# PCML: Project-I by Group BEIJING

Fei Mi
EPFL

Yumeng Hou
EPFL

November 2, 2015

## 1 Introduction

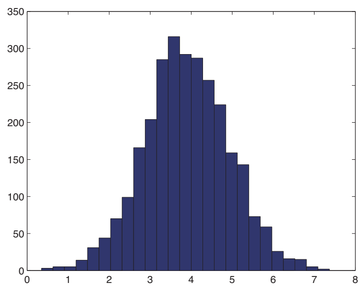
This project is about implementing and exploring regression and classification tasks on the given datasets. For the regression data set, we made use of visualization in data preprocessing to separate the data into three groups. several variations of linear regression methods with different feature transformations and manipulations are studied and applied. The evaluation is conduct by comparing RMSE of each. As for the classification task, we applied various logistic regression in an incremental manner on the transformed data and evaluate the models using 0-1 loss on validation set through cross-validation. Through our investigation of experiment results, we tried to apply the support vector machine with *rbf* kernel which slightly outperform those variations based on logistic regression.

## 2 Regression

### 2.1 Data Description

The training data for regression consists of $N_{train} = 2800$ input and output data samples. Each input sample is a vector $x_n$ with dimensionality D = 78, containing 67 real-value variables and 11 discrete-value variables.. All input samples are stored together in the matrix X_train. The output variables are scalar, and they are stored in the vector y_train.

The test dataset X_train consists of $N_{test} = 1200$ samples. The goal is to train a regression model by using the training data X_train, and then use it to produce predictions for the samples in the test dataset.


(a) 16th feature ( real-value variables)

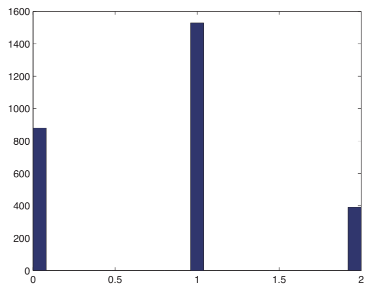

(b) 19th feature (discrete-value variables)

Figure 1: Various distributions of features in X_train.

### 2.2 Data Visualization and Preprocessing

#### 2.2.1 Data Splitting Based on Visualization

Based on the distribution of most dimensions, we guess there are possibly several data sources. When looking into the scatterplot of each dimension, we find it best to first split the dataset into subgroups according to the 41st feature.
The best boundary of separation for 41st feature is be 2.4 which minimizes the number of outliers (16). So we accordingly separate the data into 2 subgroups (Group 1 in blue and Group 2 in green) and remove the outliers as illustrated in Figure 2.

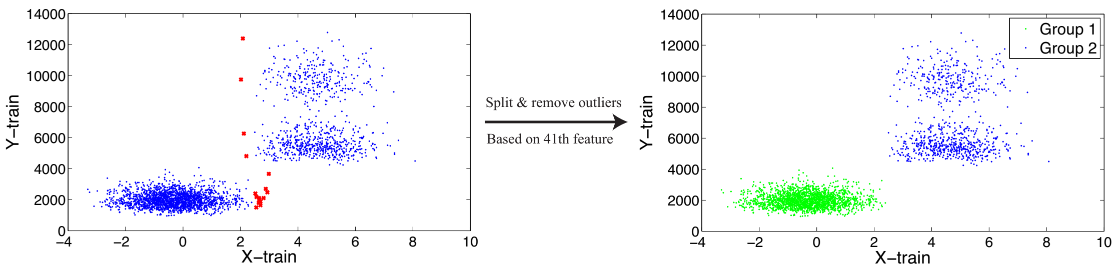


Figure 2: Separating data into Group 1 and Group 2 based on the $41^{th}$ feature.

As shown in Figure 3. Based on the visualization of both $7^{th}$ and $46^{th}$ feature, it's the case that there are two subgroups in Group 2, which comply 2 different models. Thus we got the idea that we still need to separate Group 2 into 2 groups. Therefore, we resorted to the $7^{th}$ feature, based on which we could set a new boundary to separate the data. The best boundary of separation for $7^{th}$ feature is be x =2 (as shown in 3) which minimize the number of outliers (15). So we accordingly separate Group 2 into 2 subgroups (Group 2 and Group 3) and remove the outliers by using the $7^{th}$ feature.

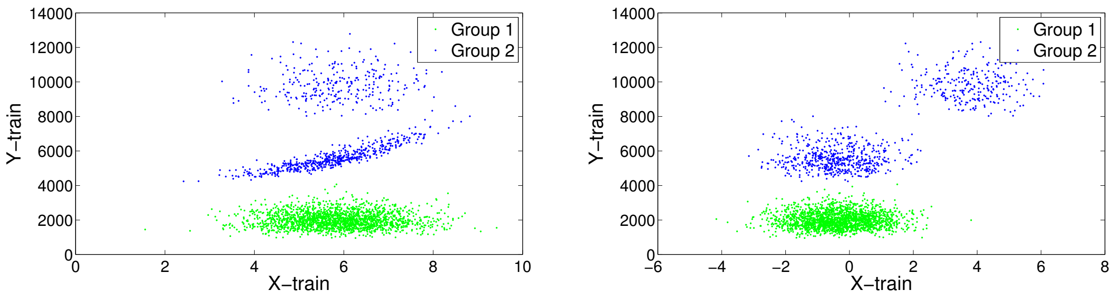


Figure 3: Visualization of $46^{th}$ (left) and $7^{th}$ (right) feature based on split data after split by $41^{th}$ feature.

After further splitting the data using the $7^{th}$ feature, we plot the results on $46^{th}$ feature, as illustrated in Figure 4, we got three clear separated subgroups (Group 1, Group 2 and Group 3). During this process, we

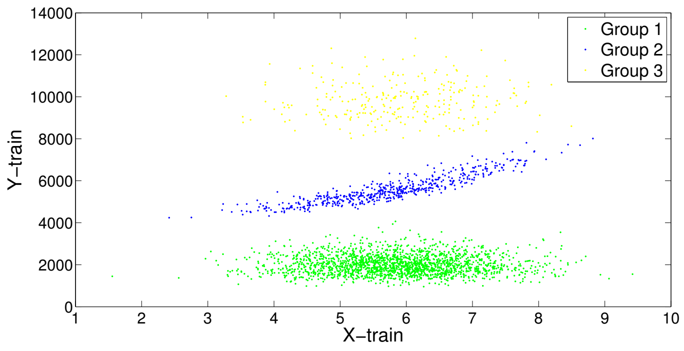


Figure 4: Visualization of $46^{th}$ feature based on re-split data.

filtered out 31 outliers in total. The filtered three groups of data are: Group 1 with 1956 vectors, Group 2 with 548 and Group 3 with 265 vectors.

### 2.2.2 Removing Outliers

After the separation process, we normalize the continuous variables of 3 split subgroups respectively and remove outliers which have at least one feature outside of the N standard deviation intervals from the mean of the corresponding feature. (N = 3 for Group 1, N = 4 for Group 2 and Group 3) During this process, we filtered out another 69 outliers. The final std-filtered version of data is: Group 1 with 1896 vectors, Group 2 with 537 and Group 3 with 265 vectors..

## 2.3 Model Exploration & Results

### 2.3.1 Experiment Setting and Evaluation Metric

For finding the best solution for the regression task we have tried several variations of linear regression methods with different feature transformations. The evaluation metric that we used to compare and test our model variations is the root-mean-squre-error (RMSE) through five-fold cross-validation on the validation set. The definition of RMSE is as follow:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2} \tag{1}$$

Figure 5 gives an overview of the RMSE on validation set for the different methods through five-fold cross validation. The hyper-parameters of the models (e.g. polynomial basis degree, lambda) were computed by applying another five-fold cross validation only on the training set.

### 2.3.2 Model Comparison

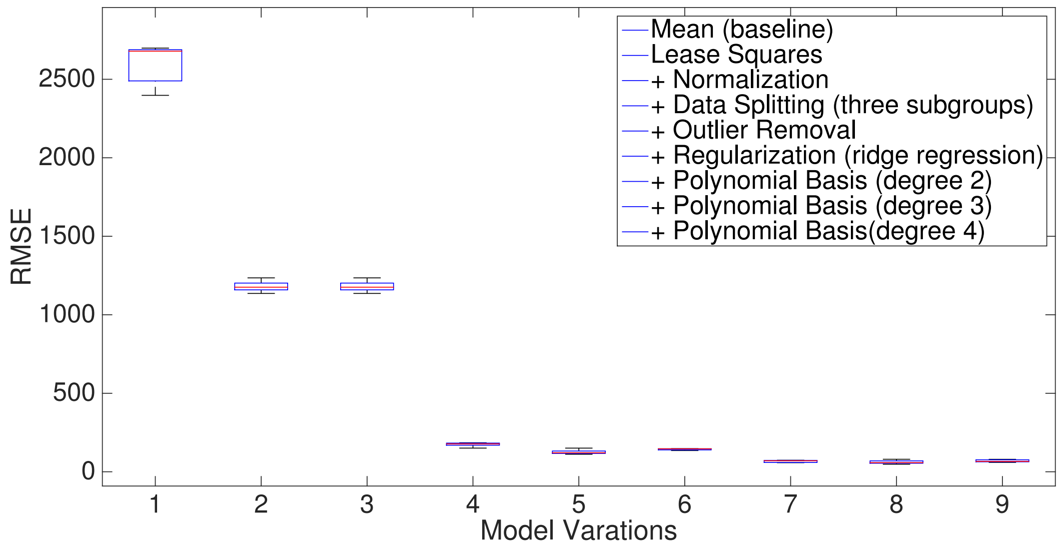


Figure 5: Model Comparison

The first algorithm in Figure 5 is considered as a naive baseline model, which is just the mean of the output variable, which does not depend on the input variables. Starting from the second model, we apply least squares. For the rest of the model variations, we incrementally add various tricks to better fulfill our prediction task, including feature normalization, data splitting, out-lier removal, regularization and adding polynomial basis.

The baseline method by just taking the mean of output is a kind of constant model, and it's a relative weak baseline as we could see from the high RMSE. Least squares significantly outperforms the baseline algorithm and achieve half of the RMSE compared with the baseline. This is expected, since we observed that the input features are correlated with the output, thus, considering the dependency between inputs and outputs starting from variation 2 will definitely help the prediction.

Comparing model variation 2 and 3, by normalizing the input variables does not provide an improvement for this dataset. Comparing model variation 3 and 4, we could see that splitting the dataset into three subgroups as we introduced in Section 2.2.1 and train three models separately on three subgroups could significantly boost the performance. Such outstanding improvement obtained thanks to our exploratory data analysis regarding the training inputs. One thing to be noted is that, the results we obtained in variation 4 after data splitting already removed part of the outliers (31) through data visualization, and the model variation 5 further removed some outliers (69) by ignoring those samples lie multiple standard deviations from the sample mean. In conclusion, we could see that out-lier removal is also an important technique to improve our model for regression task.

When applying least squares we have noticed that we have some correlated predictors. To overcome the problem of ill-conditioning we have used ridge regression. This is extremely important when we try to make use of polynomial basis. For example, when we apply polynomial basis with degree three, the number of model parameters will be $4 \times 79 = 316$ while the number of data sample in subgroup 3 is only 265, which is an extreme case of ill-conditioning with feature dimensionality greater than sample size.

We selected the optimal lambda for ridge regression (regularization) from the range $[10^{-2}, 10^{2}]$ by testing 100 values in between, and choosing the lambda which minimizes validation RMSE, by employing five-fold cross validation only on training set. The optimal lambda for three data splits are $\lambda_1 = 0.0278, \lambda_2 = 2.8818, \lambda_3 = 4.7149$,.

In algorithms 7-9 from Figure 2, we have used polynomial basis functions with degrees from 2 to 4. Such polynomial basis are applied to all three subgroups and ill-conditioning is avoided through regularization.

In conclusion, we have reached our decision that to use the model variation 7 as our final model for the prediction on testing set. Variation 7 is build on top of least squares and the most effective operation is data separation in our case, besides, out-lier removal and polynomial basis could slightly help as well with regularization ensure not falling into the case of ill-conditioning. Numerically speaking, the RMSE we obtain on validation set 61.6816, with standard deviation to be 11.89. Since not much significant improvement is obtained by add polynomial basis, we do not consider other sophisticated models for the regression task.

## 3 Classification

### 3.1 Data Description

The training data for consists of $N_{train} = 1500$ input and output data samples. Each input sample is a vector $x_n$ with dimensionality D = 33. All input samples are stored together in the matrix X_train. The output variables are observed class labels for all inputs, and they are stored in the vector y_train. Each input vector $x_n$ contains 29 real-value variables and 4 discrete-value variables. The test dataset X_test consists of another 1500 samples. The goal is to train a classification model by using the training data and then use it to produce class predictions for the test set. Test error for unseen data for our model is reported in the form of 0 -1 loss.

### 3.2 Data Visualization and Preprocessing

#### 3.2.1 Data Visualization and Transformation

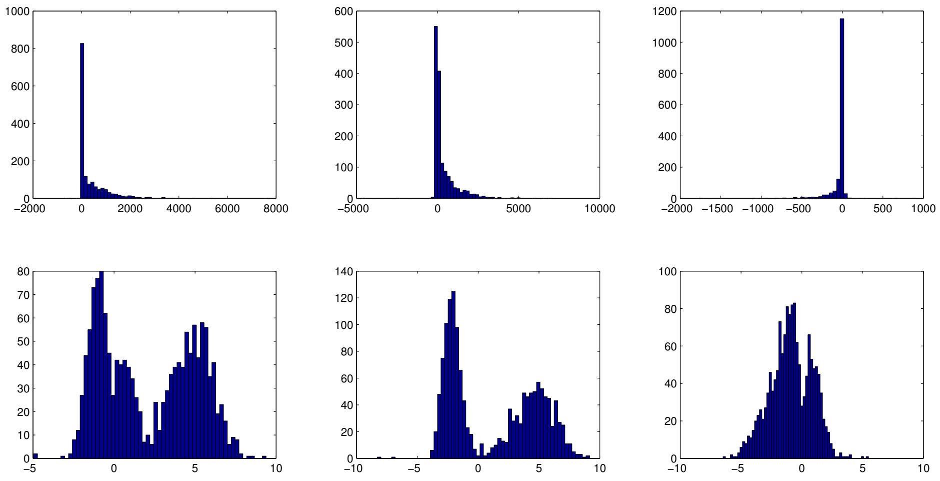


Figure 6: Visualization of original data (above) and transformed data (below) of classification.

When plotting the histogram of each dimension of X_train, we notice that most of the variables have a Poisson distribution. Thus, we transform each continuous dimension of input training matrix X_train to $\sqrt[4]{x}$ to

make the distribution closer to Gaussian. Regarding to minus zero entries, we handled the case by computing -$\sqrt[4]{-x}$
. Figure 6 shows the original distribution (above) of the first three features as well as the distribution after applying transformation (below) as comparison for a illustration.

### 3.2.2 Removing Outliers

Similarly to the regression task, we normalize the transformed continuous variables and remove outliers which have at least one feature outside of the 3 standard deviation intervals from the mean of the corresponding feature. During this process, we filtered out 23 outliers in total. The final training data set consists of 1477 vectors.

## 3.3 Model Exploration & Results

### 3.3.1 Experiment Setting and Evaluation Metric

For finding the best solution for the classification task, we have tried several variations based on logistic regression in an incremental manner. The evaluation of the classification algorithms was done similarly to the evaluation of the regression algorithms by evaluating the average validation performance through five-fold cross-validation. The evaluation metric that we used to compare and test our model variations is the expected 0-1 loss, which is defined as follow:

$$0-1 \quad loss = \frac{1}{N}\sum_{n=1}^{N} I(y_n \neq \hat{y}_n) \tag{2}$$

Figure 7 gives an overview of the 0-1 loss on validation set for the different methods. The same as regression task, the hyper-parameters of the models (e.g. polynomial basis degree, lambda) were computed by applying another five-fold cross validation only on the training set.

### 3.3.2 Model Comparison

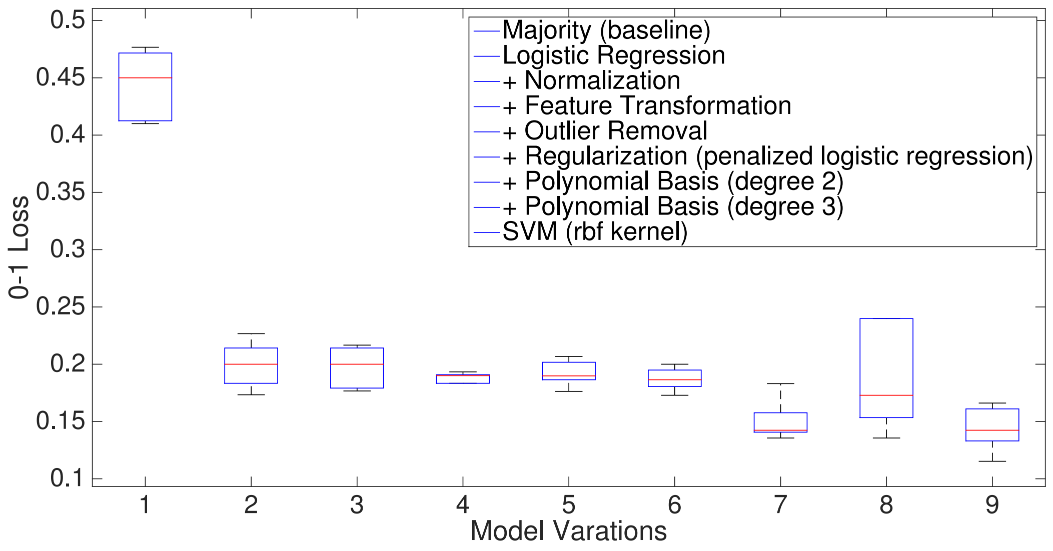


Figure 7: Model Comparison

The first algorithm in Figure ?? is considered as a naive baseline model, which simply predict all sample to the majority class. Starting from the second model variation, we apply logistic regression using gradient descent, which is a simple but effective model for classification task. For the rest of the model variations, we incrementally add various tricks to better fulfill our prediction task, including feature normalization, feature transformation, out-lier removal, regularization and adding polynomial basis.

The baseline method by predicting all samples to the majority class is a kind of constant model, and it's a extremely weak baseline as we could see from the high 0-1 loss. Logistic regression significantly outperforms the baseline algorithm and achieve roughly half of the 0-1 loss compared with the baseline. As we discussed before that logistic regression is a simple and effective model for classification task and the significant improve over baseline validate our intuition.

Comparing model variation 3 against 2, by normalizing the input variables could slightly improve the classification performance. Further investigating the difference between variation 3 and 4, we could see that using the feature transformation technique we discussed in Section 3.2.1 could boost the performance in a relative large scale and we could say that our feature transformation method is effective.

Variation 5 illustrates the performance we obtained after removing those outliers discovered in Section 3.2.2. We could see that removing those outliers did not help the classification. However, after a second though, we contend that those outliers are actually easier to classify. With this finding in mind, we look back into the performance obtained only on those "non out-lier" samples for model variation 4, and we indeed found that it does not perform as well as the version after removing those outliers. This means that variation 5 are actually doing better for those "non out-lier" samples, which are with more importance in terms of generalization value. So in the following variations, we will still remove outliers.

Variation 6 adopts the idea of regularization using penalized logistic regression. We tune the hyperparameter lambda by employing five-fold cross validation only on training set. We selected the optimal lambda for ridge regression from the range $[10^{-2}, 10^{2}]$, during the search process, we make use to a technique called *coarse to fine search*. Basically, in the search process, we first search lambda in a coarse range and gradually search in a finer range according to the validation error we observed. Through this process, the optimal lambda is 4.4. Comparing the performance of variation 5 and 6, we could see the improvement obtained by utilizing penalized logistic regression.

In algorithms 7-8 from Figure 2, we have used polynomial basis functions with degrees from 2 to 4 to achieve a nonlinear decision boundary. The significant improvement we have obtained though using polynomial basis make us to believe that this classification task in by nature a nonlinear task, that is, the decision boundary in nonlinear. And the performance drop from the degree of polynomial 2 to 3 means that we overfits the training sample with degree 3 since the model might be too complicated for our classification task.

As we saw that nonlinearity could help in our classification task, yet the polynomial basis functions are easy to overfit from degree 2 to degree 3. We researched other ways to encode nonlinearity in our model. A radial basis is a kind of band pass filter and could better deal with the case of continuous-valued input space. So we consider to try the support vector machine with rfb kernel to better handle the case of nonlinear decision boundary. Without detailed introduction about SVM, we first give the definition of rbf kernel:

$$K(\mathbf{x}, \mathbf{y}) = exp^{(-\gamma||\mathbf{x}-\mathbf{y}||^2)} \tag{3}$$

The squared exponential kernel (rbf) defines a function space that is a lot larger than that of the polynomial kernel or polynomial basis thus more flexible to handle the case of nonlinear decision boundary. There two hyperparameters of SVM with rbf kernel: penalty parameter $C$ of the error term and kernel parameter $\gamma$. We tune those two parameters using *grid search* and the optimal setting is $C = 1.8, \gamma = 0.05$. The result obtained using SVM with rbf kernel is illustrated as model variation 9, and it slightly out-performances the previous besting setting (variation 7). The implementation of SVM used a popular toolbox called libsvm.

In conclusion, the model variation 7 is the best setting for model variations based on logistic regression. Variation 7 is build on top of logistic regression and the most effective operations to achieve such performance are feature transformation and using nonlinear decision boundary. Numerically speaking, the 0-1 loss we obtain on validation set 0.1505. The SVM with rbf kernel performances slightly better than variation 7 with 0-1 loss to be 0.1444. So the final model we applied to the testing data is using SVM with rbf kernel.

## 4 Summary

In this project, we have addressed problems with one regression and one classification dataset. To solve the regression problem, we first split the data into three groups and filtered out the outliers. Then we tested a series of regression methods with different feature transformations. As for the classification dataset, we first transformed the data from Poisson-like distribution to Gaussian. Based on the transformed and filtered data, we applied various logistic regression models and one even more sophisticated model.

For both tasks, we gain experience of doing fun machine learning studies and experiments in a incremental and systematical manner.