k-NN Classifier

UNIVERSITY OF LIVERPOOL

Classification vs. Regression

- In classification
  - We are given a dataset {x, y} where each instance x must be classified into a pre-defined finite (and often small) unordered set of classes y. We are required to learn a function f(x) that predicts the class y, given x.
  - Example
    - In binary classification $f(x) = \text{sgn}(ax + b) \in [-1,+1]$
- In regression
  - We are given a dataset {x, y} where each instance x is associated with a real number y, and we are required to learn a function f(x) that predicts the value of y, given x.
  - Example
    - In linear regression $f(x) = ax + b$
  - Note that in “ordinal regression” we have integer values instead of real-values. There is a clear total ordering among the target values. Therefore, ordinal regression is “regression” and not “classification”.

Classification Algorithms

- There are loads of them...
    - k-NN, Perceptron, Naive Bayes, logistic regression, SVM, neural networks, decision trees, random forests,...
- When you learn new classification algorithms look for two main points
    - What is the loss function that the classifier minimizes?
    - What optimization method does it use for the minimization of the loss function?

World’s simplest classifier!

- Given a training dataset D_{train} of N instances {**x**,y}, simply “remember” the entire N instance in the memory (or hard-disk)
    - memory-based learning
- When we want to classify a test (previously unseen, not in D_{train}) instance $\mathbf{x}^*$, we would simply check whether $\mathbf{x}^*$ is in D_{train}
    - if $\mathbf{x}^* \in D_{train}$ the return the label of $\mathbf{x}^*$
    - otherwise make a random guess

Quiz 1

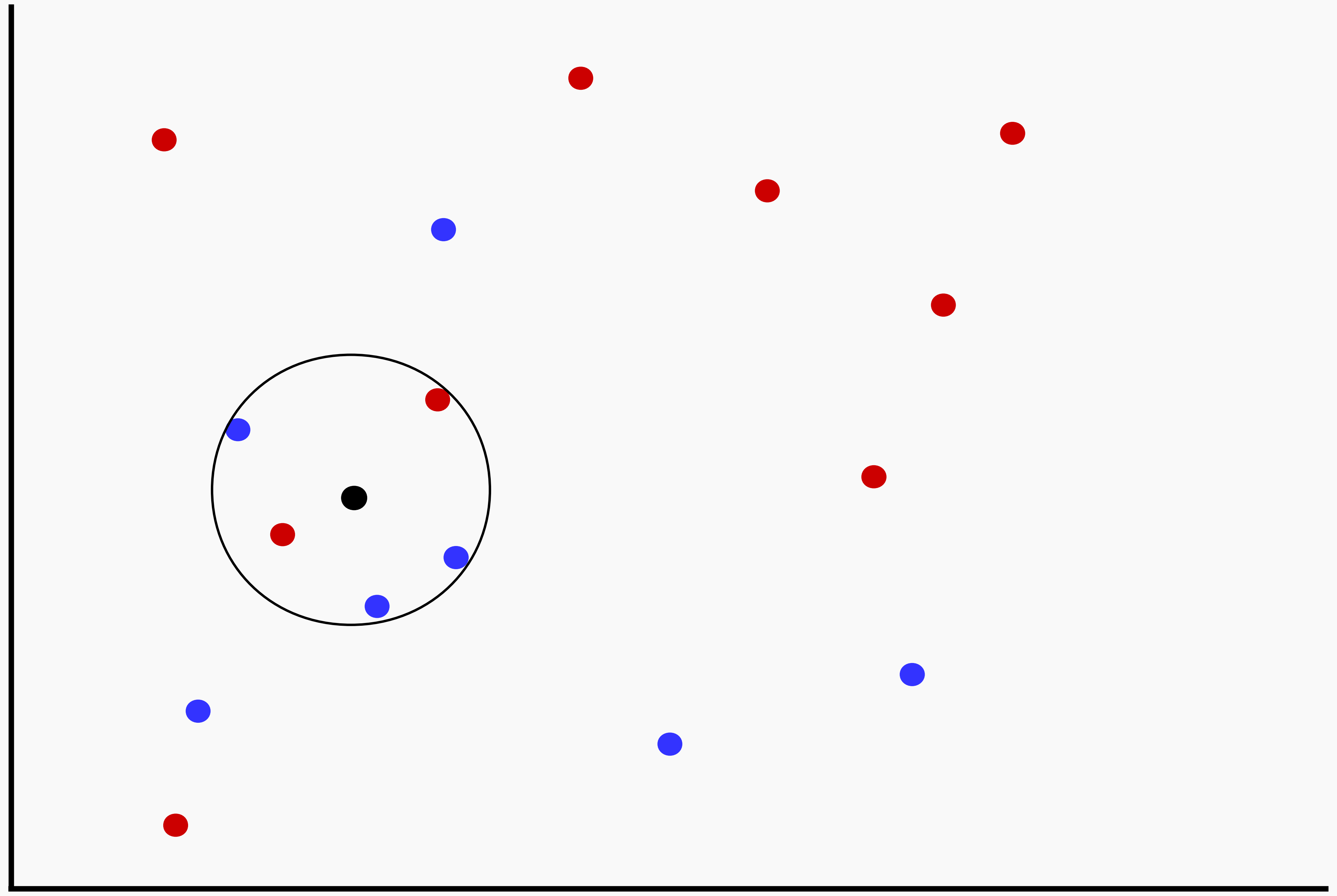
- What is the problem with the memory-based learner described in the previous slide?

k-Nearest Neighbour Classifier

1. Find the k closest (nearest) instances (neighbours) to the test instance
2. Find the majority label among those nearest neighbours
3. The majority label in the nearest neighbours is predicted to the test instance

Example

5-NN – find 5 closest to black point, 3 blue and 2 red, so predict blue



Measuring similarity/distance

- Various distance/similarity measures can be used
- Cosine similarity

$$\mathrm{CosSim}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x}^\top \boldsymbol{y}}{||\boldsymbol{x}||||\boldsymbol{y}||}$$

$$\boldsymbol{x}^\top \boldsymbol{y} = \sum_{i=1}^{D} x^{(i)} y^{(i)} \qquad ||\boldsymbol{x}|| = \sqrt{\sum_{i=1}^{D} {x^{(i)}}^2}$$

Here, $x^{(i)}$ denotes the i-th element of the vector x

Quiz 2

- Let **x** = (1, 0, -1)^T and **y** = (-1, 1, 0)^T. Compute the cosine similarity between the two vectors **x** and **y**.

Euclidean Distance

- Euclidean distance between two vectors **x** and **y** is defined as follows

$$\mathrm{EucDist}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(\boldsymbol{x} - \boldsymbol{y})^\top (\boldsymbol{x} - \boldsymbol{y})} = \sqrt{\sum_{i=1}^{D} \left(\boldsymbol{x}^{(i)} - \boldsymbol{y}^{(i)}\right)^2}$$

Quiz 3

- Let **x** = (1, 0, -1)^T and **y** = (-1, 1, 0)^T. Compute the Euclidean distance between the two vectors **x** and **y**.

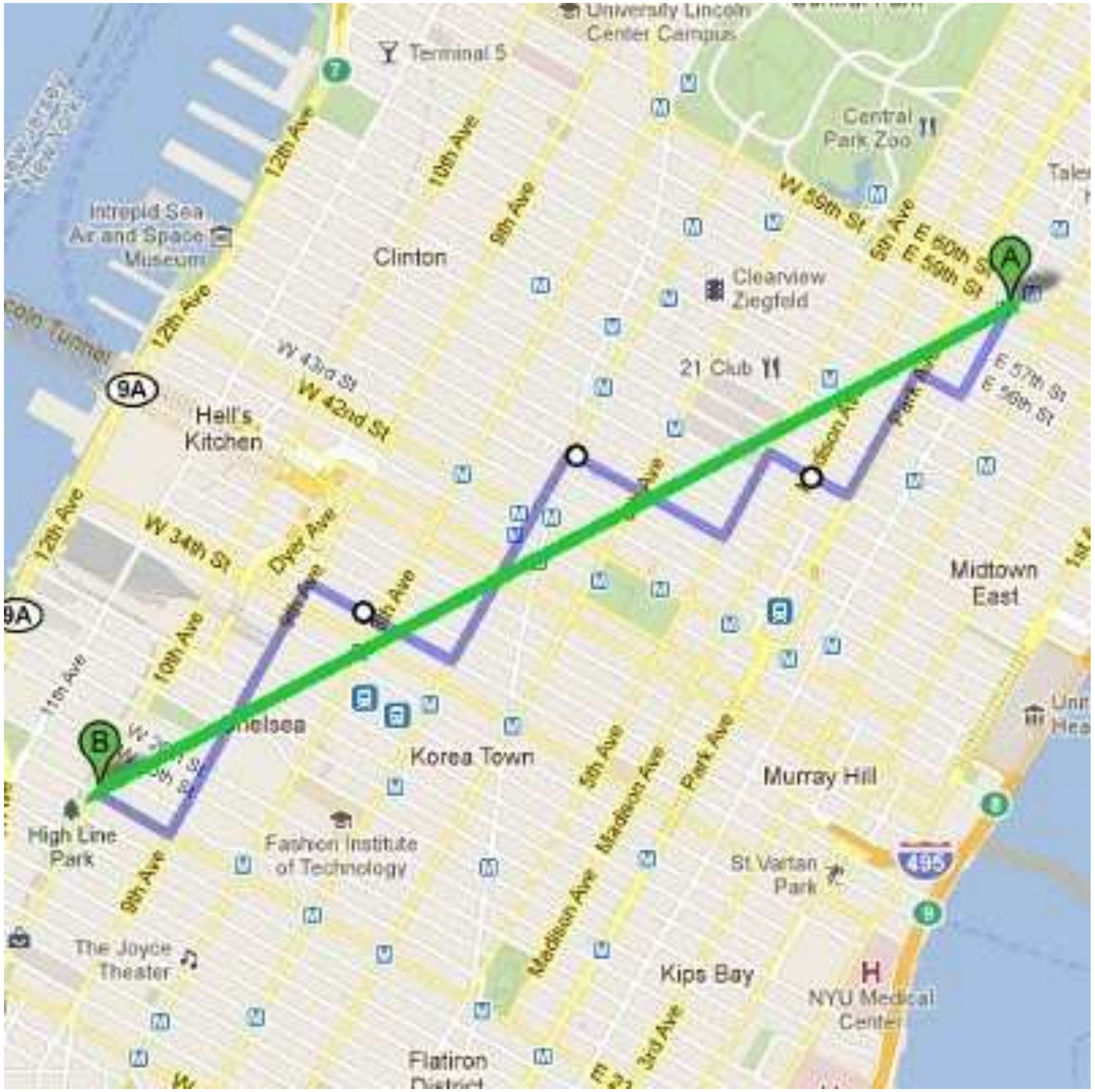
Manhattan Distance

- Manhattan (city block) distance between two vectors **x** and **y** is defined as follows

$$\mathrm{ManDist}(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}||_1 = \sum_{i=1}^{D} |\boldsymbol{x}^{(i)} - \boldsymbol{y}^{(i)}|$$

Here, $|\mathbf{x}|_1$ denotes L1 norm of the vector x

Manhattan



Central Park Zoo
Clinton
Hell's Kitchen
Chelsea
Korea Town
Midtown East
Murray Hill
Kips Bay
High Line Park
Fashion Institute of Technology
Clearview Ziegfeld
21 Club
Intrepid Sea Air and Space Museum
W 59th St
W 42nd St
W 34th St
A
B

Quiz 3

- Let **x** = $(1, 0, -1)^T$ and **y** = $(-1, 1, 0)^T$. Compute the Manhattan distance between the two vectors **x** and **y**.

Vector norms

- "norm" is a mathematical concept that expresses the "size/length" of a measure
- Popular vector norms in data mining are as follows
  - L1 norm $$||\boldsymbol{x}||_1 = \sum_{i=1}^{D} |x^{(i)}|$$
  - L2 norm $$||\boldsymbol{x}||_2 = \sqrt{\sum_{i=1}^{D} {x^{(i)}}^2}$$
  - L0 norm $$||\boldsymbol{x}||_0 = \text{no. of non-zero elements in } \boldsymbol{x}$$
  - L∞ norm $$||\boldsymbol{x}||_\infty = \max_i (x^{(1)}, x^{(2)}, \ldots, x^{(D)})$$

Quiz 4

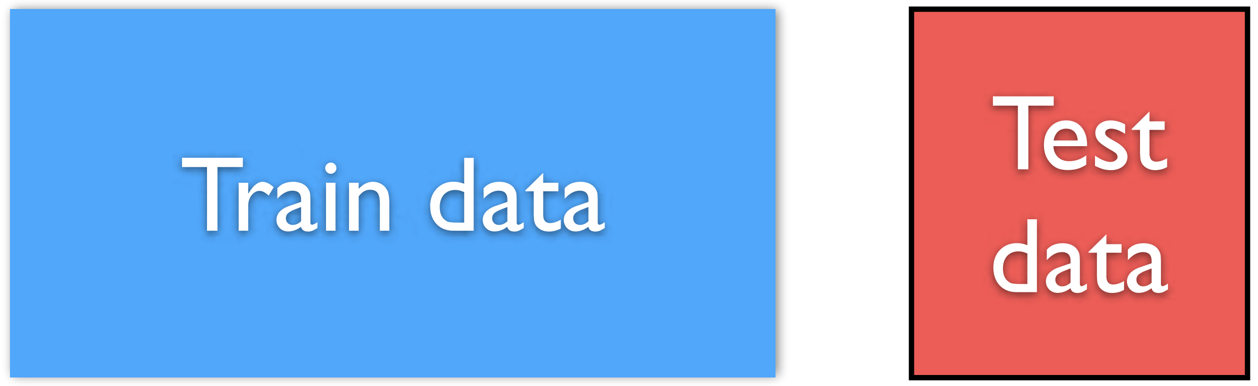
- Give the vector **x** = (1, 0, -1, 2)^T, compute the following $||\boldsymbol{x}||_1, ||\boldsymbol{x}||_2, ||\boldsymbol{x}||_0, ||\boldsymbol{x}||_\infty$

k-NN classification

- Select odd k values to avoid ties
- What value to use for k?
  - Depends on the dataset size. Large and diverse datasets need a higher k, whereas a high k for small datasets might cross out of the class boundaries
  - Calculate accuracy on a validation set for increasing values of k, and use the best value on the test data

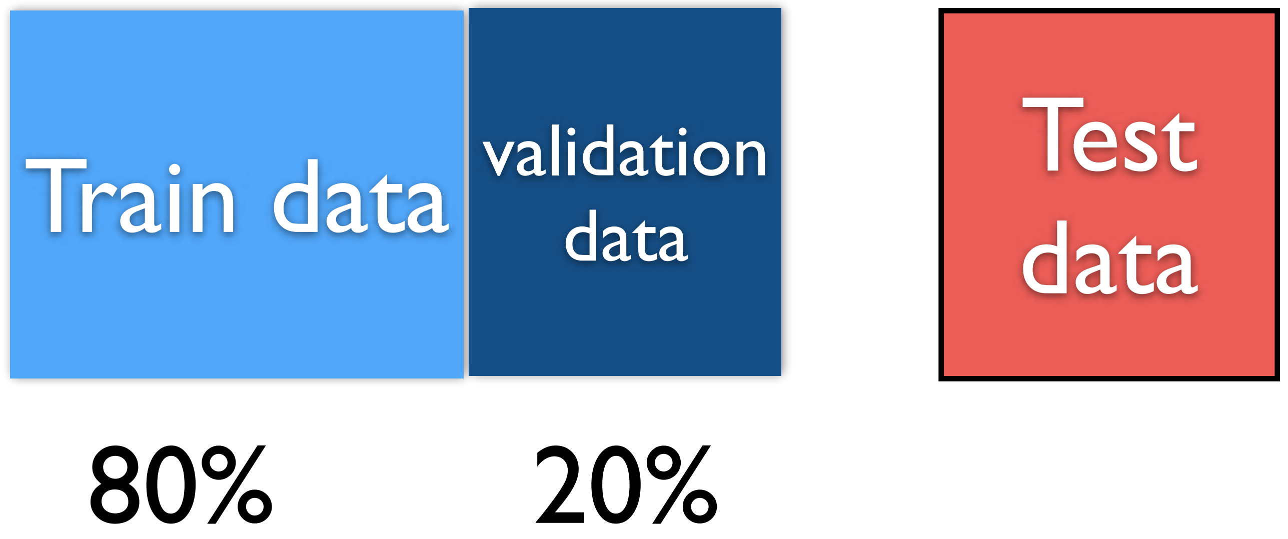
Train/Test/Validation datasets

- Validation data
  - Set aside (hold-out) a fraction of train data for validation purposes.
  - Hyper-parameters are often tuned using validation data
    - Hyper-parameter is a parameter that is NOT learnt during training, but is set BEFOR running the training algorithm. (e.g. k in k-NN is a hyper-parameter)
  - Using validation data to set hyper-parameters reduce overfitting
  - Of course, you CANNOT use test data for any tuning except for testing.



Train/Test/Validation datasets

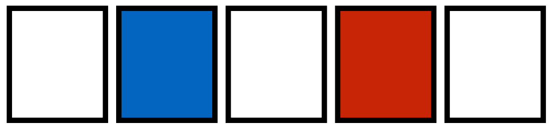
- Validation data
  - Set aside (hold-out) a fraction of train data for validation purposes.
  - Hyper-parameters are often tuned using validation data
    - Hyper-parameter is a parameter that is NOT learnt during training, but is set BEFOR running the training algorithm. (e.g. k in k-NN is a hyper-parameter)
  - Using validation data to set hyper-parameters reduce overfitting
  - Of course, you CANNOT use test data for any tuning except for testing.



Implementation considerations

- During train time
  - nothing to do
- During test time
  - Classification can be very slow when finding the k nearest neighbours.
  - In a trivial implementation it could require N number of comparisons, where N is the size of the train dataset
  - By using indexing we can speed up this look up process

Indexing trick in k-NN

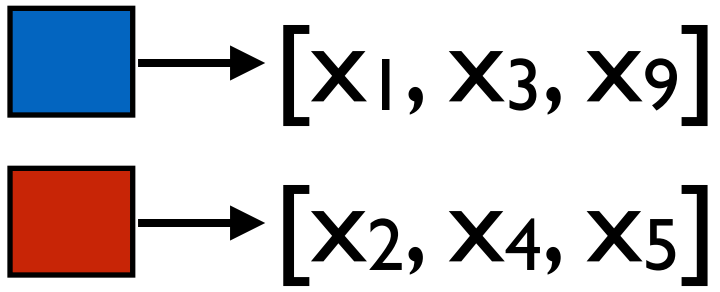


Let us assume that the test instance has a red feature and a blue feature

We would like to find train instances that are similar to this test instance

Obviously, only train instances that have at least a red or a blue feature will have non-zero similarity with this test instance

Let us create an index that lists which train instances have which color features



We only need to measure similarity between these train instances

Exploits the spareness in feature vectors