Playground

Frances Hung

10/21/2017

StoryMap

https://arcg.is/1fDKLD

Motivation

According to the CDC, suicide was the 10th leading cause of death in the US in 2015, and the 2nd leading cause of death among adolescents and young adults. Psychological disorders, particularly depression, are a significant risk factor for suicide especially when they go untreated. There is no reliable way to predict who is at risk for committing suicide, because most screening approaches depend on self-report information and people contemplating on suicide would often deny it when asked.

In the first part of our project, we hence aim to build a logistic regression model to identify important variables in predicting suicide rates. Due to the limits of our data, we consider the period from 2004-2015, within the scope of cities in California. One interesting explanatory variable we use is Google search term data (under the product of "Google Trends"). Our hypothesis is that individuals are more likely to tell the truth to Google, than on a questionnaire. In the second part of our project, we build a series of maps using the ArcGIS software. Using suicide rate and mental health treatment facilities data as well as Google search term data, our project aims to map the demand for and supply of mental health treatment in California cities.

Ultimately, we hope to shed some light on important explanatory variables correlated with suicide rates (with the regression model), and to help identify cities where there is a large treatment service gap (with the maps) so that we can address this problem in a more data-driven way.

- use result/visualization as hook

Variable choice (to be moved to preceding corresponding R chunks)

Ideally, the response variable that we are interested in is the gap between the demand and supply of mental health treatment. Which areas are over/under-served, and why? This would be very useful information to policy makers, mental health service providers, related non-profits and such. However, such a variable does not exist (or we could not find it), and we would have had to create an algorithm to derive this data from other existing variables. We could not decide on an accurate way to code "demand" (and what weights to give each component). Furthermore, even though "supply" is more straightforward, there also exists discrepancies between the size of the facilities, or the affordability of the services that would need to be captured by our variable. In the end, we decided that we would use suicide rate as a response variable, although we agreed that it would be an interesting extension to look at service gap. We also hope that our GIS maps would help our audience to begin to think about and identify areas which are under-served.

The original datasets we start with include: - List of verified mental health treatment clinics and facilities (downloaded from ReferenceUSA). We only included places with a certified psychiatrist or psychologist, and which focuses on general mental health (excluding substance abuse facilities) - Google search frequency by city on "depression" as a mood (to exclude unrelated searches on economic depression etc) from 2004-2015 (downloaded from Google Trends). The "hits" values are calculated on a scale from 0 to 100, where 100 is the location with the most popularity as a fraction of total searches, where 50 indicates a location which is half as popular and so on. - Number of suicides by zipcode from 2004-2015. We downloaded leading causes of death data from California Health and Human Services Agency and filtered for cause of death is suicide. - Demographic data downloaded from SimplyAnalytics, including racial and gender makeup, age, marriage, education level, employment, income, healthcare, etc. - Cities long lat data (if possible, could we find a dataset which has a more exhaustive list, or I could ask Warren..)

Playground

We originally intended to look at suicide rate and Google Trends data from one year, eg. 2015, but the logit model returned no significant variables as both suicide rate and depression search fluctuate a lot each year, influenced by factors like celebrity suicides which are not directly relevant to population mental health. Hence, we decided to aggregate suicide rate and Google Trends data over 12 years (constrained by data availability), from 2004 to 2015. Since demographic information is fairly stable over time, we used demographic information from the most recent year to train our model.

```
require(gtrendsR)
```

```
## Loading required package: gtrendsR
```

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(dplyr)
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
require(zipcode)
```

```
## Loading required package: zipcode
```

```
data("zipcode")
require(ggmap)
```

```
## Loading required package: ggmap
```

Making Dataframes

Longitude and latitude of cities (for mapping them later)

```
cities_longlat <- read.csv("cal_cities.csv", header = TRUE) %>% select(c(location,
    Latitude, Longitude))
```

Full gtrends data for all cities for depression search

```
gtrends_full <- read.csv("gtrends_20042015_full.csv") %>% `colnames<-`(c("location",
    "hits"))
```

Prepping facilities data to find number of facilities per city

```
facilities <- read.csv("facilities_final.csv", header = TRUE)
colnames(facilities)[7] <- "zip"
facilities$zip <- as.character(facilities$zip)
city_facs <- facilities %>% group_by(City) %>% summarise(facility_cnt = n())
colnames(city_facs)[1] <- "location"
```

Prepping suicide data to find number of suicides per city, 2004-2015

```
suis <- read.csv(file = "death.csv", header = TRUE) %>% filter(Causes.of.Death ==
    "SUI") %>% filter(Year >= 2004)
colnames(suis)[2] <- "zip"
suis$zip <- as.character(suis$zip)
suis2 <- inner_join(zipcode, suis, by = "zip")
# aggregate suicide data across all the years for each city
city_suis <- suis2 %>% group_by(city) %>% summarise(suicides = sum(Count))
colnames(city_suis)[1] <- "location"

# wrangled the data for the purpose of GIS (to use later)
zip_suis <- suis %>% group_by(zip) %>% summarise(suicides = sum(Count))
```

Adding in city demographic data for 2017

```
citydem <- read.csv("citydems.csv", header = TRUE)
citydem2 <- read.csv("citydems2.csv", header = TRUE)
citydem2$Name <- gsub(",.*", "", citydem2$Name)
citydem$Name <- gsub(",.*", "", citydem$Name)
citydem$FIPS <- NULL
citydem2$FIPS <- NULL
colnames(citydem) <- c("location", "male", "female", "healthcare.per.household",
    "bluecollar", "whitecollar", "nonfamily", "medAge", "NativeAm", "whiteNonHisp",
    "hispanic", "white", "black", "asian", "medIncome", "lessHS", "HS", "Bachelors",
    "pop", "unmarriedMpop", "unemployed")
citydem <- inner_join(citydem, city_facs, by = "location")
# Remove the Burbank and Mountain View entries that refer to
# census-designated areas (duplicate names with the actual cities will
# create problems later if not removed)
citydem <- citydem[-c(124, 38), ]
citydem2 <- citydem2[-c(636, 803), ]
citydem$facility_cnt <- citydem$facility_cnt * 1e+05/citydem$pop
colnames(citydem2) <- c("location", "healthcare.per.person", "activities.per.person",
    "socialRec.per.person", "entertainment.per.person", "poverty", "presdrugs.per.person",
    "healthcarebiz.per.1000")

# data weangling for GIS mapping
zipcode_dem <- read.csv("explansToViz-zipcode.csv")
zipcode_dem$Name <- gsub(",.*", "", zipcode_dem$Name)
zipcode_dem$FIPS <- NULL
colnames(zipcode_dem)[1] <- "zip"
zipcode_dem$zip <- as.character(zipcode_dem$zip)
zipcode_dem2 <- zipcode_dem %>% left_join(zip_suis, by = "zip") %>% left_join(zipcode,
    by = "zip") %>% filter(state == "CA")
write.csv(zipcode_dem2, "gis_zip_dem.csv")
# add city area (in square miles) info from GIS
landArea <- foreign::read.dbf("LandCity.dbf")
landArea <- landArea[, c(1, 3)]
```

```r
colnames(landArea) <- c("location", "landArea")
```

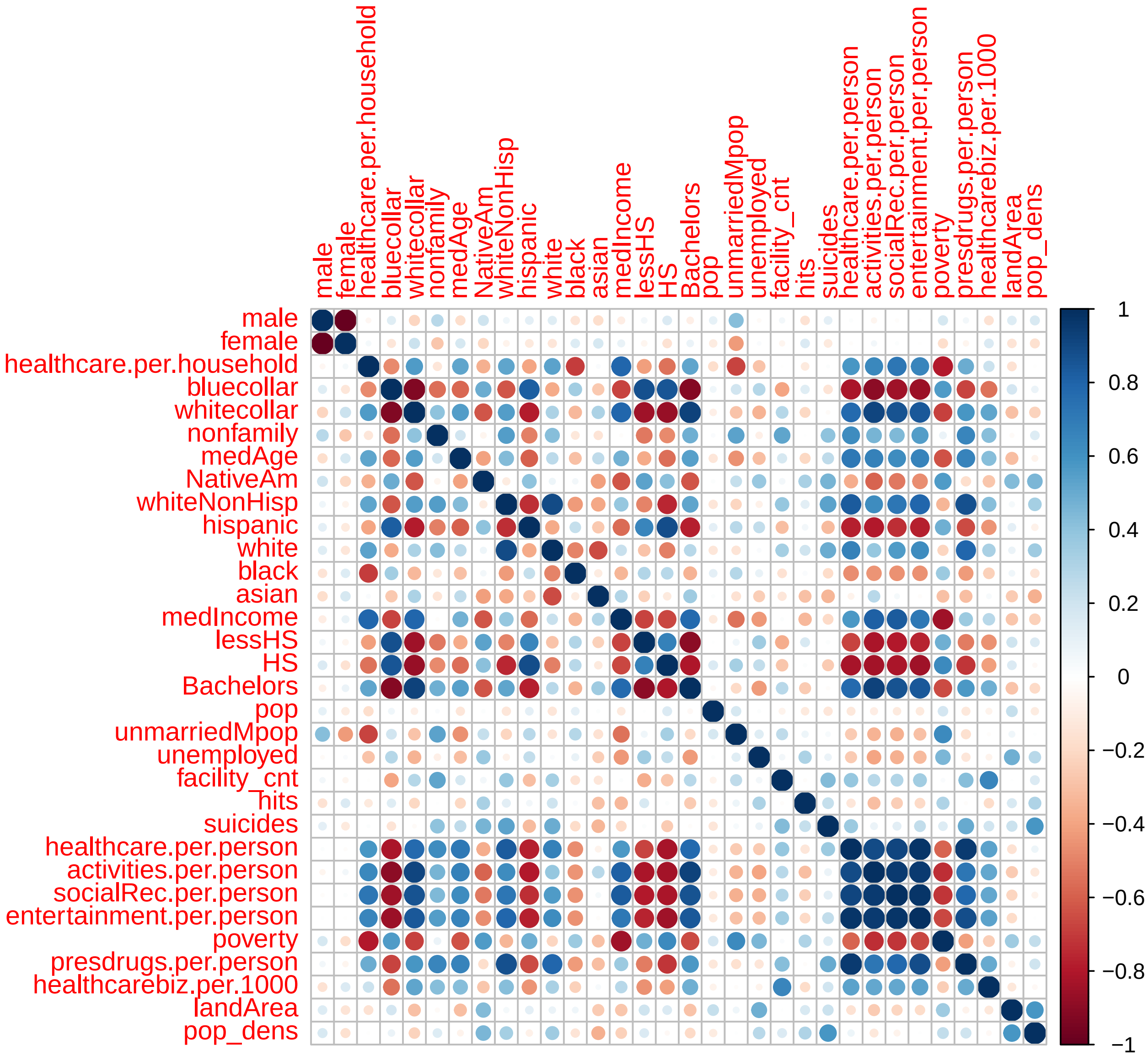
```r
logtable <- inner_join(citydem, gtrends_full, by = "location") %>% inner_join(city_suis,
    by = "location") %>% inner_join(citydem2, by = "location") %>% inner_join(landArea,
    by = "location") %>% mutate(suicides = suicides * 1e+05/pop, healthcare.per.person = healthcare.per
    activities.per.person = activities.per.person/pop, socialRec.per.person = socialRec.per.person/pop,
    entertainment.per.person = entertainment.per.person/pop, presdrugs.per.person = presdrugs.per.perso
    healthcarebiz.per.1000 = healthcarebiz.per.1000 * 1000/pop, pop_dens = landArea/pop)
```

If we want to reliably determine significant variables, we want to ensure that variables aren't collinear. Looking at the correlation plot of variables in logtable, we see significant correlation between some variables.

```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
corrplot(cor(logtable[, -1]))
```



To determine which variables to remove, we look at the vif and choose variables with the highest vifs to discount in the final analysis.

```
library(DAAG)
```

```
## Loading required package: lattice
```

```
set.seed(35)
model_full <- lm((suicides) ~ ., data = logtable[, -1])
vif(model_full)
```

```
##                    male healthcare.per.household                bluecollar
##                  2.8127                  67.6270                   17.2920
##             whitecollar                nonfamily                    medAge
##                 34.1760                  46.0270                   37.6470
##                NativeAm             whiteNonHisp                  hispanic
##                  3.3911                 831.7000                  505.1600
##                   white                    black                     asian
##                119.1100                  83.9960                  288.0200
##               medIncome                   lessHS                        HS
##                 86.4740                  14.7260                   27.0480
##               Bachelors                      pop              unmarriedMpop
##                 21.6630                   1.3614                   12.4010
##              unemployed             facility_cnt                      hits
##                  2.0731                   2.7173                    2.0191
##    healthcare.per.person    activities.per.person     socialRec.per.person
##              11670.0000                1325.1000                 2377.8000
## entertainment.per.person                  poverty     presdrugs.per.person
##               9880.4000                  13.5430                 2132.9000
##   healthcarebiz.per.1000                 landArea                  pop_dens
##                  3.2588                   3.2733                    2.9997
```

```
logtable <- logtable %>% select(-c(activities.per.person, entertainment.per.person,
    female, whiteNonHisp, healthcare.per.person, socialRec.per.person, white,
    nonfamily, healthcare.per.household))
# now the table has 174 cities, whereas the full list of gtrends had 200.
# Not a big loss
write.csv(logtable, "logtable.csv")

logtable_crop <- logtable[, -c(1)]
row.names(logtable_crop) <- logtable$location
# normalize the explanatory variables data frame
logtable_crop_normalized <- scale(logtable_crop) %>% data.frame()
row.names(logtable_crop_normalized) <- logtable$location


# for purpose of identifying which cities to map
zip.no <- zipcode_dem2 %>% group_by(city) %>% summarise(zip.n = n())
colnames(zip.no)[1] <- "location"
# gistable <- logtable %>% select(c(1,19,24)) %>%
# left_join(zip.no,by='location') We wanted to choose one city with high
# suicide rate, and one with a low suicide rate. The two cities should have
# a similar population, and should have a min of 4 zipcodes (so that
# plotting variables at the zipcode level will be more useful). We
# eventually decided on Inglewood and Santa Barbara.
```

Build a logit model to predict suicide rate at the city level

```r
library(caret)
trains <- createDataPartition(logtable_crop$suicides, p = 0.75, list = FALSE)
logtable_crop.train <- logtable_crop[trains, ]
logtable_crop.test <- logtable_crop[-trains, ]
set.seed(35)
model_full <- lm((suicides) ~ ., data = logtable_crop.train)
summary(model_full)
```

```
##
## Call:
## lm(formula = (suicides) ~ ., data = logtable_crop.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -86.348 -18.940  -3.162  16.659 216.254
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -8.358e+01  2.940e+02  -0.284 0.776716
## male                   -9.180e-02  4.411e+00  -0.021 0.983432
## bluecollar              4.694e-02  1.962e+00   0.024 0.980951
## whitecollar            -6.937e-01  1.493e+00  -0.465 0.643184
## medAge                  4.193e+00  2.340e+00   1.791 0.075993 .
## NativeAm                5.283e+01  1.131e+01   4.673 8.53e-06 ***
## hispanic               -1.092e+00  5.271e-01  -2.072 0.040603 *
## black                  -6.573e-01  9.877e-01  -0.666 0.507125
## asian                  -3.816e-01  7.449e-01  -0.512 0.609481
## medIncome               1.259e-04  4.655e-04   0.270 0.787358
## lessHS                  1.617e+00  1.938e+00   0.834 0.405956
## HS                     -1.304e+00  1.673e+00  -0.779 0.437417
## Bachelors              -5.597e-01  1.654e+00  -0.339 0.735637
## pop                    -7.572e-06  1.081e-05  -0.701 0.484985
## unmarriedMpop           8.676e-01  1.268e+00   0.684 0.495188
## unemployed             -8.615e+00  3.700e+00  -2.329 0.021720 *
## facility_cnt            1.869e+00  6.382e-01   2.928 0.004155 **
## hits                    1.354e-01  6.791e-01   0.199 0.842344
## poverty                 2.376e+00  1.425e+00   1.667 0.098333 .
## presdrugs.per.person    5.716e+02  9.138e+02   0.626 0.532907
## healthcarebiz.per.1000 -1.963e+00  2.118e+00  -0.927 0.356075
## landArea               -3.582e-03  4.559e-02  -0.079 0.937507
## pop_dens                1.557e+04  4.279e+03   3.638 0.000421 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.15 on 109 degrees of freedom
## Multiple R-squared:  0.769,  Adjusted R-squared:  0.7224
## F-statistic:  16.5 on 22 and 109 DF,  p-value: < 2.2e-16
```

Significant variables include: Facility density (#mental health facilities/100,000 people) (+), % population native (+), median age (+), healthcare spending per person (big -), social/recreation/gym club spending (big -), prescription drug spending (big +), population density (+), unemployment rate (-), and movies/parks/museum spending (big +).

```r
tests <- predict(model_full, logtable_crop.test)
toCompare <- data.frame(cbind(actuals = logtable_crop.test$suicides, predicts = tests))
cor(toCompare)
```

```
##            actuals  predicts
## actuals  1.0000000 0.8312722
## predicts 0.8312722 1.0000000
```

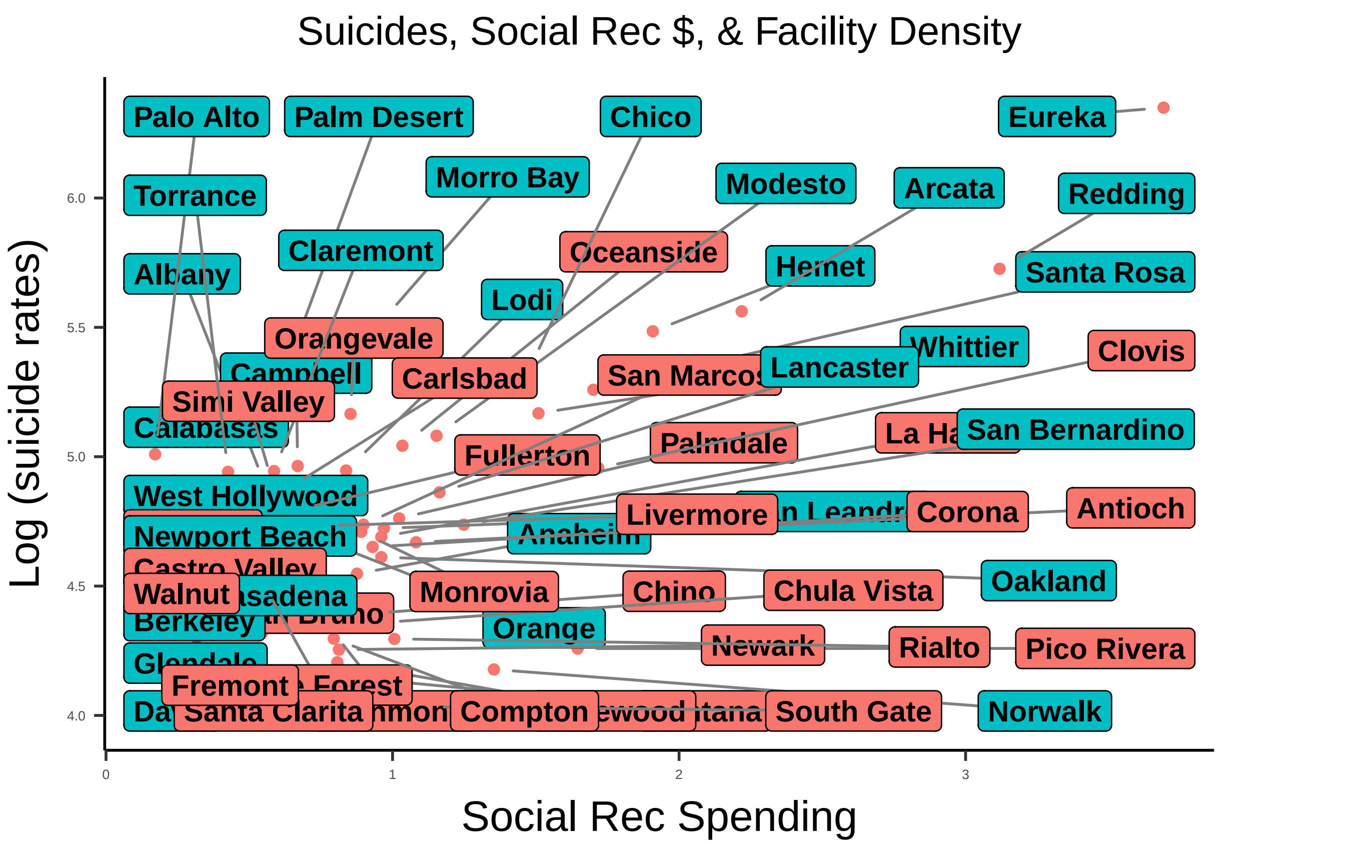
```r
(toCompare)
```

```
##                  actuals  predicts
## West Hollywood 123.11480 205.25581
## Vacaville      165.23025 156.40852
## Santa Ana       69.95208  56.46925
## Mountain View   91.32077  81.78453
## Atascadero     177.69731 240.55011
## San Francisco  121.80783 119.59885
## Salinas        109.72449  86.68892
## Santa Clara     87.62636  64.07802
## Lancaster      129.32528 180.47245
## Redwood City   167.21796 147.15297
## Santa Barbara  250.22724 209.63283
## Gilroy         102.12380 157.32435
## Napa           178.67335 211.87874
## Anaheim         94.43216  85.82320
## Fontana         62.77373  43.47071
## Fremont         74.99414  74.19682
## Livermore      113.60441 175.01110
## Chico          209.40649 264.66368
## Orangevale     175.07865 173.97275
## Hayward        122.48867 105.68701
## Santa Clarita   54.26561 109.53487
## Fairfield      123.20074 127.88079
## Yucaipa        191.26789 164.47000
## Thousand Oaks   93.08074 112.17246
## Lodi           140.70592 158.79241
## Pleasanton      94.60125 126.32952
## Mission Viejo  117.81195 133.31367
## Campbell       143.17568 143.40807
## Modesto        160.92483 147.98396
## Irvine          78.57901  65.90735
## Antioch        106.59548 126.83759
## Torrance       139.98912 108.69880
## Compton         73.44360  47.37406
## Tustin          88.95794  77.67593
## Vallejo        138.59514 133.93500
## Bellflower      66.68764  92.23991
## Camarillo      167.09797 144.25556
## Cerritos        78.59009  99.17101
## Menlo Park     125.79720 118.84428
## Inglewood       67.05665  66.14706
## Palm Desert    225.44525 226.69318
## Loma Linda      94.71647  67.63777
```

```r
require(ggrepel)
```

```
## Loading required package: ggrepel
```

```r
set.seed(35)
part <- sample(rownames(logtable_crop[logtable_crop$suicides > 0, ]), 60)
locs.toUse <- logtable_crop[part, ]
ggplot(locs.toUse, aes(x = NativeAm, y = log(suicides))) + geom_point(aes(color = facility_cnt >
    0.4)) + geom_label_repel(aes(fill = facility_cnt > quantile(logtable_crop$facility_cnt,
    0.5), label = part), fontface = "bold", color = "black", box.padding = 0.5,
    point.padding = 1, segment.color = "grey50") + theme_classic(base_size = 16) +
    theme(legend.position = "none", plot.title = element_text(size = 15, hjust = 0.5),
        axis.text.x = element_text(size = 5), axis.text.y = element_text(size = 5),
        ) + labs(title = "Suicides, Social Rec $, & Facility Density", y = "Log (suicide rates)",
    x = "Social Rec Spending")
```




```r
# clustering_table<-logtable_full_crop_normalized %>%
# select(presdrugs,socialRec,healthcarepp,facility_cnt,asian,AmInd)
# rownames(clustering_table)<-logtable_full[,1]
# dist_whole<-dist(clustering_table)
# cluster_whole<-hclust(dist_whole,method='centroid') plot(cluster_whole,
# labels=logtable_full[,1]) groups=cutree(cluster_whole,k=20) groups
# x<-cbind(clustering_table, groups) suis_cluster<-function(clus) {
# sd<-logtable_full %>% filter(location %in%
# (rownames(subset(x,groups==clus)))) %>% .[['suicides']] %>% log() %>% sd()
# mean<-logtable_full %>% filter(location %in%
# (rownames(subset(x,groups==clus)))) %>% .[['suicides']] %>% log() %>%
# mean() #View(logtable_full%>% filter(location %in%
```

```
# (rownames(subset(x,groups==clus))))) return(c(mean,sd)) } suis_cluster(5)
# #(lapply(1:12,suis_cluster))
```

```
# set.seed(10) kcluster<-kmeans(clustering_table,20, nstart=20)$cluster
# kcluster y<-cbind(clustering_table,kcluster) logtable_full %>%
# filter(location %in% (rownames(subset(y,groups==17))))
```

}}