Final Exam (version of 7:15am Wednesday June 7)

CS249 — Basic Data Science — D.S. Parker © 2017

Due: 11:55 pm Sunday June 18, 2017

- IMPORTANT: There are 6 questions below. ANSWER ANY 4 OF THE 6 QUESTIONS. All will have equal weight.
- IMPORTANT: In the .zip file for the final is a directory called `my_answers/`. For grading, please put all files you produce (writing, programs, and output) into the appropriate subdirectory. Then zip it and upload `my_answers.zip` to CCLE.
- IMPORTANT: Post PRIVATE questions to Piazza if you have questions. We will delete public questions; if you find problems with the exam, we'll make a Piazza notice for them. Do not talk with anyone about this exam. DO YOUR OWN WORK.

1. Grant Applications

The [APM] textbook used in this course studies a Grant Application dataset, particularly in chapters 12 and 14.

For this problem:

- the grant application datasets `unimelb_train.csv` and `unimelb_test.csv` are in the exam .zip file.
- develop a script or notebook that yields predictions for the test set.
  NOTE: the classification variable is `Grant.Status`, in the second column; 1 means successful, 0 unsuccessful.
- submit your predictions for the test set: Put the following files in your `my_answers/grants/` directory:
  - your file of predictions for the test data (`grants/predictions.csv`).
  - the script or notebook that produced these predictions (`grants/script....`)

The [APM] book offers many insights about this dataset, but rather than use the raw data, it uses digests produced by an R script named `CreateGrantData.R` (in the `AppliedPredictiveModeling package`). If you use the methods in the book, you can benefit from running this script and executing `save.image(file="grantData.RData")`. This stores all computed results in a saved state that can be quickly reloaded with `load("grantData.RData")`. For example, this reloading command appears in four scripts in the book: `12_Discriminant_Analysis.R`, `13_Non-Linear_Class.R`, `14_Class_Trees.R`, `18_Importance.R`. All these scripts are also included for you.

2. Attractiveness

The files `facestat_train.csv` and `facestat_test.csv` contain data accumulated from a (now defunct) site called `facestat.com`. At this site, people uploaded photos with data about themselves, and also made snarky comments about photos of other people. (The idea was reminiscent of the 'Hot-or-Not' program of Mark Z. that was immortalized in the movie *The Social Network*.) The dataset is a distillation of information provided by users, and has many missing values. This dataset has about 20 columns; the final column is a boolean value that is TRUE if the person is male. The data is interesting; for example Figure 1 shows how attractiveness varies with age. This set of plots suggests that men and women differ on 'attractiveness'.

In this problem we want to differentiate between men and women with the data provided. Given values for age, weight, attractiveness, etc., can we predict gender?

(a) You are provided a simple implementation of logistic regression in R, called `IRLS_logistic_regression.R`. It cannot handle missing values. The attractiveness dataset has *many* missing values.

First, modify the program to delete all rows of the data that have missing values, so that it produces a model (does not crash) when given the attractiveness data as input.

(b) Print a summary of the model you obtain (by deleting missing values this way), and include the printed output as a comment at the end of your modified `IRLS_logistic_regression.R`,

(c) Next, modify your program further to handle missing values by first deleting columns that have *many* missing values, and then deleting all rows that still contain any missing values. (You can choose a method for determining whether a column has many missing values.)

(d) At the end of your modified program `IRLS_logistic_regression.R`, also include as a comment the printed summary of the model obtained for the attractiveness data this way. Is the resulting model 'better' in some way?

(e) Upload `my_answers/attractiveness/IRLS_logistic_regression.R`

(f) For this data, we want to discriminate between men and women: given values for age, weight, attractiveness, etc., we want to predict gender.

Using the training data `facestat_train.csv`, develop a model predicting gender. (There are many missing values, so devise a strategy for dealing with them.)

(g) Using the file `facestat_test.csv` as input, generate predictions of gender values. (Values should be TRUE if the predicted gender is male, and FALSE otherwise.)

(h) Upload your predictions as `my_answers/attractiveness/facestat_predictions.csv`

NOTE: An earlier version of this problem used a dataset called `attractiveness_train.csv` — but it only contained 6 columns. It should have used `facestat_train.csv`, which has 20 columns. Please use `facestat_train.csv`, with it the problem is more meaningful. Sorry for this confusion.

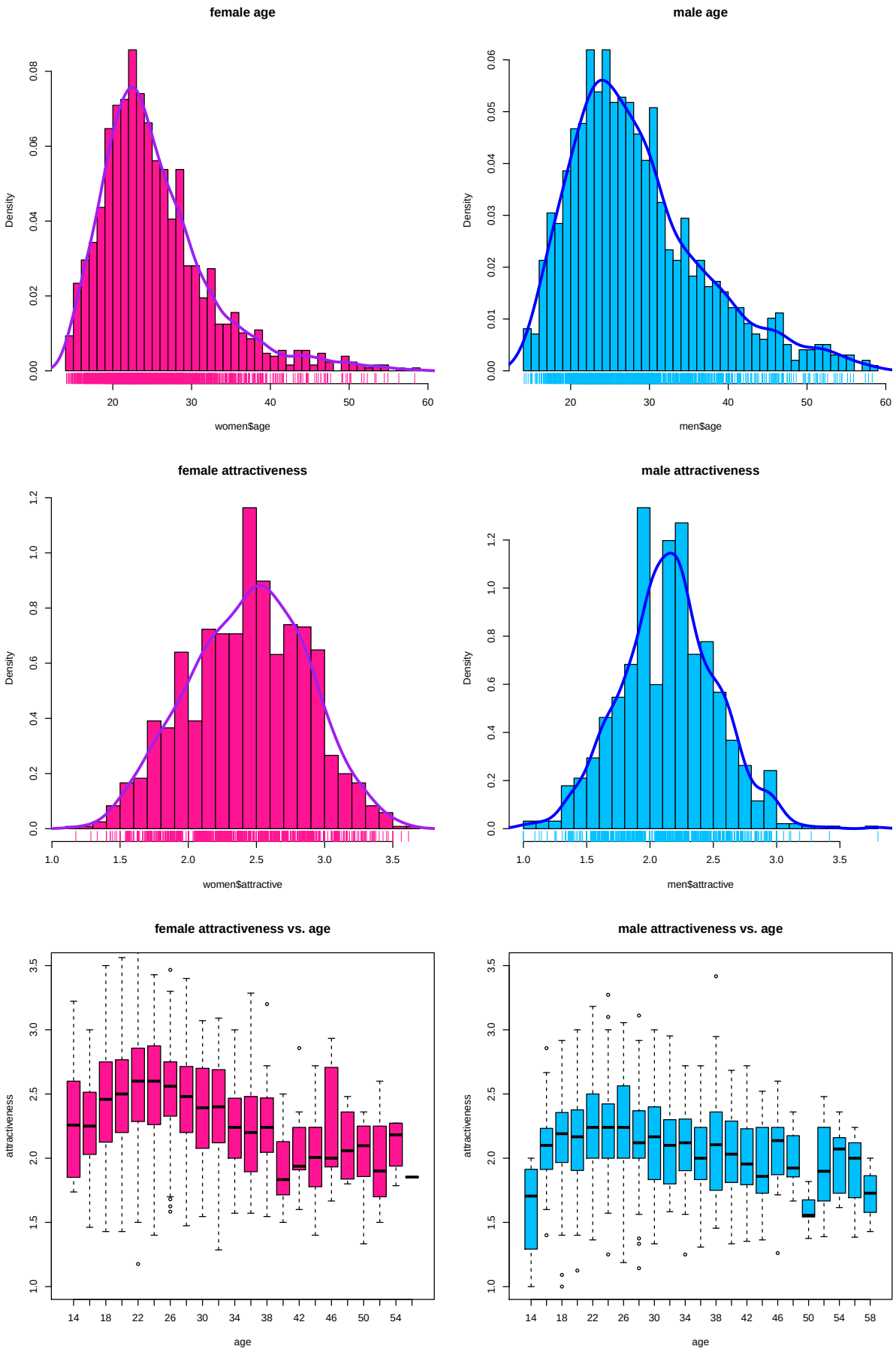


Figure 1: Age vs. Attractiveness in the facestat data

3. Unicorns

Crunchbase has a database of information about tech companies and investors. In this problem we study *unicorns* — young companies with a valuation exceeding $1B. A small database of unicorns is included as the file `unicorns.tsv`.

We also are using `a December 2015 export of the Crunchbase Data`, including two files `companies.tsv` and `investments.tsv`. These files include information on unicorns and 66,000 other companies; the information was exported in December 2015.

> In this problem we are going to construct 'social networks' linking investors and companies (and, in particular, unicorns). We want to determine whether the social network has two properities: (1) a 'rich-get-richer' or 'scale-free' structure, in which the node degree distribution follows a power law; (2) a 'small-world' structure, in which the graph can be large and locally-well-connected, but the diameter is small. (In a small world, distances between people are never large).

Figure 2 shows a subset of the unicorn–investor network, illustrating how an interactive graph of the data looks. (You can use any network analysis package. For Python, use of `networkx` is common, but `igraph` is also popular.)

In the `additional_material/` subdirectory are Jupyter notebooks showing related use of `networkx` — one reading Crunchbase, and one doing graph mining on Data Science Startups.

You are asked to construct three undirected, unweighted graphs in which the nodes represent tech companies — and two companies are connected if they have an investor in common.

- Construct a company–company graph in which the companies are *unicorns*. The `unicorns` and `investors_in_unicorns` data provide the data you need; there are about 160 unicorns, so this graph is relatively small.
- Construct a company–company graph using the entire Crunchbase dataset, where two companies are connected if they have an investor in common. The `companies` and `investments` tables have the data you need; there are over 66,000 companies, so this graph is much larger.

Then, for each of these graphs:

(a) determine the number of connected components.

(b) The *degree distribution* of a graph is a histogram of node degrees for nodes in the graph. It is often formalized as a function $P(d)$ that, for any degree value d, gives the number of nodes of degree d divided by the number of nodes in the graph. Compute the degree distribution (sequence of node degree values), and print nodes with the top 10 values.

(c) A graph is called a *scale-free graph* if its degree distribution follows a power law, i.e.: $P(d) \sim d^{-\gamma}$, where γ is a constant. (`Wikipedia page on scale-free networks`). Determine whether the graph is scale-free by plotting the histogram for $P(d)$ and determining whether $\log\ P(d)$ appears to be a linear function of d.

(d) compute eigenvector centrality of the nodes, and print nodes with the top 10 values.

(e) determine the diameter of the largest component (greatest distance between 2 companies).

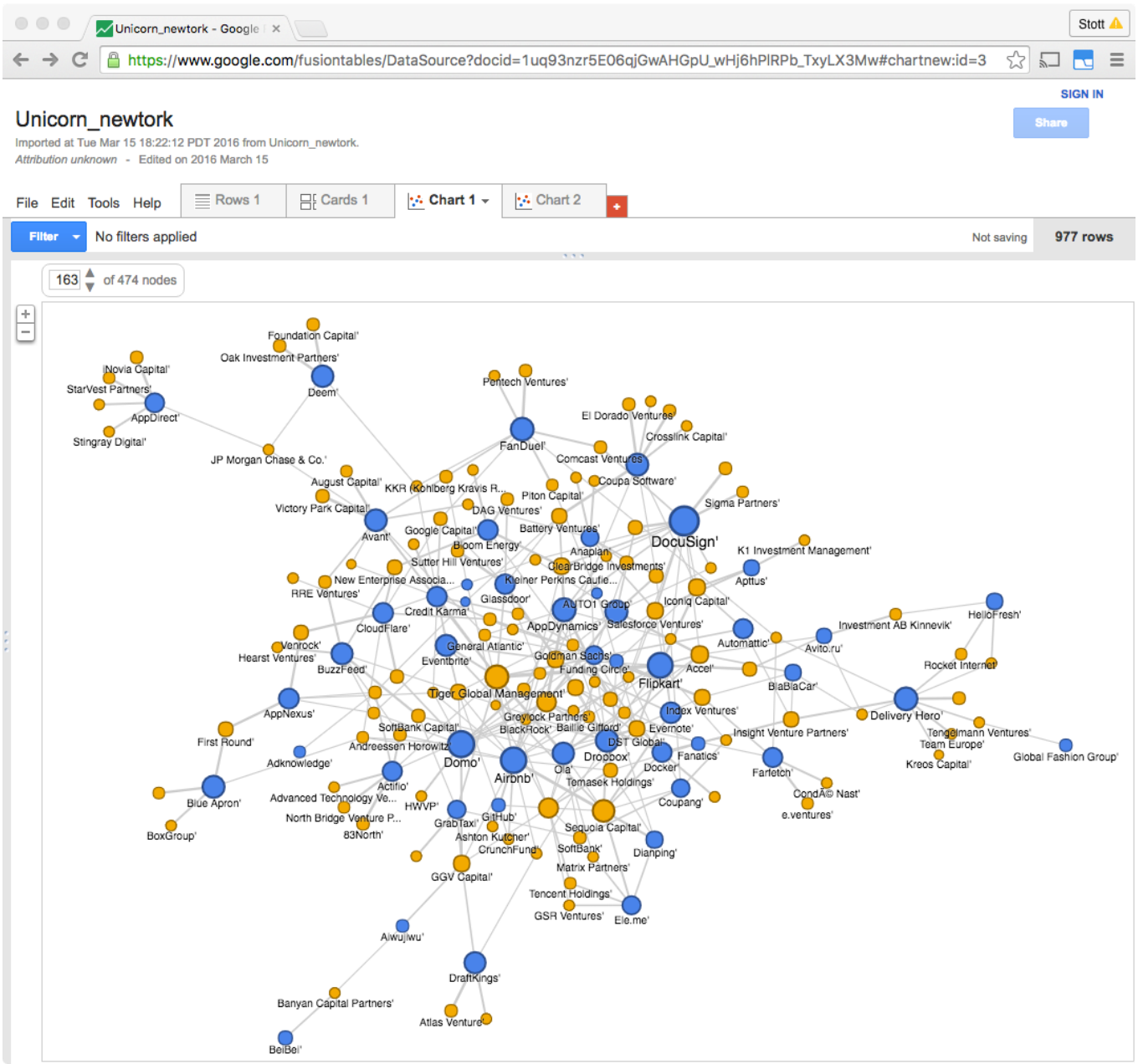


Figure 2: Image of an interactive network linking Unicorns and investors — made by Ruth Johnson at UCLA.

(f) compute the clustering coefficient $C^{\triangle}$ (= (3 times the number of triangles) / (the number of paths of length 2)), and use it to determine whether the network is a small-world network.

(g) Upload your answers in a text file named: `my_answers/unicorns/unicorns.txt`

Hint: The company-company graph is large since there are about: 66000 companies, 11000 investors who invest in more than one company, and 110000 edges between companies and these investors, yielding around 9 or 10 million edges between companies. So the company-company graph has about 10 million edges.

People often use 'all pairs shortest paths' as a way to compute the diameter on a graph:

for each node x
    using Breadth-First-Search:
        find the shortest distance from x to every other node y.

However, on a graph with n nodes and m edges, since the BFS can inspect all m edges, the total time complexity is $O(n\,m)$.

When n = 66000 and m = 10 million, $n\,m$ is about 7×10^{11}, i.e., 700 billion. This might require more than your laptop to finish before the exam deadline...

Is there another way to compute the diameter?

Yes – don't actually use the company-company graph for computing it.

Instead use the company-investor graph, which is a nice bipartite graph that has only about 110000 edges – a factor of 100 smaller.

Question: How is the diameter of the company-investor graph related to the diameter of the company-company graph?

[How long does it take to compute the diameter of the company-investor graph? For me, with nextworkx, about 90 minutes; with igraph, about 10-15 minutes. (igraph is implemented in C)]

4. Polarization of Voting in the U.S. Senate

Roll call vote is a standard method of voting in the U.S. Senate. The history of roll call votes is maintained at `voteview.com`. A recent analysis of this data (by Drew Conway, included) uses MDS (Multi-Dimensional Scaling) to display it as in Figure 3. This visualization suggests how *polarized* voting in the Senate has become, and offers some perspective on changes in 'bipartisanship' in United States government.

MDS a technique for embedding data points of high dimension (in this case, the large number of votes in each Congress) in a much lower-dimensional space (in this case, 2D plots) — in a way that preserves distances as closely as possible. Using MDS, the roll call votes for the Senate in the 102nd through 113th Congresses (each Congress covers a two-year period) display some change over time, as shown in the figure.

How can we measure polarization? This is the question we want to answer.

Collaboration between Republicans and Democrats has declined, to the point that voting patterns in the two parties appear largely decoupled. Republican senators often vote as a block (i.e., approximately all vote the same way on each roll call), independent of what Democrat senators do. Intuitively, this is what is meant by polarization. Please do the following:

(a) Extract the `.dta` data files provided in the final exam .zip file. (Complete data is available at `voteview.com/data` but in a different format that turns out to be more difficult to use.)

(b) The exam .zip file also includes a Jupyter notebook `voting.ipynb` for this data. Update this notebook to cover the 103rd through the 114th Congresses. (This is not difficult.)

(c) Extend the notebook to produce a *biclustering* visualization of the voting history of each Congress (103rd through 114th). An example is in Figure 4. In R the builtin `heatmap()` function can do this. In scikit-learn, `SpectralBiclustering()` can be used as in `scikit-learn.org/stable/auto_examples/bicluster/plot_spectral_biclustering.html` to obtain a display of the biclustered data — remember to use a final `plt.savefig()` to save the plot in a file.

(d) extend the notebook to include a model of *polarization* — a mathematical model of how distant two subsets of voters are — and explain why it is a good measure of polarization.

Polarization can be represented by any mathematical model that measures clustering or similarity of intra-party votes, and dissimilarity of cross-party votes. It should yield a single numeric value.

(e) calculate your measure of polarization for each Congress.

(f) plot the timeline showing the value of your polarization measure for each Congress.

'Is there a single right answer for the measure of polarization?' Obviously not. The question only asks you to define polarization mathematically. As an example, the minimum distance between all pairs of voting records involving senators of different parties is an almost-reasonable measure. Generate a better measure of polarization.

(If you want to find someone else's model, voteview.com was set up by people concerned about polarization. You might google for Christopher Hare and Keith Poole on polarization. But your model could be better than theirs.)

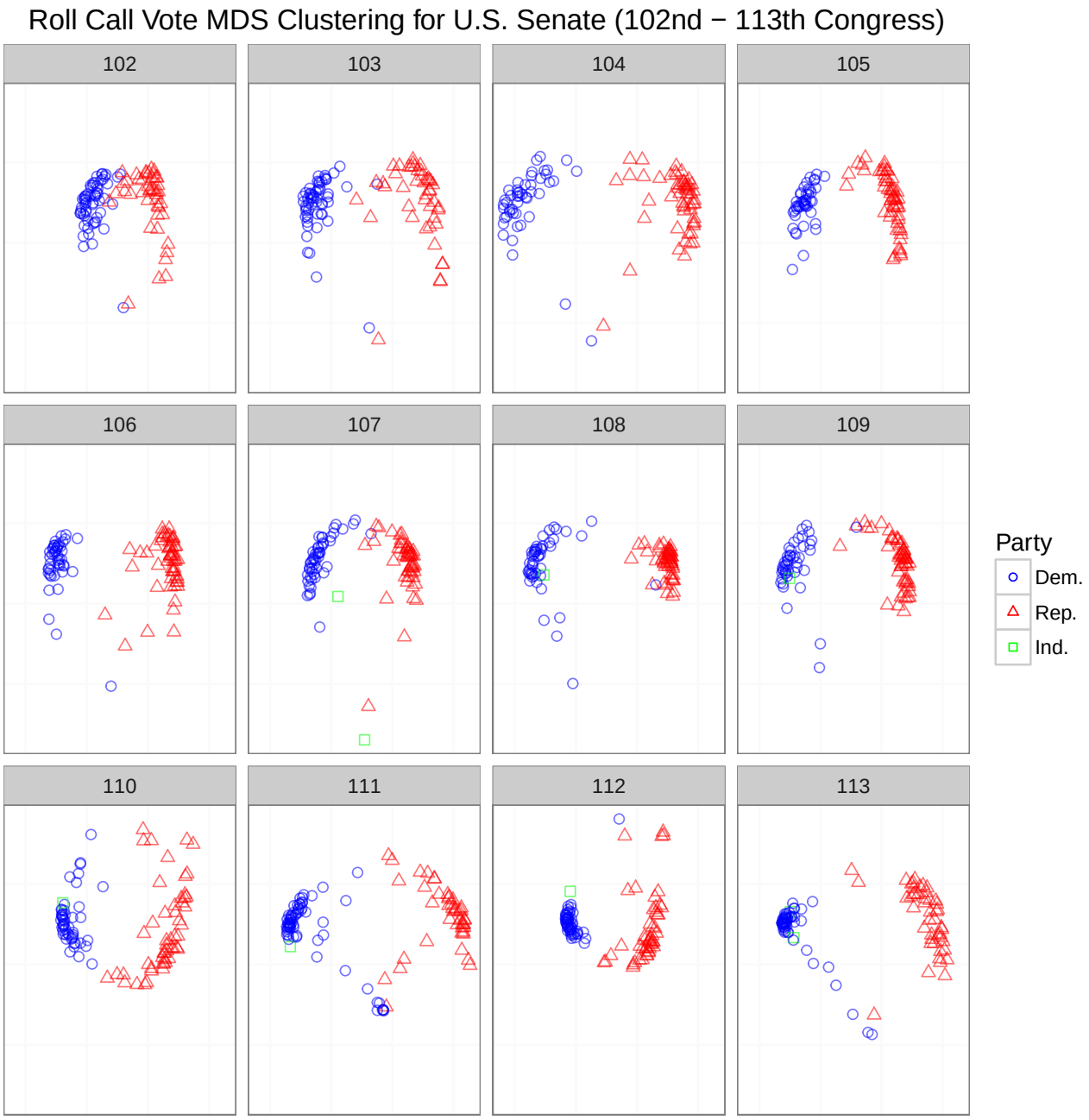
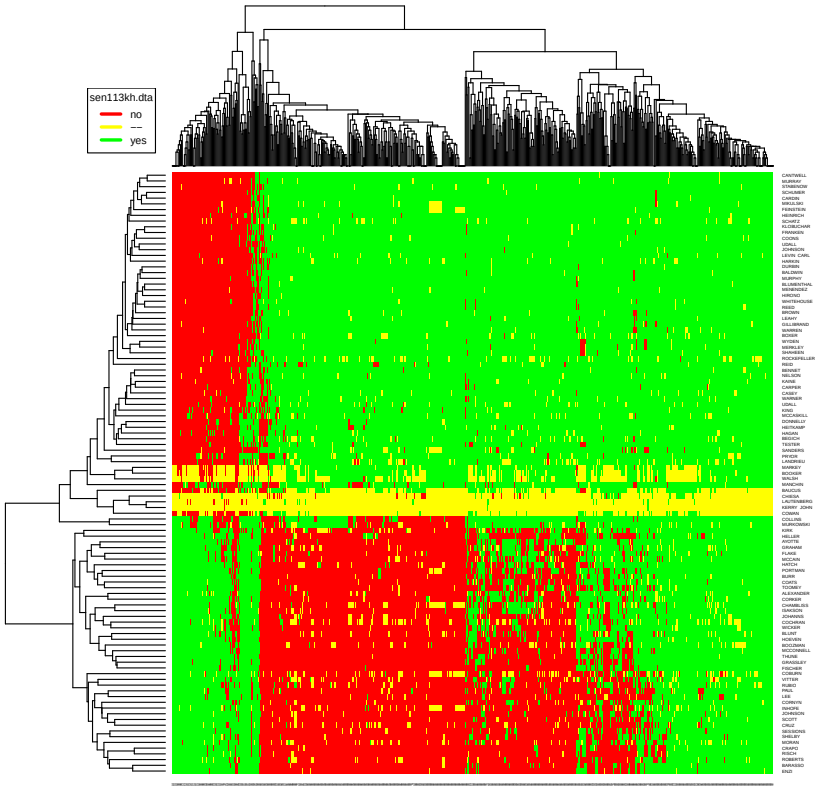


Figure 3: Change in voting patterns in the U.S. Senate over time, from the 102nd to 113rd Congress — a time period spanning from January 1992 through December 2016. YOU ARE ASKED TO UPDATE THIS TO COVER THE 103rd TO 114th CONGRESSES. The clustering of voters by party and the changes in separation between parties suggest 'polarization'.

In summary, complete the following:

- modify the `voting.ipynb` Jupyter notebook to cover the 103rd to 114th Congresses.
- extend the notebook to include biclustering displays (an example is shown in Figure 4),
- extend the notebook with your definition of *polarization*, and a plot of its values for these Congresses.
- upload your answers in a notebook named: `my_answers/voting/voting.ipynb`



Biclustered Senate voting matrix for the 113th Congress (rows = Senators, columns = Votes)
The heatmap function clusters rows and columns in order to expose similarity in voting patterns.
Notice Democrats form a voting block at the top of this display, with Republicans below.

Figure 4: A biclustering of voting data from the 113th Congress. Rows represent Senators, and columns represent votes. Rows and columns have been independently clustered by similarity to yield the hierarchical clusterings shown by the dendrograms.

5. Analyzing the Analyzers

We started off the quarter discussing how people have defined 'data scientists'. A booklet included in the exam .zip file, *Analyzing the Analyzers* (by H.D. Harris, S.P. Murphy, M. Vaisman, revised 2015) did perhaps the best job of this, defining them as essentially four kinds of people.

The four kinds of people are shown in figure 4-2 (p.22). Each is a pattern of values in a 5-dimensional vector, where the dimensions are Programming, Statistics, Math/OR, Business, and ML/Big Data. These dimensions are basically skill sets.

The methodology used to obtain their four classes of data scientist are described in *Appendix A*, on p.29–31. A job survey was used to obtain a (nonnegative) matrix of variable values for each person, and nonnegative matrix factorization was used to obtain the four patterns.

Using the job survey matrix from Homework 6 — `insights.stackoverflow.com/survey` — develop your own model of data scientist skill sets, and obtain your own classification of data scientists using nonnegative matrix factorization (such as the NMF package used in the booklet or `the Scikit-Learn method for NMF`.)

The NMF method factors a $n \times p$ nonnegative matrix X into $X = W\,H$, where W is $n \times k$ and H is $k \times p$, and k is the desired number of nonnegative factors. For example, with the first $p = 4$ columns of the iris data, and $k = 2$, we can obtain

$$H = \begin{array}{c} \begin{array}{cccc} \text{Sepal.Length} & \text{Sepal.Width} & \text{Petal.Length} & \text{Petal.Width} \end{array} \\ \left(\begin{array}{cccc} \mathbf{0.70} & \mathbf{0.49} & 0.17 & 0.02 \\ \mathbf{0.75} & 0.26 & \mathbf{0.79} & 0.29 \end{array} \right). \end{array}$$

As in PCA, the rows of H are interpreted as 'factors', in which larger entries are significant. Often people declare victory with this factorization, saying that these two factors explain irises. (They might say the first factor represents 'Sepal-size', and the second represents 'Length'.)

For this problem, there are four steps:

(a) Extract 'data science' notices (the same 4 kinds used in HW6) from the `JobSurvey` data.

(b) Repair or eliminate columns in this data that have the most missing values. (This step is described in the notebook `Analyzing_the_Analyzers.ipynb`, included.)

(c) Drop all remaining rows that have null values (using .dropna()), obtaining a matrix X.

(d) Use Nonnegative Matrix Factorization on X.

Extend the notebook `Analyzing_the_Analyzers.ipynb` with your work, and upload as `my_answers/data_scientists/Analyzing_the_Analyzers.ipynb`
Produce at least 4 matrix factors (classes of people), like the ones shown in Figure 4-2.

Notice: your alert classmate Phillip David noticed that some columns (such as the 'Star Trek; Star Wars' column) could be repaired (made null-free) because they included an empty response as a valid answer. Pandas turned empty values in the .csv file into NaNs when reading in the data. These columns have been fixed for you in the notebook. You are asked to repair or omit columns that have many missing values first, and then drop all rows that still have missing values.

An answer that obtains valid nonnegative factors is all that is required — and there is no single correct answer — but the factorization should provide some insights about *real* data scientist job postings. Include a description of the insights in your notebook. (Imagine telling data science job interviewers about your work on this.)

6. Personality

The notebook `Personality.ipynb` constructs a dataset from an online personality questionnaire data; it obtains a matrix of `ScoreValues`, in which each row (questionnaire result for a person) has 16 scores (describing 16 factors of a person's personality).

Some people want to know: do measurements of personalities tend to cluster into discrete sets? or do they spread over some larger continuum that has no obvious clusters?

Fill in the requested parts of the notebook, and also give an answer to the question:
for classifying people in the `ScoreValues` *data, is a cluster-based model (such as a k-nearest neighbor model) better than a dimensional model (such as PCA)?*

The notebook asks for you to fill in the notebook to answer this question. Specifically, it asks you to use a *gap statistic* developed by Tibshirani, Walther, and Hastie to determine a good value for the number of clusters. It also asks you to compute a PCA model of the data.

In other words, you are asked to answer the following question:

- do the rows in the ScoreValues data look like a set of points that fit fairly-clearly-defined clusters (like "classes" in a taxonomy of personalities)? If so, what is the best number of clusters (as determined by the gap statistic)?
- or do they seem more like a continuous, big ball of points in a k-dimensional ellipsoid? (where k is the number of principal components used).

Upload your answers in `my_answers/personality/Personality.ipynb`

Summary: What to Upload

Please upload your answers in the following files, as `my_answers.zip`:

```
 my_answers/
my_answers/grants/predictions.csv
my_answers/attractiveness/facestat_predictions.csv
my_answers/attractiveness/IRLS_logistic_regression.R
my_answers/unicorns/unicorns.txt
my_answers/voting/voting.ipynb
my_answers/data_scientists/Analyzing_the_Analyzers.ipynb
my_answers/personality/Personality.ipynb
```