of a heart attack.

v.) Having an Asymptomatic Angina, i.e., Chest\_Pain\_Type\_Numeric – Level 4 is associated with a 2.04 odds increase of a heart attack.

vi.) Having a Fixed Defect Thalassemia, i.e., Thalassemia – Lv6, is associated with a 0.892 odds increase for a heart attack.

vii.) Having a Reversible Defect Thalassemia, i.e., Thalassemia – Lv7, is associated with a 1.835 odds increase for a heart attack.

viii.) For a unit increase in the Number of Major vessels visible under Fluoroscopy, the heart attack odds increase by a factor of 1.174.

Although the train and the entire dataset models have overlapping variables present, we’ll consider the full dataset parsimonious model as the model of choice owing to its lower p-value, log-likelihood, and confidence interval. Even though the full dataset model has a higher AIC value, thus diminishing its out-of-sample predictability, it is still the right due to a lower record to fields ratio.

**CART Model** -> The basic data preparations steps will be the same as that executed in Logistic Regression. Here, we will be using the training dataset to carry out our analysis.

Let us have a look below at the initial tree which we obtain.

Diagram

Description automatically generated

The above tree has been obtained using a 10-fold Cross-validation indicated by the “xval” argument in the rpart. Control command.

Let us now look at the Complexity Parameter and identify the depth by which we can prune to obtain the optimal tree. The Cp values can be found below ->

A picture containing text

Description automatically generated

From the above table, we see that the 5th Complexity parameter with nine splits and at a depth of six results in the minimum relative error, and hence we will use this to prune the tree.

We will run a for-loop that takes the index equal to our depth and calculates the training, testing, and cross-validation error,

train\_error <- double(6)

test\_error <- double(6)

cv\_error <- double(6)

for(i in 1:nrow(class\_param)){

alpha <- class\_param[i, 'CP']

train <- table(heart\_train$Heart\_Disease\_Binary, predict(prune(heart\_cart, cp = alpha),

heart\_train, type = "class"))

train\_error[i] <- 1 - sum(diag(train))/sum(train)

cv\_error[i] <- class\_param[i, "xerror"] \* class\_param[i, 'rel error']

test <- table(heart\_test$Heart\_Disease\_Binary, predict(prune(heart\_cart, cp = alpha),

heart\_test, type = "class"))

test\_error[i] <- 1 - sum(diag(test))/sum(test)

}

train\_error

test\_error

cv\_error

Text

Description automatically generated

Let us now plot the above-derived errors.

matplot(class\_param[,'nsplit'], cbind(train\_error, cv\_error, test\_error),

pch=19, col=c("red", "black", "blue"), type="b",

ylab="Loss/error", xlab="Depth/# of Splits")

legend("right", c('Train', 'CV', 'Test') ,col=seq\_len(3),cex=0.8,fill=c("red", "black", "blue"))

plotcp(heart\_cart)

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

So, from the above two plots, we can see that basis the Complexity Parameters obtained, the number of splits should be > 2 as anything less will not be decipherable. However, anything excessive will render the pruning ineffective. Hence, we will choose nsplit = 4 to prune the tree.

heart\_prune <- prune(heart\_cart,cp = class\_param[4, 'CP'])

prp(heart\_prune)

Diagram

Description automatically generated

Let us now examine the confusion matrix and the accuracy of the pruned and unpruned trees.

heart\_cfm\_cart <- table(heart\_test$Heart\_Disease\_Binary,predict(heart\_prune,

type = "class", newdata = heart\_test))

heart\_cfm\_cart

cart\_accuracy <- sum(diag(heart\_cfm\_cart))/sum(heart\_cfm\_cart)

cart\_accuracy

cart\_train\_accuracy <- 1-train\_error[6]

cart\_train\_accuracy

cart\_test\_accuracy <- 1-test\_error[6]

cart\_test\_accuracy

cart\_cv\_accuracy <- 1-cv\_error[6]

cart\_cv\_accuracy

prune\_train\_acc <- 1-train\_error[4]

prune\_train\_acc

prune\_test\_acc <- 1-test\_error[4]

prune\_test\_acc

prune\_cv\_acc <- 1-cv\_error[4]

prune\_cv\_acc

Text

Description automatically generated

1. **Summary Table: Logistic Regression** ->

|  |  |  |  |
| --- | --- | --- | --- |
| Metric | Testing | Training | Full Dataset |
| Accuracy | 0.773 | 0.846 | 0.841 |
| Sensitivity | 0.780 | 0.865 | 0.868 |
| Specificity | 0.764 | 0.825 | 0.810 |
| PPV | 0.8 | 0.851 | 0.842 |
| NPV | 0.742 | 0.841 | 0.840 |
| AUC | 0.843 | 0.907 | 0.908 |

**Summary Table: CART Model** ->

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Unpruned Tree | Pruned Tree |
| Tree Size | | 9 | 4 |
| Accuracy | Train | 0.891 | 0.864 |
| Test | 0.746 | 0.786 |
| All (Cross-validated) | 0.889 | 0.875 |

1. **ROC Curves** ->
2. **Train** ->

Chart, line chart

Description automatically generated

1. **Test** ->

Chart

Description automatically generated with medium confidence

1. **Full** ->

Chart, line chart

Description automatically generated

**AUC Curves** ->

1. **Test** ->

Chart, line chart

Description automatically generated

1. **Train** ->

Chart

Description automatically generated

1. **Full** ->

Chart

Description automatically generated

1. **S-Curve** -> The S-Curve can be generated using the following code:

heart\_scurve <- glm(Heart\_Disease\_Binary ~ Num\_Major\_Vessels\_Flouro, data = heart\_clean, family = "binomial")

plot(as.numeric(Heart\_Disease\_Binary) ~ Num\_Major\_Vessels\_Flouro, data = heart\_clean,

col = "green", pch = "|", xlim = c(-1,3), ylim = c(0,1),

main = "Using Logistic Regression for Classification") +

abline(h = 0, lty = 3) +

abline(h = 1, lty = 3) +

abline(h = 0.5, lty = 2) +

curve(predict(heart\_scurve, data.frame(Num\_Major\_Vessels\_Flouro = x), type = "response"),

add = TRUE, lwd = 3, col = "dodgerblue") +

abline(v = -coef(heart\_scurve)[1] / coef(heart\_scurve)[2], lwd = 3)

Chart, line chart

Description automatically generated

1. **References** ->
2. [R-Bloggers](https://www.r-bloggers.com/2019/09/heart-disease-prediction-from-patient-data-in-r/) -> Primarily used for the EDA.
3. [UCLA](https://stats.oarc.ucla.edu/other/mult-pkg/faq/general/faq-what-are-pseudo-r-squareds/)
4. Cross-Validated -> [1](https://stats.stackexchange.com/questions/9171/aic-or-p-value-which-one-to-choose-for-model-selection), [2](https://stats.stackexchange.com/questions/8511/how-to-calculate-pseudo-r2-from-rs-logistic-regression), [3](https://stats.stackexchange.com/questions/18750/hosmer-lemeshow-vs-aic-for-logistic-regression), [4](https://stats.stackexchange.com/questions/312780/why-is-accuracy-not-the-best-measure-for-assessing-classification-models/312787#312787)
5. [CRAN](https://search.r-project.org/CRAN/refmans/generalhoslem/html/logitgof.html)