Project-I by Group Rome

Cimen Gokcen
EPFL
gokcen.cimen@epfl.ch

Petrescu Viviana
EPFL
viviana.petrescu@epfl.ch

Angelopoulos Vasileios
EPFL
vasileios.angelopoulos@epfl.ch

Abstract

This report presents conclusions after experimenting on two tasks, classification and regression. We process the given training data, investigate baseline methods and present our results here. For the regression task all of the methods perform similarly, but we chose to present the results using ridge regression. Moreover, we applied polynomial feature transformations, resulting in possible over-fitting of the model. For the classification part, both logistic regression and its penalized version performed equally well, but the feature transformations that we tried did not improve the result even more.

1 Regression

1.1 Data Description

The training data $Xtrain$ contains 1400 observations, each 43 dimensional. One training sample has 36 real valued features and 7 categorical features. Our task is to predict the values for unseen test data, consisting in 600 samples. We measure the accuracy of our estimation using RMSE.

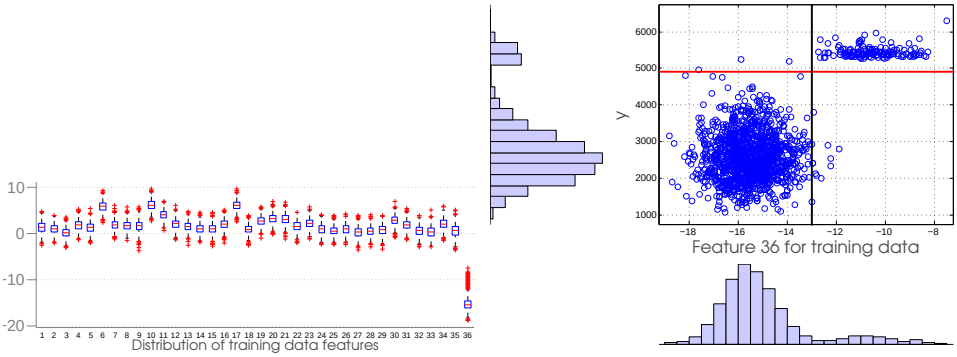
1.2 Data visualization and cleaning

Initially, our data was not centered, as seen in $Fig.$ 1a. We changed the categorical features into dummy variables, leading to a new vector of size 56. $Xtrain$ was then normalized to have 0 mean and standard deviation 1 (except the dummy variables). We applied the same operations to the test data $Xtest$ on which we will report the predicted output values.

We plotted the correlation of every feature with respect to the output and the scatter plots did not look random. We concluded the features explain the output, but we could not tell if one of them is insignificant.

The predicted values are real values in $[1000, 7000]$ and seemed to be grouped into two blobs. We believed initially the smaller blob represented outliers, but since it contained almost 10% of the data we decided not to ignore it.

After looking at every feature individually, we noticed that feature 36 offers a clear separation of the two blobs (see $Fig.$ 1b). We therefore chose to fit two models, one in which feature 36 has values greater than -13 and one in which it is smaller, corresponding to smaller predicted values. We will refer to the points grouped to the left of the black vertical line in $Fig.$ 1b as the first blob or big blob model and the points to the right as the small blob.



(a) Mean and standard deviation for the first 36 real valued variables of Xtrain. The input is not normalised and feature 36 is the only negative one.

(b) Feature 36 versus output values. X = 1.4 (black line) and y = 4900 (red line) provide a good separation of the two blobs.

Figure 1: Data visualization

We visually observed some linear correlations between certain features such as feature 2 and 24, 13 and 16, 17 and 20, but we decided to keep them since we did not have time to experiment with their removal or to test their significance. This is corroborated by the fact that $Xtrain$ is rank deficient, it has 57 columns after the use of dummy variables but rank 50.

If the input is normally distributed with mean 0 and standard deviation 1, then 99.99% of the samples appear between the values -3.891 and 3.891. We therefore remove any points that are outside this interval, considering them outliers.

1.3 Regression baseline methods

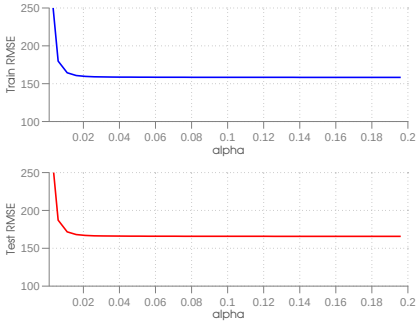
After the visualization and the analysis of the data, we applied to them the regression methods that we had implemented. More specifically, we used the least squares method (developed both using the normal equations and gradient descent) and the ridge regression method. All of them produced results of similar quality, fact that may be confirmed by the almost equal RMSE for training and test data parts, but also by the values of their resulting $beta$, which are pretty close to each other.

The following analysis, except the part of parameters computation, is referred to the results obtained from the ridge regression, but could be generalized to all of the above mentioned methods. In all of the experiments 80% of the data are used for the training part and 20% of them are used for the test part. Furthermore, K-fold cross-validation with $K = 5$ is used to obtain the error presented at this section.

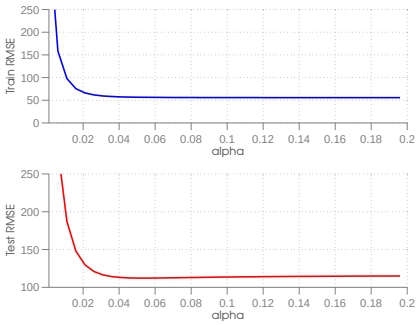
$Figures$ 2a and 2b present the results of the experiments that were held in order to find the best value of the parameter α for the least squares using gradient descent method. For this experiments 50 values of alpha that range between 10^{-3} and 0.2 were used. As we can observe, for both of our models when α has small value the RMSE that is produced is higher. The error decreases as the value of α increases up to the value 0.04. After this the performance of the method (regarding RMSE) is similar for all of the values of the parameter.

Regarding the ridge regression method we used 1000 different values for λ between 10^{-7} and 0.1. As in all of our cases 80% of the data was used for the training and the rest was reserved for the validation. $Figures$ 3a and 3b show the results, where we can see that when λ increases the RMSE grows, too. This makes us to choose a very small λ for our experiments. Moreover, the fact that our model performs better for very small values λ, makes ridge regression very close to least squares, regarding its behaviour. That is one more evidence that the two methods perform similarly to our data, as we mentioned above.

For the last part of this section's experiments we tried to study how the amount of the data that are used for training affects the resulting errors. We kept a constant amount of the data (20% of the given) for the training and we used the rest of them in proportional way for training. We started

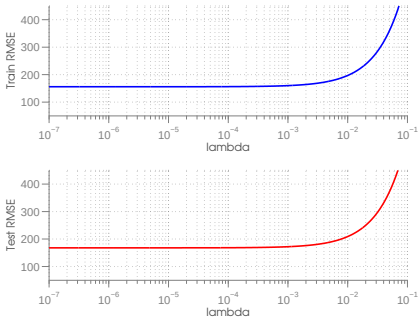


(a) Results for the big blob.

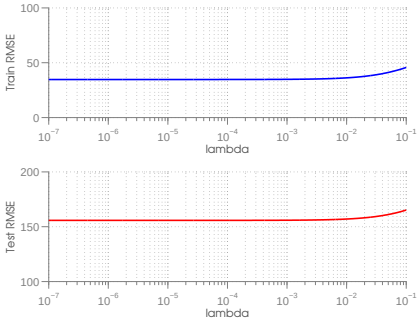


(b) Results for small blob.

Figure 2: RMSE produced for 50 values of α ranging between 10^{-3} and 0.2. Results refer to both b fitted by least squares using gradient descent.



(a) Results for the big blob.



(b) Results for small blob.

Figure 3: RMSE produced for 1000 values of λ ranging between 10^{-7} and 0.1. Results refer to both models fitted by ridge regression.

using 10% of the training part for our training and we continued up to 100%. Each experiment was held 50 times with 50 different randomly selected subgroups of data and the results shown represent the average of these experiments.

Figure 4 presents the RMSE obtained for the two models, after using each proportion of the training data. As we can see, when the proportion increases the RMSE for training and the RMSE for validation tend to some specific values.

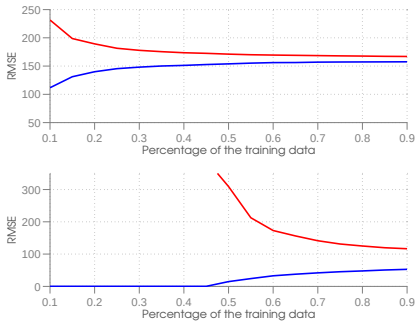


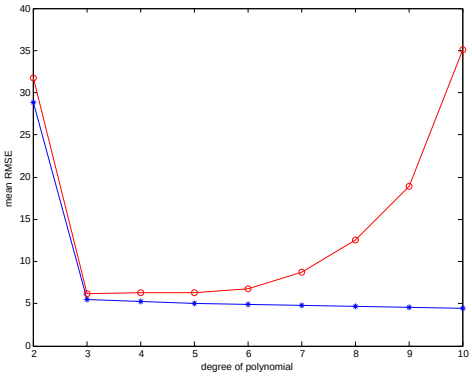
Figure 4: Learning curve produced for our model using ridge regression. The blue lines represents the train RMSE and the red the test RMSE. Moreover, the top figure represents the big blob and the bottom figure the small blob.

1.4 Feature transformations

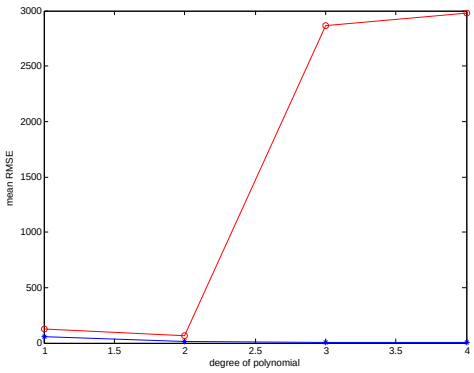
We tried polynomial and exponential transformations of the features. The exponential transformation did not prove to work well and we focused on polynomial regression. We only transformed the first 36 variables and kept the categorical variables as they are since polynomial transformation of categorical variables does not have an intuitive interpretation.

In all our experiments we used 5-fold cross-validation. The degree of the polynomial was varied from 1 to 10 for both blob-models. In Fig 5a, 5b we plotted the mean training and validation RMSE for the degree varying only between 2 - 10 and 2 - 6 respectively for readability (for the other values the RMSE was too big).

We tried both λ and α with values in the set $\{10^i | i = -5, ..., 2\}$. Using 5-fold cross validation with $\lambda = 10^{-7}, \alpha = 0.1$ we notice that a polynomial of degree 3 is the best fit for the first blob. Increasing it further leads to over-fitting, since the training errors becomes very small and the validation error increases. For the second blob model, using $\lambda = 0.001, \alpha = 0.1$ we obtain a best fit for a polynomial of degree 2.



(a) First Blob. Mean RMSE for training set (blue curve) and testing (red curve) versus polynomial degree. The errors were computed using 5-fold cross-validation.



(b) Second Blob. Mean RMSE for training set (blue curve) and testing (red curve) versus polynomial degree. The errors were computed using 5-fold cross-validation.

Figure 5: Selection of polynomial degree

We increased lambda to prevent over-fitting, since we have 10 times less samples for the second model. However, we still noticed that our RMSE for the validation set was bigger for the second blob. This might be due to the fact that we have a very small set for both training and testing of the second model.

Both polynomial regression models significantly outperform the normal ridge regression, which gave a RMSE greater than 100. Table 1.4 summarizes our best results. We notice that for the first model, our training and testing errors were quite similar and the standard deviation was quite small. For the second model, the validation error was much bigger than the train error, which is an indicate of over-fitting.

	RMSE train	RMSE test
Blob1 model	5.37 ($\pm$ 0.044)	6.25 ($\pm$ 0.4)
Blob2 model	7.16 ($\pm$ 1.85)	58.10 ($\pm$ 16.2)

Table 1: Estimated Train and Test RMSE for the two blob models.

We note that our assumption that in our training set we have a clear separation between the two blobs using the value of feature 36 is not a realistic approximation. The problem is that there are some samples with feature 36 less than -13 whose predicted value should be estimated using model 1 and some which should use model 2. This can also be seen in Fig. 1b. Because of this, we expect our error on new unseen data to be big for samples that have feature 36 value in the range $[-12, -14]$.

To test this, we split $Xtrain$ into 20% for testing and 80% for training and validation. We applied the same data processing as before, we estimated model parameters as above and the RMSE per model proved to be very similar with the ones in Table 1.4.

However, as we expected, our test error on the held out 20% got much worse. The total RMSE for our 20% test data was 295.78 (RMSE= 73.5 for blob 1, RMSE = 887.24 for blob 2 and 295.78 RMSE combined). If we remove all the samples in the held out test data which have feature 36 in the interval [-12,-14], we obtain a total RMSE = 68.62 (RMSE = 71.6 for model 1, RMSE = 82.77 for model 2). We see that the bad fit for blob 2 decreased and it is now comparable with the one from blob 1. This validates our hypothesis that high error values come from that part of the model. To sum up, we have two models that work well separately but perform bad for samples with feature 36 near the threshold value.

Our model is a piecewise polynomial and can suffer from various problems. Since we do not enforce continuity at the 36th feature, our model will generate high errors for samples that have the value of feature 36 near the threshold value, in our case -13. This happens because feature 36 does not give a clear separation of the data into two distinct models. A possible solution for further research would be to enforce smoothness at the thresholded value.

1.5 Summary

For the regression section we performed experiments on the given data using least squares and ridge regression methods. All of them have pretty similar results considering the produced RMSE. The test error estimation considering both blobs is 160.233.

The feature transformations that were introduced seemed to improve the results, decreasing the errors, but a more careful approach shows that the models that are produced very probably over-fit the data.

2 Classification

2.1 Data description

We have a two-class classification problem. The training data $Xtrain$ contains 1500 observations, each 32 dimensional. One training sample has 31 real valued features and 1 categorical feature, the 14th feature. Our task is to predict the category for unseen test data, consisting in 1500 samples. We measure the accuracy of our estimation using RMSE, $0-1loss$ and $log-loss$.

2.2 Data visualization and cleaning

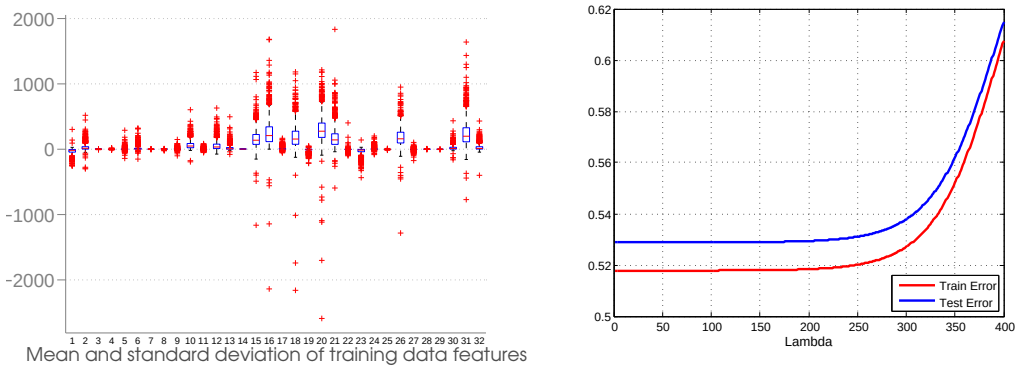
We noticed the samples were not equally distributed among classes, about 35% of the samples were coming from one class and 65% from the other. We did not have time to study the implications of this on our model estimation. We changed the labelling from -1/1 to 0/1 because our cost function was better expressed using 0 and 1.

The training data were again not centered as seen in $Fig.$ 6a After changing feature 14 to a dummy variable we normalized the data to have 0 mean and standard deviation 1. We noticed more outliers in the data of classification than in the one from regression. We removed outliers in a similar way as in the regression task which left us with 1424 samples (5% of the data were removed).

2.3 Logistic Regression

We applied Logistic Regression (logRegression) and Penalized Logistic Regression (penLogRegression) using gradient descent with learning rate α. We obtained the optimal α for logRegression and penalty factor λ for penLogisticRegression using the cross validation technique with a $50\%-50\%$ split for training and validation data.

In penLogRegression, the choice of penalty factor λ is crucial. We sampled 400 values for λs in the interval 10^{-2} to 10^{3}. As can be seen in $Fig.$ 6b, for small λ values, training error is much lower than the test error while for large λ values the test error gets as high as the training error which is a sign



(a) (b)

Figure 6: (a) Mean and standard deviation for the training data features. The input is not normalised and contains outliers. (b) Train and test error assessment with penalty factor (λ) for penalized logistic regression for 50-50 split.

of under-fitting. For small λ values, the training and test error estimated in penLogisticRegression show very similar behavior with the ones for logRegression.

The same α value that is obtained for logRegression gives also the best accuracy for penLogisticRegression. Increasing the value of α decreases all error estimations until around $\alpha = 0.8$ value which gives the best prediction accuracy for our data.

$Fig.$ 7 shows the classification accuracy percentage of logRegression and penLogRegression on the test data using 5-fold cross validation. The improvements with penLogRegression is little but might perform better on unseen data since it is more robust to outliers. Our best accuracy on the test data of 96.348% is obtained using penLogRegression with $\lambda = 0.03$ and $\alpha = 0.8$.

For our best setups, we report three types of errors using RMSE, $0-1loss$ and $log-loss$ in table 2. penLogisticRegression consistently performs better.

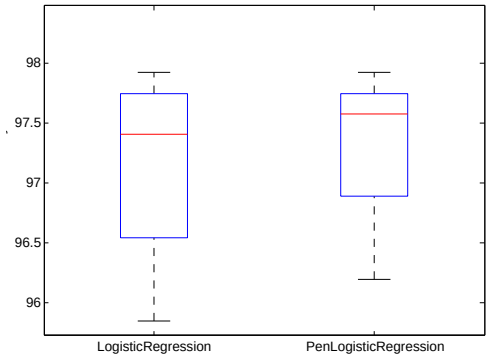


Figure 7: Comparison of logistic regression and penalized logistic regression based on validation accuracy.

Method	RMSE	0-1-Loss	Log-Loss
LogRegression	0.17	0.033	161.32
PenLogRegression	0.15	0.0267	132.34

Table 2: Best predicted error estimates for the test data.

2.4 Feature transformation

We experimented with polynomial and exponential transformation of the values but did not manage to significantly improve the previous results obtained using penLogRegression. Our best model used a polynomial of degree 2 transformation of every feature ($\alpha = 2$ and $\lambda = 0.1$) which had a recognition rate of approximately 95.77 (with standard deviation 1.056). All other models performed much worse. Since, in general, it is better to explain the correlation in the data using simpler models, we decided to report our best predictions using penLogisticRegression.

2.5 Summary

In this section, we experimented with a classification problem using logistic and penalised logistic regression. Both models gave very similar results, with best classification accuracy of 97% obtained using penLogisticRegression. Although logRegression performed very similarly, we expect penLogisticRegression to perform better on unseen data.

None of the feature transformations we tried improved the results. Using a polynomial of degree 2 for every variable performed similarly with the other models, but since the model just added complexity and no significant improvement in accuracy it was not considered relevant.

Acknowledgments

We would like to thank the course teaching assistants that helped us a lot during this project preparation, not only during the exercise session, but also at their office hours. Furthermore, the help that our teacher Mohammad Emtiyaz Khan offered us was very important, and we would want to thank him, too. All of the code used and submitted along with this report was written by equally by all of our group members.