**Random Forest**

1. **Why Random Forest? Differences from the Decision tree?**

* **Low Bias, Moderate Variance:** Each decision tree has a high variance, but low bias. But because we average all the trees in a random forest, we are averaging the variance as well so that we have low bias and moderate variance model.
* **Quick Prediction/Training Speed:** It is faster to train than decision trees because we are working only on a subset of features in this model, so we can easily work with hundreds of features. Prediction speed is significantly faster than training speed because we can save generated forests for future uses.
* **Great with High dimensionality:** Random forests are great with high dimensional data since we are working with subsets of data.
* **Parallelizable:** They are parallelizable, meaning that we can split the process to multiple machines to run. This results in faster computation time.
* **Impressive in Versatility:** Whether you have a regression or classification task, a random forest is an applicable model for your needs. It can handle binary features, categorical features, and numerical features. There is very little pre-processing that needs to be done. The data does not need to be rescaled or transformed.
* Random forest algorithms can be used for **both classifications and regression** tasks.
* It provides **higher accuracy**.
* Random forest classifier will **handle the missing values** and maintain the accuracy of a large proportion of data.
* If there are more trees, it **won’t allow overfitting** trees in the model.
* It has the power to **handle a large data set with higher dimensionality.**

1. **Random Forest:**

Random forest improves on bagging because it **decorrelates** the trees with the introduction of splitting on a **random subset of features**. **This means that at each split of the tree, the model considers only a small subset of features rather than all of the features of the model. That is, from the set of available features n, a subset of m features (m=square root of n) is selected at random. This is important so that variance can be averaged away.** Consider what would happen if the data set contains a few strong predictors. These predictors will consistently be chosen at the top level of the trees, so we will have very similar structured trees. In other words, the trees would be highly correlated.

* Random sampling of training data points when building trees.
* Random subsets of features considered when splitting nodes.

1. **Applications of Random Forest (real-life):**

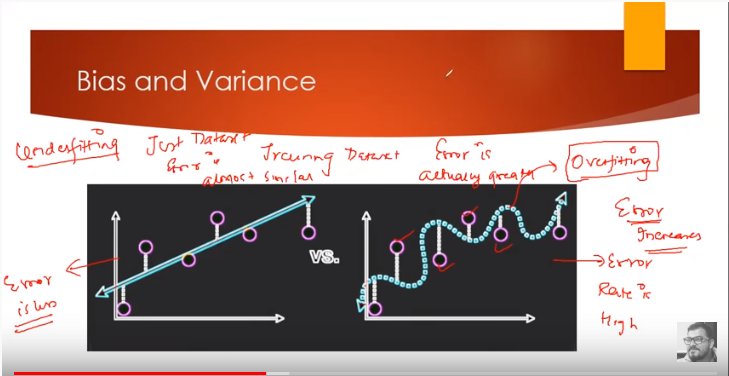
**Banking Sector:** The banking sector consists of most users. There are many loyal customers and also fraud customers. To determine whether the customer is a loyal or fraud, Random forest analysis comes in. With the help of a random forest algorithm in machine learning, we can easily determine whether the customer is fraud or loyal. A system uses a set of a random algorithm which identifies the fraud transactions by a series of the pattern.

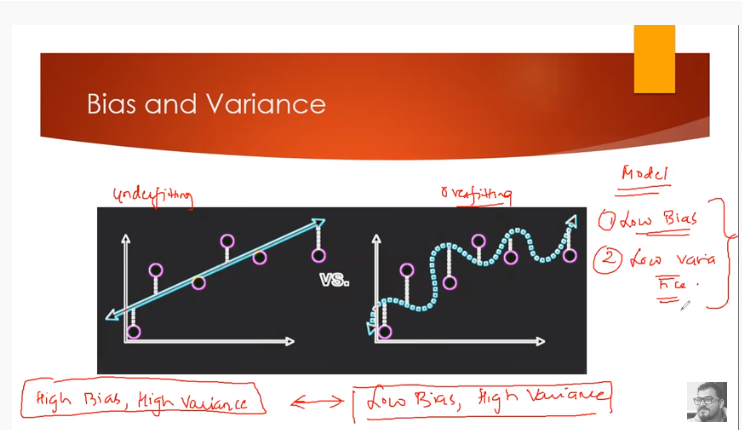
**Medicines:** Medicines need a complex combination of specific chemicals. Thus, to identify the great combination in the medicines, Random forest can be used. With the help of a machine learning algorithm, it has become easier to detect and predict the drug sensitivity of a medicine. Also, it helps to identify the patient’s disease by analyzing the patient’s medical record.

1. **How Does Random Forest work**:

In the random forest, we grow multiple trees in a model. To classify a new object based on new attributes each tree gives a classification and we say that tree votes for that class. The forest chooses the classifications having the most votes of all the other trees in the forest and takes the average difference from the output of different trees. In general, Random Forest built multiple trees and combines them together to get a more accurate result.

1. **Bias and Variance:**

****

****

* **High Bias + High Variance: Underfitting**
* **Low Bias and High Variance:** **Overfitting** ( this is where we use Random Forest to minimize the variance by splitting the data into chunks of features/data and train it.

****

1. **Jupyter Notebooks of RF :**

* Random\_Forest\_testing
* Random\_Forest\_Classifier

1. **Random Forest Example :**

**Very Useful:**

[**https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76**](https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76)

[**https://towardsdatascience.com/random-forest-in-python-24d0893d51c0**](https://towardsdatascience.com/random-forest-in-python-24d0893d51c0)

# O**verfitting: Or Why a Forest is better than One Tree:**

**Impo.:** You might be tempted to ask why not just use one decision tree? It seems like the perfect classifier since it did not make any mistakes! A critical point to remember is that the tree made no mistakes on the training data. We expect this to be the case since we gave the tree the answers and didn’t limit the max depth (number of levels). The objective of a machine learning model is to generalize well to new data it has never seen before.

Overfitting occurs when we have a [very flexible model](http://qr.ae/TUNozZ) (the model has a high capacity) which essentially memorizes the training data by fitting it closely. The problem is that the model learns not only the actual relationships in the training data but also any noise that is present.

The reason the decision tree is prone to overfitting when we don’t limit the maximum depth is that it has unlimited flexibility, meaning that it can keep growing until it has exactly one leaf node for every single observation, perfectly classifying all of them. If you go back to the image of the decision tree and limit the maximum depth to 2 (making only a single split), the classifications are no longer 100% correct. We have reduced the variance of the decision tree but at the cost of increasing the bias.

As an alternative to limiting the depth of the tree, which reduces variance (good) and increases the bias (bad), we can combine many decision trees into a single ensemble model known as the random forest.

1. **Bagging(**[**bootstrap aggregating**](https://en.wikipedia.org/wiki/Bootstrap_aggregating)**) in Random Forest :**

* Bootstrap Aggregation (or Bagging for short), is a simple and very powerful ensemble method.
* Bagging (Bootstrap Aggregation) is used when our goal is to **reduce the variance** of a decision tree. Here idea is to create several subsets of data from the training samples chosen randomly with replacement. Now, each collection of subset data is used to train their decision trees. As a result, we end up with an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree.
* An ensemble method is a technique that combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model.
* Bootstrap Aggregation is a general procedure that can be used to reduce the variance for those algorithms that have high variance. An algorithm that has high -variance is decision trees, like classification and regression trees (CART).
* An ensemble method is a technique that combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model.
* Bootstrap Aggregation is a general procedure that can be used to reduce the variance for those algorithms that have high variance. An algorithm that has high variance are decision trees, like classification and regression trees (CART).
* Decision trees are sensitive to the specific data on which they are trained. If the training data is changed (e.g. a tree is trained on a subset of the training data) the resulting decision tree can be quite different and in turn, the predictions can be quite different.
* Bagging is the application of the Bootstrap procedure to a high-variance machine learning algorithm, typically decision trees.
* The only parameters when bagging decision trees is the number of samples and hence the number of trees to include. This can be chosen by increasing the number of trees on run after run until the accuracy begins to stop showing improvement (e.g. on a cross-validation test harness). Very large numbers of models may take a long time to prepare, but will not overfit the training data.

1. **Ensemble methods**, which combines several decision trees to produce better predictive performance than utilizing a single decision tree. The main principle behind the ensemble model is that a group of weak learners come together to form a strong learner.

Few techniques to perform ensemble decision trees:

**1. Bagging (explained above)**

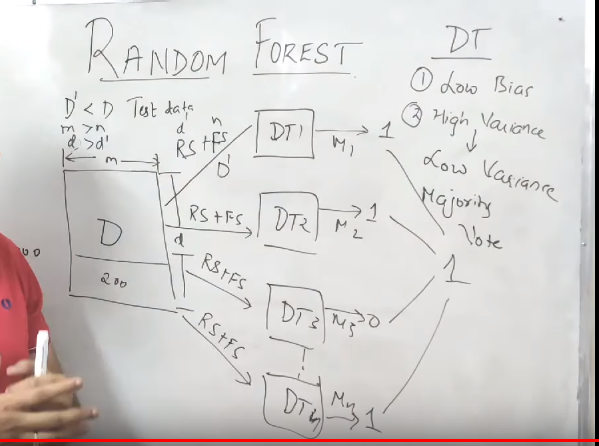
**2. Boosting**

1. **Important Note:** Random Forest is an extension over bagging. It takes one extra step where in addition to taking the random subset of data, it also takes the random selection of features rather than using all features to grow trees. When you have many random trees. It’s called Random Forest.
2. **Disadvantages of using Random Forest technique**:

Since the final prediction is based on the mean predictions from subset trees, it won’t give precise values for the regression model.

1. Links to follow: 1) <https://towardsdatascience.com/decision-tree-ensembles-bagging-and-boosting-266a8ba60fd9>

2) <https://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/>



**“Use hyperparameter optimization to choose the number of trees in forest”**

**In the diagram : RS** is random sampling of data with replacement and **FS** is feature sampling.