Scribes: Makamba Sackey, Apoorv Walia, Ishita Kundaliya, Avi Singhal

February 14, 2023
Bagging and Boosting

COMP 553 - Machine Learning

Disclaimer: *These lecture notes are intended to develop the thought process and intuition in machine learning. The materials are not thoroughly reviewed and can contain errors.*
The first section of the class was spent discussing the project and how to develop the abstract for the project. We discussed potential project topics and the associated work expected for the projects. One specific example discussed was sentiment analysis which included devoting the project to improving the analysis accuracy, decreasing cost of analysis and several other options.

1 Decision Trees

Decision trees are a popular machine learning algorithm that is used for both regression and classification tasks. They are simple and easy to understand, yet can be powerful and accurate when used correctly. A decision tree consists of a tree-like structure, where each internal node represents a test on a feature, each branch represents the outcome of the test, and each leaf node represents a prediction or decision.

1.1 How Decision Trees work

The goal of a decision tree is to recursively split the data into smaller and more homogeneous subsets, until each subset consists of only one class or a single value. The splitting process is done based on a criterion that measures the purity or impurity of the subsets, such as the Gini index or entropy.
Let X be a set of n observations, and let y_i be the target variable for the i-th observation. A decision tree partitions the data into subsets $X_1, X_2, ..., X_m$, where each subset corresponds to a different path from the root node to a leaf node. Each internal node t in the tree represents a test on a feature j_t, where j_t is the index of the feature and s_t is the threshold for the test. The test splits the data into two subsets, $X_{t,1}$ and $X_{t,2}$, based on the condition:

$$x_{i,j_t} \leq s_t \text{ for } X_{t,1} \text{ and } x_{i,j_t} > s_t \text{ for } X_{t,2}$$

The splitting process continues recursively on each subset, until all subsets are pure or a stopping criterion is met.
The criterion for measuring the impurity of a subset depends on the task. For classification, the most commonly used criteria are the Gini index and entropy. The Gini index is defined as:

$$\text{Gini}(p) = \sum_{k=1}^{K} p_k(1 - p_k)$$

where K is the number of classes and p_k is the proportion of observations in class k in the subset. The entropy is defined as:

$$\text{Entropy}(p) = -\sum_{k=1}^{K} p_k \log_2(p_k)$$

The criterion for regression is usually the mean squared error (MSE), which is defined as:

$$\text{MSE}(y) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

where $\bar{y}$ is the mean of the target variable.
The stopping criterion for a decision tree can be based on several factors, such as the maximum depth of the tree, the minimum number of samples required to split a node, or the minimum improvement in the impurity criterion.

1.2 Benefits of Decision Trees

There are several benefits to using decision trees for machine learning tasks:

- Simplicity: Decision trees are easy to understand and interpret, and can be visualized to provide insights into the decision-making process.
- Non-parametric: Decision trees make no assumptions about the distribution of the data, and can capture complex nonlinear relationships.
- Robustness: Decision trees can handle missing values and outliers in the data, and are not affected by the scale of the features.
- Feature importance: Decision trees can provide insights into the importance of each feature in the dataset, which can be useful for feature selection and model interpretation.
- Ensemble methods: Decision trees can suffer from overfitting and instability, which can be addressed by combining them with other machine learning algorithms, such as random forests and gradient boosting, to improve performance.

2 Ensembles

In ensemble learning, you train multiple models to solve one problem and combine the models to get better results. Considering decision trees as an example, there are many factors that we can consider when making a decision tree, we can choose a subset of the features for making the tree, we can have trees with different heights, different thresholds, a subset of the data can be used for making the tree and many such other factors. Ensemble learning is a good approach when there are so many options to choose from and we dont know the best option, in these cases rather than just relying on one decision tree and expecting to make the right decision at each split, ensemble methods allow to take a sample of decision trees into account, calculate which features to use at each split and make a final predictor based on the aggregated results of the sampled decision trees. Some of the commonly used ensemble methods are discussed in the next section.

- The models used to train the final model are called **weak learners**.
- Usually, the weak learners are not as accurate by themselves due to high bias or large amount of variance.

2.1 Bagging

Bagging is a process that focuses on generating an ensemble model that has **less variance** than the weak learner models. In this technique, we choose randomly choose a subset of the data and create a new model using only that subset.Random forest is a good example of the bagging technique, with some smart tricks. When deciding where to split and how to make decisions, bagged decision trees use all the features to choose from. Therefore, even though the bootstrapped samples are different, the data will mostly split at the same features throughout each model. In contrary, Random forest decides where to split based on a random selection of features. Rather than splitting at similar features at each node throughout, Random forest models have a good level of differentiation because each tree will split based on different features (it can be random). This differentiation helps give a better ensemble and thereby producing a better model.

- In bagging, we take an ensemble of several **independent classifiers** and **average their predictions** in order to reduce variance (e.g. Random Forest)
- Bagging should improve accuracy
- One of its major **advantages** is being able to run **parallel** since the models are fit independently
- If the data only includes **numerical values** (i.e. data in a tabular form) then decision trees, Random Forest and XGBoost will result in **much higher accuracy than any other deep learning algorithm**

$$Accuracy of XGBoost > Accuracy of Random Forest$$

2.2 Boosting

Boosting is a process that focuses on generating an ensemble model that has **less bias** than the weak learner models. The ensemble is not limited to a single type of model i.e the ensemble can contain decision tress, neural networks, linear regressors etc.

- If the classifier does slightly better than random then it can be given a boost to perform better
- In boosting, models are fit using an iterative/sequential method (so not parallel like with Bagging). This also means the fits are not independent (also unlike Bagging), because the training of a model at any step depends on what the fit was at previous steps
- It is initialized with a shallow decision tree (shallow = depth of 2-3 layers)

$$f(x_i) = y_i$$

where x_i is given to the shallow tree and $\hat{y}_i$ is the output received

$$f_1 = D = (x_i, R_i^1)_{i=1}^n$$

Now we calculate the difference between this output and the actual output.

$$y_i - \hat{y}_i = R_i^1$$

Next, a new classifier (which can be another shallow decision tree) will predict R_i'

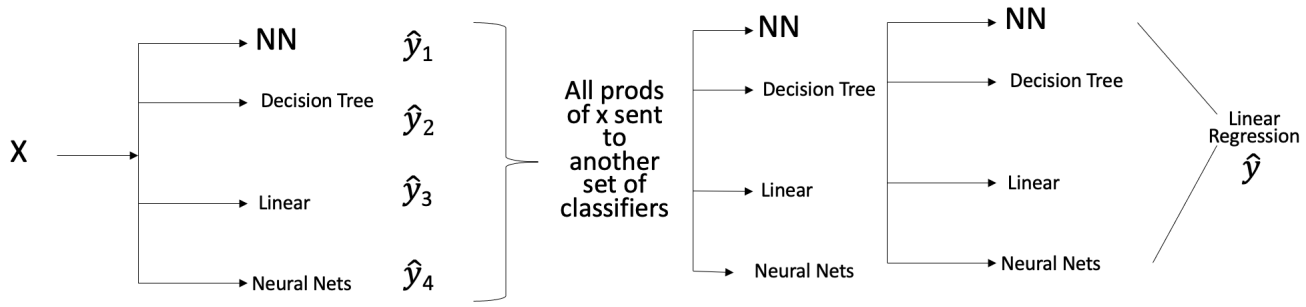

Figure 1: This is a visual representation of Boosting

$$f_2 = D = (x_i, R_i^1)_{i=1}^n$$

We continue this process until the gap between y_i and R_i^x reaches a minimum. Note: it is possible that the gap reaches 0.

$$f(x) = n_1 f_1(x) + n_2 f_2(x) + ...$$

$n_1, n_2, ...$ are the step sizes (also known as shrinkage)

2.3 Disadvantages of Boosting

- Training **must be done sequentially** when boosting. This is one of the advantages of bagging compared to boosting, since training can be done in parallel for bagging. This can result in boosting becoming too **expensive**
- Boosting has a **high inference time**

3 XGBoost

XGBoost (eXtreme Gradient Boosting) is a popular machine learning algorithm that is based on the gradient boosting framework. It is designed to minimize a cost function by iteratively adding weak learners (i.e., decision trees) that correct the errors of the previous trees. XGBoost is known for its speed and accuracy, and has been widely used for regression, classification, and ranking problems.

3.1 How XGBoost works

The XGBoost algorithm works by building an ensemble of decision trees in a way that minimizes the sum of the loss function and a regularization term. Let y_i be the target variable for the i-th observation, and let $\hat{y}_i$ be the predicted value for that observation. The objective function of XGBoost is given by:

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}i) + \sum j = 1^m \Omega(f_j)$$

where θ is the set of model parameters, $l(y_i, \hat{y}_i)$ is the loss function that measures the difference between the true target value and the predicted value, m is the number of trees in the ensemble, f_j is the j-th decision tree, and $\Omega(f_j)$ is a regularization term that penalizes complex models.
The XGBoost algorithm uses gradient boosting to optimize the objective function. Gradient boosting is an iterative method that builds the model by adding weak learners that correct the errors of the previous learners. At each iteration, the algorithm calculates the negative gradient of the loss function with respect to the predicted values:

$$r_i^{(t)} = -\left[\frac{\partial l(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}i^{(t-1)}}\right] \hat{y}_i = \hat{y}_i^{(t-1)}$$

where $\hat{y}_i^{(t-1)}$ is the predicted value for the i-th observation at iteration $t-1$. The negative gradient is also called the residual, since it represents the amount by which the predicted value differs from the true value.
The algorithm then fits a new decision tree f_j to the negative gradient, using the observations and features that have the highest information gain. The information gain is calculated using the gain function:

$$\mathcal{G} = \frac{1}{2}\left[\frac{\sum_{i \in I_j} r_i}{\sum_{i \in I_j} w_i + \lambda}\right]^2 + \frac{\gamma}{2} - \frac{1}{2}\frac{\sum_{i \in I_j} w_i^2}{\sum_{i \in I_j} w_i + \lambda}$$

where I_j is the set of observations that are assigned to node j, w_i is the weight of the i-th observation, λ and γ are regularization parameters, and the first term represents the reduction in the loss function, while the second and third terms represent the regularization penalty.
The algorithm then adds the new tree f_j to the ensemble and updates the predicted values: where x_i is the feature vector for the i-th observation, and η is the learning rate, which controls the step size of the optimization.
The algorithm continues to add trees to the ensemble until a stopping criterion is met, such as a maximum number of trees or a minimum improvement in the objective function. The final predicted value for an observation is the sum of the predicted values from all the trees in the ensemble.

3.2 Benefits of XGBoost

There are several benefits to using XGBoost for machine learning tasks:

- Speed: XGBoost is designed to be fast and scalable, and can handle large datasets with millions of observations and thousands of features.
- Accuracy: XGBoost is known for its high predictive accuracy, and has been used to win many machine learning competitions.
- Regularization: XGBoost includes several regularization techniques, such as L1 and L2 regularization, to prevent overfitting and improve generalization.
- Missing values: XGBoost can handle missing values in the dataset by learning how to make predictions based on the available data.
- Feature importance: XGBoost can provide insights into the importance of each feature in the dataset, which can be useful for feature selection and model interpretation.

4 AutoML

Automated Machine Learning (AutoML) is a process of automating the iterative tasks of machine learning model development by creating pipelines in parallel to try different algorithms and parameters.

- AutoML is a **grid type method** that automates machine learning
- It **automatically applies hyperparameters tuning** to the all possible classifiers and gives the best accuracy.
- If we look at XGBoost - It considers all possible sizes for the shallow tree and various shrinkage values.
- The **advantage** of AutoML is that it **prevents having to randomly guess**
- The **disadvantage** of AutoML is that it requires **a lot of time for computation**

5 AutoGluon

AutoGluon is an AutoML deep learning method with a focus on automating stack ensembling.

- AutoGluon can be used for various real world applications including image, text and tabular data
- The AutoGluon method does not allow for tuning the parameters. It **automates the tuning of hyperparameters**, which otherwise can take a lot of iterations and human effort to do because it is often unclear how to make modifications to hyperparameters.
- An **advantage** of AutoGluon is that it is useful in the case of **data sparsity** e.g. if you were modeling fraud and had 10 cases rather than 1000+
- The data is fed to a bunch of classifiers. The output received from this along with the input is fed to the next set of classifiers and this process continues through hundreds of classifiers
- It takes the hundreds of classifiers and passes them to another classifier to aggregate the score of these classifiers. It then gives the best accuracy.
- One of the **disadvantages** of AutoGluon is that it is **expensive to train**

References

- https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205
- https://xgboost.readthedocs.io/en/stable/tutorials/model.html
- https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052
- https://medium.com/@analyttica/gini-coefficient-or-gini-index-in-our-data-science-analytics-platform-d0408fc837
- https://towardsdatascience.com/autogluon-deep-learning-automl-5cdb4e2388ec: :text=AutoGluon
- https://www.ibm.com/topics/decision-trees: :text=A