Methods to predict Drug Sensitivity

Team: Akhan Ismailov

Abstract

This report will go through findings and methods that gave MSE of 3.171 in Kaggle drug sensitivity-2 contest. It will show ideas on how to deal with sparsity of feature matrix. How to carefully divide training set into 3 parts for 3 fold cross validation in condition of relatively small sample size. Will show benefits of recursive variable selection for preventing overfitting and improving generalization. Will present results of regression methods on selected variables and result of k-nearest neighbors regression with motivation for using it.

1 Dataset Description

Y is real valued output logarithmic IC50, X is binary features of the mutation status of 60 cancer genes. Training sample size 542, prediction sample size 100. Number of features is 60.

2 Sparse feature matrix

Our training X matrix is 542 by 60, where only 4% of entries are filled with ones, everything else are zeros. Some features are very rare that appear only once or twice, total number of features that appear less than 5 times is 23 out of 60 possible features. This may negatively affect on some models' generalization on a test set. During fitting linear model and solving coefficient on rare feature, that coefficient cannot be certainly determined due to possible noise affecting small number of samples containing rare feature. In addition, there are features that do not appear in test set at all, 16 out of 60. However, if you continue using those 16 features to fit training set, coefficients that helped to fit the training data will not improve results on the test set, which will cause optimistically misleading error results during training and cross validation. For these reasons, we will remove columns (features) that have small number of filled values, cutting them at some threshold, where value of threshold will be verified during cross validation time. I saw a noticeable improvement on the performance of my linear models after introducing this threshold.

3 Partition of the training data

During my different partitions of training data into 3 parts for 3 fold cross validation. Where we use one part as cross validation and other two parts as training set, repeating this three times and averaging errors. I noticed that the performance of model could vary very much during every repetition, and error for each part was related to the variance of that part. This could be explained by if one predicts mean value for each y, MSE will be equal to the variance of y. So it is important to divide data so that each part have equal variance and preferably mean values, especially when dealing with small sample size, where these statistics could vary noticeably from partition to partition. We do it by looping through different values of seed for random shuffling function and finding seed with minimum sum of pairwise absolute difference of variance and mean between each part.

4 Recursive variable selection

Selecting best subset of features that generalize well is essential for performance. When model use all possible features it can fit the training data well, but generalization on new unseen data will be poor. To prevent this we use recursive variable selection with particular method (linear regression, lasso). Recursive selection repeatedly adds feature to the model that decrease cross validation error or remove feature that does not improve result until the best subset is found.

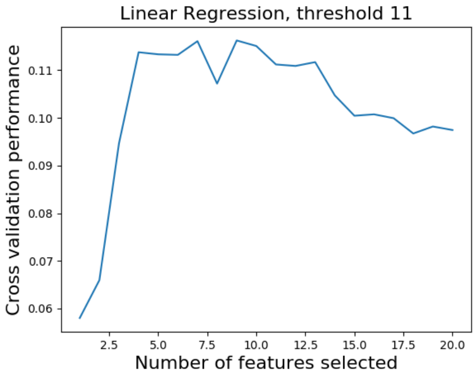

Figure 1: Linear Regression

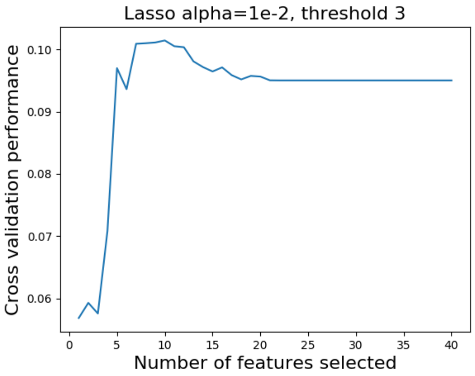

Figure 2: Lasso $\alpha = 1e - 2$

Recursive variable selection with linear regression selects 9 features, while lasso selects 10. They both have different threshold values (see section 2), which was choosen in the next steps of the pipeline by looping through possible values and only best threshold plots are shown here. These two subsets share 7 common features. At the end we will try both subsets on 3 fold cross validation and will see scores on both of them. As can be seen during recursive variable selection using linear regression and lasso particular subset of possible features generalize better comparatively to using all features and gives better performance on cross validation set. You may see that in linear regression x axis is from 1 to 20 features and in lasso from 1 to 40, not to total 60. This is due to thresholding that was described in section 2. Also can be noticed that the best threshold chosen for lasso selection is lower, which leaves more features for recursive selection. I explain it due to intrinsic better generalzation ability of lasso given more variables that arises from shrinkage.

5 Regression Methods

After selecting 9 features with linear regression and 10 features with lasso. We reduced our problem to regression with small number of features, that can be tried to be fitted by many regression methods. Table 1 is a table of regression methods' performance on 3 fold cross validation.

Where baseline is a default solution where all y_cross_val values are predicted as a mean value of all y_train. Same baseline solution exist on Kaggle leaderboard.

From above models three models were chosen for prediction on Kaggle. Linear regression using subset 1 achieved MSE 3.17638, using subset 2 achieved MSE 3.17124. Ridge regression using subset 2 achieved MSE 3.17183.

Table 1: Mean Square Error

Subset/ Method	Linear Regression	Lasso $\alpha = 0.01$	Ridge $\alpha = 1$	SVR	Decision Tree Regressor	Random Forest Regressor	Baseline
Recursive variable selection with linear regression, Threshold 11, subset 1	2.83	2.82	2.82	3.17	2.92	2.92	3.22
Recursive variable selection with Lasso, Threshold 3, subset 2	2.77	2.8	2.76	3.15	2.83	2.79	3.22

6 K-nearest neighbors

After performing described above pipeline and looking at the prediction of y_test on kaggle. I noticed that 67 out of 100 values were filled with the same value in linear regression model using subset 1, which equals to the constant part of a linear regression. This is because selected 9 features are all equal to zero in 67 cases out of 100 in the test set. This arises from the fact that feature matrix is sparse. However, selecting features is important and can be seen from the plots above that selecting more features makes generalization worse. This reason motivates approach that is conceptually similar to k-nearest neighbors, which will be used to predict 67 variables with motivation that a specific approach will perform better on predicting 67 variables than setting the value for all of them being equal to a constant. K-nearest neighbors set average value of k nearest neighbors in feature space to predict a new value, where distance is euclidean. In this approach, I decided to use cosine value as a measure of similarity of features. This motivated from the fact that if two set of samples does not have simultaneously ones in any column, it will give a similarity score of zero, which is the smallest possible score in given binary features. While euclidean distance in this case will still give some score which is not significantly differ from almost similar samples. As for nearest neighbors part, I decided to use average of all samples which cosine value is not smaller than some threshold rather than specifying number of neighbors, this decision was made from testing different approaches on cross validation. Conceptually, we consider two samples as neighbors if they share considerable part of ones together. It worths noticing that this method quite resistant to the fact that feature matrix is sparse, the cosine can be measured and still be meaningfull for prediction even for vectors that have only few filled entries. This was the main motivation for using this method. Eventually, this method did not improve result and gave MSE of 3.212. Despite that after ensembling this method to linear regression number of values that does not equal to a constant decreased to 24 out of 100.

7 Conclusion / Future work

In this report, ideas for dealing with sparse feature matrix were introduced. Careful way of dividing training data into different parts for cross validation was explained. Benefits of using recursive variable selection was showed. Finally, results of regression methods on selected variables and intuition for using k-nearest neighbors were presented. In the future, regression methods that carry meaningful intuition for dealing with a rareness of features, such as k-nearest neighbors regression should be studied.