Classification of physical activities

Machine learning

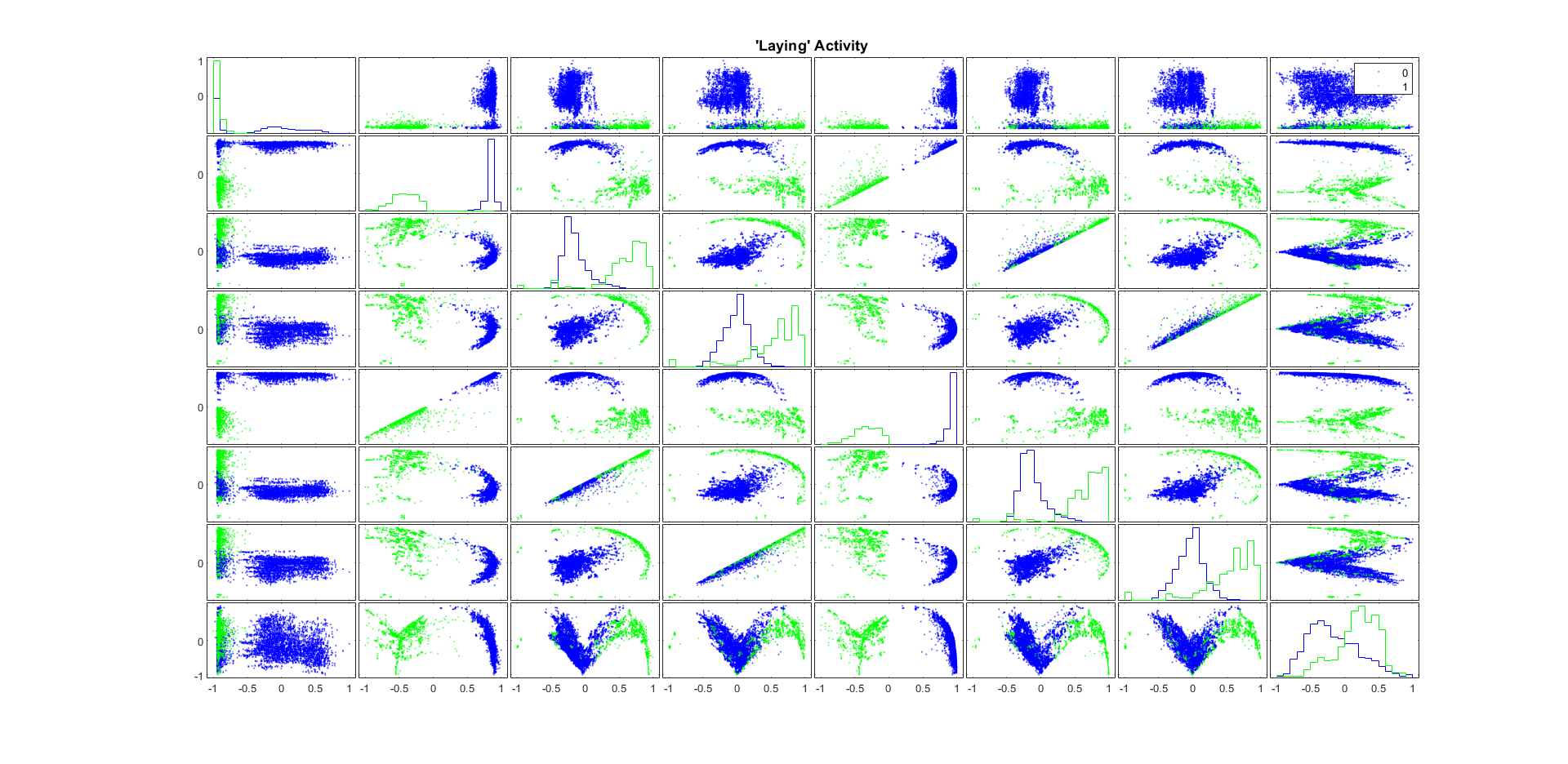
Soufiane Salama, michel banken

2019

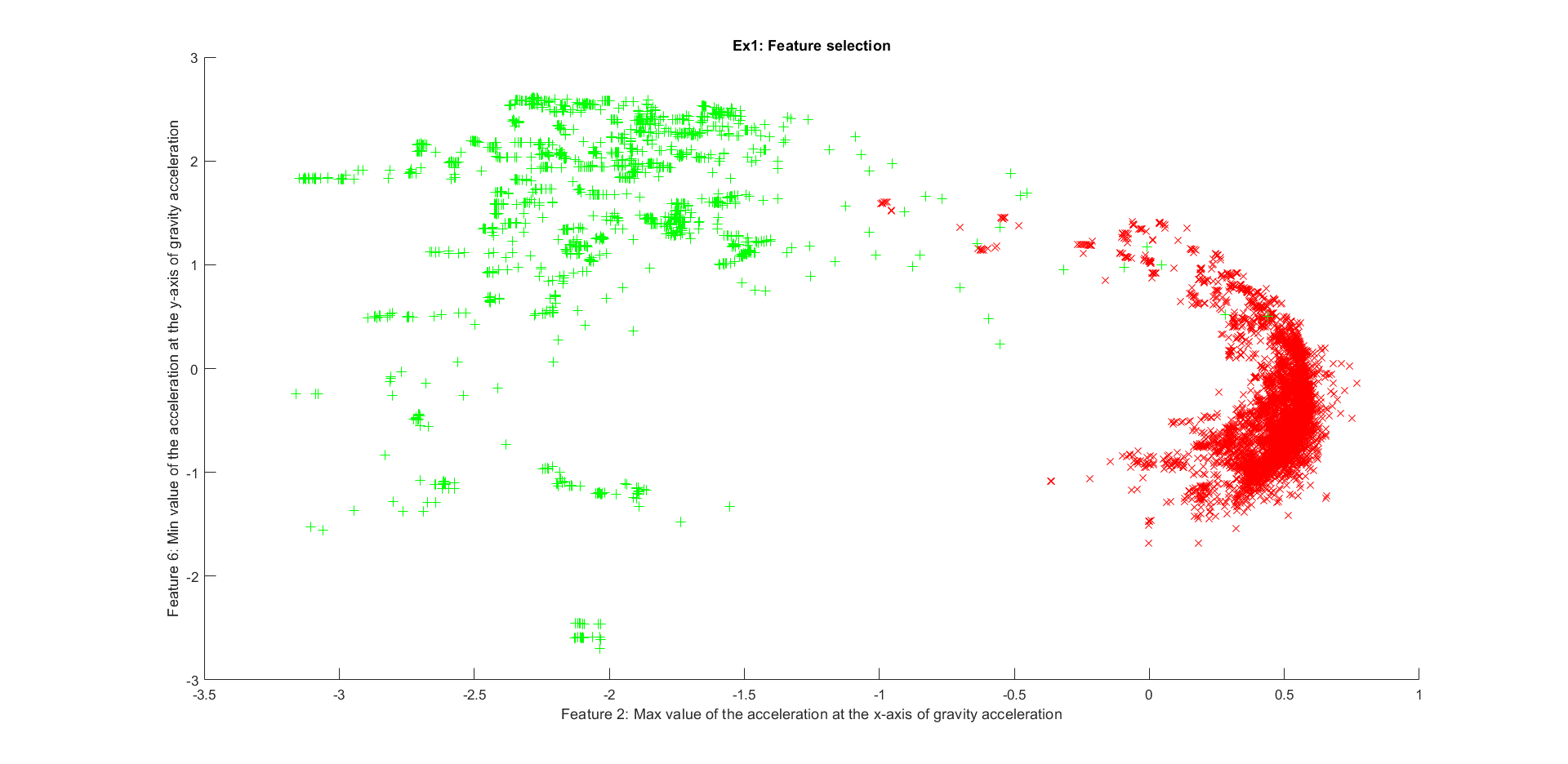
# Exercise 1: Feature selection

In the assignment one activity has to be selected as the “interested” activity for classification and two features with distinguished separation.

After plotting all the possible combinations of features we have selected the activity ‘Laying’, because multiple features according to this activity show a good and distinguished separation.



One of those sets of features consists of feature 2 and feature 6. The report will use these two features as selected features. The figure below plots the two features individually for a clear view of the features and separation.



# Exercise 2: Classification: Logistic regression

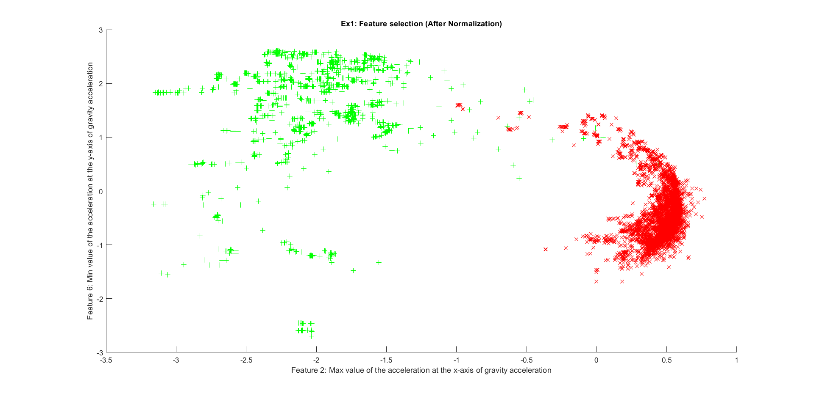
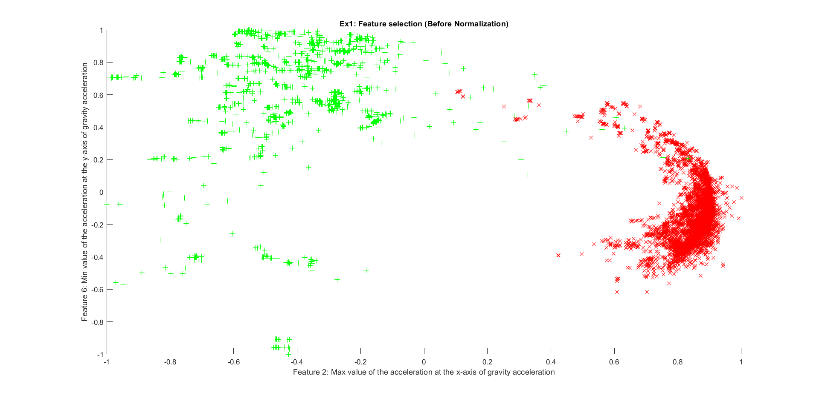
## 2.1 Cost function and gradient

Adding the MATLAB functions from programming exercise 2.

## 2.2 Linear model with 2 features

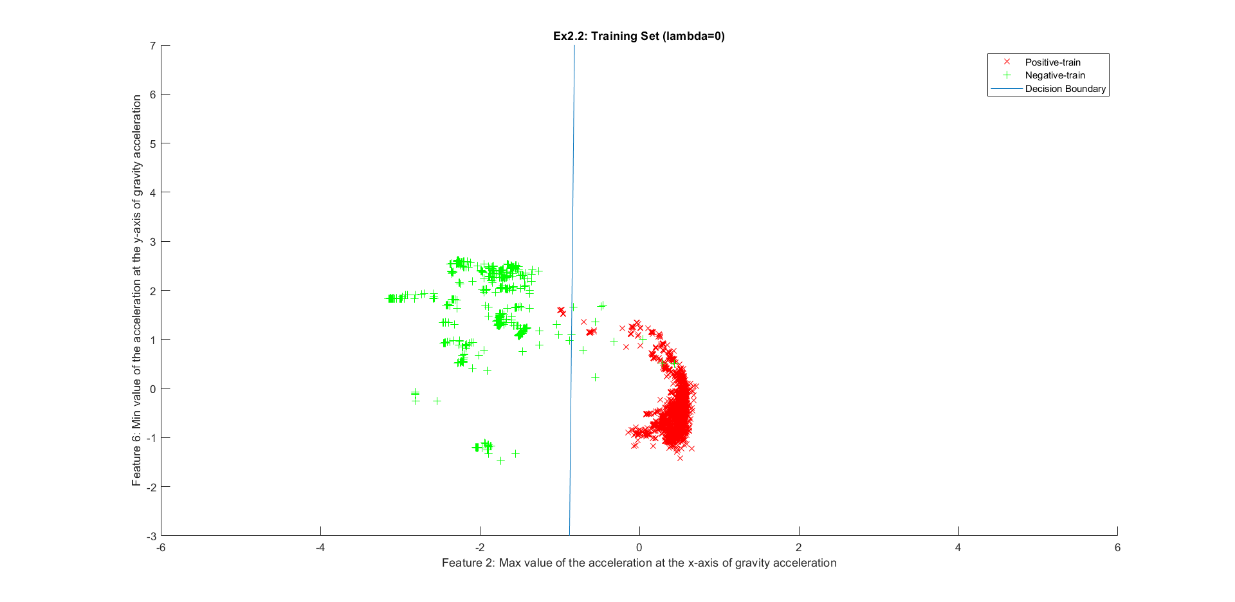
After selecting two features, normalization (Mean Normalization) was implemented. The features are now better scaled and show a better range and in general more circular shaped. The difference in values (see axes) can be … in figure .. where the features are plotted before and after normalization.

Mean Normalization:

Now that the features are normalized we need to dived the dataset into three sections, Training, Cross Validation and Test dataset with the ratio of 0.4:0.3:0.3.

The linair decision boundery for the two features in the training set looks as follows.



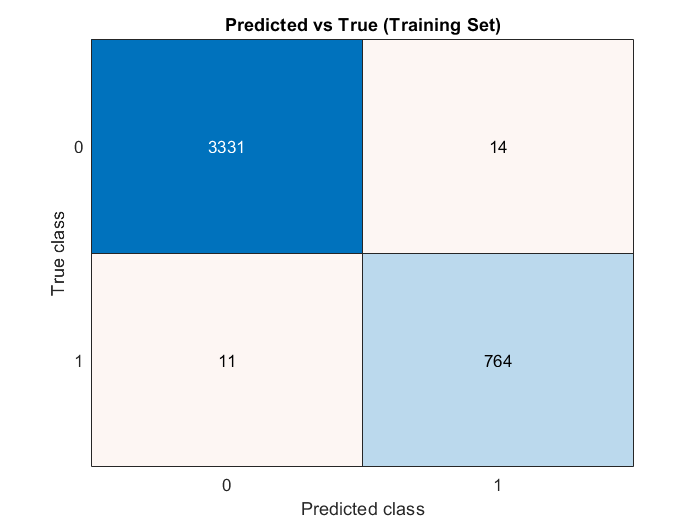
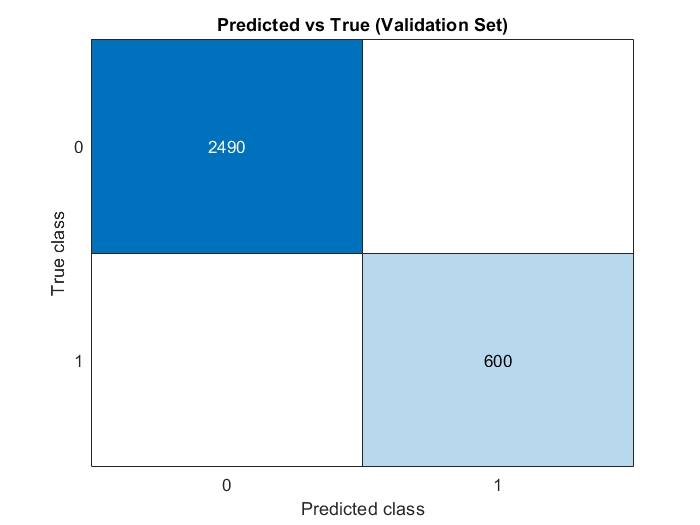
The obtained F1 scores are as followed:

* Training set -> 0.983902
* Cross Validation set -> 1.00

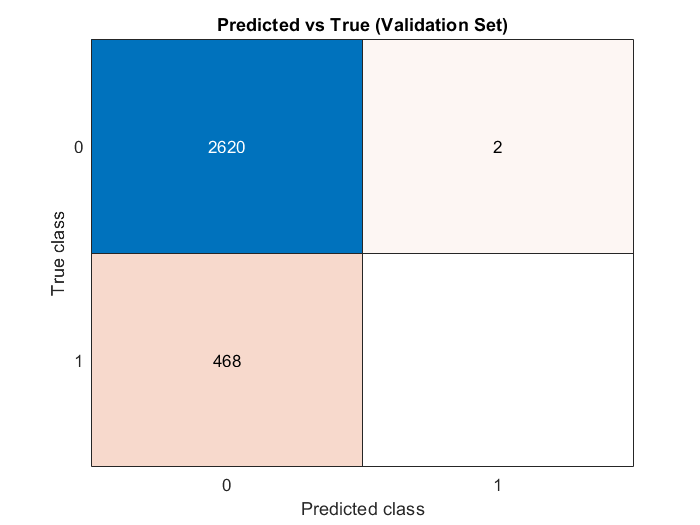
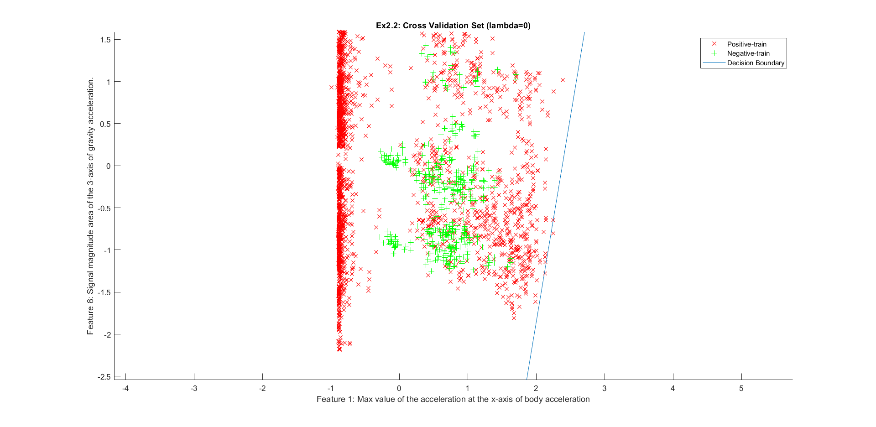
F1 Score:

where precision: and recall:

MATLAB has a built-in function “confusionmat” (and “confusiongraph”) that will calculate the ‘true positives’ (2;2), ‘false positives’(1;2) and ‘false negatives’(2;1) that are needed to calculate the precision and recall values .

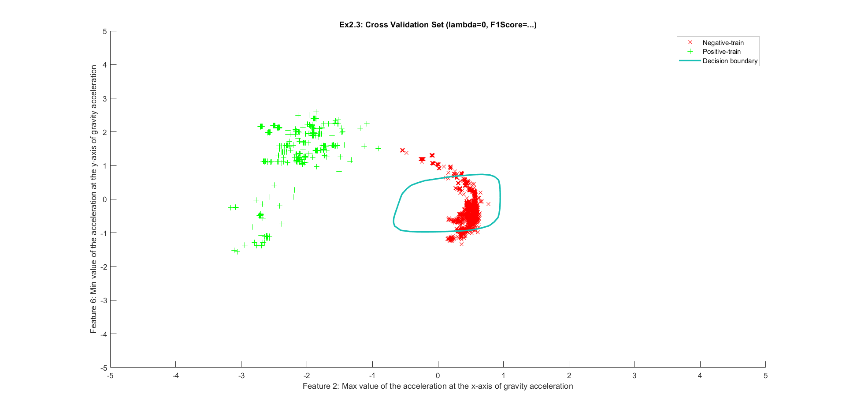
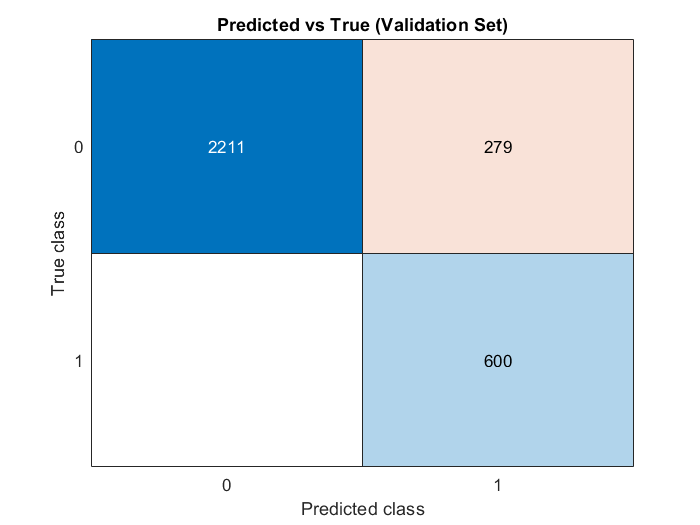
 

The obtained F1 Scores are very high, but this was already known by looking at the plotted decision boundery. If we now for example use two features and activity which doesn’t have a good seperation like features 1 and 8 for activity ‘Walking Upstairs’, we get the following (linear) decision boundary :

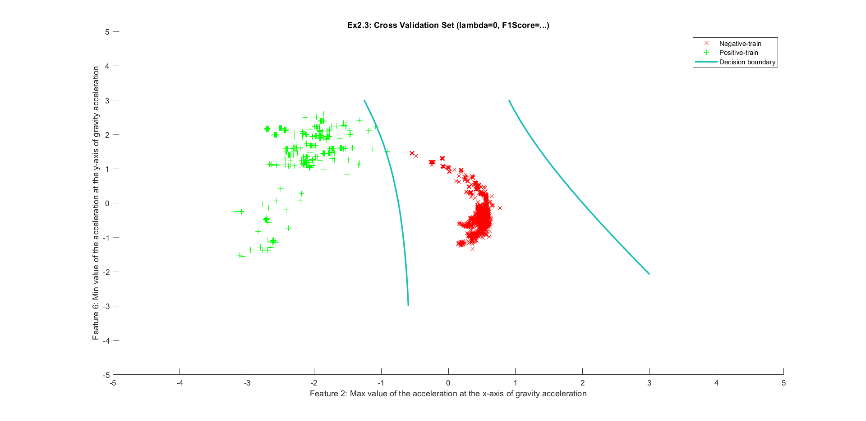
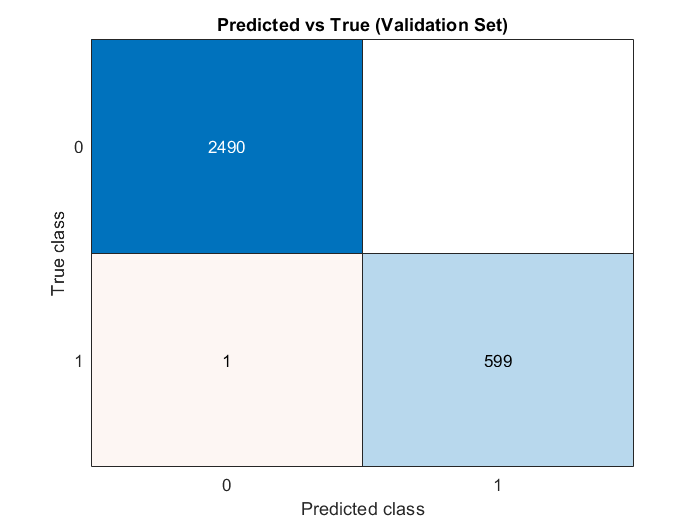


You can see that is not possible for the algorithm to draw a linear decision boundary because the positive and the negative are not clearly separated. Also the F1 Score cannot be calculated (NaN) because our hypothesis hasn’t predicted any class that is positive as you can see in fig. X. For examples like these is not enough to work with a linear hypothesis, instead it is necessary to work with a polynomial hypothesis. 2.3 Polynomial features from 2 features

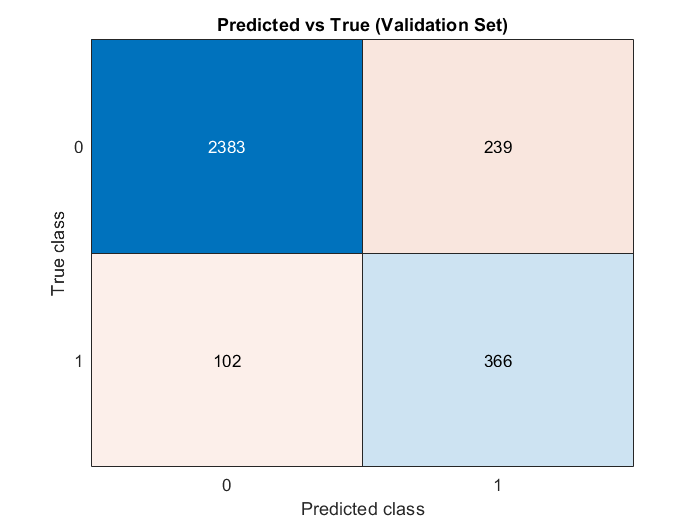
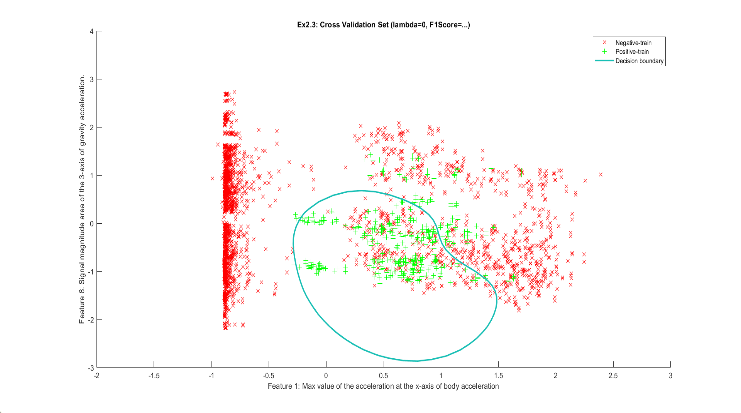
To improved our hypothesis we can map the features in polynomial terms. Depending on the situation you can see vary the degree of polynomial, for example a degree of 6 will give use 28 features. Using this feature vector the decision boundary can greatly be improved. In our case where we already had a very successful hypothesis, it wasn’t necessary to do this. In matter fact our F1 Score has even dropped to **0.811359** (Cross Validation Set) by using a 6 degree polynomial as hypothesis.

Even if we mapped the features to a polynomial of degree 2, the F1 Score (0.999166 Cross Validation Set) is still very good but the implementation of the feature mapping has no additional value.

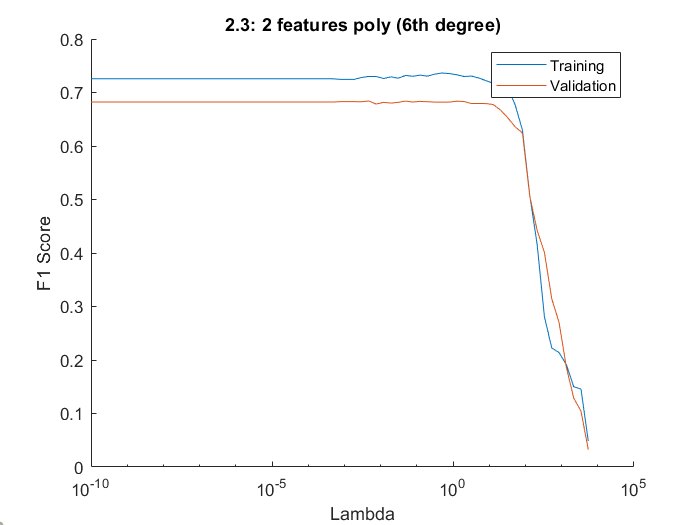
That is why we continued with features 1 and 8 for the activity of ‘Walking’ so we can still optimize the F1 Score by changing lambda. After mapping the features to a polynomial of degree 6, we get a F1 Score of **0.682199**:



Dit was allemaal met lambda = 0

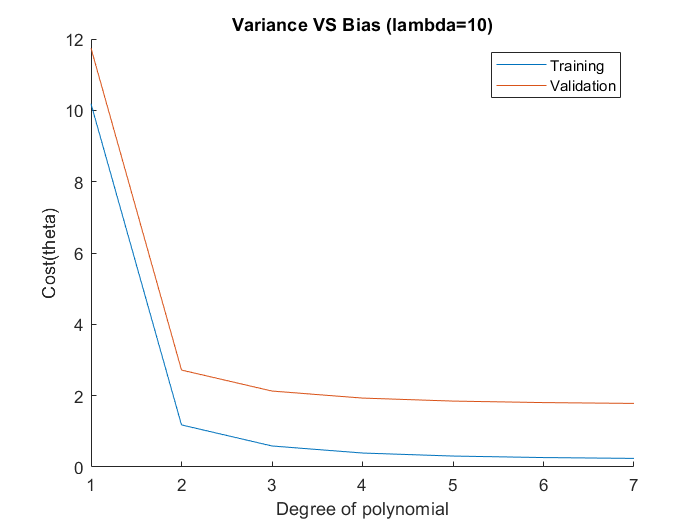
### Optimizing lambda

To optimize the F1 Score we can vary lambda between the interval -3^(10):3^(10), this can be done by using the MATLAB function logspace between -4.77 and +4.77. We used logspace because linspace didn’t give accurate values.

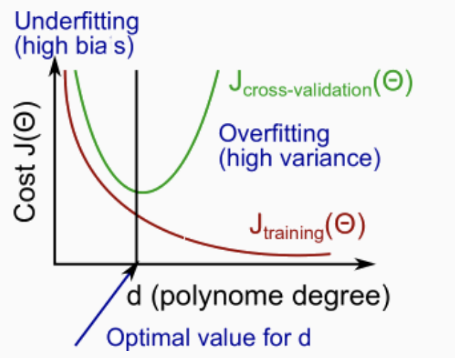


We can conclude that the F1 score keeps at a maximum for values of lambda under 10.

### Variance and bias

The variance and bias or over- and underfitting problems is an issue depending on the degree of polynomial. To see which of the two problems we have, we can visualize the cost of the training and validation set in function of the degree of polynomial.

Because we implemented regularization in the cost function, we already prevented overfitting. So the Cross Validation doesn’t increase over time.

Rule of thumb

Bias problem (underfitting):

* J\_train\_error will be high
* J\_cv\_error will be ≈ as J\_train\_error

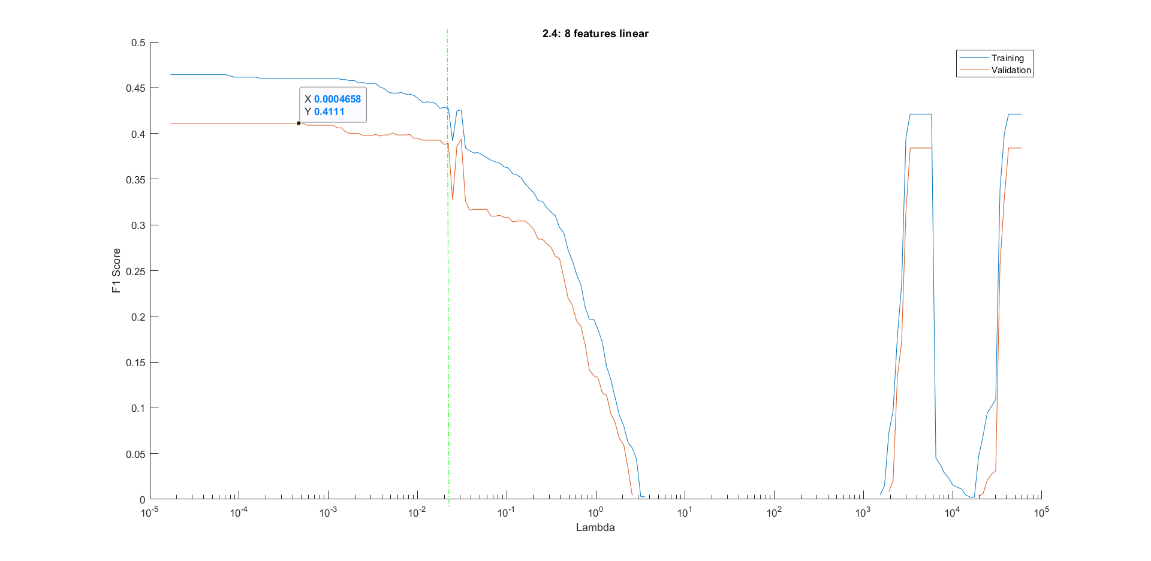
Variance problem (overfitting):

* T\_train\_error will be low
* J\_cv\_error will be much higher than J\_train\_error

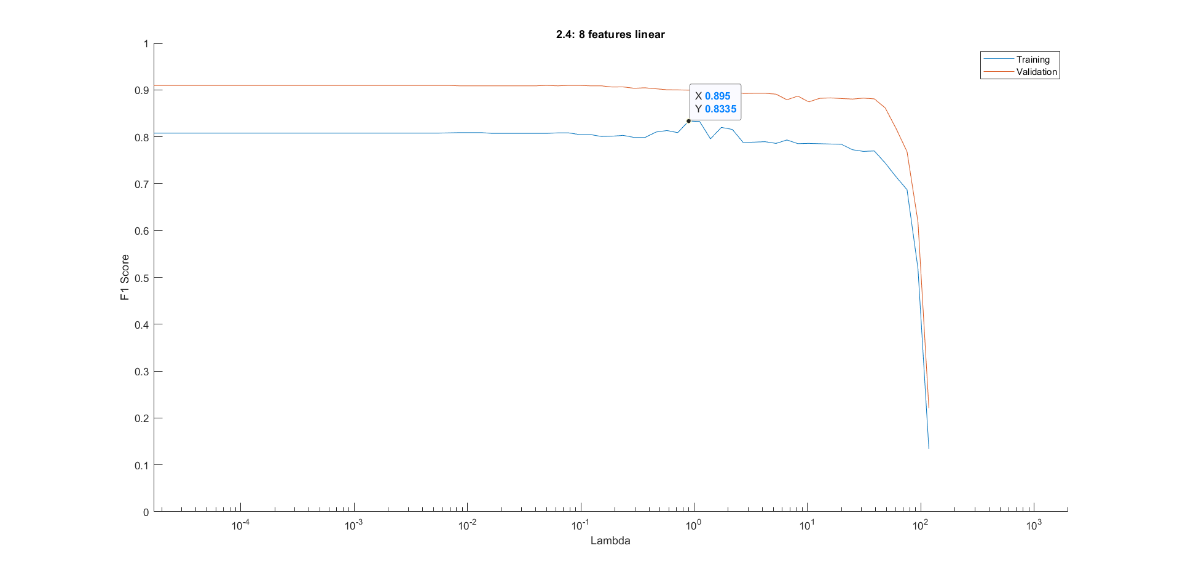
So we can conclude that for degrees of polynomials under 2 we have a bias problem (underfitting) and from there on we keep getting a lower cost. And with regularization we have no variance problem. So in our case where we mapped the features to a sixth degree, we have no variance or bias problem.

## 2.4.1 Linear classifier with 8 features

Instead of only using the two ‘ideal’ features, we now calculated the F1 score in function of the eight features. For the activity ‘Walking’, our F1 score has significantly decreased to **0.4111** and from lambda = ± 0.02,the F1 score keeps decreasing. Also if lambda has reached a value of ± 2\*10^(3), the F1 score keeps going up and down. This depends on the chosen activity.

****

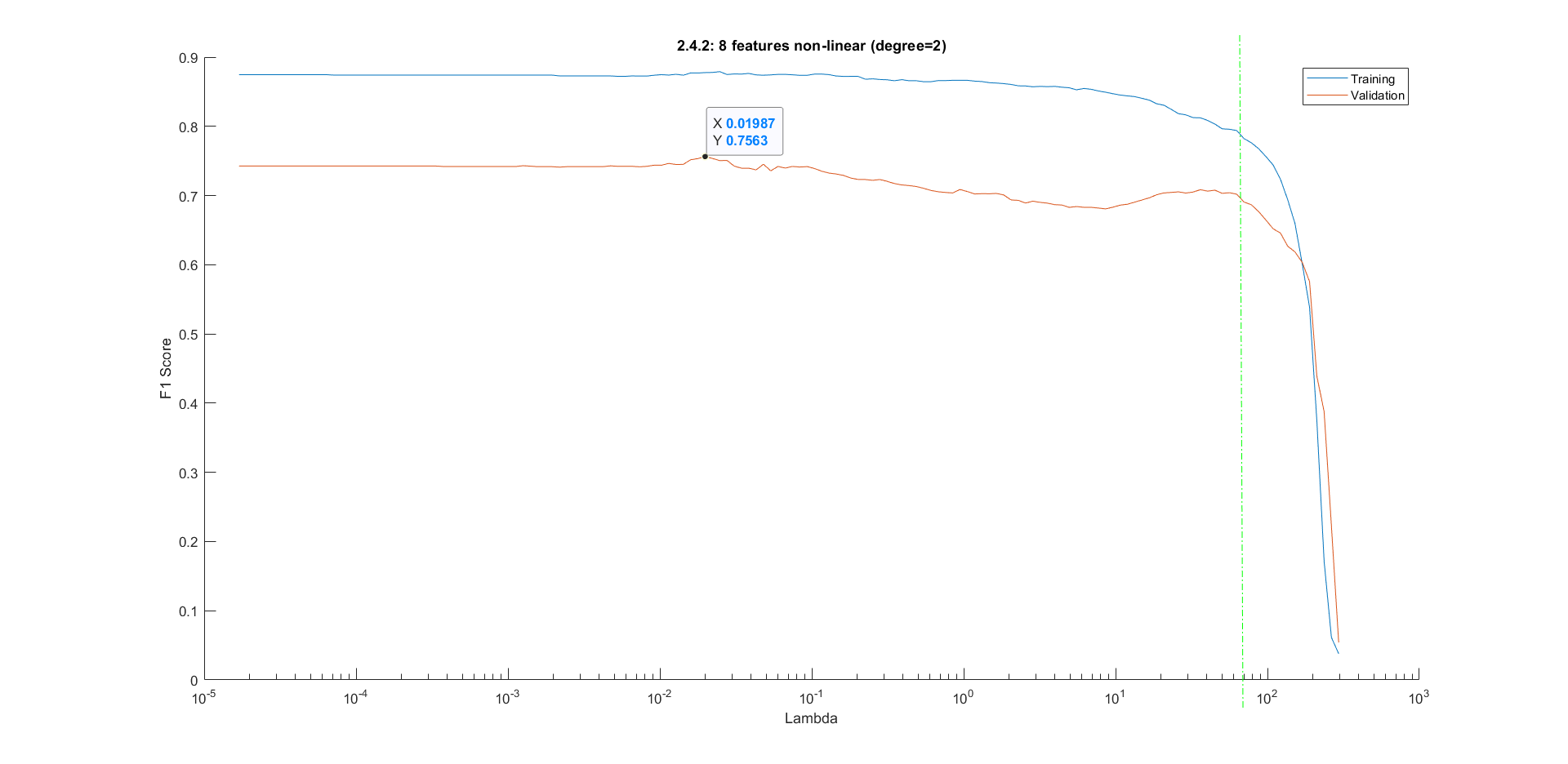
If we now for example use the activity ‘Sitting’ we get the following plot:



## 2.4.2 Non-linear classifier with 8 features

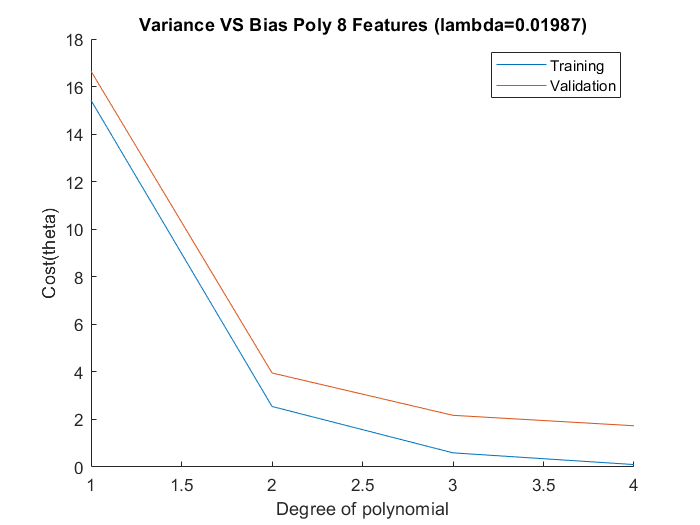
To improve our previous F1 score (0.4111), where we made use of a linear hypothesis of the eight features, we can now map these features to, for example, a quadratic (degree=2) polynomial.

As for result our F1 score has indeed increased to maximum of **0.7563.**

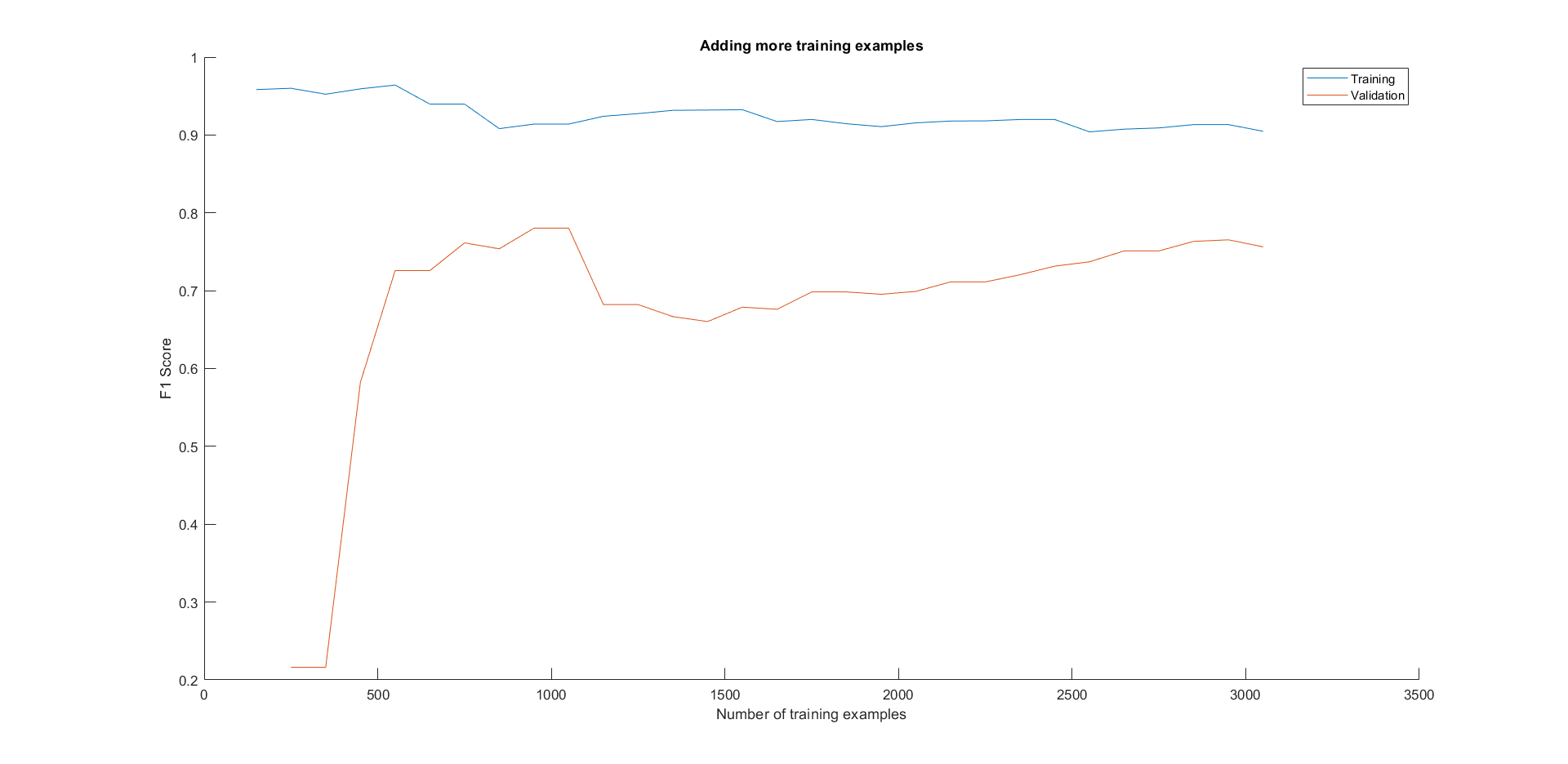


### Variance and bias

To see which of the two problems we have, we can visualize the cost of the training and validation set in function of the degree of polynomial like in chapter 2.3. For lambda we used the value that gave the best F1 score (0.01987).



You can see that for degrees of polynomials under 2 we have a bias problem (underfitting) and from there on we keep getting a lower cost. And with regularization we have no variance problem.



# Conclusion

Ookal is het niet nodig om de data random te selecteren, is het mss wel nodig om naar verhouding evenveel positieve klassen in elke onderverdeling te hebben: training heeft er 726 (normaal 688), validation heeft er 468 (normaal 517) en test heeft 526.

Future improvements: