Predictive Models: Exercise 2

Due on August 18, 2017 at 5:00pm

Professor James Scott

Matt Barrett **Timothy Lai** **Brett Scroggins** **Meyappan Subbaiah**

Problem 1

Exploratory Analysis: Austin Flight Delays

Create a figure, or set of related figures, that tell an interesting story about flights into and out of Austin. You can annotate the figure and briefly describe it, but strive to make it as stand-alone as possible. It shouldn't need many, many paragraphs to convey its meaning. Rather, the figure should speak for itself as far as possible. For example, you might consider one of the following questions:

What is the best time of day to fly to minimize delays?

What is the best time of year to fly to minimize delays?

How do patterns of flights to different destinations or parts of the country change over the course of the year?

What are the bad airports to fly to?

But anything interesting will fly.

Solution:

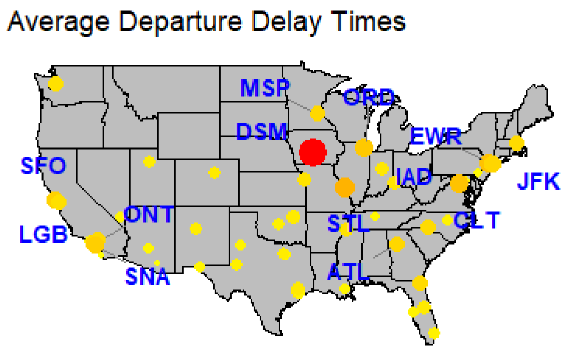
```
options(stringsAsFactors = FALSE)
austin_flights_df <- read.csv("https://raw.githubusercontent.com/jgscott/STA380/master/data/ABIA.csv")

library(tidyverse)

ggplot(austin_flights_df) +
  geom_histogram(aes(DepDelay))

### Time Series Calendar HeatMap
avg_delay <- austin_flights_df %>% group_by(Month,DayofMonth) %>%
  summarise(dep_delay = mean(DepDelay[DepDelay>0],na.rm=T),
            arr_delay = mean(ArrDelay[ArrDelay>0],na.rm=T)) %>% ungroup() %>%
  mutate(Year = 2008,
         monthweek = ceiling(DayofMonth/7),
         date = as.Date(paste(Year, Month, DayofMonth,sep="-"), "%Y-%m-%d"),
         day = weekdays(date),
         mnth = month.abb[Month],
         mnth = factor(mnth,levels=c("Jan","Feb","Mar",
                                      "Apr","May","Jun",
                                      "Jul","Aug","Sep",
                                      "Oct","Nov","Dec")),
         day = factor(day, levels= rev(c("Sunday", "Monday",
                                          "Tuesday", "Wednesday", "Thursday", "Friday", "Satur

 p1 <- ggplot(avg_delay, aes(monthweek, day, fill = dep_delay)) +
  geom_tile(colour = "white") +
  facet_wrap(~mnth,ncol=3) +
  scale_fill_gradient(low='yellow', high="red",name='Delay (minutes)',
                      breaks = c(10, 20, 30, 40, 50,60)) +
  labs(title = 'Average Flight Departures Delays',
       x = 'Week of Month',
       y = '') + theme_bw() + theme(aspect.ratio = 1)

p4 <- ggplot(avg_delay, aes(monthweek, day, fill = arr_delay)) +
  geom_tile(colour = "white") +
  facet_wrap(~mnth,ncol=3) +
  scale_fill_gradient(low='yellow', high="red",name='Delay (minutes)',
                      breaks = c(10, 20, 30, 40, 50,60)) +
  labs(title = 'Average Flight Arrival Delays',
       x = 'Week of Month',
       y = '') + theme_bw() + theme(aspect.ratio = 1)


library(maps)
library(ggmap)

# a set of personal choices for the map display
states <- map_data("state")
map.opts <- theme(panel.grid.minor=element_blank(), panel.grid.major=element_blank(), panel.background=
                  axis.title.x=element_blank(), axis.title.y=element_blank(),
                  axis.line=element_blank(), axis.ticks=element_blank(),
                  axis.text.y = element_text(colour="#FFFFFF"),
                  axis.text.x = element_text(colour = "#FFFFFF"))

### Departure Delays out of Austin
aus_dep <- austin_flights_df %>% filter(Origin == 'AUS') %>%
  group_by(Dest,Origin) %>% summarize(delay_times =
                              mean(DepDelay[DepDelay>0],na.rm=T)) %>% ungroup() %>%
  mutate(Dest = ifelse(Dest == "TUL","Tulsa",Dest),
    origin_loc = purrr::map(Origin, geocode, output = "latlon", source = "google",
                              messaging = FALSE),
        dest_loc = purrr::map(Dest, geocode, output = "latlon", source = "google",
                            messaging = FALSE)) %>% unnest() %>%
  rename("origin_lon"="lon","origin_lat"="lat","dest_lon"="lon1","dest_lat"="lat1")

## 75% quantile is 37.5
library(ggrepel)
aus_dep_text <- aus_dep %>% filter(delay_times > 37.5)

p2 <- ggplot(states) +
  geom_polygon(aes(x = long, y = lat,group=group),color='black',fill='grey') + map.opts  +
  coord_fixed(1.3) +
  geom_point(data=aus_dep,aes(x=dest_lon,y=dest_lat,size=delay_times,color=delay_times)) +
  geom_text_repel(data=aus_dep_text,aes(x=dest_lon,y=dest_lat,label=Dest),
                  fontface = 'bold', color = 'blue',
                  box.padding = unit(0.35, "lines"),
                  point.padding = unit(0.5, "lines"),
                  segment.color = 'grey50') +
  scale_color_gradient(low='yellow', high="red",name='Delay (minutes)') +
```

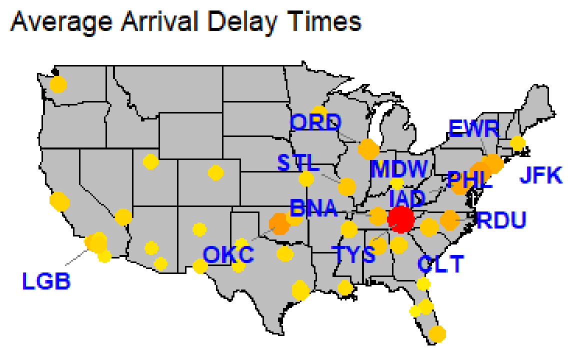
Problem 1 continued on next page...

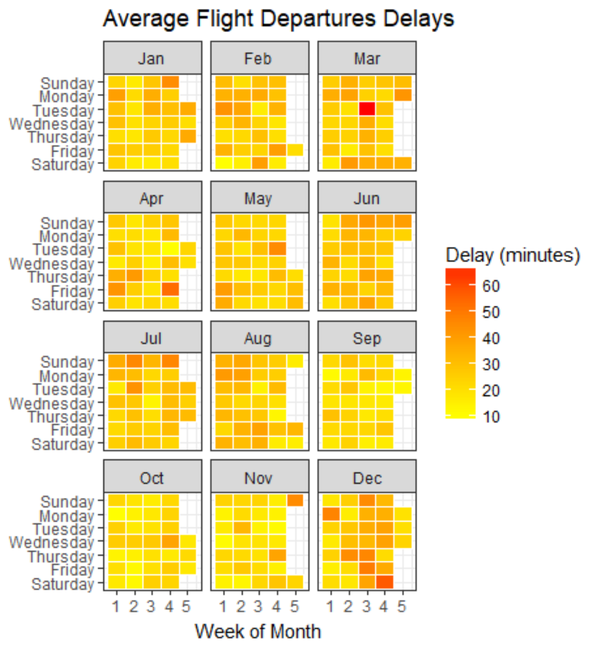
```r
  guides(color=FALSE,size=FALSE) +
  labs(title = "Average Departure Delay Times")


### Departure Delays heading to Austin.
aus_arrival <- austin_flights_df %>% filter(Dest == 'AUS') %>%
  group_by(Origin,Dest) %>% summarize(delay_times =
                                        mean(ArrDelay[ArrDelay>0],na.rm=T)) %>% ungroup() %>%
  mutate(Origin = ifelse(Origin == "TUL","Tulsa",Origin),
         origin_loc = purrr::map(Origin, geocode, output = "latlon", source = "google",
                                 messaging = FALSE),
         dest_loc = purrr::map(Dest, geocode, output = "latlon", source = "google",
                               messaging = FALSE)) %>% unnest() %>%
  rename("origin_lon"="lon","origin_lat"="lat","dest_lon"="lon1","dest_lat"="lat1")

### Quantile 36.02
aus_arr_text <- aus_arrival %>% filter(delay_times > 36.02)

p3 <- ggplot(states) +
  geom_polygon(aes(x = long, y = lat,group=group),color='black',fill='grey') + map.opts  +
  coord_fixed(1.3) +
  geom_point(data=aus_arrival,aes(x=origin_lon,y=origin_lat,size=delay_times,color=delay_times)) +
  geom_text_repel(data=aus_arr_text,aes(x=origin_lon,y=origin_lat,label=Origin),
                  fontface = 'bold', color = 'blue',
                  box.padding = unit(0.35, "lines"),
                  point.padding = unit(0.5, "lines"),
                  segment.color = 'grey50') +
  scale_color_gradient(low='yellow', high="red",name='Delay (minutes)') +
  guides(color=FALSE,size=FALSE) +
  labs(title = "Average Arrival Delay Times")

library(gridExtra)
library(grid)

grid.arrange(arrangeGrob(p2,p3,ncol=2),top = textGrob("Departure/Arrival Delays out of Austin-Bergstorm

grid.arrange(p1,p4,ncol=2)
```

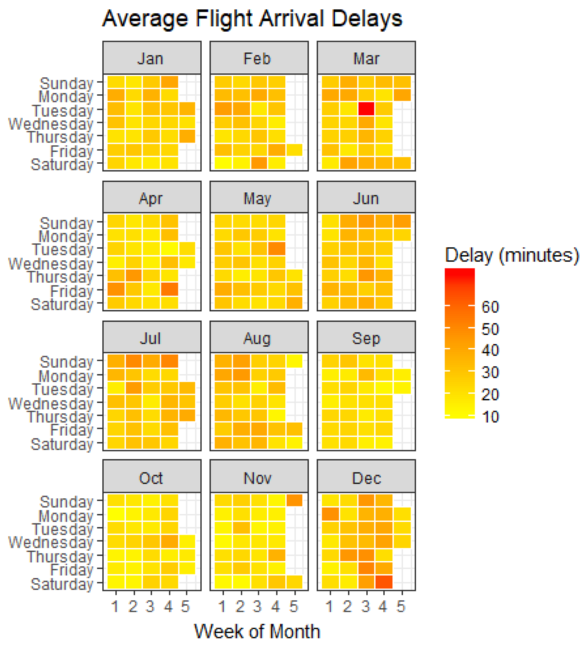
This code takes a significant amount of time to run due to geocoding requirements for each location. Currently, this chunk is set to eval=FALSE and .png files are included in place of direct code plots. If you desire to see direct code results, this can be changed and fresh plots will be created.

Departure/Arrival Delays out of Austin-Bergstorm









These plots show average delays on arrivals and departures. In just a few plots and with little external explanation, a reader can determine delays based upon calendar date and geographic location.

Problem 2

Author Attribution

Revisit the Reuters C50 corpus that we explored in class. Your task is to build two separate models (using any combination of tools you see fit) for predicting the author of an article on the basis of that article's textual content. Describe clearly what models you are using, how you constructed features, and so forth. (Yes, this is a supervised learning task, but it potentially draws on a lot of what you know about unsupervised learning!)

In the C50train directory, you have 50 articles from each of 50 different authors (one author per directory). Use this training data (and this data alone) to build the two models. Then apply your model to the articles by the same authors in the C50test directory, which is about the same size as the training set. How well do your models do at predicting the author identities in this out-of-sample setting? Are there any sets of authors whose articles seem difficult to distinguish from one another? Which model do you prefer?

Note: you will need to figure out a way to deal with words in the test set that you never saw in the training set. This is a nontrivial aspect of the modeling exercise.

Solution:

To begin our pre-processing of data, we generated a readerPlain function to read in text files individually. We also used the glob function to identify the directory names for training and test sets. To perform the bulk of our processing, we then created a Process_dir_text function. This function appends the author names provided to a vector, after we provide function parameters such as directory names and weight types. From here, we used the vector of author names in order to read in each of the files, uniquely by author name. These documents (per author) were then compiled into a corpus. The corpus needed to be transformed – specifically removing numbers and punctuation, stripping whitespace, and removing stop words. With this clean corpus, we were able to weight the words within the Document Term Matrix appropriately.

After the Process_dir_text function was created, we applied the function to both the training and test data. To ensure that we extracted the correct author names to the target column for our predictions, we filtered our author names via a regex. Finally, with all pre-processing complete, we would be able to implement our prediction models.

For the first of our two models, we decided to use an unsupervised learning approach by applying the Naïve Bayes model. To determine which the best fit model, we initially weighted the terms based on the number of times they appeared in the data set (Term Frequency – TF).

```
## [1] "Naive Bayes accuracy:  0.4744"
```

In doing this, we determined total accuracy to be approximately 47.4% in predicting the testing data set with our model. This was low, but somewhat expect as the data provided as text is difficult to process and the training set was limited (only 50 samples per author). With this small of a sample size, and the fact that there is a large amount of intersection among word choice per author, this was an average prediction. In order to possibly improve this model, we also implemented term frequency inverse document matrix (TFIDF) to vary the weight of word importance. This, however, showed a regression to 40% accuracy and therefore was not our strongest choice for Naïve Bayes. Ultimately, both models showed promise, but they left an opportunity for improved accuracy which led us to explore supervised learning methods.

Problem 2 continued on next page...

Table 1: Model accuracy by author

Authors	Accuracy
AaronPressman	0.65
AlanCrosby	0.76
AlexanderSmith	0.33
BenjaminKangLim	0.24
BernardHickey	0.62
BradDorfman	0.44
DarrenSchuettler	0.33
DavidLawder	0.35
EdnaFernandes	0.33
EricAuchard	0.44
FumikoFujisaki	0.97
GrahamEarnshaw	0.66
HeatherScoffield	0.51
JaneMacartney	0.11
JanLopatka	0.37
JimGilchrist	0.91
JoeOrtiz	0.23
JohnMastrini	0.45
JonathanBirt	0.46
JoWinterbottom	0.71
KarlPenhaul	0.55
KeithWeir	0.46
KevinDrawbaugh	0.33
KevinMorrison	0.60
KirstinRidley	0.39
KouroshKarimkhany	0.74
LydiaZajc	0.79
LynneO'Donnell	0.81
LynnleyBrowning	0.69
MarcelMichelson	0.55
MarkBendeich	0.42
MartinWolk	0.38
MatthewBunce	0.86
MichaelConnor	0.38
MureDickie	0.33
NickLouth	0.64
PatriciaCommins	0.45
PeterHumphrey	0.43
PierreTran	0.46
RobinSidel	0.83

These tables show the accuracy per author, which provided a bit more insight. The variation of accuracy shows how some authors can be found with a small sample size, but that some authors have similar enough writing styles that more articles would be required to properly and accurately differentiate between them.

Problem 2 continued on next page...

Table 2: Model accuracy by author

Authors	Accuracy
RogerFillion	0.76
SamuelPerry	0.35
SarahDavison	0.45
ScottHillis	0.13
SimonCowell	0.50
TanEeLyn	0.22
TheresePoletti	0.43
TimFarrand	0.43
ToddNissen	0.37
WilliamKazer	0.23

Next for comparison, a model was made with random forests.

Table 3: Accuracy for each Random Forest

ntreev	acc
100.00	0.34
500.00	0.36
1000.00	0.38

Prior to running our supervised learning method, we had to clean the data once more. To do this, we decided to eliminate elements in our test data for which we did not have a corresponding training point. The tradeoff here was that all of these elements would not be correctly classified upon running our prediction model. However, since this was a small subset of our testing data (approximately 3.2%), we decided as a team to move forward and found the tradeoff worth it.

Following the cleaning and elimination of some testing data elements, we implemented the Random Forest model. In the model, we tested three different forest sizes – 100, 500, and 1000 trees, respectively. Additionally, for each of these models, we settled for a default mtry parameter value and set our max nodes parameter to 20, so that the depth of each tree would not lead to overfitting. In running this model, we saw that the highest returned accuracy of 38.76% was provided by the n=500 trees random forest model. Therefore, we did not see improvement by running a supervised learning approach.

To conclude, based on the information available, the Naïve-Bayes model using Term Frequency generated our strongest prediction. We have learned that text is extremely difficult to process and analyze, given limited data. In order to improve our results moving forward, we would need more text files to improve our prediction algorithms.

Problem 3

Association Rule Mining

Revisit the notes on association rule mining, and walk through the R example on music playlists: playlists.R and playlists.csv. Then use the data on grocery purchases in groceries.txt and find some interesting association rules for these shopping baskets. The data file is a list of baskets: one row per basket, with multiple items per row separated by commas – you'll have to cobble together a few utilities for processing this into the format expected by the "arules" package. Pick your own thresholds for lift and confidence; just be clear what these thresholds are and how you picked them. Do your discovered item sets make sense? Present your discoveries in an interesting and concise way.

Solution:

Support = 1%

With a support value of 0.01, we would be able to capture as many baskets as possible with as many grocery items.

Confidence = .35

With a confidence value of 0.35, we felt this value was not too small to where it captured baskets that were not truly correlated, and it also was not too large of a confidence value to limit us to too narrow of a scope.

MaxLength = 3

Baskets would have three items or less. Most recipes or dishes implement a core of three ingredients. So it would make sense to see some form of correlation in baskets of this size.

```
rul_groc <- apriori(trans_grocery,
                         parameter=list(support=.01, confidence=.35, maxlen=3))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.35    0.1    1 none FALSE            TRUE       5    0.01      1
##  maxlen target   ext
##       3  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 98
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [88 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [89 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Problem 3 continued on next page...

```r
rul_groc_df <- data.frame(
                          lhs = labels(lhs(rul_groc)),
                          rhs = labels(rhs(rul_groc)),
                          rul_groc@quality)
```

Table 4: Top 10 results sorted by lift

lhs	rhs	support	confidence	lift
{citrus fruit,other vegetables}	{root vegetables}	0.01	0.36	3.30
{citrus fruit,root vegetables}	{other vegetables}	0.01	0.59	3.03
{root vegetables,tropical fruit}	{other vegetables}	0.01	0.58	3.02
{curd,whole milk}	{yogurt}	0.01	0.39	2.76
{rolls/buns,root vegetables}	{other vegetables}	0.01	0.50	2.59
{root vegetables,yogurt}	{other vegetables}	0.01	0.50	2.58
{tropical fruit,whole milk}	{yogurt}	0.02	0.36	2.57
{whipped/sour cream,yogurt}	{other vegetables}	0.01	0.49	2.53
{other vegetables,whipped/sour cream}	{yogurt}	0.01	0.35	2.52
{root vegetables,whole milk}	{other vegetables}	0.02	0.47	2.45

Analyzing the top 10 baskets sorted by lift, we are able to see that the baskets that show the highest lift are easily correlated. For example, we would expect baskets with fruits and vegetables to have other types of vegetables. Additionally, baskets with curd and whole milk can be expected to contain yogurt as well, as they are all dairy products. Looking at the confidence for these baskets, we can also see that several of them have high confidence values of > 0.5. This would indicate that shoppers are more likely than not to purchase the additional item.

Table 5: Bottom 10 results sorted by lift

lhs	rhs	support	confidence	lift
{berries}	{whole milk}	0.01	0.35	1.39
{long life bakery product}	{whole milk}	0.01	0.36	1.41
{fruit/vegetable juice}	{whole milk}	0.03	0.37	1.44
{citrus fruit}	{whole milk}	0.03	0.37	1.44
{dessert}	{whole milk}	0.01	0.37	1.45
{pastry}	{whole milk}	0.03	0.37	1.46
{napkins}	{whole milk}	0.02	0.38	1.47
{other vegetables,sausage}	{whole milk}	0.01	0.38	1.48
{soda,yogurt}	{whole milk}	0.01	0.38	1.50
{pork}	{whole milk}	0.02	0.38	1.50

In comparison, when we analyze the bottom 10 baskets sorted by lift, we are able to see that these baskets still show correlation. For example when we look at the first row, we see that baskets with berries are slightly expected to also have whole milk - but in considering whether we would personally buy berries with milk, it would certainly be possible. It is also worth noting that each of the bottom 10 baskets sorted by lift have whole milk as the secondary item. This also makes sense since whole milk is a common staple in households and an item that many people buy on a weekly basis.

Milk seems logical for a low value of lift due to the high level of confidence that already exists for a customer to purchase milk before other items increase it's likelihood. Lift cannot be as high for products that already

Problem 3 continued on next page...

have high purchase confidence. Finally, we see that the confidence values associated with these baskets are somewhat lower than for the top 10 (no value is above 0.4). Overall, the confidence values are still high though - one would expect shoppers with the items in the ”lhs” column in their baskets to have anywhere between a 35 and 39% chance to buy whole milk.

Table 6: Top 10 results sorted by confidence

lhs	rhs	support	confidence	lift
{citrus fruit,root vegetables}	{other vegetables}	0.01	0.59	3.03
{root vegetables,tropical fruit}	{other vegetables}	0.01	0.58	3.02
{curd,yogurt}	{whole milk}	0.01	0.58	2.28
{butter,other vegetables}	{whole milk}	0.01	0.57	2.24
{root vegetables,tropical fruit}	{whole milk}	0.01	0.57	2.23
{root vegetables,yogurt}	{whole milk}	0.01	0.56	2.20
{domestic eggs,other vegetables}	{whole milk}	0.01	0.55	2.16
{whipped/sour cream,yogurt}	{whole milk}	0.01	0.52	2.05
{rolls/buns,root vegetables}	{whole milk}	0.01	0.52	2.05
{other vegetables,pip fruit}	{whole milk}	0.01	0.52	2.03

When observing the top 10 baskets based on confidence, the results are similar in what was observed previously. Specifically, top basket in this subset is the same as in the above. What can be seen importantly with this split is the fact that all confidence values are above 50% meaning that majority of the time, shoppers with the items in the lhs column, will also purchase the item in the rhs column. Additionally, as previously stated, whole milk (a typical staple) tended to show up in a most of these top 10 baskets, which was to be expected.

Table 7: Bottom 10 results sorted by confidence

lhs	rhs	support	confidence	lift
{domestic eggs}	{other vegetables}	0.02	0.35	1.81
{other vegetables,whipped/sour cream}	{yogurt}	0.01	0.35	2.52
{berries}	{whole milk}	0.01	0.35	1.39
{tropical fruit,whole milk}	{yogurt}	0.02	0.36	2.57
{citrus fruit,other vegetables}	{root vegetables}	0.01	0.36	3.30
{long life bakery product}	{whole milk}	0.01	0.36	1.41
{butter}	{other vegetables}	0.02	0.36	1.87
{fruit/vegetable juice}	{whole milk}	0.03	0.37	1.44
{citrus fruit}	{whole milk}	0.03	0.37	1.44
{dessert}	{whole milk}	0.01	0.37	1.45

As stated above, the bottom 10 baskets based on confidence are very similar to the bottom 10 baskets based on lift. Even while this table only shows the lowest confidence, because of what our confidence level was set at, all these are significant finds with at least 35% confidence that the rhs item is in the basket with the lhs items. Lastly, what is again shown is that the rhs contains whole milk, a staple which many shoppers would already have in their baskets.