COMS 4771 HW3 (Fall 2024) Random Forest problem

Brian Chen, Andrew Yang

1 Classification Using Random Forests

In class we have discussed a classification algorithm called decision trees. Under mild conditions[1], decision trees are able to obtain 100% accuracy on training data, but this comes with the obvious issue of overfitting to the training set. *Random forests* are a type of classifier which aggregate a number of decision trees together to make a classification. In this problem, you will prove theoretically some of the advantages of random forests over decision trees and verify them empirically on a real-world dataset.

1.1 Single Decision Tree

(i) For this problem, our decision tree boundaries b will maximize information gain $IG(b; C)$. If b splits a cell C into C_L and C_R, it has an information gain given by

$$IG(b; C) = H(C) - \Pr_{i \in C}[i \in C_L] H(C_L) - \Pr_{i \in C}[i \in C_R] H(C_R),$$

where

$$H(C) = -\sum_{y \in Y} \Pr_{i \in C}[y_i = y] \ln(\Pr_{i \in C}[y_i = y])$$

is the Shannon entropy[2] of cell C over labels Y.

Can you construct a (binary-label) dataset where no decision boundary can give a positive information gain? If each feature is drawn from a continuous distribution, what is the probability of data arranging itself in such a way? How does the answer to the previous questions change for a dataset with more than two labels?

(ii) You should now implement a single decision tree that greedily maximizes information gain. The input data will be a single matrix M_D. Each row $i \in [n]$ will be of the form $(\vec{x}_i, y_i)$, where $\vec{x}_i$ are the features and y_i the label.

Recursively define the decision tree algorithm as follows:

1 Let all data be in a single cell.

2 Find the decision boundary b that maximizes the information gain on the cell. Decision boundaries should involve a single feature j and single threshold value t_j.

3 For each *new* cell created by the decision boundary b, repeat the previous step.

For finite training data, this process is guaranteed to terminate.

(iii) Download the spam dataset linked here. This dataset consists of 4601 emails, labelled as spam or not spam, and 57 observed features corresponding to the frequency of different words.

Using this dataset construct a decision tree trained on a single data point. What is the classification accuracy of that tree (tested on the rest of the dataset)? What classification accuracies can you expect from such a tree?

[1] The mild condition: There are no samples in the training data that have different labels but have identical features.

[2] If you are interested in why Shannon entropy is so widely used in machine learning, see section 6 of "A Mathematical Theory of Communication" (`https://doi.org/10.1002/j.1538-7305.1948.tb01338.x`).

(iv) We likely want to train our decision tree on more data. Let our entire dataset $D = (X, Y)$ have $n = |D|$ datapoints. Show that if we draw n samples from D uniformly and *with replacement*, the expected fraction of unique samples drawn approaches $(1 - 1/e)$ as $n \to \infty$.

Perform this uniform and with replacement draw on your spam email dataset. Construct your decision tree from the drawn (training) data and find the accuracy on the undrawn (testing) data.

1.2 Random Forest

(v) A forest is, as you can intuit, a collection of (disjoint) trees. A random forest classifier is made up of many decision trees. Split your data into training (80%) and testing (20%) sets. Then, modify your code to construct B decision trees. Each decision tree T_b should be constructed on a random subset Θ_b of the training data. For this problem, construct Θ_b by sampling the training data uniformly with replacement.

(vi) The prediction of the random forest is what a majority of your decision trees predict. If y_b is the label predicted by decision tree T_b, the random forest will make the classification $\hat{y}_{rf} = \texttt{mode}\{y_b\}_1^B$. Compute the classification accuracy (on the test data) of your random forest and compare it to the accuracy of a single decision tree for different values of B.

(vii) We can think of each tree T_b as an identically drawn random variable drawn from the distribution of all possible trees constructed from random samples of the training data. The random forest output is the expected value of the T_b output. Though each tree is trained on a different subset of data Θ_b, in practice, they can be highly (positively) correlated. For example, we may have a feature j^*, which is a very strong predictor of label. Using our greedy decision tree construction, we expect most trees to split on j^* early on. Therefore, our forest $\{T_b\}_1^B$ is a set of identically distributed, but not independent, random variables.

Let T_b be drawn from some random distribution with mean μ and variance σ^2. Define the pairwise correlation

$$\rho(T_i, T_j) = \frac{\mathbb{E}[(T_i - \mu)(T_j - \mu)]}{\sigma^2}.$$

(Since T_b are all identically distributed, there is a single pairwise correlation $\rho = \rho(T_i, T_j)\ \forall i, j \in [B]$.) Assuming that ρ is positive[3], show that

$$\texttt{Var}\left(\frac{1}{B}\sum_{b=1}^{B} T_b\right) = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2.$$

What happens to the variance as we increase the number of trees B? When can we go below the single decision tree variance σ^2? Name one way we can reduce ρ.

(viii) Why might someone use a random forest over a single decision tree? A single decision tree over a forest?

1.3 Out-of-Bag Analysis

(ix) Recall that we constructed each tree in our forest with a subset ($\sim 63\%$) of the original data. This means that around one third of the data is *not* used in each tree. For each data point $x_i \in X$, define the set of trees for which x_i was out-of-bag (not used to train each tree) as

$$\text{OOB}(x_i) = \{T_b : x_i \notin D_b\}$$

Let $\hat{y}_i^{OOB}$ be the prediction of x_i from the forest $\text{OOB}(x_i)$. We can then define OOB error as

$$\text{OOB error} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\{\hat{y}_i^{OOB} \neq y_i)$$

[3]It is possible to have negative ρ. One may use the positive semi-definiteness of the covariance matrix to show $\rho \in [-1/2, 1]$ when all pairwise correlations are the same.

Calculate the OOB error of your random forest and compare it to the test error of the random forest found in part (v). Is OOB error a biased or unbiased estimator of test error?

(x) How does your OOB error change as a function of the number of trees in your forest? How about as you change the depth m of the decision trees?

(xi) One can also use OOB data to calculate *variable importance*, which is a measure of how "important" each variable is towards the classification. Take the OOB data for tree b and count the number of correct classifications made $:= C_b$. Then permute the values of feature f_i within the OOB dataset randomly. This effectively removes any correlation between that variable and the target label. Then, recount the number of correct classifications made with the permuted-f_i data $:= C_b^{\text{perm } f_i}$. The variable importance of feature f_i is then the difference between the number of correct votes made for the permuted-f_i data and number of correct votes made on the original OOB data, averaged across all trees.

$$\texttt{variable importance}(f_i) = \frac{1}{B}\sum_{b=1}^{B}(C_b - C_b^{\text{perm } f_i})$$

Plot the variable importance for each of the 57 features on a histogram. Which features were the most "informative" and which were the least? Does this agree with your intuition based on what words would and would not separate spam and non-spam emails? When might this measure of variable importance fail, and how could one handle this issue?