# STA 141A Project: Food Insecurity

Brooke Kerstein, Gabriel Jones

2023-09-13

## Introduction

Food insecurity, a pervasive threat to a wide variety of demographics in America, is defined by a lack of accessibility to consistently supplied, adequately nourishing food necessary to live an active, healthy lifestyle. Though much effort has been made to address this basic need issue, Food insecurity has remained a persistent threat across the decades. Feeding America, a non-profit organization dedicated to addressing this issue, reports that at least 34 million adults and 9 million children experience food insecurity every year, and these numbers have only exponentiation as COVID-19 grew to a peak in 2020 and early 2021.

In this report, we wish to address the characteristics of food insecurity and its many factors, plotting data retrieved from dedicated census cites such as The United States Census Bureau, CA.gov, and Feeding America's own data caches to investigate the trends of food insecurity over time as well as factors of interest that might prove vital to understanding what exactly contributes to food insecurity. With this knowledge, we plan to construct optimal predictive models using regression techniques and machine learning to ultimately test our own findings regarding the predictors of food insecurity.

## Data Wrangling

### Libraries

### Clean Up Feed America 2019-2020

Our food insecurity data was given to us as a folder containing 11 .xlsx files. Each file contained food insecurity data for every county in the US. Upon exploring the format of each file, we noticed that there were inconsistencies with the columns between data sets, thus we would need to clean each data set individually before merging them all together. As a quick note, our initial data wrangling process was done to include all columns from the data set as we did not know which variables we wanted to use. The code provided in the appendix is cleaned to only include columns used for the rest of the analysis.

We initiated our data wrangling process by loading each file into it's own data frame. We then cleaned up the 2019-2021 data so that it was in a desirable format, this would act as our baseline format for the other data sets to match. While the data for each year had slight differences from each other, the process of cleaning them followed the same general process. First we began by selecting only California data and removing any undesired columns. We then added a "year" column to store the year that this data was obtained for. Then we manually cleaned some of the column names to remove the "20XX" and put them into tidy format using the janitor package. Lastly, we redefined the order in which columns were organized in the data frame and merged it into a single data frame called "cleandata". Below is an example of how we defined our baseline 2019-2021 data and how we cleaned yearly data for 2010-2018:

```r
# Clean up the baseline data frame that we will be adding yearly data to
cleandata2019 <- rawdata2019_2021 %>%
  select(
    -"Food Insecurity Rate among Black Persons (all ethnicities)",
    -"Food Insecurity Rate among Hispanic Persons (any race)",
    -"Food Insecurity Rate among White, non-Hispanic Persons",
    -"Weighted weekly $ needed by FI"
  ) %>%
  filter(State == "CA")
cleandata2019 <- janitor::clean_names(cleandata2019)

# Clean up other yearly data and put it in the same format as our baseline data frame
cleandata2010 <- rawdata2010 %>%
  # Select Desired Data
  filter(State == "CA") %>%
  select(-"...19") %>%
  mutate("Year" = rep(2010, length(.$State))) %>%
  # Clean Names
  rename(., "Overall Food Insecurity Rate" = "2010 Food Insecurity Rate") %>%
  rename(., "# of Food Insecure Persons Overall" = "Number of Food Insecure Persons in 2010") %>%
  rename(., "Child Food Insecurity Rate" = "2010 Child food insecurity rate") %>%
  rename(., "# of Food Insecure Children" = "Number of Food Insecure Children in 2010") %>%
  rename(., "Cost Per Meal" = "2010 Cost Per Meal") %>%
  rename(., "Weighted Annual Food Budget Shortfall" = "2010 Weighted Annual Food Budget Shortfall") %>%
  rename(., `% food insecure children in HH w/ HH incomes above 185 FPL` = `% of food insecure children
  mutate("% FI > Low Threshold" = rep("NA", length(.$Year))) %>%
  as.data.frame(.)

# Make columns the same type
cleandata2010$`# of Food Insecure Children` <-
  as.character(cleandata2010$`# of Food Insecure Children`)
cleandata2010 <- janitor::clean_names(cleandata2010) %>%
  select(
    fips,
    state,
    county_state,
    year,
    overall_food_insecurity_rate, number_of_food_insecure_persons_overall,
    low_threshold_in_state, low_threshold_type, high_threshold_in_state,
    high_threshold_type, percent_fi_low_threshold, percent_fi_btwn_thresholds,
    percent_fi_high_threshold, percent_fi_low_threshold_2,
    child_food_insecurity_rate, number_of_food_insecure_children,
    percent_food_insecure_children_in_hh_w_hh_incomes_below_185_fpl,
    percent_food_insecure_children_in_hh_w_hh_incomes_above_185_fpl,
    cost_per_meal, weighted_annual_food_budget_shortfall
  )
# Merge into one data frame
cleandata <- rbind(cleandata2019, cleandata2010)
```

### Cleaning Age Data for 2019 Correlation

### Clean Up Disability 2013-2021

### Clean Up Disability 2010-2012

### Total Disability 2010-2021

### Clean Up Unemployment & Income Data

The financial data we used was sourced from from data.ca.gov and consisted of two different data sets containing information on unemployment and median income in California. Both of these data sets were relatively clean, thus the wrangling process was fairly simple. For the unemployment data, we filtered the data to select only California counties and the years 2010-2021. We then calculated the means for each county during each year because we were given multiple data points for each county-year combination.

The median income data was cleaned in a similar fashion. Cleaning processes unique to this data set included removing unnecessary counties from the "county" column, and merging the income data with our main data frame from 2010-2020 rather than 2010-2021. This was because our income data did not contain any information from 2021.

### Final CSV

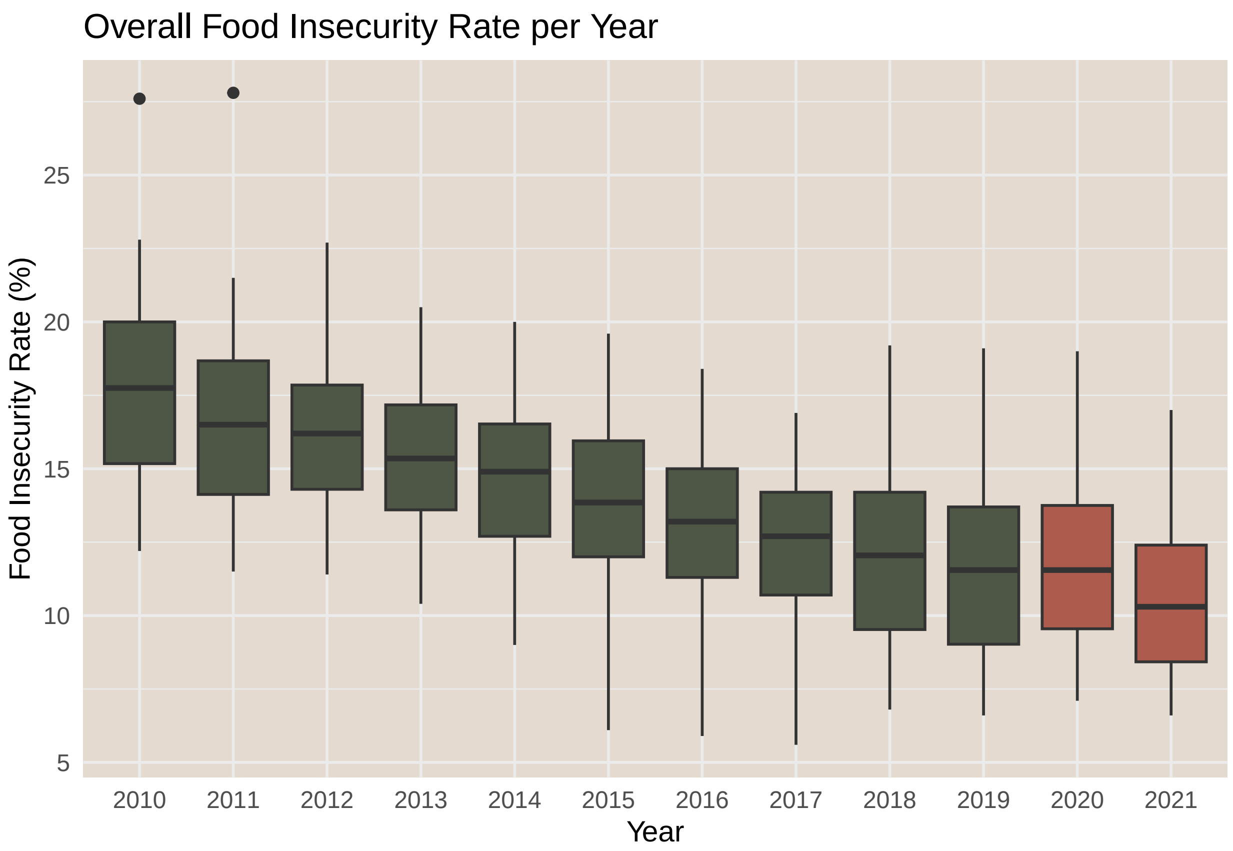
# Analysis

## Data Visualization - Yearly Trend Graphs

The second part of our preliminary findings from data exploration consisted of visualizing how overall food insecurity rate and our determining factors changed over the years. For these visualizations, we used yearly boxplots to show the inter-quantile range of each variable during each year. Through these plots, we're able to gain some extra insights as to what we would expect from the the relationships between our determining factors and food insecurity rate. This ultimately allows us to understand how to use statistical models to forecast food insecurity rate.

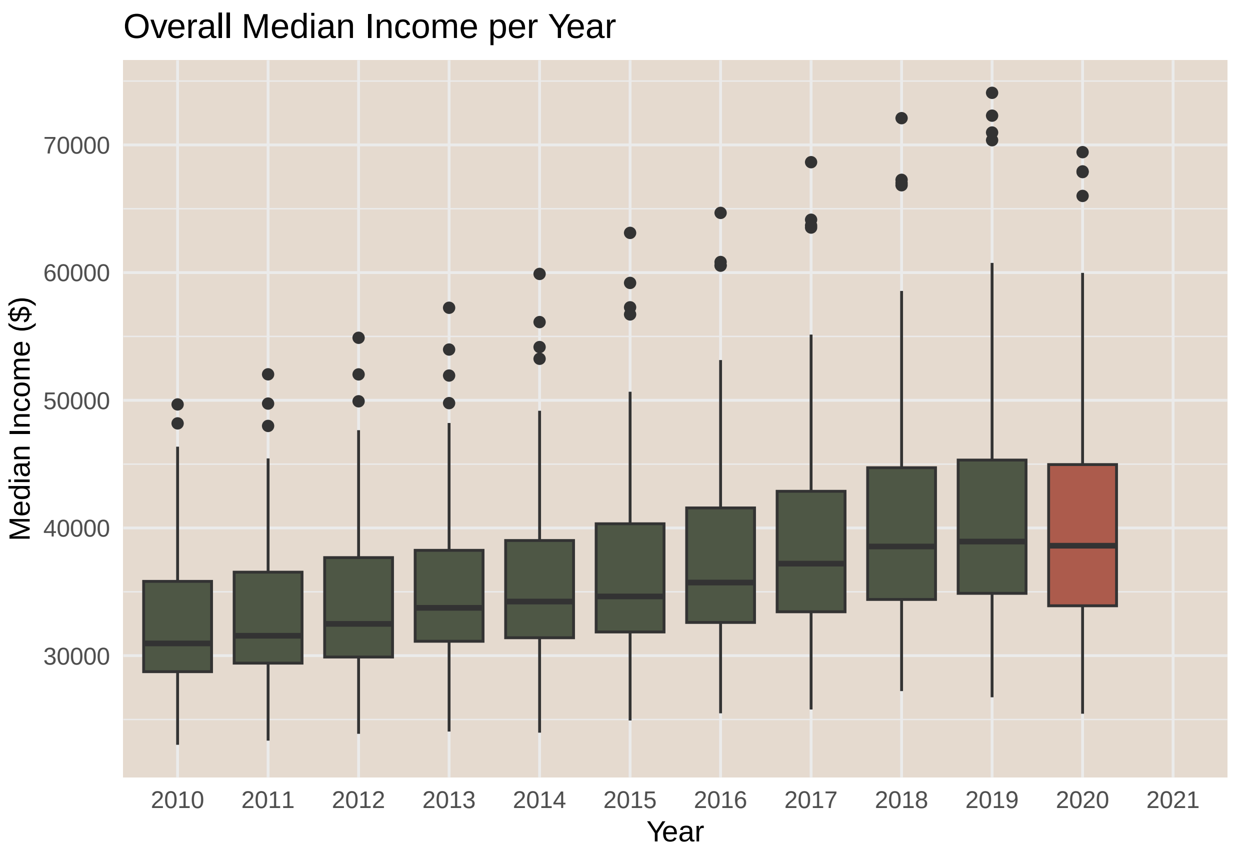
### Food Insecurity Boxplot

Starting with the yearly food insecurity rate, we notice that there is a general downturn in food insecurity each year from 2010-2019. We also notice that this downturn is disrupted in 2020 where the food insecurity rate stays relatively the same as the previous year. The downward trend then is seen to be continued as expected in 2021.



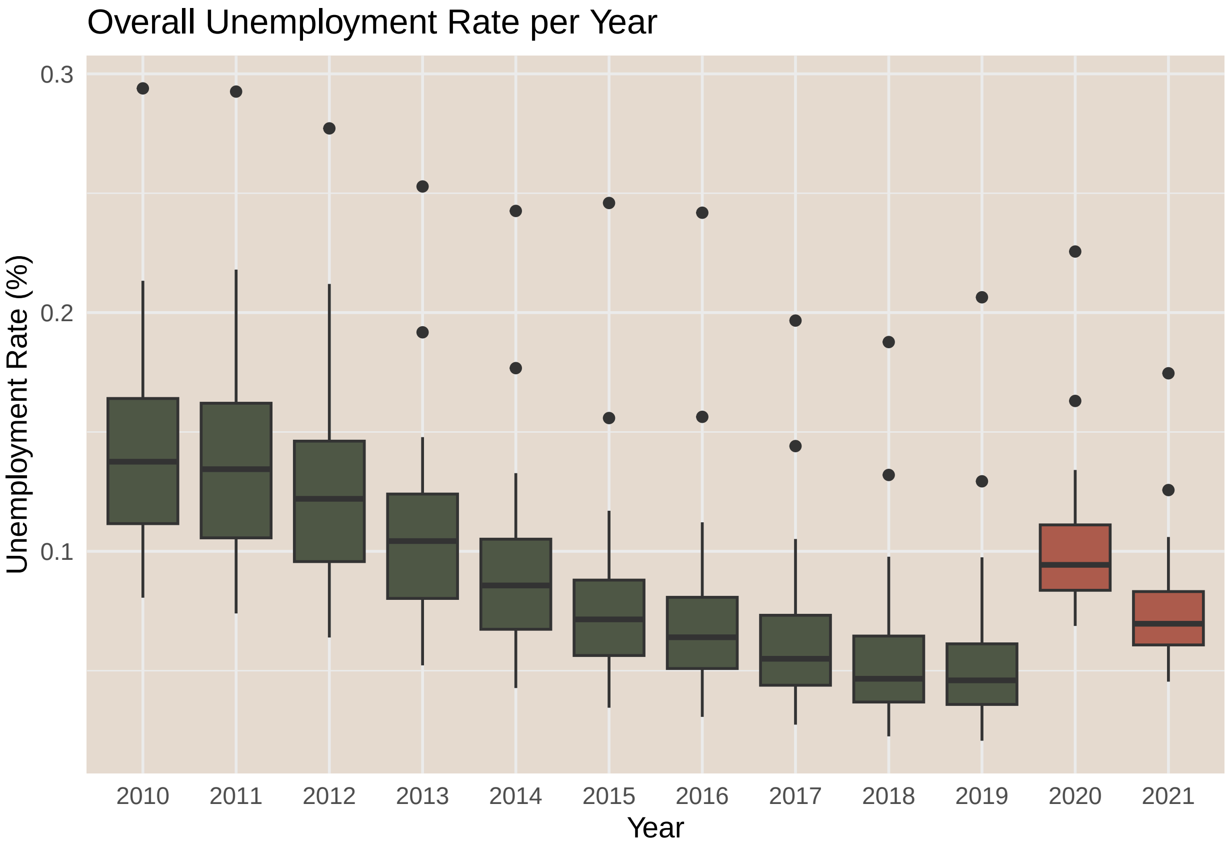
**Income Boxplot**

When performing the same visualizations for median income, we observe a similar phenomenon as the food insecurity rate. Here, we notice an upward trend in median income from 2010-2019 followed by a disruption in 2020 where the IQR remains similar to the previous year. While this provides evidence that median income would be useful in forecasting food insecurity, our predictions based on this may be disrupted by the outliers shown in the graph. These outliers are likely the median income for counties with higher income job opportunities.



**Unemployment Boxplot**

The box plot for unemployment rate in California shows a trend similar to the previous disability rate graph. Here we observe a large downturn in the unemployment rate from 2010-2019, however, we see a large increase in unemployment rate in 2020. This is likely caused by the lock down issued in 2020 where many people were no longer able to attend their jobs. This inconsistency may make predicting food insecurity in 2020 less accurate.

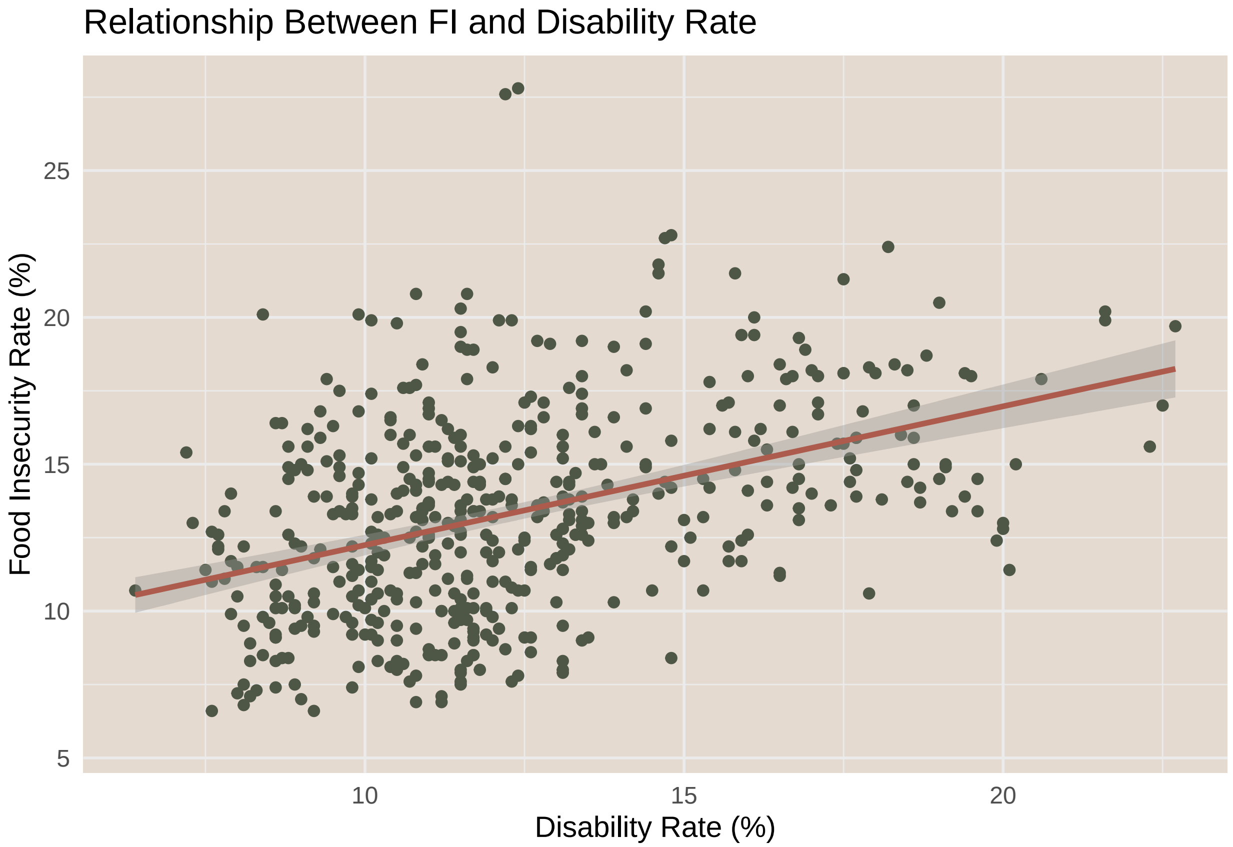


## Modeling

We began our modeling process by fitting linear regression models to all three of our predictors. The general purpose behind these visualizations was for us to get a better idea of the relationship between our predictors and food insecurity rate as we will not be able to visualize them all together in a multivariate linear model.

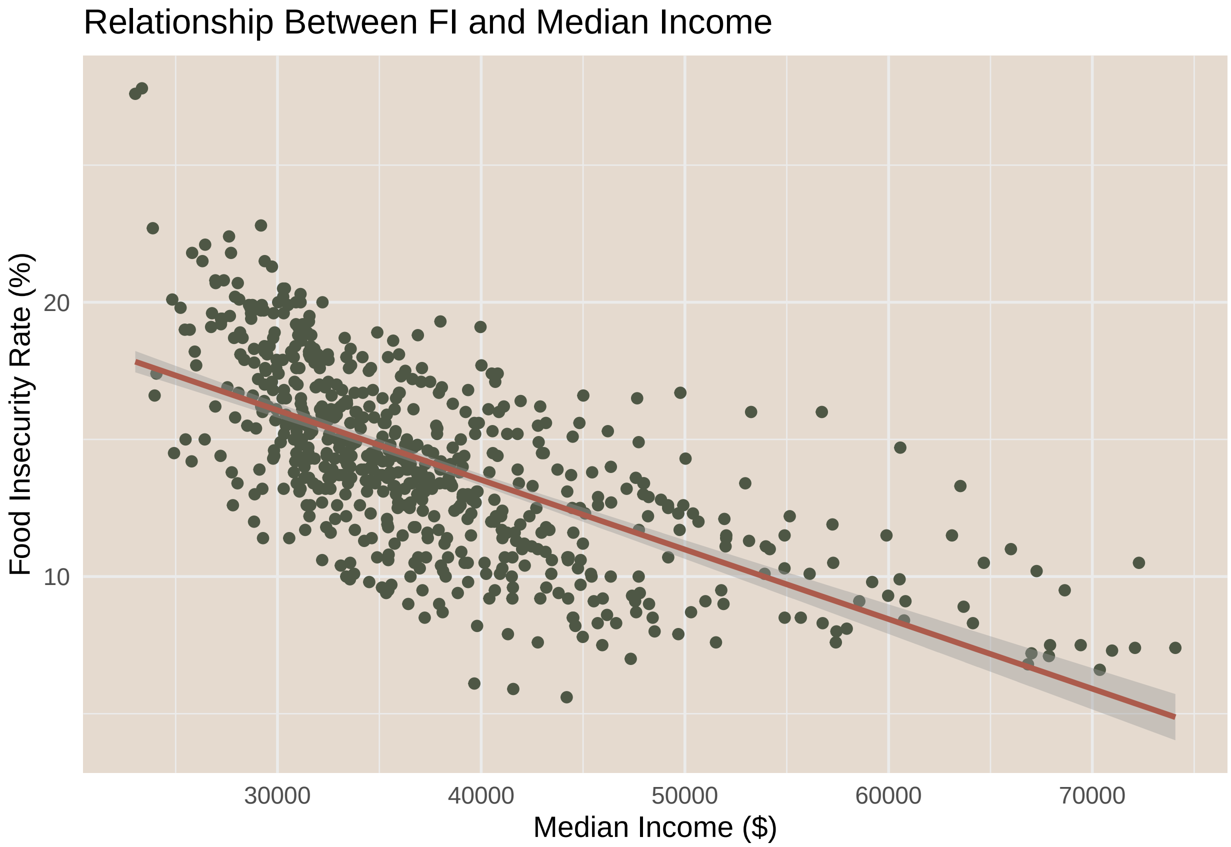
### Food Insecurity vs. Disability Linear Model

When fitting disability rate to food insecurity, we observe that the two variables have a slight positive relationship. This relationship may have innacuracies in predicting food insecurity for different counties due to the variability across each county. Fitting a model to each county individually may increase the accuracy of a multivariate model including disability rate as a predictor.



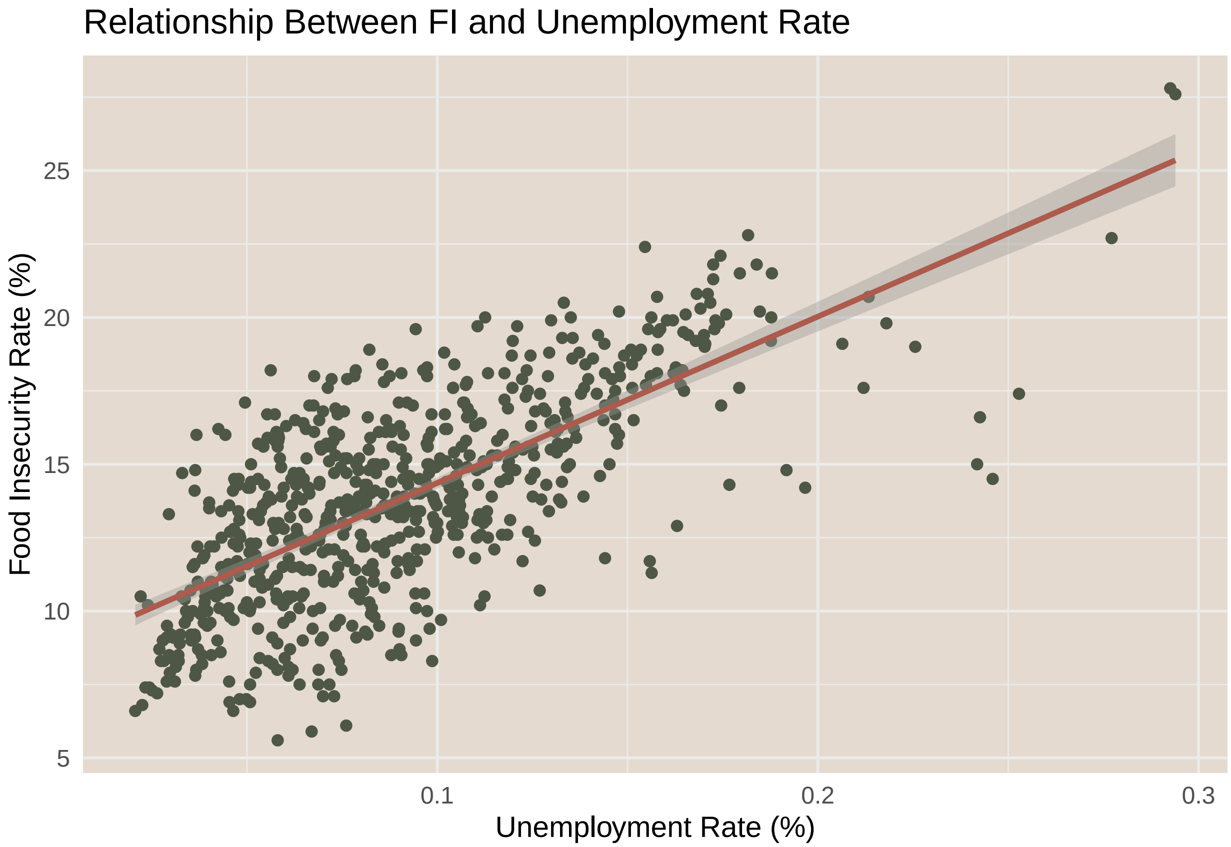
**Food Insecurity vs. Median Income Linear Model**

When fitting median income to food insecurity, we observe that the two variables have a negative relationship. This relationship is better defined in comparison to the disability rate model, however, the large number of data points in the $30000-$40000 range may make it difficult to accurately predict the food insecurity rate for counties with a median income within that range.



**Food Insecurity vs. Unemployment Linear Model**

When fitting unemployment rate to food insecurity, we observe that the two variables have a positive relationship. This model has the same concern as the median income model as counties lying in the 0-10% unemployment rate may be innaccurately estemated due to their variability. Due to the concerns of all three of these linear models, we decided to fit our predictive models for both all counties and for each county individually to compare the accuracy of each.



### Predictive Models

We decided to use two approaches to predicting food insecurity based on our predictive variables to compare which model performed best. The first model we used was an additive linear model which fit unemployment rate, disability rate, and median income together to food insecurity. We chose this model because we observed linear relationships between all three of our predictors and food insecurity. This model was also relatively simple to implement. The second model we used was a random forest model using the same formula as the additive linear model. This model was chosen as a means of reducing the variance in our data, which should increase the accuracy of our predictions.

When selecting our test data, we decided to test against both 2019 and 2020 data. This was done in consideration of the trend disruptions between values in 2019 and 2020 for all of our predictors. The expectation from this was that the predictive models would have better accuracy in predicting food insecurity for 2019 than for 2020. Thus, for our training data we used data between 2010-2018 and 2010-2019, with each being tested against 2019 and 2020 respectively.

Lastly, to further address the concerns brought up in the discussion of our single factor linear models, we wrote our predictive models using both data points regardless of county and models fit for each county individually. The expectation from this was that the models fit for each county would be significantly more accurate due to accounting for each county's individual trends. Our only concern from this was not having sufficient data points to fit these models towards.

### Predictive Model Results

After fitting our models and calculating their accuracy, we found that the general predictions for food insecurity in 2019 was more accurate than our predictions for 2020. This again, was likely due to data in

2019 following the same general trend as previous years, while data in 2020 having a disruption to that trend. When comparing within test groups, we find that the overall random forest model and by county linear model performed the best when testing against the 2019 data, with the random forest performing slightly better. When testing against the 2020 data, we found that fitting the random forest model by county performed significantly better than the other models tested against 2020.

| Model.Type | Test.Accuracy.2019 | Test.Accuracy.2020 |
|---|---:|---:|
| Overall Linear Regression | 0.8671502 | 0.6799830 |
| Overall Random Forest | 0.9324213 | 0.6688549 |
| By County Linear Model | 0.9758715 | 0.9738838 |
| By County Random Forest | 0.9946954 | 0.9951765 |

# Conclusion

## R Appendix

```r
knitr::opts_chunk$set(eval = TRUE)
# Check for installed packages before loading
list.of.packages <- c("dplyr", "tidyr", "ggplot2", "hrbrthemes", "gganimate",
                      "png", "gifski", "ggridges", "tidyverse", "tibble",
                      "mapview", "sp", "janitor", "GGally", "RColorBrewer",
                      "MASS", "knitr","matlib", "lubridate", "pdftools",
                      "stringr", "ggmap", "ggsci", "patchwork", "ddpcr",
                      "caret", "car", "paletteer")

new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, library, character.only=TRUE)
rawdata2010 <- readr::read_csv("FeedAmerica/FeedAmerica2010.csv")
rawdata2011 <- readr::read_csv("FeedAmerica/FeedAmerica2011.csv")
rawdata2012 <- readr::read_csv("FeedAmerica/FeedAmerica2012.csv")
rawdata2013 <- readr::read_csv("FeedAmerica/FeedAmerica2013.csv")
rawdata2014 <- readr::read_csv("FeedAmerica/FeedAmerica2014.csv")
rawdata2015 <- readr::read_csv("FeedAmerica/FeedAmerica2015.csv")
rawdata2016 <- readr::read_csv("FeedAmerica/FeedAmerica2016.csv")
rawdata2017 <- readr::read_csv("FeedAmerica/FeedAmerica2017.csv")
rawdata2018 <- readr::read_csv("FeedAmerica/FeedAmerica2018.csv")
rawdata2019_2021 <- readr::read_csv("FeedAmerica/FeedAmerica2019_2021.csv")

AgeData2019 <- readr::read_csv("DisabilityData/DisabilityData2019.csv")

unemploymentdataraw <- readr::read_csv("Local_Area_Unemployment_Statistics__LAUS_.csv")
unemploymentdataraw <- janitor::clean_names(unemploymentdataraw)

incomedataraw <- readr::read_csv("incomedata.csv")
incomedataraw <- janitor::clean_names(incomedataraw)

DisData2021 <- readr::read_csv("DisabilityData/DisabilityData2021.csv")
DisData2020 <- readr::read_csv("DisabilityData/DisabilityData2020.csv")
DisData2019 <- readr::read_csv("DisabilityData/DisabilityData2019.csv")
```

```r
DisData2018 <- readr::read_csv("DisabilityData/DisabilityData2018.csv")
DisData2017 <- readr::read_csv("DisabilityData/DisabilityData2017.csv")
DisData2016 <- readr::read_csv("DisabilityData/DisabilityData2016.csv")
DisData2015 <- readr::read_csv("DisabilityData/DisabilityData2015.csv")
DisData2014 <- readr::read_csv("DisabilityData/DisabilityData2014.csv")
DisData2013 <- readr::read_csv("DisabilityData/DisabilityData2013.csv")
DisData2012 <- readr::read_csv("DisabilityData/DisabilityData2012.csv")
DisData2011 <- readr::read_csv("DisabilityData/DisabilityData2011.csv")
DisData2010 <- readr::read_csv("DisabilityData/DisabilityData2010.csv")
# Clean up the baseline data frame that we will be adding yearly data to
cleandata2019 <- rawdata2019_2021 %>%
  select(
    -"Food Insecurity Rate among Black Persons (all ethnicities)",
    -"Food Insecurity Rate among Hispanic Persons (any race)",
    -"Food Insecurity Rate among White, non-Hispanic Persons",
    -"Weighted weekly $ needed by FI"
  ) %>%
  filter(State == "CA")
cleandata2019 <- janitor::clean_names(cleandata2019)

# Clean up other yearly data and put it in the same format as our baseline data frame
cleandata2010 <- rawdata2010 %>%
  # Select Desired Data
  filter(State == "CA") %>%
  select(-"...19") %>%
  mutate("Year" = rep(2010, length(.$State))) %>%
  # Clean Names
  rename(., "Overall Food Insecurity Rate" = "2010 Food Insecurity Rate") %>%
  rename(., "# of Food Insecure Persons Overall" = "Number of Food Insecure Persons in 2010") %>%
  rename(., "Child Food Insecurity Rate" = "2010 Child food insecurity rate") %>%
  rename(., "# of Food Insecure Children" = "Number of Food Insecure Children in 2010") %>%
  rename(., "Cost Per Meal" = "2010 Cost Per Meal") %>%
  rename(., "Weighted Annual Food Budget Shortfall" = "2010 Weighted Annual Food Budget Shortfall") %>%
  rename(., `% food insecure children in HH w/ HH incomes above 185 FPL` = `% of food insecure children
  mutate("% FI > Low Threshold" = rep("NA", length(.$Year))) %>%
  as.data.frame(.)

# Make columns the same type
cleandata2010$`# of Food Insecure Children` <-
  as.character(cleandata2010$`# of Food Insecure Children`)
cleandata2010 <- janitor::clean_names(cleandata2010) %>%
  select(
    fips,
    state,
    county_state,
    year,
    overall_food_insecurity_rate, number_of_food_insecure_persons_overall,
    low_threshold_in_state, low_threshold_type, high_threshold_in_state,
    high_threshold_type, percent_fi_low_threshold, percent_fi_btwn_thresholds,
    percent_fi_high_threshold, percent_fi_low_threshold_2,
    child_food_insecurity_rate, number_of_food_insecure_children,
    percent_food_insecure_children_in_hh_w_hh_incomes_below_185_fpl,
    percent_food_insecure_children_in_hh_w_hh_incomes_above_185_fpl,
    cost_per_meal, weighted_annual_food_budget_shortfall
  )
# Merge into one data frame
cleandata <- rbind(cleandata2019, cleandata2010)
FeedData2019_2021 <- rawdata2019_2021%>%
  filter(State=="CA")

FeedData2019_2021 <- FeedData2019_2021[,c(3:5)]

FeedData2019_2021 <- janitor::clean_names(FeedData2019_2021)

FeedData2019_2021[,3] <- sapply(FeedData2019_2021[,3],function(x) as.numeric(gsub("%","",x)))

FeedData2019_2021 <- FeedData2019_2021[c("year","county_state","overall_food_insecurity_rate")]
cleanFeed <- function(data){

  dataCA <- data%>%
    filter(State=="CA")

  dataCA <- dataCA[,c(3,4)]

  dataCA[,2] <- sapply(dataCA[,2],function(x) as.numeric(gsub("%","",x)))

  colnames(dataCA) <- c("county_state","overall_food_insecurity_rate")

  return (dataCA)
}

FeedData2018 <- cbind("year"=rep(2018,nrow(cleanFeed(rawdata2018))),cleanFeed(rawdata2018))
FeedData2017 <- cbind("year"=rep(2017,nrow(cleanFeed(rawdata2017))),cleanFeed(rawdata2017))
FeedData2016 <- cbind("year"=rep(2016,nrow(cleanFeed(rawdata2016))),cleanFeed(rawdata2016))
FeedData2015 <- cbind("year"=rep(2015,nrow(cleanFeed(rawdata2015))),cleanFeed(rawdata2015))
FeedData2014 <- cbind("year"=rep(2014,nrow(cleanFeed(rawdata2014))),cleanFeed(rawdata2014))
FeedData2013 <- cbind("year"=rep(2013,nrow(cleanFeed(rawdata2013))),cleanFeed(rawdata2013))
FeedData2012 <- cbind("year"=rep(2012,nrow(cleanFeed(rawdata2012))),cleanFeed(rawdata2012))
FeedData2011 <- cbind("year"=rep(2011,nrow(cleanFeed(rawdata2011))),cleanFeed(rawdata2011))
FeedData2010 <- cbind("year"=rep(2010,nrow(cleanFeed(rawdata2010))),cleanFeed(rawdata2010))
FeedData <- rbind(FeedData2010,FeedData2011,FeedData2012,FeedData2013,FeedData2014,FeedData2015,FeedDat

countyNameDisable <- FeedData$county_state

FeedData <- FeedData%>%
  mutate(county_state = gsub(", California", "", county_state))%>%
  rename(county = county_state)%>%
  arrange(county)

countyName <- FeedData$county
AgeData2019CA <- AgeData2019 %>%
  filter(NAME %in% c(countyNameDisable, 'Geographic Area Name')) %>%
  row_to_names(row_number = 1)

colnames(AgeData2019CA)[3] <- "Total Population"
```

```r
Age2019 <- AgeData2019CA %>%
  .[,!grepl("Margin of Error", colnames(.))] %>%
  .[,!grepl("Annotation", colnames(.))] %>%
  .[,!grepl("Percent", colnames(.))] %>%
  .[, grepl("Geographic Area Name|Total Population|Population under 18 years|Population 65 years and ove
  .[,!grepl("years!!|over!!", colnames(.))]

Age2019$'Geographic Area Name' <-gsub(", California", "", Age2019$'Geographic Area Name')

colnames(Age2019)[1] <- "county"

colnames(Age2019) <- gsub(".*\\Estimate!!", "", colnames(Age2019))
colnames(Age2019) <- gsub("Total civilian noninstitutionalized population!!",
                       "",colnames(Age2019))
colnames(Age2019) <-gsub("DISABILITY TYPE BY DETAILED AGE!!", "", colnames(Age2019))
colnames(Age2019) <- gsub("!!", ": ", colnames(Age2019))
colnames(Age2019) <- gsub(": :", ":", colnames(Age2019))

Age2019 <- Age2019 %>% mutate_at(-1, as.numeric)

TailAge2019 <- Age2019[,c(1:4)]
for (i in c(3:4)) {
  TailAge2019[, i] <- Age2019[, i] / Age2019[, 2]
}

colnames(TailAge2019) <- c("county","total_population","population_under_18","population_over_65")
cleanDis1 <- function(data){

  dataCA <- data %>%
    filter(NAME %in% c(countyNameDisable, 'Geographic Area Name')) %>%
    row_to_names(row_number = 1)

  colnames(dataCA)[3] <- "Total Population"

  dataCA1 <- dataCA %>%
    .[,!grepl("Margin of Error", colnames(.))] %>%
    .[,!grepl("Annotation", colnames(.))] %>%
    .[, grepl("Geographic Area Name|Percent with a disability",colnames(.))] %>%
    .[,!grepl("population!",colnames(.))]

  dataCA1$'Geographic Area Name' <-gsub(", California", "", dataCA1$'Geographic Area Name')
  colnames(dataCA1)[1] <- "county"
  colnames(dataCA1)[2] <- "Percent with a disability"

  dataCA1 <- dataCA1 %>% mutate_at(-1, as.numeric)

  return (dataCA1)

}

Dis2021 <- cbind("year"=rep(2021,nrow(cleanDis1(DisData2021))),cleanDis1(DisData2021))
Dis2020 <- cbind("year"=rep(2020,nrow(cleanDis1(DisData2020))),cleanDis1(DisData2020))
Dis2019 <- cbind("year"=rep(2019,nrow(cleanDis1(DisData2019))),cleanDis1(DisData2019))
```

```r
Dis2018 <- cbind("year"=rep(2018,nrow(cleanDis1(DisData2018)[,c(1,2)])),cleanDis1(DisData2018)[,c(1,2)]
Dis2017 <- cbind("year"=rep(2017,nrow(cleanDis1(DisData2017)[,c(1,2)])),cleanDis1(DisData2017)[,c(1,2)]
Dis2016 <- cbind("year"=rep(2016,nrow(cleanDis1(DisData2016)[,c(1,2)])),cleanDis1(DisData2016)[,c(1,2)]
Dis2015 <- cbind("year"=rep(2015,nrow(cleanDis1(DisData2015)[,c(1,2)])),cleanDis1(DisData2015)[,c(1,2)]
Dis2014 <- cbind("year"=rep(2014,nrow(cleanDis1(DisData2014)[,c(1,2)])),cleanDis1(DisData2014)[,c(1,2)]
Dis2013 <- cbind("year"=rep(2013,nrow(cleanDis1(DisData2013)[,c(1,2)])),cleanDis1(DisData2013)[,c(1,2)]
cleanDis2 <- function(data){

  dataCA <- data %>%
    filter(NAME %in% c(countyNameDisable, 'Geographic Area Name')) %>%
    row_to_names(row_number = 1)

  colnames(dataCA)[3] <- "Total Population"

  dataCA1 <- dataCA %>%
    .[,!grepl("Margin of Error", colnames(.))] %>%
    .[,!grepl("Annotation", colnames(.))] %>%
    .[, grepl("Geographic Area Name|Percent with a disability",colnames(.))] %>%
    .[,!grepl("population!",colnames(.))]

  dataCA1$'Geographic Area Name' <-gsub(", California", "", dataCA1$'Geographic Area Name')
  colnames(dataCA1)[1] <- "county"
  colnames(dataCA1)[2] <- "Percent with a disability"

  dataCA1 <- (dataCA1[,c(1,2)] %>% mutate_at(-1, as.numeric))

  return (dataCA1)
}

Dis2012 <- cbind("year"=rep(2012,nrow(cleanDis2(DisData2012))),cleanDis2(DisData2012))
Dis2011 <- cbind("year"=rep(2011,nrow(cleanDis2(DisData2011))),cleanDis2(DisData2011))
Dis2010 <- cbind("year"=rep(2010,nrow(cleanDis2(DisData2010))),cleanDis2(DisData2010))
totalDisability <- rbind(Dis2010,Dis2011,Dis2012,Dis2013,Dis2014,Dis2015,Dis2016,Dis2017,Dis2018,Dis2019
totalDisability <- totalDisability %>% arrange(county)
colnames(totalDisability) <- c("year","county","percent_disabled")
UnemploymentData <- unemploymentdataraw%>%
  filter(area_type=="County", status_preliminary_final=="Final")%>%
  filter(year>=2010 & year<2022)%>%
  filter(!area_name %in% c("Non Residential County","Resident Out of State County","Unallocated County"
  group_by(year, area_name)%>%
  summarise("unemployment_rate_avg"=mean(unemployment_rate))%>%
  distinct(.)%>%
  ungroup() %>%
  rename("county"="area_name")
IncomeData <- incomedataraw%>%
  filter(taxable_year >= 2010 & taxable_year <= 2021)%>%
  filter(!county %in% c("Nonresident","Resident Out of State County","Unallocated","Resident Out of Stat
  rename("year"="taxable_year")%>%
  mutate("county"=paste(.$county, "County"))%>%
  arrange(year,county)

IncomeData <- IncomeData[,c(1,2,6)]
majorDF <- FeedData %>% full_join(totalDisability)
majorDF <- majorDF %>% full_join(UnemploymentData)
majorDF <- majorDF %>% full_join(IncomeData)

majorDF <- majorDF %>% filter(!county %in% c("Resident Out of State County","Nonresident20 County","Resi

majorDF2019 <- majorDF %>% filter(year==2019)
majorDF2019 <- TailAge2019 %>% inner_join(majorDF2019)
majorDF[,c(1,3)] %>%
  mutate(year=as.factor(year))%>%
  ggplot(aes(x=year, y=overall_food_insecurity_rate, fill=(year==c(2020,2021)))) +
  geom_boxplot(show.legend = F)+
  scale_fill_manual(values=c("#4E5745","#AC5B4C"))+
  theme_minimal()+
  theme(panel.background=element_rect(fill="#E5DACF",color="#E5DACF",size=0.5,linetype="solid"))+
  labs(title="Overall Food Insecurity Rate per Year",x="Year",y="Food Insecurity Rate (%)")
majorDF[,c(1,6)]%>%
  mutate(year=as.factor(year))%>%
  ggplot(aes(x=year, y=median_income, fill=(year==2020))) +
  geom_boxplot(show.legend = F)+
  scale_fill_manual(values=c("#4E5745","#AC5B4C"))+
  theme_minimal()+
  theme(panel.background=element_rect(fill="#E5DACF",color="#E5DACF",size=0.5,linetype="solid"))+
  labs(title="Overall Median Income per Year",x="Year",y="Median Income ($)")
majorDF[,c(1,5)] %>%
  mutate(year=as.factor(year))%>%
  ggplot(aes(x=year, y=unemployment_rate_avg, fill=(year==2020 | year==2021))) +
  geom_boxplot(show.legend = F)+
  scale_fill_manual(values=c("#4E5745","#AC5B4C"))+
  theme_minimal()+
  theme(panel.background=element_rect(fill="#E5DACF",color="#E5DACF",size=0.5,linetype="solid"))+
  labs(title="Overall Unemployment Rate per Year",x="Year",y="Unemployment Rate (%)")
majorDF%>%
  ggplot(aes(x=percent_disabled, y=overall_food_insecurity_rate))+
  geom_point(color="#4E5745")+
  geom_smooth(method="lm", color="#AC5B4C", show.legend = F)+
  theme_minimal()+
  theme(panel.background = element_rect(fill="#E5DACF", color = "#E5DACF", size = 0.5, linetype = "soli
  labs(title="Relationship Between FI and Disability Rate",
       x="Disability Rate (%)",
       y="Food Insecurity Rate (%)")
majorDF%>%
  ggplot(aes(x=median_income, y=overall_food_insecurity_rate))+
  geom_point(color="#4E5745")+
  geom_smooth(method="lm", color="#AC5B4C", show.legend = F)+
  theme_minimal()+
  theme(panel.background = element_rect(fill="#E5DACF", color = "#E5DACF", size = 0.5, linetype = "soli
  labs(title="Relationship Between FI and Median Income",
       x="Median Income ($)",
       y="Food Insecurity Rate (%)")
majorDF%>%
  ggplot(aes(x=unemployment_rate_avg, y=overall_food_insecurity_rate))+
  geom_point(color="#4E5745")+
  geom_smooth(method="lm", color="#AC5B4C", show.legend = F)+
  theme_minimal()+
  theme(panel.background = element_rect(fill="#E5DACF", color = "#E5DACF", size = 0.5, linetype = "soli
  labs(title="Relationship Between FI and Unemployment Rate",
       x="Unemployment Rate (%)",
       y="Food Insecurity Rate (%)")
# Define Training Data (2010-2019) and Test Data (2020)
majorDFtrain <- na.omit(majorDF)%>%
  filter(year<2020)

majorDFtest <- na.omit(majorDF)%>%
  filter(year==2020)

model <- lm(overall_food_insecurity_rate ~ unemployment_rate_avg + percent_disabled + median_income, da

lm_results <- data.frame(
  "county" = majorDFtest$county,
  "predicted_food_insecurity_rate" = as.numeric(model$coefficients[1]) + as.numeric(model$coefficients[
    majorDFtest$unemployment_rate_avg + as.numeric(model$coefficients[3]) *
    majorDFtest$percent_disabled + as.numeric(model$coefficients[4]) * majorDFtest$median_income,
  "actual_food_insecurity_rate" = majorDFtest$overall_food_insecurity_rate
)

lm_overall_acc20 <- 1 - mean((abs(lm_results$predicted_food_insecurity_rate-lm_results$actual_food_inse
# Define Training Data (2010-2019) and Test Data (2019)
majorDFtrain <- na.omit(majorDF)%>%
  filter(year<2019)

majorDFtest <- na.omit(majorDF)%>%
  filter(year==2019)

model <- lm(overall_food_insecurity_rate ~ unemployment_rate_avg + percent_disabled + median_income, da

lm_results <- data.frame(
  "county" = majorDFtest$county,
  "predicted_food_insecurity_rate" = as.numeric(model$coefficients[1]) + as.numeric(model$coefficients[
    majorDFtest$unemployment_rate_avg + as.numeric(model$coefficients[3]) *
    majorDFtest$percent_disabled + as.numeric(model$coefficients[4]) * majorDFtest$median_income,
  "actual_food_insecurity_rate" = majorDFtest$overall_food_insecurity_rate
)

lm_overall_acc19 <- 1 - mean((lm_results$predicted_food_insecurity_rate-lm_results$actual_food_insecuri
# Define Training Data (2010-2019) and Test Data (2020)
majorDFtrain <- na.omit(majorDF)%>%
  filter(year<2020)

majorDFtest <- na.omit(majorDF)%>%
  filter(year==2020)

train_rf <- train(overall_food_insecurity_rate ~ unemployment_rate_avg + median_income + percent_disable
                  tuneGrid=data.frame(mtry=1:2),
                  trControl=trainControl(method="cv", number=5))

pred <- as.numeric(predict(train_rf, newdata=majorDFtest))
```

```
rf_results <- data.frame("county"=majorDFtest$county,
                          "predicted_food_insecurity_rate"= pred,
                          "actual_food_insecurity_rate"= majorDFtest$overall_food_insecurity_rate)

rf_overall_acc20 <- 1 - mean((rf_results$predicted_food_insecurity_rate-rf_results$actual_food_insecuri
# Define Training Data (2010-2019) and Test Data (2020)
majorDFtrain <- na.omit(majorDF)%>%
  filter(year<2019)

majorDFtest <- na.omit(majorDF)%>%
  filter(year==2019)

train_rf <- train(overall_food_insecurity_rate ~ unemployment_rate_avg + median_income + percent_disabl
                  tuneGrid=data.frame(mtry=1:2),
                  trControl=trainControl(method="cv", number=5))

pred <- as.numeric(predict(train_rf, newdata=majorDFtest))

rf_results <- data.frame("county"=majorDFtest$county,
                          "predicted_food_insecurity_rate"= pred,
                          "actual_food_insecurity_rate"= majorDFtest$overall_food_insecurity_rate)

rf_overall_acc19 <- 1 - mean((rf_results$predicted_food_insecurity_rate-rf_results$actual_food_insecuri
lm_trainr <- function(c,y) {
  # Define Training Data (2010-2019) and Test Data (2020)
  majorDFtrain <- na.omit(majorDF) %>%
    filter(year < y)%>%
    filter(county==c)

  majorDFtest <- na.omit(majorDF) %>%
    filter(year < y)%>%
    filter(county==c)

  if (nrow(majorDFtrain) == 0 | nrow(majorDFtest) == 0){

    rf_results <- data.frame("county"=c,
           "predicted_food_insecurity_rate"=NA,
           "actual_food_insecurity_rate"=NA)

    rf_acc <- NA

    return(list(results=rf_results, accuracy=rf_acc))
  }

  model <-
    lm(
      overall_food_insecurity_rate ~ unemployment_rate_avg + percent_disabled + median_income,
      data = majorDFtrain
    )

  lm_results <- data.frame(
    "county" = majorDFtrain$county,
    "predicted_food_insecurity_rate" = as.numeric(model$coefficients[1]) +
      as.numeric(model$coefficients[2]) *  majorDFtrain$unemployment_rate_avg +
      as.numeric(model$coefficients[3]) * majorDFtrain$percent_disabled +
      as.numeric(model$coefficients[4]) * majorDFtrain$median_income,
    "actual_food_insecurity_rate" = majorDFtrain$overall_food_insecurity_rate
  )

  lm_acc <-
    1 - mean(
      abs(
        lm_results$predicted_food_insecurity_rate - lm_results$actual_food_insecurity_rate
      ) / lm_results$actual_food_insecurity_rate
    )

  return(list(results = lm_results, accuracy = lm_acc))
}

results20 <- data.frame(county=c(), predicted_food_insecurity_rate=c(), actual_food_insecurity=c())
accuracy20 <- c()

for (c in unique(majorDF$county)){
  trainr_data <- lm_trainr(c,2020)
  #results20 <- rbind(results20, trainr_data[[1]])
  accuracy20 <- c(accuracy20, trainr_data[[2]])
}

#results20$accuracy20 <- accuracy20
lm_county_acc20 <- mean(accuracy20, na.rm=T)

results19 <- data.frame(county=c(), predicted_food_insecurity_rate=c(), actual_food_insecurity=c())
accuracy19 <- c()

for (c in unique(majorDF$county)){
  trainr_data <- lm_trainr(c,2019)
  results19 <- rbind(results19, trainr_data[[1]])
  accuracy19 <- c(accuracy19, trainr_data[[2]])
}

#results19$accuracy19 <- accuracy19
lm_county_acc19 <- mean(accuracy19, na.rm=T)
rf_trainr <- function(c,y) {
  # Define Training Data (2010-2019) and Test Data (2020)
  majorDFtrain <- na.omit(majorDF) %>%
    filter(year < y)%>%
    filter(county==c)

  majorDFtest <- na.omit(majorDF) %>%
    filter(year < y)%>%
    filter(county==c)

  if (nrow(majorDFtrain) <= 1 | nrow(majorDFtest) == 0){

    rf_results <- data.frame("county"=c,
           "predicted_food_insecurity_rate"=NA,
           "actual_food_insecurity_rate"=NA)

    rf_acc <- NA

    return(list(results=rf_results, accuracy=rf_acc))
  }

  train_rf <-
    train(
      overall_food_insecurity_rate ~ unemployment_rate_avg + percent_disabled + median_income,
      method = "rf",
      data = majorDFtrain,
      tuneGrid = data.frame(mtry = 1:2),
      trControl = trainControl(method = "cv", number = 5)
    )

  pred <- as.numeric(predict(train_rf, newdata=majorDFtest))

  rf_results <- data.frame("county"=majorDFtest$county,
                            "predicted_food_insecurity_rate"= pred,
                            "actual_food_insecurity_rate"= majorDFtest$overall_food_insecurity_rate)

  rf_acc <-
    1 - mean((
      rf_results$predicted_food_insecurity_rate - rf_results$actual_food_insecurity_rate
    ) / rf_results$actual_food_insecurity_rate
    )

  return(list(results=rf_results, accuracy=rf_acc))
}

results20 <- data.frame(county=c(), predicted_food_insecurity_rate=c(), actual_food_insecurity=c())
accuracy20 <- c()

for (c in unique(majorDF$county)){
  trainr_data <- rf_trainr(c,2020)
  #results20 <- rbind(results20, trainr_data[[1]])
  accuracy20 <- c(accuracy20, trainr_data[[2]])
}

#results20$accuracy20 <- accuracy20
rf_county_acc20 <- mean(accuracy20, na.rm=T)

results19 <- data.frame(county=c(), predicted_food_insecurity_rate=c(), actual_food_insecurity=c())
accuracy19 <- c()

for (c in unique(majorDF$county)){
  trainr_data <- rf_trainr(c,2019)
  results19 <- rbind(results19, trainr_data[[1]])
  accuracy19 <- c(accuracy19, trainr_data[[2]])
}

#results19$accuracy19 <- accuracy19
```

```r
rf_county_acc19 <- mean(accuracy19, na.rm=T)
kable(data.frame("Model Type"=c("Overall Linear Regression", "Overall Random Forest",
                                "By County Linear Model", "By County Random Forest"),
                 "Test Accuracy 2019"=c(lm_overall_acc19, rf_overall_acc19,
                                        lm_county_acc19, rf_county_acc19),
                "Test Accuracy 2020"=c(lm_overall_acc20, rf_overall_acc20,
                                       lm_county_acc20, rf_county_acc20)))
```