# **Major HW3 – Final Report**

## (Q1)

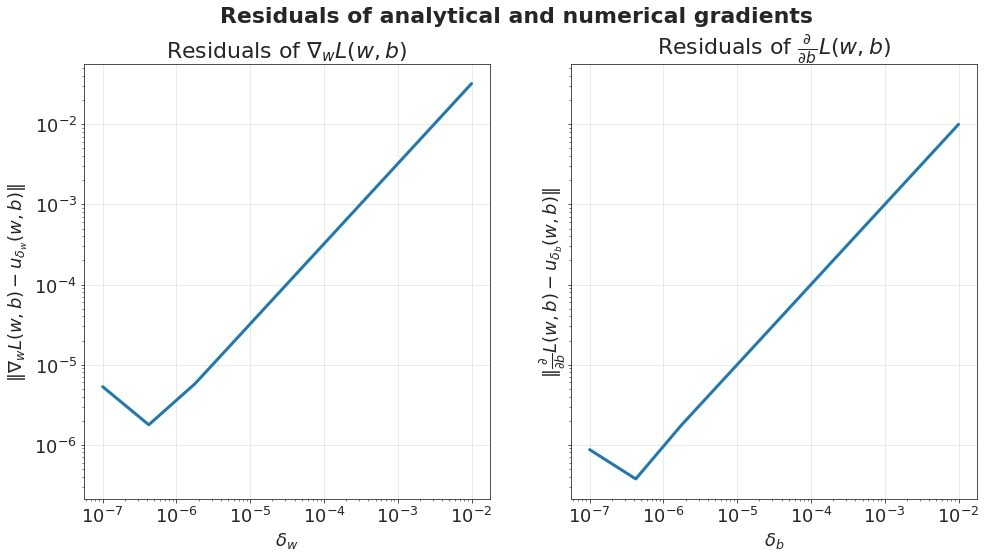
Let us observe the MSE loss:

So,

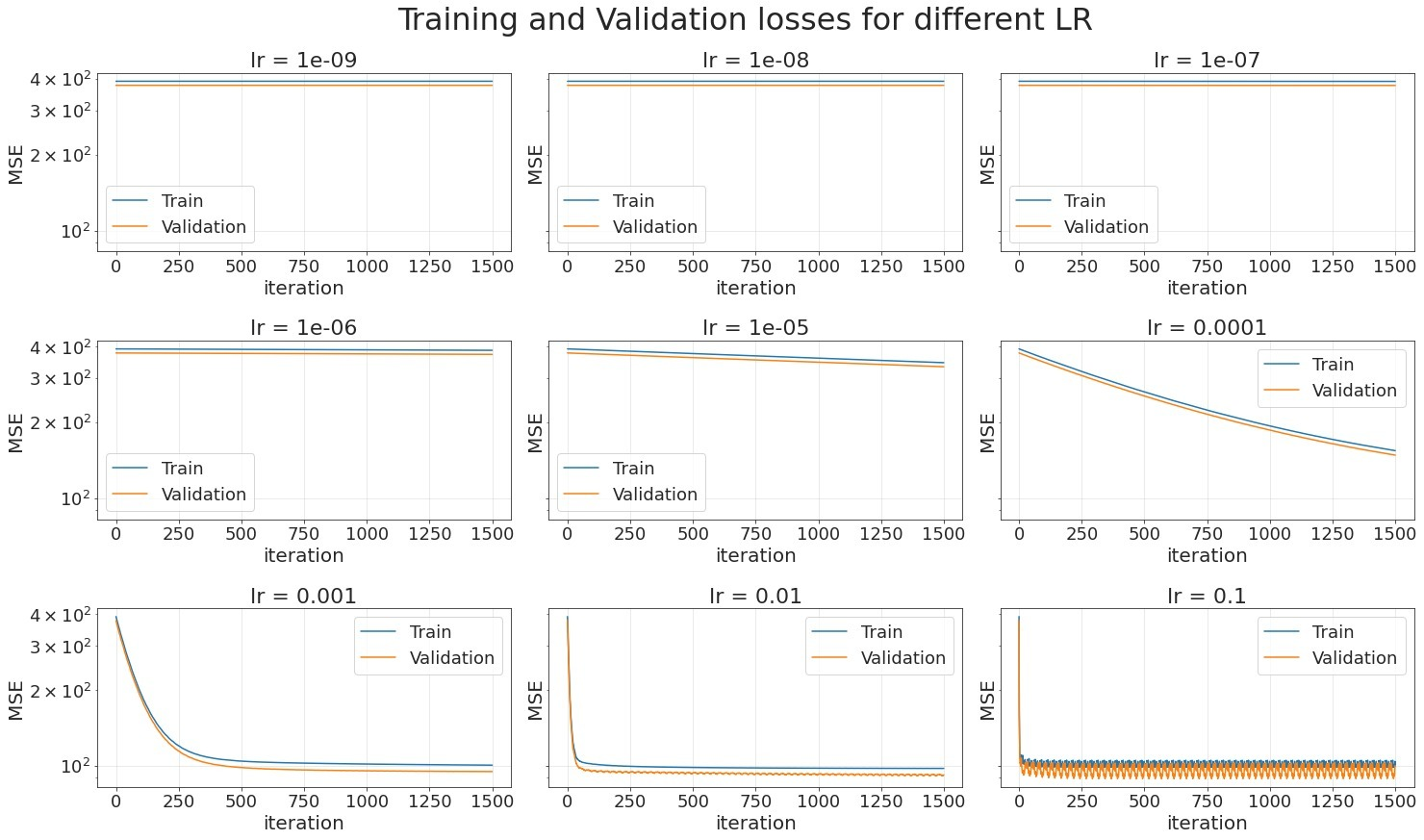
The last transition is correct since the expression means summing the entries of the vector **,** and multiplying by the scalar .

## (Q2)

The following plot shows the residuals of the numerical and analytical gradient:



## (Q3)

The following plot shows the training and validation losses as a function of iteration number, for different learning rates:

We can justify the behaviors for different learning rates:

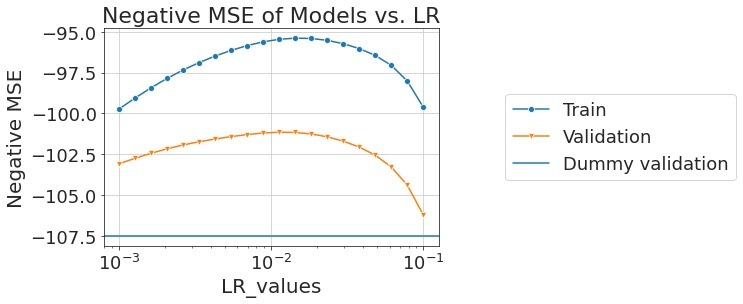
* For very small learning rates (1e-09 – 1e-06), the gradient descent algorithm barely makes progress, because the step size is very small (so it would take a very very long time to reach the minimum).
* For learning rates 1e-05, 1e-04, the rate of convergence is slightly better but still very slow.
* Learning rates 0.001, 0.01 seem best: we have fast convergence and high stability around the minimum. **The optimal learning rate is probably 0.01** since they reach virtually the same minimum point, but for 0.01 we reach it in much less iterations (however, it should be taken into account that for lr=0.01 we have small oscillations of the validation error around the minimum). **Increasing the number of steps is unnecessary** because clearly the algorithm has reached its minimum point and will not improve any further.
* For learning rate 0.1, the convergence is very fast, but the oscillations of the training and validation error are significant; this occurs since the step size is slightly too large and so the algorithm “overshoots” the minimum in every iteration.

## (Q4-5)

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Section | Train MSE | Valid MSE |
| Dummy | 2 | -107.39, -104.02, -106.93, -108.94, -108.35 | -106.69, -119.74, -108.1, -100.12, -102.81 |
| Linear | 2 | -95.66, -94.41, -94.68, -97.13, -95.45 | -101.13, -105.67, -104.84, -93.55, -100.58 |
| Lasso Linear | 3 | -96.77, -95.44, -96.08, -97.85, -96.44 | -97.51, -105.17, -100.27, -93.03, -98.12 |

The range of learning rates we chose is 20 points in logspace between and . This choice was based on the previous question: this is roughly the range in which the algorithm converged in satisfactory time to the minimum point.

The following plot shows the mean Negative MSE train and validation errors obtained from 5-fold cross validation:



As we can see, the plot behaves as expected: negative train and validation errors both rise and then fall (there is no major difference in the trend between training and validation since LR does not directly affect overfitting, only the quality of convergence of optimization). Both are better than the dummy regressor, as expected. The best validation error occurs for LR=0.0113, and the error itself is -101.15.

## (Q6)

For the dummy model, not normalizing the features has no effect whatsoever, because the prediction of the dummy regressor is the mean of the labels, regardless of the features. The labels are not normalized in any case, and so the prediction of the dummy regressor does not change.

For the linear model, assuming there are no numerical errors, the optimal training error of the model should not change, but the appropriate learning rate and number of iterations for convergence could change. The reason is, normalization of a feature is a linear transformation applied to that feature (for all samples). This is true for both standardization and min-max scaling: for standardization we have and for min-max .

Now let us observe the original data set with features, and parameters . Suppose the normalized data set is obtained from by applying a linear transformation to the ’th feature (for ), with .

So define a vector by . For all and obtained from by normalization, .

Define . Then for all we have

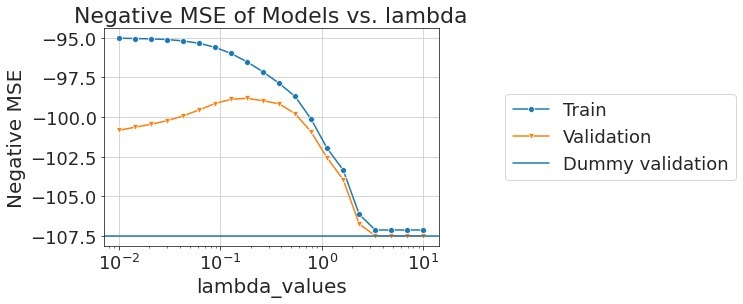
So, there is a one-to-one correspondence between parameters for and parameters for such that for every matching .

Therefore, the set of possible linear models is the same before and after normalization. This means the possible values of the training loss, and in particular the optimal value, stay the same.

Since the optimization objective (squared loss) is convex, we should reach the same minimum point, given that we chose a suitable learning rate. However, the parameter space has changed, and therefore the gradients on the graph of the function have changed, which means we might need a different learning rate (if the gradients increased, we need a smaller learning rate, and vice versa), and a different number of iterations.

## (Q7)

To tune the regularization parameter lambda, we chose (after first trying a wider range) a range of 20 points in logspace between and .



The plot behaves as expected for a tuning plot of a regularization parameter: for low values of lambda, the model is overfit and so the training error is low (negative error is high) and the validation error is high. For a lambda value of 0.183, we have optimal negative validation error of -98.82, and for high values of lambda, both training and validation errors are low since the model is underfit (in fact, the errors reach the area of the dummy model validation error).

## (Q8)

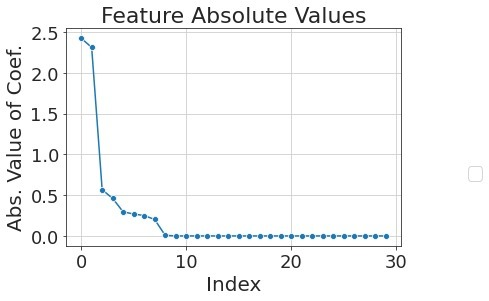
|  |  |  |  |
| --- | --- | --- | --- |
| Model | Section | Train MSE | Valid MSE |
| Dummy | 2 | -107.39, -104.02, -106.93, -108.94, -108.35 | -106.69, -119.74, -108.1, -100.12, -102.81 |
| Linear | 2 | -95.66, -94.41, -94.68, -97.13, -95.45 | -101.13, -105.67, -104.84, -93.55, -100.58 |
| Lasso Linear | 3 | -96.77, -95.44, -96.08, -97.85, -96.44 | -97.51, -105.17, -100.27, -93.03, -98.12 |

## (Q9)

The five features with the largest coefficients:

|  |  |
| --- | --- |
| Feature | Absolute value of coefficient |
| Sugar levels | 2.42 |
| PCR\_01 | 2.31 |
| PCR\_10 | 0.57 |
| blood\_type\_O | 0.46 |
| sport\_activity | 0.29 |

## (Q10)

The following plot shows the absolute value of each coefficient of the regressor:

## (Q11)

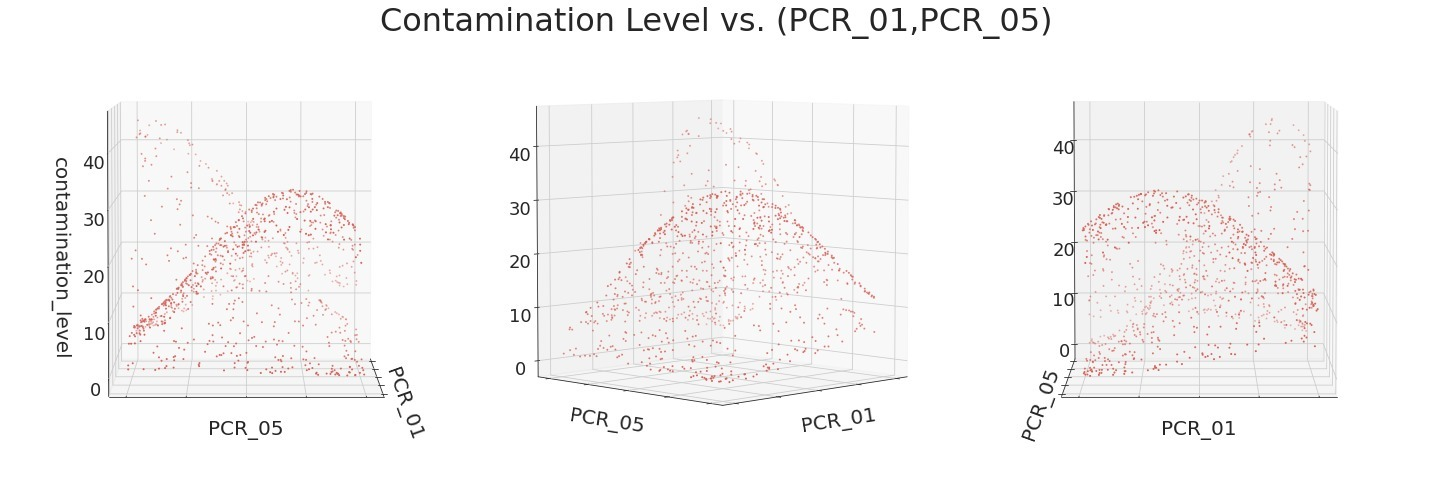
The magnitude of the coefficients is interesting for two reasons:

1. Features with high coefficients have a larger effect on the prediction , which means they are more important to the regression model.
2. In general, high coefficients indicate high model complexity, as we learned in the Lecture and Tutorial. This is why we use the norm of (whichever norm we choose) for the purpose of regularization.

## (Q12)

Yes, the training performance would change, due to the addition of the regularization component to the objective function. As we explained in Q6, normalizing a feature by a factor (neglecting the intercept for simplicity) means roughly that the corresponding coefficient for that feature is multiplied by . However, when taking into account the regularization, bigger coefficients have a higher penalty in the objective function, and therefore the optimization will prefer vectors with small coefficients, even if the coefficients deviate significantly from the coefficient resulting by multiplying by . Therefore, the algorithm will converge to parameters that define a different predictor, which results in different training performance (in contrast to Q6).

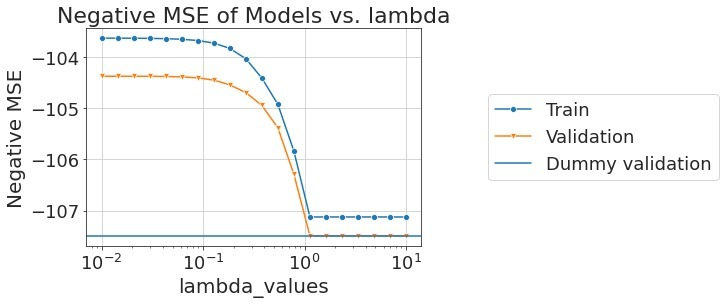
## (Q13)

The following is a visualization of Contamination Level as a function of PCR\_01, PCR\_05:

We can infer from this visualization that the dependency of *contamination\_level* on *(PCR\_01, PCR\_05)* is clearly non-linear: we may be able to estimate it with a polynomial surface, perhaps of 2nd or 3rd degree.

## (Q14)

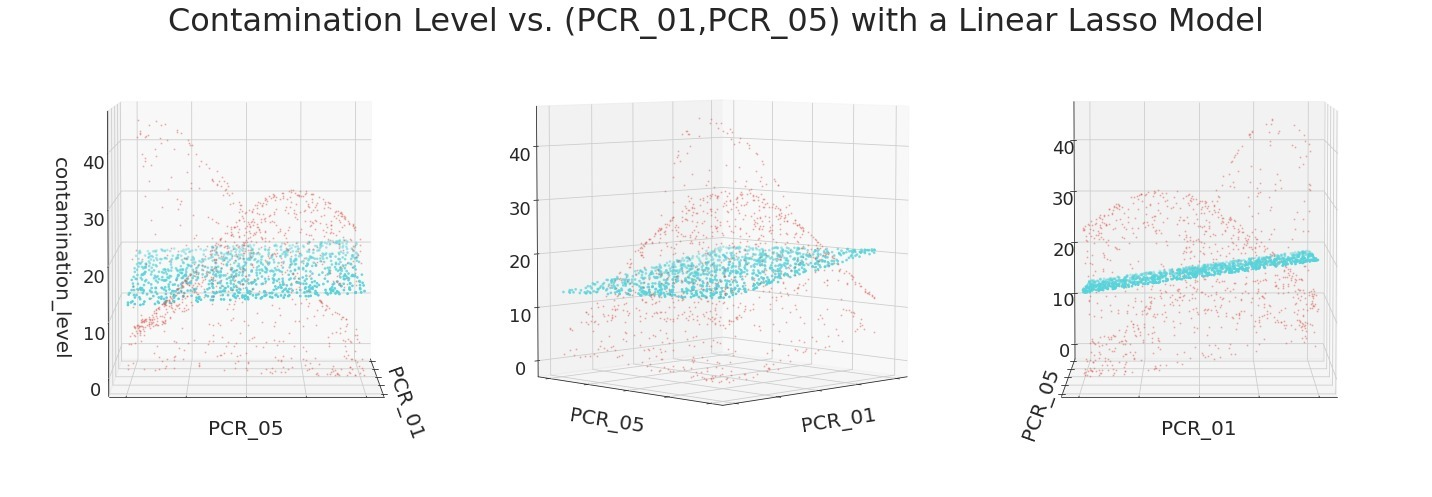
We train as a baseline a linear model for predicting of *contamination\_level* with *(PCR\_01, PCR\_05).* The following plot shows the negative MSE for train and validation, for models with different regularization parameters (lambda):



This plot is slightly irregular because there seems to be no improvement in validation error when we compare low lambda values and medium-range lambda values (we also checked lower value of lambda, the same trend continues). One explanation for this is that the linear model is so unsuitable to the data, that there is no overfitting even for small lambda values.

The optimal lambda (one of them) is 1.44e-2 and the validation score is -104.38.

## (Q15)

The following is a visualization of the true labels compared to the baseline linear model:

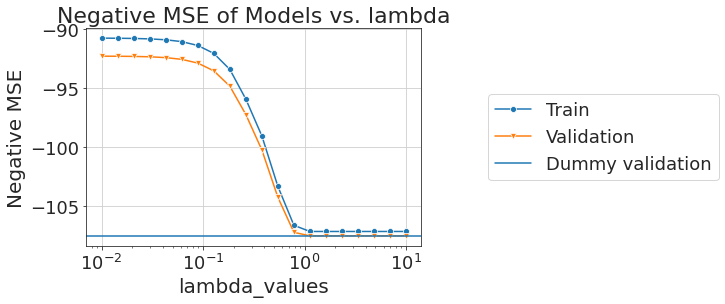
## (Q16)

Normalization after applying a polynomial mapping is important because the mapping itself ruins the existing normalization. This is because different monomials in the polynomial have different degrees: for example, in 2d polynomials we have monomials of degree 2 (), monomials of degree (,) (and even a monomial of degree ). Assuming our original features are in the range , the monomials of higher degree will have a smaller scale than monomials of degree 1. This means our new sample set (after the feature mapping), will no longer be normalized.

We know normalization is important for optimizing a linear model with SGD, because, as we saw in Lecture 8, SGD uses the same learning rate for all features. But, for features with larger scales we desire smaller learning rates, because their gradients are larger, and we to preserve a stable step size (and not diverge). This means the optimization for a non-normalized dataset may not function optimally. Therefore, a re-normalization step will allow us to improve our optimization process.

## (Q17)

We train a lasso regressor over a mapping of the data into a 2nd degree polynomial subspace to predict *contamination\_level* with *(PCR\_01, PCR\_05).* The following plot shows the negative MSE for train and validation, for models with different regularization parameters (lambda).

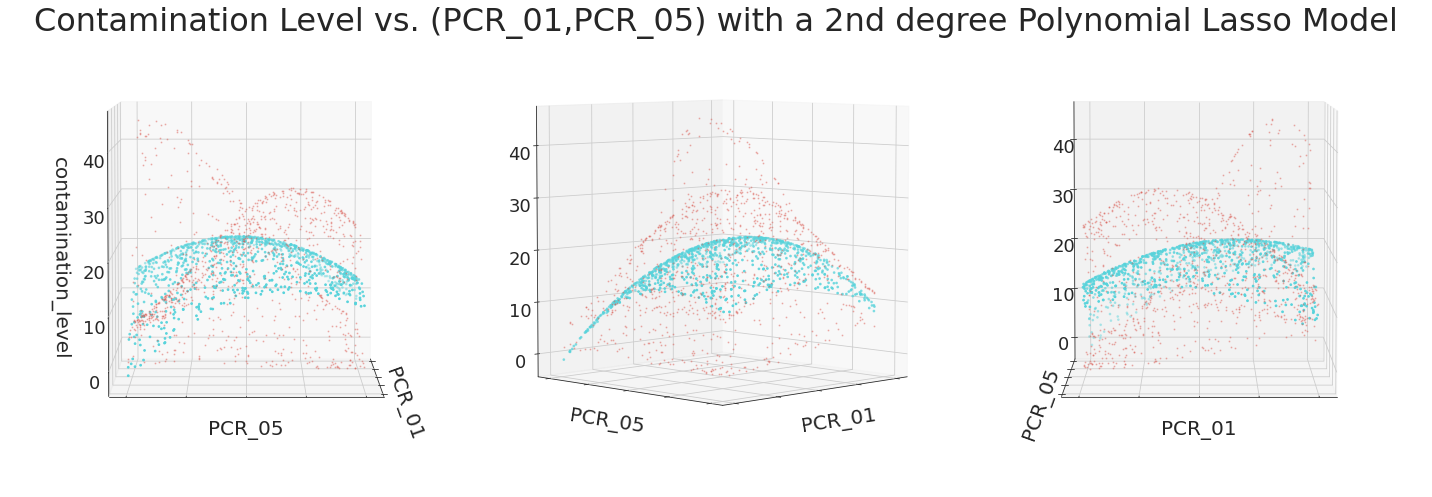


As can be seen from the plot above, for small lambda value, we have an improvement over the baseline model from Q14 of . For large lambda values (1) we see no improvement in the Negative MSE. The reason we may have an improvement is that the given data scattered in a polynomial manner, and therefore a 2nd degree polynomial may approximate it better than a linear model. The models have the same trend as the baseline models regarding the fact that there is no improvement for the Negative MSE for the train and validation datasets when we move from small to medium range lambda values (we checked also bigger range, and the same trend continues).

The optimal lambda is 0.01 and the validation score is -92.27.

## (Q18)

The following is a visualization of the true labels compared to the 2nd degree polynomial mapping with a lasso regressor:



## (Q19)

As can be seen from the Negative MSE graphs for both models, we can see that there is a noticeable improvement of for the lasso regressor model with the 2nd degree mapping over the linear model. This is because the given data is spread in a non-linear manner, which is better approximated with the models class of the 2nd degree polynomials. This could be visually seen at the 3D plots for both optimal models from each model class, as the linear model situated around the middle of the dataset, because it is the best fit it could obtain, and therefore it performs poorly over the whole training set, in contrast to the 2nd degree lasso regressor which conveniently placed between the training samples and therefore has bigger success rate (and a higher Negative MSE).

## (20)

The features we decided to transform and keep are listed below,

|  |  |  |
| --- | --- | --- |
| Feature name | Transformation | Reason |
| PCR\_01 and PCR\_05 | Polynomial Transformation with degree of 2 | We have seen throughout the assignment that this two features spread in a manner that possible to be approximated using a polynomial of degree of 2 |
| sugar\_level | Passthrough (no transformation) | We have checked the relation between contamination level and sugar level, and noticed a linear dependence, therefore we decided to keep this feature without any transformation (as it is already linear) |

We tried to keep all the features, as it has been suggested, but according to out trial-and-error checks we have noticed that whenever we use all the features the MSE over the test and validation is getting worse, and therefore concluded that it is better to remove the features that don’t contribute to the classification.

We plotted each of the features against the contamination level and noticed that most of the features has no interesting relation with the contamination level, and mostly spread uniformly, and therefore the is no reason to take them into account.

We plotted some pairs of the features together, to try and see any interesting relations between the features, but we could not identify any relations besides the listed above.

After the transformations the MSE over the validation set is . In contrast , the average MSE for the validation set prior to any transformations is .