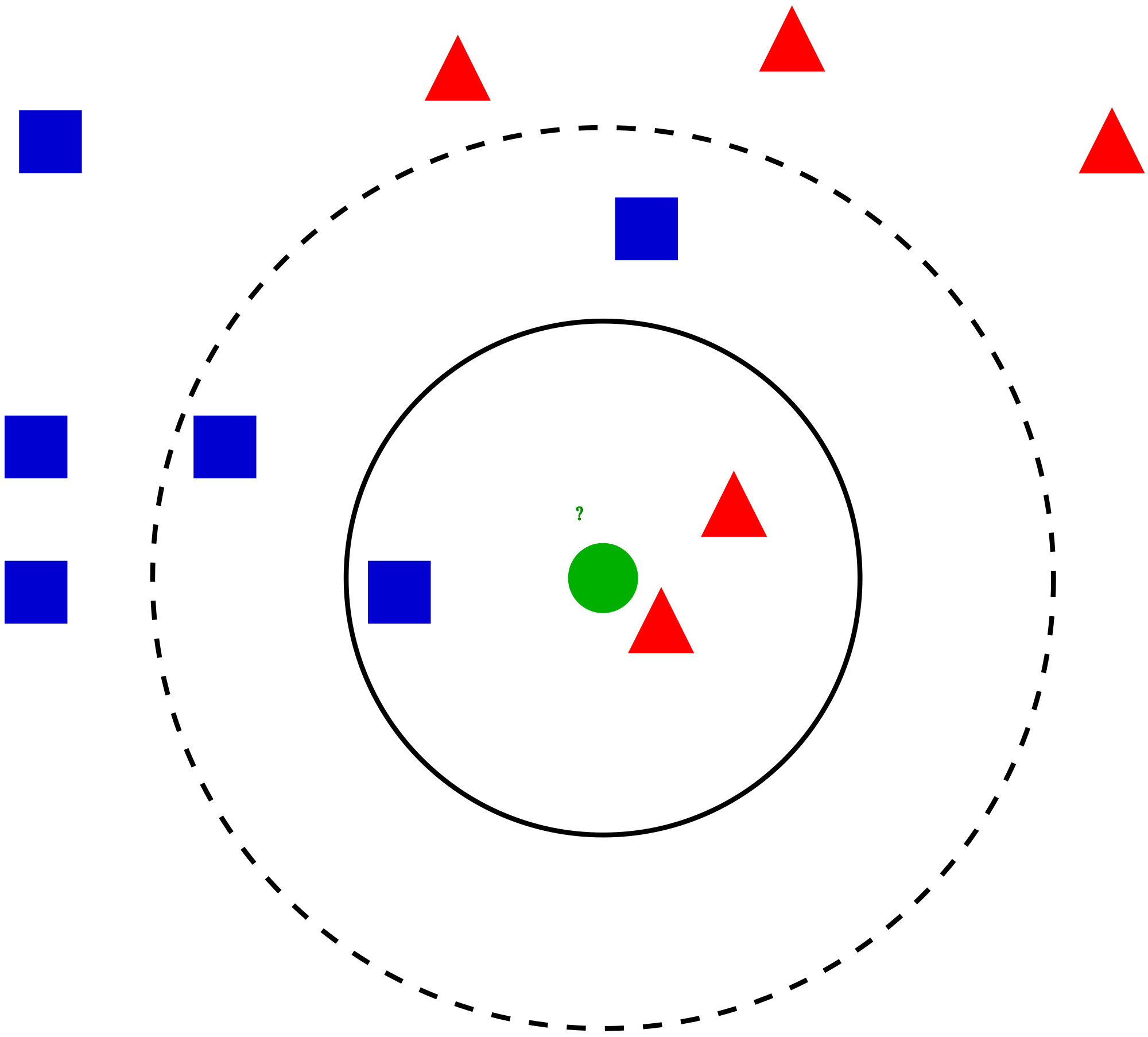
K- Nearest Neighbor

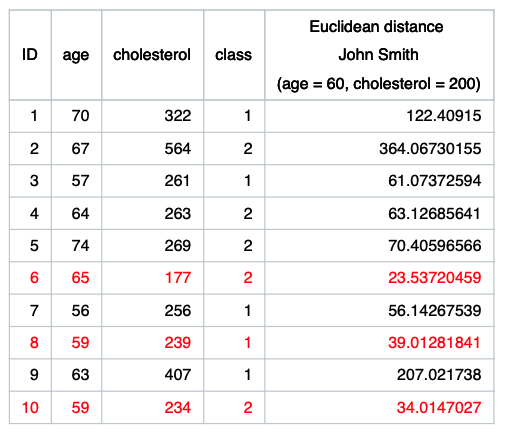
KNN is a model that classifies data points based on the points that are most similar which is depends on distance (euclidian, manhattan etc).



How does it work?

First of all we need to choose a K parameter that is the number of neighbors. Let’s say my house's green dot and I would like to visit three closest neighbors would be two red triangles and one blue square in the solid circle. Then I would like to classify my three closest neighbors based on the majority vote. There are two triangles and one square then I would classify them as red. If I choose a number of neighbors 5 then I would classify my neighbor as a square.

Let’s find John smith class {age=60, cholesterol = 200) with K=3



If we look at above example the euclidean distance between ID 1 and John smith is

sqrt((70-60)^2 + (322-200)^2)^0.5 = 122.409

The smallest three distances are highlighted with red and majority says John belongs class 2.

**Pross**

Easy to use

Quick calculation time

Does not make assumptions about the data

**Cons**

Accuracy depends on the quality of the data

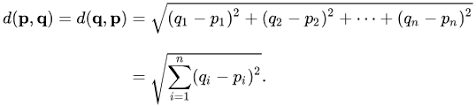
Must find an optimal k value

Poor at classifying data points in boundaries where they can be classified one way or another.

KNN is an algorithm that is considered both non-parametric and an example of lazy learning.

Non-parametric means that it makes no assumptions. The model is made up entirely from the data given to it rather than assuming its structure is normal.

Lazy leaning means that the algorithm makes no generalizations. This means that there is little training involved when using this method. Because of this, all of the training data is also used in testing when using KNN.



Python Sklearn Implementation

Import libraries

Load data

Separate futures and target

(Split data into train and test for some cases to find accuracy, Also we may need to normalize input for some ML algorithm)

Define the model

Fit the Model

Predict