1) Random-forest characteristics:

- Decision trees are building blocks of random forest model.
- consists of large number of decision trees that operates as an ensemble. Each individual tree spits out a class prediction and class with most votes becomes models prediction.
- Large number of relatively uncorrelated models operating as a committee will outperform any of the individual constituent model
- low correlation is the key.
- The trees protect each other from their individual errors.
- There needs to be some actual signal in our features so that models built using those features do better than random guessing.
- The predictions made by the individual trees need to have low correlations with each other.
- Uses bagging (Decision trees are very sensitive to data they are trained on -small changes to the training set can result in significantly different tree structures.
- Uses feature randomness when building each individual tree to try to create an uncorrelated forest of trees.

— Difference between random forest model and decision tree —

* While a decision tree is a tree-like model of decisions along with possible outcomes in a diagram, random forest is a classification algorithm consisting of many decision trees combined to get a more accurate result as compared to a single tree.

* In decision tree, there is scope of overfitting, while random forest avoids and prevents overfitting by using multiple trees.

* In decision tree, there is no accurate results, while random forest gives accurate and precise results.

* In decision trees require low computation, thus reducing time to implement and carrying low accuracy while random forest consumes more computation, time-consuming.

* To visualize the decision tree is easy but complex visualization in random forest as it determines the pattern behind the data.

~~Pseudocode~~ of random forest classification model:
1. Randomly select "k" features from total "m" features.
2. Among the "k" features, using best split point calculate node "d".

pseudocode of random forest classification model:

1. Select random k data points from the training set.
2. Build the decision trees associated with the selected data points.
3. Choose the number N for decision trees that you want to build.
4. Repeat step 1 & 2
5. For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

for example our data set

id	x_0	x_1	x_2	x_3	x_4	y
0	4.3	4.9	4.1	4.7	5.5	0
1	3.9	6.1	5.9	5.5	5.9	0
2	2.7	6.8	4.1	5.0	5.6	0
3	6.6	4.4	4.5	3.9	5.9	1
4	6.5	2.9	4.7	4.6	6.1	1
5	2.7	6.7	4.2	5.3	4.8	1

Step 1 ->

id
2
0
2
4
5
5

x_0, x_1

id
2
1
3
1
4
4

x_2, x_3

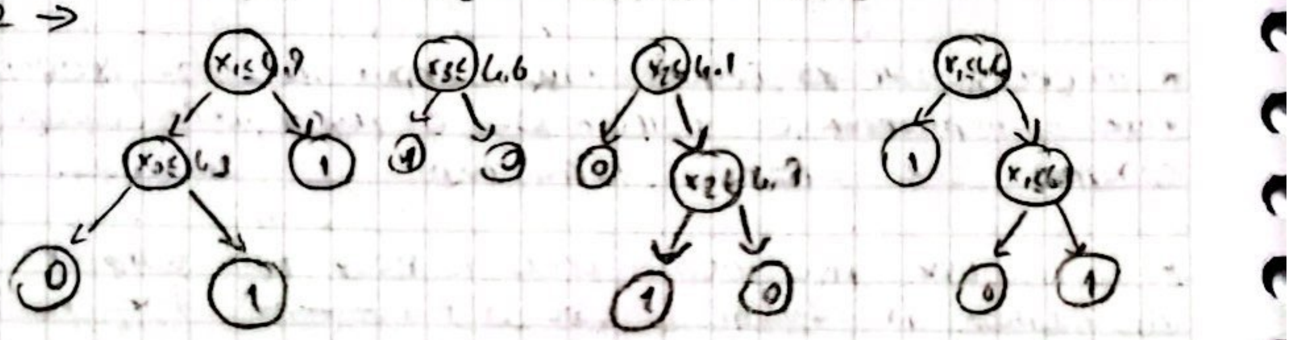
id
4
1
3
0
0
2

x_2, x_4

id
3
3
2
5
1
2

x_1, x_3

step 2 ->



step 3 ->

2,8	6,2	4,3	5,3	5,5

assume step 4 is executed.

step 5 -> for first tree -> 1
" second " -> 0
" third " -> 1
" fourth " -> 1

} winner is 1 for this prediction

2) Transfer learning:

Transfer learning is that machine learning methods store the information obtained while solving a problem and use that information when faced with another problem. With learning transfer, models that show higher success and learn faster with less training data are obtained by using previous knowledge.

- inception model

The goal of this module is act as a multi-level feature extracter by computing 1x1, 3x3 and 5x5 convolution within the same module of network. The output of these filters is then stacked along the channel dimension and before being fed into the next layer in the network.

The architecture of this model:

- 1x1 convolution with 128 filters for dimensions and reductions and rectified linear activations.
- Fully connected layer with 1024 units a rectified linear activation,
- Dropout layer with 70% ratio.
- Linear layer with softmax loss as a classifier.

3) Support vector machine

The objective is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. To separate the two classes of data points, find a plane that has the maximum margin. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence. The dimension of the hyperplane depends upon the number of features.
Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using those, we maximize the margin of the classifier.

- advantages of SVM
  - SVM works relatively well when there is a clear margin of separation between classes.
  - More effective in high dimensional spaces.
  - Relatively memory efficient.

- disadvantages of SVM
  - Not suitable for large data sets.
  - Does not perform very well when the data set has more noise.
  - In cases where number of features for each data point exceeds the number of training data samples, the SVM will underperform.
  - There is no probabilistic explanation for the classification.

4) Fasttext classification

Purpose of this classification is performing tasks of text classification and representation while processing large datasets. Fasttext is a library for efficient learning of word representations and text classification.
Starts with word representations and feed them to a linear classifier. Text representation as a hidden state that can be shared among features and classes.
It uses two methods :
- Hierarchical softmax : Based on Huffman Coding tree used to reduce computational complexity $O(kh)$ to $O(h \log(k))$, where k is the number of classes and h is dimension of text representation.

- word n-grams : Fasttext incorporates a bag of n-grams representation along with word vectors to preserve some information about the surrounding words appearing near each word.

- advantages of fasttext
  - Very fast in comparison to other methods.
  - Sentence vectors can be easily computed.
  - Works better on small datasets.

- disadvantages of fasttext
  - This is not standalone library (require another library).
  - This library has a python implementation. It is not officially supported.

5) Class imbalance: Term that referring to a training dataset in which examples of one class are heavily outnumbered by examples of another class. The minitory class consists of positive instances which signify rare but important occurences. Conversely, the majority class is built of negative instances corresponding with common events.

Techniques

- Oversampling
- Undersampling
- Threshold moving
- Ensemble techniques

- Oversampling

Oversampling works by resampling the positive tuples so that the resulting training dataset contains an equal number of positive and negative tuples.

Suppose the original training set contains 100 positive and 1000 negative tuples. In oversampling, we replicate tuples of the rarer class to form a new training set containing 1000 positive tuples and 1000 negative tuples.

Several variations to oversampling exist. They may vary, for instance, in how tuples are added or eliminated. For example, the SMOTE algorithm uses oversampling where synthetic tuples are added, which are 'close to' the given positive tuples in the tuple space.

Undersampling

Undersampling works by decreasing the number of negative tuples. It randomly eliminates tuples from the majority (negative) class until there are an equal number of positive and negative tuples

Suppose the original training set contains 100 positive and 1000 negative tuples. In undersampling, we randomly eliminate negative tuples so that the new training set contains 100 positive tuples and 100 negative tuples.

Threshold-moving

It applies to classifiers that, given an input tuple, return a continuous output value. That is, for an input tuple, x, such a classifier returns as output a mapping, $f(x) \rightarrow [0,1]$. Rather than manipulating the training tuples, this method returns a classification decision based on the output values. In the simplest approach, tuples for which $f(x) \geq t$, for some threshold, t, are considered positive, while all other tuples are considered negative.

examples of such classifier include naive Bayesian and neural network classifiers like backpropagation

Ensemble methods

Bagging, boosting, and random forests are examples of ensemble methods. An ensemble combines a series of k learned models, $M_1, M_2, \ldots, M_k$, with the aim of creating an improved composite classification model, $M*$. A given data set, D, is used to create k training sets, $D_1, D_2, \ldots, D_k$ is used to generate classifier M_i. Given a new data tuple to classify, the base classifiers each vote by returning a class prediction. The ensemble returns a class prediction based on the votes of the base classifiers.