Methods to avoid Over-fitting

- One of the major problem while training Machine learning models is "overfitting" or underfitting.
- The model will have very high training accuracy but less test accuracy when the model is overfitted.
- overfitting means the model tries to fit too hard to the data with noise (learn more about overfit and underfit in our previous posts).
- So we need to balance the error between overfitting and underfitting. we call this as generalization error.
- There are many methods to do that.

- **Cross-Validation :** Cross Validation in its simplest form is a one round validation, where we leave one sample as in-time validation and rest for training the model. But for keeping lower variance a higher fold cross validation is preferred.
- **Early Stopping:** Early stopping rules provide guidance as to how many iterations can be run before the learner begins to over-fit.

Methods to avoid Over-fitting

- **Pruning :** Pruning is used extensively while building CART models. It simply removes the nodes which add little predictive power for the problem in hand.
- **Regularization :** This is the technique we are going to discuss in more details. Simply put, it introduces a cost term for bringing in more features with the objective function. Hence, it tries to push the coefficients for many variables to zero and hence reduce cost term.

Methods to avoid Over-fitting

Cross-Validation

- Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.
- In this method we will stop one set of samples as validation and other samples as testing. Ans we do that K times (k is the only parameter in k-fold cross validation)
- It is used to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.
- few common methods used for Cross Validation

1. validation set approach
2. Leave one out cross validation (LOOCV)
3. k-fold cross validation
4. Stratified k-fold cross validation
5. Adversarial Validation
6. Cross Validation for time series

Methods to avoid Over-fitting

validation set approach

- In this approach, we reserve 50% of the dataset for validation and the remaining 50% for model training. However, a major disadvantage of this approach is that since we are training a model on only 50% of the dataset, there is a huge possibility that we might miss out on some interesting information about the data which will lead to a higher bias (underfitting).

Leave one out cross validation (LOOCV)

- Here we use only one data point in complete data set as a validation and remaining data as training.
- We do this iteratively for every data point.
- We use all data points so we can reduce the bias, but it takes lot of time to execute because we iterate this N(size of dataset) time.
- if we leave out few point as P instead of one point we call it as LPOCV

Methods to avoid Over-fitting

k-fold cross validation

- In the last two approaches we used to have high bias and high variance issues.
- So we can remove them here in this approach.
- The procedure for doing this method is as follows.

1. Shuffle the dataset randomly
2. Randomly split your entire dataset into k"folds"
3. For each k-fold in your dataset, build your model on k – 1 folds of the dataset. Then, test the model to check the effectiveness for kth fold
4. Record the error you see on each of the predictions
5. Repeat this until each of the k-folds has served as the test set.
6. The average of your k recorded errors is called the cross-validation error and will serve as your performance metric for the model.

To choose the right value of K it depends on many parameters like dataset size, test and validation dataset distribution, maintaining the balance between bias and variance as small K leads to high bias and large k leads to high variance.

Methods to avoid Over-fitting

Stratified k-fold cross validation

- Stratification is the process of rearranging the data so as to ensure that each fold is a good representative of the whole. For example, in a binary classification problem where each class comprises of 50% of the data, it is best to arrange the data such that in every fold, each class comprises of about half the instances.
- It is generally a better approach when dealing with both bias and variance. A randomly selected fold might not adequately represent the minor class, particularly in cases where there is a huge class imbalance.
- The mean response value is approximately equal in all the folds is another way of saying the proportion of each class in all the folds are approximately equal.

Methods to avoid Over-fitting

Adversarial Validation

- When dealing with real datasets, there are often cases where the test and train sets are very different. As a result, the internal cross-validation techniques might give scores that are not even in the ballpark of the test score. In such cases, adversarial validation offers an interesting solution.
- The general idea is to check the degree of similarity between training and tests in terms of feature distribution. If It does not seem to be the case, we can suspect they are quite different. This intuition can be quantified by combining train and test sets, assigning 0/1 labels (0 – train, 1-test) and evaluating a binary classification task.
- Remove the target variable from the train set. Create a new target variable which is 1 for each row in the train set, and 0 for each row in the test set. Combine the train and test datasets. Using the above newly created target variable, fit a classification model and predict probabilities for each row to be in the test set. Sort the train set using the calculated probabilities in step 4 and take top n% samples/rows as the validation set (n% is the fraction of the train set you want to keep in the validation set)

Methods to avoid Over-fitting

Cross Validation for time series

- If you have a time series data. where the data is collected sequentially through time. Random splitting of this kind of data will not work properly because the data may change over the period of time.
- Folds for time series cross valdiation are created in a forward chaining fashion.
- We progressively select a new train and test set. We start with a train set which has a minimum number of observations needed for fitting the model. Progressively, we change our train and test sets with each fold. In most cases, 1 step forecasts might not be very important. In such instances, the forecast origin can be shifted to allow for multi-step errors to be used.

Methods to avoid Over-fitting

Early stopping

- Some times it is better to stop in middle without going to end.
- Training for few iterations may underfit the model. training for huge number of iterations may lead to overfit.
- By monitoring the training and validation loss we can stop the training in between when we see a performance degrading at some point. we will discus

Methods to avoid Over-fitting

Regularization

- This is a form of regression, that constrains / regularizes or shrinks the coefficient estimates towards zero. In other words, this technique discourages learning a more complex or flexible model, so as to avoid the risk of overfitting.
- Lets take a simple linear regression equation of multivariate.
- $Y = a1X1 + a2X2 + a3X3 + ...$
- So here we fit a model to get the best fit line which has the low error value (R square or mean square).
- Here a1, a2, a3 ..... are the coefficients and X1, X2, X3,..... are data points where we we estimate a1, a2 , a3 …based on an objective function(loss function).
- Now, this will adjust the coefficients based on your training data. If there is noise in the training data, then the estimated coefficients won't generalize well to the future data.
- Now, this optimization might simply overfit the equation if x1 , x2 , x3 (independent variables) are too many in numbers.
- This is where regularization comes in and shrinks or regularizes these learned estimates towards zero.

Methods to avoid Over-fitting

Regularization

- We have 3 types of regularizers 1. Ridge, 2. Lasso and 3. elastic net

Ridge Regression

- In Ridge Regression, the OLS loss function is augmented in such a way that we not only minimize the sum of squared residuals but also penalize the size of parameter estimates, in order to shrink them towards zero(it will not go to zero) to have a very low variance.
- loss = OLS + λ * ||weights (coefficients)||*2
- The λ (shrinkage parameter) is the regularization penalty. So, setting λ to 0 is the same as using the OLS, while the larger its value, the stronger is the coefficients' size penalized.
- you can see that as λ becomes larger, the variance decreases, and the bias increases.
- Regularization term stops your coefficients going too high or too low.
- We can

Methods to avoid Over-fitting

Ridge Regression

- It shrinks the value of coefficients but doesn't reach zero, which suggests no feature selection feature
- This method uses the l2 regularization.

Methods to avoid Over-fitting

Lasso Regression

- Lasso, or Least Absolute Shrinkage and Selection Operator, is quite similar conceptually to ridge regression. It also adds a penalty for non-zero coefficients, but unlike ridge regression which penalizes sum of squared coefficients (the so-called L2 penalty), lasso penalizes the sum of their absolute values (L1 penalty).
- As a result, for high values of λ, many coefficients are exactly zeroed under lasso, which is never the case in ridge regression.
- The only difference in ridge and lasso loss functions is in the penalty terms. Under lasso, the loss is defined as
- Loss = OLS + λ|sum(Coefficients)|.
- group of predictors are highly correlated, lasso picks only one of them and shrinks the others to zero

Methods to avoid Over-fitting

Elastic Net

- ElasticNet is a hybrid of Lasso and Ridge Regression techniques. It is trained with L1 and L2 prior as regularizer. Elastic-net is useful when there are multiple features which are correlated. Lasso is likely to pick one of these at random, while elastic-net is likely to pick both.
- Loss = OLS + λ1|sum(Coefficients)| + λ2||Coefficients)||
- A practical advantage of trading-off between Lasso and Ridge is that it allows Elastic-Net to inherit some of Ridge's stability under rotation.
- It encourages group effect in case of highly correlated variables
- There are no limitations on the number of selected variables
- It can suffer from double shrinkage