Supplemental Code

Amie Thomas

2023-11-13

#load in dataset

```
art <- read_dta("nanda_artsentrec_tract_2003-2017_01P.dta") #art/entertainment
```

#use only 2017 because it has the most up to date information for analysis and also to make computation easier. Getting rid of uninformative columns

```
art_17 <- art|>
  filter(year == "2017")|>
  dplyr::select(tract_fips10, year, population, popden_7111, popden_7112, popden_712, popden_51912,
        popden_7131, popden_7132, popden_7139, popden_71394)
```

#check for 0's and handle

```
missing_values <- any(is.na(art_17))
rows_with_missing <- art_17[!complete.cases(art_17), ]
#some census tracts have a 0 population. Can't do anything with this information so best to remove
#are these true 0's? Possible limitations in the data

art_17_filt <- art_17 |>
  filter(population != 0) #no need for population of 0

miss_again <- any(is.na(art_17_filt)) #no missing values now
```

```
#Northeast (Region 1):
# Connecticut: State Code - 09
# Maine: State Code - 23
# Massachusetts: State Code - 25
# New Hampshire: State Code - 33
# Rhode Island: State Code - 44
# Vermont: State Code - 50
# New Jersey: State Code - 34
# New York: State Code - 36
# Pennsylvania: State Code - 42


#Midwest (Region 2):
# Illinois: State Code - 17
# Indiana: State Code - 18
# Michigan: State Code - 26
# Ohio: State Code - 39
# Wisconsin: State Code - 55
# Iowa: State Code - 19
# Kansas: State Code - 20
# Minnesota: State Code - 27
```

```r
# Missouri: State Code - 29
# Nebraska: State Code - 31
# North Dakota: State Code - 38
# South Dakota: State Code - 46


#South (Region 3):
# Delaware: State Code - 10
# Florida: State Code - 12
# Georgia: State Code - 13
# Maryland: State Code - 24
# North Carolina: State Code - 37
# South Carolina: State Code - 45
# Virginia: State Code - 51
# West Virginia: State Code - 54
# Alabama: State Code - 01
# Kentucky: State Code - 21
# Mississippi: State Code - 28
# Tennessee: State Code - 47
# Arkansas: State Code - 05
# Louisiana: State Code - 22
# Oklahoma: State Code - 40
# Texas: State Code - 48

#West (Region 4)
# Arizona: State Code - 04
# Colorado: State Code - 08
# Idaho: State Code - 16
# Montana: State Code - 30
# Nevada: State Code - 32
# New Mexico: State Code - 35
# Utah: State Code - 49
# Wyoming: State Code - 56
# Alaska: State Code - 02
# California: State Code - 06
# Hawaii: State Code - 15
# Oregon: State Code - 41
# Washington: State Code - 53
```

```r
#match the state census tracts to census regions to easier sampling

get_census_region <- function(code) {
  census_region_map <- c(
    "10" = "South", "12" = "South", "13" = "South", "24" = "South",
    "37" = "South", "45" = "South", "51" = "South", "54" = "South",
    "01" = "South", "21" = "South", "28" = "South", "47" = "South",
    "05" = "South", "22" = "South", "40" = "South", "48" = "South",
    "09" = "Northeast", "23" = "Northeast", "25" = "Northeast",
    "33" = "Northeast", "44" = "Northeast", "50" = "Northeast",
    "34" = "Northeast", "36" = "Northeast", "42" = "Northeast",
    "17" = "Midwest", "18" = "Midwest", "26" = "Midwest", "39" = "Midwest",
    "55" = "Midwest", "19" = "Midwest", "20" = "Midwest", "27" = "Midwest",
    "29" = "Midwest", "31" = "Midwest", "38" = "Midwest", "46" = "Midwest",
    "04" = "West", "08" = "West", "16" = "West", "30" = "West",
```

```r
    "32" = "West", "35" = "West", "49" = "West", "56" = "West",
    "02" = "West", "06" = "West", "15" = "West", "41" = "West", "53" = "West"
  )

  region <- census_region_map[substr(code, 1, 2)]
  if (is.na(region)) {
    region <- "Unknown"
  }

  return(region)
}

art_17_filt <- art_17_filt |>
  mutate(census_region = sapply(tract_fips10, get_census_region))

art_17_filt <- art_17_filt |>
  relocate(census_region, .after = tract_fips10)

unknown_regions <- art_17_filt |>
  filter(census_region == "Unknown")

#all of the tract codes are matched to their respective regions. There are 179 unknowns which don't mak

#we will change these unknowns to correspond with "south" to reflect the actual census region that it i

# Update census_region for census tracts starting with "11" to be "South"
art_17_filt$census_region[startsWith(art_17_filt$tract_fips10, "11")] <- "South"
```

#load in dataset

```r
vice <- read_dta("nanda_lqtbcon_tract_2003-2017_01P.dta") #liquor stores/tobacco
```

#use only 2017 because it has the most up to date information for analysis and also to make computation easier. Getting rid of uninformative columns

```r
vice_17 <- vice|>
  filter(year == "2017")|>
  dplyr::select(tract_fips10, year, population, popden_4453, popden_453991)
```

#check for 0's and handle

```r
missing_values_2 <- any(is.na(vice_17))
rows_with_missing_2 <- vice_17[!complete.cases(vice_17), ]
#some census tracts have a 0 population. Can't do anything with this information so best to remove
#are these true 0's? Possible limitations in the data

vice_17_filt <- vice_17 |>
  filter(population != 0) #no need for population of 0

miss_again_2 <- any(is.na(vice_17_filt)) #no missing values now
```

```r
get_census_region_2 <- function(code_2) {
  census_region_map_2 <- c(
    "10" = "South", "12" = "South", "13" = "South", "24" = "South",
    "37" = "South", "45" = "South", "51" = "South", "54" = "South",
    "01" = "South", "21" = "South", "28" = "South", "47" = "South",
```

```r
    "05" = "South", "22" = "South", "40" = "South", "48" = "South",
    "09" = "Northeast", "23" = "Northeast", "25" = "Northeast",
    "33" = "Northeast", "44" = "Northeast", "50" = "Northeast",
    "34" = "Northeast", "36" = "Northeast", "42" = "Northeast",
    "17" = "Midwest", "18" = "Midwest", "26" = "Midwest", "39" = "Midwest",
    "55" = "Midwest", "19" = "Midwest", "20" = "Midwest", "27" = "Midwest",
    "29" = "Midwest", "31" = "Midwest", "38" = "Midwest", "46" = "Midwest",
    "04" = "West", "08" = "West", "16" = "West", "30" = "West",
    "32" = "West", "35" = "West", "49" = "West", "56" = "West",
    "02" = "West", "06" = "West", "15" = "West", "41" = "West", "53" = "West"
  )

  region_2 <- census_region_map_2[substr(code_2, 1, 2)]
  if (is.na(region_2)) {
    region_2 <- "Unknown"
  }

  return(region_2)
}

vice_17_filt <- vice_17_filt |>
  mutate(census_region = sapply(tract_fips10, get_census_region_2))

vice_17_filt <- vice_17_filt |>
  relocate(census_region, .after = tract_fips10)

unknown_regions_2 <- vice_17_filt |>
  filter(census_region == "Unknown")

vice_17_filt$census_region[startsWith(vice_17_filt$tract_fips10, "11")] <- "South"
```

#load in dataset

```r
social_orgs <- read_dta("nanda_relcivsoc_tract_2003-2017_01P.dta") #social organization
```

```r
social_orgs_17 <- social_orgs|>
  filter(year == "2017")|>
  dplyr::select(tract_fips10, year, population, popden_8131, popden_8134)
```

#check for 0's and handle

```r
missing_values_3 <- any(is.na(social_orgs_17))
rows_with_missing_3 <- social_orgs_17[!complete.cases(social_orgs_17), ]

social_orgs_17_filt <- social_orgs_17 |>
  filter(population != 0) #no need for population of 0

miss_again_3 <- any(is.na(social_orgs_17_filt)) #no missing values now
```

```r
get_census_region_3 <- function(code_3) {
  census_region_map_3 <- c(
    "10" = "South", "12" = "South", "13" = "South", "24" = "South",
    "37" = "South", "45" = "South", "51" = "South", "54" = "South",
    "01" = "South", "21" = "South", "28" = "South", "47" = "South",
    "05" = "South", "22" = "South", "40" = "South", "48" = "South",
```

```
    "09" = "Northeast", "23" = "Northeast", "25" = "Northeast",
    "33" = "Northeast", "44" = "Northeast", "50" = "Northeast",
    "34" = "Northeast", "36" = "Northeast", "42" = "Northeast",
    "17" = "Midwest", "18" = "Midwest", "26" = "Midwest", "39" = "Midwest",
    "55" = "Midwest", "19" = "Midwest", "20" = "Midwest", "27" = "Midwest",
    "29" = "Midwest", "31" = "Midwest", "38" = "Midwest", "46" = "Midwest",
    "04" = "West", "08" = "West", "16" = "West", "30" = "West",
    "32" = "West", "35" = "West", "49" = "West", "56" = "West",
    "02" = "West", "06" = "West", "15" = "West", "41" = "West", "53" = "West"
  )

  region_3 <- census_region_map_3[substr(code_3, 1, 2)]
  if (is.na(region_3)) {
    region_3 <- "Unknown"
  }

  return(region_3)
}

social_orgs_17_filt <- social_orgs_17_filt |>
  mutate(census_region = sapply(tract_fips10, get_census_region_3))

social_orgs_17_filt <- social_orgs_17_filt |>
  relocate(census_region, .after = tract_fips10)

unknown_regions_3 <- social_orgs_17_filt |>
  filter(census_region == "Unknown")

social_orgs_17_filt$census_region[startsWith(social_orgs_17_filt$tract_fips10, "11")] <- "South"
```

#load in dataset

```
ses <- read_dta("nanda_ses_tract_2008-2017_04P.dta") #ses
```

```
ses_select <- ses|>
  dplyr::select(tract_fips10, totpop13_17, ped1_13_17, ped2_13_17, ped3_13_17)
```

#check for 0's and handle

```
missing_values_4 <- any(is.na(ses_select))
rows_with_missing_4 <- ses_select[!complete.cases(ses_select), ]

ses_select_filt <- ses_select |>
  filter(!is.na(totpop13_17) & totpop13_17 != 0)

miss_again_4 <- any(is.na(ses_select_filt)) #check for missing values

rows_with_missing_5 <- ses_select_filt[!complete.cases(ses_select_filt), ]

#some rows have no education information. We don't need these.

cleaned_ses <- na.omit(ses_select_filt)

miss_again_5 <- any(is.na(cleaned_ses))
```

#make region column

```
get_census_region_4 <- function(code_4) {
  census_region_map_4 <- c(
    "10" = "South", "12" = "South", "13" = "South", "24" = "South",
    "37" = "South", "45" = "South", "51" = "South", "54" = "South",
    "01" = "South", "21" = "South", "28" = "South", "47" = "South",
    "05" = "South", "22" = "South", "40" = "South", "48" = "South",
    "09" = "Northeast", "23" = "Northeast", "25" = "Northeast",
    "33" = "Northeast", "44" = "Northeast", "50" = "Northeast",
    "34" = "Northeast", "36" = "Northeast", "42" = "Northeast",
    "17" = "Midwest", "18" = "Midwest", "26" = "Midwest", "39" = "Midwest",
    "55" = "Midwest", "19" = "Midwest", "20" = "Midwest", "27" = "Midwest",
    "29" = "Midwest", "31" = "Midwest", "38" = "Midwest", "46" = "Midwest",
    "04" = "West", "08" = "West", "16" = "West", "30" = "West",
    "32" = "West", "35" = "West", "49" = "West", "56" = "West",
    "02" = "West", "06" = "West", "15" = "West", "41" = "West", "53" = "West"
  )

  region_4 <- census_region_map_4[substr(code_4, 1, 2)]
  if (is.na(region_4)) {
    region_4 <- "Unknown"
  }

  return(region_4)
}

cleaned_ses <- cleaned_ses |>
  mutate(census_region = sapply(tract_fips10, get_census_region_4))

cleaned_ses <- cleaned_ses |>
  relocate(census_region, .after = tract_fips10)

unknown_regions_4 <- cleaned_ses |>
  filter(census_region == "Unknown")

cleaned_ses$census_region[startsWith(cleaned_ses$tract_fips10, "11")] <- "South"
```

#combine all datasets

```
community_data <- left_join(art_17_filt, vice_17_filt, by = "tract_fips10") |>
               left_join(social_orgs_17_filt, by = "tract_fips10") |>
               left_join(cleaned_ses, by = "tract_fips10")

#check for NA's

test <- any(is.na(community_data))

rows_with_missing_6 <- community_data[!complete.cases(community_data), ]

#for some reason, even after cleaning it of na's they pop back in. Since we already know that this is b

community_data <- na.omit(community_data)

test_2 <- any(is.na(community_data))
```

```r
#get rid of duplicate columns

community_data <- community_data |>
  dplyr::select(-census_region.y, -year.y, -population.y, -census_region.x.x, -census_region.y.y, -popul

community_data <- community_data |>
  rename(census_region = census_region.x, year = year.x, population = population.x)

 community_data <- community_data |> #rename for understanding
   rename(
     performing_arts = popden_7111,
     spectator_sports_orgs = popden_7112,
     museums = popden_712,
     libraries = popden_51912,
     amusement_parks = popden_7131,
     casinos = popden_7132,
     recreation = popden_7139,
     fitness = popden_71394,
     liquor_store = popden_4453,
     tobacco = popden_453991,
     religious_orgs = popden_8131,
     social_orgs = popden_8134,
     less_than_hs = ped1_13_17,
     hs_some_college = ped2_13_17,
     bach_and_higher = ped3_13_17,
)
```

#make states column

```r
get_state <- function(code_a) {
  census_state_map <- c(
    "10" = "Delaware", "12" = "Florida", "13" = "Georgia", "24" = "Maryland",
    "37" = "North Carolina", "45" = "South Carolina", "51" = "Virginia", "54" = "West Virginia",
    "01" = "Alabama", "21" = "Kentucky", "28" = "Mississippi", "47" = "Tennessee",
    "05" = "Arkansas", "22" = "Louisiana", "40" = "Oklahoma", "48" = "Texas",
    "09" = "Connecticut", "23" = "Maine", "25" = "Massachusetts",
    "33" = "New Hampshire", "44" = "Rhode Island", "50" = "Vermont",
    "34" = "New Jersey", "36" = "New York", "42" = "Pennsylvania",
    "17" = "Illinois", "18" = "Indiana", "26" = "Michigan", "39" = "Ohio",
    "55" = "Wisconsin", "19" = "Iowa", "20" = "Kansas", "27" = "Minnesota",
    "29" = "Missouri", "31" = "Nebraska", "38" = "North Dakota", "46" = "South Dakota",
    "04" = "Arizona", "08" = "Colorado", "16" = "Idaho", "30" = "Montana",
    "32" = "Nevada", "35" = "New Mexico", "49" = "Utah", "56" = "Wyoming",
    "02" = "Alaska", "06" = "California", "15" = "Hawaii", "41" = "Oregon", "53" = "Washington", "11" =
  )

state <- census_state_map[substr(code_a, 1, 2)]
  if (is.na(state)) {
    state <- "Unknown"
  }

  return(state)
}
```

```r
community_data <- community_data |>
  mutate(state = sapply(tract_fips10, get_state))|>
  relocate(state, .before = year )
```

#collaspe edu levels to no college and college education

```r
community_data$no_CD <- community_data$less_than_hs + community_data$hs_some_college #collaspe into no

community_data <- community_data|> #get rid of dupes and rename
  dplyr::select(-less_than_hs, -hs_some_college)|>
  rename(CD = bach_and_higher )

community_data$college_edu <- ifelse(community_data$CD > 0.5, 1, 0) #make categorical

# 0 = more than 50% of the neighborhood population has less than a college degree
# 1 = more than 50% of the neighborhood population has a college degree
```

#sample proportionately might need to sample at all

```r
# proportions <- table(community_data$census_region) / nrow(community_data) # Calculate proportions
#
# downsize_proportions <- round(proportions * 10000) # Target 10000 total observations
#
# downsample_groups <- function(community_data, target_counts) {
#   sampled_data <- data.frame()  # Create an empty data frame to store the sampled data
#
#   for (i in 1:length(target_counts)) {  # Loop through each group's target count
#     group_subset <- subset(community_data, census_region == names(target_counts)[i])  # Subset the da
#     sampled_indices <- sample(1:nrow(group_subset), target_counts[i])  # Randomly sample indices base
#     sampled_group <- group_subset[sampled_indices, ]  # Store the sampled group data
#
#     sampled_data <- rbind(sampled_data, sampled_group)  # Append sampled rows to the sampled_data dat
#   }
#
#   return(sampled_data)  # Return the final sampled dataset
# }

# sampled_community_data <- downsample_groups(community_data, downsize_proportions)
```

EDA Begins

#proportions of no college to college degree by state

```r
#collaspe all neighborhood 0 and 1's to represent the education level for the overall state
state_proportion <- community_data |>
  group_by(state, college_edu) |>
  summarise(count = n()) |>
  group_by(state) |>
  mutate(total_state = sum(count)) |>
  mutate(proportion = count / total_state)
```

```
## `summarise()` has grouped output by 'state'. You can override using the
## `.groups` argument.
```

```r
#proportion of college education to no college education in each state
```

```r
summary <- summary(community_data)

#table that is easier to read
kable(summary, caption = "Summary of Data")
```

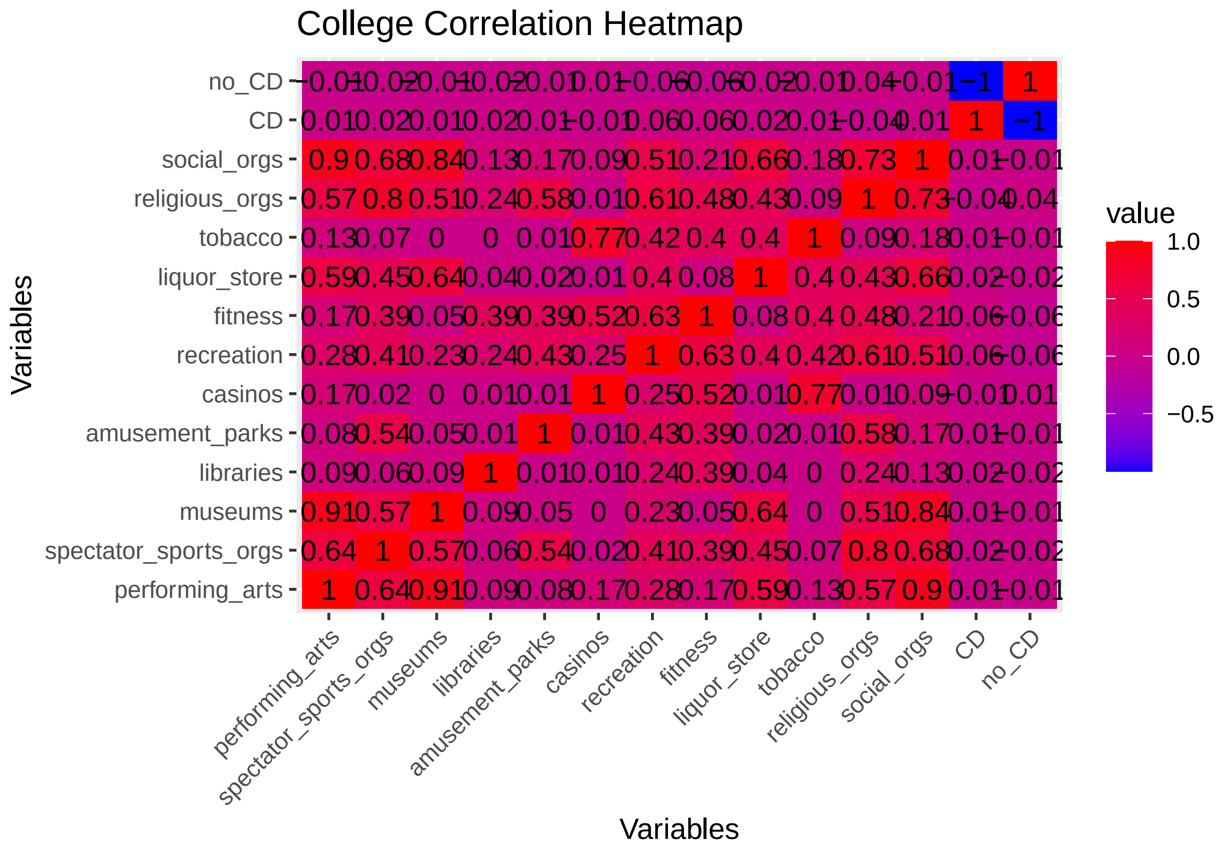
Table 1: Summary of Data

tract_c[illegible]	[illegible]	[illegible]	year	population	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	college_edu
Length:72410	Length:72410	Length:72410	Min. :2017	Min. : 3	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. : 0.0000	Min. :0.0000	Min. :0.0000	Min. :0.0000
Class :character	Class :character	Class :character	1st Qu.:2017	1st Qu.: 2946	1st Qu.: 0.0000	1st Qu.: 0.0000	1st Qu.: 0.0000	1st Qu.: 0.0000	1st Qu.: 0.0000	1st Qu.: 0.0000	1st Qu.: 0.2276	1st Qu.: 0.0000	1st Qu.: 0.0000	1st Qu.: 0.0000	1st Qu.: 0.6204	1st Qu.: 0.000	1st Qu.:0.1493	1st Qu.:0.5930	1st Qu.:0.0000
Mode :character	Mode :character	Mode :character	Median :2017	Median : 4131	Median : 0.0000	Median : 0.0000	Median : 0.0000	Median : 0.0000	Median : 0.0000	Median : 0.0000	Median : 0.5444	Median : 0.1589	Median : 0.0000	Median : 0.0001	Median : 1.1702	Median : 0.353	Median :0.2457	Median :0.7543	Median :0.0000
NA	NA	NA	Mean :2017	Mean : 4433	Mean : 0.2472	Mean : 0.08846	Mean : 0.1744	Mean : 0.1288	Mean : 0.02188	Mean : 0.0156	Mean : 0.8628	Mean : 0.3222	Mean : 0.1532	Mean : 0.0543	Mean : 1.5776	Mean : 0.715	Mean :0.2962	Mean :0.7038	Mean :0.1597
NA	NA	NA	3rd Qu.:2017	3rd Qu.: 5531	3rd Qu.: 0.2791	3rd Qu.: 0.00000	3rd Qu.: 0.1325	3rd Qu.: 0.1614	3rd Qu.: 0.0000	3rd Qu.: 0.0000	3rd Qu.: 1.0023	3rd Qu.: 0.4134	3rd Qu.: 0.2184	3rd Qu.: 0.0001	3rd Qu.: 1.9673	3rd Qu.: 0.712	3rd Qu.:0.4067	3rd Qu.:0.8507	3rd Qu.:0.0000
NA	NA	NA	Max. :2017	Max. :65528	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :[illegible]	Max. :1.0000	Max. :1.0000	Max. :1.0000

```r
edu_count <- community_data |>
  group_by(college_edu)|>
  count()
```

#correlation matrix

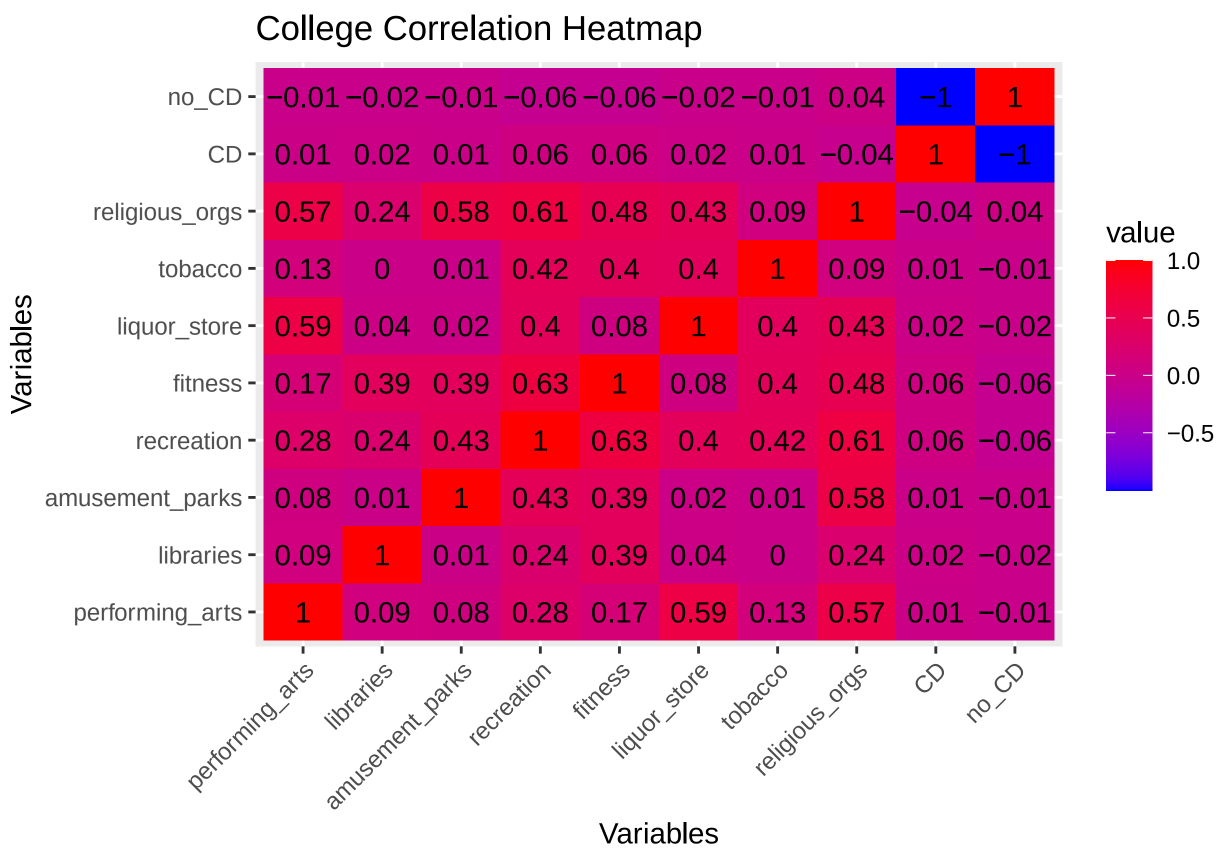
```r
subset_data <- community_data[, 6:19]

cor_matrix <- cor(subset_data)

ggplot(data = reshape2::melt(cor_matrix), aes(Var2, Var1, fill = value)) +
  geom_tile() +
  geom_text(aes(label = round(value, 2)), color = "black") +
  scale_fill_gradient(low = "blue", high = "red") +
  labs(title = "College Correlation Heatmap", x = "Variables", y = "Variables") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
#there are several variables that are correlated with each other above .70. I will compare them to see 
```

#dataset w/ new variables of interest

```
#removed social orgs, casinos, museums, and spec sports
new_com_data <- community_data|>
  dplyr::select(-7, -8, -11, -17)

subset_data_2 <- new_com_data[, 6:15]

cor_matrix_2 <- cor(subset_data_2)

ggplot(data = reshape2::melt(cor_matrix_2), aes(Var2, Var1, fill = value)) +
  geom_tile() +
  geom_text(aes(label = round(value, 2)), color = "black") +
  scale_fill_gradient(low = "blue", high = "red") +
  labs(title = "College Correlation Heatmap", x = "Variables", y = "Variables") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

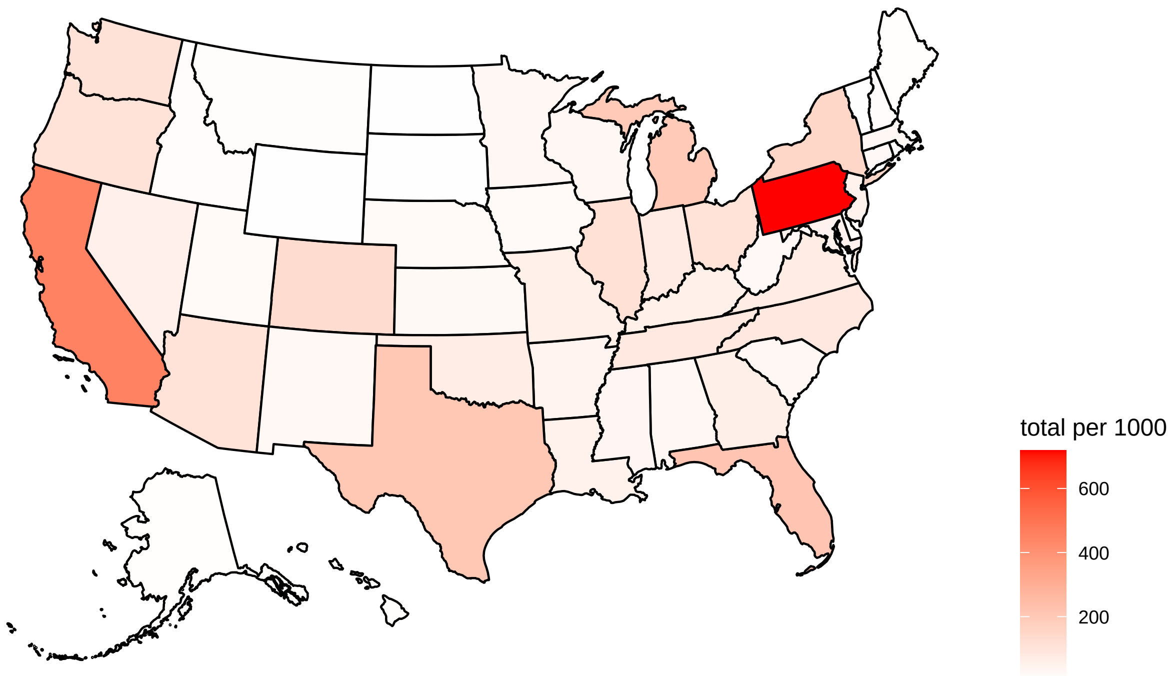


#predictor variables are no longer correlated with each other. Remove highly correlated predictors to minimize overfitting. The presence of highly correlated predictors might lead to an unstable model solution.

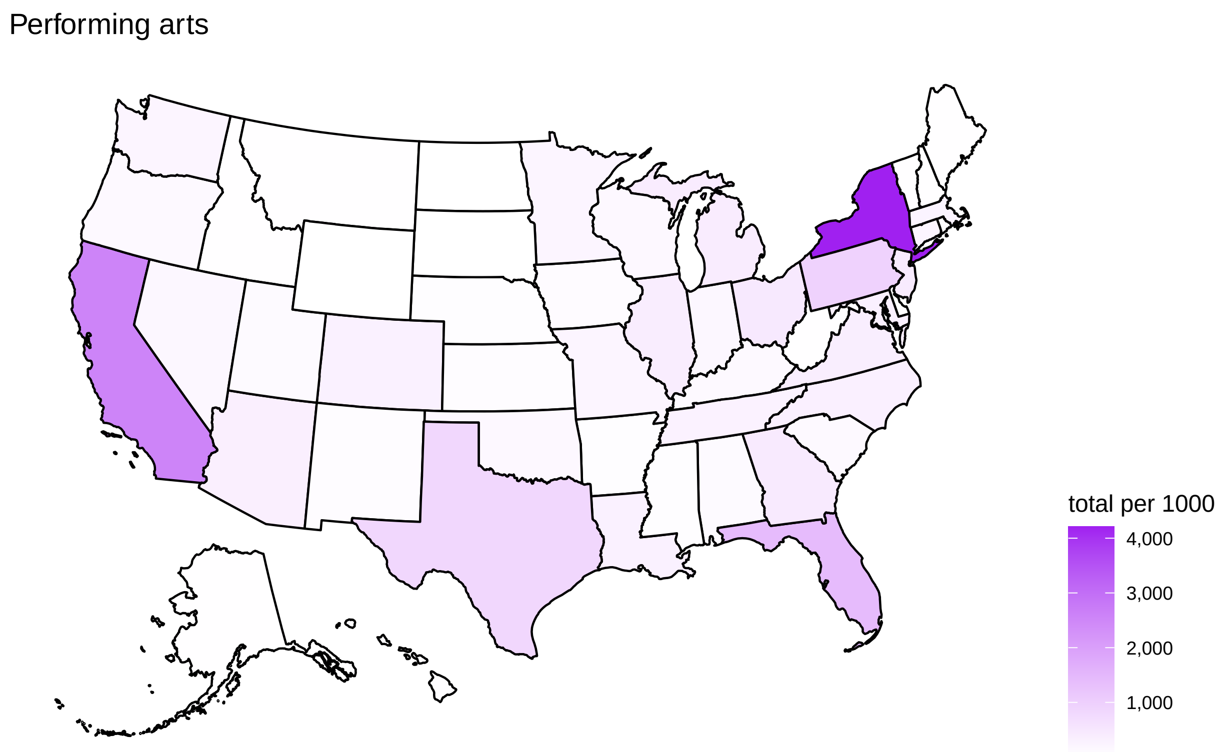
#number of tobacco stores in the US

```
tobacco_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(TotalTobaccoStores = sum(tobacco))

plot_usmap(
   data = tobacco_sum_by_state, values = "TotalTobaccoStores") +
   scale_fill_continuous(
     low = "white", high = "red", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Tobacco stores") +
   theme(legend.position = "right")
```

Tobacco stores

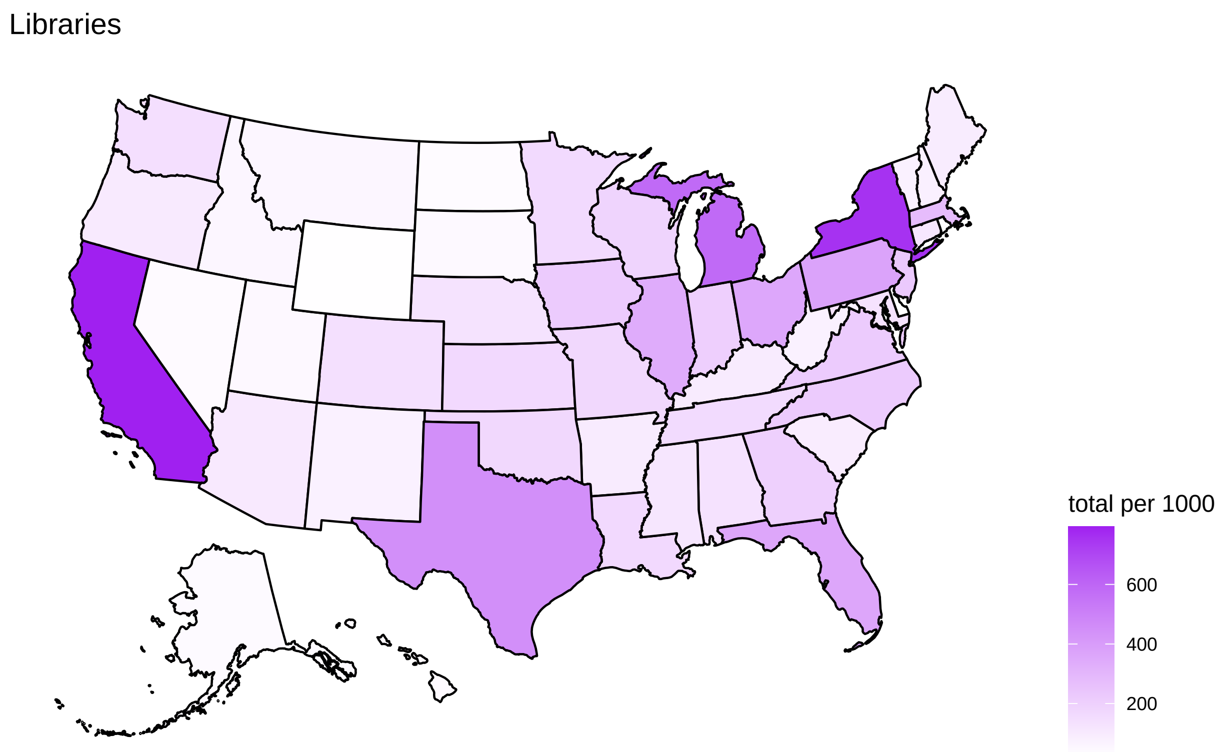



```r
perform_art_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(total_perform_art = sum(performing_arts))

plot_usmap(
   data = perform_art_sum_by_state, values = "total_perform_art") +
   scale_fill_continuous(
     low = "white", high = "purple", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Performing arts") +
   theme(legend.position = "right")
```



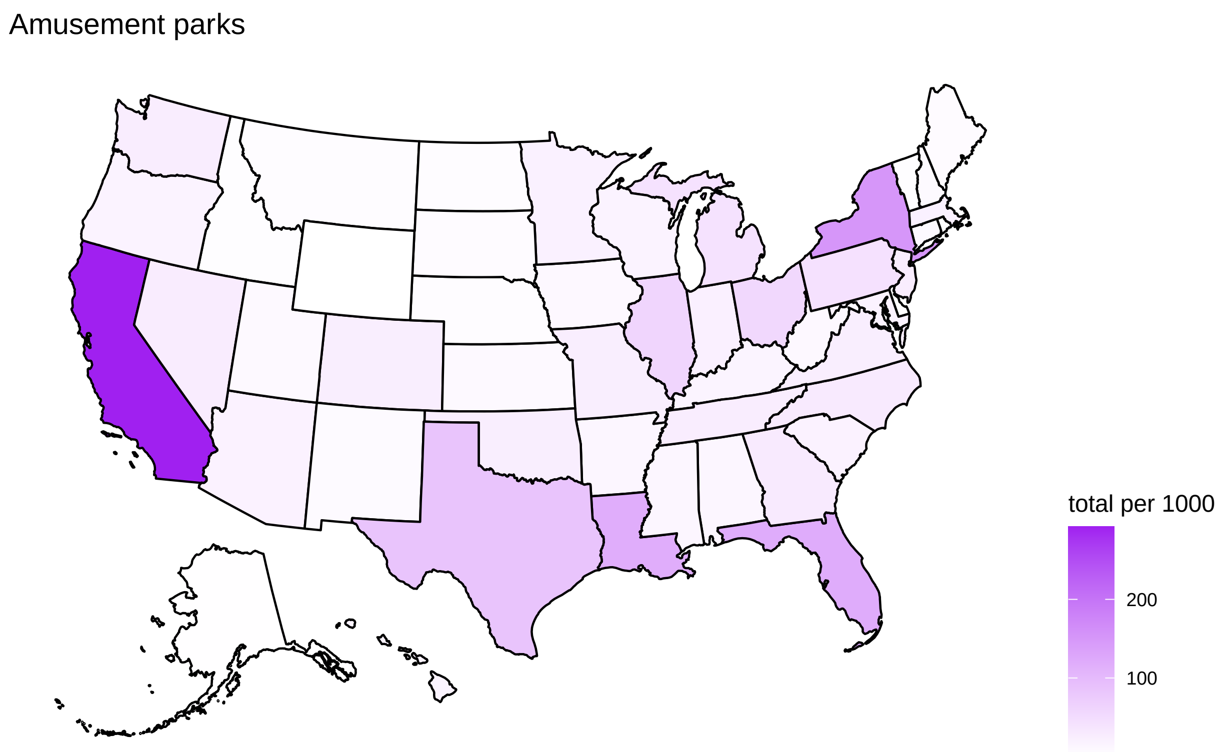
#Libraries

```
lib_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(total_lib = sum(libraries))


plot_usmap(
   data = lib_sum_by_state, values = "total_lib") +
   scale_fill_continuous(
     low = "white", high = "purple", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Libraries") +
   theme(legend.position = "right")
```



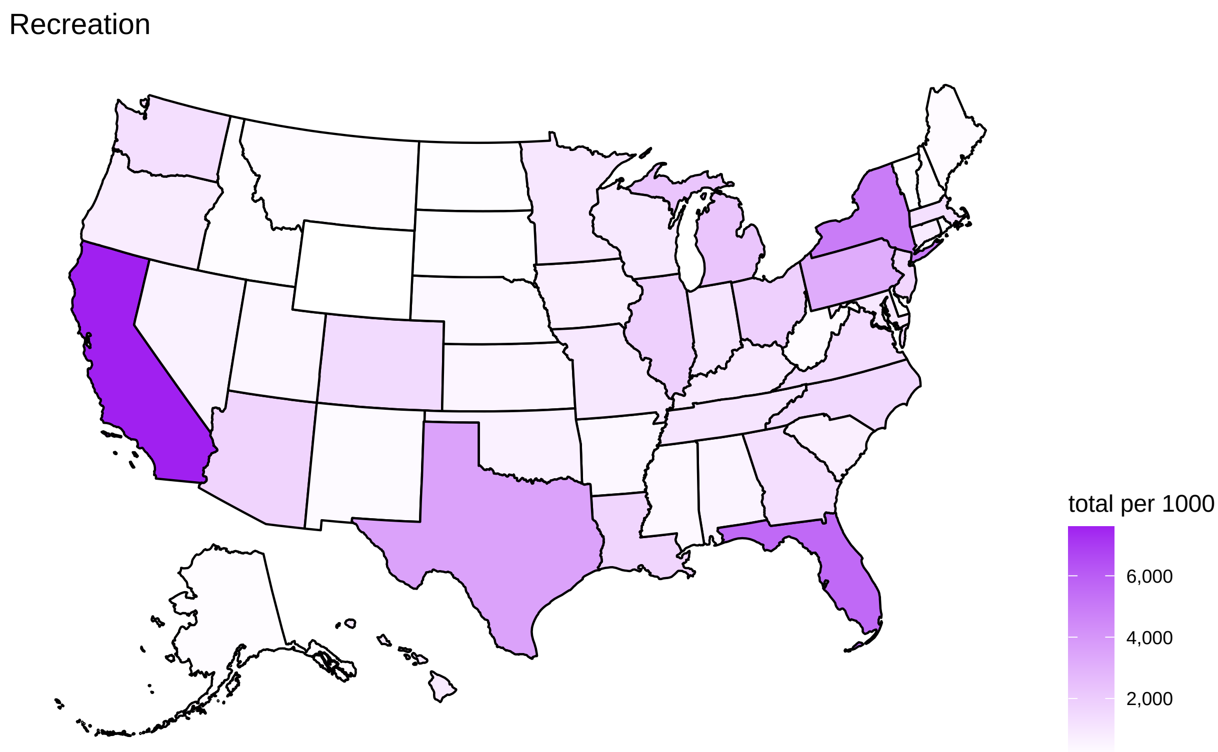
#amusement parks

```r
amuse_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(total_amuse = sum(amusement_parks))


plot_usmap(
   data = amuse_sum_by_state, values = "total_amuse") +
   scale_fill_continuous(
     low = "white", high = "purple", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Amusement parks") +
   theme(legend.position = "right")
```



#recreation

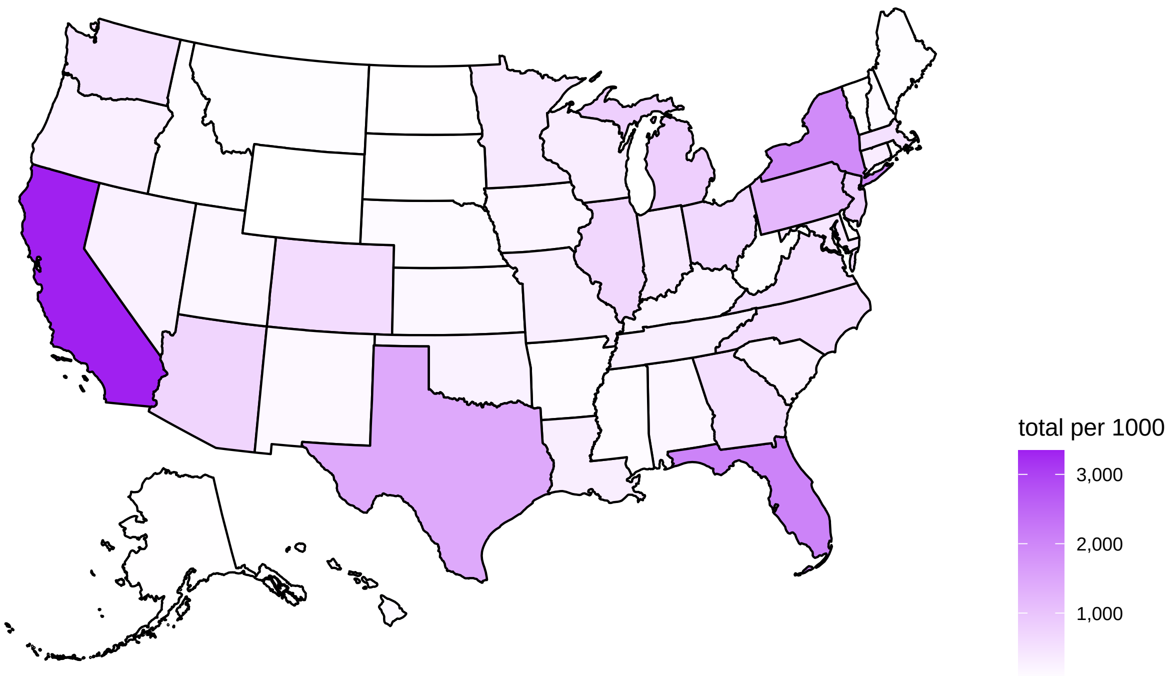
```
rec_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(total_rec = sum(recreation))


plot_usmap(
   data = rec_sum_by_state, values = "total_rec") +
   scale_fill_continuous(
     low = "white", high = "purple", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Recreation") +
   theme(legend.position = "right")
```



#fitness

```
fit_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(total_fit = sum(fitness))


plot_usmap(
   data = fit_sum_by_state, values = "total_fit" ) +
   scale_fill_continuous(
     low = "white", high = "purple", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Fitness centers") +
   theme(legend.position = "right")
```

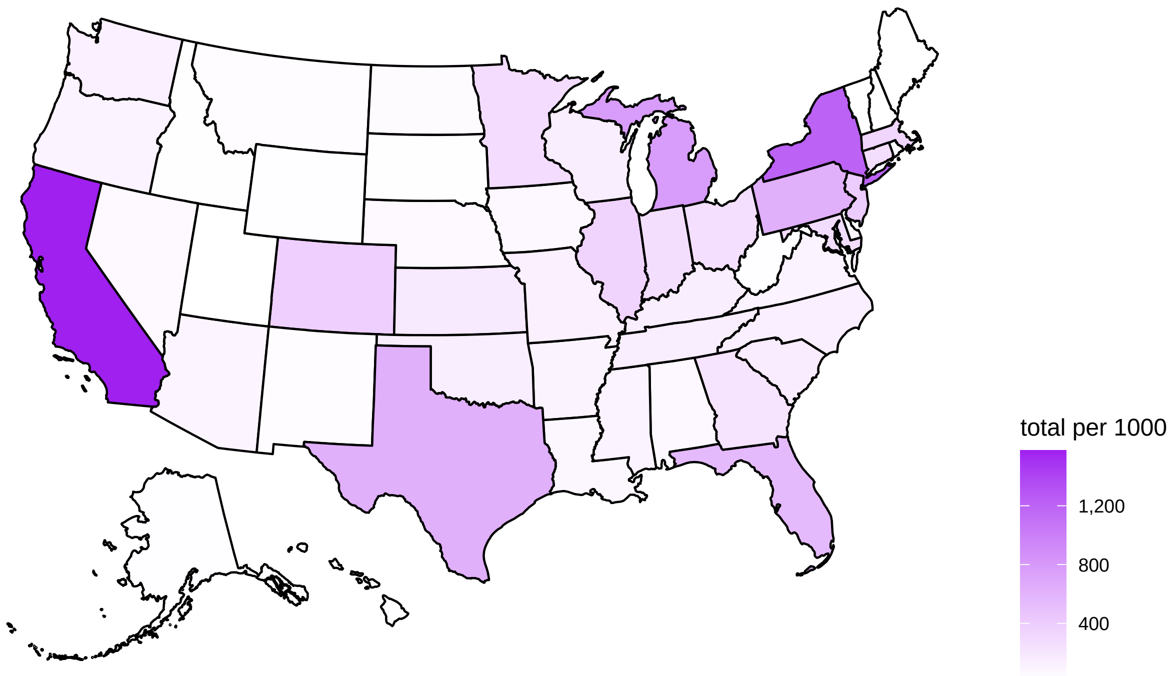
Fitness centers



#liquor

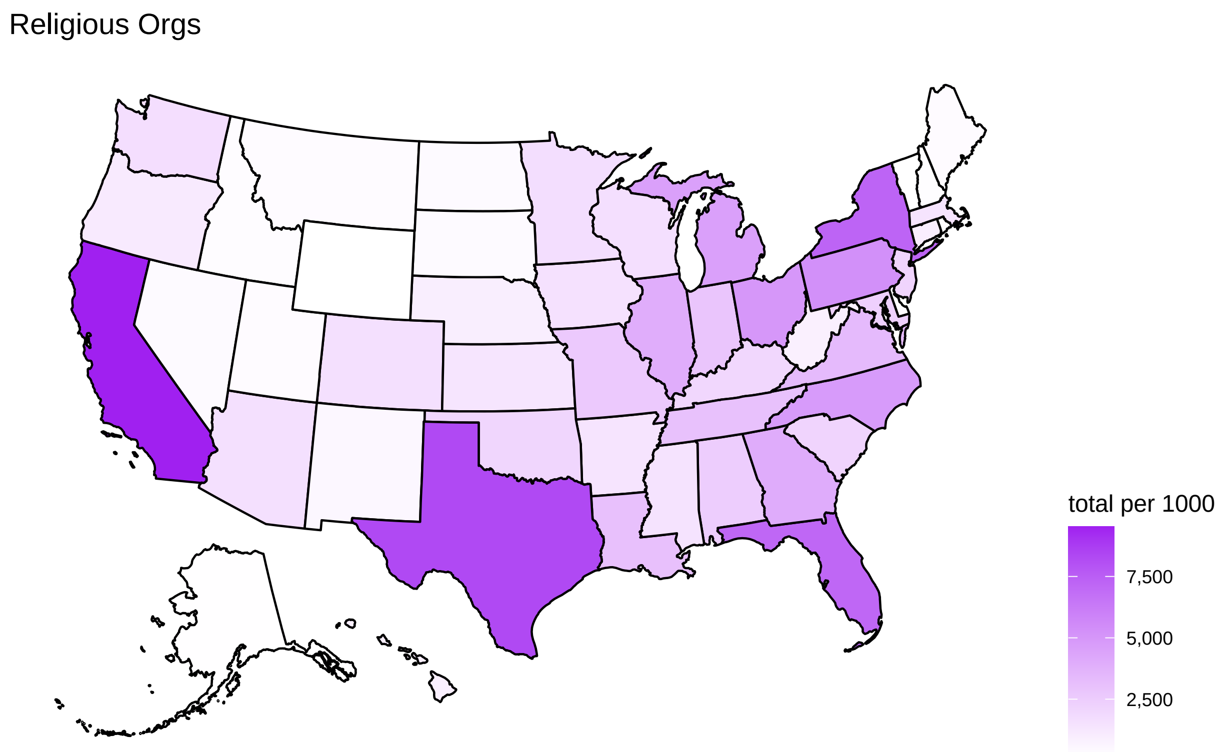
```
liq_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(total_liq = sum(liquor_store))

plot_usmap(
   data = liq_sum_by_state, values = "total_liq") +
   scale_fill_continuous(
     low = "white", high = "purple", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Liquor stores") +
   theme(legend.position = "right")
```

Liquor stores



#religious orgs

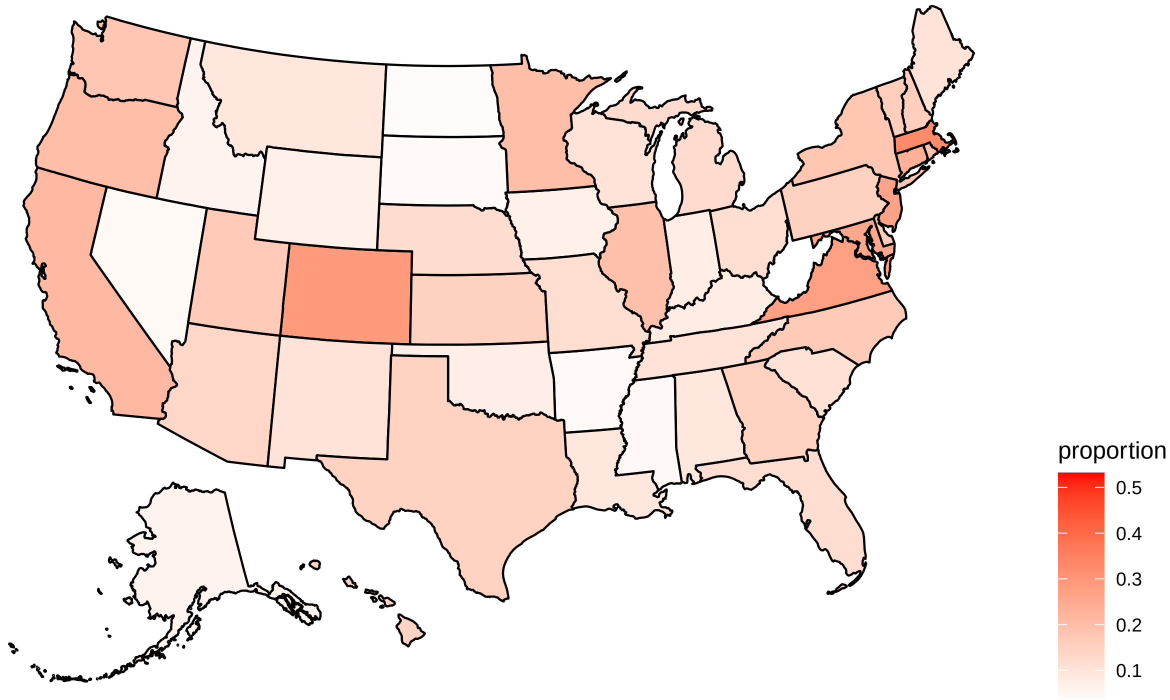
```
relig_sum_by_state <- new_com_data |>
  group_by(state) |>
  summarise(total_relig = sum(religious_orgs))

plot_usmap(
   data = relig_sum_by_state, values = "total_relig") +
   scale_fill_continuous(
     low = "white", high = "purple", name = "total per 1000", label = scales::comma
   ) +
   labs(title = "Religious Orgs") +
   theme(legend.position = "right")
```



#does not account for the different number of people in each state #dont know if data from this website is even accurate #education information is the mean of the years 2013-2017 so thats probably not that accurate either

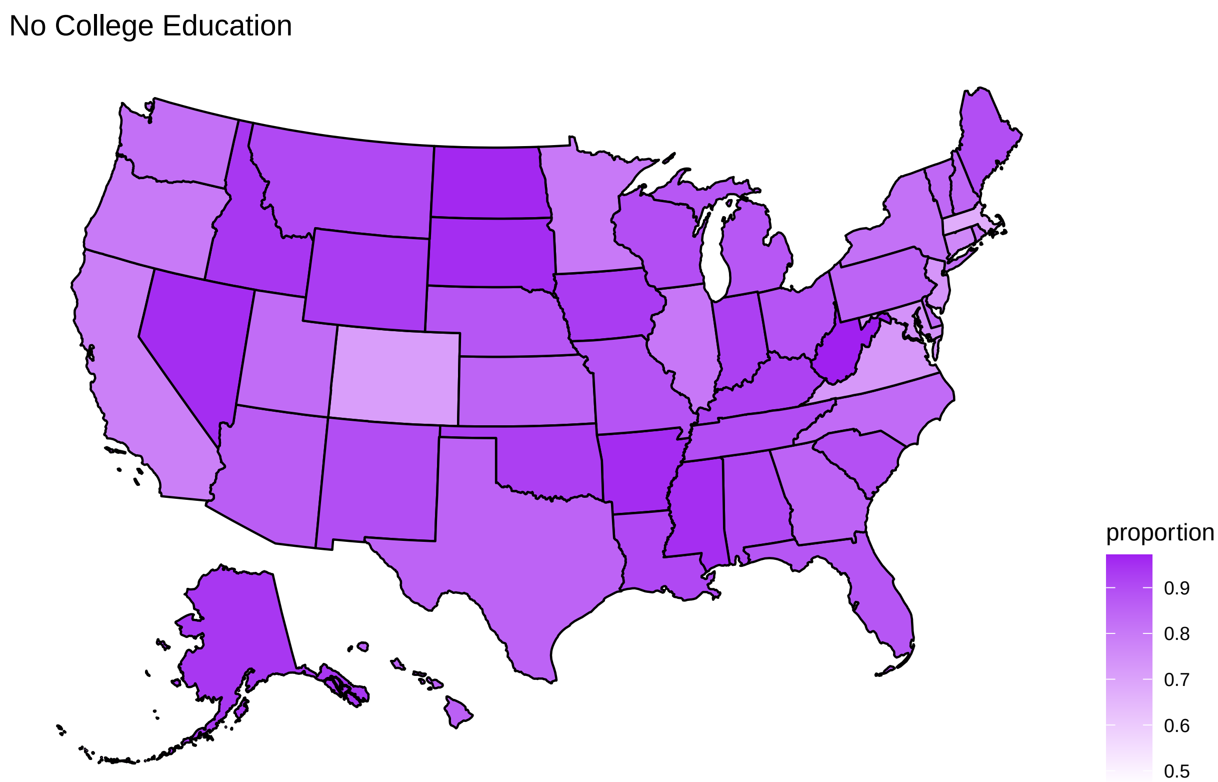
```
#map of proportion of residents that have a college degree across the US
education_state <- state_proportion|>
  group_by(state)|>
  filter(college_edu == "1")

plot_usmap(
   data = education_state, values = "proportion") +
   scale_fill_continuous(
     low = "white", high = "red", name = "proportion", label = scales::comma
   ) +
   labs(title = "College Education") +
   theme(legend.position = "right")
```

College Education




```r
no_education_state <- state_proportion|>
  group_by(state)|>
  filter(college_edu == "0")

plot_usmap(
   data = no_education_state, values = "proportion", lines = "blue"
 ) +
   scale_fill_continuous(
     low = "white", high = "purple", name = "proportion", label = scales::comma
   ) +
   labs(title = "No College Education") +
   theme(legend.position = "right")
```

```
## Warning in (function (mapping = NULL, data = NULL, stat = "identity", position
## = "identity", : Ignoring unknown parameters: `lines`
```

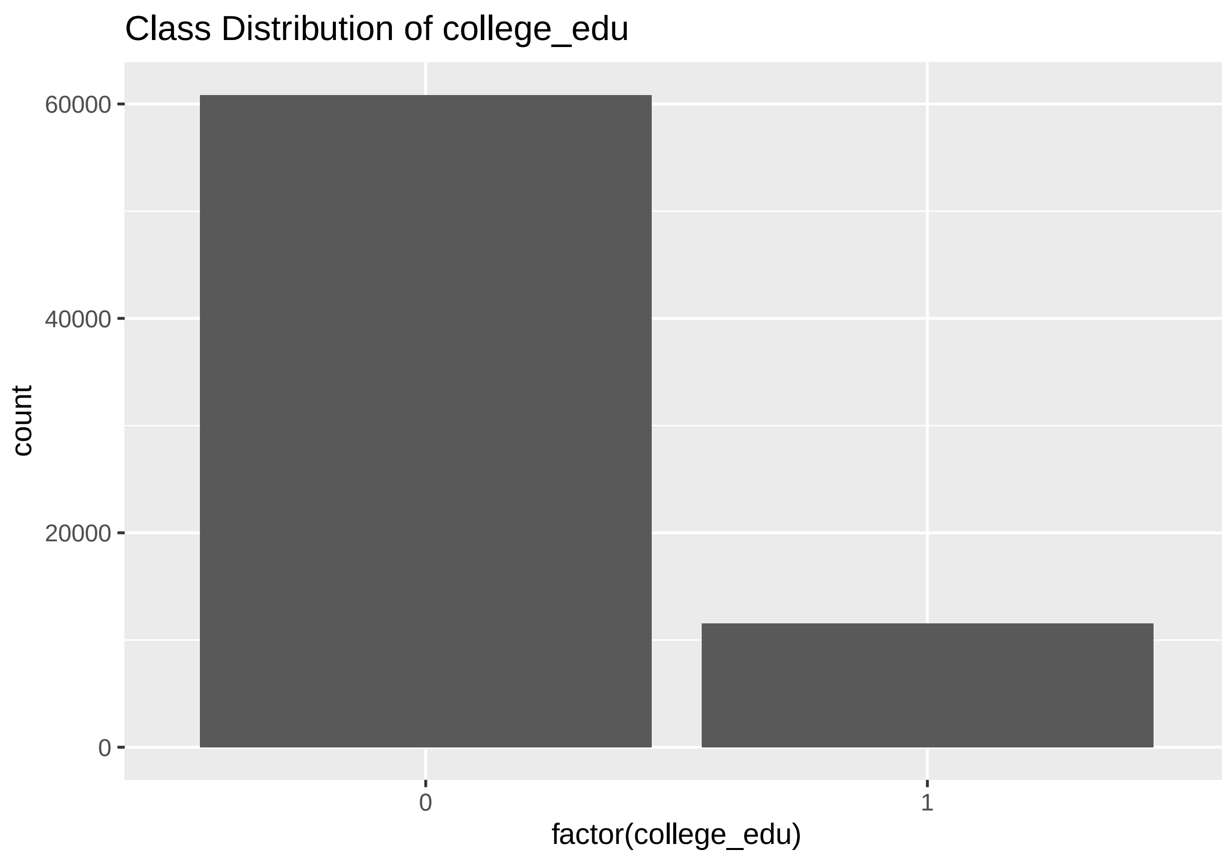


###Model building###

```
#checking class distribution
table(new_com_data$college_edu)
```

```
##
##     0     1
## 60848 11562
```

```
ggplot(new_com_data, aes(x = factor(college_edu))) +
  geom_bar() +
  labs(title = "Class Distribution of college_edu")
```



#try to fix class imbalance by undersampling and oversampling

```
balanced_com_data <- ovun.sample(college_edu ~ performing_arts +
                            libraries +
                            amusement_parks +
                            recreation +
                            fitness +
                            liquor_store +
                            tobacco +
                            religious_orgs +
                            state,
                            data = new_com_data,
                            method = "both",
                            N = 72410,
                            seed = 678)$data

table(balanced_com_data$college_edu)
```

```
##
##     0     1
## 36224 36186
```

#split into test and training

```
set.seed(678)

training_samples <- balanced_com_data$college_edu |>
  createDataPartition(p = 0.8, list = FALSE)
```

```r
train_data  <- balanced_com_data[training_samples, ]
test_data <- balanced_com_data[-training_samples, ]
```

#Null Model

```r
null <- glm(college_edu ~ 1, train_data, family = binomial)

summary(null)
```

```
##
## Call:
## glm(formula = college_edu ~ 1, family = binomial, data = train_data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.178  -1.178   1.177   1.177   1.177
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 0.0009667  0.0083097   0.116    0.907
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 80305  on 57927  degrees of freedom
## Residual deviance: 80305  on 57927  degrees of freedom
## AIC: 80307
##
## Number of Fisher Scoring iterations: 2
```

```r
#misclassification error
predicted_probs_1 <- predict(null, newdata = test_data, type = "response")
predicted_classes_1 <- ifelse(predicted_probs_1 > 0.5, 1, 0)
actual_classes_1 <- test_data$college_edu

misclassification_error_1 <- mean(predicted_classes_1 != actual_classes_1)

print(paste("Misclassification Error:", misclassification_error_1))
```

```
## [1] "Misclassification Error: 0.502278690788565"
```

#got rid of a few predictors because for some reason they were “perfect predictors”?

```r
complete_pooling <- glm(college_edu ~ performing_arts +
                              libraries +
                              amusement_parks +
                              recreation +
                              fitness +
                              liquor_store +
                              tobacco +
                              religious_orgs,
                data = train_data,
                family = binomial(link="logit"))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(complete_pooling, corr = FALSE)
```

```
##
## Call:
## glm(formula = college_edu ~ performing_arts + libraries + amusement_parks +
##     recreation + fitness + liquor_store + tobacco + religious_orgs,
##     family = binomial(link = "logit"), data = train_data)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -8.490  -1.010   0.000   1.059   8.490
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.308506   0.016279 -18.952  < 2e-16 ***
## performing_arts  1.217368   0.032194  37.813  < 2e-16 ***
## libraries        0.240761   0.036064   6.676 2.46e-11 ***
## amusement_parks -0.290154   0.066875  -4.339 1.43e-05 ***
## recreation       0.315361   0.015501  20.344  < 2e-16 ***
## fitness          0.990728   0.031077  31.880  < 2e-16 ***
## liquor_store    -0.021420   0.037824  -0.566    0.571
## tobacco         -0.573864   0.072833  -7.879 3.29e-15 ***
## religious_orgs  -0.460428   0.009614 -47.892  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 80305  on 57927  degrees of freedom
## Residual deviance: 69982  on 57919  degrees of freedom
## AIC: 70000
##
## Number of Fisher Scoring iterations: 8
```

```r
predicted_probs_2 <- predict(complete_pooling, newdata = test_data, type = "response")
predicted_classes_2 <- ifelse(predicted_probs_2 > 0.5, 1, 0)
actual_classes_2 <- test_data$college_edu

# Calculate misclassification error
misclassification_error_2 <- mean(predicted_classes_2 != actual_classes_2)

print(paste("Misclassification Error:", misclassification_error_2))
```

```
## [1] "Misclassification Error: 0.29899185195415"
```

```r
#f1 score
f1_score_2 <- F1_Score(y_pred = predicted_classes_2, y_true = test_data$college_edu)
print(f1_score_2)
```

```
## [1] 0.7165859
```

#no pooling

```r
no_pooling <- glm(college_edu ~ libraries + amusement_parks + liquor_store + tobacco + factor(state),
                          data = train_data,
                          family = binomial(link = "logit"))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(no_pooling, corr = FALSE)
```

```
##
## Call:
## glm(formula = college_edu ~ libraries + amusement_parks + liquor_store +
##     tobacco + factor(state), family = binomial(link = "logit"),
##     data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.2980  -1.1201   0.0136   1.1128   2.1119
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  -0.59050    0.07182  -8.222  < 2e-16 ***
## libraries                     0.21568    0.02858   7.547 4.45e-14 ***
## amusement_parks               0.37781    0.09041   4.179 2.93e-05 ***
## liquor_store                  0.24392    0.03376   7.224 5.04e-13 ***
## tobacco                      -0.06166    0.05118  -1.205 0.228298
## factor(state)Alaska          -0.73512    0.24355  -3.018 0.002541 **
## factor(state)Arizona          0.40374    0.09268   4.356 1.32e-05 ***
## factor(state)Arkansas        -1.02073    0.15620  -6.535 6.38e-11 ***
## factor(state)California       0.89058    0.07571  11.763  < 2e-16 ***
## factor(state)Colorado         1.33667    0.09428  14.178  < 2e-16 ***
## factor(state)Connecticut      0.90225    0.10283   8.775  < 2e-16 ***
## factor(state)DC               2.22974    0.19014  11.727  < 2e-16 ***
## factor(state)Delaware         0.39633    0.17799   2.227 0.025970 *
## factor(state)Florida          0.25015    0.08082   3.095 0.001966 **
## factor(state)Georgia          0.45430    0.08841   5.138 2.77e-07 ***
## factor(state)Hawaii           0.40265    0.14307   2.814 0.004888 **
## factor(state)Idaho           -0.60682    0.19182  -3.164 0.001559 **
## factor(state)Illinois         0.76772    0.08182   9.383  < 2e-16 ***
## factor(state)Indiana         -0.26775    0.10147  -2.639 0.008320 **
## factor(state)Iowa            -0.48994    0.12429  -3.942 8.08e-05 ***
## factor(state)Kansas           0.37624    0.11071   3.398 0.000678 ***
## factor(state)Kentucky        -0.15837    0.10876  -1.456 0.145364
## factor(state)Louisiana       -0.16479    0.10594  -1.555 0.119847
## factor(state)Maine            0.06748    0.15192   0.444 0.656902
## factor(state)Maryland         1.08605    0.09149  11.871  < 2e-16 ***
## factor(state)Massachusetts    1.48238    0.09045  16.389  < 2e-16 ***
## factor(state)Michigan         0.20033    0.08516   2.352 0.018648 *
## factor(state)Minnesota        0.80698    0.09251   8.723  < 2e-16 ***
## factor(state)Mississippi     -0.84958    0.14907  -5.699 1.20e-08 ***
## factor(state)Missouri         0.13555    0.09686   1.399 0.161672
## factor(state)Montana         -0.29382    0.17757  -1.655 0.097978 .
## factor(state)Nebraska         0.17434    0.12564   1.388 0.165242
## factor(state)Nevada          -0.59255    0.13764  -4.305 1.67e-05 ***
## factor(state)New Hampshire    0.19183    0.15070   1.273 0.203048
## factor(state)New Jersey       1.13212    0.08490  13.334  < 2e-16 ***
## factor(state)New Mexico      -0.02026    0.12983  -0.156 0.875977
## factor(state)New York         0.63170    0.07856   8.041 8.93e-16 ***
## factor(state)North Carolina   0.61699    0.08654   7.129 1.01e-12 ***
## factor(state)North Dakota    -1.10787    0.26849  -4.126 3.69e-05 ***
```

```
## factor(state)Ohio            0.15515    0.08468   1.832 0.066942 .
## factor(state)Oklahoma       -0.32951    0.10904  -3.022 0.002512 **
## factor(state)Oregon          0.92091    0.10333   8.912  < 2e-16 ***
## factor(state)Pennsylvania    0.40978    0.08204   4.995 5.89e-07 ***
## factor(state)Rhode Island    0.44054    0.15676   2.810 0.004950 **
## factor(state)South Carolina  0.22060    0.10334   2.135 0.032781 *
## factor(state)South Dakota   -1.51499    0.27566  -5.496 3.89e-08 ***
## factor(state)Tennessee      -0.01959    0.09646  -0.203 0.839055
## factor(state)Texas           0.42097    0.07836   5.372 7.79e-08 ***
## factor(state)Utah            0.62049    0.11612   5.343 9.12e-08 ***
## factor(state)Vermont         0.65370    0.17813   3.670 0.000243 ***
## factor(state)Virginia        1.20962    0.08632  14.014  < 2e-16 ***
## factor(state)Washington      0.63971    0.09219   6.939 3.95e-12 ***
## factor(state)West Virginia  -1.52571    0.20672  -7.380 1.58e-13 ***
## factor(state)Wisconsin      -0.08348    0.09880  -0.845 0.398117
## factor(state)Wyoming        -0.65155    0.28812  -2.261 0.023737 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 80305  on 57927  degrees of freedom
## Residual deviance: 76560  on 57873  degrees of freedom
## AIC: 76670
##
## Number of Fisher Scoring iterations: 6
```

```r
predicted_probs_3 <- predict(no_pooling, newdata = test_data, type = "response")
predicted_classes_3 <- ifelse(predicted_probs_3 > 0.5, 1, 0)
actual_classes_3 <- test_data$college_edu

misclassification_error_3 <- mean(predicted_classes_3 != actual_classes_3)

print(paste("Misclassification Error:", misclassification_error_3))
```

```
## [1] "Misclassification Error: 0.402775859687888"
```

```r
#f1 score
f1_score_3 <- F1_Score(y_pred = predicted_classes_3, y_true = test_data$college_edu)
print(f1_score_3)
```

```
## [1] 0.5896588
```

#partial pooling

```r
#varying intercept
partial_pooling <- glmer(college_edu ~ libraries + amusement_parks + liquor_store + tobacco + (1 | stat
                          data = train_data,
                          family = binomial(link = "logit"))

summary(partial_pooling, corr = FALSE)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: college_edu ~ libraries + amusement_parks + liquor_store + tobacco +
##     (1 | state)
```

```
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##  76844.7  76898.5 -38416.3  76832.7    57922
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1086.93    -0.93     0.01     0.93     2.72
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  state  (Intercept) 0.516    0.7183
## Number of obs: 57928, groups:  state, 51
##
## Fixed effects:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.37979    0.10186  -3.728 0.000193 ***
## libraries        0.21485    0.02853   7.530 5.07e-14 ***
## amusement_parks  0.37644    0.09006   4.180 2.92e-05 ***
## liquor_store     0.24597    0.03372   7.294 3.00e-13 ***
## tobacco         -0.06244    0.05134  -1.216 0.223965
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
predicted_probs_4 <- predict(partial_pooling, newdata = test_data, type = "response")
predicted_classes_4 <- ifelse(predicted_probs_4 > 0.5, 1, 0)
actual_classes_4 <- test_data$college_edu

# Calculate misclassification error
misclassification_error_4 <- mean(predicted_classes_4 != actual_classes_4)

print(paste("Misclassification Error:", misclassification_error_4))
```

```
## [1] "Misclassification Error: 0.40298301339594"
```

```
#f1 score
f1_score_4 <- F1_Score(y_pred = predicted_classes_4, y_true = test_data$college_edu)
print(f1_score_4)
```

```
## [1] 0.5895921
```

#partial pooling

```
#varying intercept varying slope
partial_pooling_2 <- glmer(college_edu ~ libraries + amusement_parks + liquor_store + tobacco +
                          (1 + libraries | state) + (1 + amusement_parks | state) + (1 + liquor_store |
                          (1 + tobacco | state),
                               data = train_data,
                               family = binomial(link = "logit"))


summary(partial_pooling_2, corr = FALSE)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: college_edu ~ libraries + amusement_parks + liquor_store + tobacco +
```

```
##     (1 + libraries | state) + (1 + amusement_parks | state) +
##     (1 + liquor_store | state) + (1 + tobacco | state)
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##  76167.7  76320.1 -38066.8  76133.7    57911
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -17.7054  -0.9102   0.0000   0.9268   6.9894
##
## Random effects:
##  Groups  Name            Variance Std.Dev. Corr
##  state   (Intercept)     0.14306  0.3782
##          libraries       1.57122  1.2535   0.72
##  state.1 (Intercept)     0.15241  0.3904
##          amusement_parks 2.45646  1.5673   -0.42
##  state.2 (Intercept)     0.11100  0.3332
##          liquor_store    0.54371  0.7374   -0.56
##  state.3 (Intercept)     0.06852  0.2618
##          tobacco         1.25486  1.1202   0.69
## Number of obs: 57928, groups:  state, 51
##
## Fixed effects:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.33141    0.09759  -3.396 0.000684 ***
## libraries       -0.03599    0.18471  -0.195 0.845532
## amusement_parks  0.62046    0.26717   2.322 0.020215 *
## liquor_store     0.25405    0.12297   2.066 0.038828 *
## tobacco         -0.45228    0.19664  -2.300 0.021445 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
predicted_probs_5 <- predict(partial_pooling_2, newdata = test_data, type = "response")
predicted_classes_5 <- ifelse(predicted_probs_5 > 0.5, 1, 0)
actual_classes_5 <- test_data$college_edu

# Calculate misclassification error
misclassification_error_5 <- mean(predicted_classes_5 != actual_classes_5)

print(paste("Misclassification Error:", misclassification_error_5))
```

```
## [1] "Misclassification Error: 0.415412235879022"
```

```r
#f1 score
#f1 score
f1_score_5 <- F1_Score(y_pred = predicted_classes_5, y_true = test_data$college_edu)
print(f1_score_5)
```

```
## [1] 0.5545683
```

#partial pooling varying slope

```r
partial_pooling_3 <- glmer(college_edu ~ libraries + amusement_parks + liquor_store + tobacco +
                           (libraries | state) + (amusement_parks | state) + (liquor_store | state) +
                           (tobacco | state),
```

```r
                              data = train_data,
                              family = binomial(link = "logit"))

summary(partial_pooling_3, corr = FALSE)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: college_edu ~ libraries + amusement_parks + liquor_store + tobacco +
##     (libraries | state) + (amusement_parks | state) + (liquor_store |
##     state) + (tobacco | state)
##    Data: train_data
##
##      AIC      BIC   logLik deviance df.resid
##  76167.7  76320.1 -38066.8  76133.7    57911
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -17.7054  -0.9102   0.0000   0.9268   6.9894
##
## Random effects:
##  Groups  Name            Variance Std.Dev. Corr
##  state   (Intercept)     0.14306  0.3782
##          libraries       1.57122  1.2535   0.72
##  state.1 (Intercept)     0.15241  0.3904
##          amusement_parks 2.45646  1.5673   -0.42
##  state.2 (Intercept)     0.11100  0.3332
##          liquor_store    0.54371  0.7374   -0.56
##  state.3 (Intercept)     0.06852  0.2618
##          tobacco         1.25486  1.1202   0.69
## Number of obs: 57928, groups:  state, 51
##
## Fixed effects:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.33141    0.09759  -3.396 0.000684 ***
## libraries       -0.03599    0.18471  -0.195 0.845532
## amusement_parks  0.62046    0.26717   2.322 0.020215 *
## liquor_store     0.25405    0.12297   2.066 0.038828 *
## tobacco         -0.45228    0.19664  -2.300 0.021445 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
predicted_probs_6 <- predict(partial_pooling_3, newdata = test_data, type = "response")
predicted_classes_6 <- ifelse(predicted_probs_6 > 0.5, 1, 0)
actual_classes_6 <- test_data$college_edu

# Calculate misclassification error
misclassification_error_6 <- mean(predicted_classes_6 != actual_classes_6)

print(paste("Misclassification Error:", misclassification_error_6))
```

```
## [1] "Misclassification Error: 0.415412235879022"
```

```r
#f1 score
f1_score_6 <- F1_Score(y_pred = predicted_classes_6, y_true = test_data$college_edu)
```

```r
print(f1_score_6)
```

```
## [1] 0.5545683
```

#table for AIC values. Model comparison

```r
aic_values <- c(AIC(null), AIC(complete_pooling), AIC(no_pooling), AIC(partial_pooling), AIC(partial_po

model_names <- c("Null model", "Complete pooling model", "No pooling model", "Partial pooling model var

table_data <- data.frame(Model = model_names, AIC = aic_values)

aic_ktable <- kable(table_data, caption = "AIC values of Models", align = c("l", "c"))

print(aic_ktable)
```

```
##
##
## Table: AIC values of Models
##
## |Model                               |   AIC    |
## |:-----------------------------------|:--------:|
## |Null model                          | 80307.25 |
## |Complete pooling model              | 69999.87 |
## |No pooling model                    | 76669.83 |
## |Partial pooling model vary intercept | 76844.67 |
## |Partial pooling model vary both      | 76167.70 |
## |Partial pooling model vary slope     | 76167.70 |
```

#table for Misclassification error. Model validation.

```r
misclassification_errors <- c(0.50, 0.30, 0.40, 0.40, 0.42, 0.41)

model_names <- c("Null model", "Complete pooling model", "No pooling model", "Partial pooling model var

misclassification_table <- data.frame(Model = model_names, 'Misclassification Error' = misclassificatio

missclass_ktable <- kable(misclassification_table, caption = "Misclassification Errors of Models", alig

print(missclass_ktable)
```

```
##
##
## Table: Misclassification Errors of Models
##
## |Model                               | Misclassification.Error |
## |:-----------------------------------|:-----------------------:|
## |Null model                          |          0.50           |
## |Complete pooling model              |          0.30           |
## |No pooling model                    |          0.40           |
## |Partial pooling model vary intercept |          0.40           |
## |Partial pooling model vary both      |          0.42           |
## |Partial pooling model vary slope     |          0.41           |
```

#table for F1 score.

```r
model_f1_scores <- c(0.72, 0.59, 0.59, 0.55, 0.55)

model_names <- c("Complete pooling model", "No pooling model", "Partial pooling model vary intercept",

f1_score_table <- data.frame(Model = model_names, 'F1 Test Scores' = model_f1_scores)

f1_ktable <- kable(f1_score_table, caption = "F1 test scores of Models", align = c("l", "c"))

print(f1_ktable)
```

```
##
##
## Table: F1 test scores of Models
##
## |Model                                | F1.Test.Scores |
## |:------------------------------------|:--------------:|
## |Complete pooling model               |      0.72      |
## |No pooling model                     |      0.59      |
## |Partial pooling model vary intercept |      0.59      |
## |Partial pooling model vary both      |      0.55      |
## |Partial pooling model vary slope     |      0.55      |
```