Course Project

ST451 Bayesian Machine Learning

Student 10173

5 May 2021

Contents

1. Classification - Distinguishing Signal from Cosimic Background Noise

1.1 Dataset and Problem Formulation

The *Major Atmospheric Gamma Imaging Cherenkov* (MAGIC) Telescope dataset (see UCI Repository) contains Monte Carlo generated samples with the intention to discriminate Cherenkov radiation from cosmic background noise.

Since its instruction in 2004, the signals detected by the MAGIC telescope, which is located on the Canary Islands, have allowed astrophysicists to gain insights into the nature of black holes, supernovas, and dark matter. Understanding what distinguishes signals from cosmic background noise is so crucial because it makes future discoveries more likely. Of course, this cannot be done by hand, which is why statistical models and machine learning techniques are used.

The dataset consists of ten features and a binary response variable.

Variable	Type	Description
fLength	continuous	major axis of ellipse [mm]
fWidth	continuous	minor axis of ellipse [mm]
fSize	continuous	10-log of sum of content of all pixels [in #phot]
fConc	continuous	ratio of sum of two highest pixels over fSize [ratio]
fConc1	continuous	ratio of highest pixel over fSize [ratio]
fAsym	continuous	distance from highest pixel to center, projected onto major axis [mm]
fM3Long	continuous	3rd root of third moment along major axis [mm]
fM3Trans	continuous	3rd root of third moment along minor axis [mm]
fAlpha	continuous	angle of major axis with vector to origin [deg]
fDist	continuous	distance from origin to center of ellipse [mm]
class (repsonse variable)	binary (g, h)	gamma (signal), hadron (background)

The number of signal (g) and background noise (h) events in the dataset are $12,332$ and $6,688$, respectively. In reality, event background noise constitutes the majority class.

Plot 1 below allows some insights into the relationships between a subset of the variables. Some relationships are clearly non-linear, which is a first hint that non-linear methods might be needed to achieve good results.

1.2 Methodology

This is a binary classification task. I apply the following five methods (and a null model). Note that the first two are linear methods with respect to the original features whereas the last three are non-linear.

- Logistic regression fit by maximum likelihood
- Bayesian logistic regression fit by Laplace approximation
- Logistic regression with polynomial features
- Quadratic discriminant analysis
- Gaussian process classifier

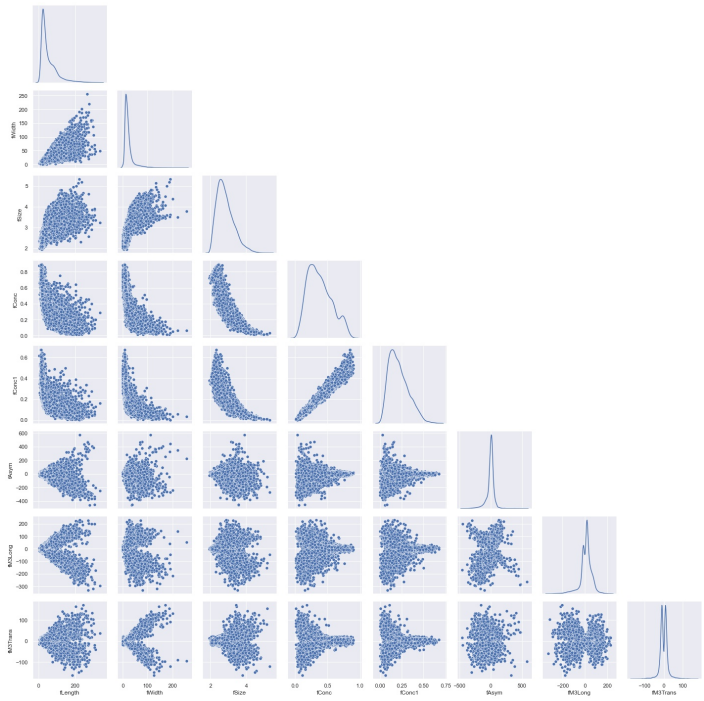

Figure 1: Plot 1: Pairsplot MAGIC Telescope Dataset (subset of all variables)

Model training. As is standard data science practice, I split the data into a training and a test set. 80 percent are used for training and the remaining 20 percent for testing. An (arguably even better) alternative would be to use cross validation. However, since the focus is on the machine learning methods themselves, I stick to the simpler train-test split.

Evaluation. First and foremost, I report the *prediction accuracy* of all methods. However, this metric is not very meaningful in this context as not all classification errors are equally bad. In particular, wrongly classifying background noise as a signal is worse than classifying a signal as background noise. For this reason, I also report the *area under the receiver operating curve* (or simply AUC), which will be my main metric for comparing the different classifiers.

1.3 Results

Null Model

I start with the **null model** (i.e. no features are used). In a classification setting, that means always predicting the majority class (which is class g). The accuracy achieved by the null model is 64.83 percent. We expect, of course, that the following methods will do better than this since they can rely on the features.

Logistic Regression - Maximum Likelihood

Even though there is no closed form solution for the MLE in the logistic regression setup, it can be found numerically by applying an iterative procedure such as the *Newton-Raphson algorithm*. The classification accuracy achieved by the MLE is 78.18 percent, which is a considerable improvement over the null model. Its AUC is 0.84.

Note that it is a linear model, so its bias is potentially large. As we will see, non-linear methods indeed outperform linear ones on this dataset.

Bayesian Logistic Regression - Laplace Approximation

The posterior distribution for logistic regression is not tractable in closed form, which is why we need to make use of an approximation. One such method is the Laplace approximation. In other words, we approximate the posterior with the normal distribution.

$$\pi(\beta|y) \rightarrow N(\beta_M, H^{-1}(\beta_M))$$

where β_M is (under some regularity conditions) the solution to $\nabla_\beta \log \pi(\beta|y) = 0$ and $H^{-1}(\beta_M)$ is the Hessian of the negative log-likelihood evaluated at β_M. Because this approximation holds only asymptotically, it is typically good for large datasets, but not necessarily for small ones. Similarly to the MLE, there is no closed from solution for β_M, but it can be found quite by applying the Newton-Raphson procedure.

The classification accuracy of this estimator is 78.18 and its AUC is 0.84, which is virtually equivalent to the results obtained by the MLE. This is no surprise as the sample size is fairly large with more than $15,000$ training observations, which is why the learned coefficients are extremely similar.

However, an important advantage of Baysian methods over frequentist ones is that they naturally allow the construction of credible intervals around point estimates. Looking more closely into these credible intervals reveals that the four variables *fWidth*, *fConc*, *fAsym*, and *fDist* are statistically insignificant. Removing them would not only result in a more interpretable model, but may also improve the prediction accuracy due to the fact that a simpler model is less prone to overfitting the training data). Like the MLE before, however, this is a linear method that is unable to model non-linearities.

Logistic Regression with Polynomial Features

As could be seen in plot 1, there exist some highly non-linear relationships between the features as well as between the features and the response variable. The two models presented above are linear methods that cannot take into account such non-linearities. Therefore, I now add polynomial features. I include only degree-2 polynomials, however. That means, every two features x and y are included as x, x^2, y, y^2, xy.

The prediction accuracy of the polynomial logistic regressing is 85.04 percent and its AUC is 0.90. The improvement over its linear counterpart stems from the increased flexibility of the model. To be even more thorough, we could test all polynomials up to a certain degree and choose the model which achieves the highest classification accuracy.

Quadratic Discriminant Analysis

Despite its name, quadratic discriminant analysis (QDA) is a generative method (all previous methods where discriminative). Because we already know that non-linear methods are better than linear ones, I only apply QDA and not LDA.

The QDA model achieves a test classification accuracy of 78.65 percent and an AUC of 0.87, which is considerably worse than the polynomial logistic regression in terms of accuracy, but quite closer in terms of AUC (see conclusion for more about this).

Gaussian Process Classifier

Unlike all previous classifiers, which are parametric methods, the Gaussian process classifier is non-parametric, which makes it very flexible. A major drawback of this classifier, however, is that it scales

cubically with the data set size, and that makes it impractical for large scale machine learning problems. As a result, I can only use a small fraction of the data as training samples (20 percent) and the remainder as test data. With respect to the kernel, I use a constant times a radial basis kernel.

Despite the small sample size, the Gaussian process classifier seems to work very well on this data set. It achieves a classification accuracy of 83.85 percent and an AUC of 0.90, which is almost on par with the polynomial logistic regression. This may be an indicator of a very non-linear decision boundary between the two classes. The radial basis kernel in particular allows very flexible decision boundaries.

1.4 Conclusion

The table below summarizes the results achieved by the six methods.

Table 2: **Summary of the Results**.
*The Gaussian process classifier was fitted on a smaller training set.

Method	Classification Accuracy (in %)	Area under the ROC
Null Model	64.83	-
Logistic Regression - MLE	78.18	0.84
Logistic Regression - Laplace Approximation	78.18	0.84
Logistic Regression with Polynomial Features	85.04	0.90
Quadratic Discriminant Analysis	78.65	0.87
Gaussian Process Classifier*	83.85	0.90

Polynomial logistic regression performed best with respect to both prediction accuracy and the more important AUC. Especially in terms of AUC, however, the Gaussian process classifier comes as a very close second. This is both surprising and unsurprising. It is surprising because it was trained on a much smaller training set, but it is not surprising as it is a highly non-liner classifier. A plot including all ROC curves can be seen below.

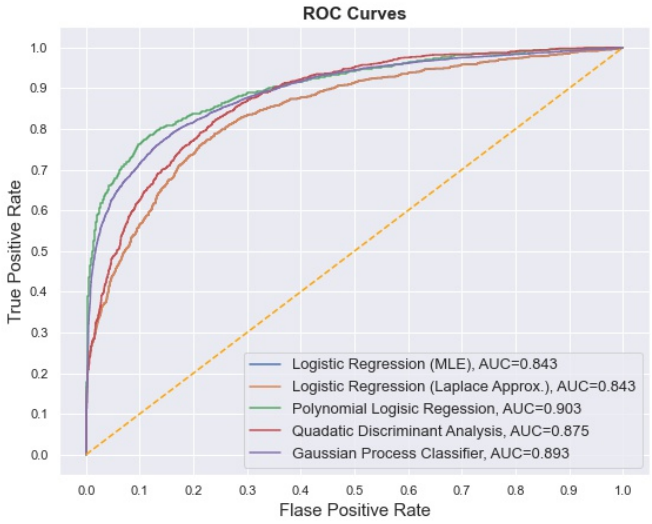

Figure 2: Plot 2: Area under the ROC for all classifiers

Intriguingly, if a very high true positive rate (i.e. sensitivity) was the main objective, the QDA would even be preferred over the polynomial logistic regression and the Gaussian process classifier, even though the latter tow have a higher AUC overall. Indeed, as already mentioned, wrongly classifying background noise as a signal is worse than classifying a signal as background noise. In other words, we care about the true positive rate of the predictions. In this sense, QDA is very competitive model and may be preferred over all other methods.

Improvements. First, because the true positive rate (sensitivity) is important, it perhaps would have been useful to report it in addition to the AUC. However, since it depends on the classification threshold used, I chose to focus on the more comprehensive AUC instead.

Second, there appear to be some very non-linear relationships between the feature variables. Therefore, higher-order polynomials, or even other basis functions, may lead to an improvement over the degree-2 logistic regression used. In addition, the performance of the Gaussian process classifier depends on the specific kernel used and there are many different options available. In fact, it would not surprise me if an optimally tuned GPC with the right kernel can outperform the degree-2 polynomial logistic regression, as it is a much more flexible method.

2. Regression - The Compressive Strength of Concrete

2.1 Dataset and Problem Formulation

Concrete is the world's most used building material. It's compressive strength depends on its ingredients as well as age. The goal is to predict the *concrete compressive strength* in megapascal (MPa) based on these variables (see UCI repository).

In fact, this problem is interesting from both a inference and prediction point of view. With respect to inference, a concrete manufacturer may want to find the best combination of ingredients for maximum strength. With respect to prediction, we may be interested in the sturdiness of the concrete used in a certain building or bridge, for example, in order to forecast if or when damages may occur.

The dataset comprises eight features and one continuous output variable.

Variable	Type	Description
Cement	continuous	kg in a m3 mixture
Blast Furnace Slag	continuous	kg in a m3 mixture
Fly Ash	continuous	kg in a m3 mixture
Water	continuous	kg in a m3 mixture
Superplasticizer	continuous	kg in a m3 mixture
Coarse Aggregate	continuous	kg in a m3 mixture
Fine Aggregate	continuous	kg in a m3 mixture
Age	*integer* $\in (1, 365)$	in days
Concrete compressive strength	continuous	in MPa (megapascal) - output Variable

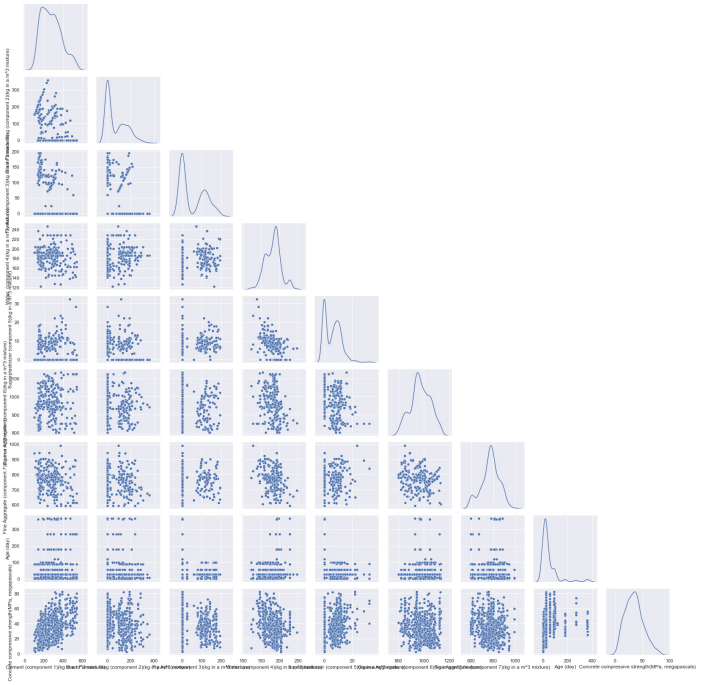

Figure 3: Plot 3: Pairsplot Concrete Dataset

2.2 Methodology

This is a regression task. The methods I am going to use are listed below. Note that only the first one is a linear method with respect to the original features.

- Linear Regression
- Polynomial Regression
- Ridge Regression (with Polynomial Features)
- Lasso Regression (with Polynomial Features)
- Gaussian Process Regression
- Sparse Bayesian Linear Regression (Horseshoe prior)

Model Training. There are 1,030 observations, of which 721 (70 percent) are used as training data and the remainder as test data. For some of the methods, I perform a model selection using cross validation over a hyperparameter-grid.

Evaluation. The mean squared error (MSE) is the main evaluation metric for regression tasks. I do not report the R-squared, for example, as it is typically more useful for inference while the main motivation in machine learning is prediction accuracy.

2.3 Results

Null Model

I start with the null model, which achieves a test mean squared error of 314.46. Of course, this is only a reference value.

Linear Regression

Linear regression is the most common statistical learning technique. The coefficients can be obtained by computing $\hat{\beta} = (X^T X)^{-1} XY$. It achieves a test MSE of 106.17, which is a considerable improvement over the null model, and a more useful baseline for the following, non-linear methods.

Polynomial Regression

A natural extension to plain linear regression is to use polynomial features in order to account for any non-linear relationships between features as well as between features and the target variable.

I use a grid search over degree-1 to degree-4 polynomials. The optimal degree is 2 with a test MSE of 61.90. Apparently, higher order polynomials overfit the training data.

Ridge Regression

An issue with more flexible models is that they are also more prone to overfitting. This is where regularization comes in. From a non-bayesian (frequentist) perspective, ridge regression is simply a $l2$ penalized linear regression. From a Bayesian perspective, ridge regression can be seen as linear regression with a Gaussian prior on β, i.e.

$$\pi(\beta) = N(0, \sigma^2\Omega_0)$$

The regularization parameter Ω_0 drives the coefficients towards 0. In other words, the stronger the regularization (the lower the variance of the prior), the smaller the coefficients and the simpler the model.

The optimal regularization parameter and degree are chosen with a grid search via cross validation. Without polynomial features (i.e. only the original features), the test MSE is 105.84, which is only a slight improvement over the plain linear regression. With polynomial features, the best model achieves a test MSE of 39.67. This model uses a regularization parameter of 0.01 and degree-4 polynomial features.

Intriguingly, this result shows that higher order polynomials are indeed useful, but sufficient regularization is required to avoid overfitting. The test MSE achieved is considerably lower than the one achieved by the degree-2 linear regression.

Lasso Regression

Lasso regression can be seen as Bayesian linear regression with a Laplace prior for β. This corresponds to $l1$-penalized linear regression in a frequentist sense. Unlike ridge regression, however, the lasso is also a variable selection method because it tends to set non-informative coefficients exactly to zero.

Without polynomial features (i.e. only the original features), the test MSE is 106.04, which is only as good as plain linear regression. With polynomial features, the best model achieves a test MSE of 39.28. This model uses a regularization parameter of $4.2 \cdot 10^{-4}$ and degree-4 polynomial features.

The results are quite similar to ridge regression, which is often (but not always) the case in practice. In contrast to ridge regression, however, the lasso also performs a variable selection, as already mentioned. In fact, more than 2/3 of the 495 coefficients are set to zero. This means that the model is much simpler simple and more interpretable than the ridge regression model, which has 495 non-zero coefficients.

Gaussian Process Regression

Gaussian process regression is a non-parametric method. There are numerous different kernels available, and the results often heavily depend on this choice. I chose a simple dot product kernel and a white kernel.

The test MSE achieved without polynomial features is 121.97, which is worse than plain linear regression. After hyperparameter optimization over degree-1 to degree-4 polynomials, the best model, which uses degree-3 polynomials, achieves a test MSE of 46.71, which is better than all linear methods but worse than polynomial lasso and ridge regression.

Sparse Bayesian Regression (Horseshoe Prior)

In order to perform Bayesian regression, I use the PyStan package. In particular, I consider the Horseshoe prior. Like lasso regression, which has been shown to be a very good model for this dataset, the Horesehoe prior also leads to a sparse coefficient vector.

The posterior of this model is not available in closed form, which is why in PyStan, the coefficients are fit using Markov Chain Monte Carlo. However, because MCMC methods are computationally very expensive, I do not run a grid search here but only use degree-4 polynomial features, which have performed best before.

The sparse Bayesian regression achieves a test MSE of 36.95. This is an improvement over the best model so far, which was the lasso. Both methods have also the advantage of producing a sparse model.

2.4 Conclusion

Most obviously, the non-linear methods clearly outperform the linear ones. This is an indication of the complex relationship between the compressive strength of concrete and its ingredients. In fact, some methods rely on degree-4 polynomials, even though sufficient regularization is required to avoid overfitting.

In the table below, I only summarize the results of the non-linear methods - except for the null model and the linear regression baseline. All other methods use higher order polynomials. The best performing method is a sparse Bayesian logistic regression with degree-4 polynomial features, followed by lasso and ridge regression (also with degree-4 polynomials).

Table 4: **Summary of the Results**.

Method	MSE
Null Model	314.46
Linear Regression	106.17
Polynomial Regression (degree-2)	61.90
Ridge Regression (degree=4, alpha=0.01)	39.67
Lasso Regression (degree=4, alpha=$4.2 \cdot 10^{-4}$)	39.28
Sparse Regression - Horseshoe prior (degree=4)	36.95
Gaussian Process Regression (degree=3)	46.71

Finally, the plot below summarizes the methods again, this time including all the linear models that were tried.

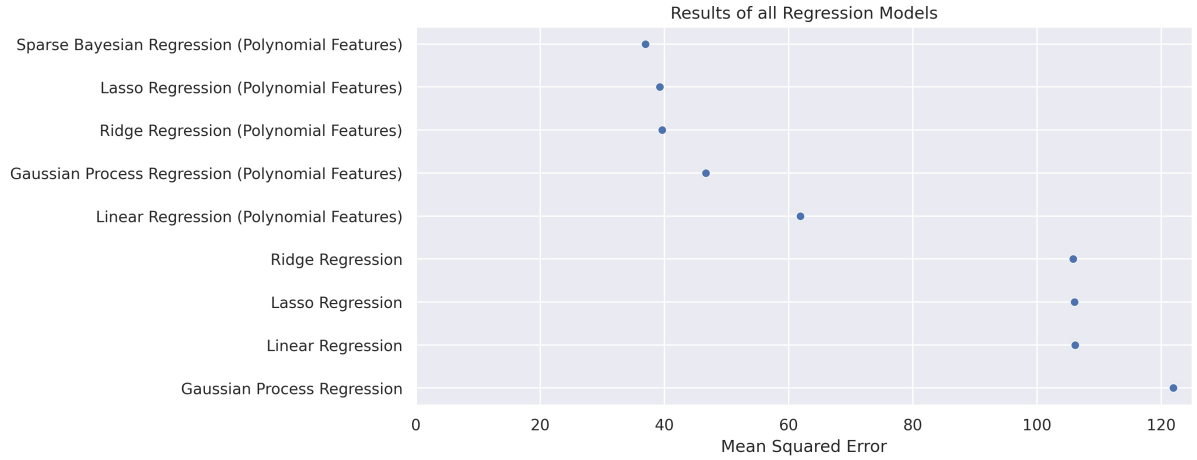

Figure 4: Plot 4: MSE of all Regression Models

Limitations. For some methods, more thorough hyperparameter tuning is necessary. Especially the Gaussian process regression has many options with respect to the different kernels available. Indeed, it is somewhat surprising that it performs considerably worse than the polynomial methods. I assume that with the right kernel, the performance gap can be minimized.

Furthermore, it would be interesting to examine the coefficients produced by the sparse Bayesian regression model in more detail. However, similar to the computer class of week 8, there have been some warnings

about the convergence of the coefficients. For point estimates, this is not necessarily concerning, but for inference, it can make a difference.