Data Preparation

`census <- read.csv("COVID-19_cases_plus_census3.csv")`

Use two data set(`2021-1-19`, `2021-1-26`) to get the delta of deaths and confirmed cases within one week.(P1-T2)

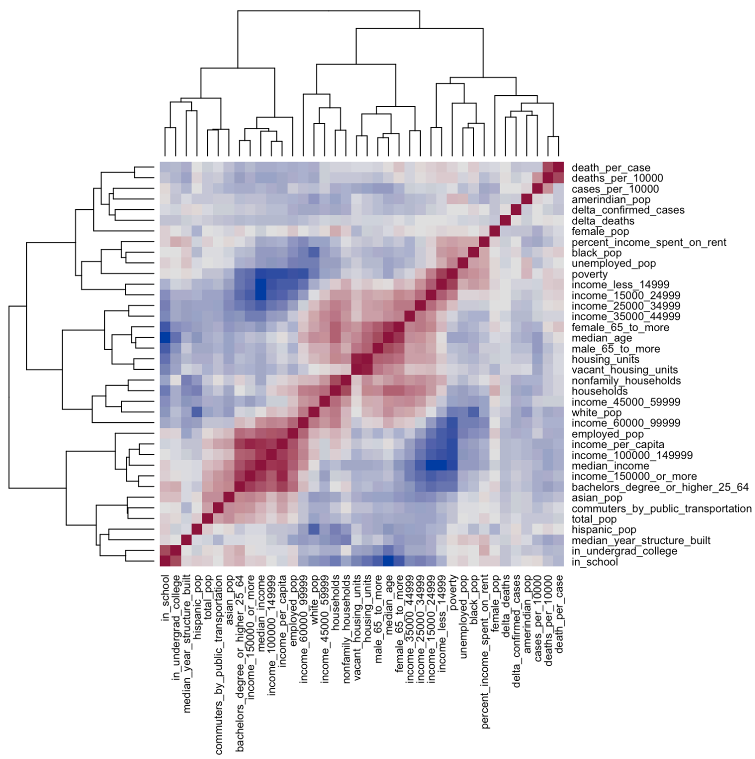
filter out zero

```
> dim(cases)
[1] 3142  261
> table(complete.cases(cases))

FALSE
 3142
> cases <- cases %>%
+   filter(confirmed_cases > 0) %>%
+   filter(deaths >= 0) %>%
+   filter(delta_deaths >= 0)
> dim(cases)
[1] 3126  261
> table(complete.cases(cases))

FALSE
 3126
```

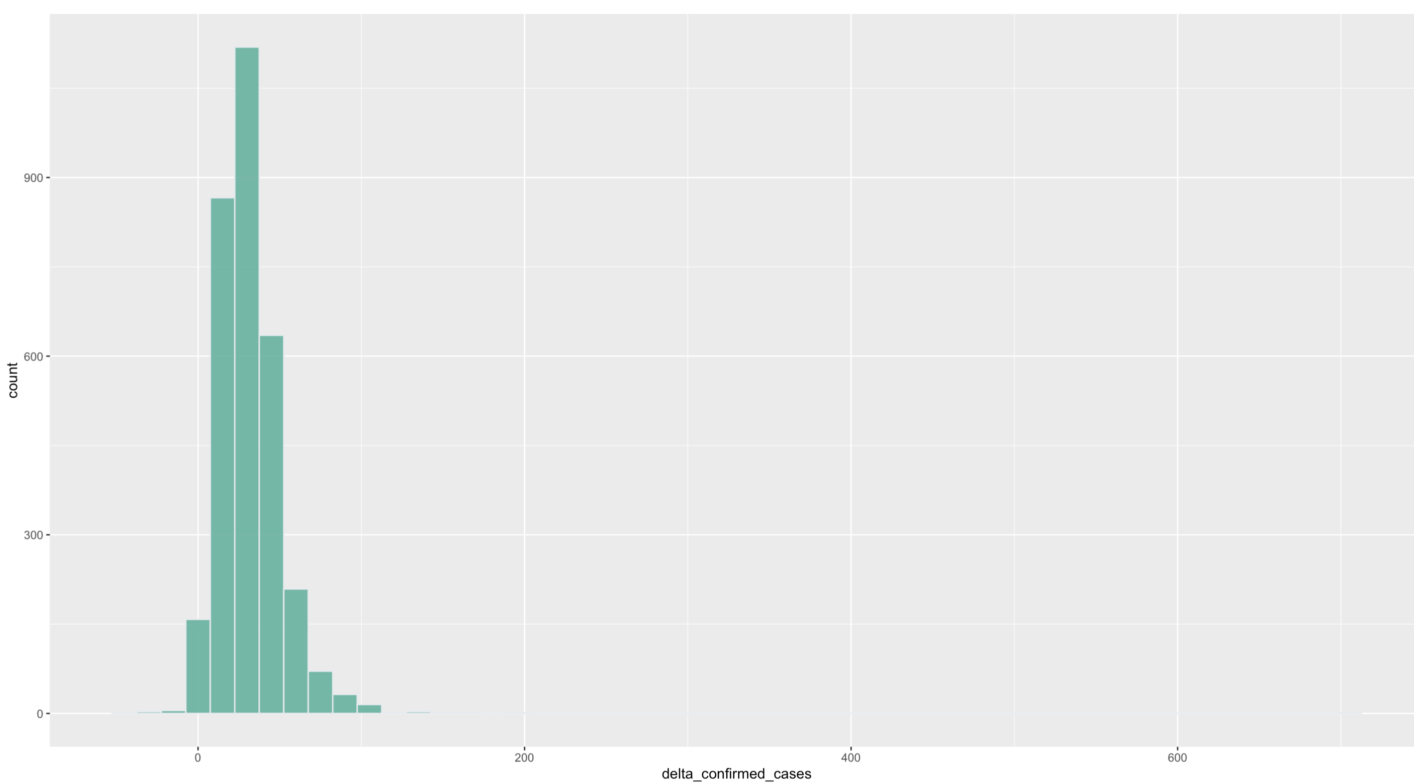
Remove N/A(P1-T3)

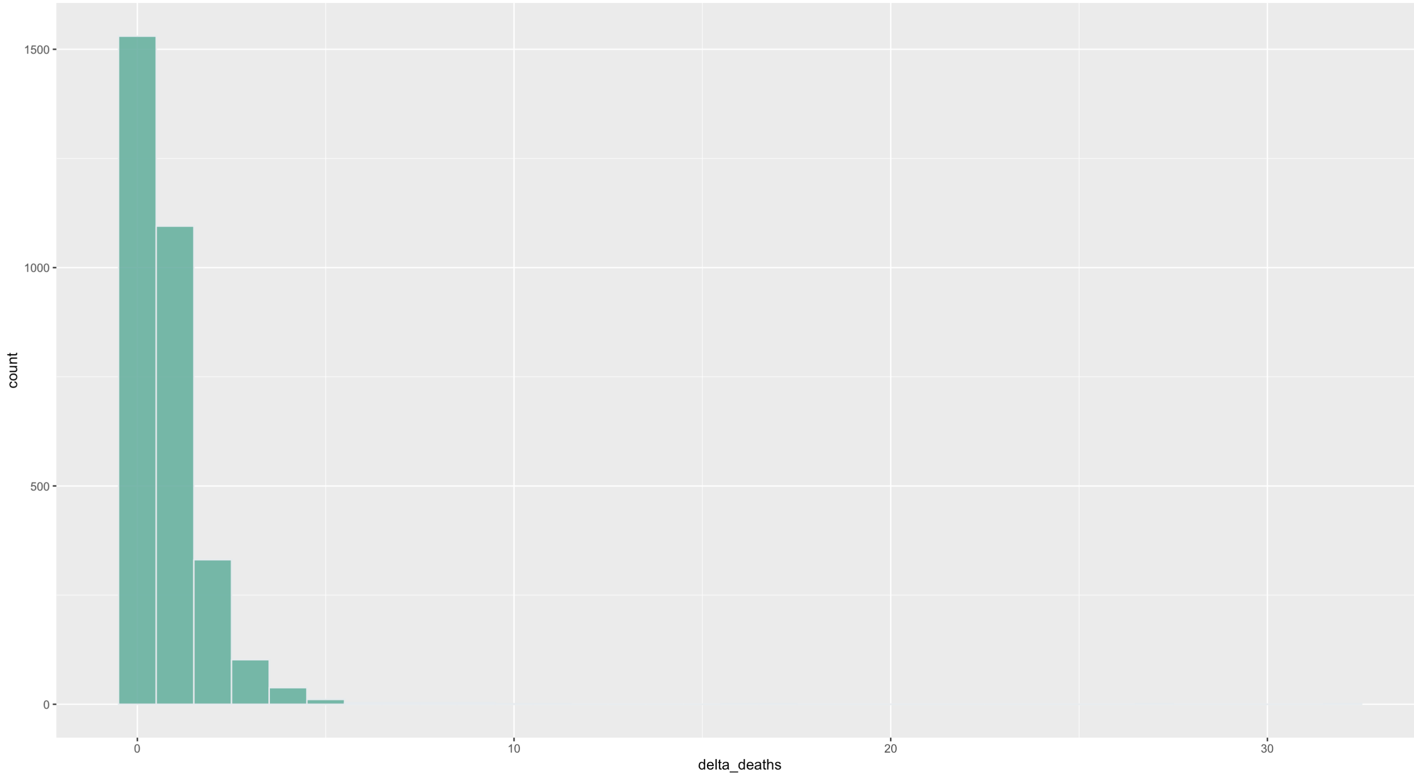
Deal with missing data(for classification models that cannot handle missing data)

```
> table(complete.cases(cases_sel))

FALSE  TRUE
    1  3125
> cases_sel <- cases_sel %>% na.omit
> table(complete.cases(cases_sel))

TRUE
3125
```



Define classes(P1-T1)

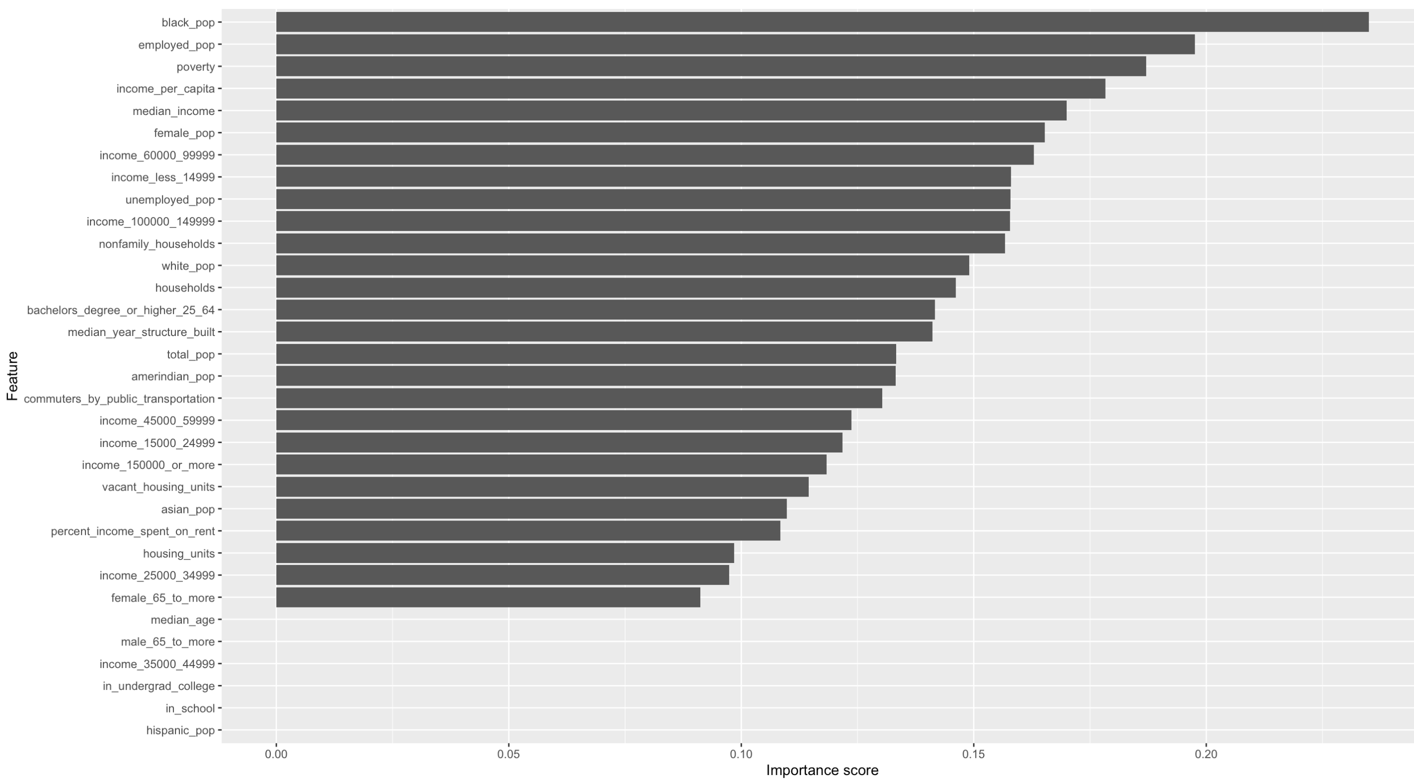





```
 female_65_to_more cases_per_10000   deaths_per_10000 death_per_case
 Min.    :0.02154   Min.    :  24.62   Min.    : 0.000    Min.    :0.00000
 1st Qu.:0.08317    1st Qu.: 592.82    1st Qu.: 7.048     1st Qu.:0.01045
 Median :0.09671    Median : 766.66    Median :12.177     Median :0.01578
 Mean   :0.09753    Mean   : 776.26    Mean   :13.796     Mean   :0.01795
 3rd Qu.:0.11019    3rd Qu.: 938.81    3rd Qu.:18.007     3rd Qu.:0.02354
 Max.   :0.27841    Max.   :3161.04    Max.   :83.587     Max.   :0.18182

  delta_deaths      delta_confirmed_cases      risk
 Min.    : 0.0000   Min.    :-41.64         Length:3125
 1st Qu.: 0.0000    1st Qu.: 19.22          Class :character
 Median : 0.5152    Median : 29.14          Mode  :character
 Mean   : 0.7979    Mean   : 31.75
 3rd Qu.: 1.1411    3rd Qu.: 40.92
 Max.   :32.0616    Max.   :701.14

cases_sel <- cases_sel %>%
  mutate(
    risk = case_when(
      delta_confirmed_cases > 29.14 & delta_deaths >= 1.1  ~ "high",
      delta_confirmed_cases > 29.14 & delta_deaths < 1.1 ~ "medium",

      delta_confirmed_cases <= 29.14 & delta_deaths >= 1.1 ~ "medium",
      delta_confirmed_cases <= 29.14 & delta_deaths < 1.1 ~ "low"
      )
```

```
  )

> cases_sel %>% pull(risk) %>% table()
.
  high    low medium 
   160   2370    595 
```

Select Features(P1-T3)

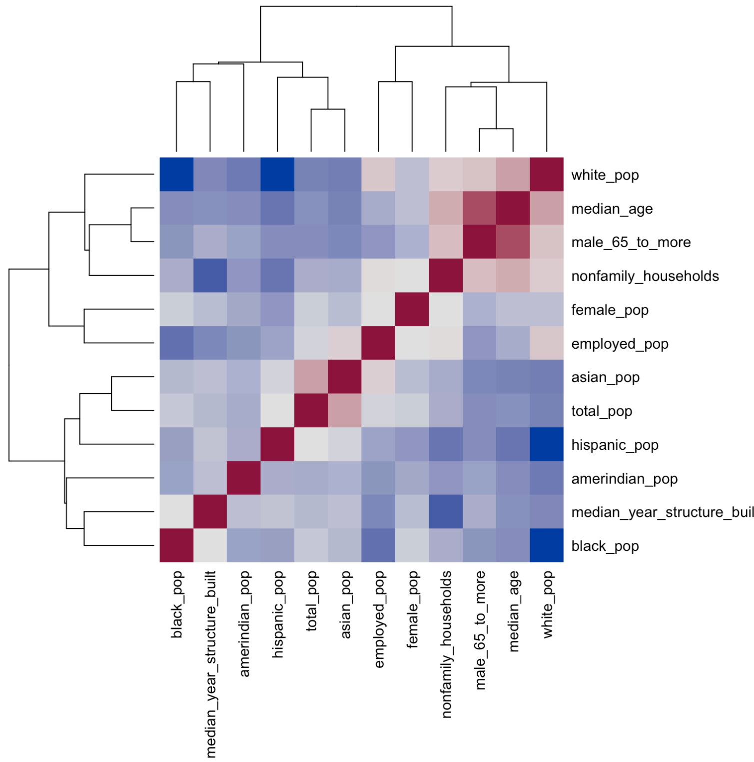



```
# These scores measure how related each feature is to the class variable.
# For discrete features (as in our case), the chi-square statistic can be used to derive a
score.
> weights
# A tibble: 33 × 2
   feature                 attr_importance
   <chr>                             <dbl>
 1 black_pop                         0.235
 2 employed_pop                      0.198
 3 poverty                           0.187
 4 income_per_capita                 0.178
 5 median_income                     0.170
 6 female_pop                        0.165
 7 income_60000_99999                0.163
 8 income_less_14999                 0.158
 9 unemployed_pop                    0.158
```

```
10 income_100000_149999            0.158

## backward
> subset
 [1] "total_pop"
 [2] "nonfamily_households"
 [3] "median_year_structure_built"
 [4] "female_pop"
 [5] "median_age"
 [6] "white_pop"
 [7] "black_pop"
 [8] "asian_pop"
 [9] "hispanic_pop"
[10] "amerindian_pop"
[11] "commuters_by_public_transportation"
[12] "households"
[13] "median_income"
[14] "housing_units"
[15] "vacant_housing_units"
[16] "percent_income_spent_on_rent"
[17] "employed_pop"
[18] "unemployed_pop"
[19] "in_school"
[20] "in_undergrad_college"
[21] "income_per_capita"
[22] "poverty"
[23] "income_less_14999"
[24] "income_15000_24999"
[25] "income_25000_34999"
[26] "income_35000_44999"
[27] "income_45000_59999"
[28] "income_60000_99999"
[29] "income_100000_149999"
[30] "income_150000_or_more"
[31] "male_65_to_more"

# greedy search heuristics
# cfs uses correlation/entropy with best first search
> cases_feature %>% cfs(risk ~ ., data = .)
 [1] "total_pop"
 [2] "nonfamily_households"
 [3] "median_year_structure_built"
 [4] "female_pop"
 [5] "white_pop"
 [6] "black_pop"
 [7] "amerindian_pop"
```

```
 [8] "commuters_by_public_transportation"
 [9] "vacant_housing_units"
[10] "employed_pop"
[11] "unemployed_pop"
[12] "income_per_capita"
[13] "poverty"
[14] "income_60000_99999"

# forward
> subset
[1] "total_pop"  "black_pop"  "households"

# best first search
> subset
[1] "black_pop"                    "asian_pop"
[3] "percent_income_spent_on_rent" "income_per_capita"
[5] "income_60000_99999"

# hill climbing
> subset
 [1] "total_pop"
 [2] "female_pop"
 [3] "median_age"
 [4] "white_pop"
 [5] "black_pop"
 [6] "commuters_by_public_transportation"
 [7] "percent_income_spent_on_rent"
 [8] "income_per_capita"
 [9] "poverty"
[10] "bachelors_degree_or_higher_25_64"
[11] "income_15000_24999"
[12] "income_45000_59999"
[13] "income_60000_99999"
[14] "income_150000_or_more"
[15] "male_65_to_more"
[16] "female_65_to_more"
```

Selected Data



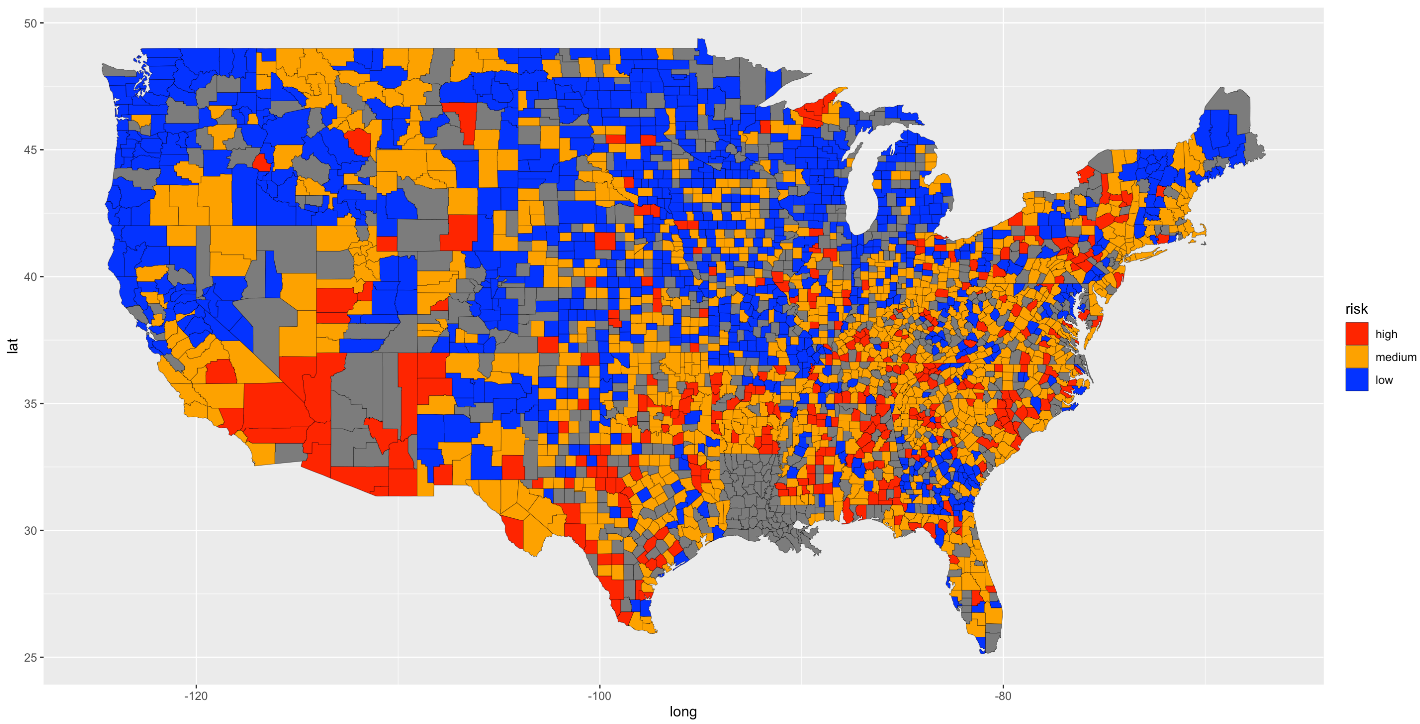

```
> summary(cases_sel)
             county_name        state          risk          total_pop
 Washington County :  30   TX      : 254   high  : 917   Min.   :      74
 Jefferson County  :  25   GA      : 159   low   : 916   1st Qu.:   11014
 Franklin County   :  23   VA      : 132   medium:1292   Median :   25801
 Jackson County    :  23   KY      : 120                 Mean   :  102633
 Lincoln County    :  22   MO      : 115                 3rd Qu.:   68328
 Madison County    :  19   KS      : 105                 Max.   :10105722
 (Other)           :2983   (Other):2240
 nonfamily_households median_year_structure_built   female_pop
 Min.   :0.03023      Min.   :1939                Min.   :0.1917
 1st Qu.:0.11069      1st Qu.:1968                1st Qu.:0.4943
 Median :0.12918      Median :1977                Median :0.5040
 Mean   :0.12907      Mean   :1975                Mean   :0.4992
 3rd Qu.:0.14619      3rd Qu.:1983                3rd Qu.:0.5110
 Max.   :0.26462      Max.   :2003                Max.   :0.5663

   median_age       white_pop          black_pop          asian_pop
 Min.   :21.60   Min.   :0.006354   Min.   :0.000000   Min.   :0.000000
 1st Qu.:37.90   1st Qu.:0.651320   1st Qu.:0.006083   1st Qu.:0.002704
 Median :41.20   Median :0.842260   Median :0.021453   Median :0.005762
 Mean   :41.13   Mean   :0.767899   Mean   :0.089202   Mean   :0.013116
 3rd Qu.:44.20   3rd Qu.:0.929433   3rd Qu.:0.099795   3rd Qu.:0.012213
 Max.   :66.40   Max.   :1.000000   Max.   :0.869207   Max.   :0.418079
```

```
  hispanic_pop      amerindian_pop       employed_pop     male_65_to_more
 Min.    :0.00000   Min.    :0.000000   Min.    :0.1017   Min.    :0.007092
 1st Qu.:0.02059   1st Qu.:0.001222   1st Qu.:0.3960   1st Qu.:0.023450
 Median :0.03987   Median :0.002700   Median :0.4429   Median :0.027303
 Mean    :0.09142   Mean    :0.017665   Mean    :0.4382   Mean    :0.028492
 3rd Qu.:0.09298   3rd Qu.:0.006319   3rd Qu.:0.4861   3rd Qu.:0.032003
 Max.    :0.99185   Max.    :0.903156   Max.    :0.7205   Max.    :0.093417
```

Modeling

Training & Testing data

```
# Training data 80%
inTrain <- createDataPartition(y = cases_sel$risk, p = .8, list = FALSE)
cases_train <- cases_sel %>% slice(inTrain)
cases_test <- cases_sel %>% slice(-inTrain)

# Training
> cases_train %>% pull(risk) %>% table()
.
  high    low medium
   372    984   1145

# Testing
> cases_test %>% pull(risk) %>% table()
.
  high    low medium
    93    245    286
```



Training

Conditional Inference Tree (ctree)

```
> ctreeFit
Conditional Inference Tree

2711 samples
  14 predictor
   3 classes: 'high', 'low', 'medium'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2441, 2440, 2439, 2438, 2440, 2441, ...
Resampling results across tuning parameters:

  mincriterion  Accuracy   Kappa
  0.01000000    0.6758104  0.4824306
  0.06157895    0.6500050  0.4410585
  0.11315789    0.6245293  0.3985362
  0.16473684    0.6013163  0.3604453
  0.21631579    0.6058618  0.3647458
  0.26789474    0.5966677  0.3502809
  0.31947368    0.5990486  0.3523090
  0.37105263    0.6014296  0.3569440
```

```
  0.42263158    0.5828753  0.3273234
  0.47421053    0.5781687  0.3194220
  0.52578947    0.5781687  0.3184235
  0.57736842    0.5666490  0.3002668
  0.62894737    0.5622118  0.2948423
  0.68052632    0.5645374  0.3026041
  0.73210526    0.5623200  0.2974762
  0.78368421    0.5624258  0.2970404
  0.83526316    0.5577746  0.2863022
  0.88684211    0.5436550  0.2618543
  0.93842105    0.5068585  0.2061354
  0.99000000    0.4952809  0.1896892

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mincriterion = 0.01.
```

Tried 20 different running parameters(mincriterion).

Best: mincriterion = 0.01

> mincriterion - 1 - P-Value Threshold

C 4.5 Decision Tree

```
> C45Fit
C4.5-like Trees

2711 samples
  14 predictor
   3 classes: 'high', 'low', 'medium'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2441, 2439, 2440, 2439, 2440, 2439, ...
Resampling results across tuning parameters:

  C            M   Accuracy   Kappa
  0.01000000   1   0.6444387  0.4589398
  0.01000000   2   0.6377966  0.4498118
  0.01000000   3   0.6234258  0.4259682
  0.01000000   4   0.6186164  0.4193127
  0.01000000   5   0.5990768  0.3926099
  0.01000000   6   0.5894990  0.3745032
```

```
0.01000000   7  0.5743725  0.3519192
0.01000000   8  0.5732573  0.3507459
0.01000000   9  0.5691983  0.3428352
0.01000000  10  0.5658786  0.3413603
0.06444444   1  0.7192870  0.5755709
0.06444444   2  0.7019492  0.5495227
0.06444444   3  0.6805836  0.5166397
0.06444444   4  0.6680361  0.4972422
0.06444444   5  0.6628645  0.4900231
0.06444444   6  0.6410973  0.4559170
0.06444444   7  0.6300203  0.4368166
0.06444444   8  0.6208060  0.4225992
0.06444444   9  0.6174782  0.4167563
0.06444444  10  0.6078976  0.4016713
0.11888889   1  0.7403122  0.6075929
0.11888889   2  0.7115352  0.5638687
0.11888889   3  0.6975226  0.5424142
0.11888889   4  0.6842546  0.5221615
0.11888889   5  0.6750404  0.5087332
0.11888889   6  0.6495831  0.4694207
0.11888889   7  0.6392345  0.4529194
0.11888889   8  0.6307732  0.4385801
0.11888889   9  0.6241216  0.4272300
0.11888889  10  0.6156413  0.4136368
0.17333333   1  0.7473193  0.6184239
0.17333333   2  0.7152212  0.5696930
0.17333333   3  0.7026791  0.5503753
0.17333333   4  0.6868349  0.5258768
0.17333333   5  0.6765178  0.5105326
0.17333333   6  0.6514254  0.4721976
0.17333333   7  0.6429327  0.4585300
0.17333333   8  0.6303974  0.4383981
0.17333333   9  0.6278144  0.4337671
0.17333333  10  0.6215468  0.4243267
0.22777778   1  0.7495319  0.6218469
0.22777778   2  0.7226041  0.5810734
0.22777778   3  0.7078520  0.5583649
0.22777778   4  0.6897883  0.5308474
0.22777778   5  0.6790981  0.5153148
0.22777778   6  0.6528973  0.4750416
0.22777778   7  0.6447669  0.4613632
0.22777778   8  0.6337158  0.4436775
0.22777778   9  0.6318680  0.4401579
0.22777778  10  0.6256031  0.4310047
0.28222222   1  0.7517405  0.6251371
0.28222222   2  0.7262900  0.5867013
```

```
0.28222222   3  0.7126463  0.5660340
0.28222222   4  0.6912657  0.5336424
0.28222222   5  0.6813081  0.5190966
0.28222222   6  0.6558466  0.4794961
0.28222222   7  0.6495626  0.4683028
0.28222222   8  0.6355608  0.4464629
0.28222222   9  0.6340875  0.4435328
0.28222222  10  0.6270805  0.4332840
0.33666667   1  0.7539532  0.6285503
0.33666667   2  0.7266563  0.5874749
0.33666667   3  0.7126423  0.5661555
0.33666667   4  0.6912630  0.5337363
0.33666667   5  0.6809472  0.5191634
0.33666667   6  0.6606423  0.4870886
0.33666667   7  0.6532376  0.4741395
0.33666667   8  0.6385033  0.4512038
0.33666667   9  0.6381384  0.4499956
0.33666667  10  0.6318694  0.4403409
0.39111111   1  0.7546926  0.6296595
0.39111111   2  0.7270267  0.5880059
0.39111111   3  0.7133803  0.5672122
0.39111111   4  0.6912644  0.5337370
0.39111111   5  0.6805782  0.5186385
0.39111111   6  0.6621211  0.4894762
0.39111111   7  0.6521293  0.4726173
0.39111111   8  0.6396035  0.4530744
0.39111111   9  0.6381357  0.4501597
0.39111111  10  0.6314949  0.4400143
0.44555556   1  0.7546926  0.6296595
0.44555556   2  0.7285000  0.5902373
0.44555556   3  0.7148563  0.5695175
0.44555556   4  0.6916320  0.5343138
0.44555556   5  0.6805782  0.5187371
0.44555556   6  0.6628591  0.4907739
0.44555556   7  0.6536026  0.4748635
0.44555556   8  0.6396035  0.4530744
0.44555556   9  0.6381357  0.4501597
0.44555556  10  0.6314949  0.4400143
0.50000000   1  0.7554306  0.6308367
0.50000000   2  0.7288677  0.5907852
0.50000000   3  0.7148563  0.5695002
0.50000000   4  0.6923673  0.5354980
0.50000000   5  0.6802092  0.5182104
0.50000000   6  0.6621238  0.4897081
0.50000000   7  0.6536026  0.4748635
0.50000000   8  0.6407065  0.4546299
```

```
  0.50000000   9  0.6385033  0.4506728
  0.50000000  10  0.6322329  0.4408686

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were C = 0.5 and M = 1.
```

Tried 10*10 different running parameters(C, M).

Best: C = 0.5 and M = 1

> C - Confidence Threshold
>
> M - Minimum Instances Pre Leaf

K-Nearest Neighbors(knn & kknn)

```
> knnFit
k-Nearest Neighbors

2501 samples
  10 predictor
   3 classes: 'high', 'low', 'medium'

Pre-processing: scaled (8)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2250, 2251, 2252, 2252, 2250, 2251, ...
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   1  0.9492427  0.9172999
   2  0.6969183  0.5071896
   3  0.7000703  0.5055052
   4  0.6597323  0.4340065
   5  0.6441465  0.4041434
   6  0.6401944  0.3933372
   7  0.6237559  0.3628774
   8  0.6077541  0.3345324
   9  0.6197750  0.3496988
  10  0.6077588  0.3275545
  11  0.6033796  0.3194782
  12  0.5877682  0.2926717
  13  0.5849650  0.2837804
  14  0.5877843  0.2889964
```

```
  15  0.5849747  0.2816312
  16  0.5853986  0.2823287
  17  0.5861795  0.2812630
  18  0.5797554  0.2690347
  19  0.5745761  0.2589911
  20  0.5785650  0.2630374

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 1.

## knn overfitting => k=3
> knnFit_choosed
k-Nearest Neighbors

2501 samples
  10 predictor
   3 classes: 'high', 'low', 'medium'

Pre-processing: scaled (8)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2251, 2252, 2250, 2252, 2251, 2251, ...
Resampling results:

  Accuracy   Kappa
  0.7053104  0.5123917

Tuning parameter 'k' was held constant at a value of 3
```

Tried 20 different running parameters(k = 1:20).

Best: k=1(overfitting) => choose k=3

> k - # Neighbors

```
> kknnFit
k-Nearest Neighbors

2711 samples
  14 predictor
   3 classes: 'high', 'low', 'medium'

Pre-processing: scaled (12)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2440, 2439, 2440, 2441, 2439, 2441, ...
Resampling results across tuning parameters:
```

```
  kmax  Accuracy   Kappa
   5    0.8885915  0.8299641
   7    0.8210974  0.7260816
   9    0.7690798  0.6458389
  11    0.7469408  0.6112530
  13    0.7266522  0.5798922
  15    0.7104405  0.5546340
  17    0.7004773  0.5391296
  19    0.6875798  0.5188131
  21    0.6857307  0.5155857
  23    0.6698716  0.4906326

Tuning parameter 'distance' was held constant at a value of 2

Tuning parameter 'kernel' was held constant at a value of optimal
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were kmax = 5, distance = 2
 and kernel = optimal.
```

Tried 10 different running parameters(max).

Best: kmax = 5, distance = 2, kernel = optimal

> Kamx - Max. #Neighbors
>
> distance - Distance
>
> kernel - Kernel

C5.0

```
> c50Fit
C5.0

2501 samples
  10 predictor
   3 classes: 'high', 'low', 'medium'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 2251, 2251, 2250, 2250, 2252, 2251, ...
Resampling results across tuning parameters:

  model  winnow  trials  Accuracy   Kappa
  rules  FALSE    1      0.5733729  0.2472232
```

```
  rules  FALSE    10      0.5721569  0.2440035
  rules  FALSE    20      0.5721569  0.2440035
  rules  FALSE    30      0.5721569  0.2440035
  rules  FALSE    40      0.5721569  0.2440035
  rules  FALSE    50      0.5721569  0.2440035
  rules  FALSE    60      0.5721569  0.2440035
  rules  FALSE    70      0.5721569  0.2440035
  rules  FALSE    80      0.5721569  0.2440035
  rules  FALSE    90      0.5721569  0.2440035
  rules  FALSE   100      0.5721569  0.2440035
  rules   TRUE     1      0.5697633  0.2403925
  rules   TRUE    10      0.5649664  0.2311124
  rules   TRUE    20      0.5649664  0.2311124
  rules   TRUE    30      0.5649664  0.2311124
  rules   TRUE    40      0.5649664  0.2311124
  rules   TRUE    50      0.5649664  0.2311124
  rules   TRUE    60      0.5649664  0.2311124
  rules   TRUE    70      0.5649664  0.2311124
  rules   TRUE    80      0.5649664  0.2311124
  rules   TRUE    90      0.5649664  0.2311124
  rules   TRUE   100      0.5649664  0.2311124
  tree   FALSE     1      0.5905715  0.2835511
  tree   FALSE    10      0.5889619  0.2795096
  tree   FALSE    20      0.5889619  0.2795096
  tree   FALSE    30      0.5889619  0.2795096
  tree   FALSE    40      0.5889619  0.2795096
  tree   FALSE    50      0.5889619  0.2795096
  tree   FALSE    60      0.5889619  0.2795096
  tree   FALSE    70      0.5889619  0.2795096
  tree   FALSE    80      0.5889619  0.2795096
  tree   FALSE    90      0.5889619  0.2795096
  tree   FALSE   100      0.5889619  0.2795096
  tree    TRUE     1      0.5957556  0.2914545
  tree    TRUE    10      0.5925460  0.2849562
  tree    TRUE    20      0.5925460  0.2849562
  tree    TRUE    30      0.5925460  0.2849562
  tree    TRUE    40      0.5925460  0.2849562
  tree    TRUE    50      0.5925460  0.2849562
  tree    TRUE    60      0.5925460  0.2849562
  tree    TRUE    70      0.5925460  0.2849562
  tree    TRUE    80      0.5925460  0.2849562
  tree    TRUE    90      0.5925460  0.2849562
  tree    TRUE   100      0.5925460  0.2849562

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were trials = 1, model = tree
```

```
and winnow = TRUE.
```

Tried 2*2*10 different running parameters(model, winnow, trials).

Best: trials = 1, model = tree, winnow = TRUE

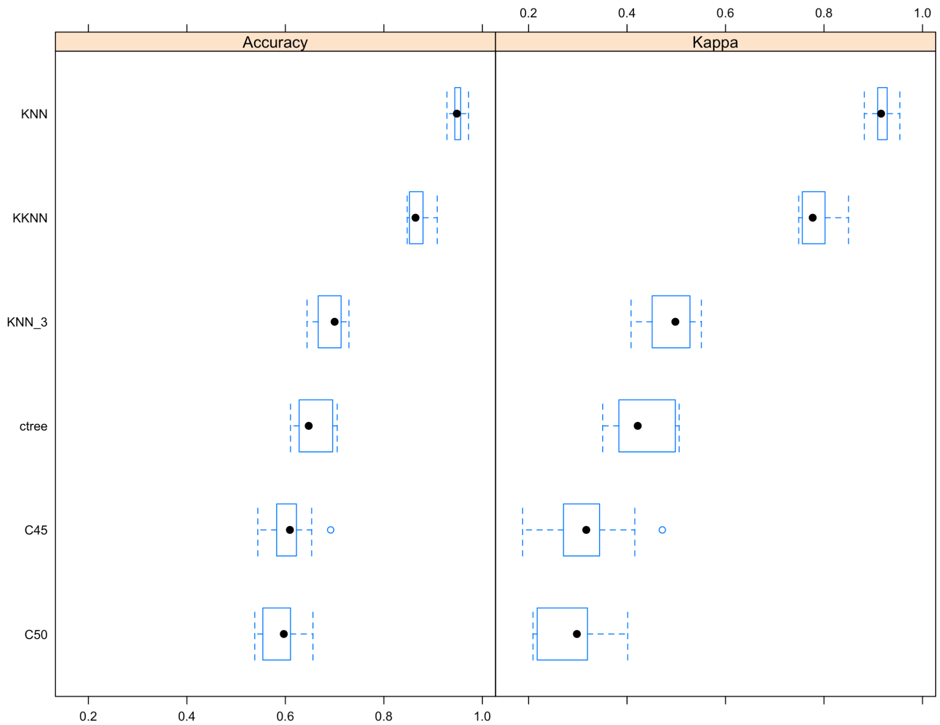
> trials - # Boosting Iterations
>
> Model - Model Type
>
> winnow - Winnow

Comparison

```
> resamps

Call:
resamples.default(x = list(ctree = ctreeFit, C45 = C45Fit, KNN
 = knnFit, KNN_3 = knnFit_choosed, KKNN = kknnFit, C50 = c50Fit))

Models: ctree, C45, KNN, KNN_3, KKNN, C50
Number of resamples: 10
Performance metrics: Accuracy, Kappa
Time estimates for: everything, final model fit
> summary(resamps)

Call:
summary.resamples(object = resamps)

Models: ctree, C45, KNN, KNN_3, KKNN, C50
Number of resamples: 10

Accuracy
           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
ctree 0.6104418 0.6296345 0.6474104 0.6584649 0.6960000 0.7051793    0
C45   0.5440000 0.5841573 0.6091968 0.6101444 0.6222460 0.6920000    0
KNN   0.9280000 0.9440558 0.9482072 0.9492427 0.9549153 0.9718876    0
KNN_3 0.6440000 0.6686747 0.7000000 0.6928878 0.7125714 0.7290837    0
KKNN  0.8473896 0.8528554 0.8642550 0.8688130 0.8787621 0.9083665    0
C50   0.5378486 0.5562249 0.5968127 0.5917601 0.6098313 0.6560000    0

Kappa
           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
ctree 0.3506211 0.3854340 0.4217182 0.4322525 0.4969543 0.5058277    0
C45   0.1878722 0.2744345 0.3172342 0.3216262 0.3430540 0.4716764    0
KNN   0.8818743 0.9089949 0.9159086 0.9172999 0.9268827 0.9540033    0
```

```
KNN_3 0.4081502 0.4523459 0.4981464 0.4906058 0.5267974 0.5509485    0
KKNN  0.7485642 0.7575549 0.7770194 0.7844456 0.8004642 0.8497945    0
C50   0.2089978 0.2181337 0.2981871 0.2877090 0.3167241 0.4013477    0

> difs

Call:
diff.resamples(x = resamps)

Models: ctree, C45, KNN, KNN_3, KKNN, C50
Metrics: Accuracy, Kappa
Number of differences: 15
p-value adjustment: bonferroni
> summary(difs)

Call:
summary.diff.resamples(object = difs)

p-value adjustment: bonferroni
Upper diagonal: estimates of the difference
Lower diagonal: p-value for H0: difference = 0

Accuracy
      ctree     C45       KNN       KNN_3     KKNN      C50
ctree            0.04832  -0.29078  -0.03442  -0.21035   0.06670
C45   0.117018            -0.33910  -0.08274  -0.25867   0.01838
KNN   1.735e-07 1.599e-08            0.25635   0.08043   0.35748
KNN_3 0.731591  0.013835  1.998e-08           -0.17593   0.10113
KKNN  3.617e-07 3.138e-07 2.422e-05 1.363e-07            0.27705
C50   0.012170  1.000000  3.559e-08 0.001074  1.415e-07

Kappa
      ctree     C45       KNN       KNN_3     KKNN      C50
ctree            0.11063  -0.48505  -0.05835  -0.35219   0.14454
C45   0.030707            -0.59567  -0.16898  -0.46282   0.03392
KNN   1.950e-07 2.970e-08            0.42669   0.13285   0.62959
KNN_3 0.658281  0.005437  1.633e-08           -0.29384   0.20290
KKNN  4.552e-07 3.860e-07 2.227e-05 1.070e-07            0.49674
C50   0.001897  1.000000  3.143e-08 0.000353  8.895e-08
```

[Analysis can see]()

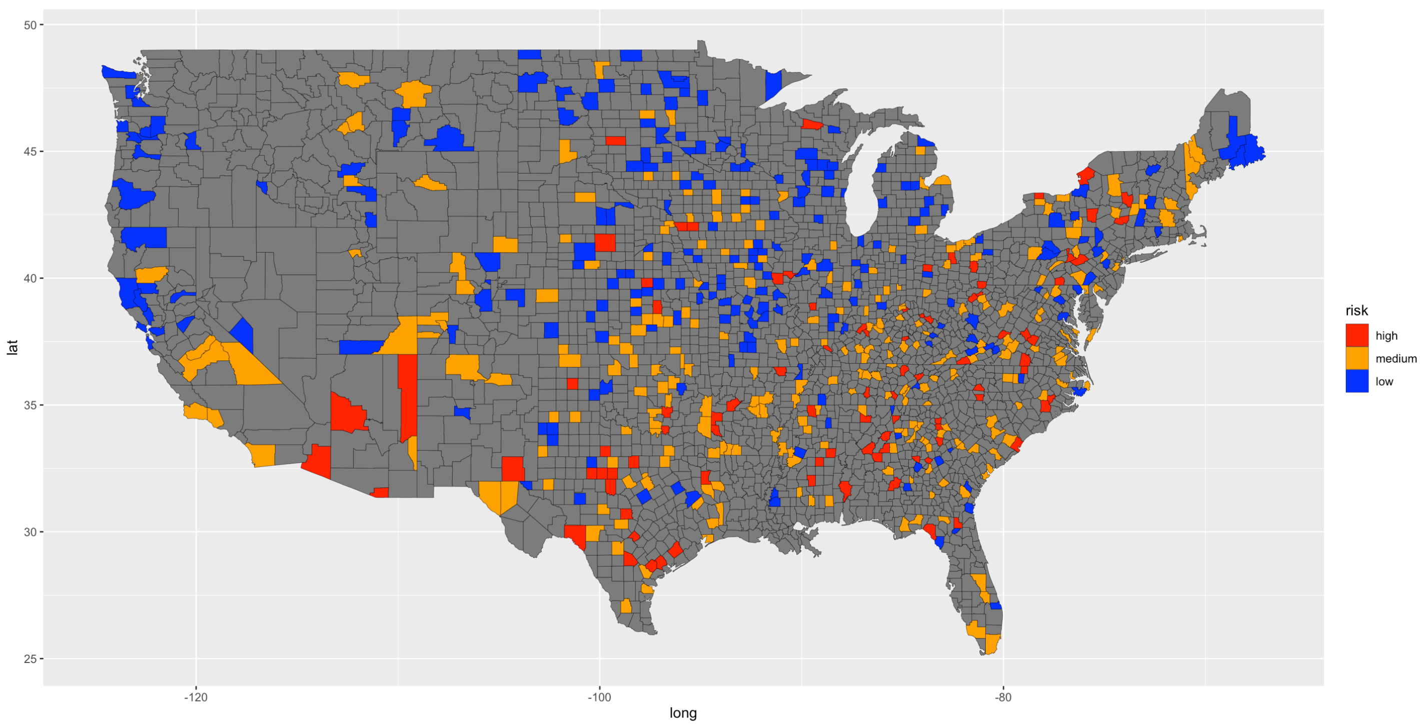


Testing

ctree

```
> confusionMatrix(data = cases_test$risk_predicted, ref = cases_test$risk)
Confusion Matrix and Statistics

          Reference
Prediction high low medium
    high     24  19     22
    low      11 113     66
    medium   58 113    198

Overall Statistics

               Accuracy : 0.5369
                 95% CI : (0.4968, 0.5765)
    No Information Rate : 0.4583
    P-Value [Acc > NIR] : 5.048e-05

                  Kappa : 0.2202

 Mcnemar's Test P-Value : 9.955e-07
```

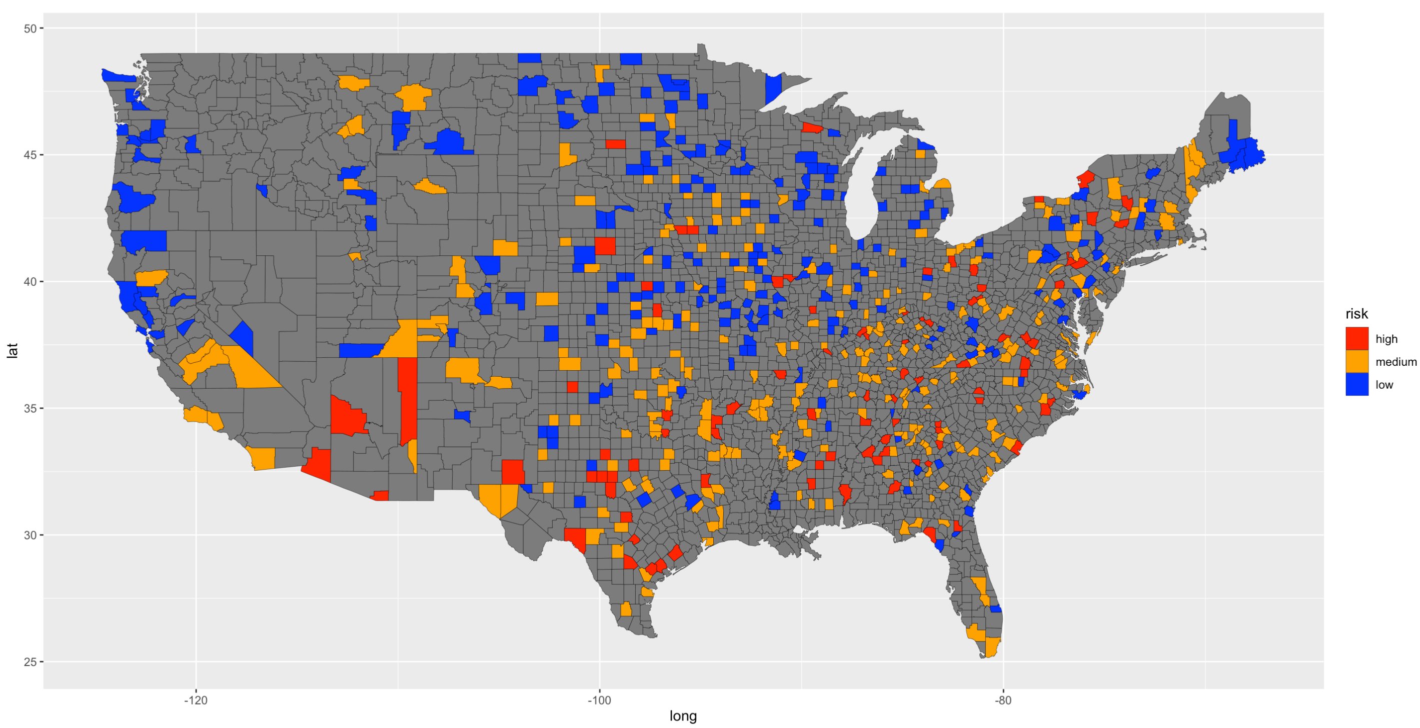
```
Statistics by Class:

                     Class: high Class: low Class: medium
Sensitivity              0.25806      0.4612         0.6923
Specificity              0.92279      0.7968         0.4941
Pos Pred Value           0.36923      0.5947         0.5366
Neg Pred Value           0.87657      0.6959         0.6549
Prevalence               0.14904      0.3926         0.4583
Detection Rate           0.03846      0.1811         0.3173
Detection Prevalence     0.10417      0.3045         0.5913
Balanced Accuracy        0.59043      0.6290         0.5932
```



C4.5

```
> confusionMatrix(data = cases_test$risk_predicted, ref = cases_test$risk)
Confusion Matrix and Statistics

          Reference
Prediction high low medium
    high     13   6      8
    low       8 123     52
    medium   72 116    226

Overall Statistics
```

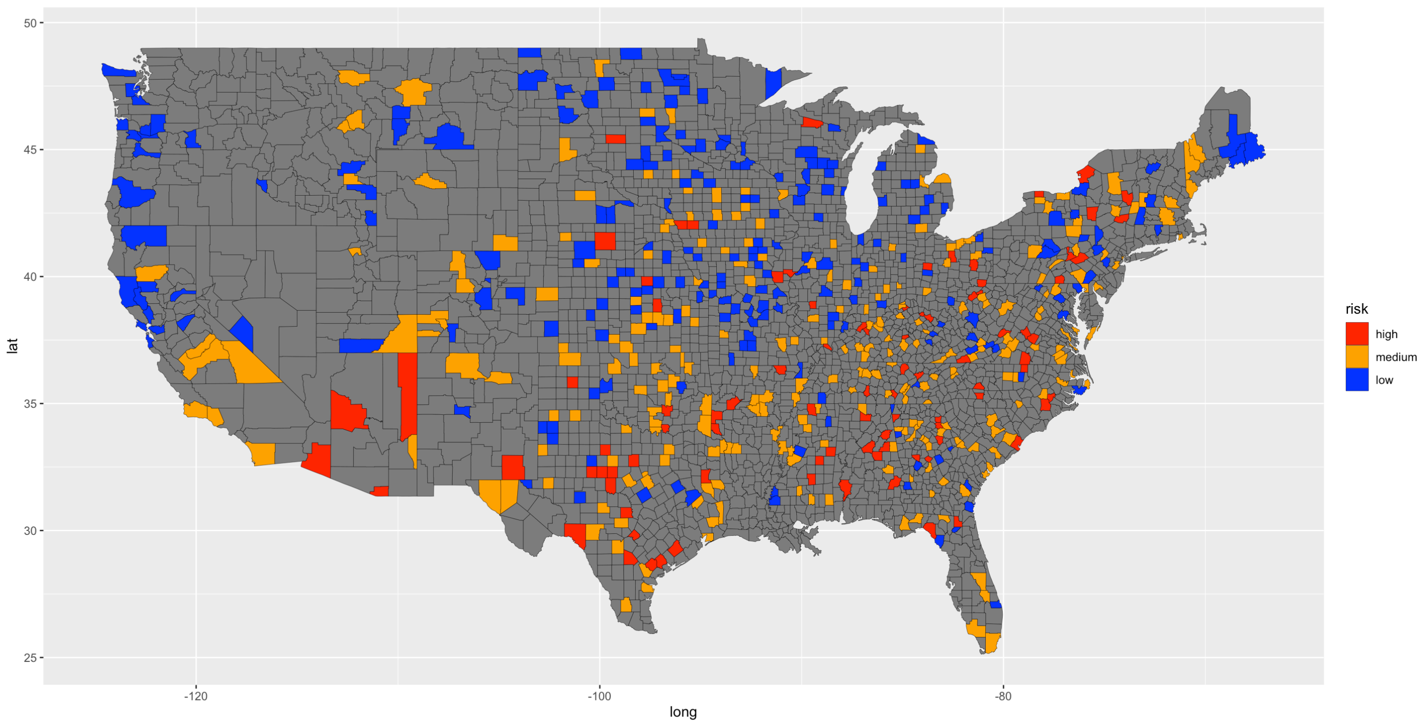
```
               Accuracy : 0.5801
                 95% CI : (0.5403, 0.6192)
    No Information Rate : 0.4583
    P-Value [Acc > NIR] : 7.018e-10

                  Kappa : 0.2689

 Mcnemar's Test P-Value : 2.362e-16

Statistics by Class:

                     Class: high Class: low Class: medium
Sensitivity              0.13978     0.5020        0.7902
Specificity              0.97363     0.8417        0.4438
Pos Pred Value           0.48148     0.6721        0.5459
Neg Pred Value           0.86600     0.7234        0.7143
Prevalence               0.14904     0.3926        0.4583
Detection Rate           0.02083     0.1971        0.3622
Detection Prevalence     0.04327     0.2933        0.6635
Balanced Accuracy        0.55671     0.6719        0.6170
```



K-Nearest Neighbors(knn & kknn)

knn

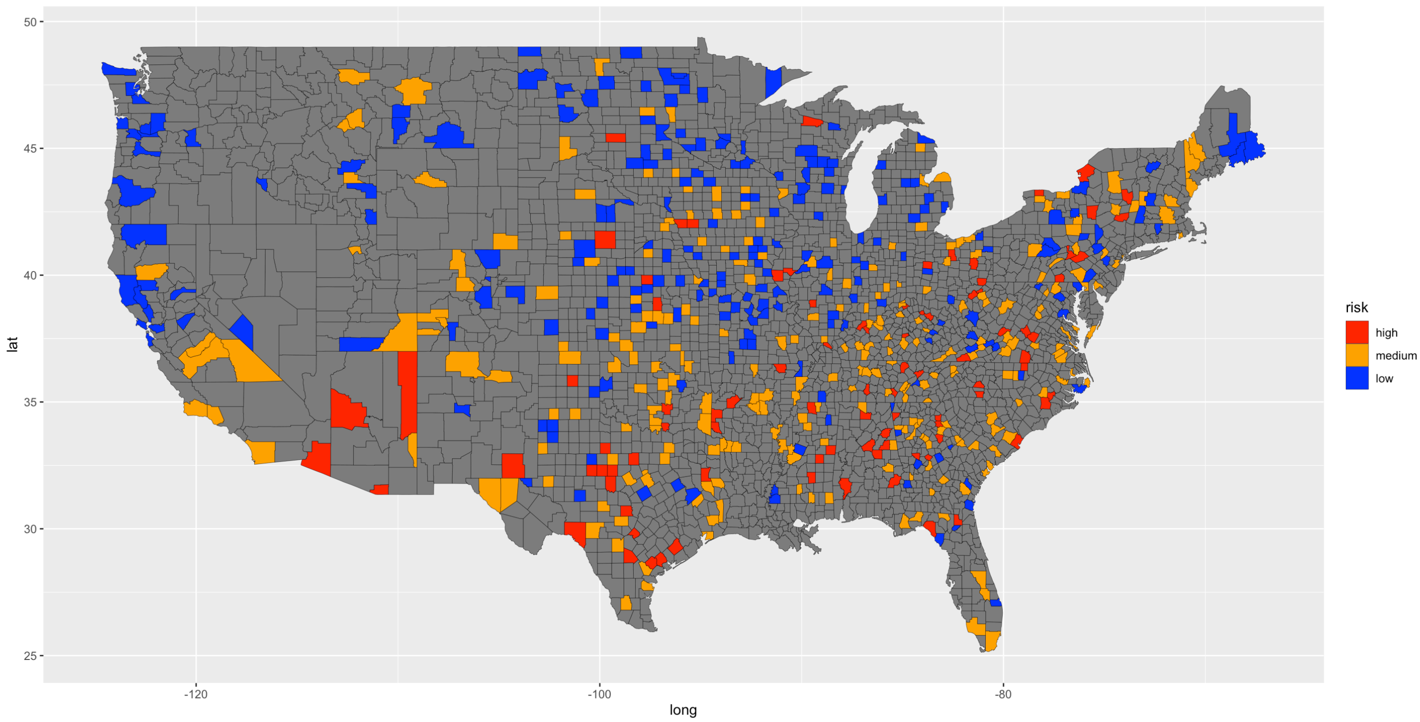
```
> confusionMatrix(data = cases_test$risk_predicted, ref = cases_test$risk)
Confusion Matrix and Statistics

          Reference
Prediction high low medium
    high     19  28     41
    low      26 123     94
    medium   48  94    151

Overall Statistics

               Accuracy : 0.4696
                 95% CI : (0.4298, 0.5096)
    No Information Rate : 0.4583
    P-Value [Acc > NIR] : 0.3005

                  Kappa : 0.1317

 Mcnemar's Test P-Value : 0.8908

Statistics by Class:

                     Class: high Class: low Class: medium
Sensitivity              0.20430     0.5020        0.5280
Specificity              0.87006     0.6834        0.5799
Pos Pred Value           0.21591     0.5062        0.5154
Neg Pred Value           0.86194     0.6798        0.5921
Prevalence               0.14904     0.3926        0.4583
Detection Rate           0.03045     0.1971        0.2420
Detection Prevalence     0.14103     0.3894        0.4696
Balanced Accuracy        0.53718     0.5927        0.5539
```



knn(k=3)

```
> confusionMatrix(data = cases_test$risk_predicted, ref = cases_test$risk)
Confusion Matrix and Statistics

          Reference
Prediction high low medium
    high     16  20     40
    low      26 138     82
    medium   51  87    164

Overall Statistics

               Accuracy : 0.5096
                 95% CI : (0.4696, 0.5495)
    No Information Rate : 0.4583
    P-Value [Acc > NIR] : 0.005758

                  Kappa : 0.1898

 Mcnemar's Test P-Value : 0.520187

Statistics by Class:

                     Class: high Class: low Class: medium
Sensitivity              0.17204     0.5633        0.5734
```

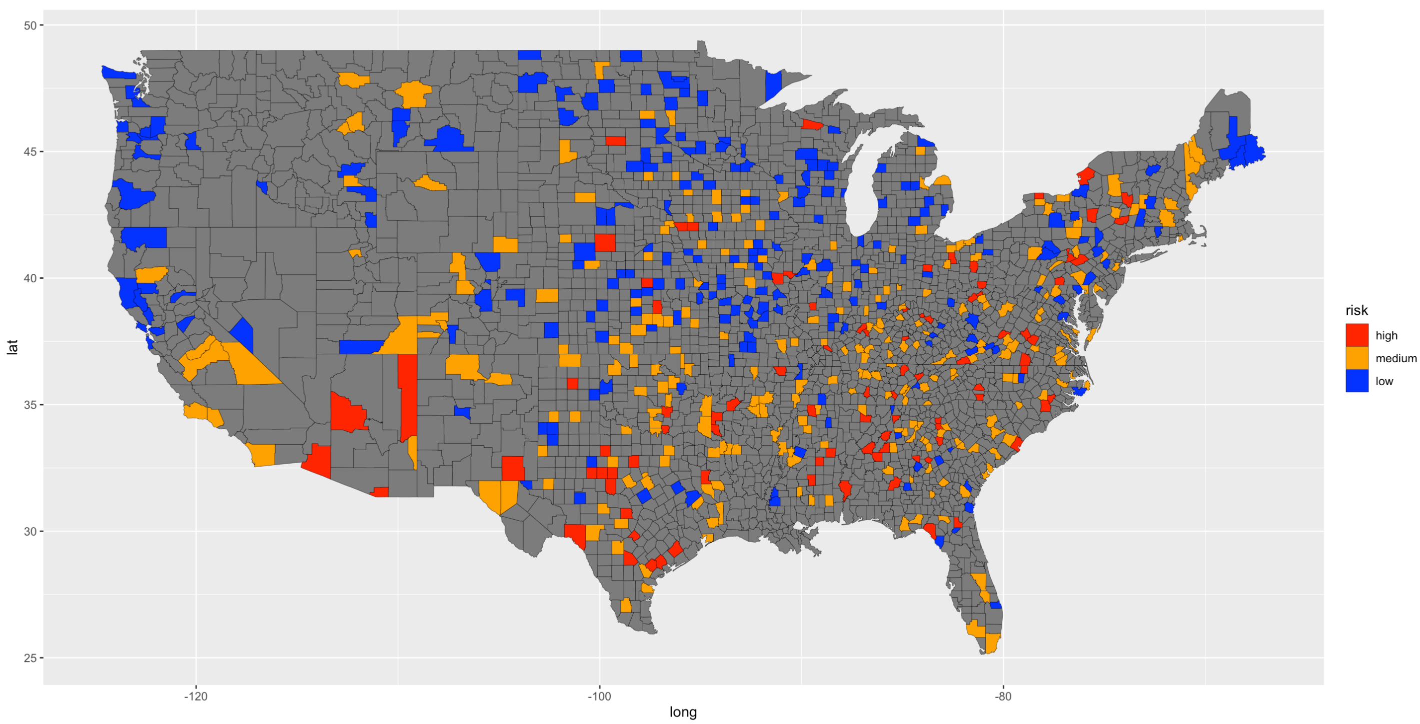
```
Specificity                 0.88701    0.7150      0.5917
Pos Pred Value              0.21053    0.5610      0.5430
Neg Pred Value              0.85949    0.7169      0.6211
Prevalence                  0.14904    0.3926      0.4583
Detection Rate              0.02564    0.2212      0.2628
Detection Prevalence        0.12179    0.3942      0.4840
Balanced Accuracy           0.52952    0.6392      0.5826
```



kknn

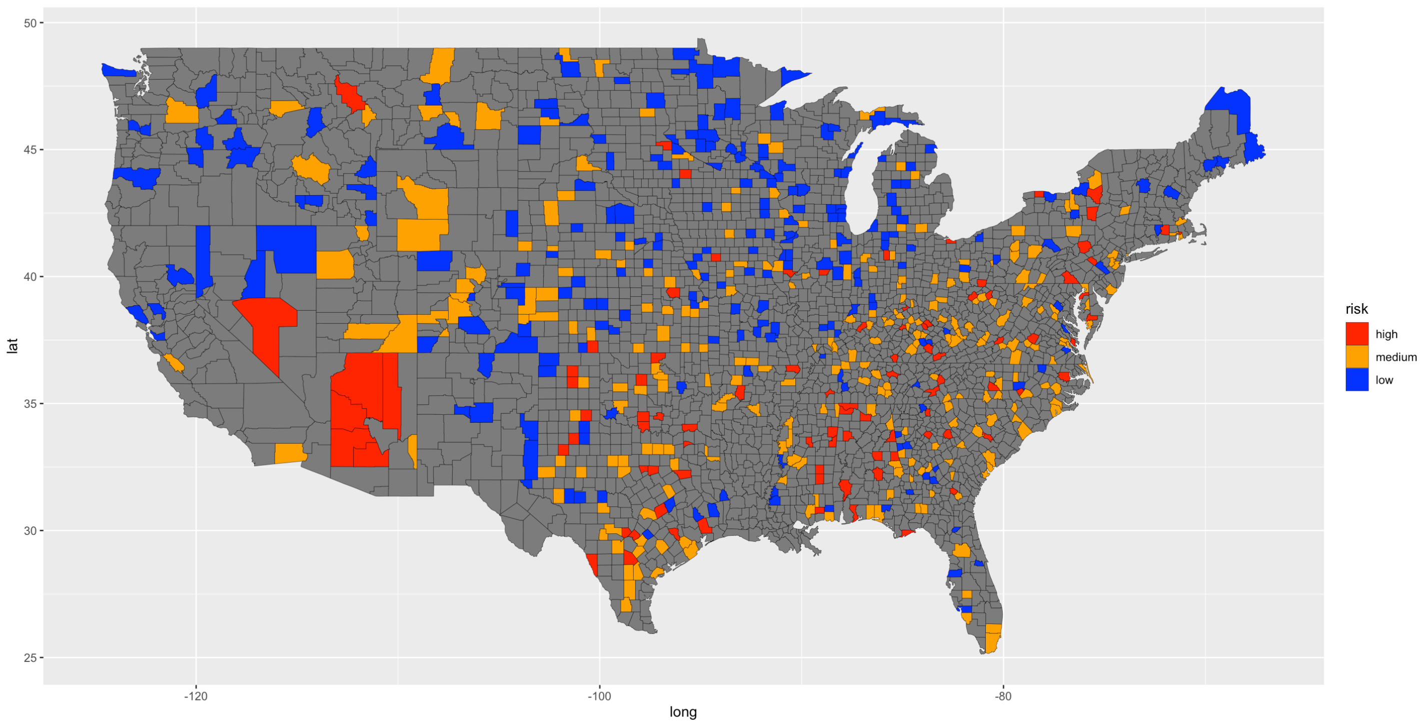
```
> confusionMatrix(data = cases_test$risk_predicted, ref = cases_test$risk)
Confusion Matrix and Statistics

          Reference
Prediction high low medium
    high     53   9     12
    low       8 146     27
    medium   32  90    247

Overall Statistics

               Accuracy : 0.7147
                 95% CI : (0.6776, 0.7499)
    No Information Rate : 0.4583
    P-Value [Acc > NIR] : < 2.2e-16
```

```
                  Kappa : 0.5225

 Mcnemar's Test P-Value : 2.375e-09

Statistics by Class:

                     Class: high Class: low Class: medium
Sensitivity              0.56989     0.5959        0.8636
Specificity              0.96045     0.9077        0.6391
Pos Pred Value           0.71622     0.8066        0.6694
Neg Pred Value           0.92727     0.7765        0.8471
Prevalence               0.14904     0.3926        0.4583
Detection Rate           0.08494     0.2340        0.3958
Detection Prevalence     0.11859     0.2901        0.5913
Balanced Accuracy        0.76517     0.7518        0.7513
```



C5.0

```
> confusionMatrix(data = cases_test$risk_predicted, ref = cases_test$risk)
Confusion Matrix and Statistics

          Reference
Prediction high low medium
    high      0   0      0
    low      24 134     90
    medium   69 111    196

Overall Statistics

               Accuracy : 0.5288
                 95% CI : (0.4888, 0.5686)
    No Information Rate : 0.4583
    P-Value [Acc > NIR] : 0.0002436

                  Kappa : 0.1702

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: high Class: low Class: medium
Sensitivity                0.000     0.5469        0.6853
Specificity                1.000     0.6992        0.4675
Pos Pred Value               NaN     0.5403        0.5213
Neg Pred Value             0.851     0.7048        0.6371
Prevalence                 0.149     0.3926        0.4583
Detection Rate             0.000     0.2147        0.3141
Detection Prevalence       0.000     0.3974        0.6026
Balanced Accuracy          0.500     0.6231        0.5764
```



References

[CDC]https://www.cdc.gov/coronavirus/2019-ncov/science/community-levels.html

[An R Companion for Introduction to Data Mining]https://mhahsler.github.io/Introduction_to_Data_Mining_R_Examples/book/classification-alternative-techniques.html