**Machine Learning Algorithms Cheat Sheet**

A quick reference guide to 5 common algorithms



Photo by [Caleb Jones](https://unsplash.com/@gcalebjones?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) on [Unsplash](https://unsplash.com/s/photos/choice?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)

The “***no free lunch”*** theorem for machine learning states that there is no single machine learning algorithm that can solve all types of machine learning problems.

Machine learning tasks can vary widely and the choice of algorithm will depend on things such as the size, dimensionality and sparsity of the data. The target variable, the quality of the data and the interactions and statistical relationships that exist both within features and between the features and target variable.

As a result, it is not possible to simply select one algorithm for a one-size-fits-all approach. Depending on exactly how they work different algorithms will be better suited to certain tasks. Data scientists will typically select the final algorithm to use by first determining a sub-set of algorithms appropriate for the specific problem and then experimenting with these algorithms to find the optimal choice.

In this article, I will provide a quick reference guide to five of the most common algorithms used for machine learning. This will provide an introduction to the inner workings of the algorithms and the considerations that make each algorithm better suited to certain tasks.

This will include a brief introduction to Linear Regression, Logistic Regression, Random Forest, XGBoost and K-means. For each algorithm I will cover the following:

1. **How the algorithm works.**
2. **An example code implementation.**
3. **Guidance on appropriate situations to use the algorithm.**
4. **Advantages and disadvantages.**

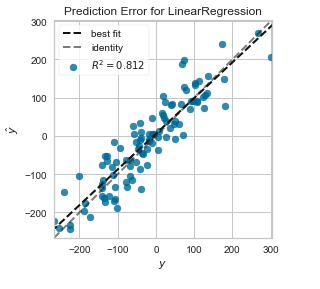
**1. Linear Regression**

Linear regression is a supervised machine learning algorithm that is used to predict a continuous target variable. For simple linear regression, where there is one **independent variable**(feature) and one **dependent variable**(target) the algorithm can be represented by the following equation.

y = a + bX

Where **y** is the dependent variable, **X** is the explanatory variable, ***b*** is the slope of the line and ***a*** is the intercept.

Simple linear regression can be visualised as a scatter plot where the x-axis contains the dependent variable and the y-axis contains the independent variable. The linear regression algorithm draws a line of best fit through the data points minimising the variation between the predicted and actual outputs.



Linear resgression. Image by Author

**Code example**

**When should it be used?**

* Linear Regression can only be used to solve regression-based problems.
* There must be a linear relationship between the dependent and independent variables.
* Residuals must form a normal distribution.
* There must be no correlation between features.
* The algorithm assumes that training data is randomly sampled.
* Best suited to regression-based problems where the relationships in the data are both linear and simple.

**Advantages**

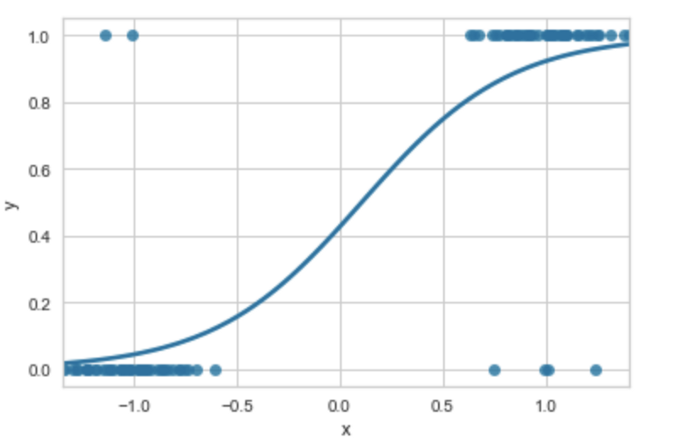
* Highly interpretable and fast to train.
* Performs very well on linearly separable data.

**Disadvantages**

* Not robust to outliers.
* Very simplistic and so it doesn't model the complexities found in real-world data well.
* This algorithm is also prone to overfitting.

**2. Logistic Regression**

Logistic regression is essentially linear regression moulded to fit a classification problem. Instead of fitting a straight line, logistic regression applies the[**logistic function**](https://en.wikipedia.org/wiki/Logistic_function) to squeeze the output of a linear equation between 0 and 1. The result is an S-shaped curve rather than a straight line through the data points, illustrated in the diagram below.



A Logistic Regression Curve. Image by Author

A threshold between 0 and 1 is chosen to separate the classes, typically this is 0.5. In essence, we draw a horizontal line across the S curve at 0.5. Any data points above this line belong to class 1 and any below to class 0.

**Code example**

**When should it be used?**

* This algorithm can only be used to solve classification problems.
* There must be a linear relationship between features and the target variable.
* The number of observations must be larger than the number of features.
* Best suited to classification problems where the relationships in the data are both linear and simple.

**Advantages**

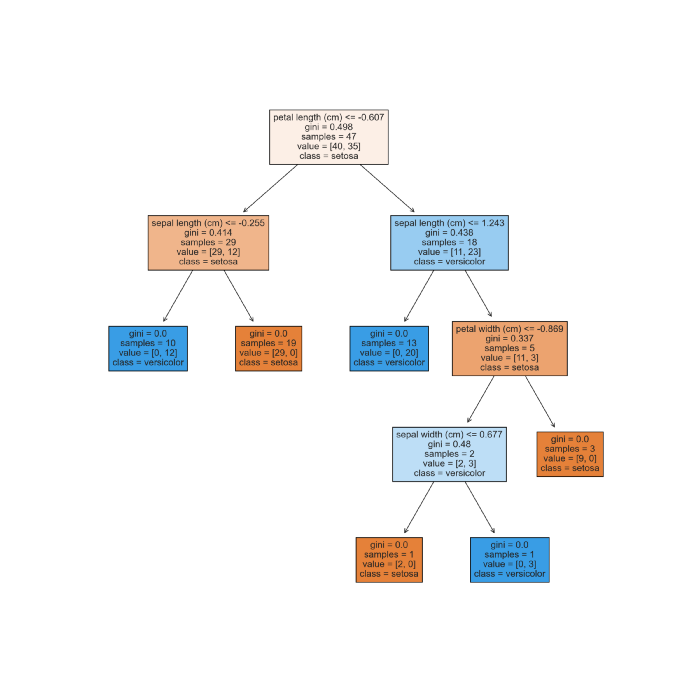
* As with linear regression, this algorithm is highly interpretable and fast to train.
* It performs very well on linearly separable data.

**Disadvantages**

* Prone to overfitting.
* As with Linear Regression, it does not model complex relationships well.

**3. Random Forest**

The random forest algorithm builds a ‘forest’ of **decision trees**. Each tree in the forest generates a prediction from a given set of features. Once all predictions have been generated a majority vote is taken and the most commonly predicted class forms the final prediction.



A single decision tree generated by Random Forest. Image by Author

A decision tree is a very intuitive algorithm. It has a flowchart-like structure containing a series of nodes representing a test. The outcome of each test results in a split and one or more **leaf nodes** are created until the final prediction is achieved. Hyperparameters determine how deep the tree grows and the node-splitting function used.

The Random Forest algorithm follows the below sequence of steps:

1. The training dataset is randomly split into multiple samples based on the number of trees in the forest. The number of trees is set via a hyperparameter.
2. The decision trees are trained in parallel using one of the data subsets.
3. The output of all trees is evaluated and the most commonly occurring prediction is taken as the final result.

**Code example**

**When should it be used?**

* This algorithm can be used to solve both classification and regression-based problems.
* It is particularly well suited to large datasets with high dimensionality as the algorithm inherently performs feature selection.

**Advantages**

* It can model both linear and non-linear relationships.
* It is not sensitive to outliers.
* Random Forest is able to perform well on datasets containing missing data.

**Disadvantages**

* Random Forest can be prone to overfitting although this can be mitigated to some degree with **pruning**.
* It is not as interpretable as linear and logistic regression although it is possible to extract feature importances to give some level of interpretability.

**4. XGBoost**

XGBoost is an algorithm based on**gradient-boosted** decision trees. It is similar to Random Forest in that it builds an ensemble of decision trees but rather than training the models in parallel, XGBoost trains the models sequentially. Each decision tree learns from the errors produced by the previous model. This technique of training models sequentially is known as**boosting**.

The gradient in XGBoost refers to a specific type of boosting where **weak learners** are used. Weak learners are very simple models that only just perform better than random chance. The algorithm starts with an initial weak learner. Each subsequent model targets the errors produced by the previous decision tree. This continues until no further improvement can be made and results in a final strong learner model.

**Code example**

**When should it be used?**

* It can be used to solve both classification and regression-based problems.
* XGBoost is generally considered one of the best and most flexible algorithms for supervised learning on structured data and is therefore suited to a wide range of datasets and problem types.

**Advantages**

* XGboost is highly flexible in that it works equally well on small and large datasets.
* It is computationally efficient and therefore faster to train models compared to other complex algorithms.

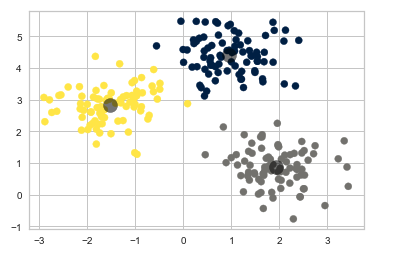
**Disadvantages**

* It does not work as well on very sparse or unstructured data.
* It is considered a black box model and is less interpretable than some other algorithms.
* XGBoost can be sensitive to outliers due to the mechanism of models learning from the errors of their predecessors.

**5. K Means**

K-means is one of the most popular algorithms for clustering, a form of unsupervised machine learning which aims to find groups of similar examples within a training dataset.

The algorithm works by first initialising random cluster centroids. Then for each datapoint a distance measure commonly the [**Euclidean**](https://en.wikipedia.org/wiki/Euclidean_distance) distance or [**Cosine**](https://en.wikipedia.org/wiki/Cosine_similarity) similarity is used to assign it to the nearest centroid. Once all data points are assigned the centroid is moved to the mean of the assigned data points. These steps are repeated until the centroid assignment ceases to change.



K-means clusters with centroids visualised. Image by Author

**Code example**

**When should it be used?**

* K-means is only suited to unsupervised clustering.
* It is generally considered a good all-rounder algorithm for these types of problems.

**Advantages**

* It is a relatively simple algorithm to implement.
* It can be used on large datasets.
* The resulting clusters are easy to interpret.

**Disadvantages**

* K-means are sensitive to outliers.
* This algorithm does not find the optimal number of clusters. This has to be determined by other techniques prior to implementation.
* The results of the clustering are not consistent. If K-means is run on a dataset multiple times it can produce different results each time.

The algorithms covered in this article are some of the most widely used for machine learning. There are many more algorithms available each of these having its own suitability for specific problems. As mentioned at the start of this article there is no one algorithm that can solve all problems currently although XGBoost is the closest thing to it for solving problems with structured data.

This introduction will provide some insight into how and why a data scientist may select one algorithm over another. Here is a brief summary of the suitability of each algorithm to specific types of problems:

* **Linear Regression:**Best suited to solving regression-based problems for data sets where there is a linear relationship present and the relationship is relatively simplistic.
* **Logistic Regression:** Best suited to solving classification problems where the data is linearly separable and the dimensionality of the dataset is low.
* **Random Forest:** Best for large, high-dimensional datasets with complex relationships.
* **XGBoost:**Suitable for a wide range of structured datasets and problems. Computationally more efficient compared to Random Forest.
* **K-means:** Best suited to solving unsupervised clustering problems.

Thanks for reading!