LA TROBE UNIVERSITY

Building AI – Module 4

AI Lifecycle

7 June 2022

Optus U

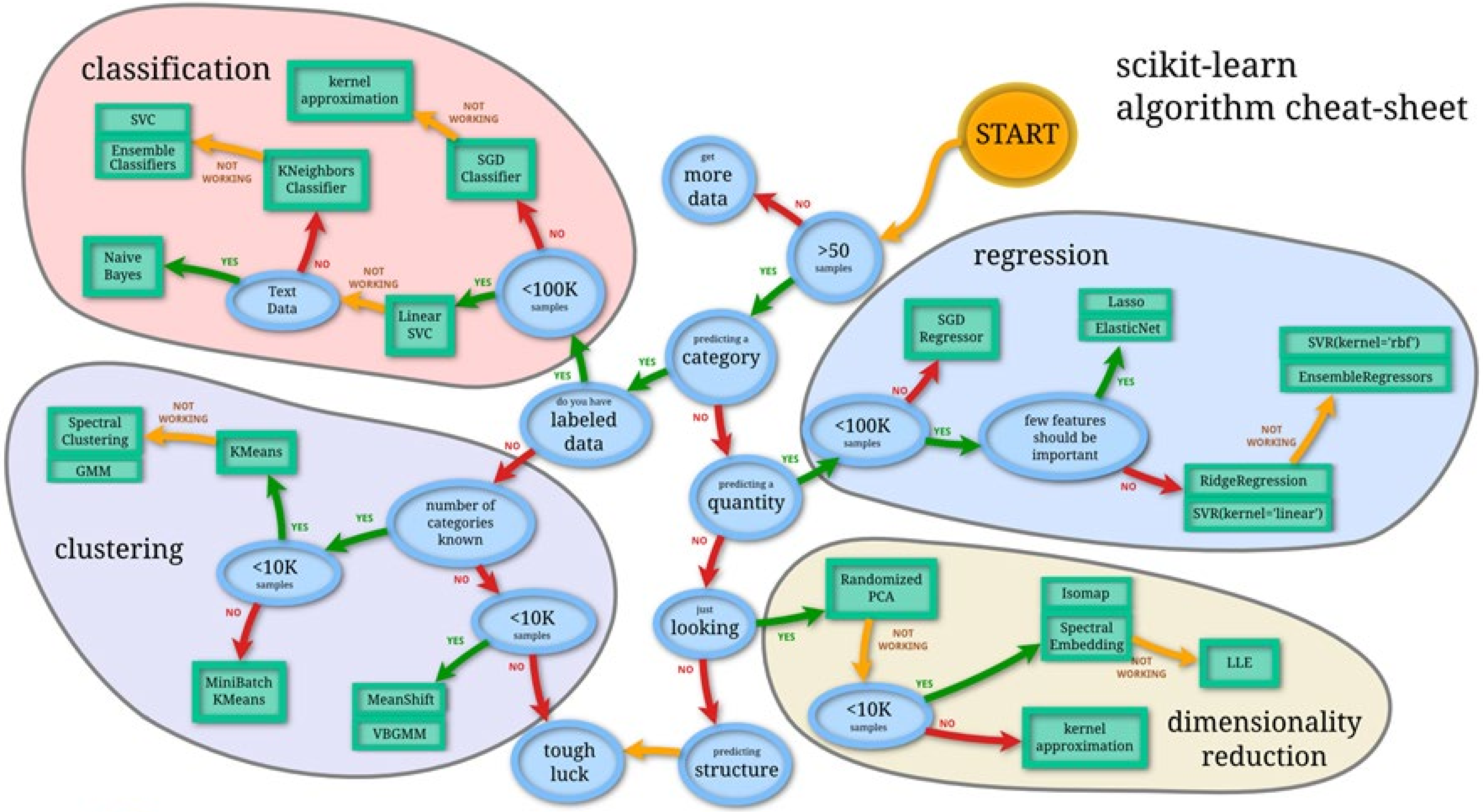
Agenda

1. Conventional AI lifecycle
2. An updated AI lifecycle
3. Phase by phase
4. Evaluation metrics

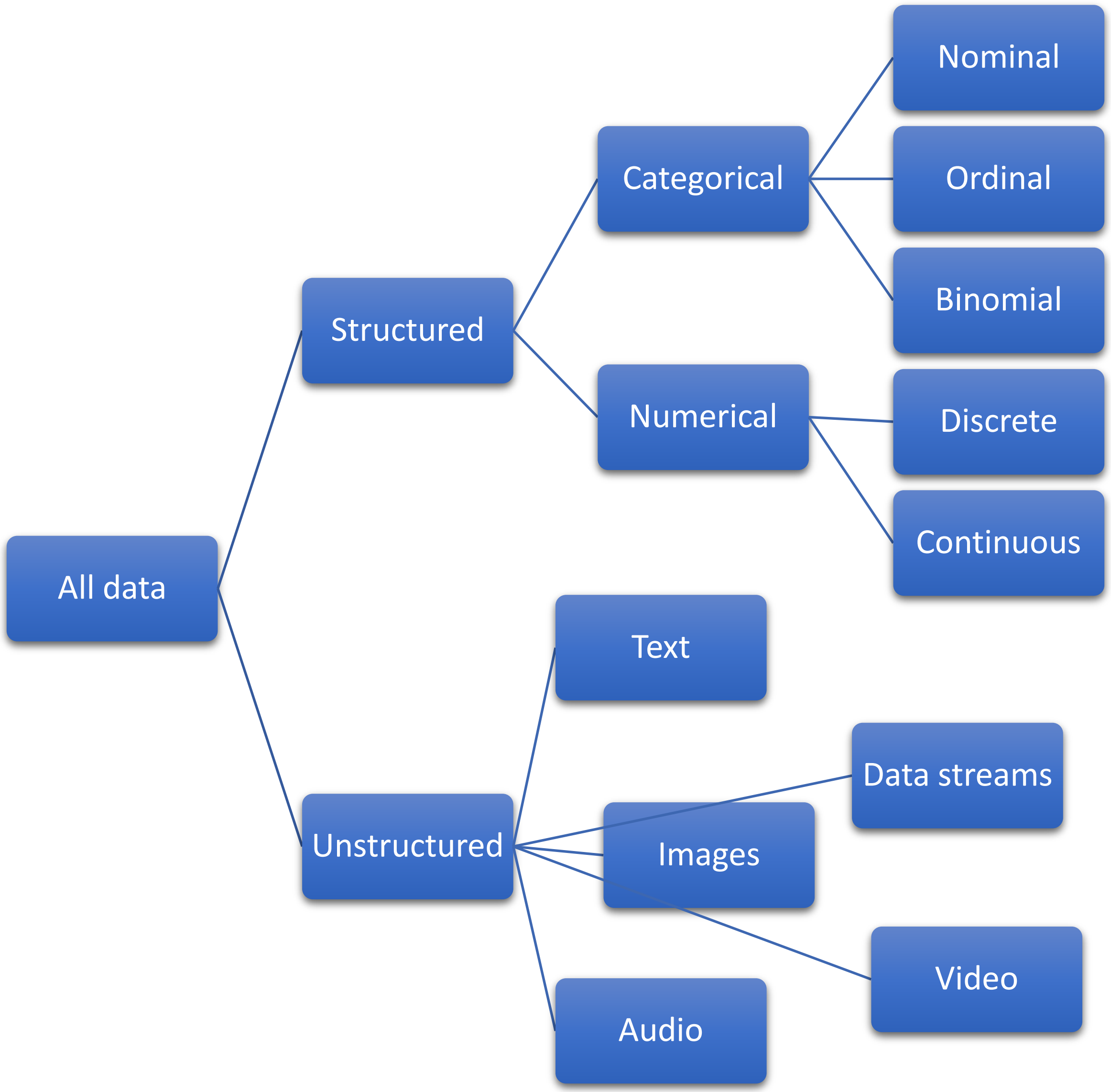
Decision-making capabilities

- "Algorithms that generate intelligence to inform decision-making"
  - How about algorithms that do not generate intelligence?
- The decision-making capabilities are,
  - A prediction – regression, classification, time series, sequence
    - Classification is also detection – object, anomaly, concept
  - An association – clustering, feature selection, dimensionality reduction
  - An optimisation – scheduling, planning, control, generation, simulation
- And these capabilities work across different modalities of data
  - Text – natural language processing/understanding (NLP/U)
  - Image, audio, video – computer vision, speech recognition

Capabilities to algorithms



A taxonomy of data types

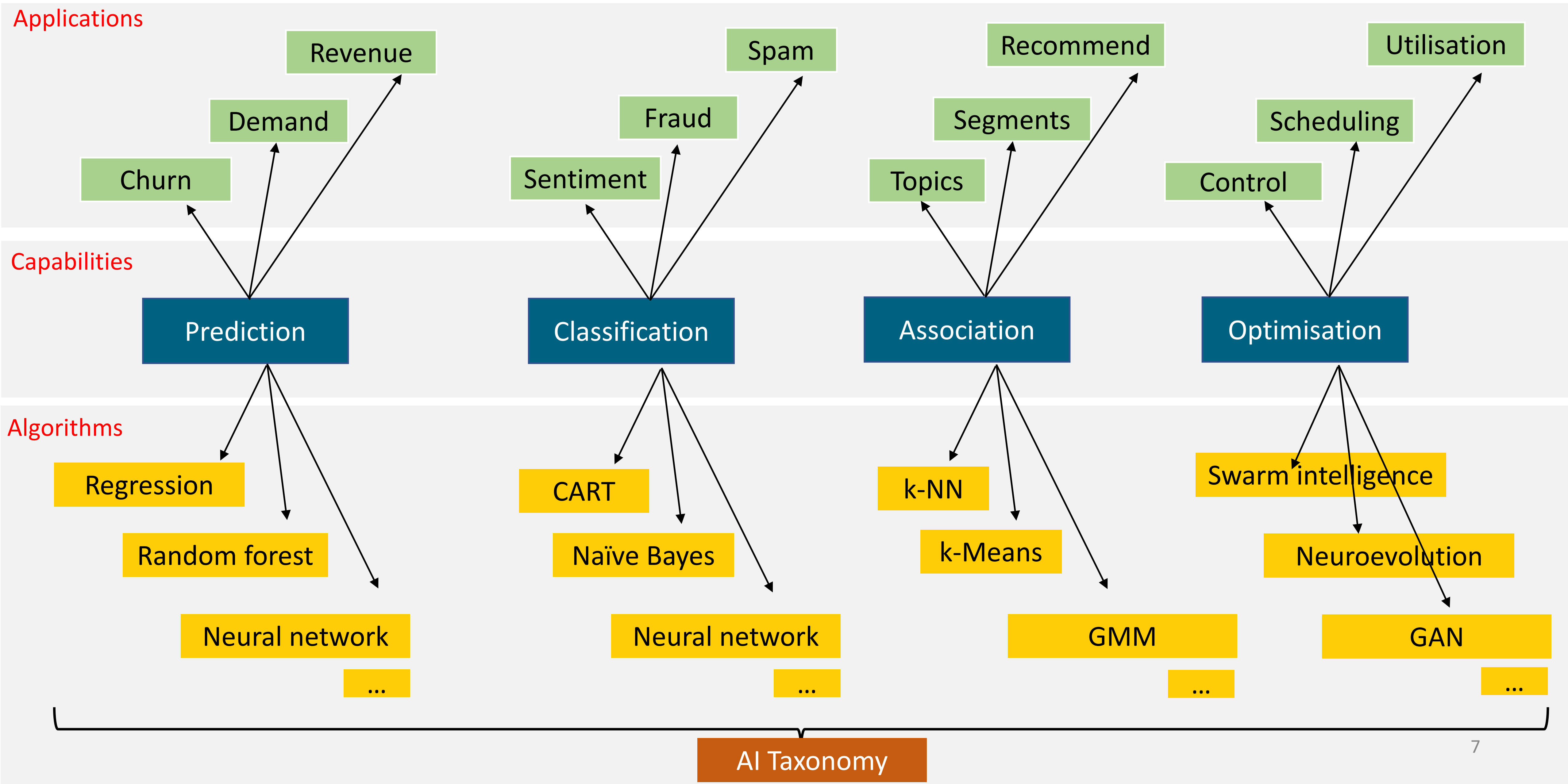


From capabilities to applications

- Prediction – demand, revenue, churn, weather, stock price, protein folding
- Classification – face recognition, spam detection, credit card fraud, sentiment
- Association – recommendation, behaviour profiling, customer segments, topics, machine translation
- Optimisation – resource utilisation (CPU, storage, networks, energy), digital twins, control, robotics

- Complex applications requiring multiple capabilities:
  - Algorithmic trading
  - Conversational chatbots
  - Self-driving vehicles

Algorithms → Capabilities → Applications



Taxonomy – Machine learning

- Supervised -
  - k-nearest neighbour, linear regression, logistic regression, naive bayes, decision trees, CART (Classification and Regression Trees), gradient boosting (XGBoost, Light GBM), neural networks (perceptron, backpropagation), support vector machines
- Unsupervised -
  - k-Means clustering, mixture models, PCA, self-organising maps, autoencoders
- Reinforcement -
  - Monte Carlo, SARSA, Q-learning
- Deep learning -
  - CNN, RNN, LSTM, DBN, Stacked autoencoders
- Fine-tuning (domain adaptation) -
  - Transfer learning, active learning, ZSL, OSL, FSL

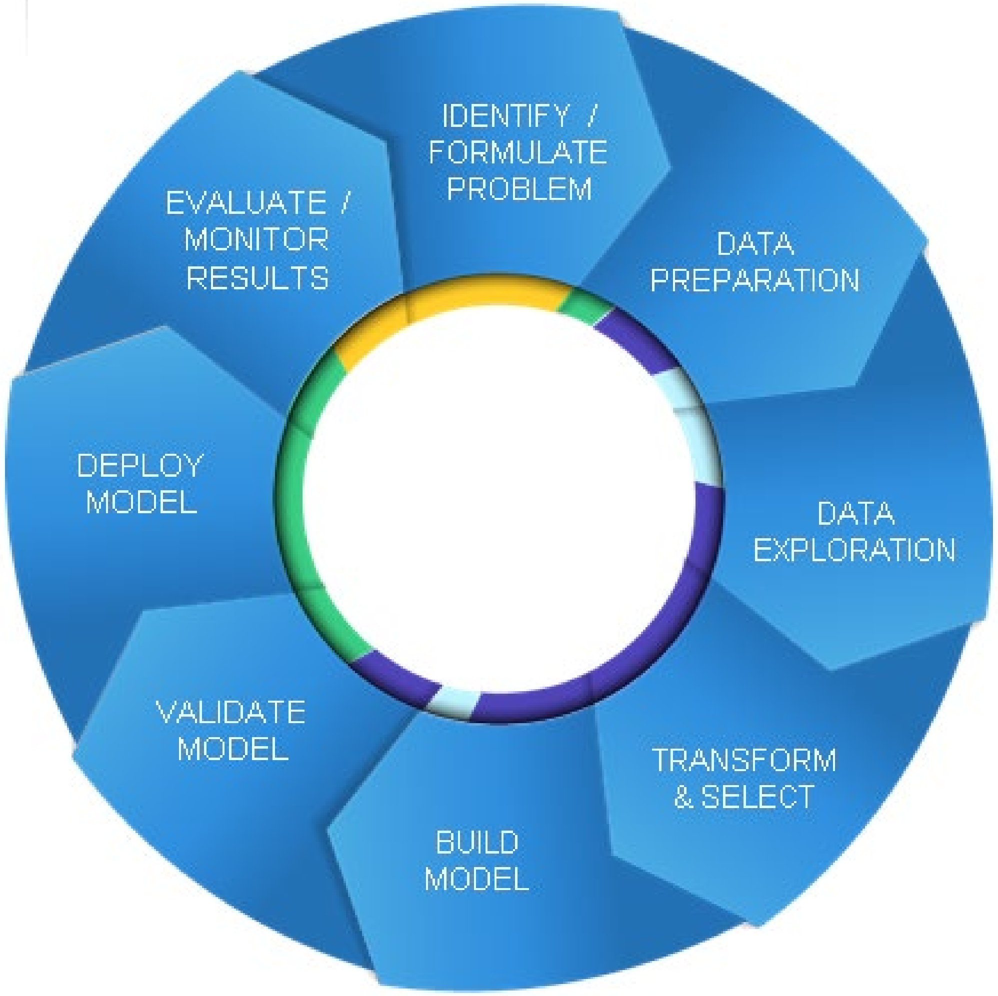
Taxonomy – Others

- Knowledge engineering/representation -
  - Decision trees, CART (Classification and Regression Trees), knowledge graphs, ontology engineering
- Machine reasoning -
  - Decision trees, Fuzzy logic, Bayesian networks, Markov models
- Evolutionary computation (metaheuristics)
  - Genetic algorithms, swarm intelligence (PSO, ACO, ABC)
- Generative modelling
  - GANs, latent spaces, representation learning

Statistical models vs Machine learning

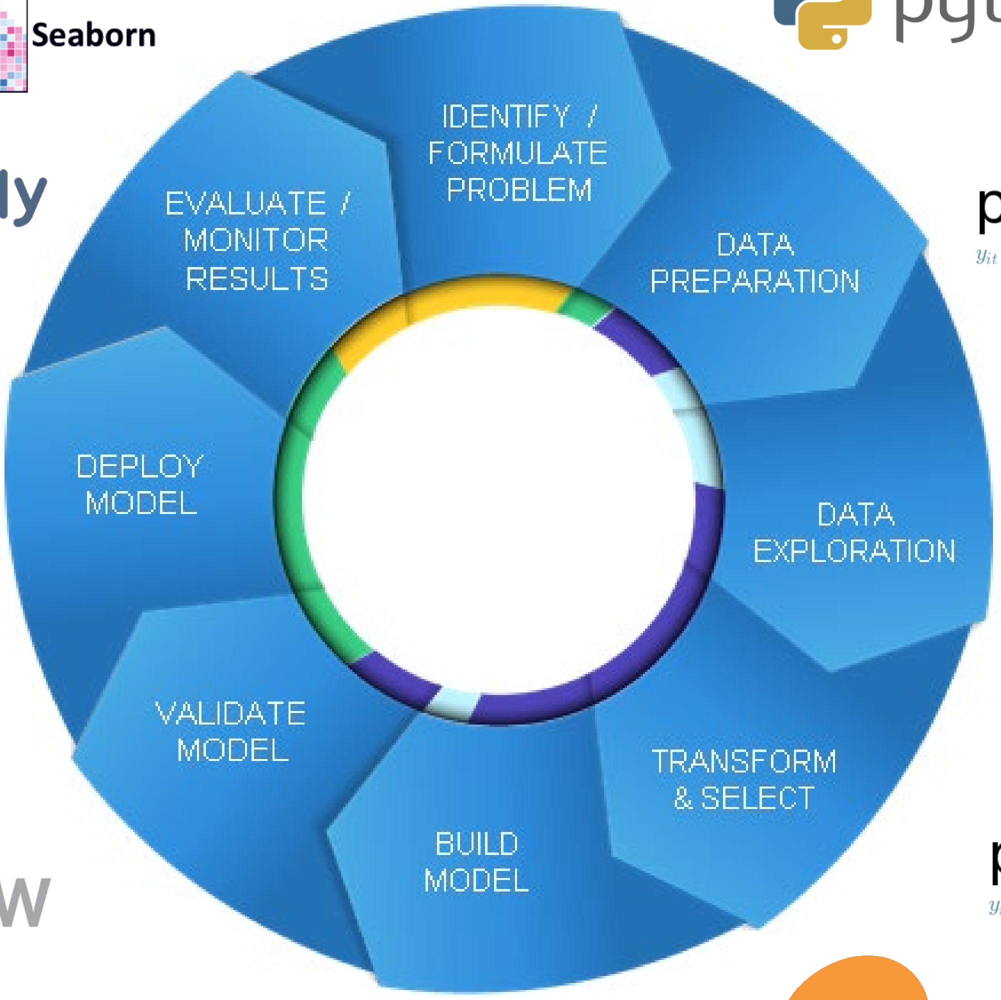
- Isn’t regression a statistical model? - fairly dated debate, but you may still come across this.
  - “When we raise money it’s AI, when we hire it's machine learning, and when we do the work it's logistic regression” (by a statistician of course)
  - Short answer – just get the job done
- Model-based vs data-driven
  - In ML, training a linear regressor to predict = In SM, best fit line to minimise the squared error
  - SM starts with a hypothesis test and a set of rules (assumptions), ML is less rigid
  - SM works on all the ‘long’ data, ML takes ‘wide’ data and splits into train/validate/test
  - But SM will also remove outliers or look for known distributions whereas ML doesn’t
  - SM focuses on causality through linearity, ML focuses on correlation through non-linearity
  - In this manner, SM fills in the unobserved, ML looks for/learns patterns (inference vs prediction)
  - ML is more technology-ready (big data, cloud, pipelines), SM is not.

The conventional AI lifecycle



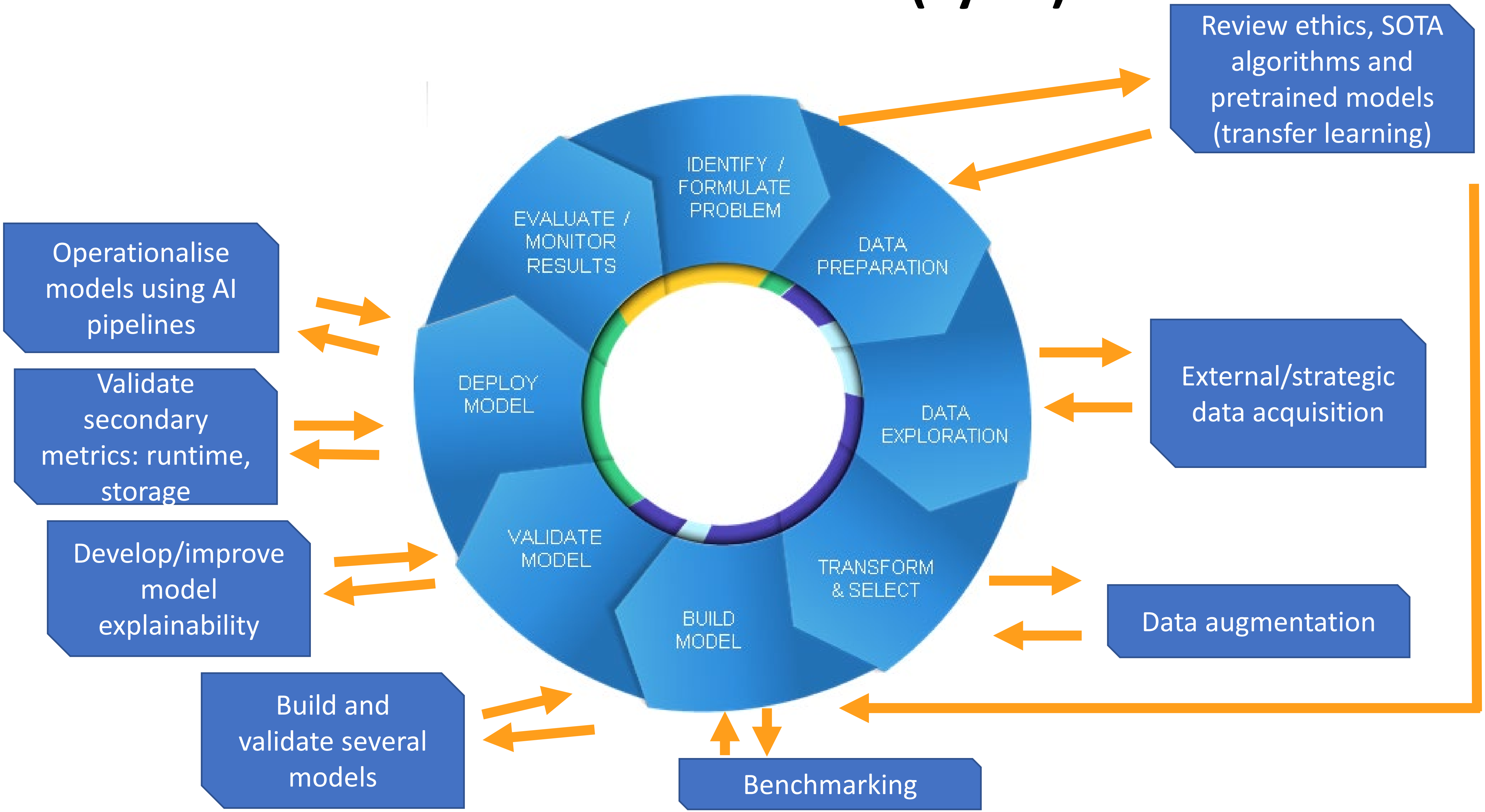
with Python

matplotlib
Seaborn
plotly
Flask
web development, one drop at a time
TensorFlow
PYTORCH

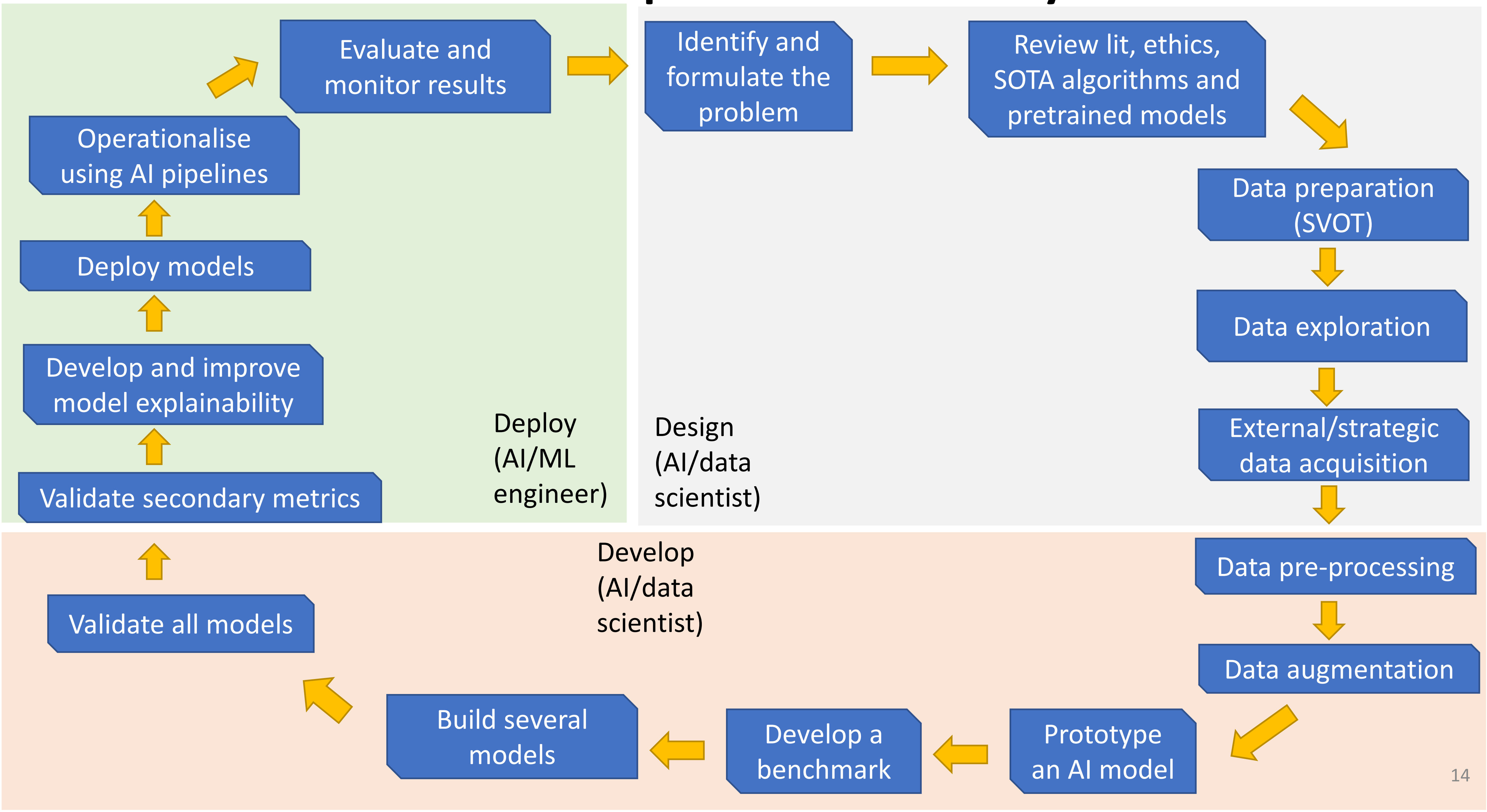


python
jupyter
NumPy
pandas
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$
SciPy
SM StatsModels Statistics in Python
Seaborn
matplotlib
plotly
pandas
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$
bonobo
Keras
scikit learn

Is there more to life(cycle)?



A more comprehensive AI lifecycle © CDAC 2019

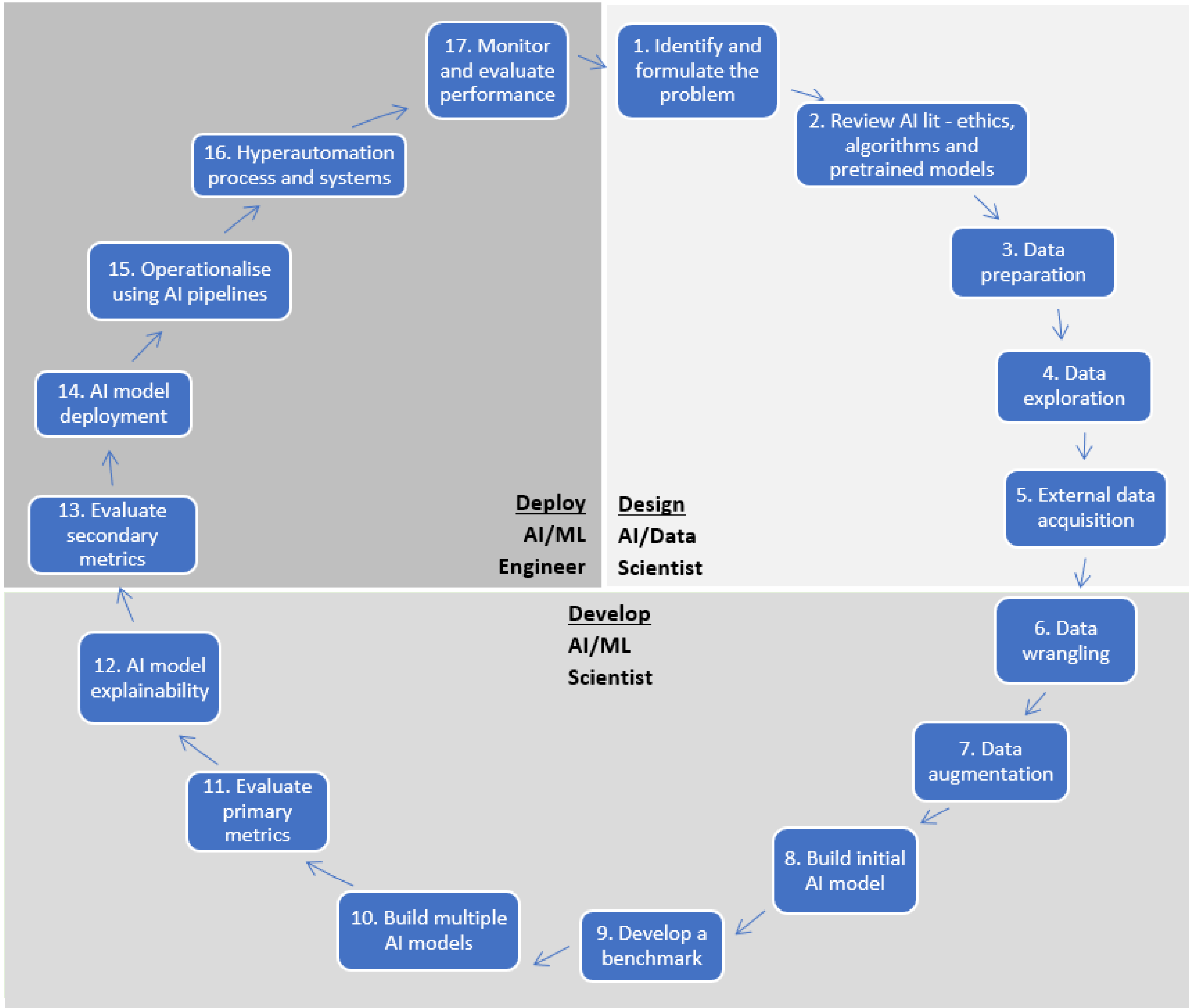


AI life cycle

And now 19 phases..

"An Artificial Intelligence Life Cycle: From Conception to Production," Patterns, 2022

https://www.sciencedirect.com/science/article/pii/S2666389922000745#fig1



Lifecycle phases 1-2

1. **Identify and formulate the problem**

- How is the problem currently solved? (improve - efficient, enhance - effective, disrupt - innovate)
- What are the steps/phases of the process?
- Where/how are the rules defined for the process?
- What data is collected, how is it stored, what historical data is available?

2. **Review literature, ethics, state-of-the-art (SOTA) algorithms and pretrained models**

- Google search/ Google scholar
- Tech news (Wired, Inside AI, MIT tech review, CIO.com, techcrunch etc.)
- Publishing platforms (Medium, Towards Data Science, etc.)
- Q&A sites (Stack Exchange, Quora, etc.)
- Research labs (CDAC!, MIT, Stanford etc.)
- Code repositories and Cloud Platforms (GitHub, Azure, AWS, GCP)
- Social media (Twitter)

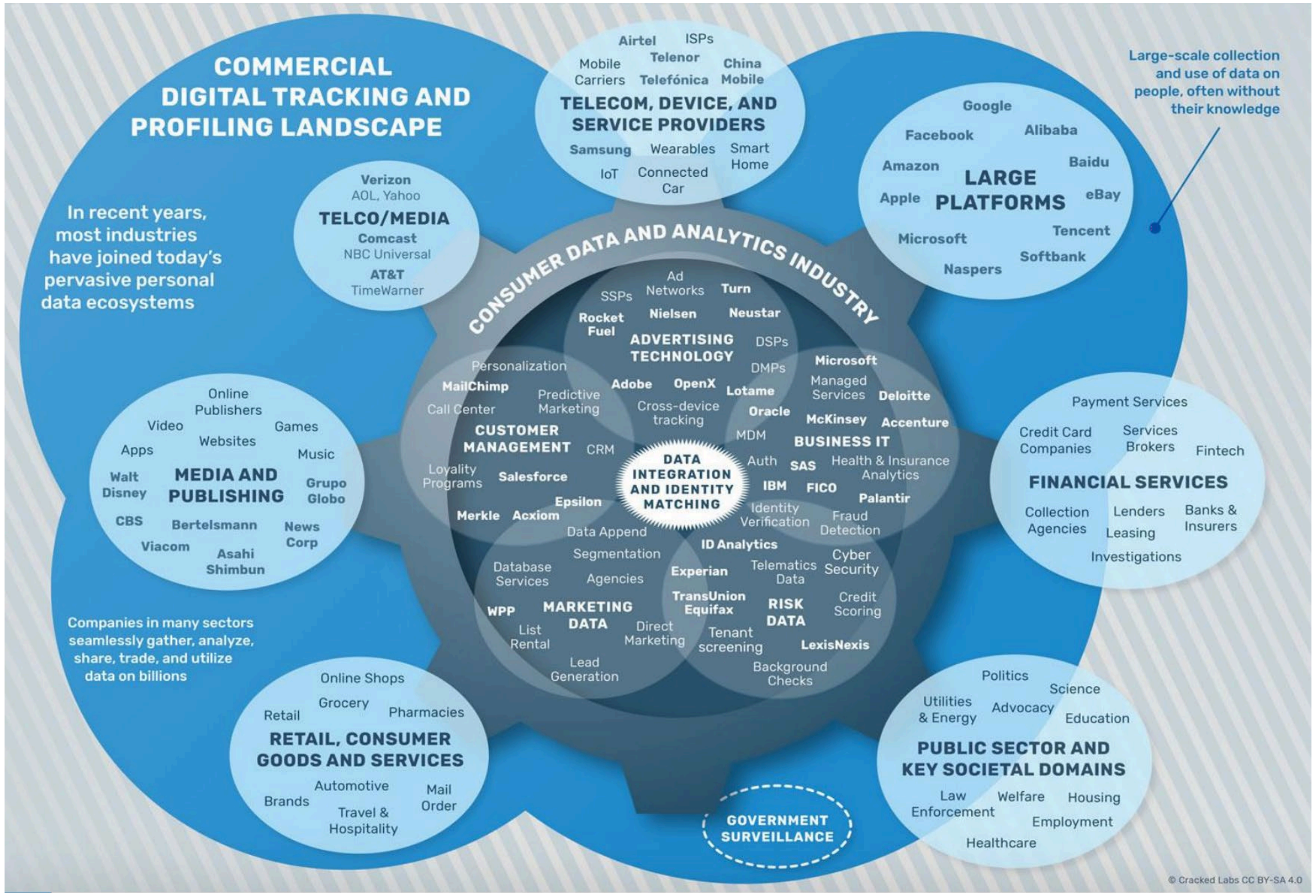
Lifecycle phases 2-4

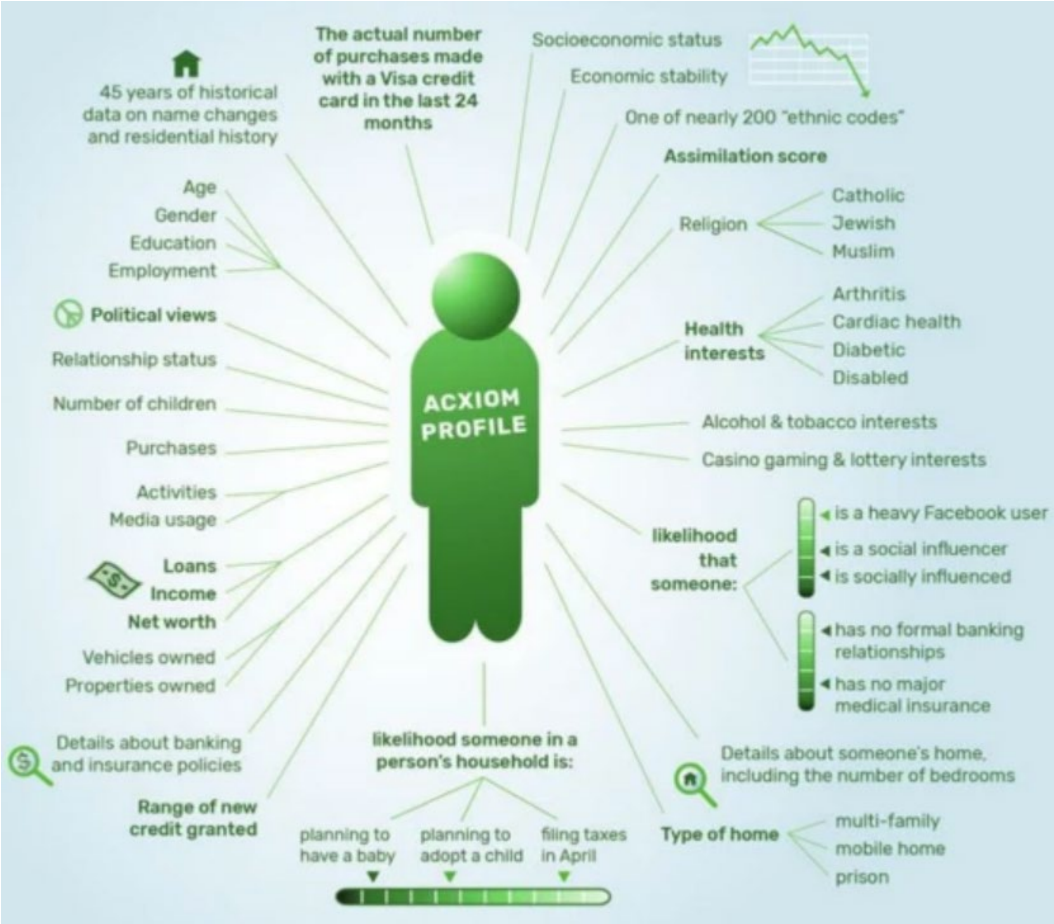
2. **(continued) Review literature, ethics, SOTA algorithms and pretrained models**
    - What to look for -
    - Literature reviews, commentaries, letters, articles, op-ed
    - Case studies, best practices, product/tool documentation
    - Tutorials, guides, Forums - Q&A, demos
3. **Data preparation (SVOT)**
    - Digital representations of the problem domain/machine learning task
    - A unified data warehouse/data lake setup
    - Access, ownership, stewardship, metadata, ethics
4. **Data exploration**
    - Comparison with industry benchmarks/ algorithmic baselines
    - Granularity, quality, relationships

Lifecycle phases 5-6

5. **External/strategic data acquisition**
    - Long-term – collected from primary stakeholders, based on trust and full transparency
    - Short-term – more than 4000 data brokers and vendors in 2019, a $200 billion per year industry
    - Acxiom is one of the largest - collects up to 1,500 different pieces of information on 500 million consumers.
    - Public records, credit scoring, social media, web history, apps, data sharing agreements
    - Maintain ethical and legal boundaries
6. **Data pre-processing**
    - Formatting – dates, representations
    - Values - duplication, missing, erroneous
    - Transformations - normalisation, relationships, hierarchies, data type changes

Data tracking





Lifecycle phases 7-9

7. **Data augmentation**

- Oversampling – for the minority class, create synthetic samples or duplicate existing data (SMOTE)
- Undersampling - for the majority class, delete or merge samples
- Transfer learning from synthetic data, warping - dynamic time warping
- Images – flip, crop, scale, rotate

8. **Prototype an AI model (Slides 4-9)**

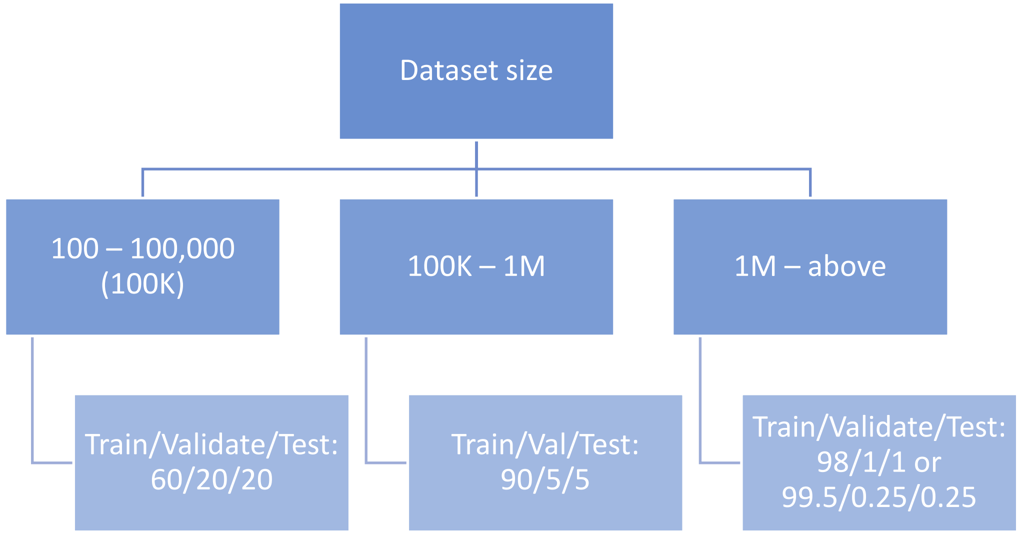
- Finally, building an AI model!
- Phases – train, validate, test
- Structure - algorithm, parameters, architecture, model

9. **Develop a benchmark**

- Develop a performance benchmark using a baseline model (like linear or logistic regression, pretrained)
- Common-sense heuristic - 10% of the time to develop, but 90% of the way to achieve reasonably good results
- A better understanding of your data – what fits/doesn't fit
- Not to be confused with benchmark datasets (MNIST, Iris, fashion MNIST)

Dataset splitting strategy

- Conventional data (or small data):
  - Train/Validation/Test: 60%/20%/20%, 70%/15%/15%
- With Big Data, dataset are in the range of a million+
- If the dataset is a million,
  - 30% is 300k for evaluation
  - 70% is 700k for training
- Instead,
  - Validate/Test with ~ 20,000 data points (2%)
  - Training with ~ 980,000 data points (98%)



Lifecycle phases 10-12

10. Build several models

- Next step - what does the baseline fail to capture?
- Parameter tuning, regularization techniques, back to data pre-processing
- Develop complex models (logistic regression → xgboost → 3-layer neural network → deep neural network)
- Increase model complexity up to overfitting and then use regularization techniques to generalise (not ideal)

11. Validate all the models

- Select the correct model evaluation metric (literature helps, API documentation, many blogs)
- https://scikit-learn.org/stable/modules/model_evaluation.html
- Holdout method → leads to high variance → k-fold cross-validation

12. Validate secondary metrics

- CPU usage – on-prem vs cloud, device vs cloud
- Memory usage – model compression

Evaluation metrics

- Depends on the type of AI capability:
  - Prediction – regression, classification
  - Association – clustering, dimensionality reduction
  - Optimisation – generic, but also specific
  - And many others which are problem-specific
- Characteristics of a good metric (what to look for)
  - Accurate (reliable, representative)
  - Robust (consistent)
  - Agnostic
  - Scalable
  - Interpretable (meaningful)

In general,

- $\text{Outcome}_i = (\text{model}_i) + \text{error}_i$
  - Error = predicted - actual
  - Error = classified - actual
  - Error = clustered - actual?
  - Error = optimised - actual?
- Predicting the weather,
  - Regression error = 24 - 22 or 20 - 22
  - Classification error = cloudy – sunny

Regression

- Predicting numerical values, a time-series or a sequence
- Mean Absolute Error (MAE)

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N} |Predicted_i - Actual_i|}{N}$$

  - Average difference between the predicted and actual value (for the entire test dataset)
  - Intuitive, same units (as the output variable), but a linear score (all errors weighted equally)

- Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

  - Square root of the averaged squared difference between the predicted values and the actual values
  - Larger errors are penalised – more effective when outliers need to be considered.
  - RMSE is always larger or equal to MAE
- Others - Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), R-squared

Classification - 1

- Predicting (or classifying) a categorical value/s (or label) - binary and multi-class
- Confusion matrix (seen this before?)
  - Type I – False Positive, Type II – False Negative
  - An error matrix – usually, predicted class in rows, actual class in columns
  - Used to check if the model (not people) is “confusing” the classes
  - Extends to multi-classification problems
- Classification accuracy (or just accuracy)

$$\frac{TP + TN}{TP + TN + FP + FN \text{ (total)}}$$

  - Intuitive, easy to calculate
  - Misleading when the dataset is imbalanced (and most datasets are)
  - 98% accuracy from a dataset containing 98 of class A and 2 of class B, accurate?
- Misclassification (error) rate

$$\frac{FP + FN}{TP + TN + FP + FN \text{ (total)}}$$

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positive (TP)	False Positive (FP)
Predicted Negative (0)	False Negative (FN)	True Negative (TN)

Actual

Predicted	A	B	C
A	TP_A	E_{BA}	E_{CA}
B	E_{AB}	TP_B	E_{CB}
C	E_{AC}	E_{BC}	TP_C

Classification - 2

- Accuracy is an unreliable metric when the dataset is imbalanced.
  - Consider an extreme example: a dataset containing 99 legitimate (A) + 1 fraudulent (B). If our model classified everything as A then we have 99% accuracy. But B is the "positive" class that our model should be identifying. So we cannot only use accuracy.
- Precision – how many of the predictions are true?
  - True positive (TP) divided by total positive predictions (TP + FP) – minimise FP
  - In the example – Precision is undefined 0/(0+0)
- Recall – how many of the truths were predicted?
  - True positive (TP) divided by total truth (TP + FN) – minimise FN
  - In the example – Recall is 0, 0/(0+1)
- F_1 score
  - Combines precision and recall using the harmonic mean (deals better with outliers)
  - In the example – F_1 is division by zero.

$$\frac{TP}{TP + FP}$$

All the P's

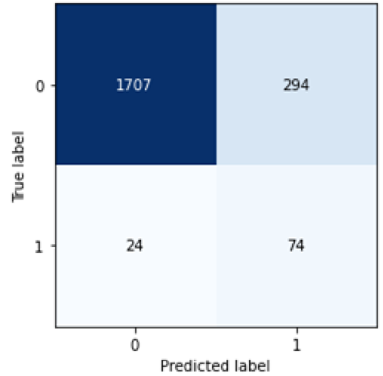
$$\frac{TP}{TP + FN}$$

Replace the last P with an N

$$F_1 = 2 \times \left(\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}\right)$$

Classification - example

- In the OD notebook,
  - Accuracy = 1781/2099 = 0.85
  - Misclassification = 318/2099 = 1 – 0.84 = 0.16
  - **Precision = 74/368 = 0.2**
  - **Recall = 74/98 = 0.76, F_1 = 0.32**



	Actually Positive	Actually Negative
Predicted Positive	(TP) 74	(FP) 294
Predicted Negative	(FN) 24	(TN) 1707

- But we are detecting outliers vs non-outliers, so what is “positive”?
  - Accuracy = 1781/2099 = 0.85 (same)
  - Misclassification = 318/2099 = 1 – 0.84 = 0.16 (same)
  - **Precision = 1707/1731 = 0.99**
  - **Recall = 1707/2001 = 0.85, F_1 = 0.91**

	Actually Positive	Actually Negative
Predicted Positive	(TP) 1707	(FP) 24
Predicted Negative	(FN) 294	(TN) 74

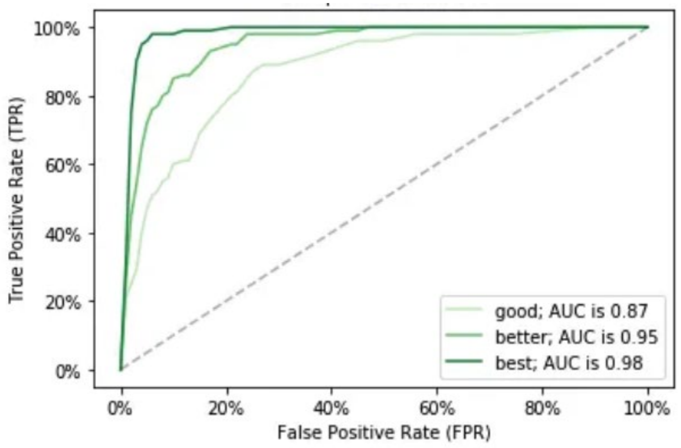
- `print(classification_report(y_test, y_preds))`

```
              precision    recall  f1-score   support

           0       0.99      0.85      0.91      2001
           1       0.20      0.76      0.32        98

    accuracy                           0.85      2099
   macro avg       0.59      0.80      0.62      2099
weighted avg       0.95      0.85      0.89      2099
```

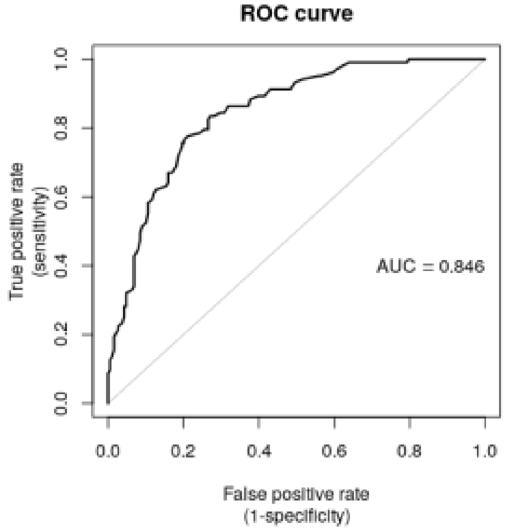
Classification - 3

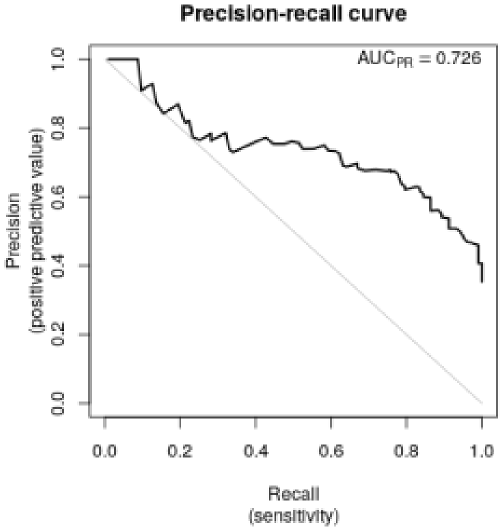


- Receiver (or relative) Operating Characteristic (ROC) curve
  - A probability curve that gets its name from first use in military radar operations
  - Predicting class labels based on the probability of each class (usual threshold=0.5)
  - Trade-off between True Positive Rate (TPR) - y-axis, False Positive Rate (FPR) - x-axis
  - Why “rate”? Measures performance of the classifier at various prediction thresholds
  - TPR (sensitivity/recall) = TP / (TP + FN) and FPR (1 - specificity) = FP / (FP + TN)
  - In other words, at probability { 0.25, 0.5, 0.75…}, how many TP/FP?
  - Each model gets one curve
- Area Under the Curve (AUC-ROC)
  - The curve is quantified using the “area under” - an aggregate measure of performance across all possible thresholds
  - A value between 0 and 1, 0 being an inverse classifier, 0.5 being no-skill line/chance, and 1 being perfect and unrealistic

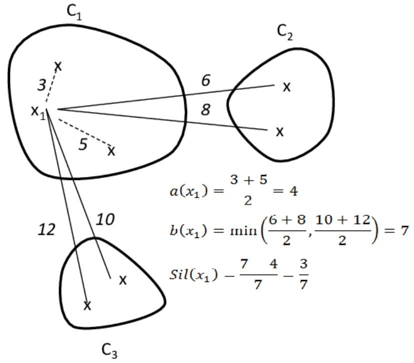
Classification - 4

- ROC curves are reliable when the dataset is balanced (i.e. an equal number of observations for each class), when the dataset is imbalanced, Precision-Recall (PR) curves should be used.
  - Plots Recall on the x-axis and Precision on the y-axis for all prediction thresholds, unlike ROC both axes focus on TP, so Pr curve is more sensitive to the positive class (i.e. the dataset is imbalanced and our interest is in the minority class)
  - In ROC, the top-left indicates good performance, while in a PR curve, it is the top-right.
  - Area under the curve PR AUC is also called Average Precision (AP)
- We know that higher Precision means the model has lower Recall, so we are looking for the point of Recall when Precision starts to drop.





Clustering

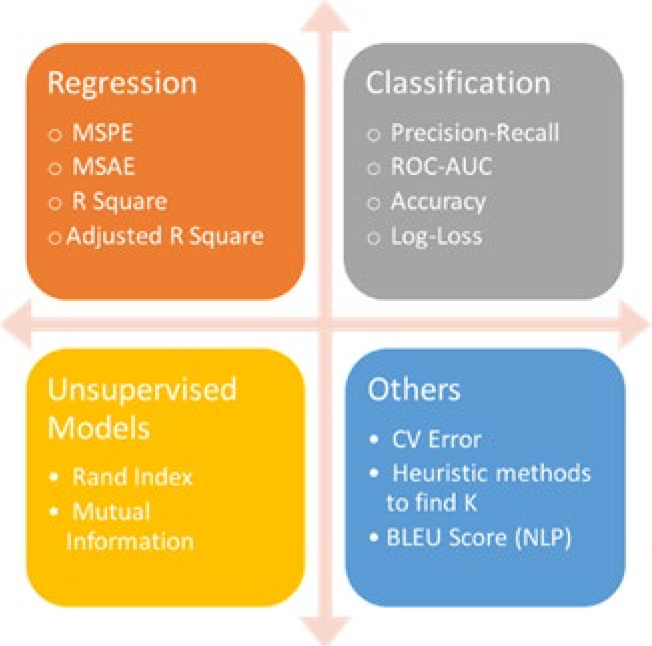


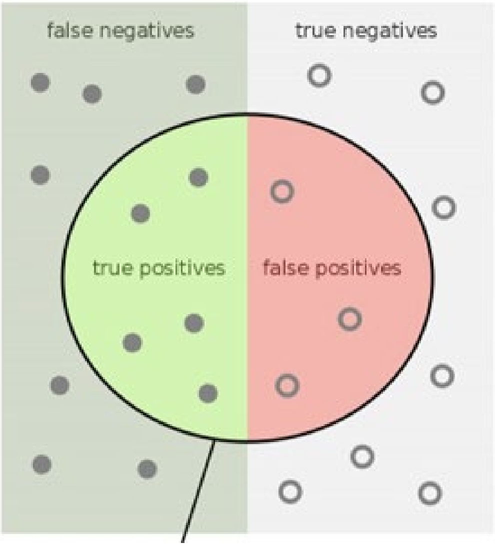
- Challenging because?
- Kleinberg's three axioms (2015),
  - Scale invariance – equal representation of proximity should not impact the clustering outcome
  - Consistency – internal and external proximity should not impact the clustering outcome
  - Richness – flexibility to create any arbitrary clustering
  - But also (equally importantly) evaluation contradicts exploration
- External – Using a labelled sample for homogeneity, completeness, Rand index, mutual information (MI), …..
- Internal - no human involvement – Elbow method (variance), DB-index, Silhouette coefficient, ……
- More intuitively, visualise/tabulate to 'see' if the clustering makes sense (start with k=10….5 or 2)
- https://scikit-learn.org/stable/modules/clustering.html

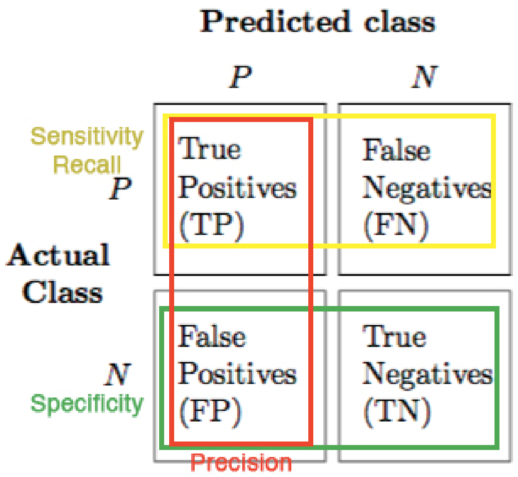
Optimisation

- We already have the solution – a population of them
- Metrics can be problem/algorithm specific, focusing on,
- Convergence – reliably reach an optimal solution (point of convergence)
  - Global vs local convergence
- Robustness – consistent result from any starting point, flow of execution
- Complexity – number of steps to generate a result
- Performance – time and computational cost (function evaluations) to achieve this result

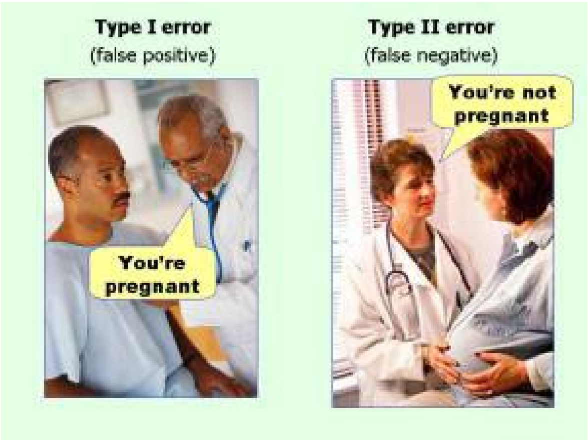
More visually..

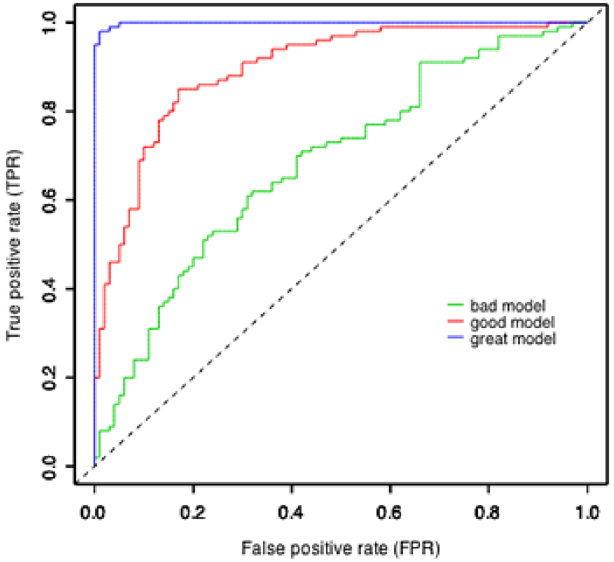






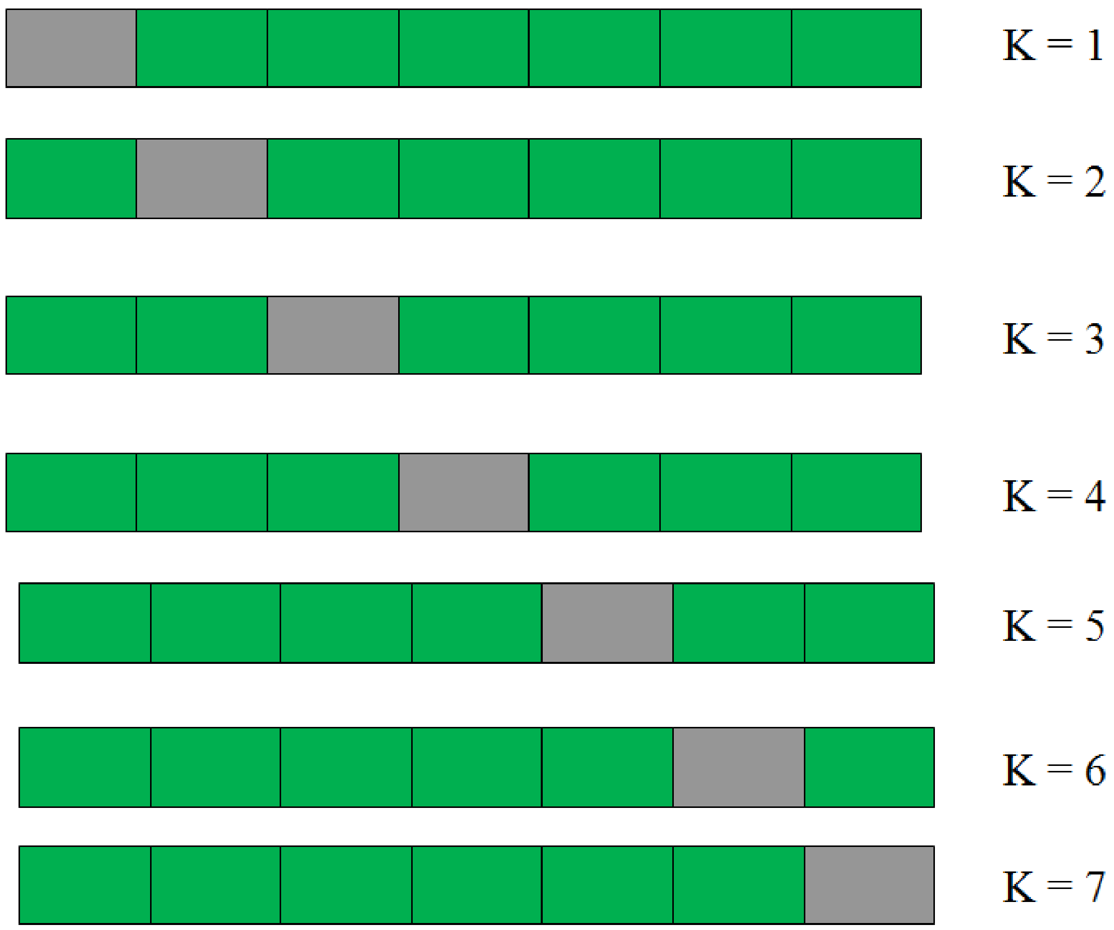
Mean squared error	$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$
Root mean squared error	$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2}$
Mean absolute error	$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n} \lvert e_t \rvert$
Mean absolute percentage error	$\text{MAPE} = \frac{100\%}{n}\sum_{t=1}^{n} \left\lvert \frac{e_t}{y_t} \right\rvert$





https://www.kdnuggets.com/2018/06/right-metric-evaluating-machine-learning-models-2.html

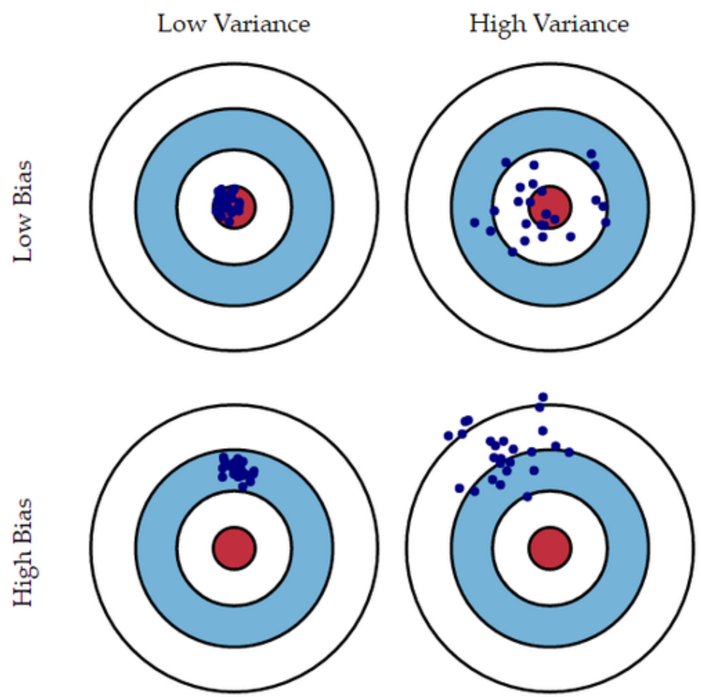
k-fold cross-validation



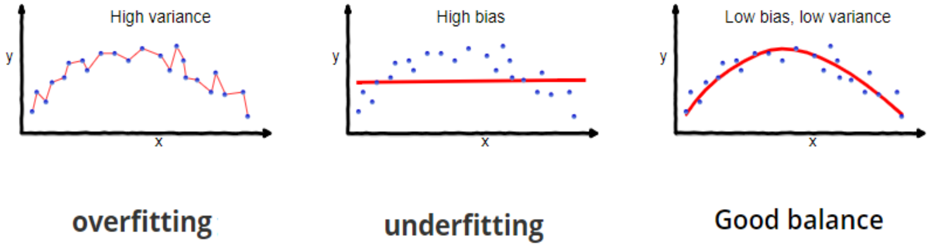
- K=7
- Divide the entire dataset into 7 equal subsets
- In the first iteration, use 6 subsets to train (green) and validate on k=1 (grey)
- Second iteration validates on k=2, trains on the rest.
- Repeat up to k=7, aggregate performance over all k.
- Each iteration, one of the k subsets is used for validation and the other k-1 subsets form the training set.
- Reduces bias as most of the data is learned, reduces variance as most of the data is also used for validation.

Bias and variance (underfitting/overfitting)

- Bias – average difference between predicted values and actual values.
  - Predictions in the training dataset itself are inaccurate - trying to force a linear model on nonlinear data
  - Every algorithm starts with some level of bias, but a high bias misses important correlations between input and output variables.
  - A high bias means that the model is too simple and does not capture the complexity of data, thus **underfitting** the training dataset.

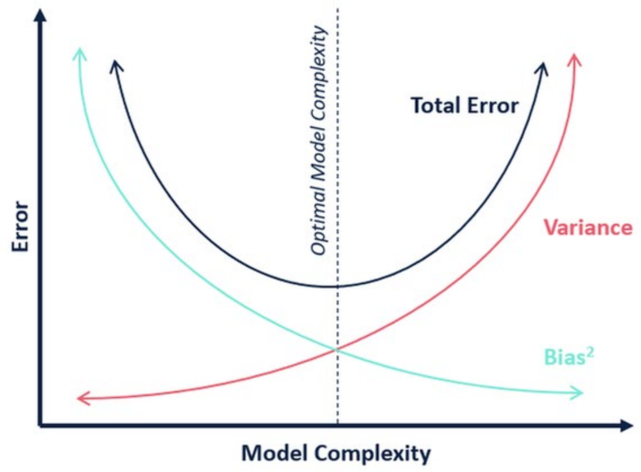


- Variance: the distribution of predicted values from actual values.
  - Variance occurs when the model fails on the test dataset
  - High variance means that the model has captured random noise present in the training data (instead of actual patterns), thus **overfitting** the training dataset



Bias-variance trade-off

- Balance underfitting and overfitting
- Rich enough to express underlying structure in data and simple enough to avoid fitting noise/spurious patterns.
- Address underfitting:
  - Select a non-linear (more complex) algorithm (xgboost to deep learning)
  - Increase model complexity – # features, # parameters, training time
- Address overfitting:
  - More training data – k-fold cross-validation
  - Regularisation – remove layers, node dropout, tree pruning
  - Reduce training time

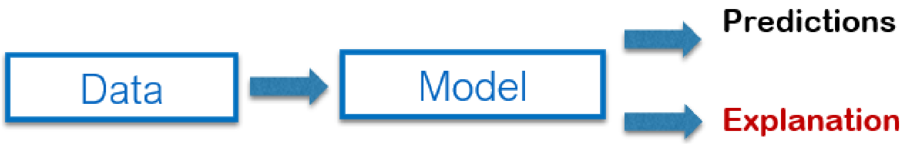


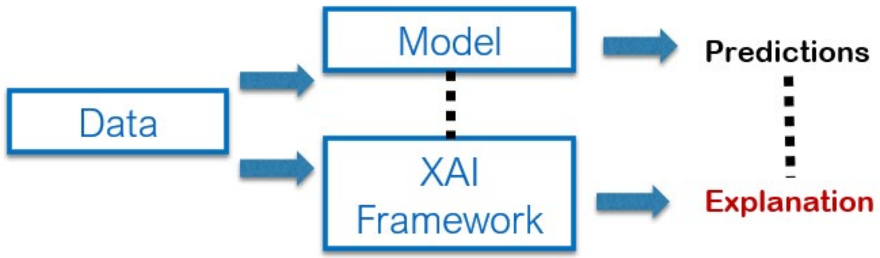
Infographic - https://elitedatascience.com/bias-variance-tradeoff

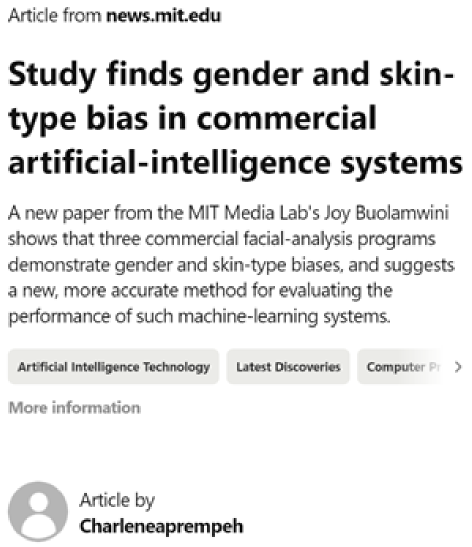
Lifecycle phases 13

13. Develop and improve model explainability

- Explainable AI (XAI)
- The other kind of "bias"
- Data bias - an algorithm is only as good as the data it learns from
- AI models are opaque (not ~~black box~~) – cannot explain how it reached the outcome/decision.
- Poor combination as opaque models will reflect human biases and prejudices
- EU GDPR - a right of explanation for "meaningful information of the logic involved when automated decision-making takes place."
- Broadly two approaches, 1) explainable by design (?) 2) explainable outcomes






Article from news.mit.edu

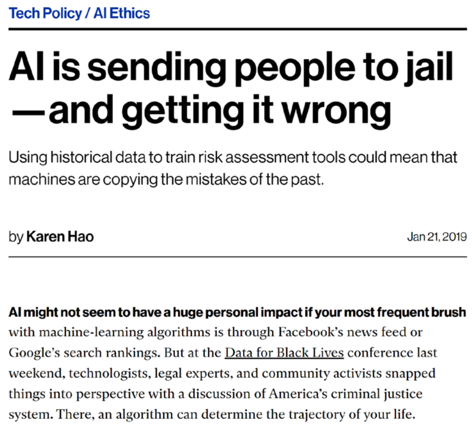
Study finds gender and skin-type bias in commercial artificial-intelligence systems

A new paper from the MIT Media Lab's Joy Buolamwini shows that three commercial facial-analysis programs demonstrate gender and skin-type biases, and suggests a new, more accurate method for evaluating the performance of such machine-learning systems.

Artificial Intelligence Technology | Latest Discoveries | Computer P

More information

Article by
Charleneaprempeh


Tech Policy / AI Ethics

AI is sending people to jail—and getting it wrong

Using historical data to train risk assessment tools could mean that machines are copying the mistakes of the past.

by Karen Hao Jan 21, 2019

AI might not seem to have a huge personal impact if your most frequent brush with machine-learning algorithms is through Facebook's news feed or Google's search rankings. But at the Data for Black Lives conference last weekend, technologists, legal experts, and community activists snapped things into perspective with a discussion of America's criminal justice system. There, an algorithm can determine the trajectory of your life.

XAI techniques

- Sensitivity Analysis
  - Analyse the effect of each input feature on model output.
  - Use Partial Dependence Plots (PDP) or Individual Conditional Expectation (ICE) plots to provide a graphical explanation.
- Local Interpretable Model Explanation (LIME)
  - Finds feature importance by capturing feature interaction (correlation and covariance analysis) between features and output using a linear model.
- Shapley Additive Explanations (SHAP)
  - An additive feature attribution method that assigns a value to each feature for the expected AI outcome.
  - The higher the value, the larger the feature's attribution to the specific AI outcome.
- Many others, an emerging field of research

Lifecycle phases 14-15

14. Deploy models

- Also known as model serving, model scoring, production
- Key responsibility of ML/AI engineers
- Adding value to the organisation:
    - Real-time vs batch execution/prediction
    - Number of end-users, applications
    - Expected format/s of output, turnaround time, frequency of use

15. Operationalise using AI pipelines

- MLOps, AIOps.. Like DevOps
- Includes deployment and.. versioning, auditing, retraining, maintenance, and monitoring
- Data pipeline – availability, collection, storage, pre-processing, versioning, ethics
- ML pipeline – model compression, device compatibility, CI/CD
- Debugging ML - Cross-functional complexity, silent errors (predicts, but incorrect), latency of full AI cycle for each fix

Lifecycle phases 16-17

16. Hyper-automation processes and systems

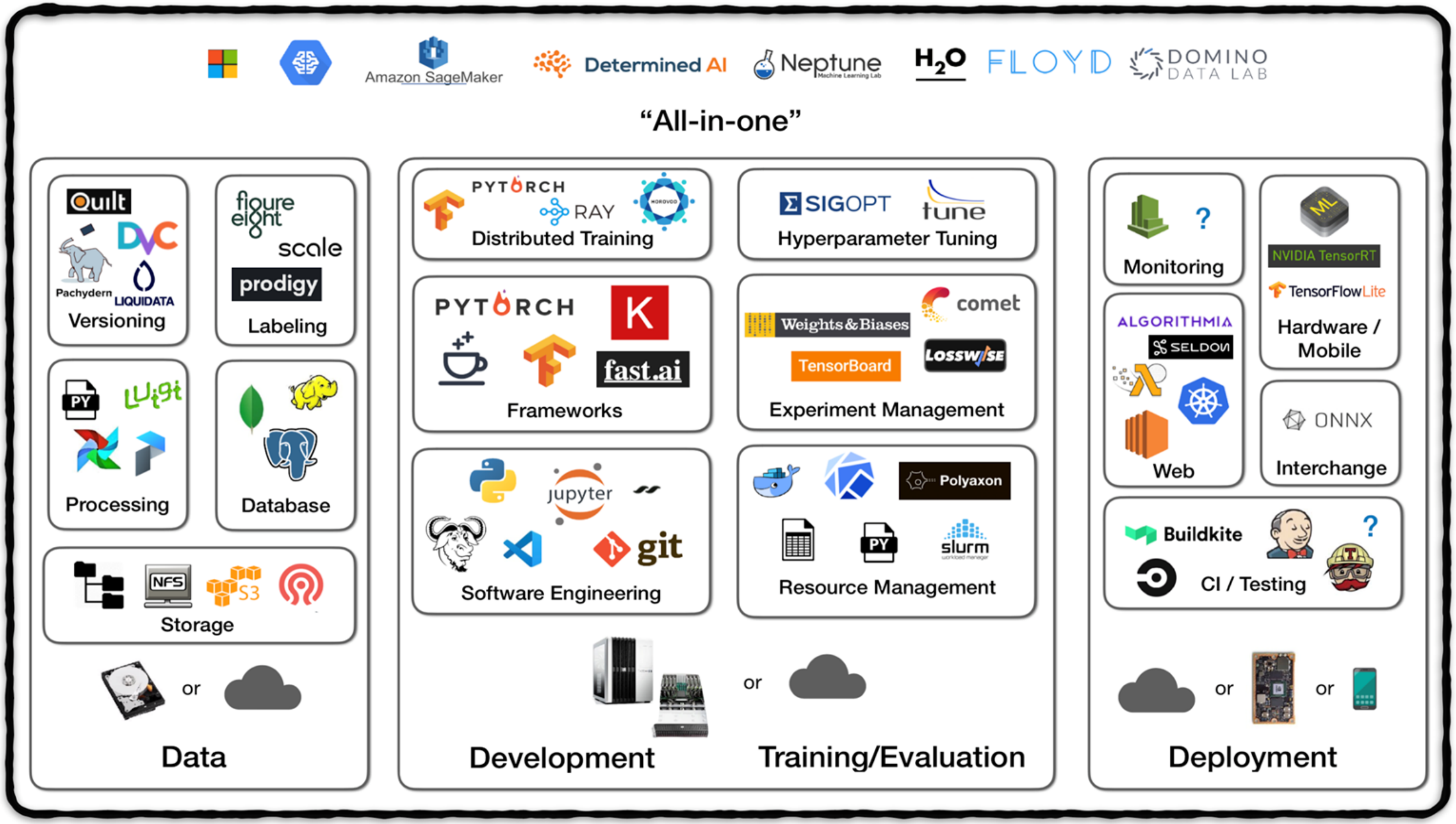
- Hyper-automation (or intelligent automation) is the integration of AI capabilities with automated components in processes and systems
- Demonstrate capabilities of the AI solution to downstream/upstream process owners and stakeholders
- Conduct a pilot phase of hyper-automation
- Evaluate the outcomes of the process/workflow due to hyper-automation
- Consider options for gains in efficiency (improve), effectiveness (enhance) or innovation (disrupt/transform)

17. Evaluate and monitor results

- Model drift – decreasing accuracy (re-train)
- Model staleness – changing data/environment (re-model)
- End-user – adoption, documentation, feedback
- ROI – reduced costs (time, effort, skill)
- ROI - increased revenue, new revenue streams, increased market share
- ROI – intangible (reduced errors, increased productivity, reduced turnover)

Optus

Pipeline tools



Sergey Karayev at Full Stack Deep Learning Bootcamp 2019, https://fullstackdeeplearning.com/