My First Classifier

by Dane Brown

The simplest solution is sometimes the best solution

cvML Methodology

- **Initialization:** Call the `cv` or `scikit` model by name to create an empty instance of the model
- **Set parameters:** can be default, e.g. k -NN: specify k for more than one neighbour
- **Train the model:** `train` or `fit` is used to fit the model to some data.
- **Predict new labels:** use `predict`, to guess the labels of new (**unseen**) data.
- **Score the model:** refer to slide 10 & 13: works for both `cv` and `scikit`

Unlike the Previous Program

2_k-NN.py

- will take in (generated) training data points
- predict the label of the test (new) data points using k-NN

- The previous program simply showed how labels work (by simulating a classifier)

k -NN in a Nutshell

The class of a new (unseen) data point is **predicted** by finding the closest data point to it in the training dataset, i.e. its nearest neighbour

Assumption: the new data point probably belongs to the same class as its neighbour.

Multiple neighbours: In this case k-nearest neighbours are considered instead of a single one.

How to k -NN (in OpenCV)

- **Acquire** some **training data** (generate mock data)
- **Acquire** some **test data** (generate mock data)
- **Create** a k**-NN object**
- **Specify** an odd number for k (optional)
- **Find** the **nearest neighbors** of a test data point, to be classified
- **Assign** the **class label** of the test data point by **majority vote**
- **Plot** the **result**

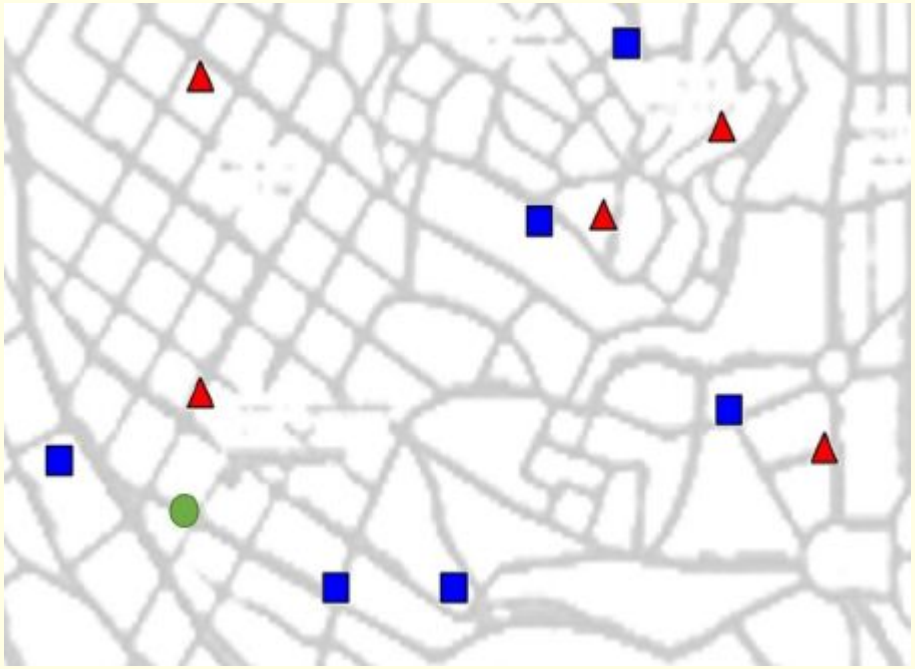
Why an odd number for k?

It is advisable to only assign odd values for **binary** classification to avoid ties

Non-binary classification is a different story

Predict a Rugby Fan

- Fans of two teams, Reds and Blues live here
- Fans always live closer to their fellow fans
- Add a new point somewhere
  - Predict whether the new point is a Reds or Blues fan



`Check it out -> CV_ML`