# Chapter 2

# The Basics of Classification

Classification is a form of supervised learning in which our goal is to learn a mapping between some features, $x$, and an output, $y$. In classification, the output, $y$, is a category. In **binary classification** (by far the most common), there are only two categories: yes or no, usually represented as "0" (no) or "1" (yes). In **multi-class classification**, there are more than two categories.

To learn an appropriate mapping, we feed **training data** to a **learning algorithm**. Different algorithms learn different types of mappings.

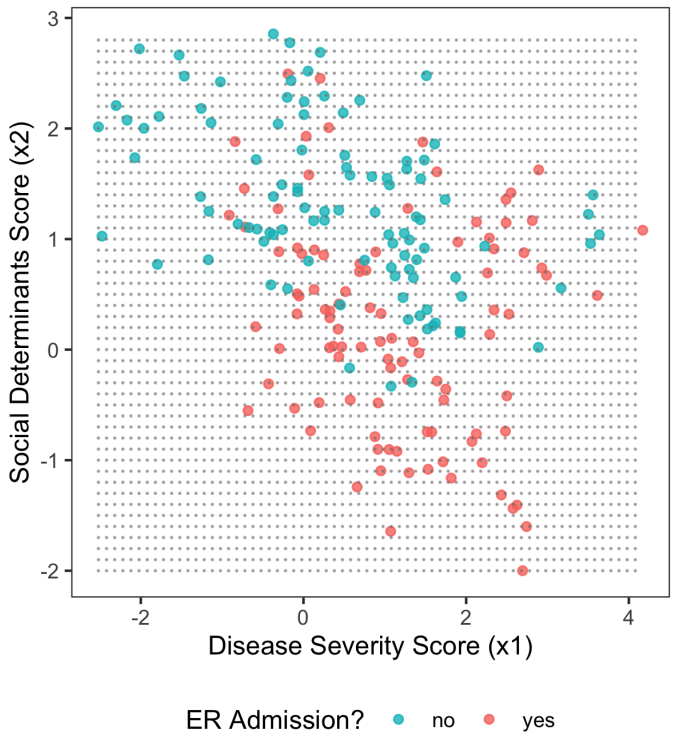
## 2.1 Definitions

- **Training data:** The data used, along with an appropriate learning algorithm, to create the mapping between input and output. It is composed of **training examples**, a.k.a. **samples**, each consisting of one or more input features and a single output.
- **Test data:** An independent dataset, not used in model training, on which the performance of a trained supervised learning model is evaluated.
- **Feature:** Also known as a **predictor**, or **covariate**, one of the inputs to a supervised learning algorithm.
- **Output:** Also known as the **outcome**, or **label**, the thing you are trying to predict.

- **Feature space:** Envisioning each feature as having its own axis that is orthogonal to all of the other features' axes, the multidimensional space spanned by those axes (or rather: unit vectors in the directions of those axes)
- **Extrapolation:** Making predictions outside the region of the feature space occupied by the training data. This will often lead to errors.

## 2.2 Visualizing the Classification Problem

Imagine we want to predict whether a patient will be readmitted to the emergency room (ER) within 30 days of hospital discharge. We gather data on two predictors: a disease severity score ($x_1$), which characterizes the severity of illness, and a social determinants score ($x_2$), which characterizes the patient's socioeconomic status. We have data on 200 patients.

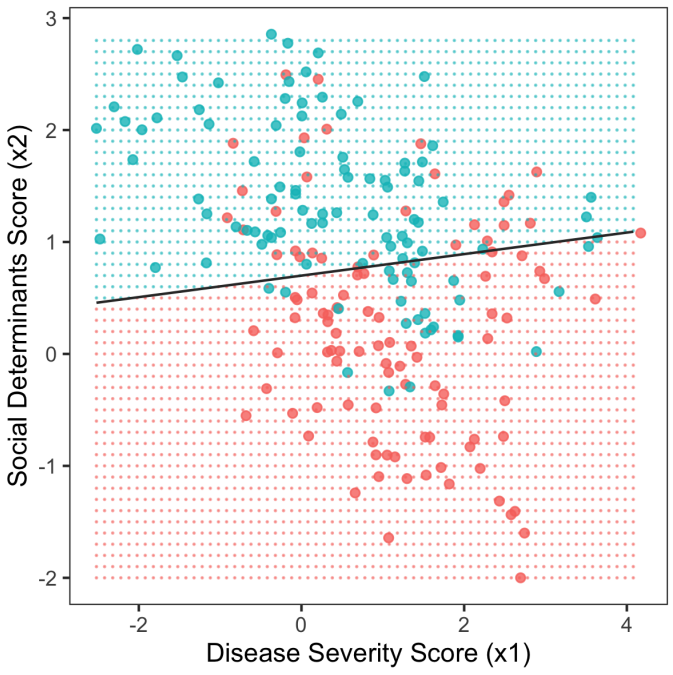


In this figure, the color refers to whether a patient was readmitted (blue = "no", red = "yes"). The location of each point is governed by the patient's disease severity score ($x_1$, horizontal axis) and social determinants score ($x_2$, vertical axis). Our goal in classification is to draw a **decision boundary** through this space, on one side of which we will predict that the patient is readmitted, and on the other side not.

## 2.3 Three Classification Algorithms

### 2.3.1 Logistic Regression

The simplest decision boundary is, arguably, a line. The logistic regression algorithm simply draws a line[1] through the feature space that divides the positive and negative training examples.




[1] In a higher-dimensional feature space, the decision boundary for logistic regression is a **hyperplane**.

The output of a fitted logistic regression model from R looks like this:

```
Call:
glm(formula = y ~ x1 + x2, family = "binomial", data = df)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.88232  -0.90614  -0.05965   0.86579   2.28489

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.9780     0.2945   3.321 0.000897 ***
x1            0.1344     0.1372   0.980 0.327272
x2           -1.3981     0.2316  -6.035 1.59e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 277.26  on 199  degrees of freedom
Residual deviance: 209.54  on 197  degrees of freedom
AIC: 215.54

Number of Fisher Scoring iterations: 4
```

The equation of the line (or, in higher dimensions, hyperplane) that forms the decision boundary in logistic regression can be obtained by setting the linear sum of coefficients of this model equal to zero.

$$0.9780 + 0.1344x_1 - 1.3981x_2 = 0$$
$$\implies x_2 = \frac{0.9780 + 0.1344x_1}{1.3981}$$

At any point, $(x_1, x_2)$, in the feature space, the model's predicted probability of a positive outcome (i.e. probability of an ER readmission) is related to the coefficients by this equation

$$\log \frac{P[Y = 1]}{1 - P[Y = 1]} = 0.9780 + 0.1344x_1 - 1.3981x_2$$

The decision boundary occurs when $P[Y = 1] = 0.5$ (total uncertainty, e.g. a coin toss). Another way to write this equation is:

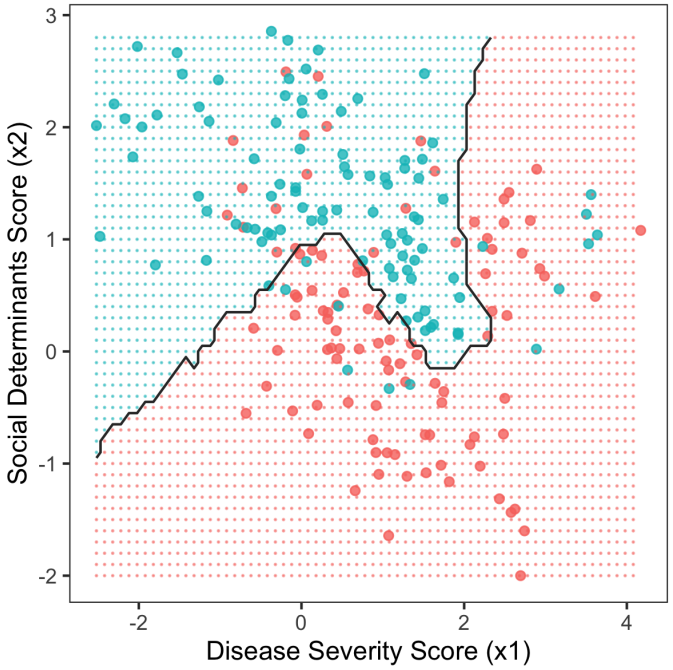
$$P[Y = 1] = \frac{1}{1 + \exp(-(0.9780 + 0.1344x_1 - 1.3981x_2))}$$

The functional form on the right, $1/(1+\exp(-z))$, is called the **logistic function**; this is how logistic regression got its name. We will learn much more about the math behind logistic regression in subsequent chapters.

### 2.3.2 K Nearest Neighbors (KNN)

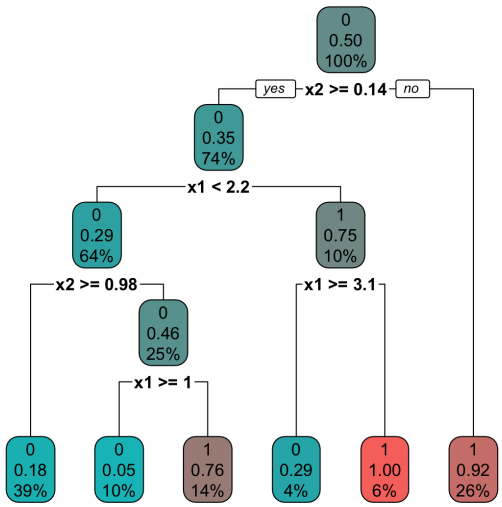
Another – completely different – approach to classification is to start with no assumptions about the shape of the decision boundary. To make a prediction about a new patient, we simply identify the $K$ nearest neighbors to that patient from our training set and allow them to vote on whether or not the new patient will be readmitted. The parameter $K$ must be set independently and is called a **hyperparameter**.

Here is the decision boundary for KNN with $K = 15$:

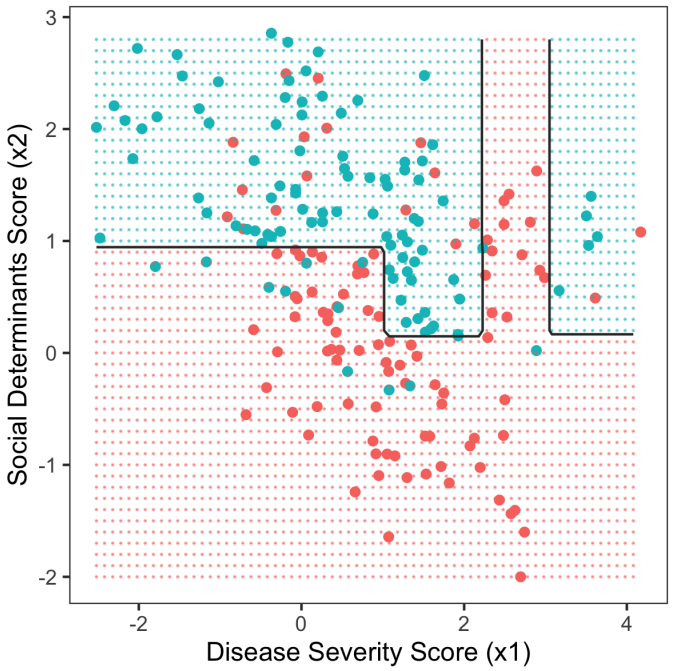


### 2.3.3 Decision Tree

Finally, we may choose to use our training data to build a decision tree, which will allow us to make predictions on new patients using a series of simple yes/no questions. There are different decision tree learning algorithms, but here is the tree produced by a famous one called CART:




And here is the decision boundary produced by this tree:




**Question 2.1**

How can you tell, just by looking at these images, which feature ($x_1$ or $x_2$) impacts the outcome the most? Which one is it?

**Question 2.2**

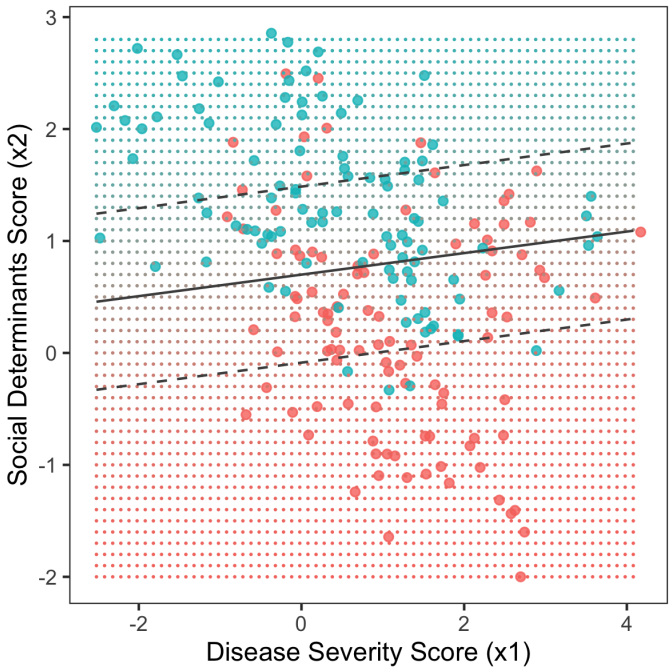
There are six rectangular regions in the picture of the decision tree decision boundary. Each corresponds to one of the six leaves of the tree. Identify all six and which leaves they correspond to on the decision tree.

## 2.4 Classification with Probabilities

We can think of classification as simply drawing a decision boundary, but underlying each algorithm is a quantitative assessment of each point in the feature space. Each algorithm is, in its own way, able to provide a degree of

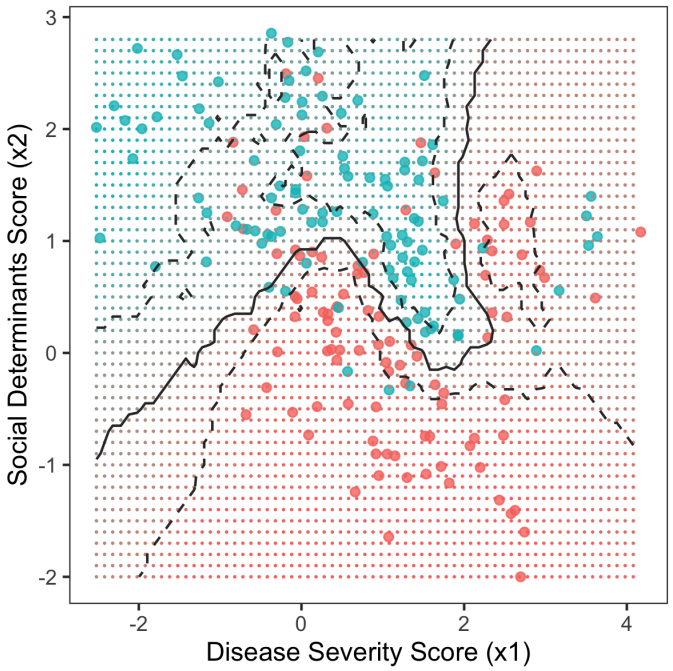
certainty, or **probability**[2], that a point belongs to the positive outcome class.

For example, here is the feature space of the example we just saw, colored by the probability, according to logistic regression, that a sample at each point should be classified as positive (i.e. the patient will be readmitted to the ER):

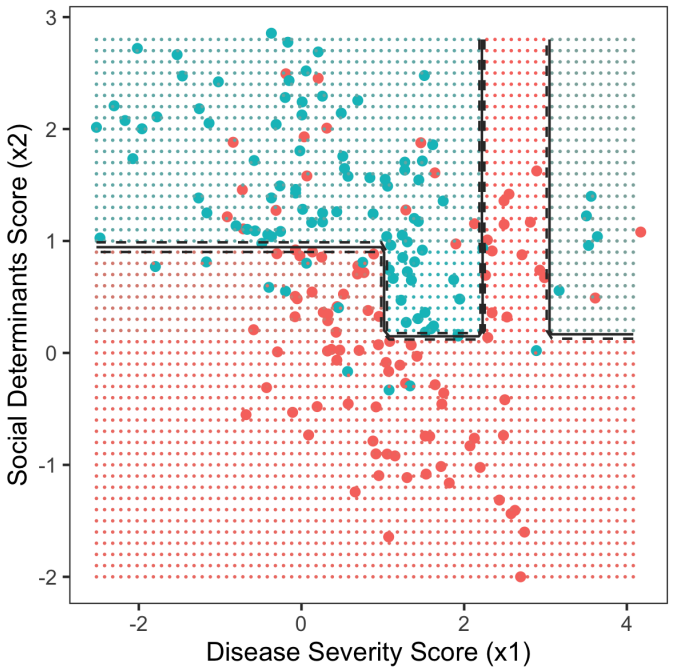



The solid line is the decision boundary, and the dashed lines indicate where the probability of a positive outcome (ER readmission) is 25% (top line) and 75% (bottom line). You can see that the color of the background gets purer red or purer blue the further you get from the decision boundary, but that near the decision boundary, the color is rather murky. That murkiness reflects the algorithm's uncertainty about the outcome. At the decision boundary, it is maximally uncertain. There the probability of a positive outcome is 50%: a coin toss. Here is a similar plot for KNN ($K = 15$):

[2]Pedantic footnote: this is a Bayesian definition of probability, as opposed to a frequentist definition. More on that later.




You can see that the shapes of the 25% and 75% probability lines have much more complex shapes than for logistic regression, but the story is the same: you have regions of pure blue or red, where the algorithm is certain, and you have a murky region near the decision boundary. Now, finally, here is the same plot for the decision tree:




The color of the background in the regions corresponding to the six leaves of the tree is the same throughout each region. That's because the probability in each rectangular region (corresponding to each leaf of the tree) is constant. It equals the number of red dots in that region divided by the total number of dots.

**Question 2.3**

What are the advantages and disadvantages of each algorithm?

1. Logistic regression?
2. KNN ($K = 15$)?
3. Decision tree?

**Question 2.4**

What makes a good classification algorithm? Consider issues of accuracy, generalizability, and speed (both to train the algorithm and to use it to make predictions on new samples).