

Slides adopted from Harvard CS 109

Taught by: Weiwei Pan, Kevin Radar, Pavlos Protopapas

Statistical Modeling

Regression vs. Classification

Error/Loss Functions

k-Nearest Neighbors

# Statistical Modeling

---

# Predicting a Variable

Let's imagine a scenario where we'd like to predict one variable from another (or set of other) variables.

Examples:

- Predicting the amount of views a YouTube video will get next week based on video length, the date it was posted, previous number of views, etc...
- Predicting which movies a Netflix customer will rate highly based on his or her previous movie ratings, demographic data, etc...
- Predicting whether or not a customer on Amazon will purchase something based on mouse clicks, demographic data, what ads they are presented, etc...

## Outcome vs. Predictor Variables

There is an asymmetry in a lot of these problems. The variable we'd like to predict may be more difficult to measure, is more important than the others, or may be directly or indirectly influenced by the levels of the other variable(s). Thus we define:

- The *outcome* or *response variable* as the variable we'd like to predict, and typically call this variable  $Y$  (and the measurements  $y_i$ ).
- The *features* or *predictor variables* as the variables we can use to do the predictions, and typically call these variables  $X = (X_1, \dots, X_p)$  (and the measurements  $x_{i,j}$ ).

Note:  $i$  represents which observation ( $i = 1, 2, \dots, n$ ) and  $j$  represents which predictor variable we are referring to ( $j = 1, 2, \dots, p$ ).

## True vs. Statistical Model

We will assume there is a general form of how the response,  $Y$ , relates to the predictors,  $X$ , in the general, theoretical sense:

$$Y = f(X) + \epsilon$$

Here  $f$  is the unknown function that relates  $Y$  to  $X$ , and  $\epsilon$  is the random error term (unrelated to the predictors,  $X$ ).

We define a statistical model as any algorithm that attempts to estimate  $f$ . This estimated function is written as  $\hat{f}$ .

A measured set of predictors for an observation  $(x_{i,1}, \dots, x_{i,p})$  can then be used to predict the response variable. This resulting predicted value is called  $\hat{y}_i$ .

## Prediction vs. Estimation

For some problems, what we care about estimating is  $f$ , the function that relates  $Y$  to  $X$ . These are often called *inference* problems.

For other problems we don't necessarily care about what the mathematical function that  $f$  takes on, but what we care about is getting  $\hat{y}_i$  as close to  $y_i$  as possible. These are called *prediction* problems.

We'll see that some algorithms are better suited for inference, some are better for prediction, and others are a little bit of both.

# Regression vs. Classification

---

# Outcome Variables

There are two main types of prediction problems that we will see this term:

- *Regression* problems are ones with a quantitative response variable.
- *Classification* problems are ones with a categorical response variable.

The distinction is important as measuring how effective a method is depends on this distinction.



# Error & Loss Functions

---

## Error & Loss Functions

In order to determine how well a modeling approach works, one can define a *Loss function*, aka Error function, to quantify this.

What would be a reasonable Loss function for a quantitative outcome variable? How about for a categorical one?

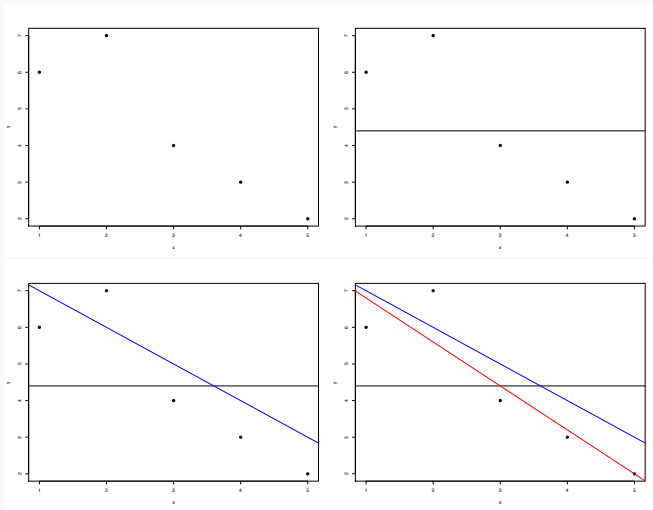
One common loss function for quantitative outcomes is called Mean Squared Error (MSE) loss:

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

For categorical outcomes a common loss function is the number of misclassified outcomes:

$$L = \sum_{i=1}^n I(y_i \neq \hat{y}_i)$$

# Line of Best Fit



Which line performs the best? How do you know?

## k-Nearest Neighbors (k-NN)

---

## k-Nearest Neighbors

One algorithm to predict an outcome variable (if quantitative) or classify an outcome variable (if categorical) is by using the k-Nearest Neighbors approach.

The approach is simple: to predict an observation's response, use the **other** available observations that are most similar to it.

For a specified value of  $k$ , each observation's outcome is predicted to be the average of the  $k$ -closest observations as measured by some distance of the predictor(s).

With one predictor, the implementation can be easily illustrated for a small example.

## k-NN: a Simple Regression Example

Suppose you have 5 observations each with an outcome and one predictor. The rectangular data set (transposed to save space) is thus:

$X$	1, 2, 3, 4, 5
$Y$	6, 7, 4, 3, 2

Calculate the predicted values for the k-NN procedure for  $k = 2$ .

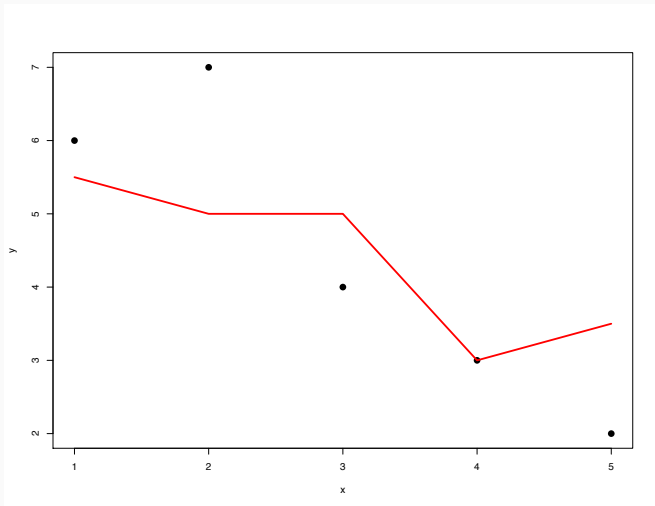
$$\hat{Y} = (5.5, 5.0, 5.0, 3.0, 3.5)$$

Calculate MSE for these predictions.

$$MSE = \frac{1}{5} ((6 - 5.5)^2 + (7 - 5.0)^2 + \dots + (3.5 - 2)^2) = 1.5$$

We could plot these predictions to see how well this procedure has performed:

## Simple example plotted



## Training vs. Testing set

The k-NN approach illustrates an important idea of determining how well an algorithm is performing in predicting a response.

The data are often split into two distinct subsets:

- The *training data* are the observations we used to estimate  $f$  or calculate  $\hat{y}$ .
- The *testing data* set are the observations we use to determine how well the model fits.

**Note:** often the entire set of data are used as the training set.



## Choice of $k$

How well the predictions perform is related to the choice of  $k$ .

What will the predictions look like if  $k$  is very small? What if it is very large?

More specifically, what will the predictions be for new observations if  $k = n$ ?

$$\bar{y}$$

A picture is worth a thousand words...

# Choice of k Matters

