O'REILLY®

Data Engineering Fundamentals

Week 1

What is Data?

Reasoning the Nature of Data

What is Data?

It may seem odd to define "data," something we all use and take for granted.

Chances are if you asked any person what data is, they might answer to the effect of "you know…data! It's…you know…information!" and not venture farther than that.

Now it seems to be marketed as the be-all and end-all. The source of not just truth…but intelligence! It's the fuel for artificial intelligence and it is believed that the more data you have, the more truth you have.

But is this the case? Why or why not?

What is Data?

Before we continue, it might be good to think philosophically what data is and what it represents.

Imagine you were provided a photo of a family:

- Do you know this family's story based on just one photo?
- What if you had 20 photos? 200 photos? 2000 photos? How many photos do you need to know their story?
- Do you need photos of them in different situations? Home, work, school, and vacation? Alone and together? With relatives and friends?

How much can we know about this family just through photos? And how many/what photos do we need?

Data is Snapshots!

Data is just like photographs; it provides snapshots of a story at discrete points in time.

The continuous reality and contexts of the story are not captured, as well as the infinite number of variables in that story.

Today there is a great deal of emphasis on data collection, promoting the idea it can intelligently tell a story, even predict what happens next, and thus create artificial intelligence.

Is this realistic? Why or why not?

Data is Snapshots!

But how realistic is this objective? How narrow must our scope be to make it feasible?

A few strategic photos of the father playing golf can easily tell us whether he is good at golf.

- A photo of him captured mid-swing
- A photo of him cheerful or lamenting at the 18th hole
- A photo of his scorecard!

But trying to decipher his entire life story through photos… that is a much harder problem.

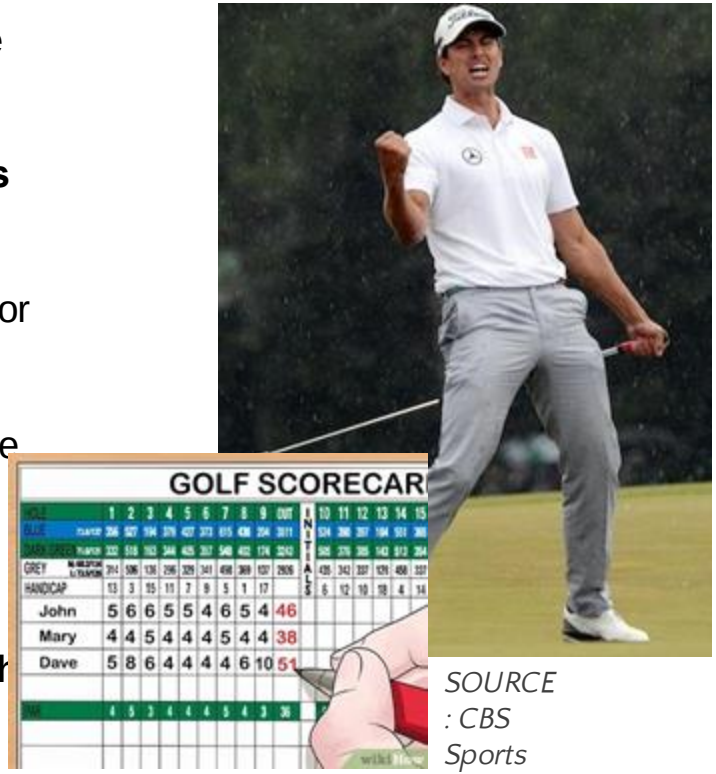
Data is Snapshots!

Even with a narrow objective, it can still be hard to determine what is ground truth, or what is actually true given the data.

Let's say we are trying to evaluate whether the father is good at golf based off those few chosen photos.

- If we catch him fist-pumping at the hole, is he cheering for himself or someone else?
- If we take a picture of the scorecard, how can we be sure it was not forged?

This just goes to show that data can be taken out of context or even forged, and it becomes all the more important to realize that data provides clues to the truth and is not actually the source of truth.



SOURCE : CBS Sports

Bird Strike Data from the FAA

INCIDENT_DATE	TIME	TIME_OF_DAY	AIRPORT_ID	AIRPORT	STATE	RUNWAY	OPERATOR	AIRCRAFT	SPECIES
8/14/2019	18:46	Dusk	KGRB	(KGRB) AUSTIN STRAUBEL INTL	WI	6	(1ASQ) ATLANTIC SOUTHEAST	EMB-145	Ring-billed gull (Larus delawa
6/9/2019	12:23	Day	CYVR	(CYVR) VANCOUVER INTL	BC	26L	(AAL) AMERICAN AIRLINES	B-737-800	Golden eagle (Aquila chrysae
6/17/2019	12:30	Day	CYVR	(CYVR) VANCOUVER INTL	BC	8L	(AAL) AMERICAN AIRLINES	B-737-800	Barn swallow (Hirundo rustic
5/8/2019	11:55	Day	ZZZZ	(ZZZZ) UNKNOWN			(1ERH) ERA HELICOPTERS	SIKORSKY S-92	Unknown bird - small
5/15/2019	09:22	Day	ZZZZ	(ZZZZ) UNKNOWN			(1ERH) ERA HELICOPTERS	AGUSTA AW 139	Unknown bird - small
6/22/2019	6:19	Dawn	PHNL	(PHNL) HONOLULU INTL ARPT	HI	4R	(AAH) ALOHA AIR CARGO	B-737-300	Scaly-breasted munia (Lonch
5/28/2019	16:20	Day	CYYZ	(CYYZ) TORONTO/LESTER B. PEARSON INTL	ON	6L	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - medium
7/22/2019		Night	KABQ	(KABQ) ALBUQUERQUE INTL SUNPORT	NM		(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - medium (Ave
6/21/2019	12:28	Day	EGLL	(EGLL) HEATHROW - LONDON	FN	27r	(AAL) AMERICAN AIRLINES	B-777-300	Gulls (Larinae)
7/26/2019	7:50	Day	EHAM	(EHAM) AMSTERDAM SCHIPHOL	FN	6	(AAL) AMERICAN AIRLINES	B-777-200	Unknown bird - small (Aves)
7/31/2019	11:10		EHAM	(EHAM) AMSTERDAM SCHIPHOL	FN	24	(AAL) AMERICAN AIRLINES	B-777-200	Rock pigeon (Columba livia)
9/5/2019	23:39	Night	KALB	(KALB) ALBANY INTL	NY	28	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - small (Aves)
9/12/2019	11:59		KALB	(KALB) ALBANY INTL	NY		(AAL) AMERICAN AIRLINES	A-319	Unknown Bird (Aves)
1/11/2019	06:25	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	8L	(AAL) AMERICAN AIRLINES	B-737-800	Mourning dove
4/27/2019	22:55	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	26R	(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - medium
4/29/2019	22:45	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA		(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird
6/29/2019	22:9	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	09R	(AAL) AMERICAN AIRLINES	A319	Perching birds (y) (Passerifor
8/11/2019	09:18	Day	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	26R	(AAL) AMERICAN AIRLINES	A-320	Unknown bird - small (Aves)
2/4/2019			KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	17L	(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - medium
3/6/2019	19:30	Night	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	17R	(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - small
3/14/2019	12:57	Night	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	35R	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - small
3/3/2019	08:24	Day	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	35R	(AAL) AMERICAN AIRLINES	B-737-800	Eastern meadowlark
4/24/2019	13:45	Day	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	35R	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - small
4/30/2019	13:35	Day	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	17R	(AAL) AMERICAN AIRLINES	B-737-800	Black-throated green warbler

What is Data Anyway?

That bird strike data is a series of snapshots, capturing a date and time as well as descriptive properties of the mishaps.

- Does the data tell a full story? Or merely clues to a story?
- Is the data comprehensive and complete? Or are there gaps, missing contexts, and unknowns?

Data provides clues and snapshots, not a full context of an event.

Like any clue it can help us find the truth, but it can also lead us to false conclusions and erroneous assumptions.

As a data engineer, it is important to know the quality of data is not exempt from these challenges, and downstream tasks like statistical modeling and machine learning cannot discern bias, correlation, or causation.

Exercise: What Causes Bird Strikes?

INCIDENT_DATE	TIME	TIME_OF_DAY	AIRPORT_ID	AIRPORT	STATE	RUNWAY	OPERATOR	AIRCRAFT	SPECIES
8/14/2019	18:46	Dusk	KGRB	(KGRB) AUSTIN STRAUBEL INTL	WI	6	(1ASQ) ATLANTIC SOUTHEAST	EMB-145	Ring-billed gull (Larus delawa
6/9/2019	12:23	Day	CYVR	(CYVR) VANCOUVER INTL	BC	26L	(AAL) AMERICAN AIRLINES	B-737-800	Golden eagle (Aquila chrysae
6/17/2019	12:30	Day	CYVR	(CYVR) VANCOUVER INTL	BC	8L	(AAL) AMERICAN AIRLINES	B-737-800	Barn swallow (Hirundo rustic
5/8/2019	11:55	Day	ZZZZ	(ZZZZ) UNKNOWN			(1ERH) ERA HELICOPTERS	SIKORSKY S-92	Unknown bird - small
5/15/2019	09:22	Day	ZZZZ	(ZZZZ) UNKNOWN			(1ERH) ERA HELICOPTERS	AGUSTA AW 139	Unknown bird - small
6/22/2019	6:19	Dawn	PHNL	(PHNL) HONOLULU INTL ARPT	HI	4R	(AAH) ALOHA AIR CARGO	B-737-300	Scaly-breasted munia (Lonch
5/28/2019	16:20	Day	CYYZ	(CYYZ) TORONTO/LESTER B. PEARSON INTL	ON	6L	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - medium
7/22/2019		Night	KABQ	(KABQ) ALBUQUERQUE INTL SUNPORT	NM		(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - medium (Ave
6/21/2019	12:28	Day	EGLL	(EGLL) HEATHROW - LONDON	FN	27r	(AAL) AMERICAN AIRLINES	B-777-300	Gulls (Larinae)
7/26/2019	7:50	Day	EHAM	(EHAM) AMSTERDAM SCHIPHOL	FN	6	(AAL) AMERICAN AIRLINES	B-777-200	Unknown bird - small (Aves)
7/31/2019	11:10		EHAM	(EHAM) AMSTERDAM SCHIPHOL	FN	24	(AAL) AMERICAN AIRLINES	B-777-200	Rock pigeon (Columba livia)
9/5/2019	23:39	Night	KALB	(KALB) ALBANY INTL	NY	28	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - small (Aves)
9/12/2019	11:59		KALB	(KALB) ALBANY INTL	NY		(AAL) AMERICAN AIRLINES	A-319	Unknown Bird (Aves)
1/11/2019	06:25	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	8L	(AAL) AMERICAN AIRLINES	B-737-800	Mourning dove
4/27/2019	22:55	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	26R	(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - medium
4/29/2019	22:45	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA		(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird
6/29/2019	22:9	Night	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	09R	(AAL) AMERICAN AIRLINES	A319	Perching birds (y) (Passerifor
8/11/2019	09:18	Day	KATL	(KATL) HARTSFIELD - JACKSON ATLANTA INTL ARPT	GA	26R	(AAL) AMERICAN AIRLINES	A-320	Unknown bird - small (Aves)
2/4/2019			KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	17L	(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - medium
3/6/2019	19:30	Night	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	17R	(AAL) AMERICAN AIRLINES	B-737-800	Unknown bird - small
3/14/2019	12:57	Night	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	35R	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - small
3/3/2019	08:24	Day	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	35R	(AAL) AMERICAN AIRLINES	B-737-800	Eastern meadowlark
4/24/2019	13:45	Day	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	35R	(AAL) AMERICAN AIRLINES	A-319	Unknown bird - small
4/30/2019	13:35	Day	KAUS	(KAUS) AUSTIN-BERGSTROM INTL	TX	17R	(AAL) AMERICAN AIRLINES	B-737-800	Black-throated green warbler

SOURCE:
https://wildlife.faa.gov/

Exercise: What Causes Bird Strikes?

The FAA estimates bird strikes cost U.S. aviation $400 million annually in damages.

You are tasked with processing FAA data to feed into an “AI model” that finds new hazard causal factors in bird strikes.

Before starting the engineering pipelines, you perform due diligence and start getting familiar with the data.

You look at four sensible variables to the right, count the incidents, and grab the top items.

What insights can you glean from these numbers? If we have a machine learning model to find new hazard causal factors and make predictions, what fallacies should we be wary of?

DATA: https://bit.ly/33uAUjI

PHASE OF FLIGHT	INCIDENTS
Approach	32063
Landing Roll	13024
Take-off Run	11743
Climb	10582
En Route	2276
Departure	1466
Descent	941
Local	590
Arrival	487
Taxi	238
Parked	71
Unknown	44

AIRCRAFT	INCIDENTS
UNKNOWN	38077
B-737-700	8748
A-320	6342
B-737-800	6239
CRJ100/200	4774
A-319	4002
EMB-170	3663
EMB-145	3498
B-757-200	2854
A-321	2584
CRJ700	2540
CRJ900	2399
B-737-300	2241
A-300	1765
B-767-300	1574
B-737-900	1548

SPECIES	INCIDENTS
Mourning dove	6327
Barn swallow	4721
Killdeer	4235
Horned lark	4050
American kestrel	3850
European starling	2359
Eastern meadowlark	2135
Red-tailed hawk	1749
Cliff swallow	1661
Rock pigeon	1504
Gulls	1409
Western meadowlark	1232
Sparrows	1176

AIRPORT	INCIDENTS
KDEN	4328
KDFW	3584
KORD	2606
KMEM	2104
KJFK	2100
KDTW	1853
KSLC	1810
KCLT	1713
KATL	1684
KPHL	1567
KLGA	1542
KMCO	1535
KSMF	1500
KEWR	1424
KMCI	1411
KAUS	1360
KSDF	1278
KIAH	1181
KLAX	1180
KBWI	1098
KBOS	1085
KMSY	1047
KCVG	1011
KMSP	1001

SOURCE:

Exercise: What Causes Bird Strikes?

Data modeling practitioners get excited when they see a dataset like this, and there are some sensible conclusions that can be gleaned:

- Most bird strikes happen during takeoff and landing phases, which makes sense given birds fly at lower altitudes.
- Doves, pigeons, gulls, and other birds with large populations are common in incidents.
- Vultures, Canadian geese, and other large birds cause the most damaging incidents.
- Bird strikes spike during the spring season (due to migration).

Note that these conclusions align to our common sense and offer sensible explanations.

- Findings that are believable and readily explained are the best kind, because causality is easier to substantiate.
- Weird and unusual findings (e.g. bird strikes spike around 2:05 PM on Tuesdays when temperature is near 70 degrees) should be dismissed as coincidental.

Bird strike on an F-16 canopy (source: Wikimedia)

Exercise: What Causes Bird Strikes?

While there are insights existing in the data, it is important to realize other variables may not be in the data:

- The type of turf grass used at airfields can attract certain species of birds.
- Stagnant and runoff water also attract birds, and poor runoff management (not rain) is the cause.
- Current weather patterns will also drive bird activity.
- Mitigating anti-bird dogs, devices, and crews can suppress incidents in high-hazard conditions, creating “false negative” data.

Data scientists and machine learning algorithms can easily treat data as inclusive and complete, leading them to identify the wrong causal factors for an event.

The late “Piper” the airport dog, tasked with keeping Cherry Capital Airport free of wildlife in Michigan.

https://abc7chicago.com/piper-airport-dog-Michigan-goggles/1229269/

T. L. DeVault, B. F. Blackwell, and J. L. Belant, editors. 2013. Wildlife in airport environments: preventing animal–aircraft collisions through science-based management. Johns Hopkins University Press, Baltimore, Maryland, USA.

Exercise: What Causes Bird Strikes?

Is it worth creating a data pipeline of bird strike data for the purposes of automatic prediction?

- Machine learning cannot automate analysis, and a human needs to be in the loop to pre-determine correlations and see if they make sense.
- We shouldn't limit ourselves to using just this data; gathering outside information (e.g. weather, bird populations, airport traffic, bird strike heuristics) would provide further context .

While history provides some insights, detailed and specific predictions for the present are better done with real-time monitoring systems.

- The DOD provides an Avian Hazard Advisory System using "bird habitat, migration, and breeding characteristics, combined with key environmental, and man-made geographic data." (http://www.usahas.com/)
- The Royal Netherlands Air Force uses an aviation radar system that allegedly decreased bird strikes around military bases by 50% (http://birdstrikealliance.com/robin-radar/).

The DOD's Avian Hazard Advisory System provides bird strike forecasts for flight planning (http://www.usahas.com/)

Why Does this Matter to Data Engineering?

As a data engineer, you are going to be the gatekeeper and have a say in how data is acquired and speak to its quality.

It is far more important to ask not just what the data says, but where it comes from.

What can bias it? Corrupt it?

Be analysis-driven, not data-driven!

Be analysis-driven, not data-driven.

Don’t just ask what the data says, but where it comes from.

What is Data To the Customer?

It is import to ask these questions about what data is and where it comes from, because your customer might not be stopping to ponder this question.

They might be treating data as a commodity that does not vary in quality, but this is a dangerous assumption.

Bad sensors, bad surveys, gaps in collection, and other problems can easily corrupt and bias data.

Data is like snapshots, capturing only what you point the camera at. What does the customer want captured?

What is Data To the Customer?

This means you need to clearly understand what the customer is using data for and watch out for blindsides they can easily fall victim to.

Does this mean you have to understand machine learning? Or statistical modeling? Yes and no, and you need to understand how data can be biased and have gaps.

While you can always blame the requirements for being wrong and push that back on the customer, a good data engineer will take ownership of the data pipeline including scrutinizing its requirements.

This prevents heartburn and wasted efforts later, and maybe you will position yourself as a knowledgeable VIP!

It will also prevent worst case, nasty political scenarios where the data engineer gets blamed for the quality of the data.

SOURCE: XKCD
https://xkcd.com/1838/

What is Data Labeling?

You might have heard of the process of labeling data, which is exactly what it sounds like: taking each datum and putting a label on it.

This is to aid tasks like machine learning and statistical prediction, to know what the desired outcome is for each data point so it can make predictions.

Take reCAPTCHAS as shown to the right: you are labeling tiles that contain a motorcycle to help machine learning efforts.

Sometimes data already contains a desired label, but oftentimes it does not.

Google's reCAPTCHA, used to verify a user is human, is used to label machine learning datasets for machine learning research.

https://www.google.com/recaptcha/intro/v3.htm

Where Does Data Come From?

Data generally comes from four different sources:

1) **Sensors** – think weather radar, cameras, satellites, movement trackers, connected factory equipment, etc.

2) **Software activity** – Logs from visiting a website, online user activity, app downloads, software usage statistics

3) **Manual Human Effort** – Surveys, click-farms, data entry, handwritten/typed reports, paperwork, etc.

4) **Simulation** –Data generated through randomized, computer-based scenarios and their outcomes.

Regardless of the source of data, there will always be the same types of problems of data corruption and bias.

New York Times wrote about the massive data entry labor required to make "AI" work.

https://www.nytimes.com/2019/08/16/technology/ai-humans.html

Why Do Data Projects Fail?

Common Mistakes Learned from Data Science

Why Do Data Projects Fail?

It is widely documented that many data projects fail[1], and many follow one or more themes:

1) Lack of focus
2) Lack of leadership and organization alignment
3) Inaccessible/bad/no data
4) Underestimating difficulty of the project
5) Lack of necessary skills


Harvard Business Review

Sign In

ANALYTICS

Why You're Not Getting Value from Your Data Science

by Kalyan Veeramachaneni

December 07, 2016

Businesses today are constantly generating enormous amounts of data, but that doesn't always translate to actionable information. Over the past several years, my research group at MIT and I have sought answers to a fundamental

[1] https://www.techrepublic.com/article/85-of-big-data-projects-fail-but-your-developers-can-help-yours-succeed/

1. Lack of Focus

Here's a common narrative: an executive at a company perceives "data science" as a competitive edge and makes it a priority.

Management creates a data science team but there's no clear objective.

When all you have is a hammer, everything starts to look like a nail.

- This team is tasked with looking for problems to solve, rather than solve known problems.
- Data science teams are notorious for having a solution (e.g. machine learning) before they have an objective.

Once a problem is found, stakeholder buy-in and aligning resources is found to be difficult, and focus starts to bounce from one low-hanging fruit to another.

2. Lack of Leadership and Aligned Resources

Leadership is needed to align resources and get buy-in from stakeholders.

- Have a clearly defined objective and roadmap
- Obtain budget to collect data and support the infrastructure
- Attain data access and negotiate data ownership
- Include stakeholder buy-in and domain knowledge
- Budget time and meetings from stakeholders

If higher leadership does not align resources and buy-in from all necessary parties, the data science project will not succeed.

There is no shortage of articles blaming data science and AI project failures on the fact companies are "not ready."

https://venturebeat.com/2015/07/22/stop-hiring-data-scientists-until-youre-ready-for-data-science/

3. Inaccessible Data

It is no secret organizations are protective of their data, but it's not just due to security or distrust concerns.

Even departments inside the same organization will not share data with each other for this reason: they do not want others doing their job… or doing it incorrectly.

- It may require *their* fulltime expertise to interpret the data, and it requires *their* domain knowledge.
- Data scientists can overestimate their ability to interpret foreign data sets and the domain knowledge needed to use it.

The solution is developing trust and buy-in with each partner, negotiate a knowledge transfer, and if needed giving them a significant role in the project.

3. Bad Data

Many project advocates say “data” is the lifeblood of data projects, but they underestimate the volume of bad data they will encounter.

- Corrupted data, rotten data, outliers, biases, and noisy data can pollute training data and require at least 85% of efforts to clean it.

- Even good data must be engineered to be consumable, and feature engineering must choose the right features to train on.

In production, bad data needs to be automatically flagged so it does not degrade the model performance.

For reinforcement learning models, data entry and corrections need to be immediate and streamlined in a way that is not tedious.

https://xkcd.com/1838/

3. No Data

An open secret with machine learning projects is they require a massive amount of labeled data, and this is often a manual effort.

- If labeled data does not exist, you are going to have a difficult/expensive time creating it.
- Tech companies rely on massive data entry workforces that label images, medical scans, traffic scenarios, social media content, and other data.
- This task is done manually by tens of thousands of low-cost data entry workers.
- Smaller AI startups must allocate a substantial part of their budget for this kind of labor.

Before embarking on a data science project make sure that necessary data will be available, and there is budget to effectively to obtain/create it.

New York Times wrote about the massive data entry labor required to make "AI" work.

https://www.nytimes.com/2019/08/16/technology/ai-humans.html

Google's reCAPTCHA, used to verify a user is human, is used to label machine learning datasets for machine learning research.

https://www.google.com/recaptcha/intro/v3.html

4. Underestimating Project Difficulty

Even world-class teams can underestimate the difficulty of a project.

- Classic example: Fully self-driving cars.
- Six years ago it was understandable to believe collecting enough data would allow coverage of fully self-driving test cases.
- Only from experience did autonomy experts realize this was much harder to achieve than initially thought.

A lot of data science teams will claim a model will work, but they "just need more data."

Effective teams will figure out how to do more with less data, utilizing heuristics and explicit algorithms to compensate for deficient data.

Source: xkcd.com

5. Lack Of Necessary Skills

Finally, many sunk data science projects are due to mismatched skills.

- An airline may hire a data scientist for their deep learning skills, but they really need skills in discrete optimization.
- Unfortunately, the hiring manager never made this distinction.

Somebody who self-professes to be a data scientist is often a wildcard, because data science training programs and skillsets are not standardized.

There are far more inexperienced data science professionals than there are veterans.

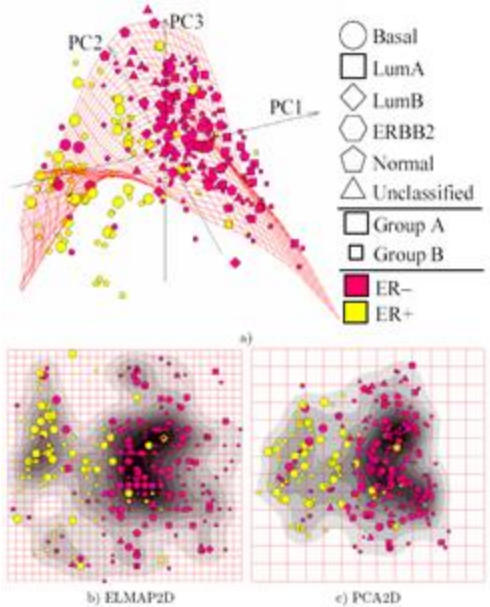
Data engineering can have this problem too.

EXERCISE:
Missing the Forest for the Trees

EXERCISE: Missing the Forest for the Trees

A vendor has approached your military aircraft operation. **They have an AI model that uses data collected from aircraft returning from combat and predicts where lightweight armor needs to be.** It uses detailed data, sometimes manifold dimension reduction, then applies K-means clustering algorithms identifying where bullet holes and damage are likely to be found. It then recommends armoring those hot spots.

What questions do you have for the vendor? Is their data pipeline and methods sound? Why or why not?



Principal component analysis (PCA) and other manifold methods are used to compress multiple dimensions into fewer dimensions, in this case flattening aircraft surfaces so bullet holes can be clustered.

SOLUTION: Missing the Forest for the Trees

This is a modernized version of a real data problem back in WWII.

https://apps.dtic.mil/docs/citations/ADA091073

The Center for Naval Analyses conducted a study on mitigating the loss of bombers. After analyzing fleets of bombers returned from missions, they conclude surfaces that statistically show the most damage should be prioritized for more armor.

But a Hungarian mathematician named Abraham Wald pointed out a fatal flaw with this heuristic.

SOURCE: Wikimedia Commons

SOLUTION: Missing the Forest for the Trees

The flaw: the data only captured survived aircraft, and therefore the heuristic was wrong.

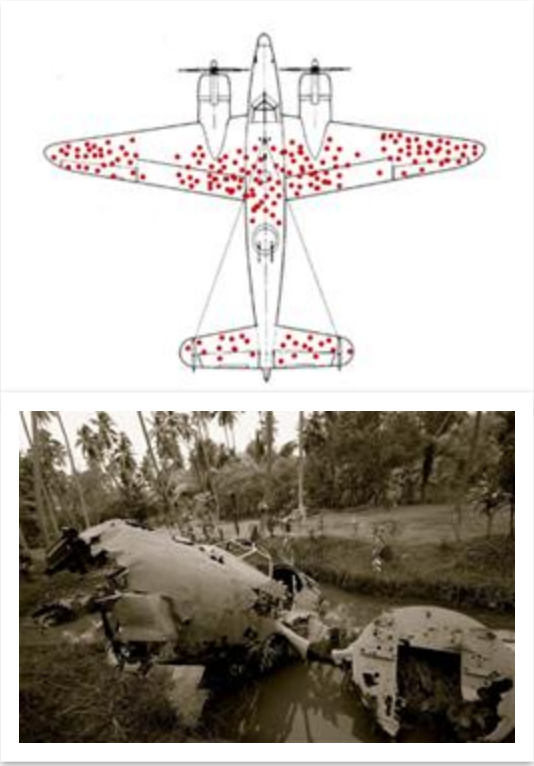
This is an example of **survival bias**, a type of selection bias where we make faulty inferences on the survived population while the deceased population is never accounted for.

While many would cynically say the data is incomplete, the data still provides a valuable clue to solve our objective.

The question we should be asking: why did the aircraft return safely despite the observed damage?

With success, Abraham flipped the theory by armoring the undamaged parts of the aircraft, inferring these were likely the critical areas causing a plane to go down and never returning to base.

This not only saved aircraft and lives but was a pivotal moment for the war effort.



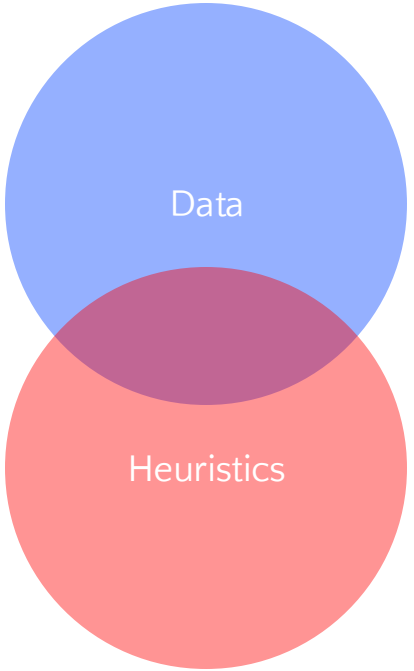
It’s Not Just About Data

Data is often meaningless without context and is therefore not a source of intelligence on its own.

We need to complement the data with our own intelligence and knowledge.

Heuristics play an important role in data acquisition and modeling, and should be evaluated just as closely:

- What is the objective? How narrowly defined is it?
- What factors drive our objective?
- Can we translate our own knowledge into a rule?
- What data do we need to aid our rule-based logic, and why?
- How reliable is the rule? Does it capture the entire domain?
- Can we compensate bad data and models with explicit heuristics?



It’s Not Just About Data

“If you’re in an environment where there is unlimited data available to learn, then you can be incredibly great at it, and there are many, many ways you can be great at it. The smarts about AI comes when you have limited data.”

- Joseph Sirosh, corporate VP for AI and research at Microsoft

“Most real-world strategic interactions involve hidden information. I feel like that’s been neglected by the majority of the AI community.”

- Noam Brown, AI Research Scientist at Facebook

Lightning Round: Identifying Biased Data

Lighting Round: Identifying Biased Data

In WWI, the British introduced the Brodie helmet that was supposed to mitigate head injuries from shrapnel. When the number of shrapnel head injuries increased, British officers thought the helmet was failing.

Should the helmet be abandoned? Why or why not?

No! As discovered later the helmet was working, because more soldiers were being injured instead of killed.

Lighting Round: Identifying Biased Data

In 1984, a psychology professor at Virginia Tech presented a research paper at the American Psychological Association.

He observed three bars near the college campus and found that students who ordered beer by the pitcher consumed twice as much as those who ordered by the glass/bottle.

He concluded that "if we banned pitchers of beer, we would have a significant impact on [reducing] drinking."

Is his theory plausible? Why or why not?

No! Subsequent studies showed that students who intend to get intoxicated will do so no matter what container they use. This is known as self-selection bias and the pitcher is a confounding variable.

Smith, G. (2016). *Standard deviations*. Gerald Duckworth & Co.

Lighting Round: Identifying Biased Data

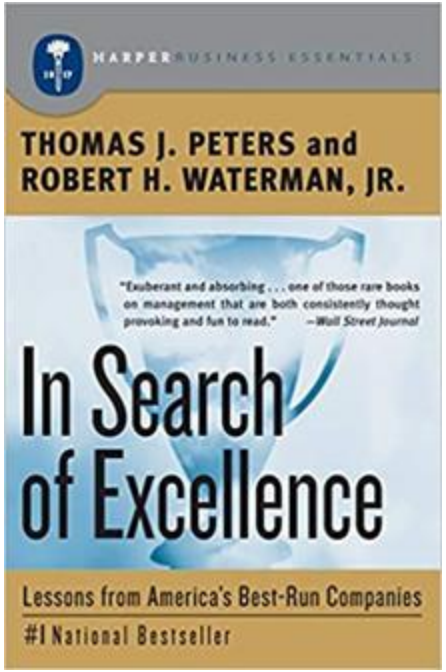
In 1982, two McKinsey & Company consultants wrote the book *In Search of Excellence* covering what they believed to be traits of 43 successful companies: bias for action, closeness to the customer, autonomy and entrepreneurship… just to name a few.

While controversial at McKinsey, the book was successful and popularly considered a classic. Should the book be treated as a blueprint of success?

No! This is survival bias, not accounting for businesses that failed in obscurity who might have had the same traits but never made it to the limelight.

One could argue the book was filled with falsifiable truisms as well: who would openly be against *bias for action*, *closeness to customer*, etc?

Any book that talks about successful traits of companies/individuals should be treated dubiously simply because of survivor bias.



Survival Bias and Success Stories

Source: xkcd.com

Lighting Round: Identifying Biased Data

An airline sent polls to recent passengers on their flights, and then advertised 90 percent of their passengers preferred them over other airlines.

Should this be a serious metric for the quality of the airline? Why or why not?

No! These passengers can be repeat customers, who obviously are satisfied with the airline. We are less likely to survey passengers who flew once and never again thus this is self-selection bias.

Also even first-time fliers have “self-selected” to fly this airline.

Lighting Round: Identifying Biased Data

A phone survey of hundreds of respondents in a school district revealed 80% of respondents support a bill to increase school funding.

Does this sample necessarily reflect the population's sentiment? Why or why not?

This is another example of self-selection bias where some respondents may be unable or choose not to respond. If the calls were made during a workday where phones are not picked up, or if certain respondents are afraid of backlash for their opinion (e.g. "Bradley Effect"), this can skew results.

Lighting Round: Identifying Biased Data

An online forum has a thread posting success stories about a new fitness trend.

41 users have posted photos of their before/after results and the number of pounds they lost, all of which are 15-35 lbs.

Is this compelling evidence the program works? Why or why not?

Not necessarily. There could be survivor/self-selection bias. Users are far more likely to share results if they were successful, while users who attempted the diet and failed are unlikely to report their results.

A control and test group is needed to objectively evaluate the efficacy of the diet.

Lighting Round: Identifying Biased Data

In 1987, veterinary studies showed that cats who fell from 6 stories or less had greater injuries than those that fell more than 6 stories.

Prevailing scientific theories said that cats righted themselves at about 5 stories, giving them enough time to brace for impact and inflict less injury.

Is this plausible? Why or why not?

No! The newspaper *The Straight Dope* posed an important question: what happened to the dead cats? People are not likely to bring a dead cat to the vet and therefore it's unreported how many cats died from higher falls.

What is Data Engineering?

History and Definitions

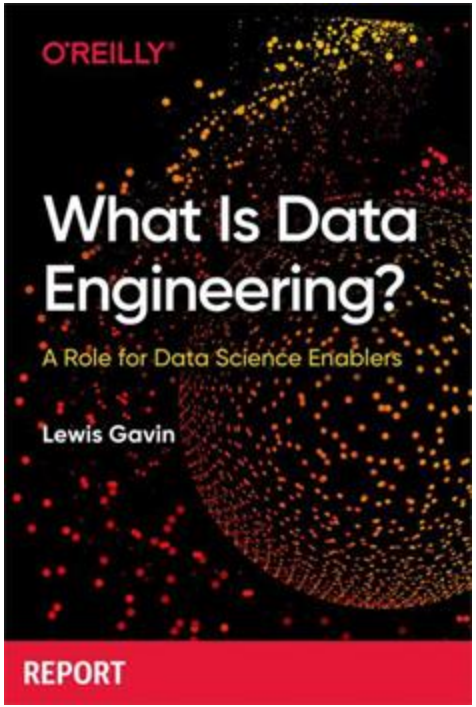
What is Data Engineering?

Just like *data science*, the term *data engineering* has been a source of confusion and unclear definition.

There are many voices on what data engineering is, but Lewis Gavin said it the most succinctly.

Data engineering is all about the movement, manipulation, and management of data.

– Lewis Gavin

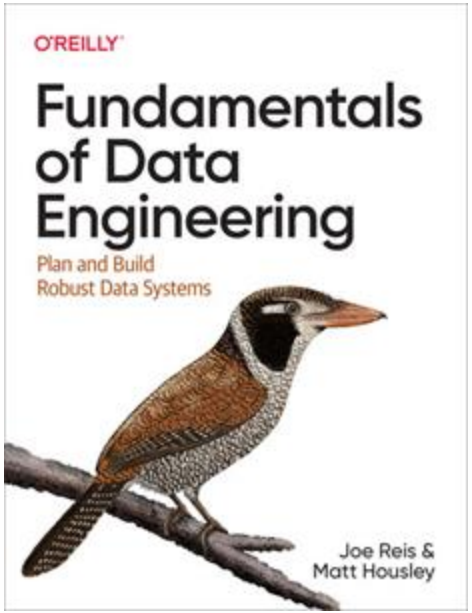


https://learning.oreilly.com/library/view/what-is-data/9781492075578/

What is Data Engineering?

Data engineering is the development, implementation, and maintenance of systems and processes that take in raw data and produce high-quality, consistent information that supports downstream use cases, such as analysis and machine learning. Data engineering is the intersection of security, data management, DataOps, data architecture, orchestration, and software engineering. A data engineer manages the data engineering lifecycle, beginning with getting data from source systems and ending with serving data for use cases, such as analysis or machine learning.

– Joe Reis and Matt Housley



https://learning.oreilly.com/library/view/fundamentals-of-data/9781098108298/

What is Data Engineering?

The first type of data engineering is SQL-focused. The work and primary storage of the data is in relational databases. All of the data processing is done with SQL or a SQL-based language. Sometimes, this data processing is done with an ETL tool. The second type of data engineering is Big Data–focused. The work and primary storage of the data is in Big Data technologies like Hadoop, Cassandra, and Hbase. All of the data processing is done in Big Data frameworks like MapReduce, Spark, and Flink. While SQL is used, the primary processing is done with programming languages like Java, Scala, and Python.

- Jesse Anderson

https://www.jesse-Anderson.com/2018/06/the-two-types-of-data-engineering/

Whew!

As you can see, data engineering sprawls across many disciplines and it is hard to restrict its scope.

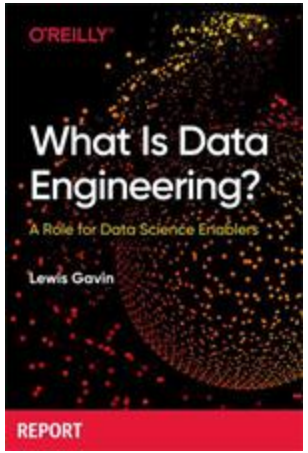
Where does software engineering end and data engineering begin? Data science? Database administration?

We'll do our best to make some distinctions but there will inevitably be overlaps with other disciplines.

Data engineering is more about thoughtful design and lifecycle management, not so much using specific tools.

Data engineering is all about the movement, manipulation, and management of data.

– Lewis Gavin



A Quick History of Data Engineering

1980's: Data warehousing became mainstream in businesses.

With efforts from IBM and Oracle, relational databases and SQL became popular.

Data was often used more to support operations, but business intelligence (BI) became nascent with the availability of data, and the need to get further value out of it.

Scalability became more ambitious, with data warehouse engineers implementing concurrent ETL processes.

A Quick History of Data Engineering

1990's: The internet started to change things rapidly, and the dot-com boom/bust set the stage for the next wave of big tech companies.

2000's: Google, Amazon, and Yahoo were collecting large amounts of data from the internet and connected devices.

Traditional platforms like Oracle could not support such scalability demands, so commodity hardware was introduced along with open-source platforms like MapReduce and Hadoop.

The stage for "big data" was now set.

A Quick History of Data Engineering

2010's: Amazon's Web Services (AWS) became the first popular cloud solution for renting out data storage and computing services.

Hadoop began to spin off a slew of other platforms and tools: Apache Spark, Hadoop Distributed File System, Pig, Cassandra, Hive, HBase, and Storm.

"Big data engineers" became a highly demanded profession and the hype of "big data" fueled an investment frenzy.

It was common businesses to use "big data" platforms to solve small data problems due to marketing bandwagons.

An Amazon data center

A Quick History of Data Engineering

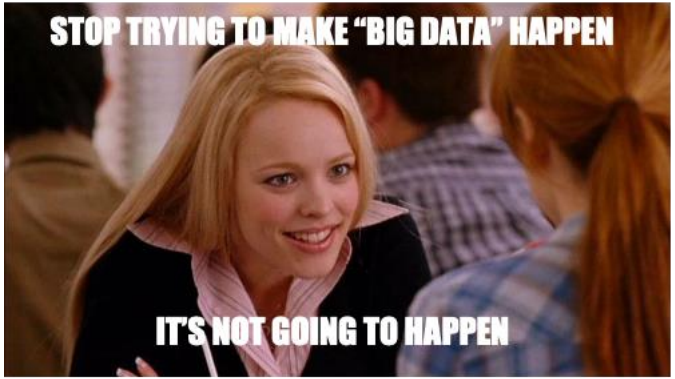
2020's: "Big data is *sooo* 2016!"

Things are simpler now. We don't have to administer all these complicated open-source tools or hire dozens of engineers.

Cloud services from Amazon, Microsoft, and Google will handle security, updates, and configurations for data and computing services.

Because these services can economically serve both "small data" and "big data" needs, we now have "data engineers."

It can also be argued "data engineers" have spun off of "data scientists" roles.



Courtesy: Paramount Pictures

Data Engineering versus Data Science

There is understandable confusion between the differences between *data engineering* and *data science.*

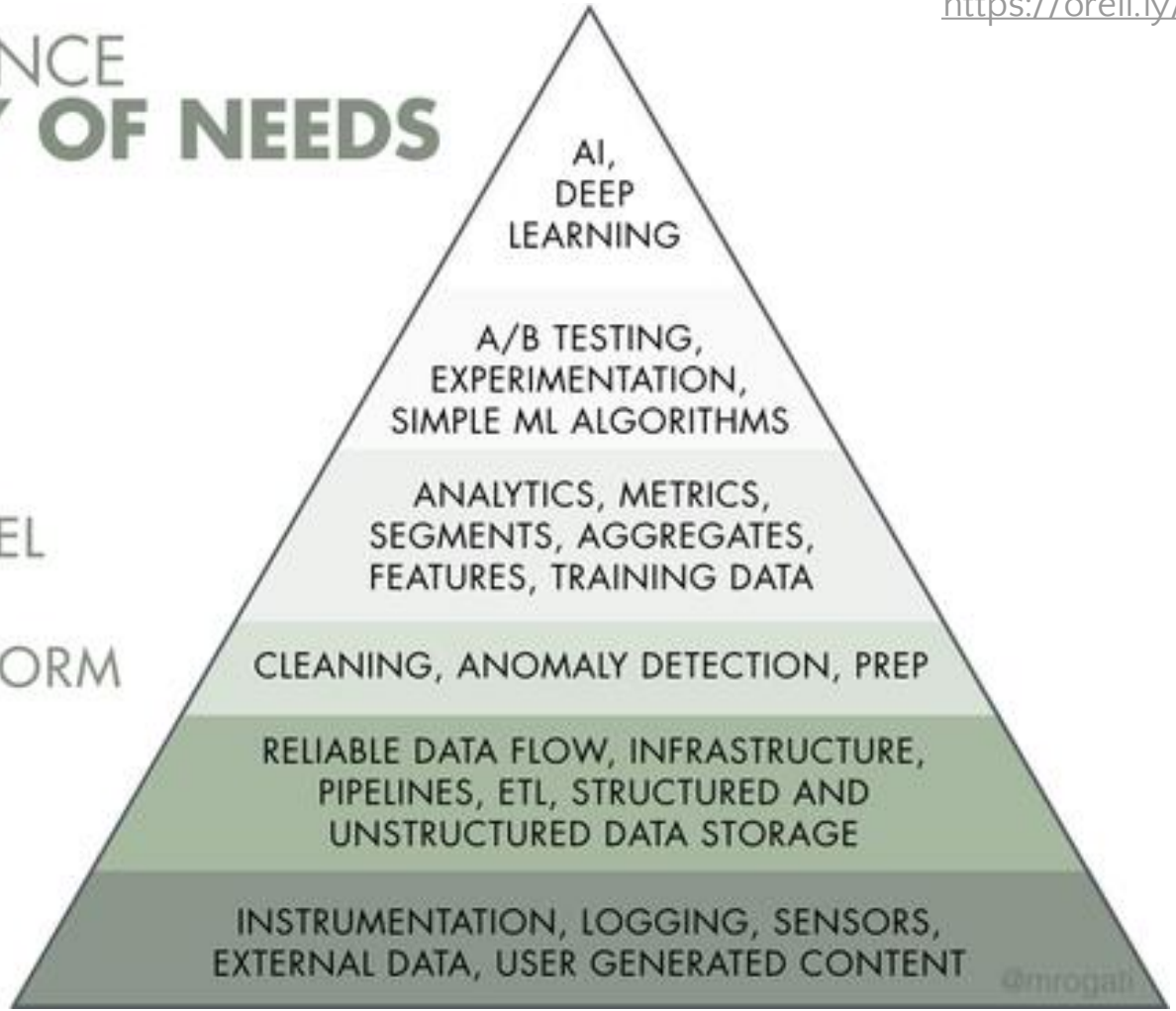
You can think of **data engineering** as upstream to data science, preparing data before it can be analyzed and modeled.

Data science is analyzing and modeling data *after* it has been engineered for consumption.

It is not uncommon to find data scientists doing their own data engineering, which in many situations is not optimal because they spend most of their time gathering, cleaning, and transforming data.

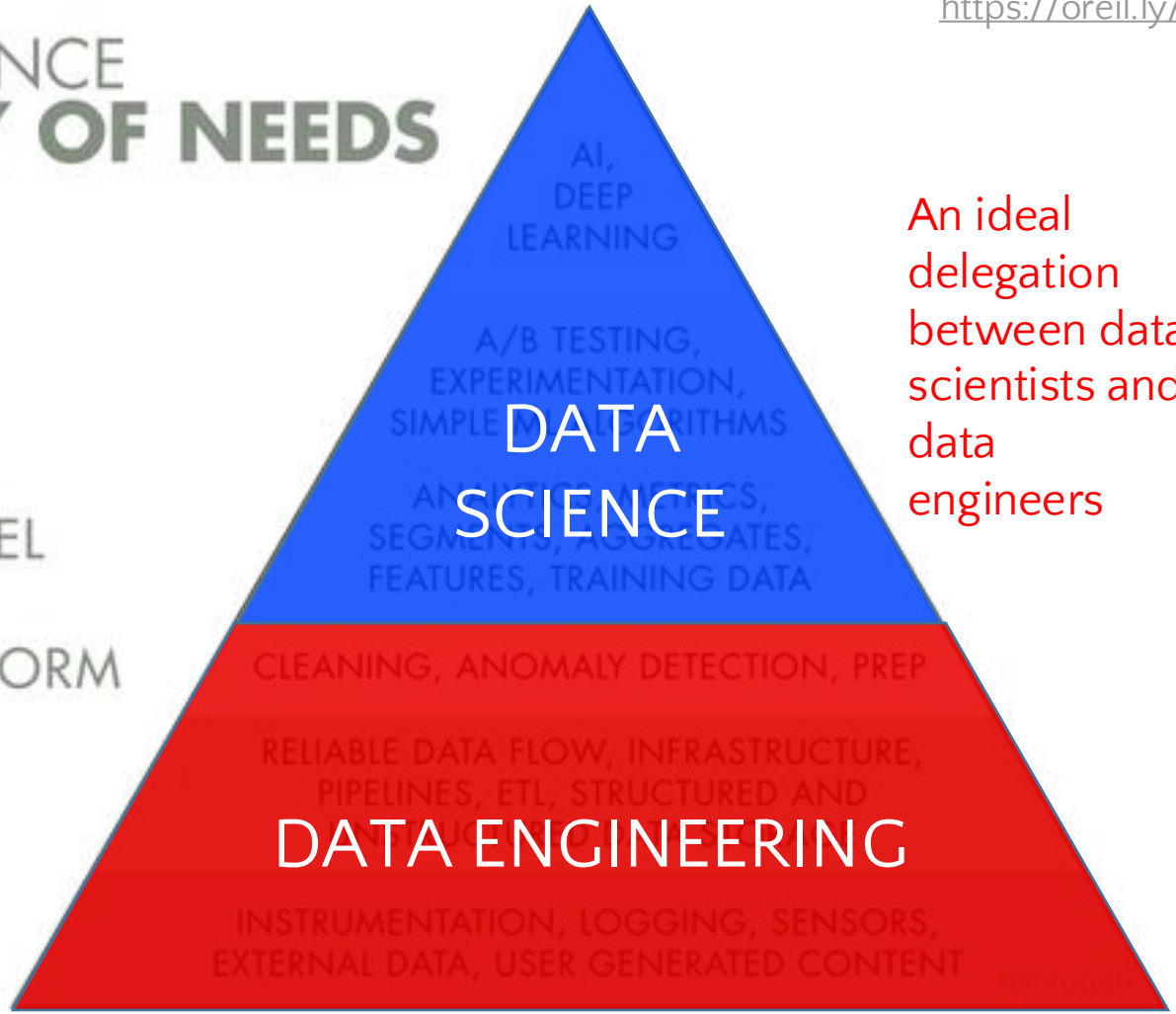
https://oreil.ly/pCg9U

THE DATA SCIENCE HIERARCHY OF NEEDS



https://oreil.ly/pCg9U

THE DATA SCIENCE HIERARCHY OF NEEDS



LEARN/OPTIMIZE

AGGREGATE/LABEL

EXPLORE/TRANSFORM

MOVE/STORE

COLLECT

An ideal delegation between data scientists and data engineers

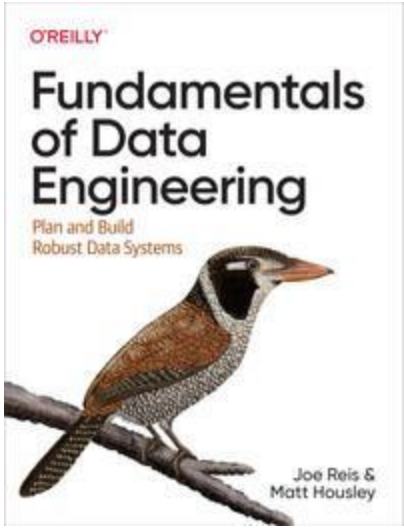
Delegating the Roles

A data engineer should ideally be doing the collecting, storing, and transforming of data… the bottom three layers of the pyramid.

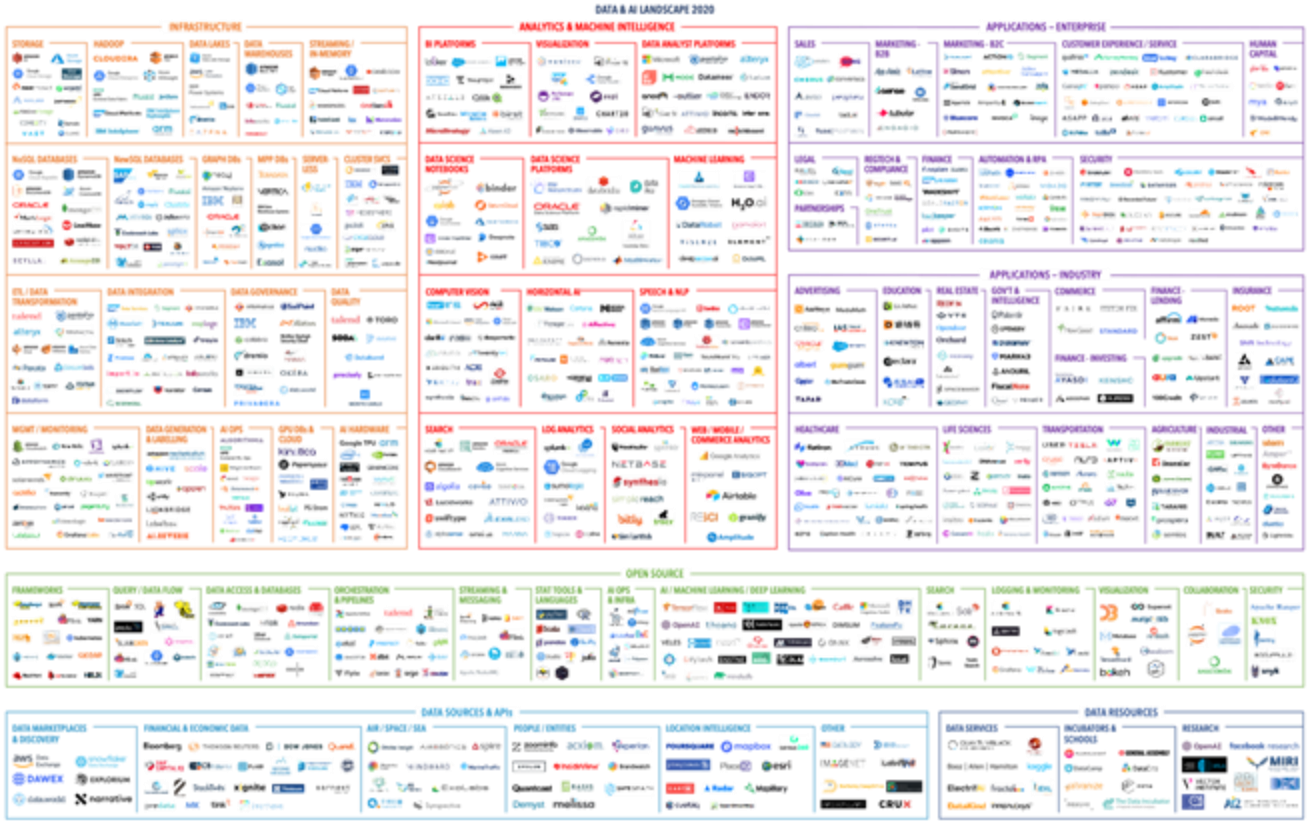
The data scientist will do the machine learning, deep learning, hypothesis testing, exploratory data analysis, and other analytical parts of working with data.

Software engineering can also be a customer to data engineering.

The data engineer should not do the roles of data scientists and software engineers, but it is in their interests to understand their roles and have strong insight into how they work.



Tools… Lots of Tools



https://mattturck.com/data2020/

We're Going to Avoid Tools

This is not a class that will teach how to use Snowflake, Azure, MySQL… or the hundreds of other platforms and tools that are available to data professionals.

Instead, we are going to focus on data engineering concepts and lifecycle management.

This way you will have knowledge that is not prone to obsolescence and can be applied to the currently accepted platforms.

Beware of the Silver Bullet Syndrome

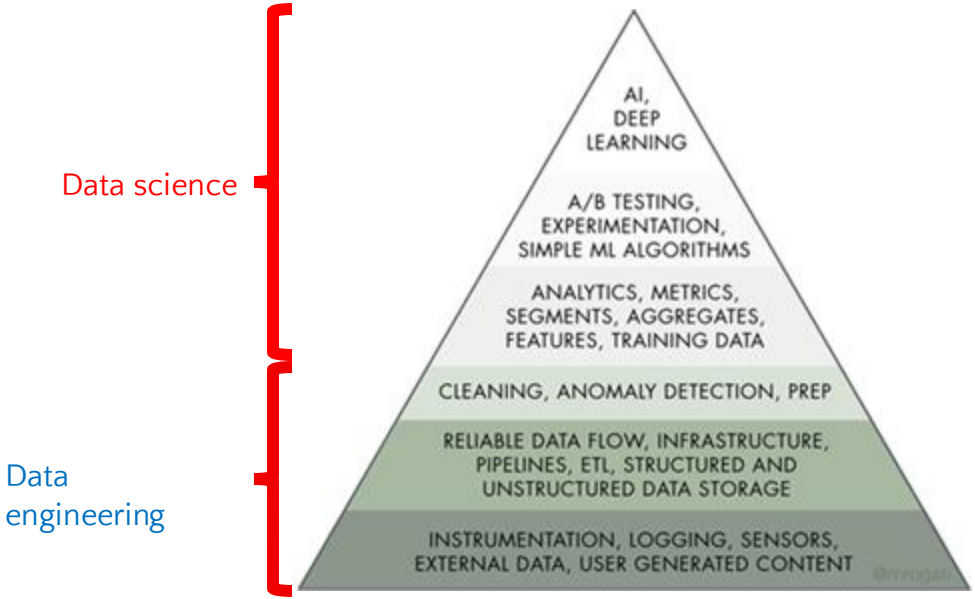
https://youtu.be/3wyd6J3yjcs

https://youtu.be/WN3CSOai_ZU

Data Engineering versus Data Science

Data scientists can easily spend 70-80% of their time on the bottom 3 layers, and they're often lacking the skills to do these tasks effectively.

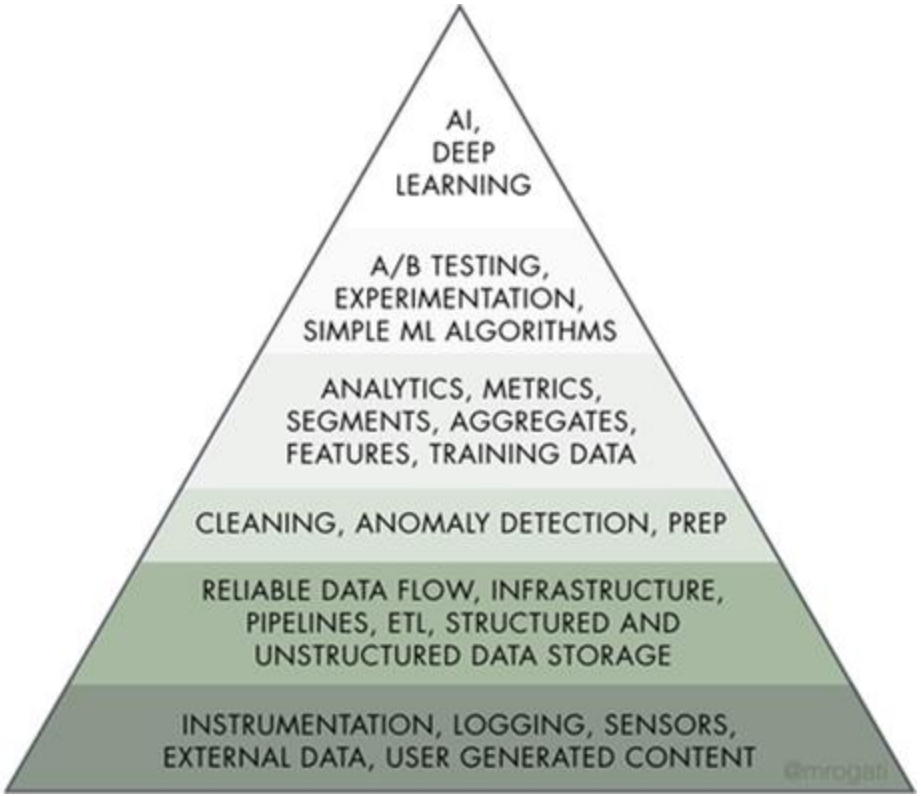
This is why we should delegate this task to data engineers and achieve these 3 layers before pursuing AI and ML.



The Data Science Hierarchy of Needs

Data science

Data engineering



https://medium.com/hackernoon/the-ai-hierarchy-of-needs-

Data Engineering Skills and Background

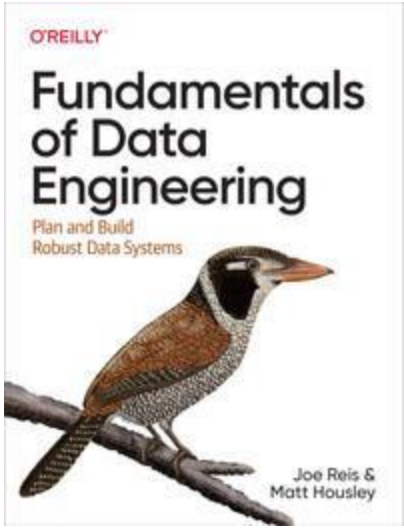
Delegating the Roles

A data engineer should ideally be doing the collecting, storing, and transforming of data… the bottom three layers of the pyramid.

The data scientist will do the machine learning, deep learning, hypothesis testing, exploratory data analysis, and other analytical parts of working with data.

Software engineering can also be a customer to data engineering.

The data engineer should not do the roles of data scientists and software engineers, but it is in their interests to understand their roles and have strong insight into how they work.



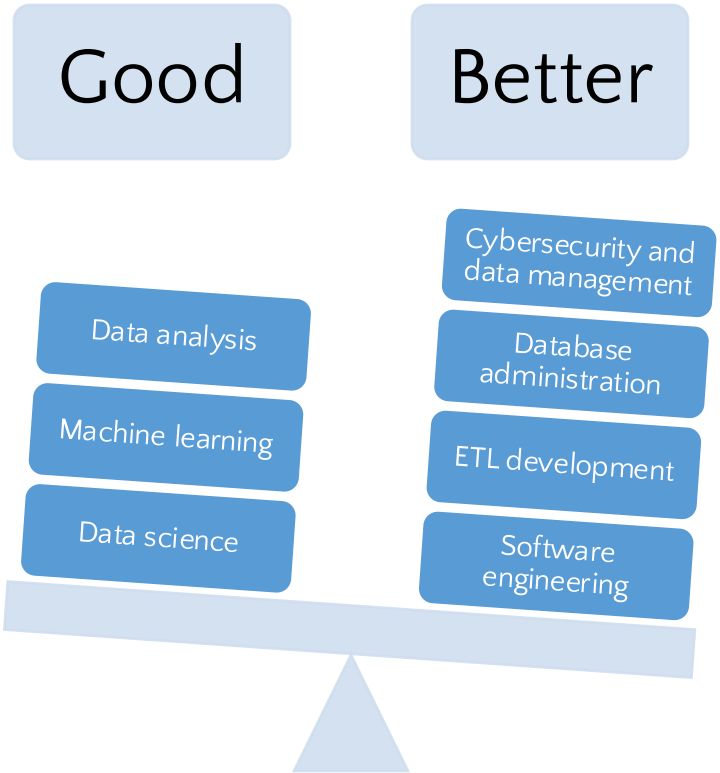
How to Become a Data Engineer

What skills should someone have to be considered a data engineer?

Because the field is new, there is little standardization in training courses, certificates, or university programs.

Like data science, this can create wildcards in people who pursue a data engineering career.

Having machine learning and data science background is helpful (many data engineers are recovering data scientists), it helps to have a systems engineering background that is data-adjacent.



But Always Be Learning…

While we do our best to scope what is and is not a data engineer, you must be curious and constantly learning.

Read, read, and read more everyday.

Be on top of what technologies and platforms are available, as well as their strengths and weakness.

Do not be driven by FOMO (fear of missing out), and always advocate what is practical rather than chasing the latest fad.

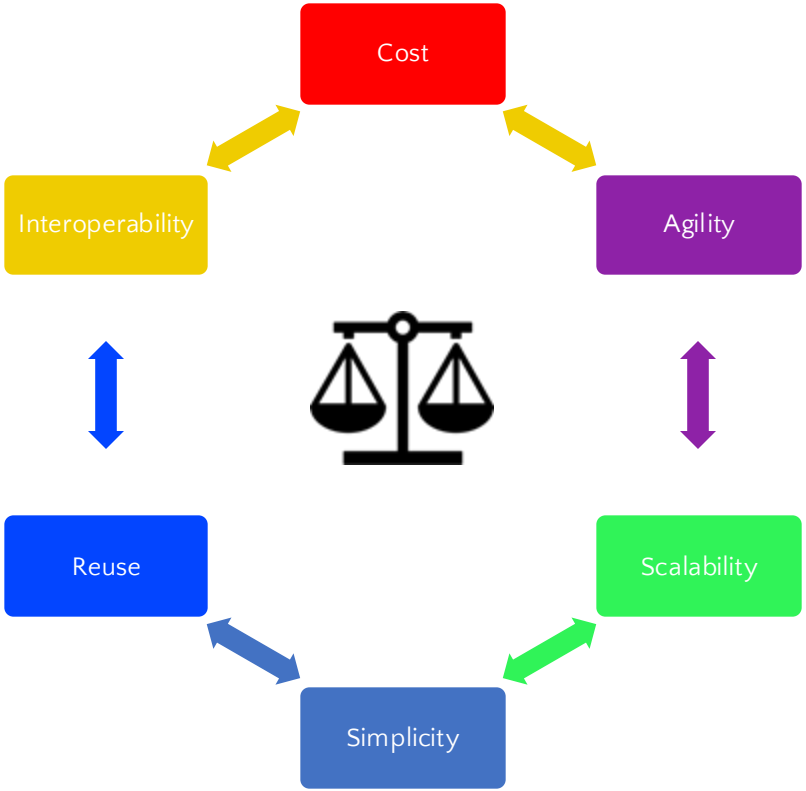
Understand the larger picture of what you are providing, and how your end users and larger organization are using the data.

Balancing these Objectives

A data engineer must optimize the 6 concerns to the right.

Underlying these concerns are activities involving data management, DataOps, security, as well as software and data architecture.

Regardless of the tools used, the data engineer does maintenance, administration, and pipeline management.



Data Engineering ≠ Data Science

Data engineers can benefit from understanding machine, KPI's, data analysis, and end user software engineering.

However, this is not their job to do these data science tasks.

Maintaining context of the end goal is always critical to deliver value, and spending time with customers to understand their roles is never a waste of time.

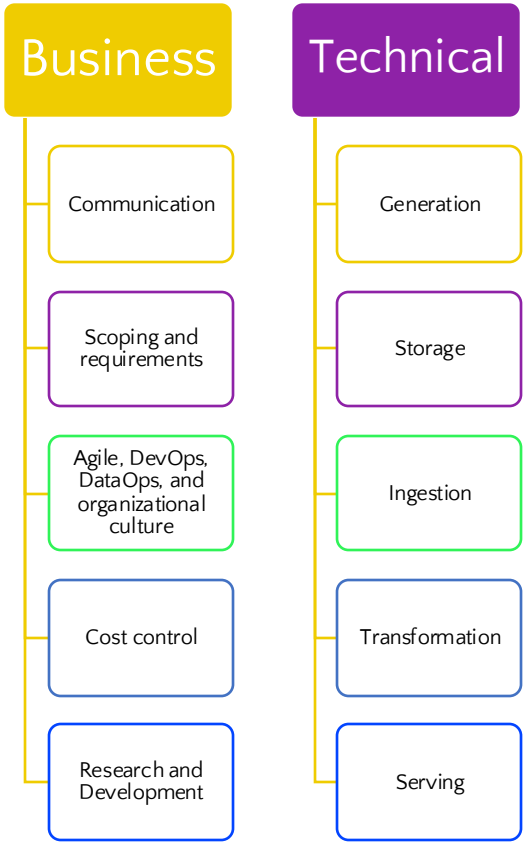
Responsibilities

On the business front, data engineering is responsible for ensuring that value is delivered at a higher return than the cost.

Being able to communicate the value and achievements, as well as understanding closely what objectives the work is aligned to, is critical for success.

On the technical side, rather than focus on the latest and greatest… focus instead on what does not change and remains steady.

Stay abreast of new technologies and how they can deliver value on a paradigm level.



Do I Need to Code?

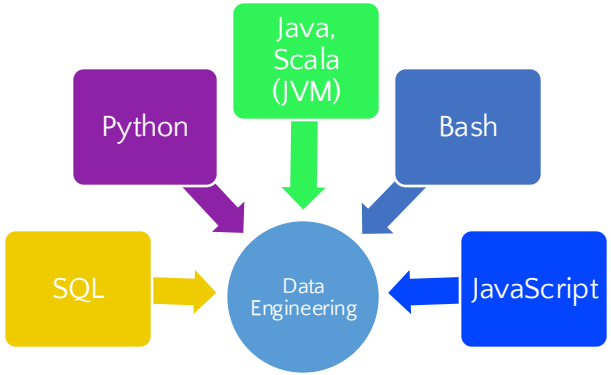
***Yes*, you not only should know how to code but how to do it well enough for production.**

Data scientists knowing a little bit of Python is not going to be enough, and understanding software engineering best practices is going to provide an advantage.

Even with all these platforms and services available, being able to write production-grade code can be a hindrance.

Platforms may have places you can "drop in" code to perform a special function or leverage a service.

Sometimes it is just helpful to "debug" a platform by having low-level understanding of code and software.

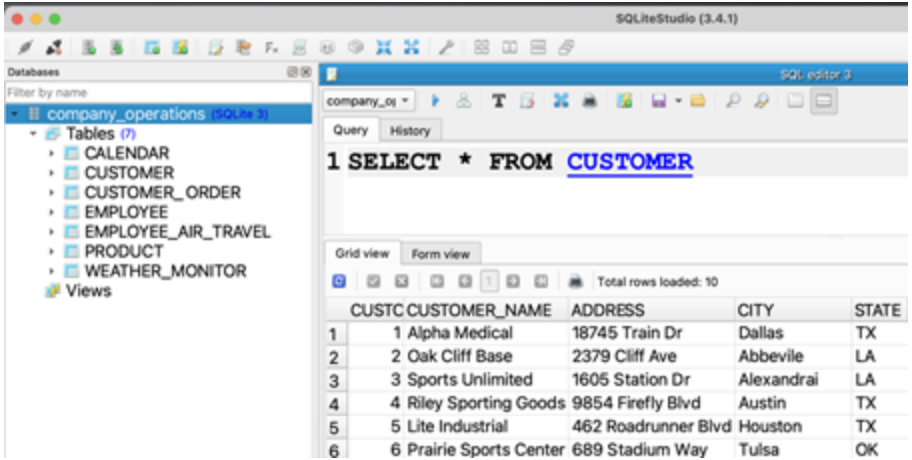


SQL

SQL (”structured query language”) is the most universal tool to querying, writing, and transforming data.

Many open-source and commercial relational database platforms support SQL such as Microsoft SQL Server and MySQL.

SQL and relational databases are so tightly coupled that “SQL” is often used in the branding of the relational database, like “MySQL” and “Microsoft SQL Server.”



Using SQLiteStudio to query a SQLite database

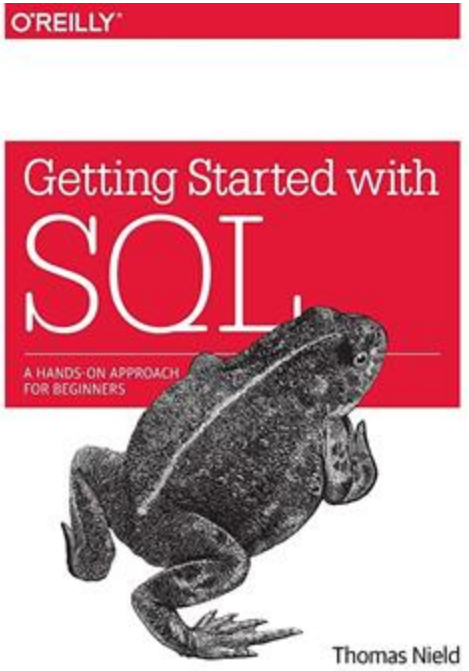
SQL

Simply put, it is hard to get anywhere as a data science professional without proficiency in SQL.

Businesses use data warehouses and SQL is almost always the means to retrieve the data.

SELECT, WHERE, GROUP BY, ORDER BY, CASE, INNER JOIN, and LEFT JOIN should all be familiar SQL keywords.

It is even better to know subqueries, derived tables, common table expressions, and windowing functions to get the most utility out of your data.



NoSQL versus SQL

NoSQL stands for *not only SQL*, and is often used to describe “Big Data” platforms that may leverage SQL but are not relational.

- NoSQL databases include MongoDB, Couchbase, Apache Cassandra, and Redis.
- These platforms store massive amounts of data in a variety of raw and unstructured formats (e.g. *documents, key-value*).
- Most of these solutions are **distributed** across multiple machines, which is difficult to do with relational databases.

Other “Big Data” solutions such as Spark SQL, Google BigQuery, Snowflake, Hive, Kafka, Beam, and Flink can be interacted with using SQL.

Therefore, SQL can be applied to non-relational database platforms and frameworks.

Caution using NoSQL and Big Data: “When all you have is a hammer, everything starts to look like a nail.”

- Do not fall into the trap of treating all data problems as Big Data problems, because most are not.
- **Be aware of the “Silver Bullet Syndrome”:** https://www.youtube.com/watch?v=3wyd6J3yjcs

SQL vs NoSQL

Feature	SQL	NoSQL	Winner
Integrity/Consistency	Data is enforced with logical relationships, minimized redundancy, and "Up-to-date" consistency.	Simple key-value and document storage does not enforce any rules or structure. Redundancy and write latency is common.	SQL
Design changes	Easy to "add" to database, but harder to modify.	NoSQL can quickly and arbitrarily change what data it stores.	NoSQL
Analysis	SQL is a universal language that makes accessing and analyzing data simple.	SQL support is sparse, and proprietary languages are esoteric and hardly universal.	SQL
Programming	Programmers of Java, Python, and .NET have to map entities to tables, which can be tedious. But data integrity is given.	Programming against a NoSQL database is quick and simple, but onus is on programmer to validate data.	*Draw*
Performance	Relational databases can store data for most use cases, but struggle with true "big data" cases. Integrity constraints also slow down performance.	NoSQL is capable of storing vast amounts of data with horizontal scaling. It also performs quickly due to horizontal scaling and no integrity constraints.	NoSQL

SQL vs NoSQL – Summary

SQL = integrity and accuracy

NoSQL = speed and scalability

SQL should be a prerequisite before learning NoSQL and “Big data”.

If you are uncertain which to use, always start with SQL.

Python

Python is the “second-best language at everything” when it comes to scripting and programming.

It is quite versatile at data science and data engineering tasks, and you will even find many commercial platforms support “plugging in” Python scripts.

It can always pay to become proficient in Python as it is the preferred tool in data science and a growingly popular tool in software engineering.

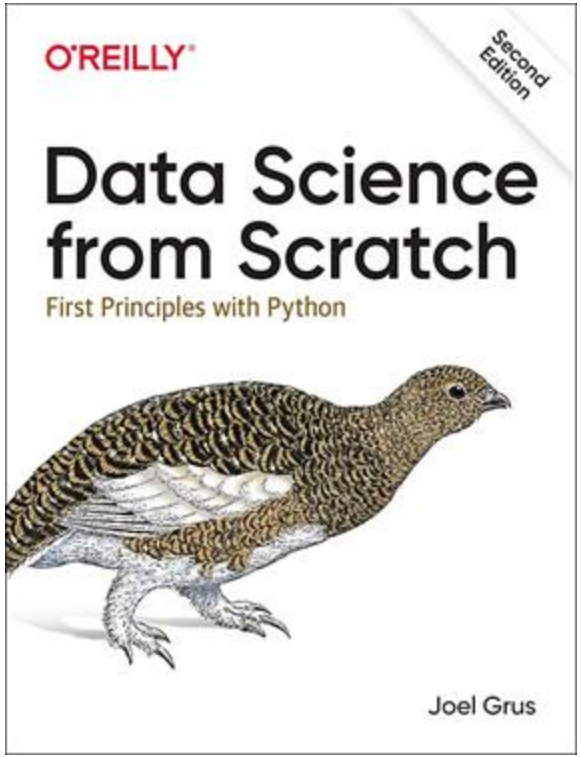
Become proficient in Pandas, NumPy, sci-kit learn, PyTorch, and PySpark

It is important to note that Python is not a fast, computationally performant platform but its libraries written in C/C++ are.

Try to get comfortable using Python outside a Jupyter notebook environment, as notebooks can create bad programming habits.

”I Don’t Like Jupyter Notebooks” – Joel Grus

https://youtu.be/7jiPeIFXb6U



Java (JVM)

You might be surprised that Java and other Java Virtual Machine (JVM) languages could be necessary to learn.

A lot of Apache open source projects like Spark, Hive, and Druid are all built on Java.

There are also Java-based languages like Scala, which also works on the JVM.

Java is faster than Python in computational speed, and Python interfaces are added to platforms like Spark to make Java talk to Python.

Understanding how Java and the Java Virtual Machine (JVM) works, as well as how bytecode is created from Java, Scala, Kotlin, and other JVM languages can set you up to better understand systems built in Java.

Recommendation: use Amazon Corretto distribution to avoid licensing headaches: https://aws.amazon.com/corretto/

Bash

The command line may look intimidating to the uninitiated, but it is enormously rewarding.

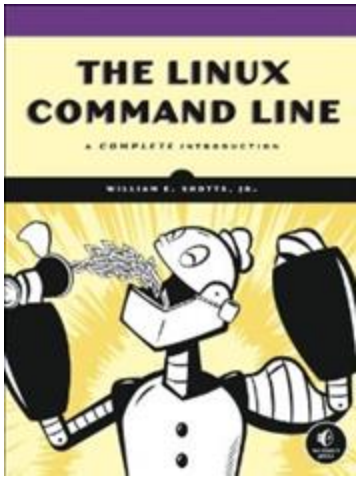
You can quickly run commands and repetitive operations through scripts.

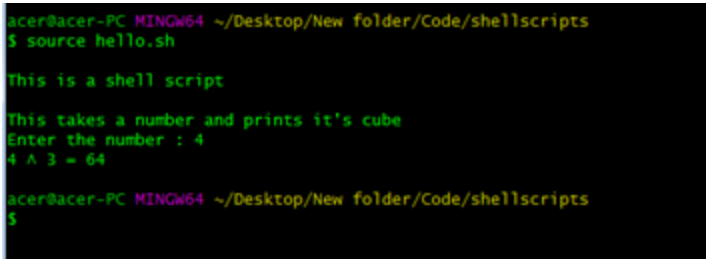
Many platforms (e.g. Microsoft Azure) will be easier to work with and maintain using bash interfaces.

Linux-based operating systems (e.g. Ubuntu) as well as macOS support bash operations in the command-line terminal.

Windows can also enable bash operations using the official Linux Bash Shell.

You can also learn the plain Windows command line or PowerShell as well.





Javascript and Other Languages

The truth is you will have to be flexible and ready to learn any programming language or technology, depending on what your organization already uses.

It is easy to have a preference, but try not to practice language tribalism as this is rarely productive.

You may have a strong preference to use Python with PyQT for user interfaces, but maybe your employer wants to use a web service with JavaScript and HTML.

Just be sure to always match your skills and knowledge with what's required for the job, and use the opportunity to learn and grow!

Always focus on what's constant and solved problems effectively, and not be overly partial to the technology used.

Data Maturity

Understanding Data Maturity

As a data engineer, understand that there is a **data maturity** reflecting a progression of data activity stages for an organization.

This affects what a data engineer works on and how their career progresses, as well as defines data utilization, capabilities, and integration of data in the organization.

Here is a model for data maturity according to Joe Reis and Matt Housley.

https://learning.oreilly.com/library/view/fundamentals-of-data/9781098108298/ch01.html

Stage 1: Starting with Data

At organizations big and small, there can be a starting stage where data objectives are vague and not quite yet tangible.

Machine learning is not practical at this stage, as trying to architect data (much less get insight from it) is not ready.

Data engineers at this stage must be prepared to wear many hats: software engineer, data scientist, systems engineer, report generator, and data architect.

The data engineer should also be winning over stakeholders and helping sell a project, with some level of leadership backing and sponsoring them.

Stage 1: Starting with Data

Don’t be surprised if you must be scrappy at this stage, cutting corners to prove a concept before more resources are deployed.

You might find yourself running processes on your desktop computer without a proper cloud instance, or cutting corners to save on storage and computation costs.

You might be running tasks that are outside of your role, like running reports and wrangling spreadsheets.

Accept this as part of “learning the process” before automating it.

Stage 1: Starting with Data

Some things to watch out for:

Always seek some quick wins even if it incurs some technical debt, so the value of data is proven to the organization.

Get away from your desk and talk to your colleagues and stakeholders to gain context, and not be disconnected from the rest of the organization.

Deliberately track what adds value to organizational goals, or else your job quickly becomes academic; this can be dangerous the moment downsizing needs to occur.

Be wary of workplaces that lack things to do, inventing busywork and constantly changing direction without purpose.

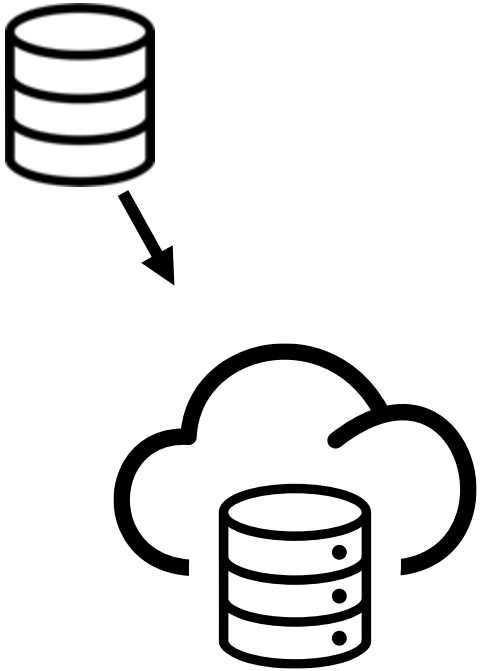
Courtesy: 20th Century Studios

Stage 2: Scaling with Data

You are in a role that successfully found a purpose for its data, congrats!

Now what's next?

This is where you now focus on scalable data architectures and formalizing the data engineering lifecycle, and specialization and role delegation starts to occur.



- Best practices for data management
- Scalable architecture
- DevOps and DataOps
- Machine learning support
- Leveraging turnkey solutions and only making custom solutions as needed

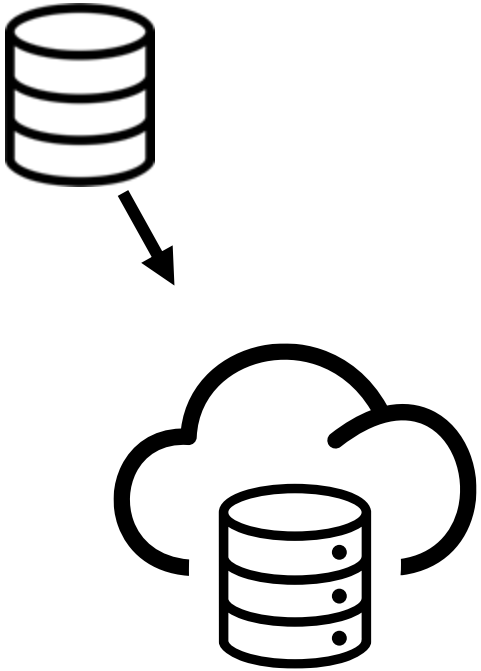
Stage 2: Scaling with Data

Things you want to watch out for include:

Jumping on trends and bandwagons, especially the latest flavors of technology platforms to build your resume. Resist the urge!

Embracing complexity instead of simplicity, and using the maximum budget and resources allowed.

Positioning yourself as a guru on the bleeding edge, rather than focusing on utilitarian solutions that seek maturity and stability.



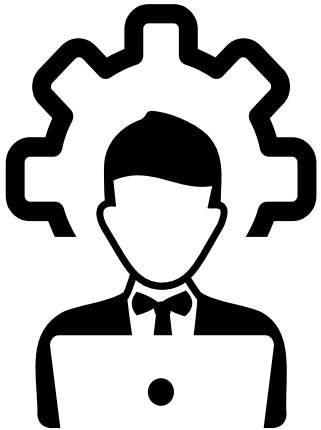
Stage 3: Leading with Data

Data processes and organizations have reached a maturity, and end users at the organization can easily perform self-service.

Data pipelines are abundant, and new ones can be added with ease.

End users are applying analytics and machine learning, and building their own tools around the data.

Data governance and management becomes more enterprise focused, and teams of ML engineers, analysts, and software engineers can easily collaborate and work together.



Stage 3: Leading with Data

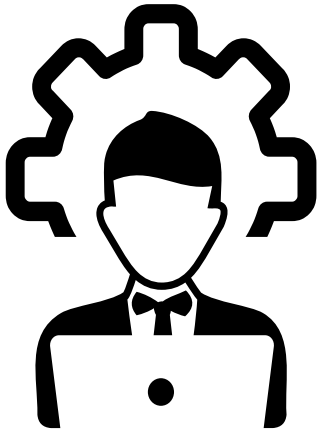
Things you need to watch out for in this stage:

Don’t let security go by the wayside, and as the organization grows consider internal and external threats if the data gets breached.

Conversely, don’t let security get so carried away that bureaucracy becomes dysfunctional and rampant, which can stifle innovation.

Advocate against cost-cutting and complacency, ensuring that necessary talent is retained and maintenance/improvement continues to be a priority.

Advocate against change for the sake of change, from both managers and other engineers, who are eager to build their resume on new projects rather than maintain operations.



Type A versus *Type B* Data Engineers

According to Reis and Housley, there are *Type A* and *Type B* data engineers, the latter which will thrive more in later stages of maturity.

Type A

- "Abstraction" – Effective at abstracting processes and concepts using off-the-shelf tools
- They will be around for all stages of maturity, but are most necessary for early stages.

Type B

- "Build" – focus on scalability and building data tools and systems on a robust production level.
- They are more likely to build custom solutions.

Exercise

Exercise – Data Strategy Team

Your executive director oversees 12 departments in a company of 2000 people, ranging from operations to accounting and IT.

She tasks you to create and lead a "data science" team of 20 people.

This data team will "rove" and aid each of the 12 departments to make them more "data-driven," apply machine learning, and break down data siloes that exist.

Is this objective reasonable? What opportunities/challenges lie ahead?

Solution – Data Strategy Team

Fortune 500 companies have tried this model with a central data science team and have them act as roving consultants.

While the idea sounds good on paper, it can be challenging especially if the company's product is not technology-centric.

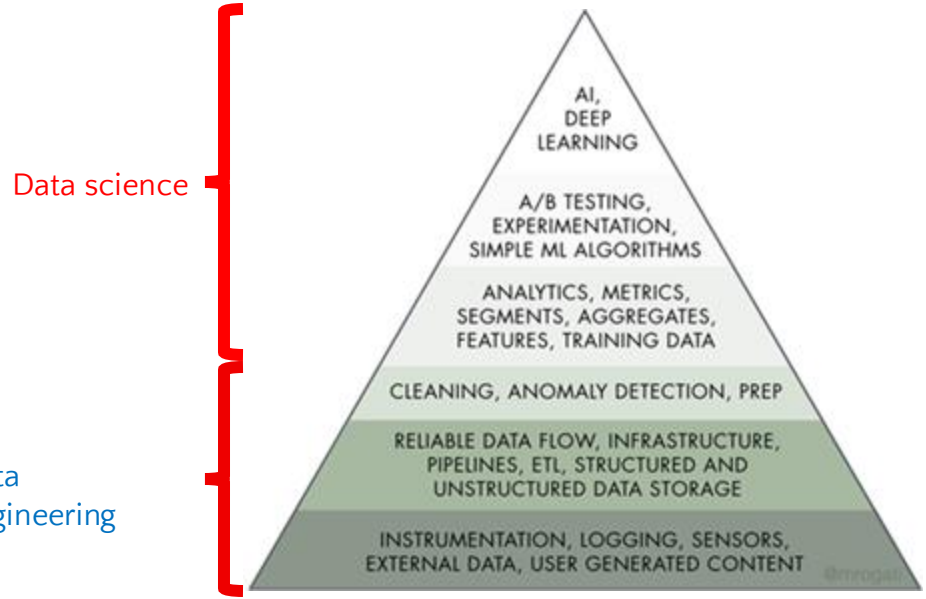
- No clear objectives, it would be better if tangible problems and roadmaps were used to justify a data team.
- Low-hanging fruit becomes the modus operandi of this team.
- Getting time, knowledge, and buy-in from departments is difficult.
- Common sentiment: "Oh yeah that project has been tried before and it didn't work, but how is your attempt different?"

Solution – Data Strategy Team

Ask the executive director what specific problems bother her the most.

Evaluate whether a "data science team" would solve these issues, or if resources would be better spent on a data engineering initiative of some kind.

If the objective is breaking down data silos, put emphasis on data engineering. Get an IT task force to centralize data and work on the bottom 3 parts of the pyramid.

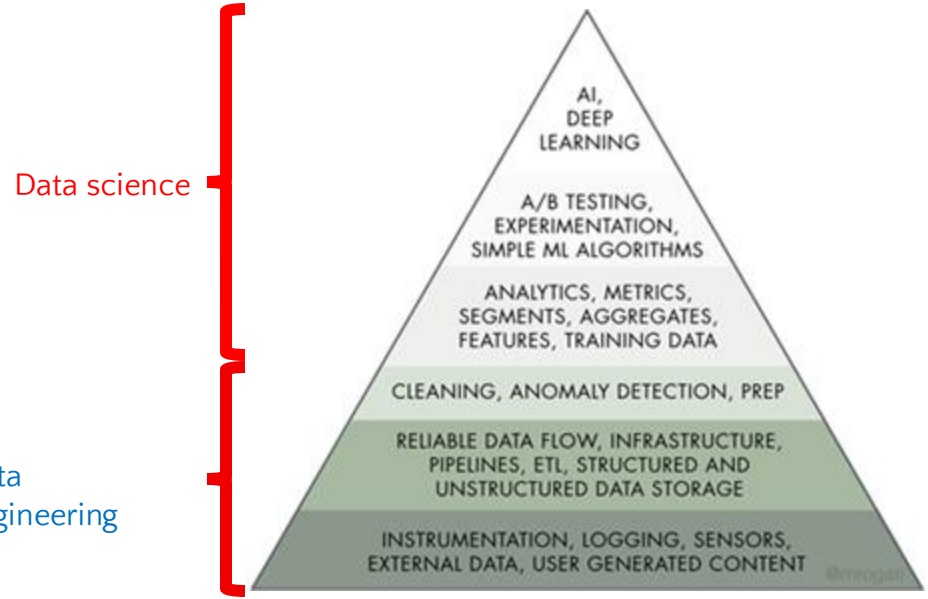


Solution – Data Strategy Team

If the executive director wants to apply machine learning, ask what problems she would like to solve with it.

If an objective is identified and a team is needed, make sure a clear roadmap is put together.

Align departments, leadership, and resources *THEN* go put a team together and get the bottom three layers of the pyramid mastered first.



O'REILLY®

Data Engineering Fundamentals

Week 2

Who Are the Customers?

Internal-Facing versus External-Facing

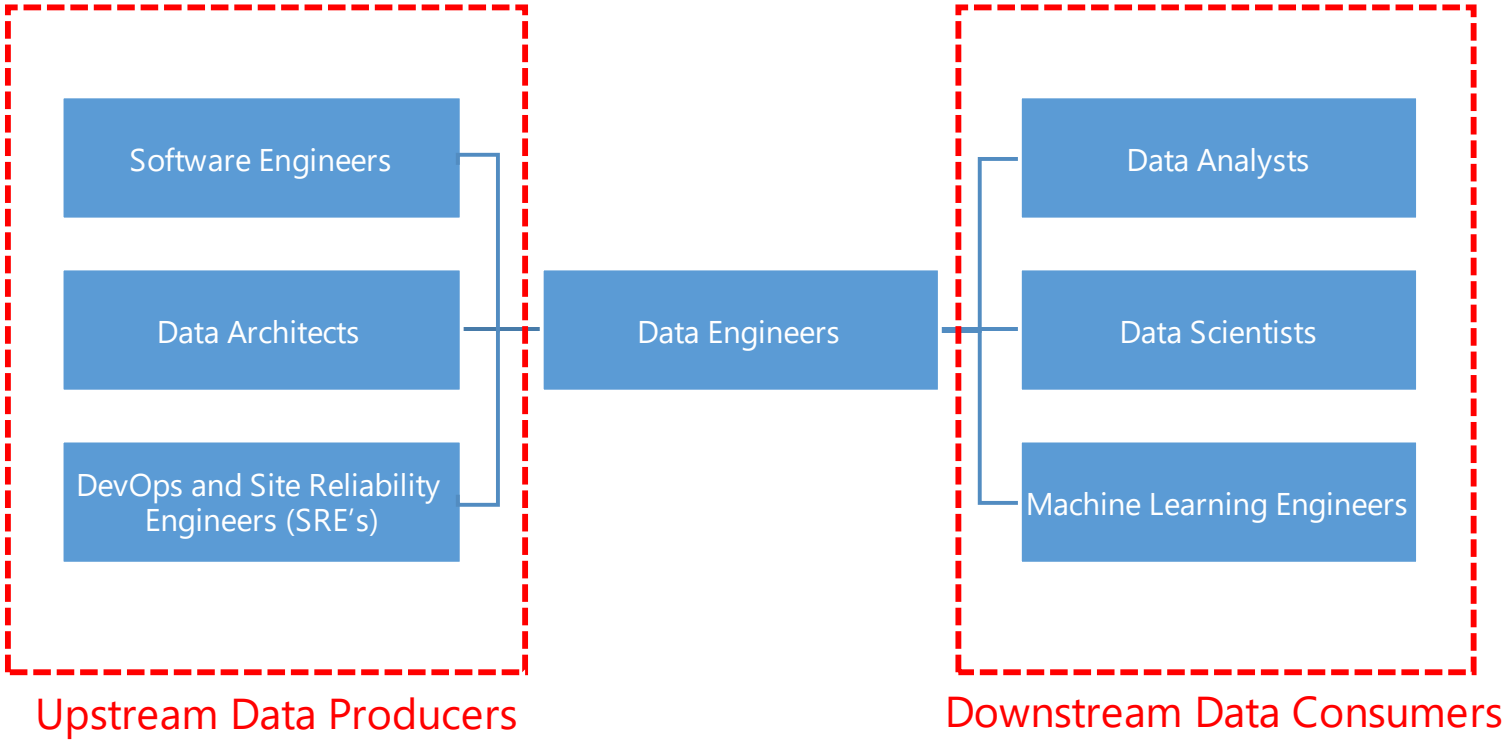
Data engineers can be internal-facing or external-facing, meaning they deal with internal or external users.

External-facing would be cyclically publishing and collecting data from consumers through social media apps, the company's shopping site, or internet-connected devices.

Internal-facing would be dealing with internal needs, like generating reports and machine learning models for internal use.

Typically, a data engineer is going to have a mix of these two roles but the internal-facing must be mastered first before such data/models are used for production.

Data Engineering Upstream/Downstream

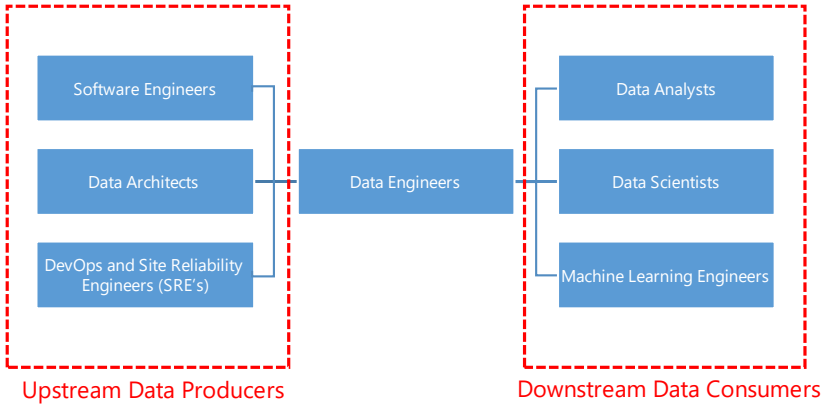


Data Engineering Upstream/Downstream

Data engineering sits between *data producers* and data *consumers*, transforming data from the former to make it usable to the latter.

The producers include software engineers, data architects (including database administrators), DevOps, and site reliability engineers (SRE's).

The consumers include data analysts, data scientists, machine learning engineers, and end user applications.
Let's understand the familiar customers:



https://learning.oreilly.com/library/view/fundamentals-of-data/9781098108298/ch01.html

Data Scientists

Data scientists use models to analyze correlations, make predictions, and automate those discoveries with machine learning.

However, they can spend a great deal of time collecting, cleaning, and preparing data and if they are limited to only a desktop computer they must cut corners in sampling and analyzing the data.

Data scientists may not be equipped to deploy data processes into production either.

This is why data engineering is necessary to serve data scientists effectively, and to better automate data science for the larger organization.

Data Analysts

Data analysts can easily be branded as “data scientists” in some organizations, but more often they associated with traditional business analytics.

Data analysts are more likely to use spreadsheets and SQL rather than machine learning and Pandas, and are more likely to have more business domain knowledge.

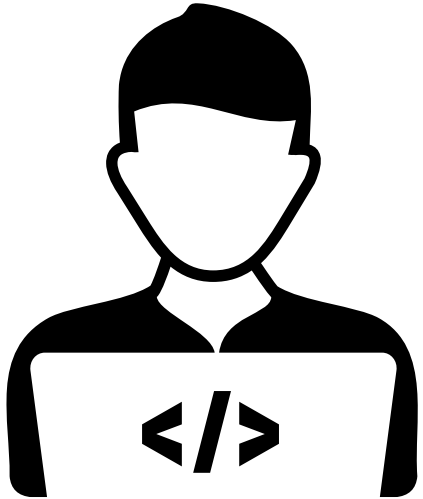
Regardless, like data scientists they still need data served to them that is available and ready, making them a customer of data engineering too.

Software Engineers

Software engineers are often focused on deploying apps and services for end users, and they can overlap some tasks with data engineering.

However, data engineering may provide some helpful support from software engineering and focus on the data-specific tasks as well as deploying for production.

This frees up the software engineer to focus on things like user interface design, web services, security, and integrating the data pipeline into those services.



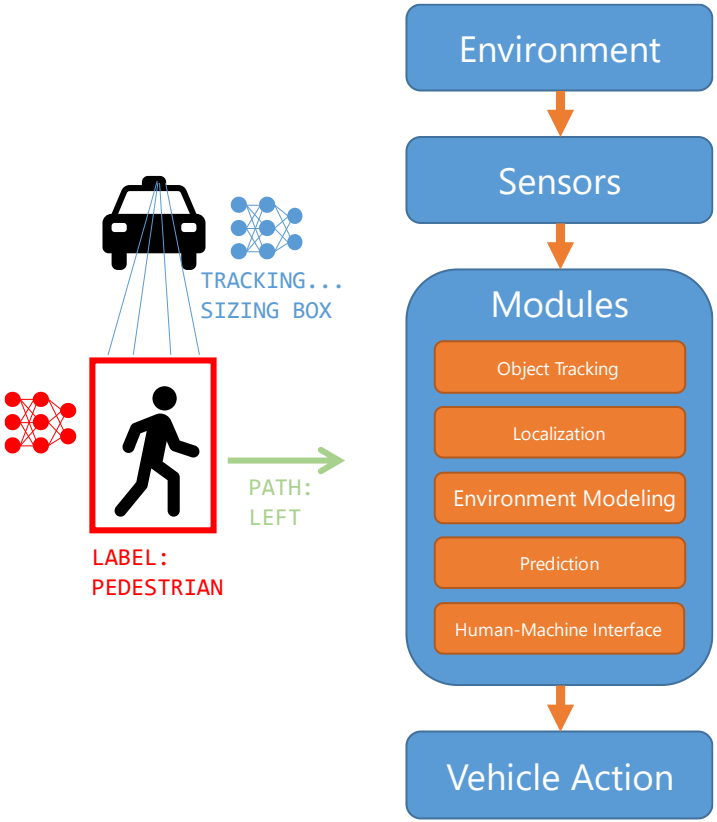
AI/ML Engineers

Machine learning engineers will often have an overlap with data scientists, software engineers, and data engineers.

However, they will likely be focused on training and testing models, and increasingly maintaining them in a production context or in a highly specialized R&D environment.

To the right shows different modules (some machine learning) that must be maintained for a ”self-driving” car. As you can imagine, a lot of infrastructure must be maintained and managed.

Therefore, data engineers serve AI/ML engineers as well.



The Data Engineering Lifecycle

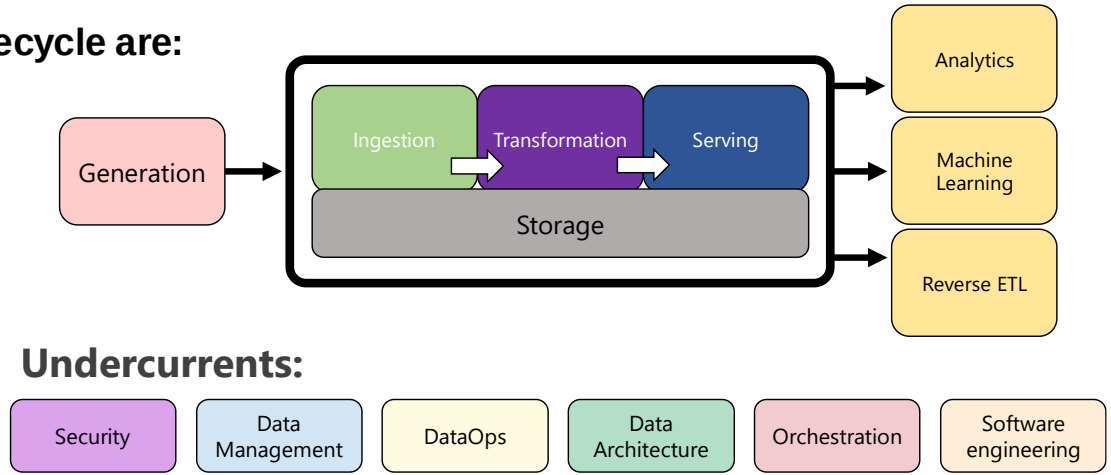
What is the Data Engineering Lifecycle?

Data engineering lifecycle is the process of turning data into useful end products.

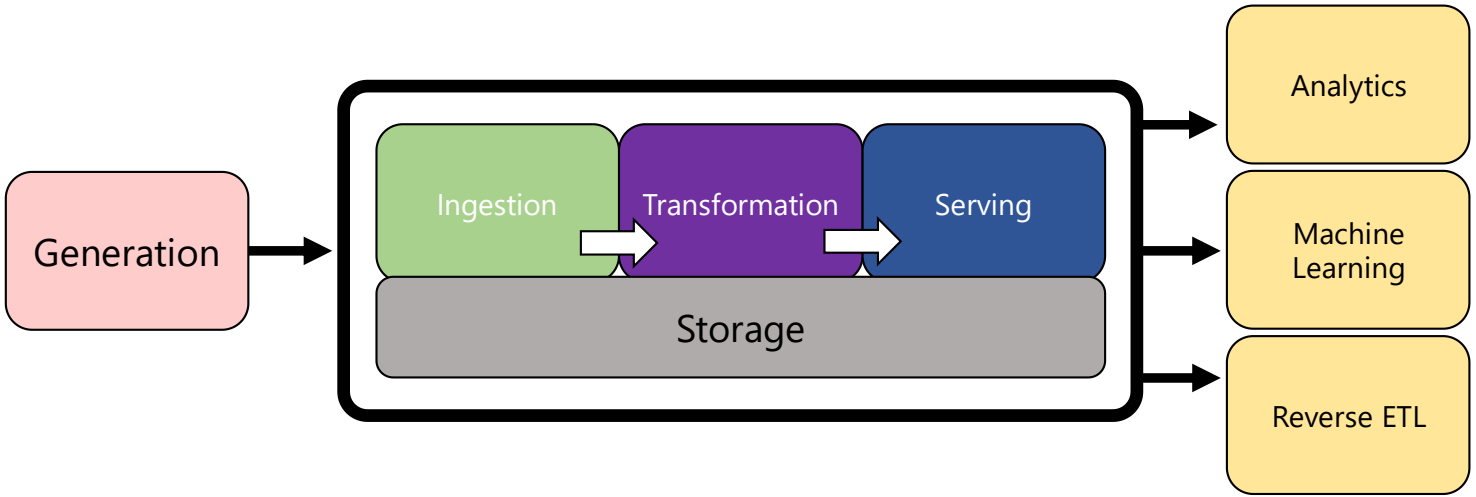
These end products are consumed by data scientists, analysts, software engineers,and machine learning practitioners.

The stages of the date engineering lifecycle are:

1) Generation
2) Storage
3) Ingestion
4) Transformation
5) Serving



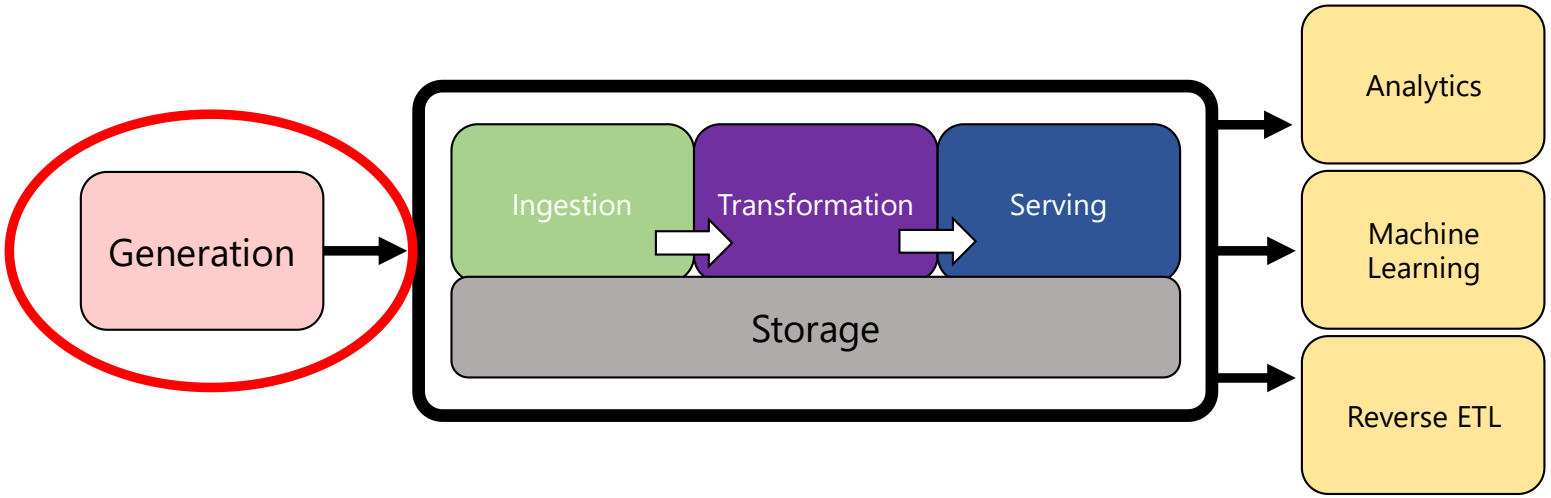
Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Generation

The source system that creates data can come from many different sources:

- User data from app usage going into a database
- Connected sensors
- Internet of things (IoT) devices
- Manual data entry

Data engineers do not have control over these sources where data is created, but they need to understand them closely.

- How frequently do these sources collect data?
- How are signals converted into data?
- What is the environment/operating domain that creates the data?
- What about the environment can bias or break it?

A set of LIDAR sensors on a car (top) and a LIDAR-generated 3D point cloud of a street.

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Generation – Data Can Come From Many Places!

Connected devices like the Apple Watch

Manual data entry

Environment sensors like LIDAR

Drones broadcasting GPS location

Aircraft sensors

Apps and websites

Surveys and paperwork

Generation – Nothing is Simple!

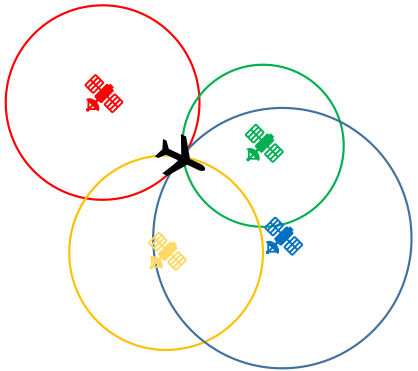
Understanding the source that generates data can be a rabbit hole.

Let's take getting a "precise" location of a vehicle using GPS. Simple right? *WRONG!!!!*

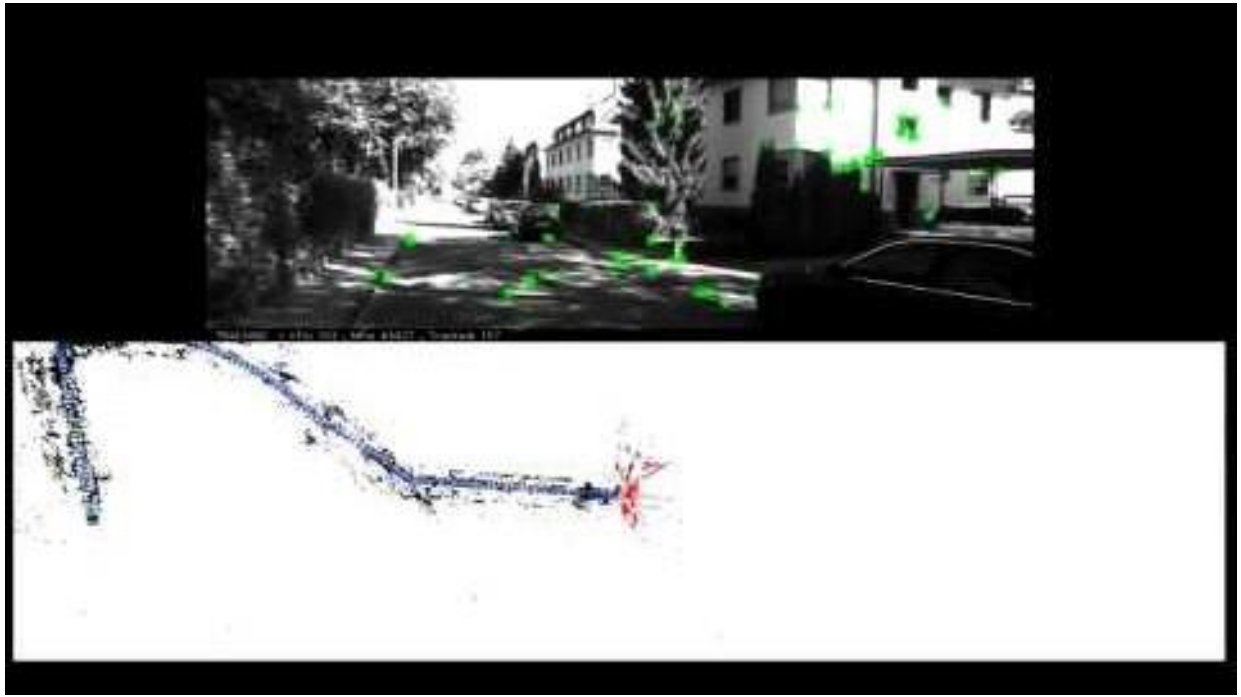
GPS approximates with 4 satellites at a time to determine location, and the faster the vehicle is going the more error it will accumulate.

Vehicles can supplement for this error using dead reckoning and radio ground stations, but faster speeds still will create errors.

This error may not matter for a food delivery service, but it does matter for a vehicle that can right-turn too early or an airplane that self-lands!



EXERCISE: Mapping an Environment with SLAM



What is odd with the above mapping operation? What is likely the cause?

SOURCE: http://webdiis.unizar.es/~raulmur/orbslam/

Evaluating Source Systems

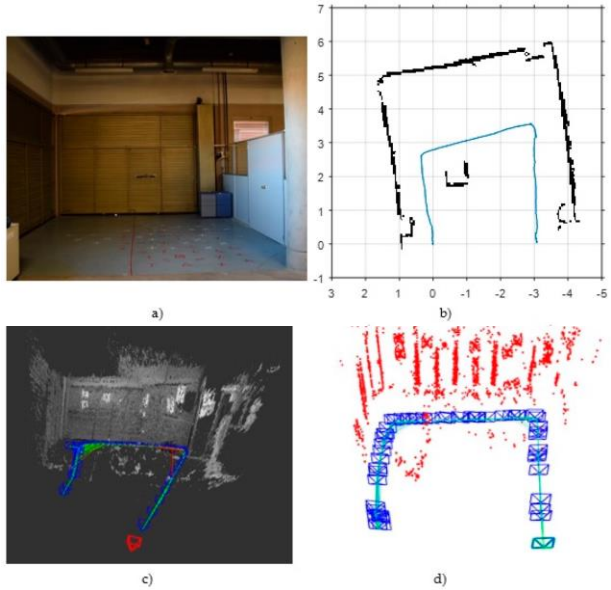
There are are a lot of things to consider when evaluating the source system:

What are the characteristics of the source creating data? How frequently and quickly, how much, and what kind of data is generated?

Is the data consistent or noisy?

What is the schema of the data? What format?

Does the environment data is being collected from change frequently?



Garcia et al. demonstrates an aerial drone mapping a poorly-lit, GPS-denied indoor environment.[1]

[1] López, E.; García, S.; Barea, R.; Bergasa, L.M.; Molinos, E.J.; Arroyo, R.; Romera, E.; Pardo, S. A Multi-Sensorial Simultaneous Localization and Mapping (SLAM) System for Low-Cost Micro Aerial Vehicles in GPS-Denied Environments. Sensors 2017, 17, 802.

Evaluating Source Systems

Can the data have gaps or biases?

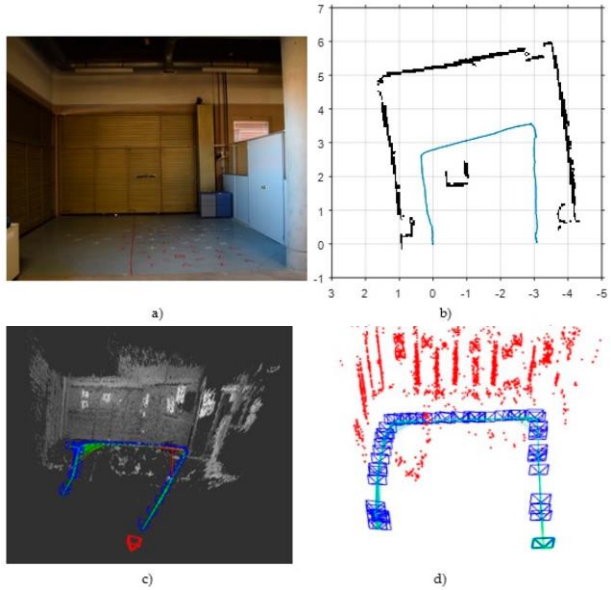
How controlled is the environment that data is being collected from? Does having more variables create further uncertainty?

How is the data converted from a signal into raw but consumable data?

Is there QA processes put in place to ensure bad data and errors do not come through?

Can all these characteristics of the source create problems with the end application?

Just to name a few!



Garcia et al. demonstrates an aerial drone mapping a poorly-lit, GPS-denied indoor environment.[1]

[1] López, E.; García, S.; Barea, R.; Bergasa, L.M.; Molinos, E.J.; Arroyo, R.; Romera, E.; Pardo, S. A Multi-Sensorial Simultaneous Localization and Mapping (SLAM) System for Low-Cost Micro Aerial Vehicles in GPS-Denied Environments. Sensors 2017, 17, 802.

Evaluating Source Systems

Whether the data source is a spreadsheet maintained by a person or a complex sensor gathering environmental data…

Understand the volume and frequency of data generation, as well as its properties and the environment it comes from!

Schemas and documentation on everything that produces the data is going to be helpful, whether it's sensor manuals or somebody's technique in hand-making the data.

It is hard to generalize knowledge about data sources because they have wide degrees of complexity and variety to them.

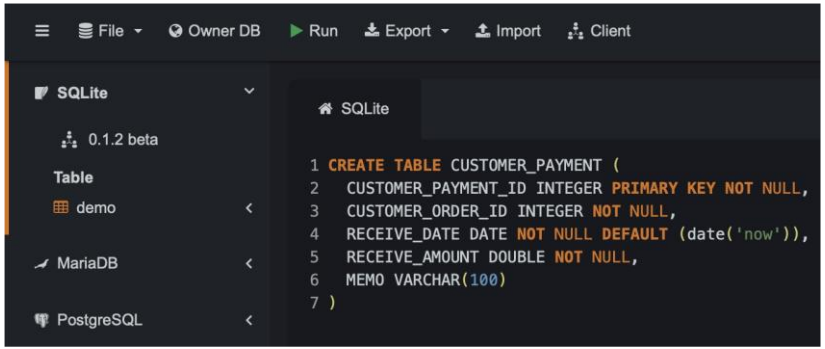
Schema versus Schemaless

Something that must be reasoned with is whether a data source enforces a schema.

A **schema** is a definition of the data, such as a table and columns, or an expected JSON output.

A **schemaless** data source is one that the application can arbitrarily define, such as PDF documents, logs, written memos, or blob data.

As a data engineer, you may find your role sometimes being taking one schema (or lack of) and converting it into another schema.



Relational databases that use SQL enforce strict schemas through table definitions.

SIDEBAR: Operating Domain

Consider the Source of the Source!

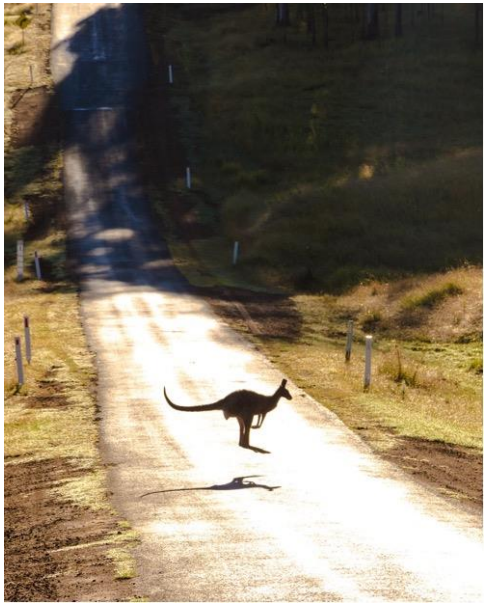
Operating Domain: Selection Bias

Despite best efforts to gather data effectively, things can still go wrong once you release a system into the wild.

A self-driving car system may have been trained to recognize pedestrians and even deer, but what if it was never trained to recognize a kangaroo?

This is exactly what happened to Volvo when they tested their autonomous vehicles in Australia.

The kangaroo was not recognized, and its jumping motion thwarted the object trajectory prediction.



A kangaroo is an example of selection bias, something never captured in the dataset due to selection being too local or not representative of the larger domain.

Operating Domain: The Problem with Outliers

Another problem is outliers, or unlikely events, that were not captured in data.

EXAMPLES: in self-driving cars, consider a fallen power line, a broken stoplight, a stop sign obscured by a tree, a locust storm, an exotic escaped zoo animal.

The **law of truly large numbers** states that with a large enough domain, any outrageous thing (an outlier) is likely to be observed.

Even untested combinations of normal conditions can create outliers (e.g. stop sign with graffiti during rainy weather).

Our data can only sample so much, and in the wild the volume of outliers is going to be high.

Operating Domain: The Problem with Outliers

To the right are examples of a well-trained neural network unable to recognize images correctly due to road objects in abnormal positions.

SOURCE:
https://arxiv.org/abs/1811.11553

Operating Domain: The S-Curve

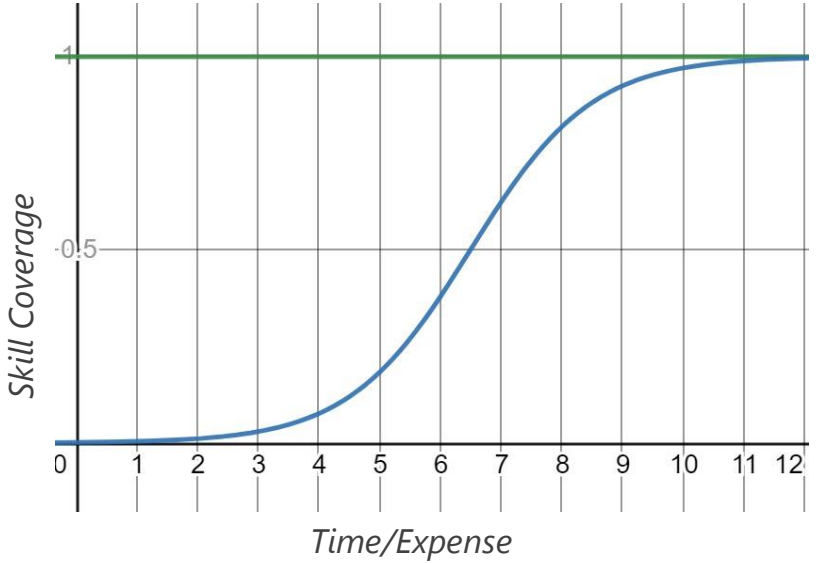
Stefan Seltz-Axmacher (CEO of defunct *Starsky Robotics*) shared an interesting industry insight on AI capability.

Take an “AI” system like virtual assistants (e.g. Alexa, Siri) or fully autonomous vehicles; many popularly believe that AI capability is exponential much akin to Moore’s Law, but it is in fact a logistic S-curve.

Time and expense is invested to create an “AI” with more data and skills, hoping to nearly reach 100% coverage of desired capabilities.

The logistic curve to the right shows an ideal situation where enough time and expense shows a nearing of 100% skills coverage.

Is this realistic though? Any guesses on what the curve looks like in reality?



https://medium.com/starsky-robotics-blog/the-end-of-starsky-robotics-acb8a6a8a5f5
http://www.autonocast.com/blog/2020/3/27/181-stefan-seltz-axmacher-on-the-end-of-starsky-and-the-future-of-autonomy

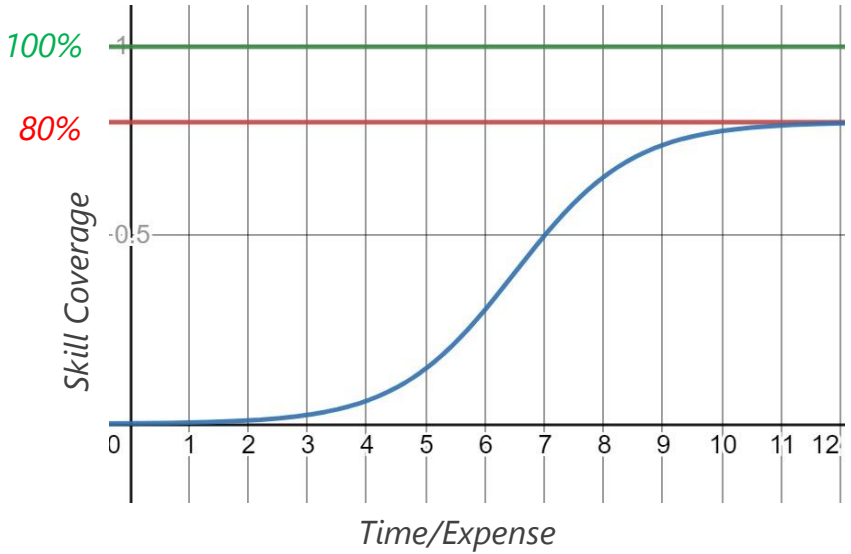
Operating Domain: The S-Curve

To the right is a more realistic outcome, where skill coverage converges on a much lower ceiling than 100%.

This is the largest problem with the “AI” industry. Several tech companies building “AI systems” found that several skills can be added in stride with initial expense, but a diminishing return for “cost per new skill added” is ultimately encountered.

Due to computational and data limits, and the high volume of outlier cases, it becomes practically impossible to create an “AI” that nears 100% coverage of needed skills without exponential expense.

This demonstrates that it is not AI capability that is exponential, but rather the expense to make it.



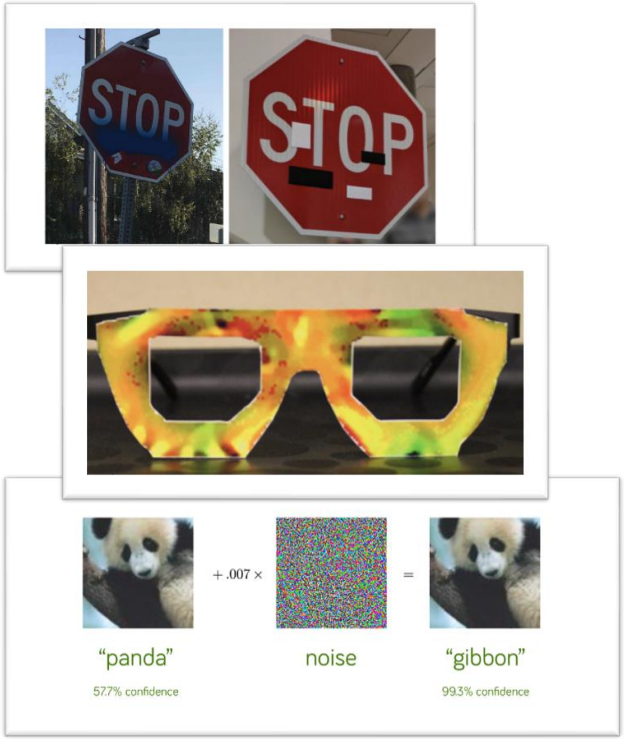
https://medium.com/starsky-robotics-blog/the-end-of-starsky-robotics-acb8a6a8a5f5
http://www.autonocast.com/blog/2020/3/27/181-stefan-seltz-axmacher-on-the-end-of-starsky-and-the-future-of-autonomy

Operating Domain: Adversarial Attacks

If biased sampling and outliers are not enough of a challenge, consider cases where synthetic outliers are created.

- A stop sign with malicious stickers is identified as a “45 MPH” sign.
- Pixelated face wear and photos of other people can thwart facial recognition software.
- Pixelated noise on a panda photo can misclassify it as a gibbon.
- Children in Halloween costumes are not recognized as pedestrians.

Adversarial attacks are hard to mitigate, and the only surefire solution is preventing exposure to uncontrolled environments or engineering safeguards to reasonable risk acceptance levels.



Operating Domain: Selection Bias

The million-dollar question: so how do we deal with outliers?

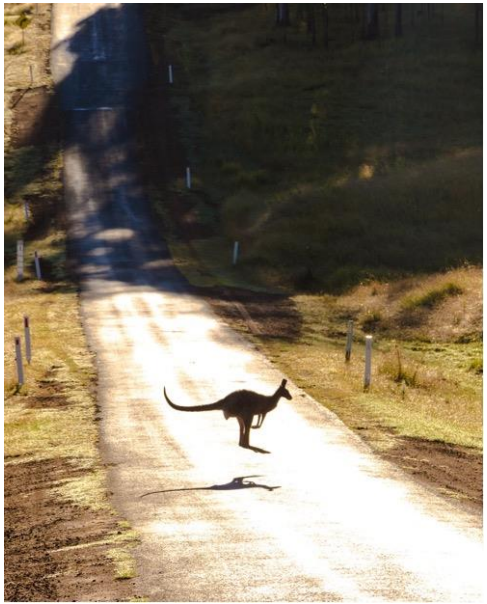
Are we ever going to collect enough data to effectively identify outliers? Compensate for data gaps?

Maybe the answer is “it depends”:

If the domain is narrow and controlled enough, maybe we can collect enough data or create explicit rules that do not rely on data.

If the domain is broad and uncontrolled, consider safe “guardrail” rules when an outlier is encountered.

In uncontrolled domains, find effective inputs and heuristics that are reliable and make sense for a given application (e.g., radar being used to emergency brake for an obstacle).



Operating Domain: Data Rot

Data used to train a model can work for awhile, but the data is likely to become stale (known as data rot).

The world will change, trends will come and go, and nothing will stay the same which renders captured data obsolete.

Airport congestion data from 2016 is hardly relevant in 2020 as new terminals, airline presence, construction projects, unusual weather seasons, and other factors disrupt.

Something as unpredictable as clothing fashions can cause problems with image recognition.

Even changes to camera resolution, hardware configurations, lighting, correction filters, and picture quality can cause training data to become outdated (contemplate this when budgeting for data acquisition!)

Operating Domain: Data Entry and Correction

In the field, consider how data will be entered and corrected!

If I have a system that predicts when aircraft parts need replacement, the user needs to input corrected data when it is wrong.

If there is a large amount of data that needs to be corrected on the field when a prediction is wrong, consider practicality for being “online” and streamline the process of inputting the data.

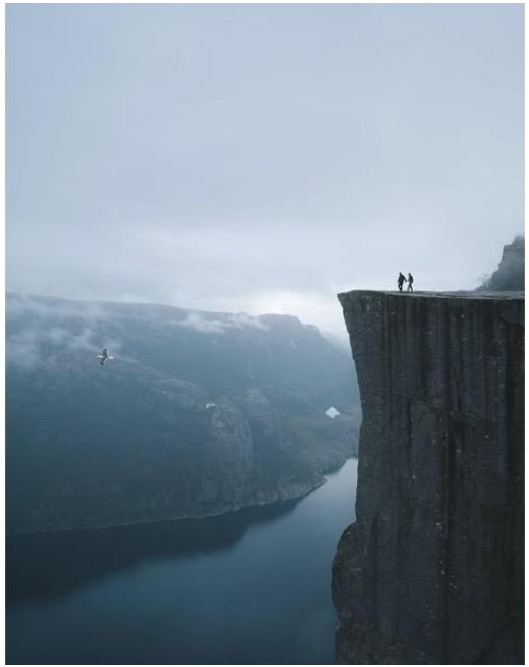
Not accounting for corrective data entry can quickly fail a data project!

Operating Domain: Data Rot

“Those of us in machine learning are really good at doing well on a test set, but unfortunately deploying a system takes more than doing well on a test set.

“All of AI, not just healthcare, has a proof-of-concept-to-production gap. The full cycle of a machine learning project is not just modeling. It is finding the right data, deploying it, monitoring it, feeding data back [into the model], showing safety—doing all the things that need to be done [for a model] to be deployed. [That goes] beyond doing well on the test set, which fortunately or unfortunately is what we in machine learning are great at.”

- Andrew Ng, Former Head of Google Brain and Deep Learning Education Pioneer

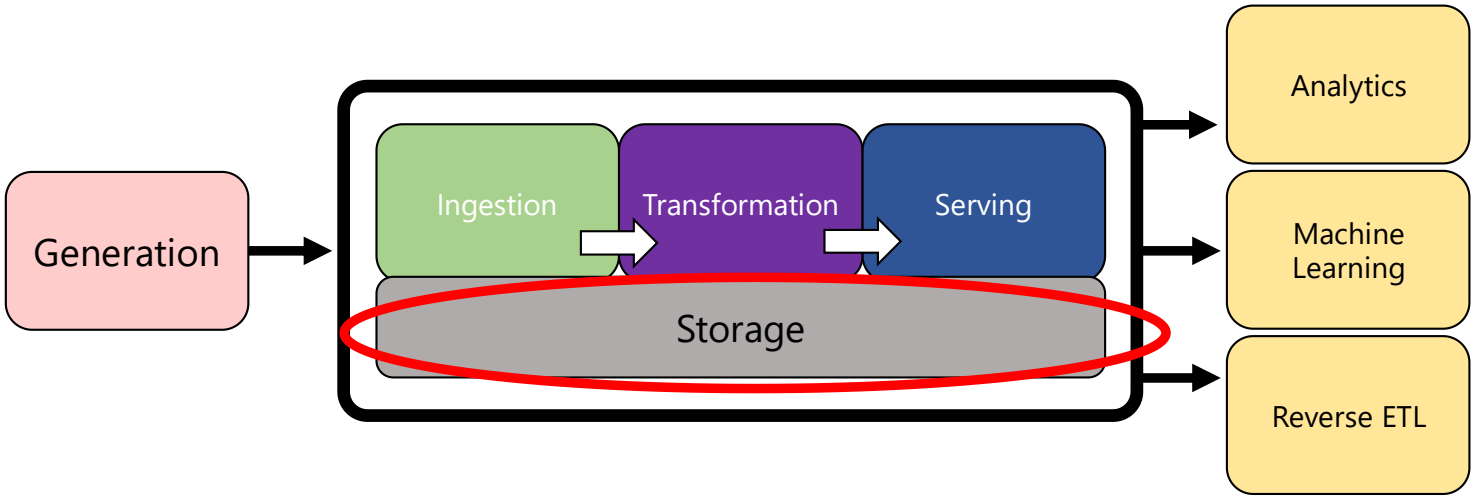


https://spectrum.ieee.org/view-from-the-valley/artificial-intelligence/machine-learning/andrew-ng-xrays-the-ai-hype

Storage

Data Does Not Persist Without It

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Storage

Storage is more than just storing data.

How frequently you access and/or write the data is going to affect what type of storage you use.

Many storage platforms provide ways to transform data and offer querying features (e.g. relational databases and SQL)

Storage is used throughout the entire data engineering lifecycle, including ingestion, transformation, and serving.

Storage Considerations

Here are some things you should consider when choosing a storage solution.

- Does the storage meet speed and capacity and requirements, and agree with the rest of the downstream processes?
- Are query capabilities supported?
- Is it schema or schemaless?
- Will it create bottlenecks?
- Where will the data geographically be stored? Is it compliant with policies and legal?
- Can redundancy and SLA agreements be met?

Storage Considerations

You need to weigh the velocity and frequency you will access data.

Data that is accessed frequently (daily) is called **hot data**.

If it is accessed less often it is **cold data**, as in once a year or so.

Lukewarm data is somewhere in between, every week or month.

Storage Considerations

Especially for cloud solutions, this can heavily affect what mix of storage solutions you use and optimizing the cost.

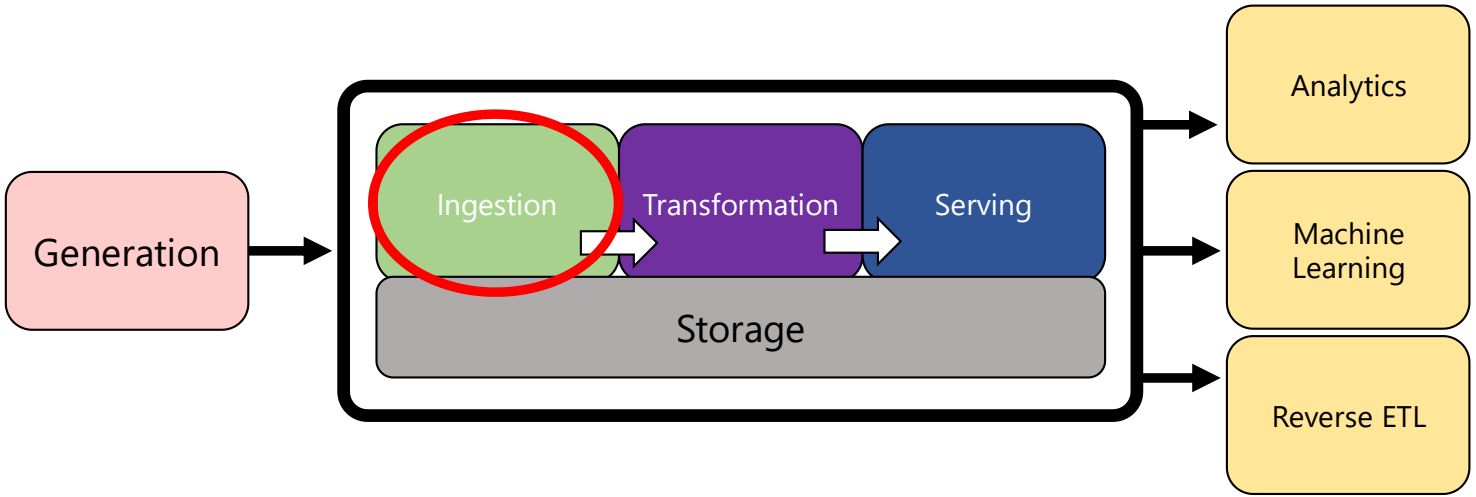
For data that is rarely retrieved (a couple times a year) you will want to use a cold archival storage solution that is only pricey on retrieval.

For data that is readily available, you will need a storage that allows fast and frequent retrieval at lower cost.

Ingestion

Taking Data from the Source

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Ingestion

Ingestion can be the most common source of bottlenecks, and data engineering rarely has control over those processes

For example, if a sensor goes down or stops transmitting the data engineer can probably only make an email or phone call to that owner.

This also includes the data storage that receives the ingestion, as the data engineer probably does not have control over that either.

Ingestion

Some considerations for ingesting data and choosing systems:

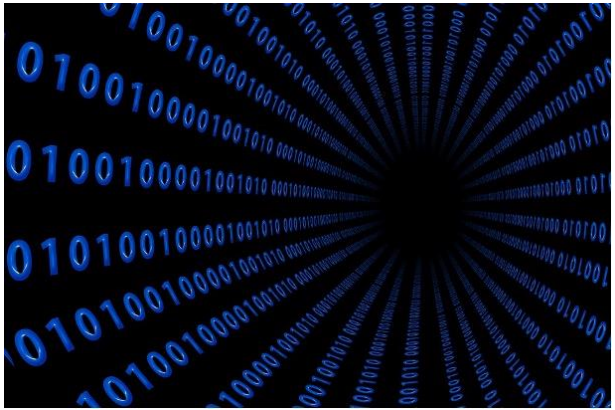
Can I minimize redundant storage by consolidating ingestion to one target, rather than several?

What volume and speed will the data be ingested, and can the platform support it?

Is the format usable for the data engineer's downstream systems?

Are transformations needed on the data during ingestion? After ingestion?

Is the data ingested and available in a timely enough matter?



Batching versus Streaming

Think for a moment and realize at the origin of the data, it is ***streaming*** and being captured in live time.

But downstream, this data gets put into ***batches*** and loaded as chunks into other systems like applications, databases, and reports.

How frequently these batches are updated depends on each application's requirements.

Streaming solutions (event-driven) that propagate live streams through the entire system are becoming more popular so data is never stale and always up-to-date.

The downside of streaming is it is more complex and expensive to implement, requiring a lot more overhead. Machine learning rarely benefits from streaming, because machine learning often trains in batches anyway.

Batching versus Streaming

Do I actually have a valid use case and need for streaming?

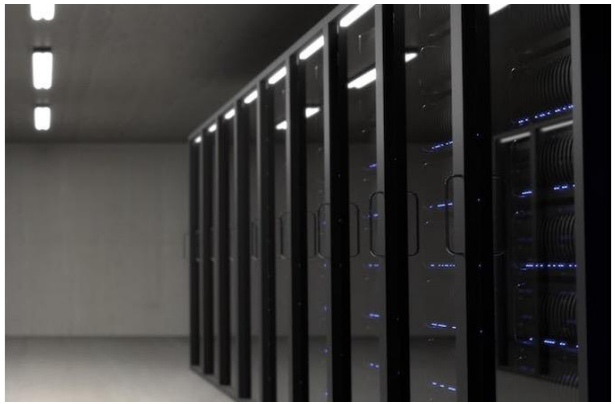
Can downstream systems handle streaming, or are they better suited for batching?

Can I use smaller and more frequent batches as opposed to live time records.

Do I have the budget and resources to implement streaming?

Do the tradeoffs make sense for streaming versus batching?

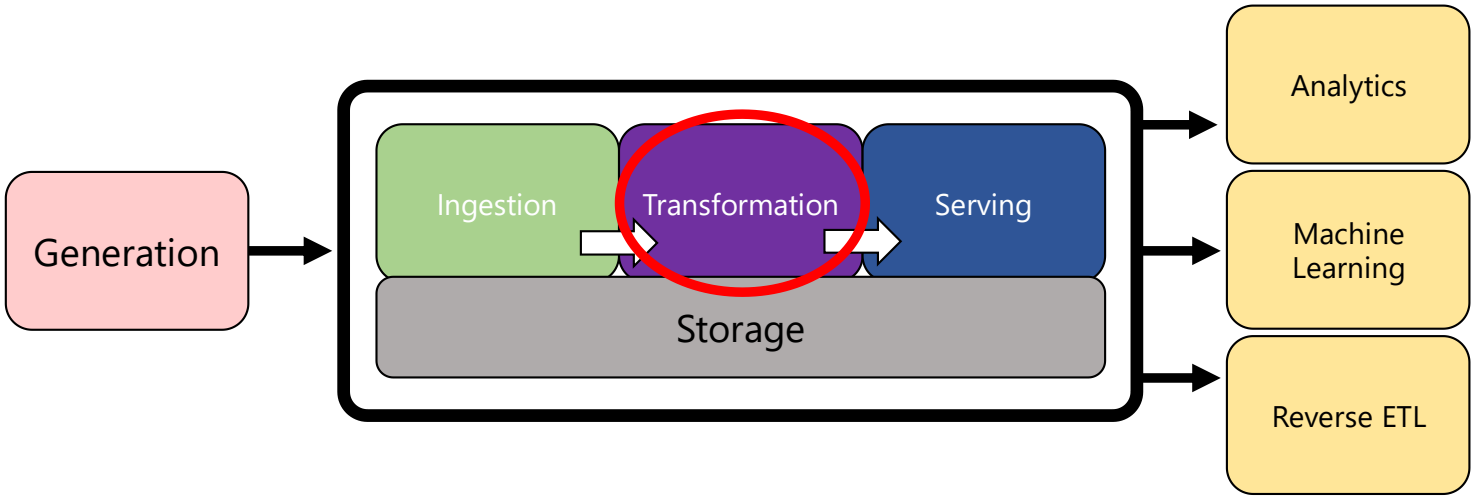
Do downstream users benefit from having the most up-to-date data? Is it used for machine learning or operational analytics and insights?



Transformation

Converting Data

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Transformation

When you have your data ingested (and now likely stored somewhere else), you will want to do tasks with it so it is useful to reports, machine learning, and analysis end users.

Transformation can include a number of large and small changes to the data, including…

- Changing dates and times stored as strings into datetime types.
- Joining to other data sources for more complete information.
- Formatting and normalizing data into row-based records.

sepal_length	sepal_width	petal_length	petal_width	species
5.9	3.0	5.1	1.8	virginica
5.4	3.0	4.5	1.5	versicolor
5.0	3.5	1.3	0.3	setosa
5.6	3.0	4.5	1.5	versicolor
4.9	2.5	4.5	1.7	virginica

sepal_length	sepal_width	petal_length	petal_width	species
5.9	3.0	5.1	1.8	2
5.4	3.0	4.5	1.5	1
5.0	3.5	1.3	0.3	0
5.6	3.0	4.5	1.5	1
4.9	2.5	4.5	1.7	2

Turning a "species" column for an iris dataset into numerical categories.

Transformation

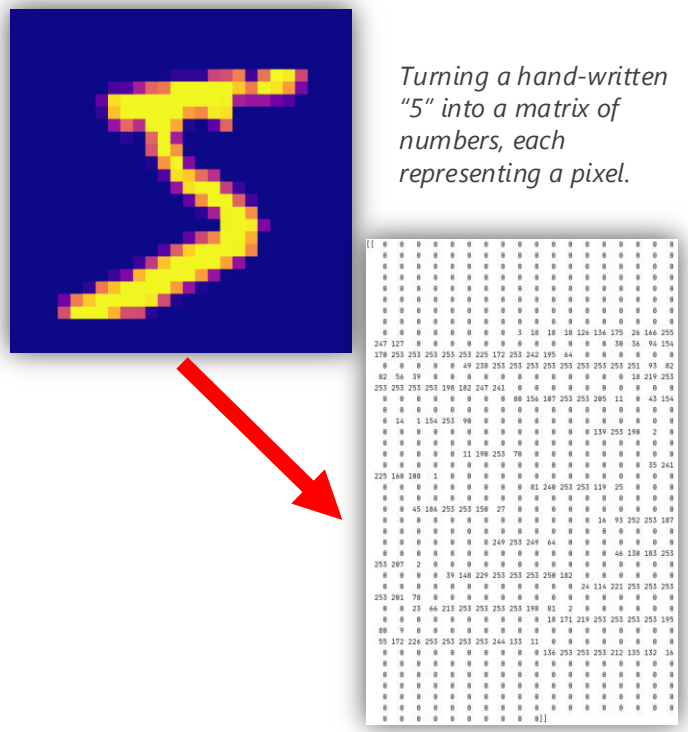
Transformation can happen anywhere in the data engineering lifecycle, including ingestion and serving.

Analysts may use SQL queries to transform data and perform aggregations and windowing functions for reporting.

Machine learning engineers may perform *featurization* where data is extracted and transformed into different features for machine learning.

Spreadsheet and Python users may add formulaic columns and models to the data, to perform tasks like forecasting.

Be sure to consider all the possible stages of transformation and what should be automated, versus self-served.

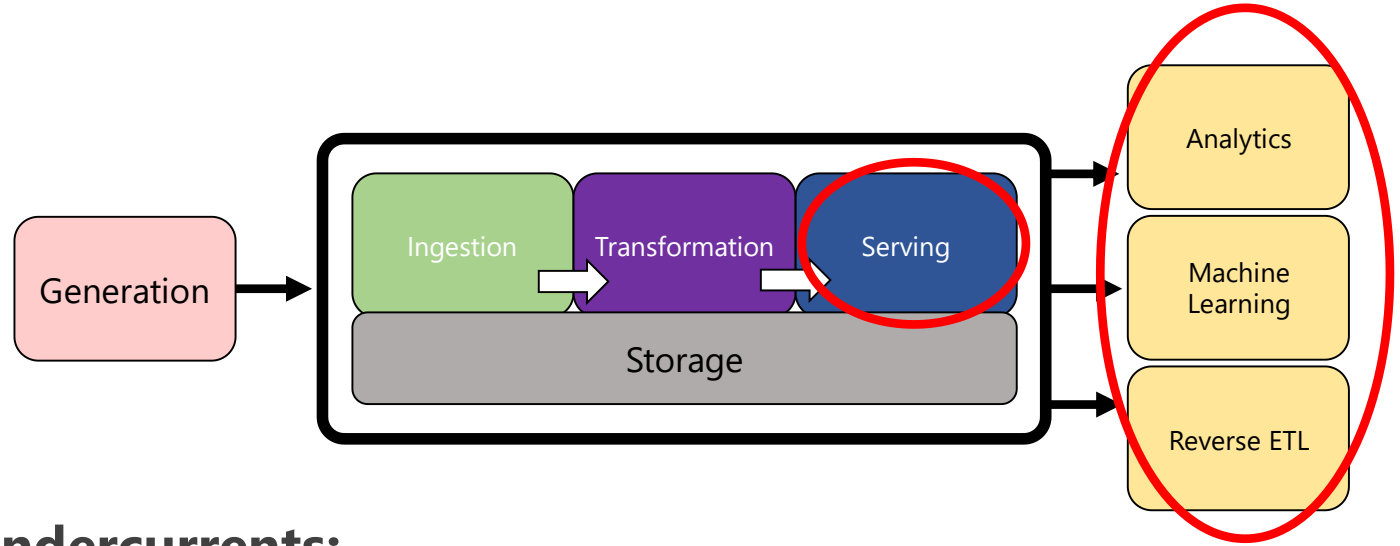


Turning a hand-written "5" into a matrix of numbers, each representing a pixel.

Serving Data

Where Value is Delivered

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

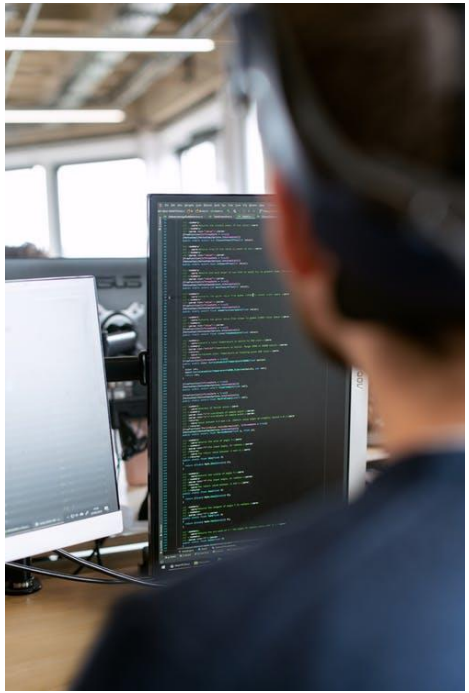
Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Delivering Value with Data

Data must be actively used to create value for an organization, and the last thing you want is for it to sit unused.

Vanity projects or “busywork projects” (including unnecessary migrations to new tools and platforms) are career liabilities, so make sure your projects are truly delivering value to aligned objectives to an organization.

Data must be intentional, managed through the data engineering lifecycle, and fulfill a specific objective that meets production… serving analytics, machine learning, and production applications.



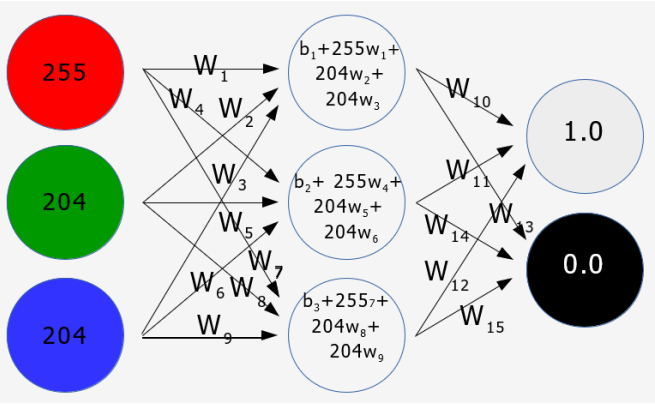
Machine Learning

Data engineers can share responsibilities with machine learning engineers and analytics users, so delegate effectively.

Feature stores are a versioning and control system for machine learning practitioners to keep track of machine learning data, and data engineering may find themselves supporting this.

While you want to maintain and delegate responsibilities effectively, data engineers should get familiar with machine learning.

Let’s get a glimpse into the machine learning workflow.



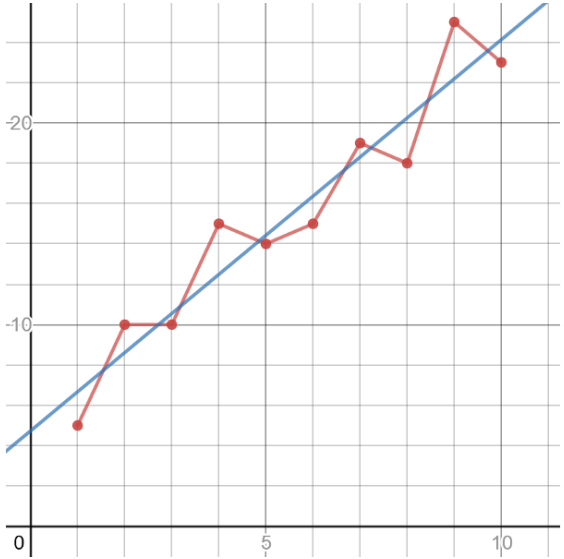
Machine Learning: Overfitting, Variance, and Bias

Overfitting means that our ML model works well with the data it was trained on but fails to predict correctly with new data.

This can be due to many factors, but a common cause is the sampled data does not represent the larger population and more data is needed.

The red “connect-the-dots” model has high **variance**, meaning its predictions are sensitive to outliers and therefore can vary greatly.

The blue line has high **bias**, meaning the model is less sensitive to outliers because it prioritizes a method (maintaining a straight line) rather than bend and respond to variance.



The red “connect-the-dots” model to the right is likely overfitted (high variance, low bias), but the blue linear regression line (low variance, high bias) is less likely to be overfit.

Machine Learning: Overfitting, Variance, and Bias

Linear regression is a highly biased method and is resilient to overfitting.

Decision trees are a low biased method with high variance, and overfit easily.

There are other remedies to mitigate overfitting, the most basic being separating training data and test data.

The model is fit to the training data, and then is tested with the test data.

If the test data performs poorly compared to the training data, there is a possibility of overfit (or just no correlations altogether).

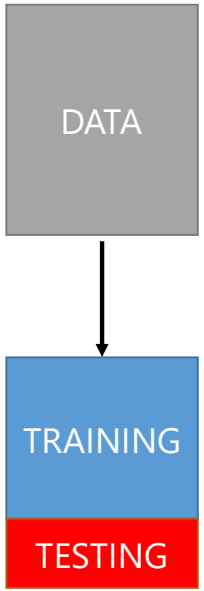
Machine Learning: Training and Testing Data

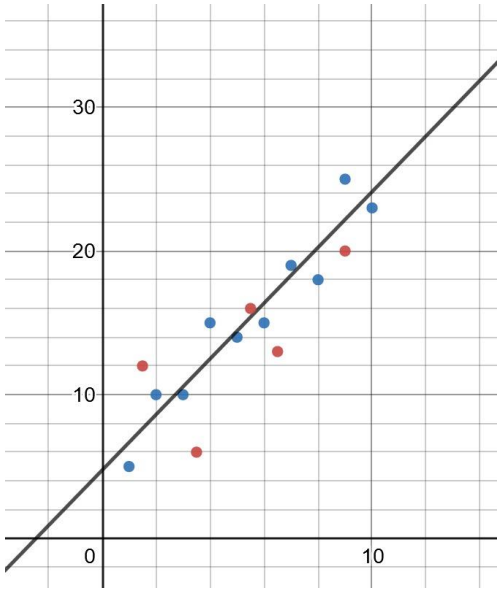
A common practice to proactively prevent overfitting in machine learning is to separate training data and testing data.

Training data is data used to fit a model and is typically 2/3 of the data.

Test data is used to test the model and is the remaining 1/3 of the data.

By omitting the testing data from training, we see how well the model works on data it has not seen before and change our parameters accordingly.





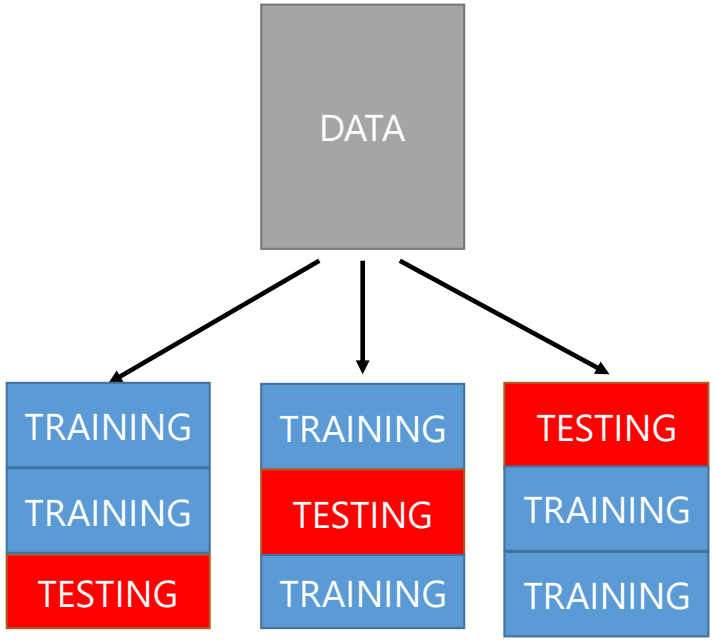
Machine Learning: Cross-Validation

We can take this concept of training/testing data a step further, and test different combinations of training and testing data.

This is known as **cross-validation**, the gold standard of validation techniques.

To the right we have **3-fold cross validation** which breaks the data into thirds and uses one of the pieces for testing.

We can then evaluate how well each of these perform, being able to compare different parameters and models (e.g. logistic regression vs decision trees) and see which setup produces the best performance.

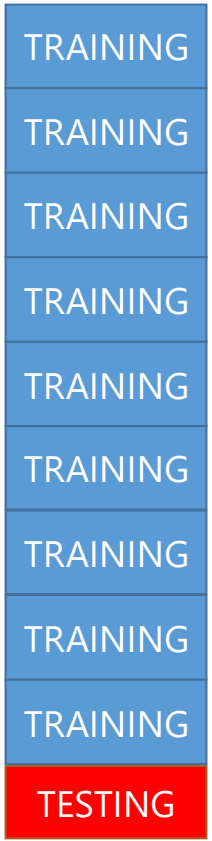


Machine Learning: Cross-Validation

Note that **k-fold cross validation** allows us to slice our data into any number and not just 3 (typically 3, 5, or 10).

For example we can do 10-fold cross validation and validate 10 different combinations of training/test data.

The most extreme form of folding is **leave-one-out cross validation**, which omits one data record for testing and uses the remaining records for training, and this is done repeatedly.

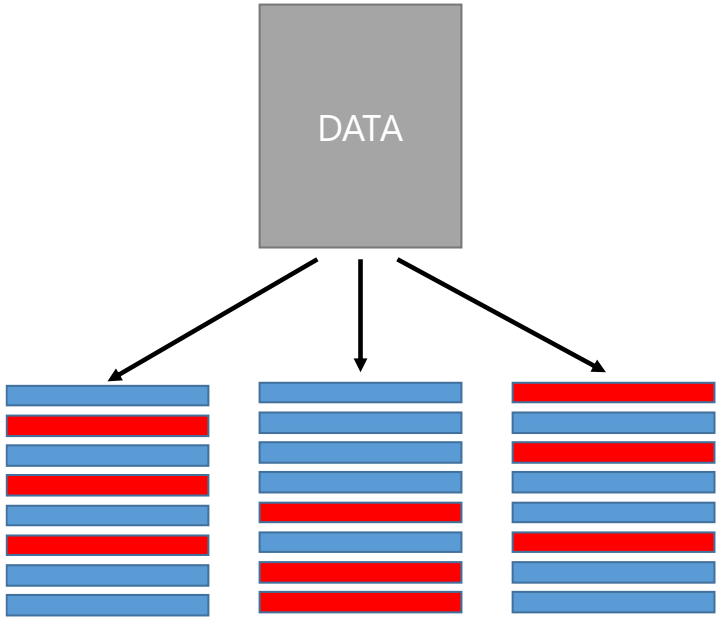


Machine Learning: Random Fold Validation

As you may be noticing, machine learning often tries to overcome data variance with randomness.

A variant with fold validation is **repeated random fold validation**, where we randomly shuffle the data and create random train/test folds as many times as desired.

This is helpful when we need to mitigate variance in the model.



Machine Learning: Which Validation to Use?

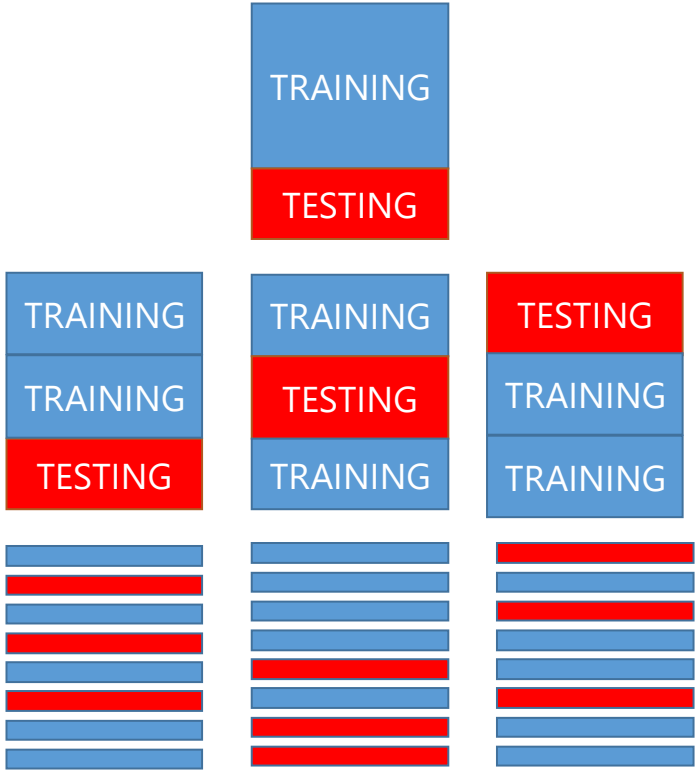
Generally, you will want to prefer k-fold validation as it is the gold standard.

A single train/test split might be warranted if performance of machine learning algorithm is slow and has enough data with lower bias.

Use the random fold split to mitigate variance in the model while balancing training speed and dataset sizes.

We will talk about class imbalance later, but if you do not have an equal number of samples for each class, you might want to consider using stratification in your k-fold validation.

Stratification means an equal proportion of data for each class is sampled for training and testing data (even it is sampled repeatedly), so that no class is neglected.



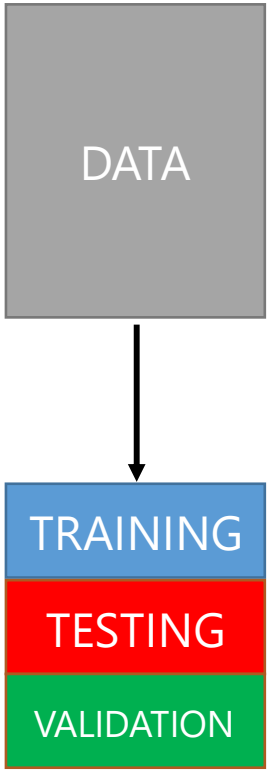
Machine Learning: Validation Data

Validation is a separate type of testing data used to compare performance of different models.

When you are comparing two or more models (e.g. logistic regression versus decision trees), you may hold back one more chunk of data for validation.

It is different than the testing data, which is used to tune parameters of the individual model, not compare different models altogether.

This is a final stopgap to stop cherry-picking, and ensure models were not shopped and leaked clues about the testing dataset to training.



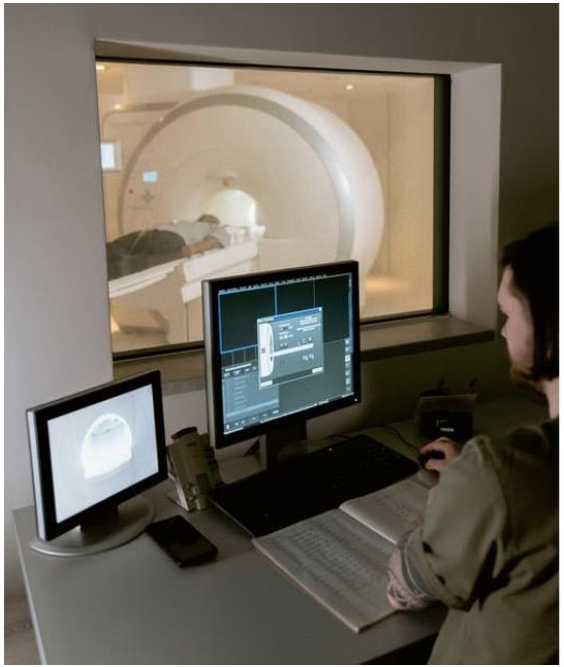
Machine Learning: Is a Train/Test Split Enough?

"It turns out that when we collect data from Stanford Hospital, then we train and test on data from the same hospital, indeed, we can publish papers showing [the algorithms] are comparable to human radiologists in spotting certain conditions.

"It turns out [that when] you take that same model, that same AI system, to an older hospital down the street, with an older machine, and the technician uses a slightly different imaging protocol, that data drifts to cause the performance of AI system to degrade significantly. In contrast, any human radiologist can walk down the street to the older hospital and do just fine.

"So even though at a moment in time, on a specific data set, we can show this works, the clinical reality is that these models still need a lot of work to reach production."

- Andrew Ng, Former Head of Google Brain and Deep Learning Education Pioneer



https://spectrum.ieee.org/view-from-the-valley/artificial-intelligence/machine-learning/andrew-ng-xrays-the-ai-hype

Machine Learning: Ground Truth

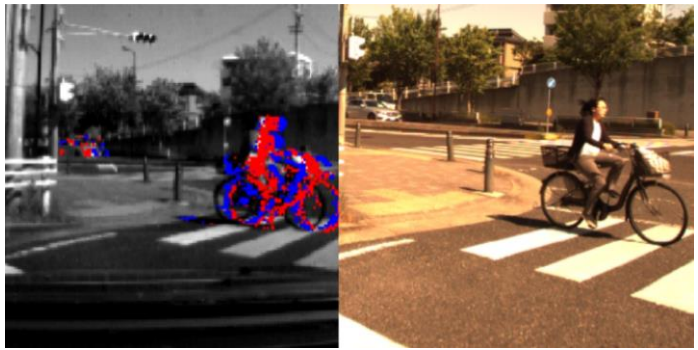
Can we trust the data is telling the truth? Or has been labelled correctly by a person or machine?

Consider a scenario where a ”self-driving” vehicle fails to recognize a pedestrian.

Is there a way for the vehicle to recognize it has failed to recognize a pedestrian?

No! This is what we call the problem of ***ground truth***, which is knowing what is actually to be true given the data or prediction.

Ground truth can be very hard to establish and comes down to reliable, focused data labels that are applied by a machine or a person.



An event camera has potential to perceive movement and activity in an environment.

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443 - 58469.

Considerations for Machine Learning

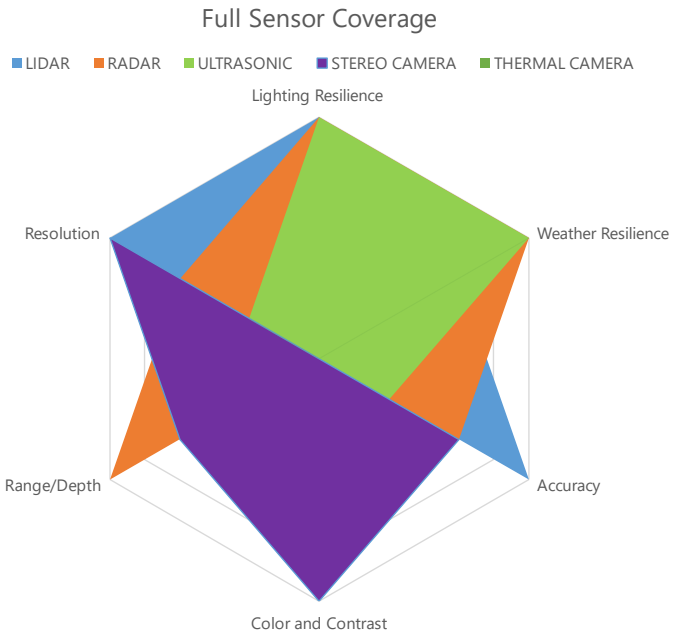
Is the data sufficient for machine learning and provides enough for training, testing, and validation?

How will roles be delegated between data engineering and ML engineering?

Is the data biased? Will it lead machine learning to be overly optimistic in accuracy?

Is the data accessible, discoverable, and available to machine learning practitioners?

Is the dataset representative of ground truth? Are things correctly labeled?



Autonomous vehicles often rely on multiple sensors in attempt to establish ground truth on what's happening in the environment.

Analytics

When you have data, obviously businesses are going to want to get insight from it: SQL queries, charts, spreadsheets, dashboards, reports, pivot tables, and other business-y activities.

As data maturity progresses, the company should gain self-service capabilities and find talent who can develop these products.

Sometimes the data engineer may take ownership of these analytics to integrate into production, embedding dashboards and analytics into software products.

Why Self-Service is Hard to Achieve

Achieving self-service can be hard in practice.

- Data has to be documented and defined, and domain knowledge has to be shared across an organization rather than siloed.
- Getting data-proficient “power users” who can perform SQL, dashboard creation, and other technical skills are not as common as one would think.
- There may be politics in sharing data and the knowledge that goes with it.
- Excessive and heavy-handed security policies can make getting access to data tedious and difficult.

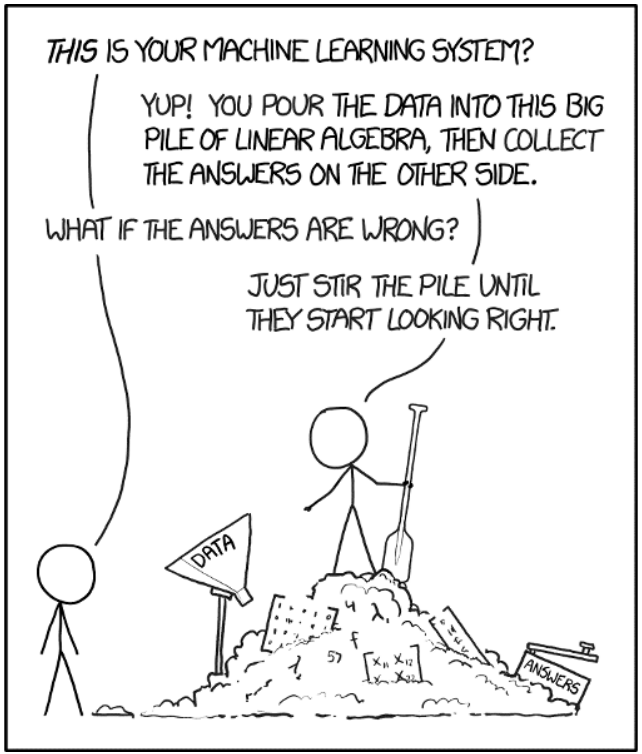
Why Self-Service is Hard to Achieve: Bad Data

Many project advocates say "data" is the lifeblood of data projects, but they underestimate the volume of bad data they will encounter.

Corrupted data, rotten data, outliers, biases, and noisy data can pollute a dataset and require at least 85% of efforts to clean it.

Because of the presence of bad data, data engineers are stuck as the gatekeepers so end users cannot run with it or make false conclusions.

In this stage, it should be asked why bad data keeps persisting or if transformations can be done to clean the data and make it available.



https://xkcd.com/1838/

Why Self-Service is Hard to Achieve: Documentation

It is common for analysts, data scientists, and other data professionals to rush in on a new dataset without documentation or context.

This can be dangerous because if there is little up-to-date documentation or unavailable experts to provide training, the data can be misused.

It just takes one missing WHERE condition on a flagged field in a SQL query to ruin an entire analysis, or worse act on it.

Keeping documentation and knowledge up-to-date easily becomes an entire team's job in itself, and companies may not be able to budget for this.

Why Self-Service is Hard to Achieve: Security

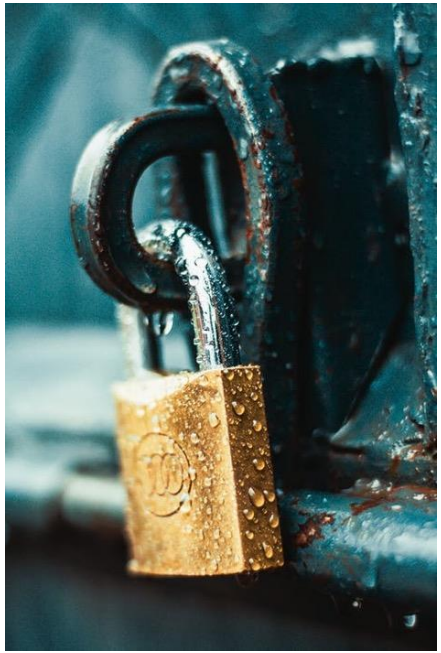
Security is necessary in any organization, but it is a difficult balancing act with nimbleness.

Maintaining security and role policies can grow to be a full-time job, and it becomes cumbersome to manage user access to data systems.

The ***principle of least privilege*** **(PoLP)** is great for preventing breaches and leaks, but it is bad for productivity!

- Bureaucratic processes and help desk tickets become barriers to getting data quickly and effectively.
- Meetings to give specific data access to specific users can take days, weeks, even months!

Sometimes these security policies are warranted, but they can be heavy-handed.



Why Self-Service is Hard to Achieve: Talent

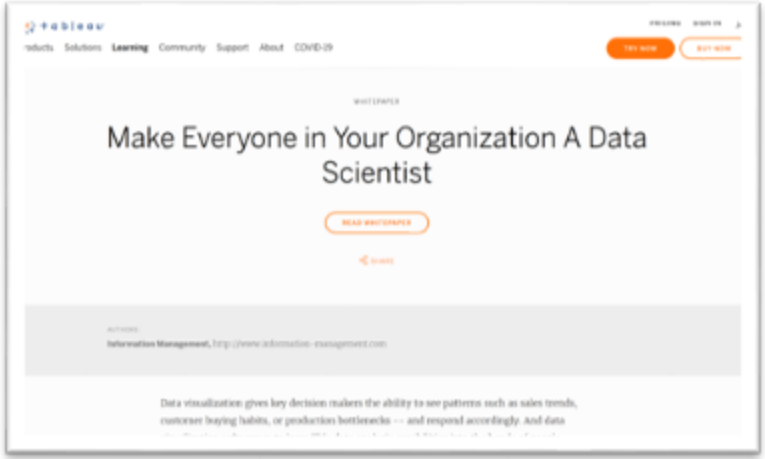
If you stay in a bubble long enough, it's easy to think *anybody* can run a SQL query or execute a Python script.

However, even in the largest Fortune 500 companies there is a sharp shortage of analysts who even know SQL.

Power users that know SQL become the gatekeepers between data warehouses and a swathe of Excel-wielding analysts.

This can make "self-service" exclusive to only people capable of performing it.

This is why there has been a growing market of "drag-and-drop" tools like Tableau and Alteryx to aid nontechnical analysts.



Tableau, a drag-and-drop visualization software, positioned its marketing by claiming their software can make anyone a data scientist.

https://www.tableau.com/learn/whitepapers/make-everyone-your-organization-data-scientist

Why Self-Service is Hard to Achieve: Office Politics

It is no secret organizations are protective of their data, but it's not just due to security or distrust concerns.

Even departments inside the same organization will not share data with each other for this reason: they do not want others doing their job… or doing it incorrectly.

- It may require *their* fulltime expertise to interpret the data, and it requires *their* domain knowledge.
- Data scientists can overestimate their ability to interpret foreign data sets and the domain knowledge needed to use it.

The solution is developing trust and buy-in with each partner, negotiate a knowledge transfer, and if needed giving them a significant role in the project.

Why Self-Service is Hard to Achieve: Shadow IT

Shadow information technology (shadow IT) is a term describing office workers who create systems outside their IT department, which can sprout like weeds in ”self-service” environments.

These systems can include databases, scripts, and processes as well as vendor and employee-made software without the IT department’s involvement.

Shadow IT can be fuel for innovation and future projects, but there is a threshold when hidden costs and security concerns accumulate and become a liability.

Nasty politics can ensue when IT departments and non-IT departments clash, accusing each other of not staying in their lane or simply co-opting roles for job security.

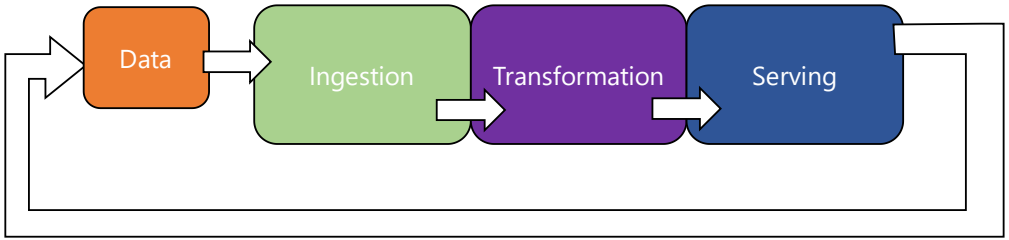
As a data engineer, be an ambassador for both sides and manage the data maturity stages accordingly, recognizing when ownership should change!

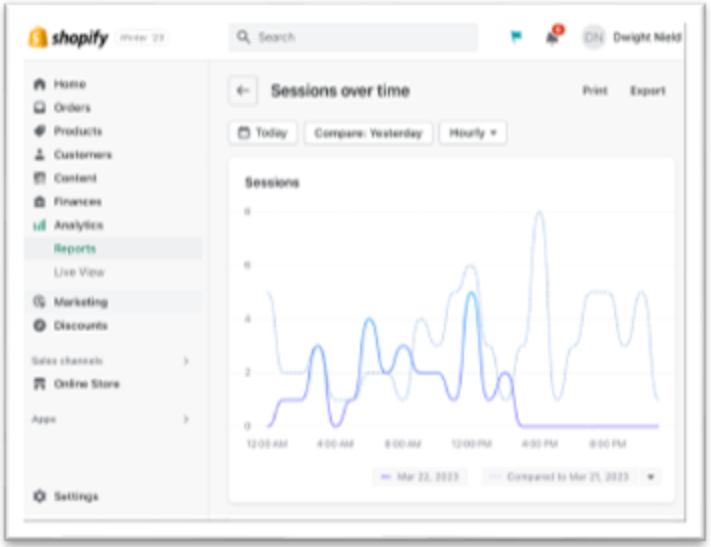
Reverse ETL

There will be moments where you want to take served data and put it back into ingestion, which is called *reverse ETL*.

For example, metrics may be generated and they want to be stored in a database and served to customers on their dashboards.

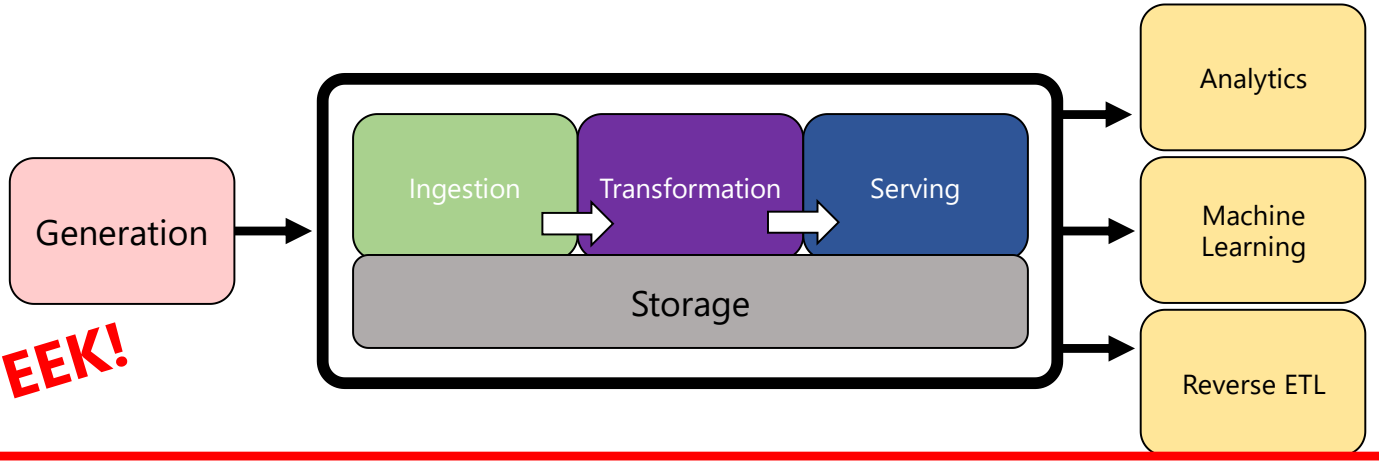
Reverse ETL is a relatively new concept, so expect it to evolve and change names over time.





Shopify allows customers to access traffic and sales metrics of their store website, which is an example of reverse ETL.

Data Engineering Lifecycle (Reis and Housley)



NEXT WEEK!

Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

EXERCISE
"There's No Correlation"

Exercise: “There’s No Correlation”

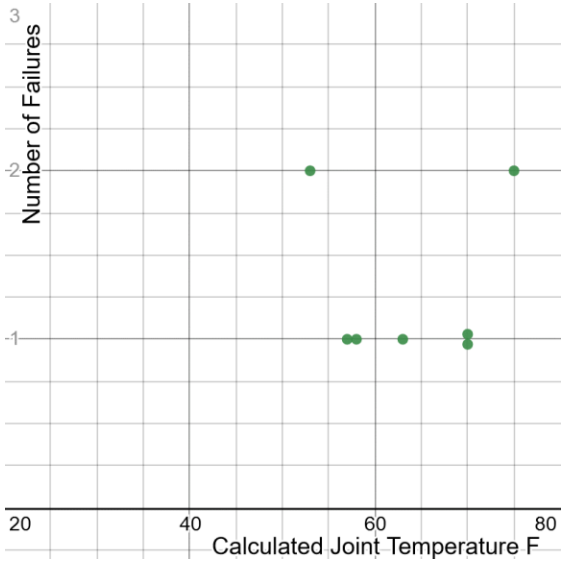
It is one day from a manned space launch.

However, your engineering team has been expressing concern about the O-Rings that seal rocket gases from releasing, and whether they perform in colder temperatures.

You share this concern with other parties and are provided data of all 7 O-Ring failures from 24 launches and the temperature (shown to the right).

The consensus from other parties is there is no correlation between number of failures and temperature.

Is this assessment correct?



temperature	o_ring_failures
53	2
57	1
58	1
63	1
70	1
70	1
75	2

Data Source: http://www.math.usu.edu/~symanzik/teaching/2009_stat6560/RDataAndScripts/sharif_abbass_project1_challenger.csv

Exercise: “There’s No Correlation”

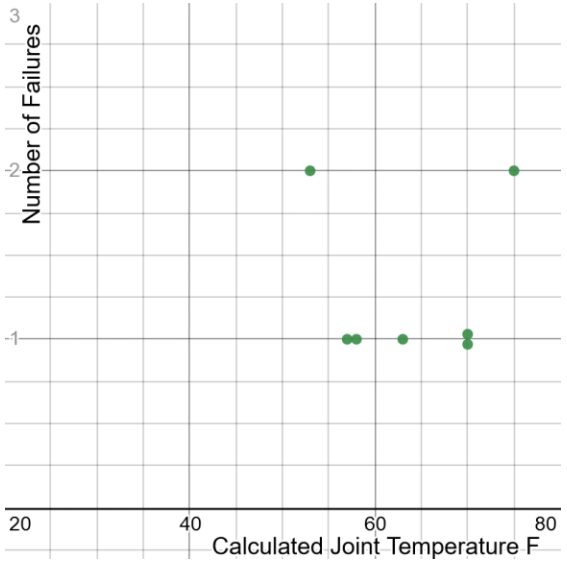
It is one day from a manned space launch.

However, your engineering team has been expressing concern about the O-Rings that seal rocket gases from releasing, and whether they perform in colder temperatures.

You share this concern with other parties and are provided data of all **7 O-Ring failures** from **24 launches** and the temperature (shown to the right).

The consensus from other parties is there is no correlation between number of failures and temperature.

Is this assessment correct? **Is anything missing**?



temperature	o_ring_failures
53	2
57	1
58	1
63	1
70	1
70	1
75	2

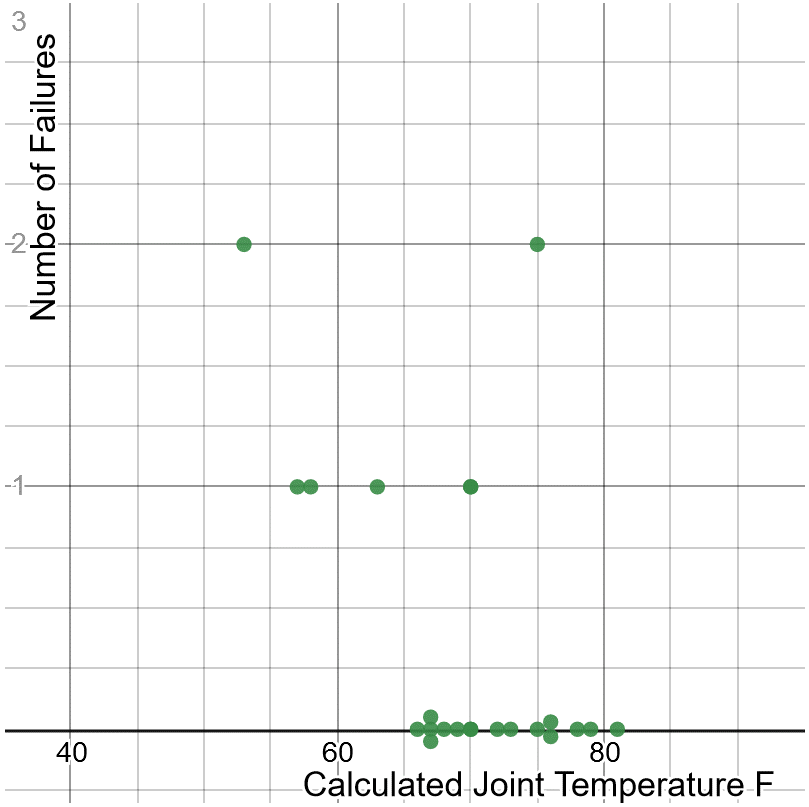
Data Source: http://www.math.usu.edu/~symanzik/teaching/2009_stat6560/RDataAndScripts/sharif_abbass_project1_challenger.csv

Exercise: “There’s No Correlation”

Notice that data from successful launches were not included, which may provide a missing piece of the story.

What are your thoughts now? Is there a correlation? Is the data showing the story?

temperature	o_ring_failures
53	2
57	1
58	1
63	1
66	0
67	0
67	0
67	0
68	0
69	0
70	1
70	0
70	1
70	0
72	0
73	0
75	0
75	2
76	0
76	0
78	0
79	0
81	0



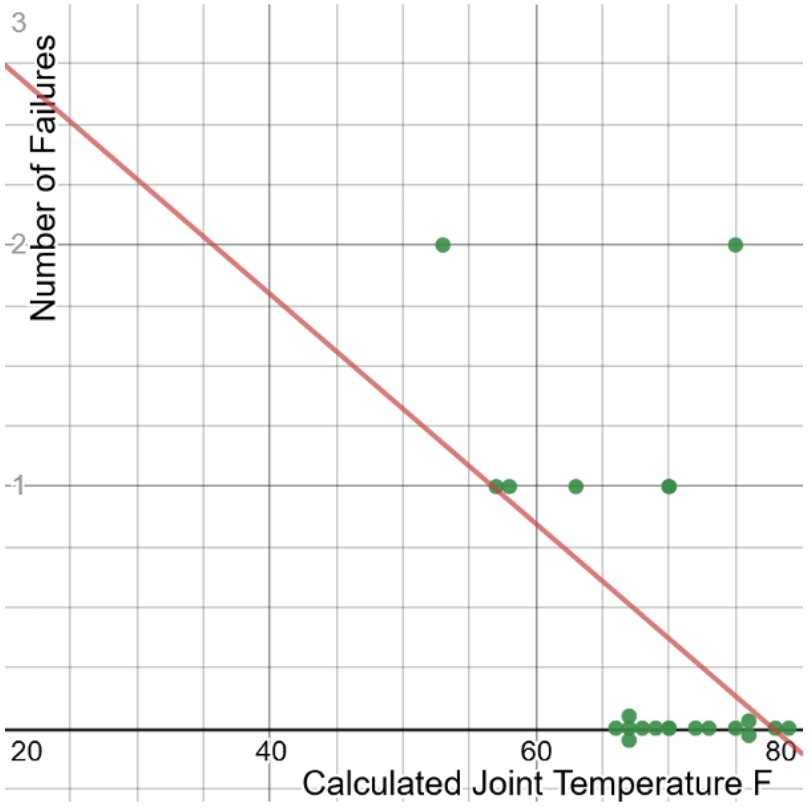
Data Source: http://www.math.usu.edu/~symanzik/teaching/2009_stat6560/RDataAndScripts/sharif_abbass_project1_challenger.csv

Exercise: “There’s No Correlation”

The data scientist tries to apply a linear regression here, and while it does show a trend it is a little awkward especially since our data is sparse.

Should we transform our data somehow?

temperature	o_ring_failures
53	2
57	1
58	1
63	1
66	0
67	0
67	0
67	0
68	0
69	0
70	1
70	0
70	1
70	0
72	0
73	0
75	0
75	2
76	0
76	0
78	0
79	0
81	0



Data Source: http://www.math.usu.edu/~symanzik/teaching/2009_stat6560/RDataAndScripts/sharif_abbass_project1_challenger.csv

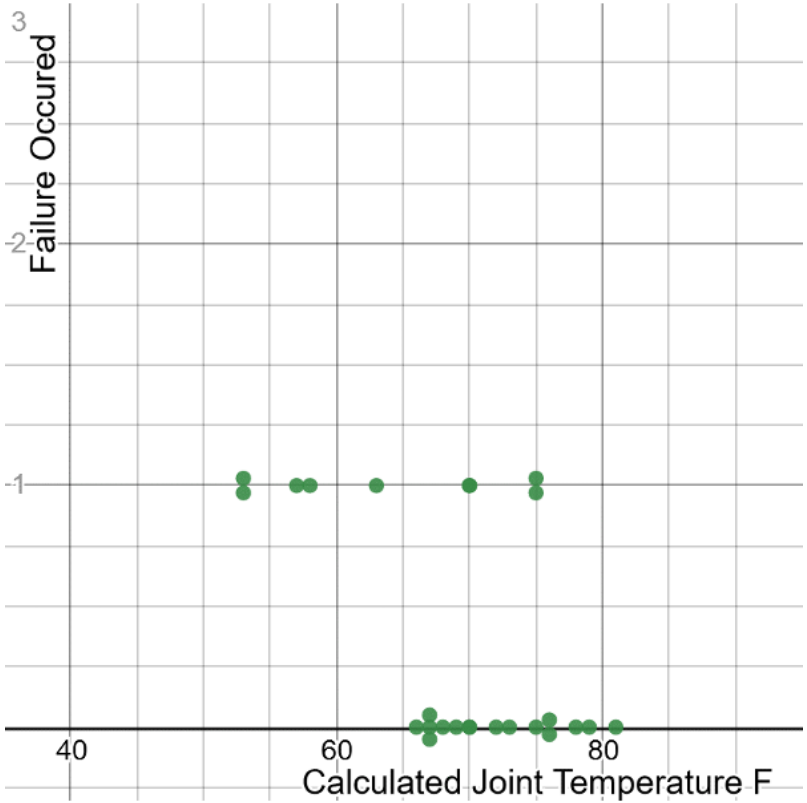
Exercise: “There’s No Correlation”

What if we converted the data to be binary, showing whether a failure occurred or not occurred, by separating each instance into its own record?

This reduces the domain of output variables to “0” and “1” creating a binary model.

Is a story now becoming clear? What model can we use to predict probability of failure at a given temperature?

temperature	o_ring_failures
53	1
53	1
57	1
58	1
63	1
66	0
67	0
67	0
67	0
68	0
69	0
70	1
70	0
70	1
70	0
72	0
73	0
75	0
75	1
75	1
76	0
76	0
78	0
79	0
81	0



Data Source: http://www.math.usu.edu/~symanzik/teaching/2009_stat6560/RDataAndScripts/sharif_abbass_project1_challenger.csv

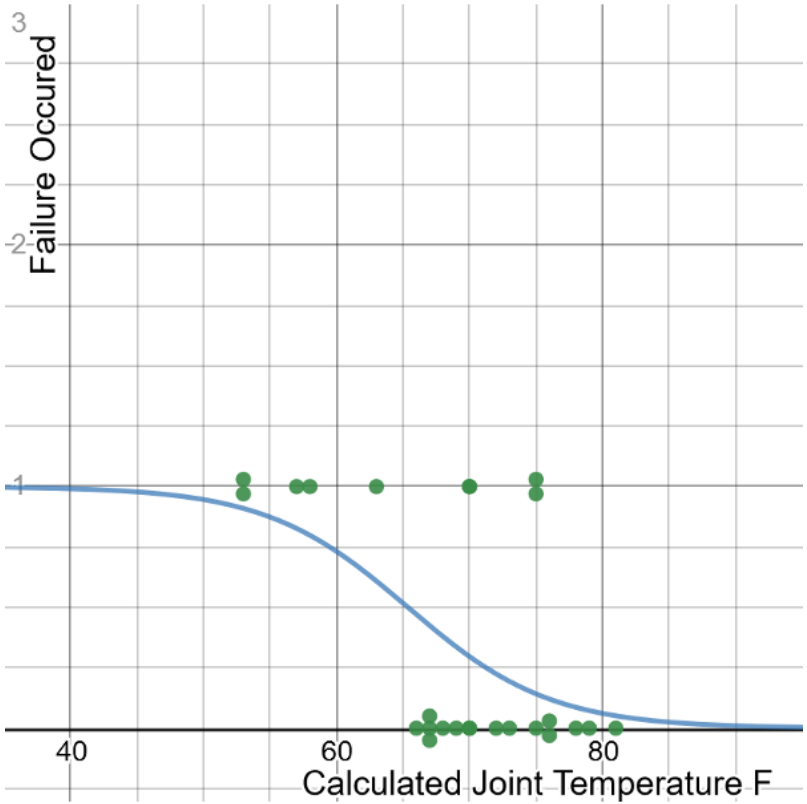
Exercise: “There’s No Correlation”

Logistic regression might be the best way to model this risk.

Even though we lack freezing temperature data, the logistic regression points to a high probability of risk for O-ring failure.

If our launch is going to happen in freezing temperatures, this does not bode well.

temperature	o_ring_failures
53	1
53	1
57	1
58	1
63	1
66	0
67	0
67	0
67	0
68	0
69	0
70	1
70	0
70	1
70	0
72	0
73	0
75	0
75	1
75	1
76	0
76	0
78	0
79	0
81	0



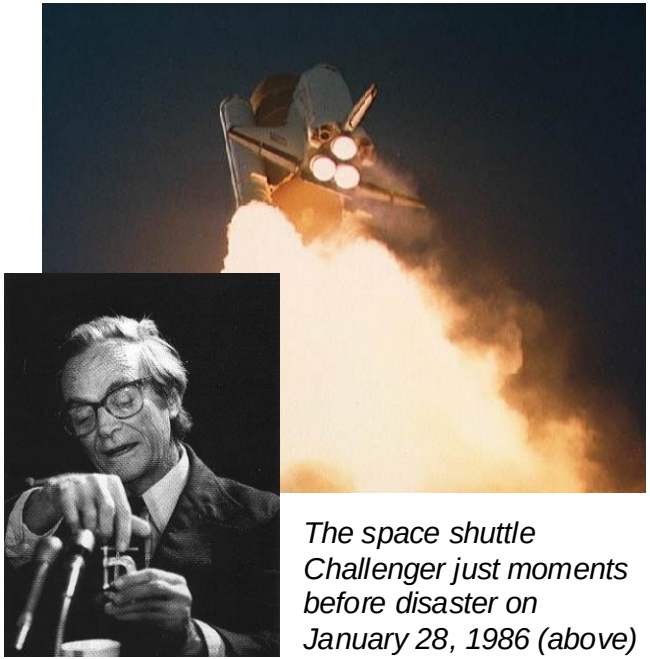
Data Source: http://www.math.usu.edu/~symanzik/teaching/2009_stat6560/RDataAndScripts/sharif_abbass_project1_challenger.csv

Exercise: “There’s No Correlation”

This is exactly what happened to the space shuttle Challenger on January 28, 1986, and if you already have not figured out already, we are doing the analysis.

Through a series of unfortunate events, only partial data was accessible and omitted non-failure data, which showed a correlation with temperature and O-ring failure.

The analysis we just did should have happened before the accident, but unfortunately it occurred afterwards.



The space shuttle Challenger just moments before disaster on January 28, 1986 (above) and Richard Feynman famously demonstrating O-ring failure with a glass of ice water (left).

Exercise: “There’s No Correlation”

What did we learn from this lesson?

You can’t take data at face value.

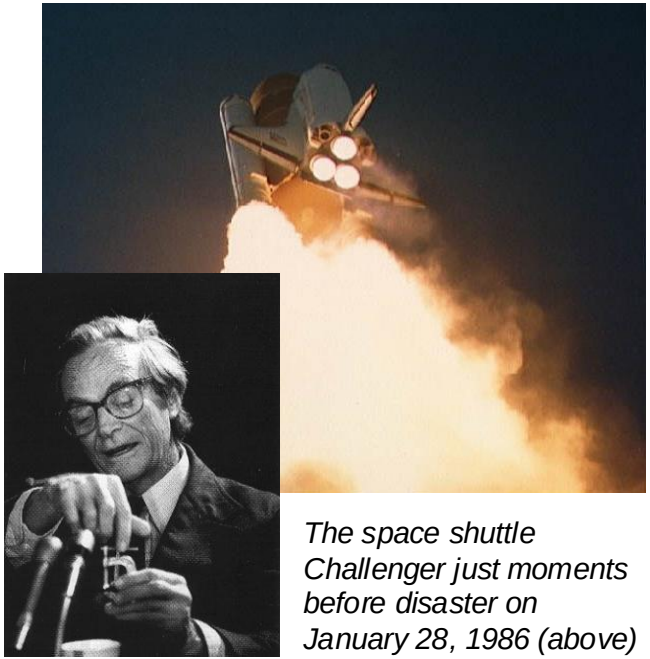
Consider the source of the data, think carefully what created it.

Look for biases and imbalances in the data that corrupted it.

Data must go through transformations to become meaningful, and you have to consider these transformations carefully.

Data is not the source of truth, but rather clues to the truth.

Be analysis-driven, not data-driven!



The space shuttle Challenger just moments before disaster on January 28, 1986 (above) and Richard Feynman famously demonstrating O-ring failure with a glass of ice water (left).

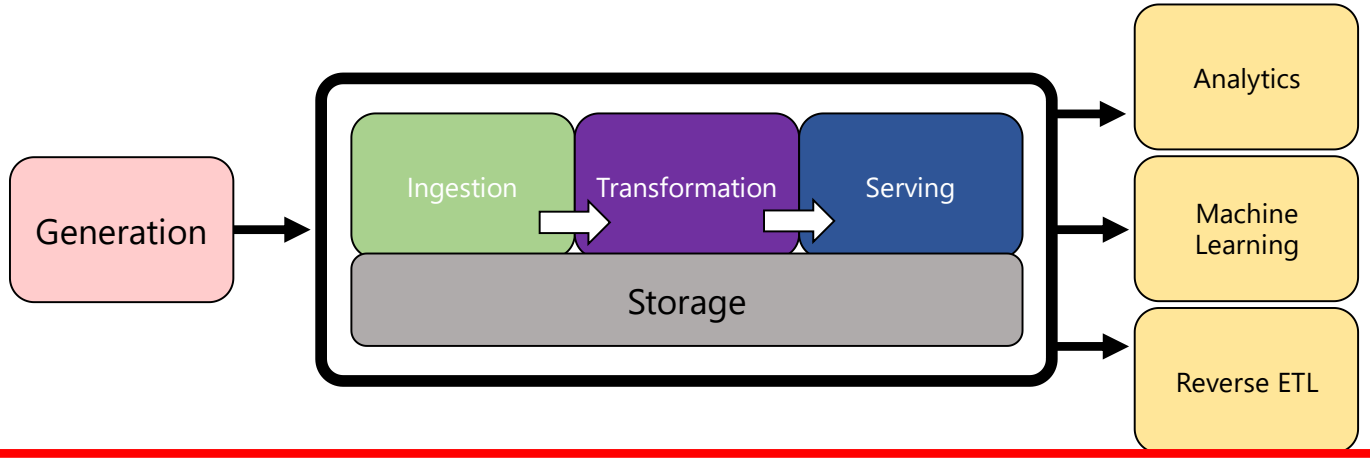
Data Engineering Fundamentals

Day 3

Major Undercurrents

Activities that Exist at All Stages of the Data Engineering Lifecycle

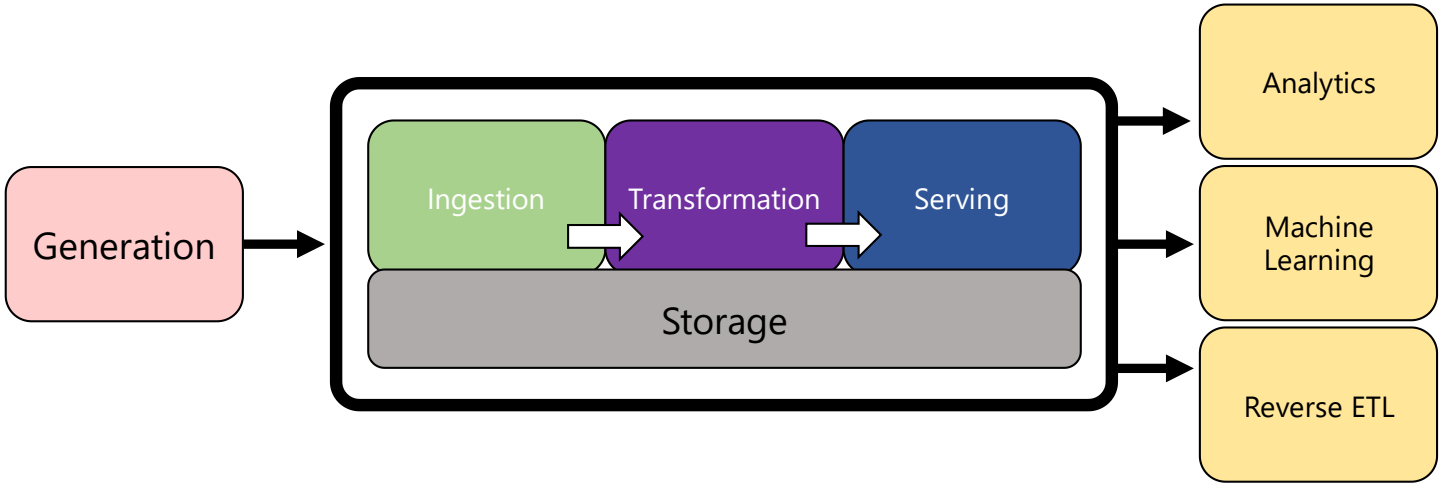
Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Undercurrents: Security

A Cybersecurity Story

No Names Named

A Cybersecurity Story

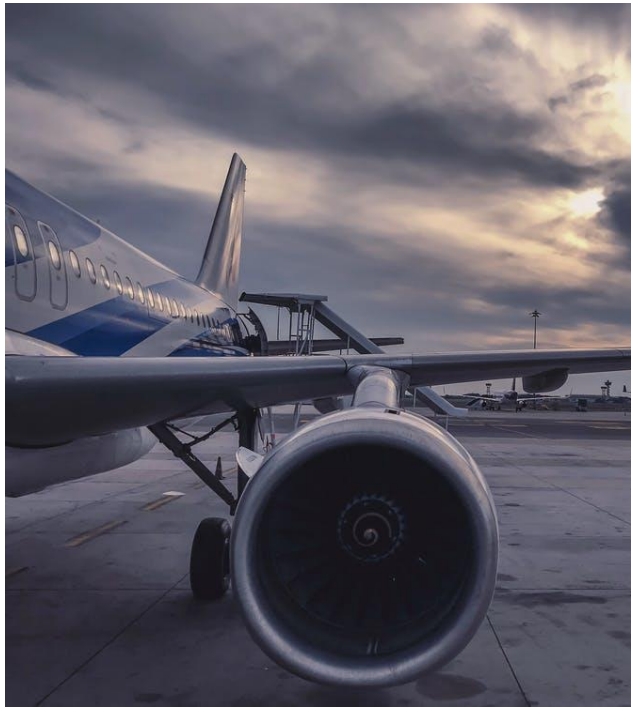
No Names Named

An airline with a global presence and large workforce was experiencing a recurring security breach in its inflight trip management system.

The internal employee portal was used to trade trips between crew members, and internal bad actors were taking advantage of this unknown cyber vulnerability.

- This vulnerability allowed one inflight member to steal another coworker's trips in an unauthorized manner, while making the swap look legitimate.

- Some bad actors allegedly created bots and web scrapers to automate thefts, monopolize high-paying trips, and sell them for a fee.



A Cybersecurity Story

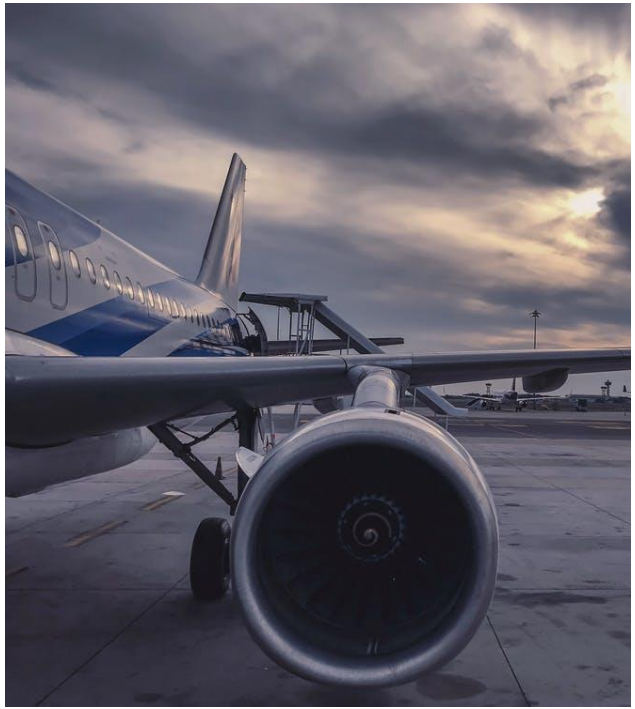
No Names Named

- Private social media inflight groups posted hundreds of members complaining about thefts they experienced, sharing stories of their interactions with the alleged cartel, and how some were offered access to the backdoor for a fee.

Middle management dismissed the allegation of ”hacker flight attendants” as uncredible rumors.

Morale plummeted as thefts accelerated, and hundreds of internal social media posts from inflight groups somehow failed to get attention from leadership.

Finally, cybersecurity got involved and confirmed there was an issue.



A Cybersecurity Story

Here Come the “Experts”

After several years, the cybersecurity team was caught in a whirlwind of legal fights with the suspects, whose lawyers said they had no hard evidence.

- The cybersecurity team ***SPENT YEARS*** trying to find how the hack was done but with no success, and bad actors kept coming and going without charges sticking.
- Since the logs showed the victims “authorized” the trip trades in question, there was no hard forensic evidence of fraudulent activity.
- The authentication framework was modernized, 2FA was introduced, many cybersecurity consultancies were brought in, but that did not stop the breaches.

A Cybersecurity Story

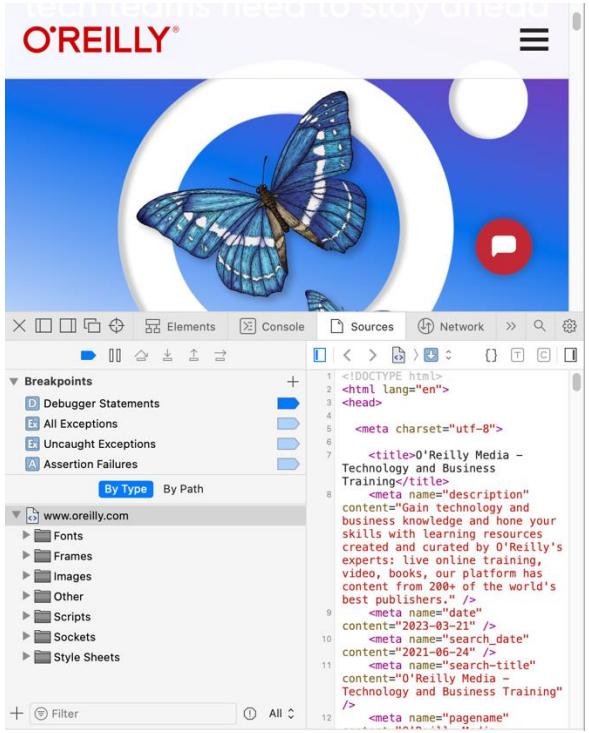
Here Come the “Experts”

Finally, an outside consultant (the instructor) was brought in by a frustrated inflight department, and he was able to discover and replicate the hack in 5 minutes.

- **The smoking gun:** the web page’s source code had links that effectively hijacked the victim party’s identity when on their page.

- Because of this authentication loophole, the bad actor could now perform actions as the victim and transfer the victim’s trips to themselves.

NOTE FROM INSTRUCTOR: I am not sharing this to self-extol or humblebrag, but simply show how an outsider thought differently, asking simple and practical questions that the established experts were blind to.



A web page can have its HTML/JavaScript source code accessed in any browser.

This can be an attack surface if not developed correctly.

A Cybersecurity Story

How Did this Happen?

What conditions allowed such a security breach happen?

- The trip trade system was over 15 years old, and since it is a cost center (and does not make revenue) it was minimally maintained and upgraded.
- The portal was designed with loose security standards as it was for employees only, and not customer-facing.
- The portal did not handle sensitive information like personal information or credit card numbers, and therefore was not deemed a security risk.
- Cybersecurity was focused more on external threats, not internal, and developed tunnel vision on protecting customer information (like credit cards) as well as company data.

A Cybersecurity Story

How Did this Happen?

Still how could a professional cybersecurity team *SPEND YEARS* investigating a vulnerability, hire penetration testers and cybersecurity third parties, but still have no results?
And an outsider party finds the vulnerability in 5 minutes?

- The cybersecurity team focused on conventional attack vectors, like login authentication and black hat tools.
- They failed to put themselves in the attacker's shoes, and ask "if I was a flight attendant with no technical background, but enough curiosity and drive, how would I breach the system?"
- Blind spots were created by following convention and orthodoxy, rather than realizing the unorthodox nature of the entire breach.

What went wrong? What caused this incident?

A Cybersecurity Story

Lessons Learned

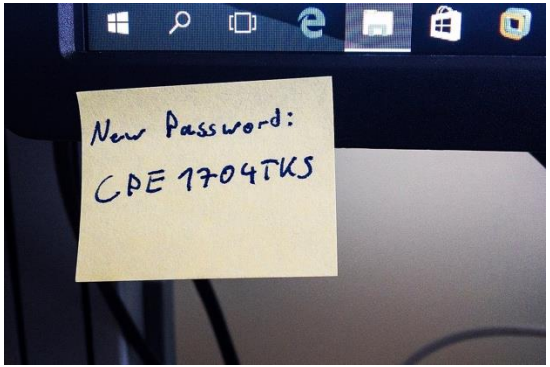
Cybersecurity is just as much (if not more) about people than technology.

- *People* inevitably will be biased, lazy, bored, distracted, trusting, deceived, and complacent creating perfect situations for exploits.
- There will never be a 100% secure system, because *people* will be using the system.

ANY **software regardless of its nature, who it is made for, and how walled off it is can be an attack surface!**

Orthodoxy, security standards, manuals, handbooks, and other conventional tools can create *security theater* that gives an illusion of security and nothing more.

Security is a cultural issue, not a technical one!



A Cybersecurity Story

Lessons Learned

The trip trade system was completely internal and made for a seemingly benign function: trading trips.

The system did not touch credit card numbers, banking information, or any sensitive information.

But it did have a devastating impact when breached: people's livelihoods (and their income) were stolen on a systematic scale.

The worst breaches are the unforeseen kind, and challenge expert and orthodox views with a never-ending game of cat-and-mouse.

You can perform security theater all day and be compliant with some standard, but that will do nothing against real and creative threats which can only be stopped by a security culture.

Courtesy: Paramount Pictures

A Cybersecurity Story

What Does this Have to do with Data Engineering?

EVERYTHING!

Data is the most critical and vulnerable asset an organization has, and security theater is not going to protect you.

You need security culture, not security theater.

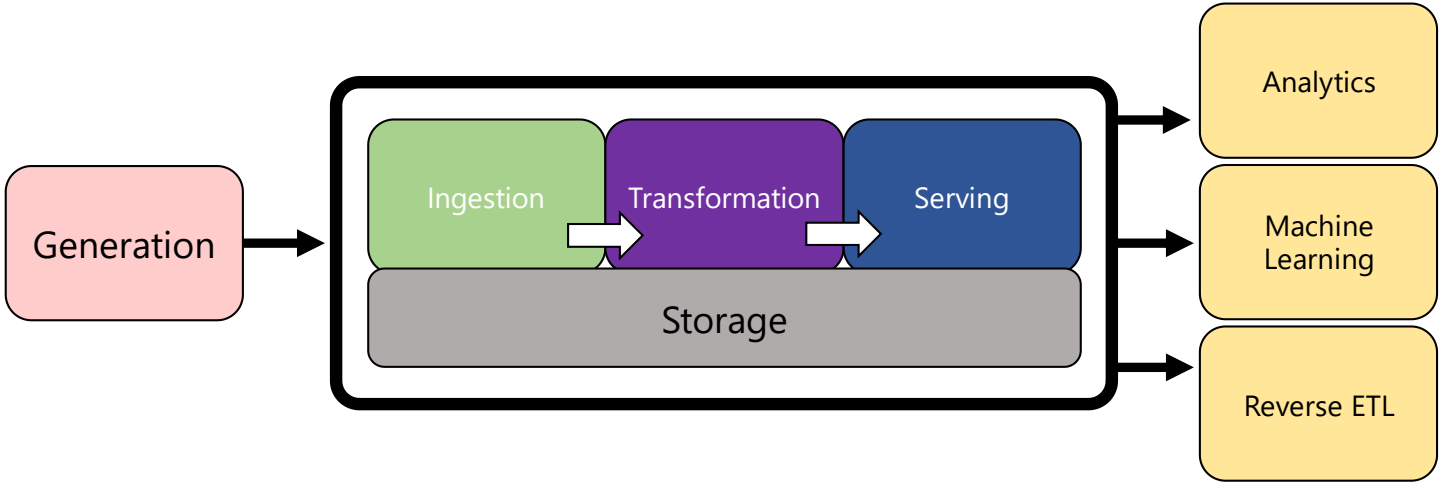
Some data may seem innocuous and not contain personal information (after all, these were just crew trips) but even innocuous data can be used in creative and destructive ways to enable theft.

As a data engineer, you will be creating many API's and if you don't authenticate and secure them properly… yikes, incidents like this one will happen.

You have a responsibility to take security culture seriously and apply it to every data activity you do.

Undercurrents: Data Management

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Data Management

Data management is the development, execution, and supervision of plans, policies, programs, and practices that deliver, control, protect, and enhance the value of data and information assets throughout their lifecycle.

- The Data Management Association International (DAMA)

Data Management

Data management naturally (perhaps redundantly) fits with data engineering, embracing principles for handling and optimizing data.

- Data governance
- Modeling and design
- Storage and encryption
- Ethics and privacy
- Integration and operation

We will talk about a few of these points.

Data Governance

“Data governance is, first and foremost, a data management function to ensure the quality, integrity, security, and usability of the data collected by an organization.”

- Data Governance: The Definitive Guide

Data governance is primarily about discoverability, security, and accountability.

Without data governance, you can run into a wide spectrum of issues from unreliable data to data breaches.

Data Governance: Discoverability

For a company to effectively use data and extract value, the data has to be discoverable.

Users should be able to find data, know where it came from, how it ties to other data, and what it means with full context.

This can be at odds with security that makes data inaccessible, so this has to be weighed accordingly or at least have documentation publicized so data access can be requested.

Data can be automatically documented or compiled in a Wiki sometimes, and this should be leveraged where possible.

Data Governance: Security

Security should be one of the first concerns for data engineering.

Principle of least privilege should always be weighed, giving people the least amount of access to do their job possible, for the shortest period possible.

Granted, this can create a dysfunctional amount of bureaucracy when carried away so be mindful of that.

Do not give admin access to all users, and give write access with discretion.

Become a competent security administrator knowing identity access management (IAM), roles, groups, password policy, and encryption policy.

Most importantly advocate a security culture and train all stakeholders in distrust.

Data Governance: Accountability

Data accountability means giving an individual ownership and stewardship of the data.

Without an owner, the maintenance and oversight of the data will cause it to be neglected.

Quality can then quickly become questionable.

The accountable party does not have to be a data engineer and can be a domain expert, a software engineer, or product manager who can coordinate with data engineering.

Data Governance: Quality

"Can I trust this data?" – Everyone

According to *Data Governance: The Definitive Guide*, data quality is defined by three things:

- Accuracy – Is the data factual and represent ground truth?
- Completeness – Is the data complete and not missing any records or values?
- Timeliness – Is the data up-to-date and available in a timely fashion?

Data Governance: Data Lineage

How do we track data as it gets passed around and transformed? This audit trail is what we call **data lineage**.

This audit trail is helpful with tracking errors, maintaining accountability, and debugging data and the systems it passes through.

This becomes especially important when subject to privacy compliance and regulations like GDPR and Sarbanes-Oxley (SOX).

Andy Petrella's Data Observability Driven Development is also a resource: https://www.kensu.io/blog/a-guide-to-understanding-data-observability-driven-development.

Data Governance: Data Integration

How do we integrate data sources across tools, applications, and processes?

Typically, general-purpose web API's are used nowadays rather than direct database connections.

These API's can be achieved with simple Python applications built with libraries like Flask or Django, and these will handle querying the databases internally but interface with external applications via a web service API.

While this simplifies data connectivity greatly, the number of these services has exploded and creates complexity in that volume.

Data Governance: Ethics and Privacy

As we have learned in the black hat story, data affects people.

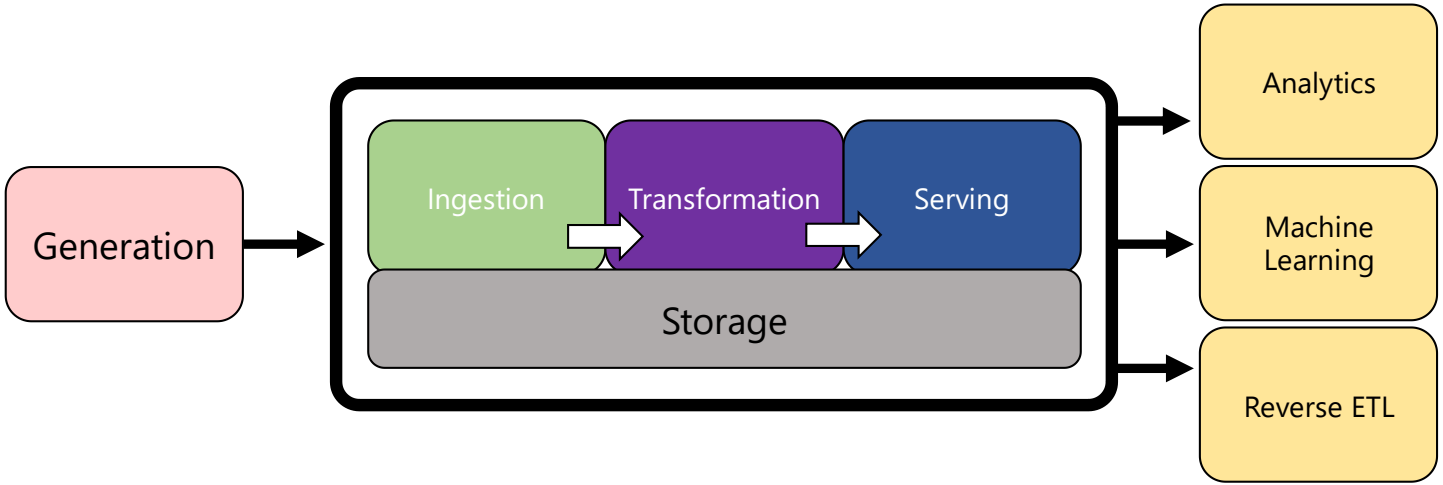
Be wary of personally identifiable information, and consider how even seemingly innocuous data can be used for harm.

Be concerned when the data is being used to make predictions on people, and consider how bias will inevitably creep into the system.

Be compliant with data regulations like GDPR and CCPA, which are only going to get more stringent and increase in penalty.

Undercurrents: DataOps

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

DataOps

DataOps is an application of DevOps principles, which emphasizes continuous and closely coordinated delivery with production.

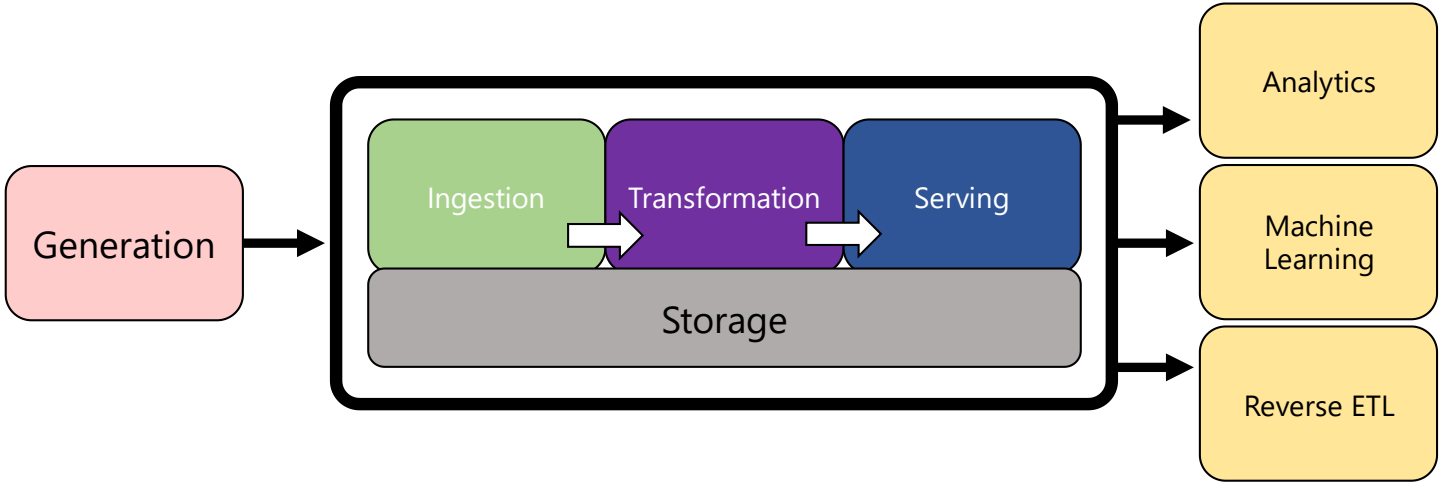
DataOps is a work in progress, but generally should strive to embrace these principles.

- Rapid innovation and experimentation
- Striving for high data quality and minimal error
- Collaboration with stakeholders and cross-functional teams
- Clear measurement and monitoring of results

Automation is an ideal for a lot of DataOps, but it will take time before we can achieve that level of efficiency and standardization of tools.

Undercurrents: Data Architecture

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Data Architecture

Data architecture is not just about a strong design, but also planning for it to evolve and change.

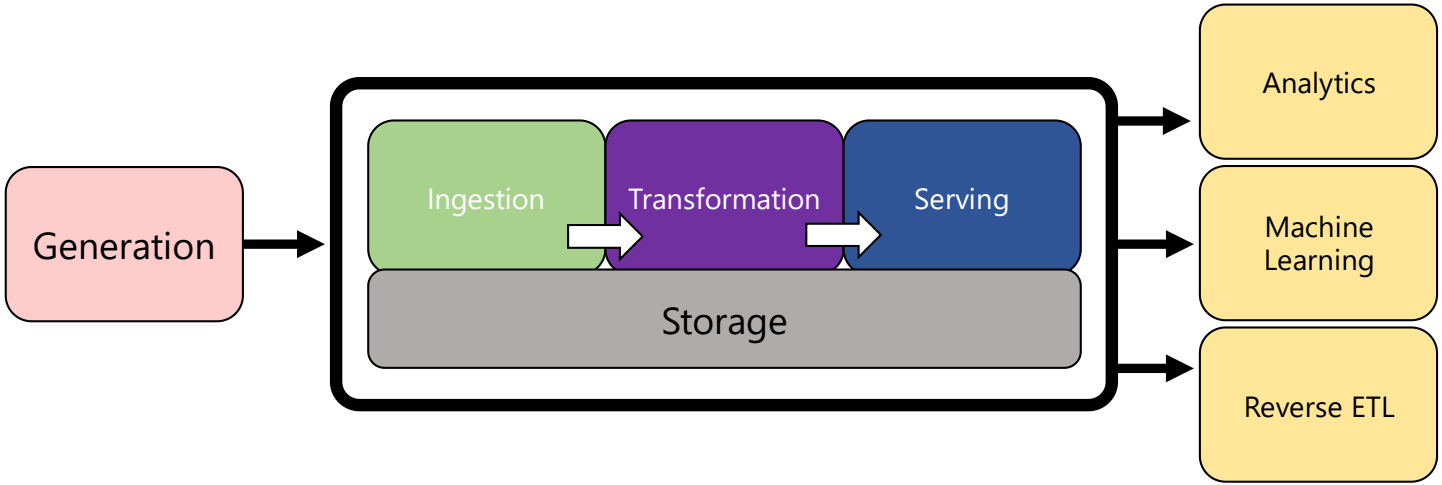
It is easier to add to a data architecture, but modifying what already exists is much harder.

Learn how to ask questions and clarify business requirements and know how to effectively normalize business entities into the needed datasets.

A data engineer is not necessarily a data architect but should understand the principles so they are able to work closely and effectively with one.

Undercurrents: Orchestration

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Orchestration

Orchestration is the coordinating and timing of several jobs to run as efficiently and quickly as possible on a schedule.

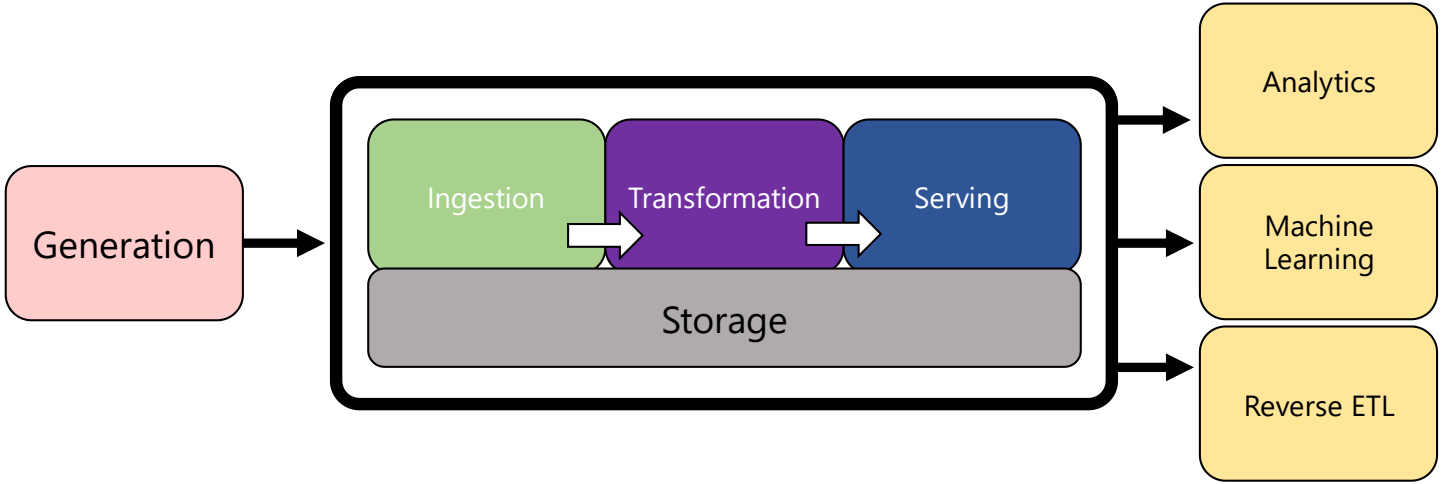
Think of a symphony orchestra, with different instruments playing on different timings, but all taking turns to fill their section of the musical piece.

Data orchestration is very much like that, where one process precedes another and dependencies come together, while no idle or underutilized moments exists for efficiency.

As a data engineer, anticipate having to work with tools to monitor and kick off activities and processes, such as directed acyclic graph (DAG) which don't just schedule on timings (like cron jobs), but execute jobs in a sequential and dependent manner.

Undercurrents: Software Engineering

Data Engineering Lifecycle (Reis and Housley)



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Software Engineering

While there are cloud services and nicely abstracted tools to minimize the amount of Java or C++ you must write, it is still important to have software engineering skills.

Being able to write unit tests and regression tests, as well as debug error messages coming from systems written in Java, more or less require some familiarity with these programming platforms.

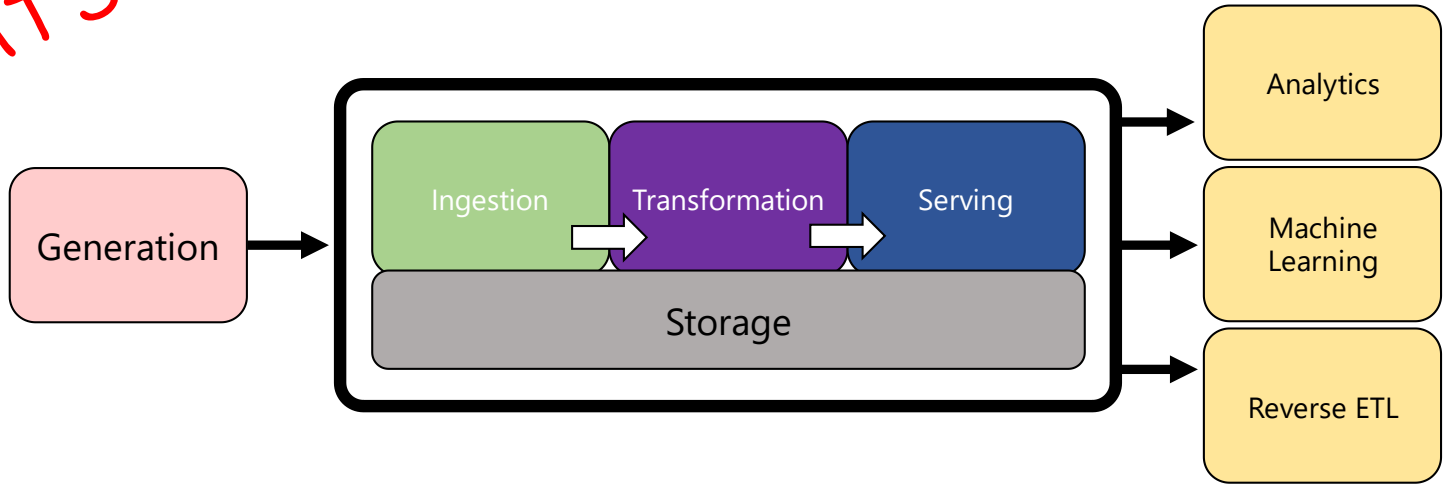
To tie together different systems you may find yourself writing code in Python, Java, or C++ to orchestrate the glue code for these systems.

You may also discover platforms that allow dropping in custom code.

Finally, there is power in being able to build a piece of a system from scratch when no existing solution exists.

Data Engineering Lifecycle (Reis and Housley)

THAT'S IT!



Undercurrents:

Security | Data Management | DataOps | Data Architecture | Orchestration | Software engineering

Case Study

“Self-Driving” Cars

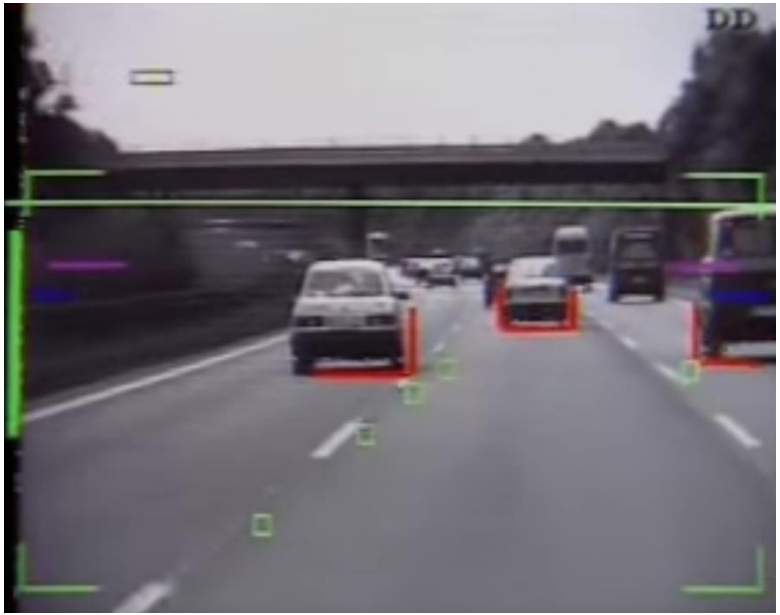
Defining ADS and ADAS

The Society of Automotive Engineers (SAE) refers to any dynamic driving automation as an *advanced driving system (ADS)* or *advanced driving assistance system (ADAS)*.

An ADS or ADAS does not exclusively refer to a fully “self-driving” vehicle, but a vehicle with any automation that dynamically reacts to its environment.

The public may refer to these systems as self-driving cars or autonomous vehicles, but these terms should be avoided because they cause confusion and misperception.

ADS systems operate in non-deterministic environments, making it difficult for them to react to events, much more predict events.



Project Prometheus environment display during demonstration in 1994.

https://www.youtube.com/watch?v=I39sxwYKlEE

A Great Data Engineering Example

ADS and ADAS vehicles (“self-driving”) are great showcases of data engineering.

You have different sensors collecting very different types of data, and it is not straightforward how this data is stitched together.

These data also come in a variety of formats and must go through several transformations.

Biases and outliers in the operating domain are highly problematic, and highlight the importance of define what environmentally creates the data.

Camera

Cameras are a key input in ambitious ADS systems, streaming picture frames often for object identification/classification.

The key input from cameras are colored pixels, which is necessary for stop lights and other color-coded symbols on roads.

Plain monocular cameras are cheap, but they are sensitive to weather and illumination, and do not provide depth data.

Thermal cameras (heat signatures), stereo cameras (depth and range), panoramic cameras (360° views), and event cameras (movements) provide ways to get more utility and data about the environment.

Weather and illumination will always need to be compensated with other instruments.

A Tesla owner claimed he broke into the Autopilot computer accessing image feeds, object identification, and classification in a Paris drive.

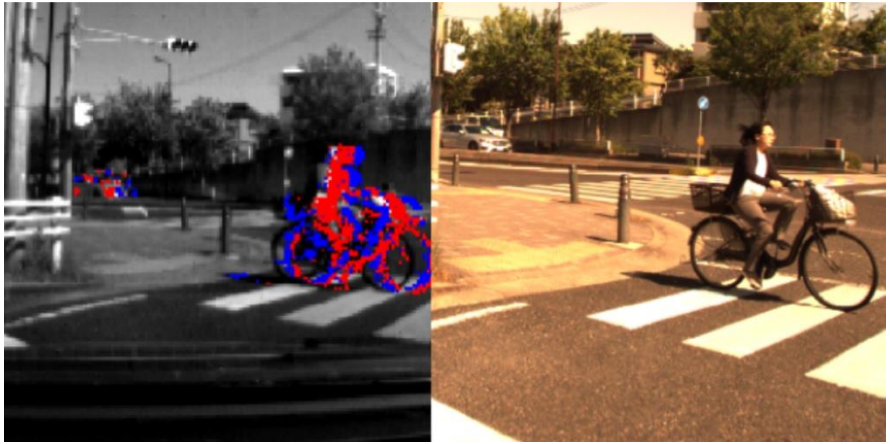
SOURCE:
https://teslamotorsclub.com/tmc/threads/seeing-the-world-in-autopilot-part-deux.129790/

Event Cameras

A special type of camera to bring attention to are event cameras, which capture events through stimulated pixels.

This hardware has limited resolution but could provide a valuable sensor input to perceive activity and movement in an environment.

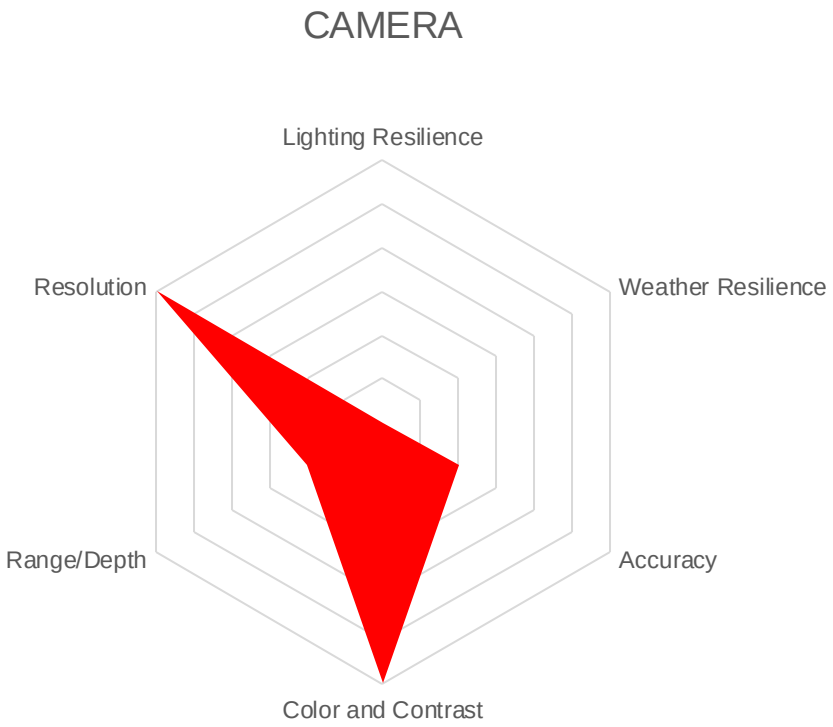
To the right shows a cyclist being identified due to its movement, and activated pixels are shaded in blue and red.

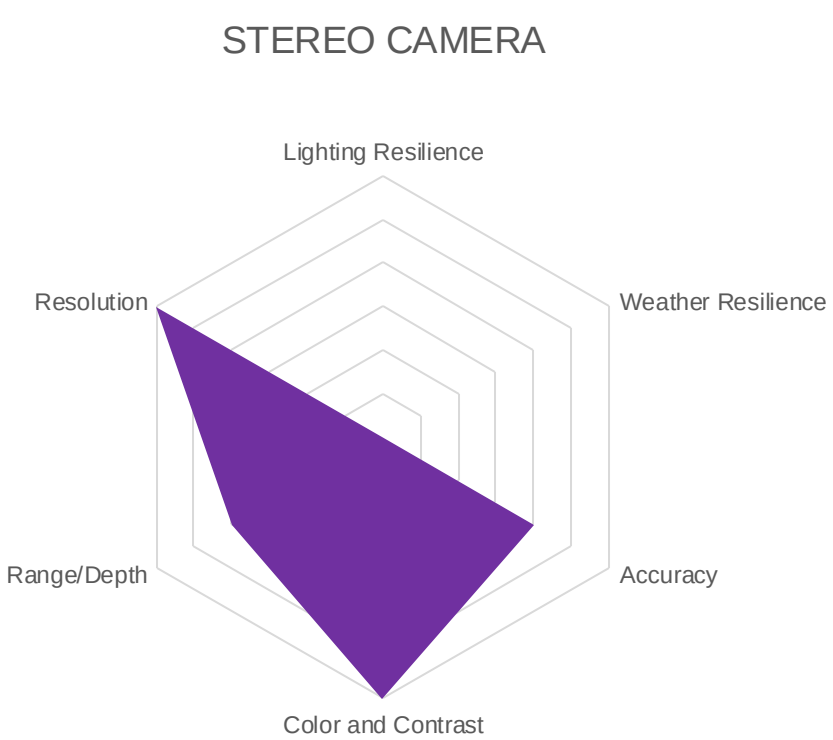


An event camera has potential to perceive movement and activity in an environment.

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Camera Strengths/Weaknesses





Radar and Ultrasonic

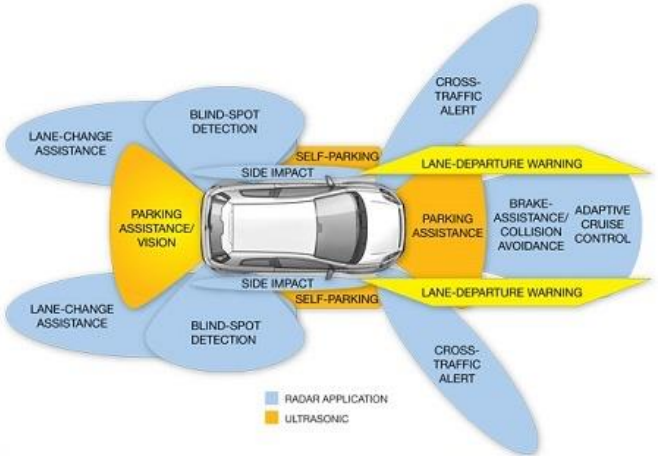
Radar emits electromagnetic waves to detect surrounding traffic and obstacles.

Cost-effective and lightweight, it can be used liberally to 3D map its environment without illumination concerns.

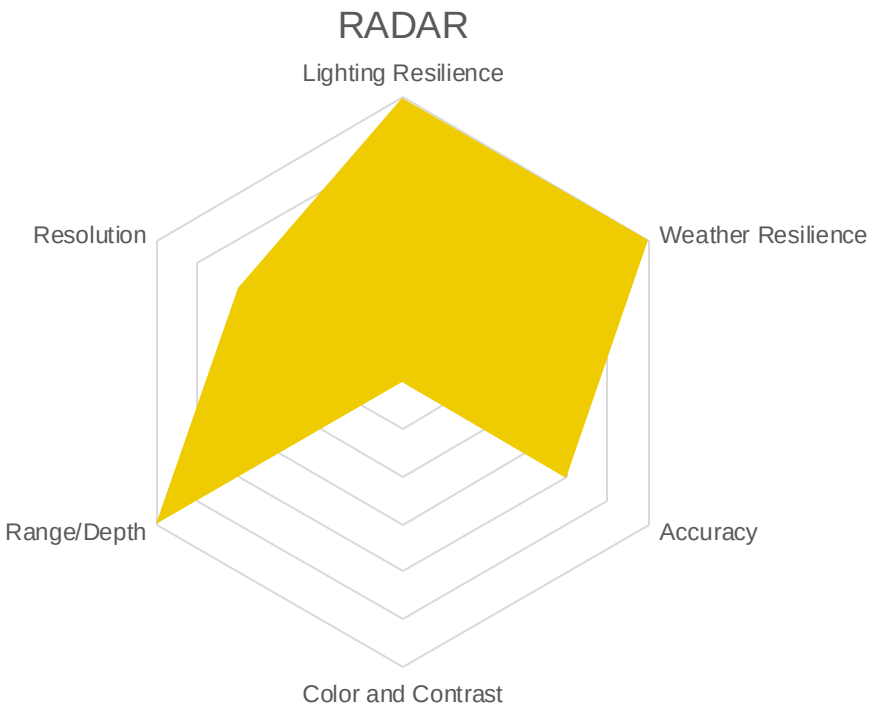
It has greater range than LIDAR, and cheaply enables many ADS functionalities like lane assist, adaptive cruise control, and emergency braking.

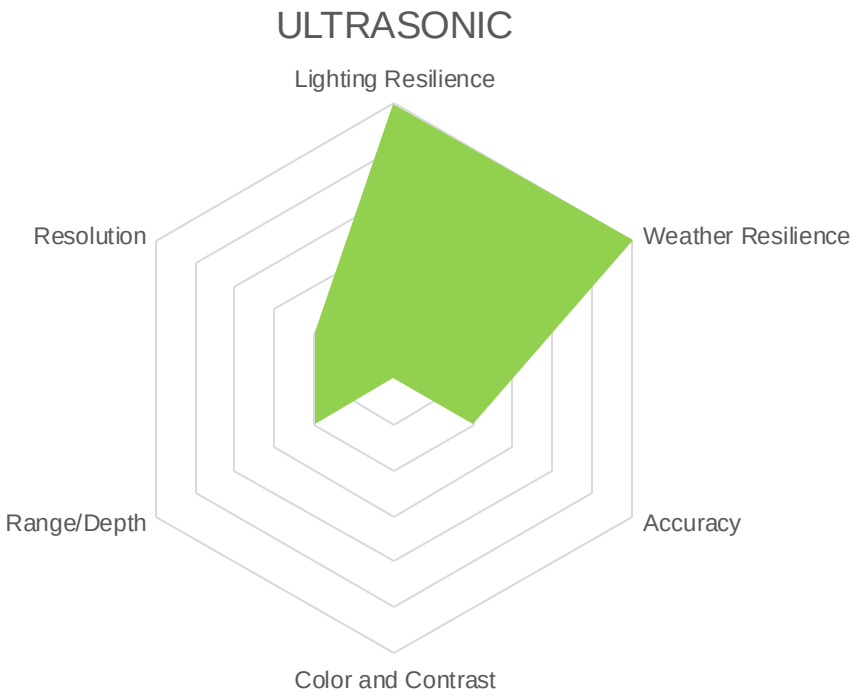
However, it actively emits radio waves which can cause interference.

Ultrasonic operates similarly to radar but uses sound waves instead of radio waves and typically used for close-range detection.



Radar and Ultrasonic Strengths/Weaknesses





LIDAR

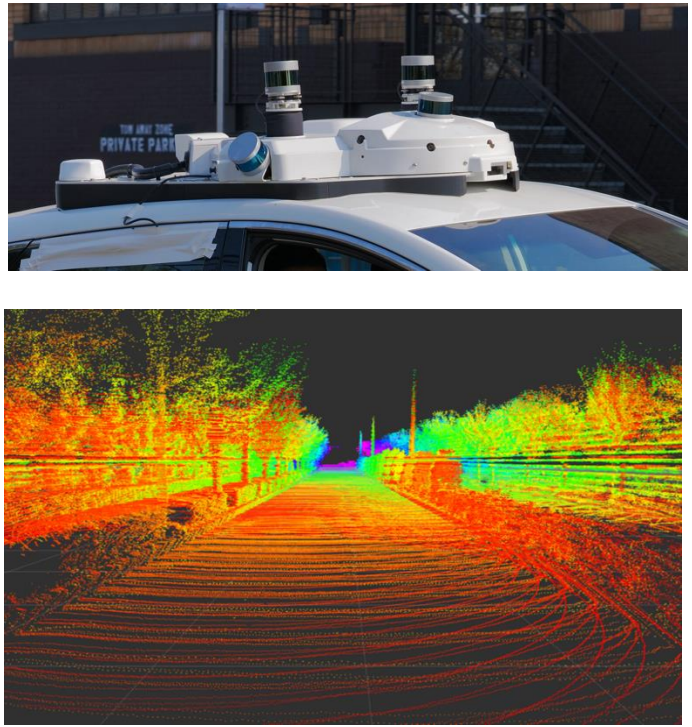
LIDAR is the most expensive and bulky sensor, but it enables a high resolution "3D Point Cloud" with moderate range.

A 3D point cloud is a 3D mapping of the environment generated by infrared lights (lasers), which will reflect off surfaces to calculate distance.

Merged with image data, colors can be applied to the 3D point cloud bringing more detail and dimensions to the mapping.

Many believe LIDAR is a key technology to unlocking effective ADS, and with cost and size coming down it may be more commonplace in the future.

Alphabet's Waymo makes heavy use of LIDAR, and the Audi A8 is the only consumer car that leverages it.



A set of LIDAR sensors on a car (top) and a LIDAR-generated 3D point cloud of a street.

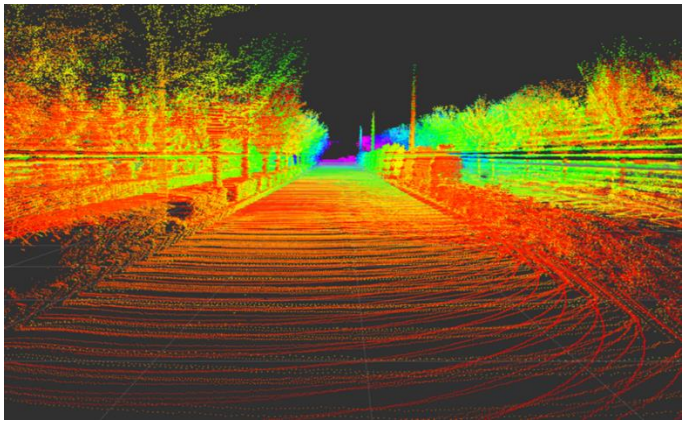
SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

LIDAR

While LIDAR has many advantages, it also has some shortcomings:

It will bounce points off irrelevant objects, which can create noise that has to be cleaned.

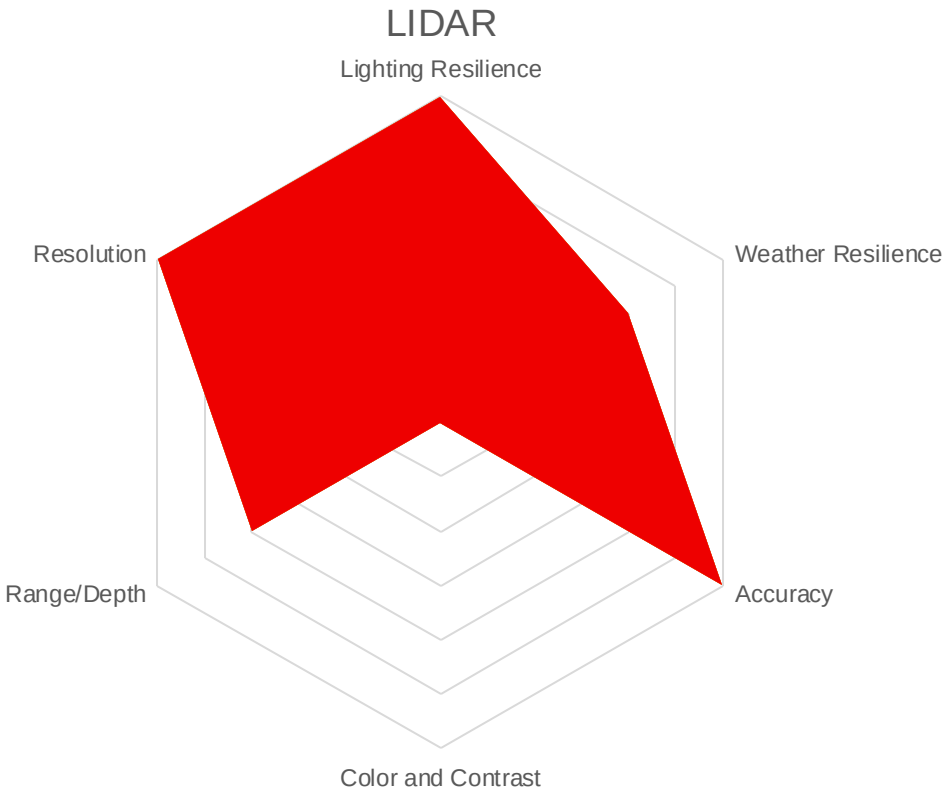
Precipitation like rain and snow will degrade performance, the latter can completely alter an environment's appearance and shape.



A set of LIDAR sensors on a car (top) and a LIDAR-generated 3D point cloud of a street.

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

LIDAR Strengths/Weaknesses



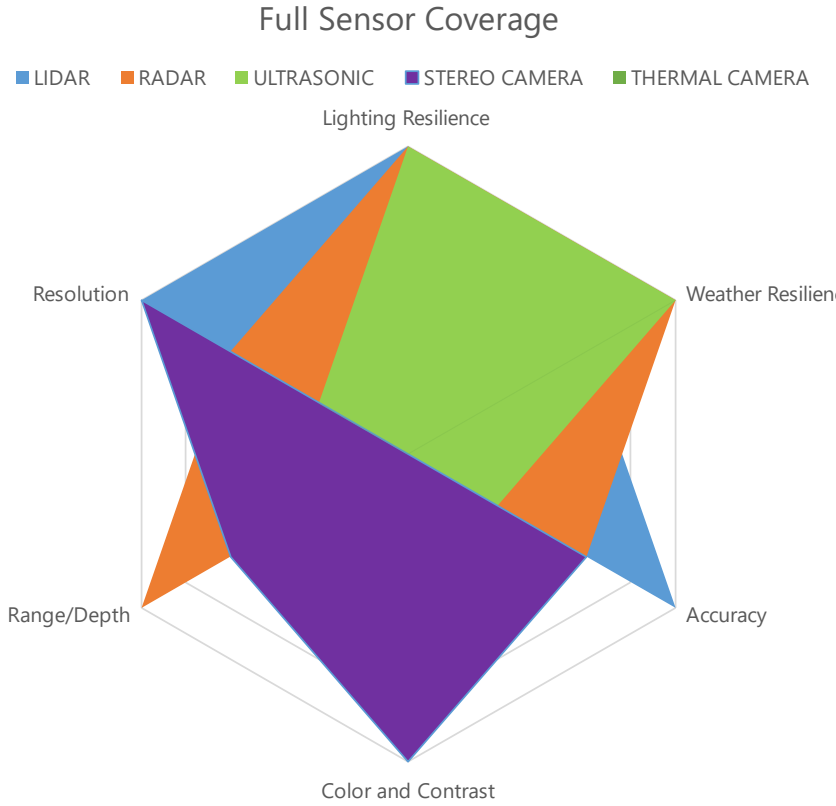
Combining for Coverage

These sensors individually have shortcomings but can contribute their strengths together for robust coverage of the environment.

Have we solved self-driving? Is data engineering done? Why or why not?

This is not an easy data engineering task, as all this sensory data (along with GPS data) must be stitched together so the environment can be modelled meaningfully and accurately.

Calibrating the sensors must be done carefully as well so they operate in tandem (e.g. cameras + LIDAR), and changes to configuration can deprecate previous data and machine learning training.

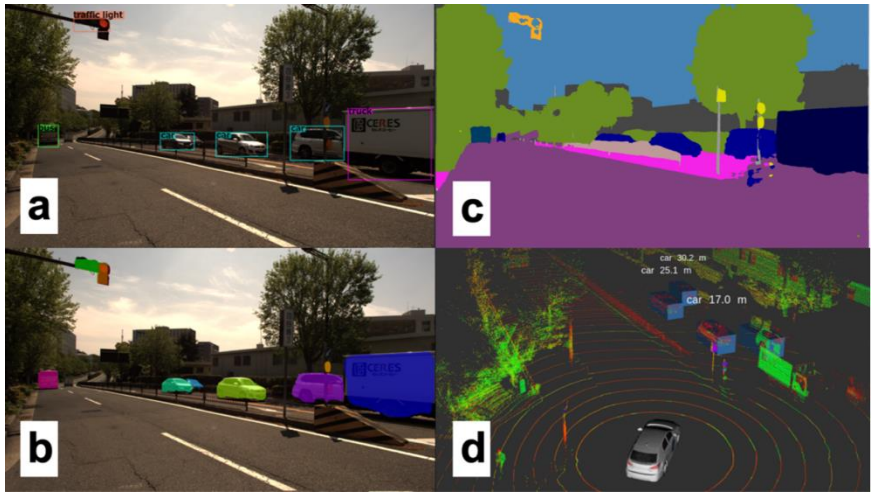


What Effect Does Hardware Calibration Have?

What happens when you calibrate or upgrade a camera? Change filters? Think carefully from a data engineering perspective!

Creating a 360° model of the vehicle's environment is challenging.

- 3D LIDAR point clouds likely need to be colored with 2D camera images.
- Cameras must be calibrated and combined carefully, and even account for mirror reflections.
- Panoramic and fish-eye distortion must be minimized and accounted for.



SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

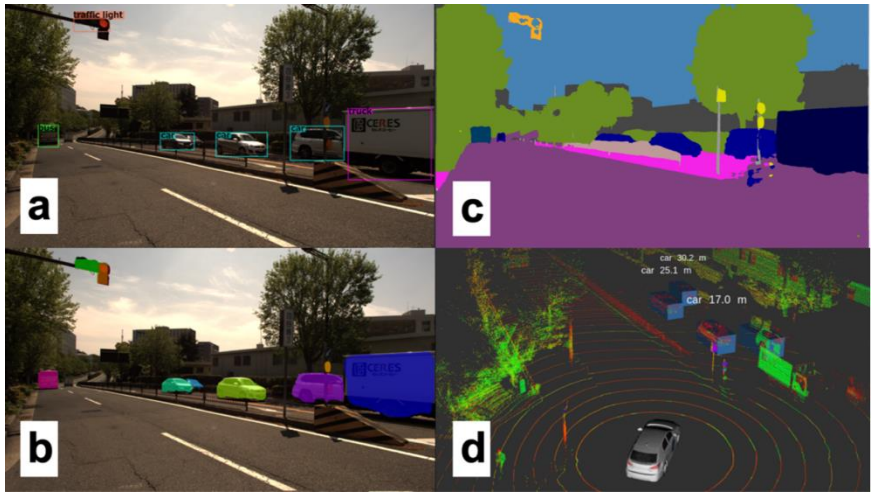
What Effect Does Hardware Calibration Have?

Poor/inconsistent illumination, shadows, weather, seasons, environment changes, bias, and outliers can also derail the model, and are major barriers to mainstream ADS adoption.

Changes to hardware and hardware configurations can rot previous data.

Didn't we spend millions of dollars on acquiring that data? Management may insist we make the previous data work!

Think of what an enormous challenge this is for data engineering.



SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

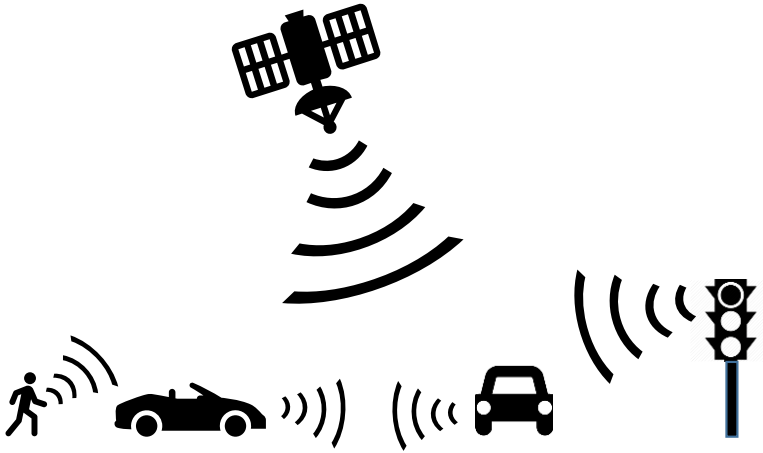
Ego versus Connected Systems

There are many ways to categorize ADS vehicles, but a fundamental distinction is whether the vehicle participates in a connected system.

Ego means the vehicle is always self-sufficient, not relying on external systems to operate and is the most common approach.[1]

Connected means the vehicle must use external systems infrastructure to operate such as GPS, radio beacons for localization, radio-controlled intersections, and even data feeds from other vehicles.[1]

Most state-of-the-art systems are ego and independent, not having to deal with connectivity challenges and perform tasks like emergency braking, lane assist, and parking assist.



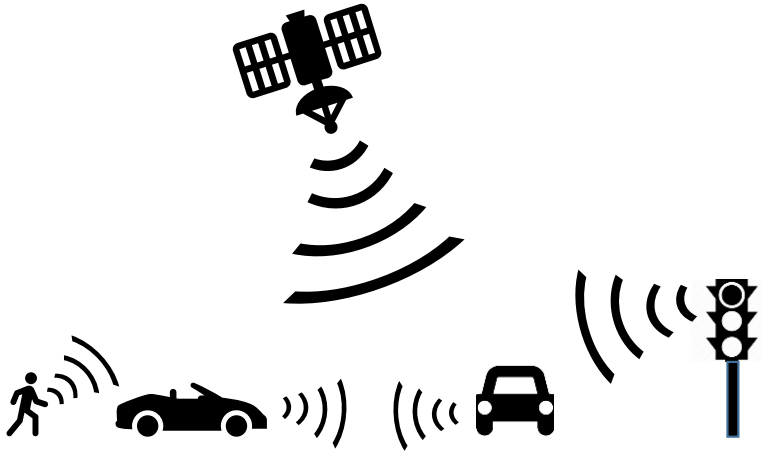
[1] https://ieeexplore.ieee.org/document/6803166

Ego versus Connected Systems

Ego means the vehicle is always self-sufficient, not relying on external systems to operate and is the most common approach.[1]

Connected means the vehicle must use external systems infrastructure to operate such as GPS, radio beacons for localization, radio-controlled intersections, and even data feeds from other vehicles.[1]

From a data engineering perspective, what do you imagine are the advantages/disadvantages of ego versus connected systems?



[1] https://ieeexplore.ieee.org/document/6803166

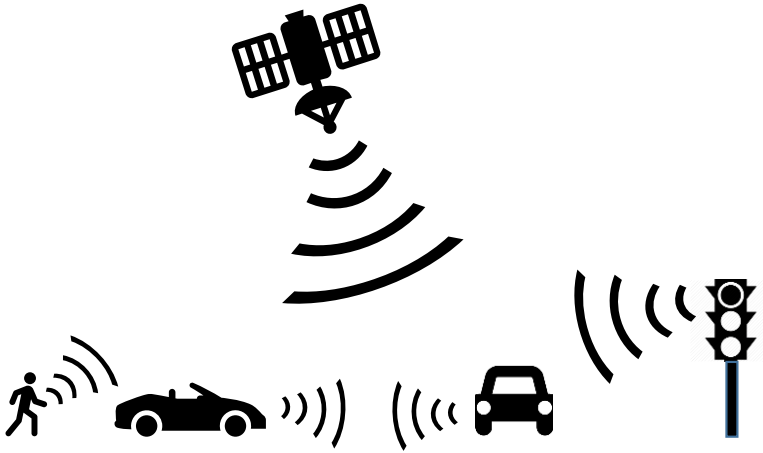
Ego versus Connected Systems

Developing a centralized/decentralized connected infrastructure is a costly initiative, and while ideas exist on nothing concrete has happened yet.

Vehicular Ad hoc NETwork (VANETs) can manage automated driving as agents, and **vehicle to everything (V2X)** paradigms can have vehicles, traffic lights, and even pedestrian mobile devices communicating with each other to collaborate traffic.[1]

VANETs can become a reality through IP networking or information-centric networking, the latter which allows vehicles to broadcast/receive to the immediate area around it rather than a central network.

The challenge is creating a connected system in busy urban areas, where thousands of agents may be active and is hard to manage activity and security.



[1] https://ieeexplore.ieee.org/document/6803166

Modular versus End-to-End Systems

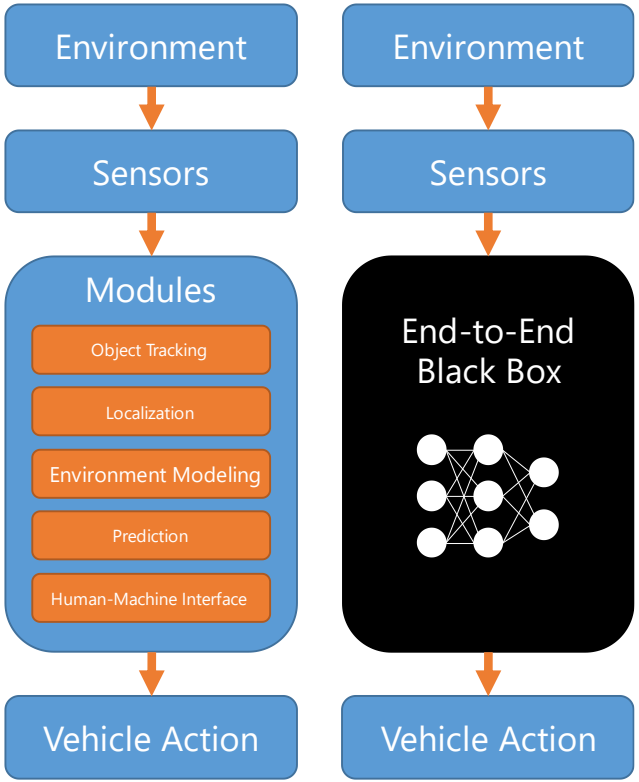
Another aspect to categorize machine learning systems are modular versus end-to-end.[1]

Modular systems "divide-and-conquer" driving into separate tasks (some using machine learning) to create a software pipeline.

End-to-end systems are often pure machine learning, ingesting sensor readings and outputting vehicle actions via deep supervised or deep reinforcement learning.

Modular systems are more prevalent and practical than end-to-end systems, as they are more interpretable and performant.

From a data engineering perspective what are the benefits of modular systems?



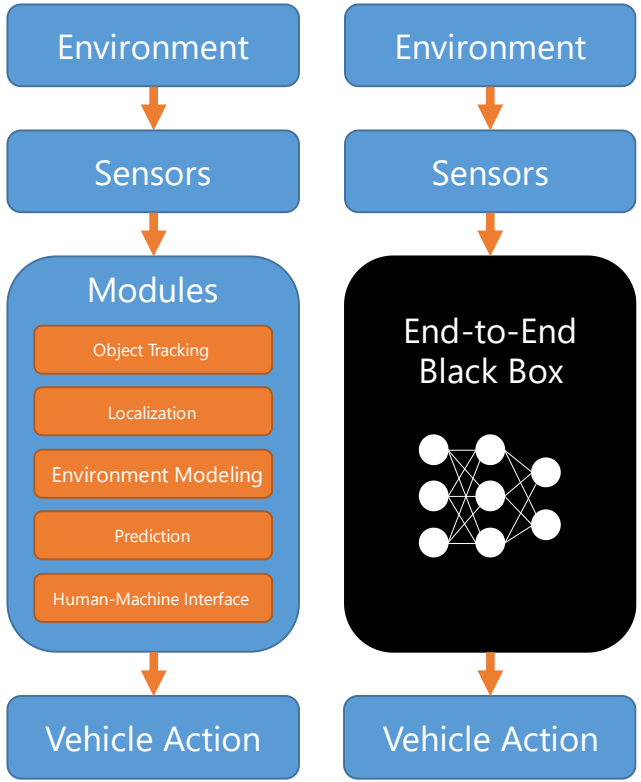
SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Modular versus End-to-End Systems

"If you have a sufficiently large deep learning model, and you train it on a dense sampling of the input-cross-output space for a task, then it will learn to solve the task, whatever that may be — *[video games] Dota*, *StarCraft,* you name it. It's tremendously valuable. It has almost infinite applications in machine perception problems. **The only problem here is that the amount of data you need is a combinatorial function of task complexity, so even slightly complex tasks can become prohibitively expensive.**

"Take self-driving cars, for instance. Millions upon millions of training situations aren't sufficient for an end-to-end deep learning model to learn to safely drive a car. Which is why, first of all, [Level 5] self-driving isn't quite there yet. And second, the **most advanced self-driving systems are primarily symbolic models that use deep learning to interface these manually engineered models with sensor data**. If deep learning could generalize, we'd have had L5 self-driving in 2016, and it would have taken the form of a big neural network."

-Francis Chollet, AI Researcher at Google and creator of Keras Tensorflow



[1] https://www.theverge.com/2019/12/19/21029605/artificial-intelligence-ai-progress-measurement-benchmarks-interview-francois-chollet-google

Inertial Sensors and IMU

Air and ground vehicles typically come with instruments to measure speed, acceleration, and other states.

Wheel encoders drive odometers, tachometers can measure speed, altimeters can measure altitude.

Inertial measurement units (IMU) can leverage velocity readings to determine location and direction.

IMU's can be paired with GPS to perform localization, position, and orientation and correct accumulated errors from dead reckoning (which approximates using speed and time to approximate location).

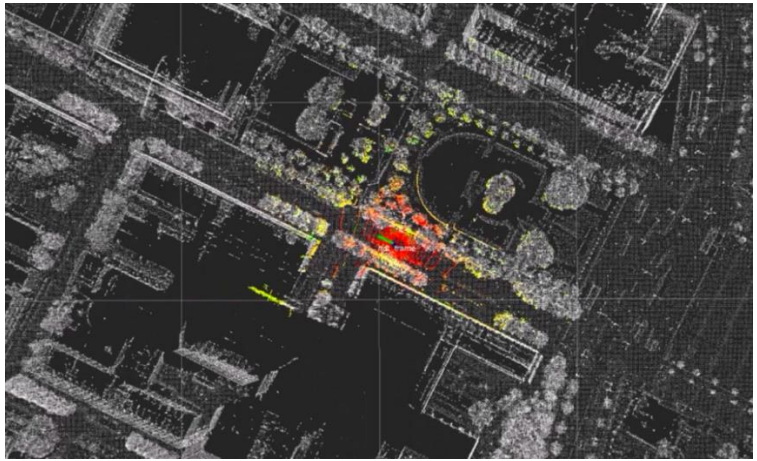
Localization: A Priori Map-Based

To get more accuracy on exact location using mapping data, there are 3 techniques:

A) **Landmark matching** – Use LIDAR, camera, and other sensors to detect a known landmark like signs and road markers and then reconcile with GPS/IMU location.

B) **Point cloud matching** – State of the art; LIDAR, camera, GPS/IMU, and other sensors generate a "3D point cloud" from immediate surroundings, and then matched to static mapping data to hone location; stochastic and probability-based matching techniques are used to estimate the fit.

C) **2D to 3D matching** – Similar to point cloud matching, but a static prior 3D point cloud is matched to online 2D images simply using camera sensors; since cameras are cost-effective this technique could gain popularity.



An offline map is localized with sensor scans from an ADS, matching colored 3D point clouds with the static white points.

Localization: Simultaneous Location and Mapping (SLAM)

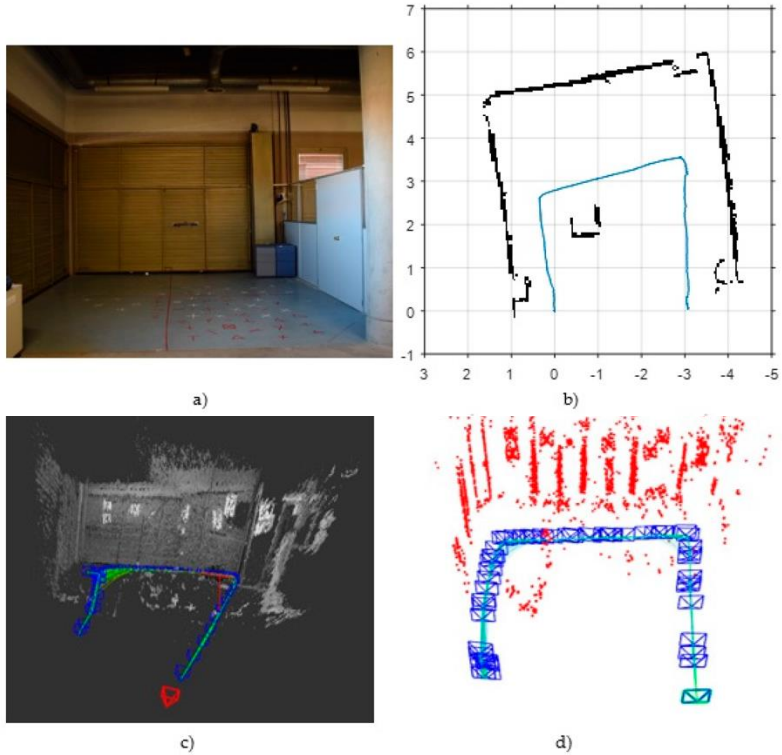
Simultaneous Location and Mapping (SLAM) is a localization technique where a vehicle generates a map while determining its location at the same time (from GPS, IMU, etc).

No prior maps or data are required, enabling it to work anywhere and without prior data collection about the environment.

It works better in indoor and sparse environments, and is used on robot vacuums, autonomous submarine vehicles, and space rovers.

Using it in outdoor and busy environments creates a lot of activity, and requires immense computing power.

But if the technology develops and improves, it can hold a lot of potential in consumer ADS applications as well as military where radio communications are denied.



Garcia et al. demonstrates an aerial drone performing SLAM in a poorly-lit, GPS-denied indoor environment.[1]

Data Cleaning: Semantic/Instance Segmentation

Consider another data transformation and cleaning task called semantic/instance segmentation, which classifies each individual pixel.

- Not every object conveniently fits into a box such as roads, lane markings, sidewalks, and building fronts.
- Identifying individual pixels in an already identified object is known as **instance segmentation**.
- Classifying pixels that were not classified previously in an object is known as **semantic segmentation**.

Mask R-CNN uses multiple neural networks to generate bounding boxes and then apply "segmentation masks," which classify each individual pixel within each bounding box.

SOURCE: https://github.com/matterport/Mask_RCNN

Data Cleaning: Semantic/Instance Segmentation

We can use these labelled pixels to eliminate noise, find relevant surfaces, and determine object paths.

With performance at 5 FPS, Mask R-CNN (shown to the right) and other segmentation algorithms are approaching acceptable speed for real-time monitoring.

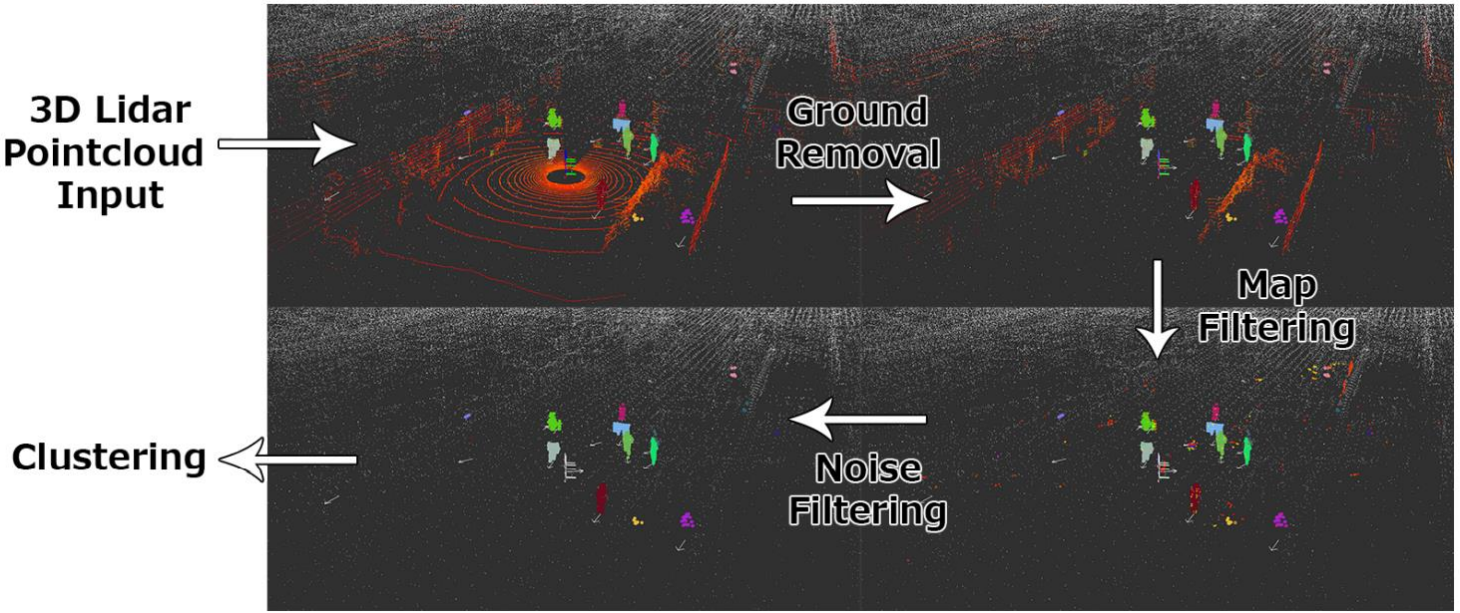
However, let's look at what happens in practice and consider how data is cleaned in a fuzzy way here.

Mask R-CNN uses multiple neural networks to generate bounding boxes and then apply "segmentation masks," which classify each individual pixel within each bounding box.

SOURCE: https://github.com/matterport/Mask_RCNN

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Data Cleaning: Semantic/Instance Segmentation



Yurtsever et al demonstrate object detection from a 3D point cloud using map filtering, noise filtering, and clustering heuristics.

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Why do you think ”self-driving” has been so difficult to achieve from a data engineering perspective?

Gathering Data in Miles Driven

“The most important thing to understand is that not all miles are the same. Most miles that we drive are very easy, and we can drive them while daydreaming… but some miles are really, really hard, and so it’s those difficult miles that we should be looking at.”

- Gill Pratt, CEO of Toyota Research Institute

Case Study

Uber Tempe Incident

Setting the Stage

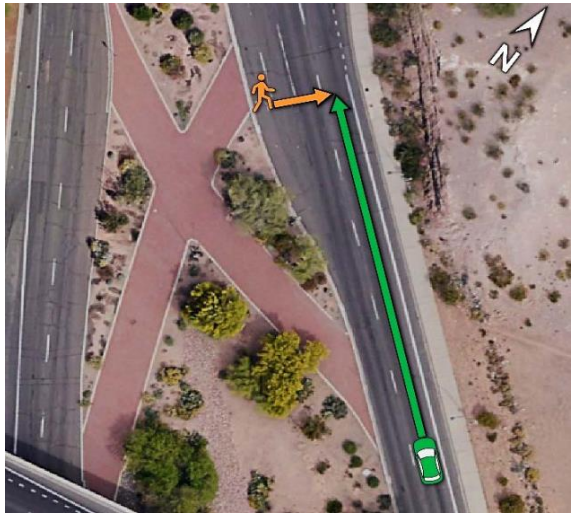
Sunday, March 18, 2018
9:58 PM
Tempe, Arizona

Uber deployed a modified 2017 Volvo XC90 into a geofenced route for testing.

The vehicle was engaged in autonomous operation but had a human operator monitoring.

A pedestrian jaywalking a bicycle started crossing Mill Avenue and into a collision path with the vehicle.

The vehicle was engaged in autonomous operation and failed to slow, fatally striking the pedestrian at 39 MPH.



SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Setting the Stage

The vehicle was equipped to be “self-driving” and had the following sensors:

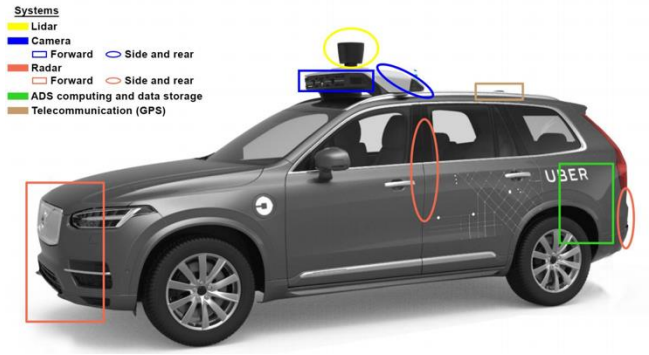
Forward/side facing cameras
Radar and ultrasonic
LIDAR
Telemetry, positioning, monitoring, and telecommunications

The system relied on detailed mapping data and localization to identify speed limits and appropriate lanes.

LIDAR, followed by computer vision, was heavily instrumental for object detection and classification, as well as create a map of the designated route up to 238 feet.

Ultrasonic sensors with 16 foot range detected immediate obstacles for collision avoidance.

The operator was tasked with intervening when automation failed, and she was distracted by her phone right before the accident. The automation was fully engaged for 19 minutes prior to the collision.



SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Opening the Black Box

6 seconds prior to impact, the vehicle LIDAR and radar pinged the pedestrian at 43 MPH.

However as it closed in on the pedestrian, misclassifications started to alternate:

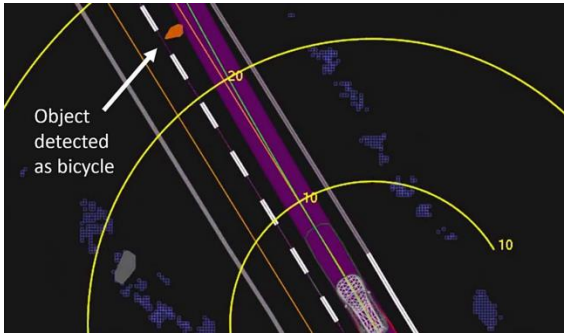
“Unknown object”

“Vehicle”

“Bicycle”

Each misclassification caused it to predict a different trajectory and travel path based on that prescribed heuristic.

1.3 seconds prior to impact, the system identified the need for emergency braking, but it was disabled to reduce “erratic vehicle behavior” and rely on operator to intervene in emergencies.



SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Opening the Black Box

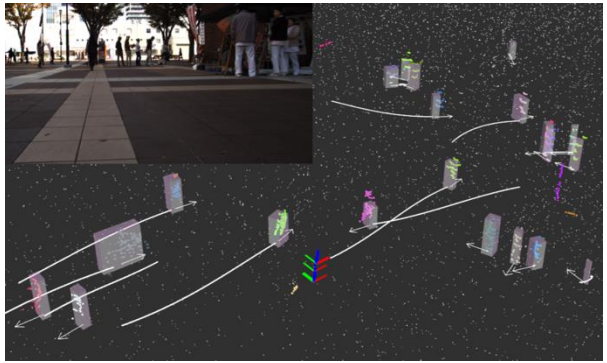
An important heuristic to note is "the ADS used a prioritization schema that promoted tracking by certain sensory systems over others, and that was also dependent on the recency of an observation" (Page 12).

"Once the ADS perception process classified a detected object, the ADS generated multiple possible trajectories—path predictions—based on the typical goal of the detected object and its tracking history" (Page 12).

"Certain object classifications ("other") were not assigned goals. Their currently detected location was viewed as static, and unless the location was directly on the path of the test vehicle, the object was not considered a possible obstacle. Pedestrians outside the vicinity of a crosswalk were also not assigned an explicit goal" (Page 13).

The system also used heuristics based on assumed contexts:

"The system never classified her as a pedestrian—or correctly predicted her path—because she was crossing N. Mill Avenue at a location without a crosswalk, and the system design did not include consideration for jaywalking pedestrians" (Page 16).



Yurtsever et al. demonstrate object tracking at a busy intersection. Pedestrians and cyclists leave a trail of their previous path and a model points an arrow to their predicted trajectory.

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Timeline

Table. Selected parameters recorded by vehicle's ADS.

Time to Impact (seconds)	Speed (mph)	Classification and Path Prediction[a]	Vehicle and System Actions[b]
-9.9	35.1	--	Vehicle begins to accelerate from 35 mph in response to increased speed limit.
-5.8	44.1	--	Vehicle reaches 44 mph.
-5.6	44.3	Classification: *Vehicle*—by radar Path prediction: *None*; not on path of SUV	Radar makes first detection of pedestrian (classified as vehicle) and estimates speed.
-5.2	44.6	Classification: *Other*—by lidar Path prediction: *Static*; not on path of SUV	Lidar detects unknown object. Object is considered new, tracking history is unavailable, and velocity cannot be determined. ADS predicts object's path as static.
-4.2	44.8	Classification: *Vehicle*—by lidar Path prediction: *Static*; not on path of SUV	Lidar classifies detected object as *vehicle*; this is a changed classification of object and without a tracking history. ADS predicts object's path as static.
-3.9[c]	44.8	Classification: *Vehicle*—by lidar Path prediction: Left through lane (next to SUV); not on path of SUV	Lidar retains classification *vehicle*. Based on tracking history and assigned goal, ADS predicts object's path as traveling in left through lane.

Note this interesting progression in how the pedestrian is classified

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Timeline

Table. Selected parameters recorded by vehicle's ADS.

Time to Impact (seconds)	Speed (mph)	Classification and Path Prediction[a]	Vehicle and System Actions[b]
-3.8 to -2.7	44.7	Classification: alternates between *vehicle* and *other*—by lidar Path prediction: alternates between *static* and left through lane; neither considered on path of SUV	Object's classification alternates several times between *vehicle* and *other*. At each change, tracking history is unavailable; ADS predicts object's path as static. When detected object's classification remains same, ADS predicts path as traveling in left through lane.
-2.6	44.6	Classification: *Bicycle*—by lidar Path prediction: *Static*; not on path of SUV	Lidar classifies detected object as *bicycle*; this is a changed classification of object and object is without a tracking history. ADS predicts bicycle's path as static.
-2.5	44.6	Classification: *Bicycle*—by lidar Path prediction: Left through lane (next to SUV); not on path of SUV	Lidar retains *bicycle* classification; based on tracking history and assigned goal, ADS predicts bicycle's path as traveling in left through lane.
-1.5	43.8[d]	Classification: *Other*—by lidar Path prediction: *Static*; partially on path of SUV	- Lidar detects unknown object; because this is an unknown object, it lacks tracking history and is not assigned a goal. ADS predicts object's path as static. - Although detected object is partially in SUV's lane of travel, ADS generates motion plan around object (maneuver to right of object); motion plan remains valid—avoiding object—for next two data points.

Note this interesting progression in how the pedestrian is classified

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Timeline

Table. Selected parameters recorded by vehicle's ADS.

Time to Impact (seconds)	Speed (mph)	Classification and Path Prediction[a]	Vehicle and System Actions[b]
-1.2	43.2	Classification: *Bicycle*—by lidar Path prediction: Travel lane of SUV; fully on path of SUV	- Lidar detects bicycle; although this is a changed classification and without a tracking history, it is assigned a goal. ADS predicts bicycle to be on SUV's path. - ADS motion plan (generated 0.3 seconds earlier) for steering around bicycle no longer possible; situation becomes hazardous (emergency situation). - Action suppression begins.
-0.2	40.5	Classification: *Bicycle*—by lidar Path prediction: Travel lane of SUV; fully on path of SUV	- Action suppression ends 1 second after it begins. - Situation remains hazardous; ADS initiates plan for gradual vehicle slowdown. - Auditory alert indicates that ADS is engaging and controlled slowdown is initiating.[e]
-0.02	39.0	--	Vehicle operator takes control of steering wheel, disengaging ADS.
Impact			
0.7	37	--	Vehicle operator brakes.

Note this interesting progression in how the pedestrian is classified.

SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

From a data engineering perspective, what went wrong?

Opening the Black Box

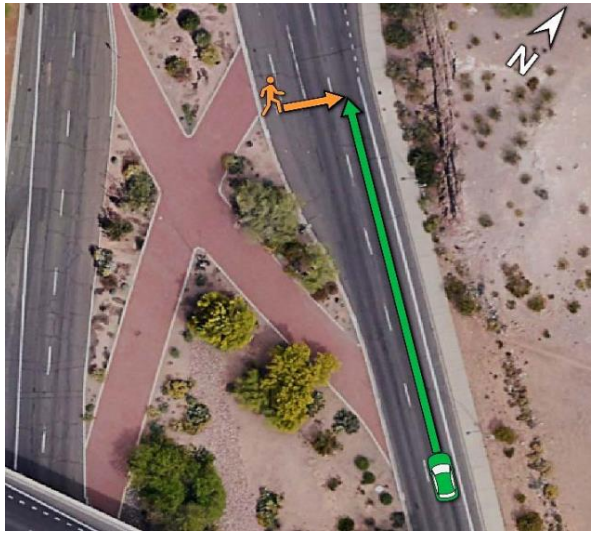
1) Fuzzy logic with a stochastic/probabilistic nature requires immense amounts of guardrails and heuristics.
Computer vision, LIDAR, and other pixelated techniques can have infinite combinations of inputs, which creates uncertainty requiring safeguards that restore safe and predictable behavior.

2) Challenge assumptions about operating domains; it is easy to assume a perfect world where pedestrians use crosswalks and operating domains stay consistent, but reality is chaotically disruptive, and upsets must be accounted for.

3) Hunt down and capture logical inconsistencies; the moment misclassification/path prediction of the pedestrian started flickering, a cautious heuristic should have been used to slow down the vehicle and get operator's attention.

4) Have a safety management system (SMS) program! Uber did not have a formal safety management system in place, and the system design clearly shows this.

What other lessons do you think exist?



SOURCE: Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K. (2020). A survey of autonomous driving: Common practices and emerging technologies. IEEE Access, 8, 58443-58469.

Read the Reports

Here are some reports and documents surrounding the Uber Tempe crash in 2018.

https://www.ntsb.gov/investigations/AccidentReports/Reports/HAR1903.pdf

https://1drv.ms/b/s!AoxKpPBUQ5LnltoUnQExu3QDb-ihkg?e=cNV79a

Knowing what you know now, how would you approach “self-driving” as a data engineer?

What do you think is most important?

The Data Engineering Problem with Self-Driving

What is the hardest part about self-driving from a data engineering perspective?

Is it computing power? Storage space? Needing more data?

No. It's operating domain.

If the operating domain is not controlled, the sensors are too complex and fuzzy, and too many actions are allowed, the self-driving initiative is not going to succeed.

Even day/night cycles, weather, graffiti on stop signs, and other conditions create a combinatorial problem approaching infinity.

A simple child in a Halloween costume reveals how many infinite cases there are that cannot be captured in data collection.

The Data Engineering Problem with Self-Driving

Automatic braking using radar is one thing; that's a narrow application using narrow data on a narrow action.

Theoretically, a radio-connected infrastructure where EVERYTHING is a connected agent (including pedestrians) and talking to each other could also work.

But classifying infinite possibilities on the road using fuzzy sensors like camera and LIDAR, and mapping into driving actions (steer, gas, brake) safely is an impractical data engineering problem.

The only way these fuzzy sensors will work is to control the operating domain, and thus limit the amount of data that needs to be collected.

IN SUMMARY, DON'T OVERLOOK THE OPERATING DOMAIN! IF YOU CAN, CONTROL IT SO THE DATA IS PREDICTABLE.

Lessons Learned

You are unlikely to be working on self-driving cars anytime soon, but we can learn from the struggles of companies that tried.

When your data is unpredictable because of outliers and an uncontrolled domain, it can be catastrophic for a data project.

You cannot underestimate the source of the data, particularly the environment that creates it, and not account for the chaos it inflicts in your application.

The more variables (e.g. pixels, inputs, sensors) you implement, the more complex and expensive (combinatorial) the data becomes.

Try to scope and simplify as much as possible, and encourage pursuing narrow objectives rather than broad ones.