Project-I by Group Rome

Gocken Cimen
EPFL
gcimen@epfl.ch

Viviana Petrescu
EPFL
viviana.petrescu@epfl.ch

Angelopoulos Vasileios
EPFL
vasileios.angelopoulos@epfl.ch

Abstract

We present our on two tasks, classification and regression on data that is not known. We process the data, investigate baseline methods and present our results here. For the regression task, out best model was ¡¿ and for classification was ¡¿.

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 > -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.

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 signifcance. This is corroborated by the fact that $Xtrain$ is rank deficient, it has 57 columns after the use of dummy variables but rank 50.
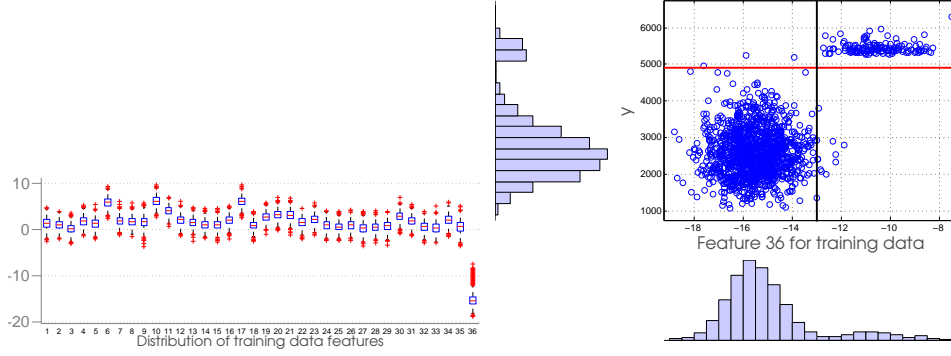
(a) Mean and standard deviation for the first 36 real valued variables of Xtrain. The input is not normalised and feature 36 is the only negative one.

(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.

If the input is normally distributed with mean 0 and std 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 Ridge regression Baseline methods

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 Fig2a,2b 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 $\lambda = 1e - 5, \alpha = 0.1$ we notice that a polynomial of degree 3 is the best fit for the first blob. Increasing it further leads to overfitting, 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.

We increased lambda to prevent overfitting, 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 ¿ 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 overfitting.

	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.

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.
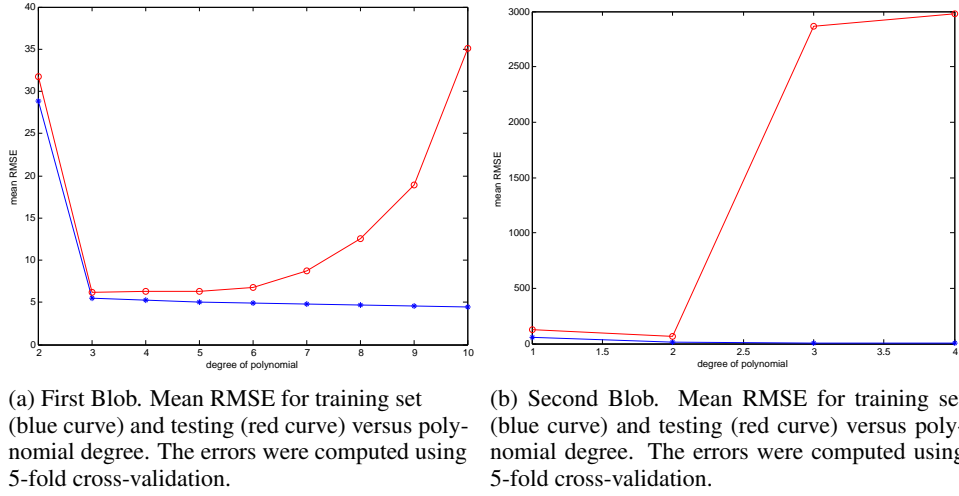
Figure 2: Selection of polynomial degree

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 < -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 splitted $Xtrain$ 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 Table1.4.

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 blob2 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 piecewiese polynomial and can suffer from various problems (In the book **?** for details on how to prevent this). Since we do not enforce continuity at the 36th feature, our model will generate high errors for samples that have the value of feature36 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. Enforcing smoothness at the thresholded value could have solved the problem, but we did not thave the time to try this.

## 1.5   Summary

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 labeling from -1/1 to 0/1 because our cost function was better expressed using 0 and 1.

The training data was again not centered as seen in $Fig.$3a After changing feature 14 to a dummy variable we normalized the data to have 0 mean and std 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).
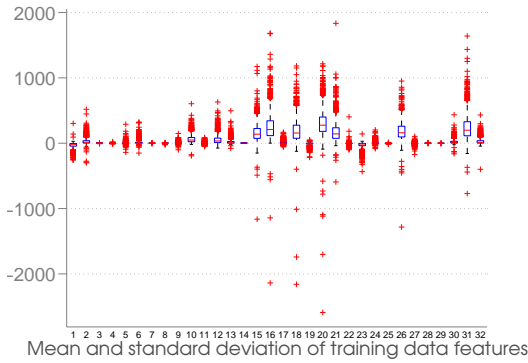
(a) Mean and standard deviation for the training data features. The input is not normalised and contains outliers.
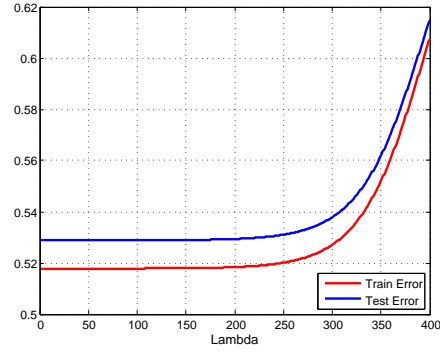
(b) Train and test error assessment with penalty factor (λ) for penalized logistic regression for 50-50 split.
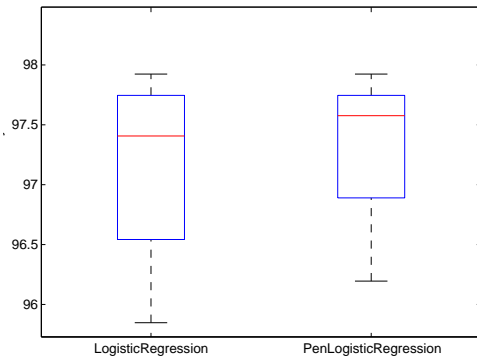
2.3 Logistic Regression

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.$3b, 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 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 also gives 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.$4 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% (std of)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 4: 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, $\alpha = 2\ \lambda = 0.1$, which had a recognition rate of approx 95.77 (with std 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 report, we experimented with a classification problem using logistic and penalised logistic regression. Both models gave very similar results, with best classification accuracy of 97% (std ..) 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.

3 Summary

Acknowledgments

References

G. James, D. Witten, T. Hastie and R. Tibshirani: An introduction to statistical learning,