

Data Science Engineering Methods and Tools

Lecture 1

Northeastern University
College of Engineering

INFO 6105 – Spring 2024
Abdolreza Mosaddegh

Course Overview

- Course Title: Data Science Engineering Methods and Tools
- Course Number: INFO 6105
- Term and Year: Spring 2024
- Credit Hour: 4
- Course Format: On-Ground
- Instructor: Abdolreza Mosaddegh
- Email : a.mosaddegh@northeastern.edu
- TA: Shubham Krishnakumar Nair
- TA Email: nair.shu@northeastern.edu
- Assignments and Resources: Canvas

About this course

Introduces the fundamental concepts and techniques for data science engineering including data preprocessing, analytical models, and visualization methods. Discusses a variety of machine learning algorithms in supervised learning and unsupervised learning based on real-world applications.

- Introduction to Data Science
- Data Preprocessing
- Linear Classifiers
- Non-Linear Classifiers
- Decision Trees
- Ensembles and Super learners
- Dimensionality Reduction
- Clustering Methods
- Association Rules
- Introduction to Neural Networks / Deep Learning
- Introduction to Big-data Analysis

Course Prerequisites

- Graduate Level CSYE 6200 Minimum Grade of B- or Undergraduate Level INFO 5100 Minimum Grade of B- or Graduate Level INFO 5100 Minimum Grade of B-
- Statistics and Linear Algebra
- Basic Programing proficiency
 - Python is used as primary programing languages in this class but other common programing languages such as R, Java, and C# are also acceptable.

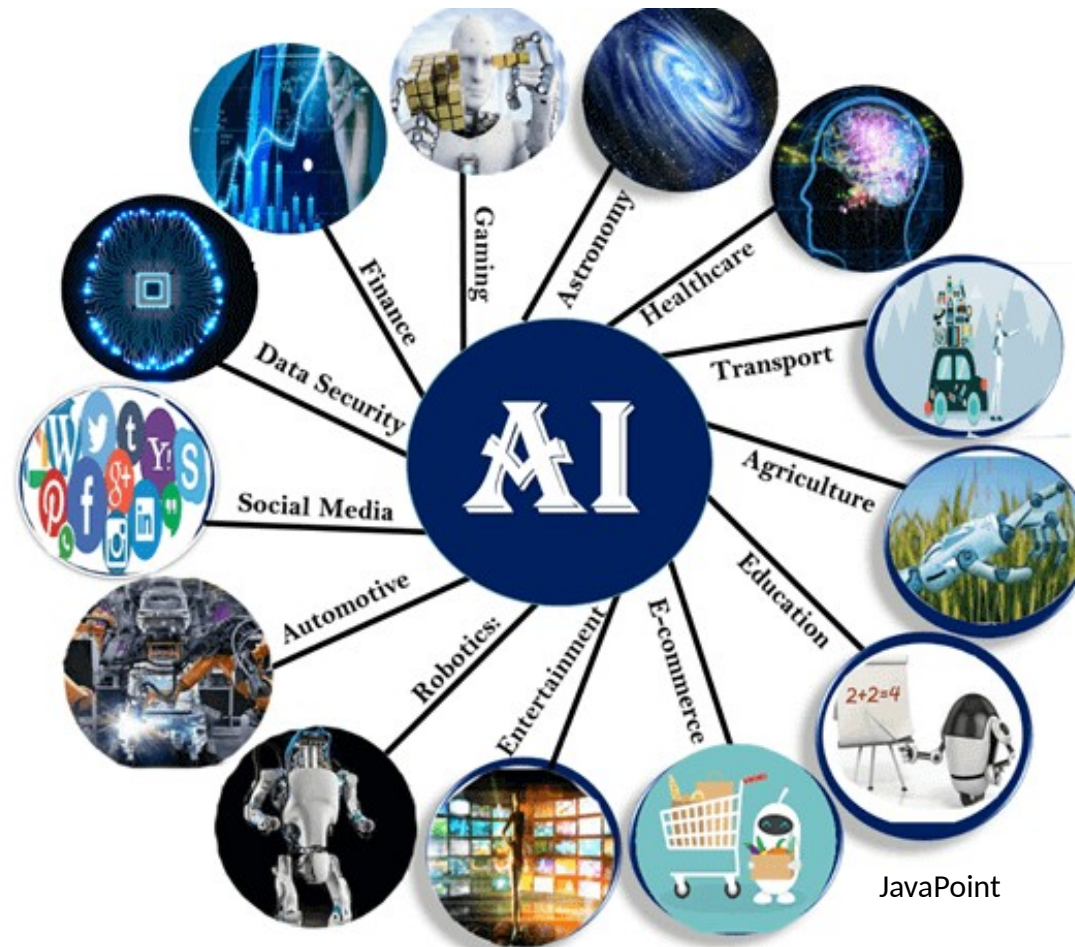
Grading

- Assignments: 40%
- Midterm Exam: 10%
- Final Exam: 20%
- Final Project: 30%

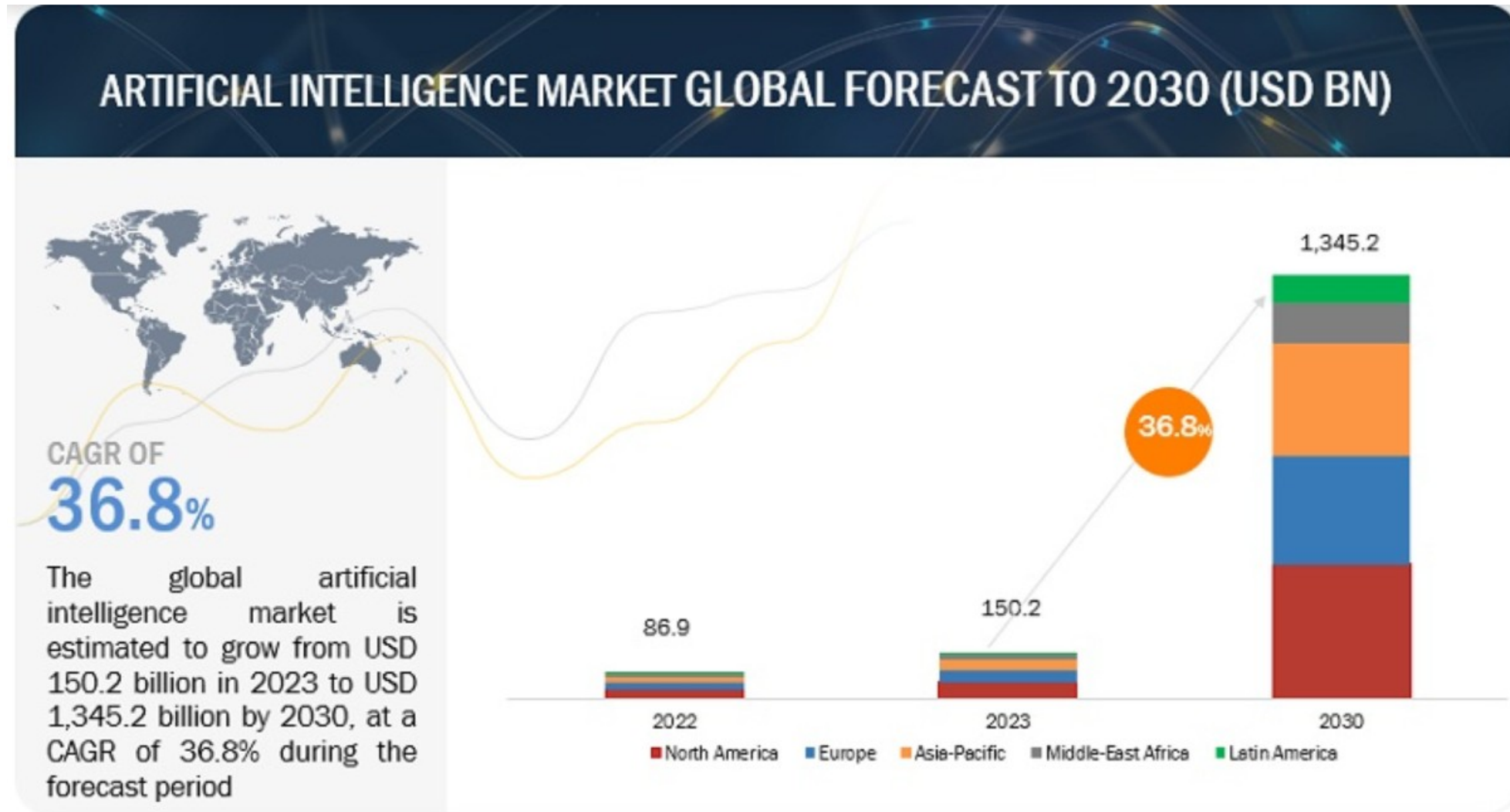
AI is the new electricity

Electricity changed how the world operated. It upended transportation, manufacturing, agriculture, health care. AI is poised to have a similar impact

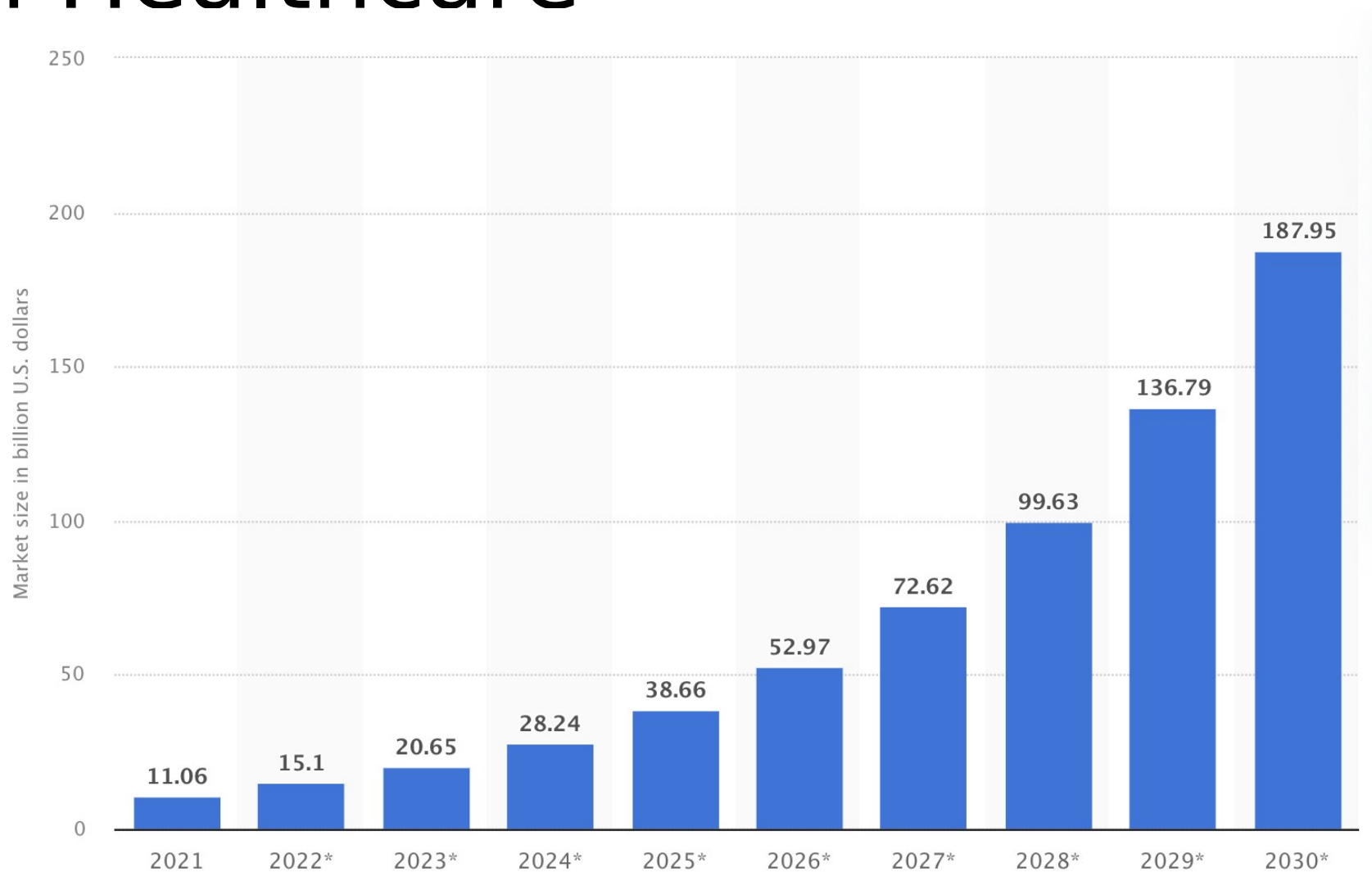
Andrew NG



Market Size of AI

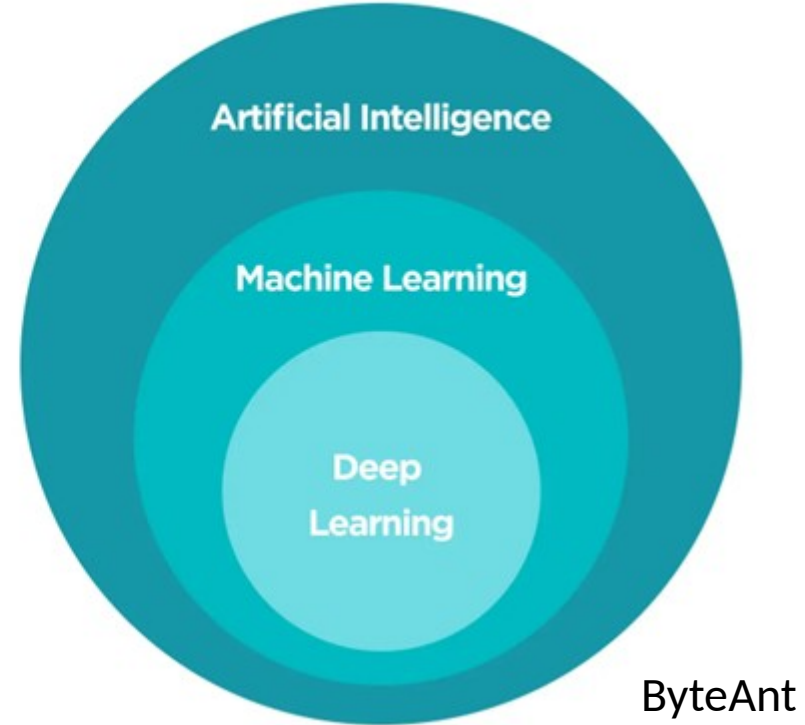


AI in Healthcare



Stewart

- **Artificial Intelligence** - a field of computer science enables machines to imitate human intelligence
- **Machine Learning** - brach of AI provides machines the ability to automatically learn
- **Deep Learning** - subfield of machine learning based on artificial neural networks



“Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.”

Arthur Samuel

AI does not always imply a learning-based system
e.g., Rule based system

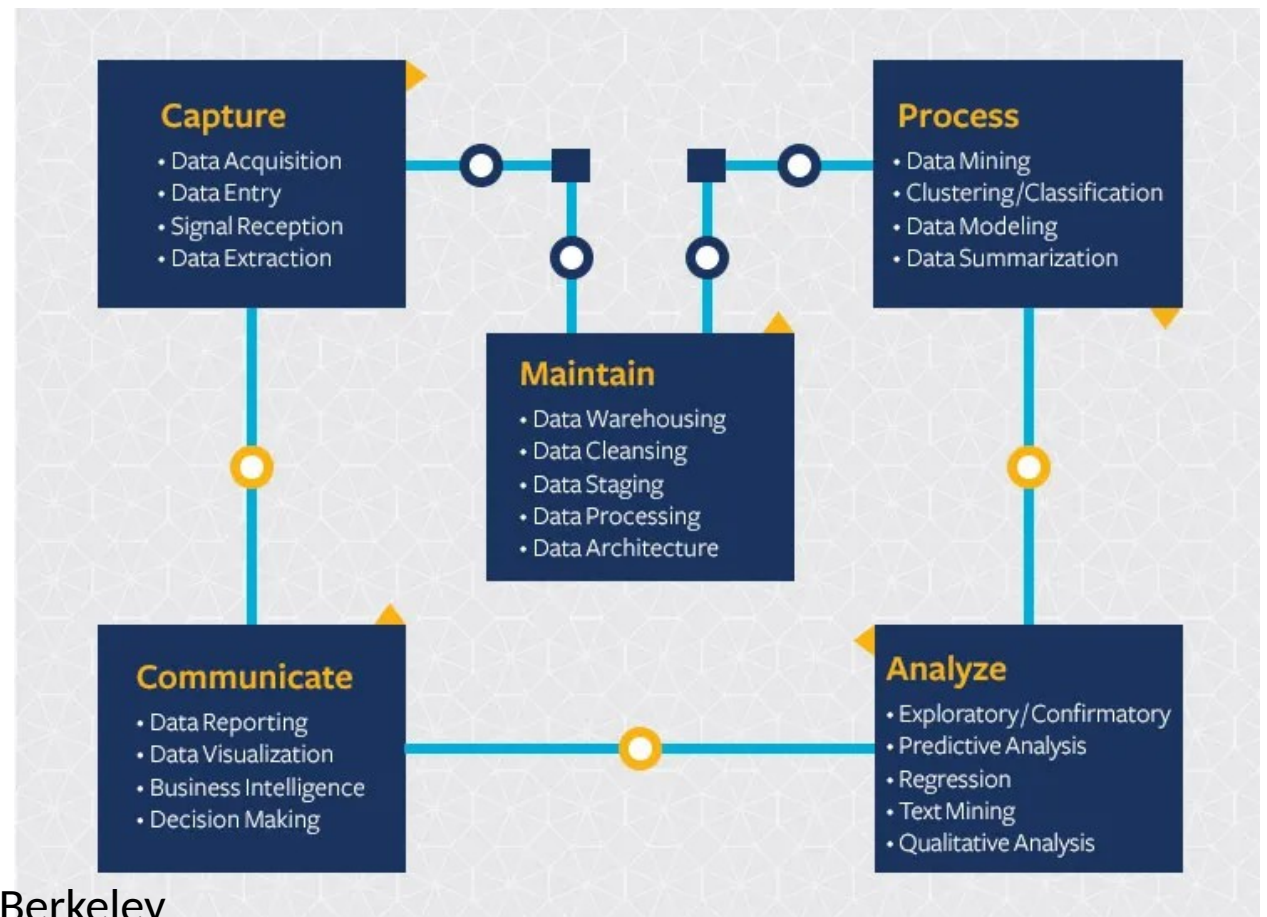
Why AI not human ?

- The problem **size is too vast** for human reasoning, calculating and memorizing capabilities
- The problem needs to be addressed with **minimum latency**
- The problem needs an **accuracy rate** which is not pragmatic for humans
- The problem needs a lot of human resources, and it is possible to **reduce costs** using AI
- The problem can be addressed by humans, but **AI makes life easier !**

What is data science?

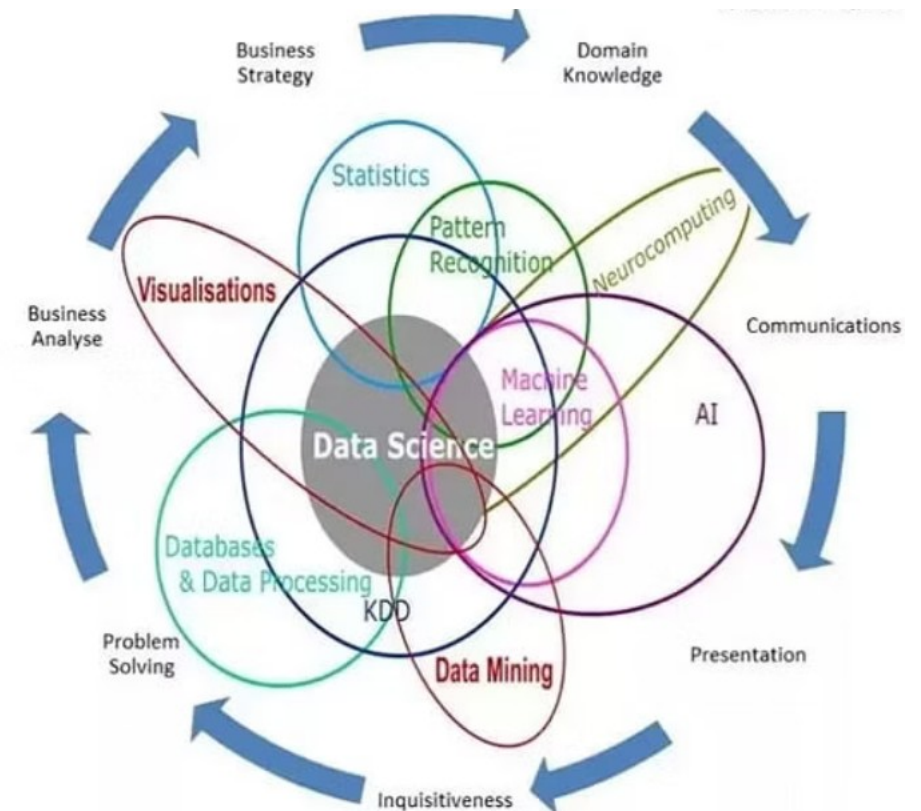
“The ability to take data — to be able to understand it, to process it, to extract value from it, to visualize it, to communicate it.”

Hal Varian

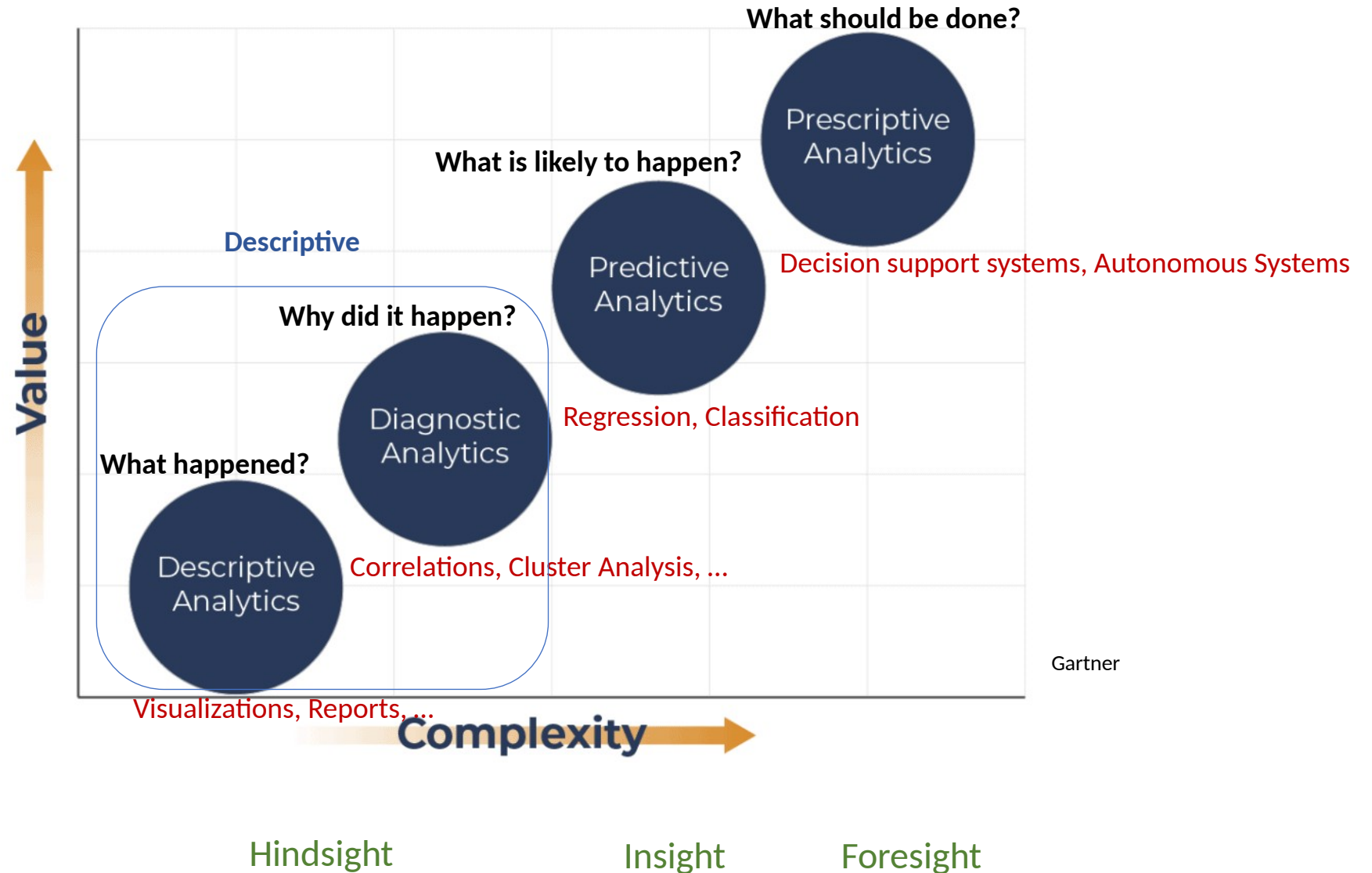


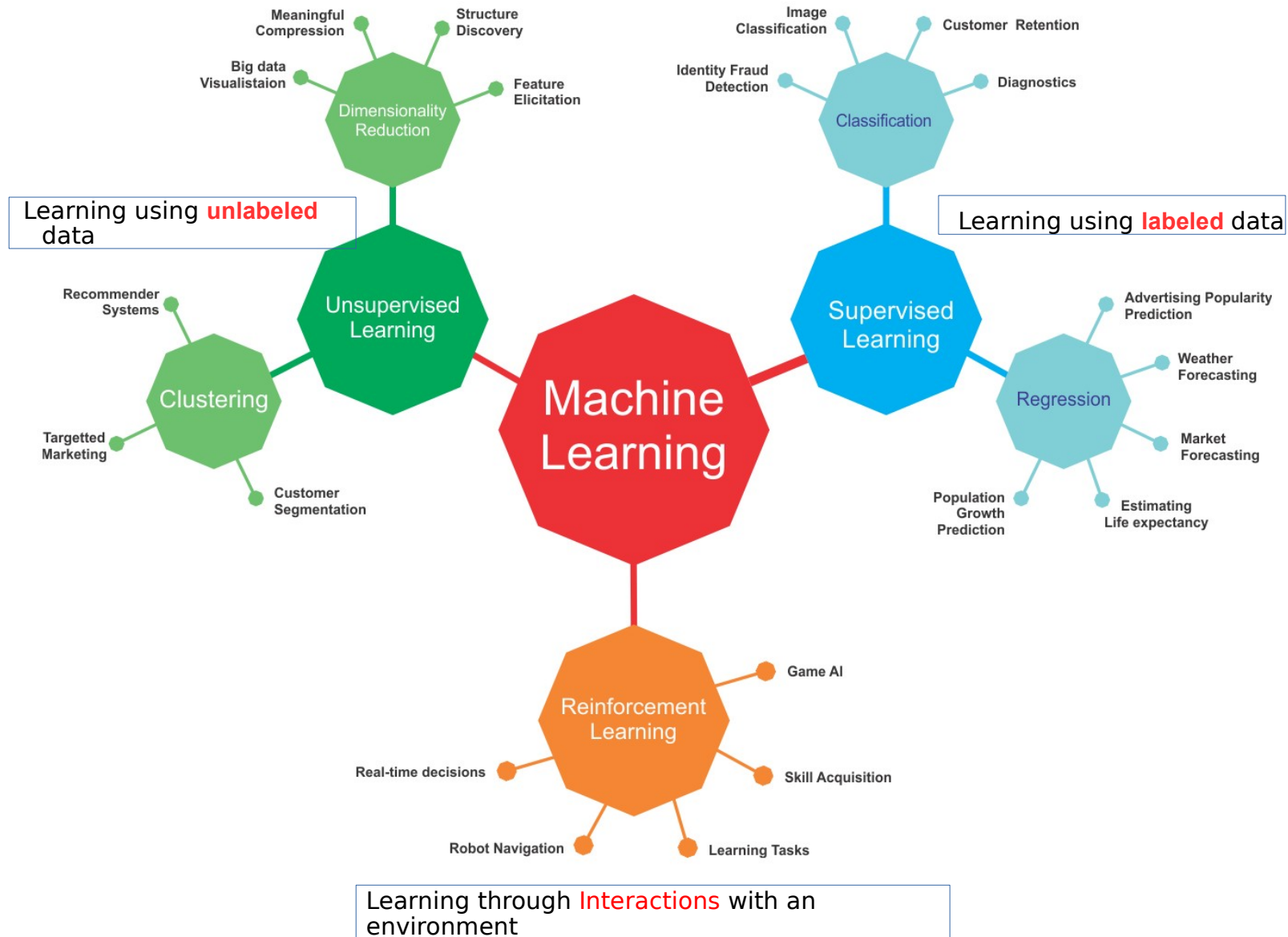
Data Science vs. Machine Learning

Data science is a **broad term** not only focuses on algorithms and statistics but also takes care of the entire data processing methodology includes data cleansing, and visualization. Machine learning is used in data science to help discover patterns in processing and analyzing data.



Analytics



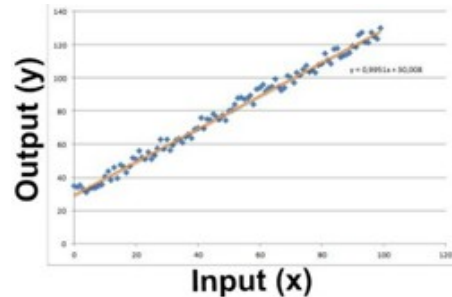


Some Supervised Approaches

Regression

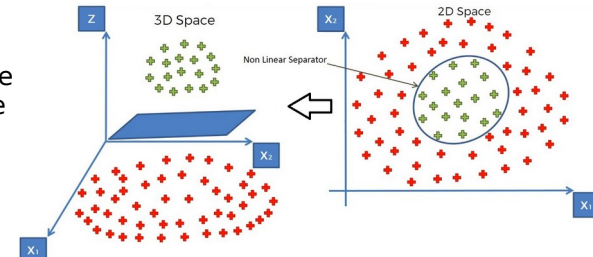
Learn a line/curve (the model) using training data consisting of Input-output pairs. Use it to **predict the outputs for new inputs**

Linear Regression
Logistic Regression,
Support Vector Machines



SVM

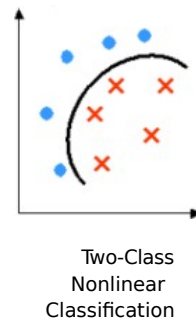
Support Vector Machine (SVM) models have the ability to perform a non-linear regression / classification by **mapping their inputs into high-dimensional feature spaces**



Classification

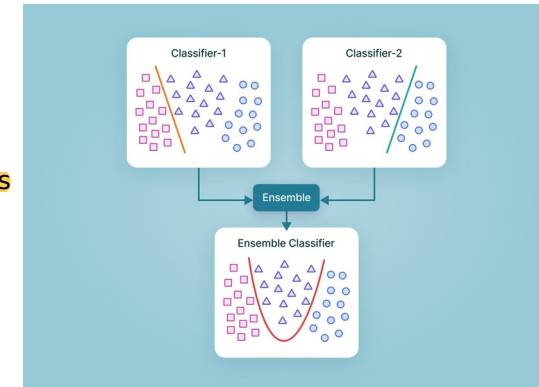
Learn to separate different classes (the model) using training data consisting of input-output pairs. Use it to **identify the labels for new inputs**

Support Vector Machines,
Decision Trees,
Neural Networks

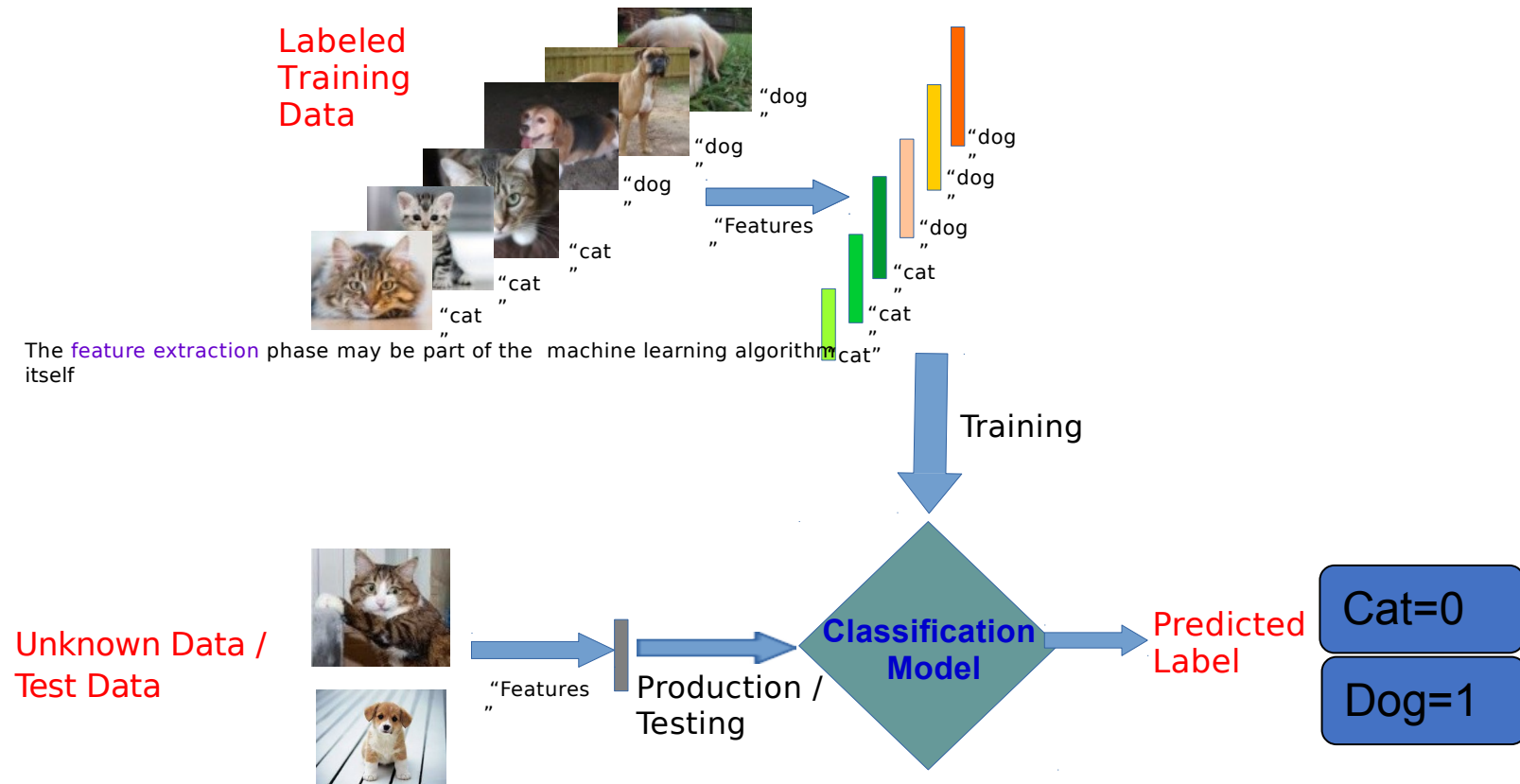


Ensembles

Ensemble methods are machine learning techniques that **combines several models in order to produce optimal models**



A Typical Supervised Learning



Training Model

- Our training data comes in pairs of inputs (x,y)
- $D=\{(x_1,y_1),\dots,(x_n,y_n)\}$

x_i : input vector of the i_{th} sample (feature vector)

y_i : label of the i_{th} sample

D : Training dataset

The goal of supervised learning is to develop a model h :

$h(x_i) \approx y_i$ for all $(x_i, y_i) \in D$

Training Steps

1-selecting the **type of machine learning** algorithm appropriate for a particular learning problem. This defines the hypothesis class H , i.e. the set of functions we can possibly learn.

2- finding the best function within this class, $h \in H$ that makes the **fewest mistakes** within our training data.

For step 2, we need to evaluate functions. This is where the **loss function** comes in.

- A loss function evaluates a hypothesis $h \in H$ on our training data and tells us how bad it is. The higher the loss, the worse it is
- A loss of zero means it makes perfect predictions. It is common practice to normalize the loss by the total number of training samples, n , so that the output can be interpreted as the average loss per sample (and is independent of n).

Loss function

- The goal of learning is to **reduce** the value of the **loss function**
- In supervised learning, our **training data provides us with the correct or desired output – known as the label – for each corresponding input.**
- The loss function **compares the label against the output** that our system currently predicts. A loss value of zero means perfect performance.

Zero-one loss

$$\mathcal{L}_{0/1}(h) = \frac{1}{n} \sum_{i=1}^n \delta_{h(\mathbf{x}_i) \neq y_i}, \text{ where } \delta_{h(\mathbf{x}_i) \neq y_i} = \begin{cases} 1, & \text{if } h(\mathbf{x}_i) \neq y_i \\ 0, & \text{o.w.} \end{cases}$$

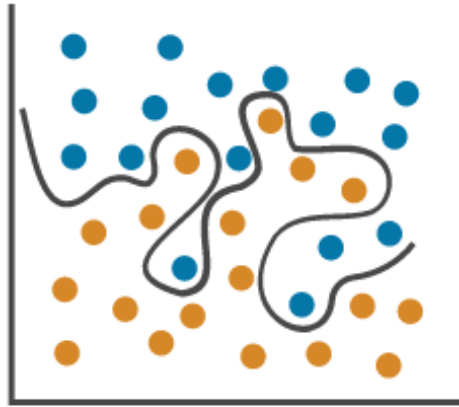
Squared loss

$$\mathcal{L}_{sq}(h) = \frac{1}{n} \sum_{i=1}^n (h(\mathbf{x}_i) - y_i)^2$$

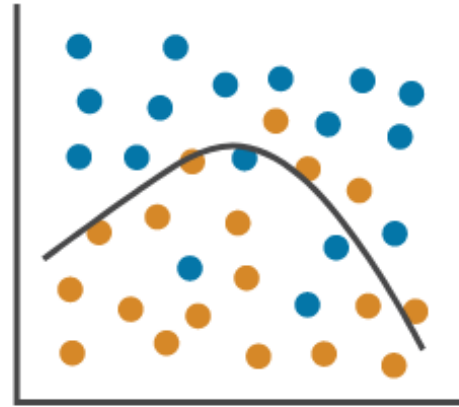
A function with near zero loss: Ideal ?

Overfitting

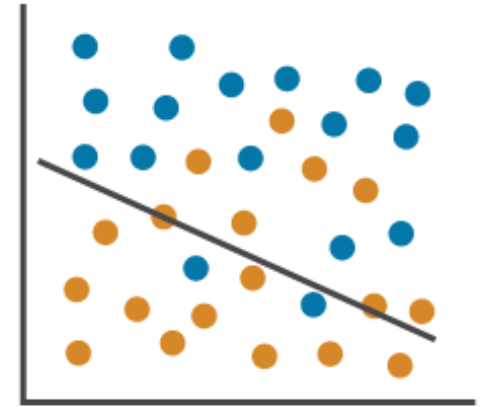
Overfitting



Right Fit



Underfitting



Mathworks

Error	Overfitting	Right Fit	Underfitting
Training	Low	Low	High
Test	High	Low	High

Our goal is not just to create a model that perform well on data used for training the model but **also perform well for new / unseen data**

Avoid Overfitting

- Reducing Model Complexity (Techniques Depends on Model)
- Feature Reduction
- Train with more data
- Ensembles / Super Learners
- Using Validation Data
- Etc.

Train / Test/ Validation

To resolve the overfitting issue, we usually split into three subsets:

- Train
- Test
- Validation

Best practice:

Train : 70%-80%

Validation : 5%-10%

Test: 10%-25%

Cross-validation

1- split dataset into k groups (k-fold cross-validation), one of the groups as testing set and the others as the training set.

2- repeat this process until each individual group has been used as the testing set

Iteration 1



Iteration 2



Iteration 3



Iteration 4



Iteration 5



Splitting Train and Test data

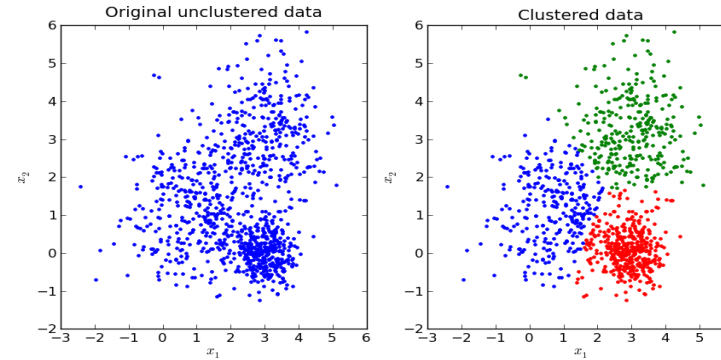
- Train and test data should be **separate** completely
- We should **not touch test** set during training
- **Before training**/learning, randomly split train and test data
- The test set must simulate a real-world scenario using **unseen data**
- Avoid over-sampling , under-sampling, etc. on **test data**

Some Unsupervised Approaches

Clustering

Learn the grouping structure for a given set of unlabeled inputs

k-Means, Hierarchical Clustering, ...



Association Rules

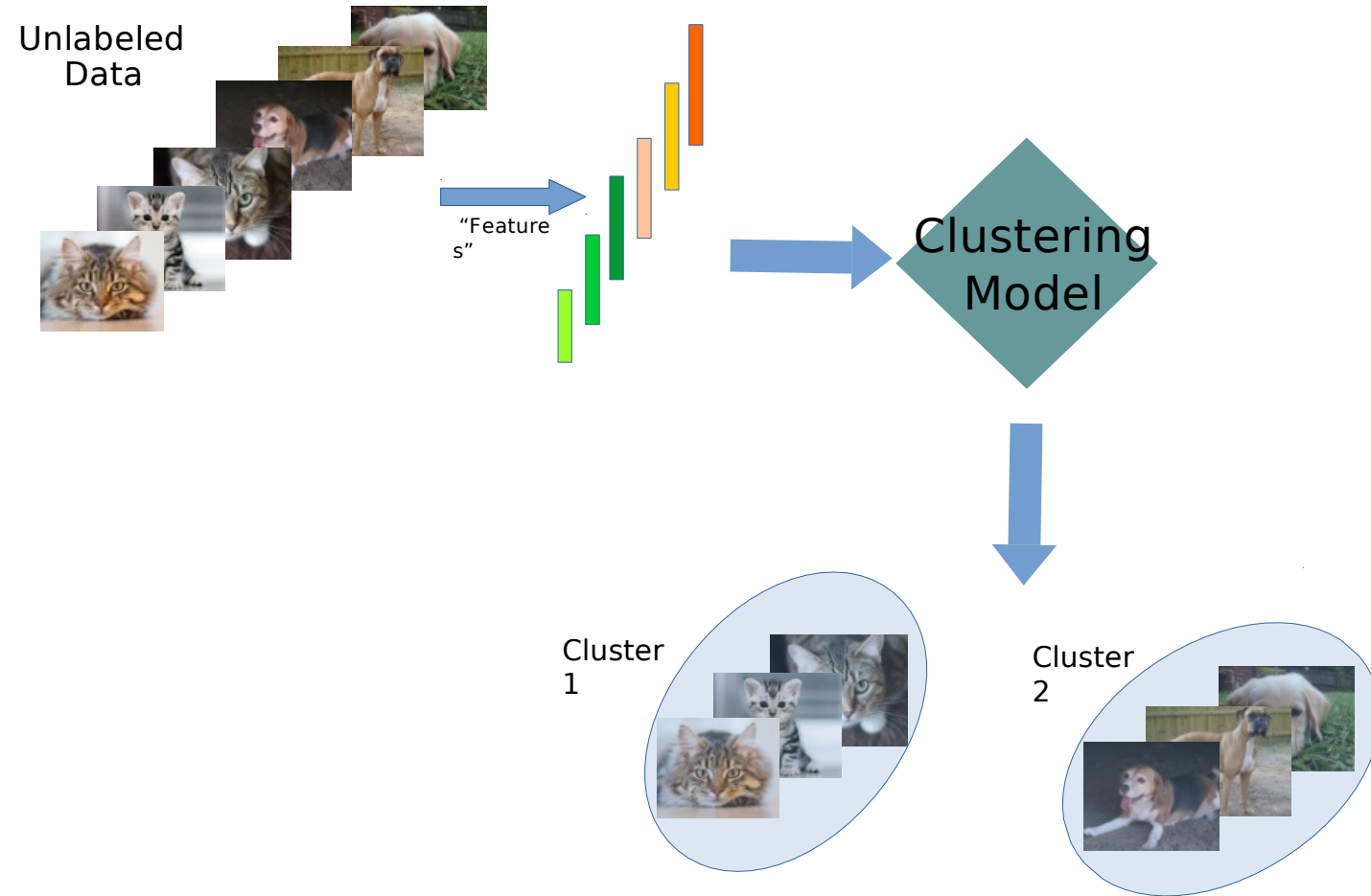
Association rule mining is a rule-based machine learning method for discovering interesting relations between variables in transactional databases.

Apriori, FP-growth, ...

$$\begin{array}{l}
 \text{Rule: } X \Rightarrow Y \\
 \begin{array}{l}
 \nearrow \text{Support} = \frac{\text{freq}(X, Y)}{N} \\
 \rightarrow \text{Confidence} = \frac{\text{freq}(X, Y)}{\text{freq}(X)} \\
 \searrow \text{Lift} = \frac{\text{Support}}{\text{Supp}(X) \times \text{Supp}(Y)}
 \end{array}
 \end{array}$$

<u>TID</u>	<u>Itemsets</u>	<u>frequent items</u>
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, b}
300	{b, f, h, j, o}	{f, b}
400	{b, c, k, s, p}	
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

A Typical Unsupervised Learning



Training in Supervised vs Unsupervised

In supervised learning we deal with **pairs of features and labels**

- Goal is to **predict labels from features** using a ML model

In unsupervised learning, we deal with **features without labels**

- Goal is to build a model, in order to:
 - **Reveal structure of data**
 - **Detect similarities / distances**
 - **Detect anomalies**
 - **Etc.**

Loss Function in Supervised vs Unsupervised

Supervised learning

- A loss function is used to choose a particular model with the lowest loss value
- After training and testing the model, we don't care about the loss function

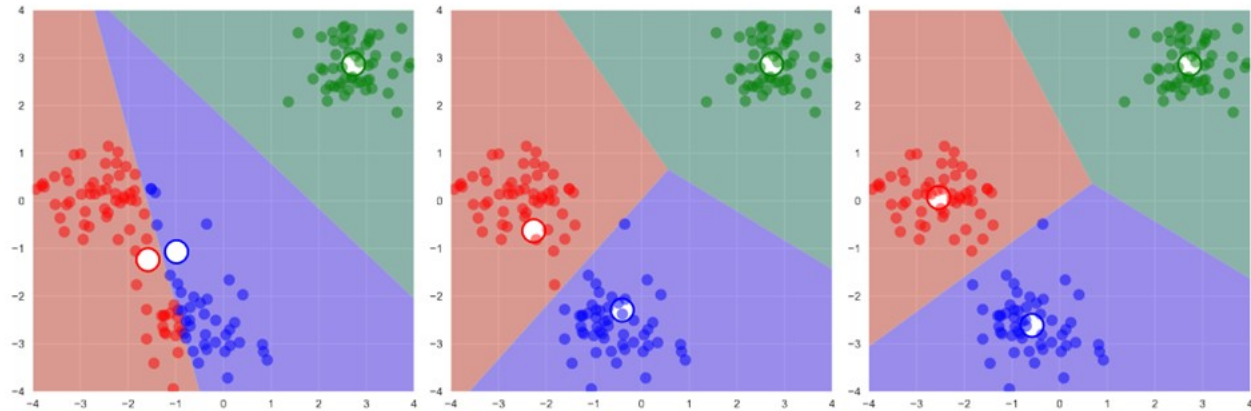
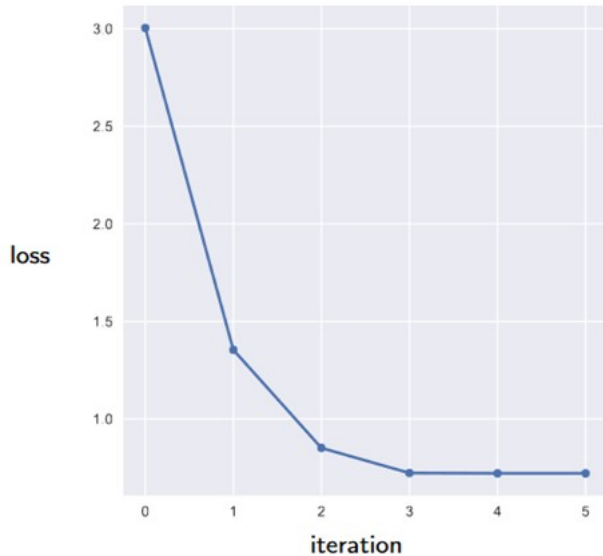
Unsupervised learning

- A loss function characterizes what the data looks like
- The loss function is our data model, and is itself our primary goal

Example of Loss Function in Unsupervised

Clustering:

- * Clustering models learn to group observations based on **similarities** between their features
- * Loss function is defined as **distance** between observations



K-mean Clustering

Semi Supervised

- Labeled data is limited
- Finding *representations* for supervised or other unsupervised models

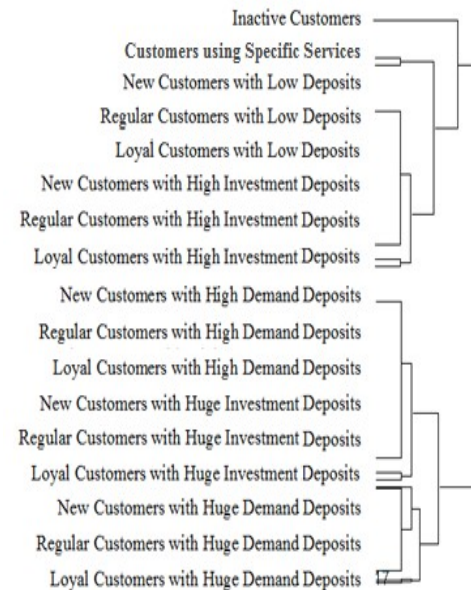
Unlabeled data

Period	Cust ID	Debit amount	Credit amount	Total balance	Average balance	Current account balance	...
201707		0	197373	22501445	22501445	2833235	
201707		7107250	7199589	19363370	22899232	47189	
201707		0	0	367000	367000	367000	
201707		1910000	1910000	591654	745847	11694	
201707		447598608	507229214	73553035	29126319	29287995	

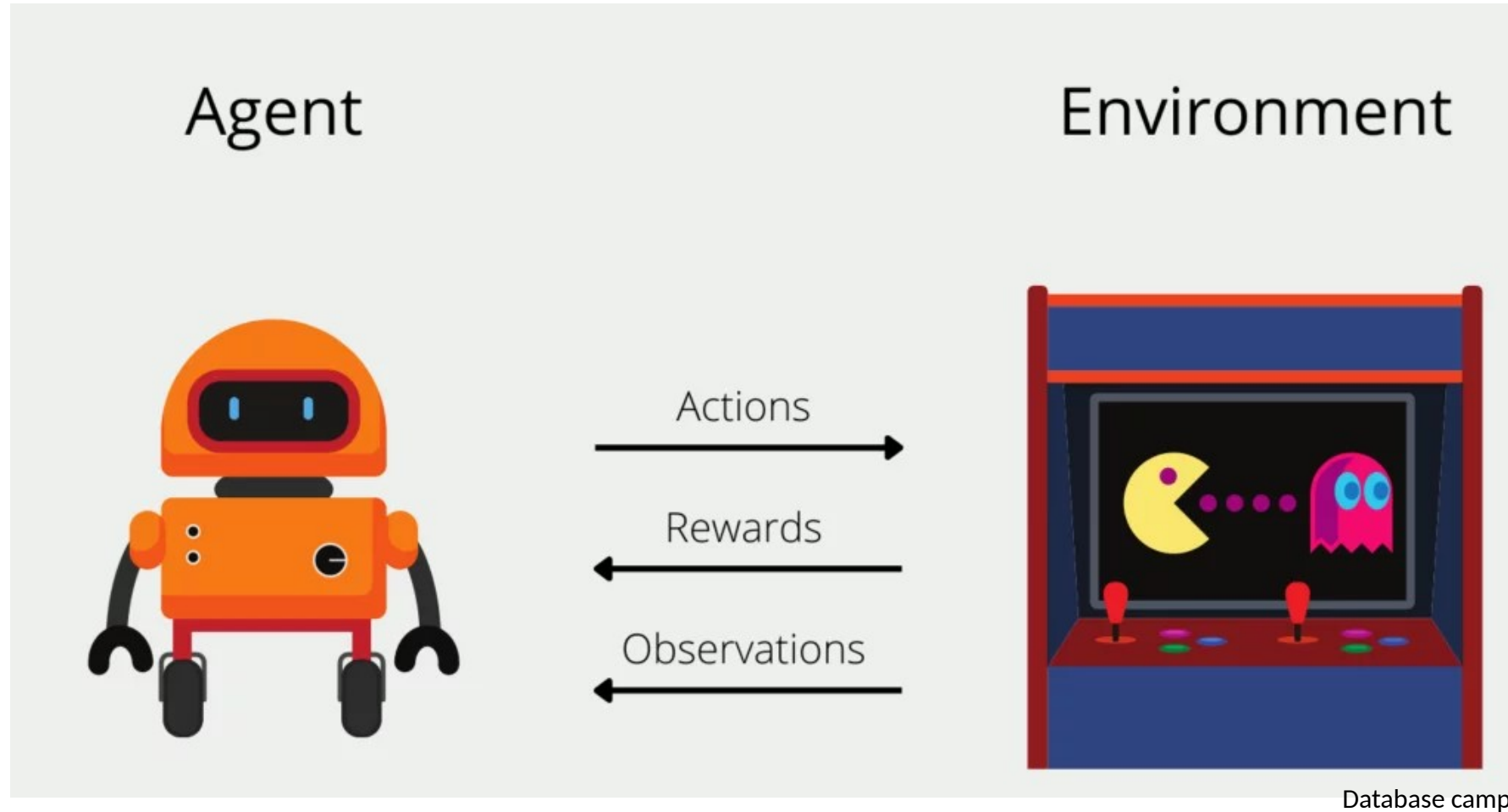
Using labeled data in other ML models

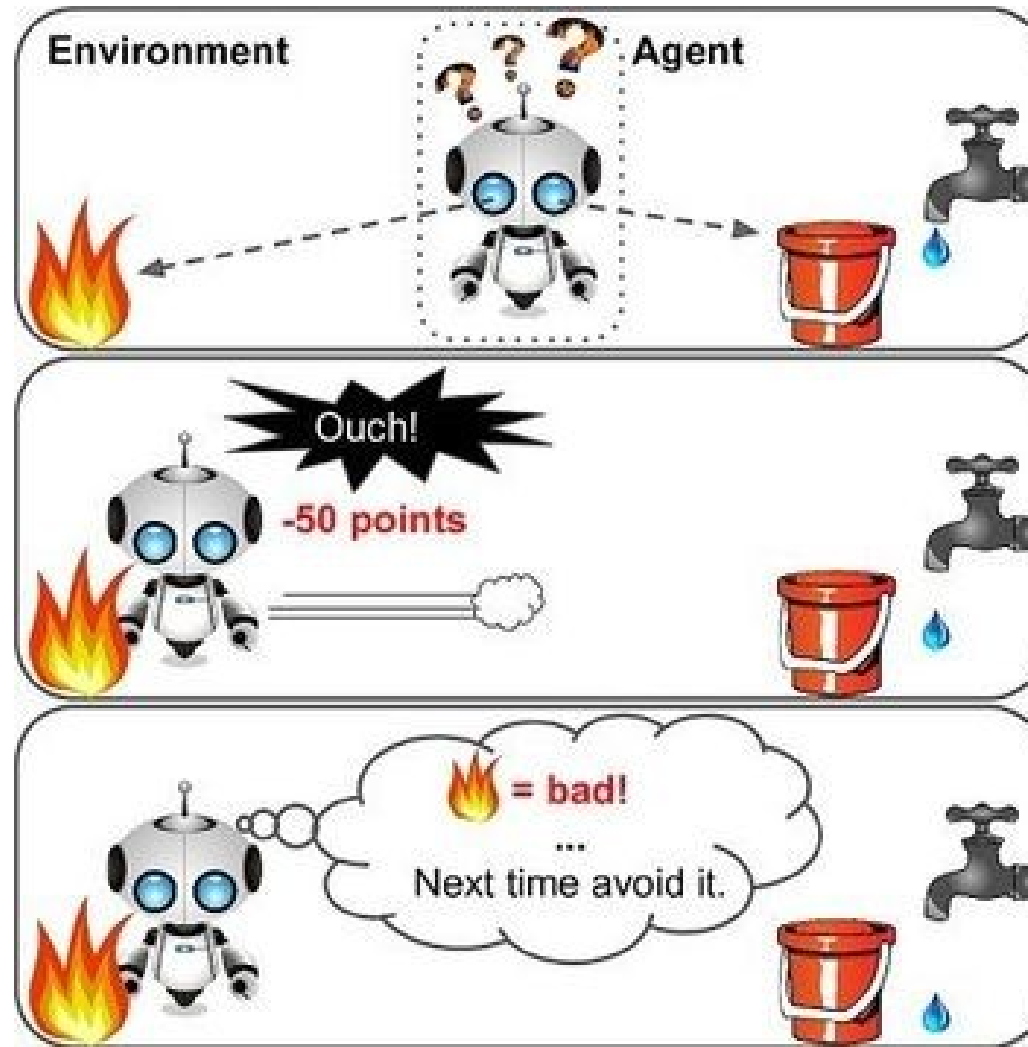
Segment no	Segment title	Time period	Members	Relative size (percent)	Financial resources (percent)
1	Inactive customers	2015.06	3955622	15.39	0.03
2	Customers using specific services	2015.06	1529249	5.95	2.64
3	New customers with low deposits	2015.06	1382009	5.38	1.94
4	Regular customers with low deposits	2015.06	2967886	11.55	3.82
5	Loyal customers with low deposits	2015.06	15797353	61.46	27.26
6	New customers with high Investment deposits	2015.06	2363	0.01	0.87
7	Regular customers with high Investment deposits	2015.06	4122	0.02	1.75
8	Loyal customers with high Investment deposits	2015.06	35542	0.14	16.67
9	New customers with high demand deposits	2015.06	1329	0.01	0.7
10	Regular customers with high demand deposits	2015.06	3029	0.01	1.47
11	Loyal customers with high demand deposits	2015.06	24721	0.1	12.47
12	New customers with huge Investment deposits	2015.06	3	0	0.72
13	Regular customers with huge Investment deposits	2015.06	8	0	0.31
14	Loyal customers with huge Investment deposits	2015.06	140	0	10.98
15	New customers with huge demand deposits	2015.06	12	0	0.73
16	Regular customers with huge demand deposits	2015.06	34	0	2.97
17	Loyal customers with huge demand deposits	2015.06	207	0	14.64

Hierarchical Clustering



Reinforcement Learning

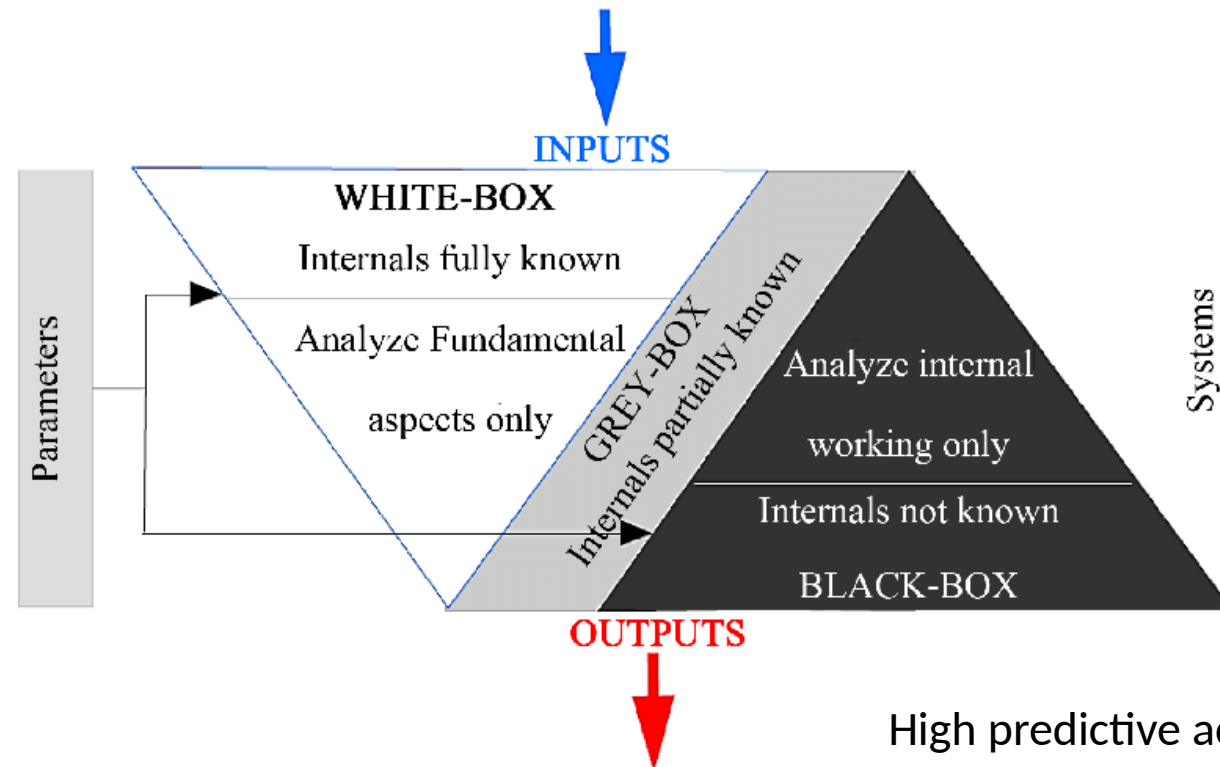




- 1 Observe
- 2 Select action using policy
- 3 Action!
- 4 Get reward or penalty
- 5 Update policy (learning step)
- 6 Iterate until an optimal policy is found

White-box vs. Black-box ML

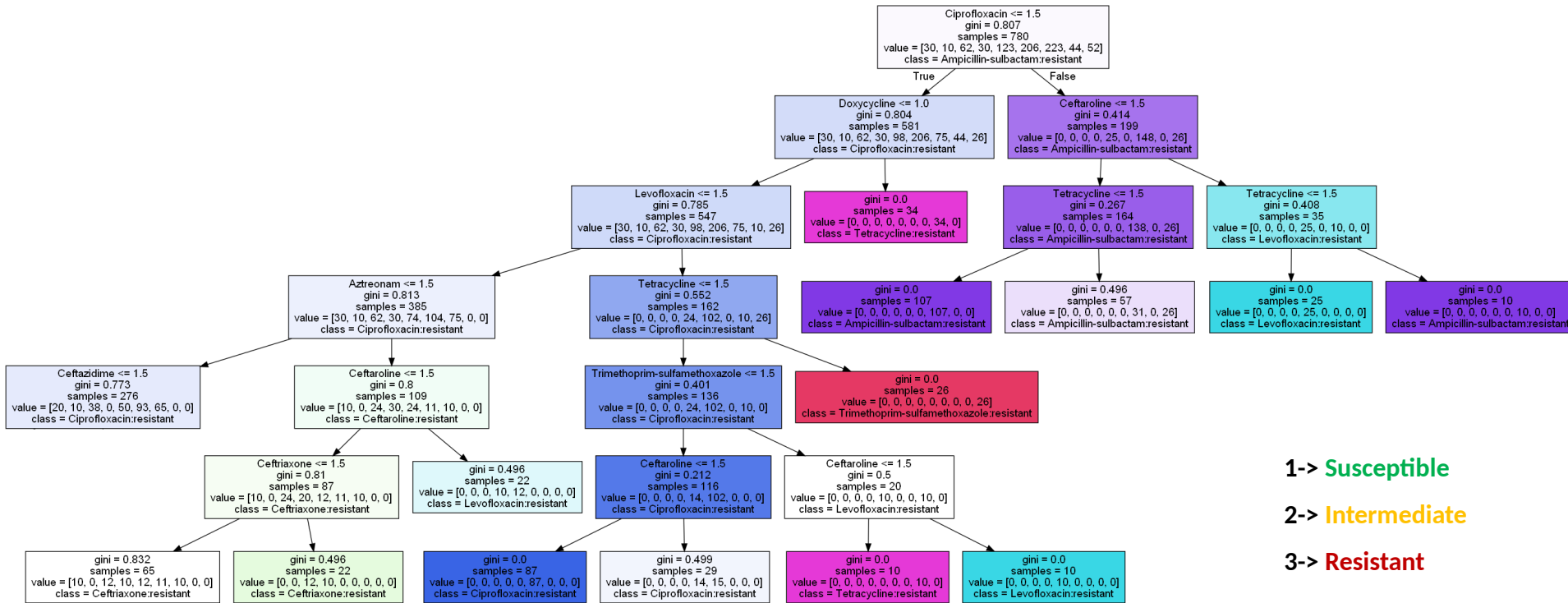
Interpretable results for developing data-driven theories
Transparency makes more reliable results than opaque models



Explainable AI

High predictive accuracy
Highly complex models

White-box Model Decision Tree (Predictive Model)



- 1-> Susceptible
- 2-> Intermediate
- 3-> Resistant

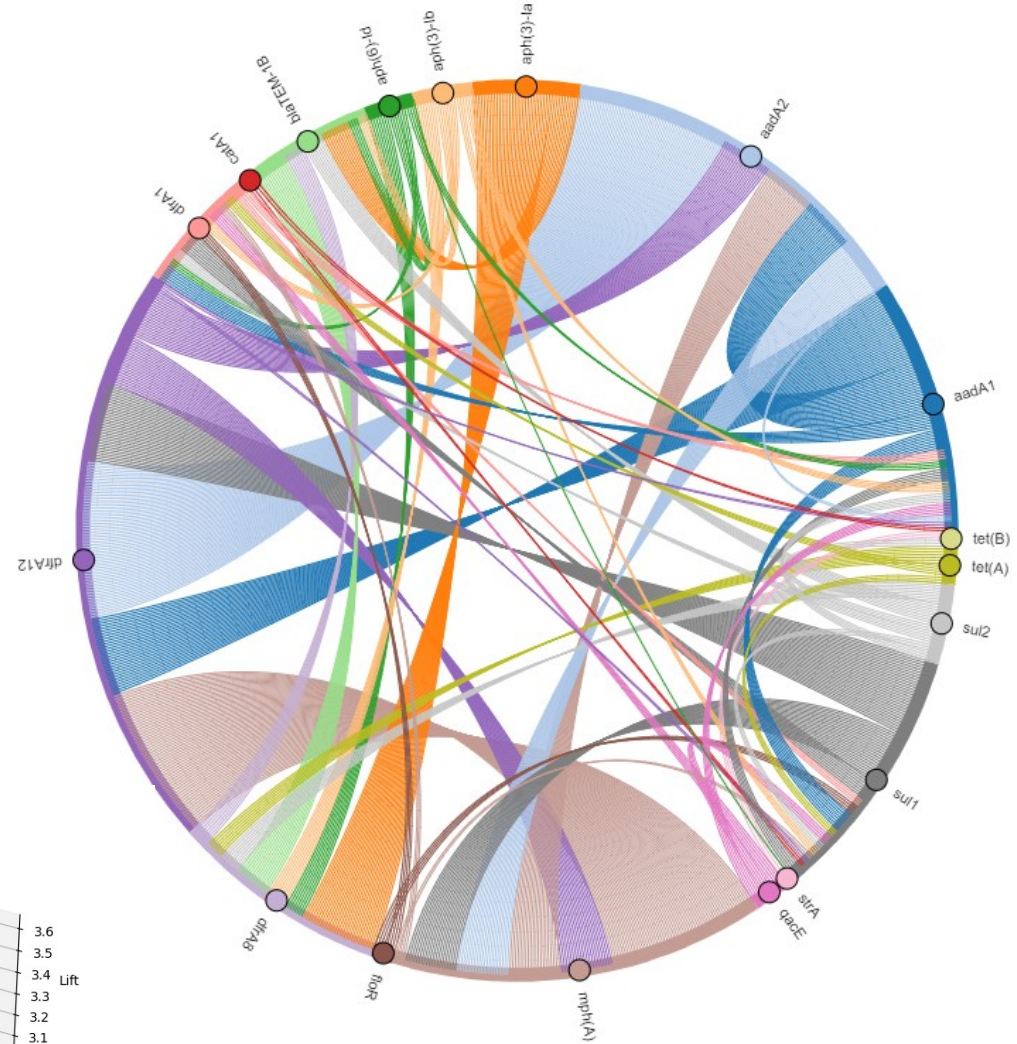
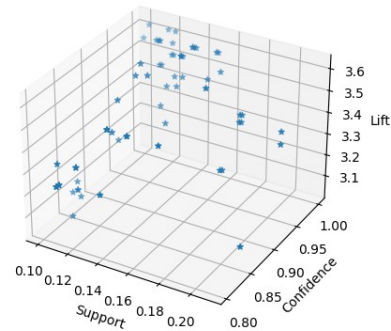
Hyperparameters

```
#####
##Parameters
#####
min_impurity_thr = 0.001      #Default min_impurity threshold for Dtree (Default is 0.001)
max_depth_thr = 15           #Default max_depth threshold for Dtree (Default is 15)
min_samples_leaf_thr = 30    #Default min_samples_leaf threshold for Dtree (Default is 30)
```

White-box Model

Association Rules (Descriptive & Predictive Model)

association	support	confidence	lift
[aadA2, mph(A)] -> [dfrA12]	0.01	1.00	80.50
[mph(A), aadA1] -> [dfrA12, aadA2]	0.01	1.00	53.67
[aadA2, sul1] -> [dfrA12, mph(A)]	0.01	1.00	53.67
[blaTEM-1B, tet(A), sul2] -> [dfrA8]	0.04	0.88	17.61
[blaTEM-1B, tet(A), aph(6)-Id] -> [dfrA8]	0.04	0.88	17.61
[blaTEM-1B, tet(A)] -> [dfrA8]	0.04	0.88	6.44
[strA, sul2] -> [tet(B)]	0.01	1.00	5.96
[strA, sul2, aph(6)-Id] -> [tet(B)]	0.01	1.00	5.96
[qacE, aadA1, tet(A)] -> [dfrA1]	0.06	1.00	2.68
[dfrA1, floR] -> [sul1]	0.35	1.00	2.64
[qacE, aph(6)-Id] -> [sul1]	0.06	1.00	2.64
[qacE, sul2] -> [dfrA1, sul1]	0.06	1.00	2.64
[mph(A)] -> [sul1]	0.02	1.00	2.64
[dfrA1, aph(6)-Id] -> [aadA1]	0.36	1.00	2.60
[dfrA1, aph(3)-Ib] -> [aadA1]	0.35	1.00	2.60
[dfrA1, sul2] -> [aadA1]	0.35	1.00	2.60
[qacE, tet(A)] -> [sul1, aadA1]	0.06	1.00	2.60
[qacE, dfrA1] -> [aadA1]	0.06	1.00	2.60
[aadA2] -> [aadA1]	0.02	1.00	2.60
[dfrA12] -> [mph(A), aadA2, sul1, aadA1]	0.01	1.00	2.60
[catA1] -> [sul1, aadA1]	0.01	1.00	2.60
[dfrA1] -> [sul1, aadA1]	0.37	0.98	2.30
[aadA1, aph(6)-Id] -> [dfrA1, sul1]	0.35	0.98	2.30
[sul1, aph(6)-Id] -> [dfrA1, aadA1]	0.35	1.00	2.30
[sul1, floR] -> [dfrA1]	0.35	1.00	2.30
[aadA1, sul2] -> [dfrA1, sul1]	0.35	0.98	2.30
[aadA1, aph(3)-Ib] -> [dfrA1, sul1]			
[sul1, sul2] -> [dfrA1, aadA1]			
[aadA1, floR] -> [dfrA1, sul1]			
[sul1, aph(3)-Ib] -> [dfrA1, aadA1]			
[qacE, sul1] -> [dfrA1, aadA1]			
[qacE] -> [dfrA1, sul1, aadA1]			
[qacE, aph(3)-Ib] -> [dfrA1, aadA1]			
[aph(3)-Ia, aph(6)-Id] -> [dfrA8, blaTEM-1B]			



Black-box Model

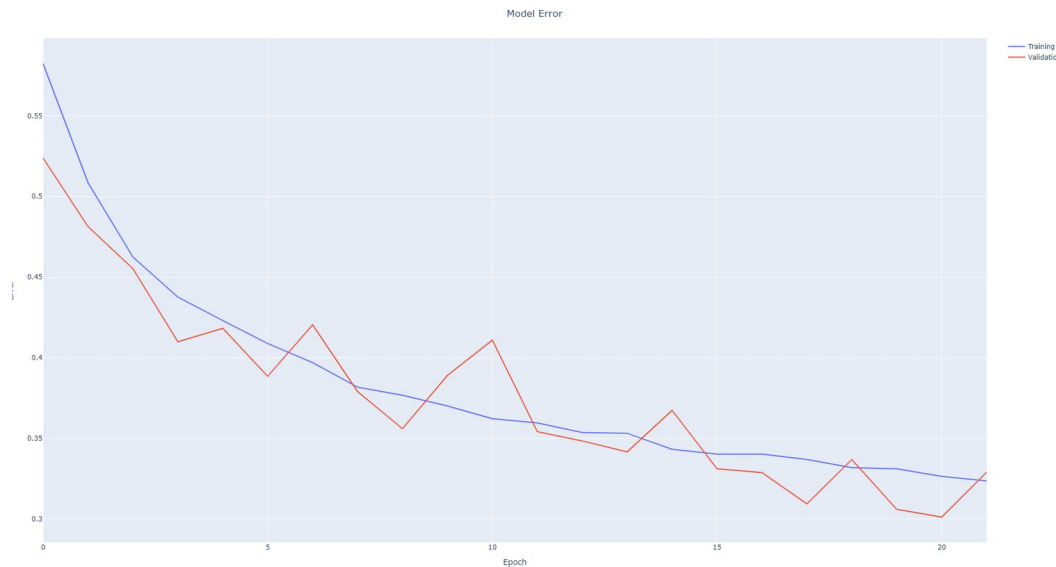
LSTM NN (Diagnostic Model)

```
def create_model(self):
    features = self.__get_features(self.X_train)
    X_val_fe = self.__get_features(self.X_val)

    # Define LSTM model architecture
    self.model = Sequential()
    self.model.add(LSTM(64, input_shape=(features.shape[1], 1)))
    self.model.add(Dense(units=self.number_of_classes, activation='softmax'))

    # Compile the model
    self.model.compile(optimizer='adam', loss=self.loss_function, metrics=['accuracy'])

    # Train the model
    self.history = self.model.fit(features,
                                   self.y_train,
                                   validation_data=(X_val_fe, self.y_val),
                                   epochs=training_epochs,
                                   batch_size=32)
```

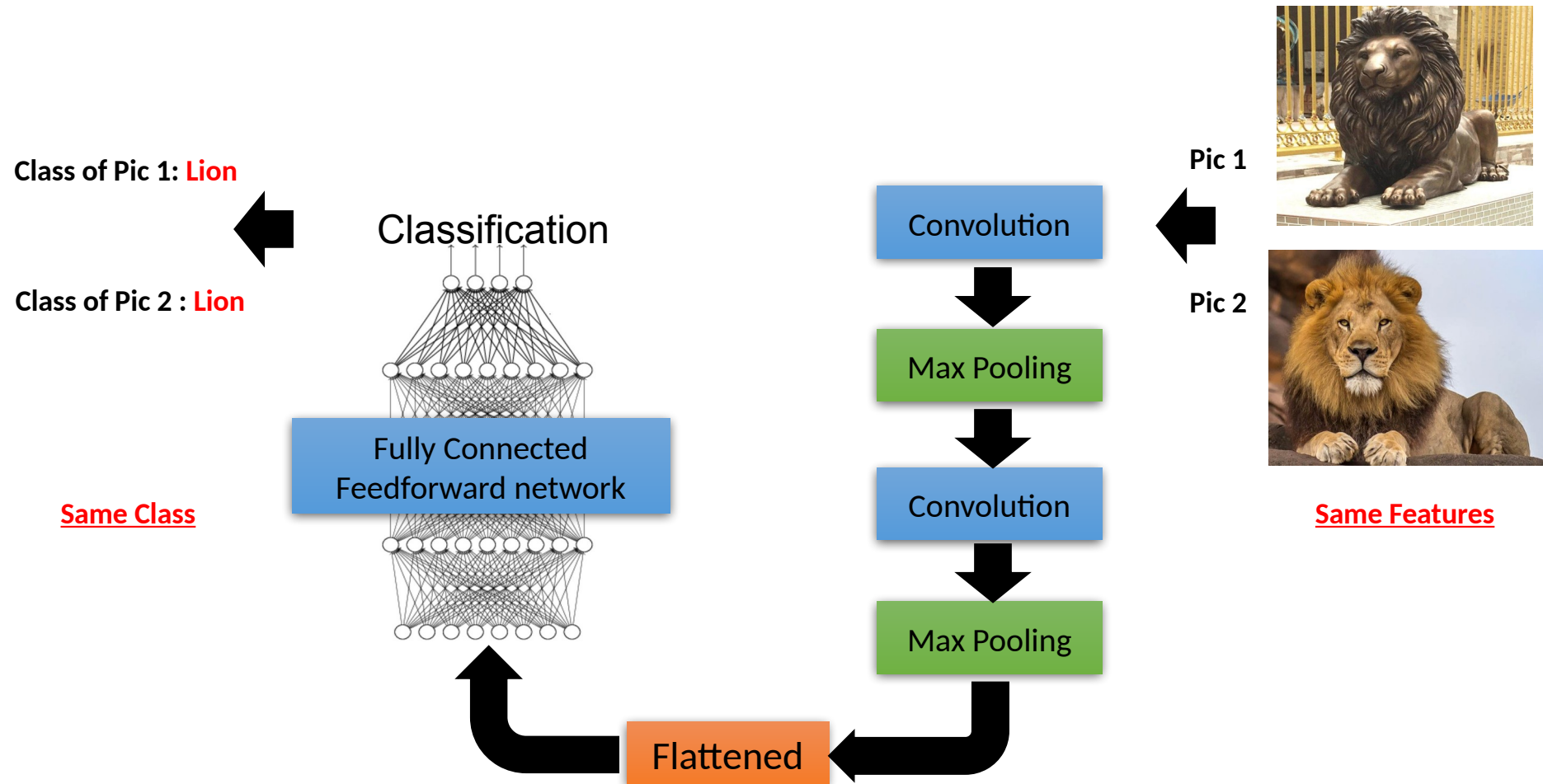


ECG Classification

Class	precision	recall	f1-score	support
0.0	0.92	0.97	0.94	14579
1.0	0.60	0.12	0.21	426
2.0	0.69	0.42	0.52	1112
3.0	0.58	0.05	0.09	145
4.0	0.76	0.84	0.80	1249
accuracy			0.90	17511
weighted avg	0.88	0.90	0.88	17511

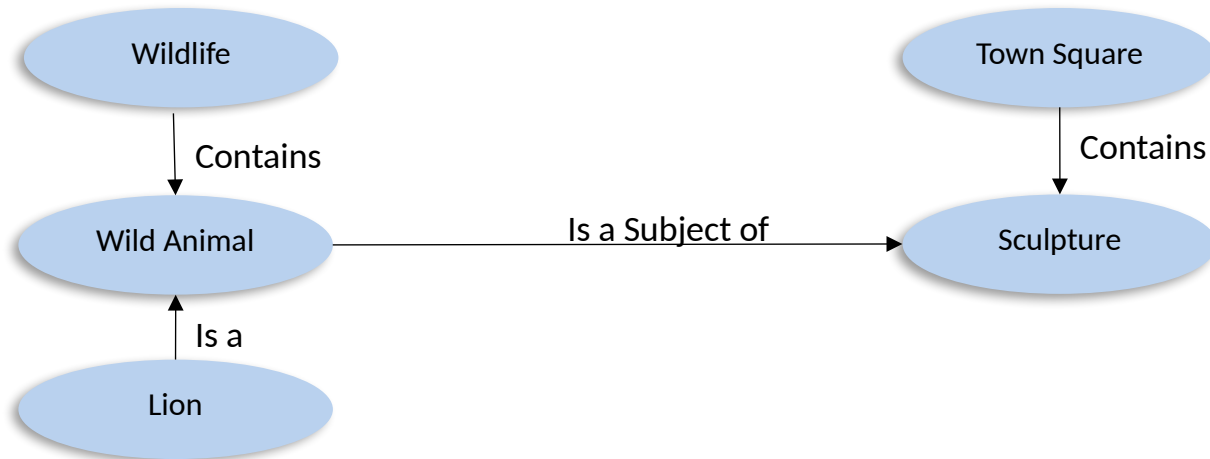
AI systems sometimes make mistakes

Image Classification using a CNN



Background knowledge

Image Understanding by a Human Being



Solution: Integrating knowledge-bases with DNNs

Moving from image classification towards image understanding

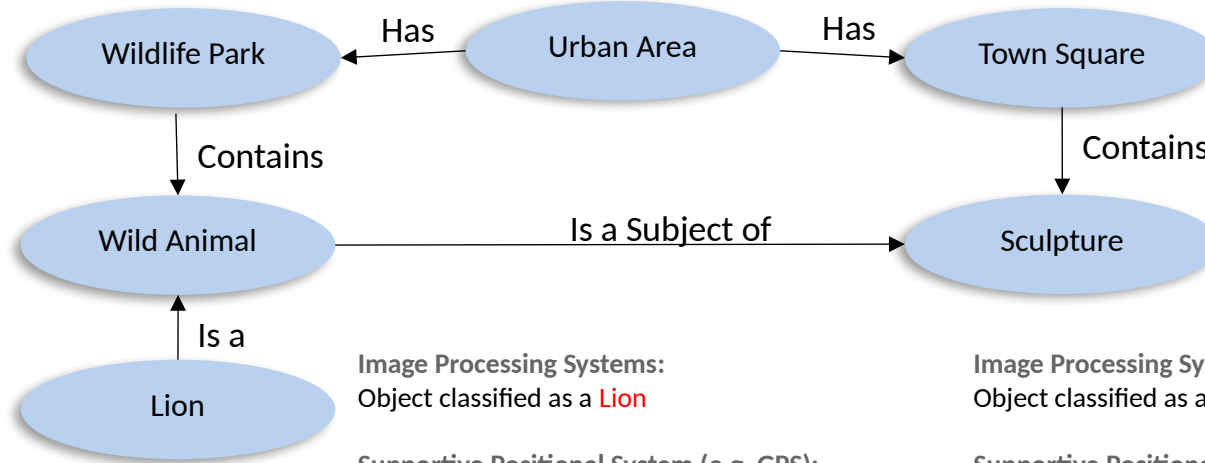


Image Processing Systems:

Object classified as a **Lion**

Supportive Positional System (e.g. GPS):

Approximate Location: **National Park**

Ontology:

National Park is a Wildlife Park

Wildlife Park Contains Wild Animals

Lion is an Animal

Results By the Proposed VPS :

Object classified as a **Lion**

Exact Position: **Section A of National Park**

Image Processing Systems:

Object classified as a **Lion**

Supportive Positional System (e.g. GPS):

Approximate Location: **Central Square**

Ontology:

Central Square is a Town Square

Town Square Contains Sculptures

Animals are Subject of Sculpture

Lion is an Animal

Results By the Proposed VPS:

Object classified as a **Sculptures of Lion**

Exact Position: **In the Middle of Central Square**



Ethical Issues in AI

The Best Algorithms Struggle to Recognize Black Faces Equally

Google's algorithm shows prestigious job ads to men, but not to women. Here's why that should worry you.

Gender and racial bias found in Amazon's facial recognition technology (again)

Do Google's 'unprofessional hair' results show it is racist?

How Amazon Accidentally Invented a Sexist Hiring Algorithm

A company experiment to use artificial intelligence in hiring inadvertently favored male candidates.

Bias, Deep fake, Etc.

Ethical Issues in Big-data

Dynamics of customer segments: A predictor of customer lifetime value

Abdolreza Mosaddegh^a, Amir Albadvi^{b,*}, Mohammad Mehdi Sepehri^b, Babak Teimourpour^b

^a Information Technology Engineering, Tarbiat Modares University, Tehran, Iran

^b Faculty of Industrial and Systems Engineering, Tarbiat Modares University, Tehran, Iran

ARTICLE INFO

Keywords:
Dynamics of customer segments
Customer lifetime value
Big data analytics
Banking sector

ABSTRACT

Most studies in the literature have focused on past behavior of customers to measure customer lifetime value, however, the rapid developments of technology and products make new conditions that cannot be predicted by past records anymore. In the era of new media and social networks, customers' needs and expectations change fast which lead to instability of customer lifetime value.

In the present study, we studied the dynamics of bank customers through value segments using big data analytics. By mining patterns of associations between customer transitions, we found six major categories, including the pattern of *Local Leaders* whose transitions are repeated by some follower groups within next two periods. Such results suggest that the dynamics of customer segments can be considered as a predictor of customer lifetime value. This approach uses the current dynamics of customers to predict CLV and therefore, unlike to the conventional method, accommodates to changing market conditions.

We also found a pattern by a few groups of *Market Trend Initiators* whose transitions are followed by overall market trends which gains insight into the dynamics of future markets.

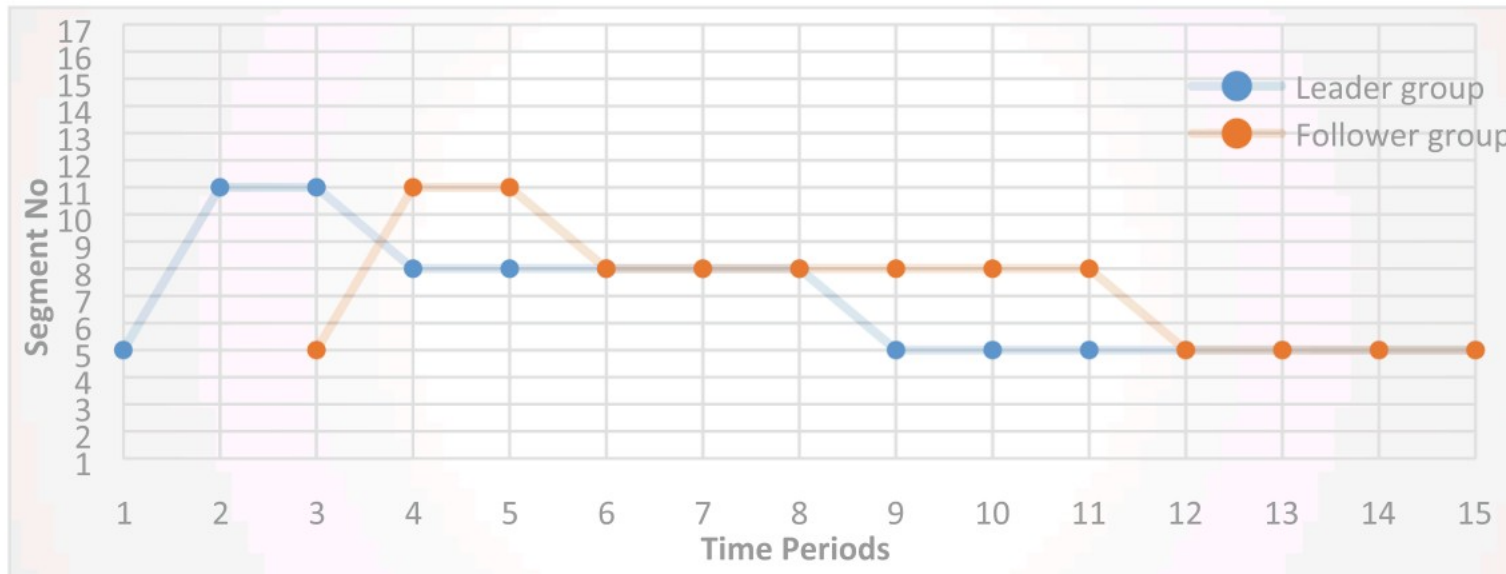
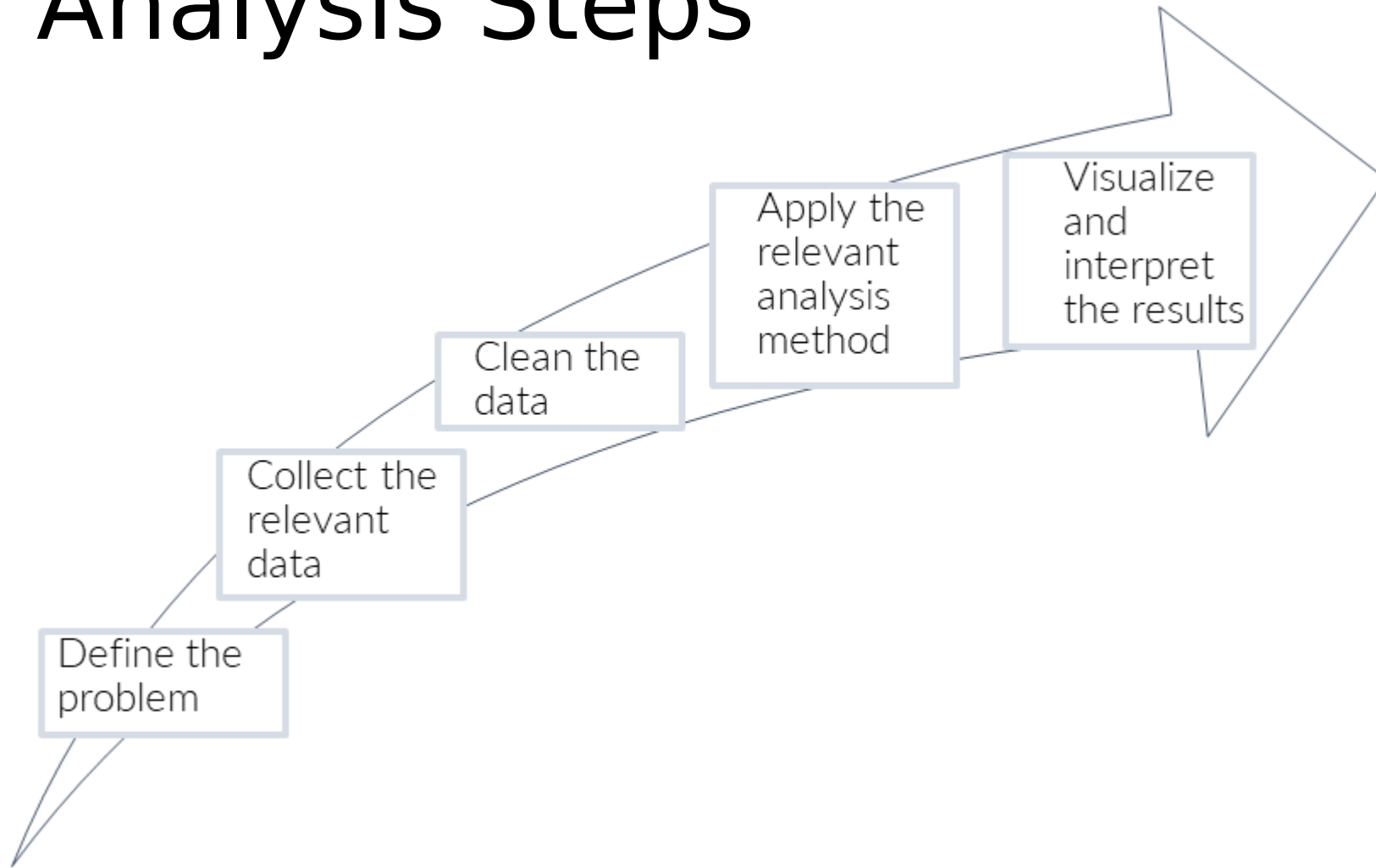


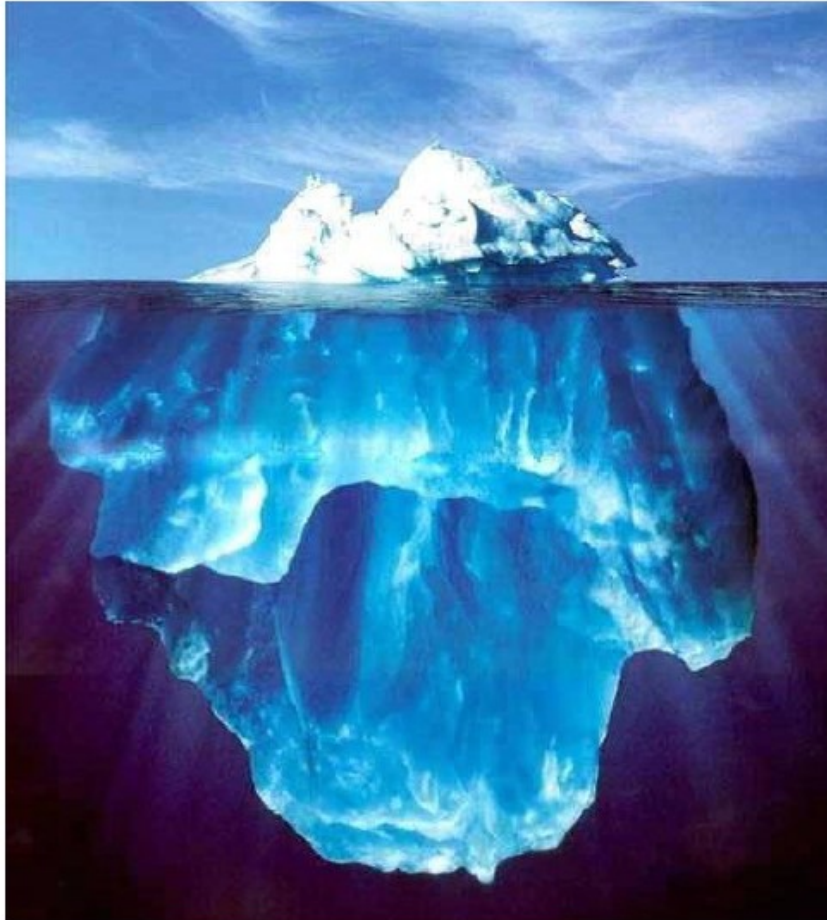
Fig. 3. Dynamics of a leader group and the corresponding follower group.

There are some ethical considerations with the results. A customer without any default, may be judged from his/her linkage to a defaulter leader group. This aspect of the dynamics should be aligned with legal requirements and ethical standards.

Machine Learning is powerful, but it can be harmful. Be mindful about **where** and **how** to use this powerful technology!

Data Analysis Steps





Data analysis and presentation is the FUN PART.



Data acquisition from myriad complex clinical, financial, administrative, and research source systems and the attendant cleansing, integration, and warehousing of these data is the HARD PART.

Oracle

“Cleaning and preparing the proper data for analysis can take up to 80% of a data scientist’s time.”

IBM

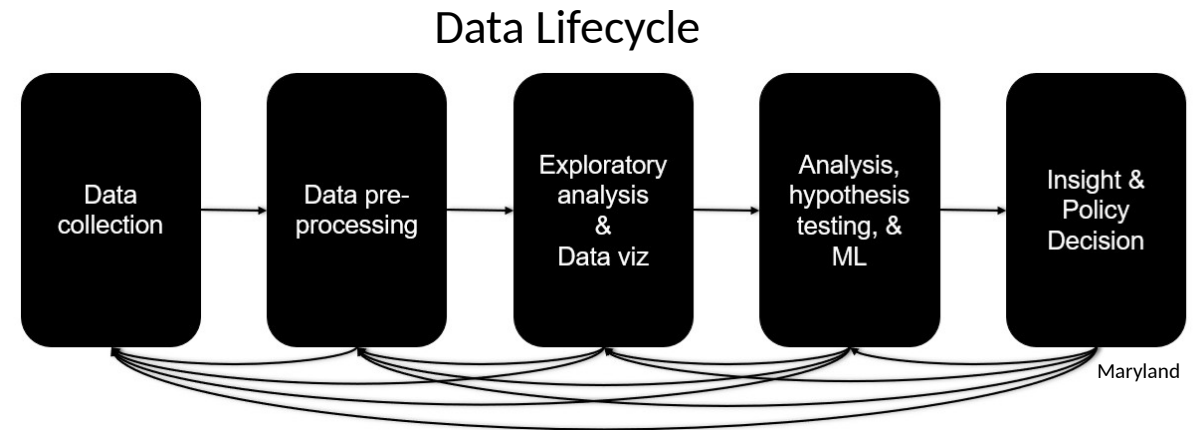
Why Data Preprocessing?

- Usually, data in the real world is not clean
 - **incomplete**: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
 - e.g., occupation=""
 - **noisy**: containing errors or outliers
 - e.g., Salary="-10"
 - **inconsistent**: containing discrepancies in codes or names
 - e.g., Age="42" Birthday="03/07/1997"
 - e.g., Was rating "1,2,3", now rating "A, B, C"
 - e.g., discrepancy between duplicate records



Data preprocessing

- The process of transforming “raw” data into data that can be analyzed to generate valid actionable insights
- Data preprocessing aka:
 - Data Wrangling
 - Data preparation
 - Data Cleansing
 - Data Transformation
 - Etc.



Major Tasks in Data Preprocessing

- **Data cleaning**
 - Fill in missing values, smooth noisy data, identify or remove outliers, remove duplicates, resolve inconsistencies and discrepancies
- **Data integration**
 - Integration of multiple databases, data warehousing
- **Data reduction**
 - Dimensionality reduction
 - Numerosity reduction
- **Data transformation**
 - Normalization
 - Discretization
- **Data Balancing, Data partitioning, etc.**

Incomplete (Missing) Data

- Data is not always available
 - E.g., have no recorded value for some attributes, such as customer income in sales data
- Missing data may be due to
 - equipment malfunction
 - inconsistent with other recorded data and thus deleted
 - data not entered intentionally or due to misunderstanding
 - certain data may not be considered important at the time of design
 - not register history or changes of the data
- Sometimes missing data may need to be inferred

Impute Missing Data

- **Ignore the tuple:** not effective when many records have missing values or dataset is small
- **Fill in the missing value manually:** not feasible when there are many missing values
- **Fill in it automatically** with
 - a global constant value: e.g., 0, “unknown”, a new class
 - last reported value in historical data: assumes no change, which is usually wrong
 - value of a record with similar other attributes (Hot Deck Imputation)
 - the attribute mean
 - the attribute mean for all samples belonging to the same class / group
- **Imputation using prediction models:** predict value of missing data using values of other attributes by a prediction model (e.g., regression)
- **Multiple Imputations (e.g., MICE)**

Multiple Imputation with Chained Equations

Imputation Cycle:

- **Step 1:** select one variable (usually with the least amount of missing data) as the target variable
- **Step 2:** Impute the missing values in other variable with temporary "place holder" values derived from the non-missing values of each variable. (e.g., mean of values)
- **Step 3:** Impute missing values of the target variable with predicted values by other variables using a prediction model (e.g., regression)
- **Step 4:** repeat steps 1-3 for other variables

Imputation Cycle can be repeated (e.g., 5 or 10)

It stops after a **number of cycles** or when **convergence** is reached (the difference between the last two imputed values is zero or very small)

MICE

Missing data is in red. There is a strong correlation between A and B, so let's try to impute A using B and C.

A	B	C
0.93	1.40	1.53
0.24	0.46	0.76
	0.80	
0.95	1.24	1.46
0.23	0.57	
0.90		1.28
0.15	0.42	
0.47	0.54	0.63
	1.14	
0.89	1.23	1.45

Missing data is filled in randomly. This dilutes the correlations, but allows us to impute using all available data.

A	B	C
0.93	1.40	1.53
0.24	0.46	0.76
0.90	0.80	1.53
0.95	1.24	1.46
0.23	0.57	1.28
0.90	0.46	1.28
0.15	0.42	1.53
0.47	0.54	0.63
0.47	1.14	1.28
0.89	1.23	1.45

A random forest is used to predict A with B and C. Notice the correlation between A and B improved.

A	B	C
0.93	1.40	1.53
0.24	0.46	0.76
0.24	0.80	1.53
0.95	1.24	1.46
0.23	0.57	1.28
0.90	0.46	1.28
0.15	0.42	1.53
0.47	0.54	0.63
0.89	1.14	1.28
0.89	1.23	1.45

After Imputing B using A and C, we have achieved a correlation between A and B much closer to the original data.

A	B	C
0.93	1.40	1.53
0.24	0.46	0.76
0.24	0.80	1.53
0.95	1.24	1.46
0.23	0.57	1.28
0.90	1.24	1.28
0.15	0.42	1.53
0.47	0.54	0.63
0.89	1.14	1.28
0.89	1.23	1.45

MICE in Python

```
import pandas as pd
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer

input_dataframe = pd.read_csv("original_dataset.csv")
print(input_dataframe)
imputer = IterativeImputer(max_iter=10, random_state=0)
imputed_dataset = imputer.fit_transform(input_dataframe)
imputed_dataframe = pd.DataFrame(imputed_dataset,
                                columns=input_dataframe.columns)
print(imputed_dataframe)
```

Noisy Data

- **Noise**: random incorrect values in a measured variable
- **Incorrect attribute values** may be due to
 - faulty data collection instruments
 - data entry problems
 - data transmission problems
 - inconsistency in data

How to Handle Noisy Data?

- **Binning**
 - first sort data and partition into (equal-frequency) bins
 - then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.
- **Regression**
 - smooth by fitting the data into regression functions
- **Clustering**
 - detect and remove outliers
- **Combined computer and human inspection**
 - detect suspicious values using queries and check by human (e.g., deal with incorrect values)

Binning (Discretization)

- **Equal-width** (distance) partitioning
 - Divides the range into N intervals of equal size: uniform grid
 - if A and B are the lowest and highest values of the attribute, the width of intervals will be:
$$W = (B - A) / N.$$
 - The most straightforward, but outliers may dominate presentation
- **Equal-depth** (frequency) partitioning
 - Divides the range into N intervals, each containing approximately same number of samples
 - Good data scaling

In Python : `pandas.qcut(data, number_of_bins)`

Binning Sample

Sorted data : 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* Partition into **equal-depth** bins:

- Bin 1: 4, 8, 9, 15
- Bin 2: 21, 21, 24, 25
- Bin 3: 26, 28, 29, 34

* Smoothing by bin means:

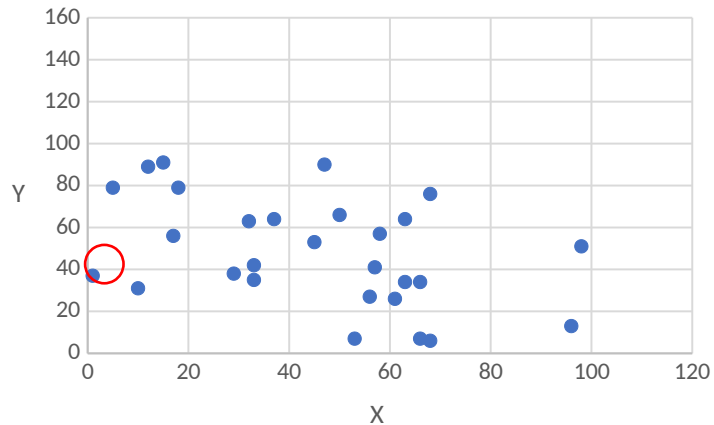
- Bin 1: 9, 9, 9, 9
- Bin 2: 23, 23, 23, 23
- Bin 3: 29, 29, 29, 29

* Smoothing by bin boundaries:

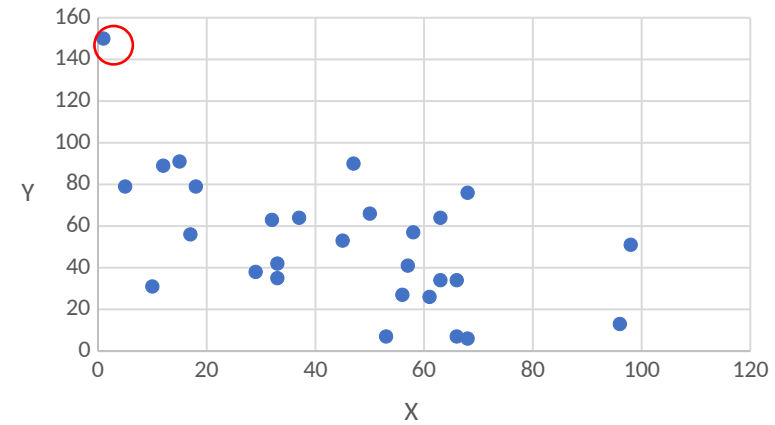
- Bin 1: 4, 4, 4, 15
- Bin 2: 21, 21, 25, 25
- Bin 3: 26, 26, 26, 34

Outliers

- Why do we care?
 - Can influence results of study



$$r = -.426$$



$$r = -.567$$

Same data, changed one value from 37 to 150

Identifying Outliers

- Identify outlier on final version of variables
 - E.g., After imputation
- Values that are greater than +3 standard deviations from the mean, or less than -3 standard deviations can be considered as outliers

SD of [34,35,41,56,**71,75**] = 16.53

- $M = 52.0$
- $M + 3 \text{ SD's} = 101.06$

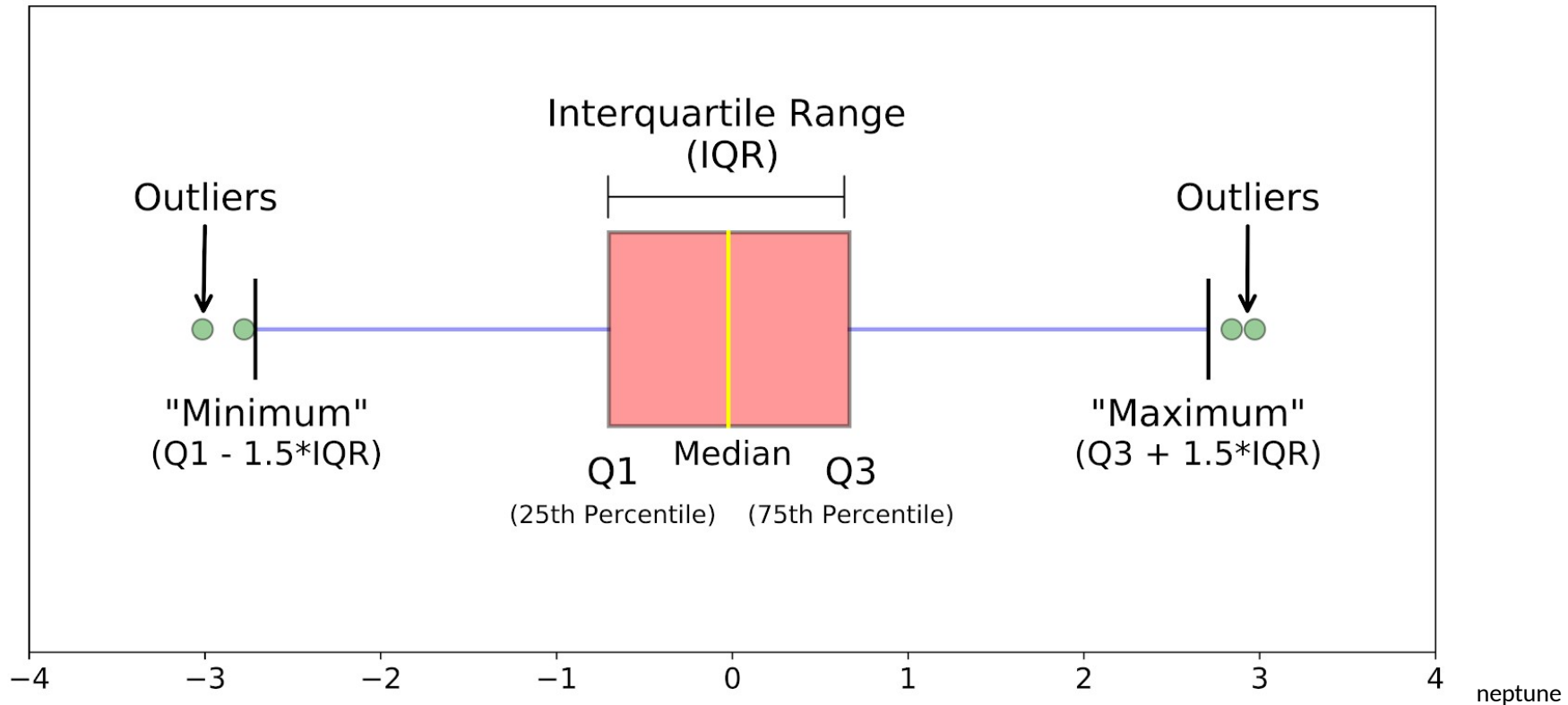
SD of [34,35,41,56,**71,150**] = 40.37

- $M = 64.5$
 - $M + 3 \text{ SD's} = 185.61$
- sometimes *SD* is influenced by extreme values. So, having extreme values makes it harder to detect extreme values

Boxplots

IQR (interquartile) method

- set up a “fence” outside of Q1 and Q3. Any values that fall outside of this fence are considered outliers.
- To build this fence take 1.5 times the IQR and then subtract this value from Q1 and add this value to Q3



In Python : `q3, q1 = numpy.percentile(data, [75, 25])`

Quiz Score
5
8
11
12
12
12
13
13
13
13
14
14
14
15
15
15
15
15

Q1

Median

Q3

$$1.5(\text{IQR}) = 1.5(3) = 4.5$$

$$\text{IQR} = 15 - 12 = 3$$

$$Q1 - 4.5 = 12 - 4.5 = 7.5$$

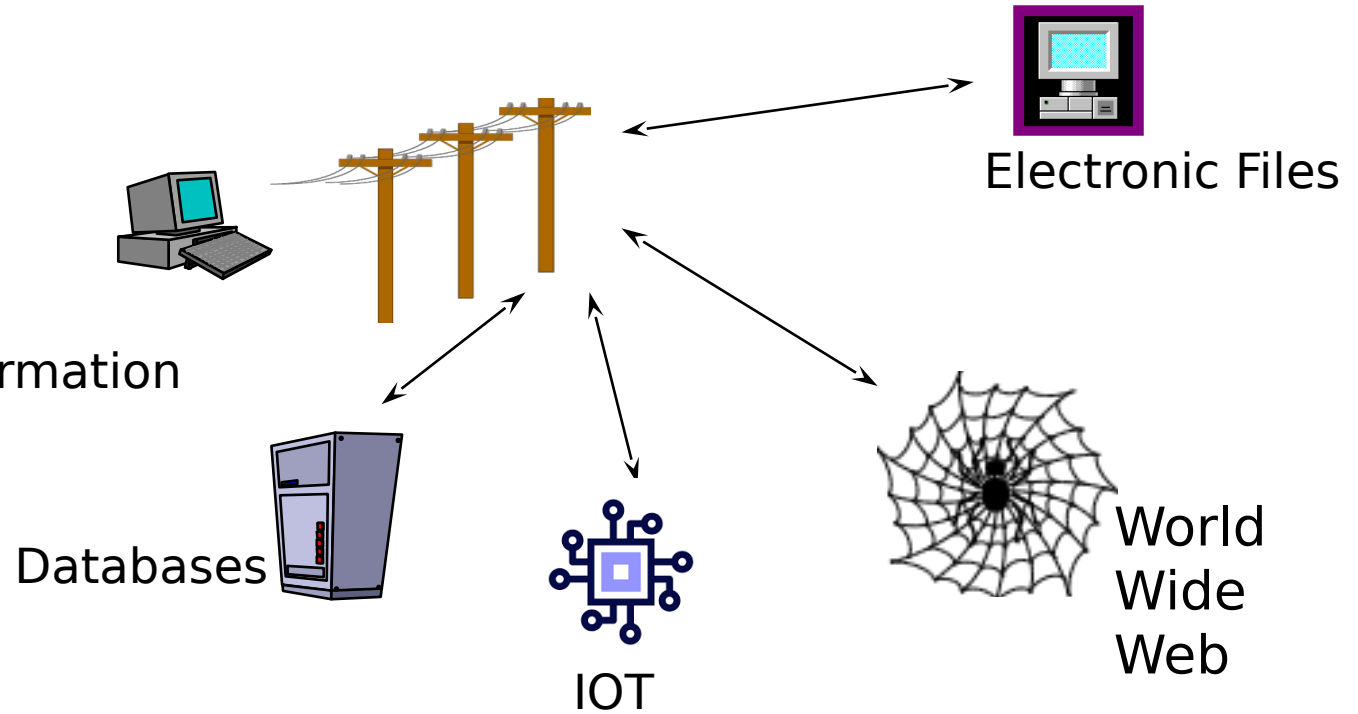
$$Q3 + 4.5 = 15 + 4.5 = 19.5$$

Addressing Outliers

- Outlier as a noncompliance (Invalid)
 - E.g., Negative age > Delete the outlier
- Outlier as a real extreme value
 - E.g., Bill Gates salary > Deleting valid cases biases your sample
- Outlier as output
 - E.g., detecting fraud > Outliers are outputs of process

Data Integration

- Different interfaces
- Different data structures
- Duplicate and inconsistent information



Structure of Data

“Looks like my V8 Chevy is running low on fuel. Didn’t I fill up just the day before?”

UNSTRUCTURED

STRUCTURED

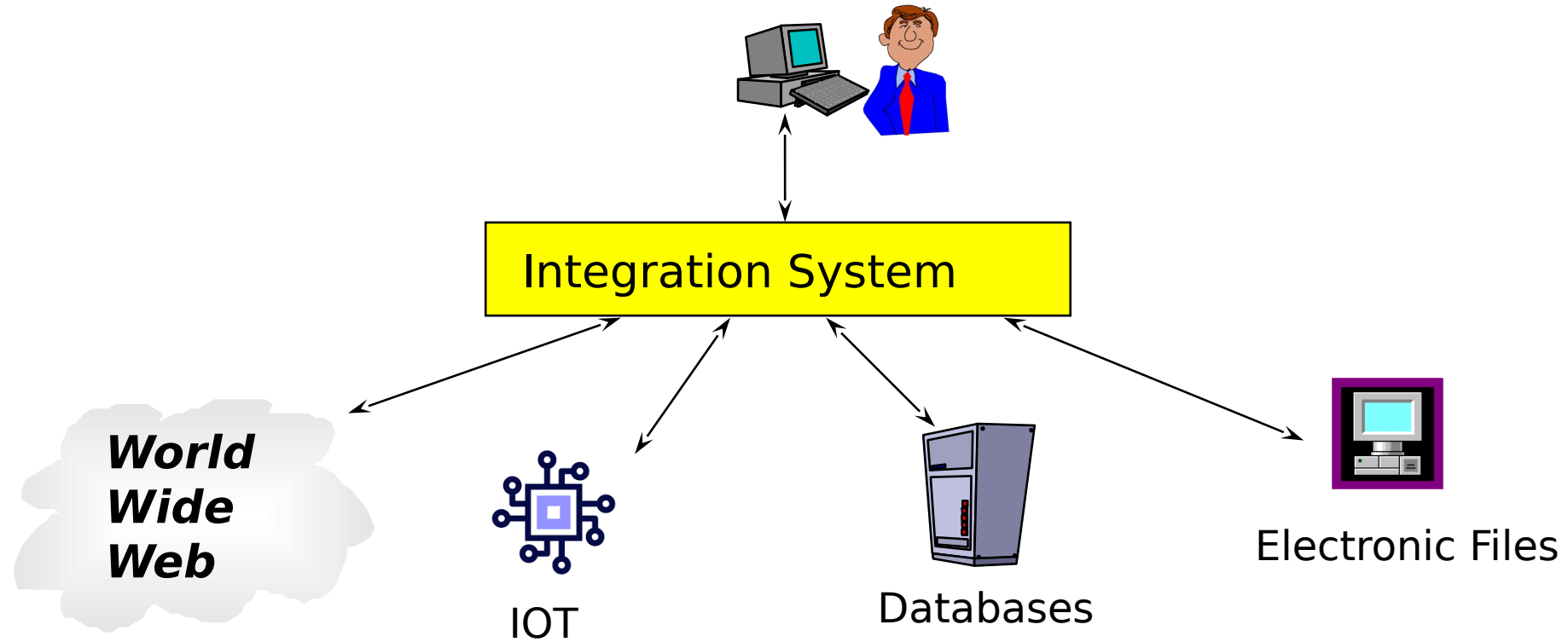


Owner	Vehicle	Type	Fuel Level	Engine	Last Fill
AK	Chevy	Gas	5%	V8	05/04/16

Semi - Structured

<Rec1> <Vehicle> Chevy </ Vehicle > <Engine> V8 </Engine></Rec1>

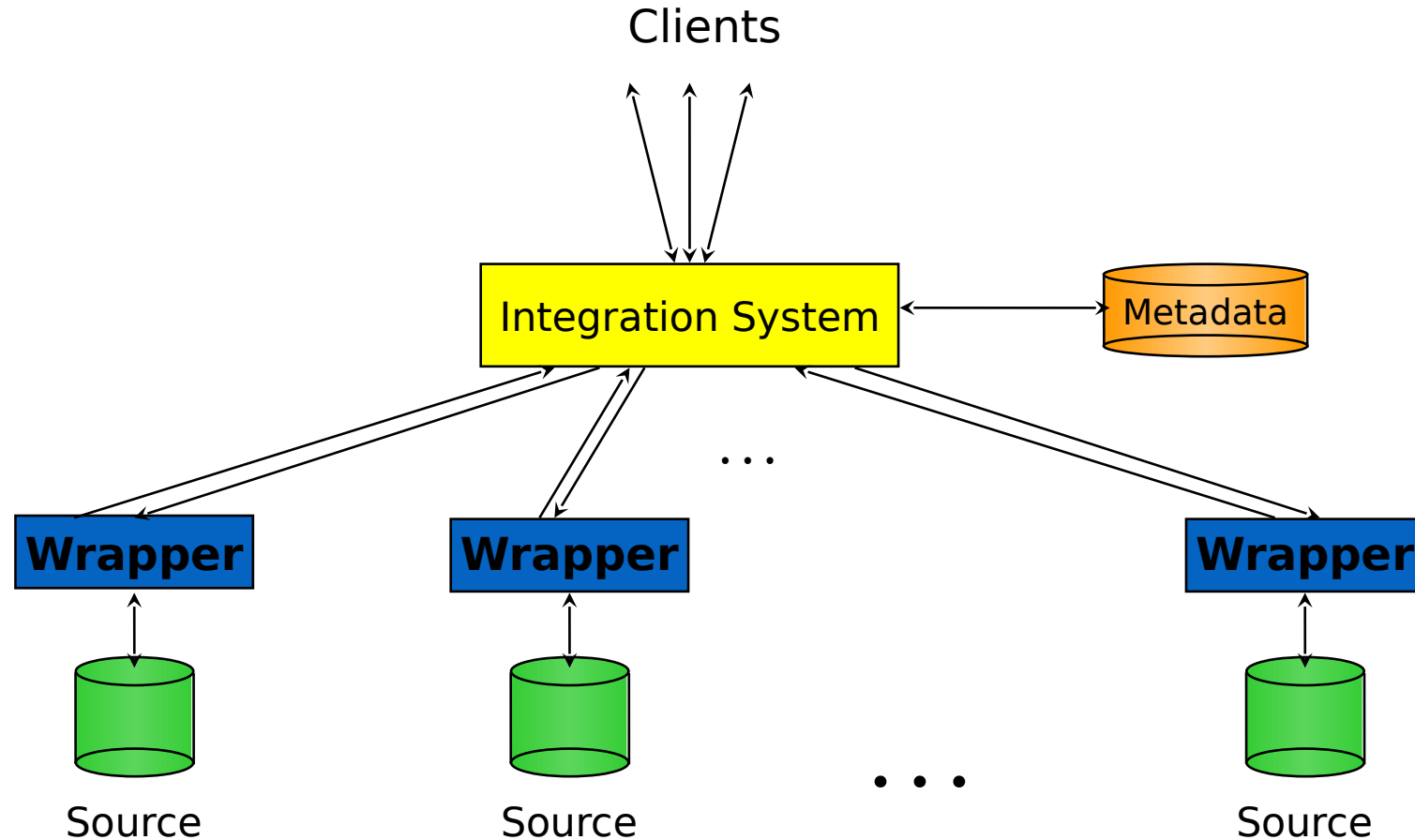
Goal: Unified Access to Data



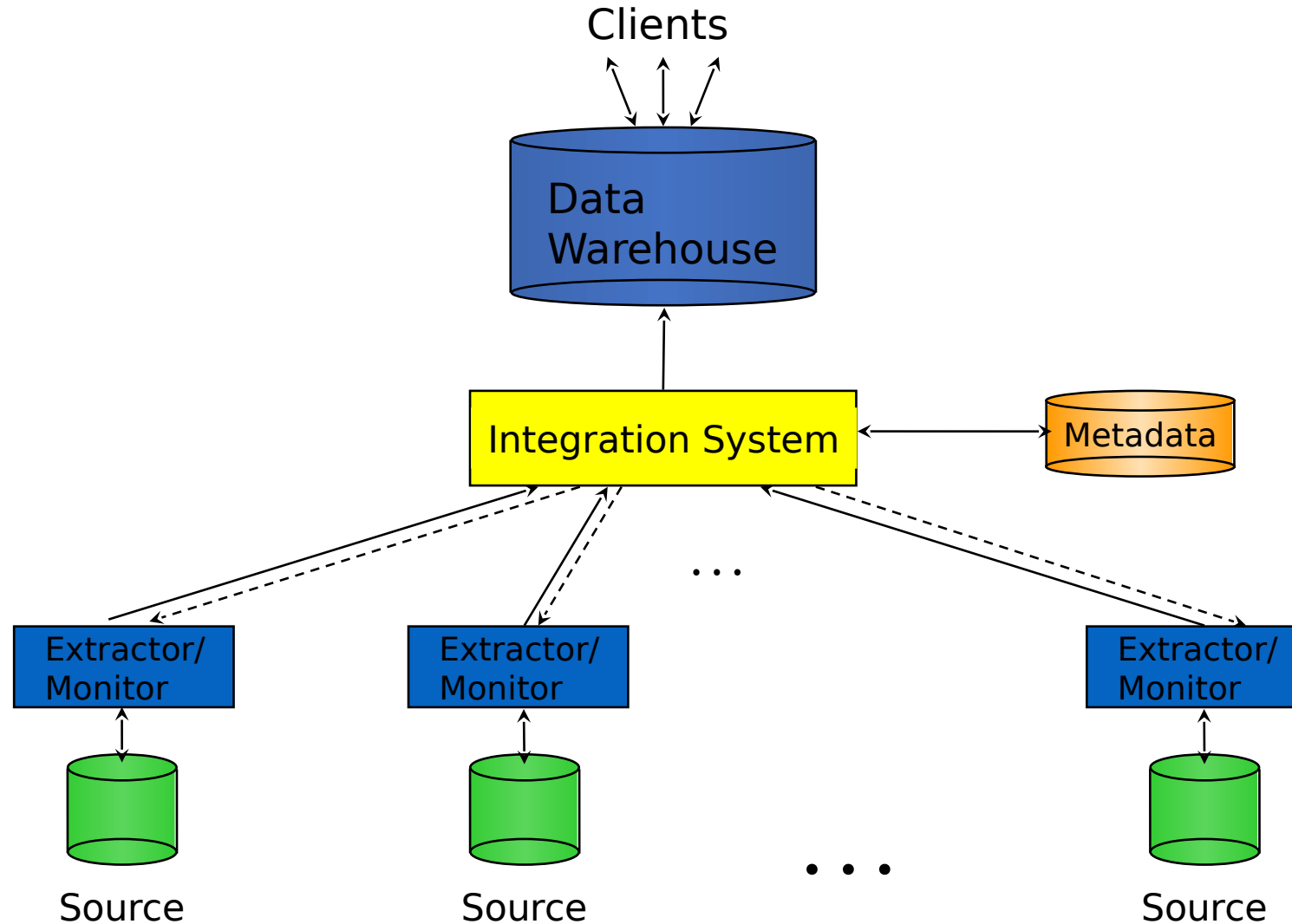
- Collects and combines information
- Provides integrated view, uniform user interface
- Supports sharing

The Traditional Approach

Query-driven (lazy, on-demand)



The Warehousing Approach



Query-Driven vs Data-warehouse

Disadvantages:

- **Slow** in query processing
- **Unavailable** information sources
- Inefficient and potentially **expensive** for frequent queries
- **Complex** integration
- **Not common** in industry

Advantages:

- Rapidly changing information
- Rapidly changing information sources
- Truly vast amounts of data from large numbers of sources
- Clients with unpredictable needs

Data Transformation

A function that maps the entire set of values of a given attribute to a new set of replacement values

- **Smoothing**
- **Aggregation and Summarization**
- **Normalization:** Scaled to fall within a smaller, specified range
 - **min-max normalization:** preserve scale but does not handle outliers well

$\text{New value} = (\text{Original value} - \text{Min}) / (\text{Max} - \text{Min})$

In Python: `preprocessing.MinMaxScaler()`

- **z-score normalization:** better handling of outliers but not exact same scale

$\text{New value} = (\text{Original value} - \text{Mean}) / \text{Standard deviation}$

In Python: `preprocessing.StandardScaler()`

Normalization can help to reduce the impact of outliers by scaling the data to a common scale, which can make the outliers less influential.

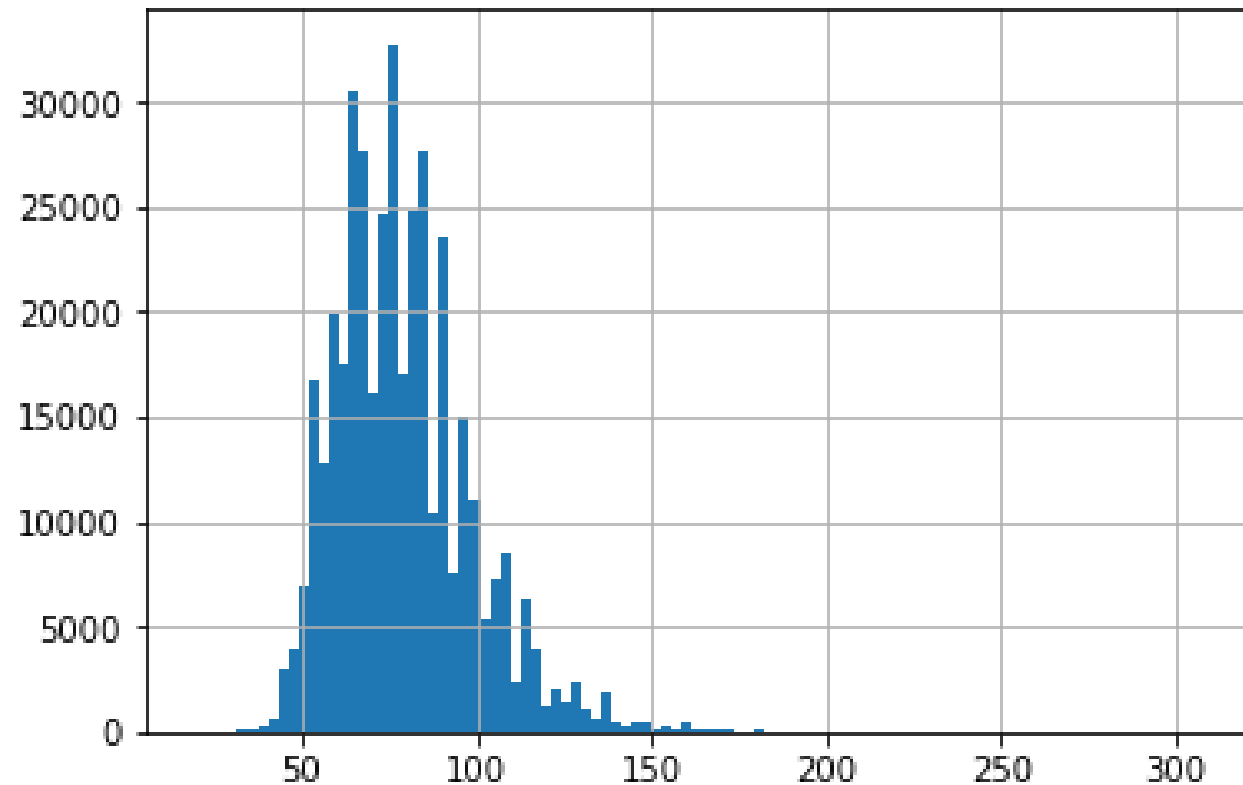
Normalization can result in a loss of information in some datasets.

Exploring Data - Statistics

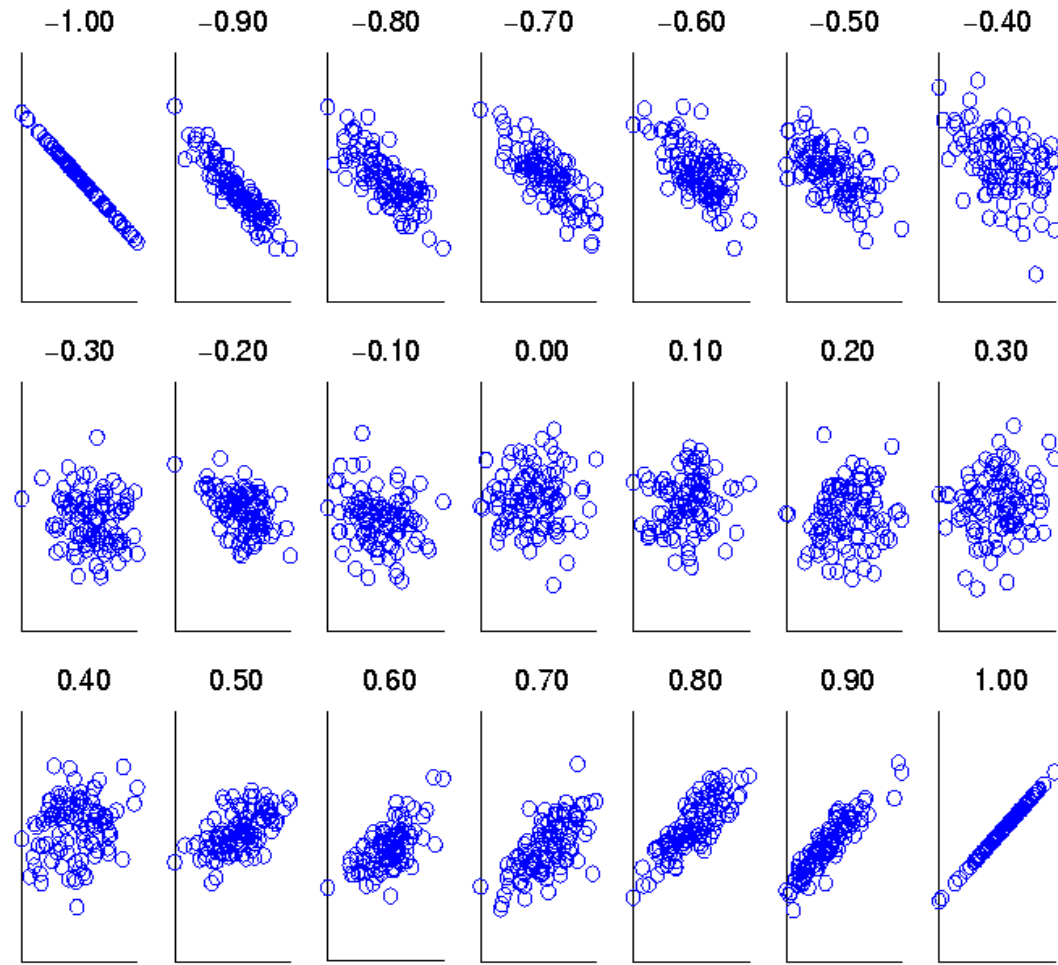
- Information that give a quick and simple description of the data:
 - Maximum value
 - Minimum value
 - Range (max – min)
 - Mean
 - Median
 - Mode
 - Quantile
 - Standard deviation
 - Etc.

Exploring Data - Distribution

Histogram allows the inspection of the data for its underlying distribution



Exploring Data - Correlations



**Scatter plots showing
the correlations
between two features
(from -1 to 1)**

Figure 5.11. Scatter plots illustrating correlations from -1 to 1.

In Python: `matplotlib.pyplot.scatter(x, y)`

Correlation only measures linear relationship

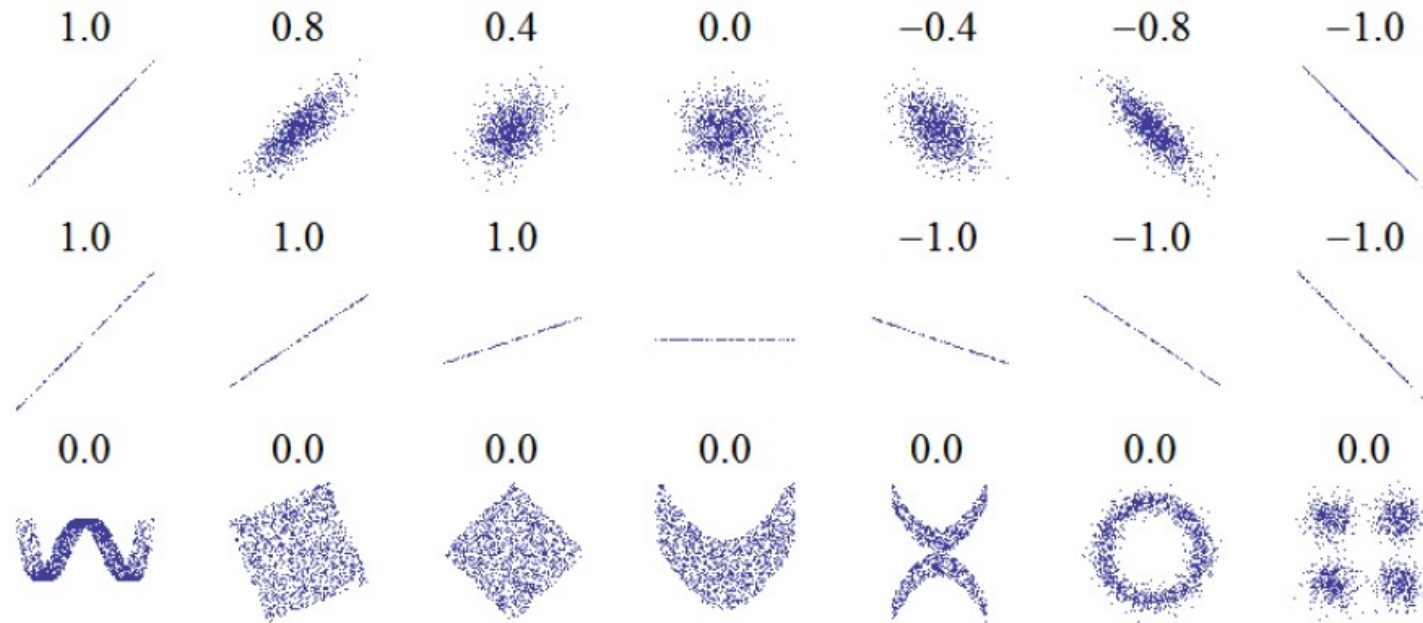


Figure 9.1: Examples of datasets with a range of correlations.

Reference Books

- 1) Data Wrangling with Python by Jacqueline Kazil
- 2) Best Practices in Data Cleaning by Jason Osborne
- 3) Bad Data by Q. Ethan McCallum

Some Open Access Datasets

- <https://www.data.gov/>
 - US-centric agriculture, climate, education, energy, finance, health, manufacturing data, ...
- <https://cloud.google.com/bigquery/public-data/>
 - BigQuery (Google Cloud) public datasets (bikeshare, GitHub, Hacker News, Form 990 non-profits, NOAA, ...)
- <https://www.kaggle.com/datasets>
 - Microsoft-owned, various (Billboard Top 100 lyrics, credit card fraud, crime in Chicago, global terrorism, world happiness, ...)
- <https://aws.amazon.com/public-datasets/>
 - AWS-hosted, various (NASA, a bunch of genome stuff, Google Books n-grams, Multimedia Commons, ...)

Assignment 1

Study data definitions and explore data in ds_dataset.csv to gain insight into the data then develop a code using a common programming language (Python, R, Java, C#, C++, etc.) over this dataset for data cleansing and describe and document each step.

- File name of Code : HW1_1_Code.[file type]
- File name of Result : HW1_1_Result.docx
- File name of Description : HW1_1_Method.docx