Introduction to Classification

Stat 220

Bastola

February 25 2022

Classification

Predicting what category a (future) observation falls into

Loading [MathJax]/jax/output/CommonHTML/jax.js

Binary Classification

We focus on the setting of binary classification where only two classes are involved (e.g., a diagnosis of either healthy or diseased)

Netflix example

> Just today, Netflix emailed subscribers notifying them of a price increase for more great entertainment

Will customers cancel their accounts?

Loading [MathJax]/jax/output/CommonHTML/jax.js

Netflix example

Possible predictor variables (a.k.a. features, attributes, inputs, independent variables)

- job status
- age of account
- age
- payment method
- location
- content ratings
- viewing habits/history
- platforms used (e.g. smartphone, Smart TV, ipad, etc.)
- competition
- `#CancelNetflix` movement
- ...and more...

More classification examples

- **Astronomy:** Whether an exoplanet is habitable (or not)
- **Filtering:** Identify spam emails
- **Medicine:** Use lab results to determine who has a disease (or not)
- **Product preference:** make product recommendations based on past purchases
- Social services: Identify which Child Welfare calls to screen in for further investigation
- Recidivism: Predict which defendants or paroles will commit another violent crime.

Let's talk about forest fires

> It would be nice to predict where the next forest fire will occur

- Dataset contains a culmination of forest fire observations
- Based on two regions of Algeria: the Bejaia region and the Sidi Bel-Abbes region.
- Timeline is from June 2012 to September 2012

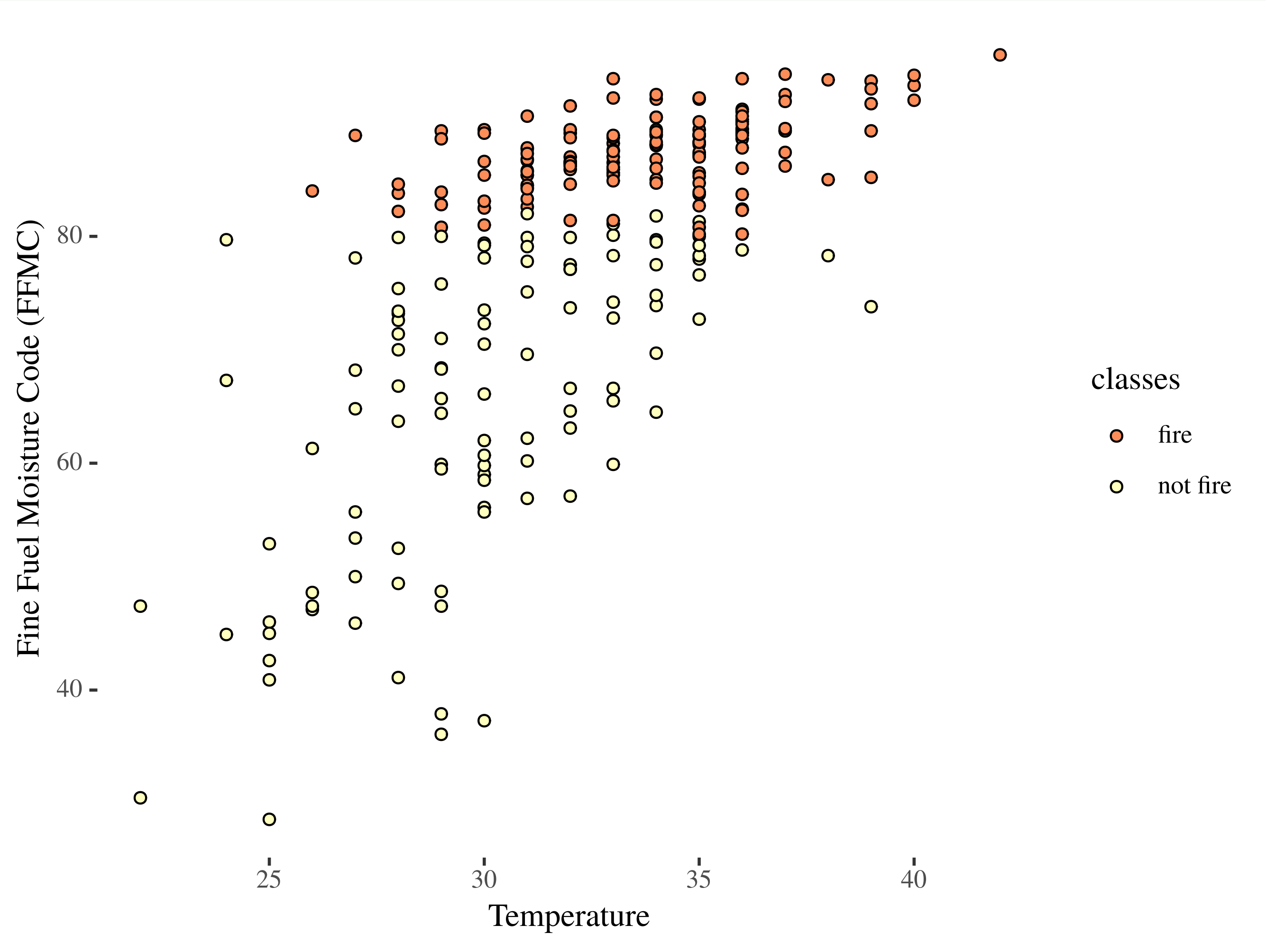
Clice here to learn more about the dataset

Variable	Description
`Date`	(DD/MM/YYYY) Day, month, year (2012)
`Temp`	Noon temperature in Celsius degrees: 22 to 42
`RH`	Relative Humidity in percentage: 21 to 90
`Ws`	Wind speed in km/h: 6 to 29
`Rain`	Daily total rain in mm: 0 to 16.8
`Fine Fuel Moisture Code (FFMC) index`	28.6 to 92.5
`Duff Moisture Code (DMC) index`	1.1 to 65.9
`Drought Code (DC) index`	7 to 220.4
`Initial Spread Index (ISI) index`	0 to 18.5
`Buildup Index (BUI) index`	1.1 to 68
`Fire Weather Index (FWI) index`	0 to 31.1
`Classes`	Two classes, namely **fire** and **not fire**

Glimpse of the data

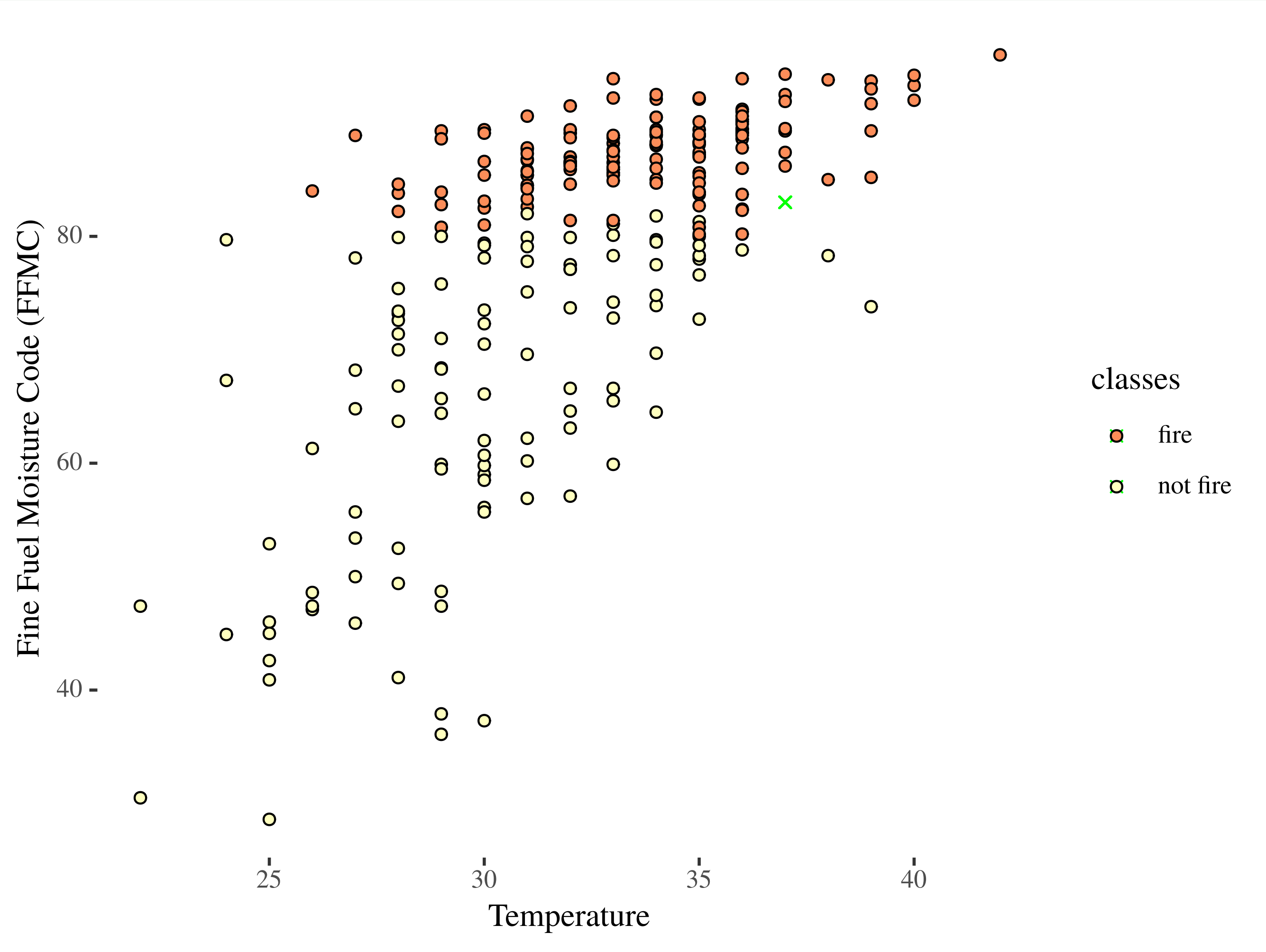
```
glimpse(fire)
Rows: 243
Columns: 14
$ day         <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,…
$ month       <dbl> 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6…
$ year        <dbl> 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012…
$ temperature <dbl> 29, 29, 26, 25, 27, 31, 33, 30, 25, 28, 31, 26, 27, 30, 28…
$ rh          <dbl> 57, 61, 82, 89, 77, 67, 54, 73, 88, 79, 65, 81, 84, 78, 80…
$ ws          <dbl> 18, 13, 22, 13, 16, 14, 13, 15, 13, 12, 14, 19, 21, 20, 17…
$ rain        <dbl> 0.0, 1.3, 13.1, 2.5, 0.0, 0.0, 0.0, 0.0, 0.2, 0.0, 0.0, 0.…
$ ffmc        <dbl> 65.7, 64.4, 47.1, 28.6, 64.8, 82.6, 88.2, 86.6, 52.9, 73.2…
$ dmc         <dbl> 3.4, 4.1, 2.5, 1.3, 3.0, 5.8, 9.9, 12.1, 7.9, 9.5, 12.5, 1…
$ dc          <dbl> 7.6, 7.6, 7.1, 6.9, 14.2, 22.2, 30.5, 38.3, 38.8, 46.3, 54…
$ isi         <dbl> 1.3, 1.0, 0.3, 0.0, 1.2, 3.1, 6.4, 5.6, 0.4, 1.3, 4.0, 4.8…
$ bui         <dbl> 3.4, 3.9, 2.7, 1.7, 3.9, 7.0, 10.9, 13.5, 10.5, 12.6, 15.8…
$ fwi         <dbl> 0.5, 0.4, 0.1, 0.0, 0.5, 2.5, 7.2, 7.1, 0.3, 0.9, 5.6, 7.1…
$ classes     <chr> "not fire", "not fire", "not fire", "not fire", "not fire"…
```

Scatterplot



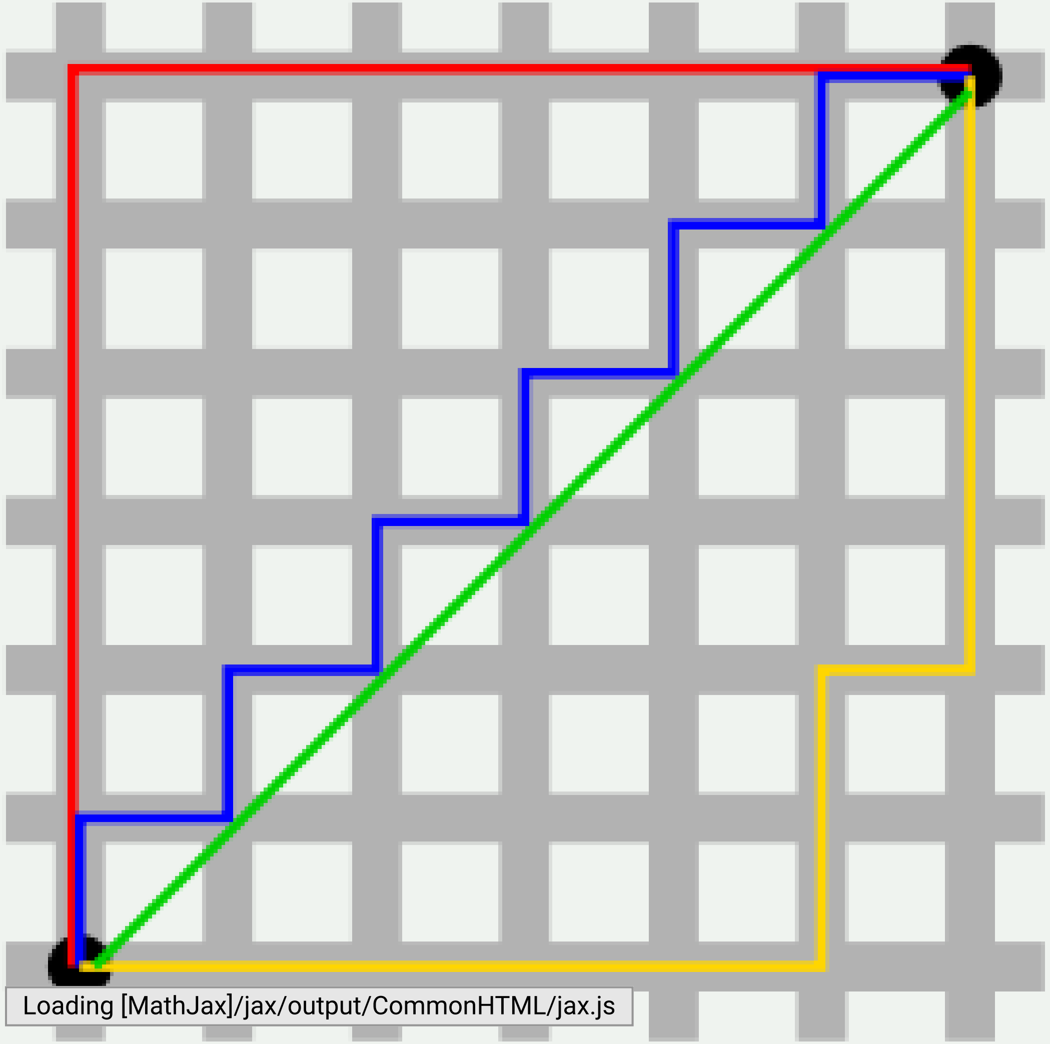
Loading [MathJax]/jax/output/CommonHTML/jax.js

How can we classify a new observation?



Loading [MathJax]/jax/output/CommonHTML/jax.js

Calculating distance



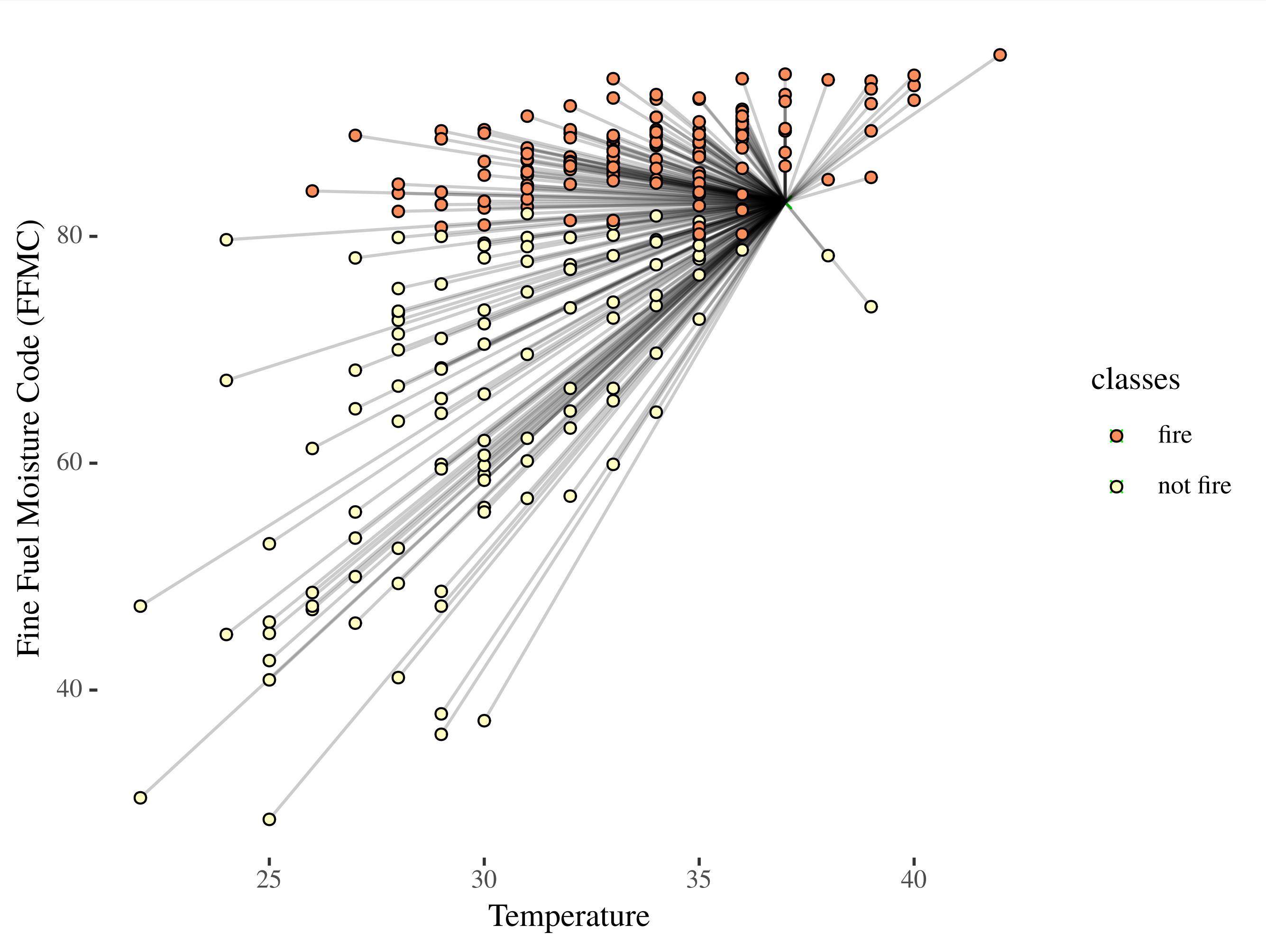
Euclidean distance: the straight line distance between two points on the x-y plane with coordinates (x_a, y_a) and (x_b, y_b)

$$\text{Distance} = \sqrt{\left(x_a - x_b\right)^2 + \left(y_a - y_b\right)^2}$$

Manhattan distance: the "taxi-cab" distance between two points on the x-y plane

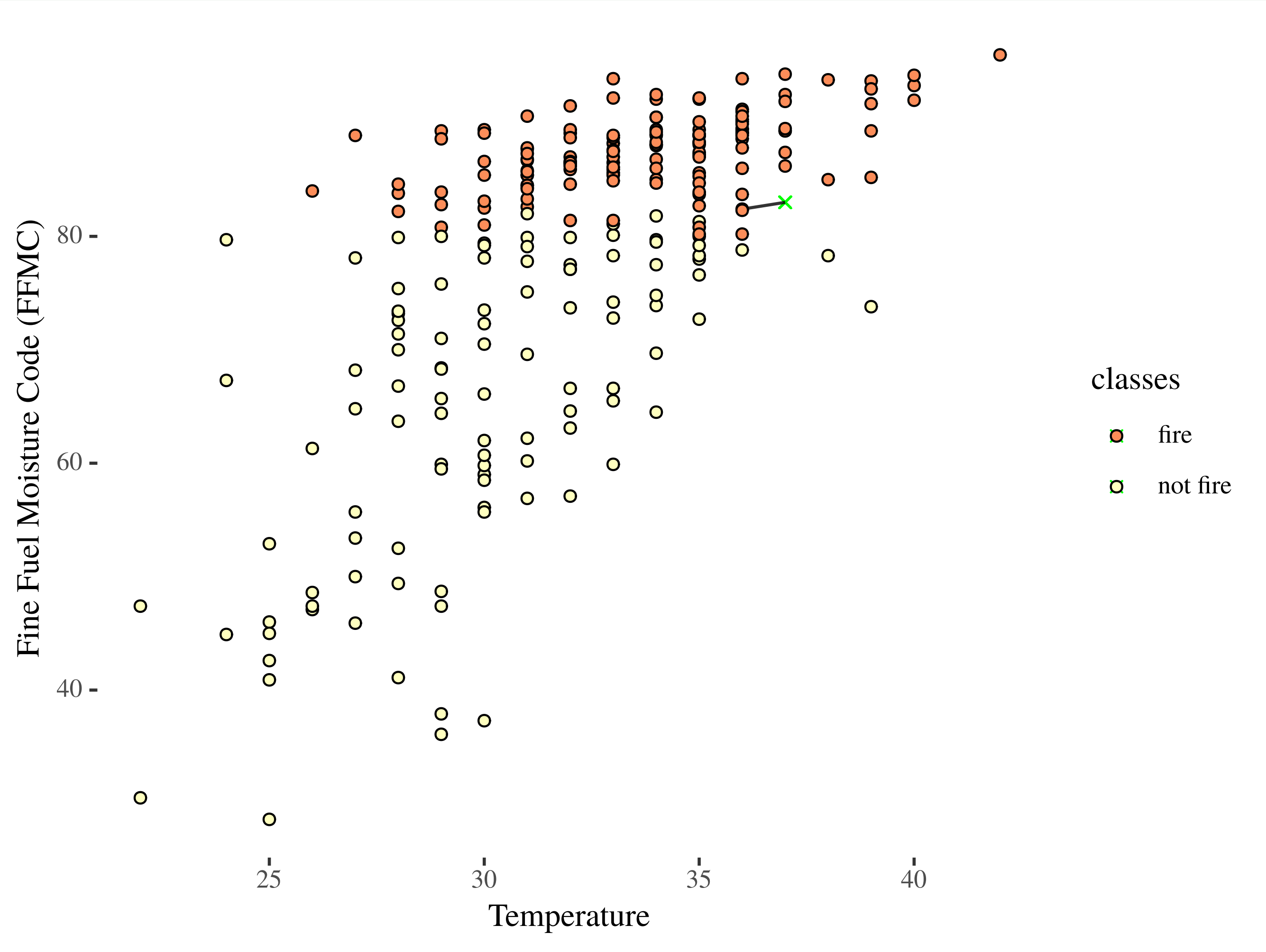
$$\text{Distance} = \left|x_a - x_b\right| + \left|y_a - y_b\right|$$

Looking at Euclidean distance



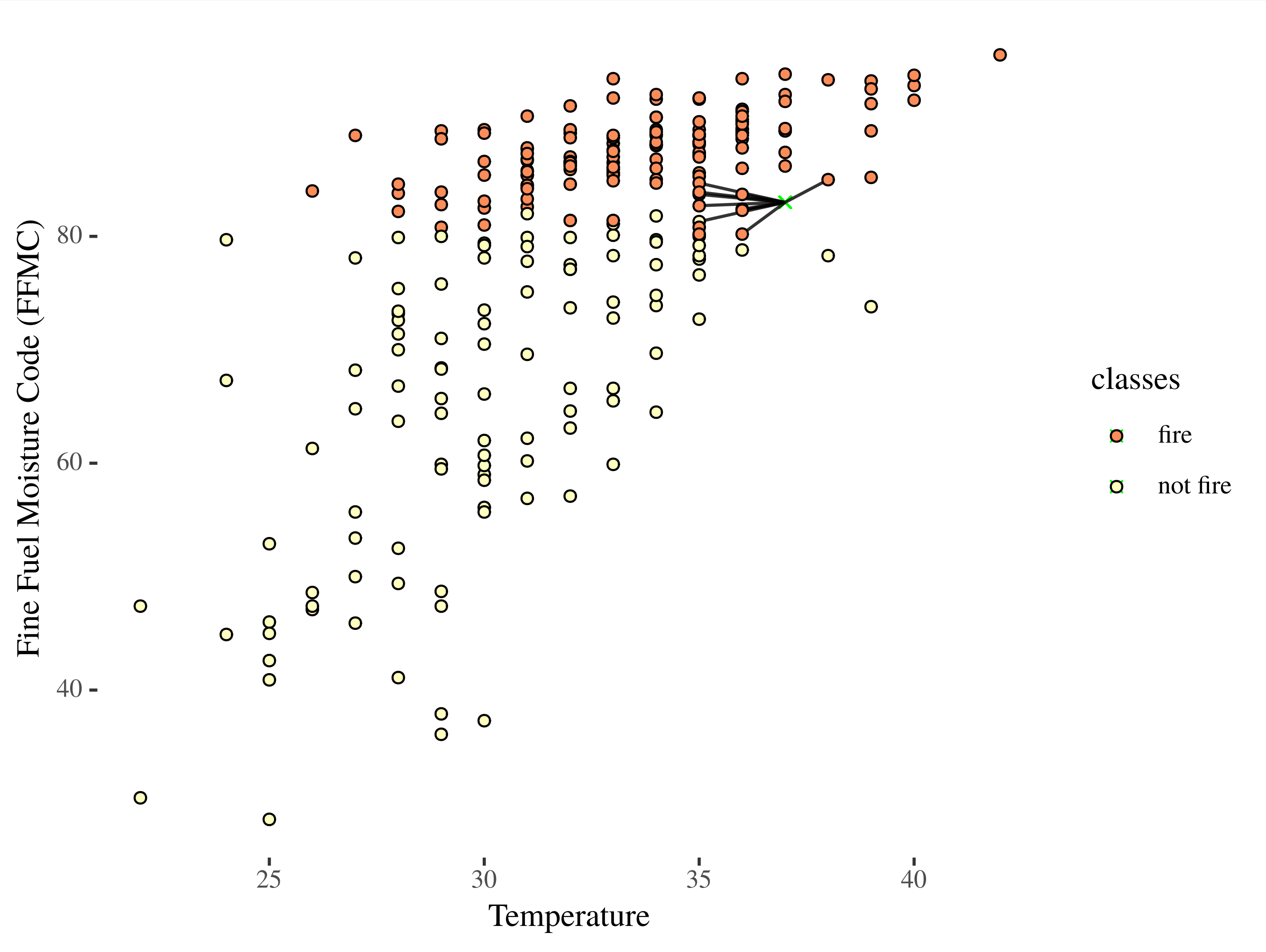
Loading [MathJax]/jax/output/CommonHTML/jax.js

1-Nearest Neighbor (NN)



Loading [MathJax]/jax/output/CommonHTML/jax.js

10-NN



Wait, something is not quite right..

Loading [MathJax]/jax/output/CommonHTML/jax.js

Need to standardize data

```
standardize <- function(x, na.rm = FALSE) {
  (x - mean(x, na.rm = na.rm)) / sd(x, na.rm = na.rm)
}
```

- Predictors with larger variation will have larger influence on which cases are "nearest" neighbors
- Methods relying on distance can be sensitive (i.e. not invariant) to the scale of the predictors
- Standardizing only shifts and rescales the variable, it doesn't change the shape of the distribution

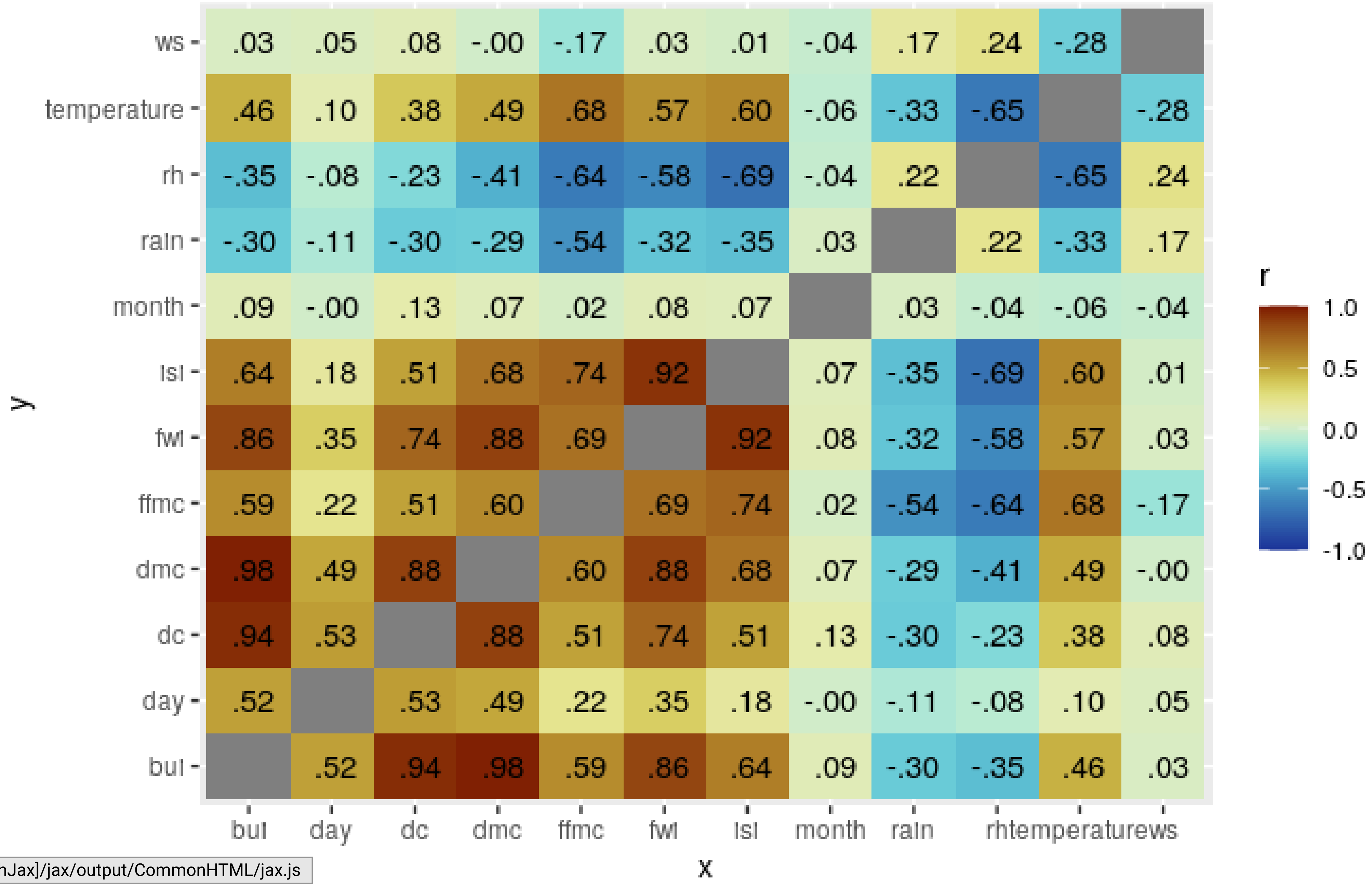

use within mutate()

Standardized data

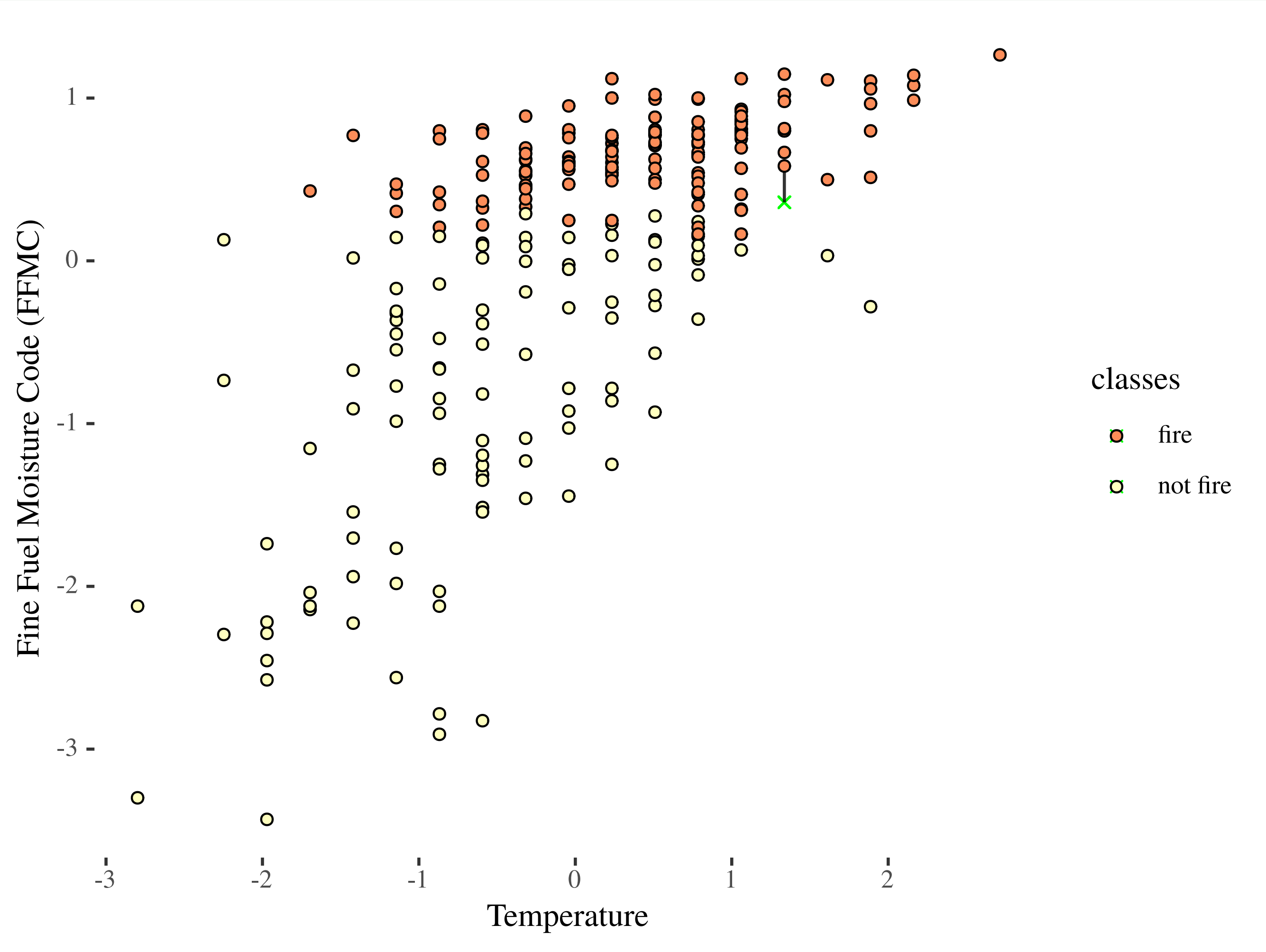
```
fire1 <- fire %>% mutate(across(where(is.numeric), standardize))
fire1 %>% summary()
      day               month               year       temperature
 Min.   :-1.66935   Min.   :-1.3474   Min.   : NA   Min.   :-2.79828
 1st Qu.:-0.87772   1st Qu.:-0.4504   1st Qu.: NA   1st Qu.:-0.59323
 Median : 0.02699   Median : 0.4467   Median : NA   Median :-0.04197
 Mean   : 0.00000   Mean   : 0.0000   Mean   :NaN   Mean   : 0.00000
 3rd Qu.: 0.81862   3rd Qu.: 0.4467   3rd Qu.: NA   3rd Qu.: 0.78492
 Max.   : 1.72334   Max.   : 1.3437   Max.   : NA   Max.   : 2.71434
                                      NA's   :243
       rh                 ws                rain              ffmc
 Min.   :-2.76778   Min.   :-3.3769   Min.   :-0.3809   Min.   :-3.4316
 1st Qu.:-0.64345   1st Qu.:-0.5313   1st Qu.:-0.3809   1st Qu.:-0.4176
 Median : 0.06466   Median :-0.1757   Median :-0.3809   Median : 0.3803
 Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.77278   3rd Qu.: 0.5357   3rd Qu.:-0.1313   3rd Qu.: 0.7288
 Max.   : 1.88552   Max.   : 4.8041   Max.   : 8.0057   Max.   : 1.2654

      dmc                dc                isi               bui
 Min.   :-1.1281   Min.   :-0.8923   Min.   :-1.1416   Min.   :-1.0957
 1st Qu.:-0.7166   1st Qu.:-0.7779   1st Qu.:-0.8046   1st Qu.:-0.7514
 Median :-0.2728   Median :-0.3426   Median :-0.2991   Median :-0.3015
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.4938   3rd Qu.: 0.4126   3rd Qu.: 0.6036   3rd Qu.: 0.4188
 Max.   : 4.1329   Max.   : 3.5868   Max.   : 3.4321   Max.   : 3.6061

      fwi             classes
 Min.   :-0.9455   Length:243
 1st Qu.:-0.8515   Class :character
 Median :-0.3811   Mode  :character
 Mean   : 0.0000
 3rd Qu.: 0.5933
 Max.   : 3.2342
```

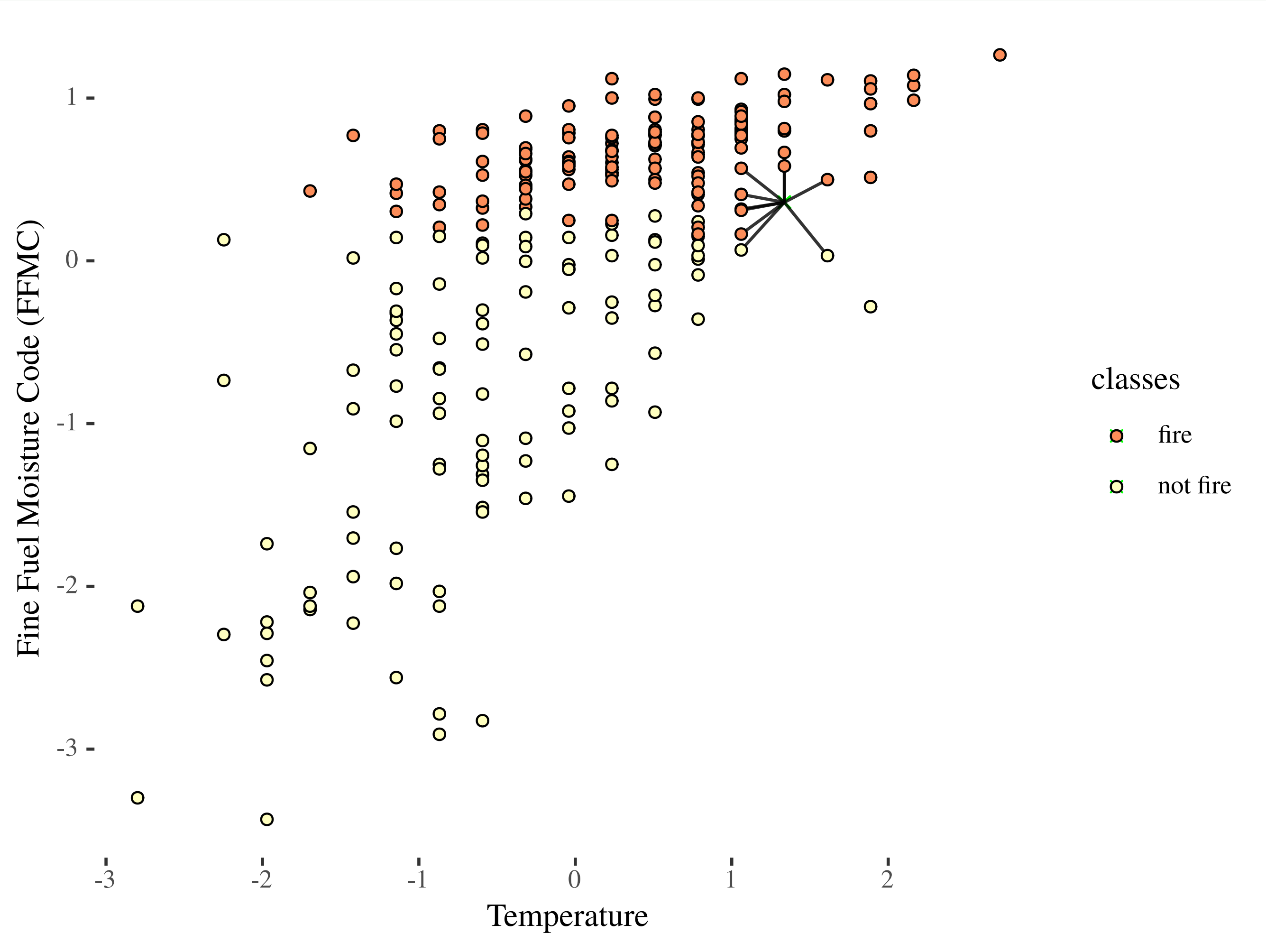

ws .03 .05 .08 -.00 -.17 .03 .01 -.04 .17 .24 -.28
temperature .46 .10 .38 .49 .68 .57 .60 -.06 -.33 -.65 -.28
rh -.35 -.08 -.23 -.41 -.64 -.58 -.69 -.04 .22 -.65 .24
rain -.30 -.11 -.30 -.29 -.54 -.32 -.35 .03 .22 -.33 .17
month .09 -.00 .13 .07 .02 .08 .07 .03 -.04 -.06 -.04
isi .64 .18 .51 .68 .74 .92 .07 -.35 -.69 .60 .01
fwi .86 .35 .74 .88 .69 .92 .08 -.32 -.58 .57 .03
ffmc .59 .22 .51 .60 .69 .74 .02 -.54 -.64 .68 -.17
dmc .98 .49 .88 .60 .88 .68 .07 -.29 -.41 .49 -.00
dc .94 .53 .88 .51 .74 .51 .13 -.30 -.23 .38 .08
day .52 .53 .49 .22 .35 .18 -.00 -.11 -.08 .10 .05
bui .52 .94 .98 .59 .86 .64 .09 -.30 -.35 .46 .03
y
bui day dc dmc ffmc fwi isi month rain rh temperature ws
x
r
1.0
0.5
0.0
-0.5
-1.0

1-NN again



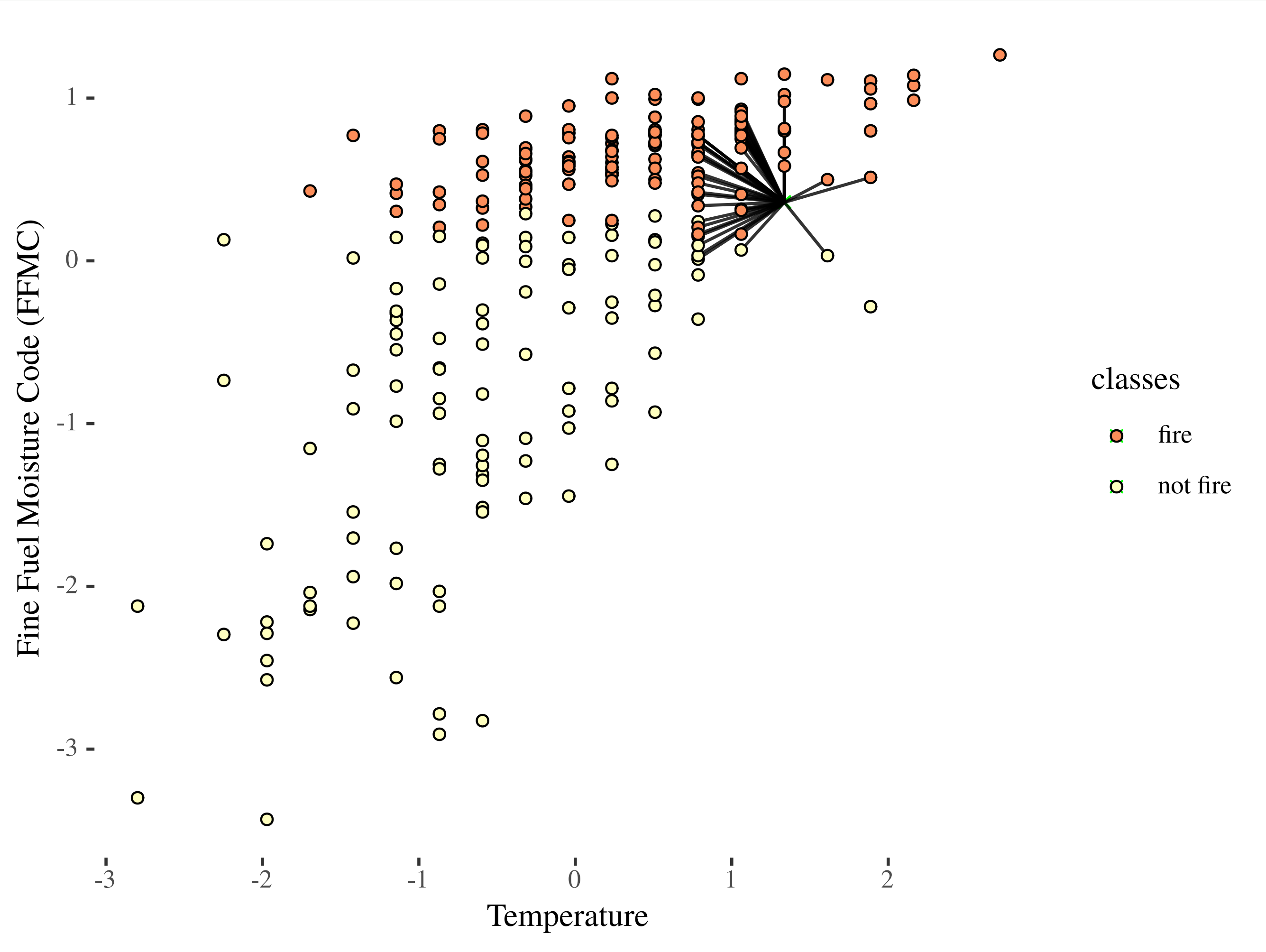
Loading [MathJax]/jax/output/CommonHTML/jax.js

10-NN again



Loading [MathJax]/jax/output/CommonHTML/jax.js

50-NN again

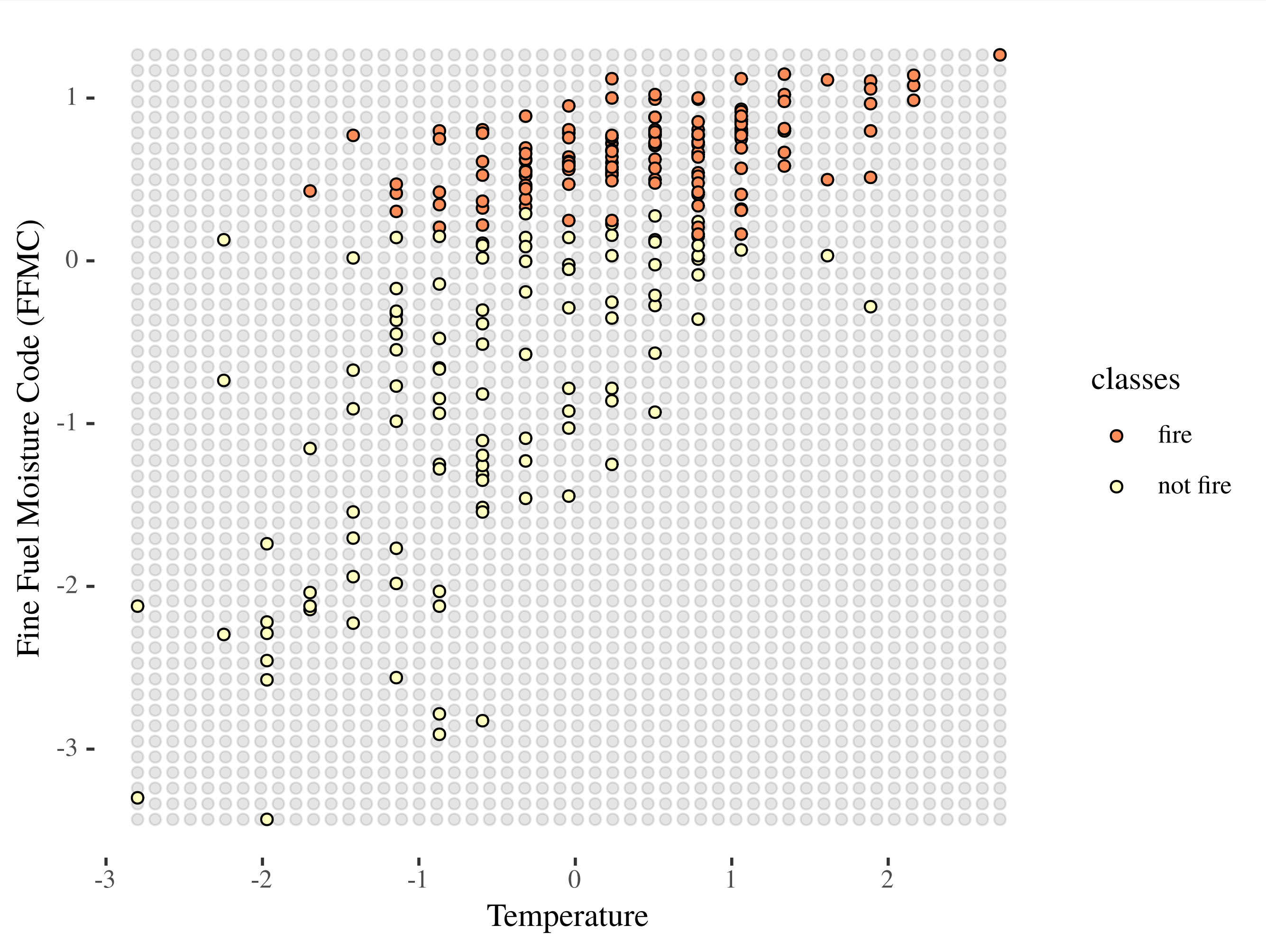


Loading [MathJax]/jax/output/CommonHTML/jax.js

Visualizing the decision boundary

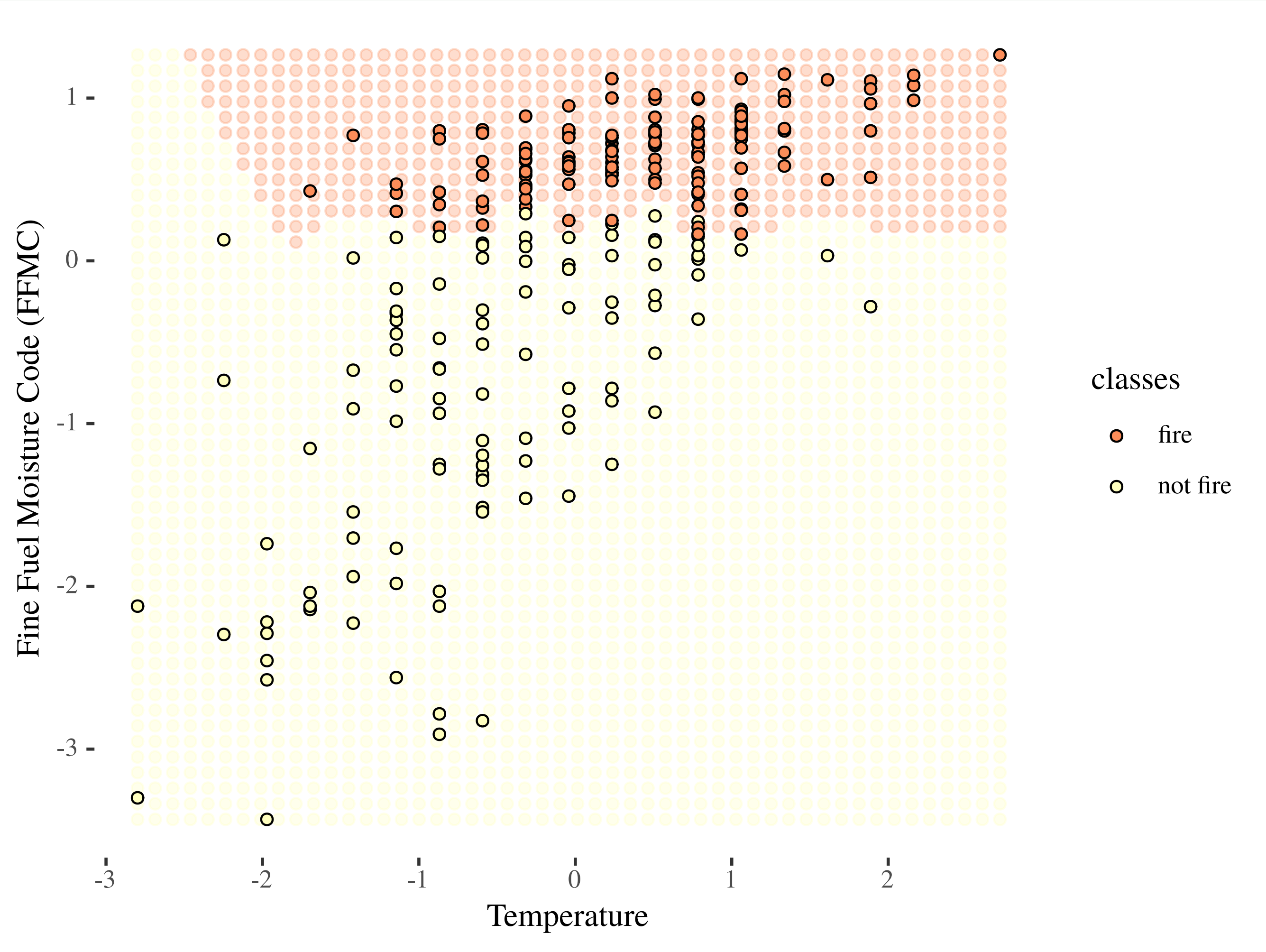
- We can map out the region in feature-space where the classifier would predict 'fire', and the kinds where it would predict 'not fire'
- There is some boundary between the two, where points on one side of the boundary will be classified 'fire' and points on the other side will be classified 'not fire'
- This boundary is called **decision boundary**

Visualizing the decision boundary



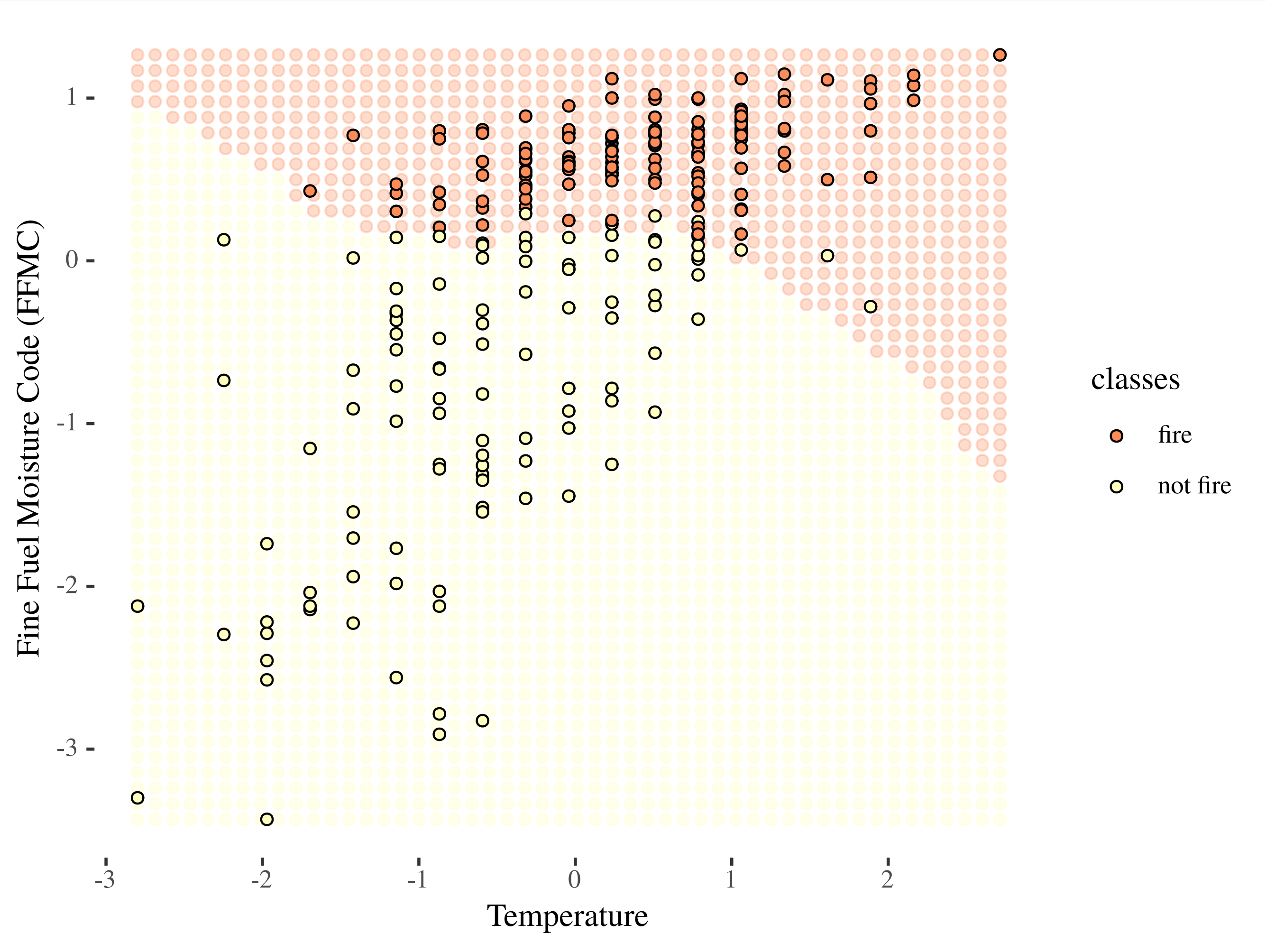
Loading [MathJax]/jax/output/CommonHTML/jax.js

1-NN decision boundary



Loading [MathJax]/jax/output/CommonHTML/jax.js

25-NN decision boundary



Loading [MathJax]/jax/output/CommonHTML/jax.js

a collection of packages for modeling and machine learning using tidyverse principles

1. Load data and convert types

```
fire_raw <- read_csv("https://raw.githubusercontent.com/deepbas/statdatasets/main/Algeriafires.c
  clean_names() %>% na.omit() %>%
  mutate(classes = as_factor(classes)) %>%
  mutate_at(c(10,13), as.numeric) %>%
  select(temperature, ffmc, classes)
```

```
head(fire_raw)
# A tibble: 6 × 3
  temperature  ffmc classes
        <dbl> <dbl> <fct>
1          29  65.7 not fire
2          29  64.4 not fire
3          26  47.1 not fire
4          25  28.6 not fire
5          27  64.8 not fire
6          31  82.6 fire
```

Loading [MathJax]/jax/output/CommonHTML/jax.js

2. Create a recipe for data preprocessing

```
fire_recipe <- recipe(classes ~ ., data = fire_raw) %>%
  step_scale(all_predictors()) %>%
  step_center(all_predictors()) %>%
  prep()
```

Loading [MathJax]/jax/output/CommonHTML/jax.js

3. Apply the recipe to the data set

```
fire_scaled <- bake(fire_recipe, fire_raw)
```

```
# A tibble: 243 × 3
   temperature   ffmc classes
         <dbl>  <dbl> <fct>
 1      -0.869 -0.846 not fire
 2      -0.869 -0.937 not fire
 3      -1.70  -2.14  not fire
 4      -1.97  -3.43  not fire
 5      -1.42  -0.909 not fire
 6      -0.318  0.332 fire
 7       0.234  0.722 fire
 8      -0.593  0.610 fire
 9      -1.97  -1.74  not fire
10      -1.14  -0.324 not fire
# … with 233 more rows
```

Loading [MathJax]/jax/output/CommonHTML/jax.js

4. Create a model specification

```r
knn_spec <- nearest_neighbor(mode = "classification",
                             engine = "kknn",
                             weight_func = "rectangular",
                             neighbors = 5)
```

5. Fit the model on the preprocessed data

```
knn_fit <- knn_spec %>%
 fit(classes ~ ., data = fire_scaled)
```

Loading [MathJax]/jax/output/CommonHTML/jax.js

6. Classify

Suppose we get two new observations, use predict to classify the observations

```
# Data frame/tibble of new observations
new_observations <- tibble(temperature = c(1, 2), ffmc = c(-1, 1))
```

```
# Making classifications (i.e. predictions)
predict(knn_fit, new_data = new_observations)
# A tibble: 2 × 1
  .pred_class
  <fct>
1 not fire
2 fire
```

Loading [MathJax]/jax/output/CommonHTML/jax.js

Further Practice: Pima Indians Diabetes

Owned by the National Institute of Diabetes and Digestive and Kidney Diseases

- A data frame with 768 observations on 9 variables.
- We have the lab results of 158 patients, including whether they have CKD
- Response variable: `diabetes` = `pos`, `neg`
- Predictor variables: *pregnant, glucose, pressure, triceps, insulin, mass, pedigree, age*

Click here for source

Variables

Variable	Description
`pregnant`	Number of times pregnant
`glucose`	Plasma glucose concentration (glucose tolerance test)
`pressure`	Diastolic blood pressure (mm Hg)
`triceps`	Triceps skinfold thickness (mm)
`insulin`	2-Hour serum insulin (mu U/ml)
`mass`	Body mass index (weight in kg/(height in m)\$^2$)
`pedigree`	Diabetes pedigree function
`age`	Age (years)
`diabetes`	diabetes case (pos/neg)

Your Turn 1

10:00

Please clone the repository on classification intro to your local folder.

```
library(mlbench)
data(PimaIndiansDiabetes2)
```

a. Tidy the data to make it ready for analysis

b. Make a correlation plot of the numerical variables in the dataset

c. Which pair of variables in the dataset have the largest correlation?

d. Using `parsnip` package, perform all the steps involved in classifying whether a patient with certain glucose and insulin would have diabetes or not.