Justin Hardy

Benjamin Frenkel

CS 4375.003

Dr. Karen Mazidi

Kernel and Ensemble Methods Reflection

I. SVM OVERVIEW

Support Vector Machines (SVMs) are linear models that can be used for both classification and regression problems. It's versatile in that it can work well with linear and nonlinear data alike. The idea behind it is that it creates a hyperplane - a straight line - through the data that separates the data into set classes. A margin is drawn on both ends of the line, with the goal of no data points being within the margin. Any instances of data that lie within the margin are known as support vectors, hence the algorithm's name. Hyperparameters can be utilized to tune the model's accuracy, which are essentially 'fake' data instances that purely exist to help generalize the model. We tend to tune our model using hyperparameters on a smaller, validation data set, in order to avoid possibly overfitting the model. SVMs have three kernel modes that can be used to help classify the data: Linear, Polynomial, and Radial.

The Linear Kernel simply attempts to separate the data linearly using a straight line, as described previously. This will generally map the data to a low dimension, as compared to the Polynomial Kernel which will map the data to a higher dimension so that it can be separated linearly in that dimension. The Radial Kernel, similarly to the Polynomial Kernel, will morph the shape of the hyperplane and its boundaries, such as in a circular-esque shape. Polynomial and Radial Kernels utilize gamma parameters to control the bias-variance tradeoff, and all of the kernels utilize cost parameters to determine the level of impact slack variables will have on the model, with lower values resulting in an increase in variance and decrease in bias, and higher values resulting in larger margins.

SVM is strong in many ways, and weak in others. It’s particularly strong in the level of flexibility the model has, with the different kernel modes that can be used, the tuning that can be done to the model, and the ability to use it for both classification and regression data problems. It’ll also generally perform better than logistic regression when the classes are well-separated. The biggest weakness we noticed in our testing is that training the model is extremely slow when working with medium to large data sets.

II. RANDOM FOREST OVERVIEW

Random Forest is interesting in that it utilizes the bagging method - bootstrap aggregation - to overcome a model’s variance. The model breaks the data down into subsets, and trains multiple different trees independently on those subsets. The results of the trees are averaged and the best tree is selected. In particular to bagging, the full set of predictors are used for each tree rather than select predictors. It’s strong in that it can automatically handle NA values as well as outliers, as well as reduce overfitting and variance in order to improve accuracy. However, it takes quite a while to train.

XGBoost is a more advanced tree that uses multithreading to build hundreds of trees, which are then aggregated together. It’s extremely scalable and capable of running up to ten times faster than algorithms that preceded it. The number of iterations it makes can be controlled to tune the model, with lower values resulting in underfitting, and higher values resulting in overfitting. Its strength lies with its efficiency and performance/accuracy, and that it can be used in both classification and regression problems. It doesn’t do too well on data that lacks structure though.

AdaBoost is an algorithm that’ll iterate through a given number of learners, and train them one-by-one. Over each iteration, weights for training examples are increased when the observations have large errors, and decreased when the observations are correct. Once learners have been trained, each learner is given a weight based on their weighted errors - so that accurate learners have high weights - and the best is selected. AdaBoost is strong in that it’s unlikely to overfit the data, and it’s fairly easy to use due to there being little need to tweak parameters. However, it’s weak in that it is extremely sensitive to outliers.