Machine Learning

What is Machine Learning

Machine learning has recently been attracting attention in the media for several reasons, mainly because it has achieved impressive results in various cognitive tasks such as image classification, natural language understanding, customer churn prediction etc. However, it has been regarded as some sort of magic formula that is capable of predicting the future, but what really is machine learning. Machine learning at its simplest form is all about making computers learn from data by improving their performance at a specific task through experience. Similar to the way humans learn by trying out new things and learning from the experience, machine learning algorithms improve their capability by learning patterns from lots of examples. The performance of these algorithms, generally improves as they are exposed to more data (experience). Machine learning is therefore a branch of artificial intelligence that aims to make machines capable of performing specific tasks without being explicitly programmed. What this means is that these algorithms are not rule-based, the entire learning process is constructed in such a way as to minimize or completely eliminate human intervention.

Machine learning algorithms are typically used for a wide range of learning problems such as classification, regression, clustering, similarity detection etc. Many applications used in the real world today are powered by machine learning. Applications such as personal assistants on mobile phones use machine learning algorithms to understand voice commands spoken in natural language, mobile keyboards predict the next word a user is typing based on previous words, email clients offer a smart reply feature whereby the content of an email is scanned and appropriate responses are generated, e-commerce applications offer recommendation to users based on previous purchases and spending habits etc. Nearly every industry would be impacted by machine learning as most processes can be automated given that there is enough training data available. Machine learning algorithms mostly excel in tasks where there is a clear relationship between a set of inputs and outputs which can be modelled by training data. Although machine learning is a rapidly improving field, there is as of now no notion of general intelligence of the form displayed by humans. This is because models trained on one task cannot generalize the knowledge gleaned to perform another task, that is machine learning algorithms learn narrow verticals of tasks.

There are three main branches of machine learning namely - supervised learning, unsupervised learning and reinforcement learning. In some cases a fourth branch is mentioned - semi-supervised learning but this is really just a special instance of supervised learning as we would see in the explanations below.

Supervised Learning Algorithms

Supervised learning is by far the most common branch of machine learning. Most of the real world value currently in the field of machine learning can be attributed to supervised learning. Supervised learning algorithms are those machine learning algorithms which are trained with labelled examples. It would be remembered that we defined machine learning as making algorithms that learn from data (examples) without being explicitly programmed. The main intuition to understand when dealing with supervised learning algorithms is that, they learn through the use of examples that are clearly annotated to show them what they are supposed to learn. The algorithms therefore try to find a mapping representation from inputs to outputs using the labels as a guide. “Supervised” in the name of these types of algorithms, point to the fact that the labels or targets provide supervision throughout the learning process. It is therefore possible for the algorithm to check its prediction against actual values stored in the labels. It then uses this error information (how far off its prediction was from the actual label) to slowly improve its performance with each iteration. The targets in a supervised learning problem can be seen as a supervisor providing feedback to the algorithm on areas where it can improve its performance. The two main applications of supervised learning algorithms are classification and regression.

Classification involves training a learning algorithm to correctly separate examples into predefined categories or classes. The classes are usually chosen ahead of time by a human expert with domain knowledge in the field where the learning problem is posed. The examples that are used to train the model are clearly labelled to indicate the category they belong to. During training, the supervised learning algorithm, uses the labels to guide its learning and at test time it is capable of correctly predicting the categories of new examples. A popular example of classification is spam detection where an email is correctly identified to belong to one of two classes - spam or not spam. Depending on what is predicted, appropriate action could be taken such as shifting spam emails to a spam folder while relevant emails are sent to a user's inbox.

Regression is a learning problem where the algorithm is interested in predicting a single real value number. Regression is used where a single numeric entity is to be predicted. An example of regression would be predicting the age of a person given a profile picture or predicting the salary of an individual given information about the individual such as level of education, work experience, age, country of residence etc. It would be observed that in both cases the final prediction is a single number.

Supervised learning algorithms are easier to train when compared to unsupervised or reinforcement learning algorithms. This is because the presence of labels simplify the learning problem since there exist a clear way of determining performance during training. However, it should be noted that most supervised learning problems can also be modelled as unsupervised if we get rid of labels. Datasets for supervised learning are more expensive to acquire as it requires meticulous human annotation of examples. The fact that most data in the world today are in an unlabelled form makes the research of unsupervised learning algorithms particularly important.

Unsupervised Learning Algorithms

Unsupervised learning involves learning directly from raw data. This type of learning takes place without the presence of a supervisor in the training loop in form of labels. Unsupervised learning algorithms are free to explore the underlying data distribution and come up with patterns that best describe the entire dataset. The training process is not guided by humans through labelled examples and as such unsupervised learning algorithms are more powerful as they can discover patterns which domain experts may not have thought of. It is however still the job of domain experts to understand the patterns so discovered and explain them because unsupervised learning algorithms do not truly have the sense of reasoning which we would ascribe to humans. Unsupervised learning algorithms merely use the data distribution or its latent (hidden) representations to unearth insights which may be in the form clusters, groups or distributions.

There are many applications of unsupervised learning algorithms such as clustering, dimensionality detection, generative models etc. Clustering is one of the popular implementations of unsupervised learning. It involves the automatic discovery of groups (clusters) of data points from raw data. Members of a group share similar features, that is they are alike. They can be thought of as belonging to the same type of entity whereas as a group they are dissimilar to other groups. Groups usually have semantic meaning which can be further explored to understand the dataset. An example of clustering would involve grouping customers of a tv streaming service into the type of shows that they watch. Users with similar interests would generally be found in the same group. This is a very powerful application as new tv shows could be recommended to users based on other users who share their interests, leading to greater engagement on the platform and increased revenues.

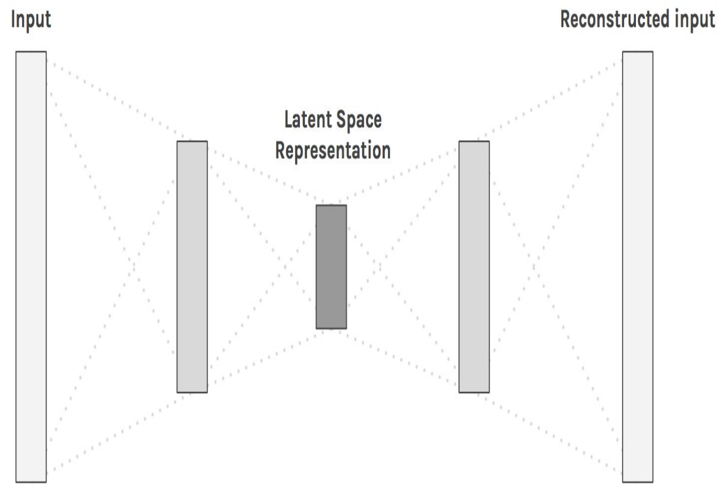
Dimensionality reduction is a machine learning technique that reduces the number of attributes (features) fed in a model to only the most relevant ones which drive predictions. It has been observed empirically, that models with greater number of features or dimensions perform worse on generalization.

That is to say that their performance suffers in the real world. By reducing the number of dimensions, a model can learn from informative features which enables it to develop valid representations about the data that aids prediction. Principal component Analysis (PCA) is probably the most popular dimensionality reduction technique and is an example of an unsupervised learning algorithm. PCA reduces the dimensions of data by identifying those axes that contain the most variability. What that means is that it discovers principal components which offer the most discriminative features.

Generative models are another popular instance of unsupervised learning algorithms. They have made recent headlines because of their ability to artificially generate photographs and works of art that look realistic. Generative Adversarial Networks (GANs) currently produce state of the art results across many image generation benchmarks and are among the most popular examples of generative models.

Semi-supervised Learning Algorithms

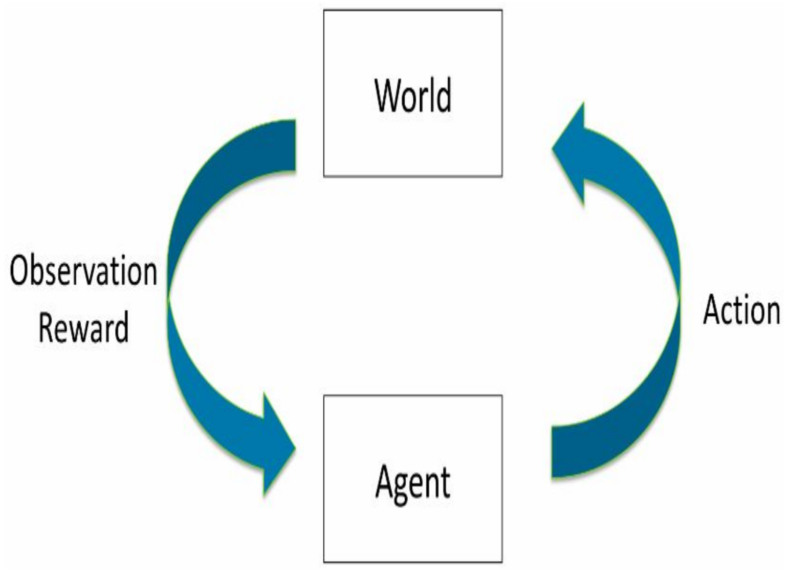
Semi-supervised learning algorithms are a special case of supervised learning algorithms. In semi-supervised learning, while there isn't an explicit label, there exist an implicit heuristic which serves as a supervisor in the training loop. Semi-supervised models do not contain any external source of labels but only rely on input features. However, the learning task is set up in such a way that supervision still takes place in the form of extraction of pseudo-labels from inputs through a heuristic algorithm. A popular example of semi-supervised learning algorithms are autoencoders. Let us look at an example to expand our understanding.



In the autoencoder above, the learning task is to reduce the dimensions of the input into a smaller latent space representing the most important hidden features, then reconstruct the input from this lower dimensional space. So given an input, example an image, an autoencoder shrinks the image into a smaller latent representation that still contains most of the information about the image, then reconstructs the original input image from this low dimensional space. Even if there are no explicit labels, it would be observed that the input serves as the supervisor since the learning task is to reconstruct the input. Once such a model is trained to compress features into a smaller dimension, the compressed features can serve as the starting point of a supervised learning algorithm similar to dimensionality reduction using PCA. The first part of the network that reduces the dimensions of the input is called an encoder while the second part that scales the encoded features back to the full size input is called the decoder.

Reinforcement Learning Algorithms

In reinforcement learning there are three main components, an agent, an environment and actions. The goal of reinforcement learning is to train an intelligent agent that is capable of navigating its environment and performing actions that maximizes its chances of arriving at some end goal. Actions carried out by the agent change the state of the environment and rewards or punishment may be issued based on the actions taken by the agent. The challenge is for the agent to maximize the accumulated rewards at the end of a specific period so that it can actualize an end goal (objective).

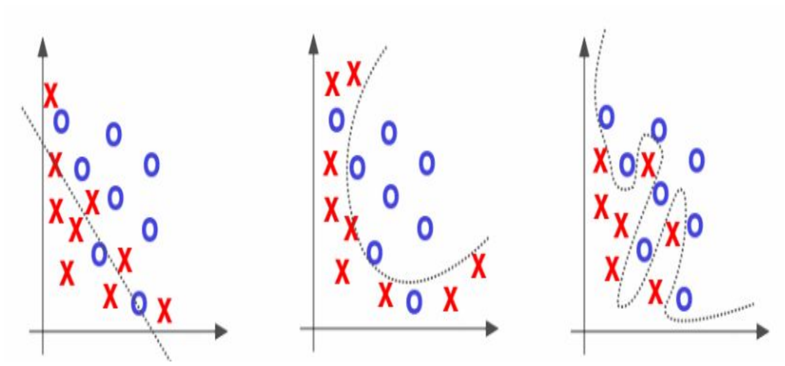


In the schematic diagram of reinforcement learning above, an agent (the reinforcement learning) interacts with the world (environment) through actions. The environment provides observations and rewards to the agent based on the kind of action taken by the agent. The agent uses this feedback to improve its decision making process by learning to carry out actions associated with positive outcomes.

Overfitting and Underfitting

Overfitting and underfitting jointly form the central problem of machine learning. When training a model we want to improve its optimization by attaining better performance on the training set. However, once the model is trained, what we care about is generalization. Generalization in a nutshell deals with how well a trained machine learning model would perform on new data which it has not seen, that is data it was not trained on. In other words, how well can a model generalize the patterns it learnt on the training set to suit real world examples, so that it can achieve similar or better performance. This is the crux of learning. A model should be able to actually learn useful representations from data that improves test time performance and not merely memorize features as memorization is not learning.

We say a model has overfit to a training set when it has failed to learn only useful representations in the data but has also adjusted itself to learn noise in order to get an artificially high training set accuracy. Underfitting means that the model has not used the information available to it but has only learnt a small subset of representations and has thrown away majority of useful information, thereby making it to make unfounded assumptions. The ideal situation is to find a model that neither underfitts nor overfitts but exhibits the right balance between optimization and generalization. This can be done by maintaining a third set of examples known as the validation set. The validation set is used to tune (improve) the performance of the model without overfitting the model to the training set. Other techniques for tackling overfitting includes applying regularization which punishes more complicated models and acquiring more training examples. Underfitting can be stymied by increasing the capacity of the learning algorithm so that it can take advantage of available features.



The plots above show three simple line based classification models. The first plot separates classes by using a straight line. However, a straight line is an overly simplistic representation for the data distribution and as a result it misclassified many examples. The straight line model is clearly underfitting as it has failed to use majority of the information available to it to discover the inherent data distribution.

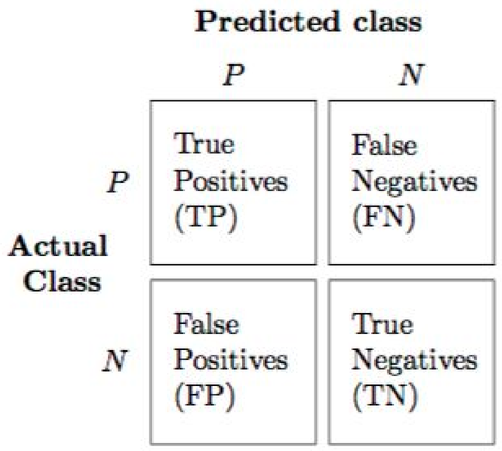
The second plot shows an optimal case where the optimization objective has been balanced by generalization criterion. Even though the model misclassified some points in the training set, it was still able to capture a valid decision boundary between both classes. Such a classifier is likely to generalize well to examples which it was not trained on as it has learnt the discriminative features that drive prediction. The last plot illustrates a case of overfitting. The decision boundary is convoluted because the classifier is responding to noise by trying to correctly classify every data point in the training set. The accuracy of this classifier would be perfect on the training set but it would perform horribly on new examples because it optimized its performance only for the training set. The trick is to always choose the simplest model that achieves the greatest performance.

Correctness

To evaluate a machine learning algorithm, we always specific measures of predictive performance. These metrics allow us to judge the performance of a model in an unbiased fashion. It should be noted that the evaluation metric chosen depends on the type of learning problem. Accuracy is a popular evaluation metric but it is not suitable for all learning problems. Other measures for evaluation include recall, precision, sensitivity, specificity, true positive rate, false positive rate etc. The evaluation used should be in line with the goals of the modelling problem.

To ensure fidelity of reported metrics, a rule of thumb is that models should never be trained on the entire dataset as any evaluation reported by metrics is likely skewed because the model’s performance when exposed to new data is unascertained. The dataset should be divided into train, validation and test splits. The model is trained on the training set, the validation set is reserved for hyperparameter tuning for best performing models and the test set is only used once at the conclusion of all experimentation.

A confusion matrix is widely used as a simple visualization technique to access the performance of classifiers in a supervised learning task. It is a table where the rows represent the instances in the actual class (ground truth) while the columns represents predictions. The order may be reversed in some cases. It is called a confusion matrix because it makes it easy to see which classes the model is misclassifying for another, that is which classes confuse the model.

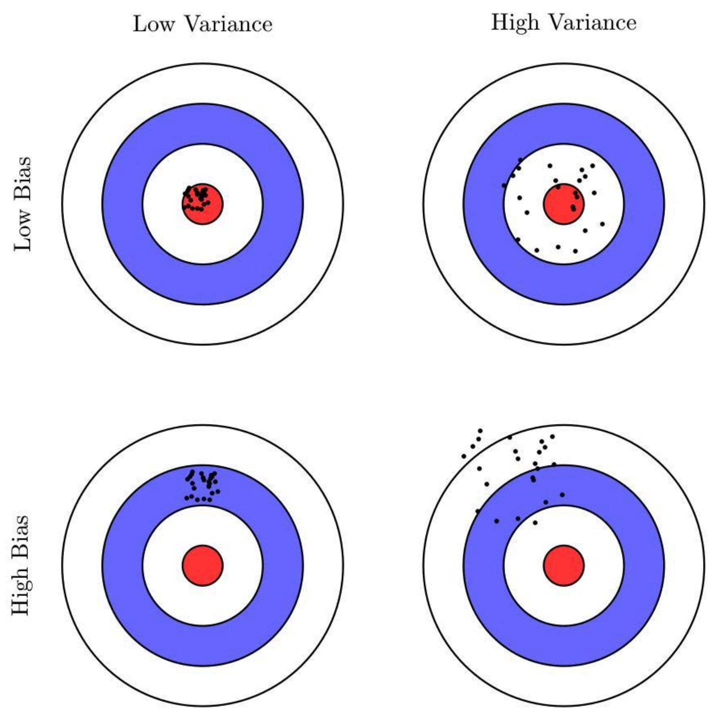


The examples which the model correctly classified are on the diagonal from the top left to bottom right. False negatives are positive classes which the classified wrongly predicted as negatives while false positives are negative instances which the classifier wrongly thought were positives. Several metrics like true positive rate, false positive rate, precision etc are derived from items in the confusion matrix.

The Bias-Variance Trade-off

The bias of a model is defined as the assumptions made by the model to simplify the learning task. A model with high bias makes assumptions which are not correlated by the data. This lead to errors because predictions are usually some way off from actuals. Variance on the other hand is how susceptible a model is to noise in the training data. How widely does the performance on the model vary based on the data it is evaluated on. A good machine learning algorithm should strive to achieve low bias and low variance. Bias and variance are related to overfitting and underfitting earlier encountered. A model with high bias is underfitting the training data because it has made simplistic assumptions instead of learning from information available. Similarly, a model with high variance is overfitting, because it has modelled noise and as a result, its performance would vary widely across the training set, validation set and test set.

Let us look use a dart analogy to further explain these concepts.



The top left image represents a model that has low bias and low variance. This is the ideal model as it has learnt to hit the target (produce correct classification) and usually hits the target most of the time (does not vary with each throw). The image at the top right shows a model that exhibits high variance and low bias. Even if it does not make a lot of assumptions, its predictions are spread all over the board which means its performance varies widely (high variance). The image on the bottom left depicts a model with high bias and low variance. The shots are not all over the board but in a specific location. This location is however far from the target meaning the model is biased because of simplistic assumptions. Finally, the image on the bottom right shows a model with high bias and high variance. The shots on the board vary widely and are far away from the target. This is the worst kind of model as it hasn't learnt any useful representation.

Feature Extraction and Selection

Feature extraction involves performing transformation on input features that produce other features that are more analyzable and informative. Feature extraction may occur by combining original features to create new features which are better suited for the modelling problem. This is similar to feature engineering where we create new features to fed into a model. An example of feature extraction is Principal Component Analysis (PCA).

Feature selection is choosing a subset of features from the original input features. The features selected are those that show the most correlation with the target variable, that is those features that drive the predictive capability of the model. Both feature extraction and feature selection leads to dimensionality reduction. The main difference between them is that feature extraction is a transformation that creates new features whereas feature selection chooses only a subset of available features. Since feature selection removes certain features, it is always advisable to first do feature extraction on a dataset, then select the most important predictors via feature selection.

Why Machine Learning is Popular

The popularity of machine learning in the artificial intelligence community and society in general is mainly because machine learning techniques have proven to be highly successful in various niches providing business value for a slew of operations from fraud detection to speech recognition to recommender systems. It is embedded in products we use every day. When you buy a product from Amazon and you are given a list of suggestions of other products that go with it, that's machine learning in action. When you open your mailbox and emails are automatically classified into folders based on their similarity, those are machine learning models doing the work behind the scenes. Even when you use your credit card online and your transaction is successful, a machine learning model approved your transaction as being normal and not fraudulent.

In light of all these, companies and organizations have poured in more money into developing better performing models through research and collaboration between industry and academia.

It wasn't always the case that machine learning was the darling of the computer science community, however in recent years three factors have conspired to give it an exalted place.

- With the dawn of the internet, more data was being collected and this lead to the age of big data.
- Computational resources became faster and cheaper with the arrival of Graphics Processing Units (GPUs), which were originally developed by the gaming industry to render graphics but are well suited for parallel computation.
- Better algorithms were developed which lead to better accuracy of models.