COMP9321:
Data services engineering

Week 7: Classification

Term1, 2022

By Mortada Al-Banna, CSE UNSW

Machine Learning for Data Analytics

1. **Define** and **Initialize** a Model
2. **Train** your Model (using your training dataset)
3. **Validate** the Model (by prediction using your test dataset)
4. Use it: **Explore** or **Deploy** as a web service
5. **Update** and **Revalidate**

https://www.slideshare.net/SetuChokshi/machine-learning-101-74017810

Supervised Learning

We are given input samples (X) and output samples (y) of a function $y = f(X)$.

We would like to “learn” f, and evaluate it on new data.

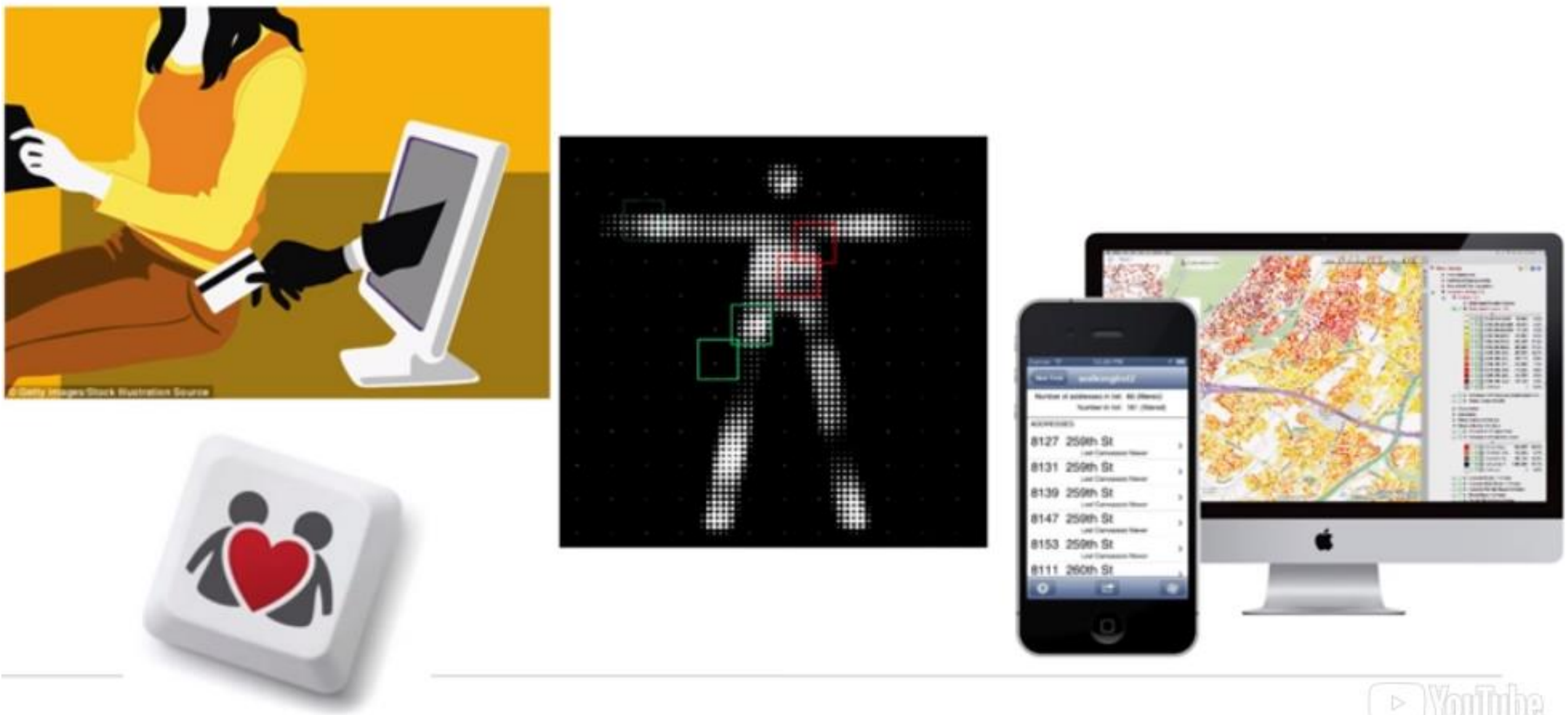
- **Classification:** y is discrete (class labels).
- **Regression:** y is continuous, e.g. linear regression.

Classification

- Supervised Learning
- You need the data labelled with the correct answer to train the algorithm
- Trained classifiers then can map input data to a category.

https://www.slideshare.net/brianjlange/machine-learning-in-5-minutes-classification

Classification Examples



Credit: Foundations of Data Science: Prediction and Machine Learning course offered by Berkley
https://courses.edx.org/courses/course-v1:BerkeleyX+Data8.3x+2T2018/course/

k-Nearest Neighbour (k-NN)

The KNN classifier is a **non parametric** and **instance-based** learning algorithm.

Non-parametric means it makes no explicit assumptions about the functional form of how the prediction is made, avoiding the dangers of mismodeling the underlying distribution of the data.

Instance-based learning means that our algorithm doesn’t explicitly learn a model. Instead, it chooses to memorize the training instances which are subsequently used as “knowledge” for the prediction phase. Concretely, this means that only when a query to our database is made (i.e. when we ask it to predict a label given an input), will the algorithm use the training instances to spit out an answer.

https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/

https://www.section.io/engineering-education/parametric-vs-nonparametric/

k-Nearest Neighbors

Given a query item:

Find k closest matches
in a labeled dataset ↓

k-Nearest Neighbors

Given a query item:

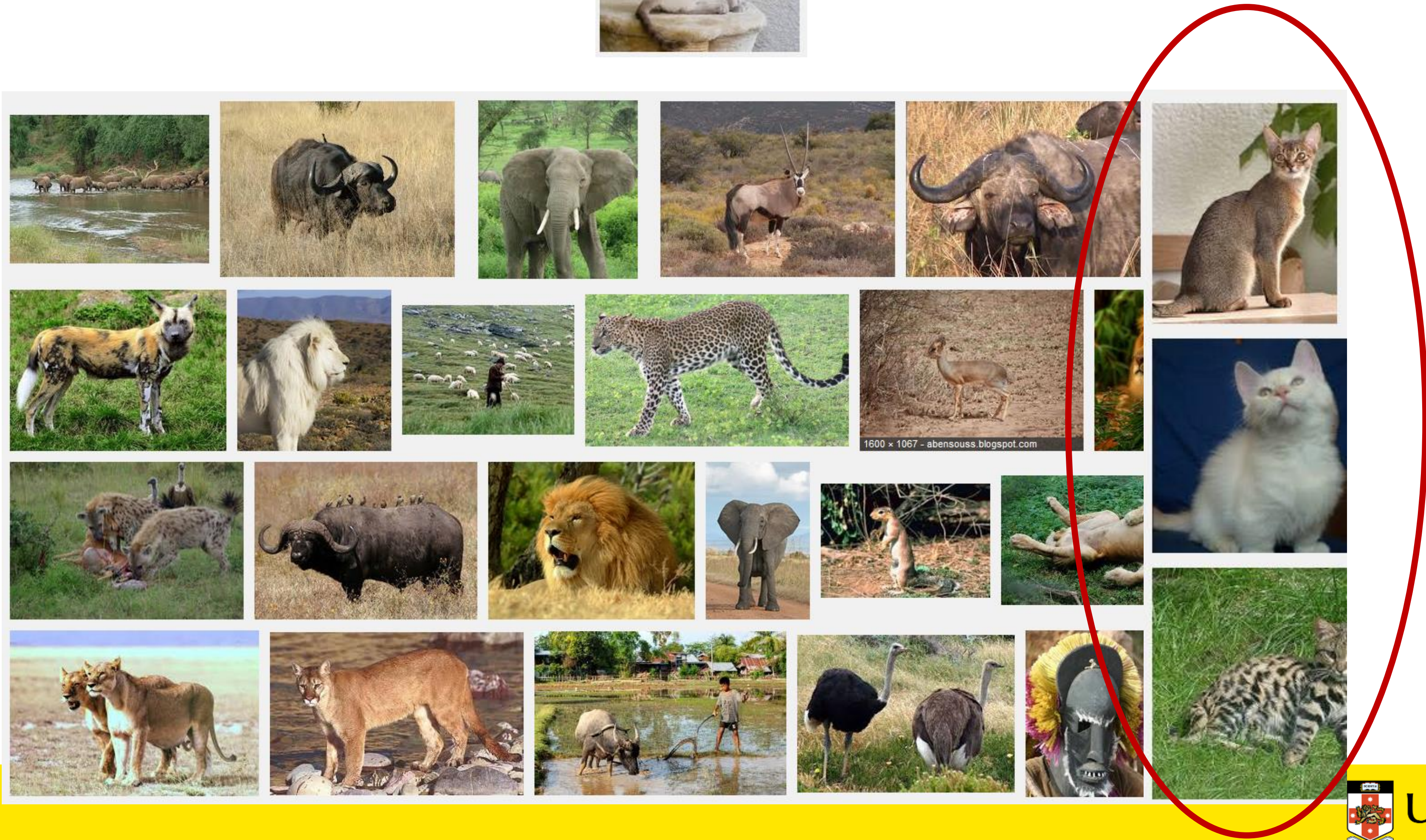
Find k closest matches

Return the most

Frequent label

k-Nearest Neighbors

k = 3 votes for “cat”



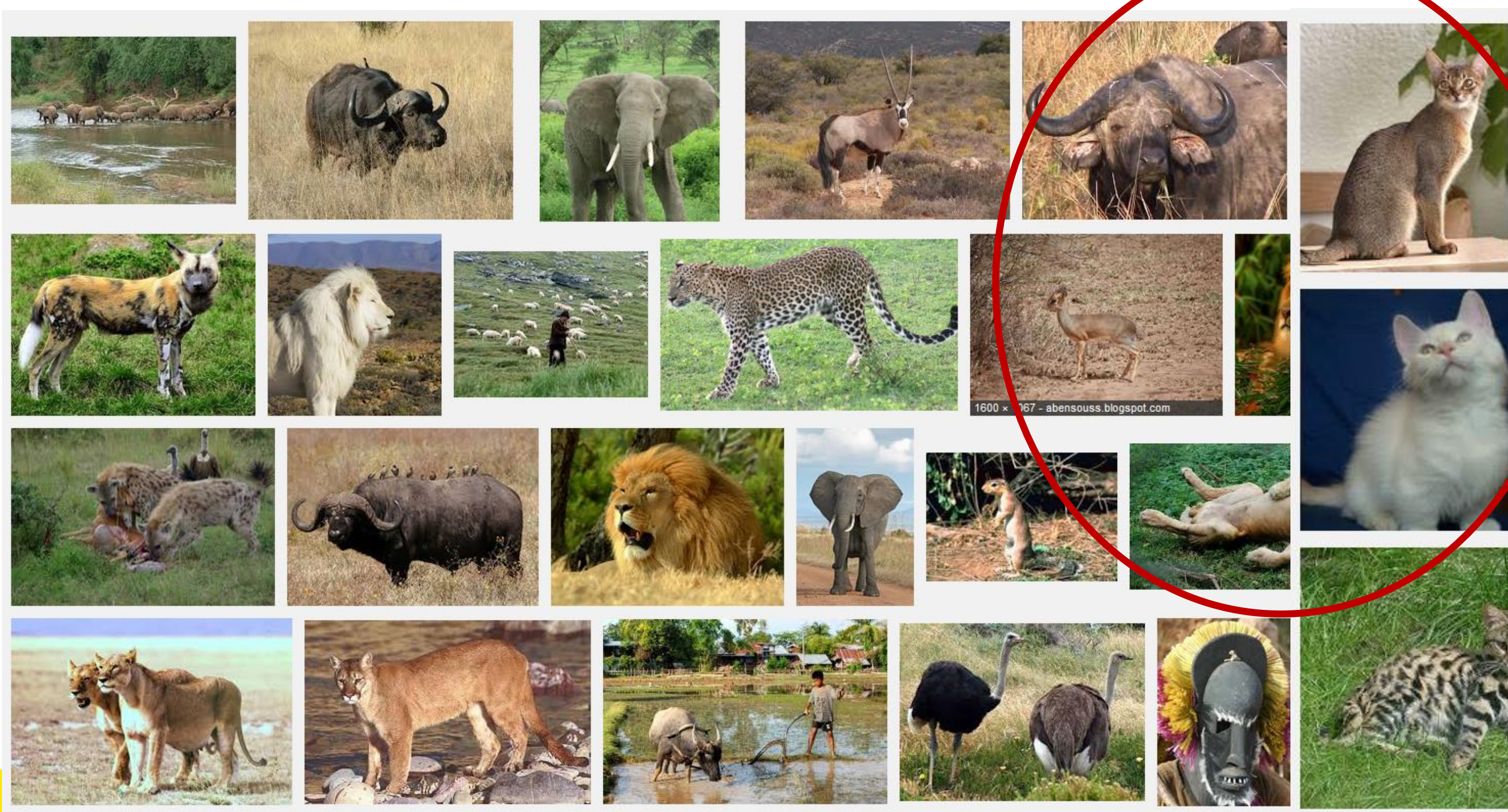
k-Nearest Neighbors

2 votes for cat,

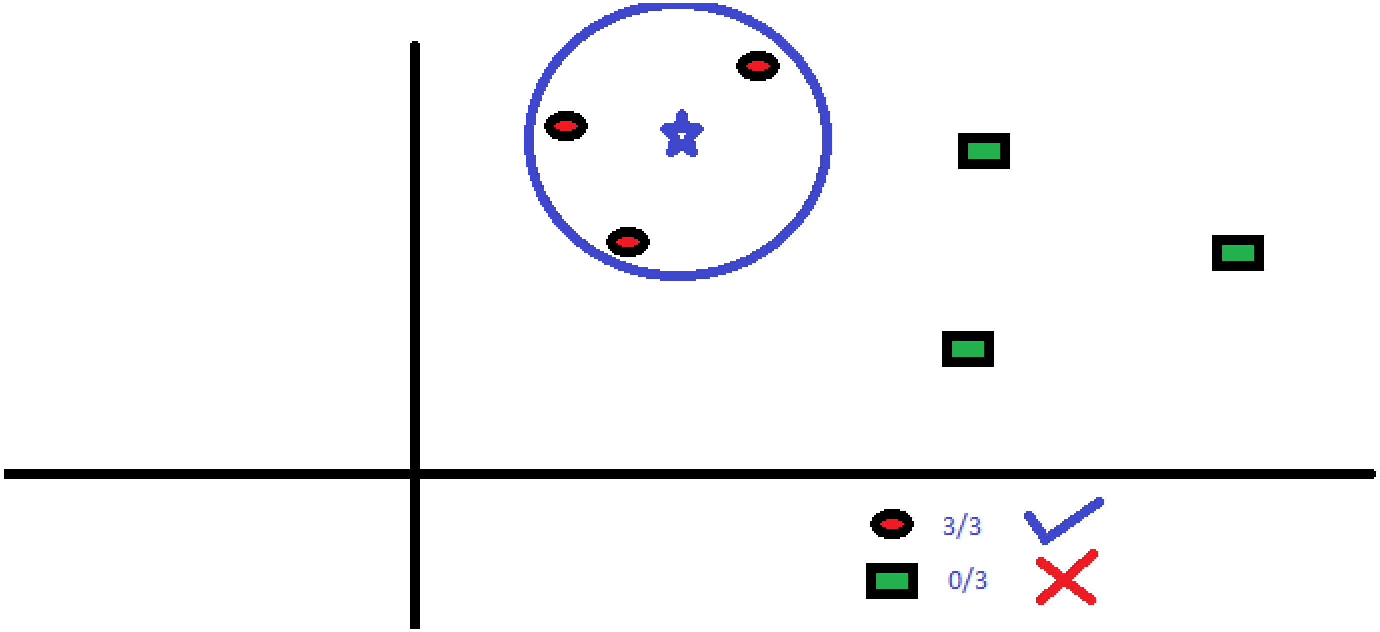
1 each for Buffalo,

Deer, Lion

Cat wins…



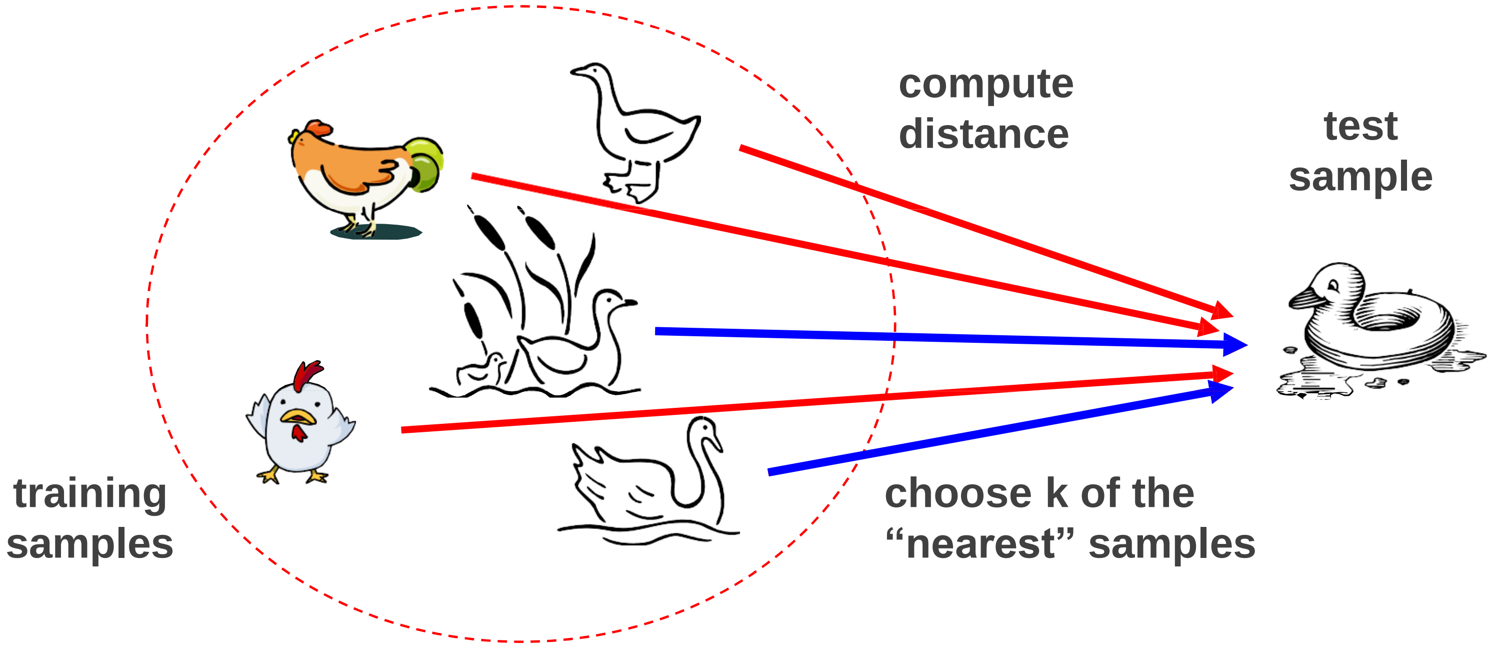
k-Nearest Neighbour (k-NN)



https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

Nearest neighbor classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



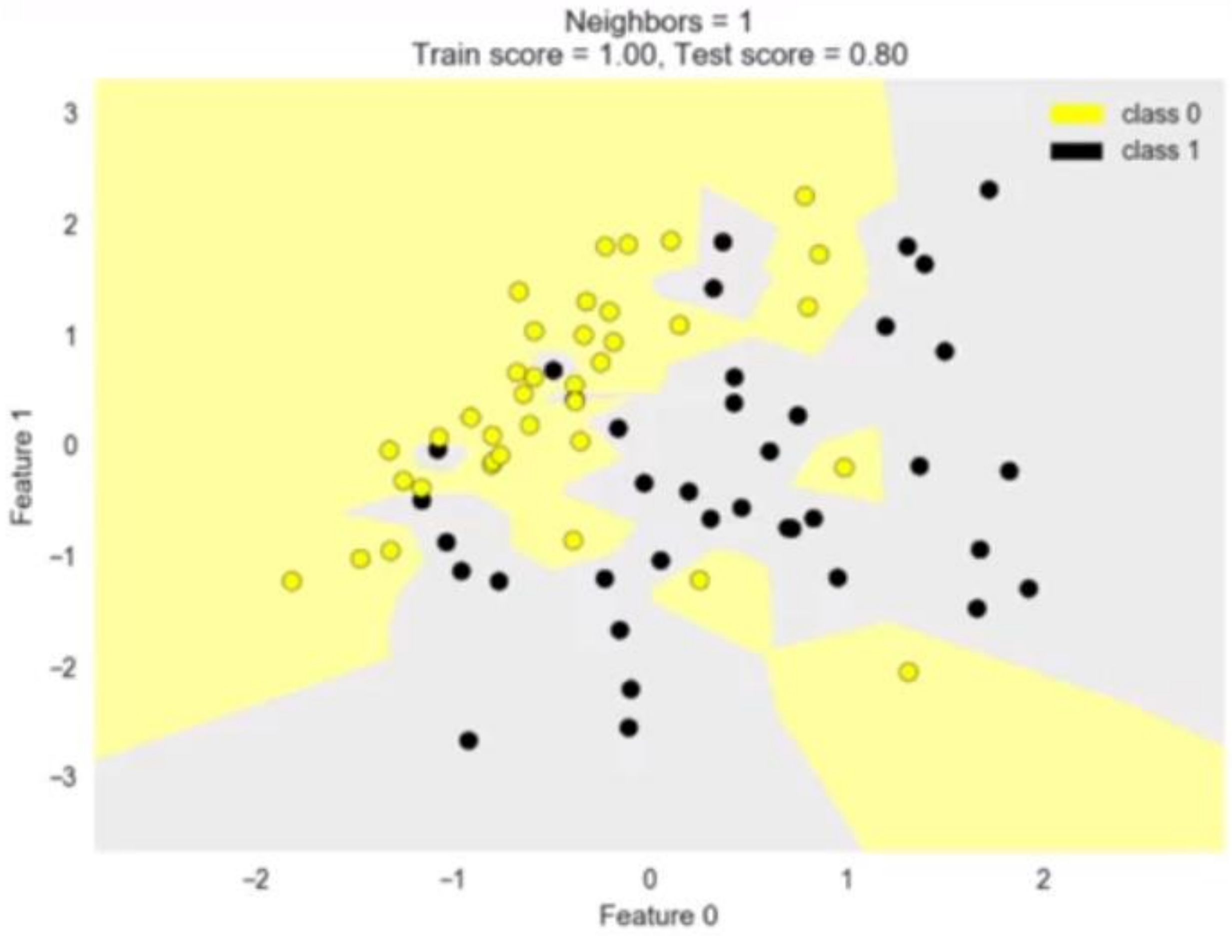
k- Nearest Neighbour Classifier Algorithm

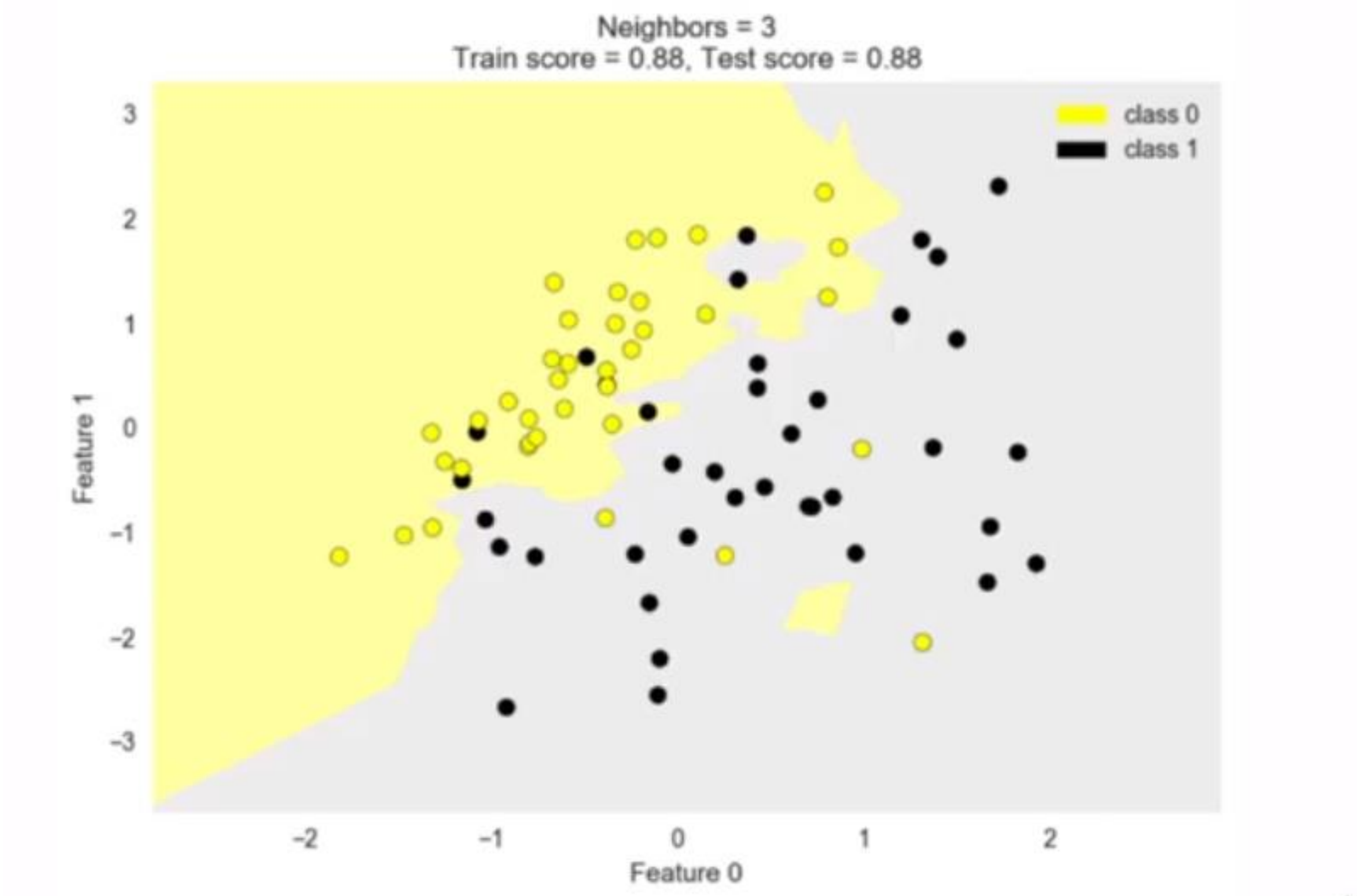
Give a training set X_train with lables y_train and given a new instance x_test to be classified:

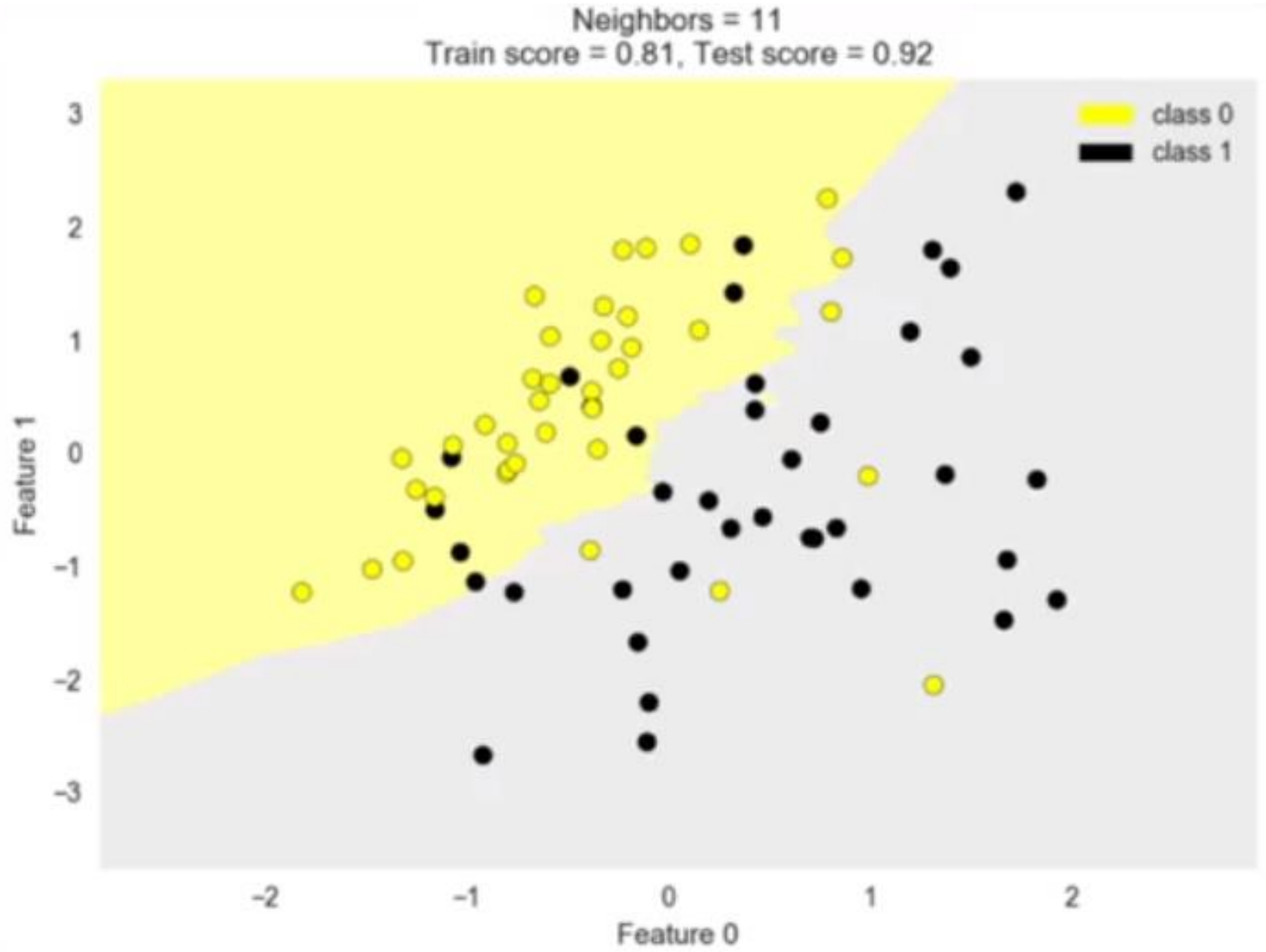
1. Find the most similar instances (let's call then X_NN) to x_test that are in X_train.
2. Get the labels y_NN for the instances in X_NN.
3. Predict the label for x_test by combining the labels y_NN (e.g., using majority rule)

Nearest Neighbour Need Four things Specified

1. A distance Metric (e.g., Euclidean)
2. How many nearest neighbours to look at (e.g., Five)
3. Optional Weighting function on the neighbours points (e.g., closer points are weighted higher than farther points)
4. How to aggregate the classes of neighbours points (e.g., simple majority voting)

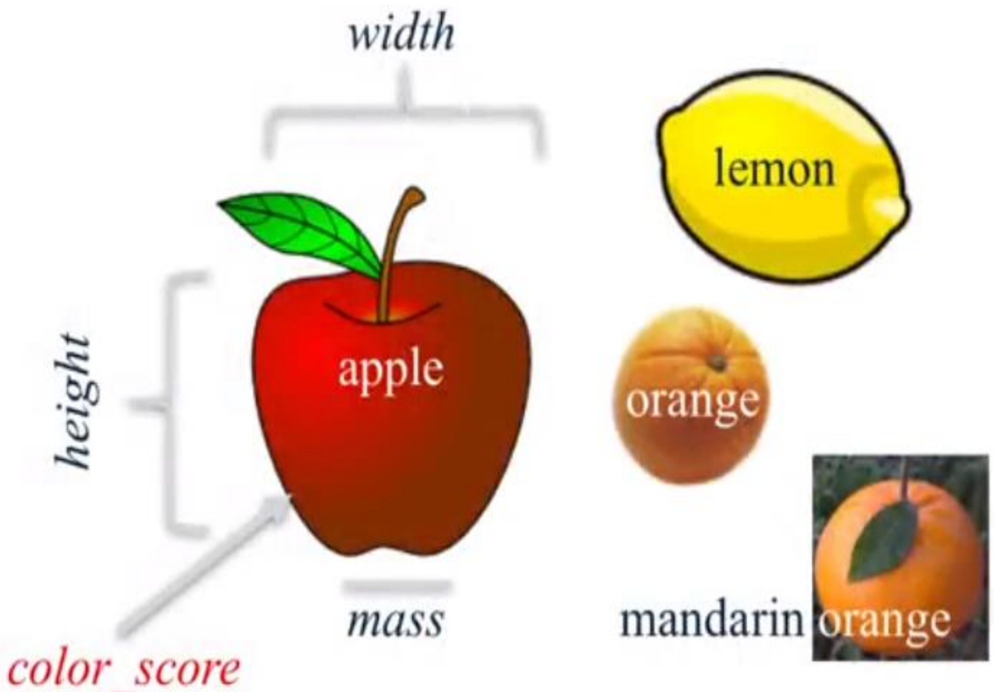

Neighbors = 1
Train score = 1.00, Test score = 0.80
class 0
class 1
Feature 1
Feature 0
3
2
1
0
−1
−2
−3
−2
−1
0
1
2


Neighbors = 3
Train score = 0.88, Test score = 0.88
class 0
class 1
Feature 1
Feature 0
3
2
1
0
-1
-2
-3
-2
-1
0
1
2


Neighbors = 11
Train score = 0.81, Test score = 0.92
class 0
class 1
Feature 1
Feature 0
3
2
1
0
−1
−2
−3
−2
−1
0
1
2

Classification Data Set Example

The Fruit Dataset



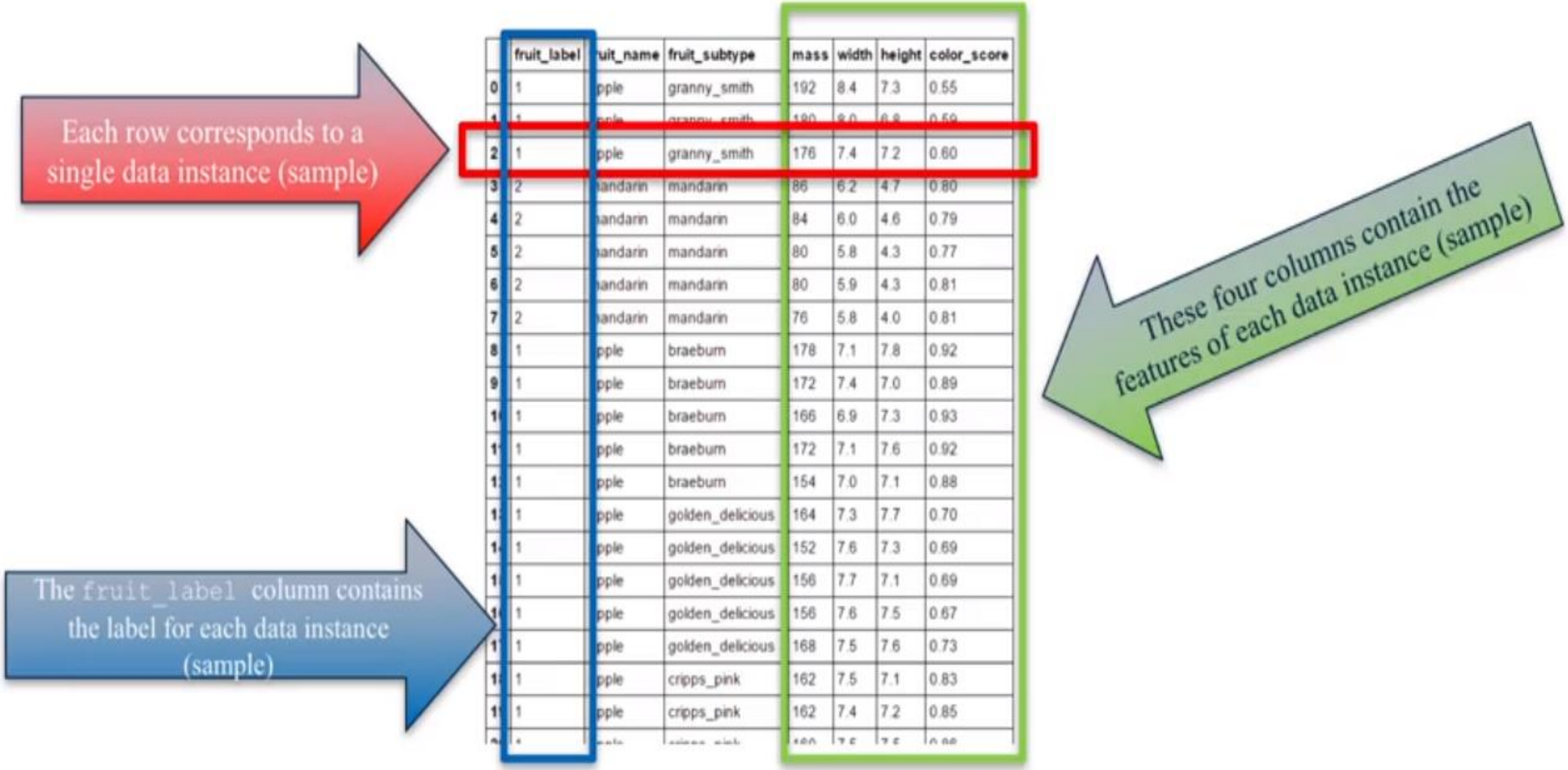
	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89
10	1	apple	braeburn	166	6.9	7.3	0.93
11	1	apple	braeburn	172	7.1	7.6	0.92
12	1	apple	braeburn	154	7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69
15	1	apple	golden_delicious	156	7.7	7.1	0.69
16	1	apple	golden_delicious	156	7.6	7.5	0.67

fruit_data_with_colors.txt

Credit: Original version of the fruit dataset created by Dr. Iain Murray, Univ. of Edinburgh

Example taken from Applied Machine Learning in Python course by the University of Michigan
https://www.coursera.org/learn/python-machine-learning

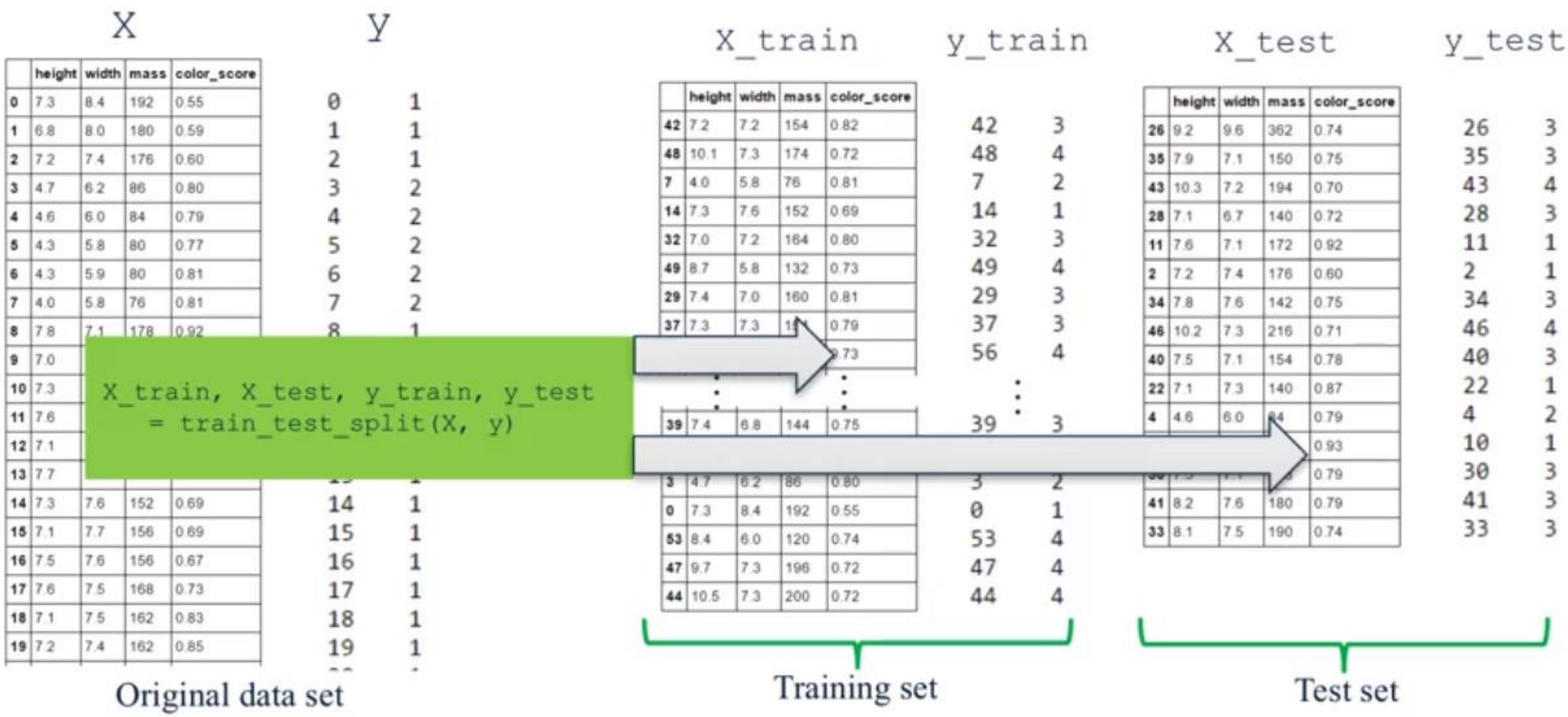
Data As a Table



	fruit_label	uit_name	fruit_subtype	mass	width	height	color_score
0	1	pple	granny_smith	192	8.4	7.3	0.55
1	1	pple	granny_smith	180	8.0	6.8	0.59
2	1	pple	granny_smith	176	7.4	7.2	0.60
3	2	andarin	mandarin	86	6.2	4.7	0.80
4	2	andarin	mandarin	84	6.0	4.6	0.79
5	2	andarin	mandarin	80	5.8	4.3	0.77
6	2	andarin	mandarin	80	5.9	4.3	0.81
7	2	andarin	mandarin	76	5.8	4.0	0.81
8	1	pple	braeburn	178	7.1	7.8	0.92
9	1	pple	braeburn	172	7.4	7.0	0.89
1	1	pple	braeburn	166	6.9	7.3	0.93
1	1	pple	braeburn	172	7.1	7.6	0.92
1	1	pple	braeburn	154	7.0	7.1	0.88
1	1	pple	golden_delicious	164	7.3	7.7	0.70
1	1	pple	golden_delicious	152	7.6	7.3	0.69
1	1	pple	golden_delicious	156	7.7	7.1	0.69
1	1	pple	golden_delicious	156	7.6	7.5	0.67
1	1	pple	golden_delicious	168	7.5	7.6	0.73
1	1	pple	cripps_pink	162	7.5	7.1	0.83
1	1	pple	cripps_pink	162	7.4	7.2	0.85

Example taken from Applied Machine Learning in Python course by the University of Michigan
https://www.coursera.org/learn/python-machine-learning

Training Set and Test Set



Example taken from Applied Machine Learning in Python course by the University of Michigan
https://www.coursera.org/learn/python-machine-learning

Always Remember to inspect your Data

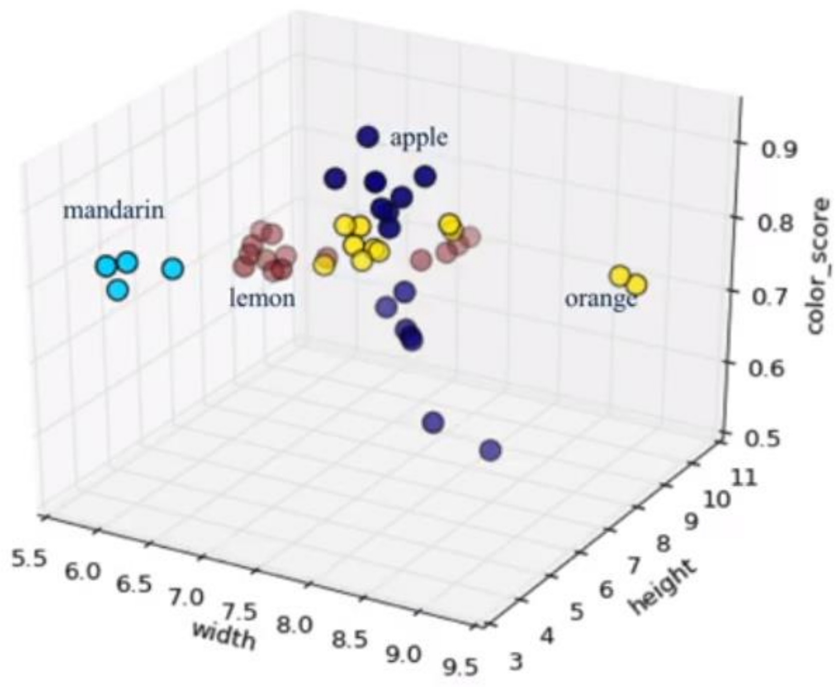
Examples of incorrect or missing feature values

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	192
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	apple	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn		7.4	7.0	0.89
10	1	apple	braeburn		6.9	7.3	0.93
11	1	apple	braeburn		7.1	7.6	0.92
12	1	apple	braeburn		7.0	7.1	0.88
13	1	apple	golden_delicious	[illegible]	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69

Example taken from Applied Machine Learning in Python course by the University of Michigan
https://www.coursera.org/learn/python-machine-learning

Plot your Data

A three-dimensional feature scatterplot



Example taken from Applied Machine Learning in Python course by the University of Michigan
https://www.coursera.org/learn/python-machine-learning

Choosing the k in k-Nearest Neighbors

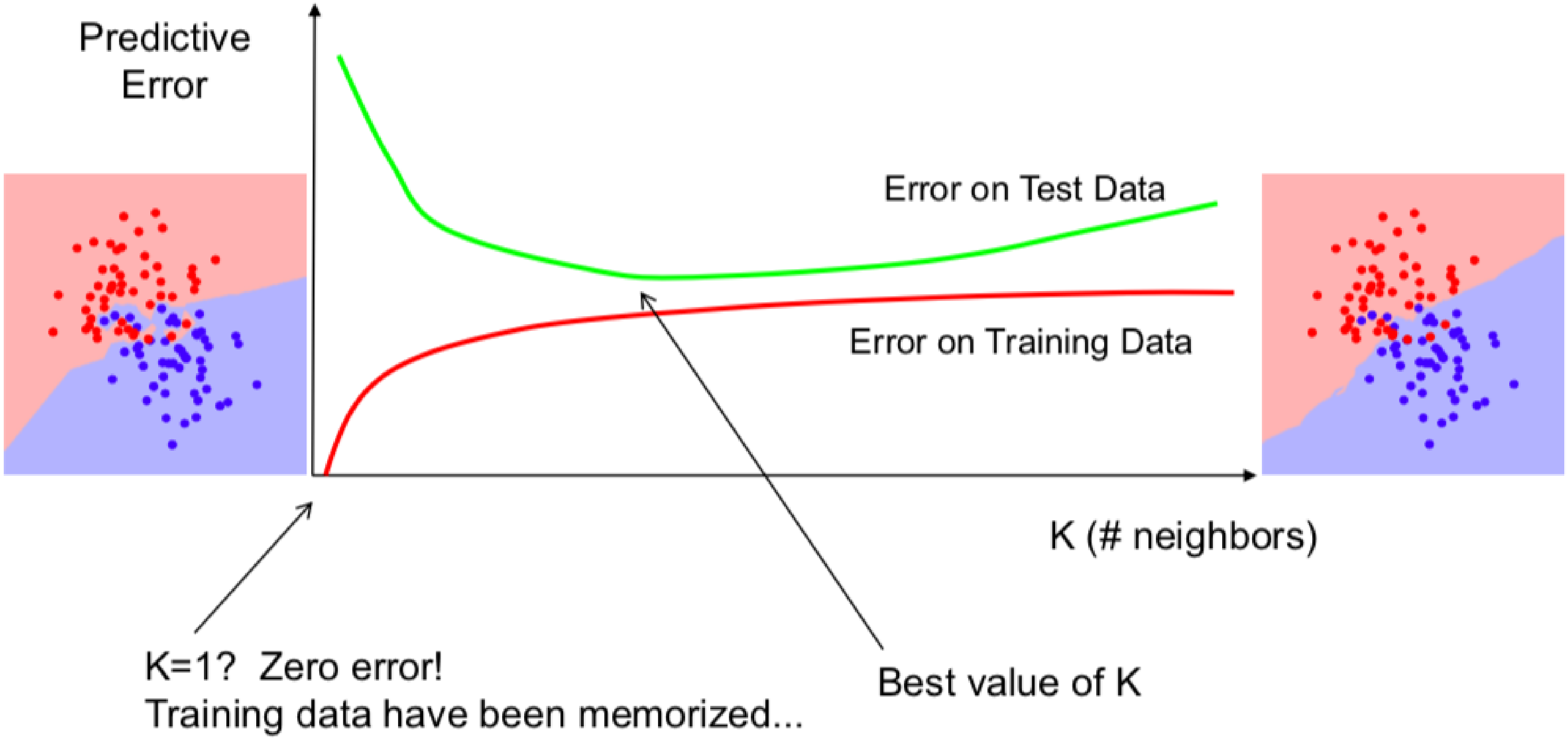
How do we choose k?

- Larger k may lead to better performance
- But if we set k too large we may end up looking at samples that are not neighbors (are far away from the query)
- We can use cross-validation to find k
- Rule of thumb is $k < sqrt(n)$, where n is the number of training examples

[Slide credit: O. Veksler]

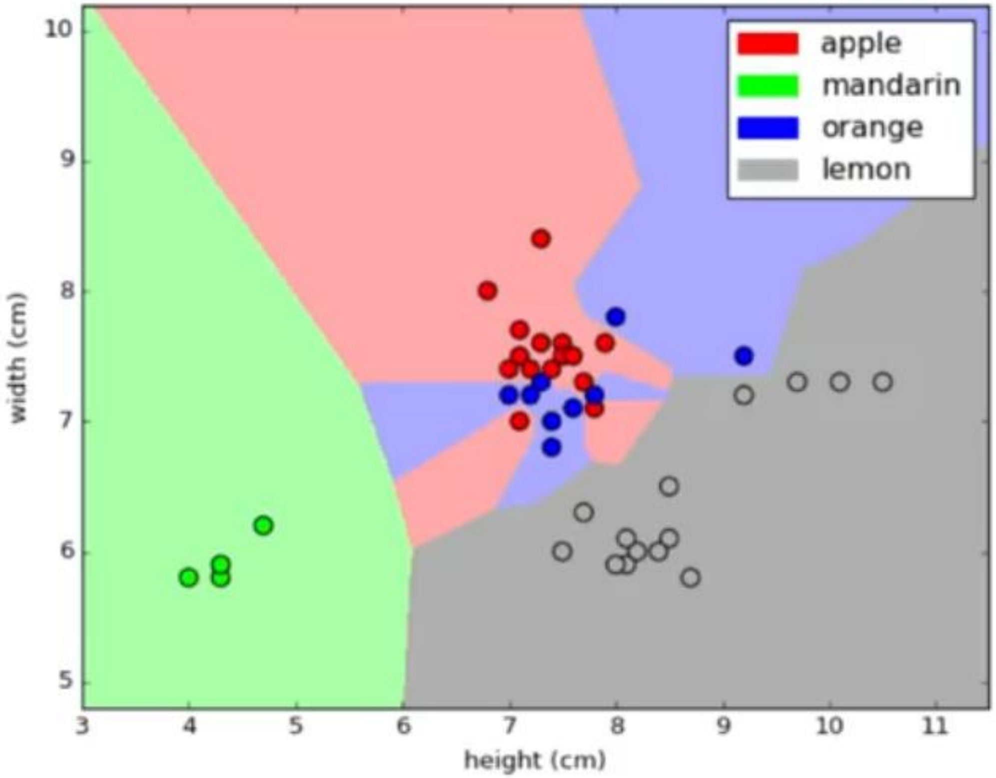
Choosing k in k-Nearest Neighbors

- The training error rate and the validation error rate are two parameters we need to assess on different K-value



Generalization, Overfitting and Underfitting

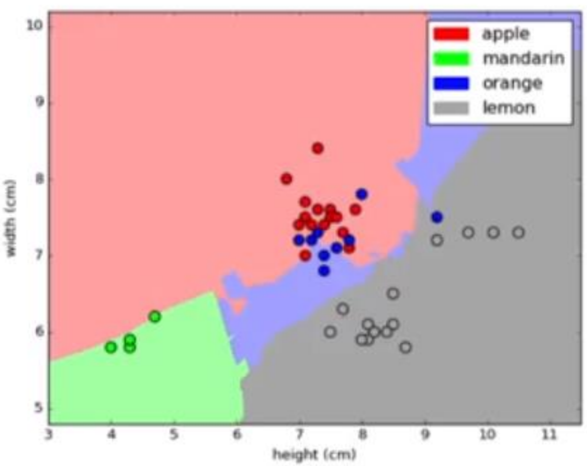
- Generalization ability refers to an algorithm's ability to give accurate predictions for new, previously unseen data.
- Assumptions:
  - Future unseen data (test set) will have the same properties as the current training set
  - This, models that are accurate on the training set are expected to be accurate on the test set
  - But that may not happen if the trained model is tuned too specifically to the training set.
- Models that are too complex for the amount of training data available are said to **overfit** and are not likely to generalize well to new data instances.
- Models that are too simple, that do not even do well on the training data, are said to **underfit** and also not likely to generalize well.



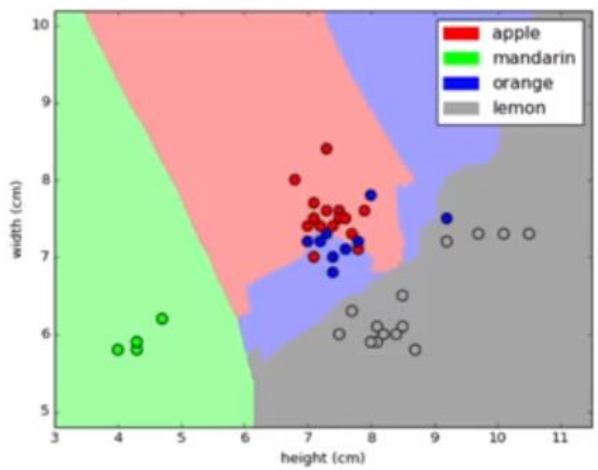
Fruit dataset
Decision boundaries
with k = 1

Example taken from Applied Machine Learning in Python course by the University of Michigan
https://www.coursera.org/learn/python-machine-learning

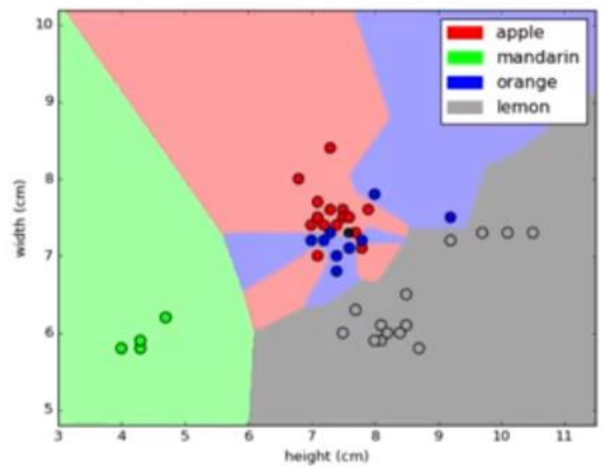
Overfitting with k-NN classifiers



K=10



K=5



K=1

Nearest Neighbor

When to Consider

- Instance map to points in R^n
- Less than 20 attributes per instance
- Lots of training data

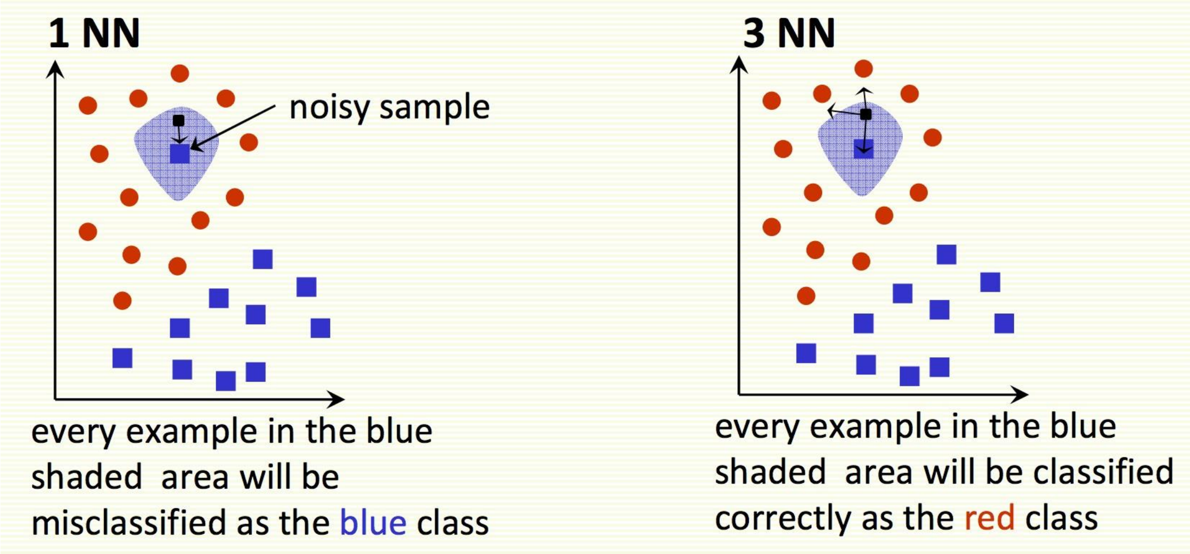
Advantages

- Training is very fast
- Learn complex target functions
- Do not lose information

Disadvantages

- Slow at query
- Easily fooled by irrelevant attributes

k-Nearest Neighbors is Sensitive



Nearest neighbors sensitive to mis-labeled data ("class noise"). Solution?

Smooth by having k nearest neighbors vote

k-Nearest Neighbors: Complexity

Expensive at test time: To find one nearest neighbor of a query point **x**, we must compute the distance to all N training examples. Complexity: $O(kdN)$ for kNN

- Use subset of dimensions
- Compute only an approximate distance (e.g., LSH)
- Remove redundant data (e.g., condensing)

[Slide credit: David Claus]

k-Nearest Neighbors: Complexity

Storage Requirements: Must store all training data

- Remove redundant data (e.g., condensing)
- Pre-sorting often increases the storage requirements

High Dimensional Data: “Curse of Dimensionality”

- Required amount of training data increases exponentially with dimension
- Computational cost also increases

Fun Example:
Where on Earth is this Photo From?

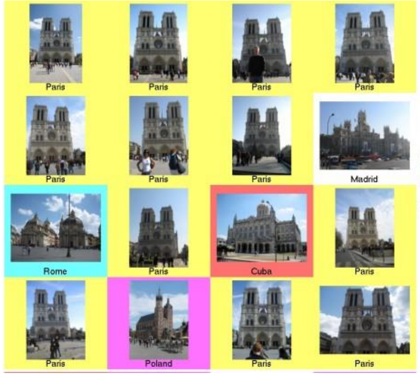
Problem: Where (e.g., which country or GPS location) was this picture taken?

[Paper: James Hays, Alexei A. Efros. im2gps: estimating geographic information from a single image. CVPR'08. Project page: http://graphics.cs.cmu.edu/projects/im2gps/]

Fun Example:
Where on Earth is this Photo From?

Problem: Where (e.g., which country or GPS location) was this picture taken?

- Get 6M images from Flickr with GPs info (dense sampling across world)
- Represent each image with meaningful features
- Do kNN!



[Paper: James Hays, Alexei A. Efros. im2gps: estimating geographic information from a single image. CVPR'08. Project page: http://graphics.cs.cmu.edu/projects/im2gps/]

Fun Example:
Where on Earth is this Photo From?

Problem: Where (eg, which country or GPS location) was this picture taken?

- Get 6M images from Flickr with gps info (dense sampling across world)
- Represent each image with meaningful features
- Do kNN (large *k* better, they use *k* = 120)!

[Paper: James Hays, Alexei A. Efros. im2gps: estimating geographic information from a single image. CVPR'08. Project page: http://graphics.cs.cmu.edu/projects/im2gps/]

Machine Learning Evaluation

- There are various metrics and methods to evaluate machine learning algorithms
- They differ according to the algorithm being supervised or unsupervised and they differ according to the task
- Let’s look at some of the metrics and concepts regarding evaluation

Accuracy

- This is the simplest metric
- Number of correct predictions divided by the total number of predictions, multiplied by 100.

$$\text{Accuracy} = \frac{\#\text{correct predictions}}{\#\text{total instances}}$$

Accuracy with Imbalanced Classes

- Suppose you have two classes:
  - The positive class
  - The negative class
- Out of 1000 randomly selected items, on average:
- One item belong to the positive class
- The rest of items (999 of them) belong to the negative class
- The Accuracy will be

$$\text{Accuracy} = \frac{\#\text{correct predictions}}{\#\text{total instances}}$$

Accuracy with Imbalanced Classes

- When you build a classifier to predict the items (positive or negative), you may find out that the accuracy on the test set is 99.9%.
- Be aware that this is not an actually presentation of how good your classifier is.
- For comparison, if we have a “dummy” classifier that do not consider the features at all but rather blindly predict according to the most frequent class

Accuracy with Imbalanced Classes

- If we use the same dataset mentioned in the previous slide (the 1000 data instance with 999 negative and 1 positive). What do you think the accuracy of the dummy classifier would be?

Answer:

$$\text{Accuracy}_{\text{Dummy}} = 999/1000 = 99.9\%$$

- Hence the accuracy alone sometime not a good metric to measure how good the model is

Dealing with Imbalanced Classes

- Data pre-processing
  - ➢ Random Under Sampling
  - ➢ Random Over Sampling
  - ➢ Cluster-Based Over Sampling
  - ➢ Synthetic Minority Over-sampling
- Select More suitable Metrics to Evaluate Imbalanced Classes
  - ➢ Precession and Recall
  - ➢ F1-Score

Precision and Recall

Precision

Precision attempts to answer the following question:

> What proportion of positive identifications was actually correct?

Precision is defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall

Recall attempts to answer the following question:

> What proportion of actual positives was identified correctly?

Mathematically, recall is defined as follows:

$$\text{Recall} = \frac{TP}{TP + FN}$$

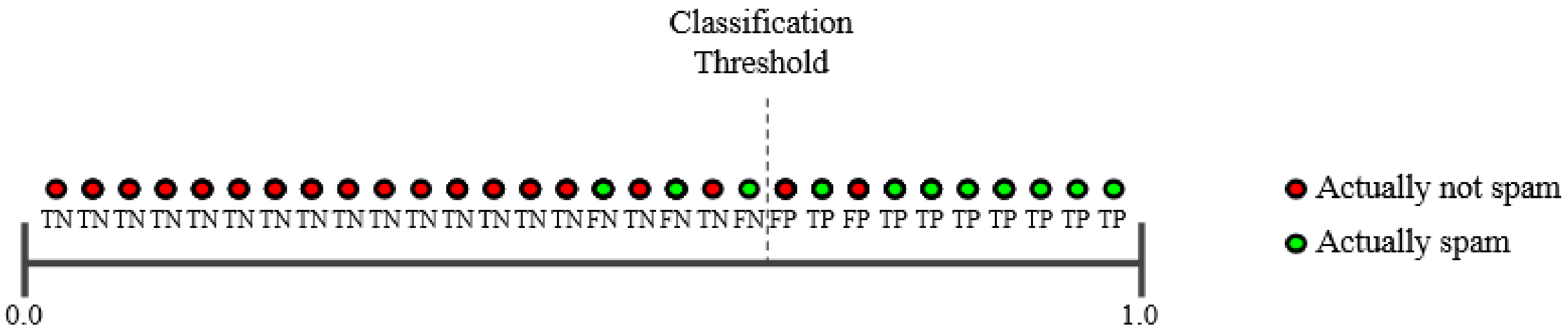
TP: True Positive

FP: False Positive

FN: False Negative

https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

Precision and Recall



True Positives (TP): 8	**False Positives (FP): 2**
False Negatives (FN): 3	**True Negatives (TN): 17**

Precision measures the percentage of **emails flagged as spam** that were correctly classified—that is, the percentage of dots to the right of the threshold line that are green

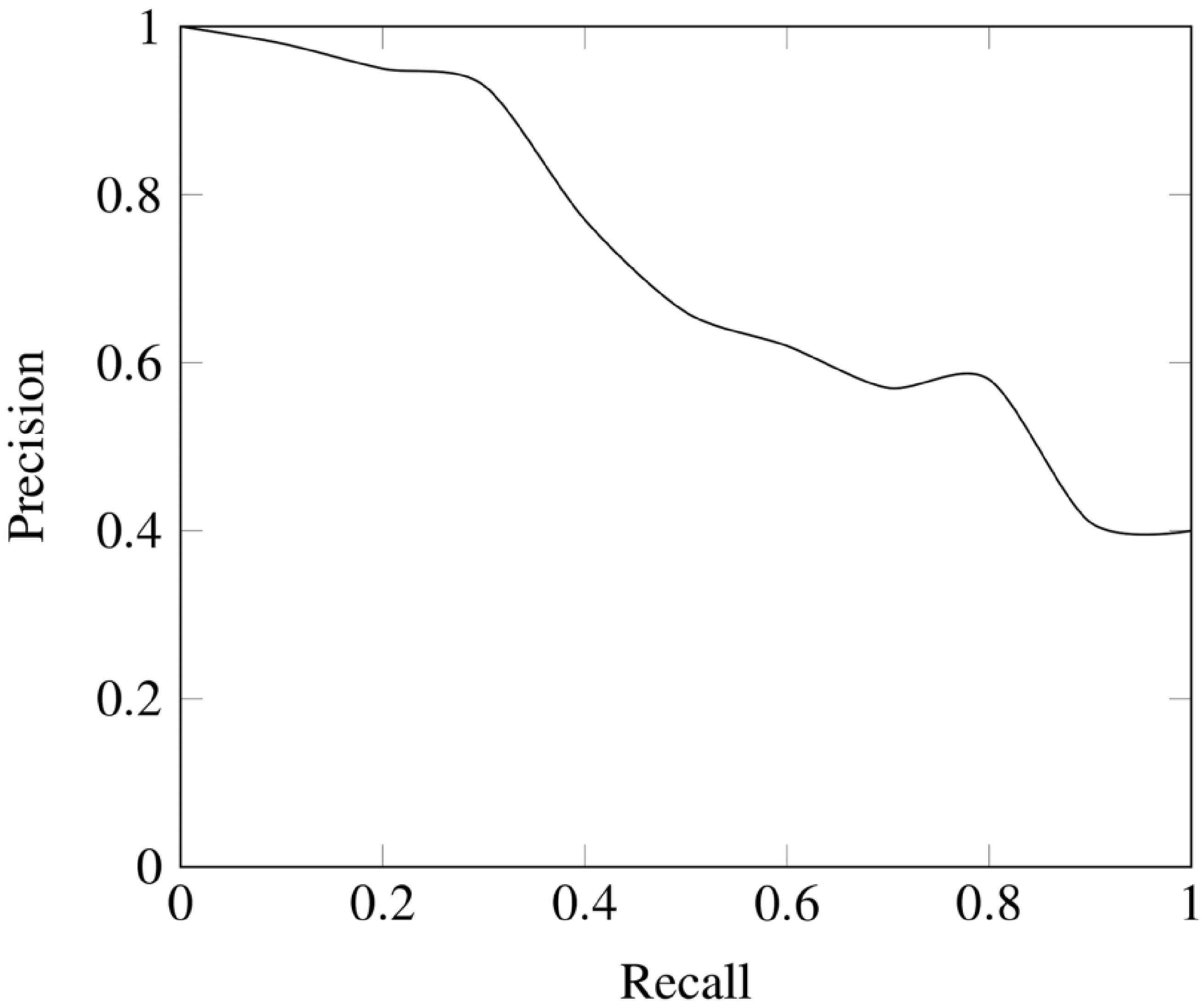
$$\text{Precision} = \frac{TP}{TP + FP} = \frac{8}{8+2} = 0.8$$

Recall measures the percentage of **actual spam emails** that were correctly classified—that is, the percentage of green dots that are to the right of the threshold line

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{8}{8+3} = 0.73$$

https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

Precision and Recall



https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c

Confusion Matrix

		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	TP	FP
	NEGATIVE	FN	TN

		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	560	60
	NEGATIVE	50	330

https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/#:~:text=A%20Confusion%20matrix%20is%20an,by%20the%20machine%20learning%20model.

F1 Score

- A metric which combines precision and recall
- Harmonic mean of precision and recall
- Instead of balancing precision and recall, we can just aim for a good F1-score and that would be indicative of a good Precision and a good Recall value as well

$$F1\,Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Useful Resources

- Mathematician hacking dating site https://www.wired.com/2014/01/how-to-hack-okcupid/
- https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7
- https://towardsdatascience.com/building-improving-a-k-nearest-neighbors-algorithm-in-python-3b6b5320d2f8
- https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/
- https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/
- https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f

Questions?