**Machine Learning Algorithms: Advantages and Disadvantages**

First off, we need to be clear what exactly we mean by advantages. People have argued the relative benefits of trees vs. logistic regression in the context of [interpretability](https://www.quora.com/What-makes-a-model-interpretable), robustness, etc. But let’s assume for now that all we care about is out of sample predictive performance. Again, you may need to specify what kind of predictive performance you need: accuracy, ranking, probability estimation!

**Logistic regression**:

Pros:

* It is more robust: the independent variables don’t have to be normally distributed or have equal variance in each group.
* It does not assume a linear relationship between the IV and DV
* There is no homogeneity of variance assumption

Cons:

* Logistic regression tends to underperform when there are multiple or non-linear decision boundaries. They are not flexible enough to naturally capture more complex relationships.

**K-NN classifiers**:

Pros:

* It’s non-parametric since it does not make any assumption on data distribution (the data does not have to be normally distributed). It simply classifies objects based on feature similarity (feature = input variables). Classification is computed from a simple majority vote of the k nearest neighbors of each point
* This algorithm is simple to implement, robust to noisy training data, and effective if training data is large

Cons:

* It is lazy since it does not really learn any model and make generalization of the data (It does not train some parameters of some function where input X gives output y).
* Need to determine the value of K and the computation cost is high.

**Support Vector Machines (SVM):** It uses orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables

Pros:

* SVMs provide a good out-of-sample generalization, if the parameters C and r (in the case of a Gaussian kernel) are appropriately chosen. This means that, by choosing an appropriate generalization grade, SVMs can be robust, even when the training sample has some bias.
* It is effective in high dimensional spaces and is effective in cases where number of dimensions is greater than the number of samples.

Cons:

* When the target variable is overlapping i.e. data has noise then SVM fails.
* Takes a bit of time if data is high dimension.
* It is prone to overfit if not trained properly and if parameters are not tuned properly.

**Decision Trees**:

Pros:

* Can handle missing values.
* Able to handle both categorical and continuous data.
* Wont effect by outliers.
* Easy to interpret.

Cons:

* Overfitting occurs when the algorithm captures noise in the dataset.
* The prediction model gets unstable with a very small variance in data.
* A highly complicated Decision tree tends to have a low bias which makes it difficult for the model to work with new data.

**Random Forest:**

Pros**:**

* A single decision tree tends to overfit the data. The process of averaging or combining the results of different decision trees helps to overcome the problem of overfitting.
* extremely flexible and have very high accuracy.
* They also do not require preparation of the input data. You do not have to scale the data. It also maintains accuracy even when a large proportion of the data are missing.
* They provide a reliable feature importance estimate
* They offer efficient estimates of the test error without incurring the cost of repeated model training associated with cross-validation

Cons:

* They are much harder and time-consuming to construct than decision trees
* For data including categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels. Therefore, the variable importance scores from random forest are not reliable for this type of data. Methods such as partial permutations were used to solve the problem.

**Naïve Bayes Classifier:**

Pros:

* Computationally fast and easy to interpret
* Works well with high dimensions
* If the NB conditional independence assumption holds, a Naive Bayes classifier will converge quicker than discriminative models like logistic regression, so you need less training data.

Cons:

* suffer multicollinearity
* no variable dependency, which is much unjustified for the real-life data
* Relies on independence assumption and will perform badly if this assumption is not met.

**Stochastic Gradient Boosting trees (GBM)**

Pros**:**

* Like bagging but learns sequentially and builds off previous trees. It builds trees one at a time, where each new tree helps to correct errors made by previously trained tree. With each tree added, the model becomes even more expressive.
* A great application of GBM is anomaly detection in supervised learning settings where data is often highly unbalanced such as DNA sequences, credit card transactions or cyber security.
* It performs the optimization in function space (rather than in parameter space) which makes the use of custom loss functions much easier. Boosting focuses step by step on difficult examples that gives a nice strategy to deal with unbalanced datasets by strengthening the impact of the positive class.
* Since boosted trees are derived by optimizing an objective function, basically GBM can be used to solve almost all objective function that we can write gradient out

Cons:

* GBMs are more sensitive to overfitting if the data is noisy.
* GBMs are harder to tune than RF. There are typically three parameters: number of trees, depth of trees and learning rate, and each tree built is generally shallow.

Here is the result from a small case study of different model performance: **Comparison of Machine Learning Classification Models for Credit Card Default Data** (Naveen Krishna, Data Scientist Altair). We will focus on the time taken to train the model and the accuracy to measure the models performance



