Data science seminar on data curation and model interpretation

Dr. Seid Muhie Yimam
House of Computing and Data Science (HCDS)
Universität Hamburg

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

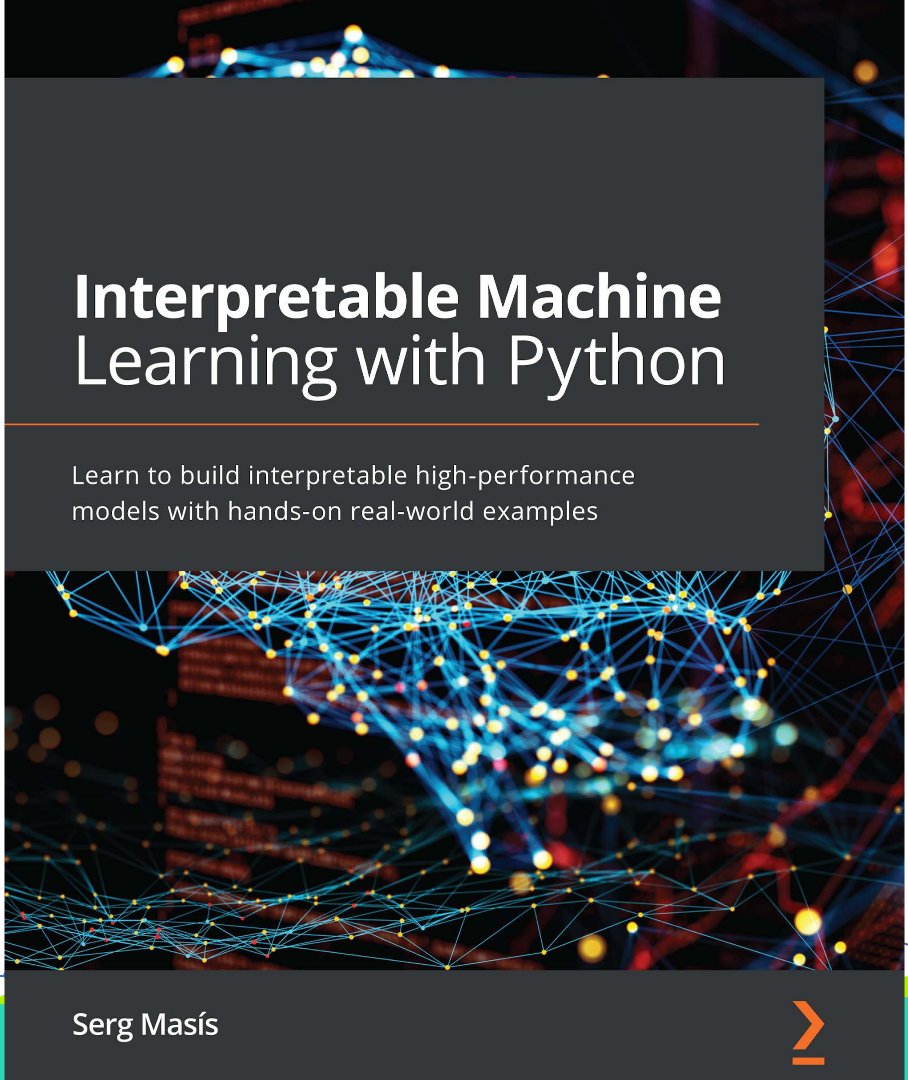
HOUSE OF COMPUTING & DATA SCIENCE

12. January 2023

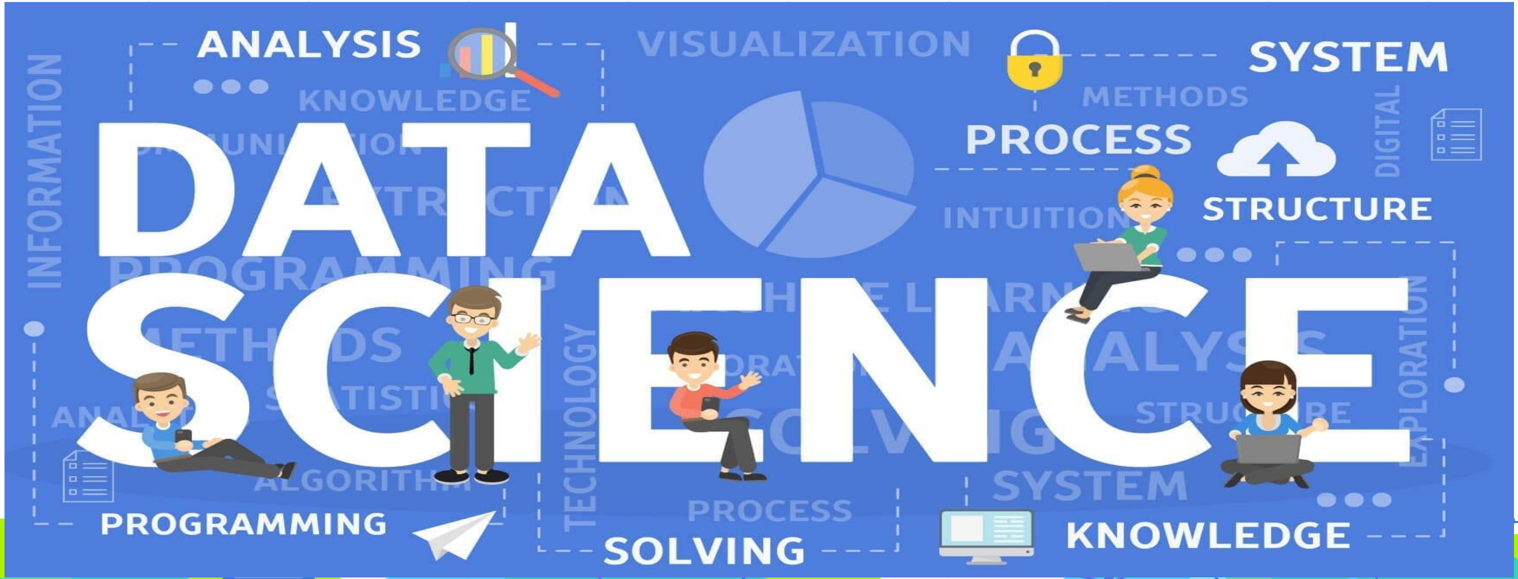
Outlines

- Part one:
  - Introduction to data science
  - Data Sources, collection
  - Annotation tools
- Part two:
  - Machine learning
  - Model building
  - Frameworks
  - Evaluation metrics
- Part three:
  - Model interpretation
  - Explainablity and Bias
  - Visualization

Reference

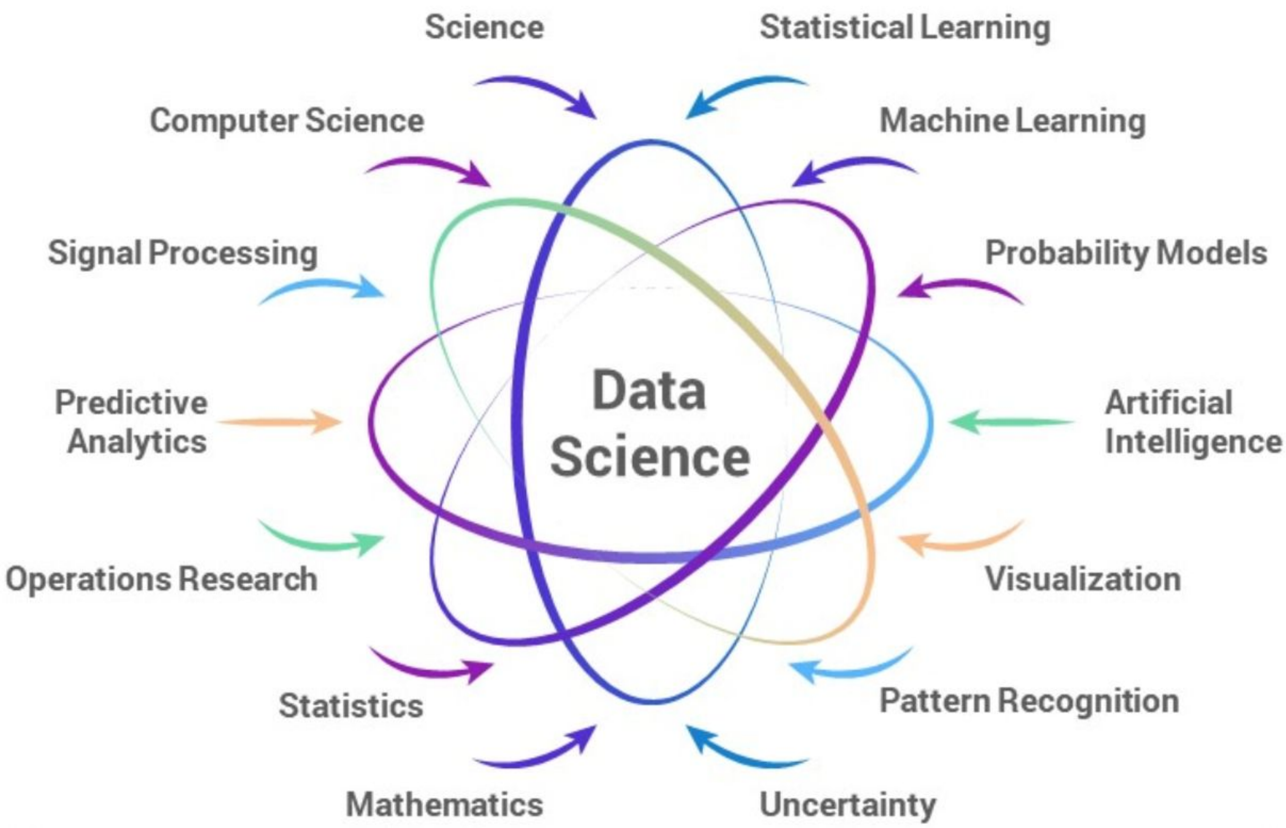

Interpretable Machine Learning with Python
Learn to build interpretable high-performance models with hands-on real-world examples
Serg Masís

Data Science
Introduction, basics

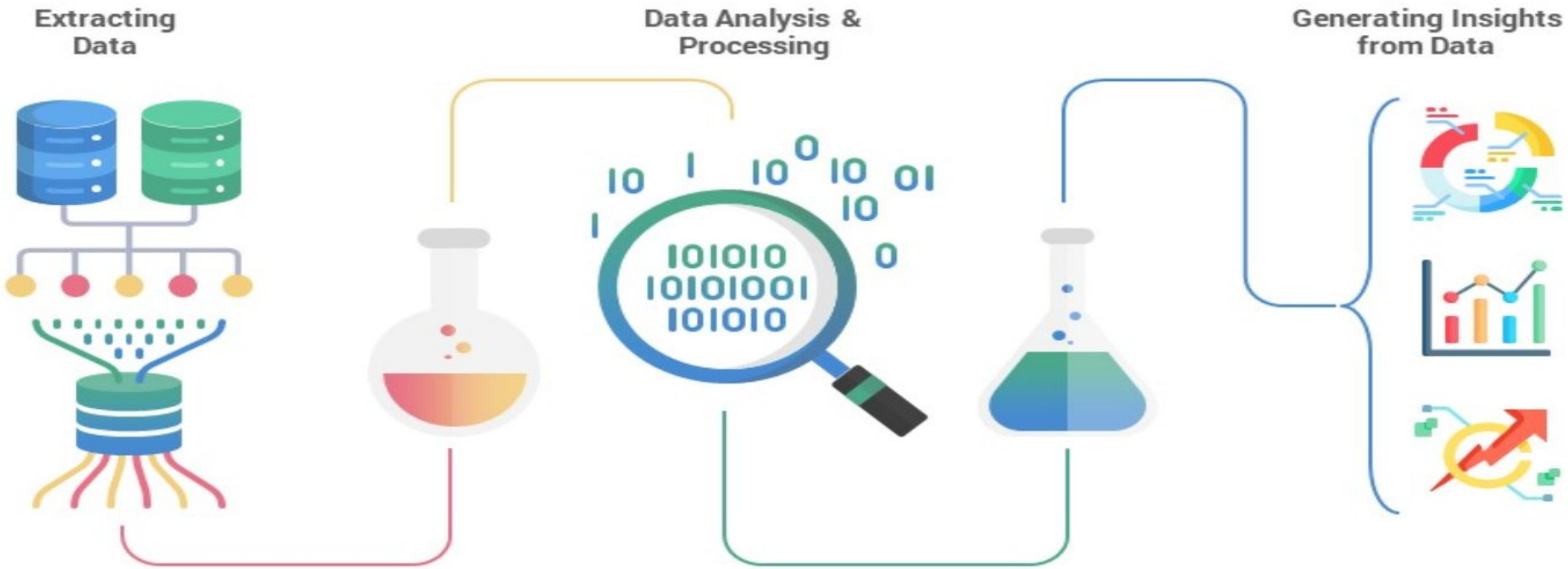


https://www.dataquest.io/wp-content/uploads/2019/05/what-is-data-science-1.jpg

Interdisciplinary



Why Data Science



Introduction

- Data science has been behind resolving some of **our most common daily tasks** for several years.
- It is rooted in **datafication**, the process of rendering into data aspects of the world that have never been quantified before.
  - **business** networks, the **lists of books** we are reading, the **films** we enjoy, the **food** we eat, our **physical** activity, our **purchases**, our **driving** behavior, and so on.
- Other ingredient of data science is the **democratization** of data analysis.
- Access to **cloud computing** allows any individual to analyze huge amounts of data in short periods of time.
- Data science is commonly defined as a methodology by which **actionable insights** can be inferred from data.

Data Science strategies

1. **Probing reality**: Data can be gathered by passive or by active methods (the **response** of the world to our actions). Analysis of those responses can be extremely valuable when it comes to taking decisions about our subsequent actions.
2. **Pattern discovery**: Datified problems can be analyzed automatically to discover useful patterns and natural clusters that can greatly simplify their solutions.
3. **Predicting future events**: Predictive analytics allows decisions to be taken in response to future events.
4. **Understanding people and the world**: Understanding natural language, computer vision, psychology and neuroscience.

Toolboxes for data scientists

- There are lot of programming language, but **Python** is the leading one
- Why Python?
  - Easy to read and code!
  - Interpreted language: **executed immediately** on console/Notebooks
  - Reach environment: Console, Ipython/Notebook, IDE

Fundamental Python Libraries for Data Scientists

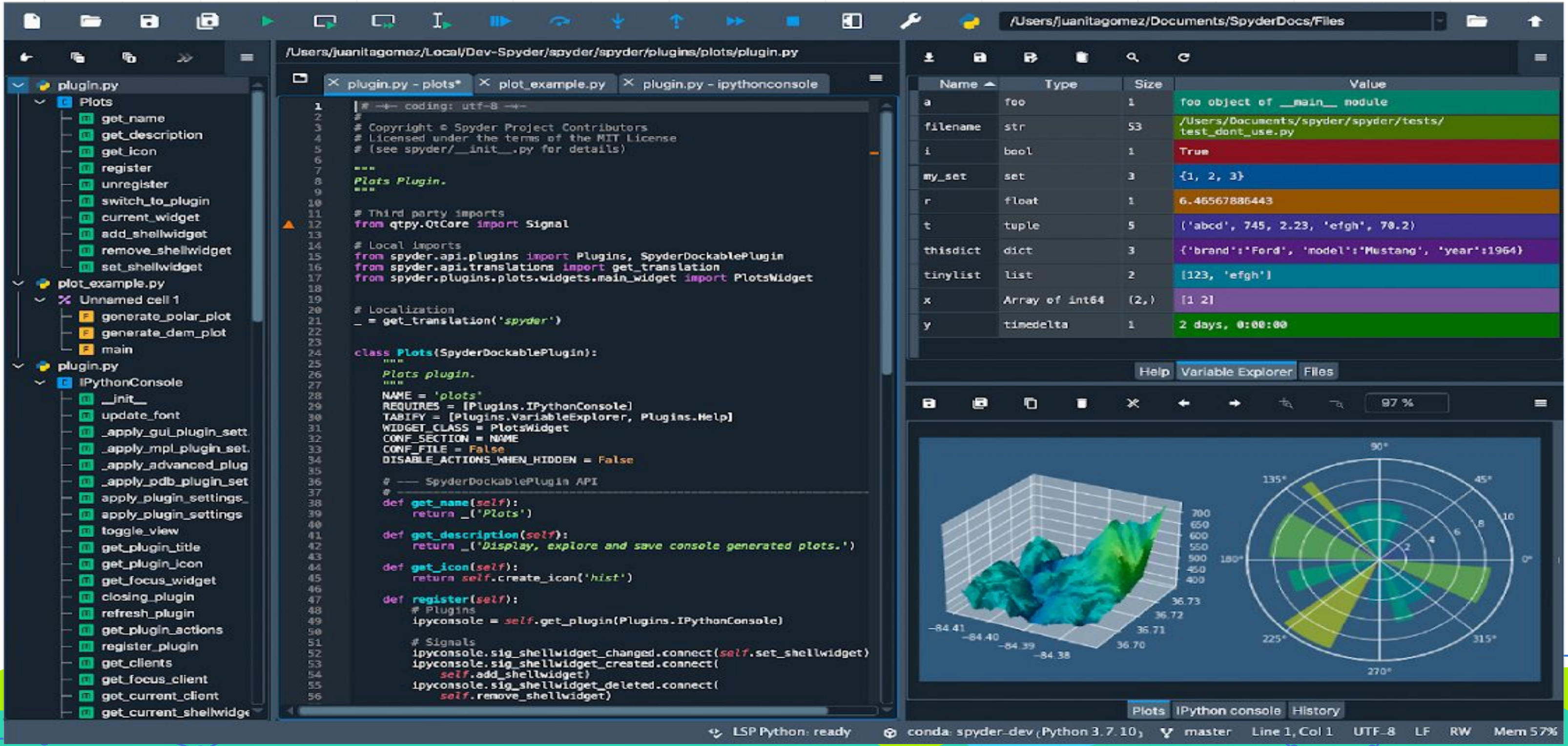
- **Numpy**: support for multidimensional arrays with basic operations on them and useful linear algebra functions.
- **SciPy**: provides a collection of numerical algorithms and domain-specific toolboxes, including signal processing, optimization, statistics, and much more
- **Pandas**: provides high-performance data structures and data analysis tools. The key feature of Pandas is a **fast** and **efficient DataFrame** object for **data manipulation** with integrated indexing.
- **Scikit-Learn**: is a machine learning library built from NumPy, SciPy, and Matplotlib. Scikit-learn offers simple and efficient tools for common tasks in data analysis such as **classification**, **regression**, **clustering**, **dimensionality reduction**, model selection, and preprocessing.
- **Matplotlib**: Used to plot or visualize results, facilitate extracting **insights** from data

Integrated Development Environments (IDE)

- The pieces of any IDE are:
  - the editor,
  - the compiler, (or interpreter) and
  - the debugger
- **NetBeans**, **Eclipse**, **PyCharm** are some general-purpose IDEs
- **Spyder** is IDE customized with the task of the data scientist in mind

spyder

The Scientific Python Development Environment

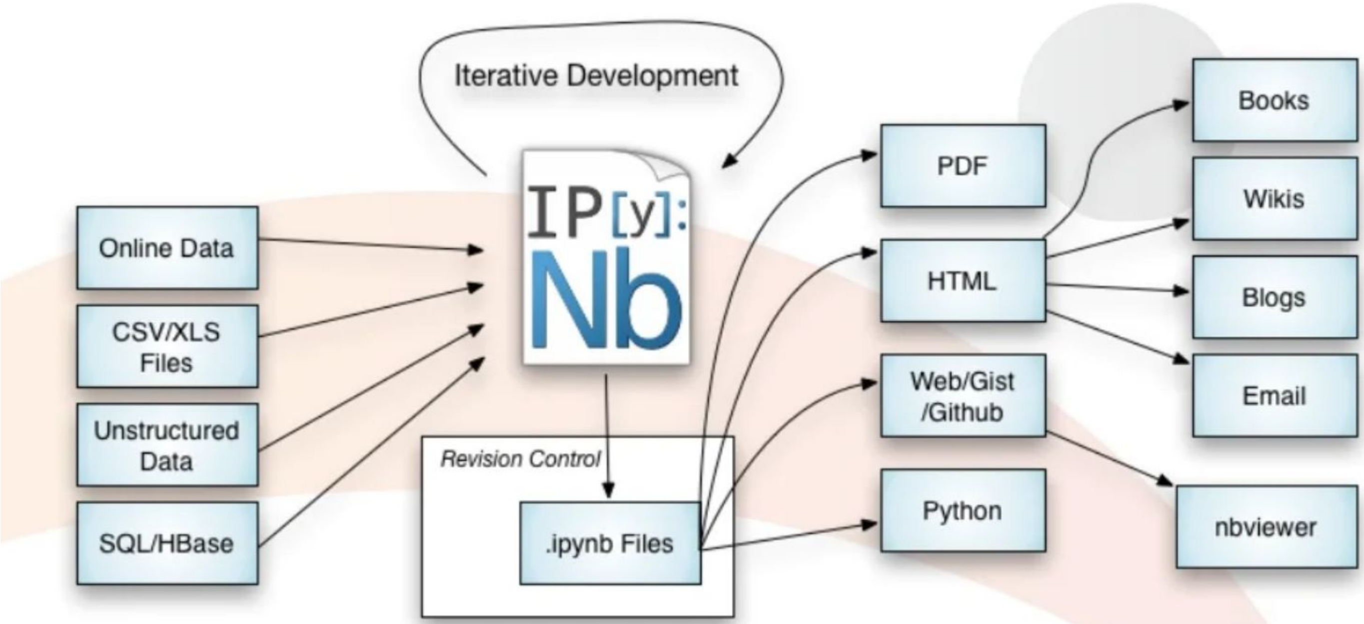


Web Integrated Development Environment (WIDE): Jupyter

- Notebooks
  - Used in classrooms
  - Used to **show results**
  - Based on IPython
  - Allow code to produce web-rich representation
    - **Image**, **sound**, **video**, **math**
  - Browser, Server, and kernels can be on different
  - .ipynb files – json based files embedding input and output

Adapted from Matthias Bussonnier slide - 2015

The Notebook Fileformat (`.ipynb`)



Adapted from Matthias Bussonnier slide - 2015

Installing/Accessing Jupyter

jupyterhub

https://code.min.uni-hamburg.de

Jupyter Notebook

Install the classic Jupyter Notebook with:

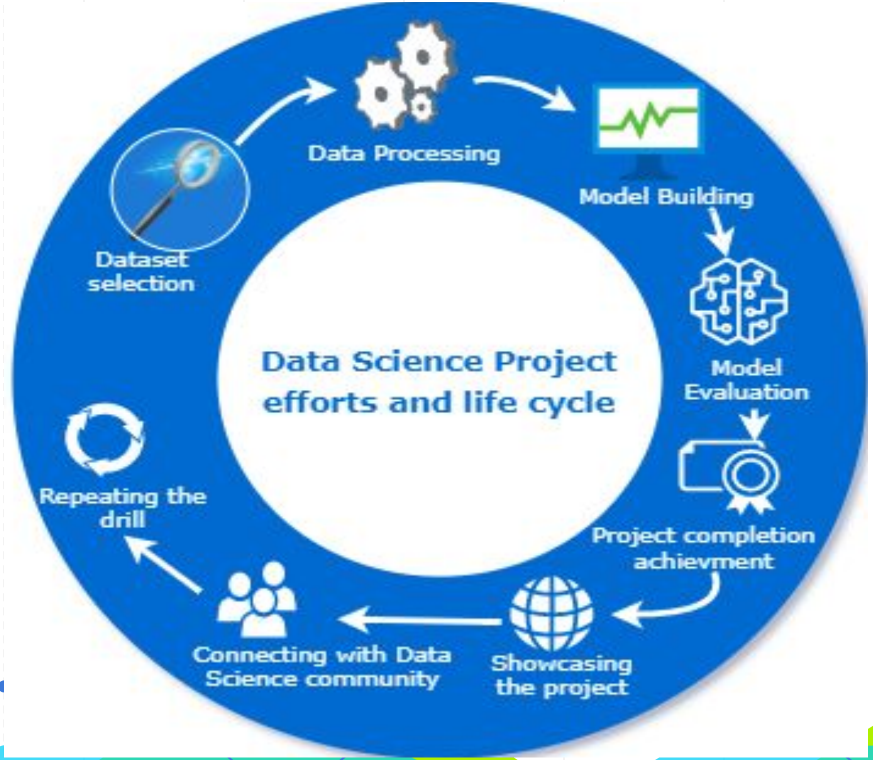
```
pip install notebook
```

Colaboratory - Google

https://colab.research.google.com

Install Anaconda and Jupyter Notebook

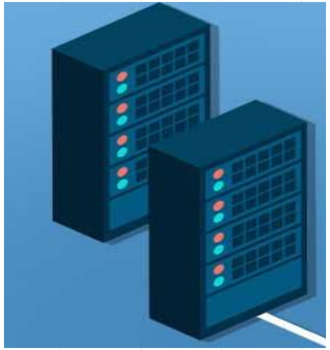
Data sources
Searching, Collection, Preparation



https://editor.analyticsvidhya.com/uploads/92084av_blog_img.PNG

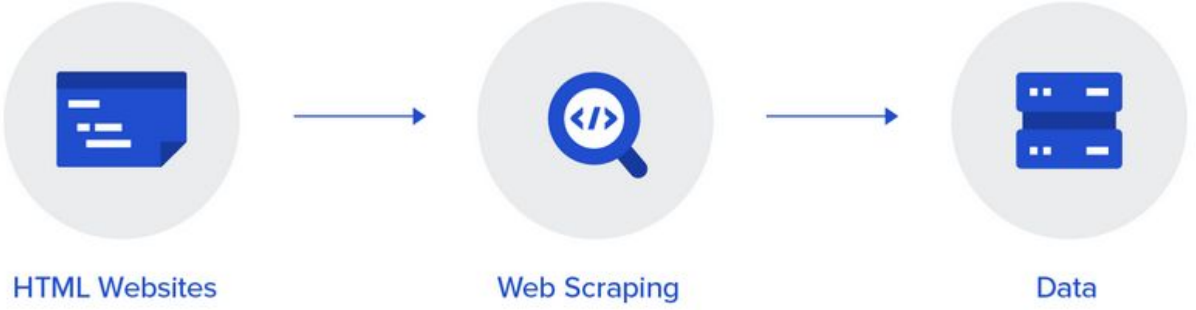
Data sources

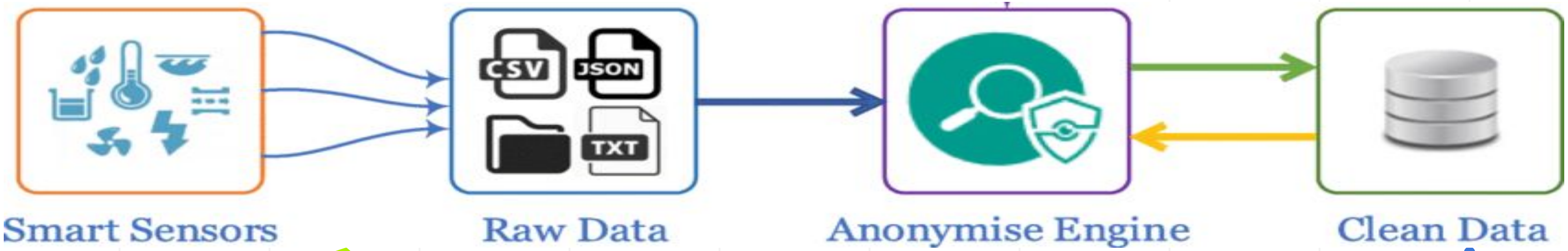
- **Primary data** – collected from primary data source
- **Preliminary data** – information gathered from primary data sources
- **Primary data sources**: Databases, files, measurements from devices (IoT), scraped from online sources, Social media, streaming data, and so on


Data collection strategies

- Data source should be identified and gathered





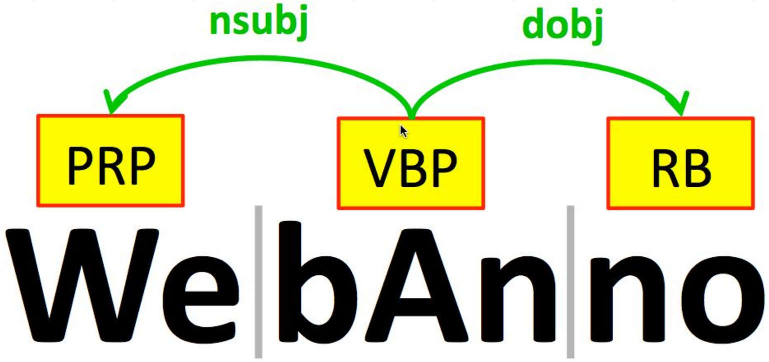
https://www.toptal.com/python/web-scraping-with-python
https://www.researchgate.net/publication/351494565/figure/fig1/AS:1022508253855759

Data Processing and Preprocessing

- Keep the original data intact, **ALWAYS**
- Data processing includes:
  - **Transformation** -> make is appropriate for model preparation
  - **Denoising** -> remove noise from data
  - **Normalization** -> organize data for more efficient access
  - **Feature extraction** -> extract relevant features or attributes that could represent the processed data

WebAnno

Annotation, curation, Automation, Agreement



What is WebAnno?

- General purpose **web-based** annotation tool
- Covers a wide range of linguistic annotations including various layers of **morphological, syntactic, and semantic** annotations
- Custom annotation layers can be defined, allowing WebAnno to be used also for **non-linguistic annotation tasks**

What is WebAnno?

- **Multi-user** tool, also different roles such as annotator, curator, and project manager
- Progress and quality of annotation projects can be monitored and measured in terms of inter-annotator agreement
- Multiple annotation projects can be conducted in parallel

What is WebAnno?

- Different modes of annotation:
  - a correction mode to review externally pre-annotated data
  - automation mode in which WebAnno learns and offers annotation suggestions
  - Curation mode to adjudicate annotation disagreements

- Fully web-based, a modern web-browser is sufficient

- After installation on a web-server, all settings can be reached through the browser

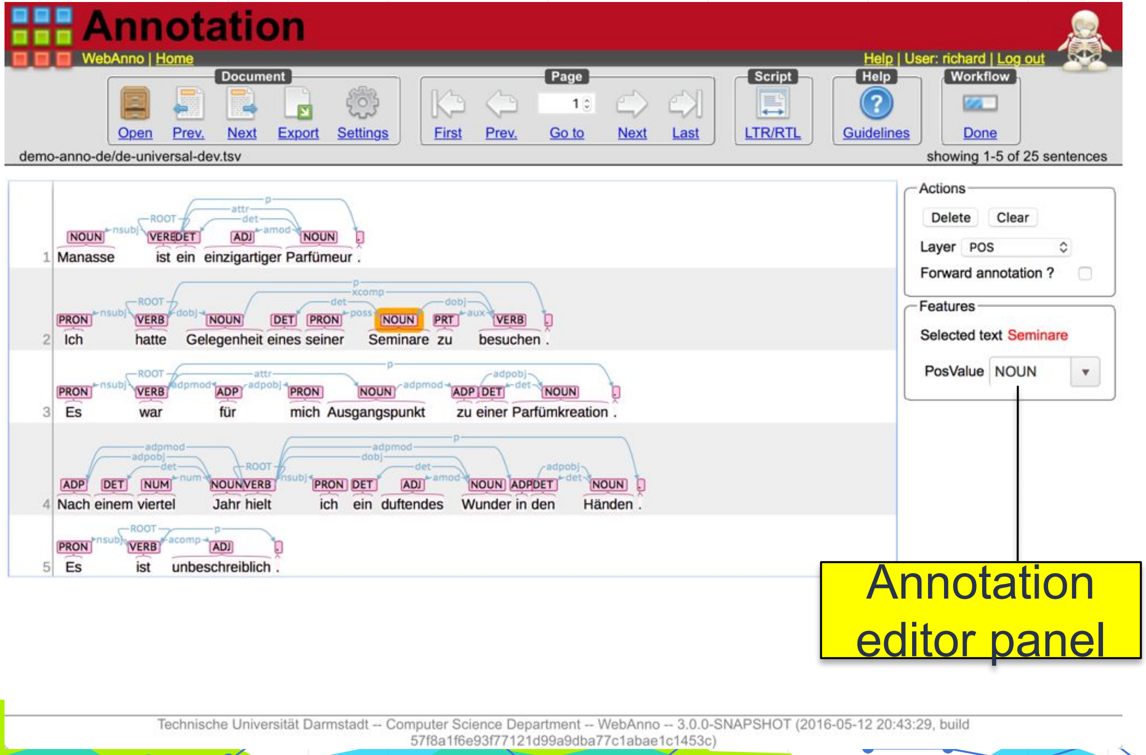
- **Open-source**

Main menu

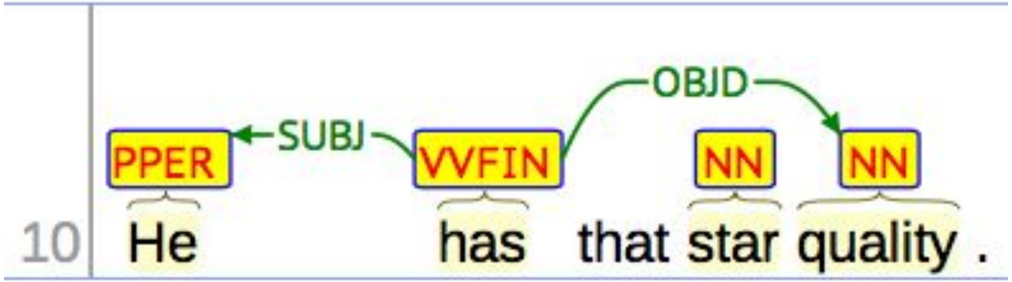
- Annotate texts from scratch
- Review and correct previously annotated documents
- Employ integrated machine learning capabilities
- Compare annotations from different annotators and merge them
- Assign workload to annotators and monitor their progress
- Create new projects

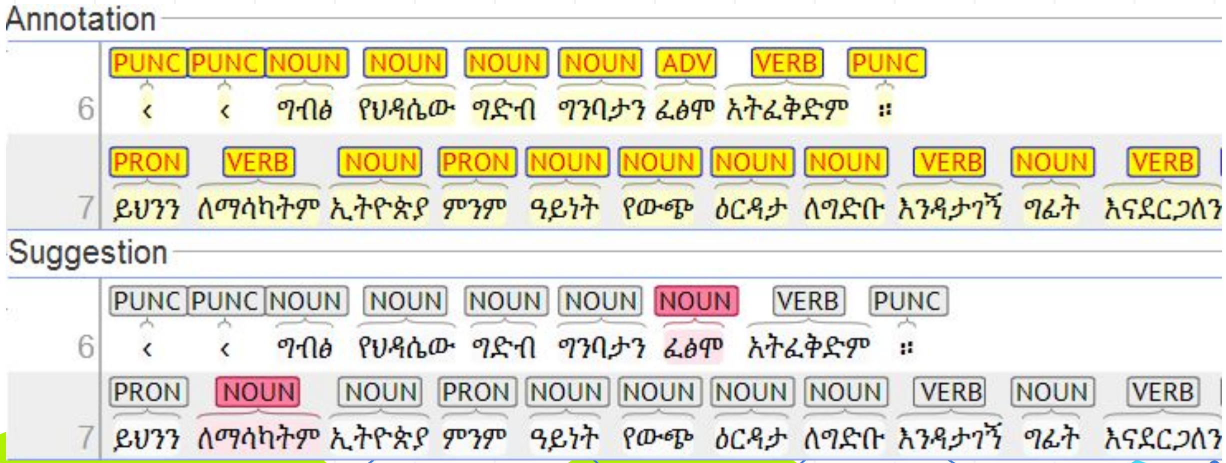
Annotation interface

- Editing elements always visible; changes take effect immediately



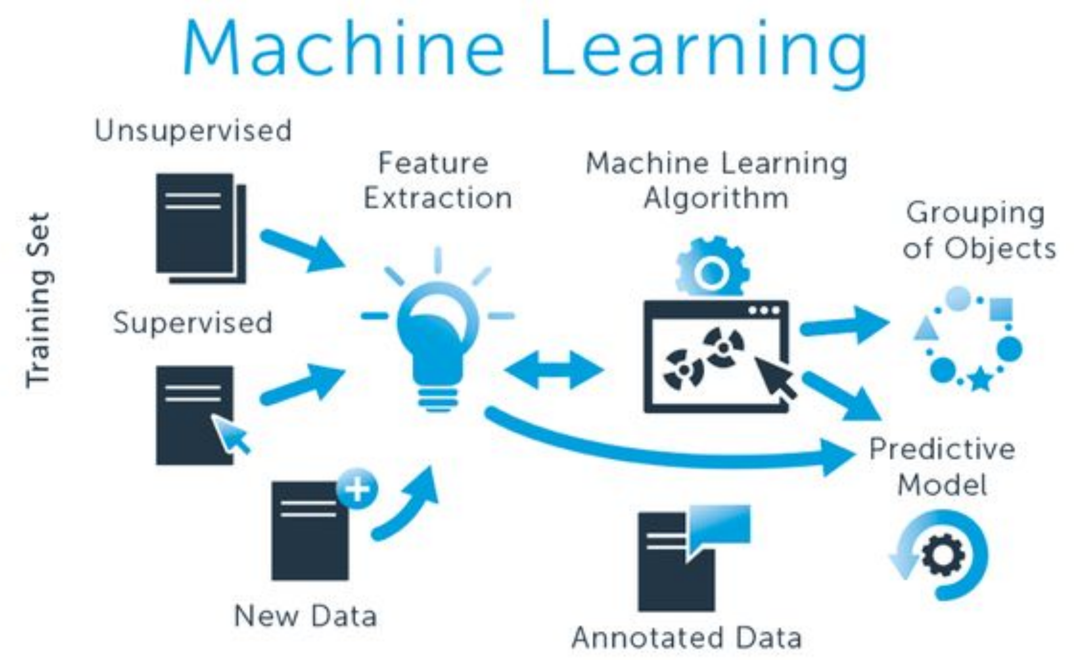
POS and dependency parsing





Machine Learning

Model building, Frameworks, Evaluation metrics



https://i.pinimg.com/originals/84/0c/ae/840cae86750d66930bff80331f8b9b79.png

What is learning

- Herbert Simon: “Learning is any process by which a system **improves performance** from **experience**.”
- “A computer program is said to **learn** from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**.”

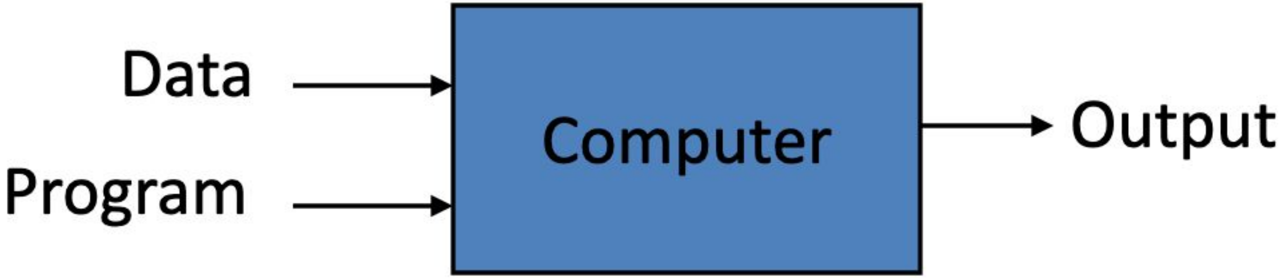
–Tom Mitchell

What is machine learning

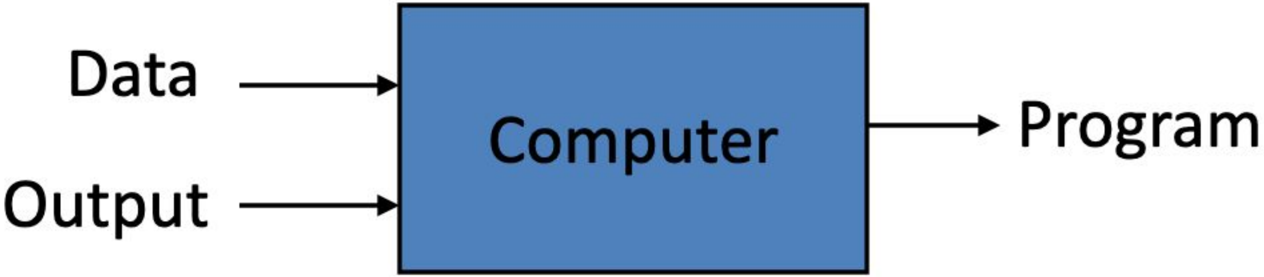
- ML is a branch of **artificial intelligence**:
  - Uses computing based systems to make **sense out of data**
    - Extracting **patterns**, **fitting** data to **functions**, **classifying** data, etc
  - ML systems can **learn** and **improve**
    - With **historical data**, **time** and **experience**
  - Bridges **theoretical computer science** and **real noise data**.

Slide adapted from Eric Eaton

Traditional Programming



Machine Learning

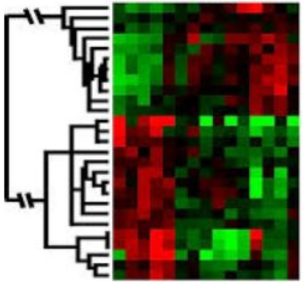


When do we use machine learning?

ML is used when:

- Human expertise does not exist (navigating on Mars)
- Humans can't explain their expertise (speech recognition)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)



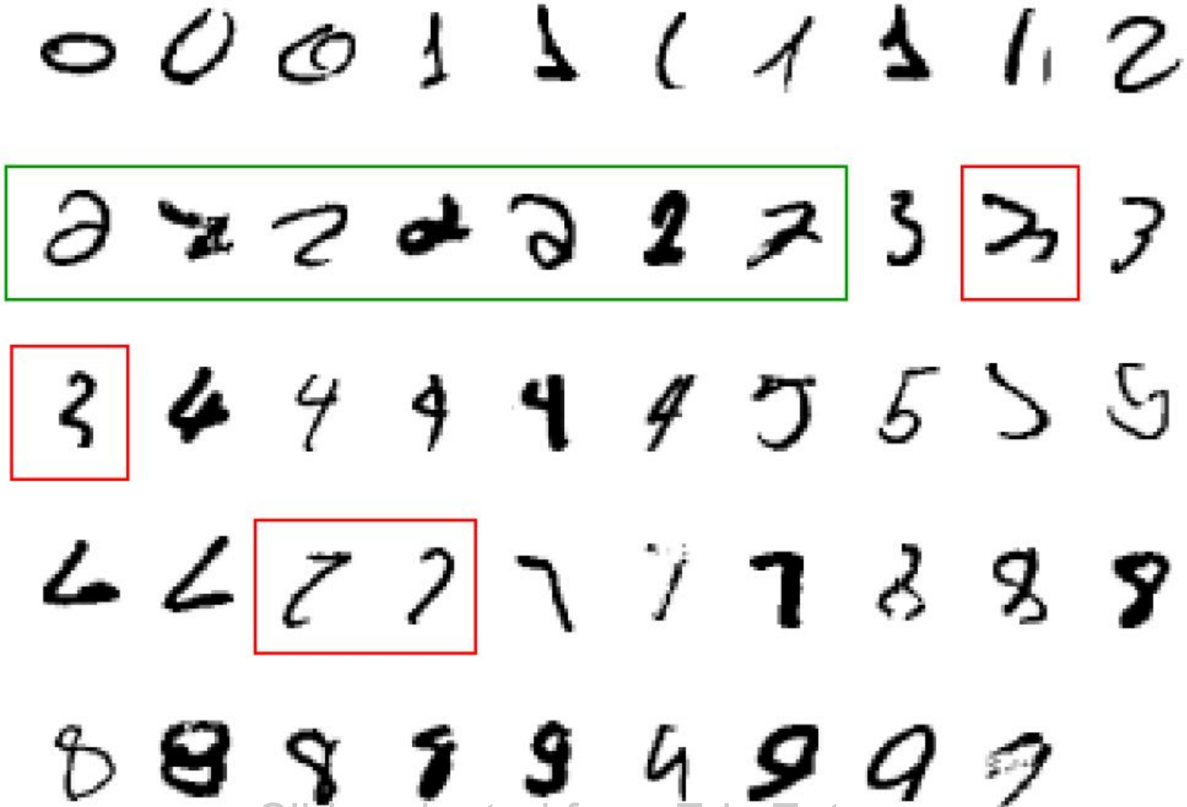



Learning isn't always useful:

- There is no need to "learn" to calculate payroll

Slide adapted from Eric Eaton

A classic example of a task that requires machine learning: It is very hard to say what makes a 2



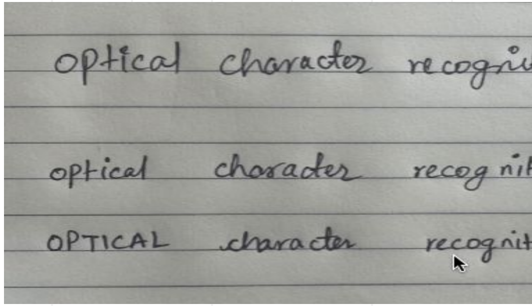
Slide adapted from Eric Eaton

Defining the learning task

Improve on task T, with respect to performance metric P, based on experience E

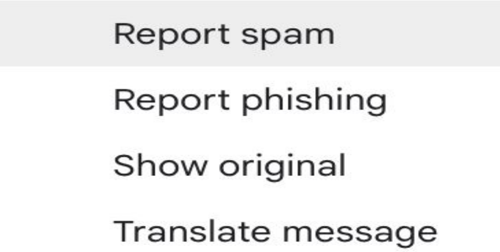
T: Playing checkers
P: Percentage of games won against an arbitrary opponent
E: Playing practice games against itself

T: Recognizing hand-written words
P: Percentage of words correctly classified
E: Database of human-labeled images of handwritten words



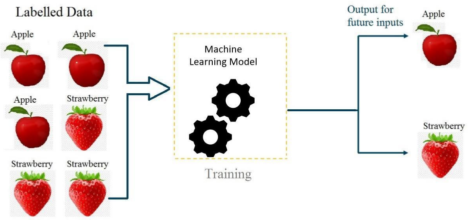
T: Driving on four-lane highways using vision sensors
P: Average distance traveled before a human-judged error
E: A sequence of images and steering commands recorded while observing a human driver.

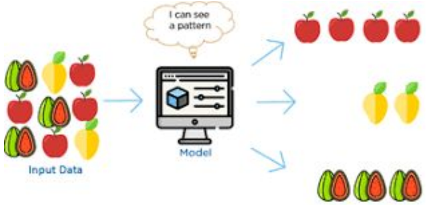
T: Categorize email messages as spam or legitimate.
P: Percentage of email messages correctly classified.
E: Database of emails, some with human-given labels

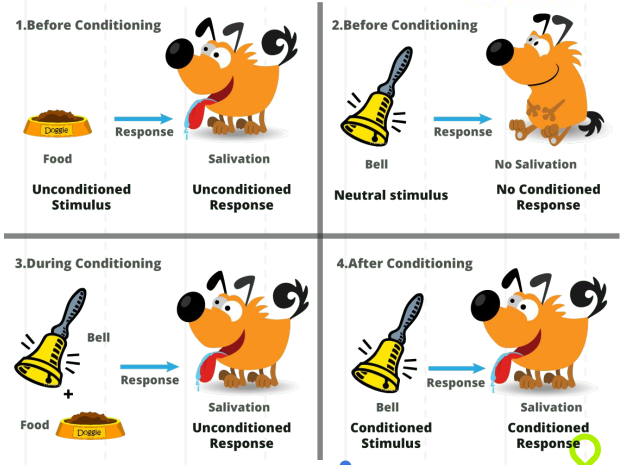


Types of learning

- Supervised (inductive) learning
  - Given: training data + desired outputs (labels)
- Unsupervised learning
  - Given: training data (without desired outputs)
- Semi-supervised learning
  - Given: training data + a few desired outputs
- Reinforcement learning
  - Rewards from sequence of actions

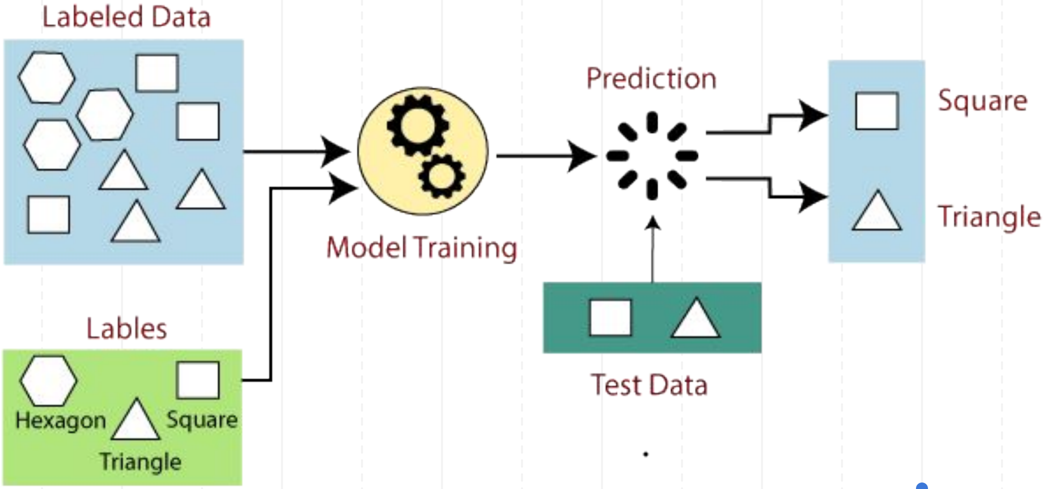






Supervised learning

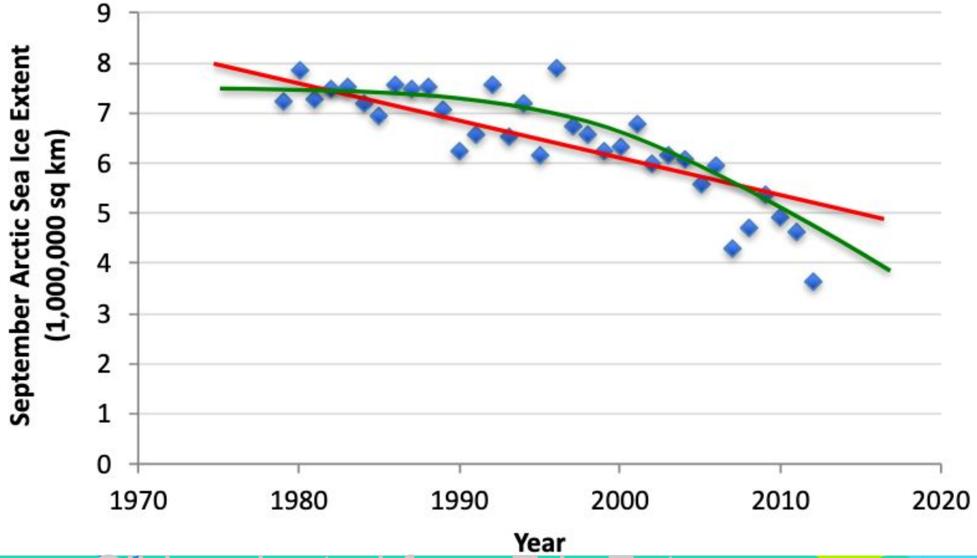
- For every example in the data there is always a predefined outcome
- Models the relations between a set of descriptive features and a target (Fits data to a function)
- 2 groups of problems:
  - Classification
  - Regression



https://static.javatpoint.com/tutorial/machine-learning/images/supervised-machine-learning.png

Supervised learning - regression

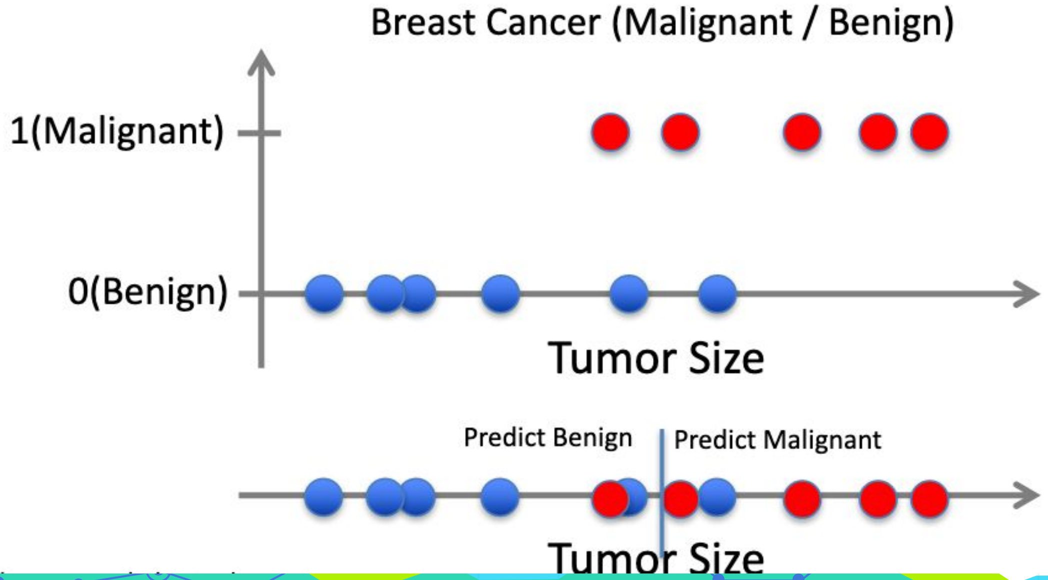
- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x
  - y is real-valued == regression



Slide adapted from Eric Eaton

Supervised learning - classification

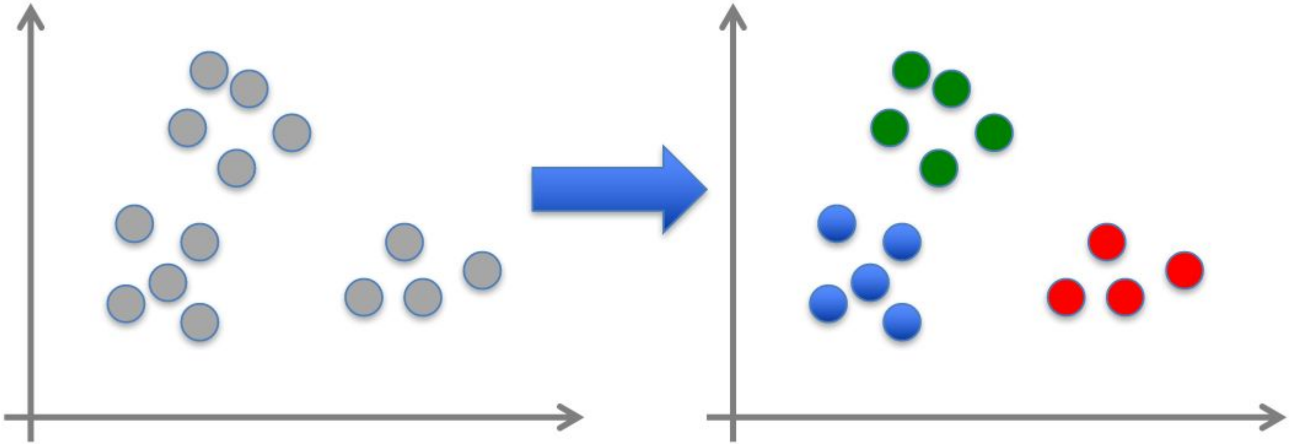
- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function $f(x)$ to predict y given x
  - y is categorical == classification



Slide adapted from Eric Eaton

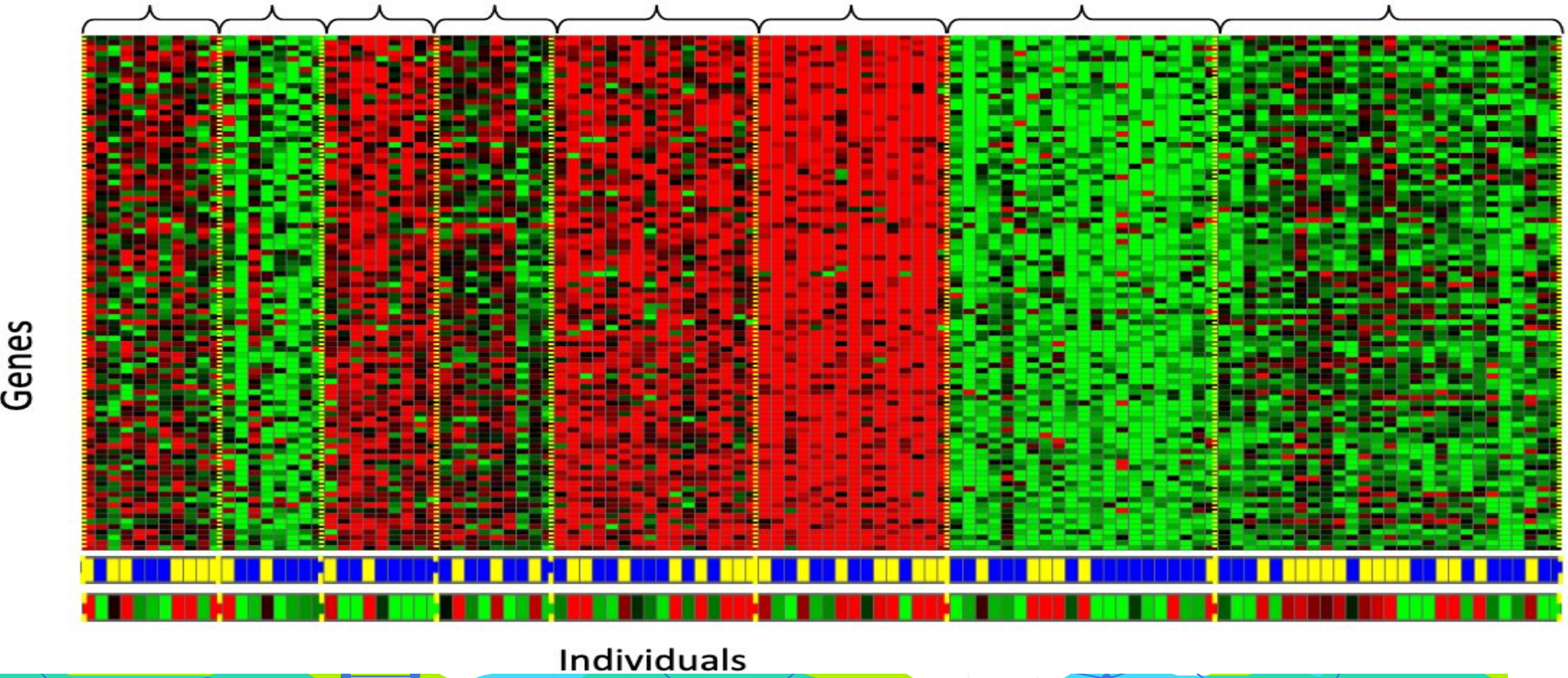
Unsupervised learning

- Given x_1, x_2, …, x_n (without labels)
- Output hidden structure behind the x's
  - E.g., clustering



Slide adapted from Eric Eaton

Clustering of gene-expression

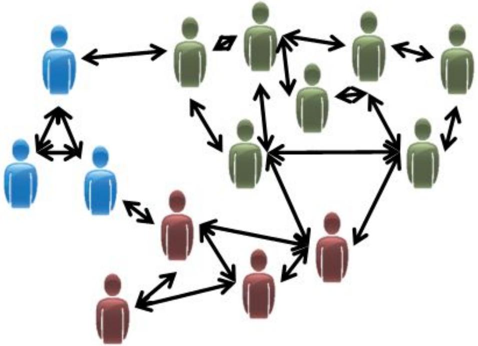


Slide adapted from Eric Eaton

Unsupervised learning

Slide adapted from Eric Eaton

Organize computing clusters



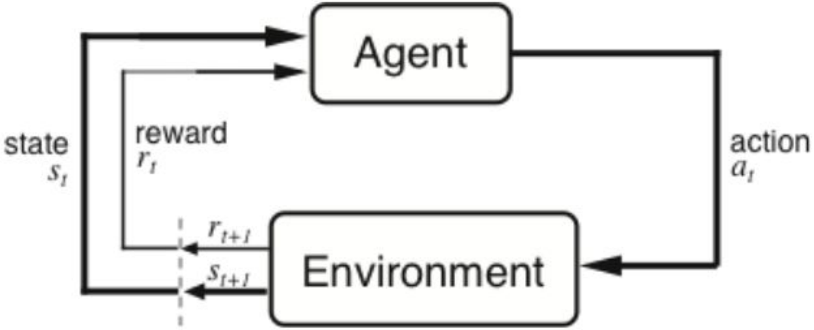
Social network analysis

Market segmentation

Astronomical data analysis

The agent-environment interface

Slide adapted from Eric Eaton



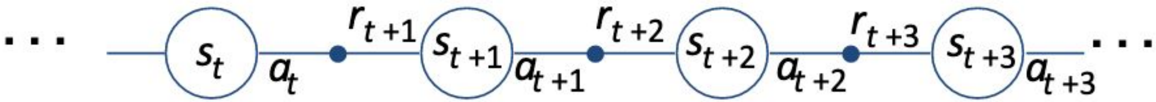
Agent and environment interact at discrete time steps : $t = 0, 1, 2, K$

Agent observes state at step t: $s_t \in S$

produces action at step t: $a_t \in A(s_t)$

gets resulting reward : $r_{t+1} \in \Re$

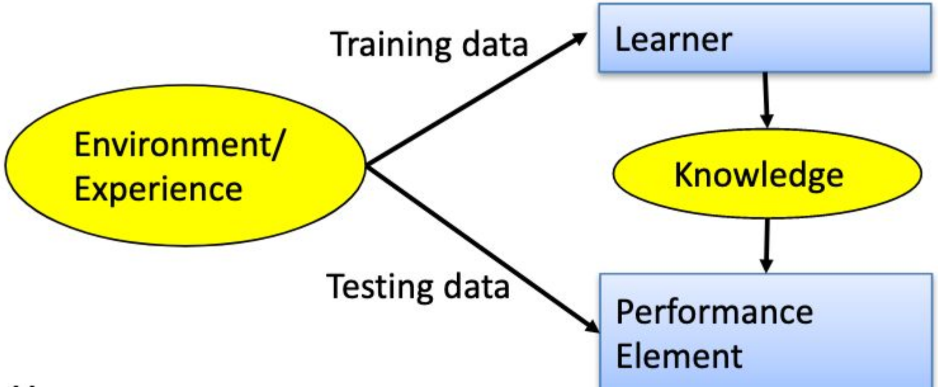
and resulting next state : s_{t+1}



Designing a learning system

Slide adapted from Eric Eaton

- Choose the training experience
- Choose exactly what is to be learned
  - i.e. the ***target function***
- Choose how to represent the target function
- Choose a learning algorithm to infer the target function from the experience



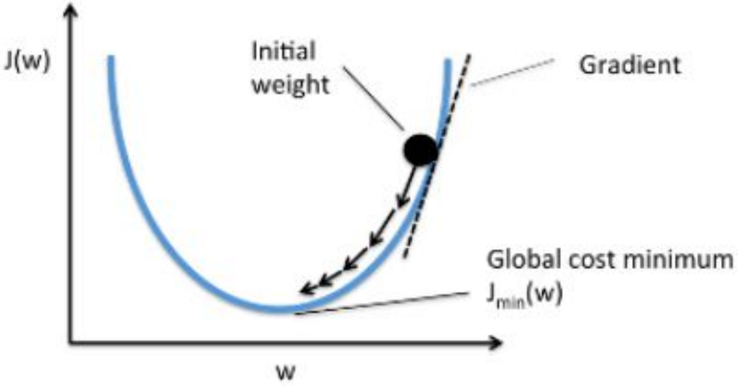
Learning algorithm – linear regression

- F(x) = WX + b
  - W = Weights to learn
  - X = Features from the input
  - b = bias term
- The task ***T*** is to predict *y, which is F(X),* from *X*, we need to measure performance ***P*** to know how well the model performs.
- First calculate error of each example i as : $e_i = abs(\hat{y}_i - y_i)$
- Finally calculate the mean for all records: $1/m \sum_i abs(\hat{y}_i - y_i)$
  - Mean Absolute Error (MAE) =

$$1/2m \sum_i (\hat{y}_i - y_i)^2$$

Learning algorithm – linear regression

- The main aim of training the ML algorithm is to adjust the weights ***W*** to reduce the **MAE** or **MSE**
- This is called the **cost function**, J(w) □ minimaxing the error is minimizing the cost function J
- Gradient decent Algorithm
  - Jmin □ minimum cost for W
  - Gradient decent algorithm:



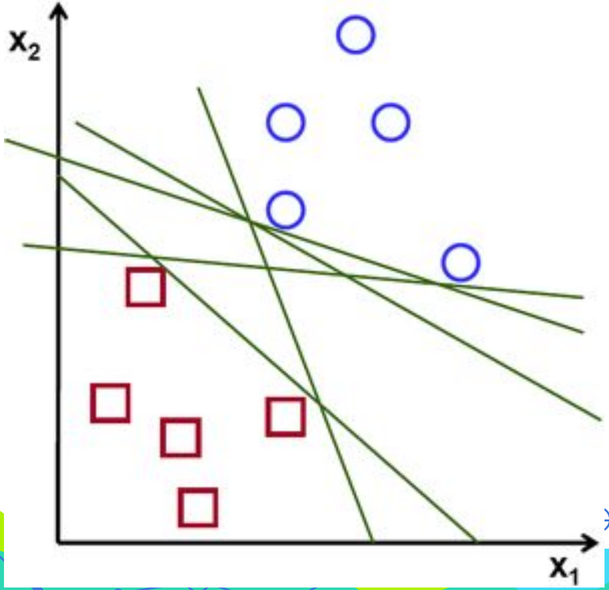
repeat until minimum cost: {

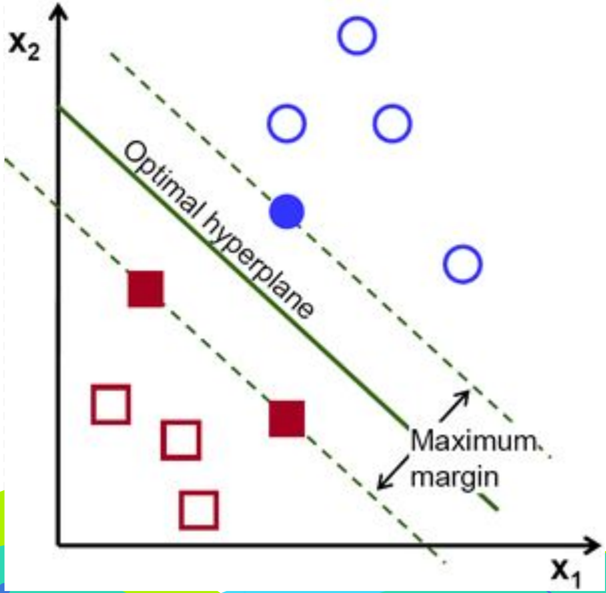
$$w_j = w_j - \alpha \partial/\partial w_j J(W)$$

}

Learning algorithm - SVM

- The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

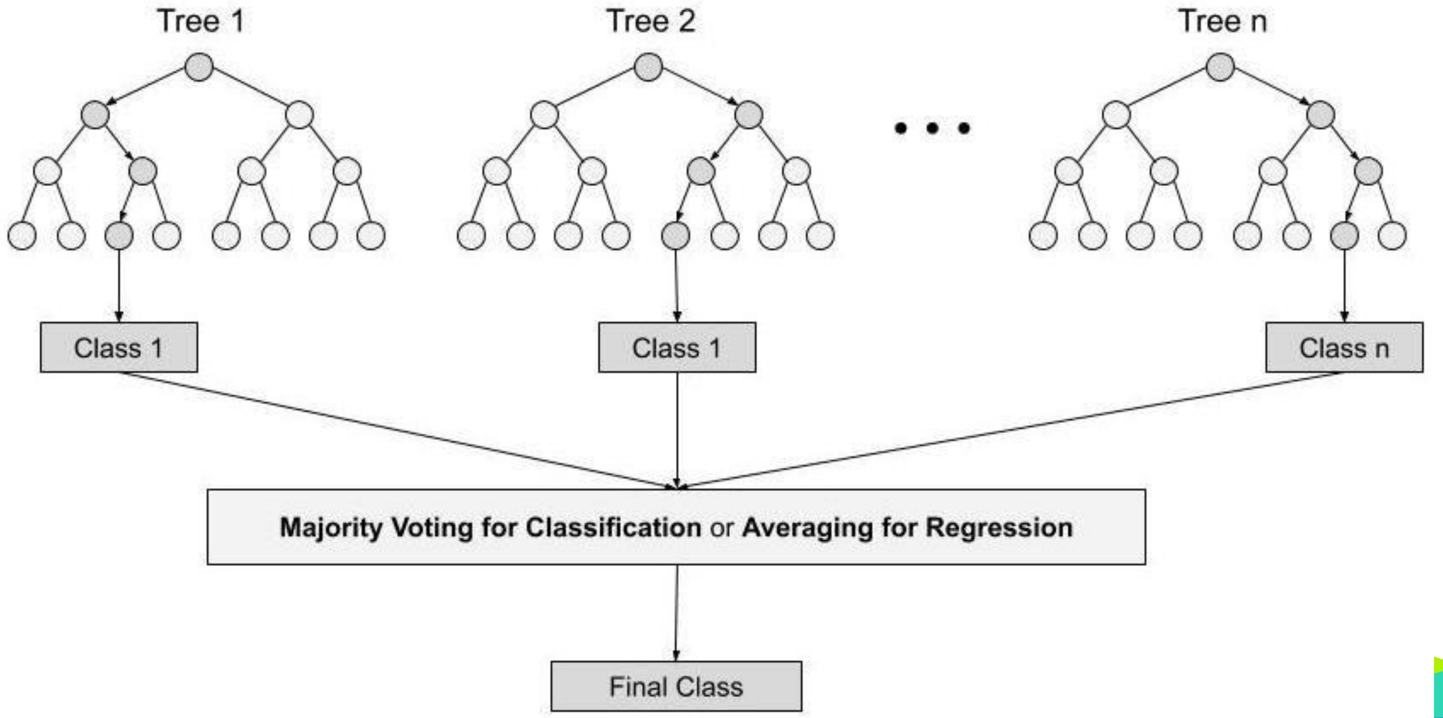




https://miro.medium.com/max/600/0*9jEWNXTAao7phK-5.png

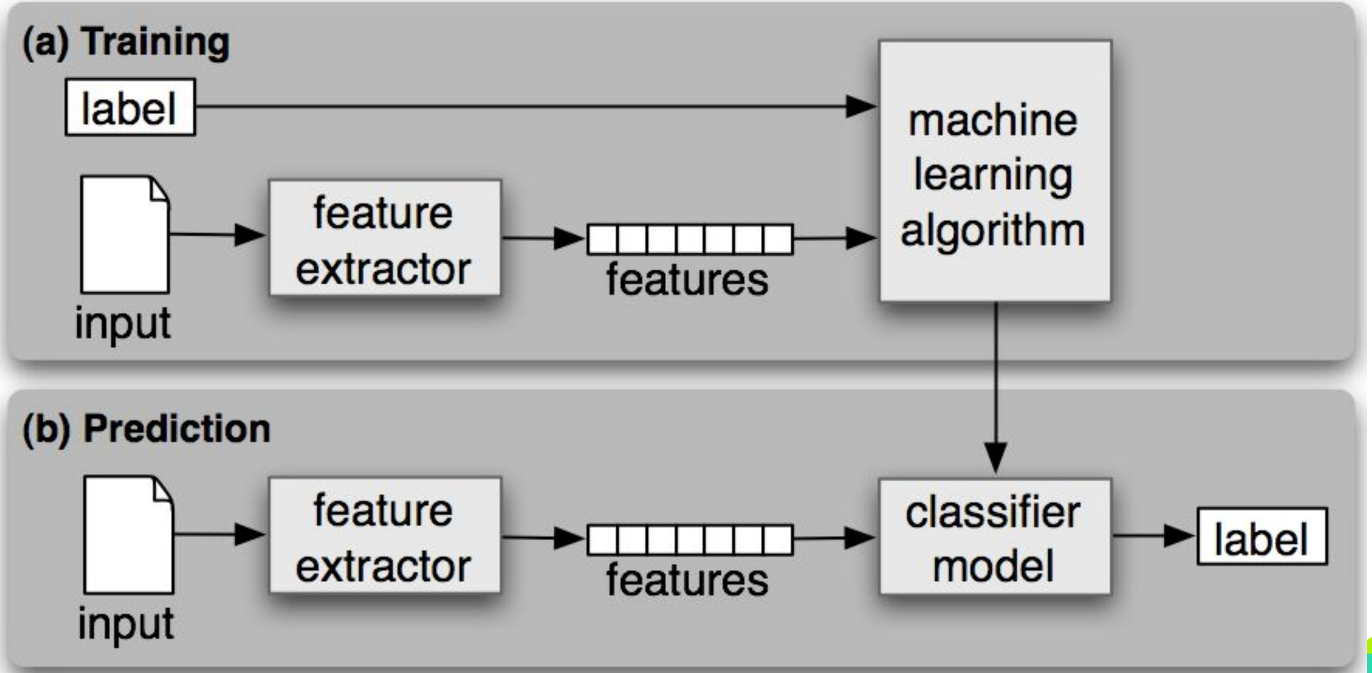
Learning algorithm – Random forest

- Random Forest builds decision trees on different samples and takes their majority vote for classification and average in case of regression.



Feature extraction

The pipeline for supervised classification looks like the following:



Feature extraction

Problem: Sentiment Analysis (detect positive and negative attitude of text)

Given: Training data

Instance	Class Label
I like hamsters very much.	True
I cannot stand dogs.	False
I love my cat.	True

Extract Features

like	love	hate	I	Class Label
1	0	0	1	True
0	0	0	1	False
0	1	0	1	True

Train a model which is able to predict the class label

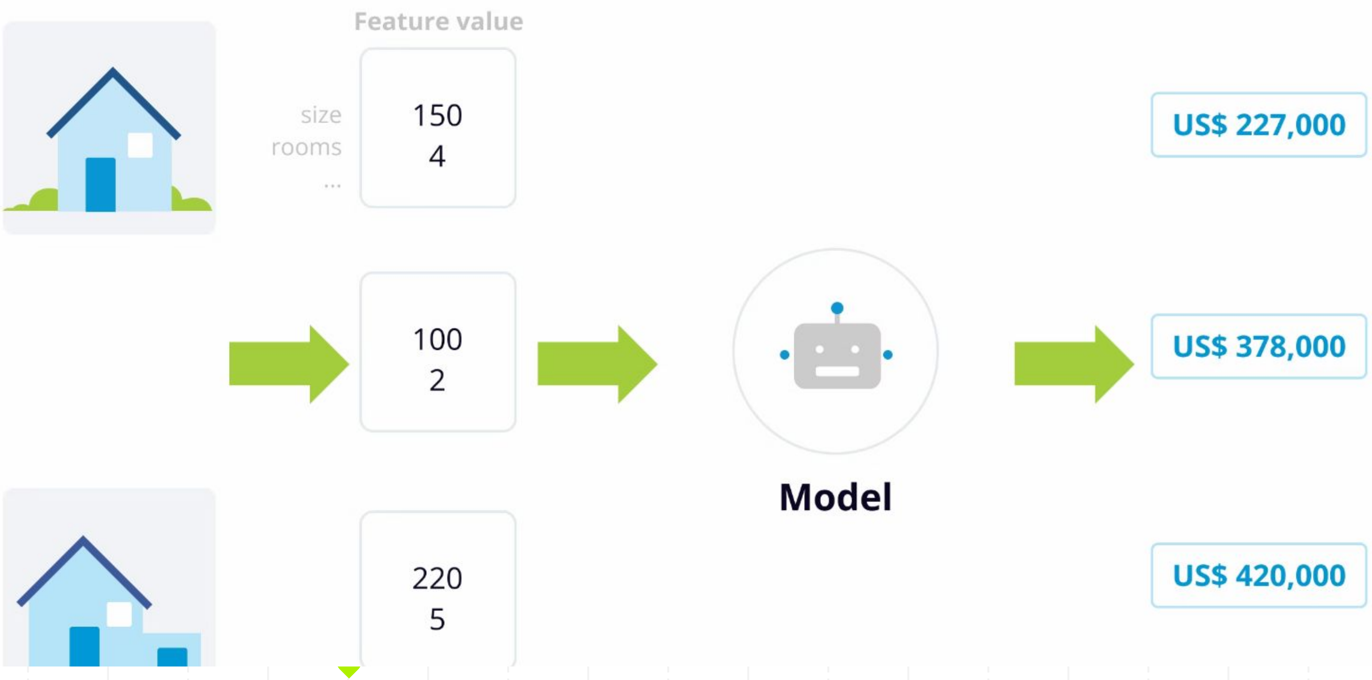
Rental price prediction

- Predicting the rental price of single-family houses will have the these features:

Features

- Number of bedrooms
- Number of bathrooms
- Living area
- Number of stories
- Year built
- Furnished/not furnished
- Fireplace/no fireplace
- Heating/no heating
- ZIP code
- Latitude and longitude

House sales prediction



Named entity and Part-of-speech tagging

- Data with annotation

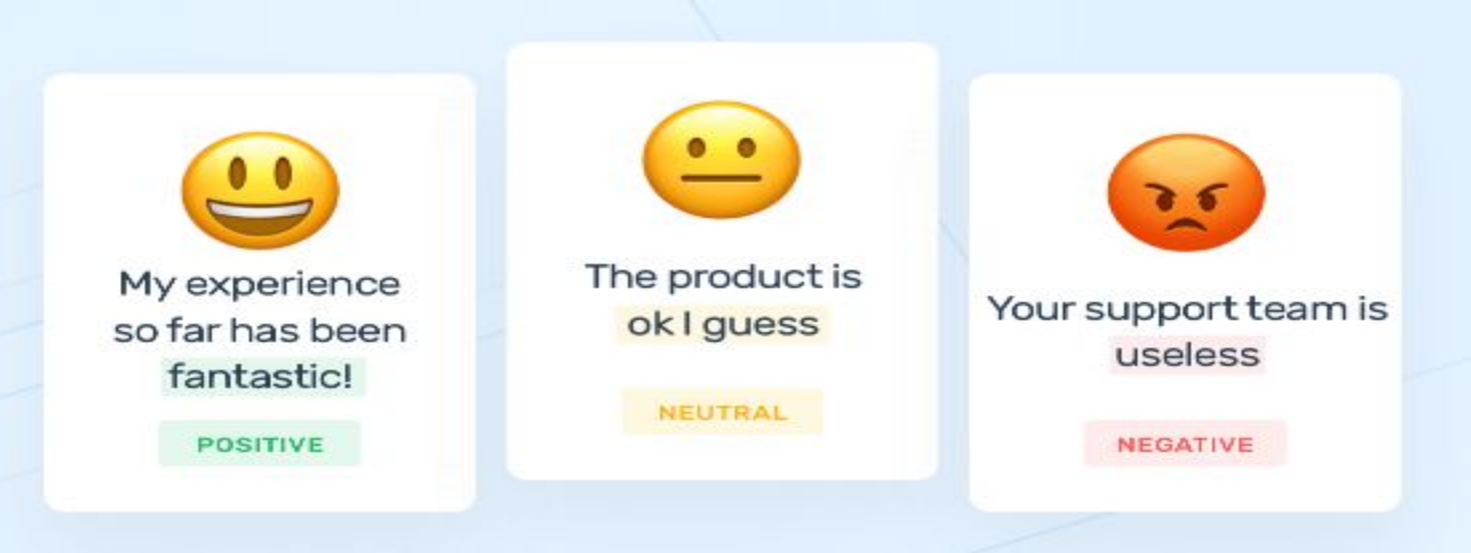
U.N.	NNP	I-NP	I-ORG
official	NN	I-NP	O
Ekeus	NNP	I-NP	I-PER
heads	VBZ	I-VP	O
for	IN	I-PP	O
Baghdad	NNP	I-NP	I-LOC
.	.	O	O

- Features:

  IsFirstUpper, **prefex-n**, **suffix-n**, the **token**, **length**, **lemma**, **PoS**, **isInGazetter**, **isGeoLocation**,

- Class Labels: PER, ORG, LOC, OTH,

Sentiment classification



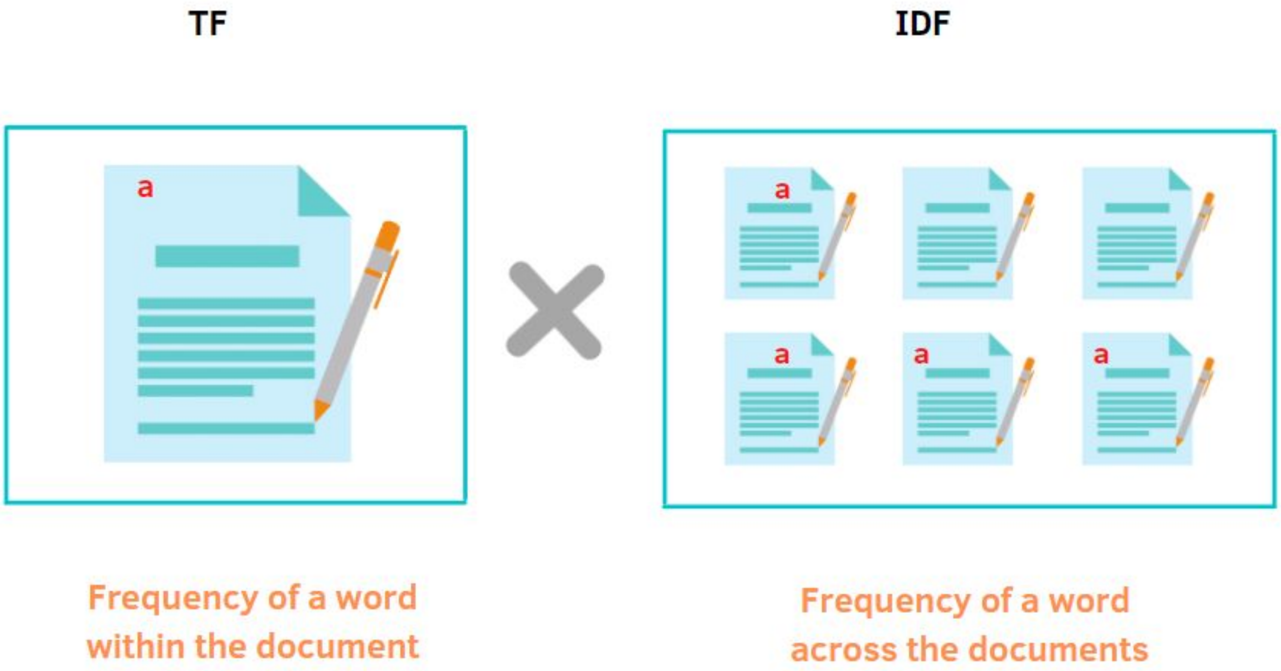
- Features
  - **Bag of words**, **bag-of-ngrams**, **TFIDF**, word vectors (**embeddings**)

Sentiment features - bag of words

	about	bird	heard	is	the	word	you
About the bird, the bird, bird bird bird	1	5	0	0	2	0	0
You heard about the bird	1	1	1	0	1	0	1
The bird is the word	0	1	0	1	2	1	0

https://user.oc-static.com/upload/2020/10/23/16034397439042_surfin%20bird%20bow.png

TF-IDF

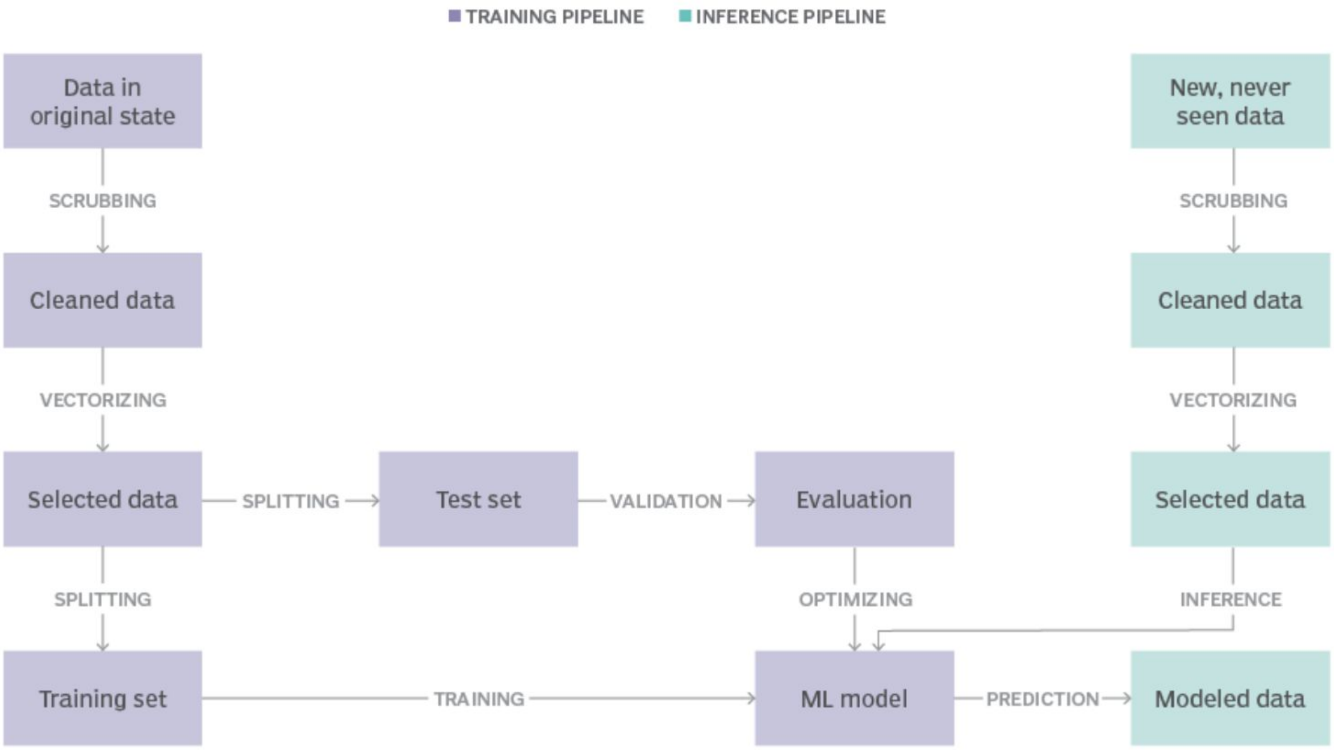


https://miro.medium.com/max/943/1*HZvxT29V9B4HxT2wx8M4XQ.png

Image classification – feature extraction

0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	125	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	255	249	255	240	255	129	0	5
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	14	1	0	6	6	0	0

Model building pipeline



https://cdn.ttgtmedia.com/rms/onlineimages/enterprise_ai-machine_learning_03-f.png

ML frameworks

- Classical ML Algorithms in Sickit-Learn
- Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python.
  - Is built upon **NumPy**, **SciPy** and **Matplotlib**.
  - Can be installed with Anaconda, conda , or pip

```
pip install -U scikit-learn
```

```
conda install scikit-learn
```

Sklearn

Supervised Learning algorithms: Almost all the popular supervised learning algorithms, like Linear Regression, Support Vector Machine (SVM), Decision Tree etc., are the part of scikit-learn.

Unsupervised Learning algorithms: On the other hand, it also has all the popular unsupervised learning algorithms from clustering, factor analysis, PCA (Principal Component Analysis) to unsupervised neural networks.

Clustering: This model is used for grouping unlabeled data.

Cross Validation: It is used to check the accuracy of supervised models on unseen data.

Dimensionality Reduction: It is used for reducing the number of attributes in data which can be further used for summarisation, visualisation and feature selection.

Ensemble methods: As name suggest, it is used for combining the predictions of multiple supervised models.

Feature extraction: It is used to extract the features from data to define the attributes in image and text data.

Feature selection: It is used to identify useful attributes to create supervised models.
Open Source: It is open source library and also commercially usable under BSD license.

https://www.tutorialspoint.com/scikit_learn/scikit_learn_tutorial.pdf

Sklearn example iris dataset

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
iris = load_iris()
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(105, 4)
(45, 4)
(105,)
(45,)
```

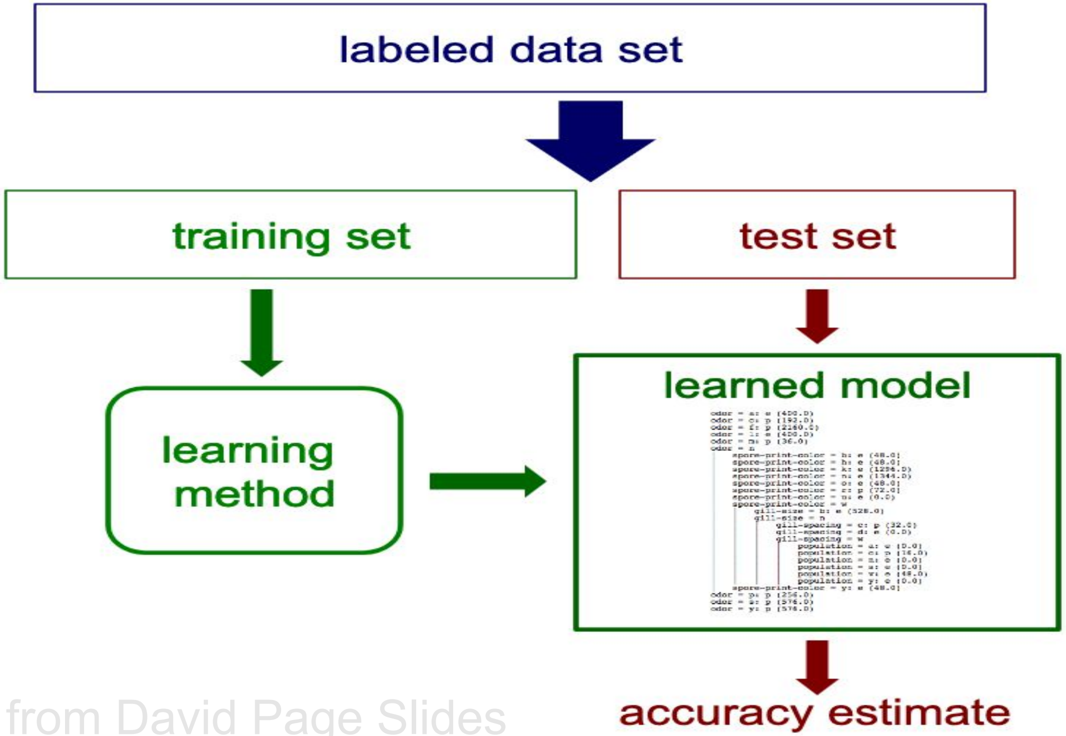
Sklearn – train a model

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
classifier_knn = KNeighborsClassifier(n_neighbors=3)
classifier_knn.fit(X_train, y_train)
y_pred = classifier_knn.predict(X_test)
# Finding accuracy by comparing actual response values(y_test)
# with predicted response value(y_pred)
print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
# Providing sample data and the model will make prediction out of that data
sample = [[5, 5, 3, 2], [2, 4, 3, 5]]
preds = classifier_knn.predict(sample)
pred_species = [iris.target_names[p] for p in preds]
print("Predictions:", pred_species)
```

```
Accuracy: 0.9777777777777777
Predictions: ['versicolor', 'virginica']
```

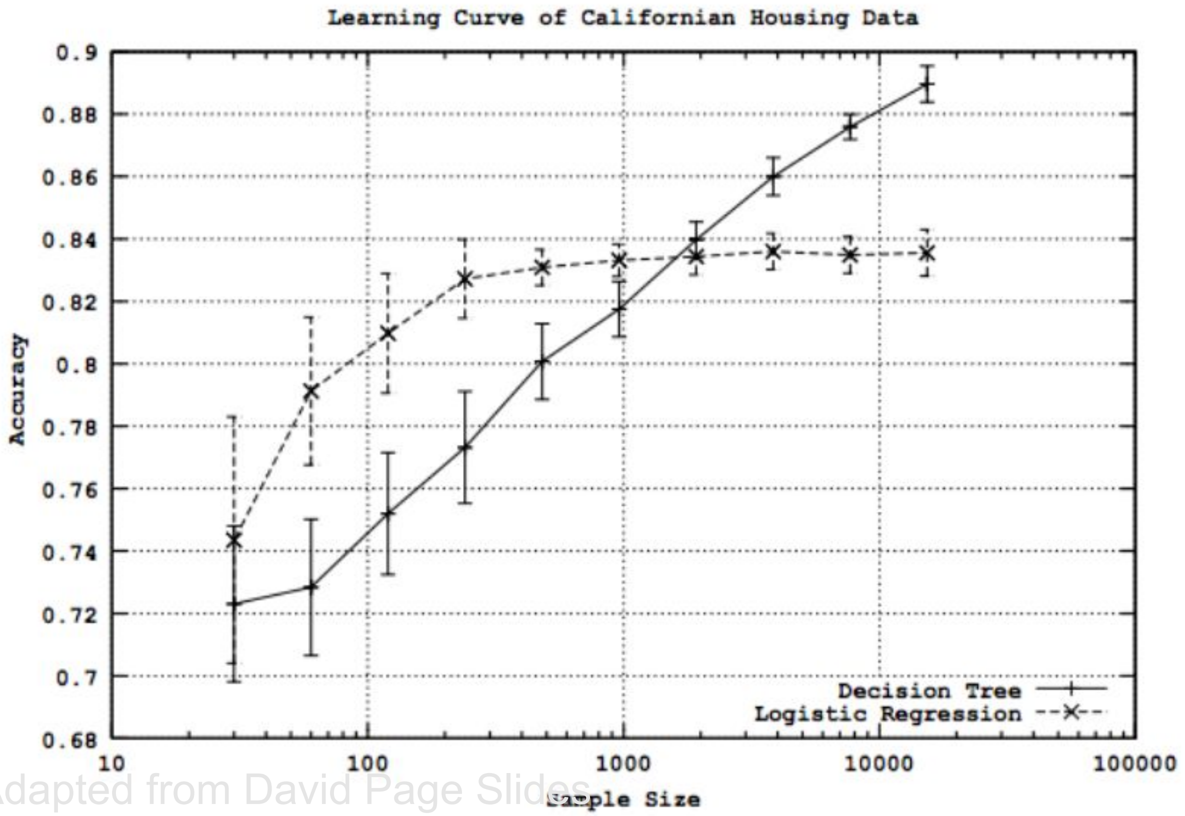
Model evaluation – test sets

How can we get an unbiased estimate of the accuracy of a learned model?



Adapted from David Page Slides

Learning curve



Adapted from David Page Slides

Confusion matrix

- Helps to learn mistakes the model makes

activity recognition from video

actual class

	bend	jack	jump	pjump	run	side	skip	walk	wave1	wave2
bend	100	0	0	0	0	0	0	0	0	0
jack	0	100	0	0	0	0	0	0	0	0
jump	0	0	89	0	0	0	11	0	0	0
pjump	0	0	0	100	0	0	0	0	0	0
run	0	0	0	0	89	0	11	0	0	0
side	0	0	0	0	0	100	0	0	0	0
skip	0	0	0	0	0	0	100	0	0	0
walk	0	0	0	0	0	0	0	100	0	0
wave1	0	0	0	0	0	0	0	0	67	33
wave2	0	0	0	0	0	0	0	0	0	100

predicted class

Confusion matrix for 2-class problems

		actual class	
		positive	negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

$$\text{accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

Is accuracy an adequate measure of predictive performance?

- accuracy may not be useful measure in cases where
  - there is a large class skew
    - Is 98% accuracy good if 97% of the instances are negative?
  - there are differential misclassification costs – say, getting a positive wrong costs more than getting a negative wrong
    - Consider a medical domain in which a false positive results in an extraneous test but a false negative results in a failure to treat a disease
  - we are most interested in a subset of high-confidence predictions

Other accuracy metrics

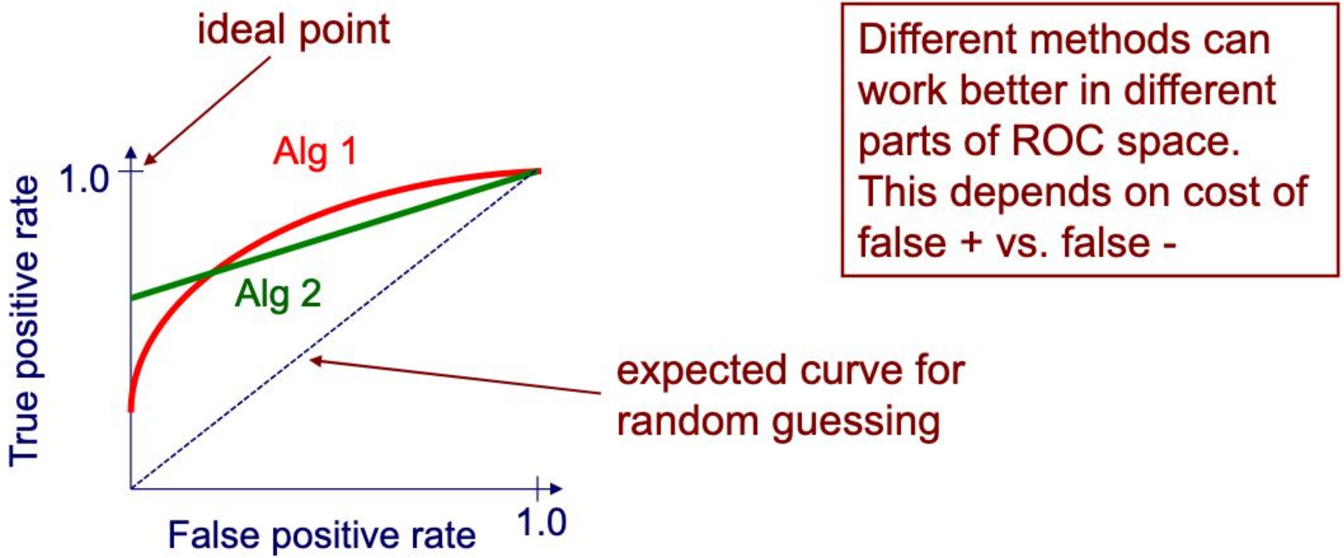
predicted class \ actual class	positive	negative
positive	true positives (TP)	false positives (FP)
negative	false negatives (FN)	true negatives (TN)

$$\text{true positive rate (recall)} = \frac{\text{TP}}{\text{actual pos}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{false positive rate} = \frac{\text{FP}}{\text{actual neg}} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

ROC curves

- A Receiver Operating Characteristic (**ROC**) curve plots the TP-rate vs. the FP-rate as a threshold on the confidence of an instance being positive is varied



Other accuracy metrics

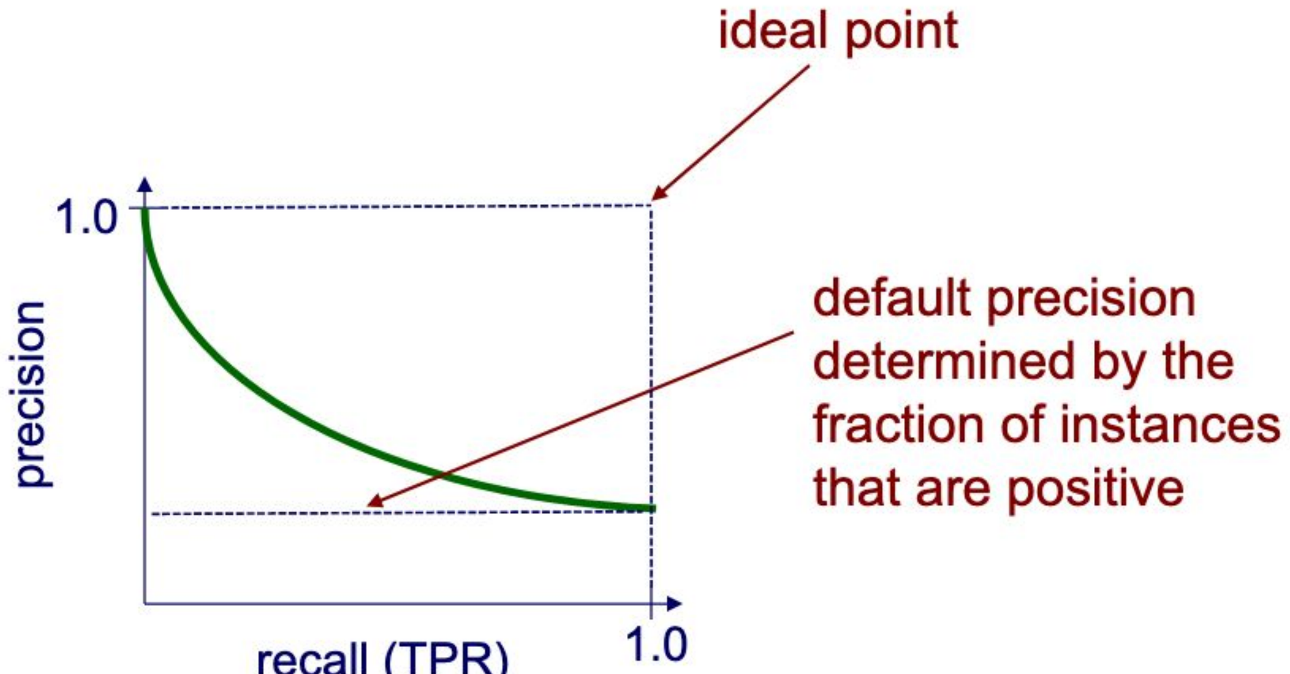
		actual class: positive	actual class: negative
predicted class	positive	true positives (TP)	false positives (FP)
	negative	false negatives (FN)	true negatives (TN)

$$\text{recall (TP rate)} = \frac{\text{TP}}{\text{actual pos}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{precision} = \frac{\text{TP}}{\text{predicted pos}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision/recall curves

- A **precision/recall** curve plots the precision vs. recall (TP-rate) as a threshold on the confidence of an instance being positive is varied



F1-score

$$F1 = \frac{2 * precision * recall}{precision + recall}$$

$$F1 = \frac{2 \times 0.3 \times 0.1}{0.3 + 0.1} \qquad \therefore F1 = 0.15$$

https://pmirla.github.io/assets/harmonic_mean.gif

Overfitting and underfitting

- **Bias:** Assumptions made by a model to make a function easier to learn. It is actually the error rate of the training data. When the error rate has a high value, we call it High Bias and when the error rate has a low value, we call it low Bias.
- **Variance:** The difference between the error rate of training data and testing data is called variance. If the difference is high then it's called high variance and when the difference of errors is low then it's called low variance. Usually, we want to make a low variance for generalized our model.

- **Underfitting**: is a scenario where a model is unable to capture the relationship between the input and output variables accurately, generating a high error rate on both the training set and unseen data.
- **Overfitting:** occurs when a model fits exactly against its training data but does not make accurate predictions on testing data.

Overfitting and underfitting

◉Reasons for Underfitting:

1. High bias and low variance
2. The size of the training dataset used is not enough.
3. The model is too simple.
4. Training data is not cleaned and also contains noise in it.

◉Techniques to reduce underfitting:

1. Increase model complexity
2. Increase the number of features, performing feature engineering
3. Remove noise from the data.

https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

Overfitting and underfitting

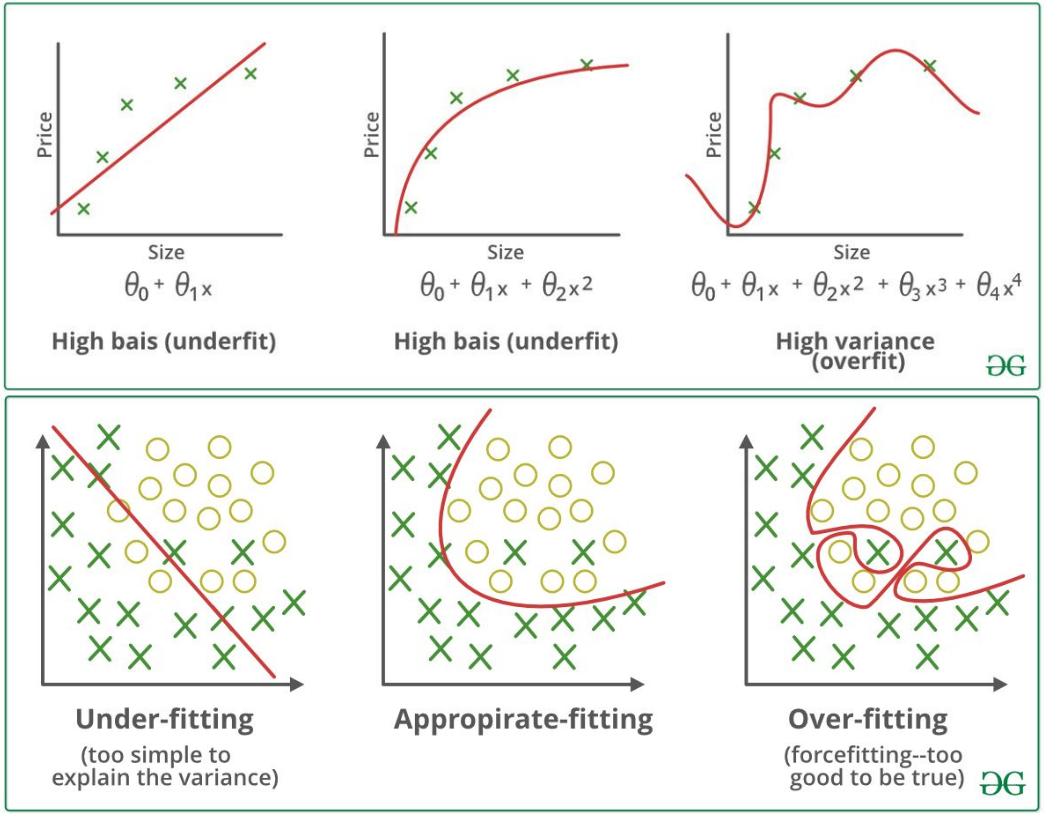
Reasons for Overfitting are as follows:

1. High variance and low bias
2. The model is too complex
3. The size of the training data

Techniques to reduce overfitting:

1. Increase training data.
2. Reduce model complexity.
3. Early stopping during the training phase (have an eye over the loss over the training period as soon as loss begins to increase stop training).

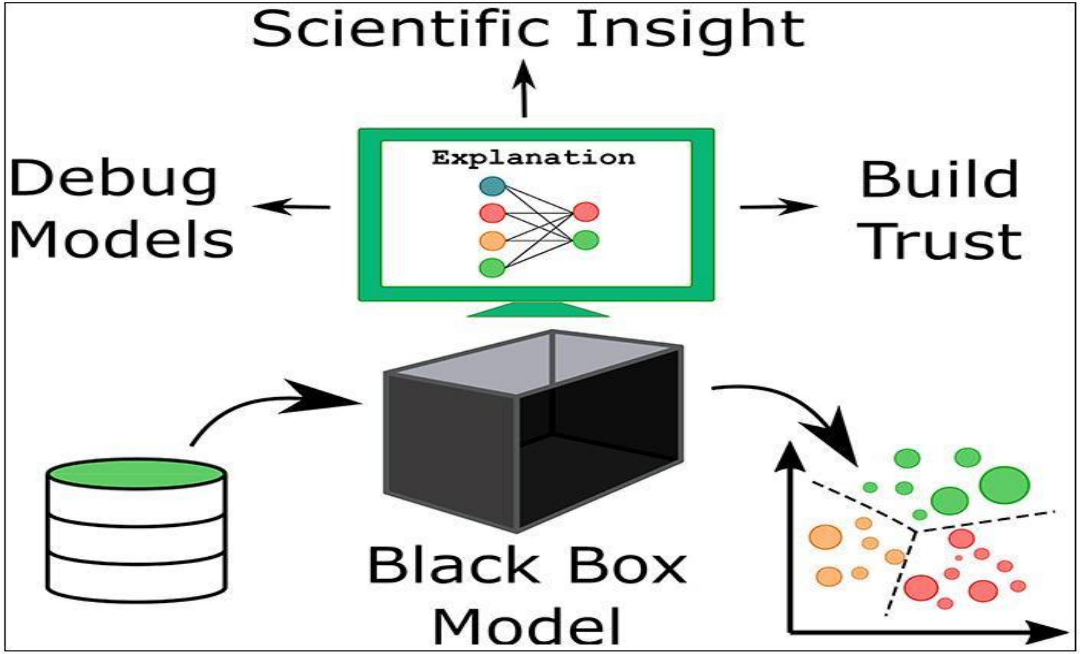
Overfitting and underfitting



https://media.geeksforgeeks.org/wp-content/cdn-uploads/20190523171258/overfitting_2.png

Model Interpretation

Interpretability, Explainability, Bias



https://pubs.acs.org/cms/10.1021/accountsmr.1c00244/asset/images/accountsmr.1c00244.social.jpeg_v03

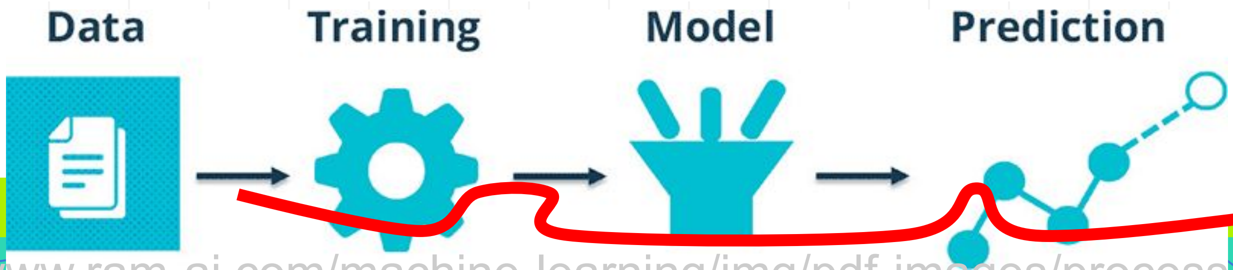
Interpretability needs

- Financial institutions train a model
  - On thousands of outcomes
  - Using dozens of variables
- Models determine
  - Likelihood that you would **default** on a mortgage (with **higher accuracy**)
- If you are a loan officer to stamp approval/denial based on the models decision:
  - How will you be sure it is right?
  - How will you be sure it is wrong?

https://photos2.fotosearch.com/bthumb/CSP/CSP990/loan-approved-rubber-stamp-shows-credit-stock-illustration__k9865316.jpg

Interpretability needs

- AI is at the root of many products and solutions, as intelligent machines are now powered by learning, reasoning, and adaptation capabilities.
- Compliment human excellence, leveraged by machines, AI is helping to predict accurately, **near zero-human innervation**.
- But it is an urgent need to understand how the **machines arrived at those decisions**.
- To interpret decisions made by a machine learning model is
  - to find **meaning in it**
  - **trace it back** to its source and the process that **transformed** it.



https://www.ram-ai.com/machine-learning/img/pdf-images/process-icons.png

What is machine learning interpretation?

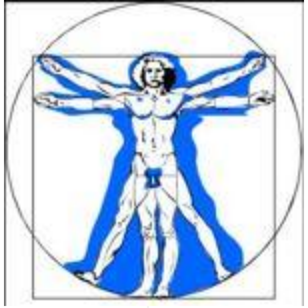
- To **interpret** something is to **explain the meaning of it**.
- That something in ML is an **algorithm**!
- That **algorithm** is a **mathematical** one that takes input data and produces an output, much like with any formula

$$\hat{y} = \beta_0 + \beta_1 x_1$$

$\hat{y}$ □ weighted sum of **x features** with β coefficients

- ŷ: The predicted value for the response variable
- β_0: The mean value of the response variable when x = 0
- β_1: The average change in the response variable for a one unit increase in x
- x: The value for the predictor variable

Example - 25,000 Records of Human Heights (in) and Weights (lbs)

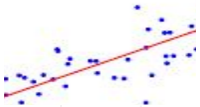


- Human Height and Weight are mostly hereditable, but lifestyles, diet, health and environmental factors also play a role in determining individual's physical characteristics. The dataset contains 25,000 synthetic records of human heights and weights of 18 years old children. These data were **simulated** based on a 1993 by a Growth Survey of 25,000 children from birth to 18 years of age recruited from Maternal and Child Health Centres (MCHC) and schools and were used to develop Hong Kong's current growth charts for **weight**, **height**, **weight-for-age**, **weight-for-height** and **body mass index** (BMI).

http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_Dinov_020108_HeightsWeights

Example...

- For our example, we use **only 200** (from the web pages home page)
- Fit a **linear regression** model
- Use **height** to **predict** the **weight**



Index	Height(Inches)	Weight(Pounds)
1	65.78	112.99
2	71.52	136.49
3	69.40	153.03
4	68.22	142.34
5	67.79	144.30
6	68.70	123.30
7	69.80	141.49
8	70.01	136.46
9	67.90	112.37
10	66.78	120.67
11	66.49	127.45

Packages

```
import math
import requests
from bs4 import BeautifulSoup
import pandas as pd
from sklearn import linear_model
from sklearn.metrics import mean_absolute_error
import matplotlib.pyplot as plt
from scipy.stats import pearsonr
```

Fetching the data from the web page

```
url =  \
'http://wiki.stat.ucla.edu/socr/index.php/SOCR_Data_Dinov_020108_HeightsWeights'
page = requests.get(url)
```

- Extract content

```
soup = BeautifulSoup(page.content, 'html.parser')
tbl = soup.find("table",{"class":"wikitable"})
```

```
<table class="wikitable" style="text-align:center; width:30%" border="1">

<tr>
<th>Index</th><th>Height(Inches)</th><th>Weight(Pounds)
</th></tr>
<tr>
<td>1</td><td>65.78</td><td>112.99
</td></tr>
```

Source view of HTML page

```
height_weight_df = pd.read_html(str(tbl))[0]\
[['Height(Inches)','Weight(Pounds)']]
```

Dataframe content

- Count records

```
num_records = height_weight_df.shape[0]
print(num_records)
```

```
200
```

- Show top 5 of the records

```
height_weight_df.head()
```

	Height(Inches)	Weight(Pounds)
0	65.78	112.99
1	71.52	136.49
2	69.40	153.03
3	68.22	142.34
4	67.79	144.30

Sklearn model

- Prepare the data for **sklearn** data format (feature **matrix** and target **vector**)

```
x = height_weight_df['Height(Inches)'].values.reshape(num_records, 1)
y = height_weight_df['Weight(Pounds)'].values.reshape(num_records, 1)
```

- Initialize the sklearn **LinearRegression** model and **fit** it with the training data

```
model = linear_model.LinearRegression()
_ = model.fit(x,y)
```

- Extract the fitted linear regression model intercept and coefficients

```
print("ŷ = " + str(model.intercept_[0]) + " + " + \
      str(model.coef_.T[0][0]) + " x₁")
```

```
ŷ = -106.02770644878137 + 3.4326761292716297 x₁
```

What does the model tells us?

- On average, for **every additional pound**, there are **3.4 inches** of **height**.
- But the actual outcomes and the predicted outcomes are not the same for the training data.
- The difference between the two outcomes is called the **error**/**residuals**.
- Use the **mean_absolute_error** to measure the deviation between the predicted values and the actual values

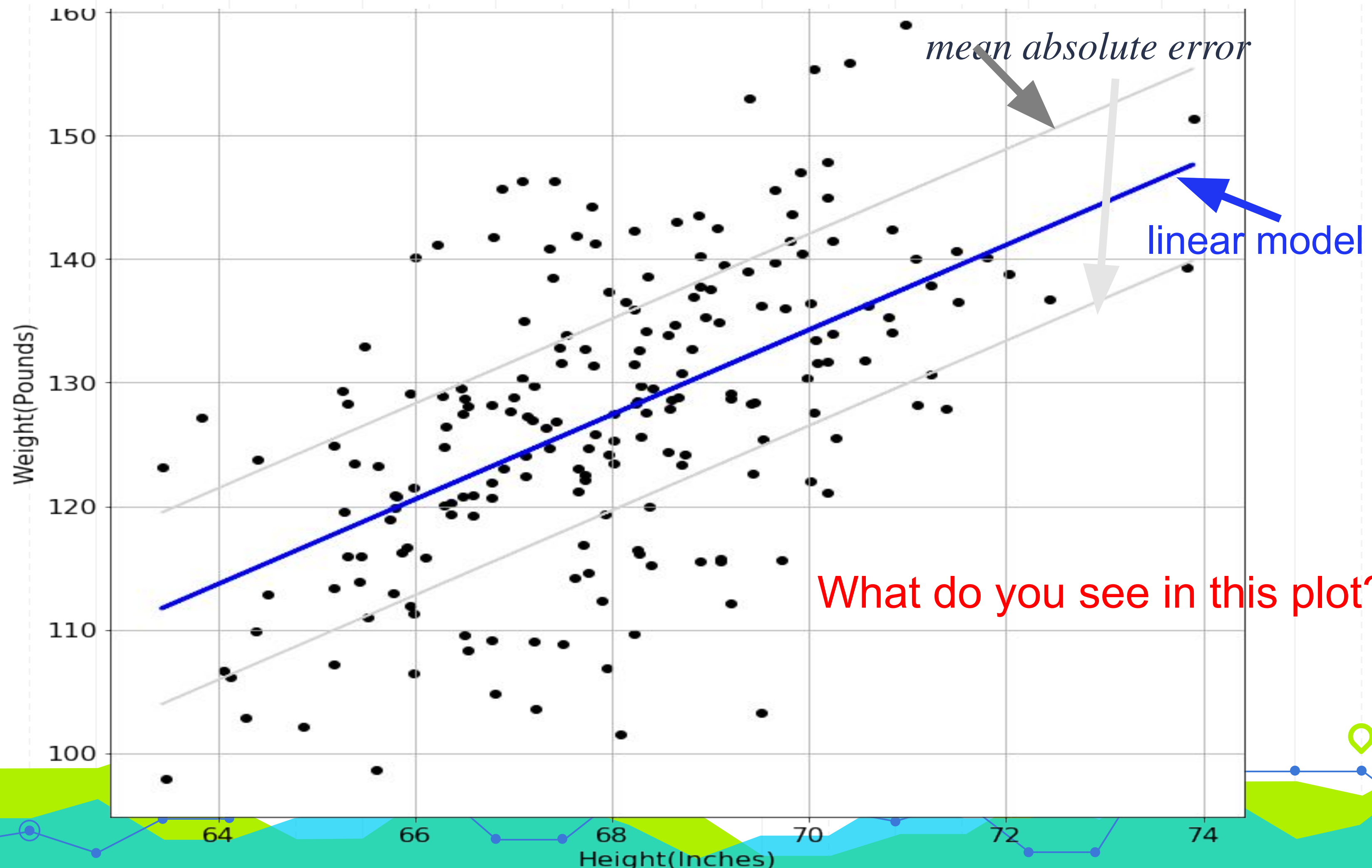
```
y_pred = model.predict(x)
mae = mean_absolute_error(y, y_pred)
print(mae)
```

```
7.7587373803882205
```

What does MAE tells us?

- A 7.8 **mean absolute** error means that, **on average**, the prediction is deviated 7.8 pounds from the **actual amount**.
- **Visualizing** the linear regression model can **shed some light** on how accurate these predictions truly are.

```
plt.figure(figsize=(12,12))
plt.rcParams.update({'font.size': 16})
plt.scatter(x, y,  color='black')
plt.plot(x, y_pred, color='blue', linewidth=3)
plt.plot(x, y_pred + mae, color='lightgray')
plt.plot(x, y_pred - mae, color='lightgray')
plt.title('')
plt.xlabel('Height(Inches)')
plt.ylabel('Weight(Pounds)')
plt.grid(True)
plt.show()
```



What do you see in this plot?

Exploring the plot

- Many weights are 20– 25 pounds away from the predication
- Hence, the MAE can easily fool us if we did not inspect the plot
  - Visualizing the error of the model is important to understand its distribution
- Residuals more or less equally spread out, we say it's **homoscedastic** (same variance).
- Assumptions to test for linear models includes, in addition to homoscedasticity
  - **Linearity**
  - **Normality** (normally distributed),
  - **Independence** (no relation between the different examples),
  - **Multicollinearity** (for two and more features)
- Establish a linear relationship between x **height** and y **weight**. This association is called a **linear correlation**.

Pearson’s correlation coefficient

- **Pearson’s correlation coefficient** is a statistical method that measures the association between two variables using their **covariance** divided by their **standard deviations**.
  - It is between -1 and 1
  - **0** -> weaker association, **+ve** number -> positive association, **-ve** -> negative association

```
corr, pval = pearsonr(x[:,0], y[:,0])
print(corr)
```

```
0.5568647346122995
```

- □ As height increases, weight also tends to increase, closer to 1 than 0, hence **strongly correlated**

Pearson's correlation coefficient

- We can also test the **p-value** (the probability of obtaining test results at least as extreme as the result actually observed [1])
- If we test that it's less than an error level of **5% (0.05)**, we can say there's sufficient **evidence of this correlation**.

```
print(pval < 0.05)
```

```
True
```

```
print(pval)
```

```
1.102901515126636e-17
```

[1] https://en.wikipedia.org/wiki/P-value

Explainability of the model

- Do you accept if this model predicted **134** pounds for **71** inches tall?

```
height_weight_df.describe()
```

	Height(Inches)	Weight(Pounds)
count	200.000000	200.000000
mean	67.949800	127.221950
std	1.940363	11.960959
min	63.430000	97.900000
25%	66.522500	119.895000
50%	67.935000	127.875000
75%	69.202500	136.097500
max	73.900000	158.960000

Explainability of the model

- Do you accept if this model predicted **134** pounds for **71** inches tall?
  - **Yes**, it is expected
- What if the model predicts **18 pounds more**?

```
height_weight_df.describe()
```

	Height(Inches)	Weight(Pounds)
count	200.000000	200.000000
mean	67.949800	127.221950
std	1.940363	11.960959
min	63.430000	97.900000
25%	66.522500	119.895000
50%	67.935000	127.875000
75%	69.202500	136.097500
max	73.900000	158.960000

Explainability of the model

- Do you accept if this model predicted **134** pounds for **71** inches tall?
  - **Yes**, it is expected
- What if the model predicts **18 pounds more**?
  - **Yes**, the margin is not ”so” **unusual**, even though it is **not ideal**
- What do we expect for **56 inches tall**? Reliable?

```
height_weight_df.describe()
```

	Height(Inches)	Weight(Pounds)
count	200.000000	200.000000
mean	67.949800	127.221950
std	1.940363	11.960959
min	63.430000	97.900000
25%	66.522500	119.895000
50%	67.935000	127.875000
75%	69.202500	136.097500
max	73.900000	158.960000

Explainability of the model

- Do you accept if this model predicted **134** pounds for **71** inches tall?
  - **Yes**, it is expected
- What if the model predicts **18 pounds more**?
  - **Yes**, the margin is not ”so” **unusual**, even though it is **not ideal**
- What do we expect for 56 inches tall? Reliable?
  - **No**, the model is fitted on the data of subjects no shorter than **63 inches**
- What about we measure for **9-year-old**?

```
height_weight_df.describe()
```

	Height(Inches)	Weight(Pounds)
count	200.000000	200.000000
mean	67.949800	127.221950
std	1.940363	11.960959
min	63.430000	97.900000
25%	66.522500	119.895000
50%	67.935000	127.875000
75%	69.202500	136.097500
max	73.900000	158.960000

Explainability of the model

- Do you accept if this model predicted **134** pounds for **71** inches tall?
  - **Yes**, it is expected
- What if the model predicts **18 pounds more**?
  - **Yes**, the margin is not "so" **unusual**, even though it is **not ideal**
- What do we expect for 56 inches tall? Reliable?
  - **No**, the model is fitted on the data of subjects no shorter than **63 inches**
- What about we measure for **9-year-old**?
  - **No**, the data is for **18-year-olds**

```
height_weight_df.describe()
```

	Height(Inches)	Weight(Pounds)
count	200.000000	200.000000
mean	67.949800	127.221950
std	1.940363	11.960959
min	63.430000	97.900000
25%	66.522500	119.895000
50%	67.935000	127.875000
75%	69.202500	136.097500
max	73.900000	158.960000

Explainablity of the model

- Is this model realistic?
  - **No**. we need to add more features such as gender, age, diet, activity level,…
- Explore **if it is fair** to include or not to include features
- **Selection bias**, what if the dataset is dominated by one group, for example **male** for gender?
- **Omitted variables bias**, what of important features, such as **poverty**, **pregnancy**, lifestyle choices are missed?
- **Feature importance**, which features impact model performance?
- More feature, complex model.

Model interpretation questions?

1. Can we explain that predictions were made **fairly**?

2. Can we trace the predictions reliably **back to something** or someone?

3. Can we explain **how predictions were made**? Can we explain how the **model works**?

And ultimately, the question to answer is :

Can we trust the model?

The FAT concept

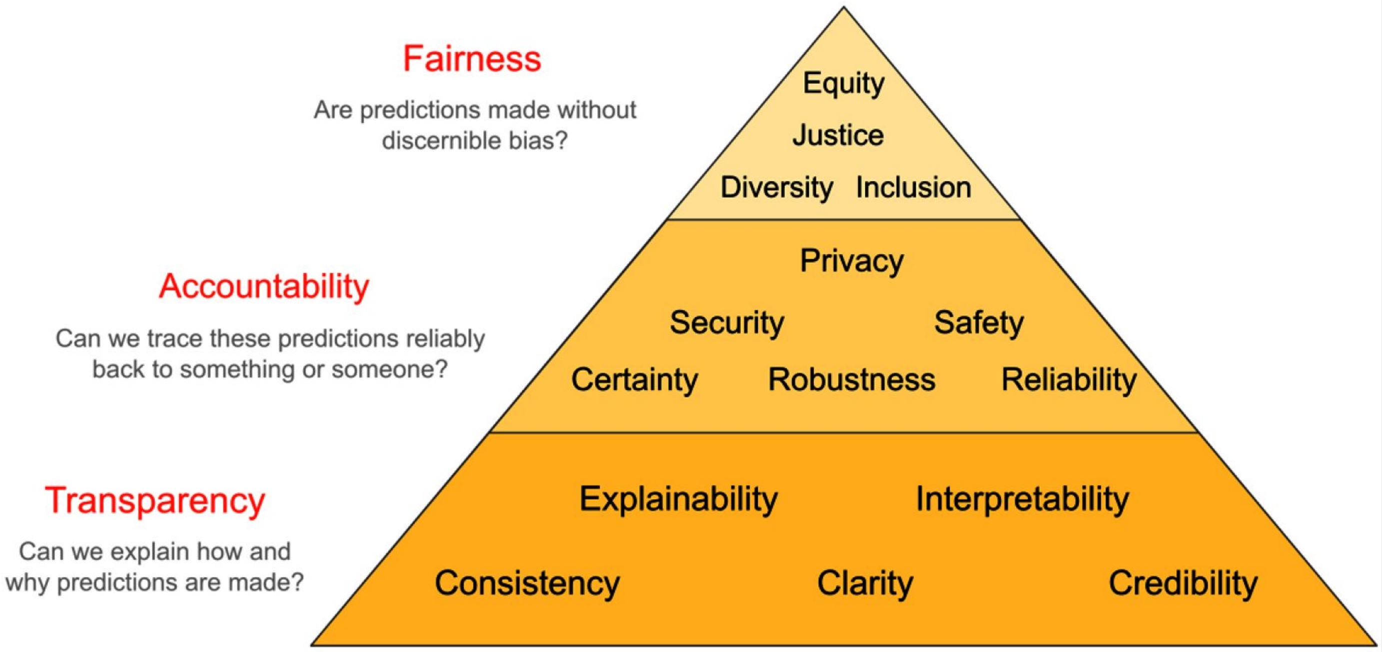


Figure 1.2 – Three main concept of Interpretable Machine Learning

Interpretability and explainability

- Interpretability and explainability are not **synonyms**
- Interpretability is the extent to which **humans**, including non-subject-matter experts, can **understand** the **cause and effect**, and **input and output**, of a machine learning model.
- Easily answer
  - *why does an input to a model produce a specific output?*
  - *What are the requirements and constraints of the input data?*
  - *What are the confidence bounds of the predictions?*
  - *why does one variable have a more substantial effect than another?*

Interpretability

- Complexity of model
  - A lot can make the model complex and difficult to interpret, such as **math involved** in the model, **dataset selection**, **feature selection**, **model training**, **parameter tuning**
- Opaque models interpretability: models which are complex
  - Post-hoc-interpretability: if the predictions are still trustworthy
  - Like we can't explain how a human brain makes a choice, but we often trust its decision

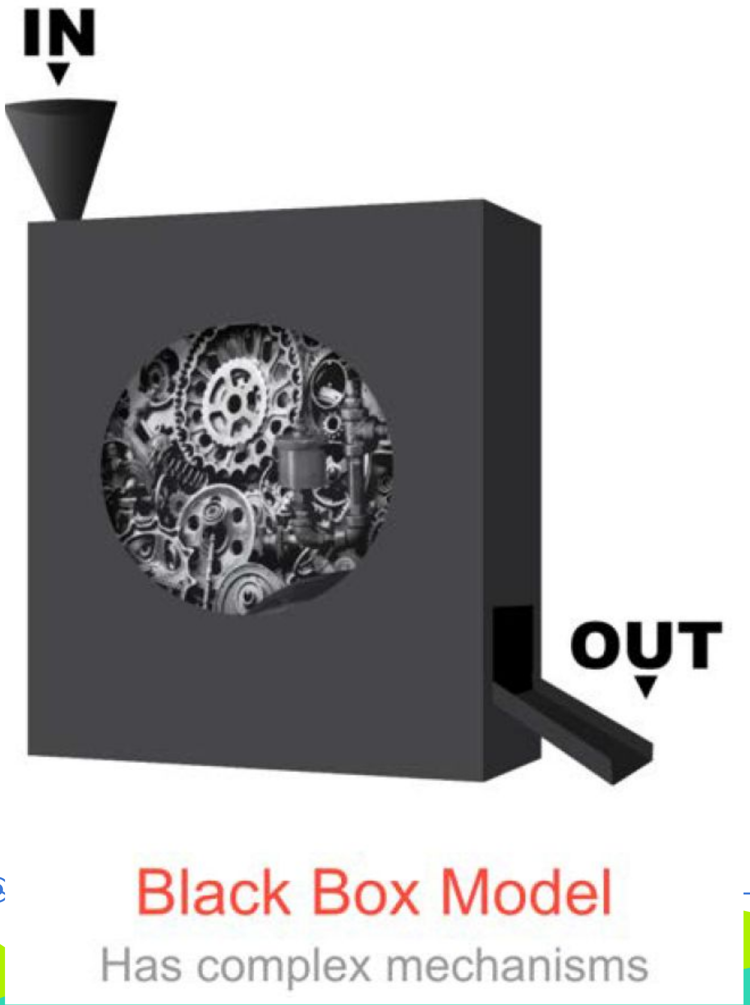
https://i0.wp.com/blog.frontiersin.org/wp-content/uploads/2016/06/shutterstock_374233666.jpg?fit=940%2C940&ssl=1

Interpretability

- When does interpretability **not that much required**?
  - When incorrect results have **no significant consequences**. Example, find and read postal code in a package. Cost of misclassification is low
  - When there are **consequences**, but these have been **studied sufficiently** and **validated enough** in the real world to make decisions without human involvement. Example, traffic-alert and collision-avoidance system (TCAS)
- Interpretability is needed for systems to have the following attributes:
  - **Minable for scientific knowledge:** example climate model
  - **Reliable and safe:** example self driving
  - **Ethical:** example gender-biased translation
  - **Conclusive and consistent**

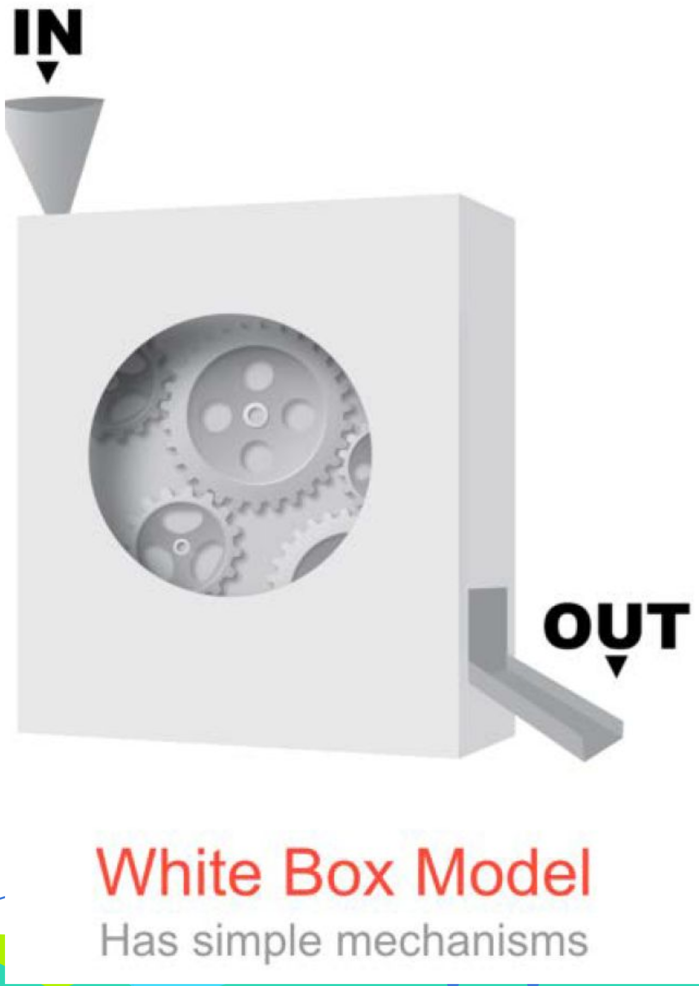
Black-box models

- **Black-box/opaque models** – only the **input and outputs** are observable but can not see the input transformation process.
- The **mechanisms** are not easily understood



White-box models

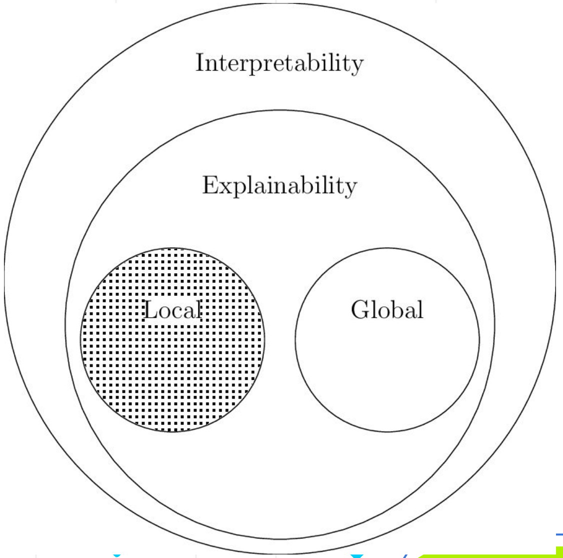
- White-box/transparent models achieve a **total or near-total** interpretation transparency
- They are intrinsically interpretable



Explainability

- Explainability encompasses everything interpretability is.
  - Goes deeper on the **transparency requirement** than interpretability
  - Demands **human friendly explanations** for a model's inner workings and the model training process, not just model inference
  - Model, design, and algorithmic transparency

https://www.researchgate.net/publication/346680834/figure/fig1/AS:966175886409733@1607365690658/Interpretability-and-explainability-algorithms-The-present-work-is-focused-on-local.png

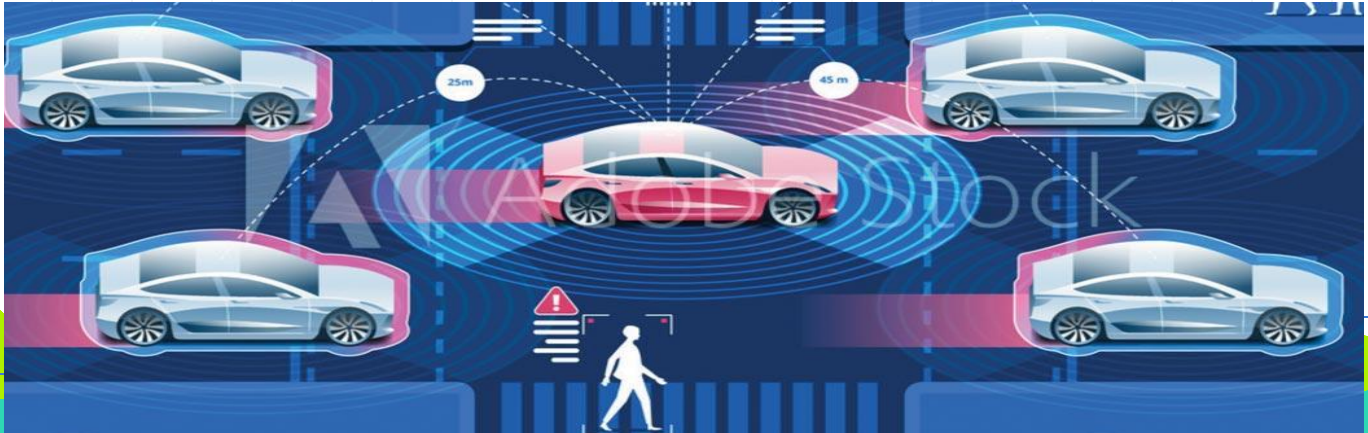


Explainability

- **Model transparency**: Being able to explain how a model is trained step by step.
  - In the prev. example, how the optimization method called **ordinary least squares** finds the β **coefficient** that minimizes errors in the model.
- **Design transparency**: Being able to explain **choices made**, such as model **architecture** and **hyperparameters**. For instance, choices based on the size or nature of the training data .
- **Algorithmic transparency**: Being able to explain automated optimizations such as grid search for hyperparameters

Transparency requirements

- **Scientific research**: for reproducibility
- **Clinical trials**: reproducible and statistically grounded
- **Consumer product safety testing**: when life-and-death safety is a concern
- **Public policy and law**: algorithmic governance, **one day**, government could be entirely run by algorithms
- **Criminal investigation and regulatory compliance audits**: danger due to algorithms, such as at chemical factory or autonomous vehicle, decision trial is needed



A business case for interpretability

- Better decisions: models are trained and evaluated against a desired evaluation metrics. Models are deployed once they pass **held-out/test** datasets, but they can fail **once deployed in real time application**, for example:
  - **Trading algorithm** crash stock market
  - **Smart home devices** terrifying their users
  - **License recognition** system fine the wrong driver
  - **Racially biased surveillance** system, wrong shoot
  - **A self-driving car** could mistake snow for a pavement

Why?

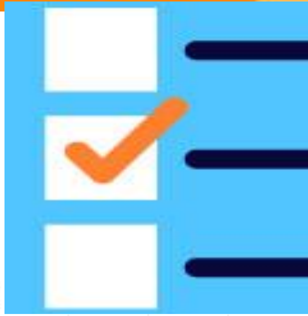
A business case for interpretability

- Focusing on just optimizing metrics can be a **recipe for disaster**
- In the lab the model might perform well, but you have to ask **why**?
  - You might miss an **opportunity to improve it otherwise**
- Example
  - What the self-driving car thinks a **road is not enough**, why so?
    - If the reason is that the road is light-colored, **this is dangerous**
    - If you know why, you could add road images from **winter**
- Making the model **more interpretable** is not to **make it less complex**, it is to make it learn different aspects of the environment.

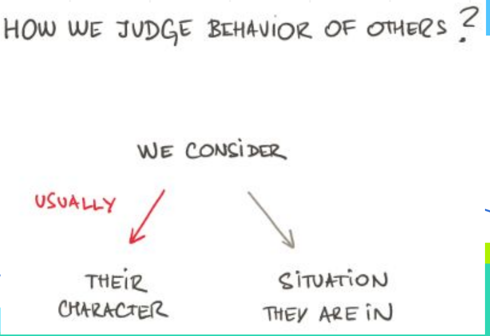
Decision biases

- **Conservatism bias**: new information evolve but our prior belief won't change

- **Salience bias**: some features might be prominent, we need to consider others too

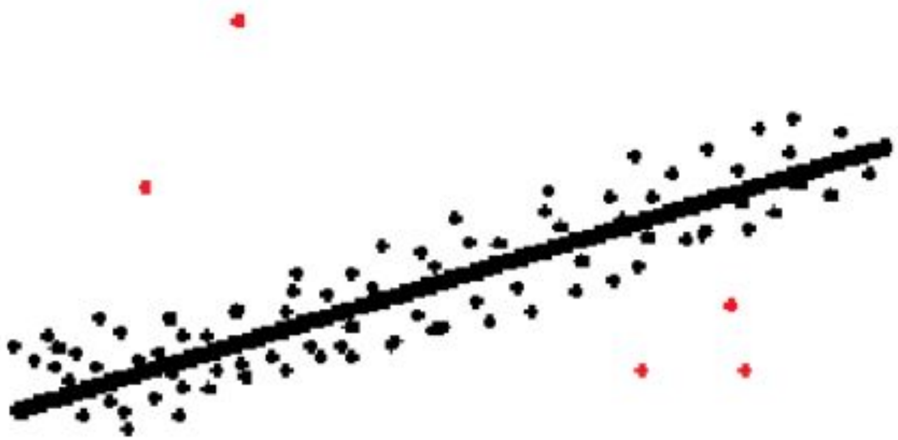


- **Fundamental attribution error**:
  - attribute outcomes to **behavior** rather than **circumstances**, **character** rather than **situations**, **nature** rather than **nurture**.



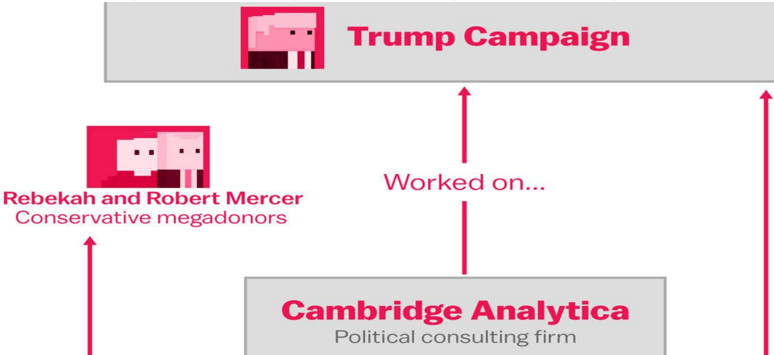
Outliers

- One crucial **benefit** of model interpretation is **locating outliers**. These outliers could be a potential new source of **revenue** or a **liability** waiting to happen. Knowing this can help us to prepare and strategize accordingly.



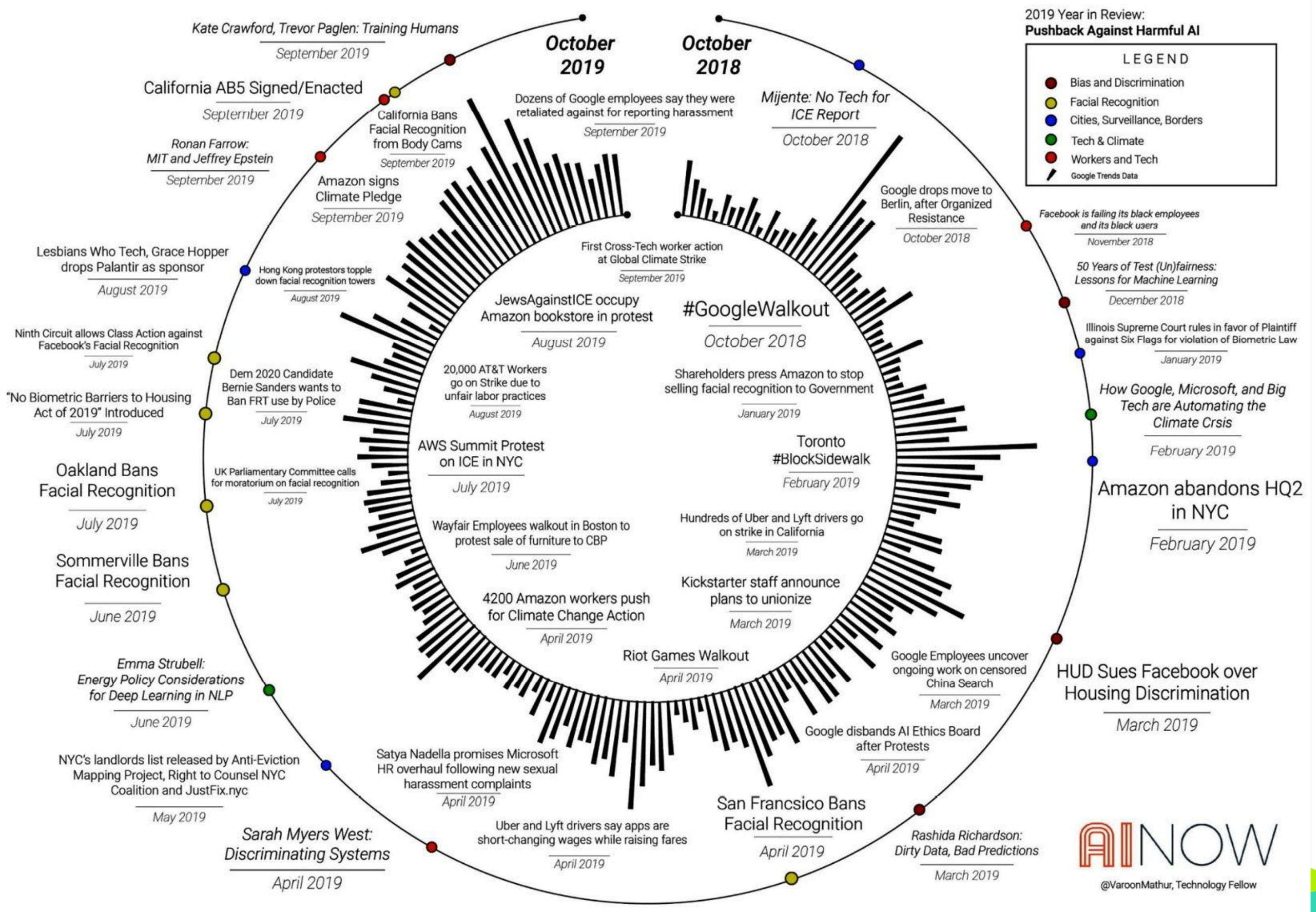
More trusted brands

- Trust is defined as a belief in the reliability, ability, or credibility of something or someone
- **Organization** – trust is their reputation
- **Court** – all it takes is one accident, controversy, or fiasco to lose trust
- Example: **Boeing** after the **737 MAX debacle** or **Facebook** after the **2016 presidential election scandal**
  - Short-sighted decisions optimized a single metric, forecasted plane sales or digital ad sales!
  - Organizations resort to **fallacies** to **justify reasoning**, **confuse public**, distract media narratives
  - Lose credibility (what they do, what they say)



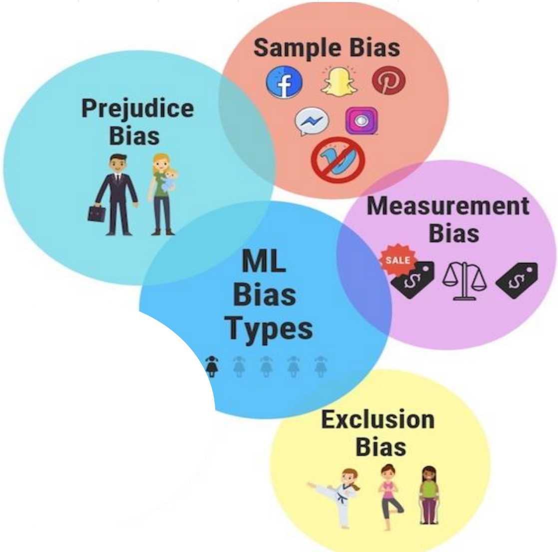
XAI - Trust

- Due to trust issues, many **AI-driven technologies** are **losing public support**, to the detriment of both companies that **monetize AI** and users that could benefit from them.


2019 Year in Review:
Pushback Against Harmful AI
LEGEND
Bias and Discrimination
Facial Recognition
Cities, Surveillance, Borders
Tech & Climate
Workers and Tech
Google Trends Data
October 2019
October 2018
Mijente: No Tech for ICE Report
October 2018
Google drops move to Berlin, after Organized Resistance
October 2018
Facebook is failing its black employees and its black users
November 2018
50 Years of Test (Un)fairness: Lessons for Machine Learning
December 2018
Illinois Supreme Court rules in favor of Plaintiff against Six Flags for violation of Biometric Law
January 2019
How Google, Microsoft, and Big Tech are Automating the Climate Crsis
February 2019
Amazon abandons HQ2 in NYC
February 2019
HUD Sues Facebook over Housing Discrimination
March 2019
Rashida Richardson: Dirty Data, Bad Predictions
March 2019
San Francsico Bans Facial Recognition
April 2019
Uber and Lyft drivers say apps are short-changing wages while raising fares
April 2019
Sarah Myers West: Discriminating Systems
April 2019
Satya Nadella promises Microsoft HR overhaul following new sexual harassment complaints
April 2019
NYC's landlords list released by Anti-Eviction Mapping Project, Right to Counsel NYC Coalition and JustFix.nyc
May 2019
Emma Strubell: Energy Policy Considerations for Deep Learning in NLP
June 2019
Sommerville Bans Facial Recognition
June 2019
Oakland Bans Facial Recognition
July 2019
"No Biometric Barriers to Housing Act of 2019" Introduced
July 2019
Ninth Circuit allows Class Action against Facebook's Facial Recognition
July 2019
Lesbians Who Tech, Grace Hopper drops Palantir as sponsor
August 2019
Ronan Farrow: MIT and Jeffrey Epstein
September 2019
California AB5 Signed/Enacted
September 2019
Kate Crawford, Trevor Paglen: Training Humans
September 2019
California Bans Facial Recognition from Body Cams
September 2019
Amazon signs Climate Pledge
September 2019
Hong Kong protestors topple down facial recognition towers
August 2019
Dem 2020 Candidate Bernie Sanders wants to Ban FRT use by Police
July 2019
UK Parliamentary Committee calls for moratorium on facial recognition
July 2019
Dozens of Google employees say they were retaliated against for reporting harassment
September 2019
First Cross-Tech worker action at Global Climate Strike
September 2019
JewsAgainstICE occupy Amazon bookstore in protest
August 2019
#GoogleWalkout
October 2018
20,000 AT&T Workers go on Strike due to unfair labor practices
August 2019
Shareholders press Amazon to stop selling facial recognition to Government
January 2019
AWS Summit Protest on ICE in NYC
July 2019
Toronto #BlockSidewalk
February 2019
Wayfair Employees walkout in Boston to protest sale of furniture to CBP
June 2019
Hundreds of Uber and Lyft drivers go on strike in California
March 2019
4200 Amazon workers push for Climate Change Action
April 2019
Kickstarter staff announce plans to unionize
March 2019
Riot Games Walkout
April 2019
Google Employees uncover ongoing work on censored China Search
March 2019
Google disbands AI Ethics Board after Protests
April 2019
AI NOW
@VaroonMathur, Technology Fellow

Ethical Issues

- A **Machine learning model's** programming has no **programmer** because the "**programming**" was **learned from data**, and there are things a **model can learn from data that can result in ethical transgressions**. Top among them are biases such as the following:
  - **Sample bias**: When your data, the sample, doesn't represent the environment accurately, also known as the population
  - **Exclusion bias**: When you omit features or groups that could otherwise explain a critical phenomenon with the data
  - **Prejudice bias**: When stereotypes influence your data, eithe directly or indirectly
  - **Measurement bias**: When faulty measurements distort your data



Take home messages

- A ML model learns from data – **nothing more**
- The more you work on your **data quality**, the more your model is interpretable
- Focus on **deployment test**, that is where the model will be **realistically evaluated**
- If you can explain your model, you know how to **fix the drawbacks easily**
- You have to to take predictions from models deployed by others with **a grain of salt**, make sure the model is explainable, reproducible!

Thank you!