## 1 Linear Regression

### 1.1 Derivation of regularized Linear Regression

#### 1.1.1 Preliminary questions

1. The design matrix X is derived from the Error function of polynomial functions for writing it as a product of a vector and a matrix. In this error function we need to find a partial derivate for , for getting equations. The fact that we introduce an x0 (for convention = 1) involves that the matrix (and tor will have a size of m x (n +1) and in the case of the vector of 1 x (n +1).
2. If we consider the cost function , the gradient represents a vector containing the partial derivative of the cost function with respect to every component of the vector and it can be calculated as follows.

Let

Then

The result is an columns vector.

If we want to give a meaning, the gradient denotes the direction of greatest change in the cost function.[[1]](#footnote-1) To minimize the cost function (e.g. find the best parameters ), the gradient must be set equal to zero.

1. The Jacobian matrix is a generalization of the gradient. In general we talk about gradient when we are working on functions that maps , so the Jacobian matrix in this case would just be a vector. We talk about Jacobian matrix when working on functions mapping .[[2]](#footnote-2)

If we have a vector function that maps , where

The Jacobian matrix is defined as

1. The Jacobian matrix is a rows, columns matrix of the form

The first row of the matrix contains the partial derivative of the first element of the hypothesis vector w.r.t. each of the parameters contained in the vector .

#### 1.1.2 Regularized linear regression optimal parameters

The cost function for the regularized linear regression is defined as follows.

We want to show that

Where represents the best fitting parameters to minimize the cost function.

We start by taking the partial derivative of with respect to .

Here we used the fact that .

Now we want to set and isolate .

If we use the fact that , we get

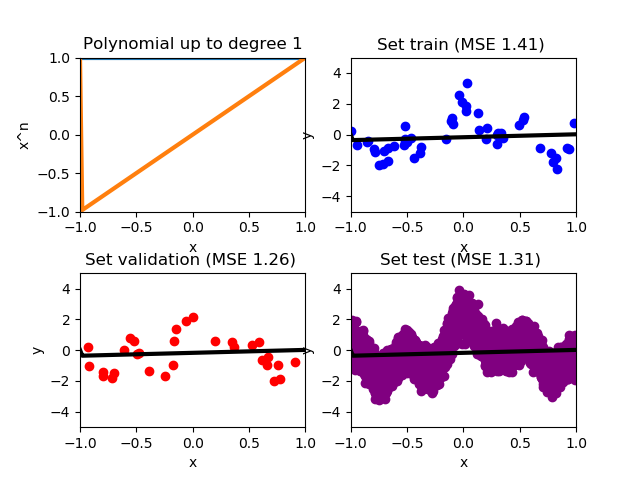
## 1.2 Linear Regression with polynomial features

### 1) Plot\_poly with different degrees

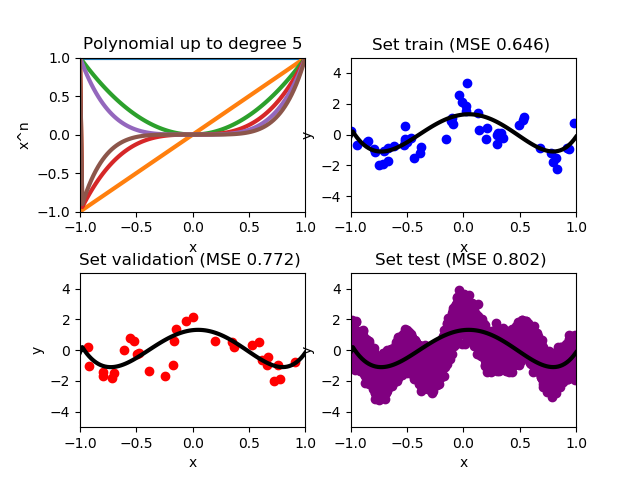
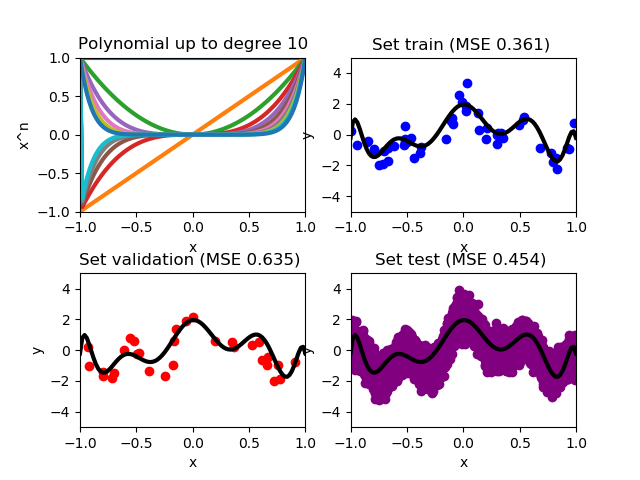
This table shows the mean square error values for each data set and for a function degree of 1, 5, 10 and 22. As you can see except for the error for testing set with degree = 22 the biggest errors are found in the graphs with function degree 1. This is because it is difficult to approximate a data set to a line (function of degree = 1).

What is interesting to note is that with higher degree functions the error is reduced but up to a certain point above which a higher degree negatively influences the value of the error (the smaller error for each data set is highlighted in green). Only for the training set the best degree corresponds to the largest possible.

|  |  |  |  |
| --- | --- | --- | --- |
| **Degrees / Data error** | **Training set MSE** | **Validation set MSE** | **Testing set MSE** |
| **1** | 1.4099 | 1.2592 | 1.3098 |
| **5** | 0.6457 | 0.7717 | 0.8021 |
| **10** | 0.3612 | 0.6348 | 0.4537 |
| **22** | 0.3390 | 0.9190 | 5.2024 |

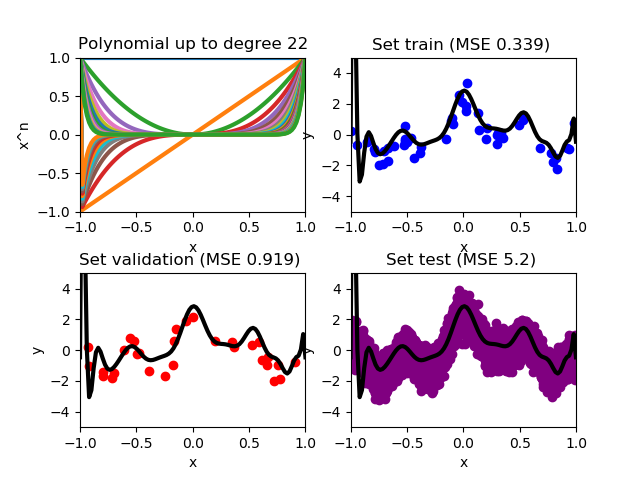
Here below are the graphs representing the data in the table.

Function of degree = 1



Function with degree = 10

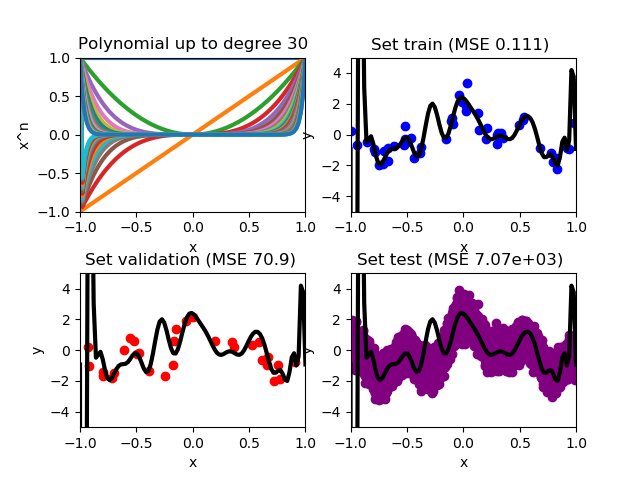
Function with degree = 5



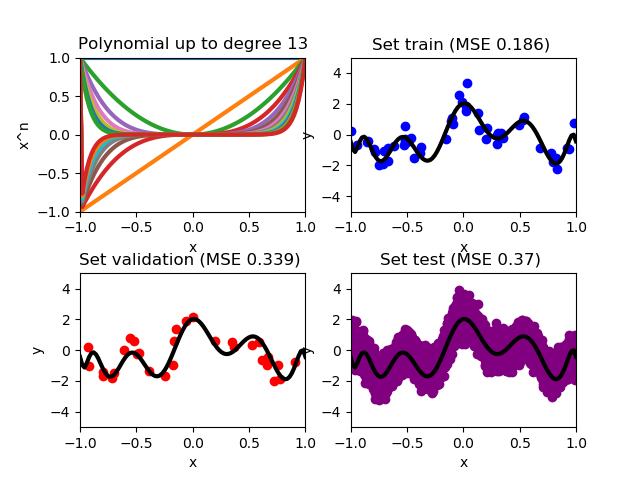
Function with degree = 22

### 2) Best degree for training set

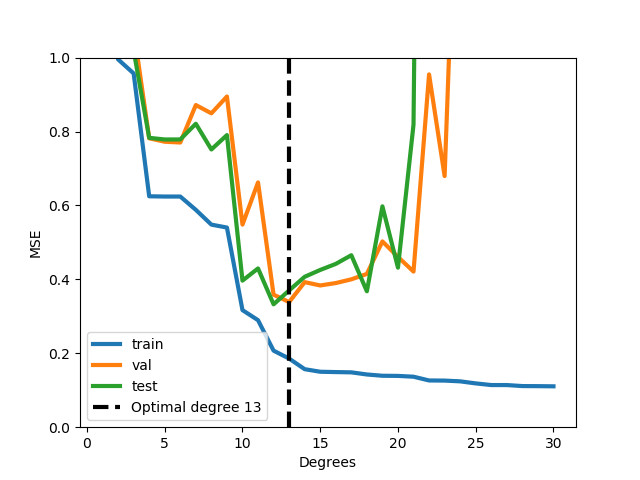
As we can see the ideal polynomial to describe this dataset will be of maximum grade, this is because, the algorithm will continue to "learn" from the dataset and with higher degrees polynomials it will have a greater possibility to describe the dataset more accurately. As we can see the MSE of the validation set by using the polynomial founded with the training set is very high, due the overfitting i.e. is an excessive accuracy of the function that therefore leads to a lack of generalization.



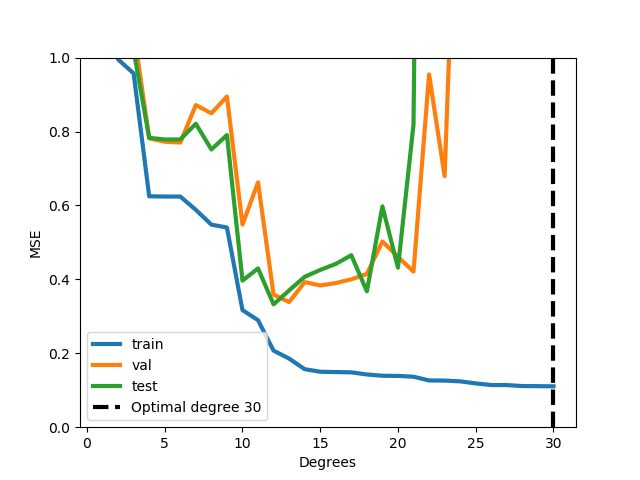
### 3) Best degree for validation set

Without regularization the function resulting from the validation set will be a grade 13 polynomial. This function here will be useful for us to notice the overfitting in the training set. As we will see later on, the error function of the validation set is very close to that of the testing set.

### 4) Error functions

Here below the graph of the function of errors for the different sets of data is shown. As we can see the functions have the optimal cost that coincide with the optimal degree.

As previously mentioned, the cost function of the validation set resembles that of the test set. This is because the validation set (graph above) is a simplified testing set from which all outstanding values have been removed. Also the optimal degree of the two datasets is similar (12 for testing and 13 for validation).

 As you can see from the graph above, the training set has a cost function that continues to decrease. This is due to the fact that the data set is not regularized and so it will continue to improve as opposed to the validation and testing set. This causes the overfitting phenomenon that can be solved through regularization (which we did by mistake and that we attach at the end of the document).

### 5) Discussion

As we were able to see from the results, when using the training set, an increasing polynomial degree means a decreasing MSE. This is because at some degree, the function has enough “power”/complexity to match every input in the training set to its corresponding output. But the best degree when using other data sets is not always the highest.

In fact, a very complex polynomial might be able to minimize the error on some pre-learned data set, but this complexity is going to hurt when trying to predict the output of some new data. This is called over-fitting.

The scope of the polynomial function is to match the structure of some data set and not to exactly match the training set input-output pairs. This is why we need a validation set. A validation set is important to make guesses about how your design is going to perform when used with new data (e.g. the training set). With the help of a validation set we can make decisions on the hyper-parameters, in our case the degree of the polynomial, without having to worry about noise in the data set.

## 2 Logistic Regression

### 2.1 Derivation of Gradient

Given

Show that

To make the derivation easier, we simplify by using the correlation and the definition of the sigmoid function and some of the basic logarithm rules.

Now that is greatly simplified we can start the derivation.

q.e.d.

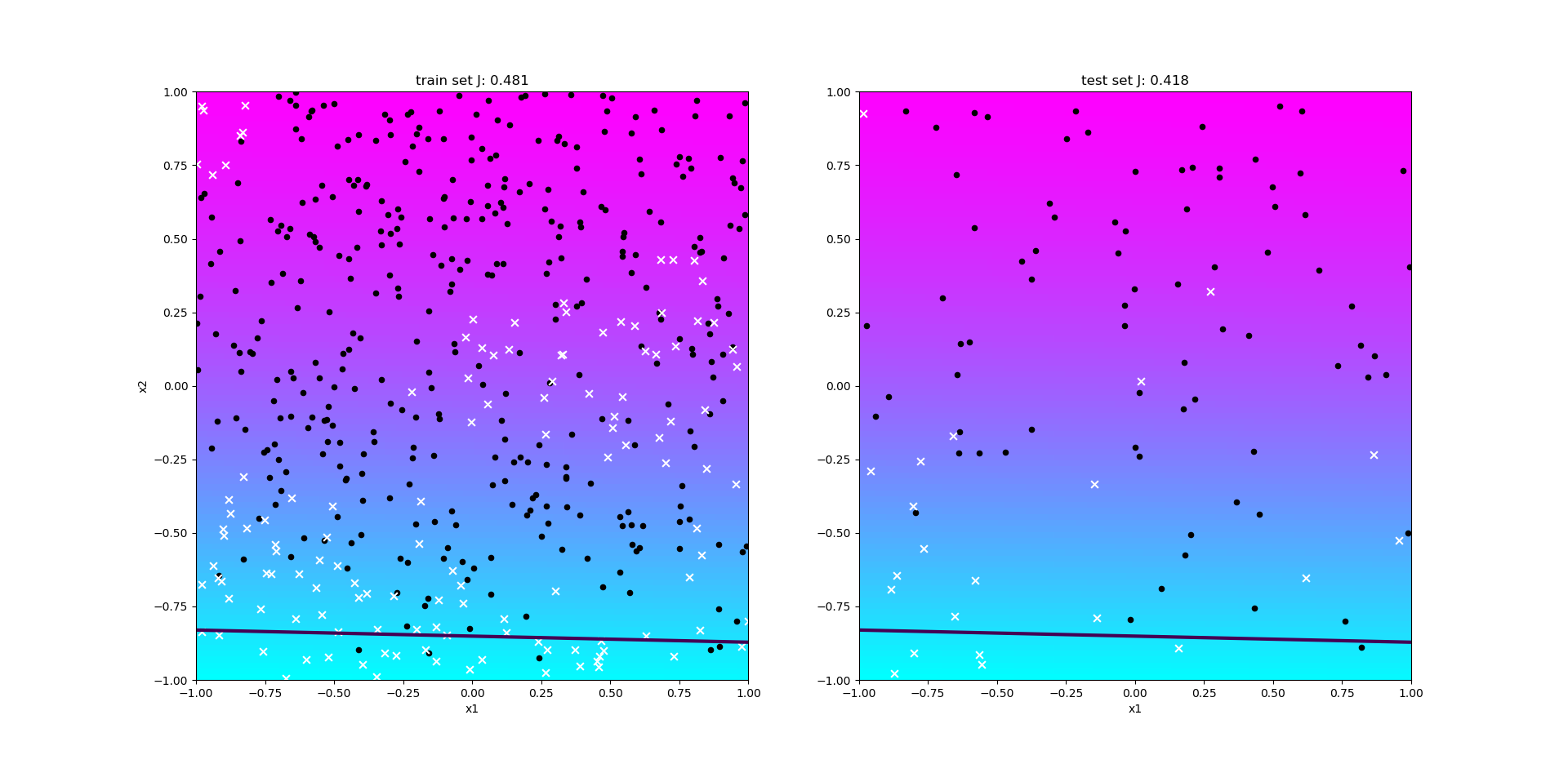
### 2.2 Logistic Regression training with gradient descent

#### 2.2.1 What is check\_gradient doing?

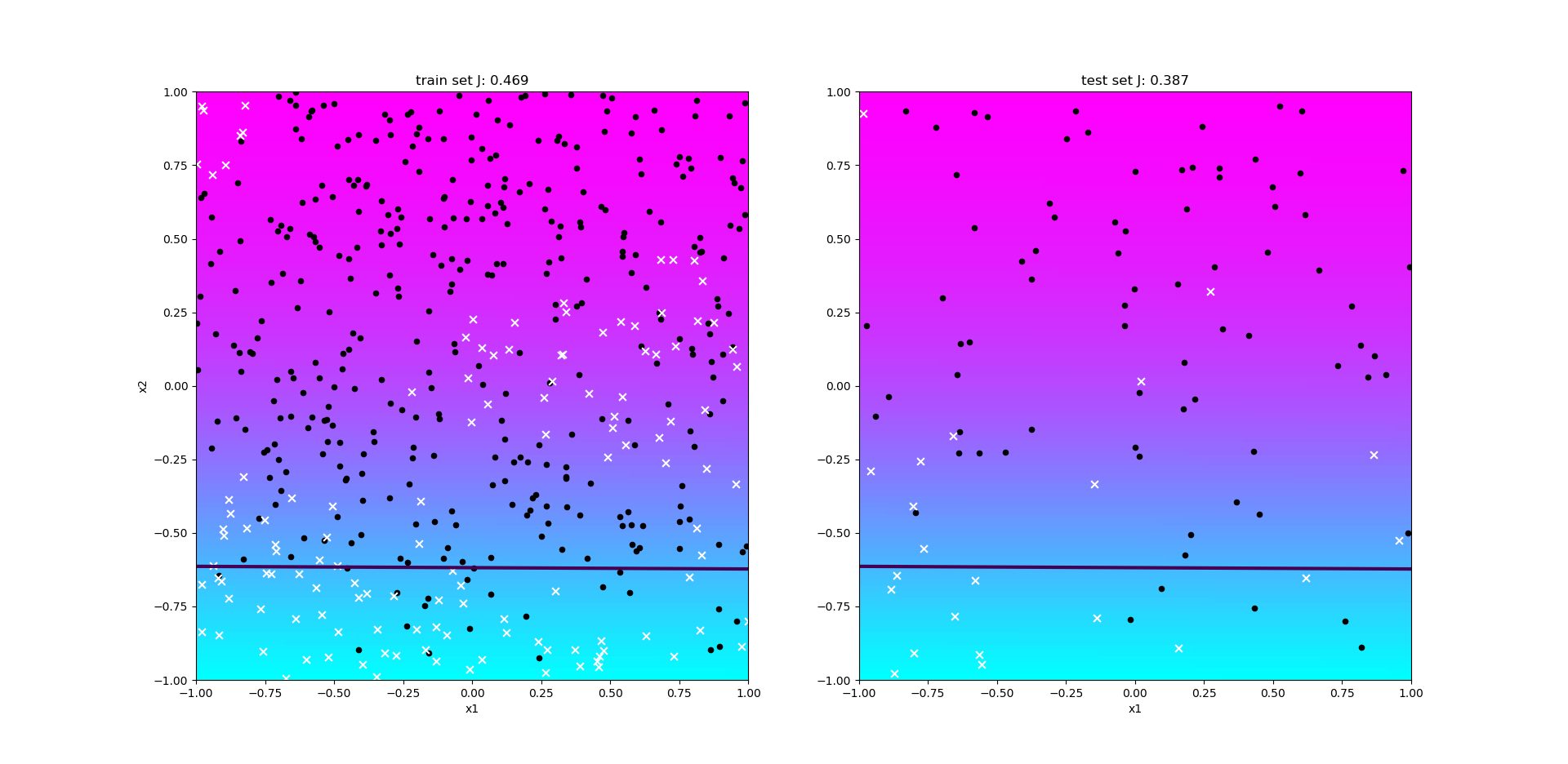
The function manually calculates a gradient and compares it to our implementation, if the calculated error is smaller than the given limit, our gradient passes the test.

##### 2.2.2

For degree l = 1 run GD for 20 and 2000 iterations with learning rate eta = 1



Training and test results with eta = 1 and 20 iterations



Training and test results with eta = 1 and 2000 iterations

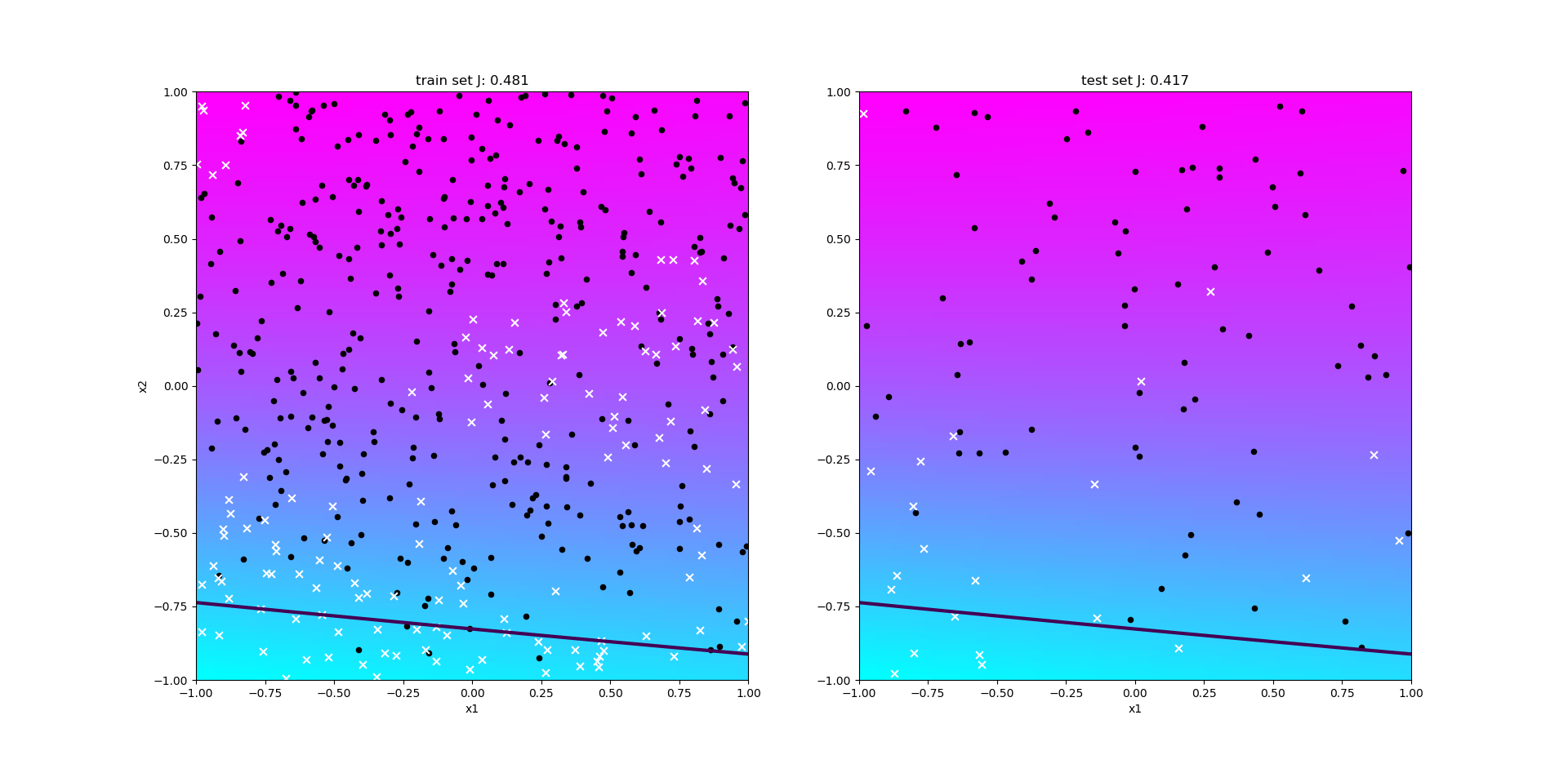
Using 20 iterations results in a training error of 0.481 and a testing error of 0.418.

Using 200 iterations results in a training error of 0.469 and a testing error of 0.387.

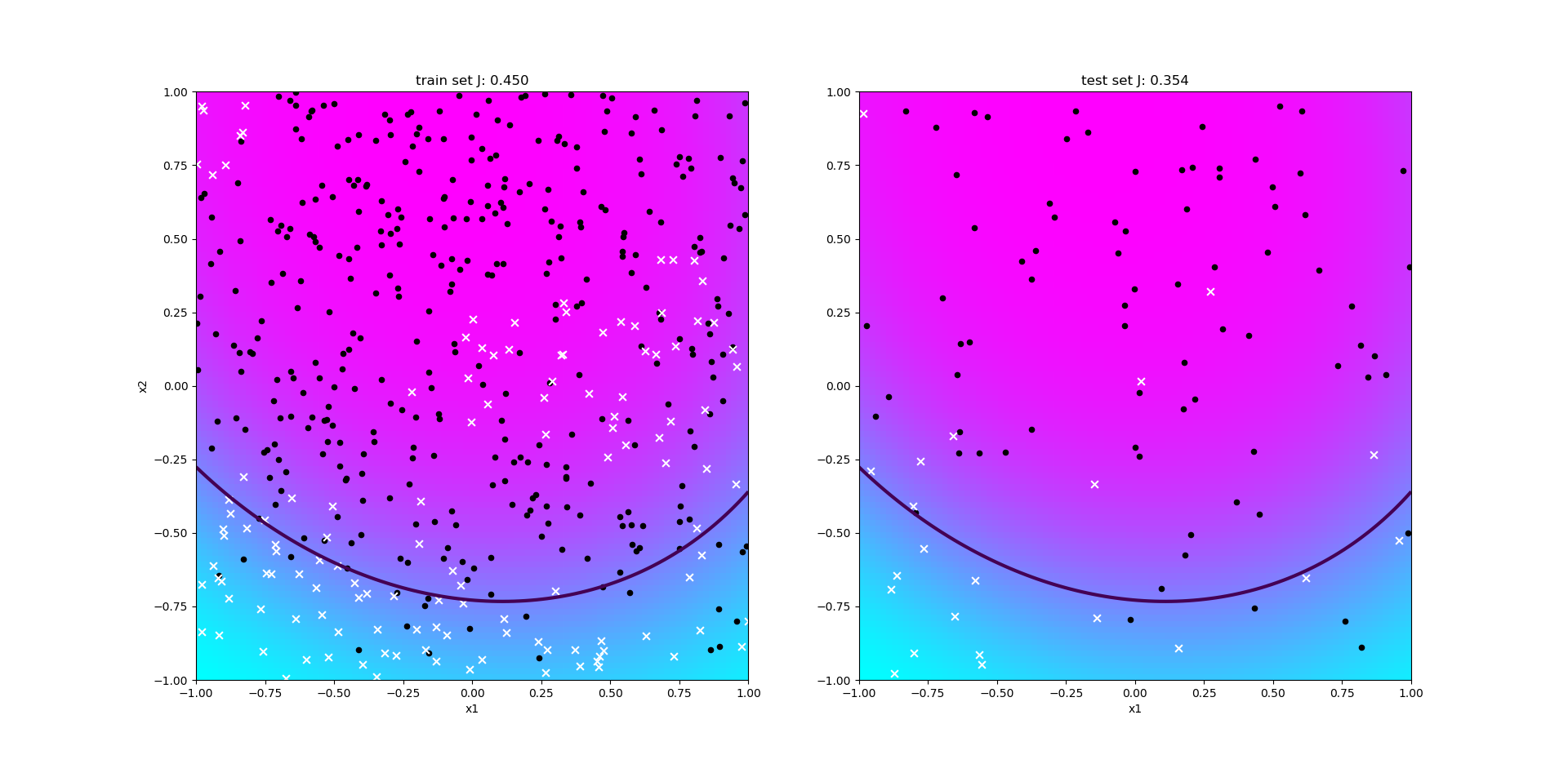
Using less iterations results in a bigger error, while too much iterations take a very long time to compute.

#### 2.2.3

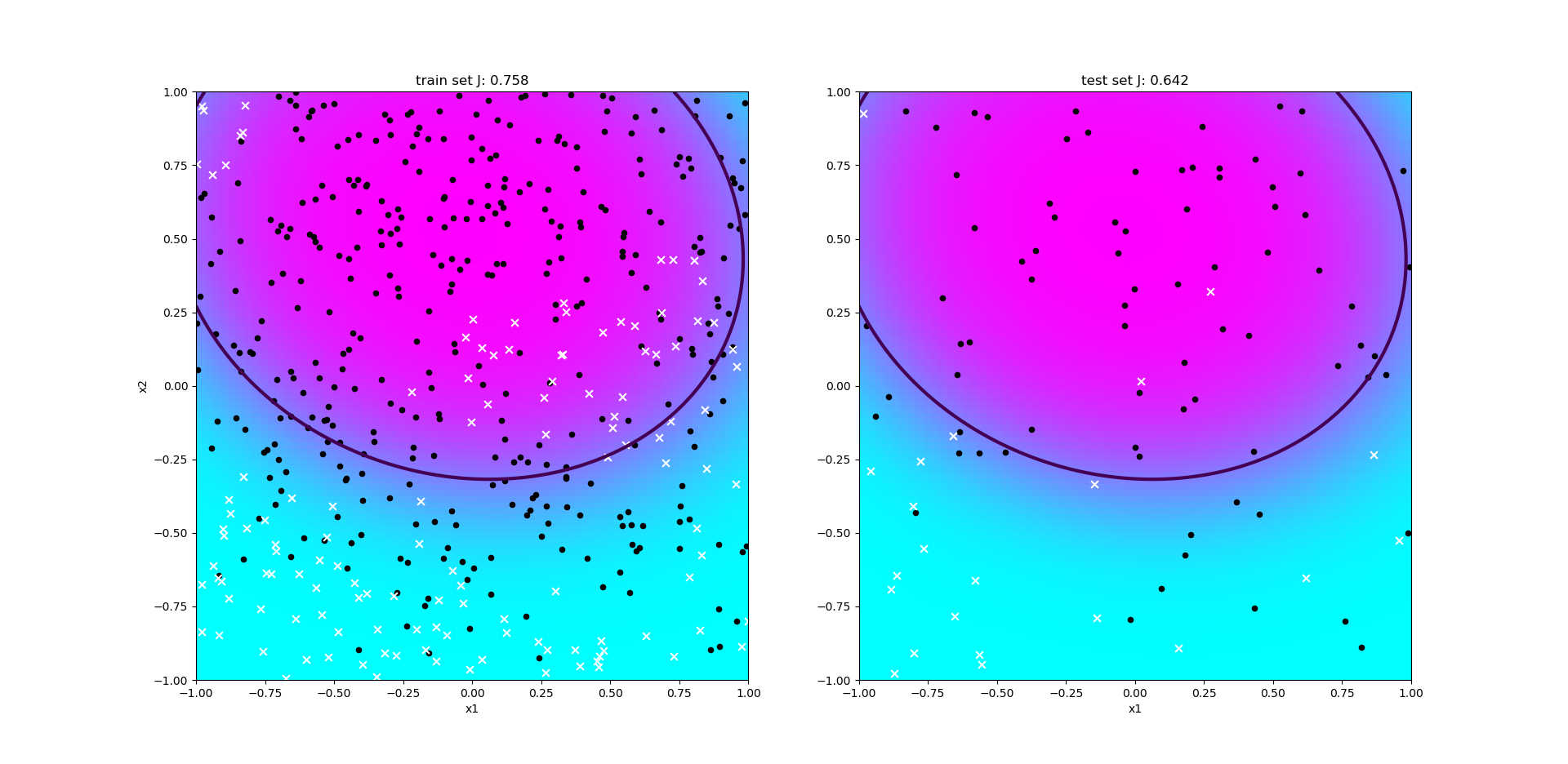
For degree l = 2 run GD for 200 iterations with learning rates of eta = 0.1, eta = 2 and eta = 20.



Training and test results with eta = 0.1 and 200 iterations



Training and test results with eta = 2 and 200 iterations



Training and test results with eta = 20 and 200 iterations

Using an eta of 0.1 and 200 iterations results in a training error of 0.481 and a testing error of 0.417.

Using an eta of 2 and 200 iterations results in a trainings error 0.450 of and a testing error of 0.354.

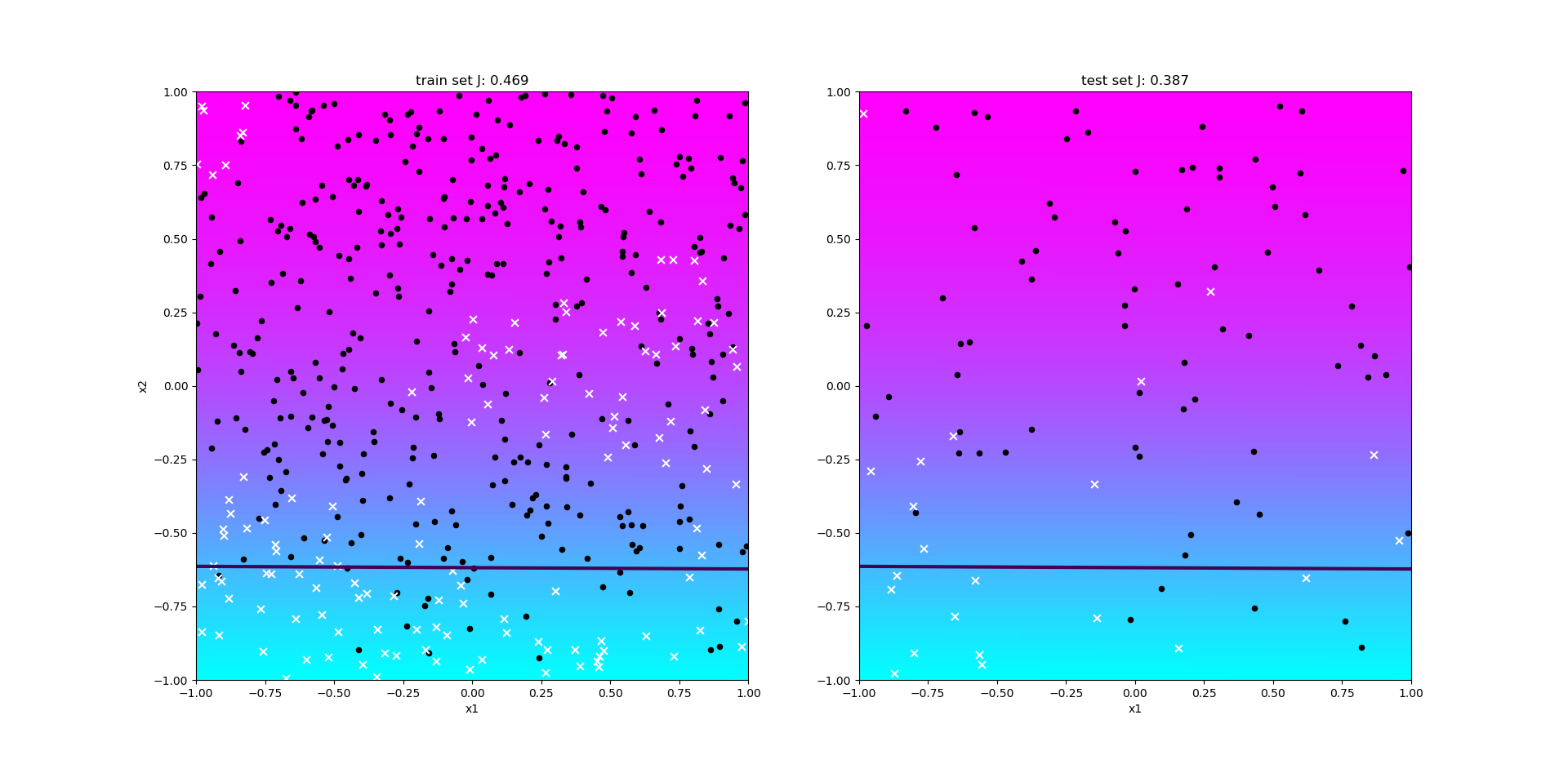
Using an eta of 20 and 200 iterations results in a trainings error of 0.758 and a testing error of 0.642.

By using an eta that is to small the problem of underfitting occurs, and by using an eta that is to big overfitting occurs

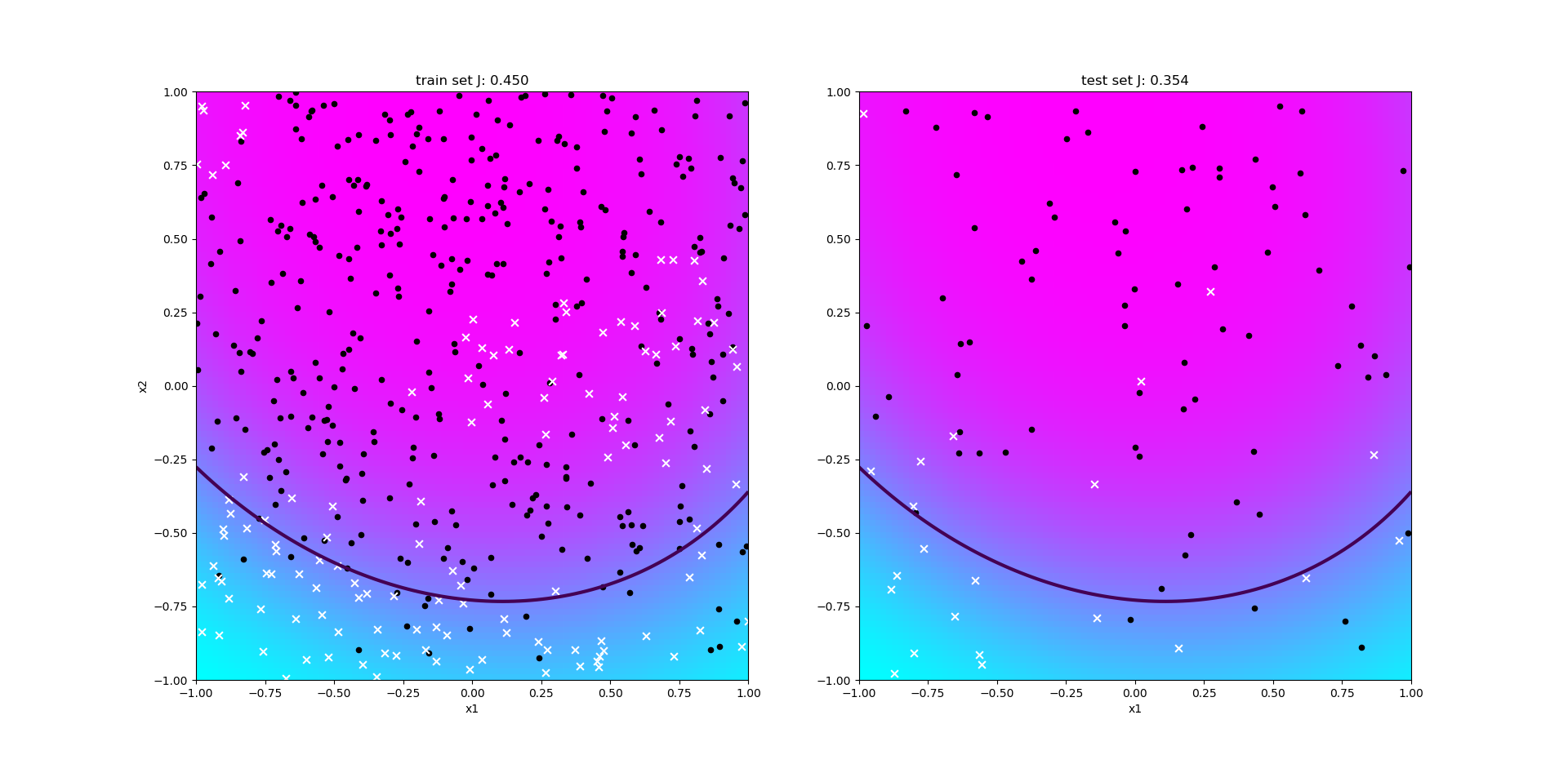
#### 2.2.4

Identify reasonably good pairs of values for eta and max\_iter for the degrees l = {1,2,7,20}

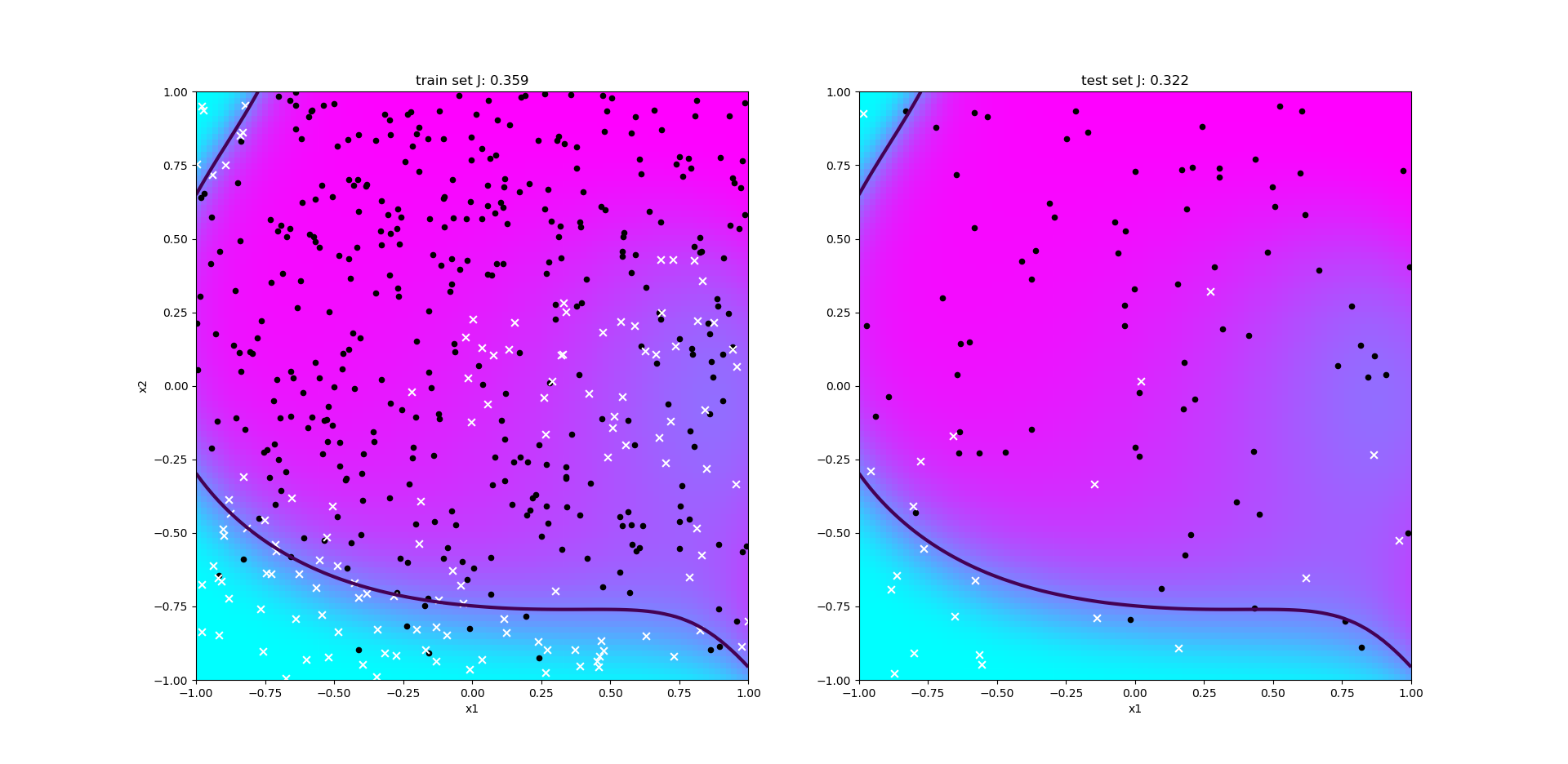
By taking the results of 2.2.2 and 2.2.3 into consideration, we have chosen eta = 2 and 200 iterations for all degrees.



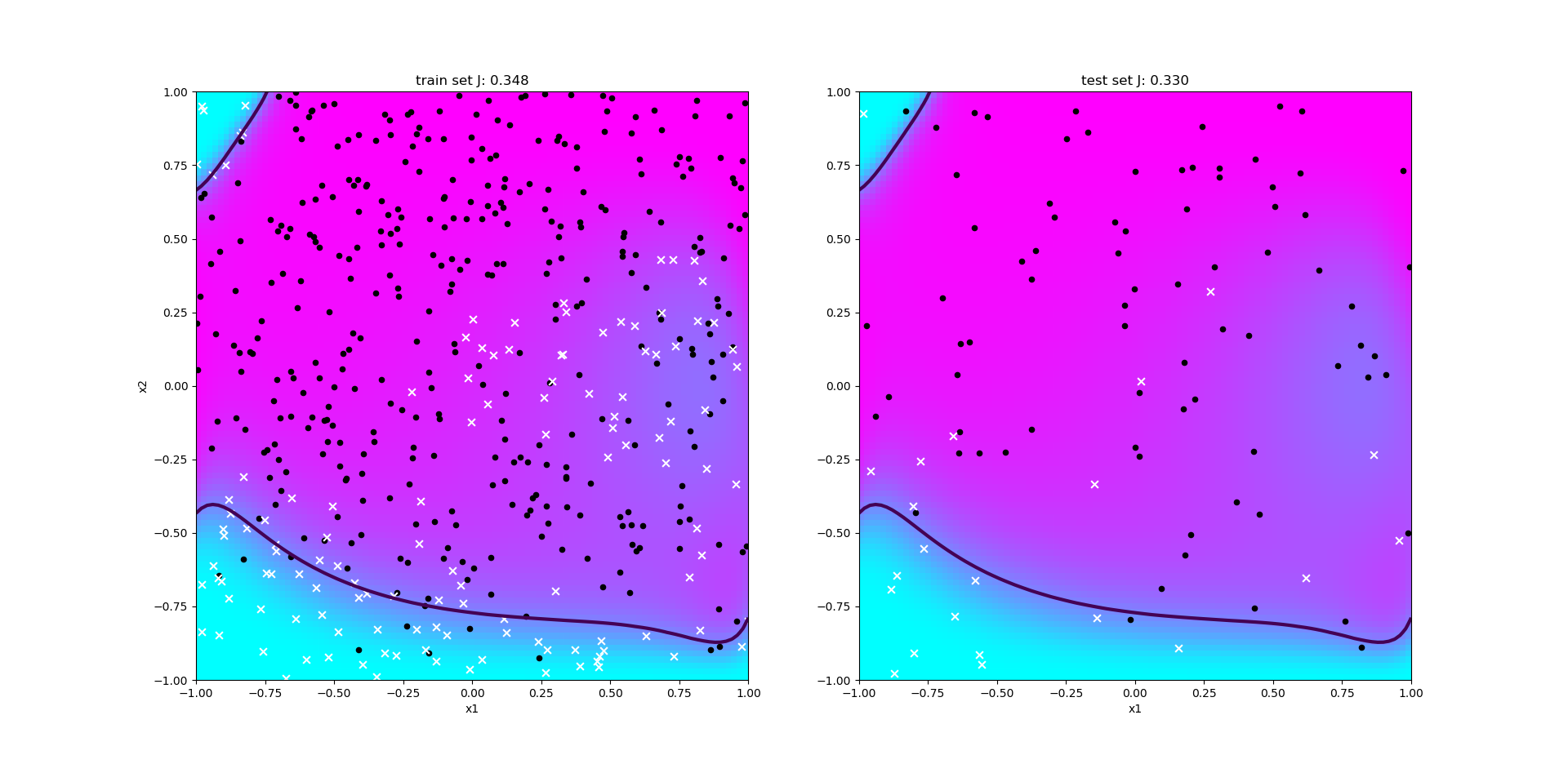
Training and test results with degree = 1, eta = 2 and 200 iterations



Training and test results with degree = 2, eta = 2 and 200 iterations



Training and test results with degree = 7, eta = 2 and 200 iterations



Training and test results with degree = 20, eta = 2 and 200 iterations

Using said eta and iterations following results have been computed:

degree = 1: training error = 0.469, test error = 0.387

degree = 2: training error = 0.450, test error = 0.354

degree = 7: training error = 0.359, test error = 0.322

degree = 20: training error = 0.348, test error = 0.330

Even though it is possible to compute the result in a short amount of time, degrees 1 and 2 have the problem of underfitting. And degree 20 takes a long time to compute and also has the problem of overfitting. Therefore degree 7 seems the most appropriate to fit the given data.

#### 2.2.5 Describe a possible stopping criterium using the gradient of the cost function.

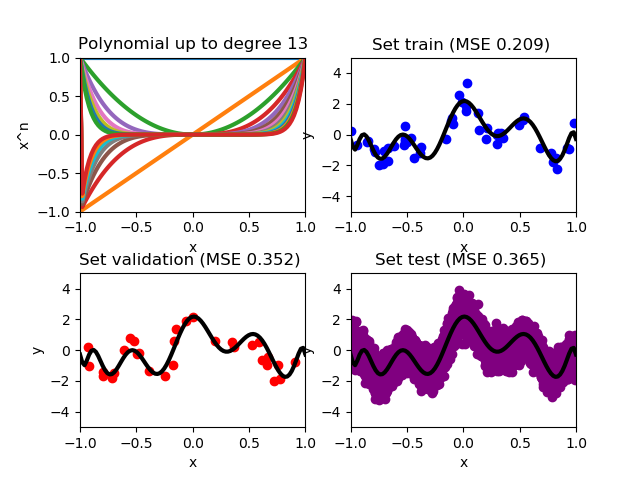
A possible stopping criterium could be the convergence of the gradient, that means to stop, once the gradient converts towards 0.

## Attachment A

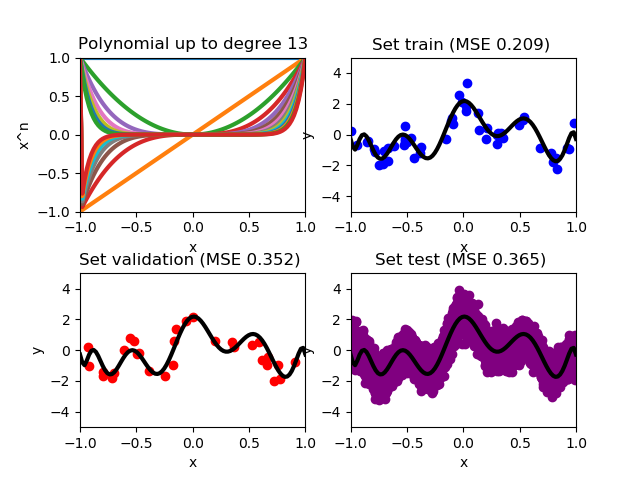
The following part of the document is a mistake that we made at the beginning. We didn’t read the comments in the .py skeleton and used a regularized cost function.

The result were very interesting and this is why at the end we didn’t want to completely throw away what we did and we decided to leave it here as an attachment.

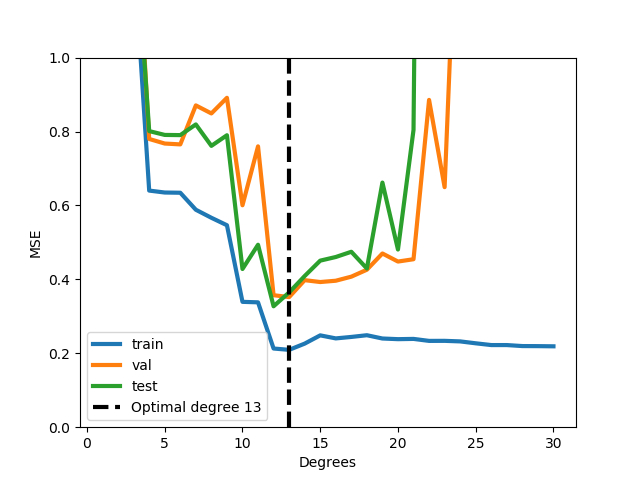
2) Best degree for training set

For this step we have chosen to take Lambda = 0.55 to have a strong regulation, as we will see later the bigger Lambda the more the training set and validation set will look like each other. Here below the graph of the function in every set of data with the optimal degree founded for the training set (13). With a Lambda even slightly lower and therefore a weaker regularization we would have completely different results.

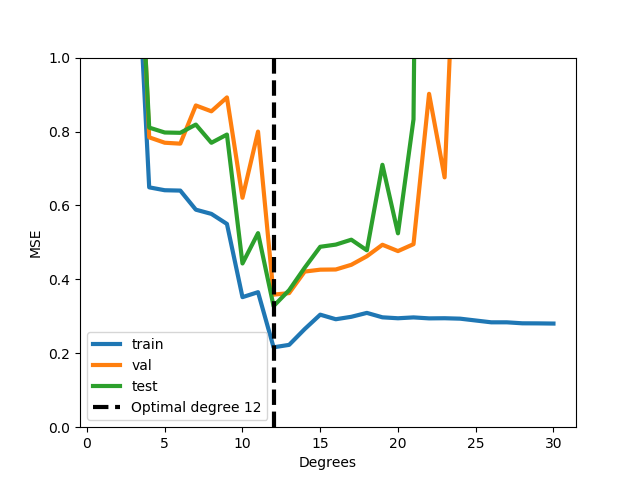
### 3) Best degree for validation set

Here below the graph of the function in every set of data with the optimal degree founded for the validation set (13). Lambda is always = 0.55.

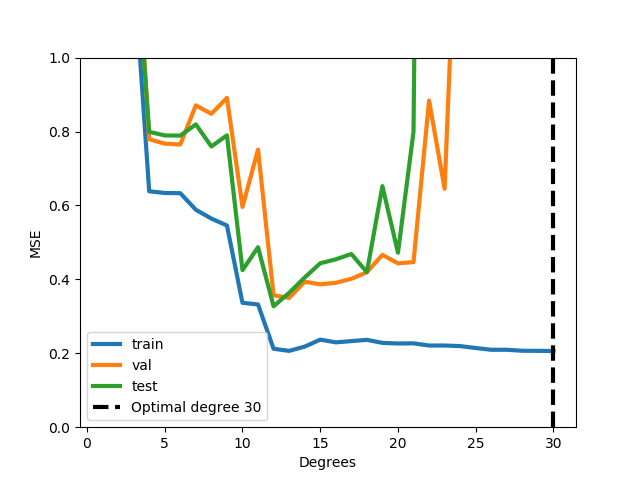
### 4) Error functions

Here below the graph of the function of errors for the different sets of data is shown. As we can see the functions have the optimal cost that coincide with the optimal degree.

The test set has a MSE that is smaller than the MSE of function with degree = 13 for a function with degree = 12. This is probably due a not so high regularization. In fact if we increase Lambda = 0.8 we get an optimal degree for validation and training set = 12 (as shown below).

If instead of increasing lambda we reduce it (e.g. lambda= 0.5) we obtain different optimal degrees for each data set:

END GRAPH



Cost functions and optimal degree for the training set with lambda = 0.5

### 5) Discussion

Cost functions and optimal degree for the validation set with lambda = 0.5

What we can deduce from the derived data and graphs is that thanks to the regularization we can improve the values of the training set in order to make them more truthful and reliable and removing the problem of overfitting. The validation set in this case serves just to this, in the moment in which the results obtained from the training set approach those obtained from the validation set we could consider them valid.

In more we have discovered the differences between a weak regularization and a strong regularization, in fact with the variation of lambda also changes the degree of optimal function. In our case we took for the final result lambda = 0.8 so that the optimal functions for the 3 data sets coincide in a function of degree 12.

### 6) Final answers to question 2 & 3

Considering the fact that a strong regularization is better than a weak one we have chosen as final value of lambda 0.8 obtaining therefore, as already said before, a function of grade 12 as function that reduces to the minimum the costs for all data sets. (Graph showing the error functions is shown above – END GRAPH). With a Lambda of value 0.55 in fact the training set will present the same function as the validation set but both of them will differ from the testing set.

1. <https://en.wikipedia.org/wiki/Gradient> [↑](#footnote-ref-1)
2. <https://math.stackexchange.com/questions/1519367/difference-between-gradient-and-jacobian> [↑](#footnote-ref-2)