Final Report

due November 16, 2021 by 11:59 PM

Enzo, Layla, Madeleine

November 16, 2021

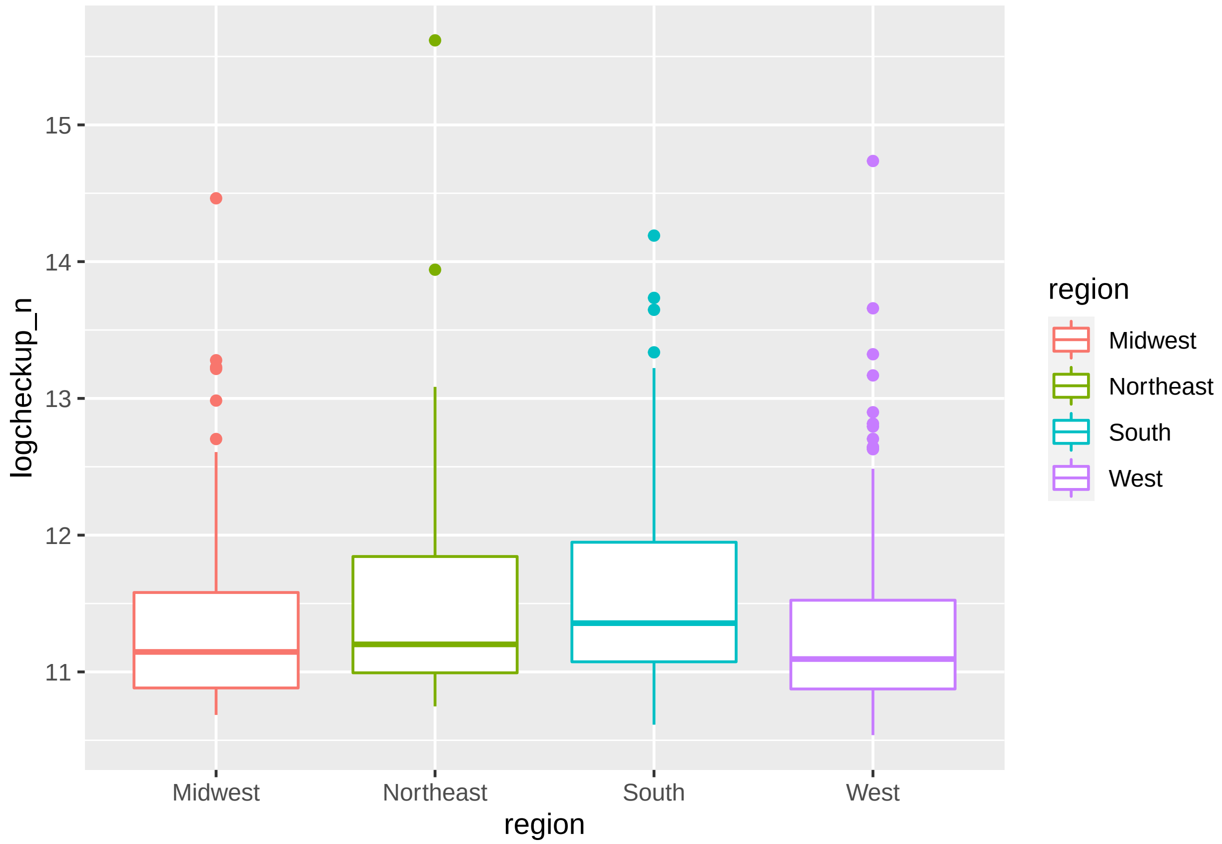
```
cities <- initial_data %>%
  select(StateAbbr, PlaceName, Population2010, PlaceFIPS, Geolocation,
         ACCESS2_AdjPrev,CANCER_AdjPrev, CHD_AdjPrev, CHECKUP_AdjPrev,
         COPD_AdjPrev, COLON_SCREEN_AdjPrev,COREM_AdjPrev, COREW_AdjPrev,
         KIDNEY_AdjPrev, MAMMOUSE_AdjPrev, PAPTEST_AdjPrev) %>%
  dplyr::rename(state = StateAbbr, city = PlaceName, population = Population2010,
         health_access = ACCESS2_AdjPrev, cancer = CANCER_AdjPrev,
         heart_disease = CHD_AdjPrev, checkup = CHECKUP_AdjPrev,
         chronic_lung_disease = COPD_AdjPrev,
         colon_screen = COLON_SCREEN_AdjPrev,
         men_colorectal_cancer_screen = COREM_AdjPrev,
         women_colorectal_cancer_screen = COREW_AdjPrev,
         chronic_kidney_disease = KIDNEY_AdjPrev,
         mammogram = MAMMOUSE_AdjPrev,
         pap_test = PAPTEST_AdjPrev) %>%
  mutate(large_metro = (population >= 1500000),
         metro = (population >= 500000 & population < 1500000),
         med_urban = (population >= 200000 & population < 500000),
         small_urban = (population >= 50000 & population < 200000)) %>%
  mutate(west = (state %in% c("WA", "OR", "ID", "MT", "WY", "CA", "NV", "UT",
                              "CO", "AZ", "NM")),
         midwest = (state %in% c("ND", "SD", "NE", "KS", "MN", "IA", "MO",
                                 "WI", "IL", "IN", "MI", "OH")),
         northeast = (state %in% c("PA", "NY", "VT", "NH", "MA", "CT", "RI",
                                   "NJ", "ME", "DC")),
         south = (state %in% c("OK", "TX", "AR", "LA", "MS", "AL", "TN", "KY",
                               "WV", "VA", "MD", "DE", "NC", "SC", "GA",
                               "FL"))) %>%
  mutate(region = ifelse(west == TRUE, "West",
                         ifelse(midwest == TRUE, "Midwest",
                                ifelse(northeast == TRUE, "Northeast",
                                       ifelse(south == TRUE,
                                              "South", NA))))) %>%
  mutate(city_size = ifelse(large_metro == TRUE, "Large Metropolitan",
                         ifelse(metro == TRUE, "Metropolitan",
                                ifelse(med_urban == TRUE, "Medium-Size Urban",
                                       ifelse(small_urban == TRUE,
                                              "Small-Size Urban", NA))))) %>%
  na.omit(city_size) %>%
  na.omit(region) %>%
  mutate(checkup_n = (population*(checkup/100)),
```

```
         colon_n_screened = (population*(colon_screen/100)),
         m_colorectal_n_screened = (population*(men_colorectal_cancer_screen/100)),
         w_colorectal_n_screened =
           (population*(women_colorectal_cancer_screen/100)),
         mammogram_n_screened = (population*(mammogram/100)),
         pap_n_screened = (population*(pap_test/100))) %>%
  mutate(no_checkup_n = (population - checkup_n),
         no_colon_n_screened = (population - colon_n_screened),
         no_m_colorectal_n_screened = (population - m_colorectal_n_screened),
         no_w_colorectal_n_screened =
           (population - w_colorectal_n_screened),
         no_mammogram_n_screened = (population - mammogram_n_screened),
         no_pap_n_screened = (population - pap_n_screened))

# checked the residual plot withouth this and need todo this to make model more accurate
#this is the training data set called model_cities
set.seed(100)

split <- initial_split(cities, prop = 3/4, strata = region)

model_cities <- training(split)
cities_test <- testing(split)

#for introductory visualizations
west_cities <- cities %>%
  filter(west)

midwest_cities <- cities %>%
  filter(midwest)

northeast_cities <- cities %>%
  filter(northeast)

south_cities <- cities %>%
  filter(south)
```

#boxplot of region and city size

Introductory Visualisations of Data Set

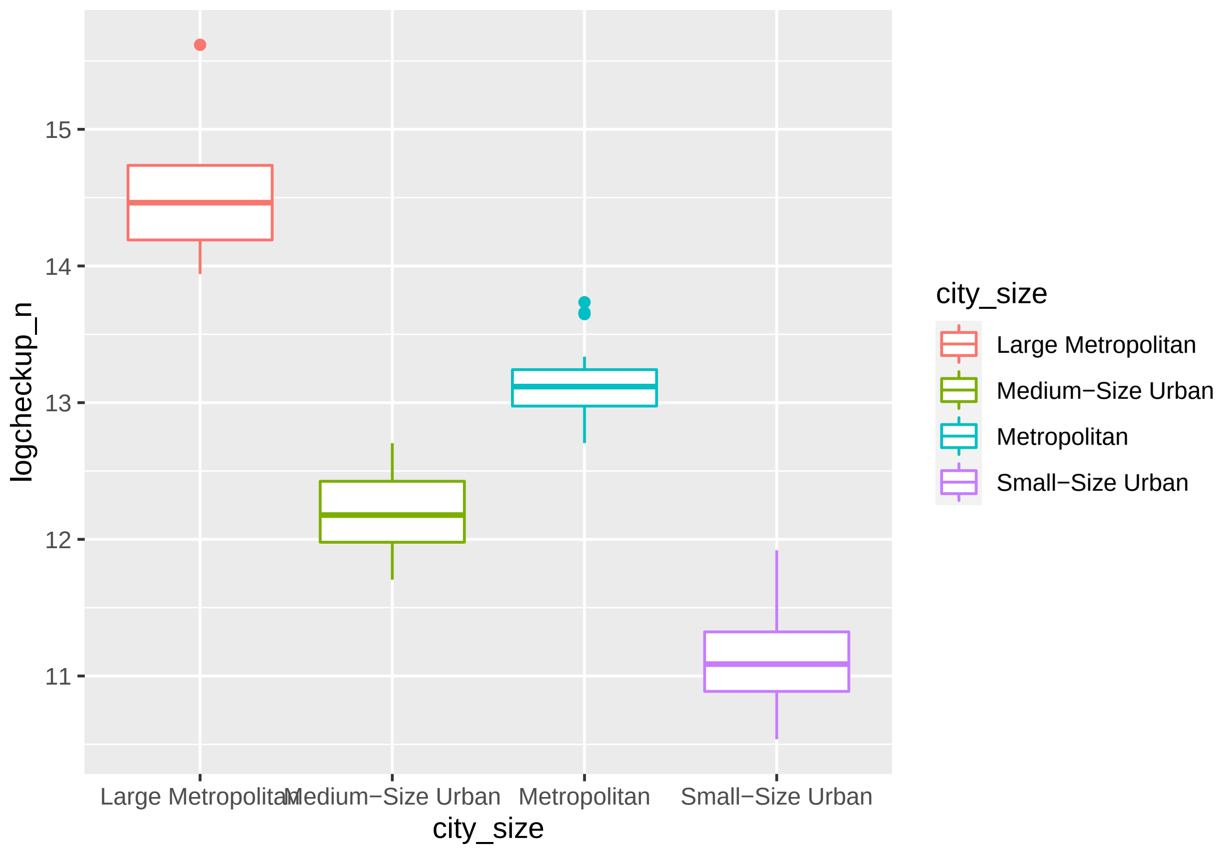
```
cities2 <- cities %>%
  filter( region != "NA") %>%
  mutate(logcheckup_n = log(checkup_n))

#boxplots of checkup (n and log) by region
ggplot(data = cities2,
       aes(x = region, y = logcheckup_n,
           color = region)) +
  geom_boxplot()
```




```
#anova shows means of checkup (n and log) are different among the regions
anova1 <- aov(logcheckup_n ~ region, data = cities2)
tidy(anova1)
```

```
## # A tibble: 2 x 6
##   term          df  sumsq meansq statistic  p.value
##   <chr>      <dbl>  <dbl>  <dbl>     <dbl>    <dbl>
## 1 region         3   6.32  2.11       4.16  0.00638
## 2 Residuals    447 226.    0.507     NA    NA
```

```
#boxplots of checkup (n and log) by city size
ggplot(data = cities2,
       aes(x = city_size, y = logcheckup_n,
           color = city_size)) +
  geom_boxplot()
```



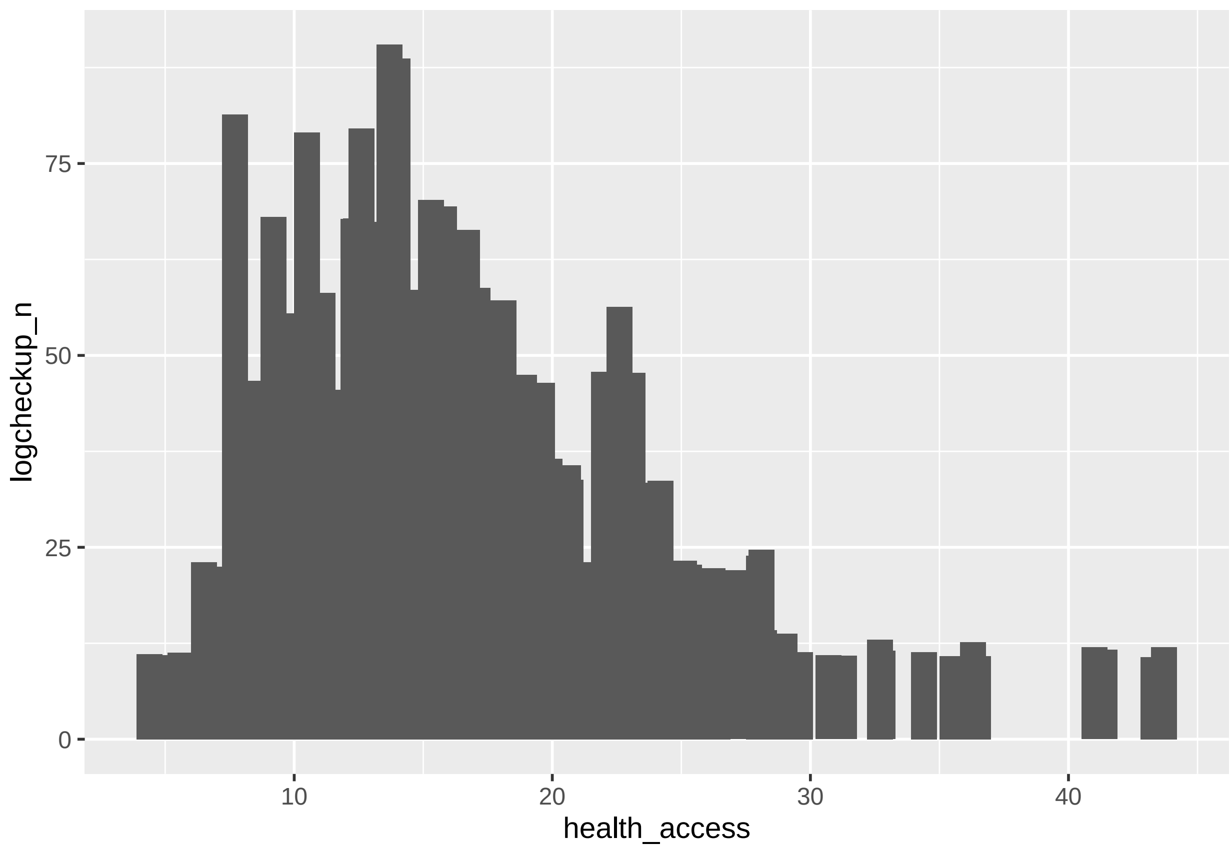

```
#anova shows means of checkup (n and log) are different among the city sizes
anova2 <- aov(logcheckup_n ~ city_size, data = cities2)
tidy(anova2)
```

```
## # A tibble: 2 x 6
##   term         df sumsq  meansq statistic    p.value
##   <chr>     <dbl> <dbl>   <dbl>     <dbl>      <dbl>
## 1 city_size     3 191.   63.5        671.  4.15e-165
## 2 Residuals   447  42.3  0.0946       NA   NA
```

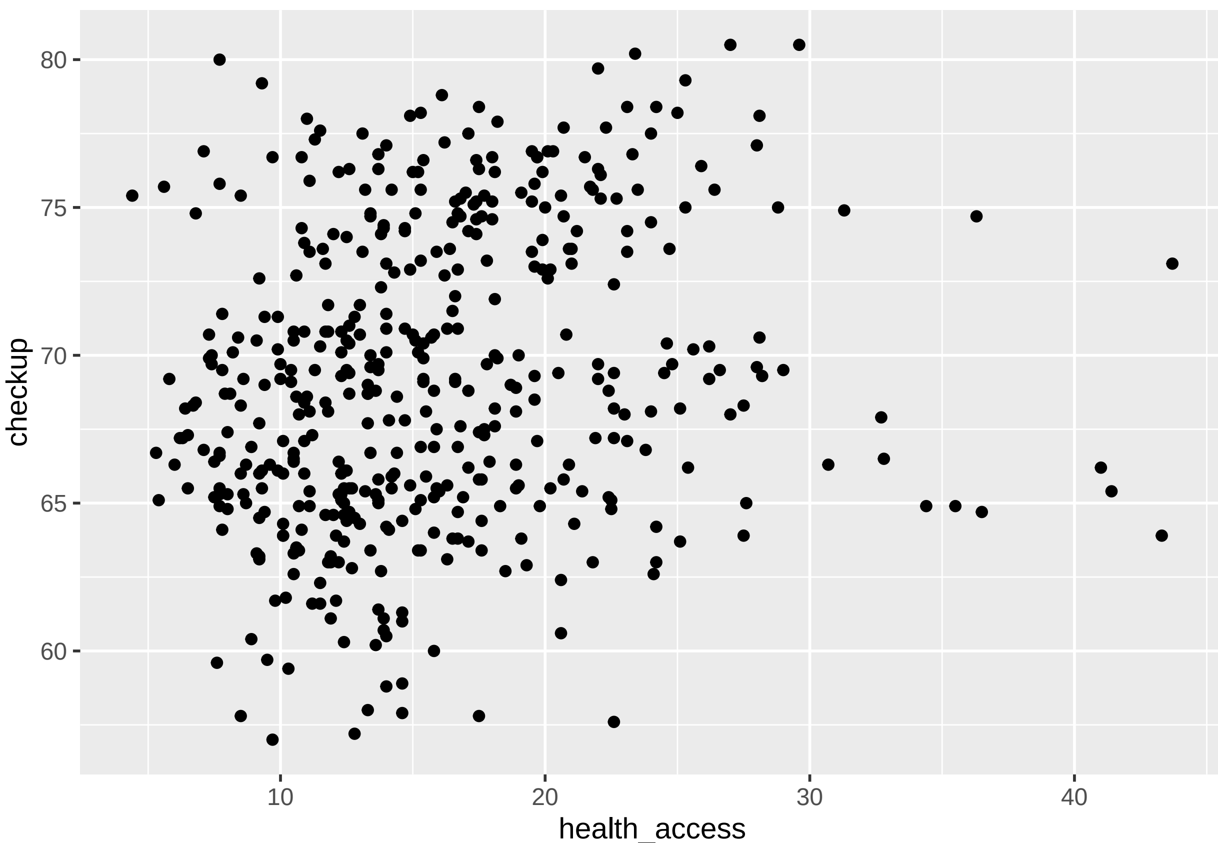
```
#enzo's graph for health access vs checkup
ggplot(data = cities2,
       aes(x = health_access, y = logcheckup_n)) +
  geom_col(aes(width = 1))
```

```
## Warning: Ignoring unknown aesthetics: width
```

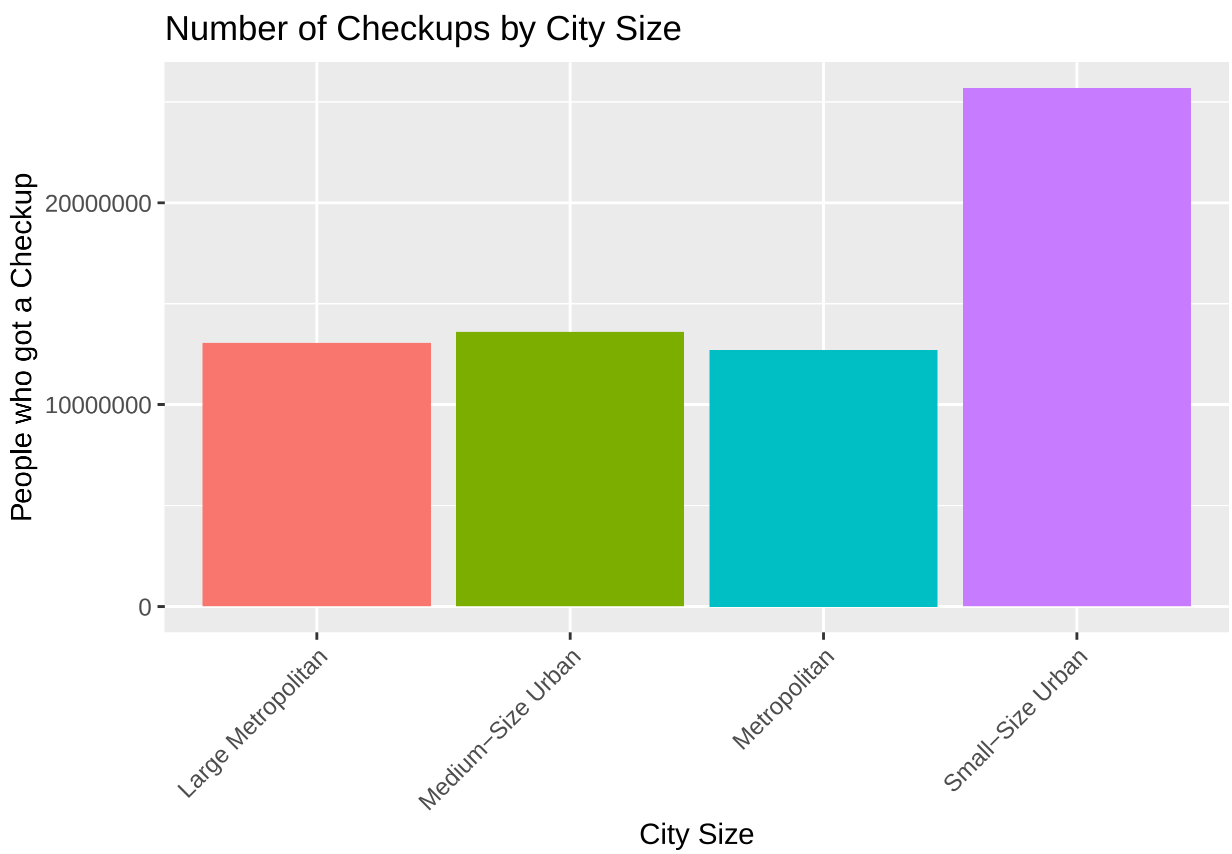
```
## Warning: position_stack requires non-overlapping x intervals
```

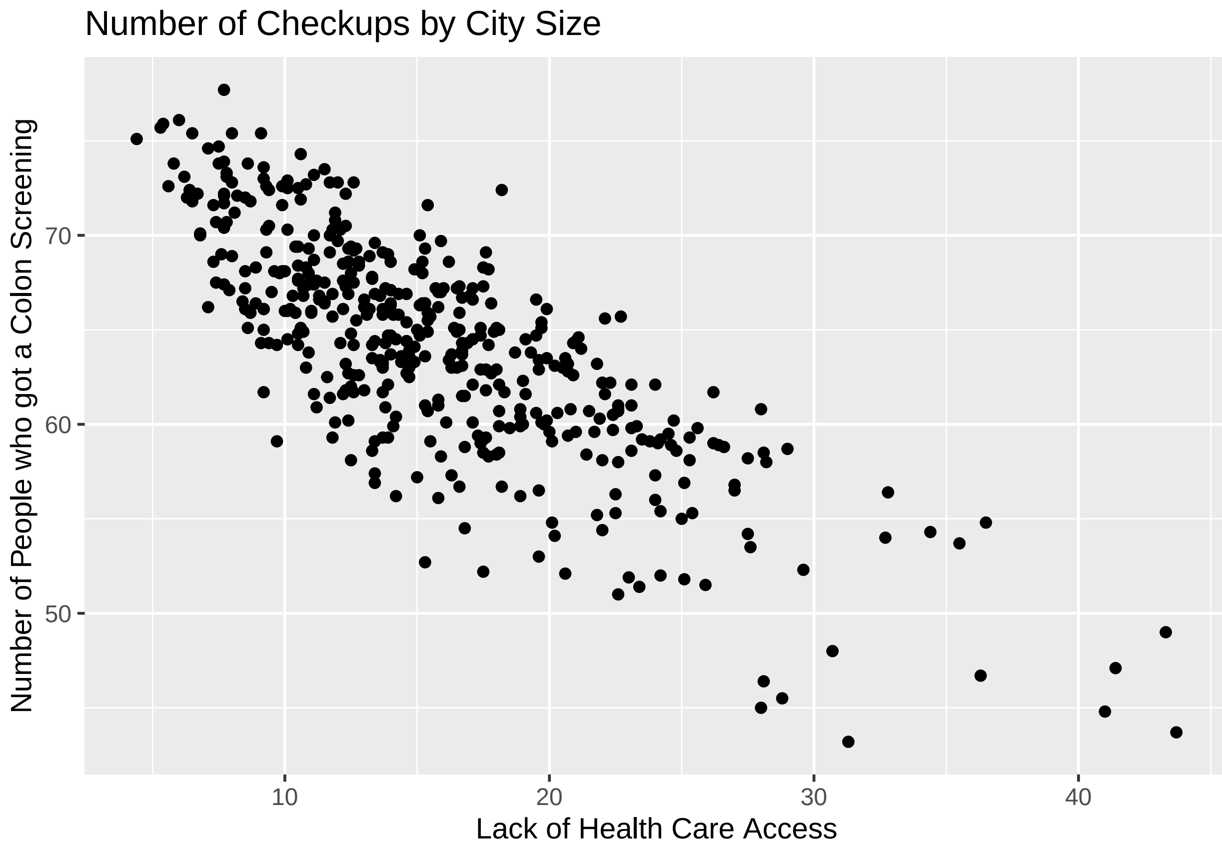


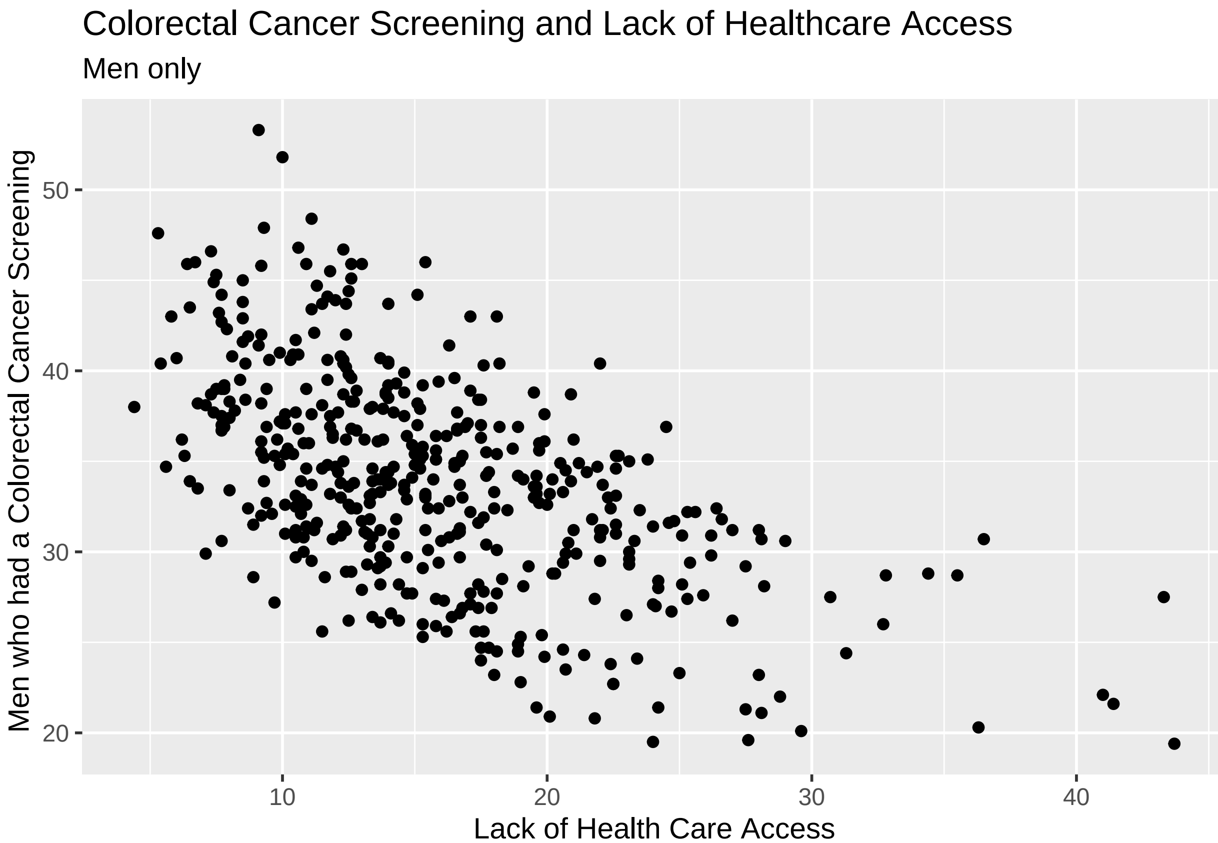

```r
#madeleine's graph for health access vs checkup
ggplot(cities, aes(x = health_access,
                   y= checkup))+
  geom_point()
```

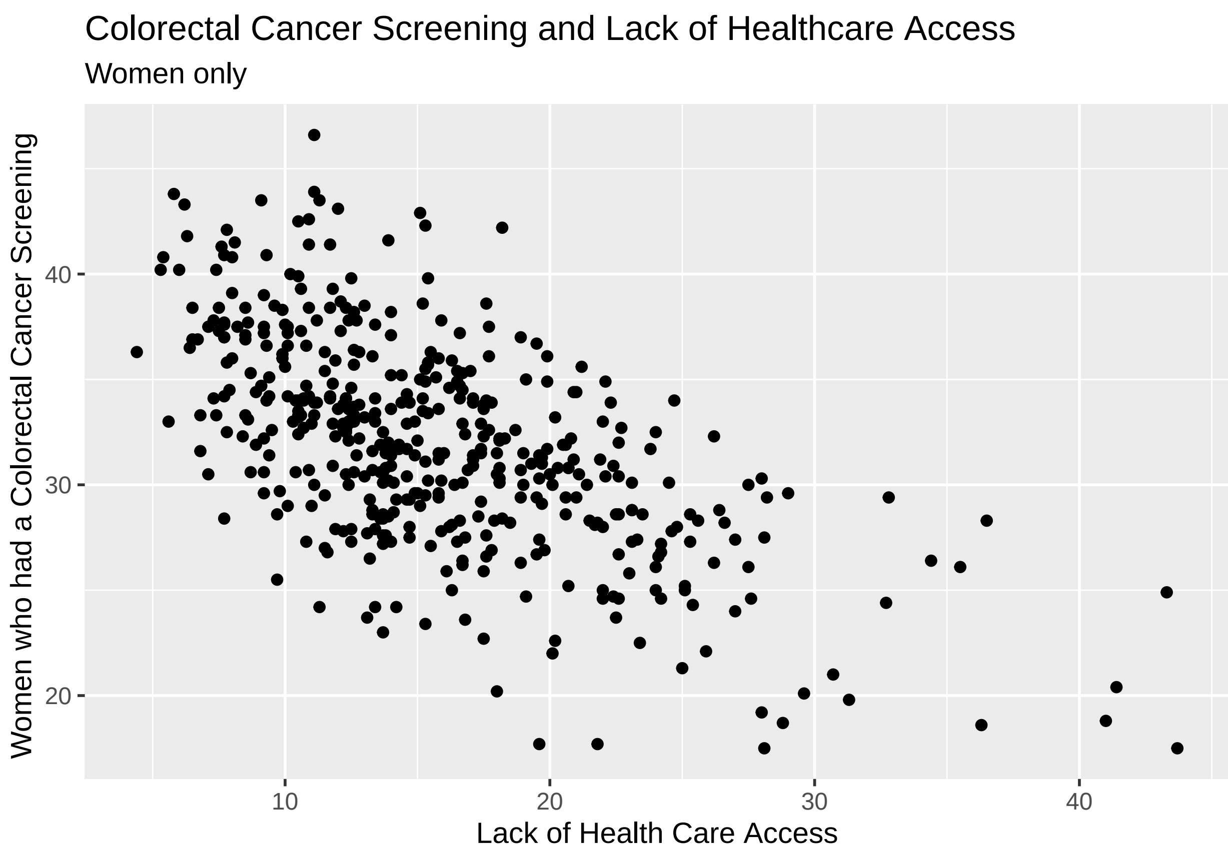


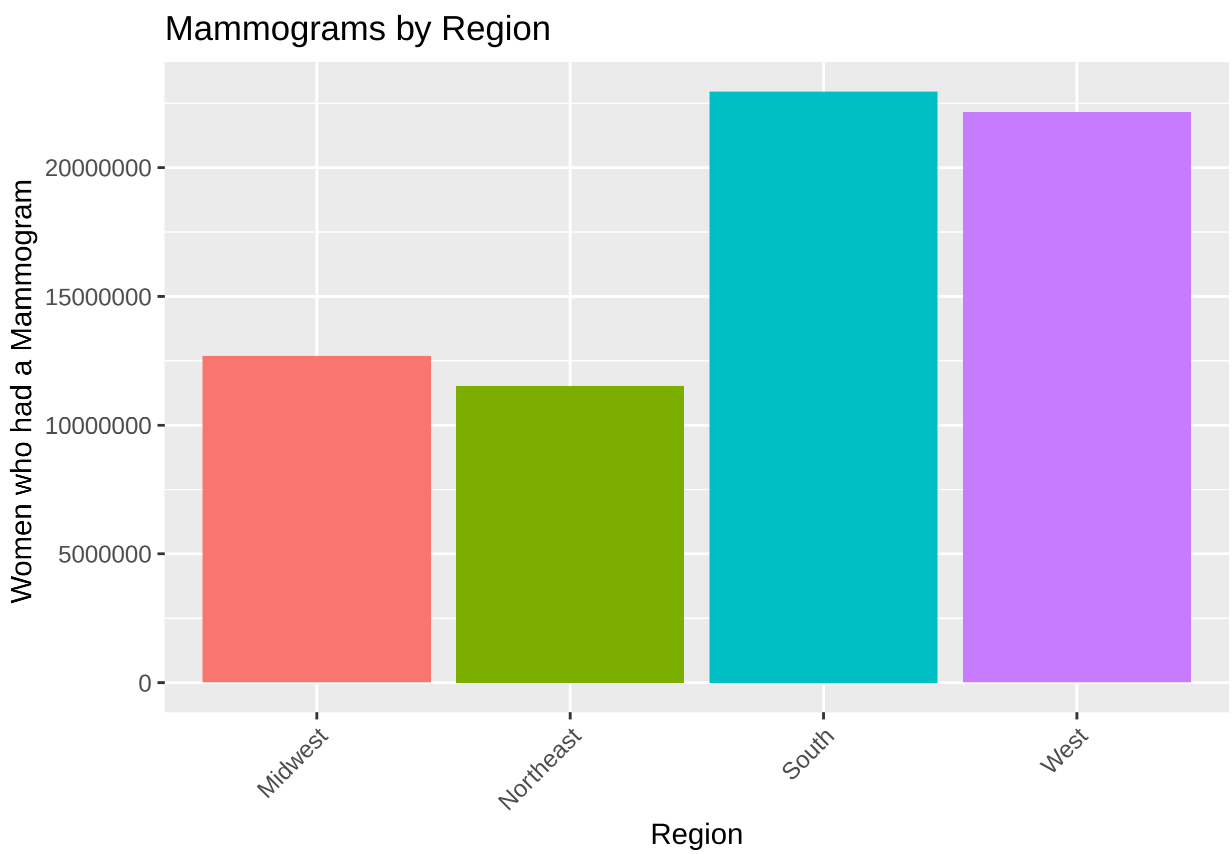
80
75
70
65
60
checkup
10
20
30
40
health_access


Number of Checkups by City Size
People who got a Checkup
20000000
10000000
0
Large Metropolitan
Medium–Size Urban
Metropolitan
Small–Size Urban
City Size

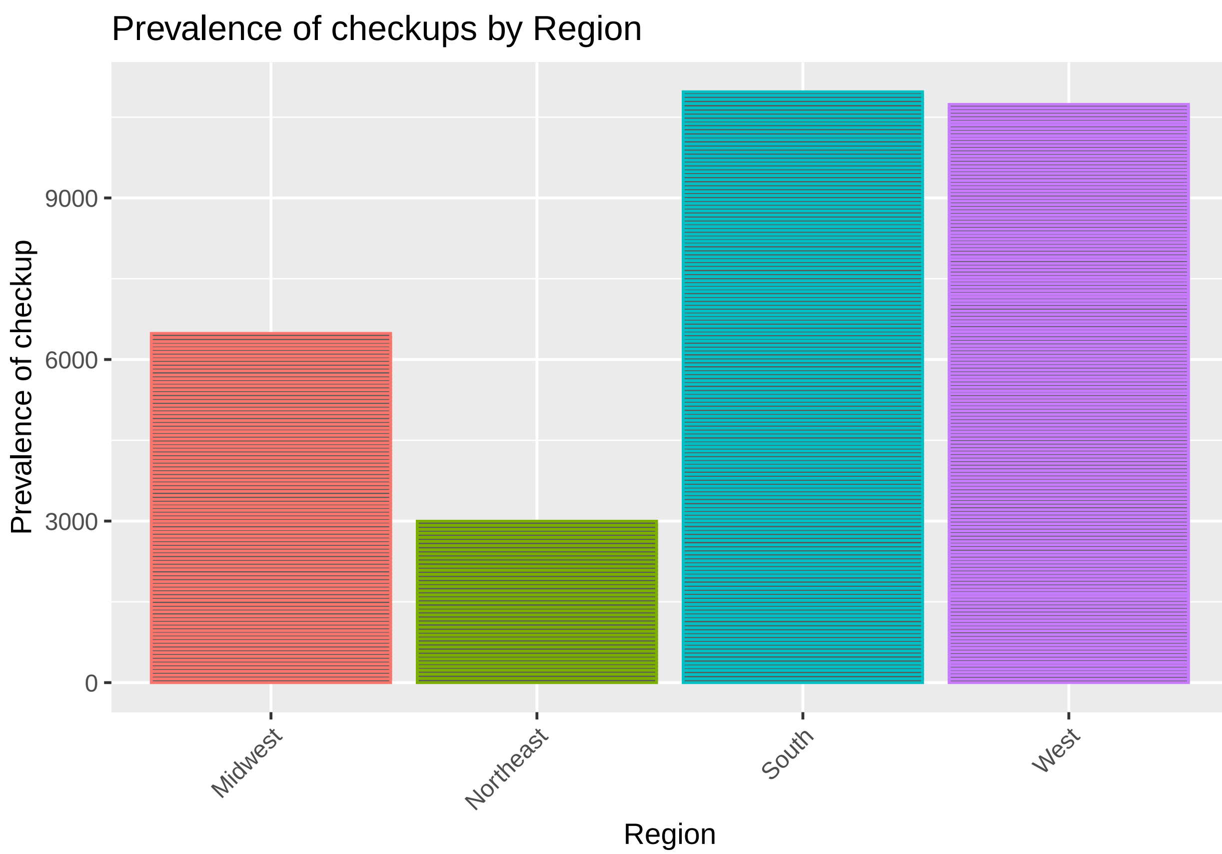

Number of Checkups by City Size
Number of People who got a Colon Screening
70
60
50
10
20
30
40
Lack of Health Care Access


Colorectal Cancer Screening and Lack of Healthcare Access
Men only
Men who had a Colorectal Cancer Screening
50
40
30
20
10
20
30
40
Lack of Health Care Access

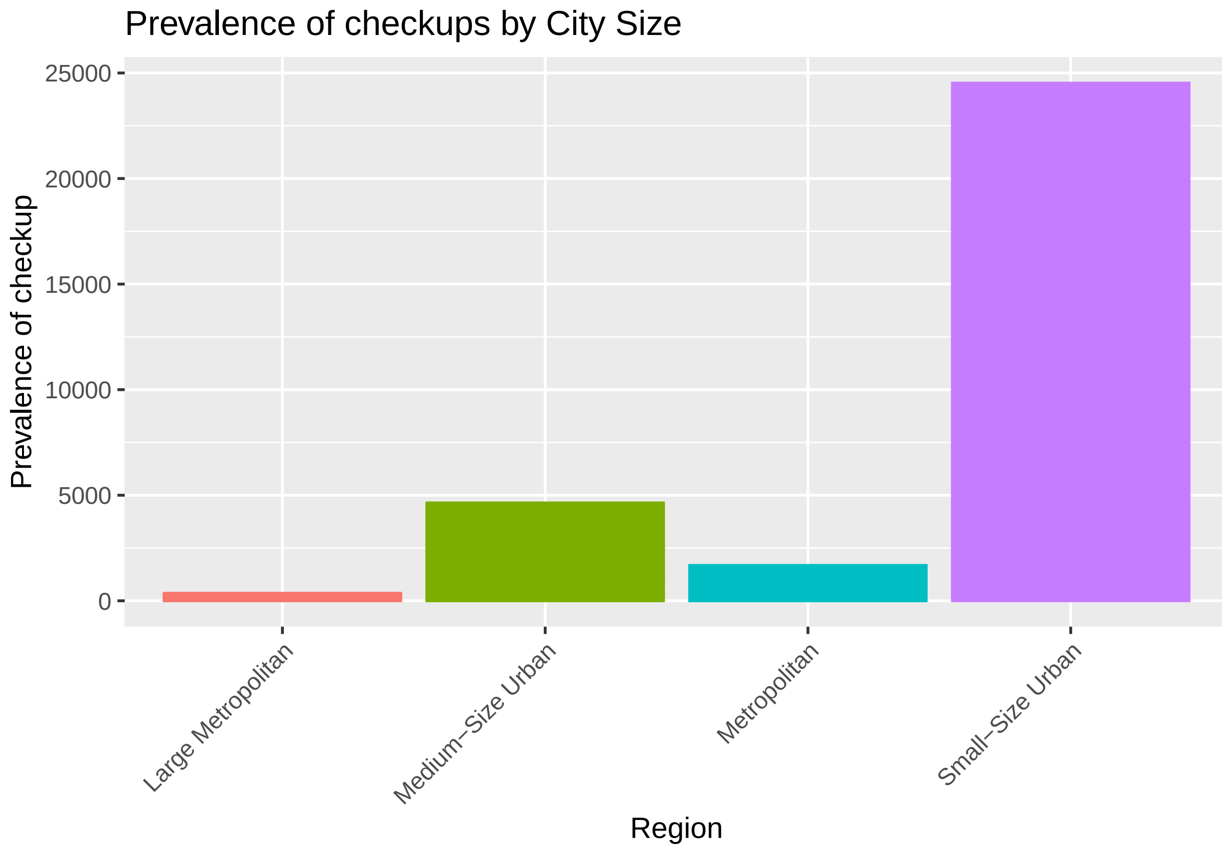

Colorectal Cancer Screening and Lack of Healthcare Access
Women only
Women who had a Colorectal Cancer Screening
40
30
20
10
20
30
40
Lack of Health Care Access




```r
cities2 <- cities %>%
  filter( region != "NA")

ggplot(cities2, aes(x = region,
                    y = checkup,
                    color = region))+
  geom_bar( stat = 'identity' )+
  labs( x = "Region",
        y = "Prevalence of checkup",
        title = "Prevalence of checkups by Region") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))+
  theme(legend.position = "none")
```

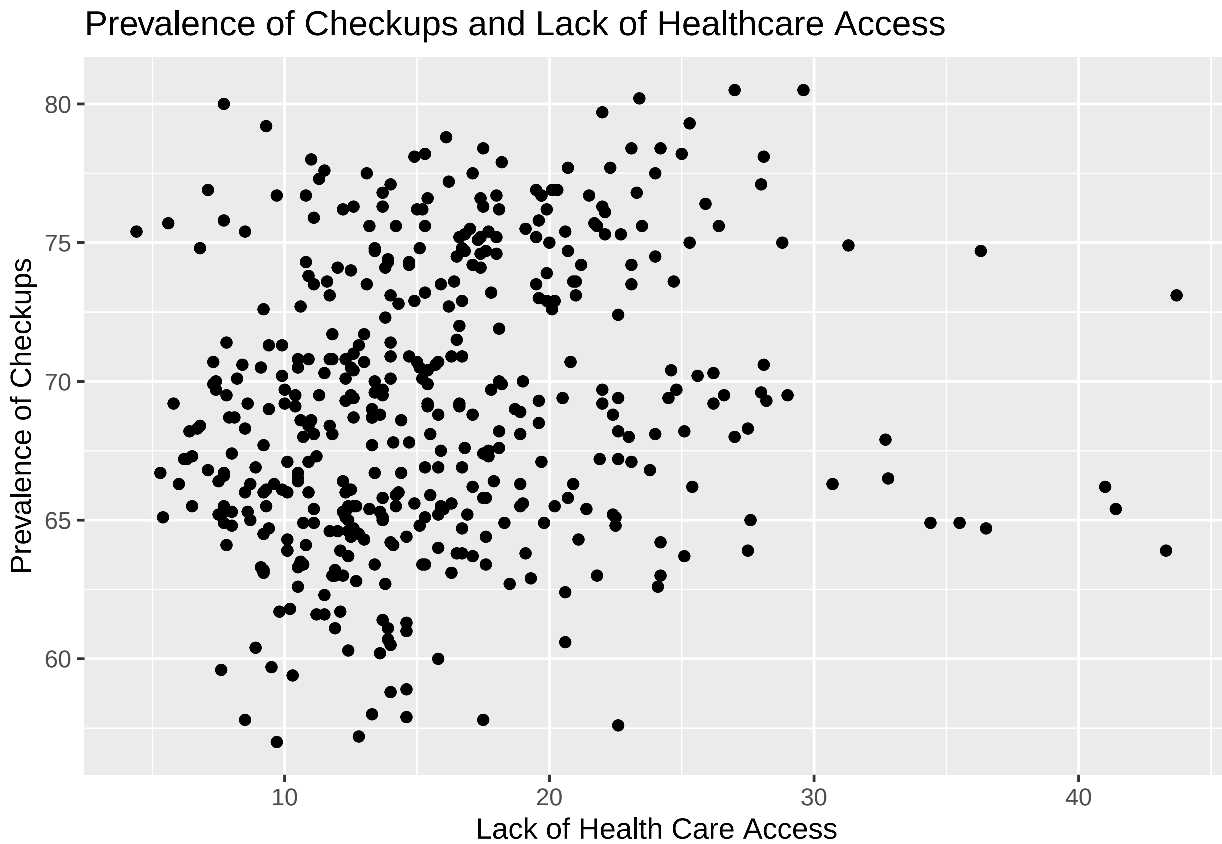



```r
cities2 <- cities %>%
  filter( city_size != "NA")

ggplot(cities2, aes(x = city_size,
                    y = checkup,
                    color = city_size))+
  geom_bar( stat = 'identity' )+
  labs( x = "Region",
        y = "Prevalence of checkup",
        title = "Prevalence of checkups by City Size") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))+
  theme(legend.position = "none")
```

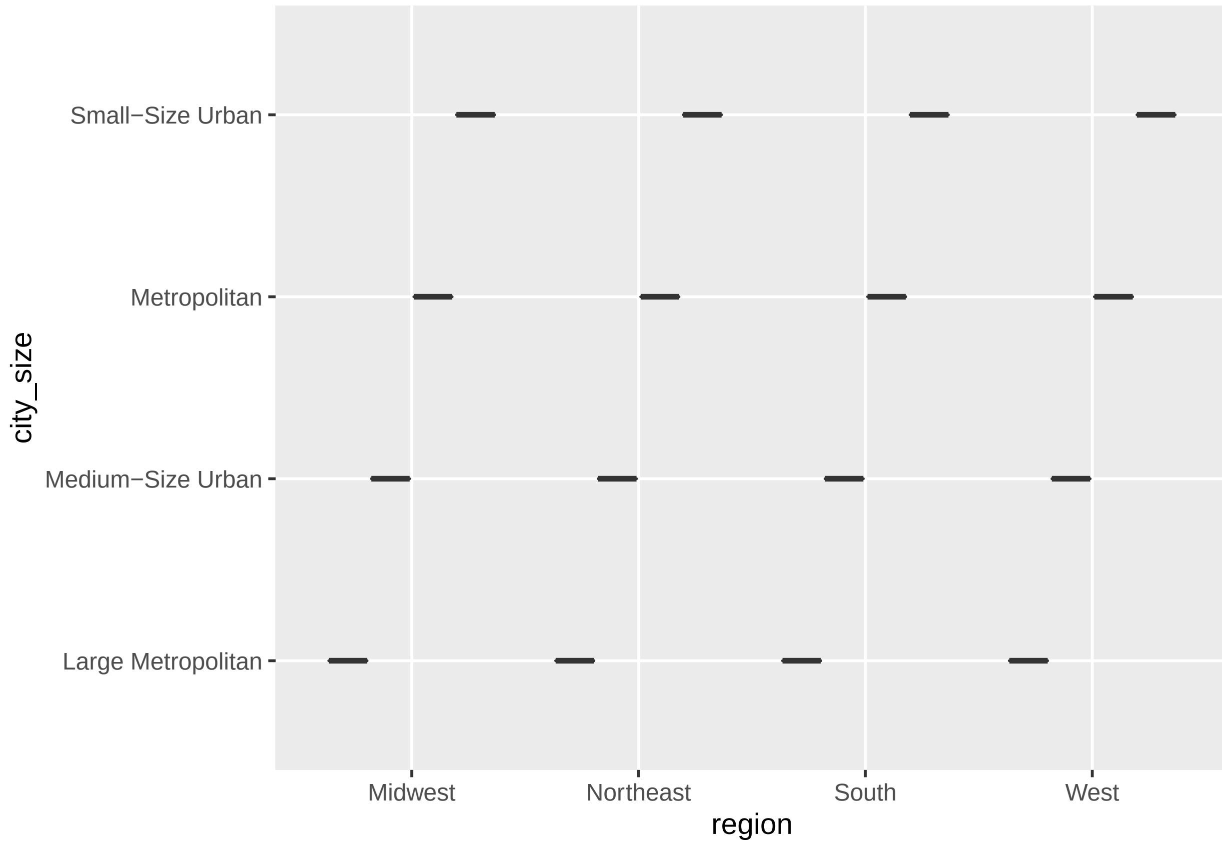



```
#doesn't look that helpful tbh won't hurt my feelings if u trash it
```

```
ggplot(cities, aes(x = health_access,
                   y= checkup))+
  geom_point()+
  labs( x = "Lack of Health Care Access",
        y = "Prevalence of Checkups",
        title = "Prevalence of Checkups and Lack of Healthcare Access")
```




```
#cloud of points means something-- don't forget this later on
```




```r
# cities_heatmap <- melt(cities, id.vars = "checkup", measure.vars =
#                        c("colon_screen", "mammogram", "pap_test"))

#head(cities_heatmap)
# ggplot(cities, aes(checkup, )) +


correlation1 <- cor.test(cities$checkup_n, cities$pap_n_screened,
                         method = "pearson")
correlation2 <- cor.test(cities$checkup_n, cities$colon_n_screened,
                         method = "pearson")
correlation3 <- cor.test(cities$checkup_n, cities$mammogram_n_screened,
                         method = "pearson")
print(correlation1)
```

```
##
##  Pearson's product-moment correlation
##
## data:  cities$checkup_n and cities$pap_n_screened
## t = 332.26, df = 449, p-value < 0.00000000000000022
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9975606 0.9983151
## sample estimates:
##       cor
## 0.9979726
```

```
print(correlation2)
```

```
##
##  Pearson's product-moment correlation
##
## data:  cities$checkup_n and cities$colon_n_screened
## t = 294.91, df = 449, p-value < 0.00000000000000022
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9969063 0.9978629
## sample estimates:
##       cor
## 0.9974287
```

```
print(correlation3)
```

```
##
##  Pearson's product-moment correlation
##
## data:  cities$checkup_n and cities$mammogram_n_screened
## t = 437.63, df = 449, p-value < 0.00000000000000022
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9985920 0.9990276
## sample estimates:
##       cor
## 0.9988299
```

Multiple Logistic Regression Model

```
##
## Call:
## glm(formula = cbind(checkup_n, no_checkup_n) ~ region + city_size +
##     health_access, family = binomial, data = model_cities)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -99.359  -18.975   -0.627   14.963  130.327
##
## Coefficients:
##                             Estimate  Std. Error  z value
## (Intercept)               0.97037012  0.00122436  792.555
## regionNortheast           0.19874789  0.00093326  212.960
## regionSouth               0.10411063  0.00084367  123.402
## regionWest               -0.27722312  0.00071380 -388.374
## city_sizeMedium-Size Urban  0.01336343  0.00086185   15.506
## city_sizeMetropolitan    -0.00289017  0.00084854   -3.406
## city_sizeSmall-Size Urban  -0.01782009  0.00077588  -22.968
## health_access            -0.00596421  0.00005375 -110.965
##                                       Pr(>|z|)
## (Intercept)              < 0.0000000000000002 ***
## regionNortheast          < 0.0000000000000002 ***
## regionSouth              < 0.0000000000000002 ***
## regionWest               < 0.0000000000000002 ***
```

```
## city_sizeMedium-Size Urban < 0.0000000000000002 ***
## city_sizeMetropolitan                 0.000659 ***
## city_sizeSmall-Size Urban  < 0.0000000000000002 ***
## health_access              < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 808649  on 336  degrees of freedom
## Residual deviance: 290529  on 329  degrees of freedom
## AIC: 294628
##
## Number of Fisher Scoring iterations: 3
```

```
##
## Call:
## glm(formula = cbind(checkup_n, no_checkup_n) ~ region + city_size +
##     health_access + region * city_size, family = binomial, data = model_cities)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -103.564   -17.554     0.418    14.078    86.320
##
## Coefficients:
##                                              Estimate  Std. Error  z value
## (Intercept)                                 0.97009830  0.00162902  595.512
## regionNortheast                             0.15804724  0.00156374  101.070
## regionSouth                                -0.01826191  0.00211966   -8.616
## regionWest                                 -0.21440752  0.00172584 -124.234
## city_sizeMedium-Size Urban                 -0.05649839  0.00178519  -31.648
## city_sizeMetropolitan                       0.09483272  0.00188009   50.440
## city_sizeSmall-Size Urban                  -0.06367042  0.00163329  -38.983
## health_access                              -0.00488135  0.00005675  -86.017
## regionNortheast:city_sizeMedium-Size Urban  0.16894949  0.00322129   52.448
## regionSouth:city_sizeMedium-Size Urban      0.20281749  0.00254564   79.672
## regionWest:city_sizeMedium-Size Urban      -0.01553915  0.00228157   -6.811
## regionNortheast:city_sizeMetropolitan       0.10447464  0.00301573   34.643
## regionSouth:city_sizeMetropolitan          -0.04324966  0.00254176  -17.016
## regionWest:city_sizeMetropolitan           -0.21259142  0.00239770  -88.665
## regionNortheast:city_sizeSmall-Size Urban   0.04923987  0.00237171   20.761
## regionSouth:city_sizeSmall-Size Urban       0.18529679  0.00234383   79.057
## regionWest:city_sizeSmall-Size Urban       -0.04092672  0.00205398  -19.926
##                                                        Pr(>|z|)
## (Intercept)                                < 0.0000000000000002 ***
## regionNortheast                            < 0.0000000000000002 ***
## regionSouth                                < 0.0000000000000002 ***
## regionWest                                 < 0.0000000000000002 ***
## city_sizeMedium-Size Urban                 < 0.0000000000000002 ***
## city_sizeMetropolitan                      < 0.0000000000000002 ***
## city_sizeSmall-Size Urban                  < 0.0000000000000002 ***
## health_access                              < 0.0000000000000002 ***
## regionNortheast:city_sizeMedium-Size Urban < 0.0000000000000002 ***
## regionSouth:city_sizeMedium-Size Urban     < 0.0000000000000002 ***
```

```
## regionWest:city_sizeMedium-Size Urban          0.00000000000971 ***
## regionNortheast:city_sizeMetropolitan      < 0.0000000000000002 ***
## regionSouth:city_sizeMetropolitan          < 0.0000000000000002 ***
## regionWest:city_sizeMetropolitan           < 0.0000000000000002 ***
## regionNortheast:city_sizeSmall-Size Urban  < 0.0000000000000002 ***
## regionSouth:city_sizeSmall-Size Urban      < 0.0000000000000002 ***
## regionWest:city_sizeSmall-Size Urban       < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 808649  on 336  degrees of freedom
## Residual deviance: 252886  on 320  degrees of freedom
## AIC: 257003
##
## Number of Fisher Scoring iterations: 3
```

checkup_fit <- logistic_reg() %>%

set_engine("glm") %>%

fit(as.factor(checkup_n) ~ region + city_size + health_access, data=cities,

family="binomial", exponentiate = TRUE)

result <- tidy(checkup_fit, conf.int=TRUE)

print(result)

#how do i visualize all the log reg models can i augment when using a log reg model that uses cbind? can i make a regression plot without using augment function?

```
##
## Call:
## glm(formula = cbind(colon_n_screened, no_colon_n_screened) ~
##     region + city_size + health_access, family = binomial, data = model_cities)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -98.034  -13.699   -1.068   18.543   65.912
##
## Coefficients:
##                            Estimate  Std. Error z value           Pr(>|z|)
## (Intercept)              0.96017646  0.00115479  831.47 <0.0000000000000002
## regionNortheast          0.01927332  0.00086022   22.41 <0.0000000000000002
## regionSouth              0.24458352  0.00079496  307.67 <0.0000000000000002
## regionWest               0.11740260  0.00068781  170.69 <0.0000000000000002
```

```
## city_sizeMedium-Size Urban  0.02151338  0.00081370   26.44 <0.0000000000000002
## city_sizeMetropolitan       0.02745954  0.00079974   34.34 <0.0000000000000002
## city_sizeSmall-Size Urban   0.02848220  0.00073139   38.94 <0.0000000000000002
## health_access              -0.03364736  0.00005098 -660.05 <0.0000000000000002
##
## (Intercept)                ***
## regionNortheast            ***
## regionSouth                ***
## regionWest                 ***
## city_sizeMedium-Size Urban ***
## city_sizeMetropolitan      ***
## city_sizeSmall-Size Urban  ***
## health_access              ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 763964  on 336  degrees of freedom
## Residual deviance: 221991  on 329  degrees of freedom
## AIC: 226119
##
## Number of Fisher Scoring iterations: 3
```

```
##
## Call:
## glm(formula = cbind(mammogram_n_screened, no_mammogram_n_screened) ~
##     region + city_size + health_access, family = binomial, data = model_cities)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -91.766  -17.040    4.115   18.677   90.943
##
## Coefficients:
##                             Estimate Std. Error  z value             Pr(>|z|)
## (Intercept)                1.2647938  0.0012665  998.664 < 0.0000000000000002
## regionNortheast            0.1026001  0.0009550  107.435 < 0.0000000000000002
## regionSouth                0.1759165  0.0008741  201.243 < 0.0000000000000002
## regionWest                 0.0265853  0.0007513   35.384 < 0.0000000000000002
## city_sizeMedium-Size Urban -0.0133440  0.0008966  -14.883 < 0.0000000000000002
## city_sizeMetropolitan      -0.0038478  0.0008831   -4.357            0.0000132
## city_sizeSmall-Size Urban  -0.0075516  0.0008070   -9.357 < 0.0000000000000002
## health_access              -0.0166378  0.0000557 -298.684 < 0.0000000000000002
##
## (Intercept)                ***
## regionNortheast            ***
## regionSouth                ***
## regionWest                 ***
## city_sizeMedium-Size Urban ***
## city_sizeMetropolitan      ***
## city_sizeSmall-Size Urban  ***
## health_access              ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 401462  on 336  degrees of freedom
## Residual deviance: 288696  on 329  degrees of freedom
## AIC: 292760
##
## Number of Fisher Scoring iterations: 3
```

```
##
## Call:
## glm(formula = cbind(pap_n_screened, no_pap_n_screened) ~ region +
##     city_size + health_access, family = binomial, data = model_cities)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -153.489   -15.237     2.326    16.525    85.826
##
## Coefficients:
##                              Estimate  Std. Error z value           Pr(>|z|)
## (Intercept)                1.66586484  0.00133834 1244.73 <0.0000000000000002
## regionNortheast           -0.07942884  0.00100571  -78.98 <0.0000000000000002
## regionSouth                0.12056658  0.00092792  129.93 <0.0000000000000002
## regionWest                 0.01680779  0.00080678   20.83 <0.0000000000000002
## city_sizeMedium-Size Urban -0.10257377  0.00094715 -108.30 <0.0000000000000002
## city_sizeMetropolitan     -0.04724339  0.00093719  -50.41 <0.0000000000000002
## city_sizeSmall-Size Urban  -0.06835666  0.00085439  -80.01 <0.0000000000000002
## health_access             -0.02159918  0.00005833 -370.29 <0.0000000000000002
##
## (Intercept)                ***
## regionNortheast            ***
## regionSouth                ***
## regionWest                 ***
## city_sizeMedium-Size Urban ***
## city_sizeMetropolitan      ***
## city_sizeSmall-Size Urban  ***
## health_access              ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 392788  on 336  degrees of freedom
## Residual deviance: 242902  on 329  degrees of freedom
## AIC: 246928
##
## Number of Fisher Scoring iterations: 3
```

Model Validation

model that was made from the training data set compared to the rest of the test data points from the cities data

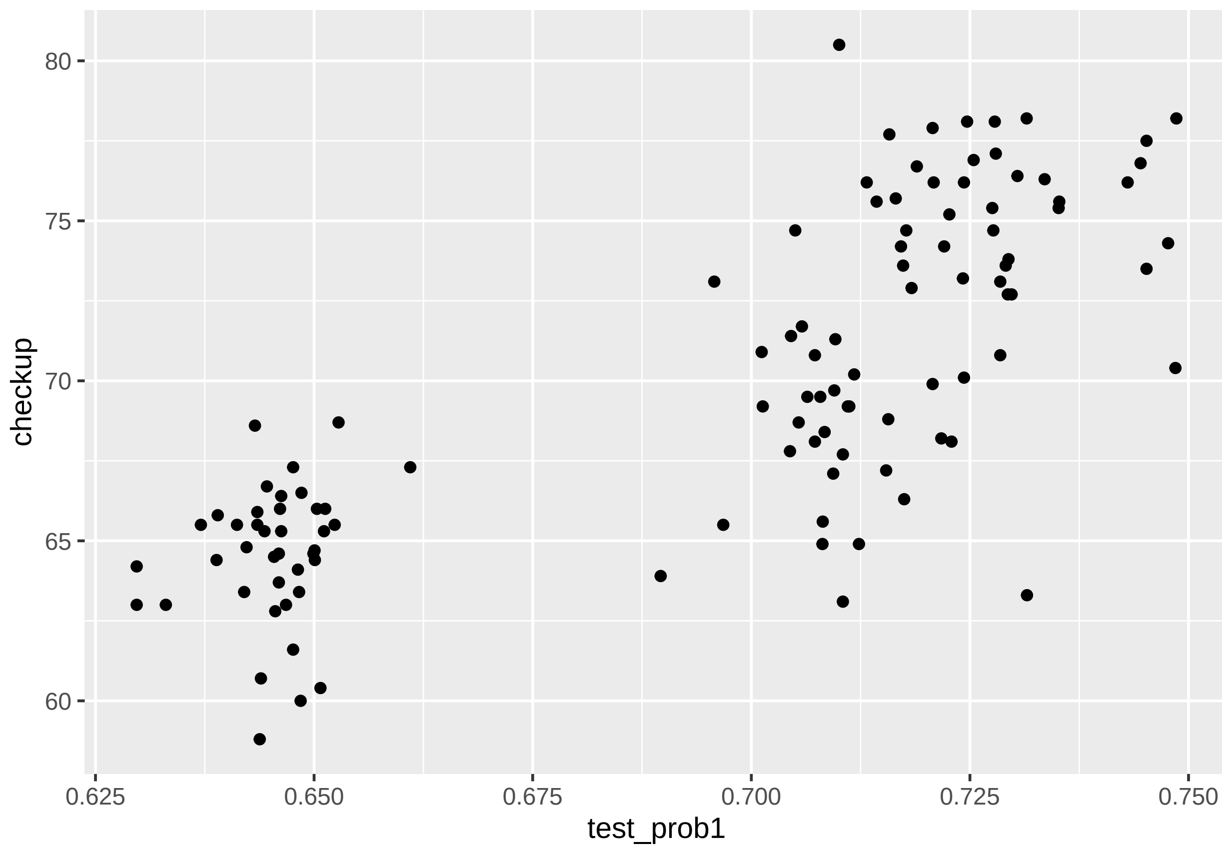
```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
# predictions from no interaction model

test_prob1 = predict(checkup_fit, newdata = cities_test, type = "response")
print(test_prob1)
```

```
##         1         2         3         4         5         6         7         8
## 0.6455606 0.7314895 0.7011851 0.7228949 0.7112079 0.7050252 0.7078892 0.6422785
##         9        10        11        12        13        14        15        16
## 0.6523540 0.7276781 0.6500896 0.7057882 0.7093670 0.6459699 0.6957643 0.7177244
##        17        18        19        20        21        22        23        24
## 0.7083823 0.7351469 0.7208554 0.7154229 0.7243224 0.6459699 0.7294139 0.6503223
##        25        26        27        28        29        30        31        32
## 0.7157870 0.7352215 0.6481492 0.7284711 0.7143288 0.7110314 0.6446047 0.6527981
##        33        34        35        36        37        38        39        40
## 0.7096128 0.7054165 0.7094899 0.7451987 0.7226520 0.6461062 0.7220539 0.7156657
##        41        42        43        44        45        46        47        48
## 0.7207354 0.7451987 0.7174826 0.7044241 0.7430410 0.7104724 0.6389831 0.7072722
##        49        50        51        52        53        54        55        56
## 0.7123092 0.6467878 0.6435109 0.7165145 0.7284711 0.6454241 0.7445186 0.7242033
##        57        58        59        60        61        62        63        64
## 0.7335473 0.6511356 0.7117661 0.7243224 0.7476684 0.7315275 0.7064071 0.6512711
##        65        66        67        68        69        70        71        72
## 0.6370547 0.7013101 0.6485571 0.7217288 0.7072722 0.6484598 0.7081709 0.6439212
##        73        74        75        76        77        78        79        80
## 0.6297147 0.6896344 0.6297147 0.7254268 0.7297669 0.7100500 0.7293296 0.7290941
##        81        82        83        84        85        86        87        88
## 0.6609976 0.6507291 0.7171198 0.6437845 0.7104724 0.7486131 0.6499197 0.7189311
##        89        90        91        92        93        94        95        96
## 0.7278425 0.6500510 0.7045482 0.6411815 0.6967931 0.7279606 0.7207354 0.6476049
##        97        98        99       100       101       102       103       104
## 0.7131987 0.6388455 0.6420044 0.7485008 0.7246796 0.6462426 0.7173617 0.6443314
##       105       106       107       108       109       110       111       112
## 0.6432372 0.7183281 0.7304339 0.6462426 0.6435109 0.6482852 0.6330461 0.6476049
##       113       114
## 0.7081358 0.7275599
```

```r
new_cities_test <-
  merge(cities_test, test_prob1, by = "row.names", all.x = TRUE)

ggplot(cities_test, aes(x = test_prob1, y = checkup)) +
  geom_point()
```

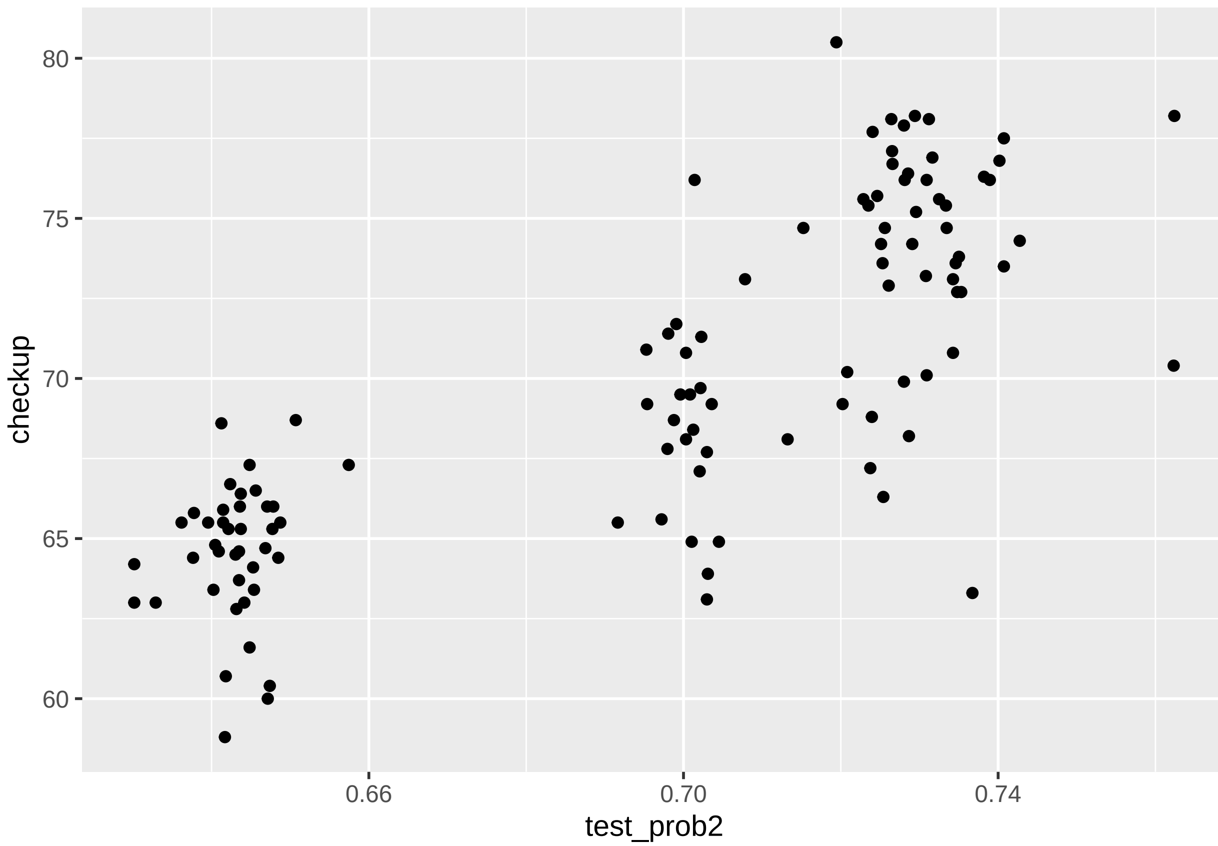



```r
# prediction from model with interaction

test_prob2 = predict(checkup_fit2, newdata = cities_test, type = "response")
print(test_prob2)
```

```
##         1         2         3         4         5         6         7         8
## 0.6431587 0.7294199 0.6952775 0.7132379 0.7035892 0.7152382 0.7008332 0.6404655
##         9        10        11        12        13        14        15        16
## 0.6487404 0.7334590 0.6484848 0.6990904 0.7020599 0.6434947 0.7078243 0.7256013
##        17        18        19        20        21        22        23        24
## 0.7012424 0.7235076 0.7281210 0.7237509 0.7309143 0.6434947 0.7350225 0.6470701
##        25        26        27        28        29        30        31        32
## 0.7240436 0.7324917 0.6452845 0.7342612 0.7228717 0.7202237 0.6423741 0.6507070
##        33        34        35        36        37        38        39        40
## 0.7022641 0.6987823 0.7021620 0.7407306 0.7295681 0.6436067 0.7290863 0.7239460
##        41        42        43        44        45        46        47        48
## 0.7280244 0.7407306 0.7254069 0.6979597 0.7389454 0.7029780 0.6377634 0.7003212
##        49        50        51        52        53        54        55        56
## 0.7045046 0.6441664 0.6414765 0.7246284 0.7342612 0.6430467 0.7401677 0.7308183
##        57        58        59        60        61        62        63        64
## 0.7382032 0.6477387 0.7208134 0.7309143 0.7427469 0.7367302 0.6996036 0.6478500
##        65        66        67        68        69        70        71        72
## 0.6361832 0.6953809 0.6456196 0.7286605 0.7003212 0.6471484 0.6972072 0.6418132
##        73        74        75        76        77        78        79        80
## 0.6301744 0.7030993 0.6301744 0.7316419 0.7353076 0.7194361 0.7347929 0.7346026
##        81        82        83        84        85        86        87        88
```

```
## 0.6574440 0.6474045 0.7251151 0.6417010 0.7029780 0.7624053 0.6409192 0.7265722
##        89        90        91        92        93        94        95        96
## 0.7264229 0.6468471 0.6980626 0.6395658 0.6916457 0.7265199 0.7280244 0.6448374
##        97        98        99       100       101       102       103       104
## 0.7014155 0.6376507 0.6402407 0.7623169 0.7312022 0.6437187 0.7253097 0.6421498
##       105       106       107       108       109       110       111       112
## 0.6412520 0.7260870 0.7285519 0.6437187 0.6414765 0.6453962 0.6329005 0.6448374
##       113       114
## 0.7010378 0.7333636
```

```
new_cities_test <-
  merge(cities_test, test_prob2, by = "row.names", all.x = TRUE)

ggplot(cities_test, aes(x = test_prob2, y = checkup)) +
  geom_point()
```




```
#calculate root mean standard error of both models

rmse(new_cities_test, truth = checkup, estimate = test_prob1)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        69.0
```

```
rmse(new_cities_test, truth = checkup, estimate = test_prob2)
```

```
## # A tibble: 1 x 3
```

```
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        69.0
```