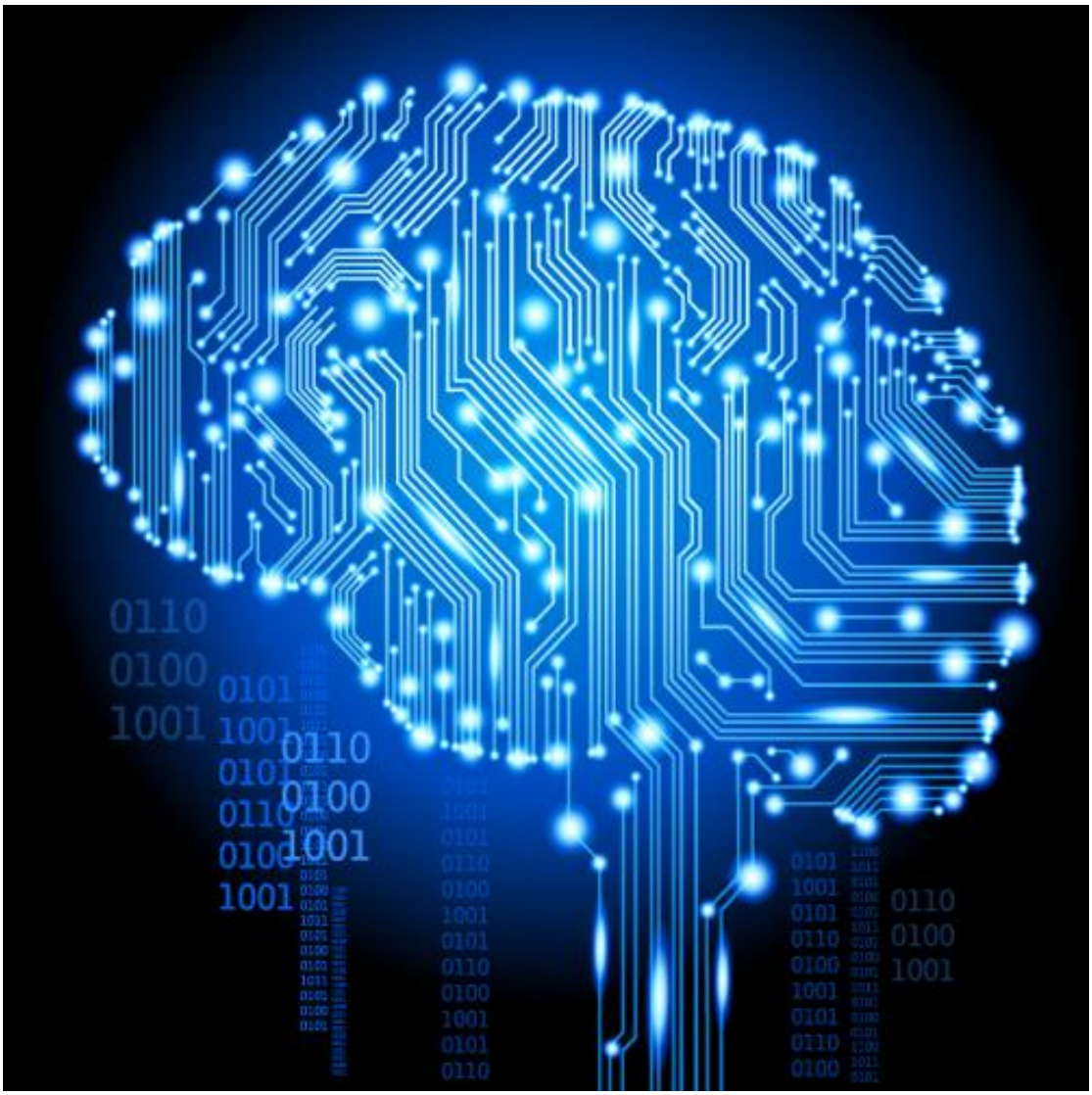
COMP3009 Machine Learning

Lab 2: Introductory Exercises

Computer Based Coursework Manual – Autumn 2021



1. Introduction

This session is done individually and is not assessed.

Now that you have been introduced to working in MATLAB, you can try your hands at the following exercises. Please note that these exercises will not be evaluated, and it is not expected that you complete each and every part of this exercise. Instead, pick one that takes your fancy. Attempting these exercises will help you to get a good grasp of working in MATLAB and some of the basic concepts of machine learning.

2. Classification using K-Nearest Neighbours

K-Nearest Neighbours is a classification algorithm which assigns a label to a feature *x* according to a majority vote taken over the labels of the K training instances with smallest distance to *x* (see Fig below).

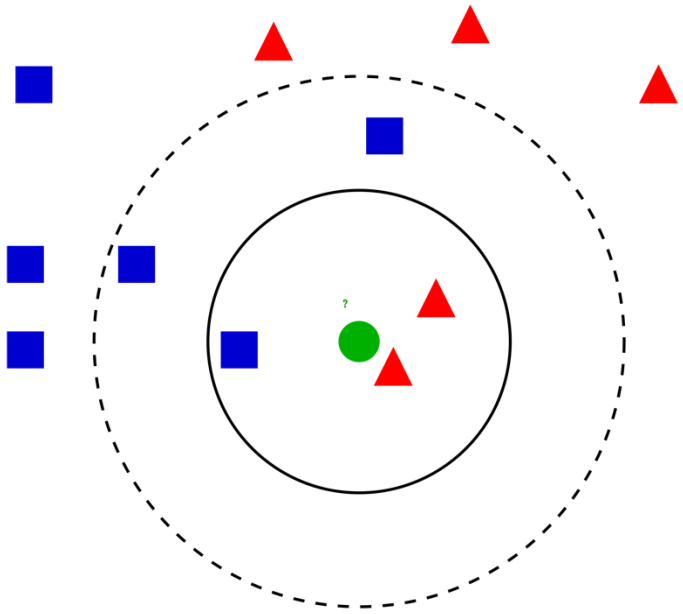


Fig. 6 k-Nearest Neighbours. If k = 3 (solid circle), then the new green data point will be assigned the label 'red triangle', as that's the majority label. If k = 5 on the other hand (dashed circle), it will be assigned the label 'blue square'.

Task: Write a function *label = knn(training_instances, training_labels, test_instance)* in MATLAB which takes as input the training instances, training labels and a test instance. The output should be a label for the test instance according to k-nearest neighbour algorithm. You can use the Euclidean distance as a distance measure for finding the nearest neighbours.

Use the first 1,000 or so data points from the forest cover type dataset in the file forest_covers.mat (available on Moodle), included in the "Lab Toy Data" zip file, to test your function. Note that for efficiency reasons, forest_covers is stored as a sparse matrix. You can use it as you would a full matrix, but if you want to turn it into a 'normal' dataset, you could do:

```
load("forest_covers.mat");
x_train = full(training_vectors(1:1000, :));
y_train = training_labels(1:1000);
```

Additional Exercises:

1. Think of a reasonable performance measure.
2. What is the dimensionality of the forest_covers feature vectors?
3. What data should you test on to measure the generalisation error?
4. Try different values of k and measure their performance.

3. Clustering using K-Means

K-Means is a clustering algorithm that is used for finding K clusters in a given set of data points where each data point belongs to a cluster with the nearest mean. It basically consists of an iterative procedure consisting of the following steps:

1. Initialise cluster centres $\boldsymbol{\mu}_k$ randomly.
2. Assign each point to one of the *k* clusters based on their distance to the current centres $\boldsymbol{\mu}_k$.
3. Update the cluster centre locations by calculating the new mean of the points assigned to each cluster.
4. Repeat steps 2 and 3 until convergence.

You can look up k-means on line for more detail.

Task: Write a function *cluster_labels = k_means(data_points)*, which takes as input a set of data points and outputs a label for each point according to the cluster to which it belongs. Use the dataset from the file OldFaithful.mat to test your function.

Additional Exercises:

1. Write a function to visualise your end result
2. Write a function to visualise the intermediate results after each iteration of the algorithm
3. Try cluster centres in the range 2:5, and visualise the end results for each
4. Try different runs of the algorithm with the same number of clusters. Are the results always the same? Explain why.

4. Linear regression using a first order polynomial model

Regression is used for modelling the relationship between a response variable and one or more predictor variables. In linear regression, the response variable is modelled as a linear combination of the model parameters and the predictor variables.

Task: Write a function model_parameters = learnRegressionModel (predictions, responses), where predictions is a matrix containing the observed values for predictor variables in each row and responses is a column vector containing the corresponding values for response variables. The function should return a vector containing the model parameters. Learning the model parameters requires the minimization of the least squares objective function. Write the above function first by

1. Using brute force approach, and then
2. Using gradient descent method.

Use the data from the file texas_temp.mat to test your function by learning a regression model to predict the temperature (response variable) from the predictor variables (latitude, elevation and longitude).