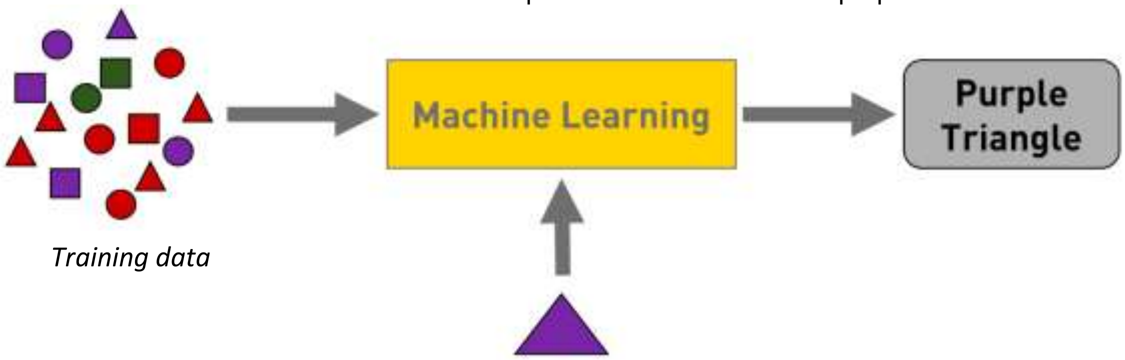
Software Development IV – Advanced .NET 420-411-DW
Lab Exercise 1 – K-Nearest Neighbors

Review your C# knowledge - work with files, classes and methods

In this exercise, we are introduced to machine learning data sets and a first algorithm in machine learning.

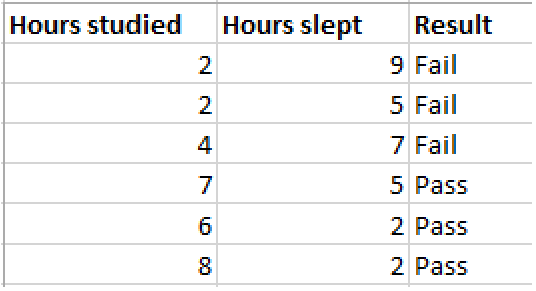
Machine learning is a subset of AI related to using existing data and finding correlations and relationships within the data to predict new outcomes or classify new data. Supervised learning start with a set of labeled data (called training data) which is used to make a mathematical or statistical model. The model is used for either prediction or classification purposes.



In this exercise, we are using a famous data set in AI, Fisher's Iris flower dataset used for classifying irises. Fisher was a statistician and biologist who measured features on three related species of irises. Fun fact - there is a Quebec connection: 2 of the 3 species were measured from a pasture in the Gaspésie!

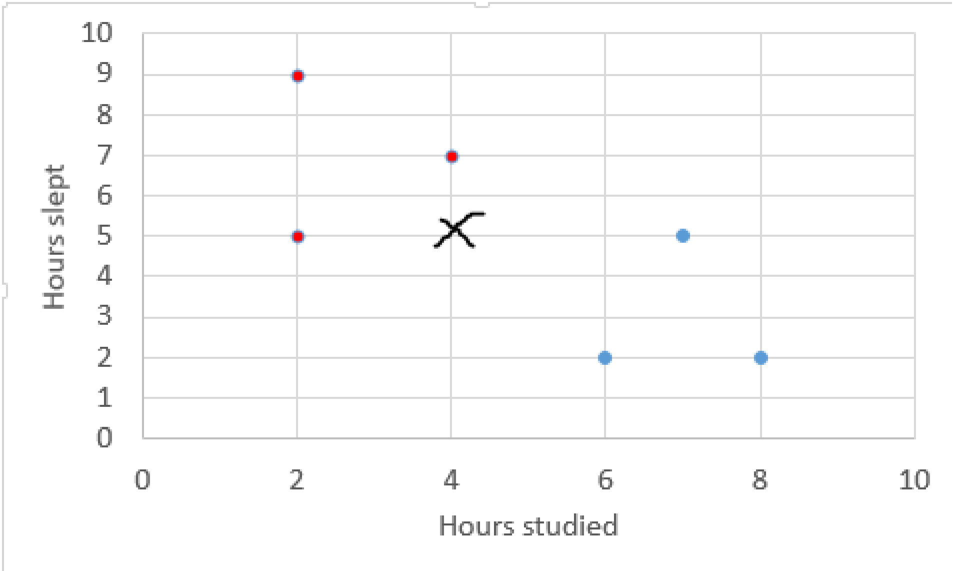
Background: Understanding the classification problem

Consider an example where we ask students prior to taking an exam how many hours they slept during the previous 12 hours, and how many hours they studied during that same time. We then get pass/fail results from the exam. The results might look something like this *sniff*:



Hours studied	Hours slept	Result
2	9	Fail
2	5	Fail
4	7	Fail
7	5	Pass
6	2	Pass
8	2	Pass

It is often easier to visualize the data through a chart. In this case, the colour legend indicates if the student passed (blue) or failed (red).



At the point $\boldsymbol{X}$: would you predict that this student passed or failed the exam?

Your instincts probably tell you that the poor student most likely failed - because you noticed *clusters*.

K-Nearest Neighbours algorithm

This algorithm basically checks which training data points are close to the new point $\boldsymbol{X}$, and predicts its classification based on the neighbours. $\boldsymbol{k}$ indicates the number of nearest neighbours who get a vote. Let's say we say that $\boldsymbol{k} = \mathbf{3}$ for the dataset above. Which are the three closest neighbours to $\boldsymbol{X}$?

The easiest distance measure is *Euclidean* distance. In our example, we have two variables, or dimensions, or *features*. Recall, in two dimensions, the distance between point $p\ (p_1\ , p_2)$ and point $q\ (q_1\ , q_2)$ is:

$$dist(p\ , q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

Since $\boldsymbol{k} = \mathbf{3}$, we use Euclidean distance to find the 3 closest example students to the student $\boldsymbol{X}$ at $(4, 5)$(4 hours studied, 5 hours slept). We see the three closest points are $(4, 7)$ (distance of 2), $(2, 5)$ (distance of 2), and $(7, 5)$ (distance of 3). Two out of 3 neighbours fails, thus we would predict that this students fails also.

Generalizing the algorithm

When you have *n* dimensions, then the Euclidean distance between 2 points is calculated as:

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

The alggorithm:

1. First calculate the distance beween the point you want to classify and all the existing training points
2. Then find the k training points that are closest
3. Get the categories (labels) for the k closest points

4. Count the frequencies of the k labels. The highest frequency is the predicted label

Lab exercise

We will be looking at the Iris flower dataset and using it to classify a new flower. This is one of the first datasets that is typically used in machine learning exercises.

The flower dataset contains measurements of 3 different but related species of irises. For each species, there are 50 flowers which were measured; and each flower had 4 attributes measured:

- sepal length
- sepal width
- petal length
- petal width

You have been provided iris.csv.

Steps

1. Create a new Console App (.NET Framework) application. Call it KNearestNeighbours
2. Notice that we have provided you with 2 classes, the FourDPoint and KNN classes. Select Project – Add Existing Item and add these two files.
3. Fill in all the TODO places to complete the application, starting with FourDPoint (a 4-dimensional Point is represented with an array with 4 elements and a string label). Don't forget to delete the NotImplementedException when you are done.
4. In your Program class, instantiate a KNN object, passing it the string "iris.csv" and k equal to 3. Predict the label for an iris with the following features: { 2.2, 3.6, 5.1, 2.5}
5. Make your application generic, so that it works with any number of features (2 or more), instead of only four. Change the name to the class FourDPoint using the refactoring tool.
6. Call your instructor over when you are done or are confused/stuck.

Note: The better way to deal with files: In Visual Studio, add existing item (be sure to choose all files) and select the iris.csv file. In the Properties panel, set Build Action to Content and Copy to output directory to Copy if Newer. This will copy the files to your bin directory automatically and you can see the file in Visual Studio. The other big benefit is that version control will see the file.

Note: if you run in Debug mode, the console will disappears as soon as the program terminates. To keep on the screen, either run without debugging (Ctrl + F5) or add a Console.ReadKey() to the end of your Main method or don't write to console, use System.Diagnotics.Debug.Writeline(...) and write to the Debug tab in the output window.

Note: You may have to change the project properties to .NET Framework 4.7 to access C# 7.0 features (Solution Explorer – right click project – Properties – Application – Target Framework