Introduction to ML

Core Materials:

1- Finding patterns in data, and using them to make prediction.

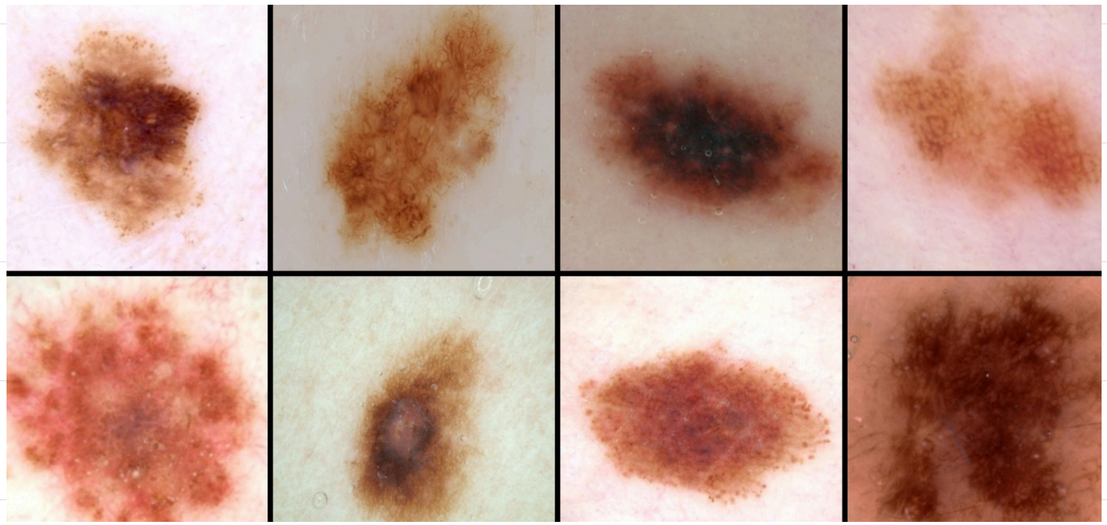
2- Models and statistics help us understand patterns.

3- Optimization algorithms "learn" the patterns.

The most important ingredient is data

Classification:

- The most common task that you hear people doing with ML is classification, e.g.,
  - to classify an input image as cat or dog.
  - to classify an individual as someone who would default on their Credit card payment or not.
  - to classify skin lesion as malignant Melanoma or benign nevi.



← Dangerous malignant melanomas

← benign nevi

Let's formalize what we do here.

- Collected training points with class labels.

Some observations belong to class C; some don't.

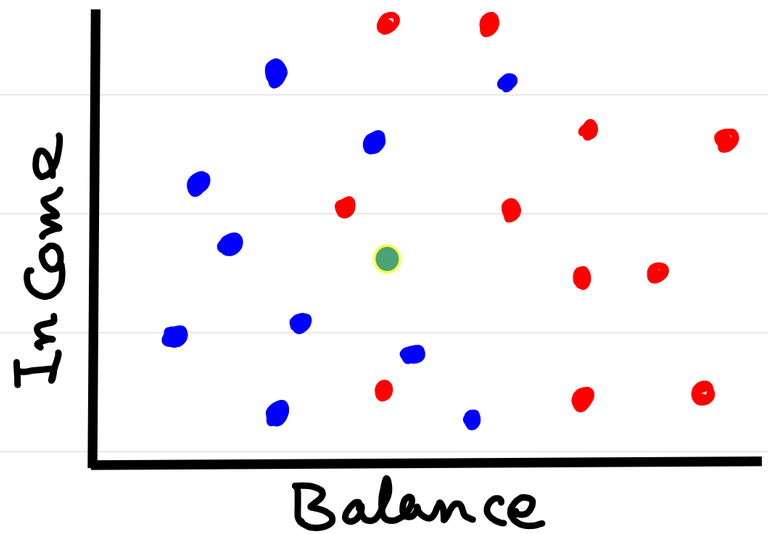
  - e.g., reliable debtors or defaulted debtors
  - e.g., Cancerous or benign

Remark: Hence, we are given a sample of N observations, each with d features. Some observations belong to class C; some not.

Remark: We represent each observation as a point in d-dimensional space, called a sample point / feature vector / independent variables. ← in case of regression.

- Evaluate new data points – predict their class

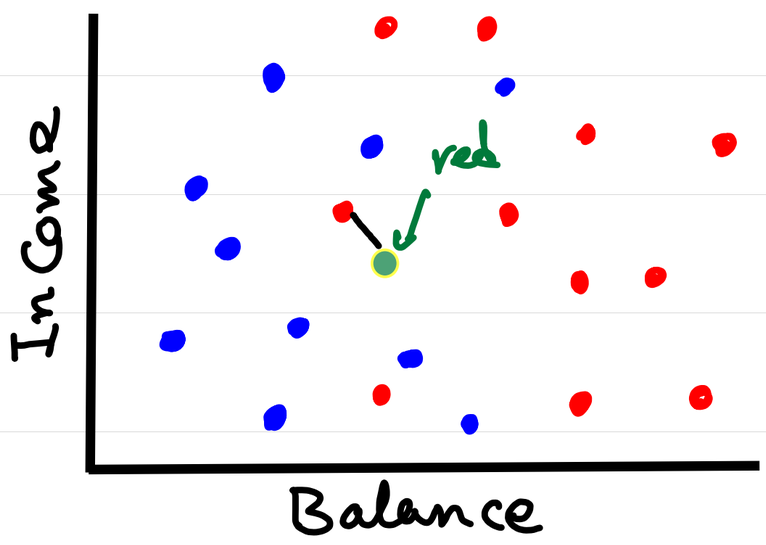
Let's see a simple example



- reliable debtor
- defaulted debtor
- the new point we want to classify

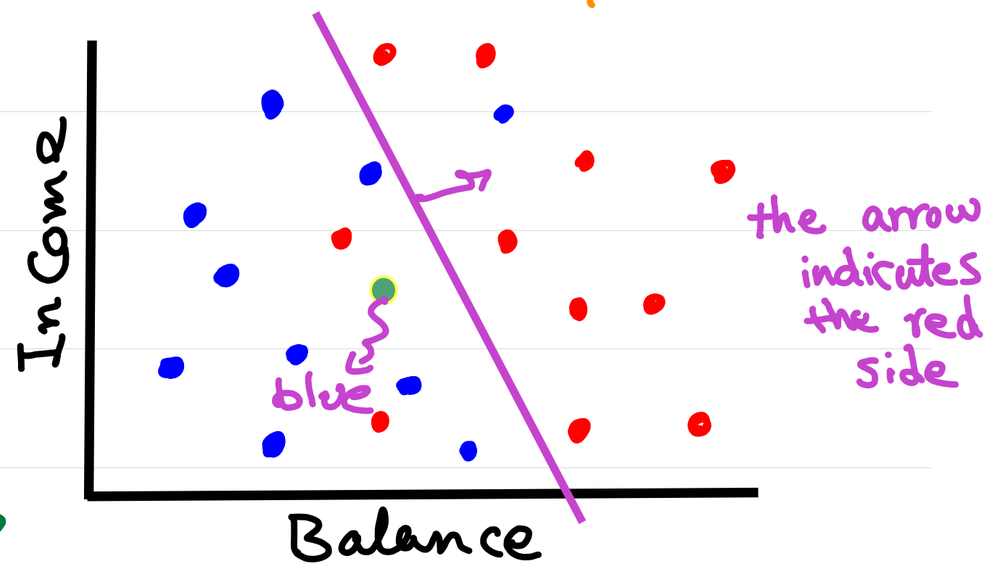
Look at their nearest neighbor

1-nearest neighbor



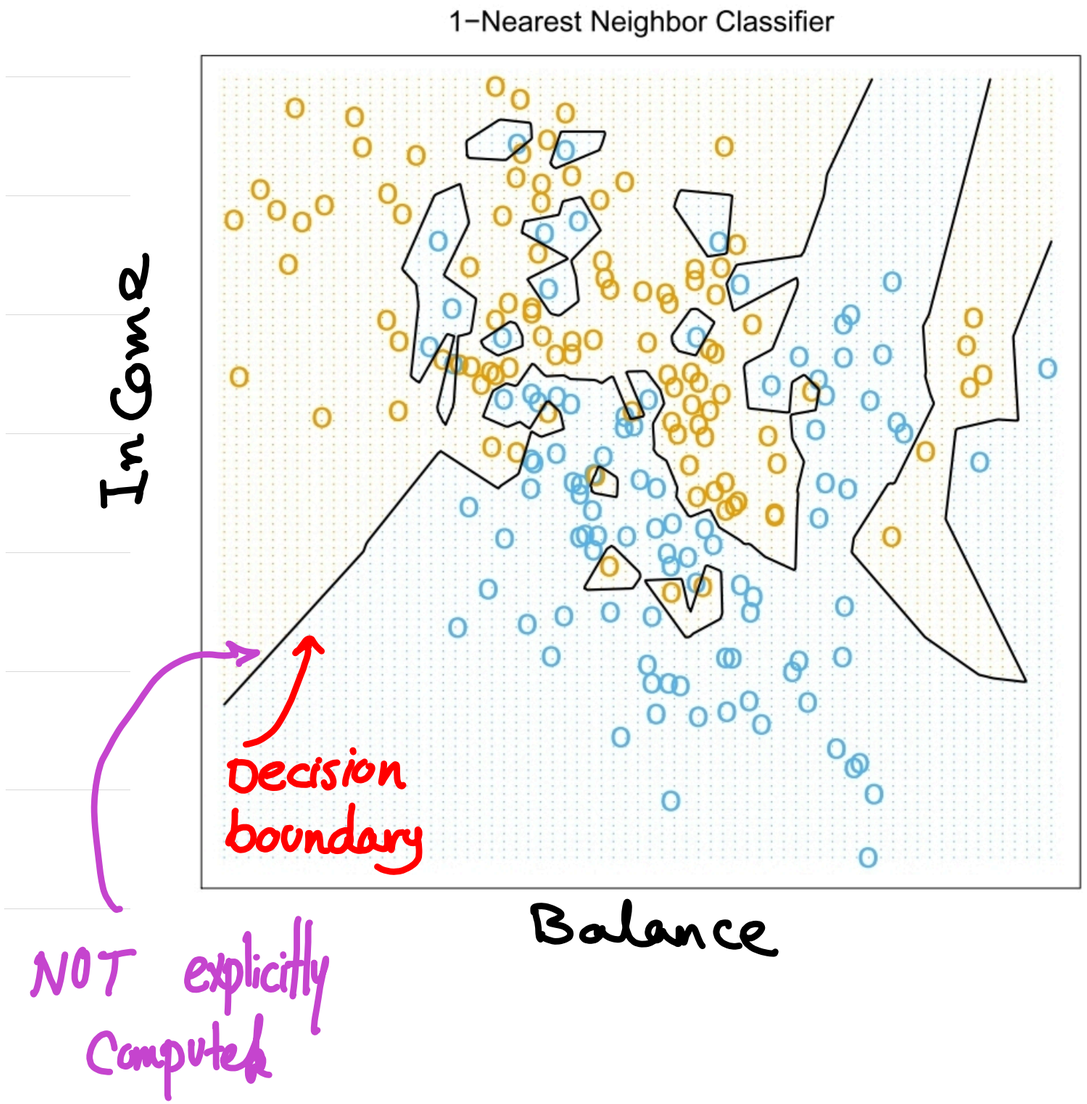
Which one is better?

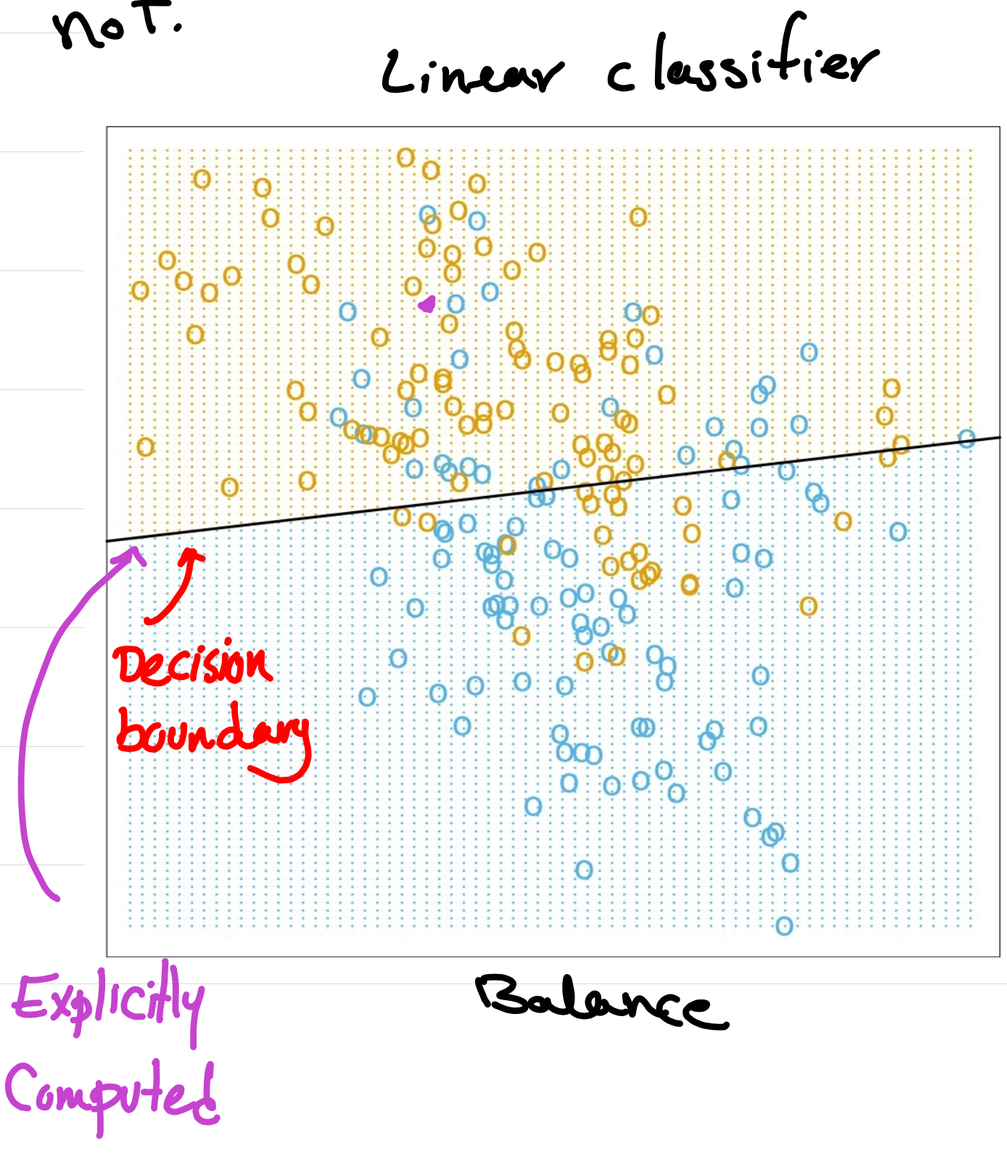
draw a line to separate them



Let's take a look at a more complex dataset: which one is better?

Decision boundary: the boundary by our classifier to separate items in the class from those not.





Q) which one computes the correct answer for each training point?

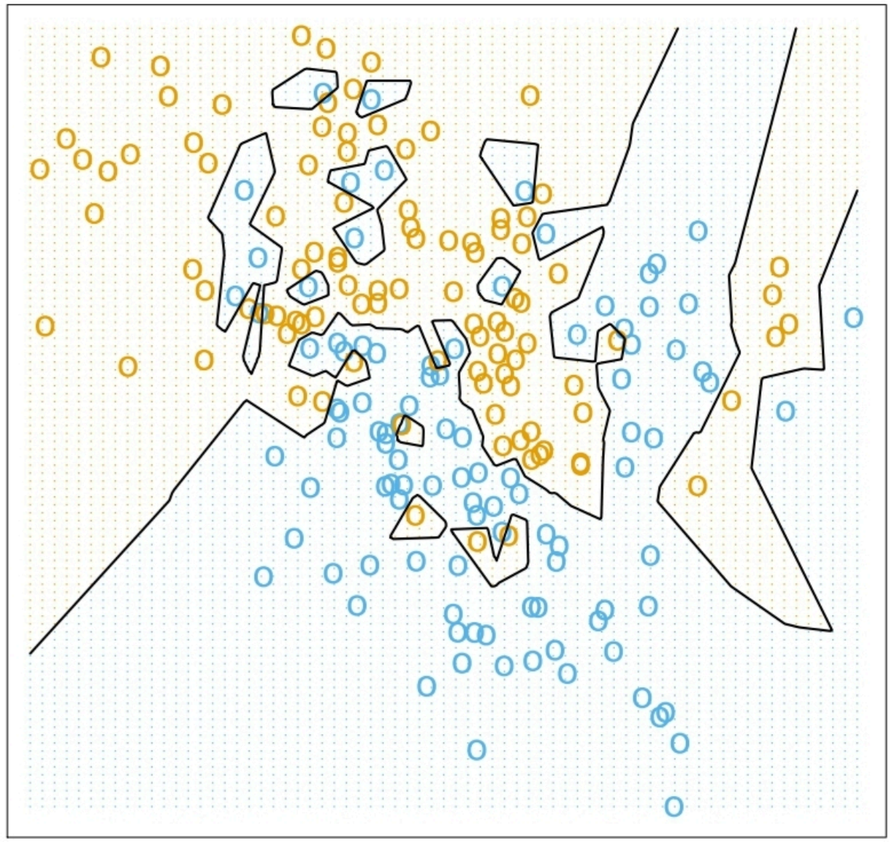
1-nearest neighbor ☑ linear classifier ☐

Q) which one conforms to some real insight about the data?

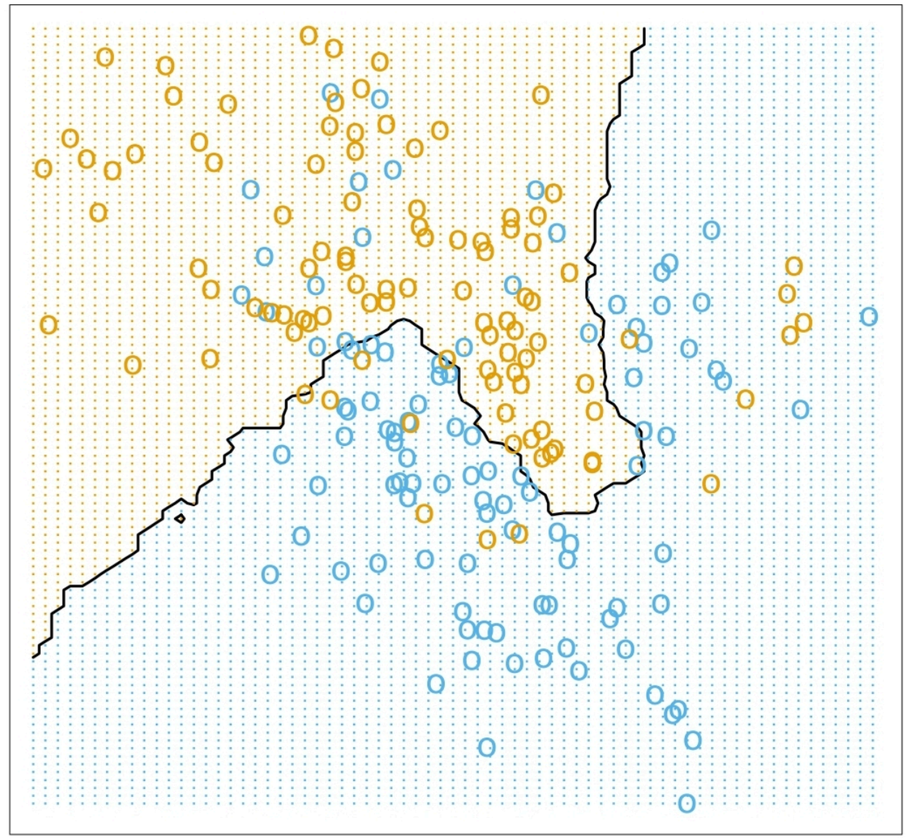
1-nearest neighbor ☐ linear classifier ☑

1-nearest neighbor VS. 15-nearest neighbor

1−Nearest Neighbor Classifier



15-Nearest Neighbor Classifier



overfitting

Training error, Test error, Validation error

How do we make sure that the model that we trained has statistical validity?

Here's how we do classification:

1 - we are given labeled data - sample points with class labels.

2 - hold back a subset of the labeled points, called the validation set. Maybe 20% of data. The other 80% is called the training set.

Training error, Test error, Validation error - Cont'd

3- train one or more classifiers: they learn to distinguish the two classes. use training set to learn weights / parameters of the model. Do NOT use your validation data to train!!!

Note: We usually train multiple learning algorithms, or one algorithm with multiple hyperparameter setting, or both.

4- Validate the trained classifiers on the validation set. Choose classifier/hyperparameters with lowest validation error.

Training error, Test error, Validation error - Cont'd

Note: When we do validation, our classifiers are not Learning anymore!

Note: We use validation error to judge our classifiers.
We do NOT use training error to judge our classifiers.

5 (optional)_ Test the best classifier on a test set of NEW data. Typically you do not have the labels. A third party judge you using the test error.

Training error, Test error, Validation error - Cont'd

- **Training Error**: The fraction of the training set not classified correctly.

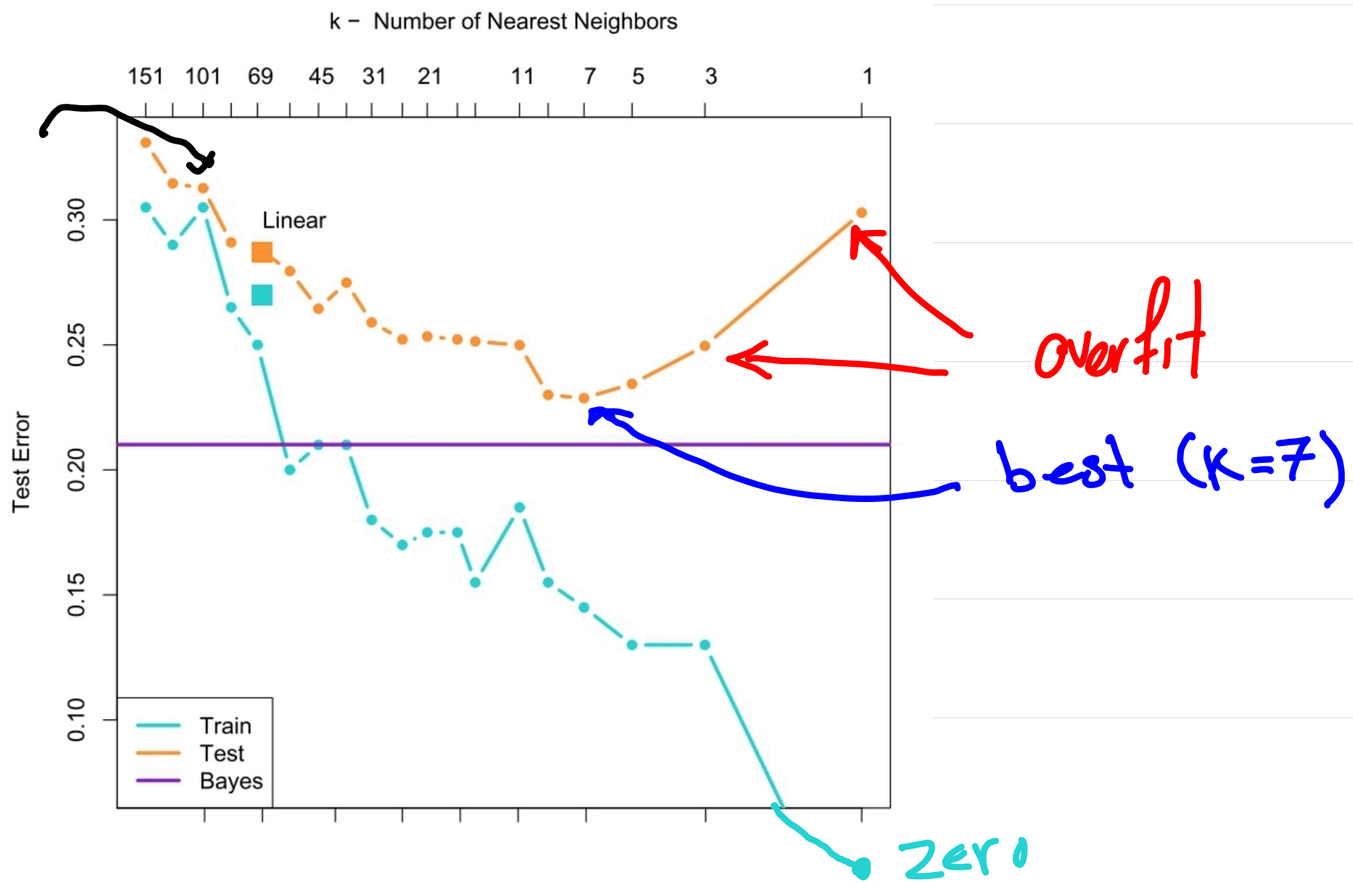
- **Validation Error**: The fraction of the validation set not classified correctly.
(This is what we are going to use to choose our classifier / hyperparameters)

- **Test Error**: The fraction of the test set not classified correctly.
(This is used to evaluate you)

Hyperparameters

Most ML algorithms have a few hyperparameters that (often) control the trade-off between over/underfitting, e.g., K in K-nearest neighbors.

underfit



Over/underfitting, Outliers

- **Overfitting:** When the validation/test error deteriorates because too sensitive outliers or other spurious patterns.

- **underfitting:** When the validation/test error deteriorates because the classifier is not flexible enough to fit patterns.

- **outliers:** points with atypical labels (e.g., dve to data entry error, rich borrower who defaulted anyway). The more outliers you have, the more they increase the risk of overfitting.

- In ML, the goal is to create a classifier that generalizes to new examples we haven't seen yet.

- overfitting and underfitting are both counterproductive to that goal.

- So, we are always seeking a compromise. We want decision boundaries that make fine distinctions without being downright superstitious.

Encoding

- Before we get mathematical, Let's think about how we should prepare the Data for ML Algorithms.
- Many ML arlgorithms can only operate on numeric values.
- Hence, we need to feed datapoints with numeric feature value. The numeric values should be reasonable. Very large or very small values can cause numeric issues and instability.

- Transforming numeric data
  - Normalizing the features
    - transforms the features to be on similar scale
    - improves the performance and training stability
  - Sometimes the Dataset contains extreme outliers
    - By feature clipping, you can cap all feature values above or below.
- Tranforming categorical data (e.g., movie ratings "terrible," "bad," "OK," "good," "awsome". OR Color)
  - Ordinal Encoding, e.g. "terrible" → 0, "bad" → 1, ...
  - Sometimes ordering the categories does not make Scence. You should do one-hot encoding <demo>
    A vector: set Corresponding elements to 1, set all other elements 0.