Machine Learning Challenge

Ali Baloch

October 24, 2017

Case Study & Model Selection

As we have 143 weeks data of customers visits in mall, and after informal analysis of given data-set specially “visits” column in train_set.csv file, it is clear that data is of discrete type and after transformation of data, it is stated that data has a variable “visits” which can take any value between 0 to 7 (weekdays), so we have to use any of the following discrete probability distributin method to get next possible visit of any particular customer i.e.

1. Bionomial probability distribution
2. Hypergeomatric probability distribution
3. Multinomial probabilty distribution
4. Negative Bionomial distribution

1. Binomial probability distribution: It’s experiments can consits of any repeated trials but works for a case in which any two possible outcomes of it’s trails are possible, so by any chance it can’t be used in our case as we have 0 to 1 possible outcomes.

2. Hypergeomatry Probabilty distribution: It focus on selecting random sized values from a dataset/sample without their replacement in dataset population and tend to classify these selected values as success or failure, which is probabily not suitable for our case, as it needs a lot of complex operation on our dataset transformation with unclear and unoptimized outcomes.

3. Negative Probability distribution: It only focus on each trail/variable who must have two outcomes, success or failure but trails are independent, and it is good for disccrete classification with only two possible outcomes.

4. Multinomial probability Distribution: An extended form of Binomial probabilty distribution,

it's popular due to it's commonly known example of dice rolling, it is good for dataset who has multiple finite known possible outcomes and also its trails are independent and it is very effective for multi-lable data and its classification.
So it's been decided that Multinomial Probability distribution Method is best fit for our dataset

Environment Setup

```
library(dplyr)
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
library(nnet)
library(reshape2)
library(ggplot2)
```

Data Files Reading

```
train_data_set= read.table(file="train_set.csv", header = TRUE,
sep=",")
head(train_data_set,10)
test_data_set = read.table(file="test_set.csv", header=TRUE, sep=",")
head(test_data_set,10)
```

Data Tranformation

```
train_data_set$visits <- as.character(train_data_set$visits)

   #vectorizatin of visits column
Sunday = vector(mode="numeric", length=nrow(test_data_set))
Monday = vector(mode="numeric", length=nrow(test_data_set))
Tuesday = vector(mode="numeric", length=nrow(test_data_set))
Wednesday = vector(mode="numeric", length=nrow(test_data_set))
Thursday = vector(mode="numeric", length=nrow(test_data_set))
Friday = vector(mode="numeric", length=nrow(test_data_set))
Saturday = vector(mode="numeric", length=nrow(test_data_set))
```

```
avg_visit_day_gap = vector(mode="numeric", length=nrow(test_data_set))
visits_total = vector(mode="numeric", length=nrow(test_data_set))
days_since_last_visit = vector(mode="numeric",
length=nrow(test_data_set))

# mapping visits column data into weekdays
for (i in 1:nrow(train_data_set)) {
  visits = unlist(strsplit(train_data_set[i,2]," "))
  visits_gaps = 0
  visits_total[i] = length(visits)-1
  size_of_visits = length(visits)
  for(j in 2:size_of_visits){
  current_visit_day = as.numeric(visits[j])
    if(current_visit_day%%7 == 0){
      Sunday[i] = Sunday[i]+1
    } else if (current_visit_day%%7 == 1){
      Monday[i] = Monday[i]+1
    } else if (current_visit_day%%7 == 2){
      Tuesday[i] = Tuesday[i]+1
    } else if (current_visit_day%%7 == 3){
      Wednesday[i] = Wednesday[i]+1
    } else if (current_visit_day%%7 == 4){
      Thursday[i] = Thursday[i]+1
    } else if (current_visit_day%%7 == 5){
      Friday[i] = Friday[i]+1
    } else if (current_visit_day%%7 == 6){
      Saturday[i] = Saturday[i]+1
    }

    #calculating average gap between visits
    if(j!=2){
      last_visit_day = as.numeric(visits[j-1])
      visits_gaps = visits_gaps + ( current_visit_day -
last_visit_day)
      avg_visit_day_gap[i] = visits_gaps/(size_of_visits-2)
    }

  }
  #claculating
  days_since_last_visit[i] = 1001-as.numeric(visits[size_of_visits])
}

transf_df <- data.frame(visitor_id = train_data_set$visitor_id,
Sunday, Monday,
                  Tuesday, Wednesday, Thursday, Friday, Saturday,
                    avg_visit_day_gap, visits_total,
days_since_last_visit,
                      next_visit_day = test_data_set$next_visit_day
```

```
                  )
head(transf_df, 10)
```

Data Normalization

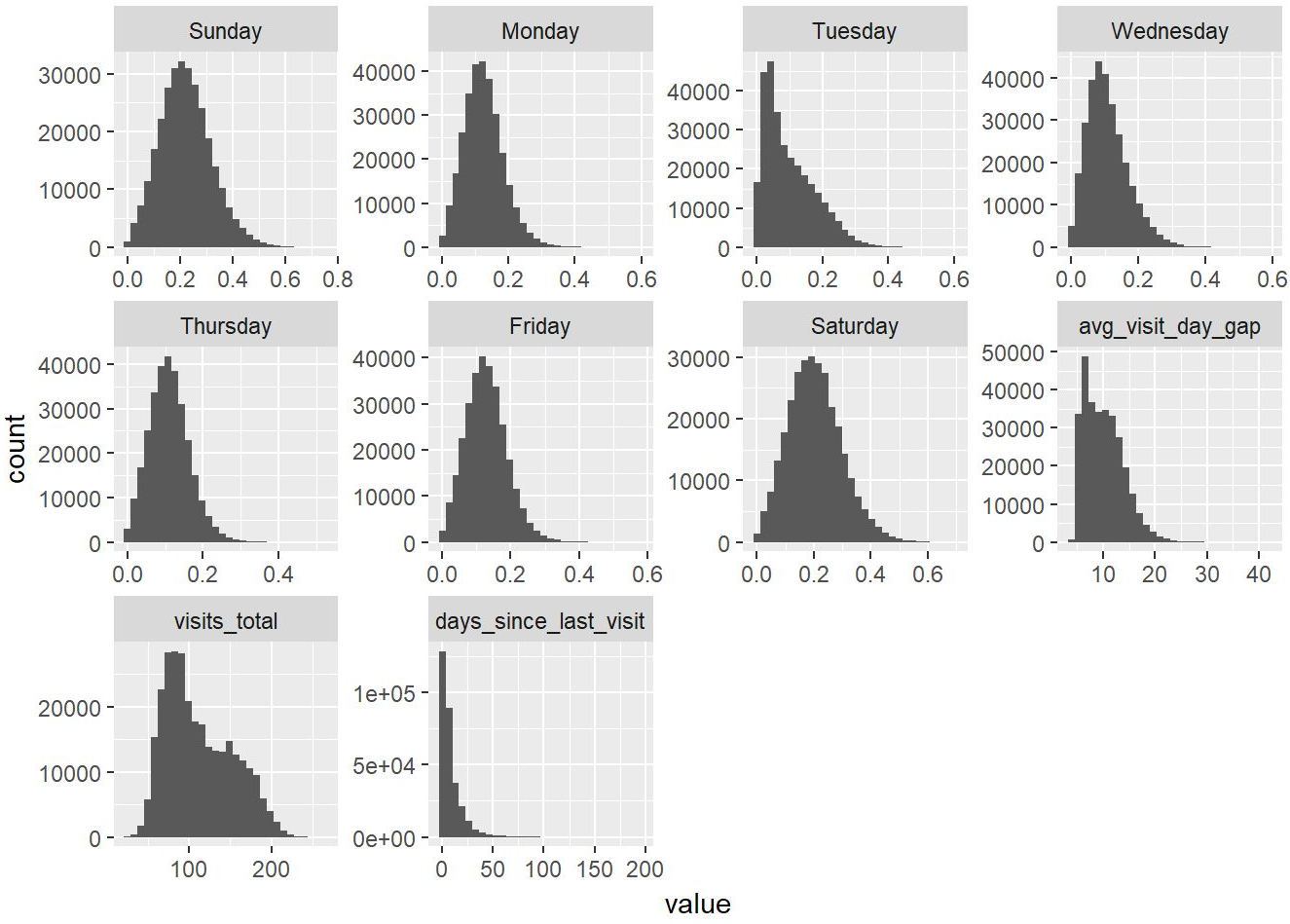
```
transf_df$Sunday = transf_df$Sunday / transf_df$visits_total
transf_df$Monday = transf_df$Monday / transf_df$visits_total
transf_df$Tuesday = transf_df$Tuesday / transf_df$visits_total
transf_df$Wednesday = transf_df$Wednesday / transf_df$visits_total
transf_df$Thursday = transf_df$Thursday / transf_df$visits_total
transf_df$Friday = transf_df$Friday / transf_df$visits_total
transf_df$Saturday = transf_df$Saturday / transf_df$visits_total

transf_df$next_visit_day = factor(transf_df$next_visit_day)

transf_df$next_visit_day  = relevel(transf_df$next_visit_day, ref="0")
```

Visualization of Customers Visit(Histograms)

```
plot_hitograms = function(data){
  d <- melt(data)
  ggplot(d,aes(x = value)) +
    facet_wrap(~variable,scales = "free") +
    geom_histogram()
}
plot_hitograms(select(transf_df, 2:11))
## No id variables; using all as measure variables
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Train/Test Data Spliting

```
#70/30 training and testing random split
train<-sample_frac(transf_df, 0.7)
sid<-as.numeric(rownames(train))
test<-transf_df[-sid,]
```

Data Modeling

```
model_weekdays = multinom(next_visit_day ~ avg_visit_day_gap +
visits_total
                          + days_since_last_visit + Sunday + Monday +
Tuesday
                          + Wednesday + Thursday + Friday + Saturday,
                         data = train)
## # weights:  96 (77 variable)
```

```
## initial  value 436682.723754
## iter  10 value 410157.806265
## iter  20 value 408724.512416
## iter  30 value 395402.653421
## iter  40 value 394131.791422
## iter  50 value 392854.750622
## iter  60 value 392290.063670
## iter  70 value 392197.244160
## iter  80 value 392182.817622
## iter  90 value 392175.103436
## final  value 392174.517312
## converged
summary(model_weekdays)
## Call:
## multinom(formula = next_visit_day ~ avg_visit_day_gap +
visits_total +
##     days_since_last_visit + Sunday + Monday + Tuesday + Wednesday +
##     Thursday + Friday + Saturday, data = train)
##
## Coefficients:
##   (Intercept) avg_visit_day_gap visits_total days_since_last_visit
## 1    2.587240       -0.04765037   0.04243888           -0.01495714
## 2    2.469895       -0.04672153   0.04208751           -0.01391397
## 3    2.550745       -0.05015928   0.04080894           -0.01276733
## 4    2.513057       -0.05398828   0.04027703           -0.01393786
## 5    2.550346       -0.05488452   0.03918718           -0.01387742
## 6    2.860122       -0.06834290   0.03771772           -0.01416056
## 7    3.054614       -0.07346942   0.03547466           -0.01405730
##       Sunday    Monday    Tuesday Wednesday  Thursday    Friday
Saturday
## 1 -0.1565825  2.582365  0.4142369 -2.085846 -2.398080 1.1964507
3.034697
## 2 -1.0183354 -1.409442  5.7050620 -1.063934 -2.331901 0.4658788
2.122566
## 3 -0.3477295 -1.864468  0.9400392  2.067710 -2.219706 1.1304436
2.844455
## 4  0.1346955 -1.824801 -0.2418917 -2.642592  2.383325 1.2664894
3.437831
## 5  0.4668861 -1.749112 -0.4474497 -2.429309 -2.611608 5.5921770
3.728761
## 6  0.9260112 -1.591768 -0.2296239 -2.394088 -2.334788 1.4145607
7.069818
## 7  3.6154042 -1.360258  0.1088694 -2.358619 -2.220838 1.5781397
3.691916
##
## Std. Errors:
##   (Intercept) avg_visit_day_gap visits_total days_since_last_visit
## 1  0.06273863        0.01335101  0.002793624           0.004858243
```

```
## 2  0.07205967        0.01347086  0.002801347           0.004862660
## 3  0.07120665        0.01347418  0.002799737           0.004862060
## 4  0.07301527        0.01353732  0.002798801           0.004869062
## 5  0.07195131        0.01352660  0.002797553           0.004868501
## 6  0.06562071        0.01342991  0.002793204           0.004860710
## 7  0.06820958        0.01345159  0.002795845           0.004860498
##       Sunday     Monday    Tuesday  Wednesday   Thursday     Friday
## 1 0.04869383 0.07537234 0.07010696 0.08104334 0.08457891 0.07454617
## 2 0.06060821 0.08681846 0.07234650 0.08765050 0.09724084 0.08949807
## 3 0.05653201 0.08708510 0.07713937 0.08671746 0.09558839 0.08559190
## 4 0.05488513 0.09011325 0.08348947 0.09572975 0.09402208 0.08517355
## 5 0.05280256 0.08860148 0.08348247 0.09475065 0.09556816 0.07923701
## 6 0.04612336 0.07878198 0.07560416 0.08521192 0.08445974 0.07275510
## 7 0.04584228 0.08061945 0.07632350 0.08680454 0.08677371 0.07512696
##     Saturday
## 1 0.05105833
## 2 0.06380653
## 3 0.05937069
## 4 0.05739833
## 5 0.05532481
## 6 0.04644522
## 7 0.05062057
##
## Residual Deviance: 784349
## AIC: 784489
predictions_prob = predict(model_weekdays, test, type="prob")
predictions_class = predict(model_weekdays, test, type="class")

head(predictions_prob)
##              0         1          2          3          4
5
## 4  0.020546435 0.2258540 0.12901630 0.11396119 0.09183634
0.25670254
## 8  0.017306461 0.1699460 0.14605138 0.25559419 0.13923192
0.06101905
## 12 0.003326970 0.1773886 0.15361652 0.11521845 0.09019813
0.12229986
## 18 0.009661837 0.2062049 0.10250648 0.09723162 0.06062390
0.09218992
## 19 0.017638109 0.3192490 0.17325217 0.14646280 0.06444500
0.07908134
## 23 0.005046692 0.2780451 0.07776082 0.08932540 0.07699206
0.16277268
##             6          7
## 4  0.08545795 0.07662531
## 8  0.14902920 0.06182175
## 12 0.23601147 0.10193999
## 18 0.09371362 0.33786776
```

```
## 19 0.10349307 0.09637855
## 23 0.21895865 0.09109858
head(predictions_class)
## [1] 5 3 6 7 1 1
## Levels: 0 1 2 3 4 5 6 7
```

Confusion/Evaluation Matrix

```
predictions = predict(model_weekdays, test)

(confusion_matrix = table(predictions,test$next_visit_day))
##
## predictions    0    1    2    3    4    5    6    7
##           0    0    0    0    0    0    0    0    0
##           1    8 5223 3343 3204 2782 2772 3443 3188
##           2    4 2157 2870 1817 1371 1250 1440 1485
##           3    4  719  757  960  482  483  546  602
##           4    6  340  289  272  392  294  320  346
##           5    1  515  302  376  412  648  560  569
##           6    5 4156 2034 2852 3393 3907 7092 4583
##           7    8 2254 1220 1455 1807 2068 2626 3988
(Misclassification =
1-sum(diag(confusion_matrix))/sum(confusion_matrix))
## [1] 0.7647444
```

Significance Test

```
significance=
summary(model_weekdays)$coefficients/summary(model_weekdays)$standard.
errors
p = (1 -pnorm(abs(significance),0,1)) * 2
p
##   (Intercept) avg_visit_day_gap visits_total days_since_last_visit
## 1           0      3.582822e-04            0           0.002078955
## 2           0      5.236827e-04            0           0.004217865
## 3           0      1.971654e-04            0           0.008641756
## 4           0      6.660269e-05            0           0.004202662
## 5           0      4.959525e-05            0           0.004365752
## 6           0      3.602245e-07            0           0.003576657
## 7           0      4.714218e-08            0           0.003826137
##         Sunday Monday      Tuesday Wednesday Thursday      Friday
Saturday
## 1 1.301474e-03      0 3.449403e-09         0        0 0.00000e+00
0
```

```
## 2 0.000000e+00      0 0.000000e+00         0        0 1.93515e-07
0
## 3 7.698597e-10      0 0.000000e+00         0        0 0.00000e+00
0
## 4 1.412244e-02      0 3.764227e-03         0        0 0.00000e+00
0
## 5 0.000000e+00      0 8.331220e-08         0        0 0.00000e+00
0
## 6 0.000000e+00      0 2.387979e-03         0        0 0.00000e+00
0
## 7 0.000000e+00      0 1.537470e-01         0        0 0.00000e+00
0
```

Model Evaluation:

Model used in this case is a Multinomial model with the power of neural network in logistic regression, we say it a regression with one layer neural network.
As logistic regression can be generalize to 'k' number of classes(more than two) with the help of multinomial logistic regression, as we have more than two classes i.e. total 8 classes (7 weekdays visits, and 1 no visit at all). So multinomial logistic model via neural network (vectorize inputs) been used to fit the dataset, as it allows to predict a factor of multiple level in one shot. Our data is neural in neural in nature so let neural network to overcome the data comparison and concatenation of probability. "multinorm" function will do the all and allow us to observe the probability of each subset to interpret our data.
Before feeding our data to multinorm function the data have been transformed into 7 weekdays and their standard derived attributes, which after have been splits into 70/30 ratio as a train test split, so that data can be tested on real training data, not a self assumed testing data, due to this best possible generalization level can be attain and can avoid over-fitting and under-fitting of model in an optimized way.

Result Evaluation:

At Data Modelling phase , 'maxiter' variable is set to its default iteration value i.e. 100, and at 100th iteration it gives a final value- a global minima which is 392231.566 with the neural network nodes weight of 96 solution with lowest possible error and gives the "Converged" signal, which shows that model went as far as it could, In last three iteration, the outcome value's first 4 decimal number before point "3922#.####" show the progress of our model numerical computation towards global minima. We have two prediction outcome forms "class" and "prob".
in **class** outcome is 0 to 7 levels of classes, which means we have 8 level predictions which cover all weekdays and no visit at all attrribute.
From **Prob** outcomes we can observe and analyse the predictions for each class associated to it.
From **summary(model_weekdays)** method a judgements can be made form the standard

error of error of each visitor_id against each week-day visit probability and its derived attributes. as next_visit_day is dependent variable and all weekdays and their derived attributes is independent variable, so the comparsion of odds between independent vvaribales against dependent and continous varibale.
Coefficient size of each independent variable in our dataset determines the expected increase or decrease of our dependent variable, as our coefficients outcomes are balance with both negative and positive values which show the coefficients do not have the straightforward interpretation which needs to discussed and analyse in a more advance way. In **standard Errors** in which estimated values in the regression line are being compared to the actual values and the difference between them show our model optimization level by using standard erorr formula on each prediction, if standard value is accurate 0.000 it means our model is overfit and if it is some higher value closer to 1, it shows the under fitting of our model, and it can seen that standard error is closer to zero in our case which show the accuracy of the model.
From **Confusion Matrix** each number of prediction(successful and unsuccessful) agaisnt its relative class (0 to 7) can be analyzed and observed, e.g. for “1” class/level we can see that our model predict 1 zero times when it should predict 0, and predict 1 5296 times when it should predict 1, predict 1, 3476 times when it should have to predict 2, and so on for others. From **Significance** it can be observed that the p-value is been tested from the confidence level 95 % and values against the intercepts coefficient divided by its standard error of each independent variable in a row which has p value outcome that is very low and less than .00 which shows the higher significance confidence level. as it can be observed that the lowest p-values can be found on sunday, Tuesdays, and Friday, which show these independent varibales have high effect strong association on our depend variable next_day_visit probability.