Friday, 4 July 2025

Machine learning notes

Cross validation

* Since there are many machine learning models, we might be confused which model to choose, here comes the cross valdation
* There are two steps involved
  + we need to estimate parameters for the model which is also known as training the data
  + Next we need to check how well the model performs
* Suppose we have a large block of data we can use 75% of data for training the model and the rest 25% of data to train the data. But we dont know which 75% of data to take.
* Cross validation takes each 75% data once and estimate the parameters and test the model on the rest 25%. and gives an average proformance of the model.

Confusion matrix

* A confusion matrix is a matrix that tells us what our algorithm did right and what it did wrong
* Example is a model that predicts whether a person has a heart disease or not. So a confusion matrix is such that the rows indicate the prdicted decision and colums represent true decison

|  | Person has heart disease | Person does not have hear disease |
| --- | --- | --- |
| Person has heart disease | 1 | FALSE POSITIVES |
| Person does not have heart disease | FALSE NEGATIVES | 1 |

Sensitivity and specificity

* After making the confusion matrix, we can find the sensitivity and specificity which are defined as follows
  + Sensitivity is defined as the percentage of true positives over true positivies and false negatives.
  + Specificity is defined as the percentage of true negatives over true negatives and fasle positives

Example

|  | Person has heart disease | Person does not have hear disease |
| --- | --- | --- |
| Person has heart disease | 139 | 20 |
| Person does not have heart disease | 32 | 112 |

Sensitivity calculation:

True positives = 139

false negatives = 32

therefore sensitivity = 139/(139 + 32) = 81.3%

Specificity calculation:

True negatives = 112

false positives = 20

therefore specificity = 112/(112 + 20) = 85%

Example for 3 x 3 confusion matrix

|  | Movie A | Movie B | Movie C |
| --- | --- | --- | --- |
| Movie A | 12 | 102 | 93 |
| Movie B | 112 | 23 | 77 |
| Movie C | 83 | 92 | 17 |

Sensitivity for movie A:

True positives = 12

False negatives = 112 + 83 = 195

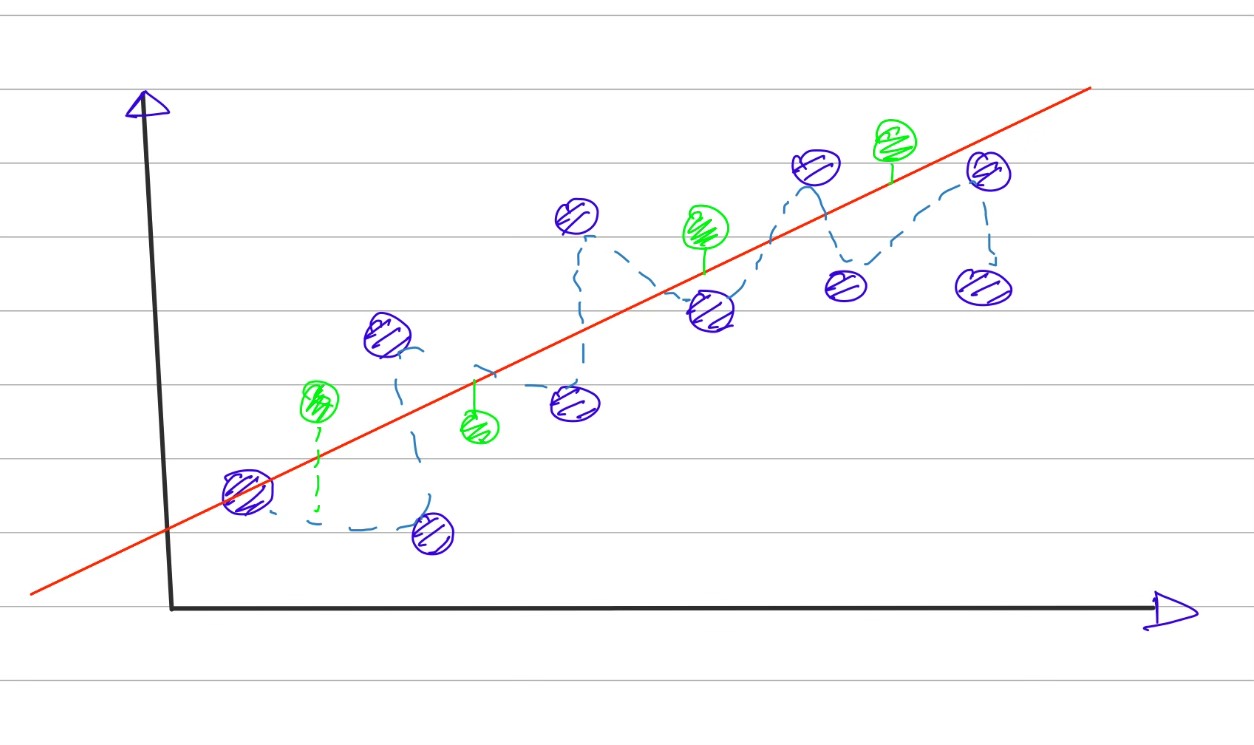
sensitivity(movie A) = 12/(12 + 195) = 6%

specificity for movie A:  
True negatives = 23 + 77 + 92 + 17 = 209

False positives = 102 + 93 = 195

specificity (movie A) = 209/(209 + 195) = 51.73%

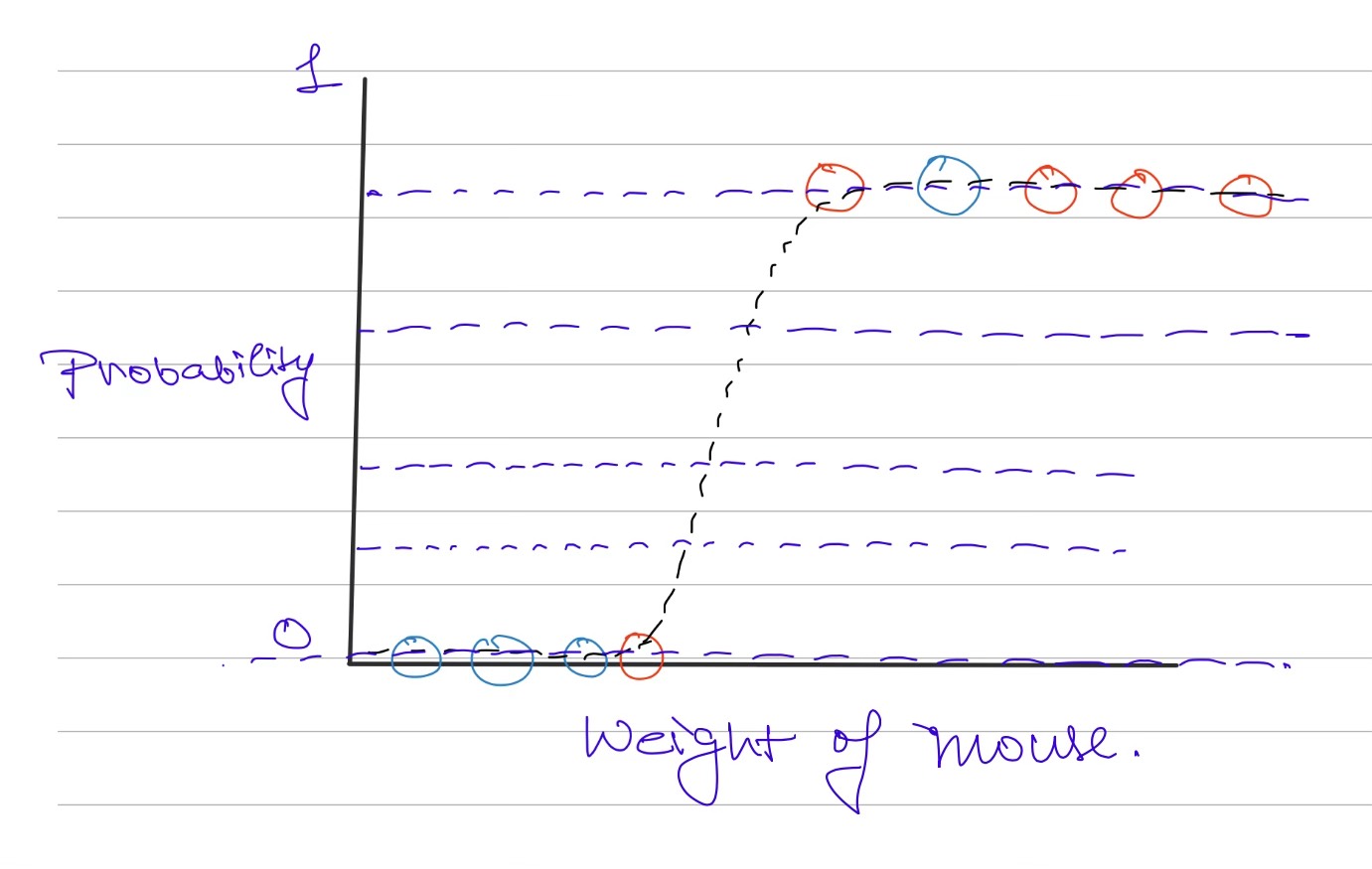
Bias and variance

* Variance is defined as how the model fits the datasets , example:

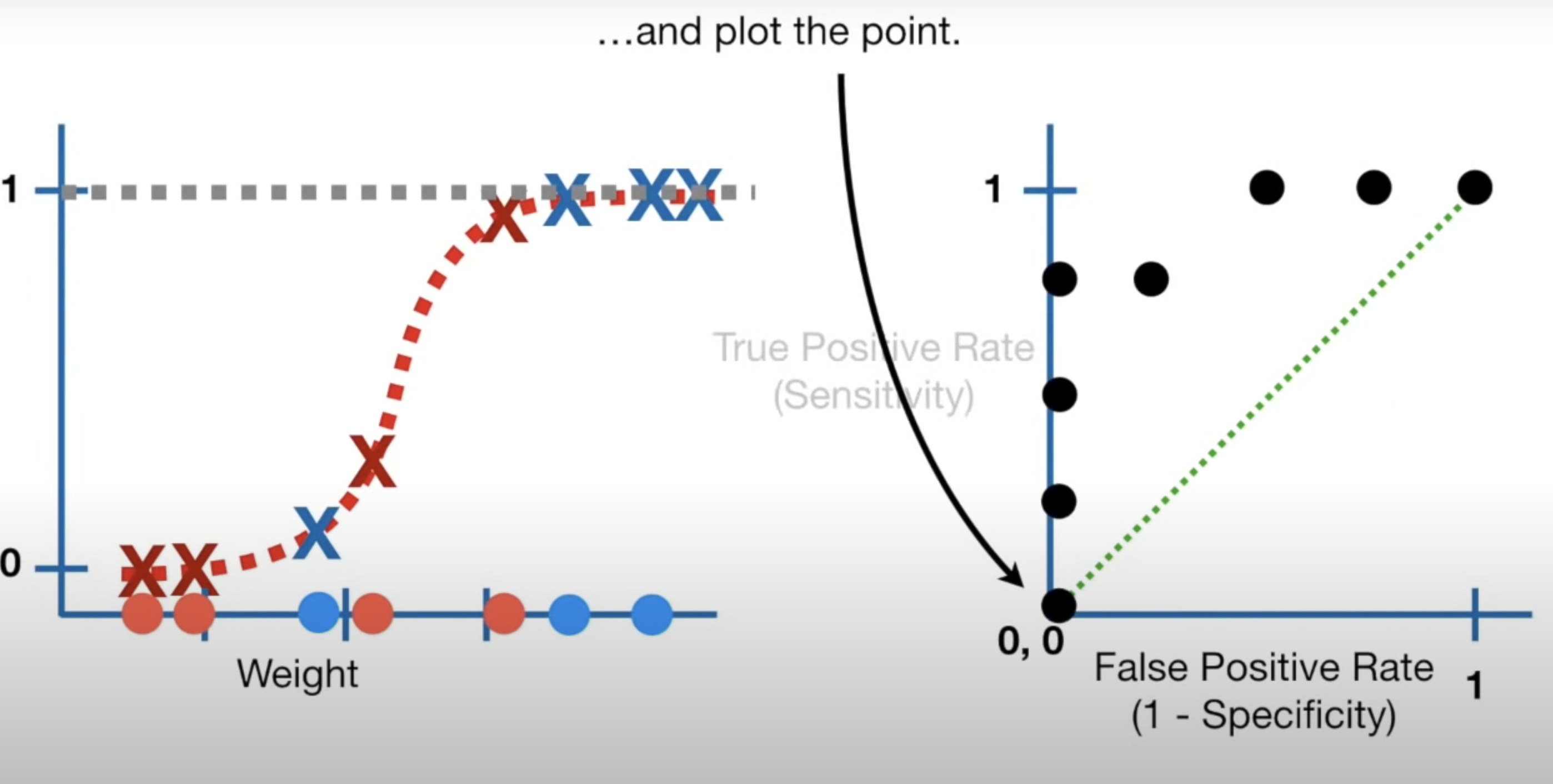
The squigly line has more variability than the straight line

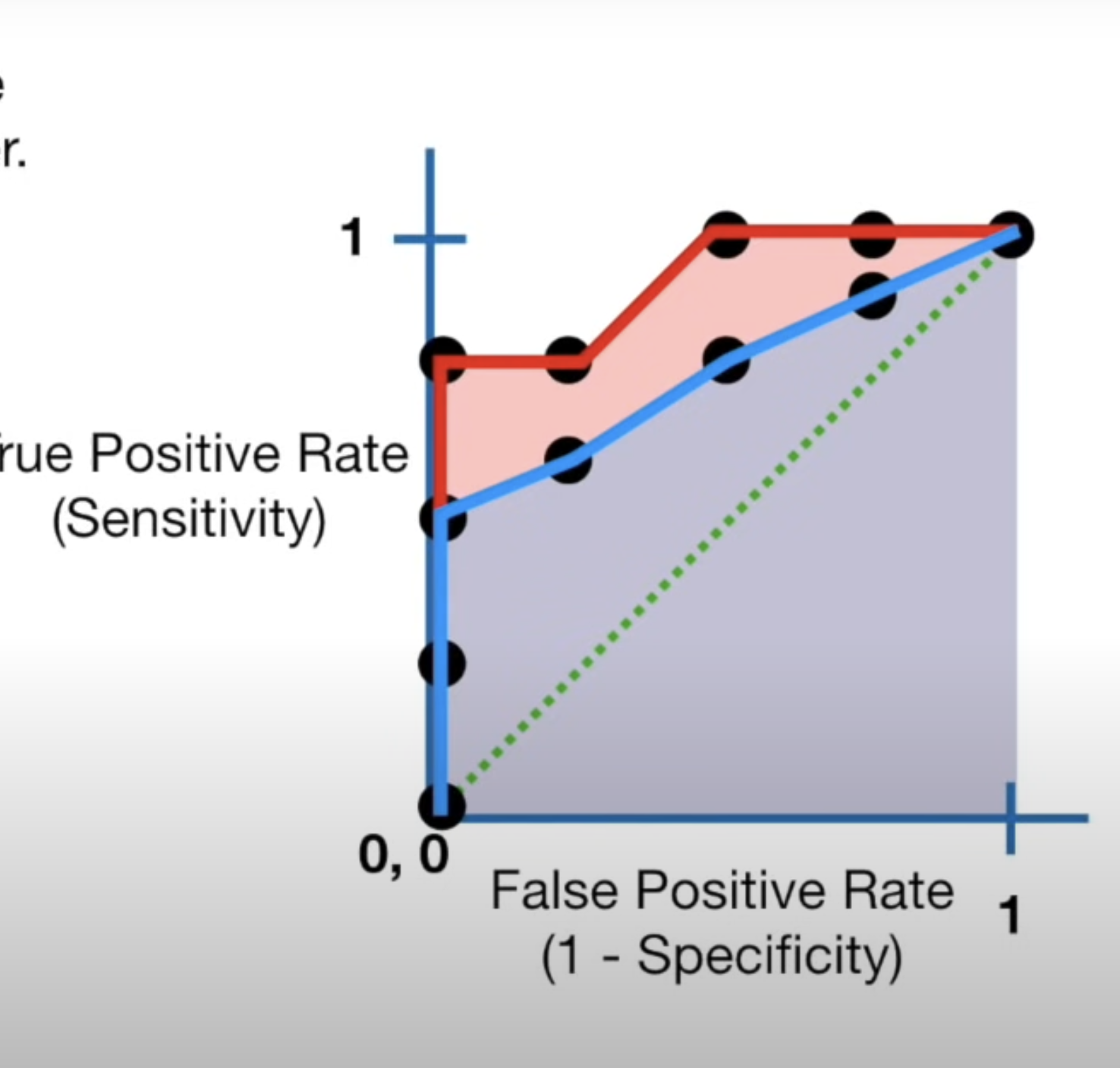
* Bias: It is defined as how well the model fits the real world data. The squigly line has less bias than the straight line.

ROC AND AUC

* Suppose we want to classifiy subjects on basis of some of its characterstics like height, weight etc
* For eg, we want to classify mice as obese and non obese on the basis of their weights

As you can see we need to set the threshold so that we can classify the mice

Each threshold value creates a confusion matrix .These large number of confusion matrices can be confusing so we plot a graph called ROC(reciever operator characterstic) which plots rate of false positives (1 - specificity) vs rate of true positives (sensitivity)

The area under the ROC gives AUC (area under the curve) . The larger the AUC the better the classification model

Red is AUC for logistic regression and blue is for random forest

Thus logisitic regression is better than random forest method

Entropy

* Before learning about entropy , we need to know about surprise
* Surprise can be seen as some sort of inverse of probability
* Instead of directly taking inverse of probability, we take logarithm of inverse of probability
* We define entropy as expected value of surprise
* suppose we have a coin with probability of head as 0.9 and probability of tails as 0.1, and we flip the coins 100 coins, we estimate the surprise of heads as (0.9x100)xs(head) and surprise of tails as (0.1x100)xs(tail). Therefore total surprise of 100 flips is estimated as (0.9x100)xs(head) + (0.1x100)xs(tail).
* Then we divide this estimated value by number of flips and thats how we get entropy. Thus we can write entropy as