Classification

Aleezah Athar

This dataset is from https://archive.ics.uci.edu/ml/datasets/Bank+Marketing (https://archive.ics.uci.edu/ml/datasets/Bank+Marketing) First we set the seed to 1234 to get the same results each time Then we read in the csv file which contains our data

```
set.seed(1234)
df <- read.csv("bank-full.csv", sep = ";", quote = "")
df
```

X.age. <int>	X.job. <chr>	X.marital. <chr>	X.education. <chr>	X.default. <chr>	X.balance. <int>	X.housing. <chr>	X.loan <chr>
58	"management"	"married"	"tertiary"	"no"	2143	"yes"	"no"
44	"technician"	"single"	"secondary"	"no"	29	"yes"	"no"
33	"entrepreneur"	"married"	"secondary"	"no"	2	"yes"	"yes"
47	"blue-collar"	"married"	"unknown"	"no"	1506	"yes"	"no"
33	"unknown"	"single"	"unknown"	"no"	1	"no"	"no"
35	"management"	"married"	"tertiary"	"no"	231	"yes"	"no"
28	"management"	"single"	"tertiary"	"no"	447	"yes"	"yes"
42	"entrepreneur"	"divorced"	"tertiary"	"yes"	2	"yes"	"no"
58	"retired"	"married"	"primary"	"no"	121	"yes"	"no"
43	"technician"	"single"	"secondary"	"no"	593	"yes"	"no"

1-10 of 10,000 rows | 1-8 of 17 columns Previous **1** 2 3 4 5 6 ... 1000 Next

We change all the character variables to factor variables so we can do classification. Next, we divide into 80/20 train/test by randomly selecting 80% of the rows to be the training data and 20% to be the testing data.

```
for(i in 1:ncol(df)){
  if(is.character(df[,i])){
    df[,i] <- as.factor(df[,i])
  }
}

levels(df$X.y.)<-c("no", "yes")

ind <- sample(2, nrow(iris), replace=TRUE, prob=c(0.8, 0.2))
df.train <- df[ind==1, 1:16]
df.test <- df[ind==2, 1:16]
df.trainLabels <- df[ind==1, 17]
df.testLabels <- df[ind==2, 17]

alternative.df.train <- df[ind==1,]
alternative.df.test <- df[ind==2,]
```

We use at least 5 R functions for data exploration, using the training data. We can use the names, dimension, summary, structure and head functions to get information about the data. We can also use the colSums and is.na functions together to check for any NA values.

```
names(df.train)
```

```
##  [1] "X.age."       "X.job."       "X.marital."   "X.education." "X.default."
##  [6] "X.balance."   "X.housing."   "X.loan."      "X.contact."   "X.day."
## [11] "X.month."     "X.duration."  "X.campaign."  "X.pdays."     "X.previous."
## [16] "X.poutcome."
```

```
dim(df.train)
```

```
## [1] 37074    16
```

```
summary(df.train)
```

```
##      X.age.              X.job.          X.marital.        X.education.
##  Min.   :18.00   "blue-collar":7983   "divorced": 4244   "primary"  : 5646
##  1st Qu.:33.00   "management" :7763   "married" :22294   "secondary":18965
##  Median :39.00   "technician" :6199   "single"  :10536   "tertiary" :10932
##  Mean   :40.92   "admin."     :4257                      "unknown"  : 1531
##  3rd Qu.:48.00   "services"   :3411
##  Max.   :95.00   "retired"    :1848
##                  (Other)      :5613
##  X.default.      X.balance.    X.housing.     X.loan.         X.contact.
##  "no" :36402   Min.   :-8019   "no" :16435   "no" :31183   "cellular" :24038
##  "yes":  672   1st Qu.:   75   "yes":20639   "yes": 5891   "telephone": 2360
##                Median :  457                               "unknown"  :10676
##                Mean   : 1366
##                3rd Qu.: 1435
##                Max.   :98417
##
##      X.day.         X.month.      X.duration.      X.campaign.
##  Min.   : 1.00   "may"  :11284   Min.   :   0.0   Min.   : 1.000
##  1st Qu.: 8.00   "jul"  : 5661   1st Qu.: 103.0   1st Qu.: 1.000
##  Median :16.00   "aug"  : 5118   Median : 180.0   Median : 2.000
##  Mean   :15.82   "jun"  : 4381   Mean   : 257.5   Mean   : 2.759
##  3rd Qu.:21.00   "nov"  : 3256   3rd Qu.: 318.0   3rd Qu.: 3.000
##  Max.   :31.00   "apr"  : 2406   Max.   :3881.0   Max.   :58.000
##                  (Other): 4968
##     X.pdays.       X.previous.         X.poutcome.
##  Min.   : -1.00   Min.   :  0.0000   "failure": 3994
##  1st Qu.: -1.00   1st Qu.:  0.0000   "other"  : 1497
##  Median : -1.00   Median :  0.0000   "success": 1240
##  Mean   : 40.03   Mean   :  0.5742   "unknown":30343
##  3rd Qu.: -1.00   3rd Qu.:  0.0000
##  Max.   :871.00   Max.   :275.0000
##
```

```
str(df.train)
```

```
## 'data.frame':	37074 obs. of  16 variables:
##  $ X.age.      : int  58 44 33 47 35 28 42 58 43 41 ...
##  $ X.job.      : Factor w/ 12 levels "\"admin.\"","\"blue-collar\"",..: 5 10 3 2 5 5
3 6 10 1 ...
##  $ X.marital.  : Factor w/ 3 levels "\"divorced\"",..: 2 3 2 2 2 3 1 2 3 1 ...
##  $ X.education.: Factor w/ 4 levels "\"primary\"",..: 3 2 2 4 3 3 3 1 2 2 ...
##  $ X.default.  : Factor w/ 2 levels "\"no\"","\"yes\"": 1 1 1 1 1 1 2 1 1 1 ...
##  $ X.balance.  : int  2143 29 2 1506 231 447 2 121 593 270 ...
##  $ X.housing.  : Factor w/ 2 levels "\"no\"","\"yes\"": 2 2 2 2 2 2 2 2 2 2 ...
##  $ X.loan.     : Factor w/ 2 levels "\"no\"","\"yes\"": 1 1 2 1 1 2 1 1 1 1 ...
##  $ X.contact.  : Factor w/ 3 levels "\"cellular\"",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ X.day.      : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ X.month.    : Factor w/ 12 levels "\"apr\"","\"aug\"",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ X.duration. : int  261 151 76 92 139 217 380 50 55 222 ...
##  $ X.campaign. : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ X.pdays.    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ X.previous. : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X.poutcome. : Factor w/ 4 levels "\"failure\"",..: 4 4 4 4 4 4 4 4 4 4 ...
```

```
head(df.train)
```

	X.age. <int>	X.job. <fct>	X.marital. <fct>	X.education. <fct>	X.default. <fct>	X.balance. <int>	X.housing. <fct>	X.loar <fct>
1	58	"management"	"married"	"tertiary"	"no"	2143	"yes"	"no"
2	44	"technician"	"single"	"secondary"	"no"	29	"yes"	"no"
3	33	"entrepreneur"	"married"	"secondary"	"no"	2	"yes"	"yes"
4	47	"blue-collar"	"married"	"unknown"	"no"	1506	"yes"	"no"
6	35	"management"	"married"	"tertiary"	"no"	231	"yes"	"no"
7	28	"management"	"single"	"tertiary"	"no"	447	"yes"	"yes"

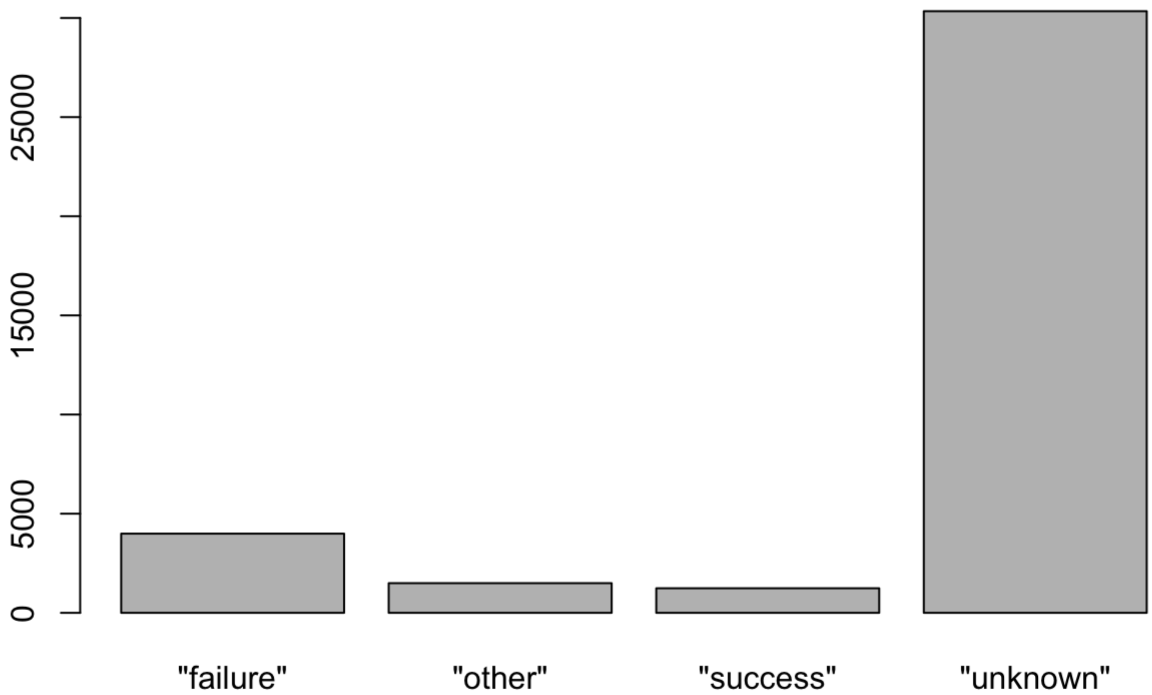
6 rows | 1-9 of 17 columns

```
colSums(is.na(df.train))
```

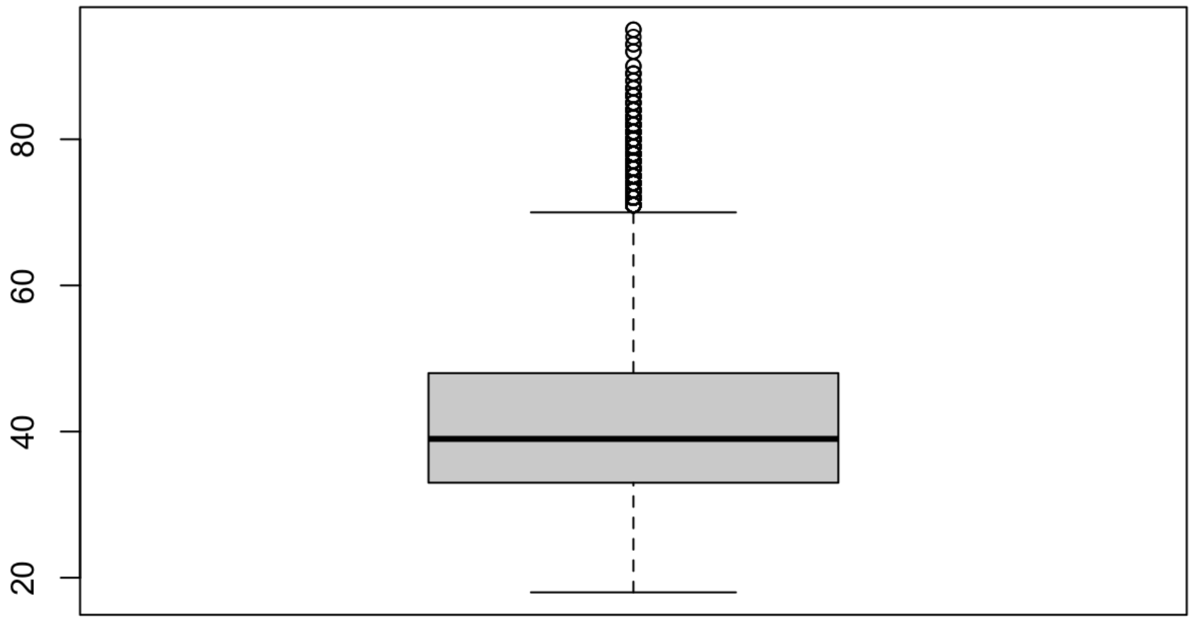
```
##       X.age.       X.job.   X.marital. X.education.   X.default.   X.balance. 
##            0            0            0            0            0            0 
##   X.housing.      X.loan.   X.contact.       X.day.     X.month.  X.duration. 
##            0            0            0            0            0            0 
##  X.campaign.     X.pdays.  X.previous.  X.poutcome. 
##            0            0            0            0
```

Creating 2 informative graphs using the training data.

```
plot(df.train$X.poutcome.)
```




```
boxplot(df.train$X.age.)
```



##

Logistic Regression

Building a logistic regression model and outputting the summary using the glm and summary functions.

```
glm1<-glm(X.y.~., data=alternative.df.train, family=binomial)
summary(glm1)
```

```
##
## Call:
## glm(formula = X.y. ~ ., family = binomial, data = alternative.df.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.9466  -0.3714  -0.2495  -0.1491   3.4705
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -2.461e+00  2.033e-01 -12.104  < 2e-16 ***
## X.age.                   2.772e-03  2.450e-03   1.131 0.257885
## X.job."blue-collar"     -3.425e-01  8.019e-02  -4.271 1.94e-05 ***
## X.job."entrepreneur"    -3.453e-01  1.383e-01  -2.497 0.012525 *
## X.job."housemaid"       -6.836e-01  1.538e-01  -4.446 8.75e-06 ***
## X.job."management"      -2.211e-01  8.078e-02  -2.736 0.006211 **
## X.job."retired"          1.505e-01  1.081e-01   1.392 0.163831
## X.job."self-employed"   -3.716e-01  1.252e-01  -2.968 0.002996 **
## X.job."services"        -2.851e-01  9.347e-02  -3.051 0.002282 **
## X.job."student"          4.358e-01  1.194e-01   3.651 0.000261 ***
## X.job."technician"      -2.115e-01  7.613e-02  -2.779 0.005459 **
## X.job."unemployed"      -2.401e-01  1.236e-01  -1.942 0.052090 .
## X.job."unknown"         -2.822e-01  2.551e-01  -1.106 0.268572
## X.marital."married"     -1.998e-01  6.526e-02  -3.062 0.002201 **
## X.marital."single"       7.819e-02  7.457e-02   1.049 0.294381
## X.education."secondary"  1.934e-01  7.191e-02   2.690 0.007155 **
## X.education."tertiary"   4.092e-01  8.341e-02   4.906 9.29e-07 ***
## X.education."unknown"    1.689e-01  1.164e-01   1.452 0.146616
## X.default."yes"          1.211e-02  1.791e-01   0.068 0.946101
## X.balance.               1.020e-05  5.938e-06   1.718 0.085797 .
## X.housing."yes"         -7.041e-01  4.870e-02 -14.458  < 2e-16 ***
## X.loan."yes"            -4.250e-01  6.671e-02  -6.370 1.89e-10 ***
## X.contact."telephone"   -1.430e-01  8.323e-02  -1.719 0.085662 .
## X.contact."unknown"     -1.567e+00  8.103e-02 -19.342  < 2e-16 ***
## X.day.                   8.975e-03  2.764e-03   3.248 0.001164 **
## X.month."aug"           -6.858e-01  8.744e-02  -7.843 4.40e-15 ***
## X.month."dec"            8.425e-01  1.974e-01   4.269 1.97e-05 ***
## X.month."feb"           -1.349e-01  9.908e-02  -1.361 0.173384
## X.month."jan"           -1.300e+00  1.357e-01  -9.584  < 2e-16 ***
## X.month."jul"           -7.735e-01  8.563e-02  -9.032  < 2e-16 ***
## X.month."jun"            3.909e-01  1.044e-01   3.744 0.000181 ***
## X.month."mar"            1.599e+00  1.329e-01  12.036  < 2e-16 ***
## X.month."may"           -3.704e-01  8.053e-02  -4.599 4.24e-06 ***
## X.month."nov"           -9.222e-01  9.448e-02  -9.761  < 2e-16 ***
## X.month."oct"            8.695e-01  1.204e-01   7.219 5.23e-13 ***
## X.month."sep"            8.811e-01  1.320e-01   6.676 2.45e-11 ***
## X.duration.              4.251e-03  7.189e-05  59.135  < 2e-16 ***
## X.campaign.             -9.975e-02  1.153e-02  -8.650  < 2e-16 ***
## X.pdays.                -5.347e-04  3.395e-04  -1.575 0.115319
## X.previous.              5.998e-03  6.325e-03   0.948 0.342975
## X.poutcome."other"       2.377e-01  9.911e-02   2.398 0.016478 *
## X.poutcome."success"     2.292e+00  9.111e-02  25.161  < 2e-16 ***
```

```
## X.poutcome."unknown"    -2.257e-01  1.022e-01  -2.209 0.027187 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 26746  on 37073  degrees of freedom
## Residual deviance: 17516  on 37031  degrees of freedom
## AIC: 17602
##
## Number of Fisher Scoring iterations: 6
```

Predicting using Logistic Regression and computing the accuracy

```
probs <- predict(glm1,newdata=alternative.df.test, type="response")
pred <- ifelse(probs>0.5,2,1)
pred <- as.factor(pred)
levels(pred) <- list("no"="1","yes"="2")
acc <- mean(as.integer(pred)==as.integer(alternative.df.test$X.y.))
print(paste("glm1 accuracy: ", acc))
```

```
## [1] "glm1 accuracy:  0.897013641391176"
```

```
table(pred, alternative.df.test$X.y.)
```

```
##
## pred    no  yes
##   no  6983  639
##   yes  199  316
```

KNN

First I had to change all factor variable to numeric variables in order to perform KNN on the data.

```
invisible({capture.output({


new.df<-df
new.df.train<-df.train
new.df.test<-df.test
new.df.trainLabels<-df.trainLabels
new.df.testLabels<-df.testLabels

for(i in 1:ncol(new.df)){
  if(is.factor(new.df[,i])){
    new.df[,i] <- as.numeric(new.df[,i])
  }
}
#str(new.df)

for(i in 1:ncol(new.df.train)){
  if(is.factor(new.df.train[,i])){
    new.df.train[,i] <- as.numeric(new.df.train[,i])
  }
}
#str(new.df.train)

for(i in 1:ncol(new.df.test)){
  if(is.factor(new.df.test[,i])){
    new.df.test[,i] <- as.numeric(new.df.test[,i])
  }
}
#str(new.df.test)

as.numeric(new.df.trainLabels)
as.numeric(new.df.testLabels)

})})
```

Using KNN and computing the accuracy

```
library(class)
df_pred <- knn(train=new.df.train, test=new.df.test, cl=new.df.trainLabels, k=15)
results <- df_pred == new.df.testLabels
acc <- length(which(results==TRUE)) / length(results)
# or combine into one line:
#acc <- length(which(iris_pred == iris.testLabels)) / length(iris_pred)
table(results, df_pred)
```

```
##        df_pred
## results   no  yes
##   FALSE  738  200
##   TRUE  6982  217
```

```
acc
```

```
## [1] 0.8847241
```

Decision Trees

Using rpart

```
library(rpart)
tree_df <- rpart(X.y.~., data=df, method="class")
tree_df
```

```
## n= 45211
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 45211 5289 no (0.88301520 0.11698480)
##    2) X.duration.< 521.5 40238 3106 no (0.92280928 0.07719072)
##      4) X.poutcome.="failure","other","unknown" 38941 2301 no (0.94091061 0.05908939)
*
##      5) X.poutcome.="success" 1297  492 yes (0.37933693 0.62066307)
##       10) X.duration.< 162.5 360  113 no (0.68611111 0.31388889) *
##       11) X.duration.>=162.5 937  245 yes (0.26147279 0.73852721) *
##    3) X.duration.>=521.5 4973 2183 no (0.56102956 0.43897044)
##      6) X.duration.< 827.5 3191 1147 no (0.64055155 0.35944845)
##       12) X.poutcome.="failure","other","unknown" 3047 1030 no (0.66196259 0.3380374
1) *
##       13) X.poutcome.="success" 144   27 yes (0.18750000 0.81250000) *
##      7) X.duration.>=827.5 1782  746 yes (0.41863075 0.58136925) *
```

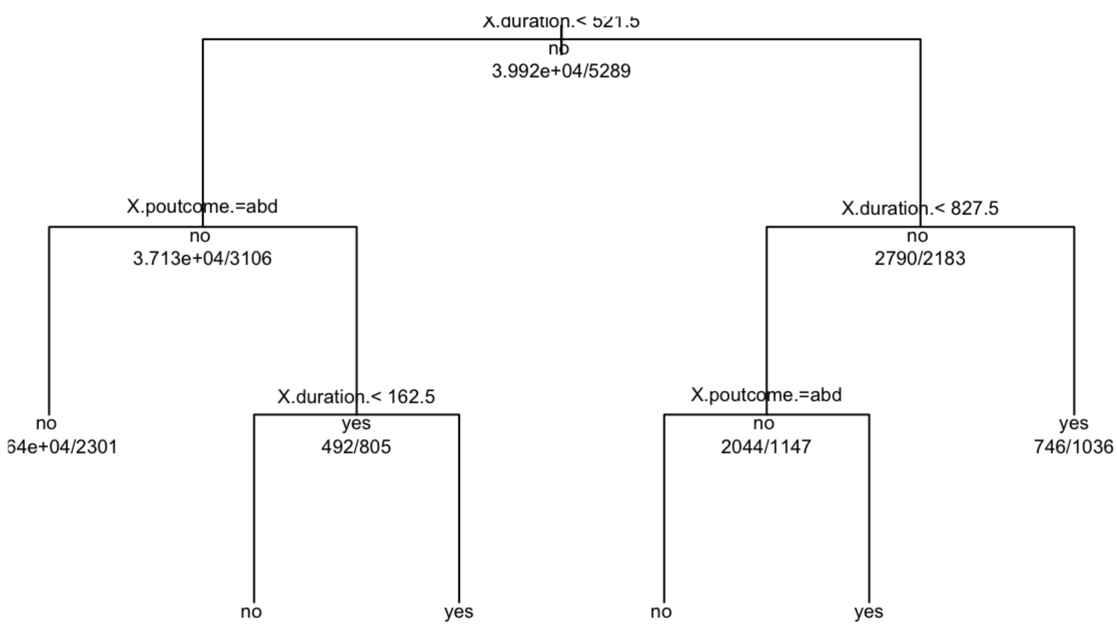
```
summary(tree_df)
```

```
## Call:
## rpart(formula = X.y. ~ ., data = df, method = "class")
##   n= 45211
##
##           CP nsplit rel error    xerror       xstd
## 1 0.03800340      0 1.0000000 1.0000000 0.01292104
## 2 0.02533560      3 0.8859898 0.8895822 0.01227563
## 3 0.01701645      4 0.8606542 0.8644356 0.01212074
## 4 0.01000000      5 0.8436377 0.8512006 0.01203794
##
## Variable importance
## X.duration. X.poutcome.
##          61          38
##
## Node number 1: 45211 observations,    complexity param=0.0380034
##   predicted class=no   expected loss=0.1169848  P(node) =1
##     class counts: 39922  5289
##    probabilities: 0.883 0.117
##   left son=2 (40238 obs) right son=3 (4973 obs)
##   Primary splits:
##       X.duration. < 521.5   to the left,   improve=1158.5880, (0 missing)
##       X.poutcome. splits as  LLRL,         improve= 879.1218, (0 missing)
##       X.month.    splits as  LLRLLLLRLLRR, improve= 518.1925, (0 missing)
##       X.pdays.    < 8.5     to the left,   improve= 265.4958, (0 missing)
##       X.previous. < 0.5     to the left,   improve= 261.3207, (0 missing)
##
## Node number 2: 40238 observations,    complexity param=0.0380034
##   predicted class=no   expected loss=0.07719072  P(node) =0.8900046
##     class counts: 37132  3106
##    probabilities: 0.923 0.077
##   left son=4 (38941 obs) right son=5 (1297 obs)
##   Primary splits:
##       X.poutcome. splits as  LLRL,         improve=791.6882, (0 missing)
##       X.month.    splits as  LLRLLLLRLLRR, improve=509.8801, (0 missing)
##       X.pdays.    < 9.5     to the left,   improve=254.5628, (0 missing)
##       X.previous. < 0.5     to the left,   improve=251.3338, (0 missing)
##       X.duration. < 205.5   to the left,   improve=228.0176, (0 missing)
##
## Node number 3: 4973 observations,    complexity param=0.0380034
##   predicted class=no   expected loss=0.4389704  P(node) =0.1099954
##     class counts:  2790  2183
##    probabilities: 0.561 0.439
##   left son=6 (3191 obs) right son=7 (1782 obs)
##   Primary splits:
##       X.duration. < 827.5   to the left,   improve=112.62690, (0 missing)
##       X.poutcome. splits as  LLRL,         improve= 61.04304, (0 missing)
##       X.contact.  splits as  RRL,          improve= 58.36740, (0 missing)
##       X.month.    splits as  LRRLLLLRLLRR, improve= 32.23210, (0 missing)
##       X.marital.  splits as  RLR,          improve= 25.82492, (0 missing)
##   Surrogate splits:
##       X.balance.  < -2385.5 to the right, agree=0.642, adj=0.001, (0 split)
##       X.campaign. < 23.5    to the left,  agree=0.642, adj=0.001, (0 split)
```

```
##       X.previous. < 17.5    to the left,  agree=0.642, adj=0.001, (0 split)
##       X.age.      < 88      to the left,  agree=0.642, adj=0.001, (0 split)
## 
## Node number 4: 38941 observations
##   predicted class=no   expected loss=0.05908939  P(node) =0.8613169
##     class counts: 36640  2301
##    probabilities: 0.941 0.059 
## 
## Node number 5: 1297 observations,    complexity param=0.0253356
##   predicted class=yes  expected loss=0.3793369  P(node) =0.02868771
##     class counts:   492   805
##    probabilities: 0.379 0.621 
##   left son=10 (360 obs) right son=11 (937 obs)
##   Primary splits:
##       X.duration. < 162.5   to the left,   improve=93.793010, (0 missing)
##       X.housing.  splits as  RL,           improve=17.230810, (0 missing)
##       X.month.    splits as  RRRRLRRRLLRR, improve=17.001180, (0 missing)
##       X.campaign. < 3.5     to the right,  improve= 9.391481, (0 missing)
##       X.job.      splits as  LLLLRRRRRLRR, improve= 8.978844, (0 missing)
##   Surrogate splits:
##       X.contact.  splits as  RRL,         agree=0.729, adj=0.025, (0 split)
##       X.campaign. < 6.5     to the right, agree=0.726, adj=0.014, (0 split)
##       X.pdays.    < 1.5     to the left,  agree=0.724, adj=0.006, (0 split)
##       X.default.  splits as  RL,          agree=0.723, adj=0.003, (0 split)
##       X.previous. < 14.5    to the right, agree=0.723, adj=0.003, (0 split)
## 
## Node number 6: 3191 observations,    complexity param=0.01701645
##   predicted class=no   expected loss=0.3594484  P(node) =0.07058017
##     class counts:  2044  1147
##    probabilities: 0.641 0.359 
##   left son=12 (3047 obs) right son=13 (144 obs)
##   Primary splits:
##       X.poutcome. splits as  LLRL,         improve=61.90733, (0 missing)
##       X.contact.  splits as  RRL,          improve=51.40018, (0 missing)
##       X.pdays.    < 0       to the left,   improve=31.42642, (0 missing)
##       X.previous. < 0.5     to the left,   improve=31.42642, (0 missing)
##       X.month.    splits as  LRRLLLLRLLRR, improve=30.51943, (0 missing)
## 
## Node number 7: 1782 observations
##   predicted class=yes  expected loss=0.4186308  P(node) =0.03941519
##     class counts:   746  1036
##    probabilities: 0.419 0.581 
## 
## Node number 10: 360 observations
##   predicted class=no   expected loss=0.3138889  P(node) =0.007962664
##     class counts:   247   113
##    probabilities: 0.686 0.314 
## 
## Node number 11: 937 observations
##   predicted class=yes  expected loss=0.2614728  P(node) =0.02072504
##     class counts:   245   692
##    probabilities: 0.261 0.739
```

```
##
## Node number 12: 3047 observations
##   predicted class=no   expected loss=0.3380374  P(node) =0.0673951
##     class counts:  2017  1030
##    probabilities: 0.662 0.338
##
## Node number 13: 144 observations
##   predicted class=yes  expected loss=0.1875  P(node) =0.003185066
##     class counts:    27   117
##    probabilities: 0.188 0.812
```

```
plot(tree_df, uniform=TRUE)
text(tree_df, use.n=TRUE, all=TRUE, cex=.6)
```



###

Using tree

```
#install.packages("tree")
library(tree)
tree_df2 <- tree(X.y.~., data=df)
tree_df2
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 45211 32630.0 no ( 0.883015 0.116985 )
##    2) X.duration. < 410.5 37668 18640.0 no ( 0.932383 0.067617 )
##      4) X.poutcome.: "failure","other","unknown" 36493 14570.0 no ( 0.949634 0.050366
)
##        8) X.month.: "aug","jan","jul","jun","may","nov" 30822  7908.0 no ( 0.971838
0.028162 )
##         16) X.duration. < 154.5 16152  1357.0 no ( 0.992942 0.007058 ) *
##         17) X.duration. > 154.5 14670  5945.0 no ( 0.948603 0.051397 )
##           34) X.contact.: "unknown" 5364   263.6 no ( 0.996271 0.003729 ) *
##           35) X.contact.: "cellular","telephone" 9306  5137.0 no ( 0.921126 0.078874
)
##             70) X.month.: "aug","jan","jul","may","nov" 8987  4285.0 no ( 0.935796 0.
064204 ) *
##             71) X.month.: "jun" 319   442.1 no ( 0.507837 0.492163 ) *
##        9) X.month.: "apr","dec","feb","mar","oct","sep" 5671  5189.0 no ( 0.828954 0.
171046 )
##         18) X.month.: "apr","feb" 4320  3039.0 no ( 0.887500 0.112500 ) *
##         19) X.month.: "dec","mar","oct","sep" 1351  1763.0 no ( 0.641747 0.358253 ) *
##      5) X.poutcome.: "success" 1175  1578.0 yes ( 0.396596 0.603404 ) *
##    3) X.duration. > 410.5 7543  9888.0 no ( 0.636484 0.363516 )
##      6) X.duration. < 647.5 4351  4929.0 no ( 0.746265 0.253735 ) *
##      7) X.duration. > 647.5 3192  4423.0 yes ( 0.486842 0.513158 ) *
```

```
summary(tree_df2)
```

```
##
## Classification tree:
## tree(formula = X.y. ~ ., data = df)
## Variables actually used in tree construction:
## [1] "X.duration." "X.poutcome." "X.month."    "X.contact."
## Number of terminal nodes:  9
## Residual mean deviance:  0.4884 = 22080 / 45200
## Misclassification error rate: 0.1098 = 4962 / 45211
```

```
plot(tree_df2)
text(tree_df2, cex=0.5, pretty=0)
```

train and test

```
set.seed(1234)
i <- sample(1:nrow(df), nrow(df)*0.8, replace=FALSE)
train <- df[i,]
test <- df[-i,]
tree_df3 <- tree(X.y.~., data=train)
pred <- predict(tree_df3, newdata=test, type="class")
table(pred, test$X.y.)
```

```
##
## pred    no  yes
##   no  7593  524
##   yes  411  515
```

```
mean(pred==test$X.y.)
```

```
## [1] 0.8966051
```

Conclusion

Comparing the results: I noticed that the accuracy was very similar using the 3 different techniques. Logistic regression had an accuracy of about 90%. KNN had an accuracy of 88%. And decision trees had an accuracy of about 90%. Providing some analysis on why the results were most likely achieved given how the algorithms work: There can be differences in the accuracies because of several reasons. The first is that these algorithms work based on different assumptions and model data using different approaches. Like logistic regression assumes a linear relationship but on the other hand decision trees can capture non-linear relationships. Therefore if the relationship is not linear, decision trees can outperform logistic regression. In our case that did not happen as both gave similar accuracies. It is also important to note that KNN performs well on well-defined clusters and logistic regression works better with linearly separable data. There is also a possibility that the choice of K in KNN was not the best causing it to not perform as well as the other two. There is also a possibility of overfitting.