

DISCUSSION: “A FEW USEFUL THINGS TO KNOW ABOUT MACHINE LEARNING”

Scott O'Hara

Metrowest Developers Machine Learning Group

10/14/2020

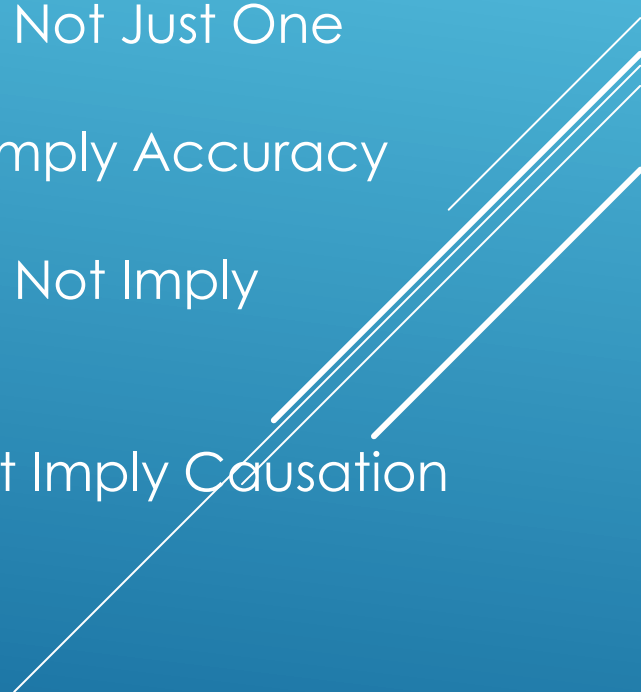
A FEW USEFUL THINGS TO KNOW ABOUT MACHINE LEARNING

- Author: Dr. Pedro Domingos
(<https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf>)
- Also author of the popular book: **The Master Algorithm**
- Written in 2012, Domingos attempted to convey the “black art” of machine learning in an article covering 12 topics i.e., the “folk knowledge” and rules of thumbs that are known by ML practitioners but that are not taught in universities.

A FEW USEFUL THINGS TO KNOW ABOUT MACHINE LEARNING

- Though the article is old, much of it is still relevant.
- However, some points probably need to be updated in light of advances in deep learning.
- I will try to point these out as best I can and welcome input from meetup participants.
- The article exclusively focuses on **supervised learning**, which requires labeled training data.

12 THINGS TO KNOW ABOUT ML

1. Learning = Representation + Evaluation + Optimization
 2. It's Generalization That Counts
 3. Data Alone Is Not Enough
 4. Overfitting Has Many Faces
 5. Intuition Fails in High Dimensions
 6. Theoretical Guarantees Are Not What They Seem
 7. Feature Engineering is the Key
 8. More Data Beats a Cleverer Algorithm
 9. Learn Many Models, Not Just One
 10. Simplicity Does Not Imply Accuracy
 11. Representable Does Not Imply Learnable
 12. Correlation Does Not Imply Causation
- 
- Three white lines of varying lengths and slopes are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

REFERENCES

- A Few Useful Things to Know about Machine Learning:
<https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf>
- James Le's Blog: 12 Useful Things to Know about Machine Learning,
<https://towardsdatascience.com/12-useful-things-to-know-about-machine-learning-487d3104e28>
- Xiangfeng Zhu's Blog: <https://xzhu0027.gitbook.io/blog/machine-learning/untitled/ml>
- Jaeyoon Han's Blog: <http://otzslayer.github.io/machine-learning/domingos2012/>
- Jon Haitz Legarreta Gorroño's Blog post:
<https://vitalab.github.io/article/2018/10/18/UsefulThingsML.html>

1. LEARNING = REPRESENTATION + EVALUATION + OPTIMIZATION


Supervised learning algorithm can be reduced to three components:

- **Representation**
 - **Evaluation**
 - **Optimization**
- 
- A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide, creating a modern, abstract graphic element.


1. LEARNING = REPRESENTATION + EVALUATION + OPTIMIZATION

- A **classifier** must be represented in some formal language that the computer can handle.
- Choosing a representation is tantamount to choosing the set of classifiers that can be learned. This is called the ***hypothesis space***.
- If a classifier is not in the hypothesis space, it cannot be learned.

1. LEARNING = REPRESENTATION + **EVALUATION** + OPTIMIZATION

- An **evaluation function** is needed to distinguish good classifiers from bad ones.
 - Sometimes called the **objective function** or the **scoring function**.
- 
- Several thin, parallel white lines of varying lengths and orientations are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

1. LEARNING = REPRESENTATION + EVALUATION + **OPTIMIZATION**


- A method is needed to search the set of classifiers for the highest-scoring one.
 - The choice of optimization technique determines the efficiency of the learner.
 - The optimization technique uses the evaluation function.
- 
- A series of three parallel white diagonal lines are located in the bottom right corner of the slide, extending from the middle of the right edge towards the bottom left.

1. LEARNING = REPRESENTATION + EVALUATION + OPTIMIZATION

Table 1. The three components of learning algorithms.

Representation	Evaluation	Optimization
Instances	Accuracy/Error rate	Combinatorial optimization
K-nearest neighbor	Precision and recall	Greedy search
Support vector machines	Squared error	Beam search
Hyperplanes	Likelihood	Branch-and-bound
Naive Bayes	Posterior probability	Continuous optimization
Logistic regression	Information gain	Unconstrained
Decision trees	K-L divergence	Gradient descent
Sets of rules	Cost/Utility	Conjugate gradient
Propositional rules	Margin	Quasi-Newton methods
Logic programs		Constrained
Neural networks		Linear programming
Graphical models		Quadratic programming
Bayesian networks		
Conditional random fields		

2. IT'S GENERALIZATION THAT COUNTS

- The fundamental goal of machine learning is to generalize beyond the examples in the training set.
 - No matter how much data we have, it is very unlikely that we will see those exact examples again at test time.
- 
- A series of three parallel white diagonal lines in the bottom right corner of the slide, extending from the middle of the right edge towards the bottom left.

2. IT'S GENERALIZATION THAT COUNTS

Training Data: $\{x_i, y_i\}$

- examples used to train the classifier.
- e.g., emails labeled 'ham' or 'spam'.

Test Data: $\{x_i, ?\}$

- examples set aside that the classifier has never seen before.
- or, emails that arrive tomorrow.

Want to do well on future data, not training data

- can be easy to train perfectly on training data e.g., decision trees or kNN.
- better to have 70% / 70% accuracy on training/test data than 100% / %50%.

3. DATA ALONE IS NOT ENOUGH

- To learn a classifier, it is not enough to have the data.
- Every learner must embody some knowledge or assumptions beyond the data it is given in order to generalize beyond it. This is called the ***inductive bias*** of the learner.


3. DATA ALONE IS NOT ENOUGH

Consider learning a Boolean function of (say) 100 variables from a million examples.

- There are $2^{100} - 10^6$ examples whose classes you do not know.
- In addition, there are $2^{(2^{100})}$ possible Boolean functions that one could learn.
- How do you figure out what those classes are? In the absence of further information, there is just no way to do this that beats flipping a coin.

3. DATA ALONE IS NOT ENOUGH

Some general assumptions that work well:

- Smoothness
 - Similar examples have similar classes
 - Limited dependency of classes
 - Limited complexity of classes (Occam's Razor).
 - The future will be like the past
- 
- A series of white diagonal lines of varying lengths and thicknesses, located in the bottom right corner of the slide, creating a modern, abstract graphic element.

4. OVERFITTING HAS MANY FACES

- Overfitting refers to a model that models the training data too well.
- For example, when your learner outputs a classifier that is 100% accurate on the training data but only 50% accurate on the test data, it clearly has overfit.
- Generalization error can be decomposed into **bias** and **variance**.

4. OVERFITTING HAS MANY FACES

Bias: Bias is a learner's tendency to consistently learn the same wrong thing.

- **Low Bias:** Suggests weaker assumptions about the form of the target function.
- **High-Bias:** Suggests stronger assumptions about the form of the target function.

For example, a linear learner has high bias, whereas decision trees are an example of a low-bias machine learning algorithm.

4. OVERFITTING HAS MANY FACES

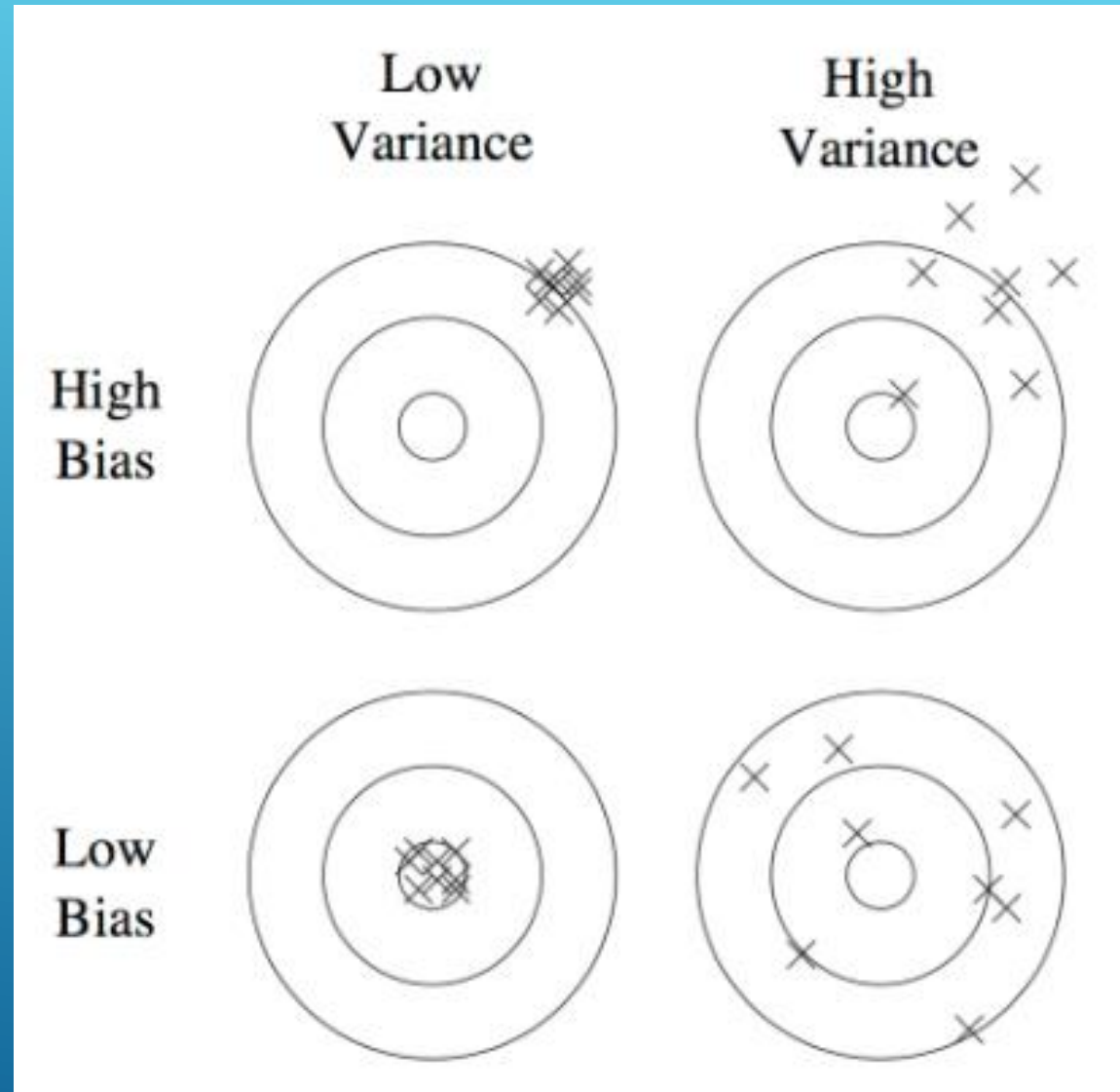
Variance: Variance is the tendency to learn random things irrespective of the real signal.

- **Low Variance:** suggests changes to the training dataset will result in small changes to the estimates of the target classifier.
- **High Variance:** suggests changes to the training dataset will result in large changes to the estimates of the target classifier.

For example, decision trees learned on different training sets generated by the same phenomenon are often very different.

4. OVERFITTING HAS MANY FACES

Bias and variance in dart-throwing.



4. OVERFITTING HAS MANY FACES

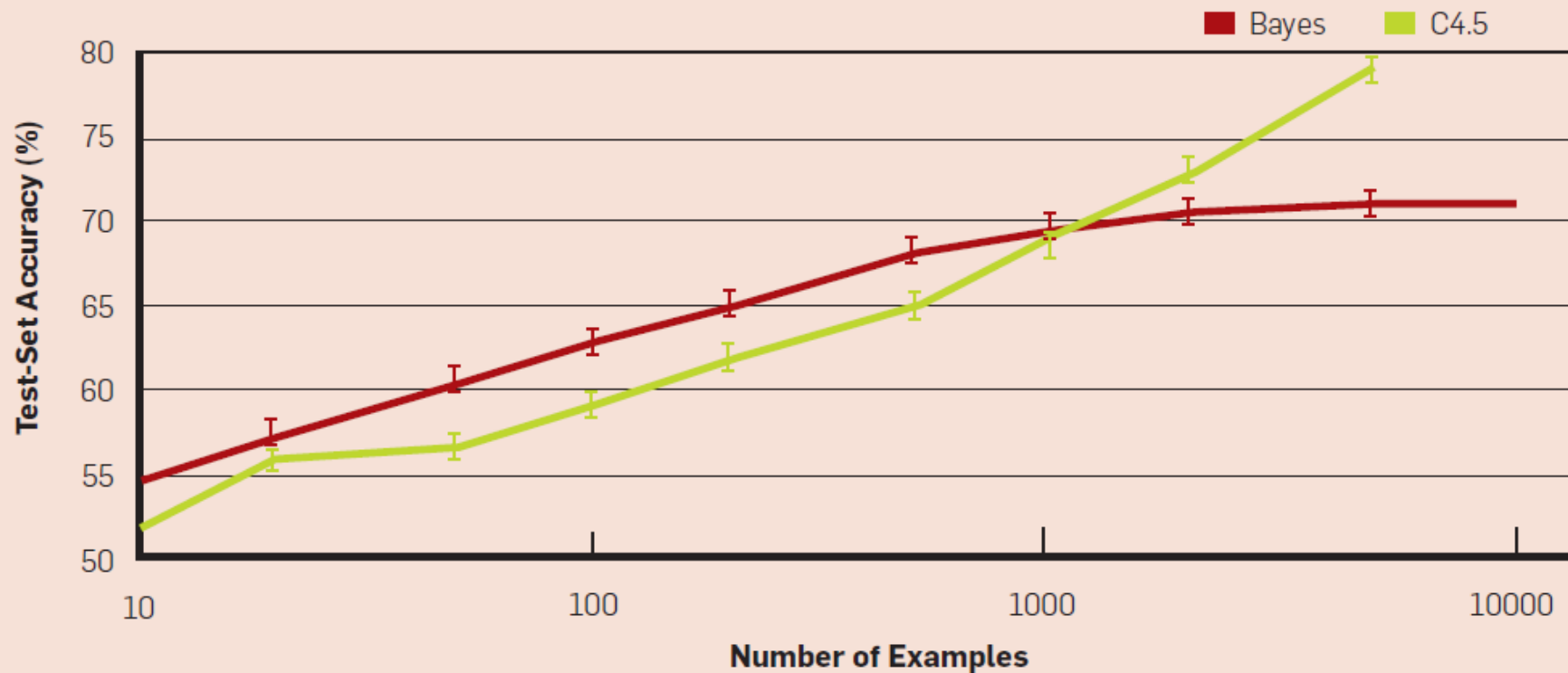
Techniques for Dealing with Overfitting:

- **Regularization** functions added to the evaluation function help ensuring that the method explains the data with the simplest possible model.
- Perform a **statistical significance test** like chi-square before adding new structure.
- Use **cross-validation** to train the model to limit overfitting to a subset of the data.

4. OVERFITTING HAS MANY FACES

Strong false assumptions can be better than weak true ones, because a learner with the latter needs more data to avoid overfitting.

Figure 2. Naïve Bayes can outperform a state-of-the-art rule learner (C4.5rules) even when the true classifier is a set of rules.



5. INTUITION FAILS IN HIGH DIMENSIONS

The Curse of Dimensionality

- One might think that adding more features never hurts, since at worst they provide no new information about the class. However, their benefits can be outweighed by the curse of dimensionality.
- The similarity-based reasoning that machine learning algorithms depend on breaks down in high dimensions.
- In addition, our intuitions, which come from a three-dimensional world, often do not apply in high-dimensional ones.
- As the number of features (dimensionality) grow, the amount of data needed to fit a reasonable model grows exponentially. For example, 20 Boolean features imply a thousand (2^{10}) possible items, 30 Boolean features a billion (2^{30}), 40 features, a trillion and so on.

5. INTUITION FAILS IN HIGH DIMENSIONS

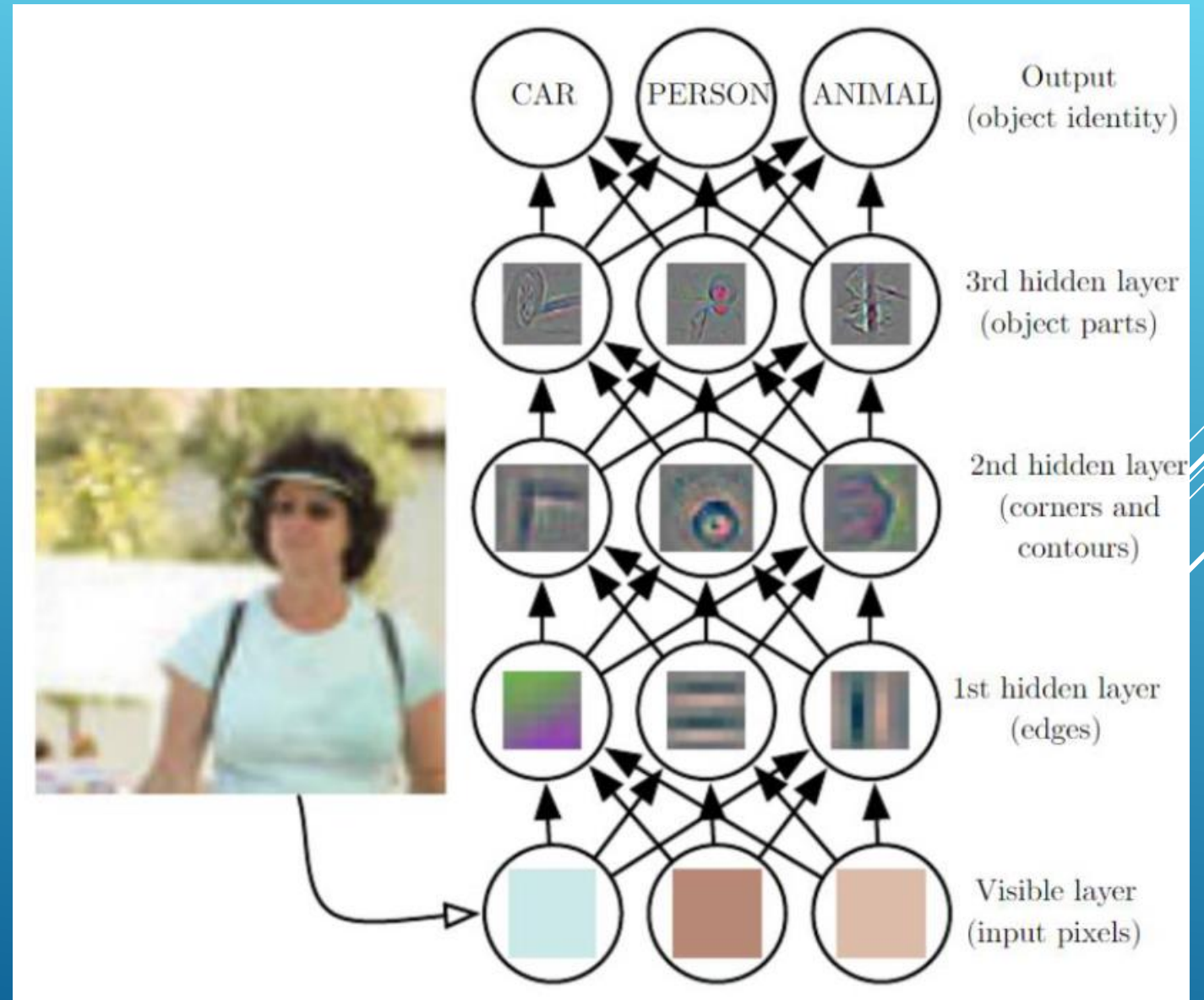
Non-Uniformity

- In most applications, examples are not spread uniformly throughout the instance space, but are concentrated on or near a lower-dimensional manifold.
- For example, kNN works quite well for handwritten digit recognition even though images of digits have one dimension per pixel, because the space of digit images is much smaller than the space of all possible images.
- Learners can implicitly take advantage of this lower effective dimension directly or use dimensional reduction algorithms such as Principle Component Analysis (PCA).

5. INTUITION FAILS IN HIGH DIMENSIONS

Deep Neural Networks

- Deep neural networks had not gotten much attention at the time of the writing of this paper.
- Deep neural networks handle the problem of higher dimensions by successively creating features at each level of the neural network, which are then used as inputs to the next level. This effectively automates dimensionality reduction.



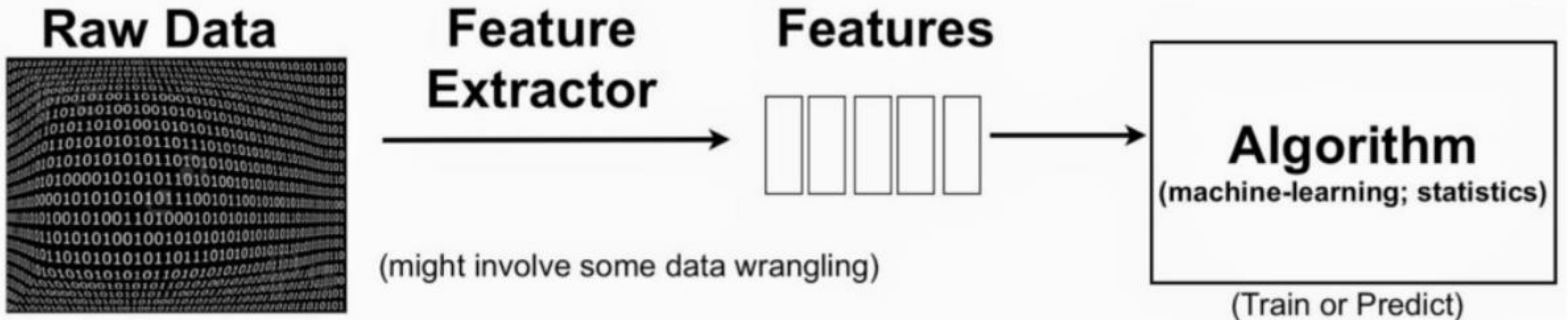
6. THEORETICAL GUARANTEES ARE NOT WHAT THEY SEEM

The main role of theoretical guarantees in machine learning is not as a criterion for practical decisions, but as a source of understanding and driving force for algorithm design.

- The most common type of theoretical guarantees is a bound on the number of examples needed to ensure good generalization.
- Another common type is asymptotic: given infinite data, the learner is guaranteed to output the correct classifier.
- Note that because of the bias-variance trade-off, if learner A is better than learner B given infinite data, B is often better than A given finite data.

7. FEATURE ENGINEERING IS THE KEY

- One of the most important factor in machine learning are the features used.
- Often the raw data is not in a form that is useful for learning; you need to construct useable features from it.



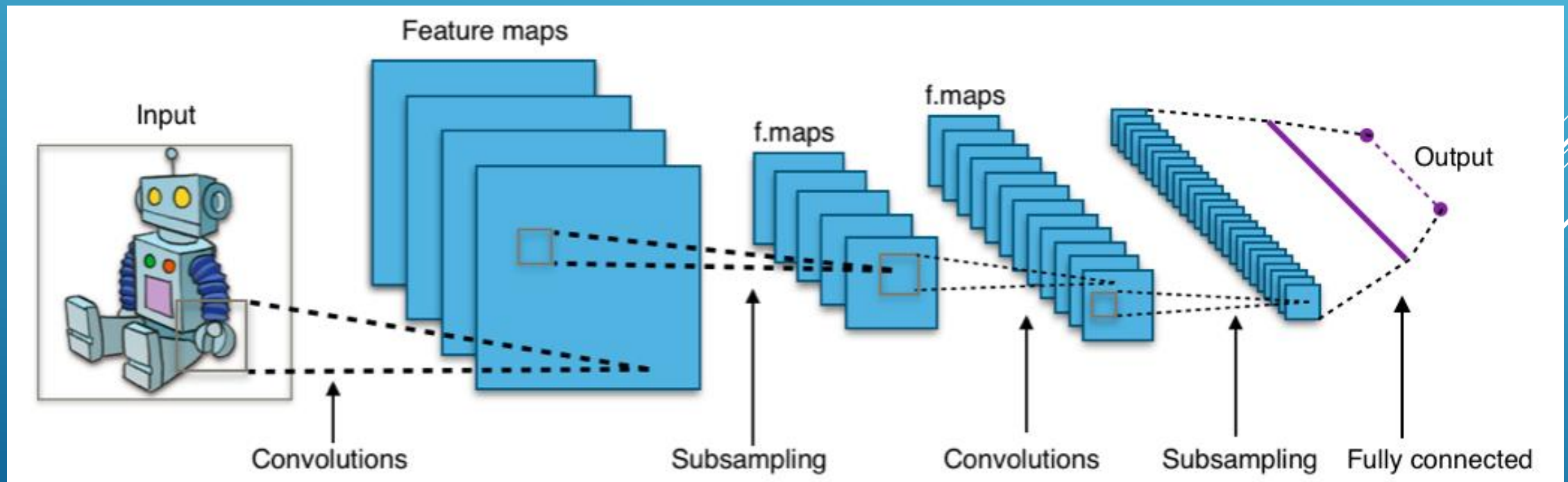
7. FEATURE ENGINEERING IS THE KEY

- Machine learning is not a one-shot process of building a data set and running a learner, but rather an iterative process of running the learner, analyzing the results, modifying the data and/or the learner, and repeating.
- Feature engineering is difficult because it is domain-specific.




7. FEATURE ENGINEERING IS THE KEY

Deep-learning has had a significant impact on this point since deep-learning optimization can create its own features. For example, with RGB images, engineered features are not used anymore.



8. MORE DATA BEATS A CLEVERER ALGORITHM

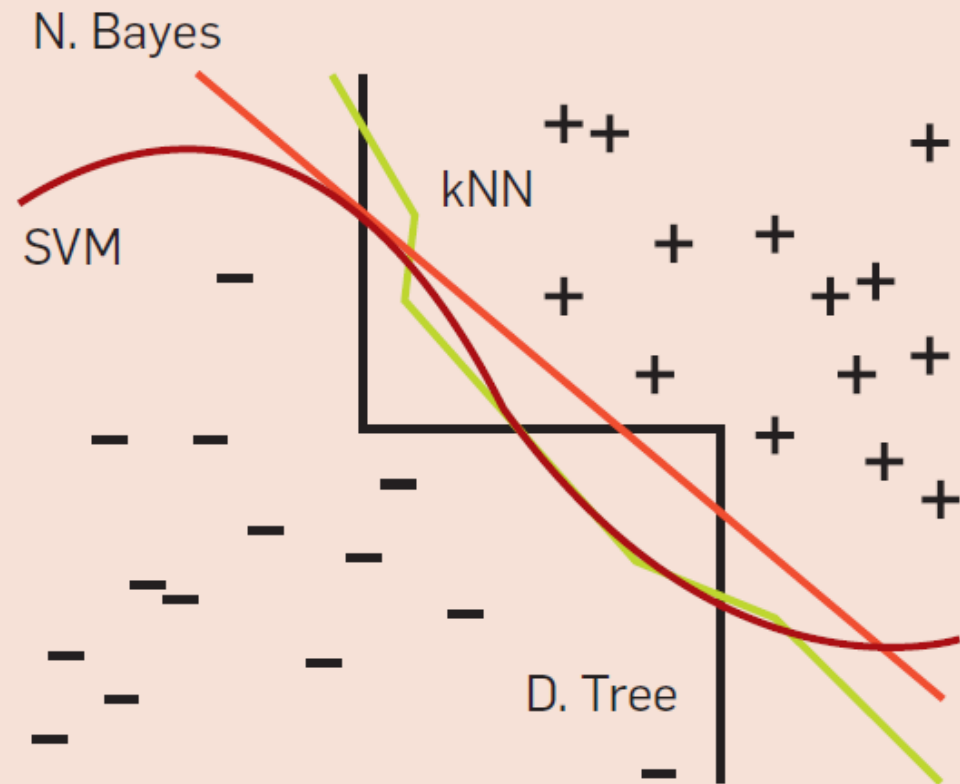
Which is better, design a better learning algorithm or gather more data?

- Often, the quickest path to success is to just to get more data.
 - However, this can lead to another problem: **scalability**.
- 
- Several thin, white, parallel diagonal lines are positioned in the bottom right corner of the slide, extending from the middle of the right edge towards the bottom left.

8. MORE DATA BEATS A CLEVERER ALGORITHM


Part of the reason using a cleverer algorithm has a smaller payoff than you might expect is that, to a first approximation, they all make similar predictions.

Figure 3. Very different frontiers can yield similar predictions. (+ and – are training examples of two classes.)



8. MORE DATA BEATS A CLEVERER ALGORITHM

Which is better, design a better learning algorithm or gather more data?

- Often, the quickest path to success is to just to get more data. However, this can lead to another problem: **scalability**.
 - Complex learners may be able to use less data, but they are usually harder to use and tune, which requires more experimentation.
- 
- A series of three parallel white diagonal lines on a blue background, located in the bottom right corner of the slide.

8. MORE DATA BEATS A CLEVERER ALGORITHM

Learners can be divided into two major types:


Parametric learners (e.g., linear regression) have a fixed-size representation and therefore can only take advantage of only so much data.

- HOWEVER, deep neural networks can use huge numbers of parameters. For example, The largest GPT-3 model has 175 B parameters.

Non-parametric learners (e.g., decision trees) can grow their representation with more data.

- In principle, can learn any function given sufficient data but may be limited by computational costs.
- No amount of data may be enough because of the curse of dimensionality.


9. LEARN MANY MODELS, NOT JUST ONE

- In early ML research, everyone had their favorite learner with some reasons to believe in its superiority. Most effort went into trying many variations of it and selecting the best one.
 - Finally, it was realized that the best learner varies from application to application and systems containing many different learners started to appear.
 - Currently it is understood that instead of selecting the best variation found, better results can be obtained by combining the predictions from many model variations.
- 
- A series of white diagonal lines of varying lengths and thicknesses are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

9. LEARN MANY MODELS, NOT JUST ONE

- In **bagging**, we simply generate random variations of the training set by resampling the training data, learning a classifier on each, and combining the results by voting. This works because it greatly reduces variance while only slightly increasing bias.
- In **boosting**, training examples have weights, and these are varied so that each new classifier focuses on the examples the previous ones tended to get wrong.
- In **stacking**, the outputs of individual classifiers become the inputs of a “higher-level” learner that figures out how best to combine them. NOTE: deep neural networks implement a kind of stacking.
- The **random forest algorithm** combines random decision trees with bagging to achieve very high classification accuracy.

10. SIMPLICITY DOES NOT IMPLY ACCURACY

- Occam's razor can be applied in machine learning, but there are many counterexamples to it. For example, ensemble learning, support vector machines and deep neural networks.
 - Simpler hypotheses are often preferred because simplicity can be a virtue in its own right, not because of a hypothetical connection with accuracy. For example, simpler hypotheses are easier to understand, apply and modify.
- 
- A series of white lines of varying lengths and orientations are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

10. SIMPLICITY DOES NOT IMPLY ACCURACY

- The “No Free Lunch Theorems” – Wolpert et al. suggest that there is no universal learning algorithm.
 - A model is a simplification of reality.
 - Simplification is based on assumptions (model bias).
 - Assumptions fail in certain situations.

How well an algorithm will do is determined by how aligned the algorithm is with the actual problem at hand. Depending on the problem, it is important to assess the trade-offs between speed, accuracy, and complexity of different models and algorithms and find a model that works best for that particular problem.

11. REPRESENTABLE DOES NOT IMPLY LEARNABLE

- Essentially all representations used in variable-size learners have associated theorems of the form “Every function can be represented, or approximated arbitrarily closely, using this representation.” Reassured by this, fans of the representation often proceed to ignore all others.
- However, just because a function can be represented does not mean it can be learned. For example, standard decision tree learners cannot learn trees with more leaves than there are training examples.
- In continuous spaces, representing even simple functions using a fixed set of primitives often requires an infinite number of components.

11. REPRESENTABLE DOES NOT IMPLY LEARNABLE

- Further, if the hypothesis space has many local optima of the evaluation function, as is often the case, the learner may not find the true function even if it is representable. Given finite data, time and memory, standard learners can learn only a tiny subset of all possible functions, and these subsets are different for learners with different representations.
- Therefore the key question is not “Can it be represented?”, to which the answer is often trivial, but “Can it be learned?” And it pays to try different learners (and possibly combine them).

12. CORRELATION DOES NOT IMPLY CAUSATION

- The point that correlation does not imply causation is made so often that it is perhaps not worth belaboring. Many researchers and philosophers believe that causality is only a convenient fiction.
- However, the correlations discovered by machine learning are often treated as causation. If we find that beer and diapers are often bought together at the supermarket, then perhaps putting beer next to the diaper section will increase sales.
- ML practitioners and users should be aware of this distinction when applying the results and insights obtained from machine learning.

REVIEW: 12 THINGS TO KNOW ABOUT ML

1. Learning = Representation + Evaluation + Optimization
 2. It's Generalization That Counts
 3. Data Alone Is Not Enough
 4. Overfitting Has Many Faces
 5. Intuition Fails in High Dimensions
 6. Theoretical Guarantees Are Not What They Seem
 7. Feature Engineering is the Key
 8. More Data Beats a Cleverer Algorithm
 9. Learn Many Models, Not Just One
 10. Simplicity Does Not Imply Accuracy
 11. Representable Does Not Imply Learnable
 12. Correlation Does Not Imply Causation
- 
- Three white lines of varying lengths and slopes are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

CONCLUSION

- Machine learning has a lot of “folk wisdom” that can be difficult to come by but is crucial for success.
- The article “A Few Useful Things to Know about Machine Learning” summarized some of the most salient items.

<https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf>

- Of course, learning these techniques must be complemented by the more conventional study of machine learning such as coursework, implementations, reading paper and books etc.
- While the paper was published in 2012 and the emergence of deep neural networks has changed some things, the techniques and issues discussed in the paper are still relevant today.