## Theory

1. Concept learning

A concept can be viewed as a mental category to be used for classification;

these classifications can be viewed as a binary dicision of either inside or

outside the concept category. Concept learning is learning a stratergy for

classifying these concepts, usually based on a training set.

An example can be some domain containing a certian set of attributes and an

output result, e.g: Answering if it will rain on a day based on the days

before humidity, rain, temprature, weekday and month (Some of these are

obviously more relevant than others)

1. Function approximation

Function approximation is to find the best fitting function to a certain data

set, often in machine learning viewed as a search for the best fitting

function through a space of hypothesis functions. This is essential for

machine learning as it enables us to find the best hypothesis based on the

training data.

1. Inductive bias

Inductive bias is the nature of assuming an output given an input, based on

training data where the given input does not occur. This is important in

machine learning as it enables us to make a conclusion of "unknown" input.

The inductive bias for Candidate elimination is such that the hypothesis space contains the target concept, whilst for decision trees is a preference for short trees over long trees.

1. Over and underfitting

Overfitting is modeling the training data too well, thus also modelling excessive details and noise.

Underfitting on the other hand is modeling the training data too simply, thus missing out essential tendencies within the data. Both models will perform poorly.

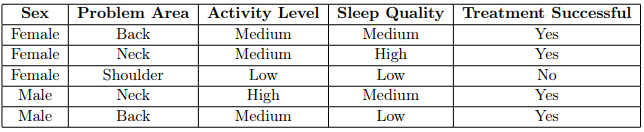
Validation Set is a set of data to estimate how well the model has been trained and also to estimate model properties.

Cross-validation runs the model multiple times using subsets of the prepared data as trainingsets and validation sets. This enforces the model to not be specialized on the given training data, in turn making it more generalized, thus mitigating the damage done when overfitting. Since only doing this once may become a problem as the split itself could be an issue multiple iterations of splitting and training/testing is used in cross-validation runs.

1. Candidate model

Version space is the validated hypothesis, a subset of all the hypothesis, that is consistent with the training data. The resulting version space, specific and general boundary is computed below.

We see here that all of them resulted in the general boundary, as there are no consistent attributes troughout the training data regarding the successful treatment.



S0 = {Ø,Ø,Ø,Ø}

G0 = {?,?,?,?}

Case 1 = Female, Back, Medium, Medium → Yes (Specific is made more general)

S1 = {Female, Back, Medium, Medium}

G1 = { ?, ?, ?, ?}

Case 2 = Female, Neck, Medium, High →Yes (Specific is made more general)

S2 = {Female, ?, Medium, ?}

G2 = {?, ?, ?, ?, ?}

Case3 = Female, Shoulder, Low, Low → No (General is made more specific)

S3 = {Female, ?, Medium,?}

G3 = {?,?,Medium,?}

Case4 = Male, Neck, High, Medium → Yes (Specific boundary becomes too general to hold any value)

S4 = {?,?,?,?}

G4 = {?,?,?,?}

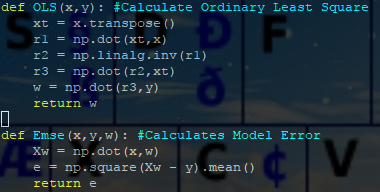
Case5 = Male, Back, Medium, Low → Yes (No change as we at the most general boundary)

S5 = {?,?,?,?}

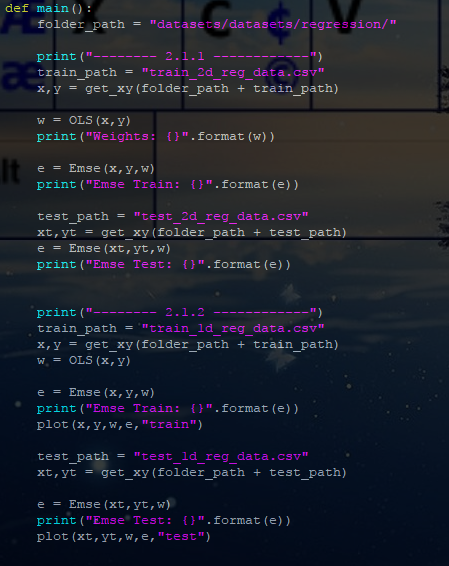
G5 = {?,?,?,?}

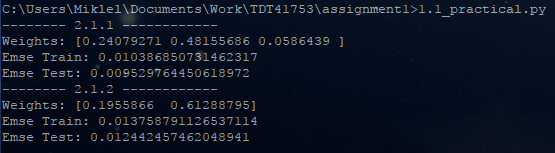
Result = {?,?,?,?}

## Task 2.1

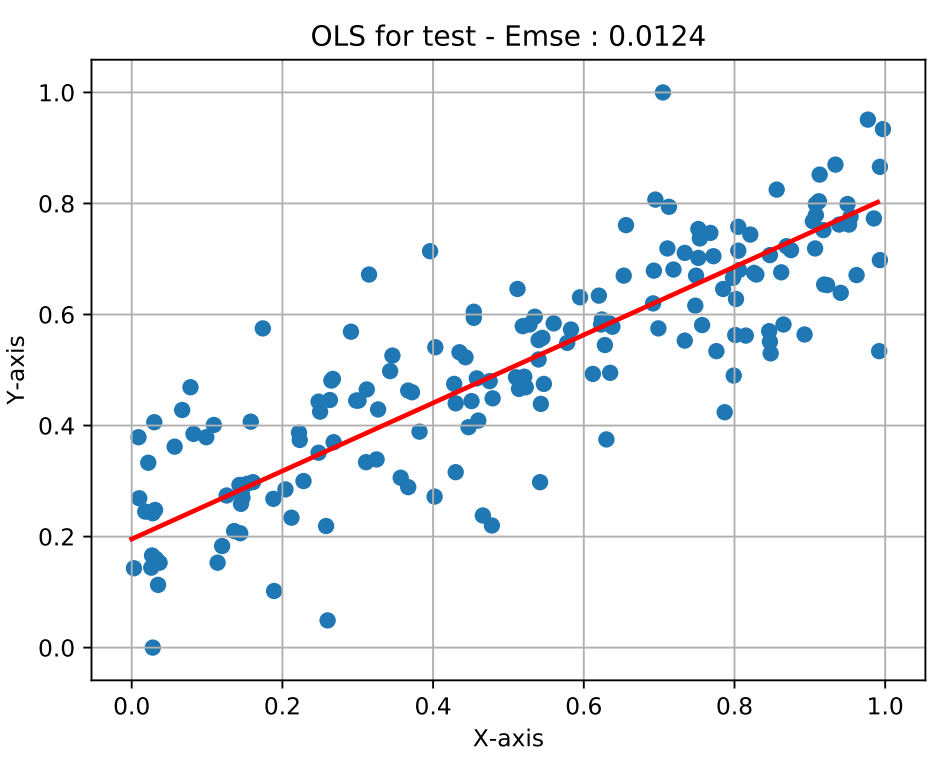
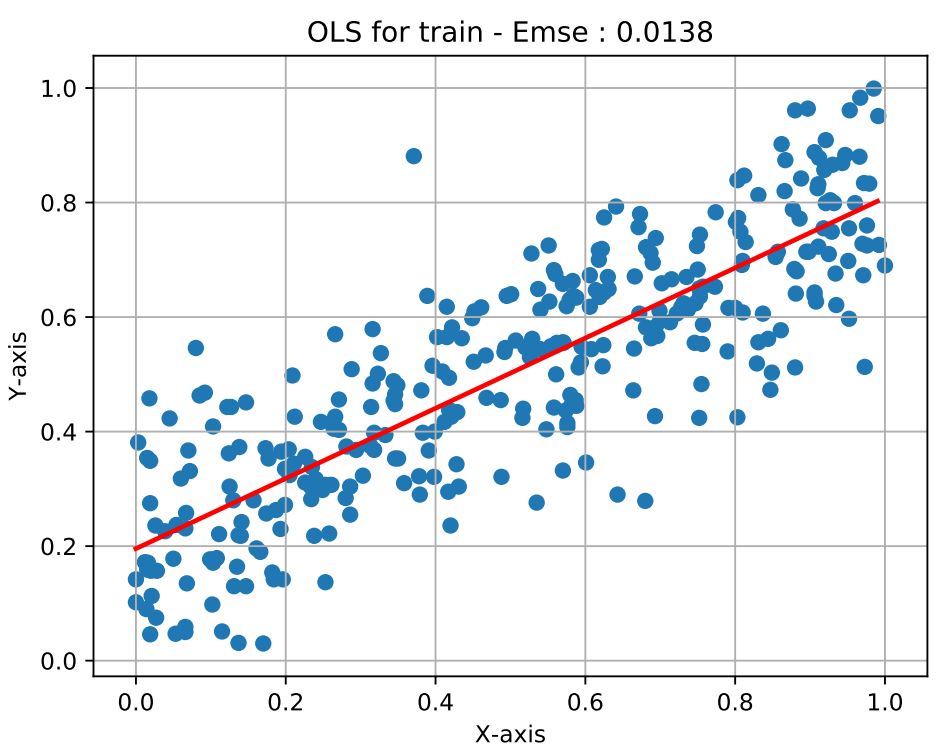


Main Function:



Corresponding output with plots from 2.1.2 on next page: 

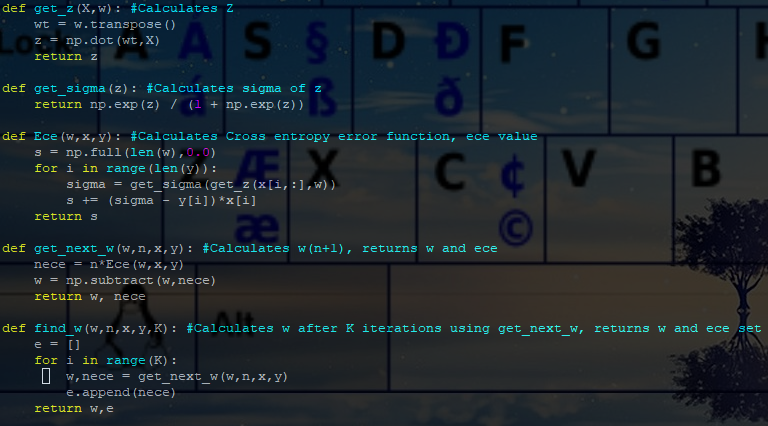
I believe the model is well generalized as we see the model error is low for both train and test data sets (actually lower in test)



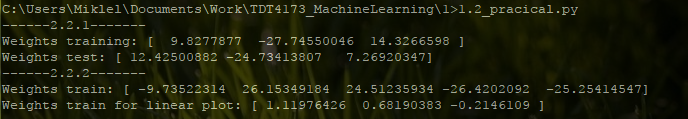
## Task 2.2

I chose learning rate of 0.1 and all starting weights of 1.0, as this was proven a well balance between generalized and specialized attributes.

The main functionality performing the calculations is shown in the following screenshot:



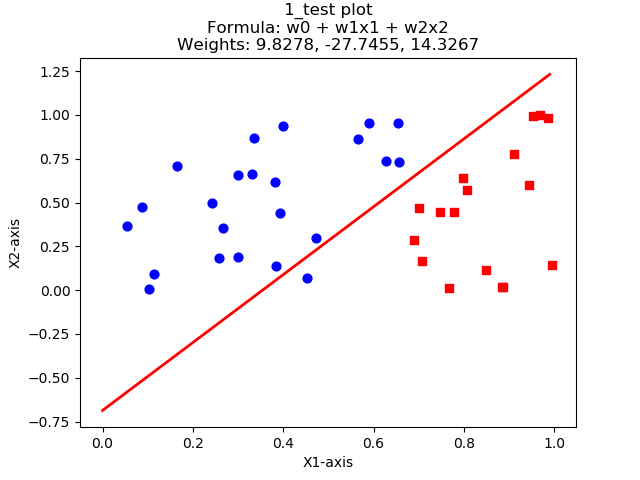
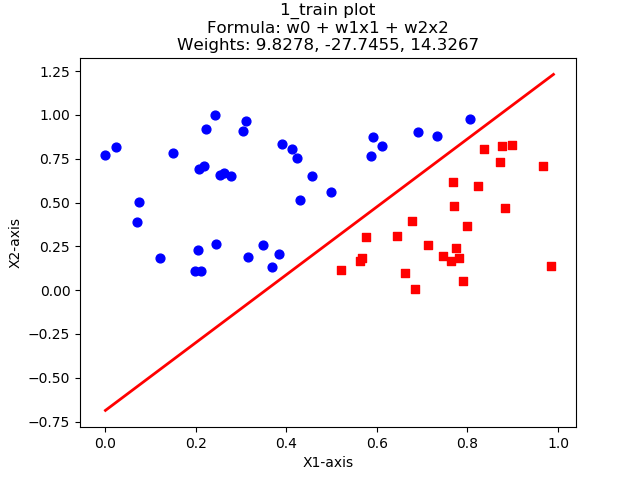
The main function running is shown below, which produced the following output in addition to the plots.





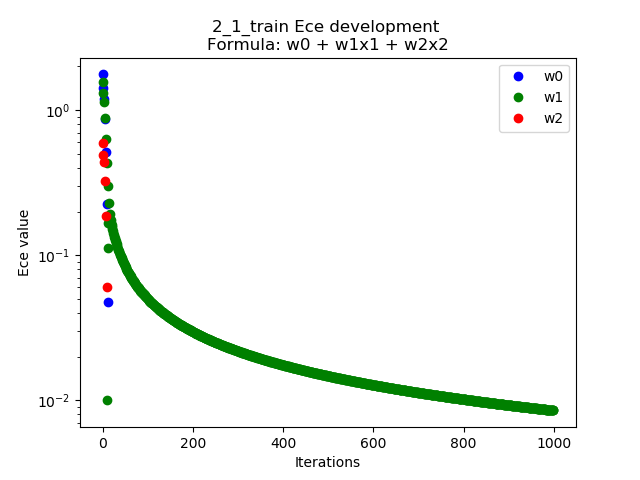
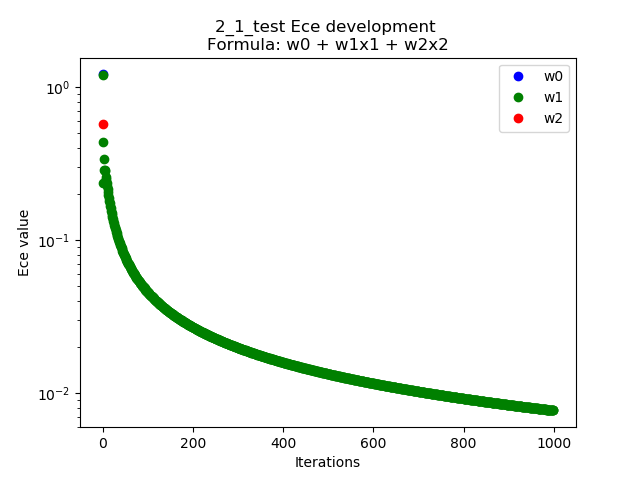
### 2.2.1

In assignment 2.1 the program produced the following plots with all starting weights equal 1.0 and learning rate 0.1, when training on Train and testing on Test.



This data is linearly separable, as we can clearly see that a single decision surface can separate the classes, here marked red and blue. The model is well generalized as it almost perfectly separates the test data as well.

In addition the assignment wanted us to train on both Train and Test in order to plot the error development. This was clarified on the discussion forum, as shown on the bottom of the report. These data produced the following plots:



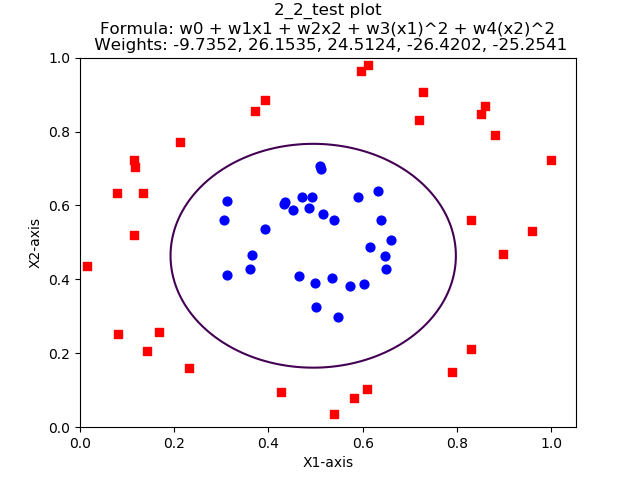
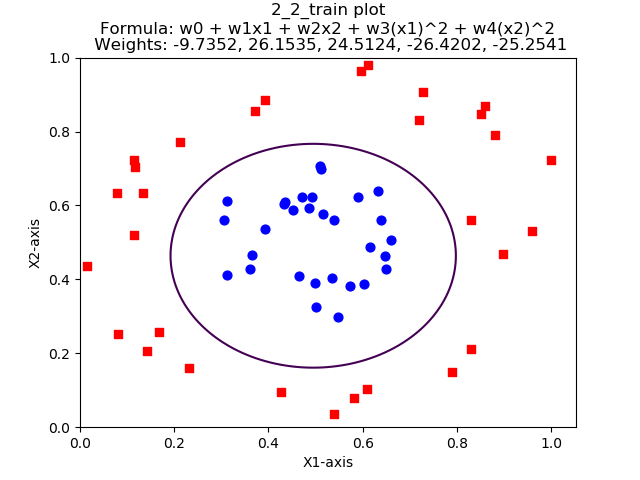
### 2.2.2

The dataset of 2.2 was shown to require a circular non-linear boundary, instead of a linear boundary as our model currently was.

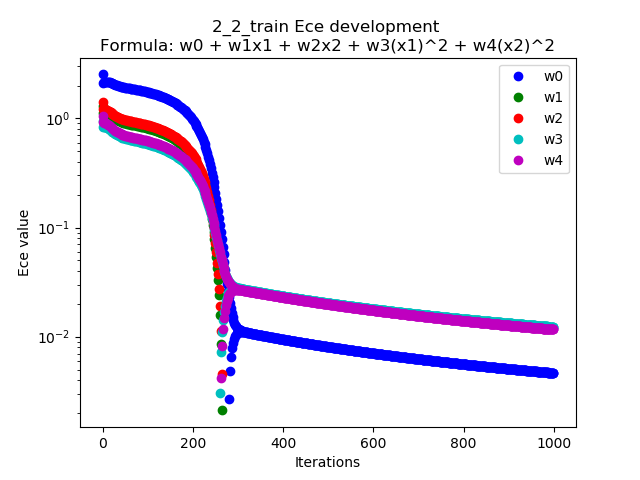
I therefor rewrote the data from the form x1, x2 to 1.0, x1, x2, (x1)^2 , (x2)^2.

This allowed for a model on the form w0 + w1x1 + w2x2 + w3(x1)^2 + w4(x2)^2 ≥ 0

This resuled in the following plots when training on Train and Testing on Test:



Here we once again see a well generalized model, as it perfectly separates the Test data. The error development when training is shown below:



The plot of the linear model on this circular data is shown below, showing a clear offset. I did not tune the data to match it to the best of my ability, as it was clear it would not be a correct boundary

