# Designing a Machine Learning System -II

## Types of Machine Learning Systems

ML systems can be categorized based on three aspects:

- How their ML models serve their predictions (batch prediction vs. online prediction)
- Where the majority of computation is done (edge computing vs. cloud computing)
- How often their ML models get updated (online learning vs. offline learning)

## Batch prediction vs. online prediction

- Two options for predictions offered by major cloud providers
- Valid when using your own data centers too
- In both cases you pass input data to a cloud-hosted machine-learning model and get inferences for each data instance

Google Cloud

## Batch prediction vs. online prediction

- Batch prediction:
  - Generate predictions periodically
  - Predictions are stored somewhere (e.g. SQL tables, CSV files)
  - Retrieve them as needed
  - Allow more complex models
- Online prediction:
  - Generate predictions as requests arrive
  - Predictions are returned as responses

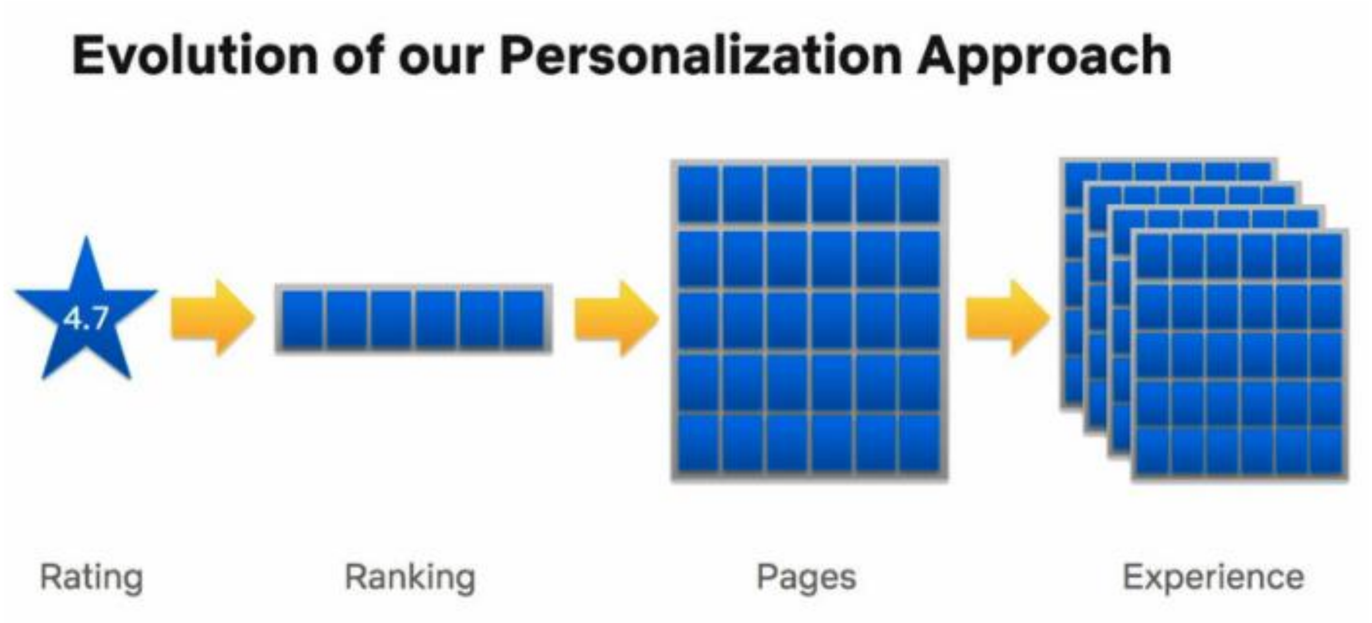
| | **Batch prediction** | **Online prediction** |
|---|---|---|
| **Frequency** | Periodical (e.g. every 4 hours) | As soon as requests come |
| **Useful for** | Processing accumulated data when you don't need immediate results (e.g. recommendation systems) | When predictions are needed as soon as data sample is generated (e.g. fraud detection) |
| **Optimized** | High throughput | Low latency |
| **Input space** | Finite: need to know how many predictions to generate | Can be infinite |
| **Examples** | ● TripAdvisor hotel ranking<br>● Netflix recommendations | ● Google Assistant speech recognition<br>● Twitter feed |

## Netflix

- Netflix is now synonymous to most people as the go-to streaming service for movies and tv shows.
- It has transformed from a mail service posting DVDs in the US (late 90s) to a global streaming service with 182.8 million subscribers.

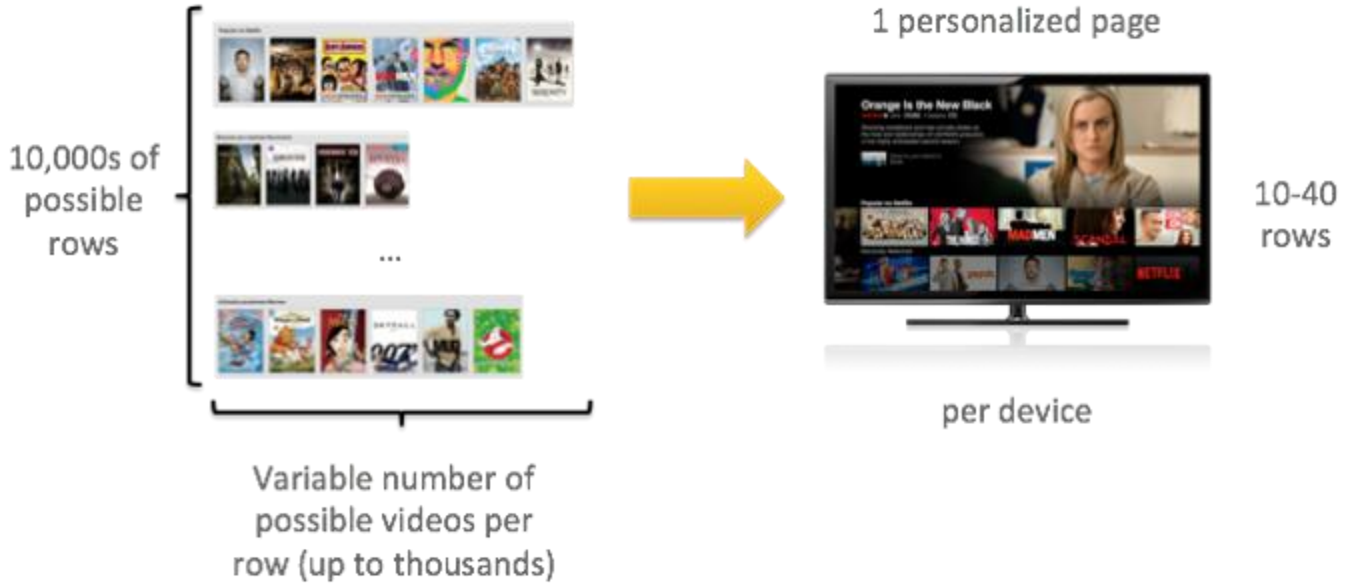
https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## Netflix



https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

# Netflix Recommender System

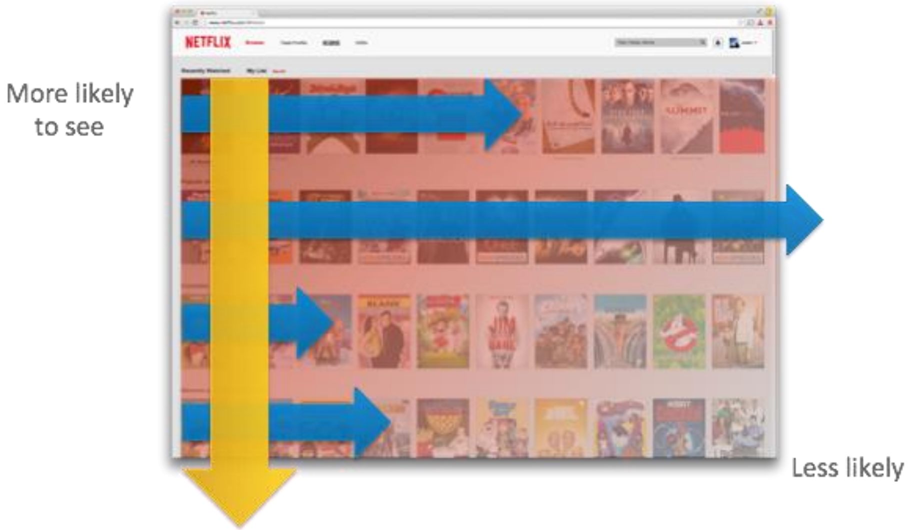


https://netflixtechblog.com/learning-a-personalized-homepage-aa8ec670359a

## Netflix Recommender System

Netflix utilizes a two-tiered row-based ranking system, where ranking happens:

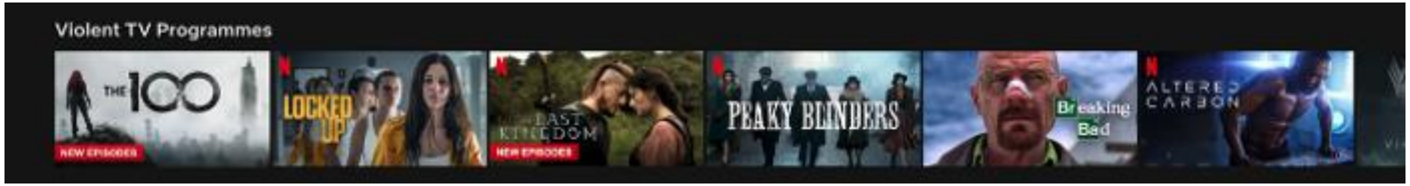
- Within each row (strongest recommendations on the left)
- Across rows (strongest recommendations on top)



https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## Netflix – Ranking Algorithms

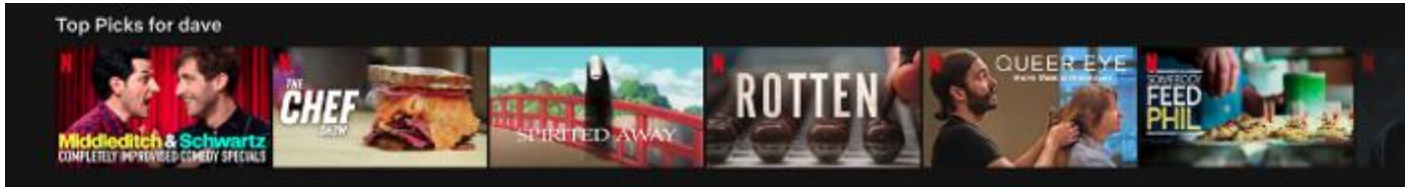
**Personalised Video Ranking (PVR)** — A general-purpose algorithm, which filters down the catalog by a certain criteria (e.g. Violent TV Programmes, US TV shows, Romance, etc), combined with side features including user features and popularity. Title recommendations use **batch predictions**



https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

# Netflix – Ranking Algorithms

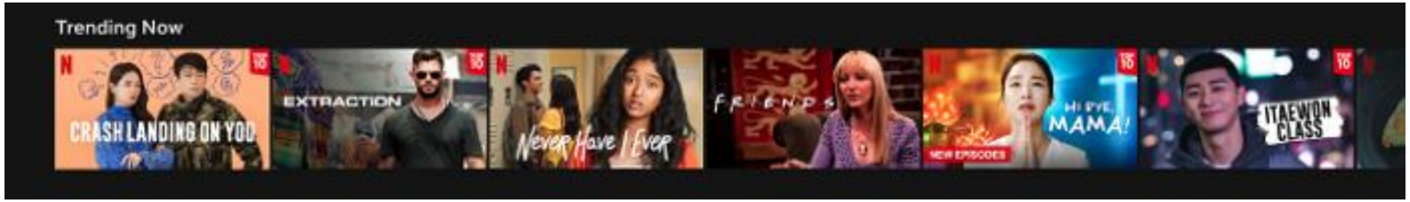
**Top-N Video Ranker** — Similar to PVR except that it is optimized using metrics that look at the top percentiles of the catalog ranking (e.g. MAP@K, NDCG).



https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

# Netflix – Ranking Algorithms

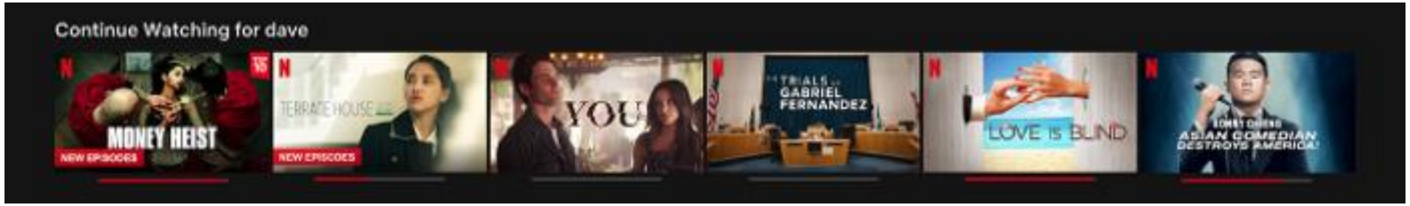
- **Trending Now Ranker** — This algorithm captures temporal trends which Netflix deduces to be strong predictors. These short-term trends can range from a few minutes to a few days.
- These events/trends are typically:
  - Events that have a seasonal trend and repeat themselves (e.g. Valentines day leads to an uptick in Romance videos being consumed)
  - One-off, short term events (e.g. recent disasters leading to short-term interest in documentaries about them)



https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## Netflix – Ranking Algorithms

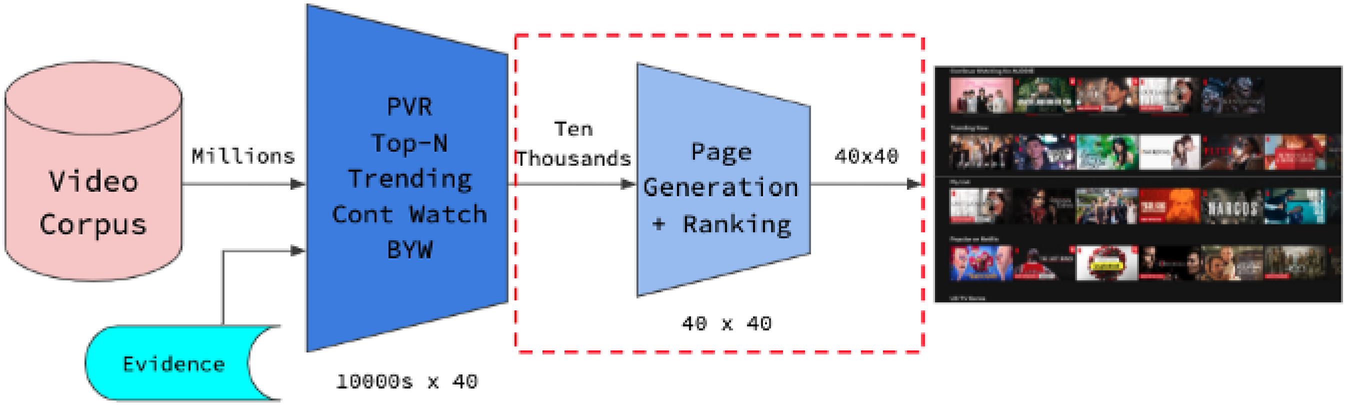
- **Continue Watching Ranker** — This algorithm looks at items that the member has consumed but has not completed, typically:
- Episodic content (e.g. TV series)
- Non-episodic content that can be consumed in small bites (e.g. movies that are half-completed, series that are episode independent such as Black Mirror)
- The algorithm calculates the probability of the member continue watching and includes other context-aware signals (e.g. time elapsed since viewing, point of abandonment, device watched on).


Continue Watching for dave
MONEY HEIST
NEW EPISODES
TERRACE HOUSE
NEW EPISODES
YOU
THE TRIALS OF GABRIEL FERNANDEZ
LOVE IS BLIND
ASIAN COMEDIAN DESTROYS AMERICA!

https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## Netflix Page Generation Process

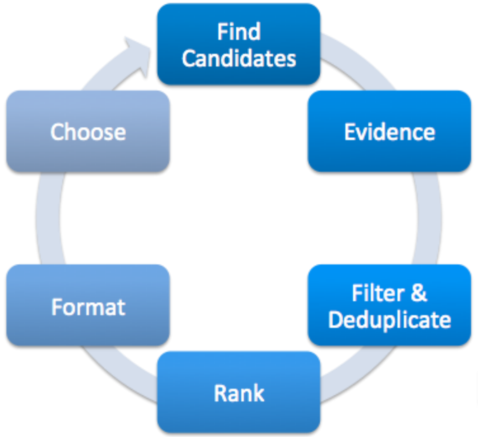
- After the algorithms generate candidate rows (already ranked within each row vector), it now needs to decide which of these 10,000s of rows to display?



https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## Netflix Page Generation Process

- Now there is a set of possible rows to consider for a page, it is time to assemble the homepage from them
- First find candidate groupings that are likely relevant for a member based on the information available for that member.
- This also involves coming up with the evidence (or explanations) to support the presentation of a row, for example the movies that the member has previously watched in a genre.

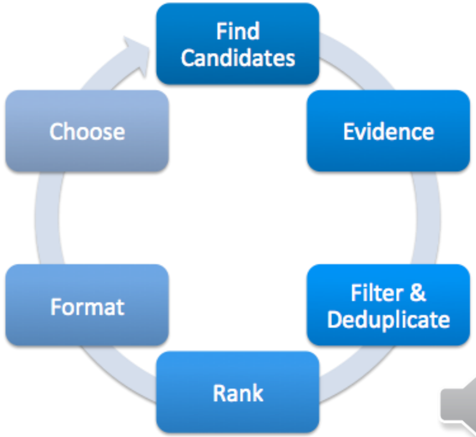


https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## Netflix Page Generation Process

- Next, each group is filtered to handle concerns like maturity rating or to remove some previously watched videos.
- After filtering, the videos in each group are ranked according to a row-appropriate ranking algorithm, which produces an ordering of videos such that the most relevant videos for the member in a group are at the front of the row.
- From this set of row candidates a row selection algorithm is run to assemble the full page.

Row selection use **online predictions**



https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

# Edge computing vs. cloud computing

| | **Cloud computing** | **Edge computing** |
|---|---|---|
| **Computations** | Done on cloud (servers) | Done on edge devices (browsers, phones, tablets, laptops, smart watches, activity watchers, cars, etc.) |
| **Requirements** | **Network connections**:<br>availability and speed for data transfer | **Hardware**:<br>memory, compute power, energy for doing computations |
| **Examples** | Most queries to Alexa, Siri, Google Assistant<br><br>Google Translate for rare language pairs (e.g. English - Yiddish) | Wake words for Alexa, Siri, Google Assistant<br><br>Google Translate for popular language pairs (e.g. English - Spanish)<br><br>Predictive text<br><br>Unlocking with fingerprints, faces |

## Benefits of edge computing

- Can work without (Internet) connections or with unreliable connections
  - Many companies have strict no-Internet policy
  - **Caveat**: devices are capable of doing computations but apps need external information
    - e.g. ETA needs external real-time traffic information to work well
- Don’t have to worry about network latency
  - Network latency might be a bigger problem than inference latency
  - Many use cases are impossible with network latency
    - e.g. predictive texting
- Fewer concerns about privacy
  - Don’t have to send user data over networks (which can be intercepted)
  - Cloud database breaches can affect many people
  - Easier to comply with regulations (e.g. GDPR)
  - **Caveat**: edge computing might make it easier to steal user data by just taking the device
- Cheaper
  - The more computations we can push to the edge, the less we have to pay for servers

# Offline learning vs. Online learning

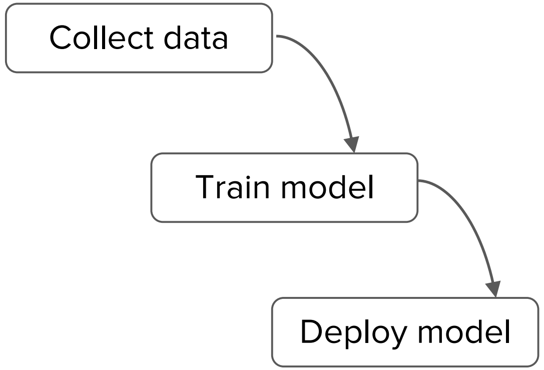
| | Offline learning | Online learning |
|---|---|---|
| **Iteration cycle** | Periodical (months) | Continual (minutes) $\neq$ continuous |
| **Batch size** | batch (thousands -> millions of samples)<br>GPT-3 125M params: batch size 0.5M<br>GPT-3 175B params: batch size 3.2M | microbatch (hundreds of samples) |
| **Data usage** | Each sample seen multiple times (epochs) | Each sample seen at most once |
| **Evaluation** | Mostly offline evaluation | Offline evaluation as sanity check<br>Mostly relying on online evaluation (A/B testing) |
| **Examples** | Most applications | TikTok recommendation system<br><br>Twitter hashtag trending |

## Online learning vs. offline learning

- Both can be done together to create more stable systems
- **If the infrastructure is set up right**, there's no fundamental difference between online learning and offline learning, just a hyperparameter to tune.

## ML in production: expectation

1. Collect data
2. Train model
3. Deploy model



**Waterfall model**

Slide from: CS 329 | Chip Huyen | cs329s.stanford.edu

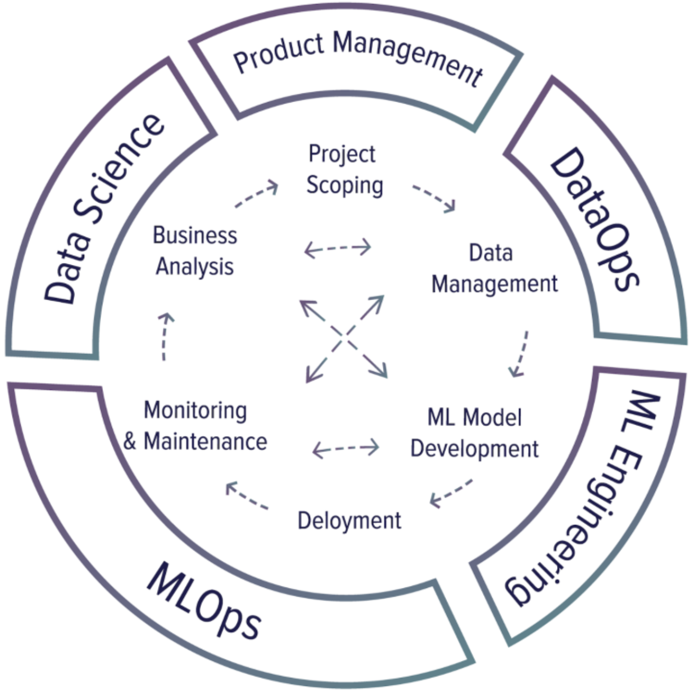
## ML in production: reality

1. Choose a metric to optimize
2. Collect data
3. Train model
4. Realize many labels are wrong -> relabel data
5. Train model
6. Model performs poorly on one class -> collect more data for that class
7. Train model
8. Model performs poorly on most recent data -> collect more recent data
9. Train model
10. Deploy model
11. There are complaints that model biases against one group! -> revert to older version
12. Get more data, train more, do more testing
13. Deploy model
14. Model performs well but revenue decreasing
15. Choose a different metric
16. Start over

Slide adapted from: CS 329 | Chip Huyen | cs329s.stanford.edu

## ML in production: reality

1. Choose a metric to optimize
2. Collect data
3. Train model
4. Realize many labels are wrong -> relabel data
5. Train model
6. Model performs poorly on one class -> collect more data fo
7. Train model
8. Model performs poorly on most recent data -> collect more
9. Train model
10. Deploy model
11. There are complaints that model biases against one group!
12. Get more data, train more, do more testing
13. Deploy model
14. Model performs well but revenue decreasing
15. Choose a different metric
16. Start over

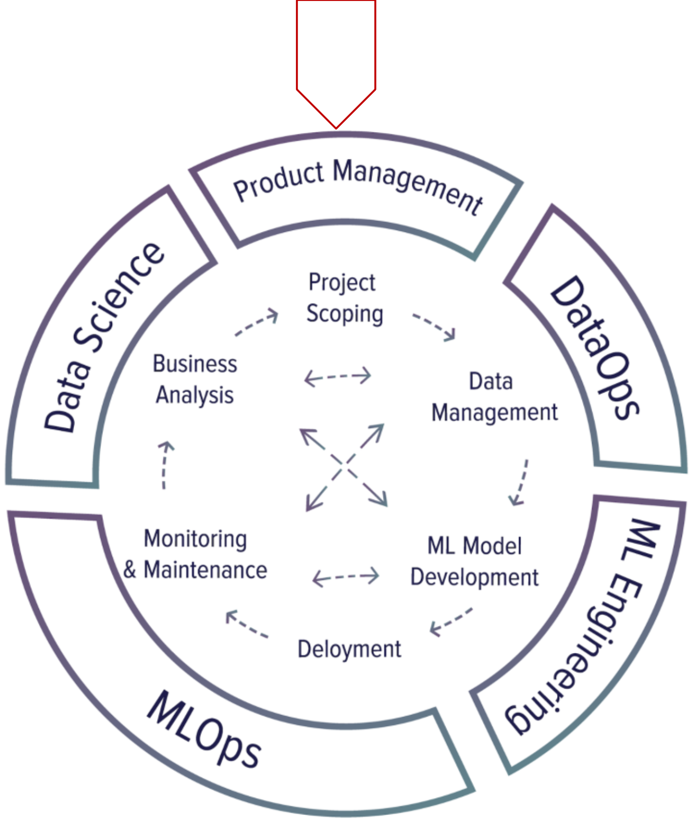


**Iterative development**

Slide from: CS 329 | Chip Huyen | cs329s.stanford.edu

## Project scoping

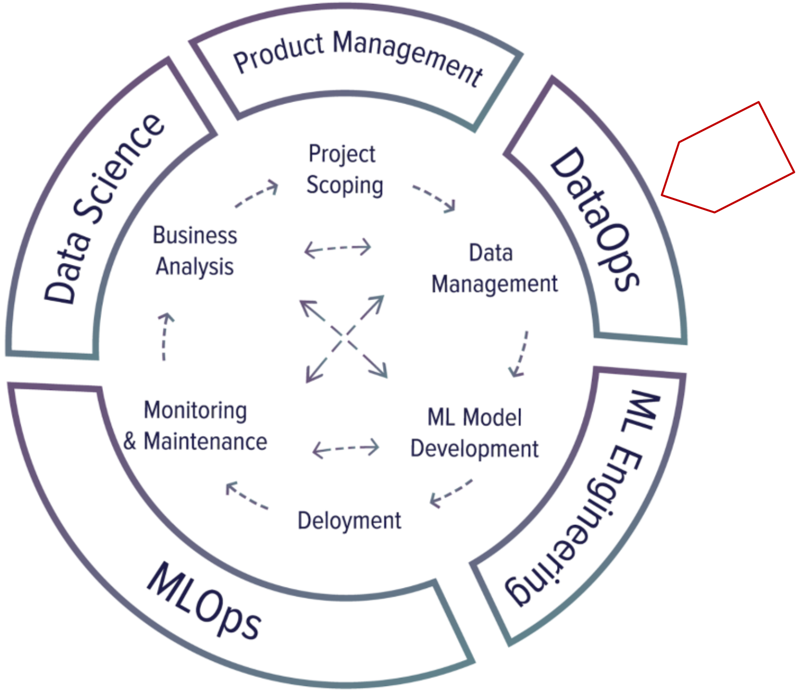
- Goals & objectives
- Constraints
- Evaluation



Slide from: CS 329 | Chip Huyen | cs329s.stanford.edu

# Data management

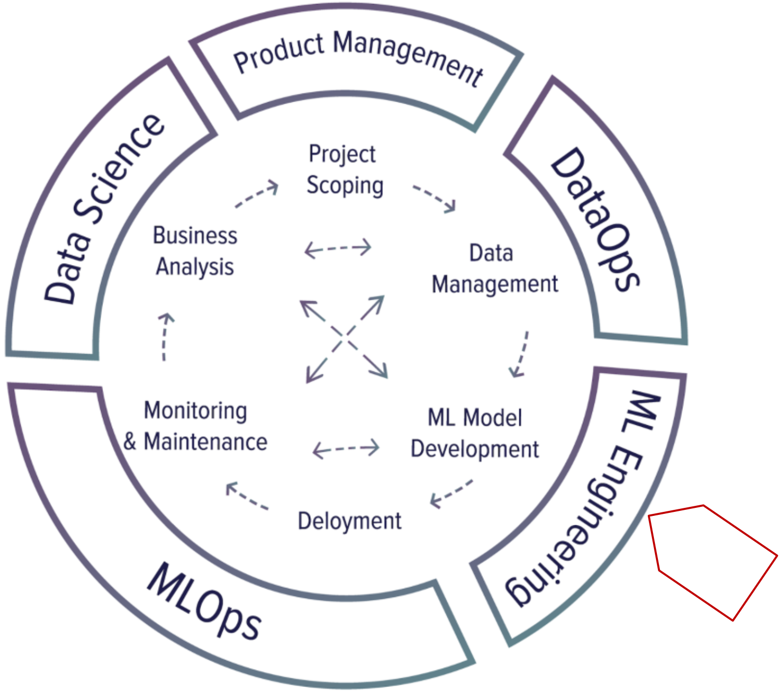
- Data sources
- Data format
- Processing
- Storage
- Data consumer
- Data controller



Slide from: CS 329 | Chip Huyen | cs329s.stanford.edu

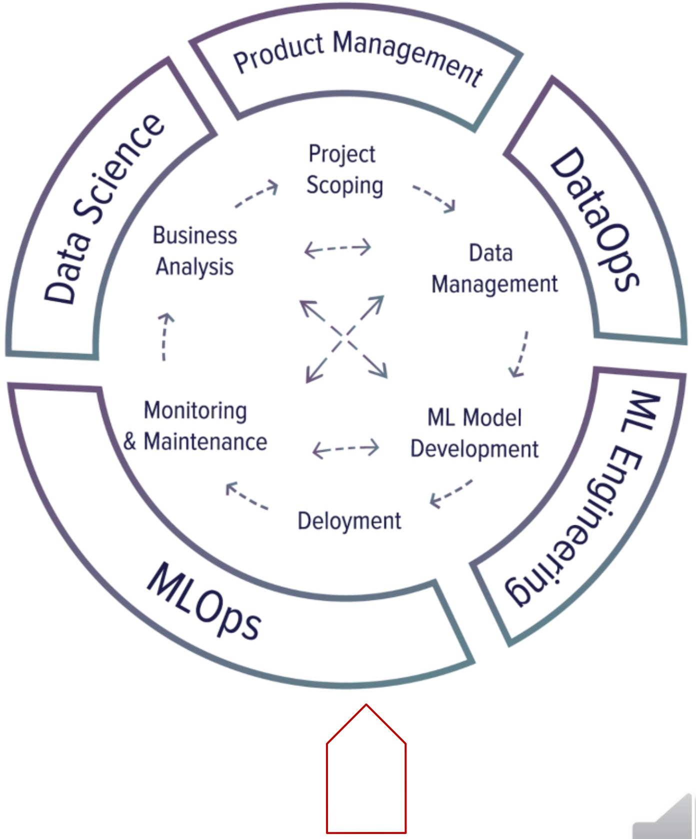
# Model development

- Dataset creation
- Feature engineering
- Model training
- Offline model evaluation



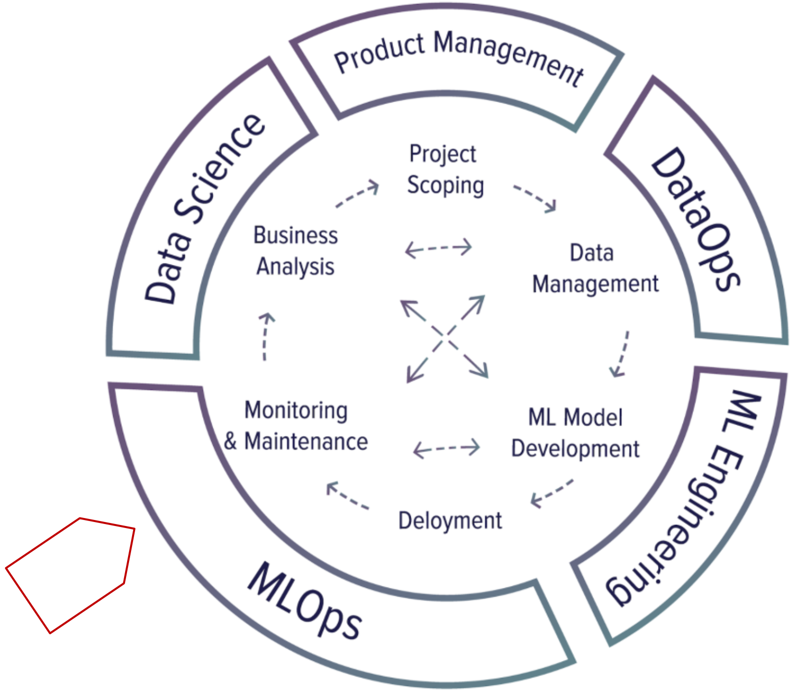
# Deployment

- Deploying and serving
- Release strategies
- Online model evaluation



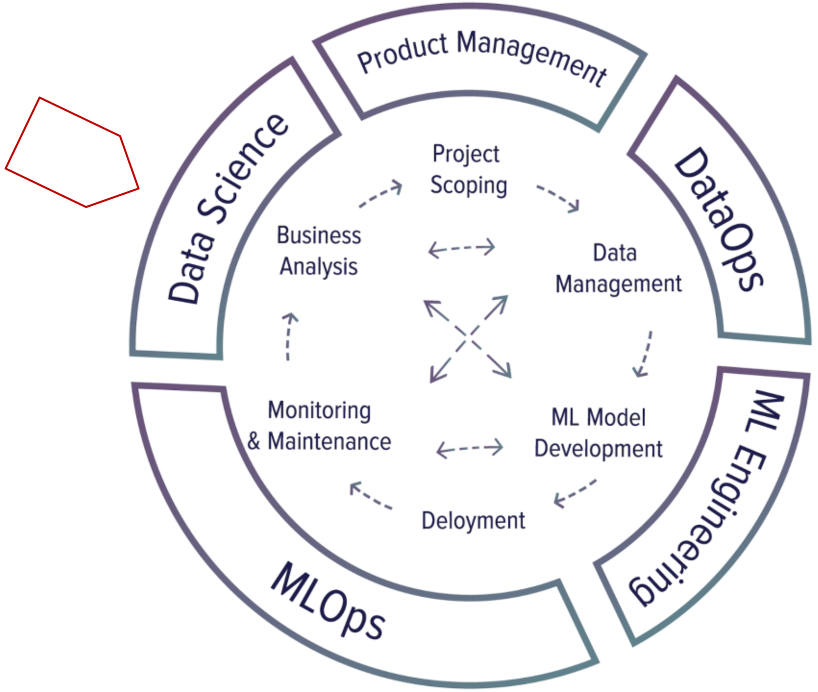
## Monitoring & maintenance

- Model performance & data monitoring
- Model retraining
- Model updates



## Business analysis

- User experience
- Tying model performance to business performance



# Project scoping

## Goals

The ultimate goal of an ML project or any project within a business is, to increase profits directly or indirectly.

- Directly: increasing sales (ads, conversion rates), cutting costs
- Indirectly: increasing customer satisfaction, increasing time spent on a website

**Non-profits are exceptions**: lots of exciting applications of AI for social good

- environment (climate change, deforestation, flood risk, etc.)
- public health
- education (intelligent tutoring system, personalized learning)

## ML to business performance can be confusing

An ML model that gives customers more personalized solutions can either:

- make them happier which makes them spend more money
- solve their problems faster which makes them spend less money

## Example

Possible goals when building a ranking system for newsfeed?

1. Minimize the spread of misinformation
2. Maximize revenue from sponsored content
3. Maximize engagement

## Objectives

| Goals | Objectives |
|---|---|
| General purpose of a project | Specific steps on how to realize that purpose |

## Example: ranking system for newsfeed

| **Goals** | **Objectives** |
|---|---|
| General purpose of a project | Specific steps on how to realize that purpose |
| Maximize users' engagement | 1. Filter out spam<br>2. Filter out NSFW content<br>3. Rank posts by engagement: how likely users will click on it |

# Example: ranking system for wholesome newsfeed

| **Goals** | **Objectives** |
|---|---|
| General purpose of a project | Specific steps on how to realize that purpose |
| **Maximize users' engagement** while **minimizing the spread of extreme views** and **misinformation** | 1. Filter out spam<br>2. Filter out NSFW content<br>3. Filter out misinformation<br>4. Rank posts by quality<br>5. Rank posts by engagement: how likely users will click on it |

## Example: ranking system for wholesome newsfeed

| **Goals** | **Objectives** |
|---|---|
| General purpose of a project | Specific steps on how to realize that purpose |
| **Maximize users' engagement** while **minimizing the spread of extreme views** and **misinformation** | 1. Filter out spam<br>2. Filter out NSFW content<br>3. Filter out misinformation<br>4. Rank posts by quality<br>5. Rank posts by how likely users will click on it |

How to combine these two objectives?

## Multiple objective optimization (MOO)

- Rank posts by quality
  - Predict posts’ quality
  - Minimize **quality_loss:** difference between predicted quality and true quality

- Rank posts by how likely users will click on it
  - Predict posts’ engagement
  - Minimize **engagement_loss**: difference between predicted clicks and true clicks

## MOO: one model with combined loss

- Rank posts by quality
  - Predict posts' quality
  - Minimize **quality_loss:** difference between predicted quality and true quality

- Rank posts by how likely users will click on it
  - Predict posts' engagement
  - Minimize **engagement_loss**: difference between predicted clicks and true clicks

$$\underbrace{\text{loss} = \alpha \ \text{quality\_loss} + \beta \ \text{engagement\_loss}}_{}$$

Train one model to minimize this combined loss
Tune $\alpha$ and $\beta$ to meet your need

## MOO: one model with combined loss

- Rank posts by quality
  - Predict posts' quality
  - Minimize **quality_loss:** difference between predicted quality and true quality

- Rank posts by how likely users will click on it
  - Predict posts' engagement
  - Minimize **engagement_loss**: difference between predicted clicks and true clicks

$$\underbrace{\text{loss} = \alpha\ \text{quality\_loss} + \beta\ \text{engagement\_loss}}_{\text{Every time you want to tweak } \alpha \text{ and } \beta\text{, you have to retrain your model!}}$$

# MOO: combine different models

- Rank posts by quality
  - Predict posts' quality
  - Minimize **quality_loss:** difference between predicted quality and true quality

- Rank posts by how likely users will click on it
  - Predict posts' engagement
  - Minimize **engagement_loss**: difference between predicted clicks and true clicks

$\mathbf{M_q}$: optimizes **quality_loss**
$\mathbf{M_e}$: optimizes **engagement_loss**

Rank posts by $\alpha$ $\mathbf{M_q(post)}$ + $\beta$ $\mathbf{M_e(post)}$

Now you can tweak $\alpha$ and $\beta$ without retraining models

## Decouple different objectives

- Easier for training:
  - Optimizing for one objective is easier than optimizing for multiple objectives
- Easier to tweak your system:
  - E.g. $\alpha$ % model optimized for quality + $\beta$ % model optimized for engagement
- Easier for maintenance:
  - Different objectives might need different maintenance schedules
    - Spamming techniques evolve much faster than the way post quality is perceived
    - Spam filtering systems need updates more frequently than quality ranking systems

## Netflix Prize

- In 2000, Netflix introduced personalized movie recommendations and in 2006, launched **Netflix Prize,** a machine learning and data mining competition with a $1 million dollar prize money.
- Back then, Netflix used *Cinematch*, its proprietary recommender system which had a *root mean squared error* (RMSE) of 0.9525 and challenged people to beat this benchmark by 10%.
- The team who could achieve the target or got close to this target after a year would be awarded the prize money.

https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## Netflix Prize - 2007

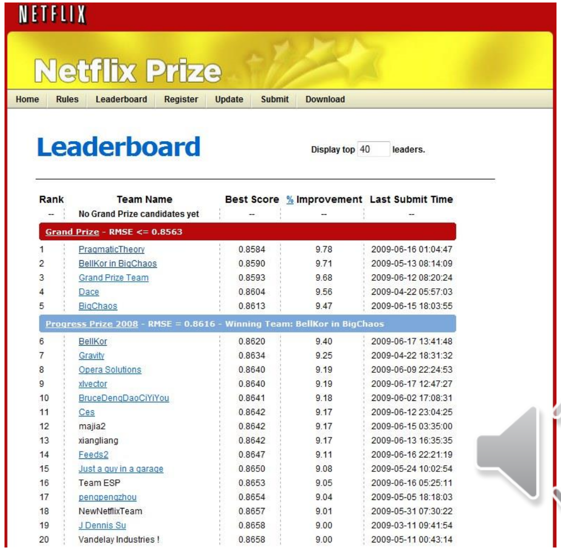
- The winner in 2007 used a linear combination of *Matrix Factorization* and *Restricted Boltzmann Machines*, achieving a RMSE of 0.88. Netflix then put those algorithms into production after some adaptations.

https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

# Netflix Prize - 2009

- Despite some teams achieving a RMSE of 0.86 in 2009, the company did not put those algorithms into production due to the engineering effort required to gain the marginal increase in accuracy.

An important point in real-life recommender systems — that there is always a positive relationship between model improvements and engineering efforts.



Leaderboard

Display top 40 leaders.

| Rank | Team Name | Best Score | % Improvement | Last Submit Time |
|---|---|---|---|---|
| -- | No Grand Prize candidates yet | -- | -- | -- |
| Grand Prize - RMSE <= 0.8563 | | | | |
| 1 | PragmaticTheory | 0.8584 | 9.78 | 2009-06-16 01:04:47 |
| 2 | BellKor in BigChaos | 0.8590 | 9.71 | 2009-05-13 08:14:09 |
| 3 | Grand Prize Team | 0.8593 | 9.68 | 2009-06-12 08:20:24 |
| 4 | Dace | 0.8604 | 9.56 | 2009-04-22 05:57:03 |
| 5 | BigChaos | 0.8613 | 9.47 | 2009-06-15 18:03:55 |
| Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos | | | | |
| 6 | BellKor | 0.8620 | 9.40 | 2009-06-17 13:41:48 |
| 7 | Gravity | 0.8634 | 9.25 | 2009-04-22 18:31:32 |
| 8 | Opera Solutions | 0.8640 | 9.19 | 2009-06-09 22:24:53 |
| 9 | xlvector | 0.8640 | 9.19 | 2009-06-17 12:47:27 |
| 10 | BruceDengDaoCiYiYou | 0.8641 | 9.18 | 2009-06-02 17:08:31 |
| 11 | Ces | 0.8642 | 9.17 | 2009-06-12 23:04:25 |
| 12 | majia2 | 0.8642 | 9.17 | 2009-06-15 03:35:00 |
| 13 | xiangliang | 0.8642 | 9.17 | 2009-06-13 16:35:35 |
| 14 | Feeds2 | 0.8647 | 9.11 | 2009-06-16 22:21:19 |
| 15 | Just a guy in a garage | 0.8650 | 9.08 | 2009-05-24 10:02:54 |
| 16 | Team ESP | 0.8653 | 9.05 | 2009-06-16 05:25:11 |
| 17 | pengpengzhou | 0.8654 | 9.04 | 2009-05-05 18:18:03 |
| 18 | NewNetflixTeam | 0.8657 | 9.01 | 2009-05-31 07:30:22 |
| 19 | J Dennis Su | 0.8658 | 9.00 | 2009-03-11 09:41:54 |
| 20 | Vandelay Industries ! | 0.8658 | 9.00 | 2009-05-11 00:43:14 |

https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48

## When to use ML

Machine learning is an approach to <u>learn</u> <u>complex patterns</u> from <u>existing data</u> and use these patterns to make <u>predictions</u> on <u>unseen data</u>.

## When not to use ML

- It is unethical.
- Simpler solutions do the trick.
- It is impossible to get the right data.
- One single prediction error can cause devastating consequences.
- Every single decision the system makes must be explainable.
- It’s not cost-effective.

## If ML is nott the solution, it can be part of it

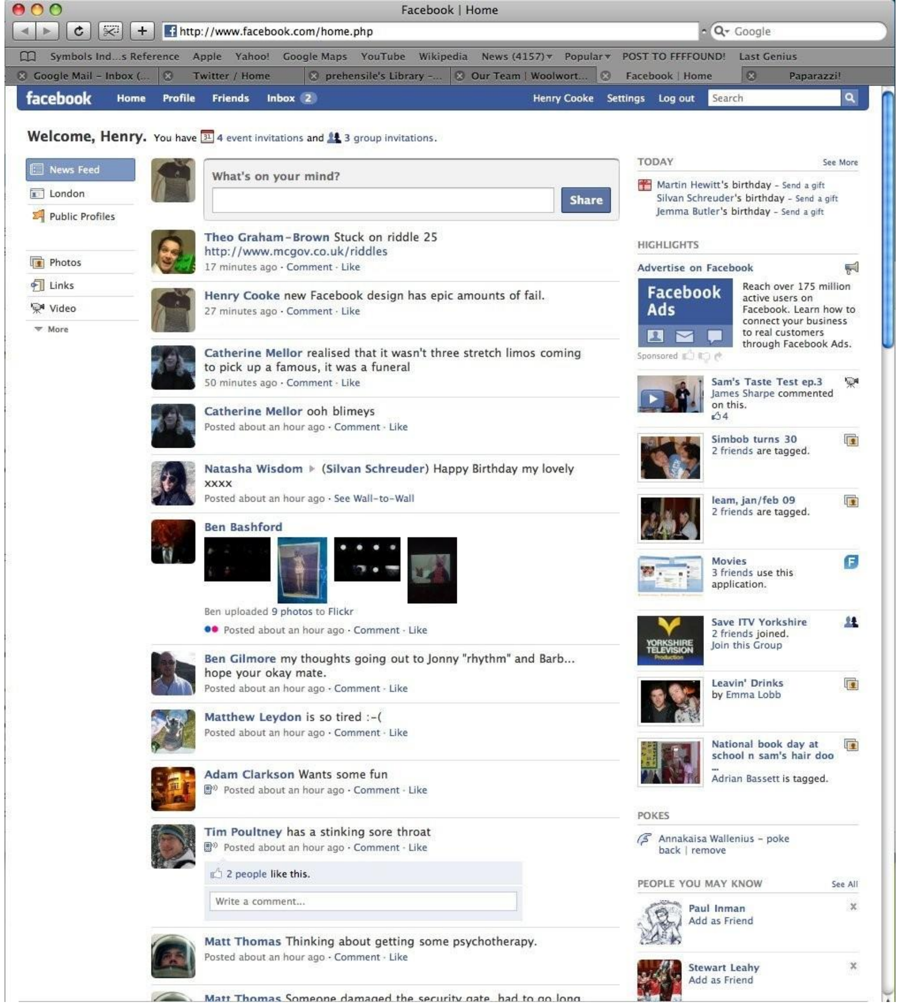
- Break your problem into smaller problems. ML might be able to solve some of them.
    - If a chatbot can’t answer all customers’ queries, build an ML model to predict whether a query matches one of the frequently asked questions.
    - Human-in-the-loop

## Four phases of ML adoption

### Phase 1: Before ML

“If you think that machine learning will give you a 100% boost, then a heuristic will get you 50% of the way there.”

Martin Zinkevich, Google


Facebook | Home

http://www.facebook.com/home.php

Symbols Ind...s Reference · Apple · Yahoo! · Google Maps · YouTube · Wikipedia · News (4157) · Popular · POST TO FFFFOUND! · Last Genius

Google Mail – Inbox (... · Twitter / Home · prehensile's Library –... · Our Team | Woolwort... · Facebook | Home · Paparazzi!

facebook · Home · Profile · Friends · Inbox 2 · Henry Cooke · Settings · Log out · Search

**Welcome, Henry.** You have 4 event invitations and 3 group invitations.

News Feed · London · Public Profiles · Photos · Links · Video · More

What's on your mind? Share

**Theo Graham-Brown** Stuck on riddle 25
http://www.mcgov.co.uk/riddles
17 minutes ago · Comment · Like

**Henry Cooke** new Facebook design has epic amounts of fail.
27 minutes ago · Comment · Like

**Catherine Mellor** realised that it wasn't three stretch limos coming to pick up a famous, it was a funeral
50 minutes ago · Comment · Like

**Catherine Mellor** ooh blimeys
Posted about an hour ago · Comment · Like

**Natasha Wisdom** ▸ (**Silvan Schreuder**) Happy Birthday my lovely xxxx
Posted about an hour ago · See Wall-to-Wall

**Ben Bashford**
Ben uploaded 9 photos to Flickr
Posted about an hour ago · Comment · Like

**Ben Gilmore** my thoughts going out to Jonny "rhythm" and Barb... hope your okay mate.
Posted about an hour ago · Comment · Like

**Matthew Leydon** is so tired :-(
Posted about an hour ago · Comment · Like

**Adam Clarkson** Wants some fun
Posted about an hour ago · Comment · Like

**Tim Poultney** has a stinking sore throat
Posted about an hour ago · Comment · Like
2 people like this.
Write a comment...

**Matt Thomas** Thinking about getting some psychotherapy.
Posted about an hour ago · Comment · Like

**Matt Thomas** Someone damaged the security gate, had to go long

TODAY · See More
Martin Hewitt's birthday – Send a gift
Silvan Schreuder's birthday – Send a gift
Jemma Butler's birthday – Send a gift

HIGHLIGHTS
Advertise on Facebook
Facebook Ads
Reach over 175 million active users on Facebook. Learn how to connect your business to real customers through Facebook Ads.
Sponsored

Sam's Taste Test ep.3
James Sharpe commented on this.
4

Simbob turns 30
2 friends are tagged.

leam, jan/feb 09
2 friends are tagged.

Movies
3 friends use this application.

Save ITV Yorkshire
2 friends joined.
Join this Group

Leavin' Drinks
by Emma Lobb

National book day at school n sam's hair doo ...
Adrian Bassett is tagged.

POKES
Annakaisa Wallenius – poke back | remove

PEOPLE YOU MAY KNOW · See All
Paul Inman
Add as Friend
Stewart Leahy
Add as Friend

## Four phases of ML adoption

### Phase 2: Simplest ML models

Start with a simple model that allows visibility into its working to:

- validate hypothesis
- validate pipeline

## Four phases of ML adoption

### Phase 3: Optimizing simple models

- Different objective functions
- Feature engineering
- More data
- Ensembling

## Four phases of ML adoption

### Phase 4: Complex ML models