Project

Park

2/14/2020

Header

```
library(readxl, lib.loc = "/Library/Frameworks/R.framework/Versions/3.6/Resources/library")
library(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
```

```
library(maps)
library(knitr)
library(geoR)
```

```
## ----------------------------------------------------------
##  Analysis of Geostatistical Data
##  For an Introduction to geoR go to http://www.leg.ufpr.br/geoR
##  geoR version 1.7-5.2.2 (built on 2016-05-02) is now loaded
## ----------------------------------------------------------
```

```
library(data.table)
```

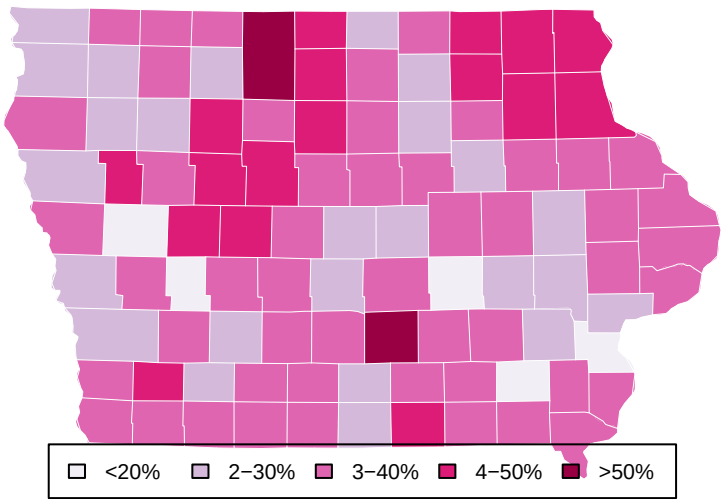
```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

Candidate percentage per county in Iowa(2020)

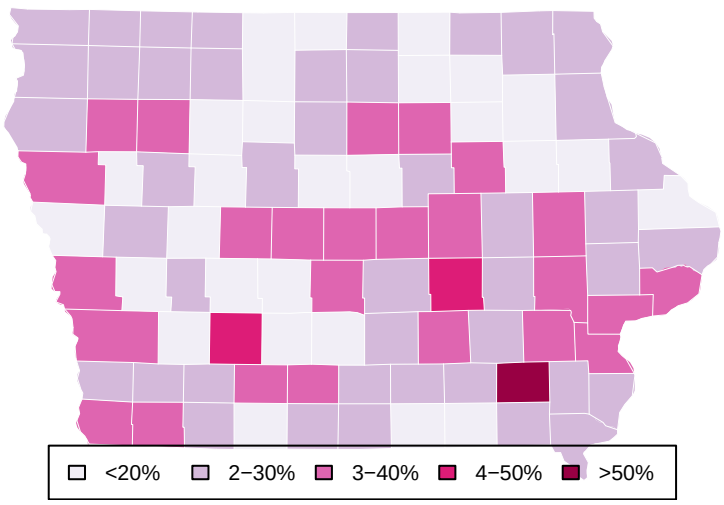
```
candidate <- read_xlsx("~/Dropbox/School/UCLA/stat c173/Iowa Project/Candidate.xlsx")
candidate <- candidate %>% group_by(County) %>% mutate(percantage = sum(Buttigieg,Sanders,Warren,Biden))

newcandidate <- data.frame(candidate[,1],candidate[,c(2,3,4,5)] / candidate$percantage)
newcandidate[,c(2,3,4,5)] <- newcandidate[,c(2,3,4,5)] %>% round(2)
```

```r
###Graph for Candidate Buttigieg
#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#DF65B0", "#DD1C77", "#980043")

newcandidate$colorBuckets <- as.numeric(cut(newcandidate$Buttigieg, c(0, 0.2, 0.3, 0.4, 0.5, 1)))

#Legend:
leg.txt <- c("<20%", "2-30%", "3-40%", "4-50%", ">50%")

colorsmatched <- newcandidate$colorBuckets

#Draw map for Iowa:
map("county", "iowa",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projection = "p

map("county","iowa", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="polyconic"

title("Candidate Buttigieg percentage per county in Iowa, 2020")
legend("bottom", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```

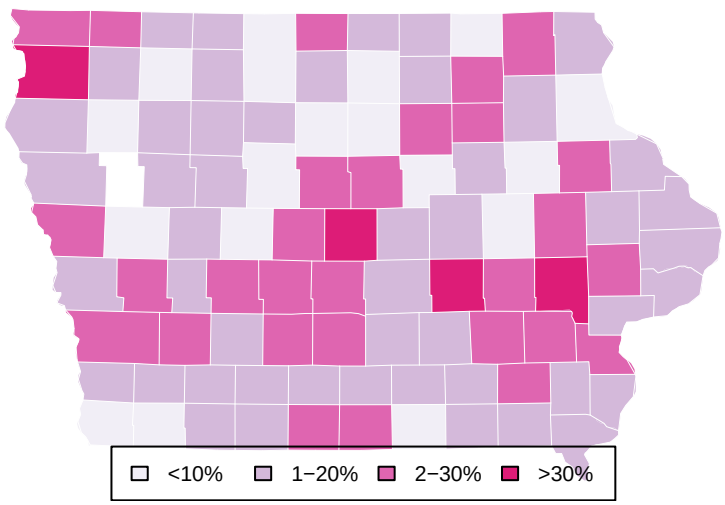



```r
###Graph for Candidate Sanders
newcandidate$colorBuckets <- as.numeric(cut(newcandidate$Sanders, c(0, 0.2, 0.3, 0.4, 0.5, 1)))

#Legend:
leg.txt <- c("<20%", "2-30%", "3-40%", "4-50%", ">50%")

colorsmatched <- newcandidate$colorBuckets
```

```
#Draw map for Iowa:
map("county", "iowa",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projection = "p

map("county","iowa", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="polyconic"

title("Candidate Sanders percentage per county in Iowa, 2020")
legend("bottom", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```

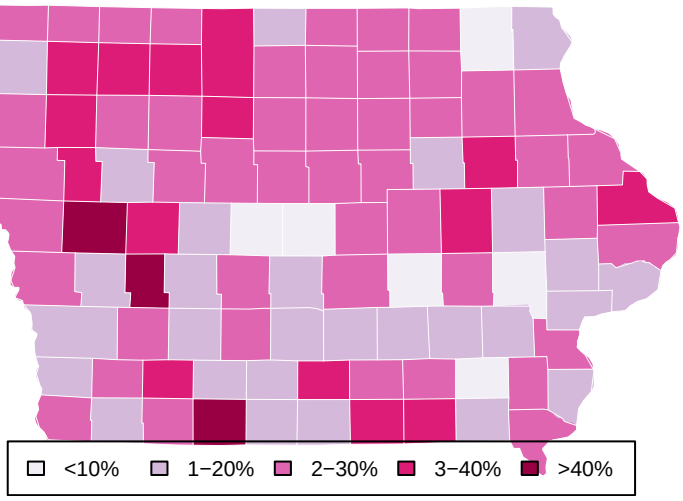



```
###Graph for Candidate Warren
#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#DF65B0", "#DD1C77")

newcandidate$colorBuckets <- as.numeric(cut(newcandidate$Warren, c(0, 0.1, 0.2, 0.3, 1)))

#Legend:
leg.txt <- c("<10%", "1-20%", "2-30%", ">30%")

colorsmatched <- newcandidate$colorBuckets

#Draw map for Iowa:
map("county", "iowa",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projection = "p

map("county","iowa", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="polyconic"

title("Candidate Warren percentage per county in Iowa, 2020")
legend("bottom", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```



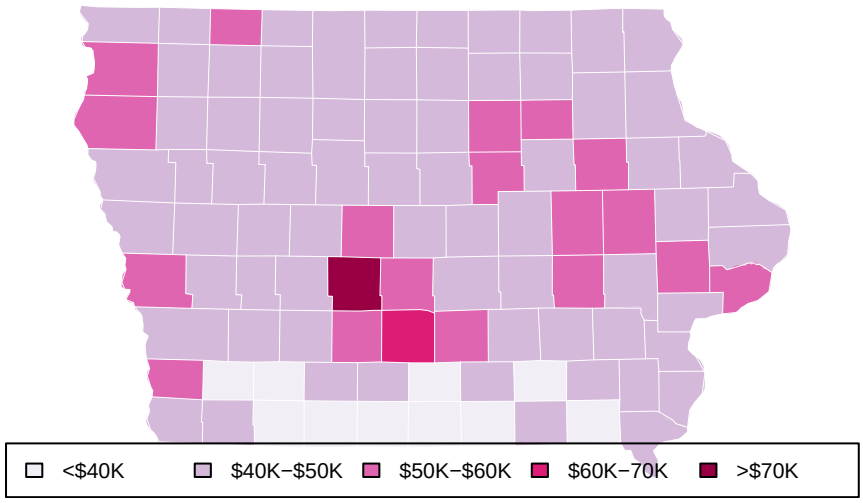

```
###Graph for Candidate Biden
#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#DF65B0", "#DD1C77", "#980043")

newcandidate$colorBuckets <- as.numeric(cut(newcandidate$Biden, c(0, 0.1, 0.2, 0.3, 0.4, 1)))

#Legend:
leg.txt <- c("<10%", "1-20%", "2-30%", "3-40%", ">40%")

colorsmatched <- newcandidate$colorBuckets

#Draw map for Iowa:
map("county", "iowa",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projection = "p

map("county","iowa", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="polyconic"

title("Candidate Biden percentage per county in Iowa, 2020")
legend("bottom", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```

Candidate Biden percentage per county in Iowa, 2020



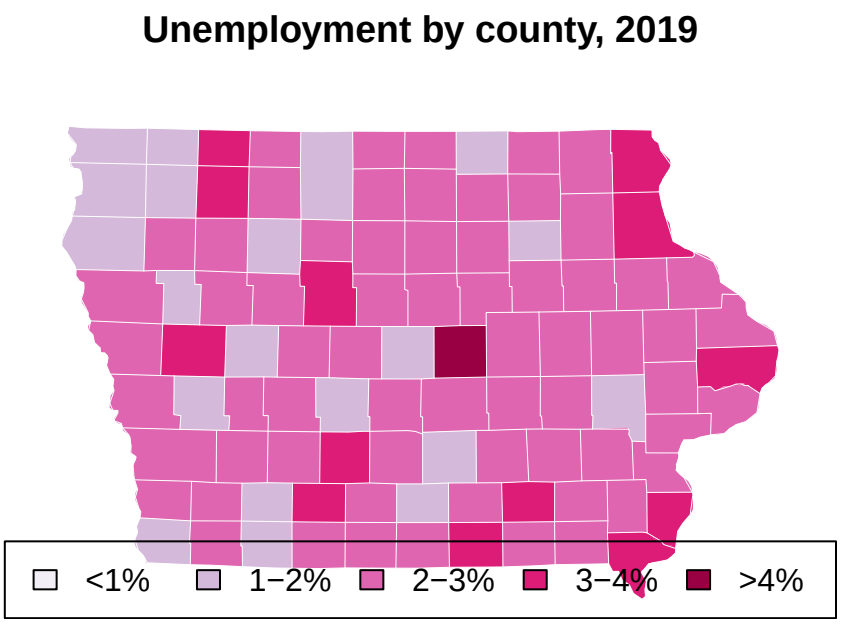
Median income per county in Iowa(2009)

```
income <- read_excel("~/Dropbox/School/UCLA/stat c173/Iowa Project/coincomemedian.xls")
colnames(income) <- c("County", "Estimate")

income$colorBuckets <- as.numeric(cut(income$Estimate, c(30000,40000,50000,60000,70000,80000)))

#Legend:
leg.txt <- c("<$40K", "$40K-$50K", "$50K-$60K", "$60K-70K", ">$70K")

colorsmatched <- income$colorBuckets

#Draw map for Iowa:
map("county", "iowa",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projection = "p

map("county","iowa", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="polyconic"

title("Median Income Estimate by county, 2009")
legend("bottom", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```



Unemployment per county in Iowa(2019)

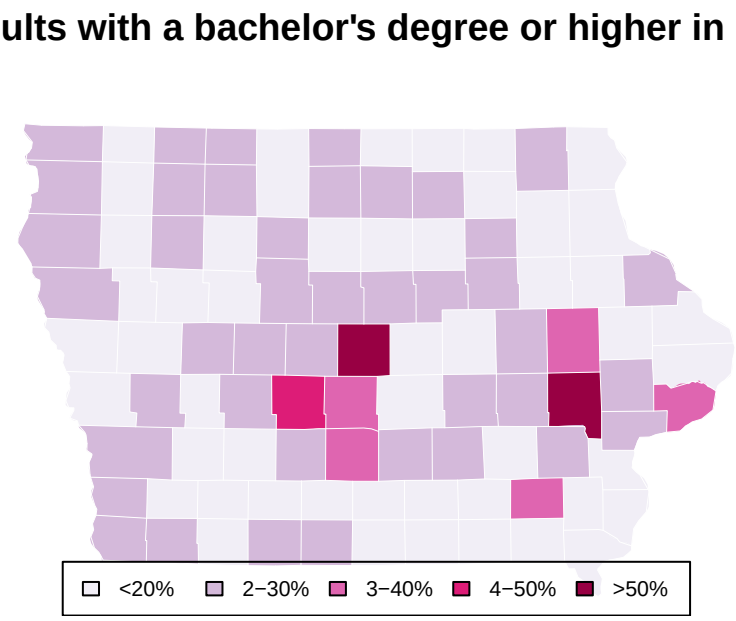
```
unemploy <- read_excel("~/Dropbox/School/UCLA/stat c173/Iowa Project/Unemployment.xls")
unemploy <- unemploy %>% filter(State == "IA")
unemploy <- unemploy[-1,]
unemploy <- unemploy[,c(1,2,53,54)]

#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#DF65B0", "#DD1C77", "#980043")

unemploy$colorBuckets <- as.numeric(cut(unemploy$Unemployment_rate_2018, c(0, 1, 2, 3, 4, 10)))

#Legend:
leg.txt <- c("<1%", "1-2%", "2-3%", "3-4%", ">4%")

colorsmatched <- unemploy$colorBuckets

#Draw map for Iowa:
map("county", "iowa",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projection = "p

map("county","iowa", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="polyconic"

title("Unemployment by county, 2019")
legend("bottom", leg.txt, horiz = TRUE, fill = colors)
```



Education per county in Iowa(2019)

```
edu <- read_excel("~/Dropbox/School/UCLA/stat c173/Iowa Project/Education.xls")
colnames(edu)[3] <- c("County")
edu <- edu %>% select(State, County, `Percent of adults with a bachelor's degree or higher, 2013-17`)
iaedu <- edu %>% filter(State == "IA")
iaedu <- iaedu[-1,]

###Graph for Candidate Buttigieg
#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#DF65B0", "#DD1C77", "#980043")

iaedu$colorBuckets <- as.numeric(cut(iaedu$`Percent of adults with a bachelor's degree or higher, 2013-

#Legend:
leg.txt <- c("<20%", "2-30%", "3-40%", "4-50%", ">50%")

colorsmatched <- iaedu$colorBuckets

#Draw map for Iowa:
map("county", "iowa",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projection = "p

map("county","iowa", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="polyconic"
```

```
title("Percent of adults with a bachelor's degree or higher in Iowa, 2013-17")
legend("bottom", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```



Adjacency Matrix for Iowa State

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
## The following object is masked from 'package:base':
##
##     union
```

```
a <- read.table("https://www2.census.gov/geo/docs/reference/county_adjacency.txt", sep="\t", fill=FALSE

iowa <- a[c(5349:6110),]

for(i in 1:761){
```

```
  if(is.na(iowa[i+1,2])){iowa[i+1,c(1,2)] <- iowa[i,c(1,2)]}
}

iowa_num <- iowa[,c(2,4)]
iowa_num <- iowa_num %>% filter(V4 < 20000 & V4 > 19000 & V2 != V4)
iowa_num <- (iowa_num - 18999)/2

iowa_adj <- get.adjacency(graph.edgelist(as.matrix(iowa_num), directed=FALSE))/2
w <- as.matrix(iowa_adj)
```

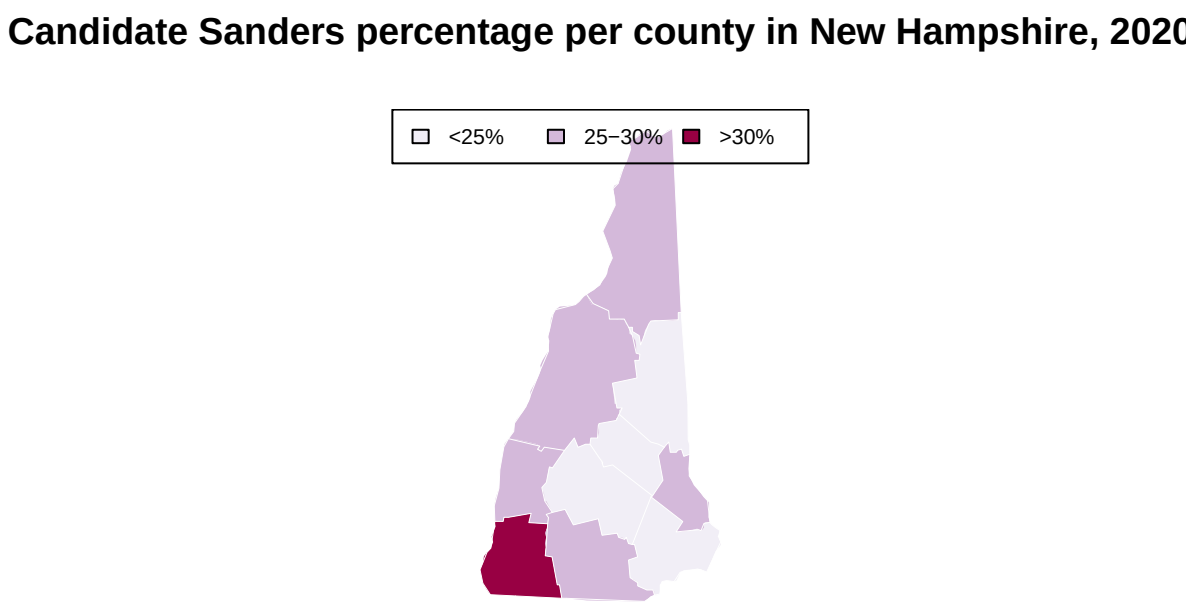
Adjacency Matrix for New Hampshire State

```
NH <- a[c(12208:12274),]

for(i in 1:66){
  if(is.na(NH[i+1,2])){NH[i+1,c(1,2)] <- NH[i,c(1,2)]}
}

NH_num <- NH[,c(2,4)]
NH_num <- NH_num %>% filter(V4 < 34000 & V4 > 33000 & V2 != V4)
NH_num <- (NH_num - 32999)/2

NH_adj <- get.adjacency(graph.edgelist(as.matrix(NH_num), directed=FALSE))/2
w2 <- as.matrix(NH_adj)
```

Candidate percentage per county in New Hampshire(2020)

```
nhcandidate <- read_excel("~/Dropbox/School/UCLA/stat c173/Iowa Project/candidate2.xlsx")
coln <- colnames(nhcandidate)
nhcandidate <- transpose(nhcandidate)
colnames(nhcandidate) <- nhcandidate[1,]
nhcandidate <- nhcandidate[-c(1,12),]
nhcandidate$County <- coln[c(2:11)]
nhcandidate[,c(1:35)] <- nhcandidate[,c(1:35)] %>% unlist %>% as.numeric()
nhcandidate[,c(1,2)] <- nhcandidate[,c(1,2)]/nhcandidate$Total

nhcandidate <- nhcandidate %>% select(`County`,`Bernie Sanders`)

###Graph for Candidate Sanders
#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#980043")
nhcandidate$colorBuckets <- as.numeric(cut(nhcandidate$`Bernie Sanders`, c(0, 0.25, 0.3, 1)))

#Legend:
leg.txt <- c("<25%", "25-30%", ">30%")

colorsmatched <- nhcandidate$colorBuckets

#Draw map for Iowa:
map("county", "new hampshire",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projec
```

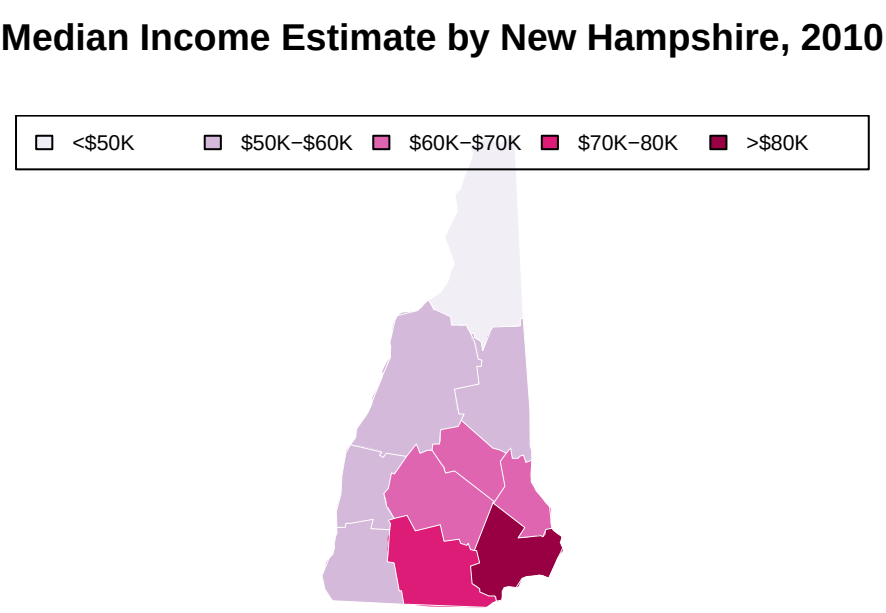
```
map("county","new hampshire", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="p

title("Candidate Sanders percentage per county in New Hampshire, 2020")
legend("top", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```



Median income per county in New Hampshire(2010)

```
nhincome <- read_excel("~/Dropbox/School/UCLA/stat c173/Iowa Project/nhincome.xlsx")

###Graph for Candidate Biden
#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#DF65B0", "#DD1C77", "#980043")


nhincome$colorBuckets <- as.numeric(cut(nhincome$`Median household income`, c(40000,50000,60000,70000,8

#Legend:
leg.txt <- c("<$50K", "$50K-$60K", "$60K-$70K", "$70K-80K", ">$80K")

colorsmatched <- nhincome$colorBuckets

#Draw map for Iowa:
map("county", "new hampshire",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projec
```

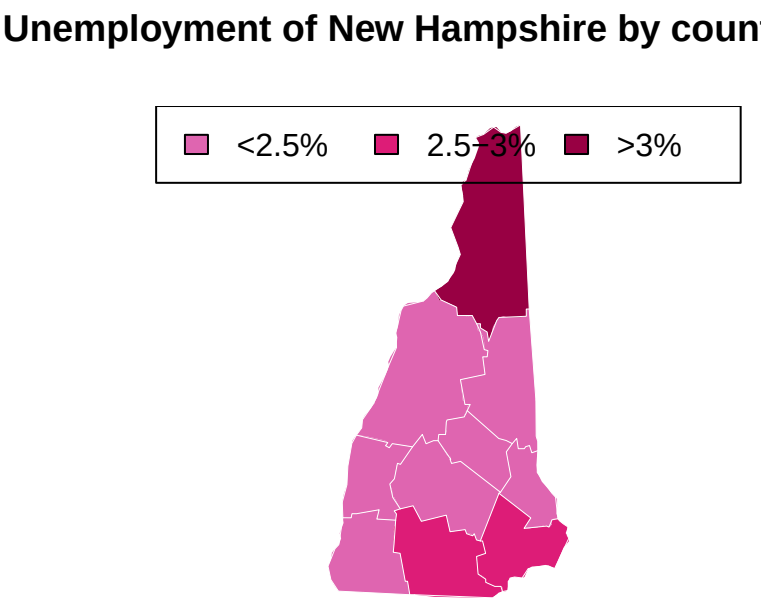
```
map("county","new hampshire", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="p

title("Median Income Estimate by New Hampshire, 2010")
legend("top", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```



Unemployment per county in New Hampshire(2019)

```
nhunemploy <- read_excel("~/Dropbox/School/UCLA/stat c173/Iowa Project/Unemployment.xls")
nhunemploy <- nhunemploy %>% filter(State == "NH")
nhunemploy <- nhunemploy[-1,]
nhunemploy <- nhunemploy[,c(1,2,53,54)]

#Define color buckets
colors = c( "#DF65B0", "#DD1C77", "#980043")

nhunemploy$colorBuckets <- as.numeric(cut(nhunemploy$Unemployment_rate_2018, c(2, 2.5, 3, 10)))

#Legend:
leg.txt <- c("<2.5%", "2.5-3%", ">3%")

colorsmatched <- nhunemploy$colorBuckets

#Draw map for Iowa:
map("county", "new hampshire",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projec
```

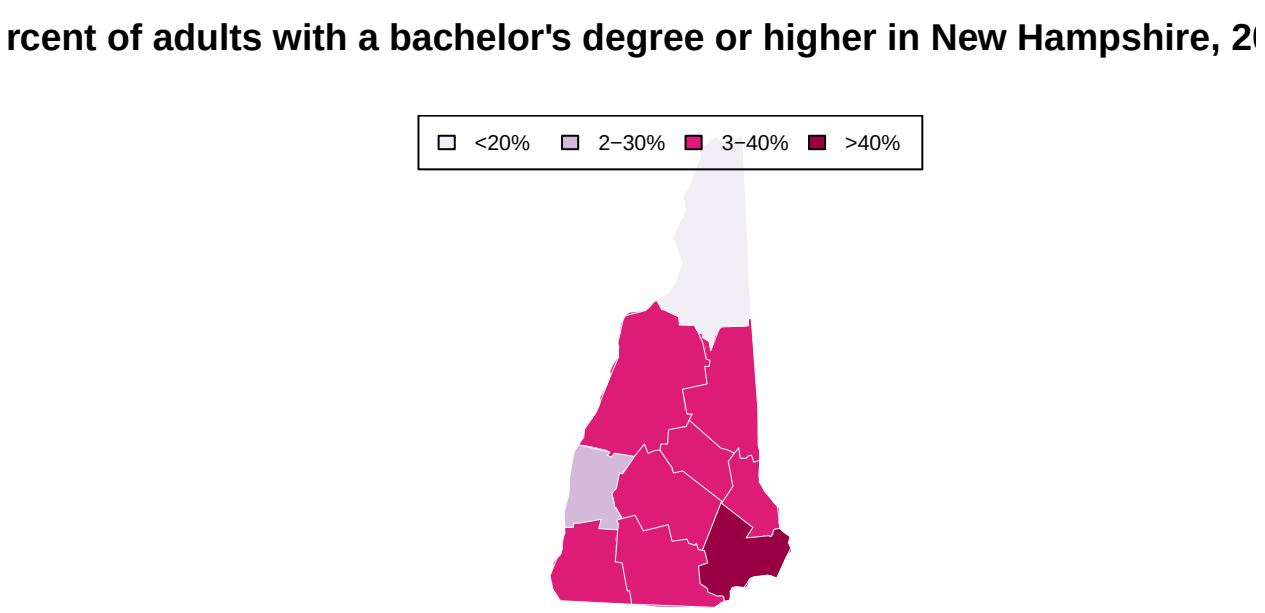
```
map("county","new hampshire", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="p

title("Unemployment of New Hampshire by county, 2019")
legend("top", leg.txt, horiz = TRUE, fill = colors)
```



Education per county in New Hampshire(2019)

```
nhedu <- edu %>% filter(State == "NH")
nhedu <- nhedu[-1,]

###Graph for Candidate Buttigieg
#Define color buckets
colors = c("#F1EEF6", "#D4B9DA", "#DD1C77", "#980043")

nhedu$colorBuckets <- as.numeric(cut(nhedu$`Percent of adults with a bachelor's degree or higher, 2013-

#Legend:
leg.txt <- c("<20%", "2-30%", "3-40%", ">40%")

colorsmatched <- nhedu$colorBuckets

#Draw map for Iowa:
map("county", "new hampshire",col = colors[colorsmatched], fill = TRUE, resolution = 0, lty = 0, projec
```

```
map("county","new hampshire", col = "white", fill = FALSE, add = TRUE, lty = 1, lwd = 0.2,projection="p
title("Percent of adults with a bachelor's degree or higher in New Hampshire, 2013-17")
legend("top", leg.txt, horiz = TRUE, fill = colors, cex = 0.7)
```



Coordinate of each county in Iowa

```
coordinate <- read.csv("~/Dropbox/School/UCLA/stat c173/Iowa Project/coordinate.csv", header = TRUE)
coordinateN <- gsub("\\s", ".",coordinate$North)
coordinateW <- gsub("\\s", ".",coordinate$West)
coordinate2 <- data_frame(County = coordinate$County,coordinateN,coordinateW)
```

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

```
coordinate2$County <-  coordinate2 %>% select(County) %>% unlist() %>% as.character()
coordinate2$coordinateN <-  coordinate2 %>% select(coordinateN) %>% unlist() %>% as.numeric()
coordinate2$coordinateW <-  coordinate2 %>% select(coordinateW) %>% unlist() %>% as.numeric()
coordinate2[71,1] <- c("O'Brien")

#Building euclidian distance matrix for all the county
coordist <- dist(coordinate2, method = "euclidean")
```

```
## Warning in dist(coordinate2, method = "euclidean"): NAs introduced by
## coercion
```

```
coordist <- as.matrix(coordist)

#First I have to combine coordinates with other datasets
coordinate2 <- coordinate2 %>% left_join(candidate, by = "County")
coordinate2 <- coordinate2 %>% left_join(income, by = "County")
coordinate2 <- data.frame(coordinate2 , Unemployment_rate_2018 = unemploy$Unemployment_rate_2018,
                          `Percent of adults with a bachelor's degree or higher, 2013-17` =  iaedu$`Per
colnames(coordinate2)[c(2,3,13)] <- c("x","y","Bachelor or above_13_17")
coordinate2 <- coordinate2[,-c(8,9,11)]
```

```
nhcoordinate <- read_excel("~/Dropbox/School/UCLA/stat c173/Iowa Project/nhcoord.xlsx")

#First I have to combine coordinates with other datasets
nhcoordinate <- nhcoordinate %>% inner_join(nhcandidate, by = "County")
nhcoordinate <- nhcoordinate %>% left_join(nhincome, by = "County")
nhcoordinate <- data.frame(nhcoordinate, Unemployment_rate_2018 = nhunemploy$Unemployment_rate_2018,
                           `Bachelor or above_13_17` = nhedu$`Percent of adults with a bachelor's degre
nhcoordinate <- nhcoordinate[,-c(5,11)]
```

BB for Iowa

```
### BB test for Buttigieg
bb_cand_Buttigieg <- 0
for(i in 1:99){
  for(j in 1:99){
    bb_cand_Buttigieg <- 1/2 * w[i,j] * newcandidate$Buttigieg[i] * newcandidate$Buttigieg[j] + bb_cand
  }
}

### BB test for Sanders
bb_cand_Sanders <- 0
for(i in 1:99){
  for(j in 1:99){
    bb_cand_Sanders <- 1/2 * w[i,j] * newcandidate$Sanders[i] * newcandidate$Sanders[j] + bb_cand_Sande
  }
}

### BB test for Warren
bb_cand_Warren <- 0
for(i in 1:99){
  for(j in 1:99){
    bb_cand_Warren <- 1/2 * w[i,j] * newcandidate$Warren[i] * newcandidate$Warren[j] + bb_cand_Warren
  }
}

### BB test for Biden
bb_cand_Biden <- 0
for(i in 1:99){
  for(j in 1:99){
    bb_cand_Biden <- 1/2 * w[i,j] * newcandidate$Biden[i] * newcandidate$Biden[j] + bb_cand_Biden
  }
}
```

```
### BB test for Median Income Estimete 2009
bb_income <- 0
for(i in 1:99){
  for(j in 1:99){
    bb_income <- 1/2 * w[i,j] * income$Estimate[i] * income$Estimate[j] + bb_income
  }
}

### BB test for Unemployment 2019
bb_unemployment <- 0
for(i in 1:99){
  for(j in 1:99){
    bb_unemployment <- 1/2 * w[i,j] * unemploy$Unemployment_rate_2018[i] * unemploy$Unemployment_rate_2
  }
}

BB <- data.frame(`Name` = c("Buttigieg", "Sanders", "Warren", "Biden","Income","Unemployment"),
                 `BB` = c(bb_cand_Buttigieg, bb_cand_Sanders, bb_cand_Warren, bb_cand_Biden, bb_income, 
kable(BB)
```

Name	BB
Buttigieg	3.460790e+01
Sanders	1.935810e+01
Warren	8.395100e+00
Biden	1.684710e+01
Income	6.287435e+11
Unemployment	1.876790e+03

BB for New Hampshire candidate Sanders

```
## BB test for Sanders
bb_nh_sanders <- 0
for(i in 1:10){
  for(j in 1:10){
    bb_nh_sanders <- 1/2 * w2[i,j] * nhcandidate$`Bernie Sanders`[i] * nhcandidate$`Bernie Sanders`[j] 

  }
}
bb_nh_sanders
```

```
## [1] 1.276085
```

BW for Iowa

```
### BW test for Buttigieg
bw_cand_Buttigieg <- 0
for(i in 1:99){
  for(j in 1:99){
    bw_cand_Buttigieg <- 1/2 * w[i,j] * (newcandidate$Buttigieg[i] - newcandidate$Buttigieg[j])^2 + bw_
  }
```

```
}

### BW test for Sanders
bw_cand_Sanders <- 0
for(i in 1:99){
  for(j in 1:99){
    bw_cand_Sanders <- 1/2 * w[i,j] * (newcandidate$Sanders[i] - newcandidate$Sanders[j])^2 + bw_cand_Sa
  }
}

### BW test for Warren
bw_cand_Warren <- 0
for(i in 1:99){
  for(j in 1:99){
    bw_cand_Warren <- 1/2 * w[i,j] * (newcandidate$Warren[i] - newcandidate$Warren[j])^2 + bw_cand_Warre
  }
}

### BW test for Biden
bw_cand_Biden <- 0
for(i in 1:99){
  for(j in 1:99){
    bw_cand_Biden <- 1/2 * w[i,j] * (newcandidate$Biden[i] - newcandidate$Biden[j])^2 + bw_cand_Biden
  }
}

### BW test for Median Income Estimate 2009
bw_income <- 0
for(i in 1:99){
  for(j in 1:99){
    bw_income <- 1/2 * w[i,j] * (income$Estimate[i] - income$Estimate[j])^2 + bw_income
  }
}

### BW test for Unemployment 2019
bw_unemployment <- 0
for(i in 1:99){
  for(j in 1:99){
    bw_unemployment <- 1/2 * w[i,j] * (unemploy$Unemployment_rate_2018[i] - newcandidate$Buttigieg[j])^1
  }
}

BW <- data.frame(`Name` = c("Buttigieg", "Sanders", "Warren", "Biden","Income","Unemployment"),
                 `BW` = c(bw_cand_Buttigieg, bw_cand_Sanders, bw_cand_Warren, bw_cand_Biden, bw_income, b
kable(BW)
```

Name	BW
Buttigieg	3.124500e+00
Sanders	3.630600e+00
Warren	3.077700e+00
Biden	3.608800e+00
Income	1.203663e+10
Unemployment	1.472363e+03

BW for New Hampshire

```
### BW test for Sanders
bw_nh_sanders <- 0
for(i in 1:10){
  for(j in 1:10){
    bw_nh_sanders <- 1/2 * w2[i,j] * (nhcandidate$`Bernie Sanders`[i] - nhcandidate$`Bernie Sanders`[j])
  }
}
bw_nh_sanders
```

```
## [1] 0.03144822
```

Moran's I(Constant Mean) for Iowa

```
s_0 <- sum(w)
n = 99
expected_i <- -1/(n-1)

### Morean's I statistic for Buttigieg
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * ((newcandidate$Buttigieg[i] - mean(newcandidate$Buttigieg)) * (newcandidate$But
      }
  a <- a + (newcandidate$Buttigieg[i])^2
}
Buttigieg_i <- n/s_0*b/a

### Morean's I statistic for Sanders
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * ((newcandidate$Sanders[i] - mean(newcandidate$Sanders)) * (newcandidate$Sanders
      }
  a <- a + (newcandidate$Sanders[i])^2
}
Sanders_i <- n/s_0*b/a

### Morean's I statistic for Warren
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * ((newcandidate$Warren[i] - mean(newcandidate$Warren)) * (newcandidate$Warren[j]
      }
  a <- a + (newcandidate$Warren[i])^2
}
Warren_i <- n/s_0*b/a

### Morean's I statistic for Biden
```

```
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * ((newcandidate$Biden[i] - mean(newcandidate$Biden)) * (newcandidate$Biden[j] - m
      }
  a <- a + (newcandidate$Biden[i])^2
}
Biden_i <- n/s_0*b/a

### Morean's I statistic for Median Income
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * ((income$Estimate[i] - mean(income$Estimate)) * (income$Estimate[j] - mean(incom
      }
  a <- a + (income$Estimate[i])^2
}
Income_i <- n/s_0*b/a

### Morean's I statistic for Unemployment
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * ((unemploy$Unemployment_rate_2018[i] - mean(unemploy$Unemployment_rate_2018)) *
      }
  a <- a + (unemploy$Unemployment_rate_2018[i])^2
}
Unemployment_i <- n/s_0*b/a

ti <- data.frame(`Name` = c("Expected", "Buttigieg", "Sanders", "Warren", "Biden","Income","Unemployment
                 `Moran I` = c(expected_i, Buttigieg_i, Sanders_i, Warren_i, Biden_i, Income_i, Unemploym
kable(ti)
```

Name	Moran.I
Expected	-0.0102041
Buttigieg	0.0104398
Sanders	0.0107400
Warren	0.0204024
Biden	0.0168096
Income	0.0060527
Unemployment	0.0027595

- Estimete Moran's I is -0.01
- Most of the value in this data has similar value to espected one, so there is a clustering in data.

Moran's I(Non Constant Mean) for Iowa

```
n = 99
s_0 = sum(w)
```

```
colnames(coordinate2)
iasand <- lm(Sanders ~ Estimate + Unemployment_rate_2018 + `Bachelor or above_13_17`, data = coordinate2
iaresid <- matrix(iasand$residuals)

e_i <- -1/(n-1)
i <- (n/s_0)*(t(iaresid) %*% w %*% iaresid)/(t(iaresid) %*% iaresid)
ti <- data.frame(`Name` = c("Expected", "Sanders"),
                 `Moran I` = c(e_i, i))
kable(ti)
```

- Moran's I statistic for sanders from the linear model has lower value than the expected I.
- We can assume that theere is a negative spatial autocorrelation in the model.

Moran's I(Constant Mean) for New Hampshire

```
s_0 <- sum(w2)
n = 10
expected_i <- -1/(n-1)

### Morean's I statistic for Sanders
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * ((nhcandidate$`Bernie Sanders`[i] - mean(nhcandidate$`Bernie Sanders`)) * (nhcan
      }
  a <- a + (nhcandidate$`Bernie Sanders`[i])^2
}
Sanders_i <- n/s_0*b/a

i <- data.frame(`Name` = c("Expected", "Sanders"),
                `Moran I` = c(expected_i, Sanders_i))
kable(i)
```

Name	Moran.I
Expected	-0.1111111
Sanders	0.0001374

- By constant mean, Moran's I statistic for Sanders has positive spatial autocorrelation.

Moran's I(Non Constant Mean) for Iowa

```
n = 10
s_0 = sum(w2)
nhsand <- lm(Bernie.Sanders ~ Median.household.income + Unemployment_rate_2018 + `Bachelor.or.above_13_1
nhresid <- matrix(nhsand$residuals)

e_i <- -1/(n-1)
i <- (n/s_0)*(t(nhresid) %*% w2 %*% nhresid)/(t(nhresid) %*% nhresid)
ti <- data.frame(`Name` = c("Expected", "Sanders"),
                 `Moran I` = c(e_i, i))
```

```
kable(ti)
```

- Moran's I statistic for sanders from the linear model has lower value than the expected I.
- We can assume that theere is a negative spatial autocorrelation in the model.

Geary's C for Iowa

```
### Geary's C statistic for Candidate Buttigieg
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * (newcandidate$Buttigieg[i] - newcandidate$Buttigieg[j])^2
      }
  a <- a + (newcandidate$Buttigieg[i] - mean(newcandidate$Buttigieg))^2
}
Buttigieg_c <- (n-1)/(2*s_0)*b/a

### Geary's C statistic for Candidate Sanders
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * (newcandidate$Sanders[i] - newcandidate$Sanders[j])^2
      }
  a <- a + (newcandidate$Sanders[i] - mean(newcandidate$Sanders))^2
}
Sanders_c <- (n-1)/(2*s_0)*b/a

### Geary's C statistic for Candidate Warren
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * (newcandidate$Warren[i] - newcandidate$Warren[j])^2
      }
  a <- a + (newcandidate$Warren[i] - mean(newcandidate$Warren))^2
}
Warren_c <- (n-1)/(2*s_0)*b/a

### Geary's C statistic for Candidate Biden
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * (newcandidate$Biden[i] - newcandidate$Biden[j])^2
      }
  a <- a + (newcandidate$Biden[i] - mean(newcandidate$Biden))^2
}
Biden_c <- (n-1)/(2*s_0)*b/a

### Geary's C statistic for Median Income Estimate 2009
b = 0
```

```
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * (income$Estimate[i] - income$Estimate[j])^2
      }
  a <- a + (income$Estimate[i] - mean(income$Estimate))^2
}
Income_c <- (n-1)/(2*s_0)*b/a

### Geary's C statistic for Unemployment 2019
b = 0
a = 0
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * (unemploy$Unemployment_rate_2018[i] - unemploy$Unemployment_rate_2018[j])^2
      }
  a <- a + (unemploy$Unemployment_rate_2018[i] - mean(unemploy$Unemployment_rate_2018))^2
}
Unemployment_c <- (n-1)/(2*s_0)*b/a

c <- data.frame(`Name` = c("Expected", "Buttigieg", "Sanders", "Warren", "Biden","Income","Unemployment
                `Geary C` = c(1, Buttigieg_c, Sanders_c, Warren_c, Biden_c, Income_c, Unemployment_c))
kable(c)
```

Name	Geary.C
Expected	1.0000000
Buttigieg	0.1479051
Sanders	0.5677339
Warren	0.2752743
Biden	0.3622504
Income	0.2677096
Unemployment	0.1615112

- For all the data, Geary's C values are all less than one, so there is a clustering.

Geary's C for New Hampshire

```
### Geary's C statistic for Sanders
b = 0
a = 0
n = 10
for(i in 1:n){
    for(j in 1:n){
      b <- b + w[i,j] * (nhcoordinate$`Bernie.Sanders`[i] - nhcoordinate$`Bernie.Sanders`[j])^2
      }
  a <- a + (nhcoordinate$`Bernie.Sanders`[i] - mean(nhcoordinate$`Bernie.Sanders`))^2
}
nh_Sanders_c <- (n-1)/(2*s_0)*b/a

c <- data.frame(`Name` = c("Expected", "Sanders"),
                `Geary C` = c(1, nh_Sanders_c))
kable(c)
```

Name	Geary.C
Expected	1.0000000
Sanders	0.1859274

- Geary's C values are all less than one, so there is a clustering.

Variogram

```
#See the variogram of percentage of vote for candidate Buttigieg
data <- coordinate2 %>% select(x,y,Buttigieg)
gdata <- as.geodata(data)
variogram <- variog(gdata, max.dist = 2)
```

```
## variog: computing omnidirectional variogram
```

```
plot(variogram)
lines.variomodel(cov.model="exp", cov.pars=c(200,0.5), nug=0, max.dist=2, lty=2)

#Fit the spherical variogram using the default option (check ?variofit manual).
fit1 <- variofit(variogram, cov.model="exp", ini.cov.pars=c(200,0.5),
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: npairs
## variofit: minimisation function used: optim
```

```
lines(fit1, lty=1)

#Use Cressies weights:
fit2 <- variofit(variogram, cov.model="exp", weights="cressie", ini.cov.pars=c(200,0.5),
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: cressie
## variofit: minimisation function used: optim
```

```
lines(fit2, lty=1, col="green")

#Use equal weights (simply OLS):
fit3 <- variofit(variogram, cov.model="exp", ini.cov.pars=c(200,0.5), weights="equal",
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: equal
## variofit: minimisation function used: optim
```

```
lines(fit3, lty=1, col="orange")

#MML:
ml <- likfit(gdata, cov.model="exp", ini.cov.pars=c(200,0.5),
       fix.nugget=FALSE, nugget=0)
```

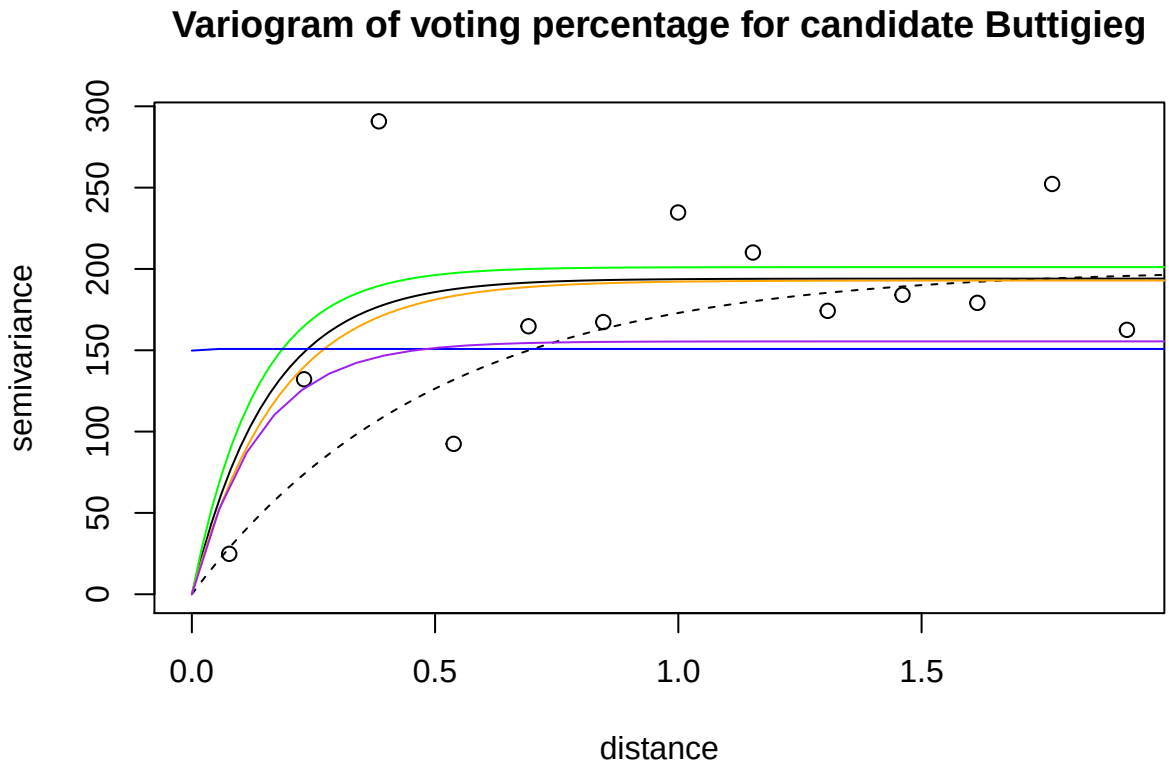
```
## kappa not used for the exponential correlation function
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
```

```
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
lines(ml, col="blue")

#REML:
rml <- likfit(gdata, cov.model="exp", ini.cov.pars=c(200,0.5),
       fix.nugget=FALSE, nugget=0, lik.method = "RML" )
```

```
## kappa not used for the exponential correlation function
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
lines(rml, col="purple")
title(main = "Variogram of voting percentage for candidate Buttigieg")
```




```
#See the variogram of percentage of vote for candidate Sanders
data <- coordinate2 %>% select(x,y,Sanders)
gdata <- as.geodata(data)
variogram <- variog(gdata, max.dist = 2)
```

```
## variog: computing omnidirectional variogram
```

```
plot(variogram)
lines.variomodel(cov.model="exp", cov.pars=c(250,0.35), nug=0, max.dist=2, lty=2)

#Fit the spherical variogram using the default option (check ?variofit manual).
fit1 <- variofit(variogram, cov.model="exp", ini.cov.pars=c(250,0.5),
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: npairs
## variofit: minimisation function used: optim
```

```
lines(fit1, lty=1)

#Use Cressies weights:
fit2 <- variofit(variogram, cov.model="exp", weights="cressie", ini.cov.pars=c(250,0.5),
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: cressie
## variofit: minimisation function used: optim
```

```
lines(fit2, lty=1, col="green")

#Use equal weights (simply OLS):
fit3 <- variofit(variogram, cov.model="exp", ini.cov.pars=c(250,0.5), weights="equal",
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: equal
## variofit: minimisation function used: optim
```

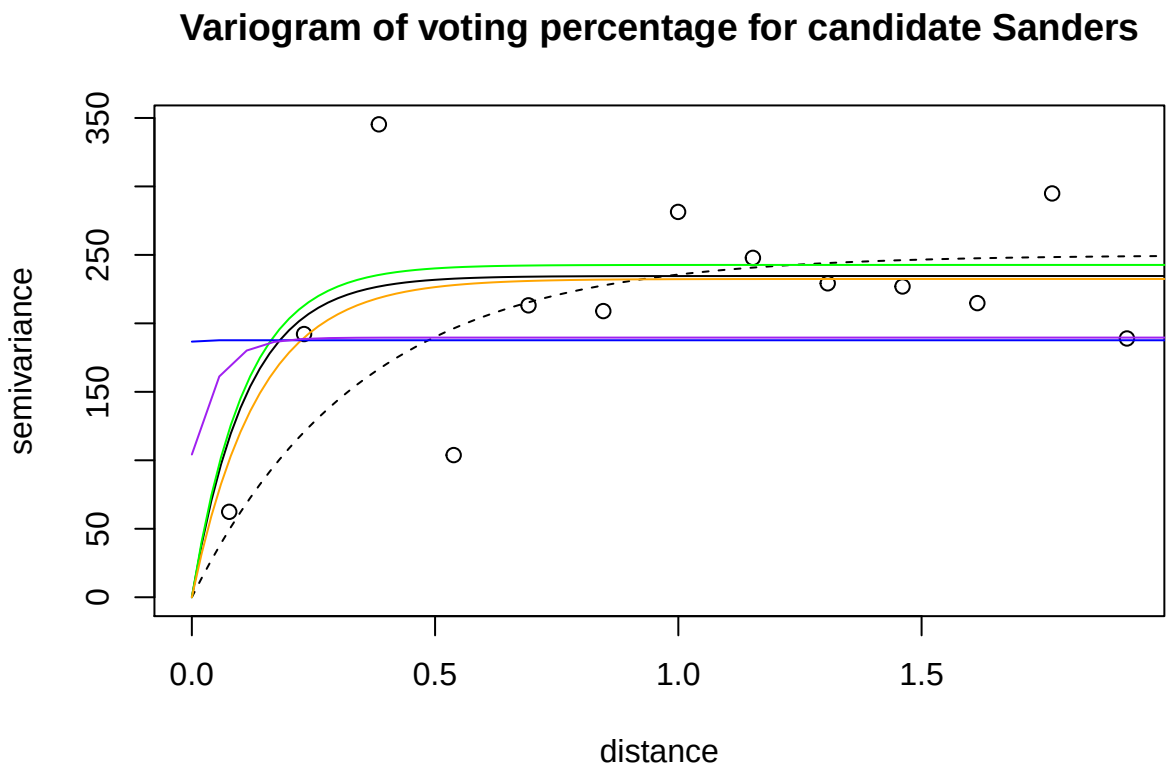
```
lines(fit3, lty=1, col="orange")

#MML:
ml <- likfit(gdata, cov.model="exp", ini.cov.pars=c(250,0.5),
       fix.nugget=FALSE, nugget=0)
```

```
## kappa not used for the exponential correlation function
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
lines(ml, col="blue")

#REML:
rml <- likfit(gdata, cov.model="exp", ini.cov.pars=c(250,0.5),
       fix.nugget=FALSE, nugget=0, lik.method = "RML" )
```

```
## kappa not used for the exponential correlation function
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
lines(rml, col="purple")
title(main = "Variogram of voting percentage for candidate Sanders")
```



Variogram

```
#See the variogram of percentage of vote for candidate Sanders
data <- nhcoordinate %>% select(West, North,`Bernie.Sanders`)
gdata <- as.geodata(data)
variogram <- variog(gdata, max.dist = 1.5)
```

```
## variog: computing omnidirectional variogram
```

```
plot(variogram)
lines.variomodel(cov.model="exp", cov.pars=c(0.001,0.35), nug=0, max.dist=2, lty=2)

#Fit the spherical variogram using the default option (check ?variofit manual).
fit1 <- variofit(variogram, cov.model="exp", ini.cov.pars=c(0.001,0.35),
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: npairs
## variofit: minimisation function used: optim
```

```
lines(fit1, lty=1)

#Use equal weights (simply OLS):
fit3 <- variofit(variogram, cov.model="exp", ini.cov.pars=c(0.001,0.35), weights="equal",
       fix.nugget=FALSE, nugget=0)
```

```
## variofit: covariance model used is exponential
## variofit: weights used: equal
## variofit: minimisation function used: optim
```

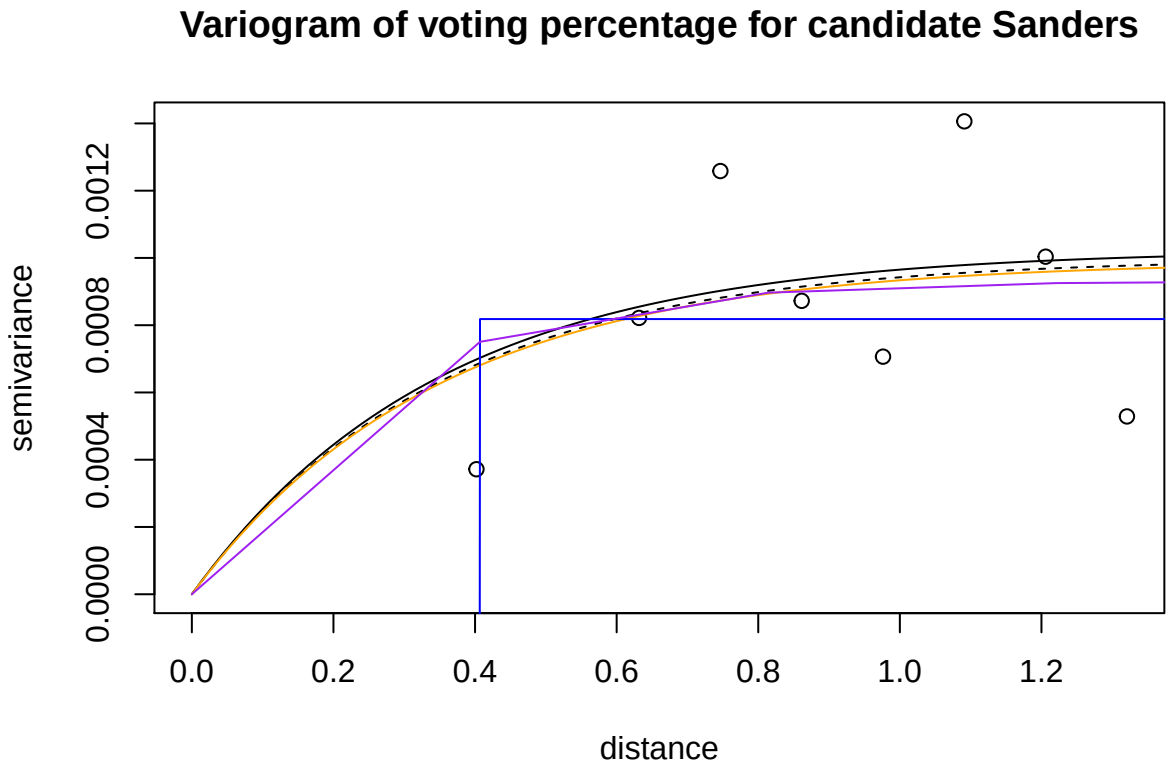
```
lines(fit3, lty=1, col="orange")

#MML:
ml <- likfit(gdata, cov.model="exp", ini.cov.pars=c(0.001,0.35),
       fix.nugget=FALSE, nugget=0)
```

```
## kappa not used for the exponential correlation function
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
lines(ml, col="blue")

#REML:
rml <- likfit(gdata, cov.model="exp", ini.cov.pars=c(0.001,0.35),
       fix.nugget=FALSE, nugget=0, lik.method = "RML" )
```

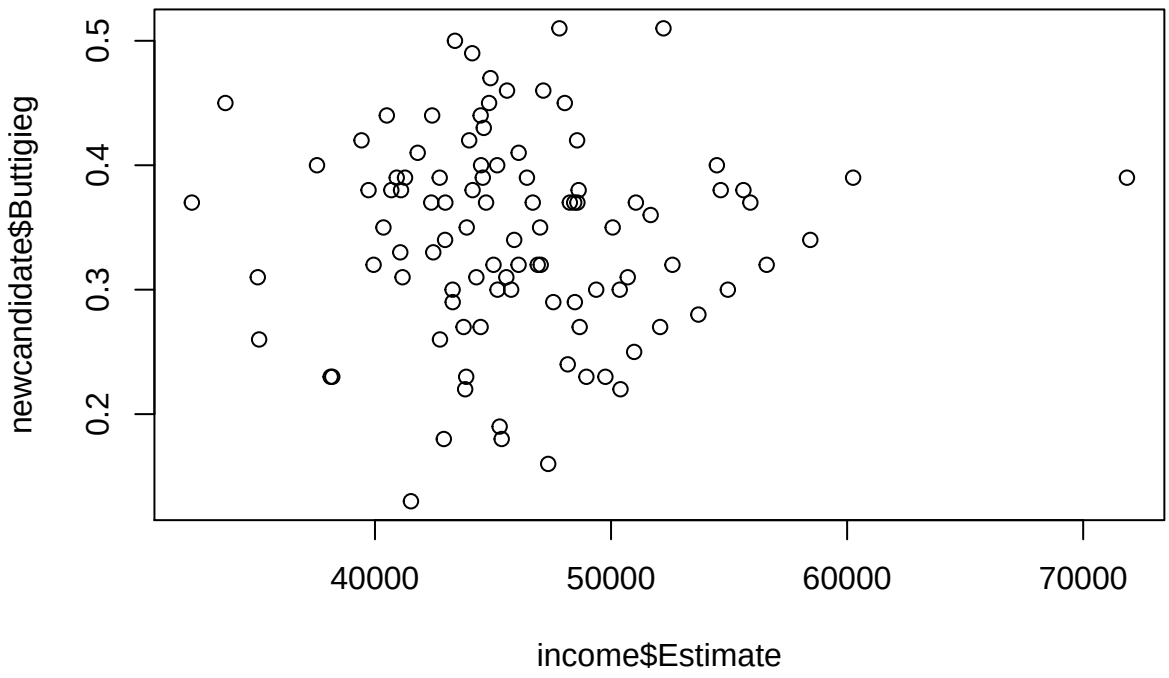
```
## kappa not used for the exponential correlation function
## ---------------------------------------------------------------
## likfit: likelihood maximisation using the function optim.
## likfit: Use control() to pass additional
##          arguments for the maximisation function.
##         For further details see documentation for optim.
## likfit: It is highly advisable to run this function several
##         times with different initial values for the parameters.
## likfit: WARNING: This step can be time demanding!
## ---------------------------------------------------------------
## likfit: end of numerical maximisation.
```

```
lines(rml, col="purple")
title(main = "Variogram of voting percentage for candidate Sanders")
```

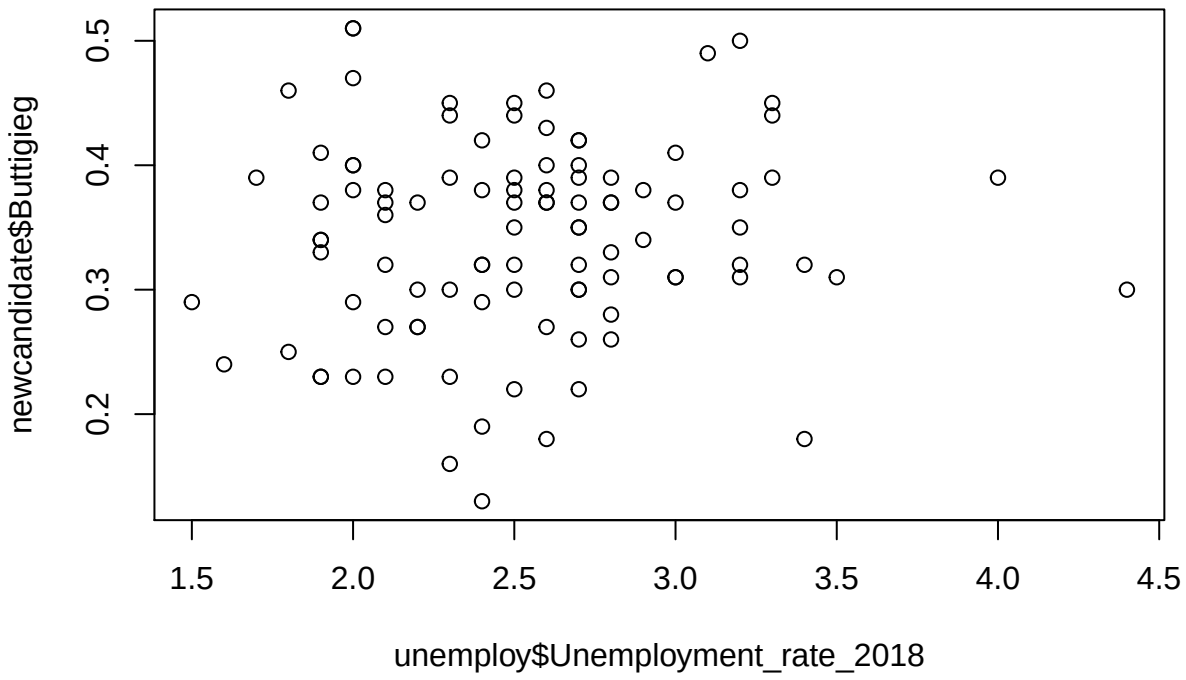


Plotting

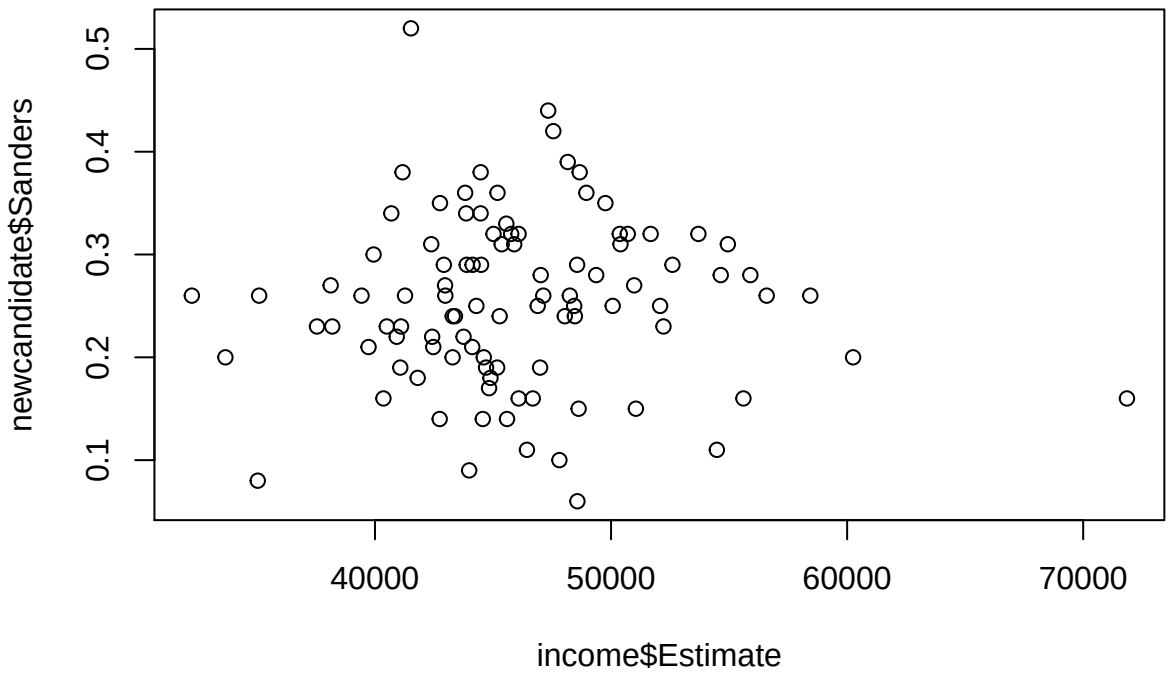
```
## Plot Buttigieg against median income and unemployment
plot(income$Estimate,newcandidate$Buttigieg)
```




```
plot(unemploy$Unemployment_rate_2018,newcandidate$Buttigieg)
```

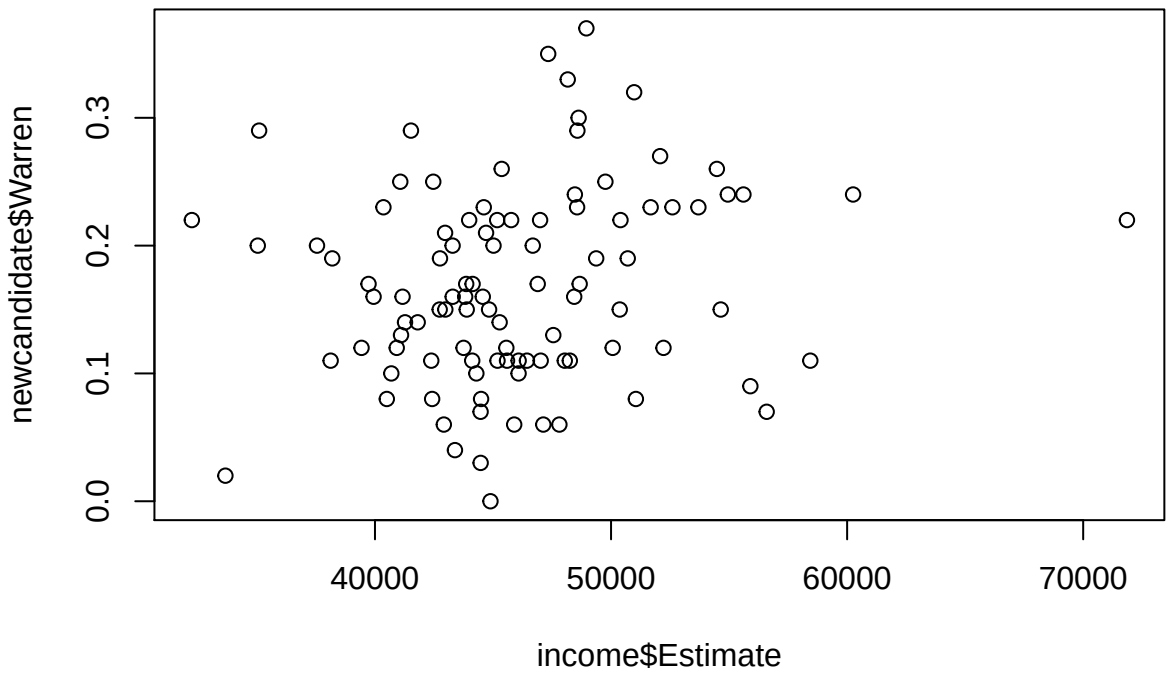



```
## Plot Sanders against median income and unemployment
plot(income$Estimate,newcandidate$Sanders)
```



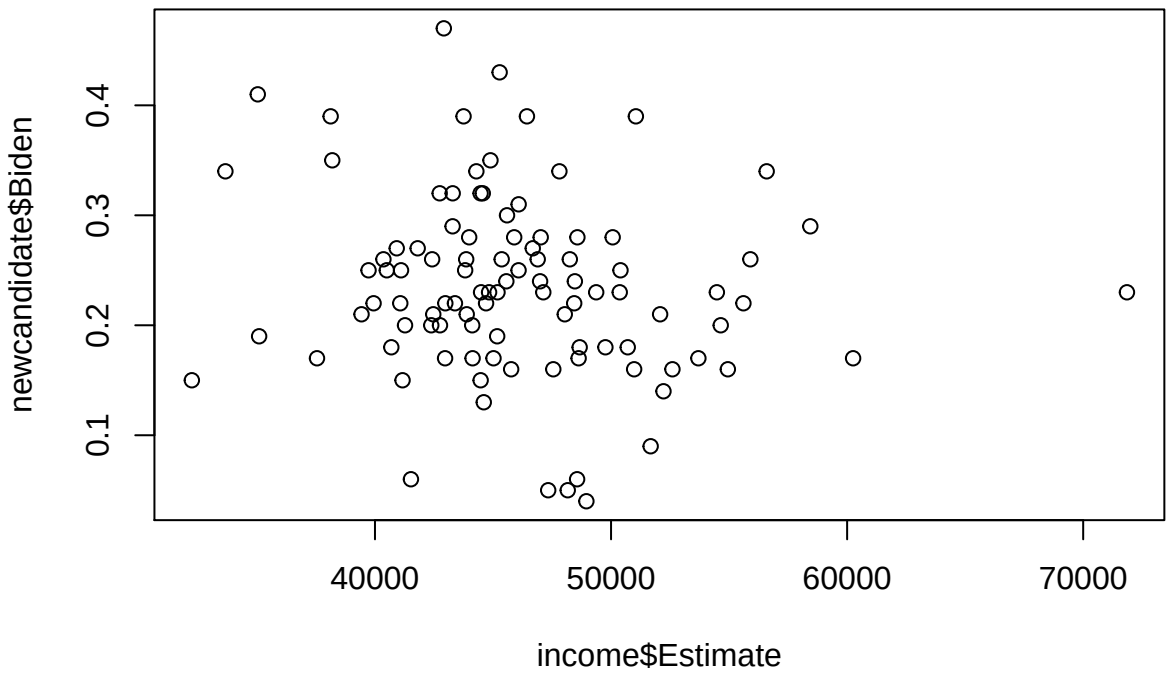

```
plot(unemploy$Unemployment_rate_2018,newcandidate$Sanders)
```


```
## Plot Warren against median income and unemployment
plot(income$Estimate,newcandidate$Warren)
```




```
plot(unemploy$Unemployment_rate_2018,newcandidate$Warren)
```


```
## Plot Biden against median income and unemployment
plot(income$Estimate,newcandidate$Biden)
```




```
plot(unemploy$Unemployment_rate_2018,newcandidate$Biden)
```


newcandidate$Biden
0.1
0.2
0.3
0.4
1.5
2.0
2.5
3.0
3.5
4.0
4.5
unemploy$Unemployment_rate_2018