Machine Learning – Project 1 (CS-433)

Axel Andersson, Alfred Clemedtson, Eric Dannetun
École Polytechnique Fédérale de Lausanne, Switzerland

***Abstract*—This report covers the findings of a first project in a machine learning course at EPFL. The aim is to build a binary classifier to predict whether measurements from CERN comes from Higgs bosons or not. The outcome of the project is a classifier with test accuracy 83.0%.**

I. Introduction

The aim of this project is to build a binary classifier to predict occurrences of Higgs bosons from data obtained from the particle accelerator in CERN. In the particle accelerator, protons are smashed together at great speed and in some cases collisions can produce Higgs bosons. The data set contains thirty-one features describing the "decay signature" of these collisions. Based on these features, the aim of the binary classifier is to determine whether the decay signature belongs to a Higgs boson or background noise.

In this project, we will use the standard Python library and NumPy to build machine learning models. We will only consider different variants of linear and logistic regression to build the classifier.

II. Models and Methods

The meta-algorithm we will use to solve this problem can be described by the following steps: (A) *Implement Machine Learning Methods*, (B) *Exploratory analysis*, (C) *Pre-processing*, (D) *Model Selection*. The first objective is to implement some machine learning methods. Exploratory analysis will help us understand the data set better and hopefully provide insights in how to proceed. Through exploratory analysis it will also become evident how the data should be processed so that meaningful signals could be extracted from it. When the data is processed and tidy we will begin to test different machine learning models and run diagnostics to determine how to minimize the test error and increase the model accuracy.

A. Implement Machine Learning Methods

A sub-task in this project is to implement some machine learning methods for later use. We will implement the common methods: linear regression with gradient descent (hereby abbreviated as GD) and stochastic gradient descent (hereby abbreviated as SGD), ridge regression and logistic regression with and without a regularizing term. The logistic regression methods will also use gradient descent as optimizing algorithm. A linear regression method which uses the normal equations will also be implemented. This makes a total of six different methods to choose from when building the binary classifier. Apart from choosing method, ridge and logistic ridge regression have an extra hyper-parameter which can be tuned, λ in 1. For linear ridge regression, we are choosing to minimize the mean squared error (MSE) with a regularizing term:

$$L(\boldsymbol{w}) = \sum_{n=1}^{N}(y_n - \boldsymbol{x}_n^T\boldsymbol{w})^2 + \lambda||\boldsymbol{w}||_2^2 \tag{1}$$

This is the loss function for linear ridge regression (lecture 3 [1]) which we will use. The regularization term $\lambda||\boldsymbol{w}||_2^2$ occurs in the loss function for logistic ridge regression as well.

B. Exploratory data analysis

By investigating the data, primarily by creating plots of values corresponding to features and feature-combinations we wish to gain insight in to how the features are connected as well as if they might hold. Depending on the eventual connections between features it might for example be beneficial to create different models for some subsets of the data.

C. Pre-Processing

In order to maximize, or at least enhance, the performance of our statistical models we want to make it easy to adapt a model to. We will do so by first removing outliers, and then standardizing the data in the following manner: Begin by temporarily removing all missing values, values from the data, as this would clearly impact the models in undesirable ways. For some percentage s ($> 50\%$), we thereafter look at every feature and then calculate the s:th percentile p_s and the $1-s$:th percentile p_{1-s} from all measurements of that feature. Using these values we calculate the inner percentile range, $ipr = p_s - p_{1-s}$ and the upper respectively lower bounds: $u = p_{1-s} + 1.5 \cdot ipr, \quad l = p_s - 1.5 \cdot ipr$. We set all feature-data which exceed this u to u, and all values under l are set to l. By doing so we minimize the risk of outliers affecting the models dependence on each feature. After performing this procedure on each feature we continue by standardizing the measurements of each feature, such that the mean value is 0 and the standard deviation becomes 1. The feature values that were removed at the beginning of the pre-processing are now set to the mean value of their respective feature, which in this case is zero.

D. Model Selection

As previously stated in section II-A, there are a lot of choices to be made regarding the model. We have six different methods to choose from and then there are several hyper-parameters which can be tuned to improve the test error and model accuracy. We decided to explore how the logistic ridge regression, as well as the linear ridge regression methods would perform as these are the most sophisticated models. It

is reasonable to begin with a simple version of the methods, i.e. see how well linear and logistic ridge regression performs without any feature augmentation or fine-tuning of other hyper-parameters. This provides some sense of which method to choose.

The choice of method aside, there are two more aspects which have to be selected in the model. A choice regarding the feature expansion must be made. This is a common strategy to improve a model. Some features may fit the output variable better if a polynomial expansion is done as in 2.

$$\phi_p(x) = w_0 + w_1 x + w_2 x^2 + ... + w_p x^p \tag{2}$$

As far as we know, it might be the case that every feature is optimally modelled by a different polynomial degree, p. A strategy to determine the polynomial degree for each feature is to keep all features fixed at degree, $p = 1$ while varying the degree of one feature.

The second hyper-parameter which has to be determined is λ in equation 1. This parameter penalizes complex models in favor of simpler ones. Since we are doing feature expansion, there is a risk of *overfitting* and a regularizing term can help to counter-act this behaviour.

To find the best choice of polynomial degrees, p and regularizing term, λ, a suitable method is K-fold cross-validation. The training data is split into K equally large batches, then training is done on $K - 1$ of the batches and a test is done on the one left out. This is done until every batch has been used for training and testing. By doing this, it is possible to estimate the generalisation error (mean of the K tests), of the model. We pick the polynomial degree, p and λ which has the smallest generalization error.

III. Results

Due to our exploratory data analysis we understood it was interesting to understand the nature of the measurement `PRI_jet_num` as it clearly corresponds to something that is not measured (as it is given as an integer 0, 1, 2 or 3). By grouping the data by this number and then plotting the values of the different features it is easy to see that measurements with the same `PRI_jet_num` correspond to certain features having value missing values. To easier handle this difference in relevant features it was decided to treat the subsets on their own, and thus create four different models. Even though the `PRI_jet_num`-values of 2 and 3 corresponded to the same relevant features we found that separating the two gave better model prediction. We call the data subset with `PRI_jet_num` $= 0$, X_0 and the two other groups are accordingly called X_1, X_2, X_3. To choose which model to use for each subset, we did a simple "cross validation" test of each model with some fixed hyper-parameters, which yielded the results presented in table I. The predictions which are displayed in table I, where done on X_0 and even though the logistic ridge regression performs slightly better on this simple test (and subset) we chose to use a ridge regression model for all four cases. This was because we found that this model had more room for further improvement meanwhile we were not able to improve the logistic ridge regression model much. .By systematically trying different inner percentile ranges we found that we obtained the best results using the 90:th percentile. By performing a 5-fold cross validation with a large amount of polynomial expansion degrees as well as value for λ, we obtained the smallest generalization error with a polynomial expansion degree of 12. To expand each feature individually was not found to have a greater outcome. The optimal λ values achieved were:

$$\lambda_0 = 3.82 \cdot 10^{-6}, \quad \lambda_1 = 1.23 \cdot 10^{-4}, \quad \lambda_2 = \lambda_3 = 6.88 \cdot 10^{-7} \tag{3}$$

TABLE I
A table showing the test loss and accuracy of each model on X_0 with varying regularization factor λ, polynomial expansion degree d and step size γ

Model	Ridge	Logistic ridge	Ridge
λ	10^{-3}	10^{-3}	$3.82 \cdot 10^{-6}$
Step size (γ)	-	1	-
Polynomial degree	1	1	12
Test loss	0.65 ± 0.0003	0.40 ± 0.0022	0.68 ± 0.0013
Test accuracy (%)	82.1 ± 0.3	82.4 ± 0.16	84.3 ± 0.3

IV. Discussion

The aim of this project was to build an accurate binary classifier and we were surprised that linear ridge regression performed better that logistic ridge regression. This is because the linear model is not actually modelling the probability of the output being zero or one. There is nothing constraining the linear model from outputting a value larger than one or smaller than zero which is problematic. The linear model should be very sensitive to both sparse and unbalanced training data (e.g. training on more 0:s than 1:s) due to this.

Despite this, the linear ridge regression model performs better than the logistic ridge regression model. We believe it has to do with the fact that in the linear model, it is possible to solve the normal equations directly, whereas the logistic model requires GD- och SGD-stepping. The best polynomial degree to use turned out to be 12 for linear ridge regression. When we tampered with polynomial expansion for logistic regression the model ceased to increase in model accuracy after the degree, $p = 2$. The idea of GD and SGD is to take steps in the steepest possible direction in order to reach a global (or local) minimum in the loss function. However, if the steps are too large in GD or SGD, convergence is not guaranteed and this is probably what happens with the logistic regression model. We tried to compensate the numerical instability with adapting the step size but did not succeed. If the step size is too small on the other hand, there is a risk of not reaching the optimum after an acceptable amount of time. To summarize, the linear model performs better than the logistic one because the linear model is tolerant of more data (expanding every feature to a larger polynomial degree) than the logistic model without becoming numerically unstable.

REFERENCES

[1] N. Flammarion and M. Jaggi, "Machine learning: lecture notes," 2022.