Reddit Analysis

M. Nelson

November 16, 2015

Problem Statement

Submissions to Reddit are scored based on other user's up-votes and down-votes. The challenge is to select and train on a set of features that will result in set of submission rules that will improve the chance of receiving a high score.

Overall Approach

Gather set of high scoring (successful) submissions for the training set. Select a set of features and run a clustering algorithm to detect groupings of combinations of these features. Define a range around each cluster centroid as a selection rule. Run a data test set against these rules to select a target subset of submissions. Test whether the mean score of submissions in this target subset is greater than the mean score of all submissions in the test set.

Detailed Steps

1. Gather top submissions over the past year from the "all" subreddit. Determine the subreddits with the highest count of appearances in this set. Rerun to gather the top submissions over the past year from each of those most successful subreddits.

```
import praw
r = praw.Reddit('Getting from subreddit 01')

#subreddits = ["all"]
subreddits = ["BlackPeopleTwitter", "pics", "me_irl", "funny", "videos", "gifs", "4chan",
"todayilearned", "worldnews", "WTF", "aww", "technology", "Showerthoughts"]
for subred in subreddits:

    submissions = r.get_subreddit(subred).get_top_from_year(limit=1000)
    filename = "YearTop-" + subred + ".txt"

    f = open(filename, 'w')
    f.write("fullname\tcreated_utc\tscore\tdomain\tsubreddit\ttitle\n")

    for submission in submissions:
        fullname = str(submission.fullname)
        created_utc = str(submission.created_utc)
        domain = str(submission.domain)
        score = str(submission.score)
        subreddit = str(submission.subreddit)
        title = str(submission.title)
        try :
            f.write(fullname + "\t" + created_utc + "\t" + score + "\t" + domain + "\t" +
subreddit + "\t" + title + "\n")
        except :
            pass

    f.close()
```

2. It is beyond the scope of this project to analyze the actual content of the submitted links. So features will be chosen from data available in the submission object only. Features selected are:

a. Day of submission

b. Time of submission

c. Length of the submission title

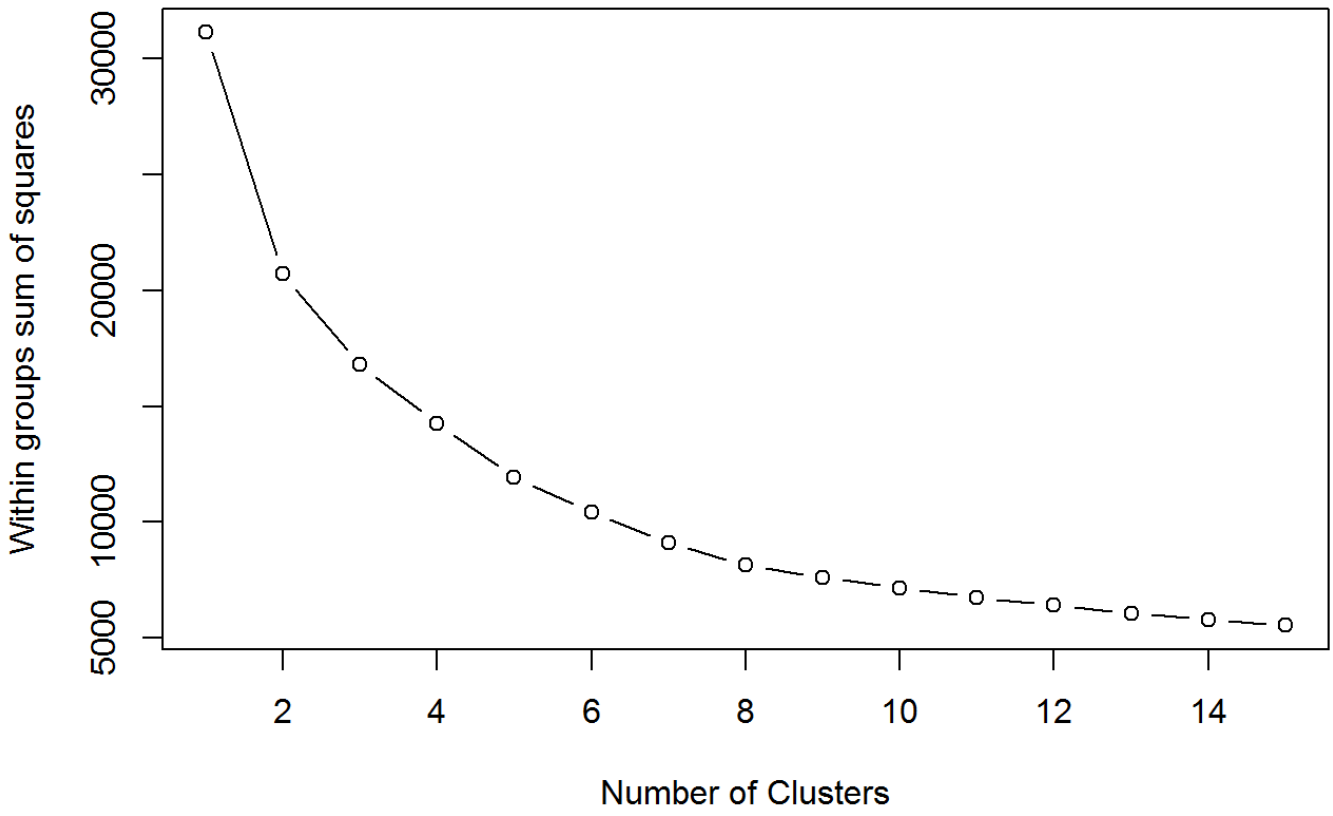
d. Estimate of whether the content is a picture, video, or text based on the domain of the url in the submission. (Note - this is assumed as an ordered factor based on the amount of effort for the user to absorb the content. Pictures require very little effort. Videos take a little longer. Text requires the most effort to read.)

3. Read the data into R and manipulate for features. Divide this data into 60% for training set and 40% for test set. NOTE: Ideally would have used a test set that was a random selection of all submissions (not a random selection of top submissions). Could not learn how to gather that data. So this test is actually trying to select best of the best and not the best of all submissions.

```
#########################################################################
# Read data. Manipulate for features.
srdata <- lapply(Sys.glob("./Data/YearTop-*.txt"), read.table, header = TRUE, sep = "\t",
quote = "", colClasses = c("factor", "numeric", "integer", "character", "factor", "charac
ter"))
srdata <- do.call("rbind", srdata)
srdata <- mutate(srdata, date = .POSIXct(srdata$created_utc, tz="UTC"))
srdata <- mutate(srdata, day = as.integer(strftime(date, format="%w")))
srdata <- mutate(srdata, time = as.integer(strftime(date, format="%H")))
srdata <- mutate(srdata, type = as.integer(ifelse(domain %in% c("i.imgur.com", "imgur.co
m"), 1,
                                          ifelse(domain %in% c("youtube.com", "youtu.be", "g
fycat.com"), 2,
                                                 3))))
srdata <- mutate(srdata, length = nchar(title))

# Divide train (60%) and test (40%) sets.
set.seed(12345)
srdata <- mutate(srdata, train = ifelse((runif(nrow(srdata)) < 0.6), 1, 0))
trdata <- filter(srdata, train == 1)
tedata <- filter(srdata, train == 0)
```

4. Select the feature columns and scale and center it. Run the train data through a series of kmeans tests to plot cost of each run by count of clusters used. Manually select the number of clusters to use from this plot.

```
#########################################################################
# Scale features, run kmeans, and plot versus cluster count. Then manually select count.
rawfeatures <- select(trdata, day, time, type, length)
features <- scale(rawfeatures)

wss <- (nrow(features)-1)*sum(apply(features,2,var))
for (i in 2:15) wss[i] <- sum(kmeans(features, centers=i, nstart = 25, iter.max = 15)$wit
hinss)

plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares")
```



5. Run kmeans with the selected number of clusters against the train set. Reverse the scale/center operation on the resulting centroids. Round the values of the centroid results to obtain a table of selection rules. Apply the rules to the test set. Calculate the mean score of the selected subset and compare to the mean score of the entire test set.

```
clustercount <-3

############################################################
# Find result rule table from running kmeans with chosen cluster count.
# Unscale and create selection rules.
fit <- kmeans(features, centers=clustercount, nstart = 25)
for (i in 1:ncol(features)) {
    fit$centers[,i] <- fit$centers[,i] * sd(rawfeatures[,i]) + mean(rawfeatures[,i])
}
results <- round(as.data.frame(fit$centers))
results <- arrange(results, day, time, type, length)



##########################################################
# Check test data against rules.
# Read test data.

# Indicate in selection column if submission meets the selection rules.
tedata <- select(tedata, score, day, time, type, length)
tedata <- mutate(tedata, selected = 0)
for (i in 1:nrow(results)) {
    rday <- results[i,]$day
    rtime <- results[i,]$time
    rtype <- results[i,]$type
    rlength <- results[i,]$length
    tedata <- mutate(tedata, selected = ifelse(
        (tedata$day == rday) &
        (tedata$time ==rtime) &
        (tedata$type == rtype) &
        (tedata$length > (rlength * 0.8)) &
        (tedata$length < (rlength * 1.2)), i, tedata$selected))
}



# Print summary results
seltd <- filter(tedata, selected > 0)
selmean <- mean(seltd$score)
nonseltd <- filter(tedata, selected == 0)
nonselmean <- mean(nonseltd$score)

print(paste("Non Selected mean = ", round(nonselmean)))
```

```
## [1] "Non Selected mean =  5460"
```

```
print(paste("Only Selected mean = ", round(selmean)))
```

```
## [1] "Only Selected mean =  6690"
```

```
for (i in 1:nrow(results)) {
    print(paste("Rule:", i, " = ", round(mean(filter(seltd, selected == i)$score))))
}
```

```
## [1] "Rule: 1  =  6678"
## [1] "Rule: 2  =  4502"
## [1] "Rule: 3  =  7812"
```

```
print(results)
```

```
##   day time type length
## 1   1   10    1     37
## 2   3   10    3    129
## 3   5   12    1     38
```

```
# Run T-test to check for statistically significant difference
print(t.test(seltd$score, nonseltd$score,
             alternative="two.sided",paired=FALSE,var.equal=TRUE)$conf)
```

```
## [1]  230.0888 2227.9294
## attr(,"conf.level")
## [1] 0.95
```

Conclusion

A hypothesis test is done using a T-test with 95% confidence interval. The null hypthesis is:

H0 = There is no difference in the mean scores of the selected and non-selected submissions.

The 95% confidence interval of the difference in means is 230 - 2228. Because zero is not within that confidence interval, the null hypothesis that there is no difference between the means of the two groups of submissions is rejected. Within the confidence of this hypothesis, the selection rules discovered does choose submissions with improved scores.