University College London

Faculty of Engineering Sciences

INSTG083/INSTM083 - Foundations of Machine Learning and Data Science

MSc Computer Science

Machine Learning - Regression Analysis

GROUP C6

Supervisors: *Email:*

Luke Dickens l.dickens@ucl.ac.uk

Authors: *Authors Email:*

Alexander Charles (alexander.charles.17@ucl.ac.uk)

Jake Connolly (jake.connolly.13@ucl.ac.uk)

Jan Wadolowski (jan.wadolowski.17@ucl.ac.uk)

23rd February 2018

Abstract

Abstract text

Contents

List of Tables

List of Figures

List of Acronyms

CS: Computer Science

UCL: Univeristy College London

1 Introduction

This report evaluates the performance of different regression models at predicting the quality of a variety of Portuguese wines. A dataset has been used which describes red wine via 11 different features, including alcohol content, fixed acidity and quality. The task has been to assess the efficacy of different machine learning models in predicting the quality of a wine given the other features.

This reports discusses four regression models for that were created to predict wine quality. This includes 3 linear basis function models polynomial, bayesian and radial basis as well as K-Nn. These models were selected to sample a selection of some of the most popular Linear regression models. By optimising each model, a fair comparison of each models efficacy was made using the same scoring for each.

1.1 Performance Evaluation

The metrics with which we have evaluated the performance of these models are as follows:

- Root mean square error (E_{RMS}): this is measure of prediction accuracy. RMSE disproportionately affects the points further from the actual results, therefore favouring predictions with small variance. This ensures that the metric is more sensitive to outlier values. This is the primary metric that we have used to asses our models. (E_{RMS}) is described by:

$$E_{RMS} = \sqrt{\frac{2E(\mathbf{w}^*)}{N}} \tag{1}$$

- Mean absolute error (E_M): this is a measure of the actual distance of the error from the prediction. This is less sensitive to large errors than E_{RMS}, but gives a clear measure of how far predictions on average deviate from the target.
- Mean absolute percentage error ($E_{M\%}$): presenting mean absolute error as a percentage can help gauge the performance of the model in a more conceptual way, helping to gauge how big the error is
- Median absolute error ($E_{\tilde{x}}$): This is the middle most error of predictions. This measure is useful as it is unaffected by outliers giving a picture of how the general accuracy of the model
- Variance (σ^2): this measure highlights the precision marked by the spread of predictions. A low variance indicates that prediction are all made in a similar region with fewer outliers.

However a model could have some issues with it's accuracy, this is why the variance needs to be used in conjunction with a mean value.

A function named `error_score.py`, was created which analysed predictions, making sure all the models were evaluated equally; enabling direct comparisons to be made.

1.2 Cross Validation

To maximise the data that we have at our disposal we train our models using k-fold cross validation. This separates the data into k separate train and test folds, which are then used by the models to predict target values. We then take the average error values of all the folds. This ensures as it enables our models to be run k-times as opposed to once.

2 kNN Regression

Table of Errors

2.1 Overview

The k-nearest neighbors regression splits data into target and training data sets. It predicts the target set by finding the average of the k-nearest values in the training set of data. It is therefore considered a 'lazy' algorithm as it does not feature a training phase and therefore doesn't generate a function which can later be tested.

The distance between two inputs is calculated using the Minkowski distance function. This function takes a parameter **p** which determines whether the function takes the Euclidian (p = 1) or the Manhattan distance (p = 2) between the input rows.

$$D(x_i, x_j) \quad = \quad \left(\sum_{l=1}^{d} |x_{il} - x_{jl}|^{1/p} \right)^{p} \tag{2}$$

2.2 Method

refer to figure **??**

2.3 Results

3 Linear Basis Polynomial Regression Model

3.1 Overview

A linear basis model is a method of supervised machine learning, that takes a series of features and assumes that a (basis) function can be applied to these features to predict a target. These

Table 3.1: Polynomial Model Error Scores

E_{RMS}	E_M	$E_{\tilde{x}}$	E_{MP}	σ^2
0.6544	0.5171	0.4353	9.2190	0.4258

basis function are summed together to create a linear combination of function, given it the *linear* name. A linear model can be generalised given the following:

$$y(x) = w_0 \quad + \quad w_1\phi_1(x_1) \quad + \quad w_2\phi_2(x_2) + ... + w_n\phi_n(x_n) \tag{3}$$

$$where \quad \phi_n(x_n) = n \text{ basis functions} \tag{4}$$

Equation **??** is a generalisation of linear regression that essentially replaces each input with a function of the input. In the case where a the identity matrix is used as the basis function, is simply just linear regression. The type of basis functions (i.e. the type of function given by ϕ) is chosen to suitably model the non-linearity in the relationship between the inputs and the target.

For this model, a polynomial basis function evaluated, that can be defined as:

$$\phi_j(x) = x_j \tag{5}$$

Through varying the degree factor a range of differing polynomial values could be evaluated. The degree factor could be optimised by looping through a range to find the factor which minimised E_{RMS}. It is worth noting that at a factor of 1, the polynomial model is actually evaluating a basic linear regression model.

As a second factor to optimise the model ridge regression was applied to the models weights. The regularisation coefficient is used to penalise weights terms with large values, smoothing out the curve of by reducing peaks.

3.2 Method

- Extract data from CSV data file
- Split data across a series of folds for cross validation phase reserve a small percentage of data to test the final validated model
- For each fold in the dataset run the polynomial training model
- Each dataset is iterated through each degree factor within a range and then each regression coefficient.

- For each degree, a matrix is created which expands out the columns of the inout matrix multiplying by the factorial of the polynomial degree. i.e. for a degree factor of 2 the matrix is expanded from x to be $1, x, x^2$.
- A regularised weight function is then used which creates a unique weight for every polynomial factor (column) in the matrix. Therefore if theres was a factor of 2 for the 11 features of the dataset than 22 different weights would be created.
- The weights values are regularised against the regularisation coefficient following the rules of regularised weights linear model. This helps smooth out the function.
- A prediction function is then defined which takes the inputs of the weights and the degree factor to setting the parameters of the linear basis function to be tested
- The new prediction function takes test data and uses this to compare it's prediction against the actual target in the data. Returning an error score and the weights matrix of the prediction.
- This process is iterated through all the folds of the data. After taking all the different cross validation results from the model; these are aggregated to produce an final error score for the model.
- To test the validated model the optimum degree factor and regression coefficient is taken and the corresponding weights array's are found across all of the folds. These weights are aggregated to create a final validated model that is then given new testing data (reserved at the start of the method)
- A final confirmation of the models performance can then be found based on training and validation phases

3.3 Results

Figure 3.1: E_{RMS} Variation with Change in Polynomial Degree, $\lambda = 0$

Figure 3.2: E_{RMS} Variation with Regression Coefficient (λ) where $Degree = 3$

3.3.1 Discussion

On first observation of the the performance of the Polynomial model, it could be witnessed that there was a significant improvement from using a polynomial basis function over a basic linear function on predicting wine quality. Figure **??** shows the change in model efficacy as the degree factor is increased, showing an parabolic shape which plateaus at a degree factor of 3 with a minimal rise at 2 and 4. When evaluating the the best models for the multiple folds, the models

best degree factor varies between these three values. The models sensitivity therefore eludes to requiring more data required generates a stable prediction across the folds.

Through holding out data in the validation phase, over fitting could be avoided due to comparing the model to data which hast been included in the training phase.

4 Bayesian Polynomial Regression Model

Table of Errors

4.1 Overview

A Baysian Model for linear regression relies on using a prior on mu MLE, overfitting MAP, no represenation of uncertinaity for weights or targets

The error must be defined for for the baysian approach as a continuous gaussian distrubtion, with a mean of 0 and constant variables.

$$y(x) = w_0 \quad + \quad w_1\phi_1(x_1) \quad + \quad w_2\phi_2(x_2) + ... + w_n\phi_n(x_n) \tag{6}$$

$$where \quad \phi_n(x_n) = n \text{ basis functions} \tag{7}$$

The posterior mean that we gain by using a bayesian approach is a way of counterbalancing the maximum likelihood achieved via non-bayesian methods and that of the prior used.

For the implementation of our bayesian model we used a polynomial model as the basis function to which it is applied. As the polynomial model was the most accurate of our non-bayesian models, it was the natural choice.

4.2 Method

refer to figure **??**

The objective of the polynomial is to produce an error score that evaluates its efficacy as a model for our data. This is achieved by passing our training and testing data for each fold into the function `polynomial_bayesian`. This uses the optimum polynomial degrees and regression coefficeint derived from the polynomial model detailed above. As with the polynomial model, the input matrix is applied to a function that takes each value takes the powers of it in the range *0 - i* where *i* is the polynomial degree value. This returns a matrix of expanded inputs.

Next we instantiate an array of prior mean wieghts, *m0* and an alpha of value 100. Following this we deduce the noise precision of covariance. This can achieved via two methods. The first is to find the variance of the raw target data, the second is to take the residuals of the *target - predicted* values. The relative merits of these two methods are discussed below.

This Bayesian Rregression is performed using a cross validation technique similar to that used in the polynomial model. However we omit the reserving of 0.1 raction of our data to perform a final test on. This would be redundant for a Bayesian model as the purpose of reserving data to test on is to avoid overfitting. This is also a primary purpose of the Bayesian approach so to properly evaluate the efficacy of both approaches we do not apply both.

Our program also returns a graph plotting the errors

4.3 Results

Figure 4.1: E_{RMS} Variation with Change in Polynomial Degree, $\lambda = 0$

Figure 4.2: E_{RMS} Variation with Regression Coefficient (λ) where $Degree = 3$

As you can see from figure **??** the number of iterations through the `calaculate posterior weights` function (where each posterior is then used as the prior in the next recursion) negatively effects the accuracy of the model. This implies that, although the accuracy of the polynomial model is improved marginally by the application of a prior, continual iterations generate weights that predict values that are farther from the actual target values. This is particularly relevant when choosing our method of deriving a precision value. The methods outlined above that is drawn from the variance of the residuals suffers as the calculating weights posterior function is called, as the predicted values it uses as a parameter become increasingly inaccurate. This d

4.3.1 Discussion

The polynomial model appeared to be a # Radial Basis Function

4.4 Classification Method For Centres

5 Evaluation

- table of results with best models Highlighted

5.1 Comparison of Models

5.2 Model Performance

- beta and alpha of priors

6 Conclusion

7 Appendices