FEATURE LABS

Machine Learning 2.0 for the Uninitiated

A PRACTICAL GUIDE TO IMPLEMENTING ML 2.0 SYSTEMS FOR THE ENTERPRISE.

What is ML 2.0?

ML 2.0 is a new paradigm for approaching machine learning projects. The key differentiator for ML 2.0 projects is an emphasis on ***deployment*** and ***impact*** as the ultimate goals, across the entire length of the project.

Is ML 2.0 right for you?

You have ***never*** attempted to build and deploy a machine learning solution to a problem

Expect the principles and guidelines surrounding ML 2.0 to empower you to both start and finish your first machine learning project with ease.

You have ***tried but never succeeded*** at deploying a machine learning solution to a problem

ML 2.0 will take you through the finish line.

You have ***successfully deployed*** machine learning solutions in the past

ML 2.0 will increase the speed to deploy and accuracy of future machine learning solutions.

Deploy predictive models, faster

ML 2.0 defines practical steps along the entire machine learning workflow. This enables companies to tackle projects in a systematic way, and take advantage of automated tools when applicable.

The end result is a much faster route to deployment for individual projects, and a significant increase in the number of projects capable of being deployed.

Examples problems you can solve with ML 2.0

- Predict and recommend what a customer should buy next
- Identify the variables that indicate if sales will go up or down next quarter
- Warn if a piece of manufacturing equipment is likely to fail soon
- Predict if a user will upgrade to a paid account
- Optimize your supply chain by anticipating demand
- Forecast key metrics like employee attrition to better operate your business

ML 2.0 Fundamental Principles

Use automated tools & processes when possible

Focus on effective interplay between humans and automation

Automation minimizes cognitive load enabling your team to think at a higher level

- What parts of my data are relevant?
- What features would make the most sense to include?
- Am I solving the right problem?
- How will this solution be used?

Use standardized abstractions, data representations, & processes

With the right set of abstractions, the same API can be used in training and production

Tightly integrate all teams from the beginning

Maximize the chance that a solution gets deployed for real business impact

ML 2.0 Steps

1. Work from raw data, without creating redundant copies of data

2. Translate the business goal into a machine learning problem using prediction engineering

3. Make data machine learning-ready with automated feature engineering

4. Train and optimize machine learning model

5. Test and validate the system

6. Deploy and operationalize the solution

What is prediction engineering?

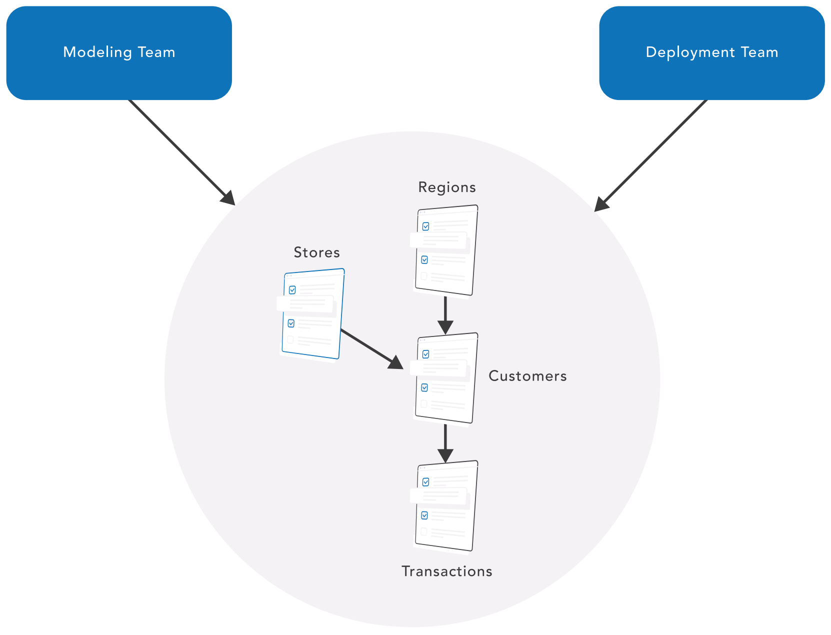
The process of translating business goals into machine learning problems. With prediction engineering, you programmatically define prediction problems so you can use software to search for the most impactful ones.

What is feature engineering?

The process of identifying and extracting predictive features from complex dataset that can be used to train machine learning algorithms. This is a required step to build high-performing machine learning models.

Work from raw data as much as possible, without creating redundant copies of data

- Start by organizing the raw data, as well as adding metadata and annotations
- This enables the same data representation to be used everywhere
- If underlying raw data changes, the effects will be known immediately
- Deployment and modeling team access data with same APIs



Representing Data

Most machine learning projects are hindered by lack of data definition and management practices. ML 2.0 introduces a unique in-memory representation of data called an Entity Set and has a number of apis to interact with Entity Sets to manage the data throughout the course of the project.

Translate the business goal into a machine learning problem using prediction engineering

Prediction engineering is the process of specifying a prediction problem such that it can be used to programmatically label data for machine learning. Prediction engineering enables you to systematically use ML to solve problems that matter for your business.

This process contains many parameters that drastically affect the actual problem being solved. For instance, one important parameter is the lead time, or the amount of time into the future to predict. In general, increasing the lead time (predicting farther into the future) make the problem harder. When undertaking a predictive machine learning project, companies need to experiment with several different lead times to find a suitable mix of predicting far enough out to be useful, yet close enough to be accurate.

Customer id	Upgraded Account	Cutoff
1	True	4/9/15
2	False	5/9/15
2	True	5/9/15
..	..	..
...	...	...

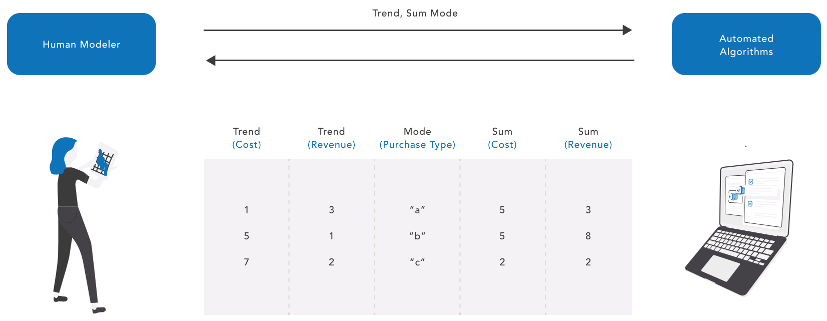
In this example, we predict "if a customer will upgrade to a paid account in 30 days after the cutoff time." Prediction engineering is used to create the examples above for training our machine learning algorithms.

Make data machine learning-ready with automated feature engineering

Feature engineering is the process of using domain knowledge and human intuition to extract new variables from raw data that make machine learning algorithms work.

Feature engineering is challenging because it is tedious, time-consuming, and error-prone. It is now possible to automate this process using Deep Feature Synthesis.

Renowned machine learning professor Pedro Domingos accurately said, „one of the holy grails of machine learning is to automate more and more of the feature engineering process.“



Trend (Cost)	Trend (Revenue)	Mode (Purchase Type)	Sum (Cost)	Sum (Revenue)
1	3	"a"	5	3
5	1	"b"	5	8
7	2	"c"	2	2

What are features?

Machine learning algorithms require data to be in a single table, with training examples as the rows and explanatory variables, or features, as the columns. This data representation for machine learning is called the feature matrix.

A *feature* is a quantitative, measurable property of a particular entity. For example, in online retail, features for a customer could be "the total amount spent in the past month“ or "time since last purchase". These features could then be used to predict, "will a customer make a purchase next month?". If we increase the quantity or quality of features, our machine learning model will perform better.

Train and optimize machine learning model

Good open source tools for machine learning models have been around for a decade or more now. They accept input in the form of a feature matrix and labels created by feature engineering and prediction engineering.

After an end-to-end solution is built, including a statistically valid way to measure its business impact, automatic algorithms can be used to optimize over alternative machine learning algorithms and hyperparameters (parameters that control how models are constructed). Be cautious about over-optimization too early in the process – engineering the right prediction problem and set of features generally has a much greater impact on the end result.

Scikit-Learn is the definitive source for classic machine learning methods, and Keras uses a very similar API for deep learning methods.

How to select the right machine learning method.

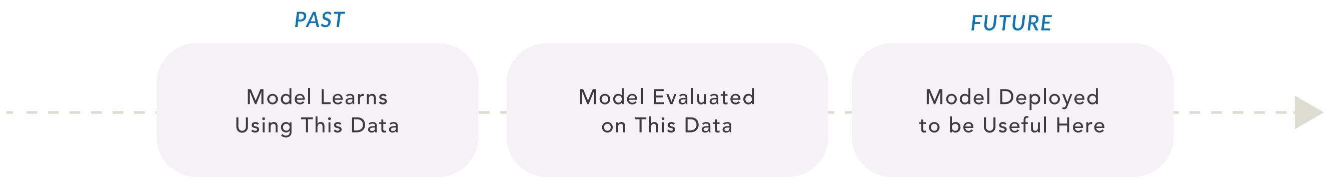
Data scientists do not know without trying which of the many modeling techniques will work best for their data. This has been an open problem in machine learning. Particular techniques work better with particular types of data — for example, support vector machines and recurrent neural networks are known to be better for natural language processing problems, while convolutional neural networks are known to do well for image recognition problems. Random forests and other tree-based models tend to perform well over behavioral or transactional data, and have the added benefit of being readily interpretable.

Test and validate the system

It is crucial to be confident in your system before deploying it in a business-critical scenario. When using machine learning to build predictive models the testing requires simulating the state of the world at different points in time.

Pay special attention to time

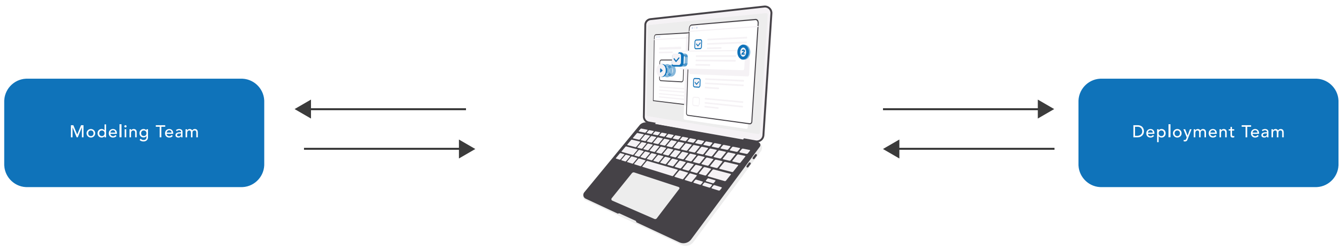
- ML systems learn from historic training data to predict the future.
- The underlying processes can change over time requiring comprehensive testing from time to time.
- Your system needs to be tested on historical data in the same way it will be deployed, using only access to data available before simulated time points in the past.


PAST
FUTURE
Model Learns Using This Data
Model Evaluated on This Data
Model Deployed to be Useful Here

Deploy and operationalize the solution

Proof-of-concepts are only valuable if they can easily be transformed into a production-ready system.

By using the same abstractions and APIs for development and production, ML 2.0 ensures you can easily deploy when you're ready.



FEATURE LABS

About Feature Labs

See Our Platform In Action - Request a Demo Today.

Feature Labs builds tools and API's to deploy impactful machine learning solutions by combining open source software and proprietary algorithms for automated feature engineering. Founded in 2015 out of the Computer Science and AI Lab at MIT - CSAIL, the company is funded by DARPA and venture investors, based in Boston, MA.

For more information, visit **www.featurelabs.com**

www.featurelabs.com | contact@featurelabs.com | 229 Berkeley St, Boston, MA 02116