Theory, Practice & Products

Three main approaches for work that happens with machine learning:

1. **Theoretical**: Focus on creating new models and mathematically proving their effectiveness.
2. **Practitioner**: Solving specific problem defined by a dataset.
3. **Product-Based**: Focus on creating a product to solve a consumers problem.

Roles in an organization dealing with Machine Learning:

- Software engineers: build a customer-facing product.
- Data analysts: use data for decision making and to inform the rest of the organization or its client
- Data engineer: create data pipelines and apply (optimized) algorithms to deliver machine-learning capabilities to the product
- Machine learning engineers - program machine learning algorithms interfacing w/ product
- Data scientist: determine the best machine learning algorithms that apply to the customer domain and fine tune it
- Research engineers: develop new machine learning algorithms and models relevant to the product domain

Theory

Theory approach focuses on how machine learning models can solve the problems they aim to solve. Machine learning = MODEL PARAMETER ESTIMATION.

Models

A models purpose is to define a representation for data that mimics the underlying principles that create the data. We define a model by defining an *objective function* that the model should minimize or maximize.

Ex. Two variables x and y that have a linear relationship. We can represent y in terms of x using:

$$f(x) = w_1 x + w_0$$

w_1 is the sloe of the line, w_0 is the bias or intercept. $f(x)$ can accurately represent y if and only if there in fact exists this linear relationship and x is the only variable that influences y. In reality, this rarely occurs. If we have a linear relationship between x and y, our real $f(x)$ is usually:

$$f(x) = w_1 x + w_0 + \epsilon$$

ϵ is a function of all the other variables that can influence y.

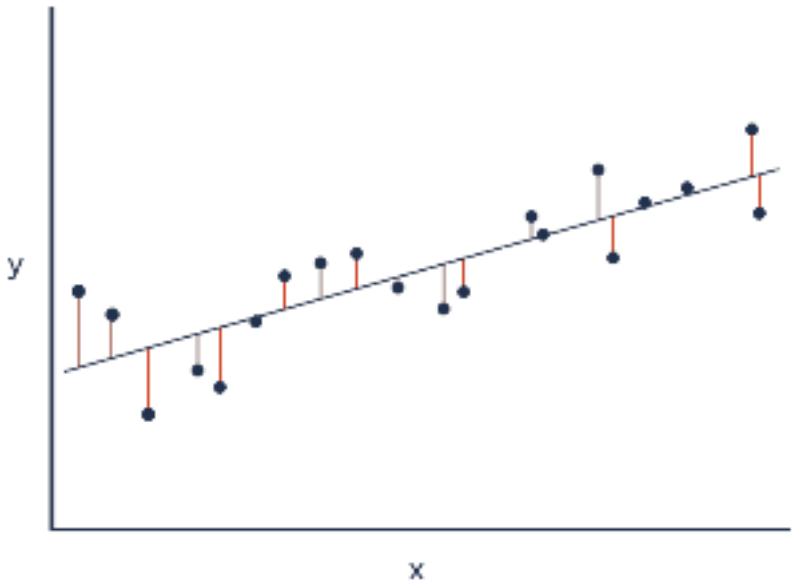

Figure 1: Example of minimization of euclidean distance

Linear Regression

Linear regression: best fit line. We do this because accurately representing all possible variables that influence y is infeasible.

Best fit lie (Linear regression): a line that minimizes the squared euclidean distance between the line and the actual data points.

$$\min_{w_1, w_0} \sqrt{\sum_{i}^{n} (y_i - (w_1 x_i + w_0))^2}^2 ,$$

Figure 2: Formula to minimize euclidean distance to 0

The w_0 and w_1 under min means that they're the two variables that will change to minimize the equation. Minimizing the equation means minimizing the square distance between actual y and estimated y.

Formulation above can be generalized for a polynomial regression by $\mathbf{Y}=\mathbf{w^T X}$ as,

$$\min_{\mathsf{w}} \ ||\mathsf{y} - \mathsf{w}^T\mathsf{X}||_2^2,$$

Figure 3: Polynomial regression generalization of formula in figure 1

$\mathbf{X}$ is an (m+1) x n matrix, $x_{ij} = x^j_i$

$\mathbf{w}$ is an (m+1) x 1 vector

and $||$ x $||_p$ is the lp-norm

$$||\mathsf{x}||_p = \sqrt[p]{\sum_i^n x_i^p},$$

Figure 4: lp-norm definition

P ≥ 1

$$\min_{\mathbf{w}} \ ||\mathbf{y} - \mathbf{w}^T\mathbf{X}||_2^2$$

$$\min_{\mathbf{w}} \left\| \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix} \begin{bmatrix} x_1^0 & x_1^1 & x_1^2 & \dots & x_1^m \\ x_2^0 & x_2^1 & x_2^2 & \dots & x_2^m \\ x_3^0 & \ddots & & & \\ \vdots & & \ddots & & \\ x_n^0 & \dots & \dots & \dots & x_n^m \end{bmatrix} \right\|_{2,1}^2$$

Figure 5: Expanded version of Polynomial regression formula

Practitioners

Practitioner: Start with a variety of available model they want to use on a particular dataset.

Data: any information collected about an object to represent/explain it.

Metadata: data about data.

Representing the data means the model can predict how one value changes when compared to others (supervised), detect a pattern or complete the data (unsupervised), or make decisions about the data (reinforcement).

Model Fitting

Procedure for model fitting is to split data into *training set* and *test set*, train the model on training set, and test the model on testing set.

Figure 6: Model fitting

Three things can happen to your model when training:

1. Overfitting: training a model that performs well on training data, but poorly on test data
2. Generalizing: training a model that learns enough from training to generalize a pattern that does well in test data
3. Underfitting: training a model that performs poorly on both training and test data

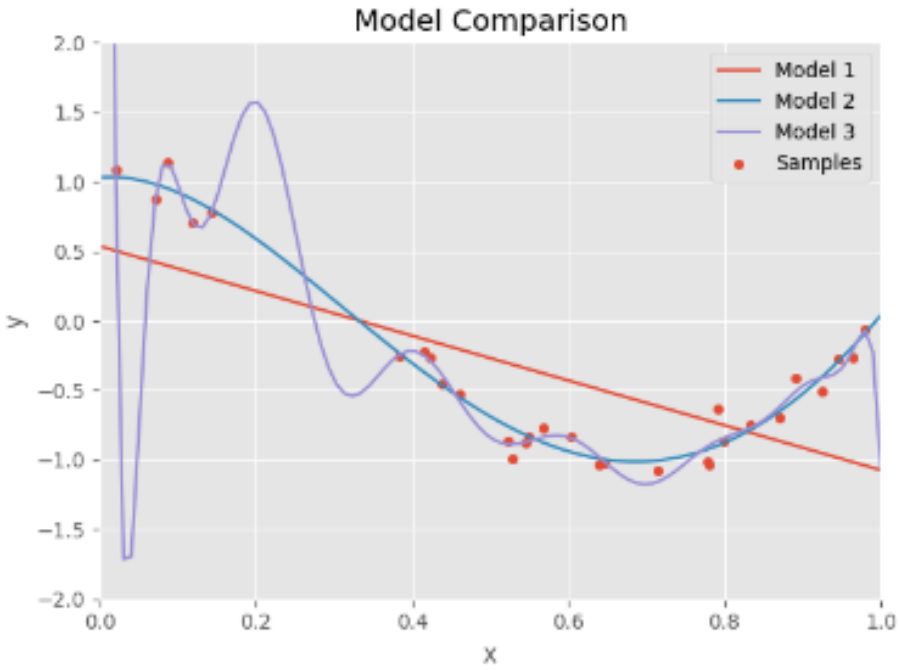

Figure 7: Model 1: underfit, Model 2: generalize, Model 3: overfit

Every model has parameters that it learns through training (ex: coefficients of a polynomial) Some models also have **hyperparameters** that are inputs to the model training (ex: the order of the polynomial to use).

Cross Validation

Cross validation: method used to determine ideal hyperparameters of a model. This is done by splitting your training set into model and validation sets.

- Model set is used to train model with different hyperparameters.
- Validation set used to validate these new models

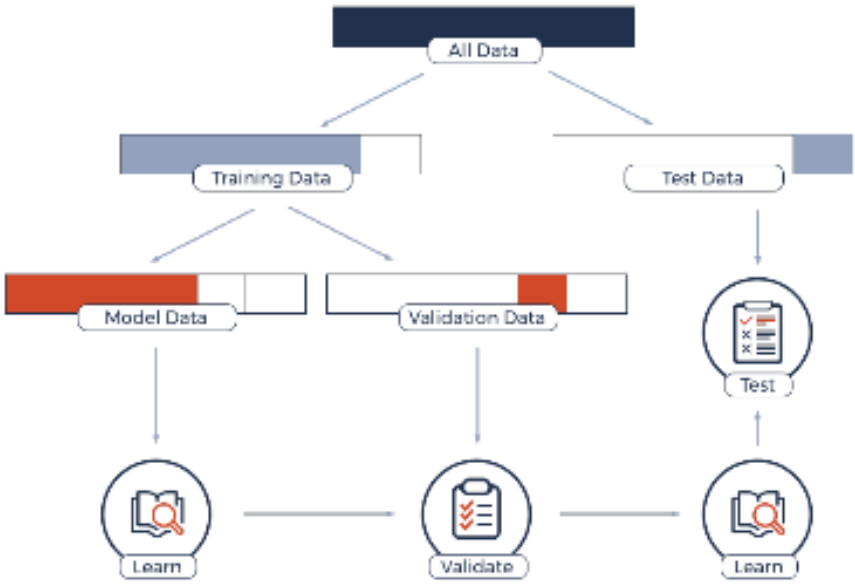

Figure 8: Cross Validation data splits

Once the best hyperparameters is determined, train a new model using them on the entire training set.

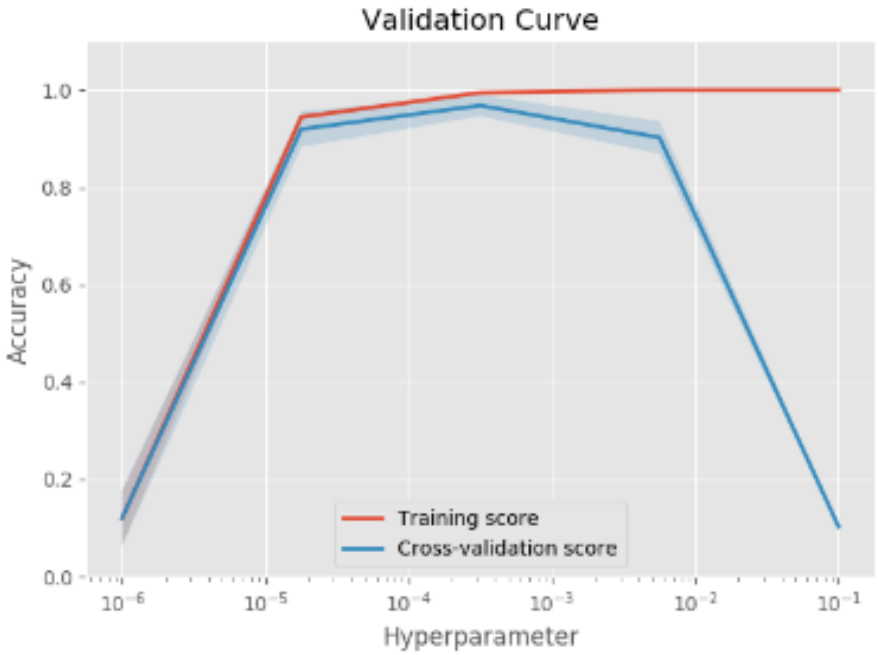

Figure 9: Validation curve showing hyper parameter change vs accuracy

The shaded blue line shows the range in cross validation score. The solid blue line shows the actual averages. As you can see there is a trade off of too much hyperparameter fitting that results in a lower cross validation score.

k-fold cross validation: process of breaking training set into k equal portions and repeating cross validation experiments k times.

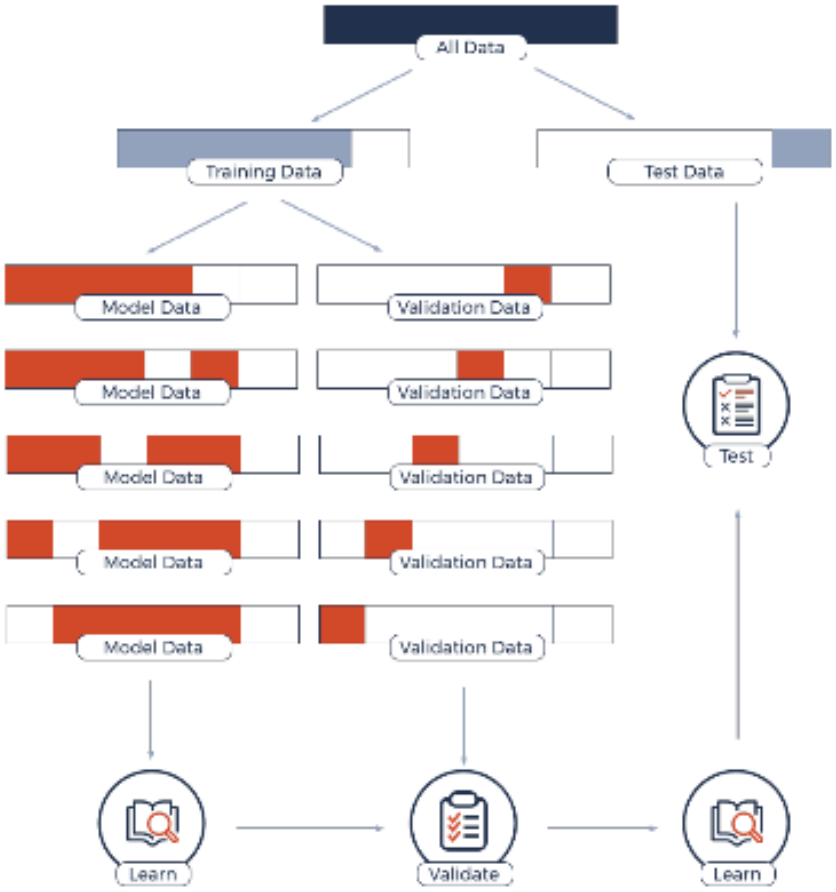

Figure 10: Example of k-fold cross validation

Product

Product: Focus on a measurable value produce for a customer/client.

Model doesnt have to be the best, the focus is on business and engineering

Figure 11: Product: Business question