Machine Learning Problem Framework

Presenters:
Bikramjeet Singh(20752928)
Priyansh Narang (20716980)

Agenda

- Background research
- Brief introduction to Machine Learning
- ML Problem: Formulation
- ML Pipeline
- Questions

Background Research

RE for ML/ ML for RE feat. Google Scholar

- We tried multiple keywords to review past work done in this space: requirement engineering, requirement elicitation, SDLC for ML
- No credible source for RE for ML
- Several papers where authors have used techniques from ML to improve Requirement Engineering:
  - Estimation of effort for tasks
  - Prioritizing requirements
- Few online publishing platforms have articles about the intersection of SE and ML
- This is an attempt at developing and end-to-end framework for systems leveraging Machine Learning

Introduction to Machine Learning

Formal definition

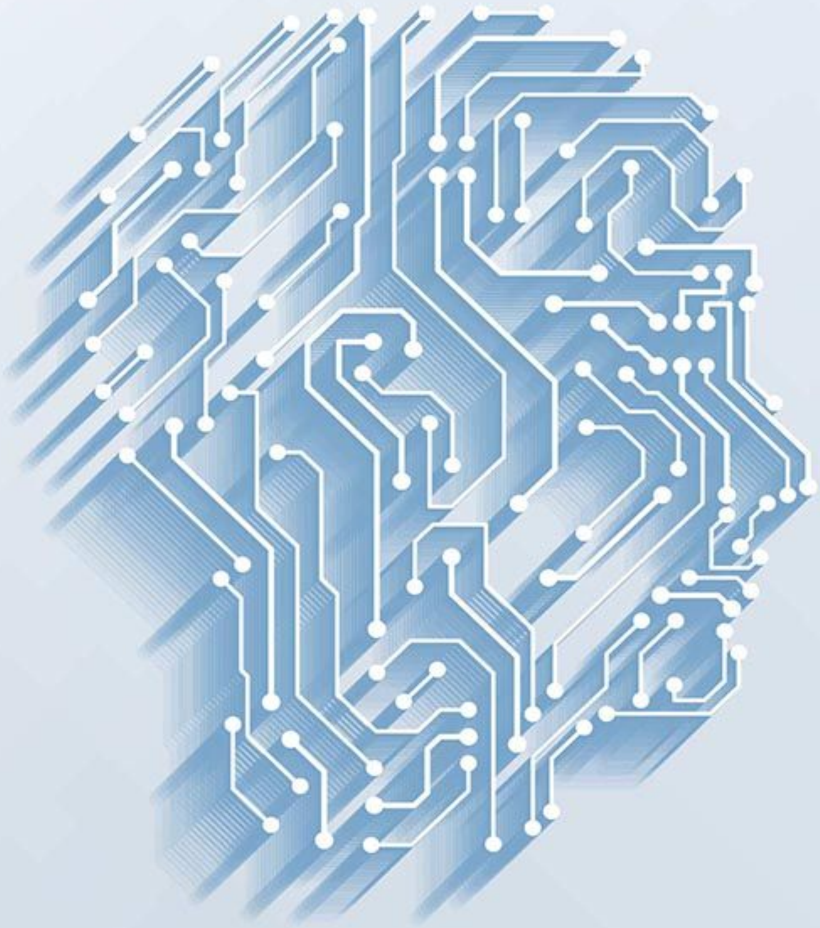
"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"

Type of Machine Learning Problems

Type of ML Problem	Description	Example
Classification	Pick one of N labels	cat, dog, horse, or bear
Regression	Predict numerical values	click-through rate
Clustering	Group similar examples	most relevant documents (unsupervised)
Association rule learning	Infer likely association patterns in data	If you buy hamburger buns, you're likely to buy hamburgers (unsupervised)
Structured output	Create complex output	natural language parse trees, image recognition bounding boxes

ML Mindset

Machines thinking like humans
or
Humans thinking like machines



Identifying suitable problems for ML

- Clear use case for ML
  - Traditional programming is rule-based
  - Problems where a clear approach for developing the solution isn’t clear: identifying objects in a picture
- Data data data
  - A rule of thumb is to have at least thousands of examples for basic linear models, and hundreds of thousands for neural networks.
  - If you have less data, consider a non-ML solution first.
- Knowing the features/signals or the intuition behind it
- Prediction vs Decisions:
  - ML is better at making decisions.
  - Statistical approaches are better suited for finding “interesting” things in the data.

Prediction	Decision
Credit limit based on past spending history	Allowed approval credit limit = 1.2 times the usual spending
What video will the user watch next?	Show those videos in the recommendation bar.

ML Problem: Formulation

1: Describing the problem using simple English

- In plain terms, what would you like your ML Model do?
- Qualitative in nature
- Real goal, not an indirect goal
- Example: We want our ML Model to predict a user's credit limit

2. What's your ideal outcome?

- Incorporating ML model in the product should produce a desirable outcome.
- This outcome may be entirely different from how the model's quality is assessed.
- Multiple outcomes of a single model possible
- Looking beyond what the product has been optimizing for to the larger objective.
- Example: reduce the man-hours spent on deciding credit limit for new applicants of credit cards.

3. What are your success metrics?

- How do you know the system has succeeded? Failed?
- Phrased independently of evaluation metrics
- Tied to the ideal outcome
- Domain/product/team specific
- Are the metrics measurable?
- When are you able to measure them?
- How long will it take for you to know that system is a success or failure?
- Example: Predict the credit limit within 10% range of the manual process
- Example: Reduce the time taken to approve the user for a certain credit limit by 90%

4. What’s the ideal output?

- Write the output you want your models to produce in plain english
- The output must be quantifiable that the machine is capable of producing
- For instance: “User did not enjoy the article” produces much worse results than “User down-voted the article”
- For your ideal output, can you obtain example outputs for training data?

5. How can you use the output?

- Predictions can be made:
  - In real-time as a response to user activity: Online
  - Batch/Cache: Offline
- Define how will the model use these predictions?
- Predictions vs Decisions: we want our model to make decisions, not just predictions.
- Example: if we are trying to predict the number of order’s an e-commerce website might receive on Black Friday, this can help determine the number of compute nodes to spin for ensuring fail proof transactions.

6. Identify the heuristics

- How would you have solved the problem without Machine Learning?
- Write down the answer to this question in plain english
- For instance: to predict the credit limit, you might take monthly average expenditure of the user and approve that as the credit limit

7. Simplify the problem

- Simpler problem formulations are easier to reason about
- Multi class classification to binary classification
- Example: predicting that a news article is fake instead of related/unrelated/agree/disagree

8. Designing data

- Know what data is currently available to the team/ developers
- Use domain expertise of Product Owners to identify what the dataset would look like in an ideal world?
- Analyze if there are requirements for data available from sources outside the current datasets?
- Analyze whether those requirements are feasible to be implemented?: time and money

8. Designing data

Input 1	Input 2	Input 3	Input 4	Input 5
Avg monthly expenditure	Avg. monthly income	Avg credit limit of other customers with similar income	Number of credit defaults	Years of association with the bank

9. Evaluation Metric

- Evaluating your machine learning algorithm is an essential part of any project
- Assess the quality of the model
- Depends on:
  - Outcome of the project
  - Problem statement
  - Dataset at hand
- Different metric for regression and classification problems

Metrics for Regression

- **Mean Absolute Error (MAE)** - average of the absolute differences between the prediction and actual values

$$MeanAbsoluteError = \frac{1}{N}\sum_{j=1}^{N}|y_j - \hat{y}_j|$$

- Gives an idea of the magnitude of the error, but no idea of the direction
- Example : House Price Prediction

Metrics for Regression

- **Mean Square Error (MSE)** - average of the square differences between the prediction and actual values

$$MeanSquaredError = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2$$

- **Root Mean Square Error (RMSE) :** Taking root of MSE and converts the units back to the original units of the output variable

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(p_i - a_i)^2}{n}}$$

- Example : House Price Prediction

Metrics for Regression

- **R Squared** - provides an indication of the goodness of fit of a set of predictions to the actual values. Also, called the coefficient of determination

$$\textit{Coefficient of Deternination} \rightarrow \quad R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

$$\textit{Sum of Squares Total} \rightarrow \quad SST = \sum (y - \bar{y})^2$$

$$\textit{Sum of Squares Regression} \rightarrow \quad SSR = \sum (y' - \overline{y'})^2$$

$$\textit{Sum of Squares Error} \rightarrow \quad SSE = \sum (y - y')^2$$

- Example : House Price Prediction

Metrics for Classification

- **Accuracy** - number of correct predictions made as a ratio of all predictions made

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

- Works well only if there are equal number of samples belonging to each class.
- Example : Classify email spam or not spam

Metrics for Classification

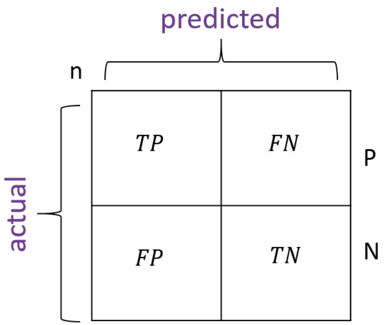
- **Log Loss** - classifier must assign probability to each class for all the samples

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log(p_{ij})$$

- The scalar probability between 0 and 1 can be seen as a measure of confidence for a prediction by an algorithm.
- Example : Classify a set of images of fruits which may be oranges, apples, or pears.

Metrics for Classification

- **Confusion Matrix-** number of correct and incorrect predictions made by the classification model compared to the actual outcomes in the data



Confusion matrix

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

- Used for imbalanced class

Metrics for Classification

- **Area Under the Curve(AUC)**- represents a model's ability to discriminate between positive and negative classes.
- Performance metric for binary classification

$$\frac{\mathrm{TP}}{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \qquad \frac{\mathrm{FP}}{N} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

- An area of 1.0 represents a model that made all predictions perfectly. An area of 0.5 represents a model as good as random
- Used for imbalanced classnvbv

Metrics for Classification

- **F1 Score** - Harmonic Mean between precision and recall. tell sow precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2}\right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

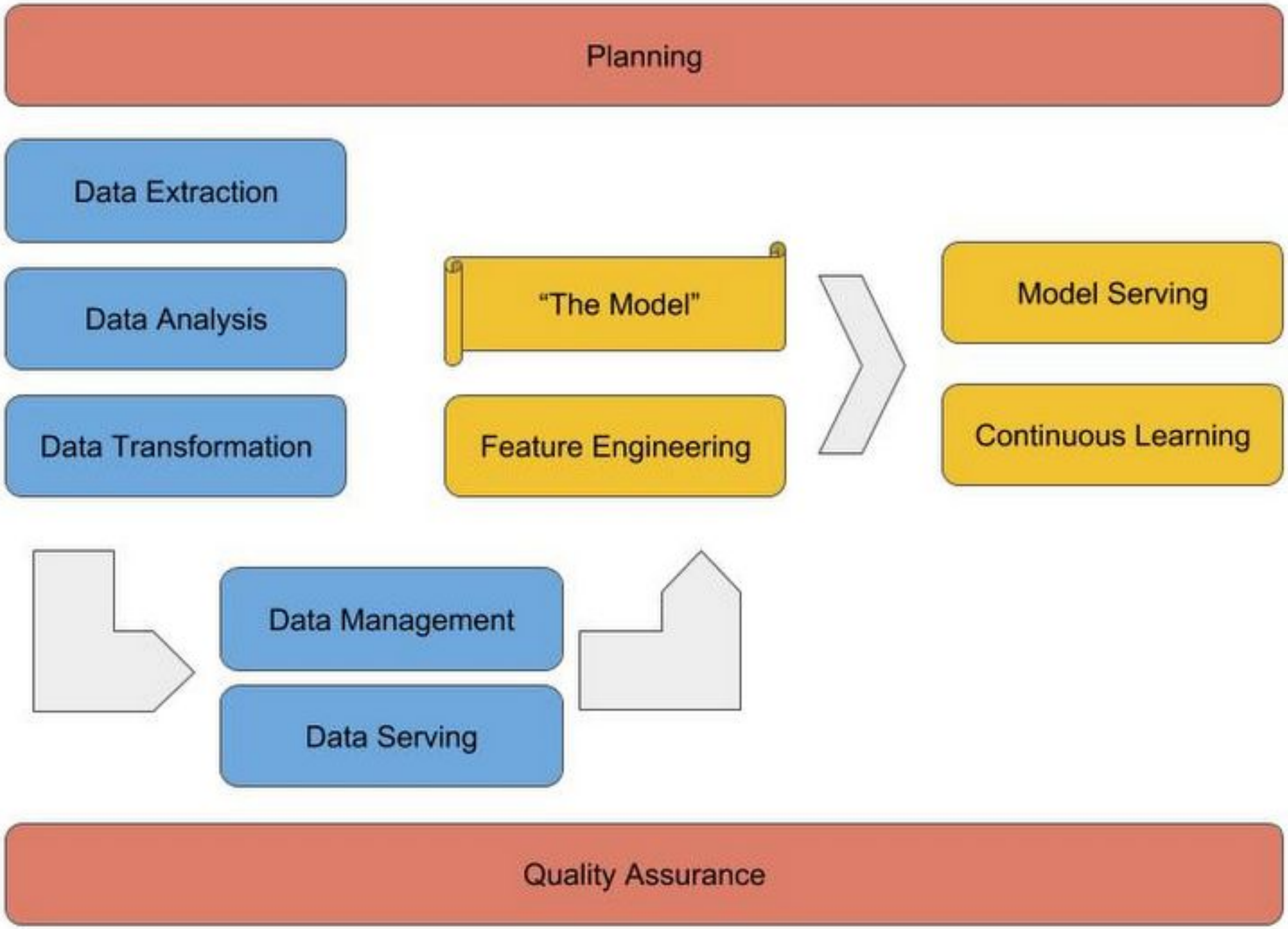
- Range from [0, 1]
- F1 Score tries to find the balance between precision and recall

10. Formalism

Example: ML Model that predicts which tweets will get retweets

- **Task** (*T*): Classify a tweet that has not been published as going to get retweets or not.
- **Experience** (*E*): A corpus of tweets for an account where some have retweets and some do not.
- **Performance** (*P*): Classification accuracy, the number of tweets predicted correctly out of all tweets considered as a percentage.

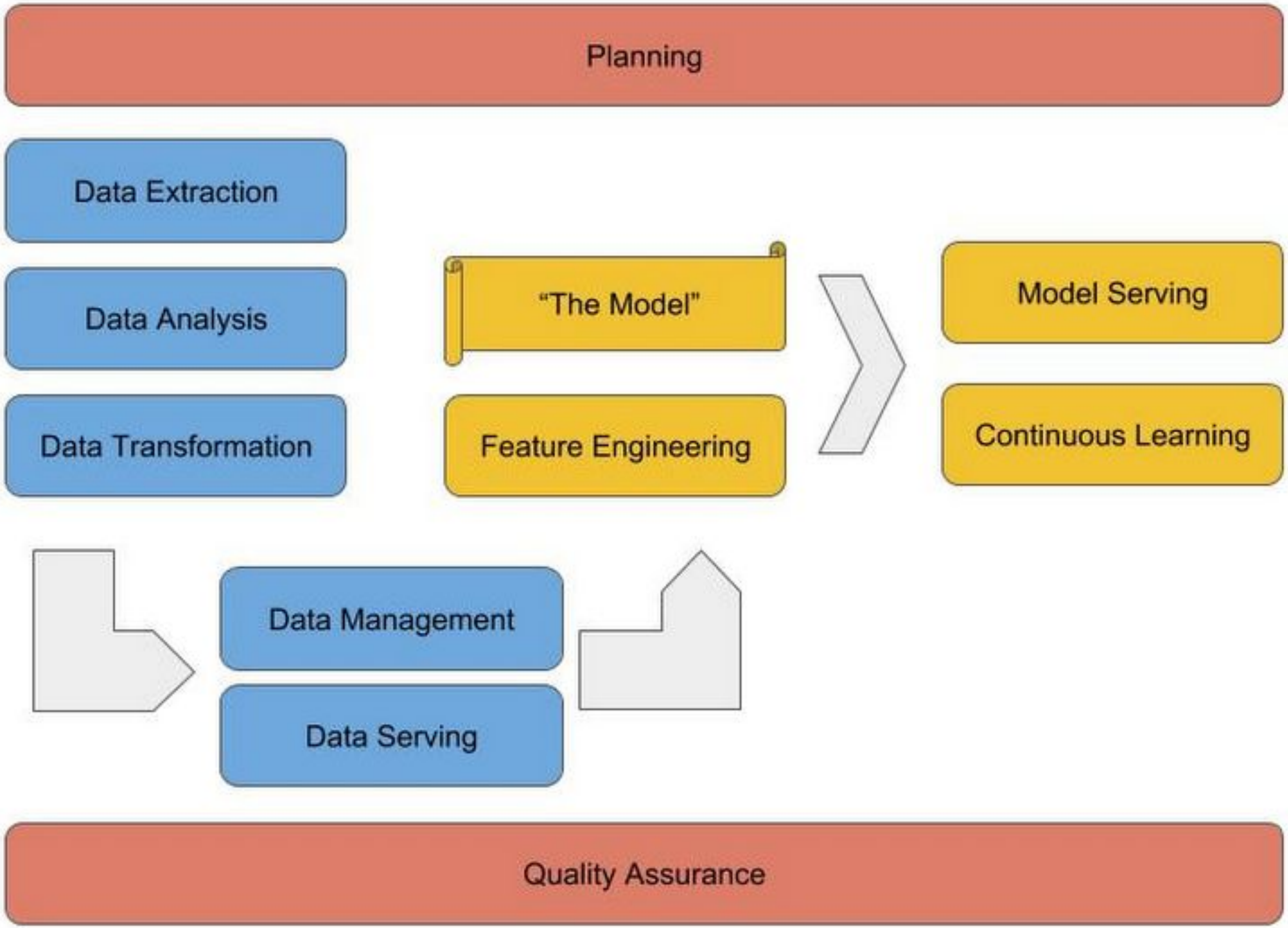
ML Pipeline



Machine Learning Ecosystem - Google, LLC ©

Planning

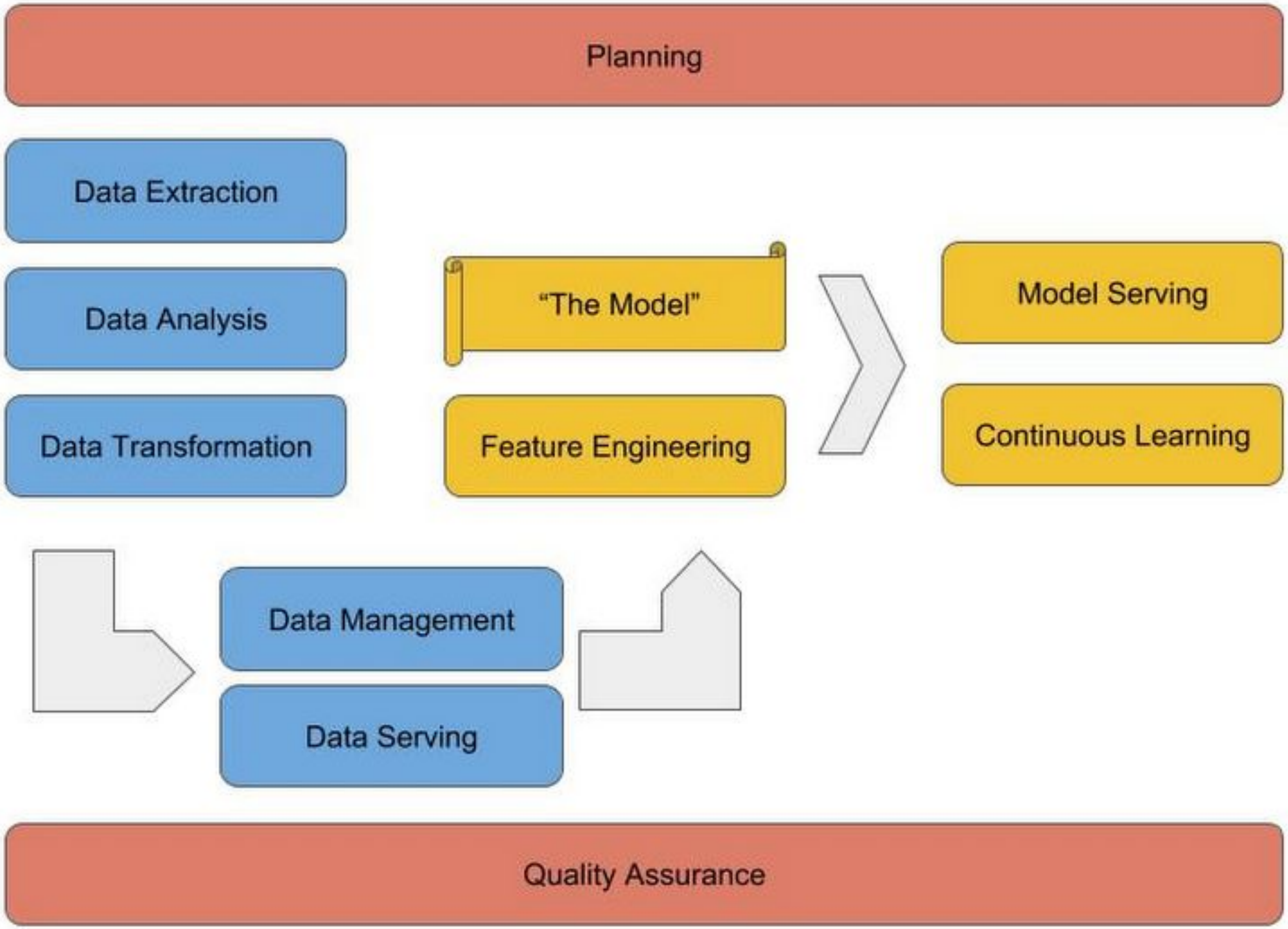
- ML model is an algorithm that is learned and updated dynamically
- Once an algorithm is released in production, it may not perform as planned prompting the team to rethink, redesign and rewrite
- New set of challenges that require Product Owners, Engineering and Quality Assurance teams to work together
- Example: daily standups
- Typically, you develop policies to address user issues in a SE application but with machine learning we are learning these policies in real-time
- Planning is embedded in all stages



Machine Learning Ecosystem - Google, LLC ©

Data Engineering

- 80% of time and resources is spent on data engineering
- Activites:
  - Data Collection
  - Data Extraction
  - Data Transformation
  - Data Storage
  - Data Serving
- Tools used: SQL/ NoSQL, Hadoop, Apache Spark, ETL Pipelines



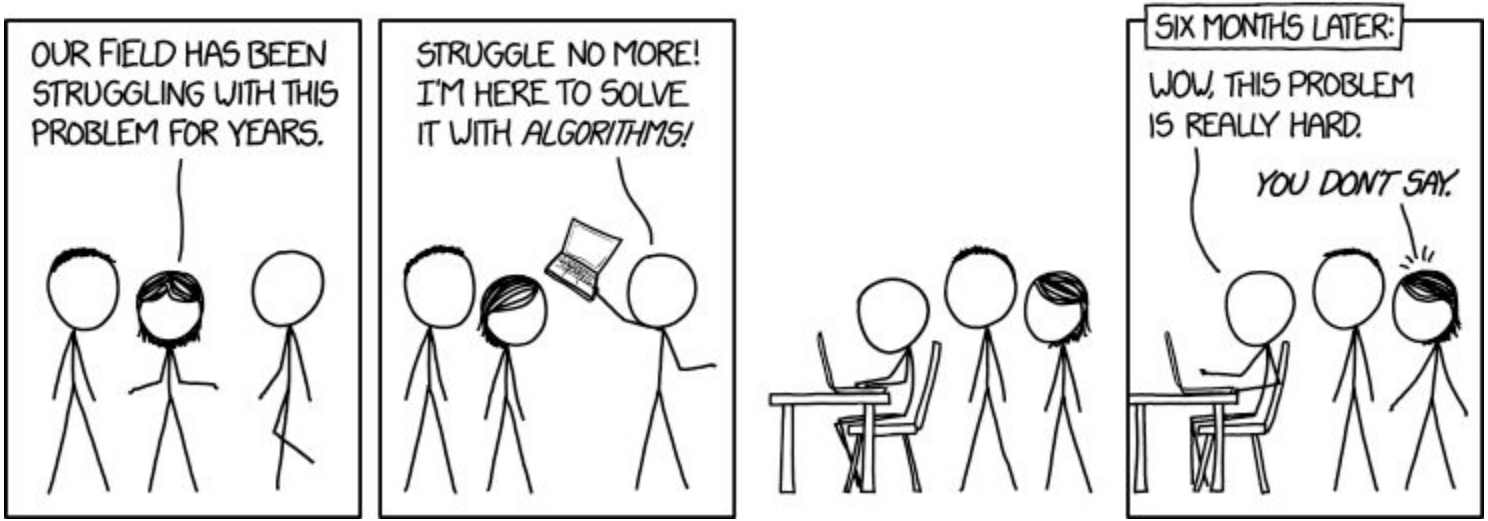
Machine Learning Ecosystem - Google, LLC ©

Modelling

- Split the data into training, validation and testing set
- Feature engineering
- Offline vs online learning
- Hyper-parameter tuning using validation set: dependent on the algorithm being used and problem that we are attempting to solve
- One-shot training is only effective in academic and single-task use cases
- Evaluation using the pre-defined metric for candidate model

References

1. Zhang, Du, and Jeffrey JP Tsai. "Machine learning and software engineering." *Software Quality Journal* 11.2 (2003): 87-119.
2. Meng, Xiangrui, et al. "Mllib: Machine learning in apache spark." *The Journal of Machine Learning Research* 17.1 (2016): 1235-1241.
3. Deploying Machine Learning Models https://christophergs.github.io/machine%20learning/2019/03/17/how-to-deploy-machine-learning-models/
4. Defining Machine Learning Problem: https://machinelearningmastery.com/how-to-define-your-machine-learning-problem/
5. Machine Learning Model from Scratch: https://towardsdatascience.com/machine-learning-general-process-8f1b510bd8af
6. Data: A key requirement for ML Product https://medium.com/thelaunchpad/data-a-key-requirement-for-your-machine-learning-ml-product-9195ace977d4


OUR FIELD HAS BEEN STRUGGLING WITH THIS PROBLEM FOR YEARS.
STRUGGLE NO MORE! I'M HERE TO SOLVE IT WITH *ALGORITHMS!*
SIX MONTHS LATER:
WOW, THIS PROBLEM IS REALLY HARD.
YOU DON'T SAY.

Questions?