STAT 515 Final Project Report

I. Why Police Stops Data?

I chose to do my final project using a police stops dataset from the Stanford Open Policing Project (SOPP). I believe the US police has perpetuated a number of systemic problems and continues to discriminate against minority and impoverished communities. Based on knowledge of previous research into police stops, I started this analysis with a few questions to answer. Does race influence police stop outcomes? Does the time of stop – day or night – affect the number of stops more for one race over another? Do more police stops happen in areas with lower socio-economic status? What are the reasons for police stops and do they vary by sex?

The SOPP has compiled a large database of police stop data from cities across the United States on their website. I chose to work with the Nashville, Tennessee dataset because it was more complete with many variables to explore over a long period of time. It technically covers a 10 year span from 2010 to 2019, but I only use 9 years as I will explain later. This includes over three million observations and forty-four variables with each row representing one police stop. In total it takes up one gigabyte of storage on my computer.

II. Additional Data Sources – Population, Income, Geospatial

To perform some analyses I added three supplemental datasets. When exploring racial components, I used 2018 Nashville census data to scale each race's police stop count by their population. This is a good scaling method, but it's not perfect as not every person stopped by police in Nashville actually lives in Nashville. To include this data, I copied each race's population from the Nashville census.gov quickfacts table into Excel and loaded with R's *read_excel* function.

I wanted to compare police stops and median income by zip code in two maps side by side. This required downloading a Davidson county shapefile from data.nashville.gov with zip code boundaries. The shapefile can be read in R through the *sf* package, which loads a dataframe with post office names, zip codes, and geometries (a list of polygon type rows with latitude, longitude coordinate pairs to draw zip code boundaries).

I got median income data through over 1,300 API requests to justicemaps.org. I created a few R functions to accomplish this – one that computes a unique set of latitudes and longitudes from the Nashville shapefile *geometry* list and one that loops over this set and makes an API *get* request for each coordinate pair. This returned median income by census tract with latitude and longitude variables that I could join back to the original police stop data.

III. Data Description – Police Stops

Returning to our main police stops dataset, Figure 1 shows a set of sixteen variables explored in this project. There are temporal variables (date, year, month, time), geospatial variables (latitude and longitude), and demographic variables for the person stopped (age, race, sex). Each stop also indicates the violation, or reason for the stop, and the outcome – either a mere warning, a citation, or an arrest. Lastly, the police recorded a few logical variables, whether a frisk was performed, a search conducted, contraband found, drugs found, or weapons found. Figure 1 uses the R *glimpse* function to show the variable types as well as the first few observations for each variable. Near the

bottom I can see contraband is False for the fifth observation when a search was conducted, while the other rows are missing. I can guess from this pattern (and confirm with a frequency table) that contraband found is only recorded when a search was conducted, and logically this makes sense.

```
Rows: 3,078,116
Columns: 16
$ date              <date> 2010-10-10, 2010-10-10, 2010-10-10, 2010-10
$ year              <dbl> 2010, 2010, 2010, 2010, 2010, 2010, 2010, 20
$ month             <dbl> 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,
$ time              <time>       NA, 10:00:00, 10:00:00, 22:00:00, 01:
$ lat               <dbl> 36.19, 36.16, 36.12, 36.09, 36.18, 36.29, 36
$ lng               <dbl> -86.80, -86.74, -86.90, -86.65, -86.81, -86.
$ subject_age       <int> 27, 18, 52, 25, 21, 26, 37, 33, 33, 49, 18,
$ subject_race      <fct> black, white, white, white, black, white, wh
$ subject_sex       <fct> male, male, male, male, male, female, male, r
$ violation         <fct> investigative stop, moving traffic violation
$ outcome           <fct> warning, citation, warning, warning, warning
$ frisk_performed   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL
$ search_conducted  <lgl> FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FALS
$ contraband_found  <lgl> NA, NA, NA, NA, FALSE, NA, NA, NA, NA, NA, NA
$ contraband_drugs  <lgl> NA, NA, NA, NA, FALSE, NA, NA, NA, NA, NA, NA
$ contraband_weapons <lgl> NA, NA, NA, NA, FALSE, NA, NA, NA, NA, NA, NA
```

Figure 1: Glimpse of Police Stops Data

The majority of the variables are categorical or logical, not continuous, and thus making a histogram or scatterplot matrix is out of the question. Instead I will begin data exploration using base R's *count* function to compute frequencies over a subset of variables. Fortunately, SOPP included a great tutorial on their website that aided my EDA process.

IV. Exploratory Data Analysis (EDA)

First up is the count of police stops by year (Table 1). Two trends immediately stand out – 2019 has many fewer stops than all other years and stops spike in 2014 then decline each following year. Regarding the former lack of counts, all further analyses will exclude 2019 as it is not a full representative year. I can confirm by looking at counts by year and month as well (Table 2).

year	n
<dbl>	<int>
2010	310622
2011	393248
2012	444146
2013	412695
2014	413114
2015	357261
2016	297248
2017	245565
2018	204217
2019	14235

Table 1: Stops by Year

	1	2	3	4	5	6	7	8	9	10	11	12
2010	28282	25731	29698	28185	19448	25065	25525	26211	27093	27246	26047	22091
2011	29657	27671	35503	33074	36627	33060	31776	37483	32953	32706	30010	32728
2012	43890	38925	39540	38375	40475	38409	33439	35429	33056	35723	35804	31081
2013	41200	35689	37861	34561	35777	35279	34797	32700	30952	33160	30573	30146
2014	41359	30539	39989	39304	37273	35599	30681	30561	32686	33051	31697	30375
2015	37520	26897	32198	34771	33258	34566	28875	28703	27467	26412	23784	22810
2016	29846	27796	29978	26222	24403	26242	24629	24323	23414	20774	19745	19876
2017	23877	21163	21723	22149	20515	19956	22387	21476	18699	19360	17746	16514
2018	25358	20609	22828	21079	18129	17014	17574	17996	14781	13770	10369	4710
2019	5814	4398	4023	0	0	0	0	0	0	0	0	0

Table 2: Stops by Year and Month

Next I examined the break down of police stops by subject race and sex in Table 3. I see that some sexes and races are missing, unknown, or other. I chose to remove these rows instead of imputing values as they don't account for many rows and there is plenty of data left after removal.

subject_race	male	female	`NA`
<fct>	<int>	<int>	<int>
asian/pacific islander	26852	14666	150
black	644046	517844	3981
hispanic	116355	47903	556
white	1004390	660698	5785
other	7628	2757	12
unknown	26654	7970	2254
NA	1118	648	84

Table 3: Stops by Race and Sex

With stop counts higher for males than females for every race, I move on to look at stop violations, or the stop reason, by sex in Figure 2. Except for child restraint at the very bottom, males are stopped by police more frequently than females across every violation category. The top two violations – moving traffic and vehicle equipment violations – make up the vast majority of stops, accounting for 2.5 out of 3 million stops.

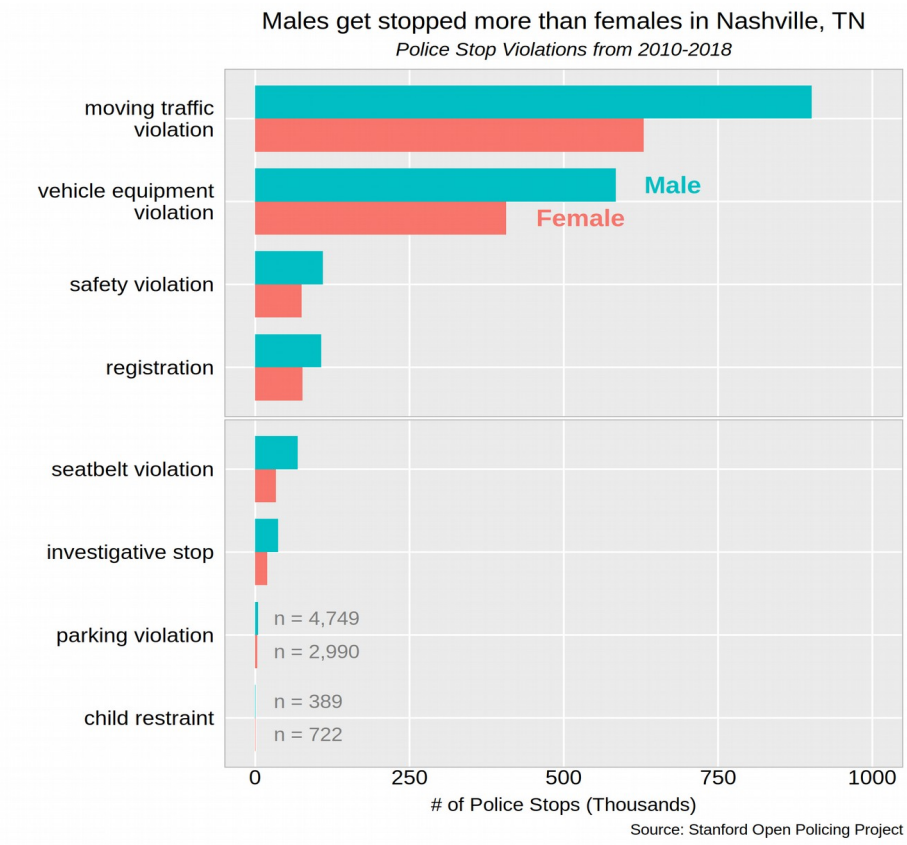

Figure 2: Stops by Violation Type and Sex

I can use the plot in Figure 3 to see if any new patterns emerge regarding outcome discrepancy by race. By loading the Nashville census data I can scale the stop counts by a race's population and show the number of stops per one thousand persons by race across the three outcomes. Since I only

added census data for 2018, I focused only on police stops in 2018 as well. I abbreviated Asian/Pacific Islander to just Asian to more effectively use plotting space. Across each outcome category – warning, citation, and arrest – black persons are stopped the most per one thousand persons, followed by white, hispanic, and lastly asian. I combined two perspectives of outcome by race in a juxtaposed plot, where the right plot with a free x scale shows an insight not revealed in the left plot – that hispanic persons are near or above white persons in stops that end with citations or arrests with many fewer stops resulting in a warning.

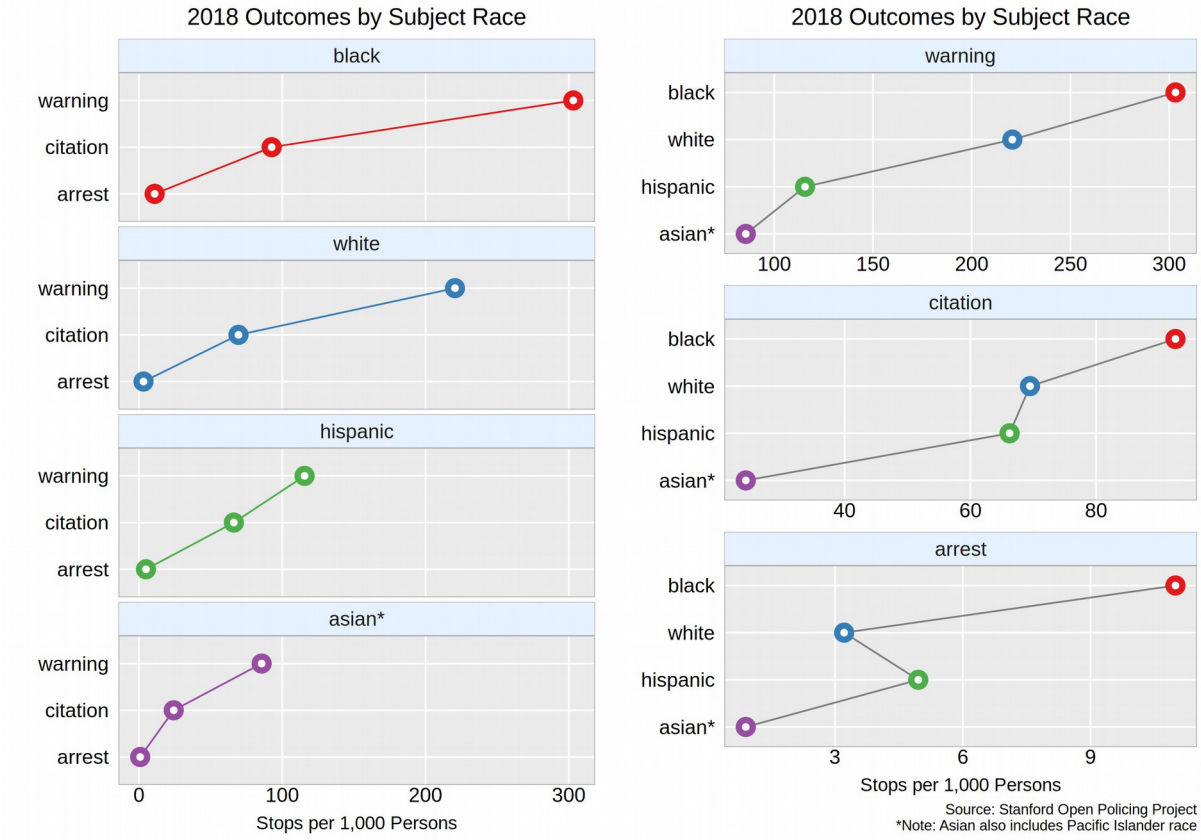

Figure 3: 2018 Stop Outcomes by Race Juxtaposed Plot

I remember reading a study by Stanford and NYU researchers who performed "a large-scale analysis of racial disparities in police stops across the United States. [Their] results indicate that

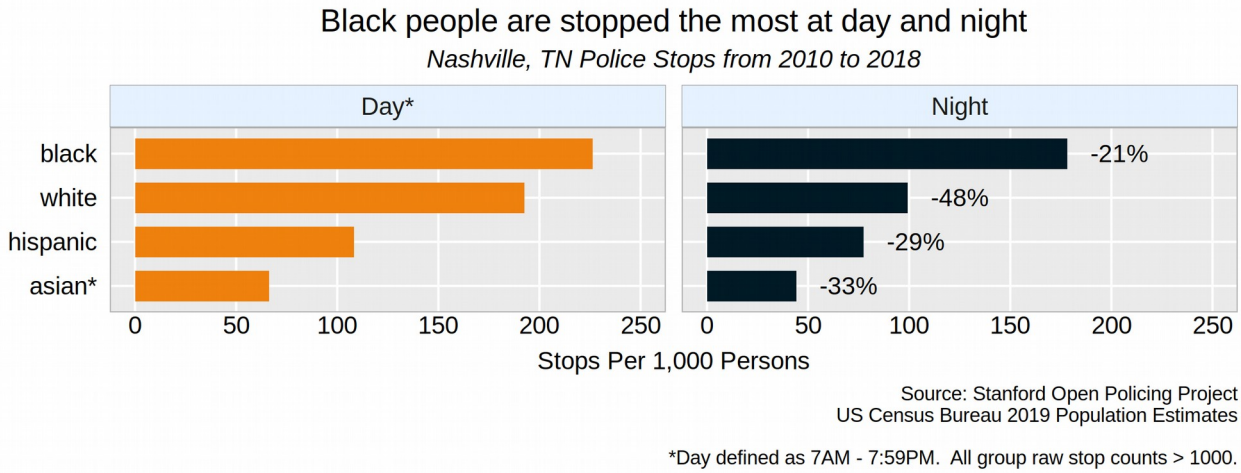

Figure 4: Stop Counts by Race Comparison - Day vs Night Hours

police stops and search decisions suffer from persistent racial bias and point to the value of policy interventions to mitigate these disparities" (E. Pierson et. al, 2020). In my limited analysis with crude day to night time definitions, I did not find a discrepancy in police stops by race with black persons having the least percentage drop in stops from day to night and the most overall stops during day and night hours. Coincidentally, the study authors published the nationwide data they collected and it is publicly available on the SOPP website – my main source for this analysis. These researchers cleverly used the daylight savings time change to test their police stop hypotheses. This method is much more statistically sound than what I did, so my findings here in no way discredit theirs. This only serves as a reminder that I should strive to collect as much accurate data as possible and transform it in appropriate ways to not mislead readers with visuals.

V. Predictive Modeling - Classification

A. Linear Discriminant Analysis (LDA)

My dataset has a natural variable to use as the response – the outcome of the police stop. I could consider performing logistic regression (LR) to predict this categorical variable. The ISLR textbook notes that linear discriminant analysis is better than LR "when the classes are well-separated, when *n* is small and the each predictor's distribution is approximately normal, and when there are more than two response classes" (Gareth, et. al, 2013). Though I didn't test for predictor normality, the data has many observations and *outcome* has 3 classes, so I will begin with LDA.

```
outcome ~ subject_age + subject_race + subject_sex + contraband_found
Call:
lda(fmla, data = train)

Prior probabilities of groups:
 warning citation   arrest
  0.3591   0.3372   0.3037

Group means:
         subject_age subject_raceblack subject_racehispanic subject_racewhite
warning        31.35            0.5172              0.04775            0.4263
citation       30.33            0.5639              0.10249            0.3286
arrest         32.15            0.5276              0.09629            0.3700
         subject_sexfemale contraband_foundTRUE
warning             0.2526               0.02297
citation            0.2068               0.35919
arrest              0.2354               0.24521

Coefficients of linear discriminants:
                           LD1       LD2
subject_age           0.001823  0.079884
subject_raceblack     0.668571 -0.218396
subject_racehispanic  1.657754  1.283185
subject_racewhite     0.291075  0.005408
subject_sexfemale    -0.164455  0.675418
contraband_foundTRUE  2.558508  0.061821

Proportion of trace:
   LD1    LD2
0.9742 0.0258

           warning citation arrest  class.error
  warning    12385     7025   7516       0.5400
  citation     654     5399   3344       0.4255
  arrest       352      382    303       0.7078
[1] "Test MSE: 0.4841"
```

Figure 5: LDA Model 1 Results, Confusion Matrix, and Test MSE

I wrote a quick R function[1] to fit the model on the training set, make predictions from the test set, and output the formula, model results, confusion matrix, and test mean squared error (MSE). The first model used a subject's age, race, sex, and the logical contraband found as predictors. This subset was based on my intuition about what might predict a police stop, but in hindsight I think using principal component analysis, random forest variable importance, or lasso regression as feature selection methods would have been better choices. The prior probabilities are stated at the top of the results summary and indicate the probabilities for each class within the given dataset. One part that stands out is the group means for contraband found with much larger means for the citation and arrest classes than the warning class. This means that subjects that are searched and found with contraband are more likely to receive a citation or be arrested rather than let off with a warning. This makes logical sense as I would expect persons with drugs or weapons found on them to be punished more harshly than those without any contraband. Surprisingly, though, the group mean for citation is actually higher than that of arrest. The confusion matrix shows that the model fit is not that great with more cases being predicted incorrectly than correctly for the warning and arrest classes. I can use the test MSE of 0.4841 to compare performance to future LDA models. LDA model 2 used the predictors time, violation, sex, and contraband drugs and had a better test MSE of 0.4662. LDA model 3 had the worst test MSE of all the LDA models at 0.4903 with the predictors subject race, sex, frisk performed, and contraband found.

B. Quadratic Discriminant Analysis (QDA)

While LDA assumes that each response class shares a common covariance matrix and has substantially lower variance for doing so, QDA assumes each class has its own covariance matrix.

```
R> fit.qda(fmla = fmla2, train = train, test = test)
outcome ~ time + violation + subject_sex + contraband_drugs
Call:
qda(fmla, data = train)

Prior probabilities of groups:
 warning citation   arrest
  0.3591   0.3372   0.3037

Group means:
          time violationinvestigative stop violationmoving traffic violation
warning  46749                    0.05376                           0.4332
citation 49077                    0.05303                           0.4165
arrest   42469                    0.09902                           0.4564
         violationparking violation violationregistration
warning                   0.002675               0.06345
citation                  0.002714               0.06300
arrest                    0.001356               0.06362
         violationsafety violation violationseatbelt violation
warning                   0.06087                     0.03398
citation                  0.05635                     0.05452
arrest                    0.05126                     0.03492
         violationvehicle equipment violation subject_sexfemale
warning                                0.3518            0.2526
citation                               0.3534            0.2068
arrest                                 0.2927            0.2355
         contraband_drugsTRUE
warning               0.0172
citation              0.3083
arrest                0.1831

           warning citation arrest   class.error
  warning    11965     7790   7808        0.5659
  citation     728     4297   2222        0.4071
  arrest       699      721   1135        0.5558
[1] 0.4656
```

Figure 6: QDA Model 2 Results, Confusion Matrix, and Test MSE

QDA can have better performance by trading higher variance for lower bias when there are many training observations and reducing variance is not critical. Between all 6 models (3 LDA, 3 QDA), the QDA model 2 with predictors time, violation, subject sex, and contraband drugs performed the best with a test MSE of 0.4656. The confusion matrix shows that this model still predicts many of the cases incorrectly, but has predicted the arrests class more accurately than the other models did.

C. Random Forest (RF) Classification

Finally, I also tested classifying the stop outcome variable with a few RF models with the default 500 trees and 3 variables tried at each split. The second model with 10 variables had an out of bag error (OOB) error rate of 45.91% and was the best of the three RF models I fit. While this model's error rate is rather close to fifty percent (a coin toss), it did seem to predict the warning class much better than either LDA or QDA did.

```
Call:
 randomForest(formula = outcome ~ ., data = data_subset,
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 3

        OOB estimate of  error rate: 45.91%
Confusion matrix:
         warning citation arrest class.error
warning    24817     3625   2950      0.2094
citation   11774    14601   3096      0.5046
arrest     11302     7383   7860      0.7039
```

Figure 7: Random Forest Model 2 Results

Variable importance plots like the one in Figure 8 can help an analyst decide which variables are influential in predicting the response. I can see that time, subject age, month, and year are often found in trees grown in the random forest model and are important predictors. This can be used for feature selection in other models or like I did in future RF models. With 10 variables, I took the top half from both plots and used them as features in my last RF model. This reduced the variable count to 7 and while I thought this would end up being the best RF model, it resulted in a worse OOB error rate of 47.16%.

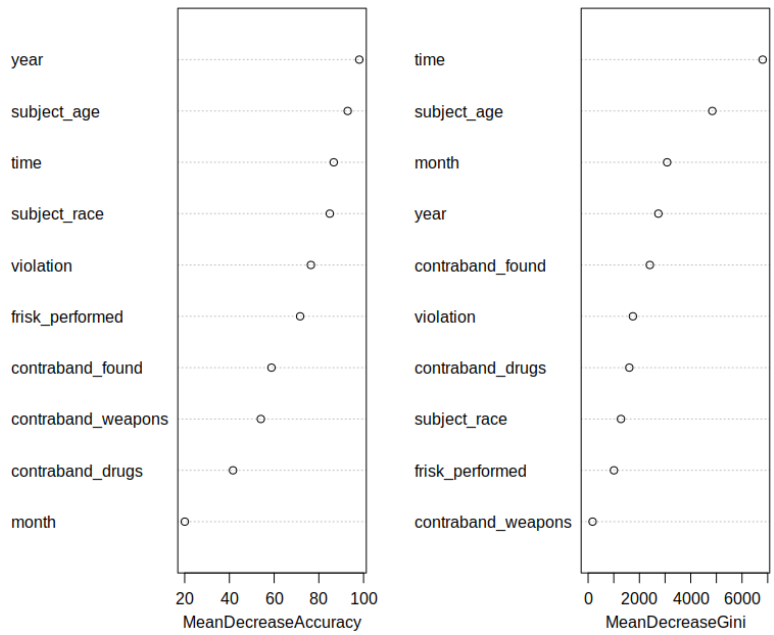

Figure 8: Model 2 Variable Importance Plot

Sometimes it may be useful to look at an outcome class prototype like the juxtaposed plot in Figure 9. I adopted code from the iris classification lab and since my dataset had many more observations I cut my data to only 150,000 observations and switched from a scatterplot to a hex bin heatmap plot. Higher bin counts are represented in red and lower in blue. A few patterns emerge here – the majority of stops that ended in a warning are of persons around age 25 late at night (10PM – 3AM); more arrests occur during this time frame as well. Younger people tend to be stopped more often older people. A few very old folks (around 100 years) are also being stopped, though it seems that only one of them got arrested.

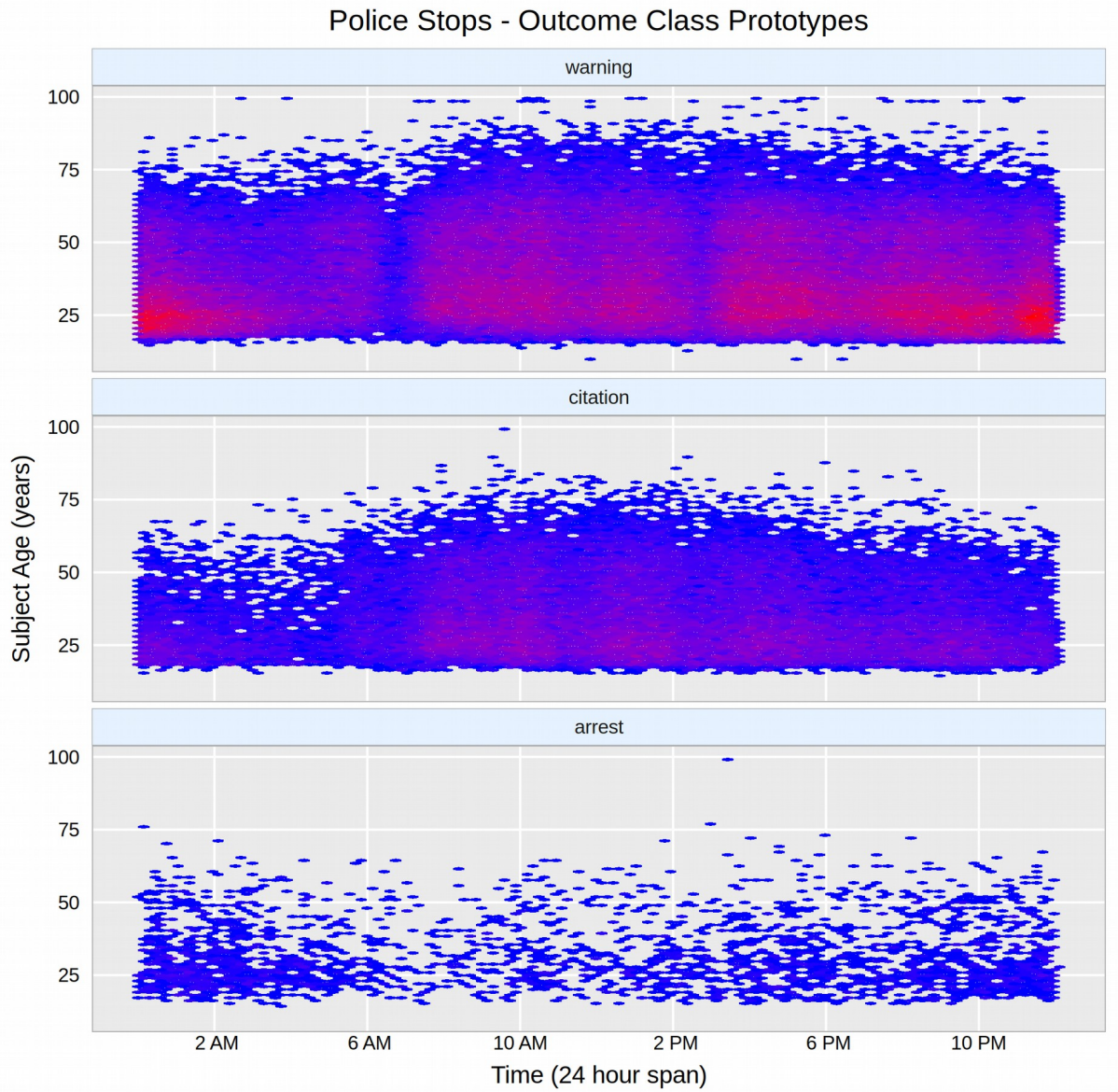

Figure 9: Police Stops Outcome Class Prototypes

VI. Interactive Juxtaposed Choropleth Map

One question left – do more police stops happen in areas with lower socio-economic status? To answer this I loaded a Davidson county zip code shapefile with the *sf* package and converted my police stops data with latitude, longitude pairs to an *sf* object with the *st_as_sf* converter function. Now both the shape object and the police stops data are *sf* objects and I can count the number of stops within each zip code polygon boundary and filter out zip codes with no stops.

Next I sent this dataframe to a new function to extract a unique set of coordinate pairs to loop over, sending each one to the justicemaps.org API in a *get* request. I linked each API result back to its coordinate pair so now I have zip codes linked to coordinates, post office names, police stops, and median income. At this point I tried many interactive plotting packages – plotly, ggmap, leaflet – but plotly's output didn't allow me to customize the popup information and juxtapose two plots, while ggmap and leaflet plain didn't work at all and only filled my terminal with errors. I tried the *tmap* package and I liked their maps much better as they could be customized with familiar ggplot2 additive syntax and they looked very polished.

This prepared data is ready to be passed to *tmap*'s suite of functions to produce an interactive choropleth map. Similar to *gridExtra*'s *arrange* function, I made two maps and combined them vertically with *tmap_arrange* with the *sync* parameter set to True. This sync setup allows me to zoom and click and drag both plots simultaneously (albeit with some lag at times). Also, tmap shows a hover tooltip with the post office name and zip code in each plot waiting for the user to click a region resulting in a pop up with that region's police stop counts and median income.

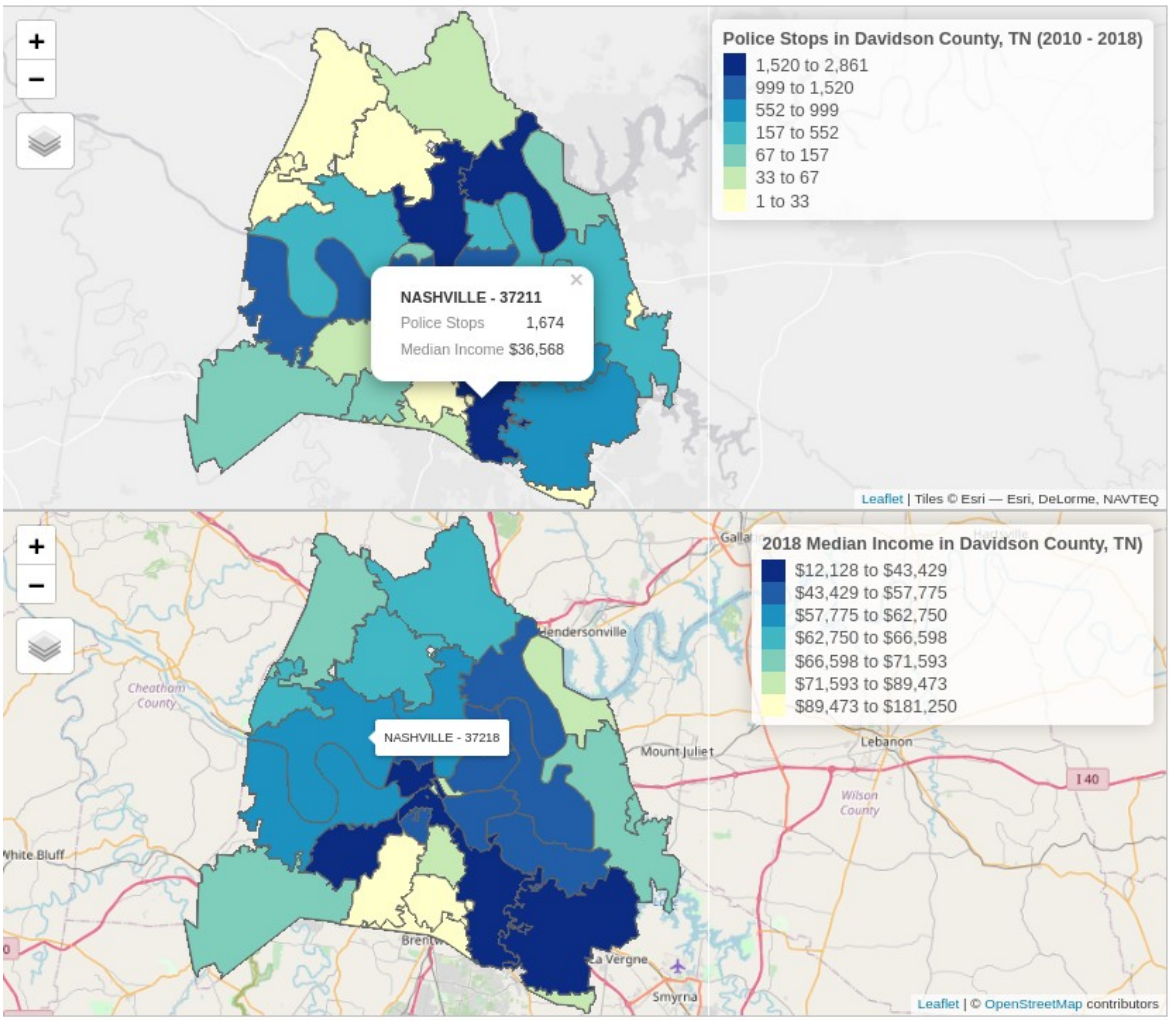

Figure 10: Juxtaposed Choropleth Plot - Police Stops (top) vs Median Income (bottom)

Police stops and median income were encoded in the sequential color scheme where dark blue indicates the highest bin of police stops and the lowest median income bin, while yellow has the lowest stop counts and the highest median income. Values in green fell in the middle bins. This means that the median income scale is reversed relative to how I would generally present this data

(low at bottom to high at top), but I felt comparison could be more effective if the parts I thought were correlated were coded in the same colors. There seems to be some correlation here between more police stops and lower median income as some zip codes with low stop counts (lighter green and yellow) have higher income and many zip codes with a high stop counts (dark blue) have lower incomes. This is not conclusive as visual determinations are often subjective, but it's a starting point and may lead to a statistically significant result if tested.

VII. Conclusion

While frustrating at times, I thoroughly enjoyed this project and the earlier graphic redesign project as they let me practice many of the concepts taught in this course and throughout this masters program. It is rather hard to correctly predict outcomes of police stops from the data I utilized, but I came away with a few new insights from twisting and turning the data and viewing it from other perspectives. Some next steps might include using data from more cities across the United States, performing analyses on a high-powered distributed computer, and testing formal hypotheses that could apply to a more generalized population.

VIII. References

E. Pierson, C. Simoiu, J. Overgoor, S. Corbett-Davies, D. Jenson, A. Shoemaker, V. Ramachandran, P. Barghouty, C. Phillips, R. Shroff, and S. Goel. "A large-scale analysis of racial disparities in police stops across the United States". Nature Human Behaviour, Vol. 4, 2020.

Engel, C. (2019). Using Spatial Data with R. Retrieved 1 May 2021, from https://cengel.github.io/R-spatial/

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. (2013). An introduction to statistical learning : with applications in R. New York :Springer, 2013.

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani. (2013). Linear & Quadratic Discriminant Analysis · UC Business Analytics R Programming Guide. (2013). Retrieved 4 May 2021, from https://uc-r.github.io/discriminant_analysis

Geospatial vector data in R with sf. Retrieved 1 May 2021, from https://ourcodingclub.github.io/tutorials/spatial-vector-sf/

Justice Map. (2020). Retrieved 1 May 2021, from http://www.justicemap.org/include_map.php

Nashville Open Data Portal. (2021). Retrieved 1 May 2021, from https://data.nashville.gov/browse?tags=shapefile

Nuno, C. (2018). Configure error in rgdal "gdal-config." Retrieved 1 May 2021, from https://stackoverflow.com/questions/48668535/configure-error-in-rgdal-gdal-config

Pascual, C. (2020). R API Tutorial: Getting Started with APIs in R – Dataquest. Retrieved 1 May 2021, from https://www.dataquest.io/blog/r-api-tutorial/

U.S. Census Bureau QuickFacts: Nashville-Davidson (balance), Tennessee; Davidson County, Tennessee. (2019). Retrieved 1 May 2021, from https://www.census.gov/quickfacts/fact/table/nashvilledavidsonbalancetennessee,davidsoncountytennessee/PST045219

You, C. (2018). how to merge a shapefile with a dataframe with latitude/longitude data. Retrieved 1 May 2021, from https://stackoverflow.com/questions/50140707/how-to-merge-a-shapefile-with-a-dataframe-with-latitude-longitude-data

Appendix

[1] This R model fitting function and its intended use is given below:

```r
fit.lda <- function(fmla, train, test) {
    print(fmla)

    lda.fit <- lda(fmla, data = train)
    print(lda.fit)

    lda.pred <- predict(lda.fit, test)

    print(table(lda.pred$class, test$outcome))
    print(paste0("Test MSE: ", round(mean(lda.pred$class == test$outcome, na.rm=TRUE), 4)))
}

# Model 1 with variables below (test mse: 0.4841)
fmla1 <- outcome ~ subject_age + subject_race + subject_sex + contraband_found
fit.lda(fmla = fmla1, train = train, test = test)
```

R scripts were used for all data collection, cleaning, preparation, modeling, and visualizations. All R scripts, input data files, and graphics produced can be viewed on my github at https://github.com/doug-cady/gmu_daen/tree/master/STAT515/M7_Final_project.