3106 Final Project

Ohad Klopman and Josef Cohen

2022-05-07

Github Repository: https://github.com/OhadKlopman/DonorLab

```
library(tidyverse)
library(ModelMetrics)
library(glmnet)
library(coefplot)
library(R.filesets)
library(moments)
```

Introduction

The nonprofit sector has struggled to make use of big data for a long time now. For many organizations, especially smaller ones, there simply isn't enough room in the budget, not to mention interest and human capital, to plan strategic decisions with a data-driven approach. Larger organizations can afford to pay for-profit consultants for fundraising and management services, and sometimes these approaches make use of data. But for the majority of the industry, there's an enormous amount of data left on the table.

In this report, we're going to introduce the beginning of a solution. Every year, the IRS collects and publicizes nonprofit tax returns in the Form 990 extract. This is a completely public data set with information on more than 200,000 of the largest organizations nationwide. The reason the data is made public is because regular people have an interest in knowing how their preferred organizations make use of the donations they collect. To that end, organizations such as Guidestar and ProPublica have built tools to let web users easily access the 990 returns for organizations. Here, though, we won't focus on one specific organization, but rather on the industry as a whole. We focus on the 2020 returns, and through a variety of methods, we'll unlock some insights for what makes organizations successful. What kinds of organizations raise the most? Can we predict revenue using other factors? Though we won't get into it in this report, we will close this report with a framework for expanding this analysis to inclue data from previous years (on the IRS website it's possible to find 990 datasets going back to 2013).

One facet of the 990 we're particularly interested in is the 501(c)3 subsections. 501(c)3 is the governing law that identifies which organizations are eligible for nonprofit tax-exempt status, but within that, there are 25 different types of eligible nonprofits ("subsections"), ranging from local food pantries, to national hospital systems, to black lung benefit trusts (that's subsection 21). What, if anything, do these subsections have in common from a financial standpoint? We will seek to answer that in this report using supervised regression and unsupervised clustering methods.

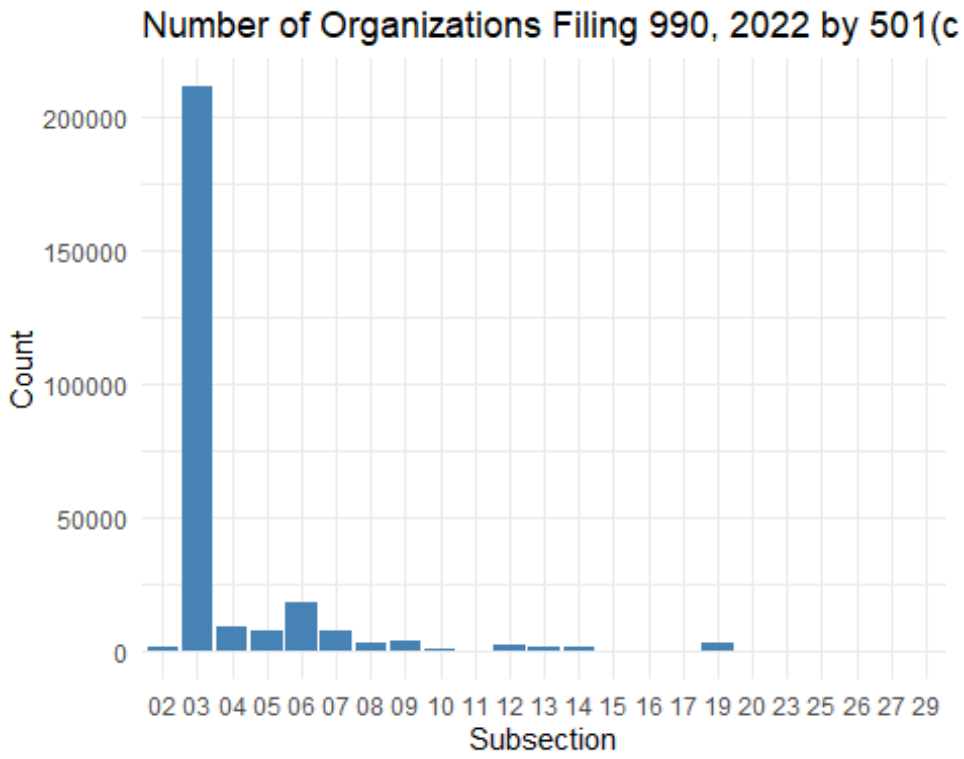
Reading in the data

```
memory.limit(size = 1e6)

## [1] 1e+06

irs <- read_csv("C:/Users/ohadk/Documents/R/DonorLab/990 Extracts/IRS 2020
990 Extract.csv")
```

Exploratory Data Analysis

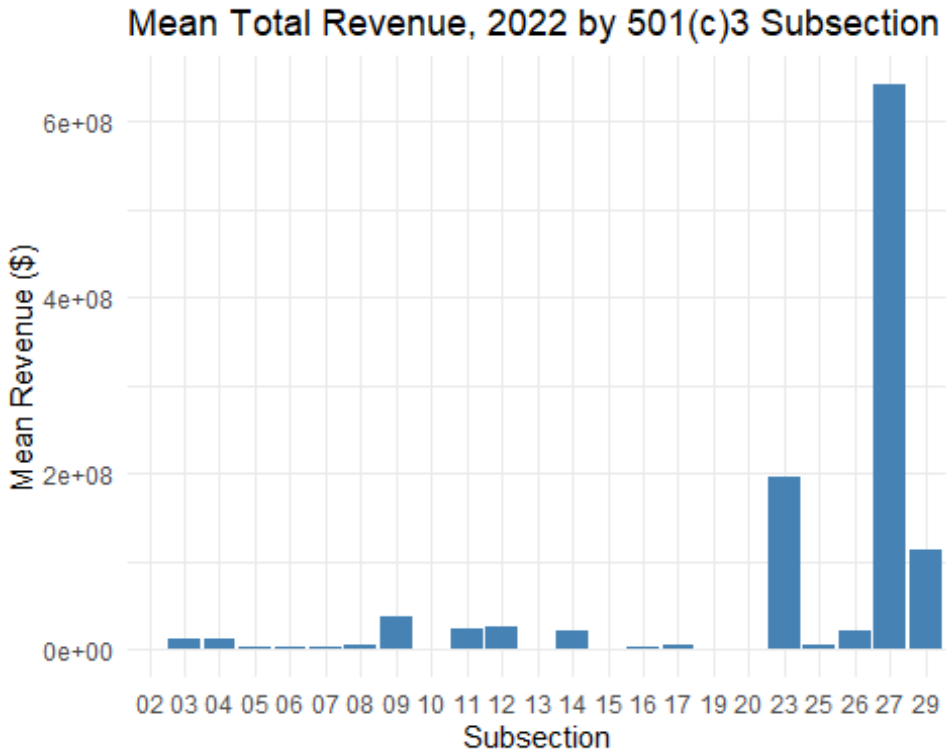
Since we're interested in the financial

```
irs%>%
  group_by(subseccd) %>%
  summarise(count = n()) %>%
  ggplot(mapping = aes(x = subseccd, y = count)) +
  geom_col(fill = "steelblue") +
  labs(title = "Number of Organizations Filing 990, 2022 by 501(c)3
Subsection",
       x = "Subsection",
       y = "Count") +
  theme_minimal()
```




```
irs%>%
  group_by(subseccd) %>%
  summarise(mean_rev = mean(totrevenue)) %>%
  ggplot(mapping = aes(x = subseccd, y = mean_rev)) +
```

```
  geom_col(fill = "steelblue") +
  labs(title = "Mean Total Revenue, 2022 by 501(c)3 Subsection",
       x = "Subsection",
       y = "Mean Revenue ($)") +
  theme_minimal()
```



The difference in revenue by subsection could not be more drastic. Subsections 3 and 4 are by far the most plentiful organizations. They are the types of organizations we interact with most often and generally associate with "nonprofit work." Section 3 alone is an umbrella group that includes 8 different types of charitable organizations:

- Charitable Corporation
- Educational Organization
- Literary Organization
- Organization to Prevent Cruelty to Animals
- Organization to Prevent Cruelty to Children
- Organization for Public Safety Testing
- Religious Organization
- Scientific Organization

And subsection 4 is a catch-all for "social welfare organizations," including food banks, homeless shelters, and anti-poverty missions. But in terms of revenue, these organizations on average bring in a tiny amount compared to some of the more specialized and financialized organizations, such as those in subsection 23 ("Associations of past and present members of the armed services") and 27 ("State-sponsored workers' compensation reinsurance plans"). It's hard, in a way, to even consider these organizations in he same legal category as churches and museums– which is why quantitative analysis is appropriate.

Processing and Wrangling the Data

Because so many questions on the tax form are Yes/No questions, the majority of the features in the dataset are binary variables, but not all of them. After reading in the data, we temporarily separate the numeric and categorical variables in order to process each one in the most fitting manner. For numeric variables, we treat them with log transformations when appropriate (more on that later). For categorical variables, we convert the dataset to a model matrix. In a model matrix, a single categorical variable is converted into a series of indicator (dummy) variables, all of them being 0 except for the column that contains the original value, which becomes 1. For example, there are 25 different types of subsections under the 501(c)3 nonprofit code. In our model matrix, every nonprofit will now have 25 columns tracking which subsection the organization falls into. If an organization is part of subsection 3 (a general charitable organization), then the variable `subsection 3` will be 1, and everything else will be 0.

```
## converting factor variables to a model matrix
col_classes <- unlist(map(irs, ~class(.)))
cat_cols <- which(col_classes == "character")
num_cols <- which(col_classes == "numeric")

## splitting between numeric and character features
irs_ein <- irs[, "ein"]
irs_categorical <- irs[, cat_cols] %>%
  select(-ein)
irs_numeric <- irs[, num_cols]

## removing columns which have less than two unique values
## and a sledgehammer to account for numeric variables that slip through and
appear as categories

count_table <- apply(irs_categorical, 2, table, useNA = "always")
one_cat_only <- which(sapply(count_table,
                             function(x) (length(x) <= 2 | length(x) > 50)))
irs_categorical <- irs_categorical[, -one_cat_only]

options(na.action = "na.pass")
cat_matrix <- as.data.frame(model.matrix(~., data = irs_categorical))
```

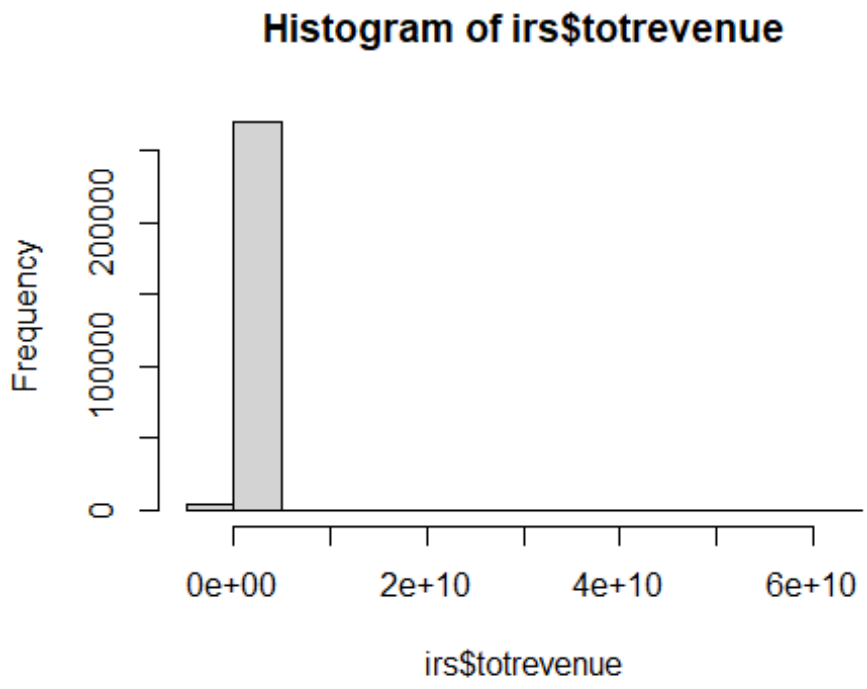
Assessing skewness in numeric variables

This dataset contains organizations of a wide variety of sizes, since the 990 form is the standard tax return for nonprofits throughout the United States. Since our goal is to predict and then understand organizational revenue, we should first take a look at the distribution of revenue in the dataset. And as is to be expected, there are a few massive organizations out there, while most are reasonably collected. This is a tell-tale example of right skew, so we pass a log transformation over it to get a better understanding of the data.

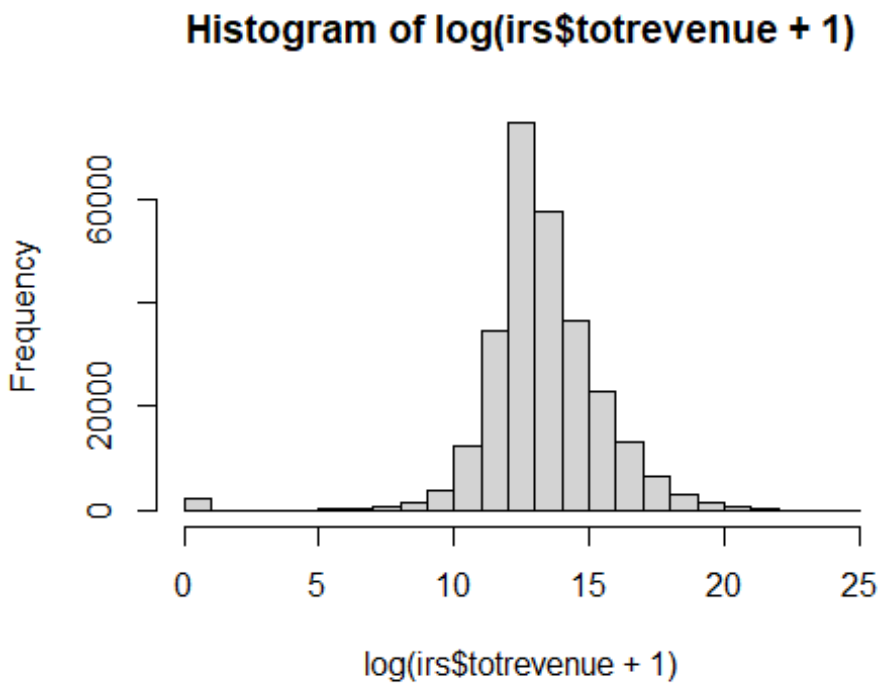
```
## applying log transformations to highly right-skewed variables
summary(irs$totrevenue)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -6.147e+07  1.989e+05  4.754e+05  1.015e+07  1.751e+06  6.252e+10
```

```
hist(irs$totrevenue)
```

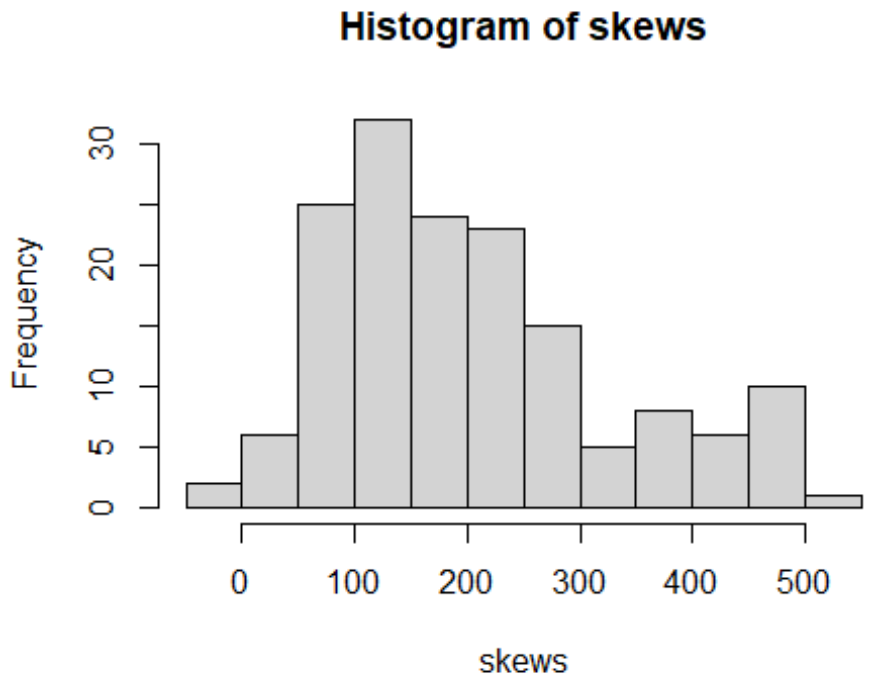



```
hist(log(irs$totrevenue + 1))
```



From here, it's worth checking the rest of the continuous variables for skewness. But with over 100 numeric features, that's too many to visualize with histograms and to judge visually. Instead, we make use of the `skewness` function in the `moments` package, which calculates a measure of non-symmetry for us. 0 represents perfect symmetry, and anything large and positive represents right skew (negative for left skew). In our case, we'll say that anything with a skew score over 5 deserves log transformation– except for `totrevenue` (total revenue). Even though we just passed a transformation over it for exploration sake, it doesn't help us to transform the outcome variable when it comes to modeling and analysis, so we re-exponentiate that afterwards.

```
skews <- apply(irs_numeric, 2, skewness)
hist(skews)
```

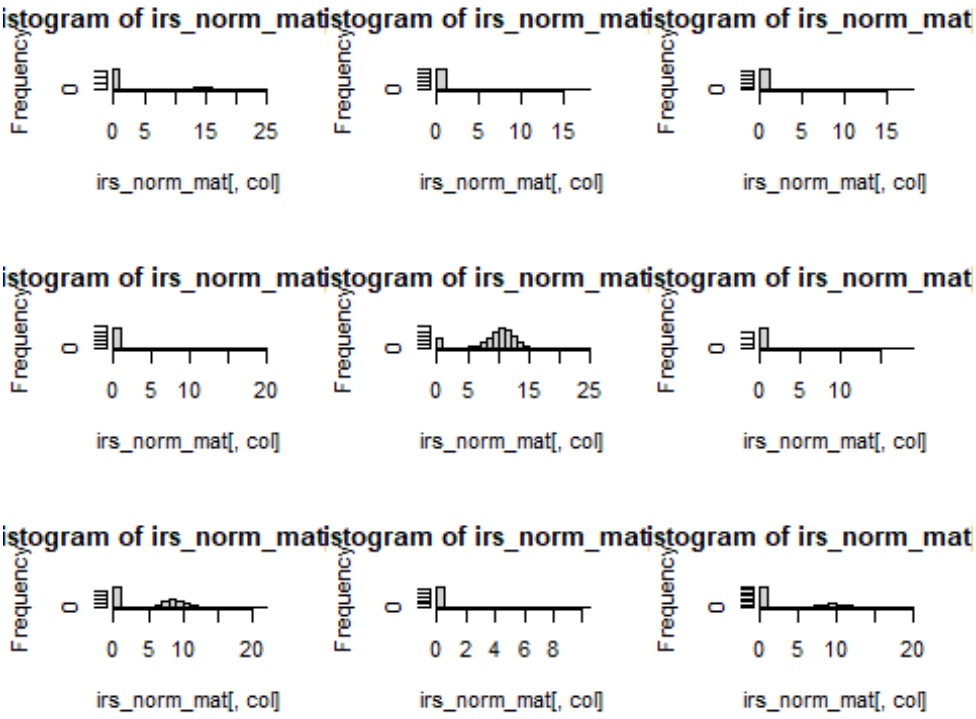



```
right_skews <- which(skews > 5)
log_transform <- function(x) {
  skew <- skewness(x)
  if ((is.na(skew)) | (length(table(x)) <= 2)) {
    return (x)
  }
  else if (skew > 5 & (length(table(x)) > 2)) {
    return(log(x + 1))
  }
}

irs_normalized <- apply(irs_numeric, 2, log_transform)
norm_lengths <- sapply(irs_normalized, length)
irs_normalized <- irs_normalized[norm_lengths == nrow(irs)]

irs_norm_mat <- do.call(cbind, irs_normalized) %>%
  as.data.frame() %>%
  filter(!is.na(totrevenue)) %>%
  mutate(num.totrevenue_2020 = exp(totrevenue)-1) %>%
  ## make dynamic in the future
  as.matrix()

irs_norm_mat[is.na(irs_norm_mat)] <- 0
```

The histogram of skew measures shows that nearly every numeric variable in the data set is right skewed, with some more dramatic than others. It may be justified in a further

analysis to try transforming these variables in different ways depending on how severe the skew is (for example, a log-log or double log transformation). For this anlaysis, we'll stick with just one log transformation. A quick sample of the transformed variables shows that they're mostly normally distributed and symmetric:

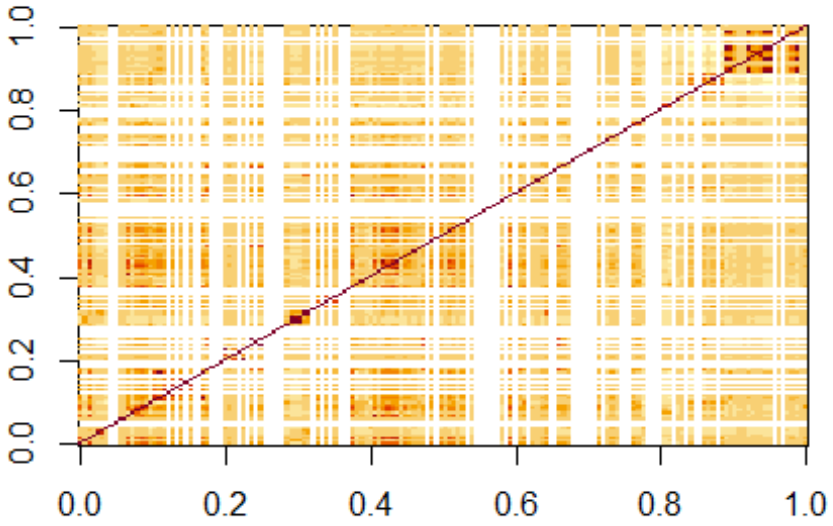
```
set.seed(12345)
par(mfrow = c(3, 3))
for (i in 1:9) {
  col <- sample(ncol(irs_norm_mat), 1)
  hist(irs_norm_mat[, col])
}
```



Now, with the categorical variables converted to a model matrix and numeric variables appropriately transformed, we can begin to analyze the data. The first step should be to determine which features are most strongly correlated with each other and with the outcome variable. For numeric variables, we can look at the correlation matrix:

To generate the correlation matrix

```
corr_matrix <- cor(irs_norm_mat)

## Warning in cor(irs_norm_mat): the standard deviation is zero

write.csv(corr_matrix, file = "irs_corrmat_2020", row.names = FALSE)

image(corr_matrix)
```



The correlation matrix shows that for the most part, the log-transformed numeric variables do not correlate too much with one another. There are small regions of overlap, and we can see that these mostly lie along the diagonal. Since the order of columns in our dataset are exactly as they appear on the 990 form, we can conclude that correlated variables appear most often next to each other on the form.

With this exploratory data analysis, we can begin to pare down the number of features to focus our sights on the ones that contribute most to organizational revenue. In this analysis, we focus on the column `totrevenue` as our outcome variable. According to the documentation provided by the IRS, `totrevenue` refers to the total revenue earned by an organization in one tax year. This includes revenue from donations, grants, programs, and the sales of goods and services. Different organizations fundraise in different ways, and even organizations that do similar work can achieve their revenue goals with completely different strategies. Some earn almost all their income from individual donations, while other focus on grants and major gifts. Because of this, we chose to model `totrevenue` as a function of the other features in the dataset, in order to avoid any sort of bias based on an individual organization's fundraising strategy.

Before we go on, it must be noted that `totrevenue` is (almost always) a linear combination of other columns in the dataset, since there are other columns that track the number of dollars raised by different methods. These methods include:

- program service revenue
- fundraising revenue
- grant revenue

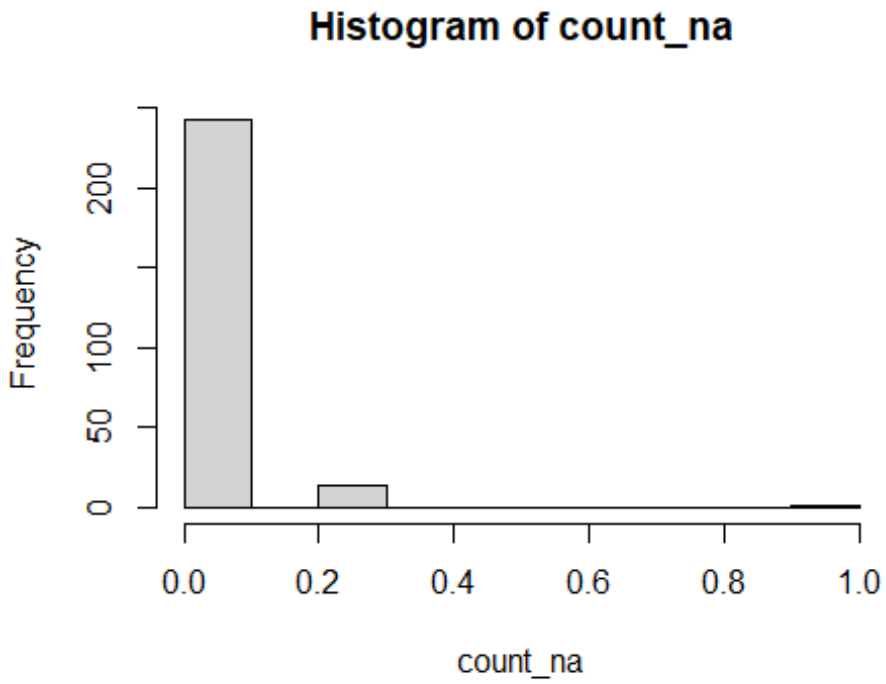
We remove these features from the data set, using a list that can be found on our GitHub repository.

Removing revenue columns

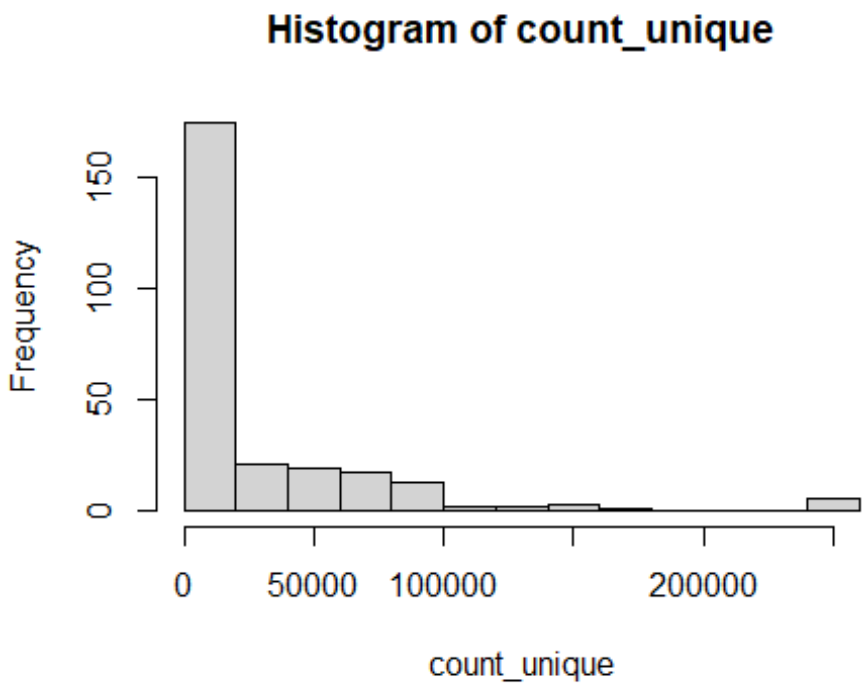
```
revenue_cols <- read_csv("C:/Users/ohadk/Documents/STAT 3106/Final
Project/Revenue Columns - IRS 990.csv")

## Rows: 22 Columns: 1

## -- Column specification --------------------------------------------------
------
## Delimiter: ","
## chr (1): Column

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this
message.

irs_numeric <- irs_numeric[, !(colnames(irs_numeric) %in%
revenue_cols$Column)]
```

Combining the data sets back together

```
## combining categorical and numeric variables back together
irs_mat <- cbind(cat_matrix, irs_numeric)

## removing columns that have too many NAs-- set as anything more than 50% of
the rows
count_na <- apply(irs_mat, 2, function(x) sum(is.na(x))) / nrow(irs_mat)
hist(count_na)
```




```
irs_mat <- irs_mat[count_na <= 0.5 ,]

## removing columns that don't have 2 or more unique non-NA variables
count_unique <- apply(irs_mat, 2, function(x) length(table(x)))
hist(count_unique)
```




```
irs_mat <- irs_mat[count_unique >= 2]

## sledgehammer to replace lingering NA values with 0 -> sparse matrix
irs_mat[is.na(irs_mat)] <- 0
```

LASSO Regression and Feature Reduction

The goal of this section and model is to identify which features are the most predictive of totrevenue in the 2020 data set. To do that, we fit LASSO regression at alpha = 1.

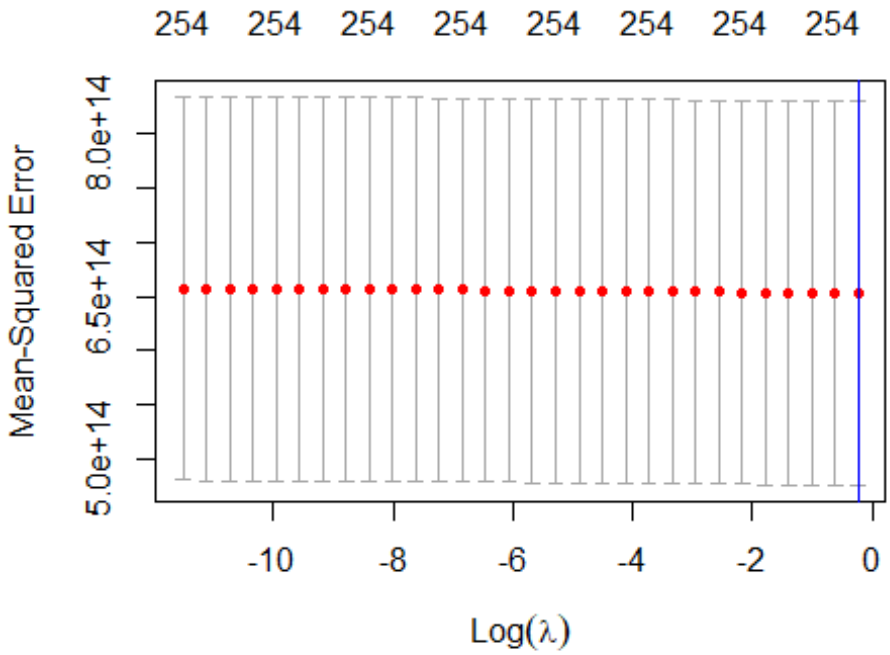
```
## splitting data into 80:20 train and test sets
set.seed(1234)

split <- sample(nrow(irs_mat), size = 0.8*nrow(irs_mat), replace = F)
train <- irs_mat[split, ]
test <- irs_mat[-split, ]

train_x <- train[, colnames(train) != "totrevenue"]
train_y <- train[, "totrevenue"]

test_x <- test[, colnames(test) != "totrevenue"]
test_y <- test[, "totrevenue"]

## fitting LASSO with a series of lambdas to find the best fit

lasso.cv <- cv.glmnet(as.matrix(train_x), as.matrix(train_y),
```

```
                      lambda = 10^seq(-5, -0.1, length.out = 30),
                      alpha = 1)

## use line below to save the fit to a local file
## saveRDS(lasso.cv, file = "lasso.cv_2020.RDS")

## use line below to read in the saved RDS Lasso file
## lasso.cv <- loadRDS("C:/Users/ohadk/Documents/STAT 3106/Final
Project/lasso.cv_2020.RDS")
```

Evaluating the LASSO Initial Fit

Below is a plot of the 30 lambda values we fit with `cv.glmnet` against the mean squared error (MSE) computed for the linear model generated using those lambda values. In order to have a confident predictive model, we want need to find a value of lambda that minimizes MSE.

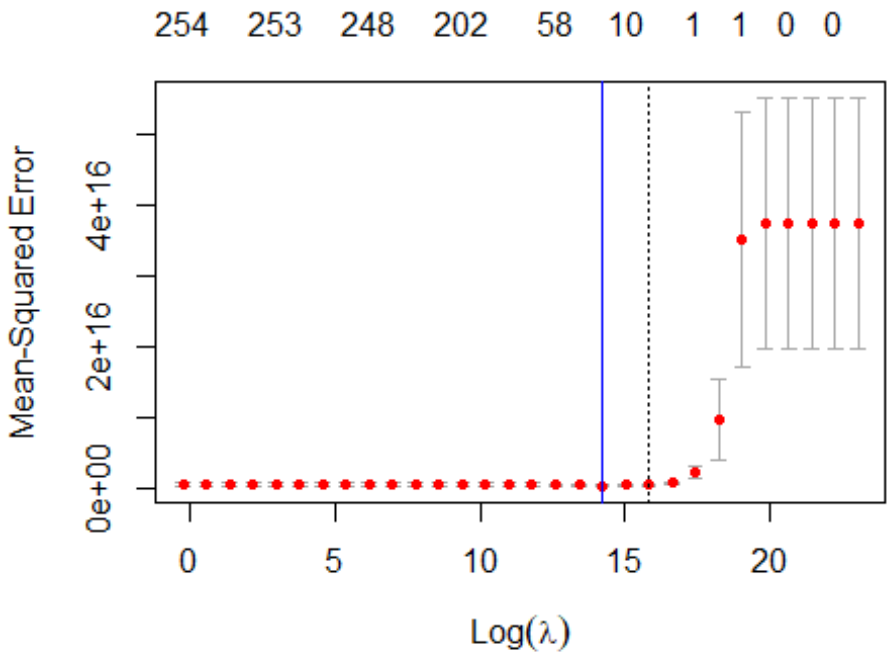
```
lasso.cv$lambda.min

## [1] 0.7943282

plot(lasso.cv)
abline(v = log(lasso.cv$lambda.min), col = "blue")
```



This initial sequence of lambda values shows that within this sequence, MSE is minimized at the right very extreme of the lambdas we tested, at lambda = 10^-0.1. However, this means there's a possibility that we haven't hit the true minimum MSE yet, since MSE could

concievably continue decreasing as lambda increases. We re-fit the model with a wider range of negative values:

```
lasso.cv2 <- cv.glmnet(as.matrix(train_x), as.matrix(train_y),
                       lambda = 10^seq(-0.1, 10, length.out = 30),
                       alpha = 1)
lasso.cv2$lambda.min

## [1] 1475589

plot(lasso.cv2)
abline(v = log(lasso.cv2$lambda.min), col = "blue")
```



Found it! Within this range is a value for lambda that minimizes MSE. From here, we can start to evaluate the penalty on each feature, which should tell us which features are most predictive and which ones we can ultimately drop from a future model. To do this, we use the best lambda value from `cv.glmnet` to fit a single model with `glmnet` alone. Then, we compare the predictions of the LASSO-adjusted model to the test set to evaluate model fit.

```
## refitting the model using lambda min
lasso_best <- glmnet(as.matrix(train_x), as.matrix(train_y),
                     lambda = lasso.cv2$lambda.min,
                     alpha = 1)

## predictions against the test set
lasso_pred <- predict(lasso_best, s = lasso.cv2$lambda.min, newx =
as.matrix(test_x))
```

```
## comparing actual vs. predicted
act_vs_pred <- cbind(actual = test_y, predicted = lasso_pred)
print(act_vs_pred[sample(nrow(act_vs_pred), size = 10, replace = FALSE), ],
digits = 3)
```

```
##          actual       s1
## 62619  21077090 21297031
## 148141  3298547  3688203
## 55178         0   105416
## 127713   601981   798670
## 142904   652455   685605
## 247998  6843850  4726654
## 143930    66487   170908
## 272827   150187   210217
## 141758  1059845  1030621
## 101961   901972   917633
```

```
## computing R-Squared and RMSE
rsq <- cor(act_vs_pred[, "actual"], act_vs_pred[, "s1"]) ^2
rsq
```

```
## [1] 0.9804791
```

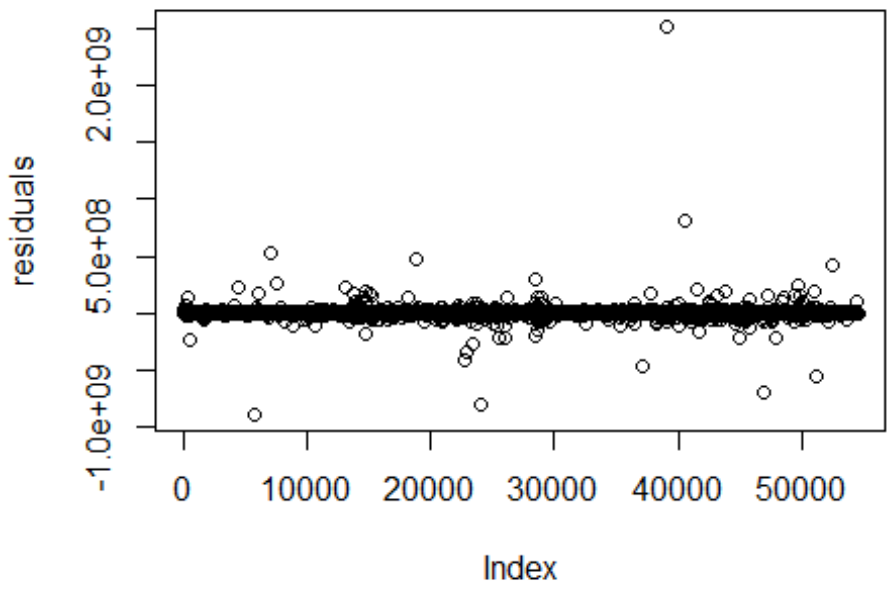
```
rmse(act_vs_pred[, "actual"], act_vs_pred[, "s1"])
```

```
## [1] 15212690
```

```
best_betas <- as.data.frame(as.matrix(coef(lasso_best))) %>%
  rownames_to_column()
colnames(best_betas) <- c("Feature", "Est")
```

This model achieves a phenomenal r-squared of 0.9794 but a comparatively weak RMSE of 15569504. From the penalized coefficients, we can see that none of the coefficients were penalized all the way to zero. This is probably due to the indicators we provided for all the various categories an organization could fit into. We can evaluate the residuals of the fitted model too. We don't necessarily expect the model to have the same characteristics as an unbiased OLS regression, but comparing predicted and actual gives us a good understanding of where the model is most effective and least effective at prediction. To do that, we plot the residuals of the cross-validation test set.

```
residuals <- lasso_pred - test_y
plot(residuals)
```



We could expect this would be the case given the extremely high r-squared value that we discussed earlier, but the residual plot shows that the model is exceptionally effective at minimizing residuals.

So to return to our questions from before: do nonprofits tend to raise revenue in the same way, or are there important differences between 501(c)3 subsections? And what characteristics are most predictive of revenue? To do that, let's begin looking at the coefficients on each of the subsections and which ones were "penalized out"– reduced to 0– by the LASSO model.

```
zeroes <- best_betas[best_betas$Est == 0, ]
survivors <- best_betas[best_betas$Est != 0, ]

survivors
```

```
##                Feature          Est
## 1          (Intercept) 8.742621e+04
## 130         invstmntinc 1.542402e-01
## 144            gnlsecur 1.586108e-01
## 146             netgnls 1.180811e-01
## 155         grntstogovt 1.428621e-01
## 164           payrolltx 5.656317e-01
## 185          othrexpnsb 1.117357e-01
## 186          othrexpnsc 1.421260e-01
## 187          othrexpnsd 7.028461e-03
## 190        totfuncexpns 9.777176e-01
## 198    invntriesalesend 1.018033e+00
```

```
## 201        invstmntsend 4.333095e-02
## 217 unrstrctnetasstsend 1.297896e-02
## 222     retainedearnend 2.445006e-02
## 223      totnetassetend 1.461058e-03
## 227     gftgrntsrcvd170 2.398048e-03
```

What survived

Out of the 255 features we entered from the 990 dataset into the model, 245 of them were penalized all the way to 0, leaving 10 behind (plus the intercept). The surviving features all have fairly small coefficients in magnitude, but it's worth recalling that many numeric features were log-transformed. To interpret `totfuncexpns`, for example, we treat the coefficient as 0.989*log(x+1), which is equivalent to exp(0.989)-1*x in while dollars. `totfuncexpns` stands for "total functional expenses," an accounting term that refers to the money an organization must spend just to stay afloat: staff, utilities, program costs, etc. Thus we can say that for every one dollar an organization spends, we can predict on average $1.60 in total revenue. Most organizations also have program expenses separate from their functional expenses, and these are recorded in a separate part of the 990. Interestingly, the only non-functional expense variable that survived the LASSO penalty is `grntstogovt`– "grants to government."

Looking at the surviving features, it seems that some variables which directly contribute to total revenue may have slipped through. `invstmninc`, for example, stands for "investment income" and may be counted towards total revenue. Nonetheless, it is a smaller coefficient in magnitude than `totfuncexpns`, despite having a more direct connection to total revenue. This is a sign that we have begun to uncover some insight into the revenue structure of nonprofits in the US. The 990 form is filed by successful, well-organized nonprofits, and to become a successful nonprofit, the organization has to raise enough revenue to cover its expenses. At the same time, functional expenses can't completely count for revenue, because the organization still has to spend money on programs in order to achieve its mission and look responsible to donors. As an organization grows, it's also bound to have higher functional expenses, a sign that the organization is growing more robust in its operations.

What didn't make it

We entered this analysis hoping to learn whether the different subsections of the 501(c)3 code also had distinctive funding structures in addition to their various specific missions, or if an organization's financial makeup didn't depend on its subsection at all. Given that every since subsection feature was cut by the LASSO regularization, we can conclude that subsection is not an effective way to predict total revenue. Fundraising and nonprofit management strategists might be interested to learn that nonprofits can have similar financial structures even if they work in dramatically different industries.

On the topic of fundraising professionals, we also see that many features which were related to fundraising expenses did not make the cut, including `lessdirfndrsng` (total fundraising expenses), `profndraising` (professional fundraising expenses), and `advrtpromo` (advertising and promotion). This is a great sign for smaller organizations

which don't have as much to spend on expensive fundraising campaigns: once you account for other aspects of an organization's financial health, the amount spent on fundraising has little predictive power for how much revenue the organization is likely to take in. This is just observational data, though, so making any kind of causal claim is suspect. Future analyses would benefit from taking a more controlled approach to the issue of fundraising expenses, either through random assignment or regression discontinuity.

Unsupervised Learning

Having studied which features are most strongly predictive of total revenue, we turn back to the other question we set out to understand: what are the similarities between organizations of different subsections? To do this we employ K-Means clustering on the 990 dataset, scanning from 2 to 30 clusters. If the subsections do actually have distinctive financial structures, then a stable cluster assignment should have a fairly homogeneous population in terms of subsection. After the initial scan, we plot the between-cluster sum of squares, the within-cluster sum of squares, and the ratio between the two. We find 16 clusters to be an appropriate number, as it maximizes between-ss and minimizes within-ss without using an overwhelming number of clusters. This is 9 fewer clusters than there are subsections, so in the following pages we will compare between 16 and 25 clusters, evaluating them based on comprehensibility, stability, and usefulness.

```
set.seed(1234)
subset_cols <- colnames(irs_mat)[grepl("subseccd", colnames(irs_mat))]

irs_mat_for_kmeans <- irs_mat %>%
  select(-subset_cols) %>%
  as.matrix()

## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(subset_cols)` instead of `subset_cols` to silence this
message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.

k_max <- 30
between_ss <- c()
within_ss <- c()
for (k in 2:k_max) {
  km_temp <- kmeans(irs_mat_for_kmeans, centers = k, iter.max = 100)
  between_ss[k-1] <- km_temp$betweenss
  within_ss[k-1] <- km_temp$tot.withinss
}

jpeg("kmeans_irs_eval.jpg")
    par(mfrow=c(3,1))
    plot(2:k_max, between_ss)
    plot(2:k_max, within_ss)
    plot(2:k_max, between_ss/within_ss)
dev.off()
```

```
## png
##   2
```

Evaluating 16 and 25 clusters

To compare between the cluster assignments and the true subsections, we produce a two-way table for the 16- and 25-cluster outcomes. We then measure homogeneity by calculating the share of each cluster that subsections are sorted into, and the share of each subsection that clusters collect.

```
set.seed(1234)

subsecs <- irs_mat[, subset_cols]
maxes <- apply(subsecs, 1, which.max)
max_names <- sapply(maxes, function (x) colnames(subsecs)[x])


k_16 <- kmeans(irs_mat_for_kmeans, centers = 16, iter.max = 100)
k_25 <- kmeans(irs_mat_for_kmeans, centers = 25, iter.max = 100)

cluster_assignments_16 <- k_16$cluster
cluster_assignments_25 <- k_25$cluster

cluster_mat_16 <- as.data.frame.matrix(table(cluster_assignments_16,
max_names))
cluster_mat_25 <- as.data.frame.matrix(table(cluster_assignments_25,
max_names))

## calculating the dominating cluster's share
apply(cluster_mat_16, 2, function(x) x[2] / sum(x))

## subseccd03 subseccd04 subseccd05 subseccd06 subseccd07 subseccd08
subseccd09
##  0.9913503  0.9949818  0.9988681  0.9989672  1.0000000  0.9962277
0.9804018
## subseccd10 subseccd11 subseccd12 subseccd13 subseccd14 subseccd15
subseccd16
##  1.0000000  0.8333333  0.9814156  0.9993827  0.8269647  1.0000000
1.0000000
## subseccd17 subseccd19 subseccd20 subseccd23 subseccd25 subseccd26
subseccd27
##  1.0000000  1.0000000  1.0000000  0.0000000  0.9908257  1.0000000
0.0000000
## subseccd29
##  1.0000000

apply(cluster_mat_25, 2, function(x) x[7] / sum(x))

## subseccd03 subseccd04 subseccd05 subseccd06 subseccd07 subseccd08
subseccd09
```

```
##  0.9786723  0.9918855  0.9974846  0.9985867  1.0000000  0.9951989
0.9659481
## subseccd10 subseccd11 subseccd12 subseccd13 subseccd14 subseccd15
subseccd16
##  1.0000000  0.8333333  0.9608541  0.9987654  0.7649603  1.0000000
1.0000000
## subseccd17 subseccd19 subseccd20 subseccd23 subseccd25 subseccd26
subseccd27
##  1.0000000  1.0000000  1.0000000  0.0000000  0.9847095  1.0000000
0.0000000
## subseccd29
##  0.9000000

table(max_names)

## max_names
## subseccd03 subseccd04 subseccd05 subseccd06 subseccd07 subseccd08
subseccd09
##     212494       9366       7951      18397       7531       2916
4082
## subseccd10 subseccd11 subseccd12 subseccd13 subseccd14 subseccd15
subseccd16
##        988          6       2529       1620       1387         85
8
## subseccd17 subseccd19 subseccd20 subseccd23 subseccd25 subseccd26
subseccd27
##         42       3159          1          2        327          6
3
## subseccd29
##         10
```

In both the 16-cluster and 25-cluster versions, there is a disappointingly uninteresting trend: one cluster dominates all the others and contains the vast majority of observations, while the rest of the clusters are mostly sparse. From the other direction, we also find that the majority of subsections are completely or almost completely consumed by this one dominating cluster. In the 16-cluster case, the dominating cluster contained at least 83% of every subsection, often reaching into the higher 90s. This is especially true for subsection 3. As mentioned earlier in this report, subsection 3 is an umbrella subsection that includes many different kinds of mission-based nonprofits. Despite being the largest subsection by number of organizations and having the widest scope of missions, over 99% of subsection 3 landed in the dominating cluster.

The cluster analysis failed to show, as we had hoped to find, distinctive financial structures between subsections. Instead, it actually showed us the opposite: across all subsections, most organizations had extremely similar finances– not on any one metric, but in the aggregate. This supports our finding from the LASSO analysis, where we saw that every subsection feature had been penalized to 0.

Where do we go from here?

In both the LASSO and clustering analysis, we found evidence that nonprofits throughout the market have more in common than not. Subsections, in particular, were neither especially useful in predicting revenue nor grouping organizations based on their finances. As analysts and consultants hoping to make a difference by serving nonprofits, this is a good sign: it means that a strategy that works in one sector of the market may be applicable elsewhere. Obviously, business strategy is much more complicated than 990 forms alone, but this analysis gives us hope that with further research, we can uncover even more insightful information– and make an impact.

Future analyses should consider incorporating 990 returns from previous years, both to analyze the time series of organizations that have been around for a long time and to understand the growth of younger ones. The IRS currently published 990 extracts dating back to 2013, and it is possible to learn which organizations have filed for all years. Clustering with other algorithms may find a more insightful way to group organizations based on their finances, and other predictive models can be fit to better understand how different features lead to revenue growth. At the same time, we can also turn our attention to the various components of revenue, and build models that predict fundraising, program revenue, or grant funding, each of which has a distinct and substantial business case separate from this analysis.

Critiques of Other Teams' Work

Of Nicki and Hart

Nicki and Hart do a great job of establishing the use case for their research and explaining how their methodology helps them achieve their stated goals. In reading their report, I appreciate the amount of detail they gave in explaining where the data came from and what information it contains, as well as the specific models and evaluation metrics used to analyze that data. Their writing is clear and concise, though it can be a bit jargon-y at times. A simple improvement would be to explain how the fitted model can improve the audience's experience and/or outcomes. It's difficult at least at this stage to know what recommendations the authors would make based on the findings of their data. Another suggestion I would offer, even before the one just mentioned, is to go into more detail in explaining the data sets. What are the types of features included? What are their ranges? Do they need scaling, normalization, or transformation? As a reader, I trust that your findings are accurate and predictive (especially given the satisfactory RMSE and sMAPE measures), but without fully understanding the data before seeing the results of the models, it's difficult to fully contextualize it. Overall, though, this research is in a great state, and I look forward to seeing what your final results are.

Of Hudson and Qi

Overall, this is a phenomenal piece of analysis. Hudson and Qi do an excellent job laying out the case for their research– to build a tool that can recognize, interpret, and classify handwritten characters in Mandarin– by describing what has already been done, what the

use cases are, and what methods they use to solve their problem. They have clearly done a tremendous amount of work on the data collection and processing side, especially given that they used a form of data completely outside the scope of our class. I am impressed both with the accuracy of the models they fit and the ease with which they evaluated their performance, and I can't help but be proud that the Random Forest model was more successful than the neural network! If I had to make a suggestion, it would be to explain in more detail why each of the three models were chosen (logistic regression, random forest, and neural network), and to put that in the context of the eventual performance. Are there situations in which a less-predictive model would be preferable to a more-predictive one?

Of Derin

I believe Derin does a wonderful job of integrating her financial knowledge into the final project. Her expertise appears when it comes time for feature engineering and applying hypothesis tests that work toward her end goal of determining the most significant indicators of development level in OECD countries. The project utilizes the WDI (world development indicator) Data set from the world bank which provides information on emerging markets. Through my reading of the code, I appreciated the amount of detail supplied when outlining what battery of models she would be attempting and her rational for doing so. At this stage in the project (looking just through the code in the comments) The writing is a little wordy but definitely progresses logically and guides me through her thought process. An improvement I could suggest, would be to simplify some of the code (using dpylr functions etc.) and make it dynamic. Additionally, it may help to walk the reader through the intention of some of the transformations. However, the way in which Derin handles the NAs is logical and effective. Another suggestion I would offer, is to go into some detail about the data sets during the write up (What are the types of features included? What are their ranges? Do they need scaling, normalization, or transformation?). The features and breakdown of the dataset aren't mentioned much in the code and I got a little confused not having seen it before. Additionally, I think the confusion matrix function for recall and precision may be useful for evaluating your models. Overall, I believe the project is off to a great start. I look forward to seeing what your final results are.