CORK INSTITUTE OF TECHNOLOGY
INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

Machine Learning

Machine Learning

Lecture: Instance-Based Learning

Ted Scully

k - Nearest Neighbour Distance Metric

- There are a number of distance measurements used to determine similarity:
  - Euclidean Distance
  - Manhattan Distance
  - Minkowski
  - Hamming Distance

Distance Metrics

- An important aspect of k-NN algorithms is how we determine which instances are the nearest to the target case. Thus, the distance metric is a measure of the similarity between two cases.

- Common distance metrics include:
  - Euclidean
  - Manhattan
  - Minkowski

- To help illustrate the various metrics let's assume we have the dataset below with n features and two instances p and q

	Feature 1	Features 2	….	Feature n
p	p_1	p_2	……	p_n
q	q_1	q_2	…….	q_n

Euclidean Distance Metric

- The most common measure of distance is **Euclidian distance**: which measures the straight-line distance between two points.
- If **p** = <p_1, p_2,..., p_n> and **q** = <q_1, q_2,..., q_n> are two points in Euclidean n-space, then the distance (d) from p to q, or from q to p is given by

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

It is important to understand that p and q here represent two data instances. Each instance consisting of a finite set of features. The instance q has the features q1, q2, ... qn.

Euclidean Distance Metric

Movie title	# of kicks	# of kiss	
California Man	3	104	
He's Not Really into Dudes	2	100	
Beautiful Woman	1	81	
Kevin Longblade	101	10	
Robo Slayer 3000	99	5	
Amped II	98	2	Action
?	18	90	Unknown

Movie title	Distance to movie "?"
California Man	20.5
He's Not Really into Dudes	18.7
Beautiful Woman	19.2
Kevin Longblade	115.3
Robo Slayer 3000	117.4
Amped II	118.9

Distance between **California man** (3, 104) and the **query** instance (18, 90) would be:

Euclidean Distance Metric

Movie title	# of kicks	# of kiss	
California Man	3	104	
He's Not Really into Dudes	2	100	
Beautiful Woman	1	81	
Kevin Longblade	101	10	
Robo Slayer 3000	99	5	
Amped II	98	2	Action
?	18	90	Unknown

Movie title	Distance to movie "?"
California Man	20.5
He's Not Really into Dudes	18.7
Beautiful Woman	19.2
Kevin Longblade	115.3
Robo Slayer 3000	117.4
Amped II	118.9

Distance between **California man** (3, 104) and the **query** instance (18, 90) would be

$$\sqrt{(3-18)^2 + (104-90)^2} = \sqrt{225+196} = 20.5$$

Manhattan Distance Metric

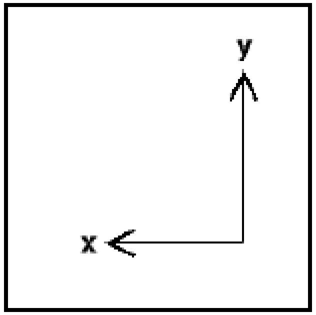
- Manhattan distance measures distance **parallel to each axis**, not diagonally (in downtown Manhattan, to get from one point to another you generally walk North-South and East-West, rather than 'as the crow flies').

- In other words take the sum of the absolute values of the differences of the coordinates

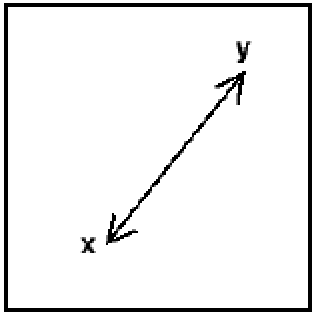
- d(p, q) = $|q_1 - p_1| + |q_2 - p_2| + |q_n - p_n| = \sum_{i=1}^{n} |q_i - p_i|$

Manhattan Distance Metric

- Lets assume we have a simple dataset containing three instances as follows. We are also given the query instance below. Calculate the Manhattan and Euclidean distance between the **first training example** and the **query**



Manhattan



Euclidean

Area	Weight	Height	Capacity
10	8	4	14
12	10	6	12
14	9	4	11

Area	Weight	Height	Capacity
5	4	4	10

Area	Weight	Height	Capacity
10	8	4	8
12	10	6	12
14	9	4	11

Area	Weight	Height	Capacity
5	4	4	3

Manhattan Distance Metric

- <u>Euclidean</u>
  - $(10-5)^2+(8-4)^2+(4-4)^2+(8-3)^2$
  - 57
  - Square root of 57 = 7.55
- <u>Manhattan</u>
  - $|10-5|+|8-4|+|4-4|+|8-3|$
  - 5+4+0+5 = 14

Area	Weight	Height	Capacity
10	8	4	8
12	10	6	12
14	9	4	11

Area	Weight	Height	Capacity
5	4	4	3

Minkowski Distance

- The Minkowski distance between a feature vector **p** = <p_1, p_2,..., p_n> and another feature vector **q** = <q_1, q_2,..., q_n> is defined as:

- $d(p,q) = \left(\sum_{i=1}^{n} |p_i - q_i|^a\right)^{\frac{1}{a}}$

- In the above equation *a* is an integer. Consider the case when a = 1 or a = 2. Any comments.

Minkowski Distance

- The Minkowski distance between a feature vector **p** = <p_1, p_2,..., p_n> and another feature vector **q** = <q_1, q_2,..., q_n> is defined as:

- $d(p,q) = \ (\sum_{i=1}^{n}|p_i - \ q_i|^a)^{\frac{1}{a}}$

- In the above equation *a* is an integer. Consider the case when a = 1 or a = 2. Any comments.
  - For a = 1 we get the Manhattan distance and for a = 2 we get the Euclidean distance.
  - Minkowski Distance is a generalization of the Euclidean and Manhattan distance metrics.

Hamming Distance Metric

- Allows us to deal with problems that have features that are categorical (discrete) rather than **continuous-valued.**
- The value 0 is assigned for each feature where both cases have the same value, 1 for each where they are different.

- d(p_i, q_i) = $$\begin{matrix} 0 \ if \ qi == pi \\ 1 \ if \ qi \ != pi \end{matrix}$$

ID	Outlook	Temp	Hum	Windy	Play?
A	sunny	hot	high	false	no
B	sunny	hot	high	true	no
C	overcast	hot	high	false	yes

What is distance between B and C?

Outlook (1) + Temp(0) + Hum(0) + Windy(1) = 2

Heterogeneous Distance Metric

- When we have a dataset that is a mix of discrete and continuous valued features we can combine distance measures such as Manhattan and Hamming.

- d(pi, qi) = $$\begin{array}{c} |q_i - p_i| \; if \;\; feature \; i \; is \; continuous \\ 0 \; if \;\; feature \; i \; is \; discrete \; and \; qi == pi \\ 1 \; if \;\; feature \; i \; is \; discrte \; qi \, ! = pi \end{array}$$

The Hamming metric is limited and provides limited information about the difference between features.

Building a K Nearest Neighbour Classifier

- **Step 1. Read information from a dataset**
  - Read data from a dataset containing classified instances. Read each feature of the dataset as well as corresponding class.

- **Step 2. Determine distance between each dataset entry and the query instance**
  - Use a suitable distance metric to calculate the distance between the query instance and all k neighbours

- **Step 3. Classify the query instance**
  - Identify k nearest data instances. Assign query instance category corresponding to most common category.

Problems Measuring Distance 1 -Scale

- Since the performance of k-NN is strongly dependent on the choice of distance metric, you need to be aware of some pitfalls.

- The first problem arises when the **features are different** from each other.
- For example, if one feature has a range between **0 and 1** and another feature has a range between **0 and 10, 000**, it hardly makes sense to add them as would happen with Euclidian or Manhattan distance metrics (for example, salary and age).

- What is the main problem that arises from the above situation?

Problems Measuring Distance (1)

- **Problem 1: Scaling**
  - Feature A has range 1-10
    Feature B has range 1-1000
  - Feature B will dominate calculations

- Example, lets calculate the distance between data instance 1 and 2
  - Data instance 1 = (5.5, 787)
  - Data instance 2 = (7.5, 567)

Problems Measuring Distance (1)

- **Problem 1: Scaling**
  - Feature A has range 1-10
    Feature B has range 1-1000
  - Feature B will dominate calculations

- Example, lets calculate the distance between data instance 1 and 2
  - Data instance 1 = (5.5, 787)
  - Data instance 2 = (7.5, 567)

$$\sqrt{(5.5-7.5)^2+(787-567)^2}$$

$$\sqrt{16+48400}$$

Problems Measuring Distance (1)

- **Problem 1: Scaling**
  - Feature A has range 1-10
    Feature B has range 1-1000
  - Feature B will dominate calculat

We can see below that the second feature is entirely dominating the distance calculation simply because it has a larger range of values compared to the first feature.

We don’t want our model to bias toward a particular feature simply because the range happens to be larger.

- Example, lets calculate the distance
  - Data instance 1 = (5.5, 787)
  - Data instance 2 = (7.5, 567)

$$\sqrt{(5.5-7.5)^2+(787-567)^2}$$

$$\sqrt{16+48400}$$

Problems Measuring Distance (1)

- Solution:

  - Normalise all dimensions independently (scale data so that it has a maximum and minimum range)

  - Using range normalization we identifying the minimum and maximum value for a specific feature. We can then apply the following formul.

    - $\boldsymbol{newValue = \frac{originalValue - minValue}{maxValue - minValue}}$

- $newValue = \frac{originalValue - minValue}{maxValue - minValue}$

- Problem 1: Scaling
  - Feature A has range 1-10
    Feature B has range 1-1000
- Normalise variables
  - Feature A
    - (5.5 - 1)/(10-1) = 0.5
    - (7.5 -1)/(10 -1) = 0.72
  - Feature B
    - (787-1)/(1000-1) = 0.78
    - (567-1)/(1000-1) = 0.56

- Before Normalization
  - Data instance 1 = (5.5, 787)
  - Data instance 2 = (7.5, 567)

- After Normalization
  - Data instance 1 = (0.5, .78)
  - Data instance 2 = (0.72, 0.56)

$$\sqrt{(0.5-0.72)^2+(0.78-0.56)^2}$$

$$\sqrt{0.048+0.048}$$

Problems Measuring Distance (1)

- When we normalize the train data, it is also important to understand:
  - We normalize each feature **independently**
  - We must normalize the test data using the same parameters for max and min (that is we still use the minValue and maxValue from the original training set).

Problems Measuring Distance – Irrelevant Features

- The other principal problem is that **all features are included equally** in the calculations we have looked at, even though some features may be **redundant** or **less relevant**.
- Therefore, a number of features may skew the result even through they might of little or no impact to the classification.

- **Solution 2A**:
  - **Assign weighting** to each dimension (Optimise weighting to minimise error)
- **Solution 2B**:
  - Give some dimensions **0 weight** (Feature subset solution)

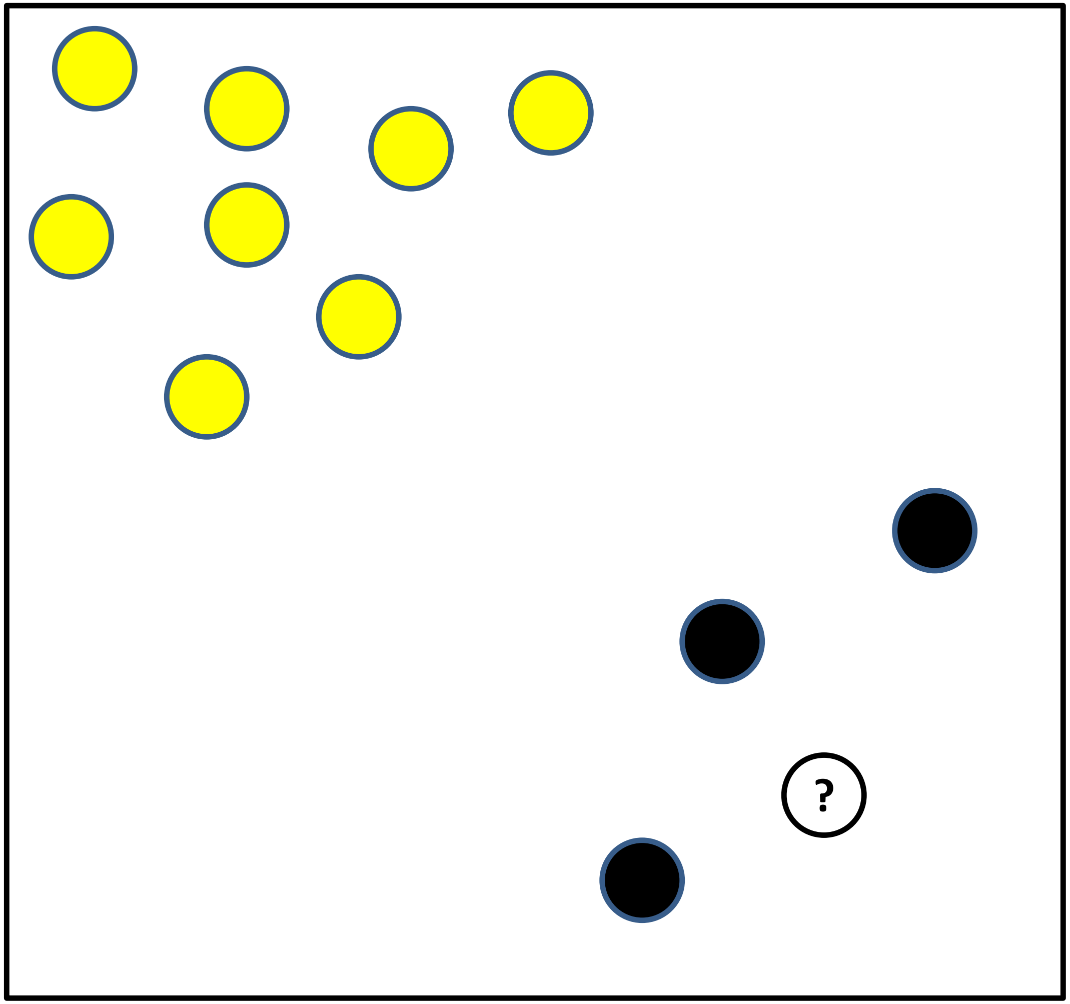
- Either way, since we cannot know in advance what weighting to give dimensions, systematic repeated experiments are needed to optimise them.
- We will look feature selection in more detail later in the module.

Problem with KNN

Assume in the following example that we are using **k=7**?

What are the potential problems?

What might you do to address this problem?



Distance-Weighted kNN

Give each neighbour weight: inverse of distance from target

Distance Weighted k-NN (Regression)

- It is the vote of each neighbour that is weighted according to how close it is to the target, so the **closer neighbour's influence the prediction more**.

- We can if we wish use all cases as neighbours, since those very far from the target will have little influence on the prediction, but none will be completely ignored.
- We do the following for a regression problem.

Given a query instance x_q,

$$f(\mathbf{x}_q) := \frac{\sum_{i=1}^{k} w_i\, f(\mathbf{x}_i)}{\sum_{i=1}^{k} w_i}$$

Where

$$w_i = \frac{1}{d(\mathbf{x}_q, \mathbf{x}_i)^2}$$

Distance Weighted k-NN (Regression)

- It is the vote of each neighbour that is weighted according to how close it is to the target, so the **closer neighbour's influence the prediction more**.

- We can if we wish use all cases as neighbours, since those very far from the target will have little influence on the prediction, but none will be completely ignored.
- We do the following for a regression problem.

Given a query instance x_q,

$$f(\mathbf{x}_q) := \frac{\sum_{i=1}^{k} w_i\, f(\mathbf{x}_i)}{\sum_{i=1}^{k} w_i}$$

Where

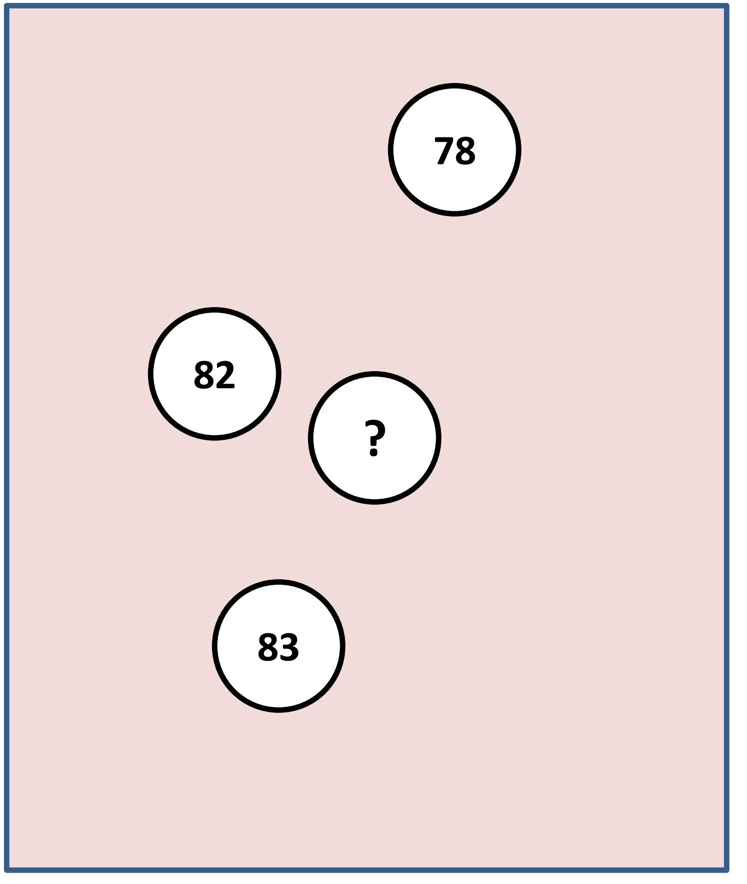
$$w_i = \frac{1}{d(\mathbf{x}_q, \mathbf{x}_i)^2}$$

Here you will notice that we use the inverse distance to the power of 2 (n=2). This is typical. However, we can also use n=1, n=3, etc

Distance Weighted k-NN Algorithm Regression Example

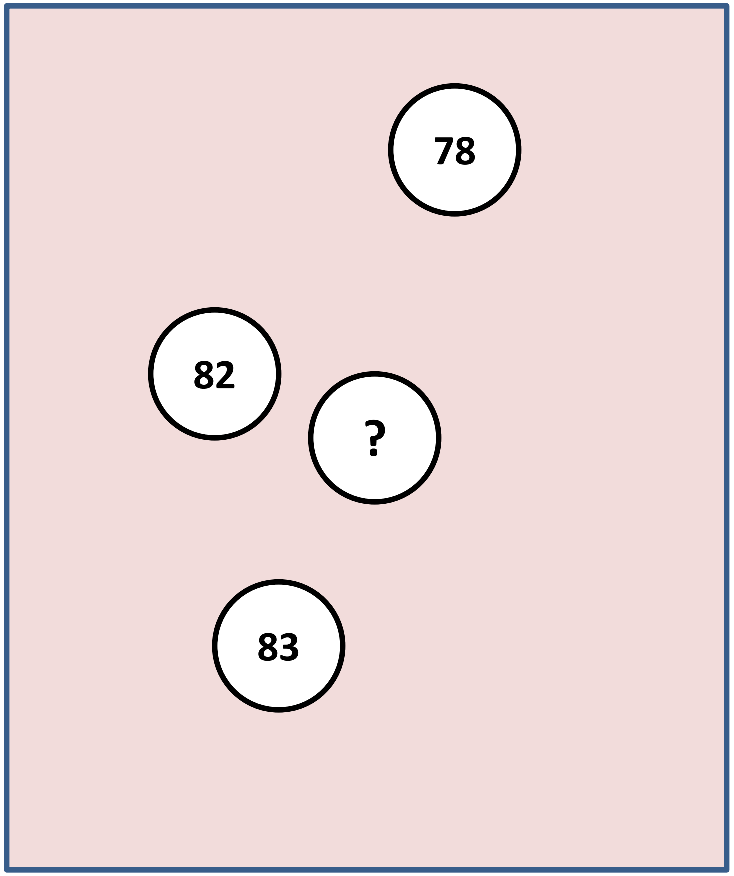
- Consider the basic regression example depicted in the slide.
- Distance between query instance and:
  - Case 82 is 2
  - Case 83 is 4
  - Case 78 is 6
- What is the value of the query instance.

$$f(\mathbf{x}_q) := \frac{\sum_{i=1}^{k} w_i f(\mathbf{x}_i)}{\sum_{i=1}^{k} w_i}$$



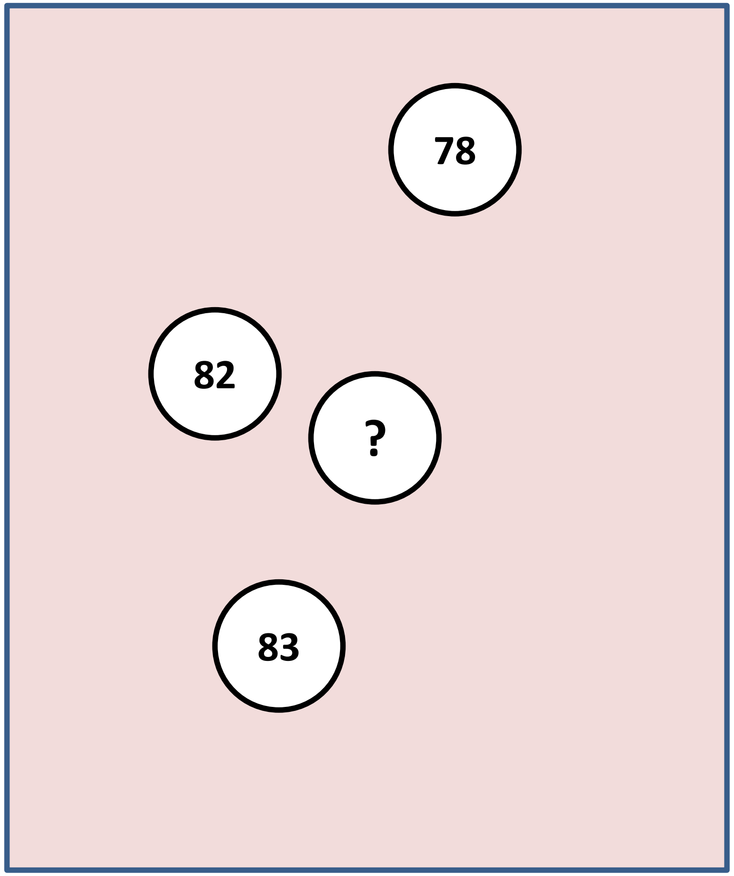
Distance Weighted k-NN Algorithm Regression Example

- Distance between query instance and:
  - Case 82 is 2
  - Case 83 is 4
  - Case 78 is 6



Distance Weighted k-NN Algorithm Regression Example

- Distance between query instance and:
  - Case 82 is 2
  - Case 83 is 4
  - Case 78 is 6



- ((1/4)(82) + (1/16)(83) + (1/36)(78)) / (1/4 + 1/16 + 1/36)
- = 27.854/0.34027777
- =81.856

Distance Weighted k-NN (Classification)

- We iterate through each class. For a specific class we identify each instances amongst the k nearest instances that belong to that class. We then add up the inverse distance for each of the identified instances.

- The class that results in the largest value is the selected class for the new query instance.

$$vote(c_j) := \sum_{i=1}^{k} \frac{1}{d(\mathbf{x}_q, \mathbf{x}_i)^n} (c_i, c_j)$$

(y_i, y_j) returns 1 if the class labels match and 0 otherwise

What do you think will be the impact of n (n must be a positive number greater than or equal to 1)

Vote(Purple Class) (n=1)

Purple

1/10 +
1/9 +
1/5 = 0.41

Yellow
1/5+
1/6+
1/5 = 0.566

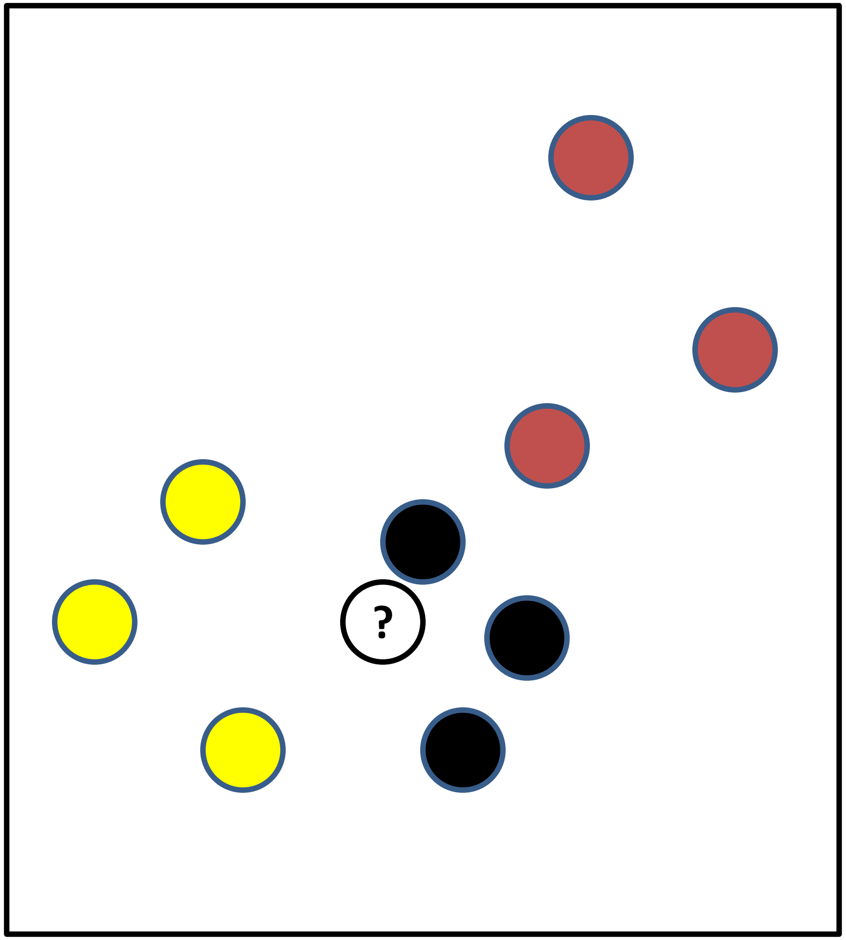
Black
1/1+
1/2+
1/2= 2

Purple

10
9
5

Yellow
5
6
5

Black
1
2
2



$$vote(y_j) := \sum_{i=1}^{k} \frac{1}{d(\mathbf{x}_q, \mathbf{x}_i)^n} (y_i, y_j)$$

Result of Voting

- Result of voting is
- V(Purple) = 0.41
- V(Yellow) = 0.566
- V(Black) = 2

- Therefore the query instance is classified as a **Black class.**

Assessing the Performance of a Regression Model

- So far we have a basic measure that we can use for assessing the performance of a classification model, which is the **accuracy**.

- Accuracy = number of test instances correctly classified/ total number of test instances.

- Later in the module we will look more comprehensively at evaluation metrics.

- So what is a common **evaluation metric** we can use for **regression**?

Basic Measures of Error (Regression)

- The $\mathbf{R^2}$ coefficient compares the **performance of a model on a test set (sum of squared residuals)** with the performance of an imaginary model that always predicts the **average values from the test set (total sum of squares)**.

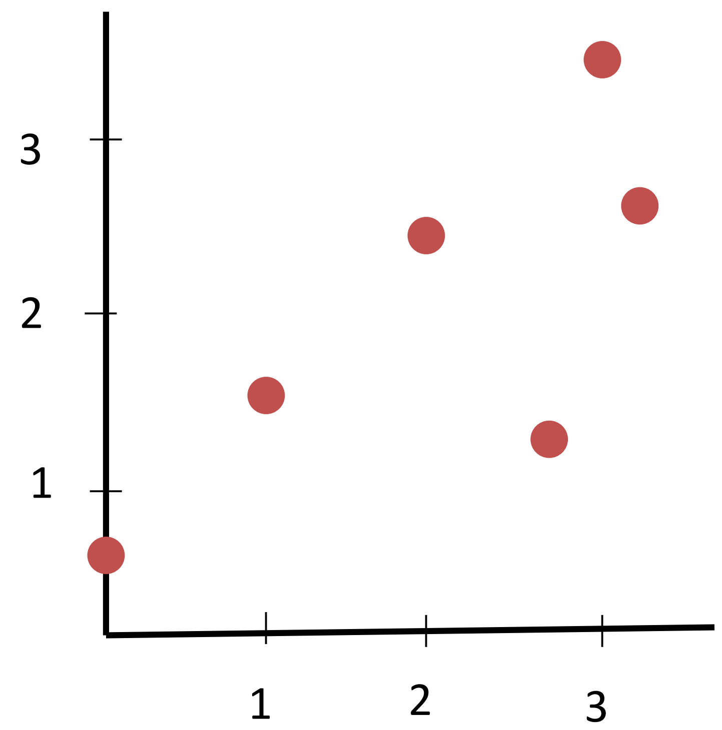
- The $\mathbf{R^2}$ coefficient is calculated as:

$$R^2 = 1 - \frac{sum\ of\ squared\ residuals}{total\ sum\ of\ squares}$$

- Where

$$sum\ of\ squared\ residuals = \sum_{i=0}^{m} \left(f\left(x^i\right) - y^i\right)^2$$

$$total\ sum\ of\ squares = \sum_{i=0}^{m} \left(\bar{y} - y^i\right)^2$$



3
2
1
1
2
3

Basic Measures of Error (Regression)

- The **R^2** coefficient values typically fall in the range [0, 1) and larger values indicate better model performance.

- The **worse the model** produced, the closer the sum of square residuals value will be to the total sum of squares value. Consequently the **smaller the total R^2**.

- The **better the model** the smaller the squared residuals (smaller error in the model) and the **larger the R^2** value.

- While it is rare, the model produced could be worse than the total sum of squares. In this case the R^2 would be **negative**. The worse the model the lower the R^2 values. It means that whatever model that you came up with is worse than predicting the mean (not a good sign!).

Eager vs. Lazy Learner (1)

- Eager Learning (Such as Bayesian classifiers, decision trees, neural networks)
  - When given training data, it **constructs a model** for future use in prediction that summarises the data
  - **Slow in model construction**, generally quick when classifying unseen instances

- Instance based learning often referred to as lazy Learners
  - **No** explicit global **model** constructed
  - **Calculations deferred** until new case to be classified
  - Creates **many local approximations**, whereas eager learners create a global approximation
  - **Significant calculations** needed to take place for each new query (can be slow)

When to use k-Nearest Neighbour

- Primary Benefits
  - **Comprehensibility**: easy to understand
  - Relatively straight-forward to **implement**
  - Can easily handle **multi-class** datasets
  - Effective classifiers for **complex target functions** => good for diverse concepts.
  - Can be used for both **regression and classification** problems and can mix **feature types** in the one dataset so they are very flexible.

- Consider using when:
  - Moderate number of **training instances**
  - Moderate number of **features** per instance (< 20) [Note: If dealing with datasets with a large number of features we can perform dimensionality reduction and still use a k-NN approach]