Kernel and Ensemble Methods

Members: Muhammad Shariq Azeem and Aarushi Pandey

SVM

In simple terms, Support Vector Machine(SVM) is an algorithm that takes data as input and creates a line or a hyperplane that separates the data into different classes. It sounds easy, but usually, there are several lines that can separate the data, and the main goal of the SVM algorithm is to find the best one.

SVM does that by finding points closest to the line from each class and computing the distance between those points and the line; this distance is called the margin. After computing the margin for each possible line, SVM selects the hyperplane that maximizes the margin, to be the optimal hyperplane.

SVM Kernels:

There are different functions, within SVM, that takes the data and transforms it into required form; these functions are called kernels. Some types of kernels are linear, polynomial, and radial. Linear kernels try find a linear hyperplane, polynomial kernels find a polynomial hyperplane, and radial kernels find circular hyperplane.

Strengths of SVM:

- Works well when there is a clear margin that separates the classes.
- More effective in high dimensional spaces.
- Relatively memory efficient.

Weaknesses of SVM:

- Not suitable for large data sets.
- No probabilistic explanation for the classification.
- Does not perform well when there is an overlap in the target classes.

Random Forests

It is an ensemble technique in which many trees are created in a de-correlated manner. At each split, a random subset of predictors is selected and then one predictor is selected from that subset.

Strengths of Random Forest:

- Can build trees that do not start with a particular strong predictor.
- Excellent performance.
- Can do classification or regression.
- Does not require normalization or scaling of data.
- Not influenced by outliers.

Weaknesses of Random Forest:

- Less interpretability than decision trees.
- Less control over what the model does (i.e. which predictor(s) it chooses)

XGBoost

For XGBoost (extreme gradient boosting), the data must be converted to a numeric matrix and labels must be 0/1 integers. (This not a prerequisite for decision trees.) The number of trees is controlled by the nrounds parameter, and the greater the nrounds the better the accuracy (up to a certain point). The hypothesis space of all the possible trees is unlimited, so we need to recursively split on the features with the highest information gain until a stopping threshold. Overfitting is avoided by pruning the tree and setting up a maximum depth (similar to a decision tree). The results of each tree are aggregated along the way to calculate the final result.

Strengths of XGBoost:

- Can do classification or regression.
- Missing values are handled internally.
- Will run on all cores of machine by default.

- Model can be saved on disk.

Weaknesses of XGBoost:

- Need to scale data.
- Sensitive to outliers.
- Does not perform well on sparse and unstructured data.

AdaBoost

Known as adaptive boosting, the weights of the examples are equal in the first training (1/N where N is the number of training examples). In subsequent iterations, the weights of observations with large errors are increased and weights for correct observations are decreased. (Over iterations, the weighting errors are used to weight the learners so that the more accurate learners are given higher weights.) It is the first designed boosting algorithm with a loss function (which random forest/decision tree does not have).

Strengths of AdaBoost:

- Less prone to overfitting (theoretically).
- Can use it with SVM.
- Can use it to improve accuracy of weak classifiers.

Weaknesses of AdaBoost:

- Sensitive to outliers.
- Does not perform well with noisy data.
- Takes more time (than XGBoost and decision tree) to run.