Andy Zhu - STAT196K

```
* Full code provided in bottom of document
```

Clustering

Resources

1. Google machine learning
2. Julia MultivariateStats

Data

The data is available in S3, and the following command will download it.

```
aws s3 cp s3://stat196k-data-examples/processed990.zip ./ --no-sign-request
```

Alternatively, you can download it from the browser: processed990.zip.

This zip file contains three Julia files containing data we produced in the [XML to matrix assignment]({% link _posts/2021-03-11-homework-xml-to-matrix.md %}).

1. `irs990extract.jldata` contains summary statistics computed from each of the XML files.
2. `terms.jldata` contains all the terms in the term document matrix.
3. `termfreq.jldata` contains the term frequency matrix. The rows correspond to the indexes of `irs990extract`, the XML files. The columns correspond to the indexes of `terms`, the dictionary of all possible words. Each row represents the relative frequency of each word in the mission statement.

Once you unzip the file, you can load each object into Julia as follows.

```
terms = Serialization.deserialize("terms.jldata")
```

1 Exploratory Data Analysis

(5 pts)

1. **Relatively how many terms appear in exactly one document?**

   > At least one term would appear in exactly one document. The name of the organization or any errors during NLP that may occur - concatenation of words, misspelling, not acquiring the corect data or format. Looking at some of the list of terms found, there includes numbers (0, 00, 000, ...) as well as words that wouldn't typically be considered used by people ("zuolay", "zylawi", "zyzzyva", "zzu").

2. **Relatively how many terms appear at least 5 times?**

   > There are 14,235 terms that appear at least 5 times.

3. **Show the 20 most frequent words. Words like "and", "to", "the" aren't especially meaningful. Which is the first word that you feel may be meaningful for characterizing the nonprofit? Why?**

   > The 20 most frequent terms were "and", "to", "the", "of", "provid", "for", "in", "a", "is", "educ", "communiti", "servic", "organ", "with", "support", "promot", "by", "through", "mission", and "that". The first meaningful term is "provid" which makes sense for a nonprofit whoms intention is to provide help to others. Other important terms would be "educ" and "communiti".

4. **How many documents contain "sacramento"?**

   > Within the mission statements, 98 documents contain the word "sacramento".

5. **What's one element in `irs990extract` where the mission contains "sacramento"?**

   - "name" => "Sacramento Public Library Foundation"
   - "volunteers" => "200"
   - "revenue" => "1117827"
   - "file" => "201900109349301480_public.xml"
   - "mission" => "TO RAISE RUNDS TO SUPPORT AND PROMOTE THE ACTIVITIES AND PROGRAMS OF THE SACRAMENTO PUBLIC LIBRARY AND COMMUNITY LITERACY PROGRAMS."
   - "employees" => "4"
   - "ein" => "680029756"

Come up with your own question similar to the questions above, and answer it.

6. **How many nonprofits listed themselves to contain zero employees and volunteers?**.

> Among 260,783 irs990extract documents, there were 45,385 or 17.4% of nonprofit organizations explicitly listing to having zero employees and volunteers? All documents that contain 'missing' for either/both employees or volunteers were excluded.

2 Selecting a Subset

(5 pts)

What do you do when your program doesn't run? Try using a subset of the data, the most important subset.

1. **Use one or more of the fields in `irs990extract` to define and pick the 10,000 largest nonprofits.**

```
function parse_vol(x)
    v = x["volunteers"]
    if ismissing(v)
        0
    else
        parse(Int, x["volunteers"])
    end
end

volunteers = map(parse_vol, irs990extract)

most_vol = sortperm(volunteers, rev = true)

top10k_idx = mostvol[1:10000]

top10k = irs990extract[top10k_idx]

top10k_freq = termfreq[top10k_idx, 1:end]
```

2. **What's the largest nonprofit based on your definition? Does it seem reasonable?**
    - "name" => "THE LEUKEMIA & LYMPHOMA SOCIETYINC"
    - "volunteers" => "3000000"
    - "revenue" => "438854761"
    - "file" => "201920589349300912_public.xml"
    - "mission" => "OUR MISSION IS TO CURE LEUKEMIA, LYMPHOMA, HODGKIN'S DISEASE AND MYELOMA, AND IMPROVE THE QUALITY OF LIFE OF PATIENTS AND THEIR

FAMILIES."
- "employees" => "1291"
- "ein" => "135644916"

> This does seem to be a reasonable largest nonprofit based on the defined subset. The Leukemia & Lymphoma Society Inc, founded in 1941, has 300k volunteers, over 430M revenue and 1291 employees all working for a worthy cause to help people all around the world. https://www.lls.org/who-we-are/about

3. **Drop all words that don't appear at least twice in this subset.**

```
# frequency of terms in subset
terms_appeared = 0 .< top10k_freq
word_count = sum(terms_appeared, dims = 1)

drop = dropdims(word_count, dims = 1)

# keep terms that appear at least two times
atleast_two = 2 .<= drop

# produce 10,000x5,230 Sparse
reduced_10k = top10k_freq[1:end, atleast_two]
```

We'll use this subset for the remainder of the assignment.

3 Principal Components Analysis

(5 pts)

Fit the first 10 principal components, i.e. project the data down into a 10 dimensional subspace.

1. **Interpret the principal ratio. What does it mean?**

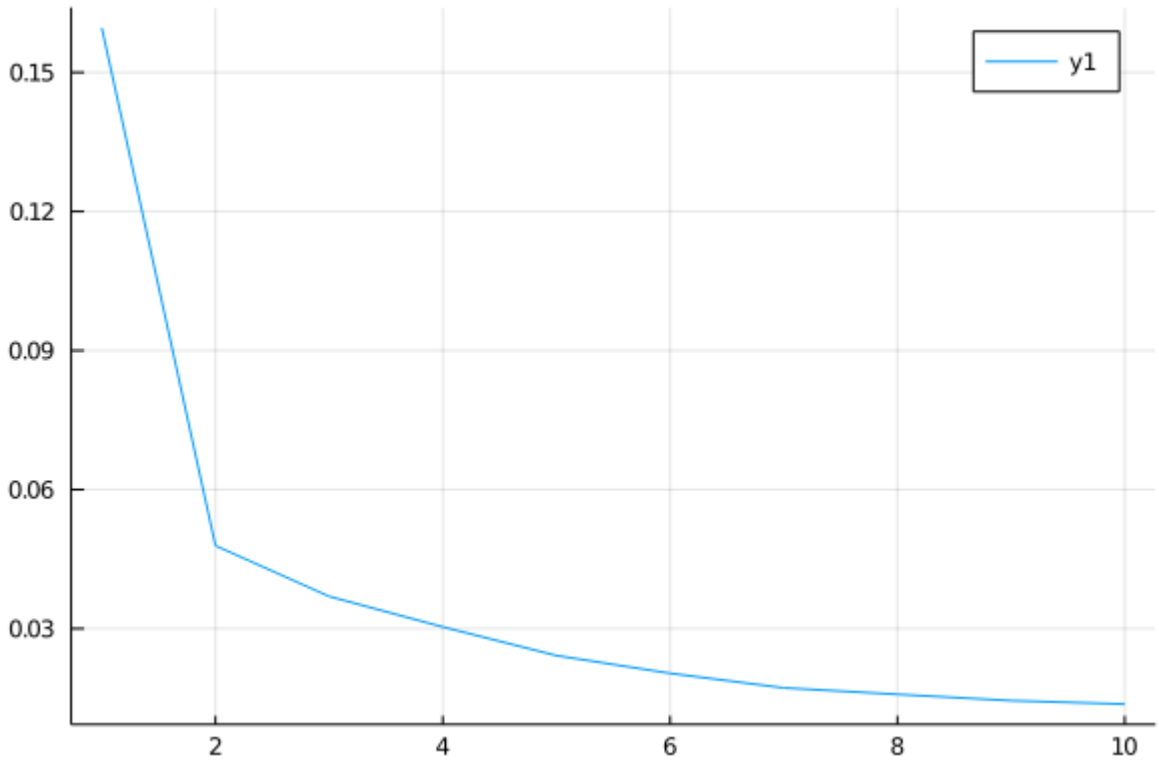
```
pca = fit(PCA, t_reduced, maxoutdim = 10) # reduced to 10 dimensions
# PCA(indim = 5230, outdim = 10, principalratio = 0.380541864408225)
```

> According to our lecture, the principal ratio tells how much the variance is explained by the principal components model. In other words, 38% of the data matches the total variance of our model. If I am understanding this correctly, this low ratio makes sense because if you are looking at the largest nonprofits based on the number of

> volunteers then the similarities between nonprofits start to greatly differ. It seems unlikely to have the largest nonprofits all doing the same thing.

2. **Plot the variances of the first 10 principal components as a function of the principal component number. What do you observe?**

```
pvars = principalvars(pca)
tvars = tvar(pca)
pcom10 = pvars/tvars
 # 0.15962540627643854
 # 0.047889318609469565
 # 0.036948747142903096
 # 0.030361496053469342
 # 0.024180251160642527
 # 0.020353853366523445
 # 0.017206042591083
 # 0.015812713715186357
 # 0.014457990296593436
 # 0.013706045195915709
plot(pvars/tvars)
```



> Based on the earlier 38% principal ratio, the majority of the data variation is in the first prinicpal component. 15.9% come from first component, 4.7% from the second, and decrements for each following component.

3. **Which words have the relatively largest loadings in the first principal component? (These the absolute values of the entries of `projection()`. Are these the kinds of words you expected? Explain.**

```
pca_proj = projection(pca) # 5230×10 Array{Float64,2}
pca_proj[1, 1:end] # Get first principal component
abs_proj = abs.(pca_proj[1, 1:end])
sortperm(abs_proj)
```

Hints:

1. Resources for interpreting principal components: Making sense of principal component analysis, eigenvectors & eigenvalues PCA and proportion of variance explained
2. Transpose the matrix to follow the structure described in the MultivariateStats documentation
3. If the program is too slow, try converting from a dense to a sparse matrix.

4 Clustering

(5 pts)

Apply k means with k = 3 to the principal components of the subset of data. This means you should be fitting k means to a data matrix with 10,000 observations, and 10 features, which are the scores for each of the 10 principal components.

1. **How many elements are in each group?**
   - sum(g1) # 4028
   - sum(g2) # 269
   - sum(g3) # 5703
2. **Which nonprofits are closest to the centroids? Feel free to use the function below.**

> Based on the function between, the nonprofits closest to the centroids are - NEW HANOVER REGIONAL MEDICAL, FEED MY STARVING CHILDREN INC, and EAST SIDE HOUSE INC.

3. **k means should find a group of mission statements that are very similar. What happened? Is it reasonable? If we were to continue this analysis, what would you do next?**

> Group 2 had clear missions statements that seperated them from the other two groups. All mission statemets in Group 2 consist of "SEE SCHEDULE O".

> Although I cannot identify the clustering pattern between Groups 1 and 3, the clustering does appear to be functional based on the results of Group 2.

```
"""
    Find the indices of the data points that are closest to the centroids defined by the kmeans clust
"""
function close_centroids(knn_model)
    groups = knn_model.assignments
    k = length(unique(groups))
    n = length(groups)
    result = fill(0, k)
    for ki in 1:k
        cost_i = fill(Inf, n)
        group_i = ki .== groups
        cost_i[group_i] = knn_model.costs[group_i]
        result[ki] = argmin(cost_i)
    end
    result
end
```

CODE

```
# Get data
using Serialization
using Statistics
using StatsBase
using MultivariateStats
using Clustering
using Plots
using Distributions

terms = Serialization.deserialize("terms.jldata")
termfreq = Serialization.deserialize("termfreq.jldata")
irs990extract = Serialization.deserialize("irs990extract.jldata")

################################################################################
# Q1

terms_appear = 0 .< termfreq
term_count = sum(terms_appear, dims = 1) # Frequency of terms
drop = dropdims(term_count, dims = 1)
```

```
atleast_five = 5 .<= drop # keep terms that appear at least five times
sum(atleast_five)

top20term = sortperm(vec(term_count), rev = true)
top20term_idx = top20term[1:20]
terms[top20term_idx]

count = 0
sac = []

for i = 1:length(irs990extract)
    m = irs990extract[i]["mission"]
    if (occursin("SACRAMENTO", m) == true)
        push!(sac, irs990extract[i])
        count = count + 1
    end
end

sac[1]

#################################################################################
# Q2

function parse_vol(x)
    v = x["volunteers"]
    if ismissing(v)
        0
    else
        parse(Int, x["volunteers"])
    end
end

volunteers = map(parse_vol, irs990extract)

mostvol = sortperm(volunteers, rev = true)

top10k_idx = mostvol[1:10000]

# pick out the top 10k from the irs990 extract
top10k = irs990extract[top10k_idx]

# pick out the top 10k from the termfreq
top10k_freq = termfreq[top10k_idx, 1:end]
```

```
top10k[1]

top10k[2]["mission"]

top10k[3]

top10k_freq[1,1:end].nzind

StatsBase.counts(top10k_freq[1,1:end].nzind)

terms_appeared = 0 .< top10k_freq

word_count = sum(terms_appeared, dims = 1) # Frequency of terms in subset

drop = dropdims(word_count, dims = 1)

atleast_two = 2 .<= drop # keep terms that appear at least two times

reduced_10k = top10k_freq[1:end, atleast_two] # 10,000x5,230 Sparse

# Q1.6 / Find all orgs that list having both zero employees and volunteers
empty_org = []
for i = 1:length(irs990extract)
    e = irs990extract[i]["employees"]
    v = irs990extract[i]["volunteers"]
    if ismissing((v))
        continue
    end
    if ismissing(e)
        continue
    end

    if ((occursin("0", v) == true) && (occursin("0", e) == true))
        push!(empty_org, i)
    end
end

length(empty_org)

##########################################################################################
# Q3

t_reduced = transpose(reduced_10k) # transpose to 5,230x10,000 Sparse
```

```
t_reduced = collect(t_reduced) # convert to 5,230x10,000 Dense

pca = fit(PCA, t_reduced, maxoutdim = 10) # reduced to 10 dimensions
# PCA(indim = 5230, outdim = 10, principalratio = 0.380541864408225)

tform = transform(pca, t_reduced) # 10×10000 Array{Float64,2}

pca_proj = projection(pca) # 5230×10 Array{Float64,2}
pca_proj[1, 1:end] # Get first principal component
abs_proj = abs.(pca_proj[1, 1:end])
sortperm(abs_proj)

approx = reconstruct(pca, trans)

pvars = principalvars(pca)
tvars = tvar(pca)

pcomponent10 = pvars/tvars

plot(pcomponent10)
##############################################################################################
# Q4

function close_centroids(knn_model)
    groups = knn_model.assignments
    k = length(unique(groups))
    n = length(groups)
    result = fill(0, k)
    for ki in 1:k
        cost_i = fill(Inf, n)
        group_i = ki .== groups
        cost_i[group_i] = knn_model.costs[group_i]
        result[ki] = argmin(cost_i)
    end
    result
end

new_k3 = Clustering.kmeans(tform, 3)
g1 = new_k3.assignments .== 1
g2 = new_k3.assignments .== 2
g3 = new_k3.assignments .== 3
close_centroids(new_k3) # 3-element Array{Int64,1}: 6765, 9, 6051

top10k[6765]
```

```
top10k[9]
top10k[6051]

sum(g1) # 4028

io = open("g1.doc", "w");
for i in 1:10000
    if(g1[i] == true)
        write(io, top10k[i]["mission"])
        write(io,"\n\n")
    end
end
close(io)

sum(g2) # 269

io = open("g2.doc", "w");
for i in 1:10000
    if(g2[i] == true)
        write(io, top10k[i]["mission"])
        write(io,"\n\n")
    end
end
close(io)

sum(g3) # 5703

io = open("g3.doc", "w");
for i in 1:10000
    if(g3[i] == true)
        write(io, top10k[i]["mission"])
        write(io,"\n\n")
    end
end
close(io)
```