Logistic-Multinomial Regression

Since Logistic Regression pertains to the use of simply 2 variables as the outcome of a classification model, I had to use Multinomial Logistic Regression (function multinom) which is a part of the library "Nnet" to predict one of the 10 categories available for the rent data.

```
##Libraries--------------------------------------------
library(tree)
library(ISLR)
library(boot)
library(xgboost)
library(tidyverse)

## ── Attaching packages
─────────────────────────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.1.0     ✔ purrr   0.3.0
## ✔ tibble  2.0.1     ✔ dplyr   0.7.8
## ✔ tidyr   0.8.2     ✔ stringr 1.4.0
## ✔ readr   1.3.1     ✔ forcats 0.3.0

## ── Conflicts ──────────────────────────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ✖ dplyr::slice()  masks xgboost::slice()

library(leaflet)
library(stringr)
library(rgdal)

## Loading required package: sp

## rgdal: version: 1.4-3, (SVN revision 828)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.1.3, released 2017/20/01
##  Path to GDAL shared files:
/Library/Frameworks/R.framework/Versions/3.5/Resources/library/rgdal/gdal
##  GDAL binary built with GEOS: FALSE
##  Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
##  Path to PROJ.4 shared files:
/Library/Frameworks/R.framework/Versions/3.5/Resources/library/rgdal/proj
##  Linking to sp version: 1.3-1

library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```r
library(forecast)
library(DT)
library(prophet)
```

```
## Loading required package: Rcpp
```

```
## Loading required package: rlang
```

```
##
## Attaching package: 'rlang'
```

```
## The following objects are masked from 'package:purrr':
##
##     %@%, as_function, flatten, flatten_chr, flatten_dbl,
##     flatten_int, flatten_lgl, flatten_raw, invoke, list_along,
##     modify, prepend, splice
```

```r
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(fastDummies)
library(caret)
library(class)
library(e1071)
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(neuralnet)
```

```
##
## Attaching package: 'neuralnet'
```

```
## The following object is masked from 'package:ROCR':
##
##     prediction
```

```
## The following object is masked from 'package:dplyr':
##
##     compute
```

```
library(nnet)


##Read the data
mergedf <-
read.csv("~/Desktop/GitAdd/Data_Mining/Files/Outputs/CrimeRentData.csv")


##Remove columns
mergedf <- within(mergedf, rm("X"))
mergedf <- within(mergedf, rm("AreaName"))
mergedf <- within(mergedf, rm("CrimeCodeDescription"))
mergedf <- within(mergedf, rm("DRNumber"))
mergedf <- within(mergedf, rm("Location.x"))
mergedf <- within(mergedf, rm("PremiseDescription"))
mergedf <- within(mergedf, rm("Variable"))
mergedf <- within(mergedf, rm("Location.y"))
mergedf <- within(mergedf, rm("Date"))
mergedf <- within(mergedf, rm("VictimDescent"))


mergedf = na.omit(mergedf)
mergedf$Year <- as.factor(mergedf$Year)
mergedf$CrimeCode <- as.factor(mergedf$CrimeCode)
mergedf$Neighborhood <- as.factor(mergedf$Neighborhood)


##Create dummy var
dff <- dummy_cols(mergedf, select_columns =
c("CrimeCode","Year","Neighborhood") )

##Remove Dummy Columns
dff<-within(dff, rm("CrimeCode"))
```

```r
dff<-within(dff, rm("Neighborhood"))
dff<-within(dff, rm("Year"))

##Remove extra columns
dff<-within(dff, rm("DateOccurred"))
dff<-within(dff, rm("Tract"))
dff<-within(dff, rm("ReportingDistrict"))
dff<-within(dff, rm("TimeOccured"))
dff<-within(dff, rm("VictimAge"))
dff<-within(dff, rm("VictimSex"))
dff<-within(dff, rm("PremiseCode"))


mergedf <- dff

##Create Amount categories

mergedf$Amount <- as.numeric(mergedf$Amount)
ra <- range(mergedf$Amount)
div <- (ra[2]-ra[1])/10
ini <- ra[1]
br <- rep(0,11)
br[1]<-ra[1]
for(i in 2:11){
  ini<-ini+div
  br[i]<-ini
}

mergedf$Renth <- cut(mergedf$Amount,
                     breaks=br,
                     labels=c("1","2","3","4","5","6","7","8","9","10"))
mergedf <- within(mergedf,rm("Amount"))

##Sample
train <- sample(nrow(mergedf),0.7*nrow(mergedf))
traindata <- data.frame(mergedf[train,])
testdata <- data.frame(mergedf[-train,])


##Neural Network - Multinomial Logistic regression
fit1<- nnet::multinom(Renth~., data=traindata, MaxNWts = 50000)
```

```
## # weights:  3510 (3150 variable)
## initial  value 1508936.970905
## iter  10 value 675514.293589
## iter  20 value 613805.723973
## iter  30 value 598021.725161
## iter  40 value 590348.162216
## iter  50 value 585748.852395
## iter  60 value 580333.917912
## iter  70 value 574204.596932
## iter  80 value 568281.523202
## iter  90 value 564258.184040
## iter 100 value 562725.178542
## final  value 562725.178542 
## stopped after 100 iterations
```

```
##Confusion Matrix
ad <- testdata$Renth
pd<- predict(fit1,newdata=testdata)
(table1 <- table(ad,pd))
```

```
##     pd
## ad        1      2      3      4      5      6      7      8      9     10
##   1    2949    347   1177     39   1005      5      9      0      0      0
##   2     905  13215  17936    582   1073    277      9     32      0      0
##   3      87   6057 123689   5485    419    978     26     30      1      0
##   4    1025    895  23607  20369   3711   3277    106     32      1      0
##   5    1383    255   3794   7198   7978   4508     90     41      0      0
##   6     243    143   4348   5210   2738  12093     71      3      0      0
##   7       0      0     33    170    193    213    184     73      0      0
##   8       0      0     15      1      0    105      8     82      0      0
##   9       0      0      1     50     30     50     13      0      0      0
##   10      0      0      0     12    115     55      6      7      0      0
```

```
##Accuracy
sum=0
for(i in 1:10){

  sum=sum+table1[i,i]
}

(acc= sum/nrow(testdata))
```

```
## [1] 0.6427692
```