Subjective Question & solutions

Assignment- Advanced Regression

Question 1

Rahul built a logistic regression model with a training accuracy of 97% and a test accuracy of 48%. What could be the reason for the gap between the test and train accuracies, and how can this problem be solved?

Answer 1:

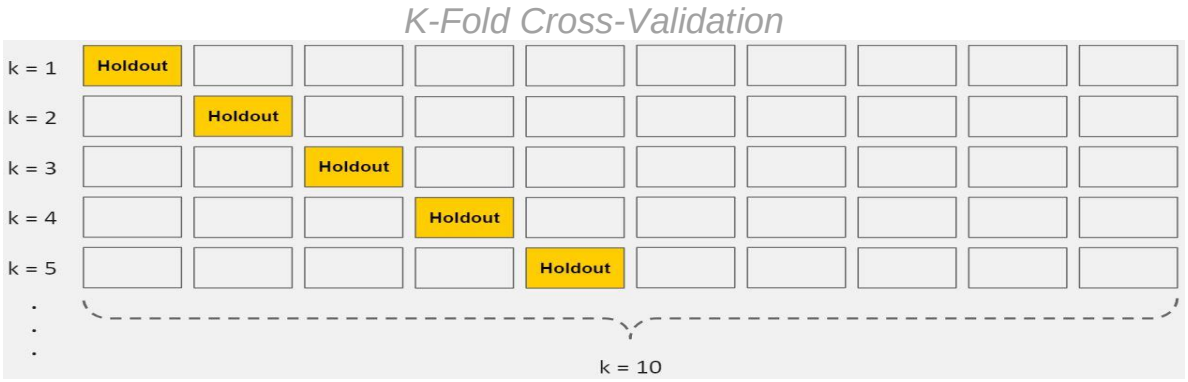
When training accuracy is higher than the testing accuracy it means we have an **overfit model**. In essence, the model has learned the particulars that help it to perform better in your training data which are not applicable to the larger data population and therefore result in worse performance.
The problem can be solved using cross validation. Its' purpose is to help avoid over fitting our models.

Cross-validation

Cross-validation is a powerful preventative measure against overfitting.

The idea is use your initial training data to generate multiple mini train-test splits. Use these splits to tune your model.

In standard k-fold cross-validation, we partition the data into k subsets, called folds. Then, we iteratively train the algorithm on k-1 folds while using the remaining fold as the test set (called the "holdout fold").



Cross-Validation Step-by-Step

These are the steps for selecting hyperparameters using 10-fold cross-validation:

1. Split your training data into 10 equal parts, or "folds."
2. From all sets of hyperparameters you wish to consider, choose a *set of hyperparameters.*
3. Train your model with that *set of hyperparameters* on the first 9 folds.
4. Evaluate it on the 10th fold, or the"hold-out" fold.
5. Repeat steps (3) and (4) 10 times with the same *set of hyperparameters*, each time holding out a different fold.
6. Aggregate the performance across all 10 folds. This is your performance metric for the *set of hyperparameters*.
7. Repeat steps (2) to (6) for all sets of hyperparameters you wish to consider.

When the data for research is limited, cross-validation methods are highly recommended.

Cross-validation allows you to tune hyperparameters with only your original training set. This allows you to keep your test set as a truly unseen dataset for selecting your final model.

Question 2

List at least four differences in detail between L1 and L2 regularisation in regression.

Answer 2:

Regularization

Regularization helps to solve over fitting problem in machine learning. Simple model will be a very poor generalization of data. At the same time, complex model may not perform well in test data due to over fitting. We need to choose the right model in between simple and complex model. Regularization helps to choose preferred model complexity, so that model is better at predicting. Regularization is nothing but adding a penalty term to the objective function and control the model complexity using that penalty term. It can be used for many machine learning algorithms.

A regression model that uses L1 regularization technique is called ***Lasso Regression*** and model which uses L2 is called ***Ridge Regression***. Differences between these two are:

1.Penalty term.

Ridge regression adds "*squared magnitude*" of coefficient as penalty term to the loss function. Here the *highlighted* part represents L2 regularization element.

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

Cost function

Here, if *lambda* is zero then you can imagine we get back OLS. However, if *lambda* is very large then it will add too much weight and it will lead to under-fitting. Having said that it's important how *lambda* is chosen. This technique works very well to avoid over-fitting issue.

Lasso Regression (Least Absolute Shrinkage and Selection Operator) adds "*absolute value of magnitude*" of coefficient as penalty term to the loss function.

$$\sum_{i=1}^{n}(Y_i - \sum_{j=1}^{p} X_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

Cost function

Again, if *lambda* is zero then we will get back OLS whereas very large value will make coefficients zero hence it will under-fit.

2.**Feature Selection**: Another **key difference** between these techniques is that Lasso shrinks the less important feature's coefficient to zero thus, removing some feature altogether. So, this works well for **feature selection** in case we have a huge number of features.

3. **Computational efficiency.** L1-norm does not have an analytical solution, but L2-norm does. This allows the L2-norm solutions to be calculated computationally efficiently. However, L1-norm solutions does have the sparsity properties which allows it to be used along with sparse algorithms, which makes the calculation more computationally efficient.

4. **Sparsity** refers to that only very few entries in a matrix (or vector) is non-zero. L1-norm has the property of producing many coefficients with zero values or very small values with few large coefficients.

Traditional methods like cross-validation, stepwise regression to handle overfitting and perform feature selection work well with a small set of features but these techniques are a great alternative when we are dealing with a large set of features.

Question 3

Consider two linear models:

L1: y = 39.76x + 32.648628

And

L2: y = 43.2x + 19.8

Given the fact that both the models perform equally well on the test data set, which one would you prefer and why?

Answer 3:

While creating the best model for any problem statement, we end up choosing from a set of models which would give us the least test error. Hence, the test error, and not only the training error, needs to be estimated in order to select the best model. This can be done in the following two ways.

1. Use metrics which take into account both model fit and simplicity. They penalise the model for being too complex (i.e. for overfitting), and thus are more representative of the unseen 'test error'. Some examples of such metrics are Mallow's Cp, Adjusted R squared, AIC and BIC.

2. Estimate the test error via a validation set or a cross-validation approach. In validation set approach, we find the test error by training the model on training set and fitting on an unseen validation set while in n-fold cross-validation approach, we take the mean of errors generated by training the model on all folds except the kth fold and testing the model on the kth fold where k varies from 1 to n. Let's look into these one by one:

1. Mallow's Cp

2. AIC (Akaike information criterion)

3. BIC (Bayesian information criterion)

4. Adjusted R squared.

AIC and **BIC** are defined for models fit by maximum likelihood estimator. We can notice that as we increase the number of predictors d, the penalty term in Cp, AIC and BIC all increase while the RSS decreases. Hence, lower the value of Cp, AIC and BIC, better is the fit of the model. Higher the Adjusted R squared , better is the fit of the model.

Question 4

How can you make sure that a model is robust and generalisable? What are the implications of the same for the accuracy of the model and why?

Answer 4: While creating the best model for any problem statement, we end up choosing from a set of models that would give us the least test error. Hence, the test error, and not only the training error, needs to be estimated in order to select the best model. This can be done in the following two ways:

Use metrics that take into account both the model fit and its simplicity. They penalise the model for being too complex (i.e., for overfitting) and, consequently, more representative of the unseen 'test error'. Some examples of such metrics are adjusted R2, AIC and BIC.

Estimate the test error via a validation set or a cross-validation approach.
In the validation set approach, we find the test error by training the model on a training set and fitting on an unseen validation set, while the in n-fold cross-validation approach, we take the mean of errors generated by training the model on all folds except the kth fold and testing the model on the kth fold, where k varies from 1 to n.

So far, you have understood that MSE of the training might not be a good estimate of the test error. The aforementioned parameters are a manipulation of the RSS (residual sum of squares), wherein a penalty term is introduced to compensate for the increase in complexity due to the increase in the number of predictors.

Question 5

You have determined the optimal value of lambda for ridge and lasso regression during the assignment. Now, which one will you choose to apply and why?

Answer 5: We found the optimal values of lambda for lasso and ridge to be 1000 and 450 respectively.

TOP 5 FEAUTURES FROM ABOVE MODELS ARE:
-> OverallQual
-> GrLivArea
-> Neighborhood
-> GarageCars
-> Condition1

1. For the same values of alpha, the coefficients of lasso regression are much smaller as compared to that of ridge regression .
2. For the same alpha, lasso has higher RSS (poorer fit) as compared to ridge regression
3. Many of the coefficients are zero even for very small values of alpha.

2. Typical Use Cases

- **Ridge:** It is majorly used to *prevent overfitting*. Since it includes all the features, it is not very useful in case of exorbitantly high #features, say in millions, as it will pose computational challenges.
- **Lasso:** Since it provides *sparse solutions*, it is generally the model of choice (or some variant of this concept) for modelling cases where the #features are in millions or more. In such a case, getting a sparse solution is of great computational advantage as the features with zero coefficients can simply be ignored.

Its not hard to see why the stepwise selection techniques become practically very cumbersome to implement in high dimensionality cases. Thus, lasso provides a significant advantage.

We will select lasso regression with this dataset.