Cory Pekkala
Leo Nguyen

Support vector machine (SVM) is a supervised machine learning model that can be used for classification or regression. SVM works by taking data in a lower dimensional space and mapping it onto a higher dimensional space so that a decision boundary can be drawn best separating different groups in the data.

The decision boundary is a hyperplane that best separates the data. This boundary can be multidimensional. For example, in a binary classification problem, a SVM might be able to identify a linear decision boundary that maximizes the distance to the nearest observation in either class. However, in classification problems for more than two targets, a SVM might not be able to find a linear relationship that does so. Instead, as involved with the kernel, SVM would map the data points to a higher dimensional space to draw a better decision boundary.

A SVM kernel is the transformer which is used to find a decision boundary. For instance, the binary classification scenario explored previously, yielded a linear decision boundary brought about by a linear kernel. In the case of more than two targets, a nonlinear decision boundary would've been determined by either a polynomial, or radial basis function kernel.

SVM kernels are feature generators. We've explored some of the inner workings of the linear kernel previously. The polynomial kernel, on the other hand, tries to combine existing and new features by computing polynomial operations on the existing ones. It does so in an attempt to find a nonlinear decision boundary that best fits the data. Whereas, the radial basis function kernel tries to create new features by computing the distance between data points to center point.

Support vector machines are strong when the data can be easily separated into classes. In addition, when dealing with high-dimensional data, even to an extent where there are more dimensions than there are observations, support vector machines can be really effective. On the other side, support vector machines tend not to perform well when dealing with a lot of data. In addition, the classification results of a SVM tend to be a little less interpretable than other models, such as decision trees for example, since the decision boundary is more or less as a result of new features being generated by a kernel in a geometric fashion.

Random Forest is an ensemble, supervised learning technique that makes use of multiple weak decision tree learners and combines their results to get a stronger learner. The idea behind this model is that the individual decision trees are uncorrelated and each makes a guess as to what a piece of data should be classified as. Once all of the trees make their classification for a given piece of data, the random forest takes the majority vote of all the trees, and classifies the data as such.

Other ensemble methods, such as XGBoost and AdaBoost classifiers are considered here as well. We chose both of these techniques in our third R notebook. The problem that ensemble techniques are trying to solve is that generally results are better if a bunch of models work together versus a single model. Both XGBoost and AdaBoost work off this premise.

XGBoost deals with gradient boosting, which is the ensemble technique used to aggregate the predictions from weak learners (decision trees) to predict a target. Each weak learner operates in a sequential manner, where the weights of the input features are continually adjusted while each learner attempts to predict their labels, if a prediction is wrong, then a penalty is incurred and reflected in the new weight value.

AdaBoost on the other hand uses adaptive boosting. In this algorithm, weak learners are ensemble and make predictions on instances. Instances that are not predicted correctly are favored more by the subsequent weak learner. As long as one of the learners in the ensemble is predicting a little bit better than randomly guessing, then the results of all the learners can form a stronger learner by the end, giving more accurate results. In general these boosting methods reduce variance among all learners.

XGBoost is one of the most popular data science algorithms right now. Many data science competitions are won using the XGBoost algorithm. On this note, there are other key strengths it brings to the table. It can work well with different datasets, large or small, as long as it is structured data. It can be used for regression or classification, so there is a wide range of applicability. It also works to reduce overfitting and the input data need not be scaled in order for the algorithm to work, reducing the amount of preprocessing necessary. Overall, because it makes use of trees, the end predictions are highly interpretable as well. All that considered, there are some costs to using XGBoost. For example, if working with a lot of unstructured data, then XGBoost would not be as effective. In addition, if that data after being transformed into structured data contained a lot of outliers, then that could impact performance. As each learner tries to correct the misclassifications of the previous learner, dealing with an outlier during all this could take up some time.

Similar to XGBoost, AdaBoost is a sequential ensemble technique which uses the error in the previous weak learners predictions to adjust weights on the input data for the next learner to predict on. It has a few key strengths. One of which is that it is able to avoid overfitting for the most part. In addition, a generally weak classifier can be converged or transformed into a stronger classifier. It is popular in the natural language processing space to classify data. But there is one disadvantage to using this technique. AdaBoost requires clean data. Data with a lot of noise or outliers in the mix have to be removed, because otherwise - similar to XGBoost, performance will take a hit. In the end, this means you'll be spending more time preprocessing the data going with AdaBoost over XGBoost.

Resources

https://www.ibm.com/docs/en/spss-modeler/saas?topic=models-how-svm-works

https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/

https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496

https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107

https://careerfoundry.com/en/blog/data-analytics/what-is-random-forest/#:~:text=Random%20Forest%20grows%20multiple%20decision,group%20than%20they%20do%20alone.

https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/

https://www.mygreatlearning.com/blog/adaboost-algorithm/#:~:text=It%20works%20on%20the%20principle,boosting%20with%20a%20slight%20difference.

https://www.geeksforgeeks.org/xgboost/#:~:text=XGBoost%20uses%20both%20Lasso%20and,to%20be%20sorted%20in%20order.

https://neptune.ai/blog/xgboost-everything-you-need-to-know#:~:text=Disadvantages,overall%20method%20is%20hardly%20scalable.