Model-Based Machine Learning

Introduction

The space of machine learning has grown over time, and researchers have come up with many algorithms for solving problems in various domains. Nevertheless, the latest computational advancements and the exponential growth in data have moved machine learning from a niche research area to the center stage of all modern software applications. Today thousands of engineers and researchers are applying machine learning to a broad range of enterprise and consumer applications. However, the effective use of machine learning, especially when employing for the first time, can be daunting because the engineers and researchers often try to map the problem to the methods they are familiar with or have applied in their experience. Though effective, this approach of "fitting the problem to the algorithm" can be limiting.

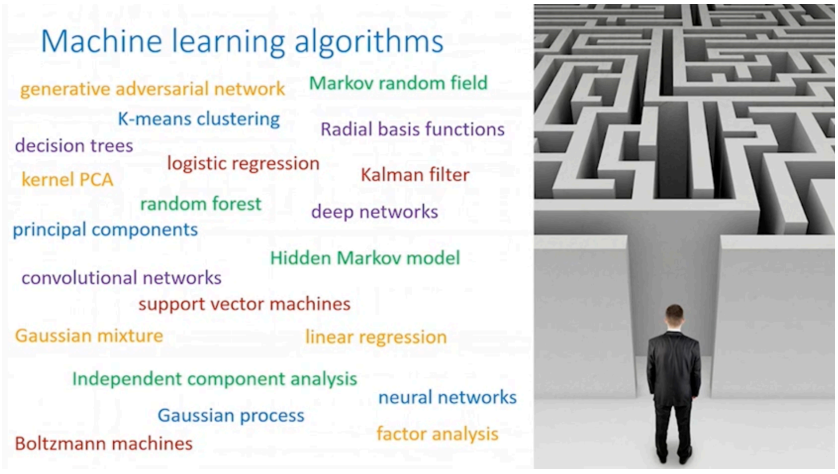

Figure 1: Dilemma of Machine Learning

In this survey, we review "**Model-Based Machine Learning**", an alternative approach for applying machine learning, where *instead of having to modify the problem to fit some standard algorithm, we design the algorithm precisely to fit the problem*.

Machine Learning Goal

In traditional machine learning a practitioner's approach to solving a new machine learning problem typically involves the following steps.

1. Understand the problem and data
2. Select a set of familiar algorithms
3. Select point values for the set of parameters of the algorithms
4. Fit the data to the algorithms
5. Evaluate the results from each algorithm
6. Perform parameter tuning
7. Report the best algorithm

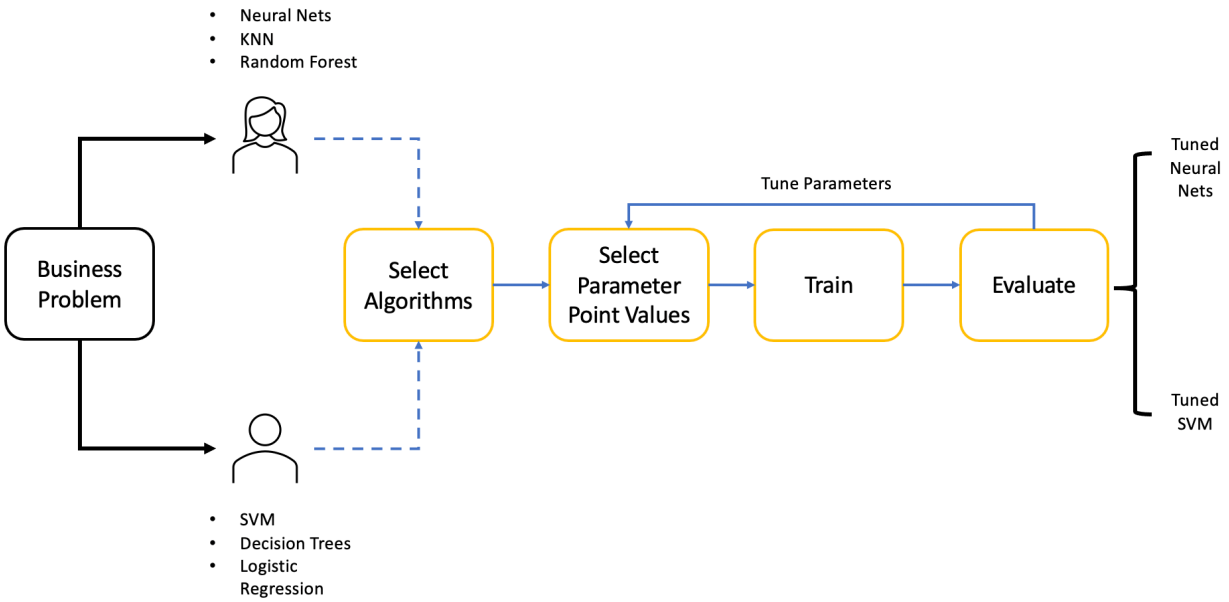

Figure 2: Traditional Machine Learning Workflow

Based on the above workflow, it is clear that the output - the best algorithm - depends on the individual's familiarity of algorithms. One possible way to overcome the limitation is to build a single algorithm that can solve any machine learning problem; however, this is impossible because of a mathematically proven theorem called "**No Free Lunch**", which states that:

> *"Averaged over all possible data-generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points (Wolpert 1996)"*

The theorem implies that given the space of all machine learning problems, every algorithm will perform better on some problem while failing on the remaining, i.e. on average all the algorithms will have the same error rate when applied to the entire space of machine learning problems. Thus, it can be concluded that there cannot be any universal algorithm, and the goal of machine learning is to find the algorithm that is best suited for the problem at hand.

Model-Based Machine Learning

With the machine learning goal in mind, let's now understand the characteristics of a machine learning algorithm. Every machine learning algorithm is a combination of two components - model and inference method - where

- The model captures a set of assumptions about a problem domain, expressed in a precise mathematical form.
- Inference methods are computational techniques used to determine the optimal set of parameters for the model.

In general, what differentiates one algorithm from another is essentially the assumptions that are made by each algorithm; however, in the case of standard algorithms, these assumptions are implicit and opaque.

The core idea of model-based machine learning is to decouple the model and the inference method, enabling us to create bespoke models by changing the list of assumptions used in the model relevant to the problem domain.

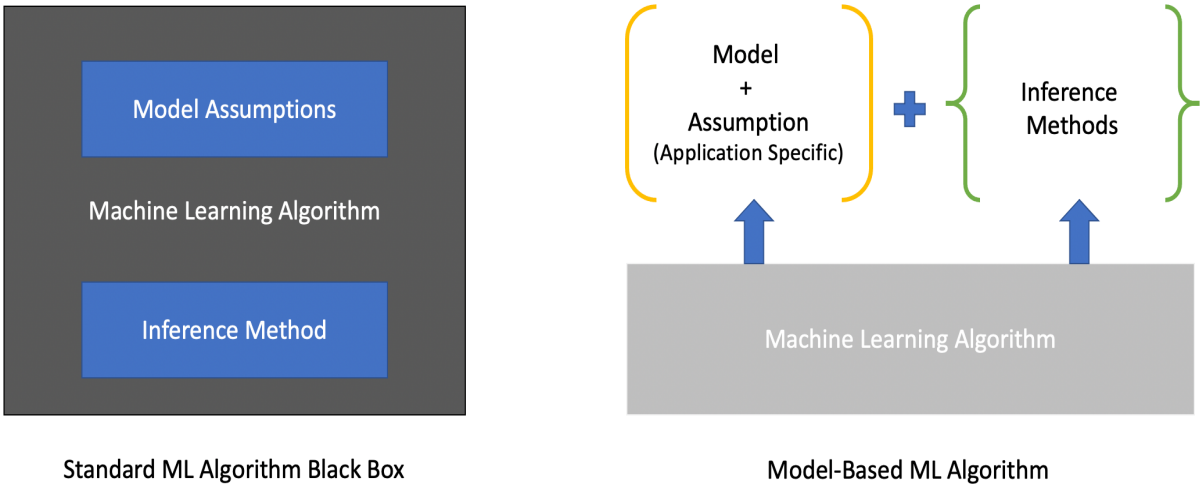

Figure 3: Standard Algorithm Vs Model-based Algorithm

Elements of Model-Based Machine Learning

This section will review the key concepts used to realize the idea of model-based machine learning.

Bayesian Inference

The first pillar supporting the idea of MBML is that of Bayesian inference or learning. In traditional machine learning, the model parameters are point estimates and determined by optimizing a proper cost function. By contrast, in the Bayesian setting, the model parameters are represented as random variables with probability distributions that get learned upon observing data using Bayes' theorem. Thus, the traditional task of parameter estimation is replaced in the Bayesian setting by inference, which is the process of computing probability distributions over random variables based on the observed data.

Bayesian methods are most effective when the supply of data is limited, and the resulting uncertainty in model parameters is significant. However, in today's world, when we are drowning in data, why should one care about the limited data. The answer lies in having a clear understanding of computationally large versus statistically large datasets. For example, in the figure below, the set of points in the plot are computationally insignificant in size (only 7), but they are statistically considerable because they can model the problem well. In contrast, a million images of an object, each containing several megapixels, maybe computationally significant in size, but statistically, they are insignificant for object detection as they cannot capture all the possible combinations of the object class.

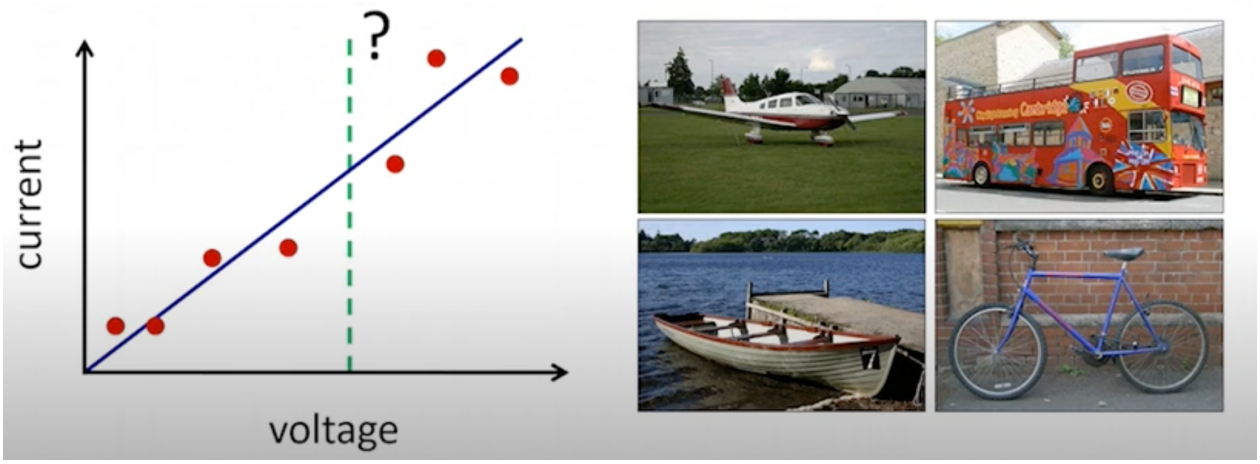

Figure 4: Predicting Current from Voltage based on 7 data points (Statistically Significant) Vs Object Identification based on several megapixel images (Statistically Insignificant)

Many new machine learning applications contain computationally large datasets but are statistically small, and Bayesian inference can play an essential role in addressing those applications.

Probabilistic Graphical Models - Factor Graphs

The second concept that supports MBML is the Probabilistic Graphical Model (PGM), specifically the Factor Graphs. In the Bayesian setting, a model represents the joint probability distribution over all the random variables (both observed and predicted) of the problem. A PGM represents this joint distribution in a graph, using the product rule from probability and captures all the assumptions made in the model.

$$p(\mathrm{y}, \mathrm{x}_1, \mathrm{x}_2) = p(\mathrm{y})\, p(\mathrm{x}_1 | \, y) p(\mathrm{x}_2 | \, \mathrm{y}, \mathrm{x}_1)$$

Joint Probabilty Distribution

Factor graphs are a particular type of PGM that consists of the following components.

1. Variable node, representing the random variable in the model and depicted as a white circle or ellipse in the graph. The observed variables are drawn as filled node
2. Factor node, representing the factors which are multiplied together to give the joint probability distribution. These nodes are depicted as black squares in the model
3. Links or edges between variable and factor nodes represent the dependencies between them

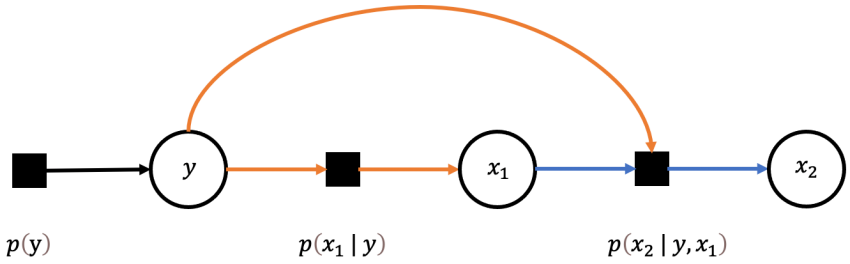

Figure 5: Factor Graph for Joint Probability Distribution

$$p(\mathrm{y}, x_1, x_2) = p(\mathrm{y})\ p(x_1 \mid y)\, p(x_2 \mid y)$$

Naive Bayes Joint Probabilty Distribution

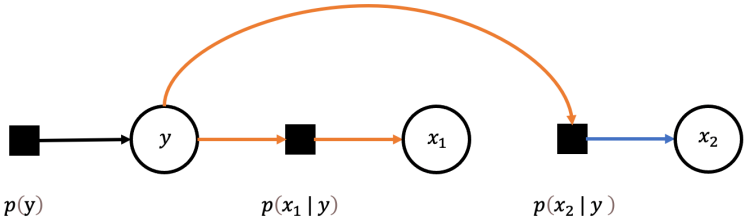

Figure 6: Naive Bayes Classifier Model Factor Graph. There is a missing link between x1 and x2 that captures the assumptions that features are independent

$$p(\mathrm{y}, x_1, x_2) = p(\mathrm{y})\ p(x_1 \mid y)\, p(x_2 \mid y)$$

Naive Bayes Joint Probabilty Distribution

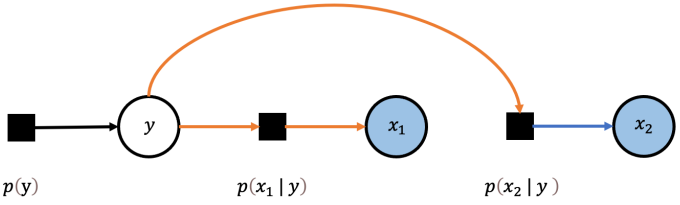

Figure 7: Naïve Bayes Classifier with having observed features x1 and x2

To summarize, the graphical representation gives us an intuitive and compact data structure to capture the high dimensional probability distributions.

Approximate Inference

In most of the applications, instead of learning about the overall joint distribution, we are interested in predicting some values based on some observed variables. In the Bayesian setting, this process is called inference where

- y is the predicted variable
- x_1, x_2 are the observed variables
- z are the latent variables

$$p(y) = \sum_{x_1}\sum_{x_2}\sum_{z} p(y, x_1, x_2, z)$$

$$p(y \mid x_1 = x1, x_2 = x2) \propto \sum_{z} p(y, x_1 = x1, x_2 = x2, z)$$

Though the above equation may seem simple, for all practical purposes computing the above posterior probability is impractical. We must therefore resort to approximations, which themselves need to be computationally efficient while achieving sufficient accuracy for the particular application. This leads us to the third pillar of MBML which are the set of approximation schemes such as Variable Message Passing, Belief Propagation, Expectation Propagation and Variational Bayes.

Case Study – Simple Speech to Text Model

Let us now apply the above ideas to build a simple application for speech to text determination. We begin by listing all the assumptions that our problem must satisfy.

1. The goal is to output a sequence of words based on the input sequence phonics signal
2. Since the correct word sequence is unknown, we model the output at each time step in the sequence as a random variable having a Dirichlet distribution
3. The prediction of a word in the output sequence at step *'i'* depends on the word predicted in the previous step *'i-1'*

Based on the above assumptions, let us look at graphical model of the application

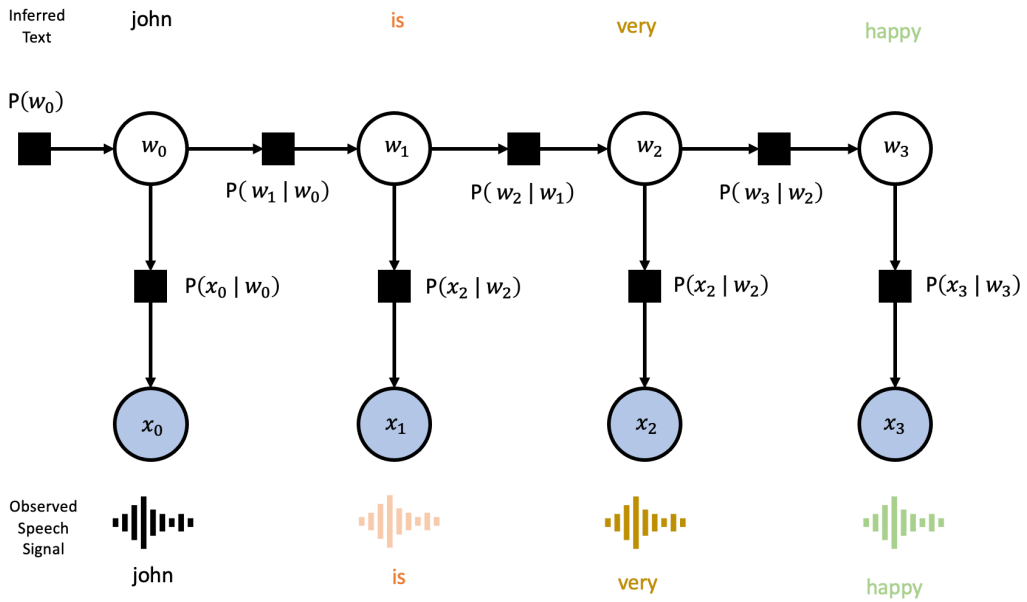

Figure 8: Factor Graph for the model – Hidden Markov Model

Using the training data, we can perform Bayesian inference and estimate or learn the value of each of the probability distributions used in the above model by leveraging any of the inference methods.

Also, what will happen if we change our assumption (3) by another - The prediction of a word in the output sequence at step *'i'* depends on the words predicted in the previous step *'i-1'* and *'i-2'*. Now the revised graphical model will look as the following figure.

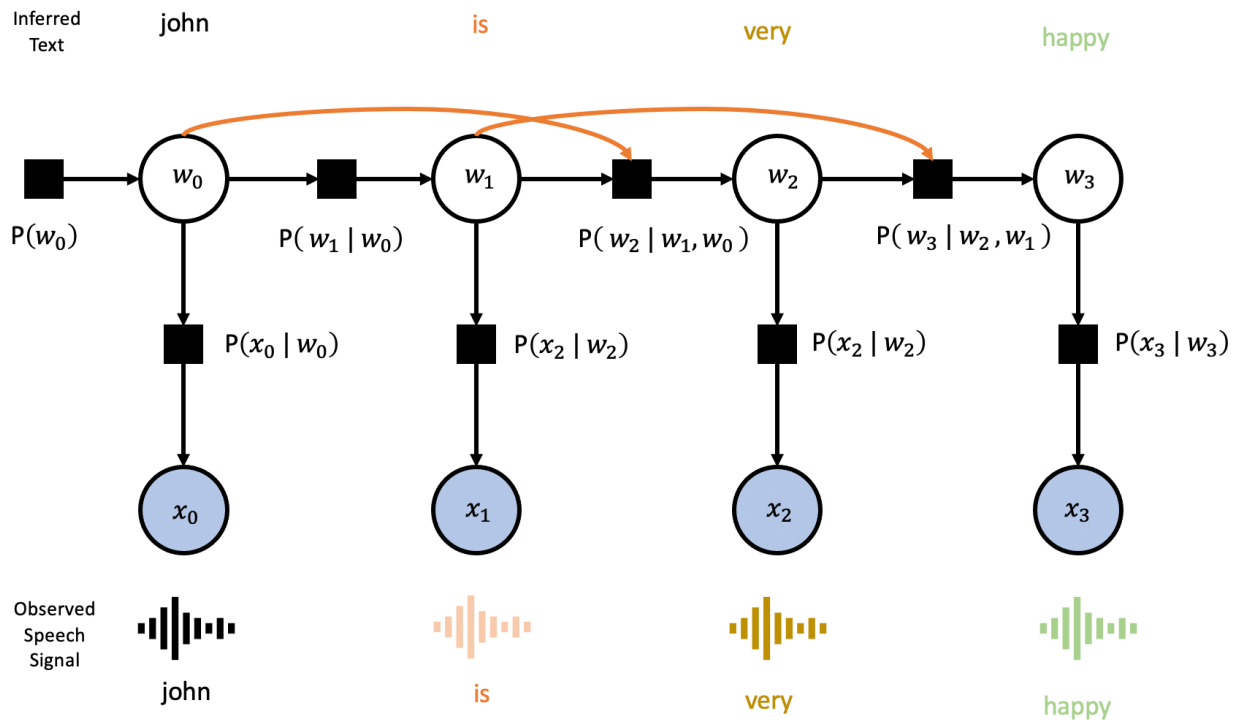

Figure 9: Revised Factor Graph of the Bespoke(new) Model

We can now re-estimate the probability distributions based the same inference method used in the previous iteration.

Conclusion

The traditional approach to machine learning has resulted in numerous successful applications, and it will undoubtedly continue to be an essential paradigm for many years to come. On the other side, model-based machine learning provides an entirely new approach to build custom bespoke models and has many advantages, including its ability to

- Create a broad range of algorithms based on the requirements of the application
- Perform standard machine learning tasks such as classification or clustering while providing additional insights and control
- Extend and adapt existing models based on the changing needs
- Provide explanations about why one model works better than another
- Easily communicate to someone else the inner workings of the model
- Efficiently debug the model when things go wrong

References

1. J. Winn, C. Bishop, and T. Diethe, Model-Based Machine Learning, Microsoft Research, http://www.mbmlbook.com/, 2015.
2. C. M. Bishop, “Model-based machine learning” Phil Trans R Soc, A 371: 20120222. http://dx.doi.org/10.1098/rsta.2012.0222, Jan. 2013
3. C. M. Bishop, “Keynote: Model-Based Machine Learning”, Microsoft Research, https://www.youtube.com/watch?v=zKUFSKRjTIo
4. C. M. Bishop, “Pattern Recognition and Machine learning”