CS342 Machine Learning: Assignment 1 Plants vs Animals

Joe Tilley, MMath 4th year, j.tilley@warwick.ac.uk, 1403899

February 11, 2018

1 Data Exploration, Feature Expansion and Feature Selection

Data Exploration: Initial observations of probeA show that the data is corrupted in some row by permuting resolutions in each compound. Hence, we write an uncorrupt function to correct this. This method works for both probeA and probeB, both by exploring the data and using an sns.pairplot analysis of probeA and probeB shows they are statistically similar. Since probeA contains data on Class and TNA, unlike probeB, we cannot use these features (directly) in our regression and classification algorithms.

Feature Expansion: We can feature expand this data by taking all possible products of each feature, i.e. monomials up to some fixed order N (we take N no larger than $N = 3$ for computational memory reasons). The discussion of whether to use or feature expand TNA in classification and class in regression or other features will be discussed in each section.

Feature Selection: Once we have expanded our dataset using feature expansion, we wish to drop some of the unimportant features which do not contribute to the our final result. We will do this by using a threshold value (a hyper parameter which we will optimise), where features determined as unimportant by a decision tree (and using the .feature_importances_ attribute) will be dropped in the classification problem, and features determined as unimportant by a ridge/lasso method (and using the .coef_ attribute) will be dropped in the regression problem.

2 Predicting TNA (Regression)

Since 'Class' is a discrete data type, we do not consider this in the continuous regression problem and drop this column entirely. Our aim now is to optimise, with respect to the mean squared error (equivalent to optimising with respect to R^2, since MSE is the normalised R^2 error) the hyper-parameters: choice of regression method (OLS/Ridge/Lasso), dimension of feature expansion (i.e. order of monomials) n, threshold value (as discussed in Feature Selection section) and the regularisation constant α. Note that during feature expansion, we also include a column of 1's, creating a bias term.

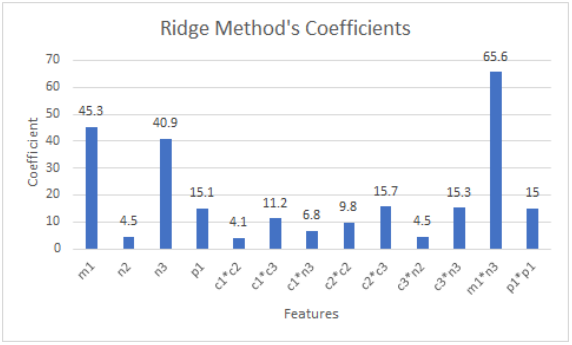

Figure 1: A graph displaying the optimised ridge method's coefficients for selected features

Using leave-one-out-cross-validation on probeA, we use a grid search method to determine the hyper parameters (with smaller increments as we narrow the ranges). Ignoring OLS (this is encapsulated in Ridge/Lasso for $\alpha = 0$), our grid search found Lasso to have optimised dimension $n = 3$, threshold $= 1$ and $\alpha = 5e - 05$ which gives MSE of 1.502. Ridge gave values $n = 2$, threshold $= 8.5$ and $\alpha = 0.0016$, giving

MSE of 1.208. Hence, we use a Ridge method with these hyper-parameters trained on the entire dataset probeA.

With this optimised method, we can determine the most important features for predicting TNA by looking at (the absolute value of) our model's .coef_ attribute (i.e. coefficients of $\mathbf{w}$), as shown in Figure 1. Any features excluded from this graph have a coefficient below the threshold, and losing such features during feature selection achieved lower errors.

3 Predicting Class (Classification)

An initial exploration of the relationship between features and class shows us that TNA is *extremely* predictive of Class compared to any of the other features, as seen via a decision tree's .feature_importances_ attribute (i.e. Gini impurity value), see Figure 2.

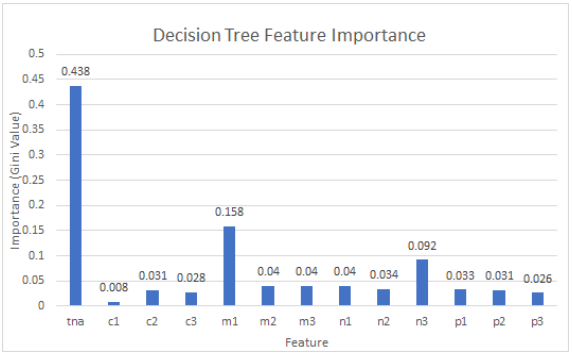

Figure 2: A graph displaying the Gini impurity for a decision tree for every feature

Whilst probeB contains no data on TNA, we could train our model on probeA including TNA values, and then use the regression method to gain predicted values of TNA for each entry in probeB. This may lead to a higher AUC than training without TNA values.

Our aim now is to optimise with respect AUC the hyper-parameters: choice of classification method (k-NN (with distance metric as sub-parameter) and Decision Tree), dimension of feature expansion (i.e. order of monomials), threshold value (as discussed in Feature Selection section) and either k in nearest neighbours or maximum depth in decision tree.

Using 100-fold cross validation (since leave-one-out cannot be used for AUC measure) and probabilistic based predictions of class (using .proba_ attribute of our models), we use a grid search method to determine the hyper parameters (of increasingly smaller increments as we narrow the ranges). Using predicted TNA values by our optimal regression method in the previous section and training with known TNA values, this method found Decision Tree to have optimised dimension $n = 1$, threshold $= 0.069$, max depth $= 7$ which gives an AUC $= 0.7491$. k-NN gave values $n = 1$, threshold $= 0.038$, $k = 32$ with Minkowski-2 metric (i.e. Euclidean) gave an AUC of 0.8022. Without predicted TNA values and training sans TNA gave smaller AUC for both k-NN and decision tree methods. Hence, we use the k-NN with predicted TNA values (via optimised regression) with these optimal hyper-parameters trained on the entire dataset probeA.

Running a simple decision tree on our feature-selected dataframe, we can find the most predictive features of class by the tree's .feature_importances_ attribute displaying Gini impurity value (i.e. alternative measure of entropy gain), see Figure 3. Any features excluded from this graph had a Gini impurity value below the threshold, and losing such features during feature selection achieved higher AUC. Training with TNA values on probeA and using predicted TNA values for probeB only slightly improved our AUC. This may not be too surprising following GIGO principles; predicted TNA values can be considered as an extremely high order feature expansion.

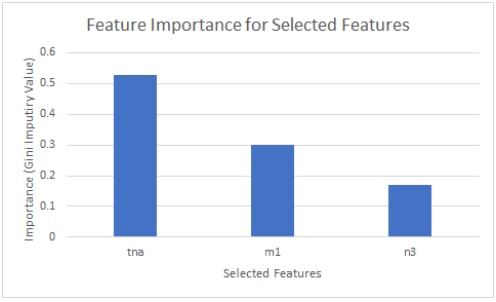

Figure 3: A graph displaying the Gini impurity for a decision tree for optimised selected features