The Most Important Machine Learn Algorithms

Pros and cons of the top 11 algorithms every machine learning engineer should know

Posted by Tobias (https://twitter.com/semanti_ca) on 03-09-2018

There are plenty of machine learning algorithms. Some of them are v useful for solving some very specific problems. Others could be appl solve a wide range of tasks.

In this post, we list the algorithms most frequently used by machine engineers at semanti.ca (https://semanti.ca) as well as those that oft Kaggle (https://www.kaggle.com) competitions.

Classification and Regression

Gradient Boosting Machine

Gradient Boosting Machine is currently (as of 2018) the most popular algorithm among Kagglers and most winning teams used Gradient B Machine in their solutions.

Pros:

- Can handle huge datasets (millions of examples and millions of dimensions);
- Extremely accurate;
- Can be used for both classification and regression tasks;
- Lots of flexibility with the choice of loss functions: can be tailore characteristics of the problem.

Cons:

- Can be slow at training, since the trees are built sequentially;
- Model is a black box (as for all ensemble methods);
- Prone to overfitting (however hyperparameter tuning helps).

Random Forest

Random Forests are still frequently used by the machine learning practitioners. They provide good accuracy and speed of training and handle big datasets.

Pros:

- Much easier to tune than Gradient Boosting Machine (two hyperparameters vs three;
- Almost always perform as well as or better than SVMs;
- Deal well with uneven datasets that have missing variables;
- Rarely overfits.

Cons:

- Slow at prediction time so less appropriate for high-speed data

processing;
- If data includes categorical variables with different numbers of l Random Forest is biased in favor of those with more levels. Ther the variable importance scores from Random Forest are not relia

Support Vector Machine

Support Vector Machines is a mature and well-studied machine lear algorithm, with a solid theoretical foundation. It supports kernels, sc handle non-linearly separable classification problems.

Pros:

- Handles both classification and regression;
- Supports very high-dimensional and sparse data (hundreds of th of dimension); highly effective in text classification where high-s and dimensionality are frequently observed in data.
- High accuracy, good theoretical guarantees regarding overfittin
- Handles very well outliers and noise.
- Extremely fast at prediction time;
- Generalize well from few data.

Cons:

- Cannot be parallelized, the whole dataset has to fit in memory;
- Not good at handling very big dataset (more than a hundred of thousands of examples);
- Sometimes hard to tune due to a wide range of possible hyperpa values.

LSTM Neural Network

LSTM Neural Network is a variant of the Recurrent Neural Network.

units effectively solve the problem of vanishing gradient. They are ty
used for sequential classification problems: text labeling, speech rec
The Encoder-Decoder architecture of LSTM networks allows buildin
machine translation systems.

Pros:

- Classify relatively long (up to 40-50 tokens) sequences well;
- Highly accurate even with a couple of hidden layers.

Cons:

- Very slow at training as cannot be parallelized between multiple
- Relatively slow at classification, since it classifies one token at a (especially the Encoder-Decoder architecture);
- Black boxes without strong theoretical foundations.

Convolutional Neural Network

As of 2018, one cannot imagine the image processing without Convol
Neural Networks. Such state-of-the-art CNN architectures as AlexN
GoogLeNet, and ResNet are all variants of a Convolutional Neural Ne

Pros:

- Very fast to train, as can be parallelized between multiple GPUs;
- Learn low-level and high-level features so can be used as featur learners for other ML algorithms;
- Very accurate at image classification and many other tasks, incl language processing.

Cons:

- Require a lot of training data (however, such techniques as **trans learning** can reduce the need for data);
- Sensible to parameter initialization and choice of hyperparamet
- Slow to train on CPU;
- Black boxes without strong theoretical foundations (as of 2018).

K-Nearest Neighbors

K-Nearest Neighbors is one of the oldest and simplest algorithms of learning. In fact, there's not so much learning happening: the predict given as the majority (classification) or the average (regression) of th in the close neighborhood of the input, unannotated example. KNN very accurate recently, however, it was for a long time considered as prediction time. However, modern libraries made this algorithm very today it can easily compete with the state-of-the-art techniques.

Pros:

- No assumptions about data: works well for non-linearly separab
- Simple to implement;
- Flexible to feature/distance choices;
- Naturally handles multi-class cases;
- Does well in practice with enough representative data
- Can be used for both classification and regression.

Cons:

- Search in a large space of examples to find nearest neighbors ca slow (mostly resolved in modern libraries);
- With a lot of training examples, the model can require a lot of m
- Sensitive to irrelevant features and the scale of the data.

Clustering and Dimensionality Reduction

K-Means

The K-means algorithm remains one of the most used clustering algo despite its simplicity.

Pros:

- Simple to use;
- Fast (when used with an appropriate data structure to store exa that preserve spatial locality);
- Results are simple to understand by a human.

Cons:

- K-means only works well when the shape of clusters are hyper-s or when clusters are far from one another;
- Different runs of the algorithm will most probably yield in differe clusterings of the same data;
- Requires the number of clusters to be known in advance.

Expectation Maximization

Expectation Maximization, as well as Latent Dirichlet Allocation (belo both examples of the so-called "soft" clustering. They assign one exa multiple clusters with a probability of membership. Expectation Maximization learns Gaussian Mixture Models which solves the prob K-Means: the clusters can be non-spherical.

Pros:

- Simplicity and ease of implementation;
- Solutions to the M-steps often exist in the closed form;
- Can often be easily parallelized and its memory requirements te modest compared to other methods;
- The algorithm is numerically very stable;
- Soft-membership in clusters is often desirable;
- Can be naturally used for finding outliers.

Cons:

- Slow linear convergence;
- Can converge to local optima;
- As in K-Means, one needs to know the number of clusters;
- One needs to carefully choose the generative model.

DBSCAN

DBSCAN (for *Density-Based Spatial Clustering of Applications with No* another popular clustering algorithm. Its difference from K-Means li fact that DBSCAN doesn't use the notion of cluster centroids. It's der based, so the number of clusters depends on the data itself. Clusters have absolutely any form: for DBSCAN a cluster is a subset of points form a dense "cloud".

Pros:

- Doesn't need the number of clusters as input;
- Clusters can have arbitrary shapes;
- Robust to noise;
- Deterministic.

Cons:

- Requires connected regions of sufficiently high density;
- The maximal distance between two points that belong to one clu to be given as input;
- Sensitive to hyperparameters;
- Sampling affects density measures.
- Datasets with varying densities are problematic.

Latent Dirichlet Allocation

LDA is one the most important unsupervised learning algorithms. It successfully applied to text analysis (topic modeling), social network (community overlaps), content recommendation, genetic population and many others.

In the nutshell, LDA takes a collection of documents, the number of and returns a distribution of topics over documents and a distributio words over topics.

LDA can be used as a soft-clustering method: every topic can be seen cluster and each document can have several topics with different probabilities, thus can belong to several clusters.

Pros:

- Excellent empirical results;
- Mitigates overfitting well;

Cons:

- High computational complexity;
- Nondeterministic (different runs can give different results);
- Doesn't work well on short documents.

UMAP

One important aspect of the machine learning practice is data visual and dimensionality reduction. Dimensionality reduction can be done visualize data (the human can only see up to three-dimensional data) make learning tractable and/or more accurate.

For a long Principal Component Analysis (https://en.wikipedia.org /wiki/Principal_component_analysis) or PCA was the most popular dimensionality reduction and t-SNE (https://en.wikipedia.org/wiki/ distributed_stochastic_neighbor_embedding) was used to visualize However, in 2018 the new algorithm, UMAP (https://github.com/lmc /umap) (for *Uniform Manifold Approximation and Projection*) was pro and efficiently implemented. UMAP combines the advantage of both t-SNE: it can be used as a dimensionality reduction techniques (t-SN and has visualization properties similar or surpassing those of t-SNE many tasks.

Pros:

- Can be used for both dimensionality reduction and visualization
- Has very fast implementation in multiple programming language including Python.

Cons:

- Has hyperparameters one has to tune to find a good visualizatio

These were the 11 most important machine learning algorithms, acco semanti.ca (https://www.semanti.ca) engineers and scientists. Did w something important? Please, let us know and we will improve this te

Read our previous post "Data Science Work Productivity Tips & Trick (https://semanti.ca/blog/?data-science-work-productivity)" or subs (https://semanti.ca/blog/?feed) to our RSS feed.

Like it? Share it!

(https://twitter.com/home?status=https://www.semant /blog/?the-most-important-machine-learning-algorithms)
(https://www.facebook.com/sharer/sharer.php?u=http //www.semanti.ca/blog/?the-most-important-machine-learni algorithms)
(https://plus.google.com/share?url=https://www.seman /blog/?the-most-important-machine-learning-algorithms)
(https://www.linkedin.com/shareArticle?mini=true&url=h //www.semanti.ca/blog/?the-most-important-machine-learni algorithms)
(http://www.reddit.com/submit?url=https://www.seman /blog/?the-most-important-machine-learning-algorithms)

(/)