Function Dictionary

Libraries

```
library(dplyr)
library(ggplot2)
library(caret)
library(rpart)
library(rpart.plot)
```

Dataset read in

```
dataname=read.csv("Datasets/filename.csv")
```

Dataset Views, can also click on the name in the environment window (upper right)

```
head(dataname)
names(dataname)
View(dataname)
```

Column selection to keep a few columns

```
newname=select(dataname, column1, column2, column3, column4, etc)
```

Create New Column

```
dataname=mutate(dataname, newColumn = formula)
dataname=mutate(dataname, newColumn1 = formula1, newColumn2 = formula2,
                newColumn3 = formula3)
```

Scatter Plots

```
ggplot(dataname, aes(x = X-Variable, y = Y-variable)) + geom_point()
```

BoxPlots

```
ggplot(dataname, aes(x = groupColumn, y = numericColumn)) + geom_boxplot()
```

Bargraph

```
ggplot(dataname, aes(x = groupColumn)) + geom_bar()
```

Histogram

```
ggplot(dataname, aes(x = groupColumn)) + geom_histogram(col='red',fill='gray',bins=25)
```

Extra Groups in plots, Faceting. Just add to any plot above.

```
+ facet_wrap( ~ groupColumn2)
```

Basic Statistics

```
summarize(dataname, Average = mean(columnName, na.rm=T))
summarize(dataname, Variance = var(columnName, na.rm=T))
summarize(dataname, SD = sd(columnName, na.rm=T))
summarize(dataname, Maximum = max(columnName, na.rm=T),Minimum = min(columnName, na.rm=T))
summarize(dataname, Median = median(columnName, na.rm=T))
summarize(dataname, Correlation = cor(columnName1, columnName2, use='complete.obs'))
summarize(dataname, Missing = sum(is.na(columnName)))

#Several at once just keep adding new ones
summarize(dataname, Average = mean(columnName, na.rm=T),
          Variance = var(columnName, na.rm=T), SD = sd(columnName, na.rm=T),
          Correlation = cor(columnName1, columnName2, use='complete.obs'))
```

Group Statistics (Put %>%View at the end to see the full results, or name it then view it)

The formulas are from the basic stats section above.

```
group_by(dataname, groupName) %>%
  summarize(StatName =Formula)

group_by(dataname, groupName) %>%
  summarize(StatName1 = Formula1, StatName2 = Formula2,StatName3 = Formula3)
```

Counts

```
group_by(dataname, groupColumn1, groupColumn2, groupColumn3, ColumnToCount) %>%
  summarize(Count=n())%>%
  mutate(Percentage=Count/sum(Count))
```

Make a subset of the dataset replace rule with selection criteria

Use == for equality and != for not equal. Also can use > and <.

```
newname=filter(dataname, RULE )
```

Order Low to High

```
dataname=arrange(dataname,columnName)
```

Order High to Low

```
dataname=arrange(dataname,desc(columnName))
```

Binomial

```
dbinom(x, n, p)
sum(dbinom(x1:x2, n, p))
```

Negative Binomial

```
dnbinom(x, k, p)
sum(dnbinom(x1:x2, k, p))
```

Poisson

```
dpois(x, lambda)
sum(dpois(x1:x2, lambda))
```

Normal

```
#Below
pnorm(x, mean, sd)

#Above
1-pnorm(x, mean, sd)

#Between
pnorm(x2, mean, sd) - pnorm(x1, mean, sd)
```

Chi-squared Test

```
observationVector = c( )   #enter data in between ( )
expectationVector = c( )
chisq.test(observationVector, expectationVector, rescale.p = T)
```

Binomial Test

```
binom.test(x, n , p)
```

Linear Models

Fitting

```
mod = lm(data = dataname, outcome ~ x1 + x2 + x3 +  . . .)
mod
summary(mod)
```

Residual PLot

Only change the dataname to what you called your dataset (or change mod if you changed it)

```
dataname = mutate(datanae, fitted = mod$fitted.values, residuals = mod$residuals)
ggplot(data=dataname, aes(x=fitted,y=residuals)) + geom_point()+geom_hline(yintercept = 0)
```

Prediction

Read in the new dataset, View the dataset at the end

```
preds = predict(mod, newdata = NEWdataNAME, type='response')
NEWdataNAME = mutate(NEWdataNAME , Predictions = preds )
View(dataname)
```

Logistic Model

Fitting

```
mod = glm(family='binomial', data = dataname, outcome ~ x1 + x2 + x3 +  . . .)
mod
summary(mod)
```

Prediction

Read in the new dataset. You can use the old data as the new dataset to evaluate the model.

```
preds = predict(mod, newdata = NEWdataNAME, type='response')
NEWdataNAME = mutate(NEWdataNAME , Predictions = preds )
NEWdataNAME=mutate(NEWdataNAME, ClassPrediction = factor(
  ifelse(Predictions > threshold ,'PrimaryClass','SecondaryClass')))

table(select(dataname,ClassPrediction,Outcome))
confusionMatrix( table(select(dataname,ClassPrediction,Outcome)) )
```

Decision Tree

Fitting

```
mod=rpart(data = dataname, outcome ~ x1 + x2 + x3 +  . . .)
rpart.plot(mod,type = 1,extra = 4)
```

Prediction

```
preds = predict(mod, newdata = NEWdataNAME, type='class')
NEWdataNAME = mutate(NEWdataNAME , Predictions = preds )

table(select(dataname,ClassPrediction,Outcome))
confusionMatrix( table(select(dataname,ClassPrediction,Outcome)) )
```

Training and Test Sets

```
n=nrow(dataname)
Trainrows = sample(1:n,0.9*n)
TrainDF = dataname[Trainrows, ]
TestDF = dataname[-Trainrows, ]
```