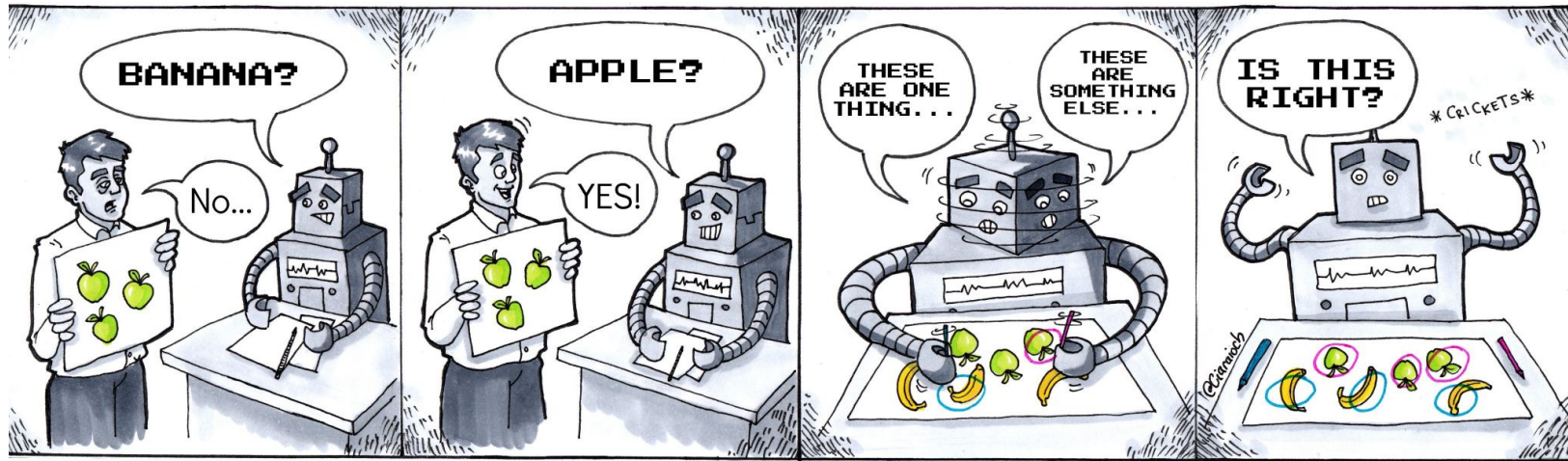


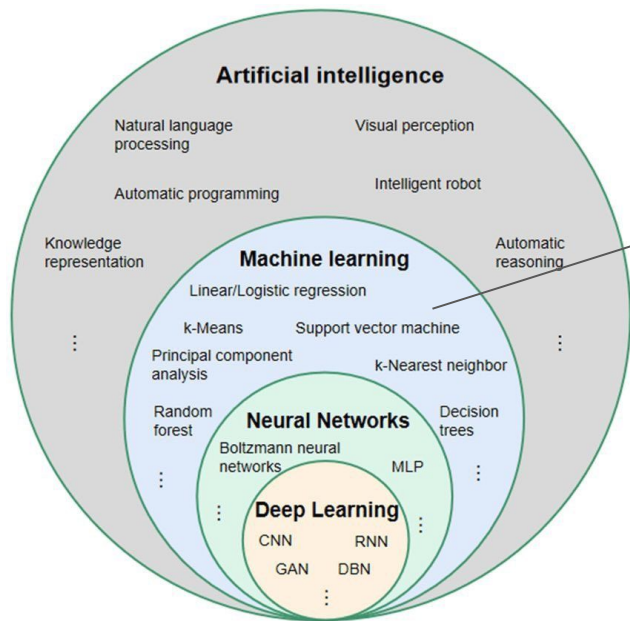
Machines that Learn



Supervised Learning

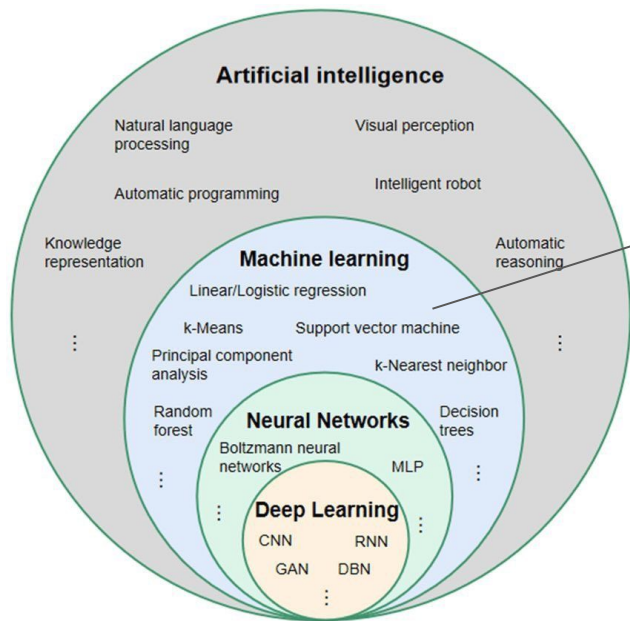
Unsupervised Learning

Recap



Supervised Learning
Unsupervised Learning
Reinforcement Learning

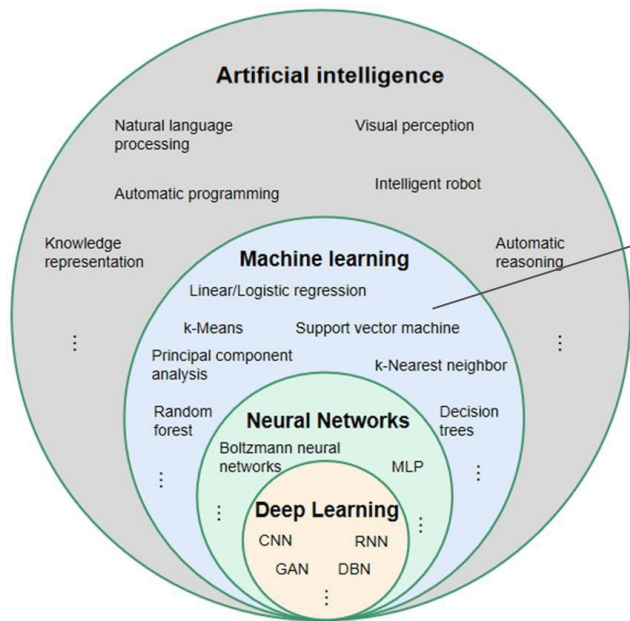
Recap



Supervised Learning
Self-Supervised Learning
Few-shot Learning
Zero-shot Learning
Transfer Learning
Unsupervised Learning
Reinforcement Learning

Federated Learning

Recap



Supervised Learning

Self-Supervised Learning

Few-shot Learning

Zero-shot Learning

Transfer Learning

Unsupervised Learning

Reinforcement Learning

Federated Learning

Recap

Supervised Learning

$$f(x) = y$$

Recap

Supervised Learning

$$f(x) = y$$



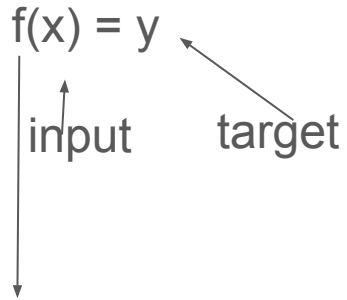
input



target

Recap

Supervised Learning



Machine learning algorithm i.e logistic classification, logistic regression, naive bayes, etc...

Recap

Supervised Learning

$$f(x) = y$$

input

target

Tries to match the prediction to the targets

We have the targets

Machine learning algorithm i.e logistic classification, logistic regression, naive bayes, etc...

Recap

Supervised Learning

$$f(x) = y$$

Getting this

P.s we don't always have labels readily available

Making sure the data is good

Finding the most suitable algorithm and training it

Recap

Supervised Learning

$$f(x) = y$$

```
graph LR; A["f(x) = y"] --> B["Getting this"]; A --> C["Evaluate if the result makes sense and is relevant for the given use-case"]; A --> D["Making sure the data is good"]; A --> E["Finding the most suitable algorithm and training it"];
```

The diagram illustrates the supervised learning process. It starts with a central box containing the equation $f(x) = y$. From this box, four arrows point outwards to different stages of the process: a horizontal arrow to the right, a curved arrow pointing down and to the right, a diagonal arrow pointing down and to the left, and a vertical arrow pointing straight down.

Getting this

Evaluate if the result makes sense and is relevant for the given use-case

Making sure the data is good

Finding the most suitable algorithm and training it

Recap

Supervised Learning

$$f(x) = y$$

```
graph LR; A["f(x) = y"] --> B["Getting this"]; A --> C["Evaluate if the result makes sense and is relevant for the given use-case"]; A --> D["Making sure the data is good"]; A --> E["Finding the most suitable algorithm and training it"];
```

The diagram illustrates the supervised learning process. It starts with a box containing the equation $f(x) = y$. From this box, four arrows point to different stages of the process: a long horizontal arrow to 'Getting this', a curved arrow to 'Evaluate if the result makes sense and is relevant for the given use-case', a diagonal arrow to 'Making sure the data is good', and a vertical arrow to 'Finding the most suitable algorithm and training it'.

Getting this

Evaluate if the result makes sense and is relevant for the given use-case

Making sure the data is good

Finding the most suitable algorithm and training it

P.s All of this without making it useful and easy to use does not mean anything.

Recap

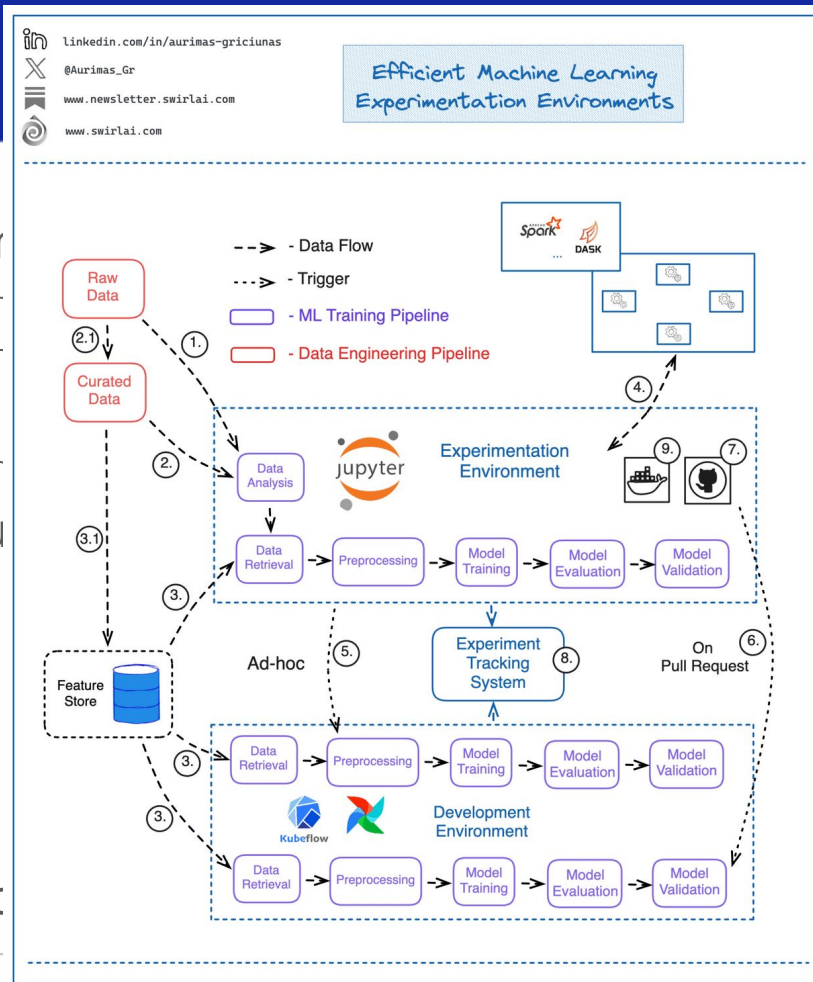
Supervised Learning

$$f(x) = y$$

Making suggestions

Finding the most successful

Pushing All of this with



Evaluate if the result makes sense and is relevant for the given use-case

es not mean anything.

Supervised Learning

This class will be about

- Being empiricists
- (more) Exploratory Data Analysis (EDA)
- Feature Engineering (more data wrangling)
- Training Models
- Evaluation

Supervised Learning

If machines can learn by themselves, just give it data.
Shouldn't it be easy ?

Supervised Learning

If machines can learn by themselves, just give it data.
Shouldn't it be easy ?



Supervised Learning

How do we separate good from garbage ?

Supervised Learning

How do we separate good from garbage ?
How do we know if it smells ?



<https://ideogram.ai/>

Supervised Learning

How do we separate good from garbage ?

How do we know if it smells ?

Exploratory data analysis, explore correlation and quickly build baseline models

Supervised Learning

How do we separate good from garbage ?

How do we know if it smells ?

Things to aim for when working on a dataset:

- Complete
- Consistent
- Relevant
- Representative
- Accurate

Supervised Learning

How do we separate good from garbage ?

How do we know if it smells ?

Things to aim for when working on a dataset:

- **Complete** - Ensure all necessary data points are included and address missing data appropriately.
- **Consistent** - Maintain uniform formats and values throughout the dataset to avoid contradictions.
- **Relevant** - Include only data directly aligned with your analysis goals or application needs
- **Representative** - Ensure the dataset accurately reflects the population or phenomenon being studied.
- **Accurate** - Verify data correctness to minimize errors or misleading insights

Supervised Learning

How do we separate good from garbage ?

How do we know if it smells ?

Things to aim for when working on a dataset:

- Complete
- Consistent
- Relevant
- Representative
- Accurate

Red flags

- Too many empties or NaN
- Outliers
- Biases
- Noisy Data
- Weird things

Supervised Learning

Data → Features

Supervised Learning

Feature Engineering

- Categorical Encoding
- Handle Missing Values
- Feature Scaling
- Creating new features

Data → Features

Supervised Learning

Feature Engineering

- Categorical Encoding
 - one-hot encoding
 - multi-hot encoding

Data → Features

Supervised Learning

Feature Engineering

Data → Features

- Categorical Encoding
 - one-hot encoding
 - multi-hot encoding

Some ways to do it:

- [df.get_dummies](#)
- [sklearn one hot encoder](#)
- [sklearn multi hot encoding](#)

Supervised Learning

Feature Engineering

- Handle Missing Values

Data → Features

Supervised Learning

Feature Engineering

Data → Features

- Handle Missing Values
 - Drop
 - Imputation (Mean, Median, Mode)
 - Forward Fill and Backward Fill
 - Random Values

[Top 4 Techniques for Handling Missing Values in Machine Learning](#)

Supervised Learning

Feature Engineering

- Feature Scaling
 - Normalization
 - Standardization

Data → Features

Supervised Learning

Feature Engineering

Data → Features

- Creating new features based on existing data

Supervised Learning

Cool ?

Supervised Learning



Supervised Learning

Data understood.
Features created.
Now it's time to train.



Supervised Learning

Linear Regression
Logistic Regression



Supervised Learning

Linear Regression

Logistic Regression

Naive-Bayes

KNN (k-nearest neighbors)

SVM (Support Vector Machines)



Supervised Learning

Naive-Bayes

Naive Bayes is a **probabilistic** machine learning algorithm based on **Bayes' Theorem**, which calculates the **probability** of a class given a set of features. It assumes that all features are independent (the "naive" assumption), which simplifies computation but may not hold true in all real-world cases. Despite this limitation, it is highly effective for **text classification** and **spam filtering** because it works well with **high-dimensional data** and **provides fast predictions**.

Reference: [Sklearn Naive Bayes](#)

Supervised Learning

Naive-Bayes

Naive Bayes is a **probabilistic** machine learning algorithm based on **Bayes' Theorem**, which calculates the **probability** of a class given a set of features. It assumes that all features are independent (the "naive" assumption), which simplifies computation but may not hold true in all real-world cases. Despite this limitation, it is highly effective for **text classification** and **spam filtering** because it works well with **high-dimensional data** and **provides fast predictions**.

Reference: [Sklearn Naive Bayes](#)

Challenge: Try to implement this algorithm from scratch

Supervised Learning

K-Nearest Neighbors (KNN)

KNN is a simple, instance-based algorithm that **predicts the class** of a data point by **considering the classes of its k closest neighbors in the feature space**. The "closeness" is typically determined using distance metrics like **Euclidean distance**. KNN is intuitive and non-parametric, meaning it makes no assumptions about the underlying data distribution, but it can **become computationally expensive as the dataset grows**.

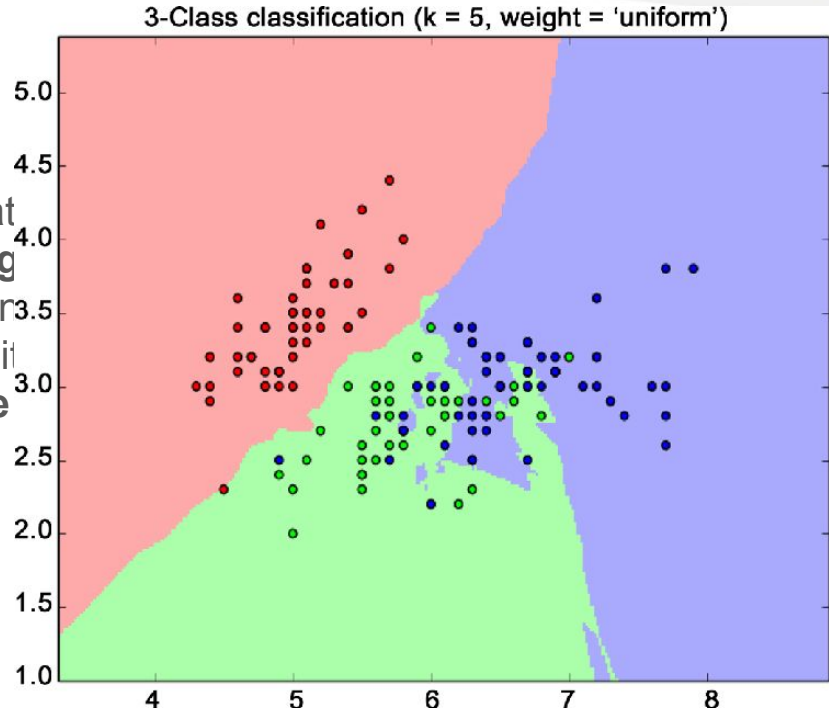
Reference: [Sklearn KNN](#)

Supervised Learning

K-Nearest Neighbors (KNN)

KNN is a simple, instance-based algorithm that **considering the classes of its k closest neighbors**. "closeness" is typically determined using distance metrics. KNN is intuitive and non-parametric, meaning it does not assume an underlying data distribution, but it can **become inefficient** as the **dataset grows**.

Reference: [Sklearn KNN](#)



Supervised Learning

Support Vector Machines (SVM)

SVM is a powerful algorithm used for **classification** and **regression** tasks. It works by finding a **hyperplane that best separates data points** of different classes in a **high-dimensional space**. SVM focuses on maximizing the margin between the classes, making it robust to outliers and well-suited for both linear and non-linear data (with the help of kernel functions). **It is especially effective for datasets with clear margins of separation.**

Reference: [Sklearn SVM](#)

Let's do some feature engineering

More EDA
Feature Engineering
More training



Let's do some feature engineering

More EDA
Feature Engineering
More training



Supervised Learning

Remember when I said we need to be empiricists ?

And act like scientists.

We need to create hypothesis and properly evaluate them.



Supervised Learning

Data = Train + Testing

Testing = Validation + Testing

Data = Train + Validation + Testing



Supervised Learning

Data = Train + Testing

Testing = **Validation** + Testing

Data = Train + **Validation** + Testing

Validation - used during development to validate hyperparameters (like max_iter of logistic regression)

* more on this on the next class



Supervised Learning

Data = Train + Testing

Testing = Validation + Testing

Data = Train + Validation + Testing

Sidenote:

In real world use-cases data is not stagnant.

Any ML models need to be constantly re-evaluated
and more often than not re-trained.



Supervised Learning

Hypothesis

We can learn a reasonable enough valid correlation between X and Y

Testing

How close did we get ?
Is it close enough ?



Supervised Learning

Testing results are only considered correct if there are no biases* during training or evaluation - i.e no **data leakage**

*data leakage is just one possible bias, we also can have selection bias, sampling bias among others



[Understanding Bias in Machine Learning Models - Arize AI](#)

Supervised Learning

Testing results are only considered correct if there are no biases during training or evaluation - i.e no **data leakage**

This means that testing data, *ideally* is locked away.



Supervised Learning

Testing results are only considered correct if there are no biases during training or evaluation - i.e no **data leakage**

This means that testing data, *ideally* is locked away.

Which means that certain **transformations** must be done **only after splitting the data** e.g Normalization, Standardization, TF-IDF,...



Supervised Learning

Hypothesis

We can learn a reasonable enough valid correlation between X and Y

Testing

How close did we get ?

Is it close enough ?



Supervised Learning

Evaluation is looking closely at the right metrics.

Let's go back to titanic and store sales.



Supervised Learning

Hypothesis

We can learn a reasonable enough valid correlation between X and Y

Testing

How close did we get ?

Is it close enough ?



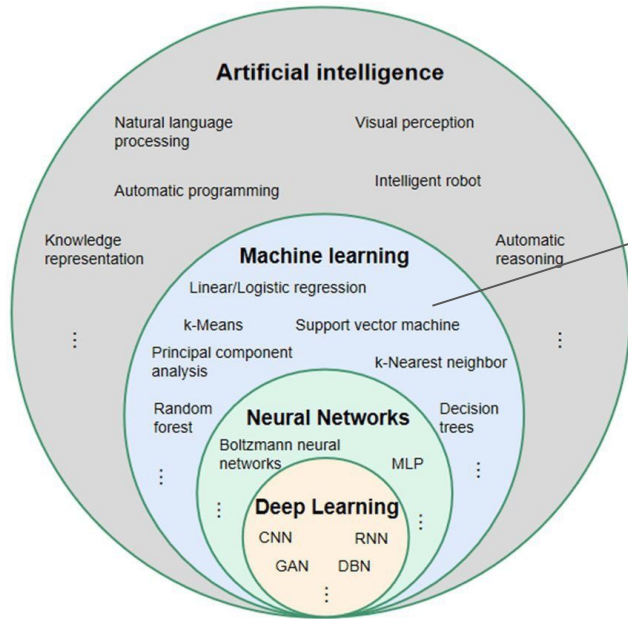
Supervised Learning

Building blocks

- [Pipeline — scikit-learn 1.5.2 documentation](#)
- [sklearn.datasets — scikit-learn 1.5.2 documentation](#)
- [sklearn.preprocessing — scikit-learn 1.5.2 documentation](#)



Where we are



Supervised Learning

Self-Supervised Learning

Few-shot Learning

Zero-shot Learning

Transfer Learning

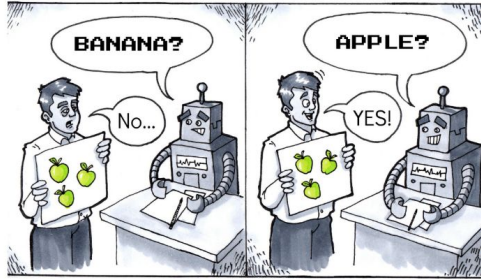
Unsupervised Learning

Reinforcement Learning

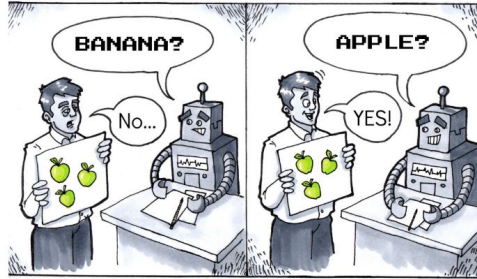
Federated Learning

Machines that Learn

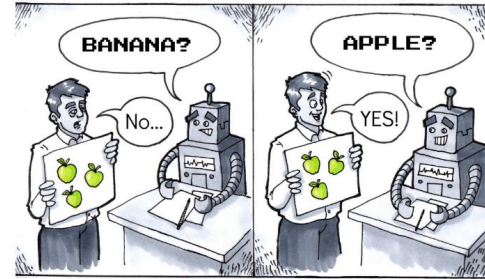
More on the next class



Supervised Learning



Supervised Learning



Supervised Learning

Hyperparameter Search
Cross-Validation