Classification on Diamond Dataset

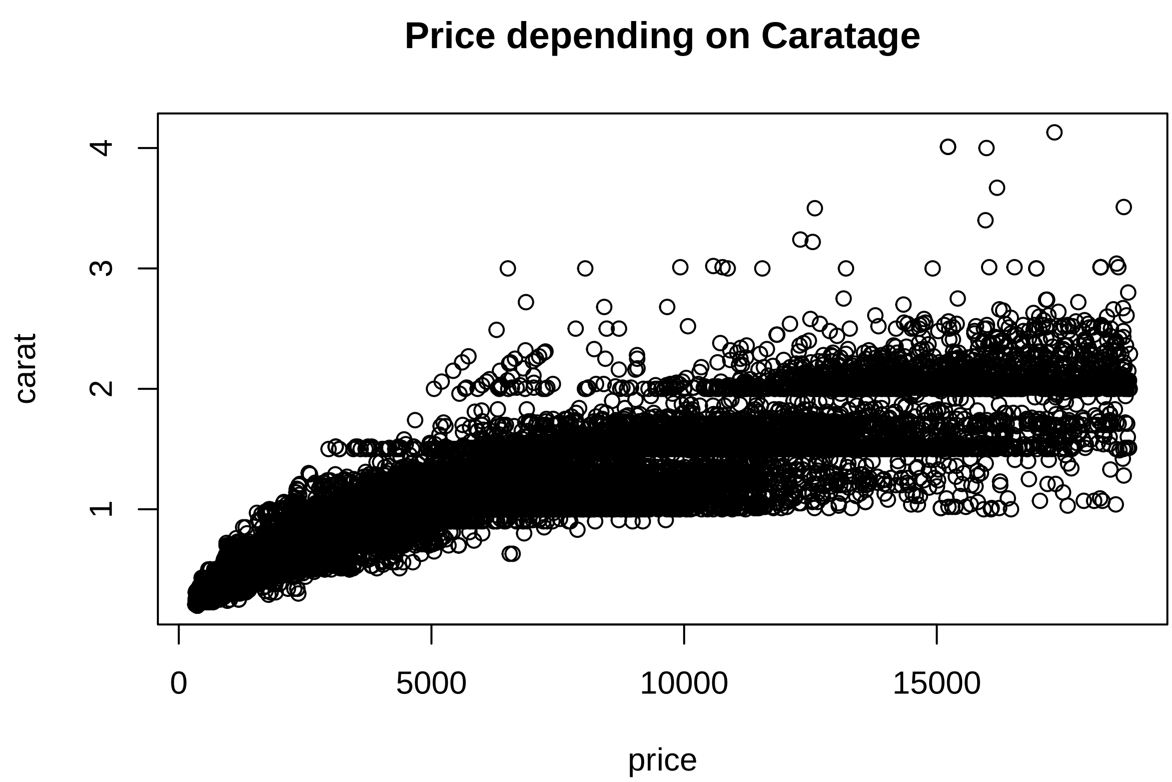
Ved Nigam and Swarn Singh

02/17/2023

```
diamonds_data <- read.csv("diamonds.csv", header = TRUE)

# Splitting data into train and test data (20-80)
set.seed(1234)
i <- sample(1:nrow(diamonds_data), nrow(diamonds_data)*.80, replace = FALSE)
train <- diamonds_data[i,]
test <- diamonds_data[-i,]

## Some basic graphs with training data
# Scatterplot of caratage and price
plot(carat~price,
     data = train,
     main = "Price depending on Caratage")
```



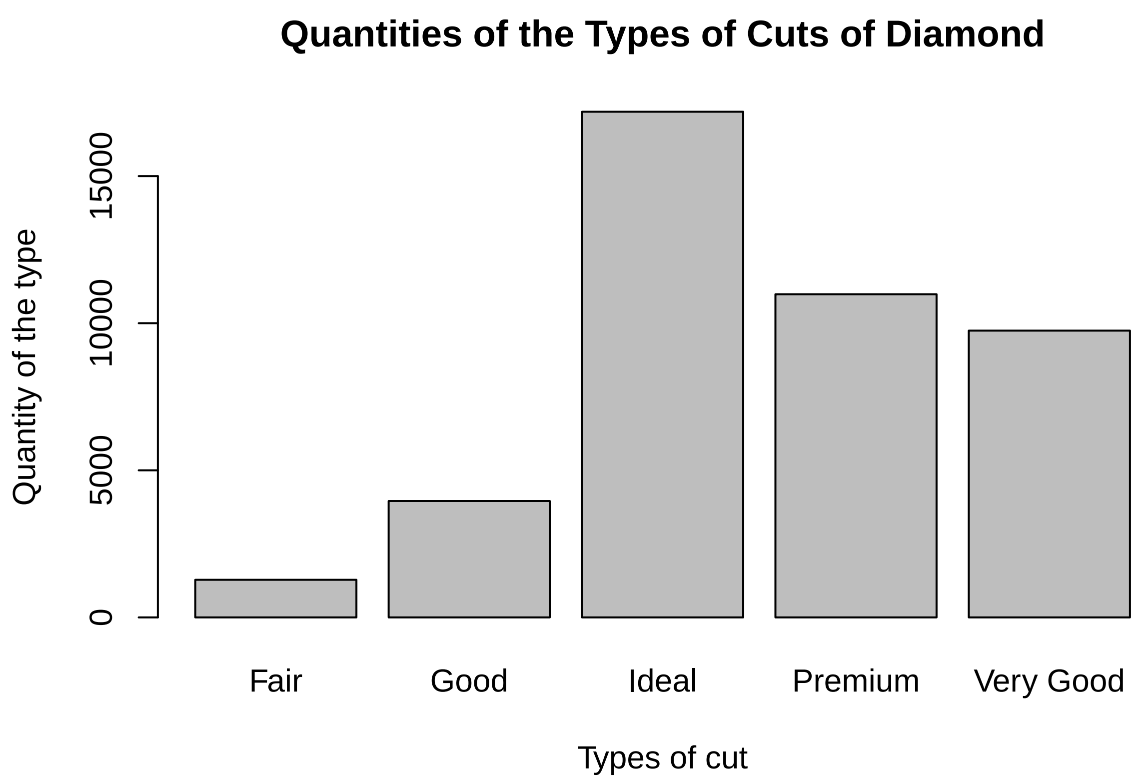
```r
# Barplot of the count of the different cuts
counts <- table(train$cut)
barplot(counts,
        data = train,
        xlab = "Types of cut",
        ylab = "Quantity of the type",
        main = "Quantities of the Types of Cuts of Diamond")
```

```
## Warning in plot.window(xlim, ylim, log = log, ...): "data" is not a graphical
## parameter

## Warning in axis(if (horiz) 2 else 1, at = at.l, labels = names.arg, lty =
## axis.lty, : "data" is not a graphical parameter

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...): "data"
## is not a graphical parameter

## Warning in axis(if (horiz) 1 else 2, cex.axis = cex.axis, ...): "data" is not a
## graphical parameter
```




```r
## Data exploration on the training set
# Number of rows in the training data
nrow(train)
```

```
## [1] 43152
```

```r
# The columns in the dataset
names(train)
```

```
##  [1] "X"       "carat"   "cut"     "color"   "clarity" "depth"   "table"
##  [8] "price"   "x"       "y"       "z"
```

```r
# Structure of the data in each column
str(train)
```

```
## 'data.frame':    43152 obs. of  11 variables:
##  $ X      : int  40784 40854 41964 15241 33702 35716 17487 15220 19838 2622 ...
##  $ carat  : num  0.61 0.53 0.23 1.33 0.3 0.3 2.01 1.12 1.02 0.74 ...
##  $ cut    : chr  "Good" "Premium" "Very Good" "Ideal" ...
##  $ color  : chr  "E" "G" "E" "J" ...
##  $ clarity: chr  "I1" "SI2" "VVS2" "VS1" ...
##  $ depth  : num  63.4 60.8 62.3 61.3 61.6 60.8 63.9 61.8 62.1 62.3 ...
##  $ table  : num  57.1 58 55 57 56 57 59 55 57 56 ...
##  $ price  : int  1168 1173 505 6118 838 911 7024 6110 8401 3226 ...
##  $ x      : num  5.37 5.21 3.9 7.11 4.3 4.34 8.01 6.64 6.43 5.76 ...
##  $ y      : num  5.43 5.19 3.93 7.08 4.34 4.31 7.92 6.7 6.45 5.79 ...
##  $ z      : num  3.42 3.16 2.44 4.35 2.66 2.63 5.09 4.12 4 3.6 ...
```

```r
# The first 2 rows of the data
head(train, n = 2)
```

```
##           X carat     cut color clarity depth table price    x    y    z
## 40784 40784  0.61    Good     E      I1  63.4  57.1  1168 5.37 5.43 3.42
## 40854 40854  0.53 Premium     G     SI2  60.8  58.0  1173 5.21 5.19 3.16
```

```r
# Range of the prices of the diamonds
range(train$price)
```

```
## [1]   326 18823
```

```r
## Logistic regression for classifying quality of a diamond to see if it is
# "Ideal"
ideal <- diamonds_data
ideal$cut <- as.factor(ifelse (ideal$cut == "Ideal", 1, 0))

x <- sample(1:nrow(ideal), nrow(ideal)*.80, replace = FALSE)
train_ideal <- ideal[x,]
test_ideal <- ideal[-x,]

glm1 <- glm(cut~., data = train_ideal, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
probs <- predict(glm1, newdata = test_ideal)
pred <- ifelse(probs > .5, 1, 0)
acc <- mean(pred == test_ideal$cut)
print(paste("accuracy = ", acc))
```

```
## [1] "accuracy =  0.766314423433445"
```

```r
table(pred, test$cut)
```

```
##
## pred Fair Good Ideal Premium Very Good
##    0  246  697  2989    1956      1628
##    1   86  253  1375     851       707
```

```r
# The above model outputs a table of how accurate the model is for predicting
# the quality of caratage. We can see that the model is for an "Ideal" diamond
# is more accurate for the ideal diamonds.

## Naïve-Bayes model
library(e1071)
nb1 <- naiveBayes(cut~., data = train_ideal)
(nb1)
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         0         1
## 0.6008528 0.3991472
##
## Conditional probabilities:
##    X
## Y       [,1]     [,2]
##   0 25590.09 15742.63
##   1 29048.65 15020.07
##
##    carat
## Y        [,1]      [,2]
##   0 0.8614590 0.4904657
##   1 0.7037935 0.4333527
##
##    color
## Y            D          E          F          G          H          I
##   0 0.12141314 0.18173403 0.17764579 0.19758562 0.16102283 0.10212897
##   1 0.12981886 0.18091036 0.17655597 0.22817000 0.14619136 0.09707385
##    color
## Y            J
##   0 0.05846961
```

```
##   1 0.04127961
##
##    clarity
## Y            I1          IF         SI1         SI2         VS1         VS2
##   0 0.018705646 0.018127121 0.270826905 0.203602283 0.141584388 0.221845110
##   1 0.007199257 0.056316767 0.198850441 0.117800743 0.166627961 0.236936832
##    clarity
## Y          VVS1        VVS2
##   0 0.049560321 0.075748226
##   1 0.094867627 0.121400372
##
##    depth
## Y       [,1]      [,2]
##   0 61.77622 1.7552528
##   1 61.71330 0.7181288
##
##    table
## Y       [,1]     [,2]
##   0 58.46540 2.191872
##   1 55.94798 1.245834
##
##    price
## Y       [,1]     [,2]
##   0 4250.197 4078.802
##   1 3470.499 3820.841
##
##    x
## Y       [,1]     [,2]
##   0 5.880090 1.135272
##   1 5.509925 1.064383
##
##    y
## Y       [,1]     [,2]
##   0 5.875677 1.126589
##   1 5.523186 1.078548
##
##    z
## Y       [,1]      [,2]
##   0 3.630451 0.7261531
##   1 3.403139 0.6580029
```

```r
# This predicts the probability of each observation being in the regression
# model.

pred1 <- predict(nb1, newdata = test_ideal, type = "class")
table(pred1, test_ideal$cut)
```

```
##
## pred1    0    1
##     0 5046 1045
##     1 1415 3282
```

```r
mean(pred1 == test_ideal$cut)
```

```
## [1] 0.7719689
```

```r
pred1_raw <- predict(nb1, newdata = test_ideal, type = "raw")
head(pred1_raw)
```

```
##              0            1
## [1,] 0.3723624 6.276376e-01
## [2,] 0.8930957 1.069043e-01
## [3,] 0.0566690 9.433310e-01
## [4,] 0.9999995 5.328312e-07
## [5,] 0.1393215 8.606785e-01
## [6,] 0.1269478 8.730522e-01
```

```r
## Comparing the two models
# In this case, it looks like the Bayes model is more accurate by about .01.
# We only made a model to see if the diamond was "Ideal", but the model could
# also have become a multi-class classifier for all the types of diamonds.
# Logistic regression is more effective for boolean outcomes whereas the Bayes
# model will be better at handling multi-class situations. A very obvious
# benefit of the Bayes model in terms of user friendliness is that a correlation
# between two variables is not required. But, this is also a strength of the
# logistic regression: it proves/disproves correlation between two variables.

## Benefits/drawbacks of the classifiers
# The benefit of the classifier we used for the regression on the quality of
# the diamond was that there were a very limited amount of categories that the
# data could have been classified into. By seeing the probabilities of the model
# (designed for an "ideal" diamond) classifying all the other types of of
# diamonds, it was very reassuring to see that the model is more accurate by
# about 20% for an "ideal" diamond than aby other diamond. I cannot think of
# any drawbacks.
```