Evaluation and Selection of Models in Machine Learning

On a high level, Machine Learning is the union of statistics and computation. The machine learning revolves around the concept of algorithms or models which are in fact statistical estimations on steroids.

However, any given model has several limitations depending on the data distribution. None of them can be entirely accurate since they are just estimations. These limitations are popularly known by the name of bias and variance.

A model with high bias will oversimplify by not paying much attention to the training points (e.g.: in Linear Regression, irrespective of data distribution, the model will always assume a linear relationship).

A model with high variance will restrict itself to the training data by not generalizing for test points that it hasn't seen before (e.g.: Random Forest with max_depth = None).

The issue arises when the limitations are subtle, like when we have to choose between a random forest algorithm and a gradient boosting algorithm or between two variations of the same decision tree algorithm. Both will tend to have high variance and low bias.

This is where model selection and model evaluation come into play!

Points to Focus:

- *What are model selection and model evaluation?*
- *Effective model selection methods (resampling and probabilistic approaches)*
- *Popular model evaluation methods*
- *Important Machine Learning model trade-offs*

What are model selection and model evaluation?

Model evaluation is a method of assessing the correctness of models on test data. The test data consists of data points that have not been seen by the model before.

Model selection is a technique for selecting the best model after the individual models are evaluated based on the required criteria.

Types of model selection

Resampling methods

Resampling methods, as the name suggests, are simple techniques of rearranging data samples to inspect if the model performs well on data samples that it has not been trained on. In other words, resampling helps us understand if the model will generalize well.

Random Split

Random Splits are used to randomly sample a percentage of data into training, testing, and preferably validation sets. The advantage of this method is that there is a good chance that the original population is well represented in all the three sets. In more formal terms, random splitting will prevent a biased sampling of data.

It is very important to note the use of the validation set in model selection. The validation set is the second test set and one might ask, why have two test sets?

In the process of feature selection and model tuning, the test set is used for model evaluation. This means that the model parameters and the feature set are selected such that they give an optimal result on the test set. Thus, the validation set which has completely unseen data points (not been used in the tuning and feature selection modules) is used for the final evaluation.

Time-Based Split

There are some types of data where random splits are not possible. For example, if we have to train a model for weather forecasting, we cannot randomly divide the data into training and testing sets. This will jumble up the seasonal pattern! Such data is often referred to by the term – Time Series.

In such cases, a time-wise split is used. The training set can have data for the last three years and 10 months of the present year. The last two months can be reserved for the testing or validation set.

There is also a concept of window sets – where the model is trained till a particular date and tested on the future dates iteratively such that the training window keeps increasing shifting by one day (consequently, the test set also reduces by a day). The advantage of this method is that it stabilizes the model and prevents overfitting when the test set is very small (say, 3 to 7 days).

However, the drawback of time-series data is that the events or data points are not mutually independent. One event might affect every data input that follows after.

For instance, a change in the governing party might considerably change the population statistics for the years to follow. Or the infamous coronavirus pandemic is going to have a massive impact on economic data for the next few years.

No machine learning model can learn from past data in such a case because the data points before and after the event have major differences.

K-Fold Cross-Validation

The cross-validation technique works by randomly shuffling the dataset and then splitting it into k groups. Thereafter, on iterating over each group, the group needs to be considered as a test set while all other groups are clubbed together into the training set. The model is tested on the test group and the process continues for k groups.

Thus, by the end of the process, one has k different results on k different test groups. The best model can then be selected easily by choosing the one with the highest score.

Bootstrap

Bootstrap is one of the most powerful ways to obtain a stabilized model. It is close to the random splitting technique since it follows the concept of random sampling.

The first step is to select a sample size (which is usually equal to the size of the original dataset). Thereafter, a sample data point must be randomly selected from the original dataset and added to the bootstrap sample. After the addition, the sample needs to be put back into the original sample. This process needs to be repeated for N times, where N is the sample size.

Therefore, it is a resampling technique that creates the bootstrap sample by sampling data points from the original dataset with replacement. This means that the bootstrap sample can contain multiple instances of the same data point.

The model is trained on the bootstrap sample and then evaluated on all those data points that did not make it to the bootstrapped sample. These are called the out-of-bag samples.

Probabilistic measures

Probabilistic Measures do not just take into account the model performance but also the model complexity. Model complexity is the measure of the model's ability to capture the variance in the data.

For example, a highly biased model like the linear regression algorithm is less complex and on the other hand, a neural network is very high on complexity.

Another important point to note here is that the model performance taken into account in probabilistic measures is calculated from the training set only. A hold-out test set is typically not required.

A fair bit of disadvantage however lies in the fact that probabilistic measures do not consider the uncertainty of the models and has a chance of selecting simpler models over complex models.

Akaike Information Criterion (AIC)

It is common knowledge that every model is not completely accurate. There is always some information loss which can be measured using the KL information metric. Kulback-Liebler or KL divergence is the measure of the difference in the probability distribution of two variables.

A statistician, Hirotugu Akaike, took into consideration the relationship between KL Information and Maximum Likelihood (in maximum-likelihood, one wishes to maximize the conditional probability of observing a datapoint X, given the parameters and a specified probability distribution) and developed the concept of Information Criterion (or IC). Therefore, Akaike's IC or AIC is the measure of information loss. This is how the discrepancy between two different models is captured and the model with the least information loss is suggested as the model of choice.

$$AIC = (2K - 2log(L))/N$$

- *K = number of independent variables or predictors*
- *L = maximum-likelihood of the model*
- *N = number of data points in the training set (especially helpful in case of small datasets)*
  The limitation of AIC is that it is not very good with generalizing models as it tends to select complex models that lose less training information.

Bayesian Information Criterion (BIC)

BIC was derived from the Bayesian probability concept and is suited for models that are trained under the maximum likelihood estimation.

$$BIC = K * log(N) - 2log(L)$$

- *K = number of independent variables*
- *L = maximum-likelihood*
- *N = Number of sampler/data points in the training set*
  BIC penalizes the model for its complexity and is preferably used when the size of the dataset is not very small (otherwise it tends to settle on very simple models).

How to evaluate the ML models?

Models can be evaluated using multiple metrics. However, the right choice of an evaluation metric is crucial and often depends upon the problem that is being solved. A clear understanding of a wide range of metrics can help the evaluator to chance upon an appropriate match of the problem statement and a metric.

Classification metrics

For every classification model prediction, a matrix called the confusion matrix can be constructed which demonstrates the number of test cases correctly and incorrectly classified.

It looks something like this (considering 1 -Positive and 0 - Negative are the target classes):

	Actual 0	*Actual 1*
Predicted 0	*True Negatives (TN)*	*False Negatives (FN)*
Predicted 1	*False Positives (FP)*	*True Positives (TP)*

- *TN: Number of negative cases correctly classified*
- *TP: Number of positive cases correctly classified*
- *FN: Number of positive cases incorrectly classified as negative*
- *FP: Number of negative cases correctly classified as positive*

Accuracy

Accuracy is the simplest metric and can be defined as the number of test cases correctly classified divided by the total number of test cases.

$$Accuracy = (TP+TN)/(TP+TN+FP+FN)$$

It can be applied to most generic problems but is not very useful when it comes to unbalanced datasets.

For instance, if we are detecting frauds in bank data, the ratio of fraud to non-fraud cases can be 1:99. In such cases, if accuracy is used, the model will turn out to be 99% accurate by predicting all test cases as non-fraud. The 99% accurate model will be completely useless.

If a model is poorly trained such that it predicts all the 1000 (say) data points as non-frauds, it will be missing out on the 10

*fraud data points. If accuracy is measured, it will show that that model correctly predicts 990 data points and thus, it will have an accuracy of (990/1000)*100 = 99%!*

This is why accuracy is a false indicator of the model's health.

Therefore, for such a case, a metric is required that can focus on the ten fraud data points which were completely missed by the model.

Precision

Precision is the metric used to identify the correctness of classification.

$$Precision = TP/(TP + FP)$$

Intuitively, this equation is the ratio of correct positive classifications to the total number of predicted positive classifications. The greater the fraction, the higher is the precision, which means better is the ability of the model to correctly classify the positive class.

In the problem of predictive maintenance (where one must predict in advance when a machine needs to be repaired), precision comes into play. The cost of maintenance is usually high and thus, incorrect predictions can lead to a loss for the company. In such cases, the ability of the model to correctly classify the positive class and to lower the number of false positives is paramount!

Recall

Recall tells us the number of positive cases correctly identified out of the total number of positive cases.

$$Recall = TP/(TP + FN)$$

Going back to the fraud problem, the recall value will be very useful in fraud cases because a high recall value will indicate that a lot of fraud cases were identified out of the total number of frauds.

F1 Score

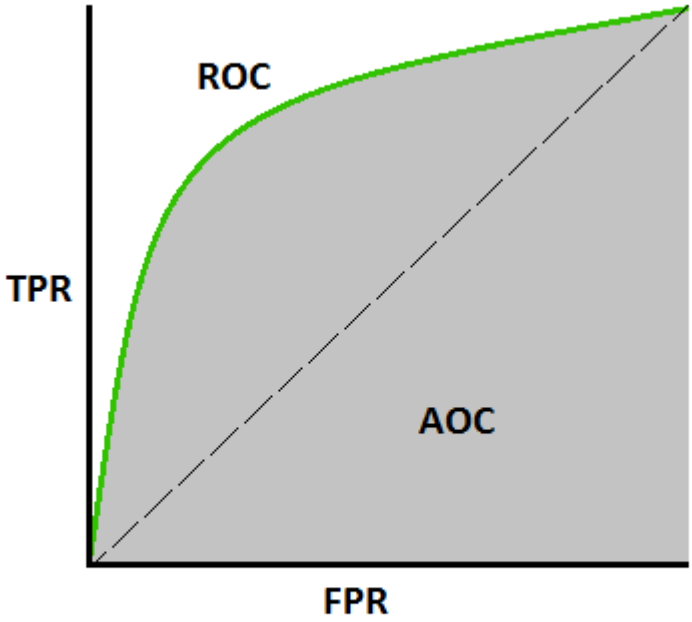
F1 score is the harmonic mean of Recall and Precision and therefore, balances out the strengths of each.

It is useful in cases where both recall and precision can be valuable – like in the identification of plane parts that might require repairing. Here, precision will be required to save on the company's cost (because plane parts are extremely expensive) and recall will be required to ensure that the machinery is stable and not a threat to human lives.

$$F1Score = 2 * ((precision * recall)/(precision + recall))$$

AUC-ROC

ROC curve is a plot of true positive rate (recall) against false positive rate (TN / (TN+FP)). AUC-ROC stands for Area Under the Receiver Operating Characteristics and the higher the area, the better is the model performance. If the curve is somewhere near the 50% diagonal line, it suggests that the model randomly predicts the output variable.



AUC - ROC Curve [Image 2] (Image courtesy: My Photoshopped Collection)

Regression metrics

Regression models provide a continuous output variable, unlike classification models that have discrete output variables. Therefore, the metrics for assessing the regression models are accordingly designed.

Mean Squared Error or MSE

MSE is a simple metric that calculates the difference between the actual value and the predicted value (error), squares it and then provides the mean of all the errors.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widetilde{y}_i)^2$$

MSE is very sensitive to outliers and will show a very high error value even if a few outliers are present in the otherwise well-fitted model predictions.

Root Mean Squared Error or RMSE

RMSE is the root of MSE and is beneficial because it helps to bring down the scale of the errors closer to the actual values, making it more interpretable.

Mean Absolute Error or MAE

MAE is the mean of the absolute error values (actuals – predictions).

$$MAE = \frac{1}{n}\sum_{i=1} |x_i - x|$$

If one wants to ignore the outlier values to a certain degree, MAE is the choice since it reduces the penalty of the outliers significantly with the removal of the square terms.

Root Mean Squared Log Error or RMSLE

In RMSLE, the same equation as that of RMSE is followed except for an added log function along with the actual and predicted values.

$$RMSLE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(log(x_i+1)) - (log(y_i+1))^2}$$

x is the actual value and y is the predicted value. This helps to scale down the effect of the outliers by downplaying the higher error rates with the log function. Also, RMSLE helps to capture a relative error (by comparing all the error values) through the use of logs.

R-Squared

R-Square helps to identify the proportion of variance of the target variable that can be captured with the help of the independent variables or predictors.

$$R{-}Squared = 1{-}(UnexplainedVariation/TotalVariation) = 1{-}(Variance(model)/V$$

R-square, however, has a gigantic problem. Say, a new unrelated feature is added to a model with an assigned weight of w. If the model finds absolutely no correlation between the new predictor and the target variable, w is 0. However, there is almost always a small correlation due to randomness which adds a small positive weight (w>0) and a new loss minimum is achieved due to overfitting.

This is why the R-squared increases with any new feature addition. Thus, its inability to decrease in value when new features are added limits its ability to identify if the model did better with lesser features.

Adjusted R-Squared

Adjusted R-Square solves the problem of R-Square by dismissing its inability to reduce in value with added features. It penalizes the score as more features are added.

$$adj\,R_2 = 1 - (1 - R^2)\frac{n-1}{n-m-1}$$

The denominator here is the magic element which increases with the increase in the number of features. Therefore, a significant increase in R^2 is required to increase the overall value.

Trade-offs

Bias vs variance

At the beginning of this article, the mention of bias and variance was important to understand the need for model evaluation and selection. But understanding it in more detail is necessary to grasp the concepts of evaluation and selection well.

Bias occurs when a model is strictly ruled by assumptions – like the linear regression model assumes that the relationship of the output variable with the independent variables is a straight line. This leads to underfitting when the actual values are non-linearly related to the independent variables.

Variance is high when a model focuses on the training set too much and learns the variations very closely, compromising on generalization. This leads to overfitting.

An optimal model is one that has the lowest bias and variance and since these two attributes are indirectly proportional, the only way to achieve this is through a tradeoff between the two. Therefore, the model selection should be such that the bias and variance intersect like in the image below.

Soure: Analytics Vidhya

This can be achieved by iteratively tuning the hyperparameters of the model in use (Hyperparameters are the input parameters that are fed to the model functions). After every iteration, the model evaluation must take place with the use of a suitable metric.

Summary

Machine learning has a lot of concepts and algorithms and this article just scratches the surface. Both model selection and model evaluation techniques can appear to be a bit extensive, but it comes easily through practice and effective investment of time. Different problems require different treatments and you should use the techniques that make sense for your project.