TEALS | AI Summer Academy

Azure Global

Microsoft

Introduction

- **Azure Global Commercial Industry – RoW**
- **What we do** – Building industry PaaS AI solutions for the following:
  - Retail & Consumer Goods
  - Supply Chain
  - Energy
  - Financial Services
  - Media, Entertainment & Gaming

Python crash course

Emmanuel Awa
Said Bleik
Max Kaznady

Why Python

- **Learning Curve**
- **Awesome for POC and Scripting**
- **Platform Support and Flexibility**
- **A library/framework for everything**
- **Testing frameworks**
- **Great Choice for Enterprise Application Integration**
- **Active Technical Community**

Why Learn Python - Top 10 Reasons to Learn Python in 2021 | upGrad blog

Python Syntax

- Type into command line or file
- Indentation
- Variables
- Comments
- Casting
- Multiple assignments, variable unpacking
- Global vs Local Variables

Python Data Structures

- Numeric
  - Int, float, complex
- Sequence
  - List, tuple, range
- Mapping
  - Dict
- Sets
  - Set, frozenset
- Boolean
  - Bool
- Binary
  - Bytes, bytearray, memoryview
- Text
  - str

Python Operators

- Arithmetic
- Assignment
- Comparison
- Logical
- Identity
- Membership
- Bitwise

Python Operators (w3schools.com)

Conditionals and Loops

- Logical conditionals
  - If, if…elif..else
- Loops
  - While and For
  - Break statements are used to terminate loops

Functions and Lambdas Introduction

- Function is a named block of code that can only run when called.
  - It can take parameters by name or arbitrarily using *args
  - Can also take keyword arguments using **kwargs

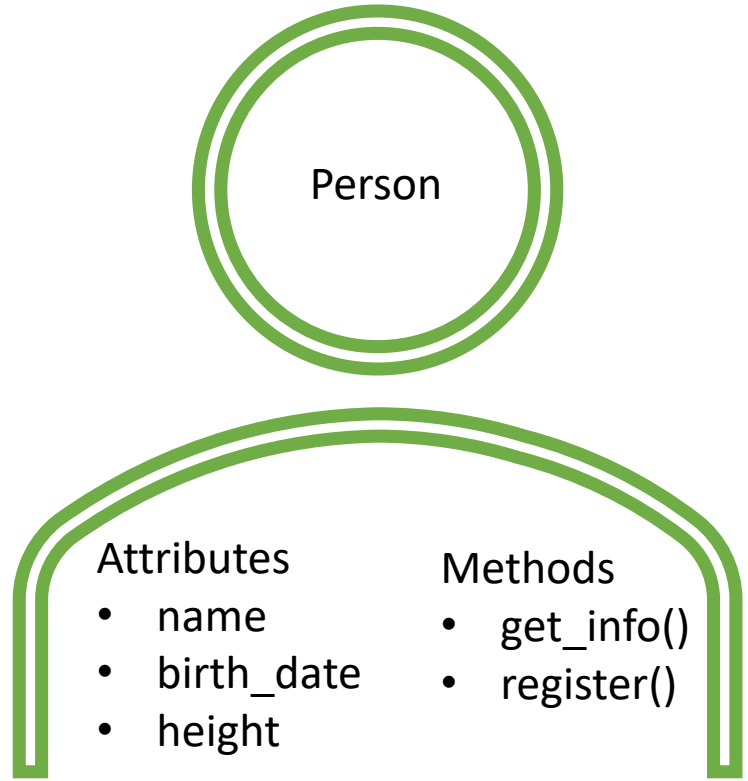
- Lambda is a nameless function.
  - Can take many arguments but can only have one expression
  - Useful for quick reusable one-liners that are required for a short time
  - Faster than functions as it’s evaluated at runtime

Go to notebook

Object Oriented Programming (OOP)

Beyond variables and values

- Programming paradigm where entities are represented as *objects*
- Abstract representation of properties and behavior of entities using *classes* (types of objects)
  - Properties <- member attributes
  - Behavior <- member methods
- Helps programmers:
  - Reuse code
  - Design better real-world systems
  - Build on top of others' work



Class *instances* (objects of type Person):
person1, person2, p3, that_person...

Object Oriented Programming (OOP)

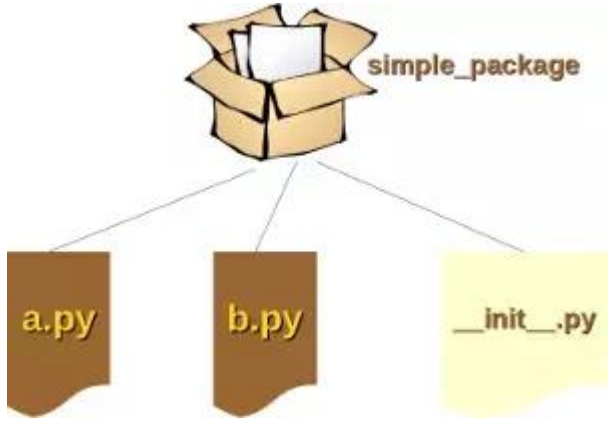
- Inheritance
  - Parent/Child or Base/Derived classes
  - Ex: *class Student(Person)* and *class Teacher(Person)*
  - Inherit attributes and methods
- Encapsulation
  - Values and functions are encapsulated within a class
  - Ex: student1.get_info() returns previously set member attributes
- Polymorphism
  - Allows functions to have different forms
  - Ex: student1.get_courses() and teacher5.get_courses()

Go to notebook

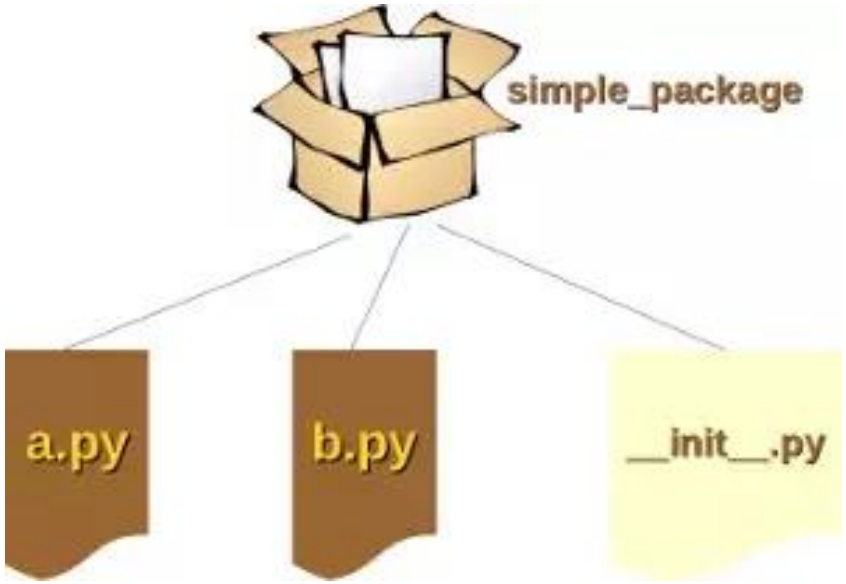
Packages

- Package is like a Toolbox
  - imports are like tools

- Toolbox example 1
  - “from toolbox import hammer
  - “hammer.strike()”
- Toolbox example 2
  - “from toolbox.hammer import strike”
  - “strike()”

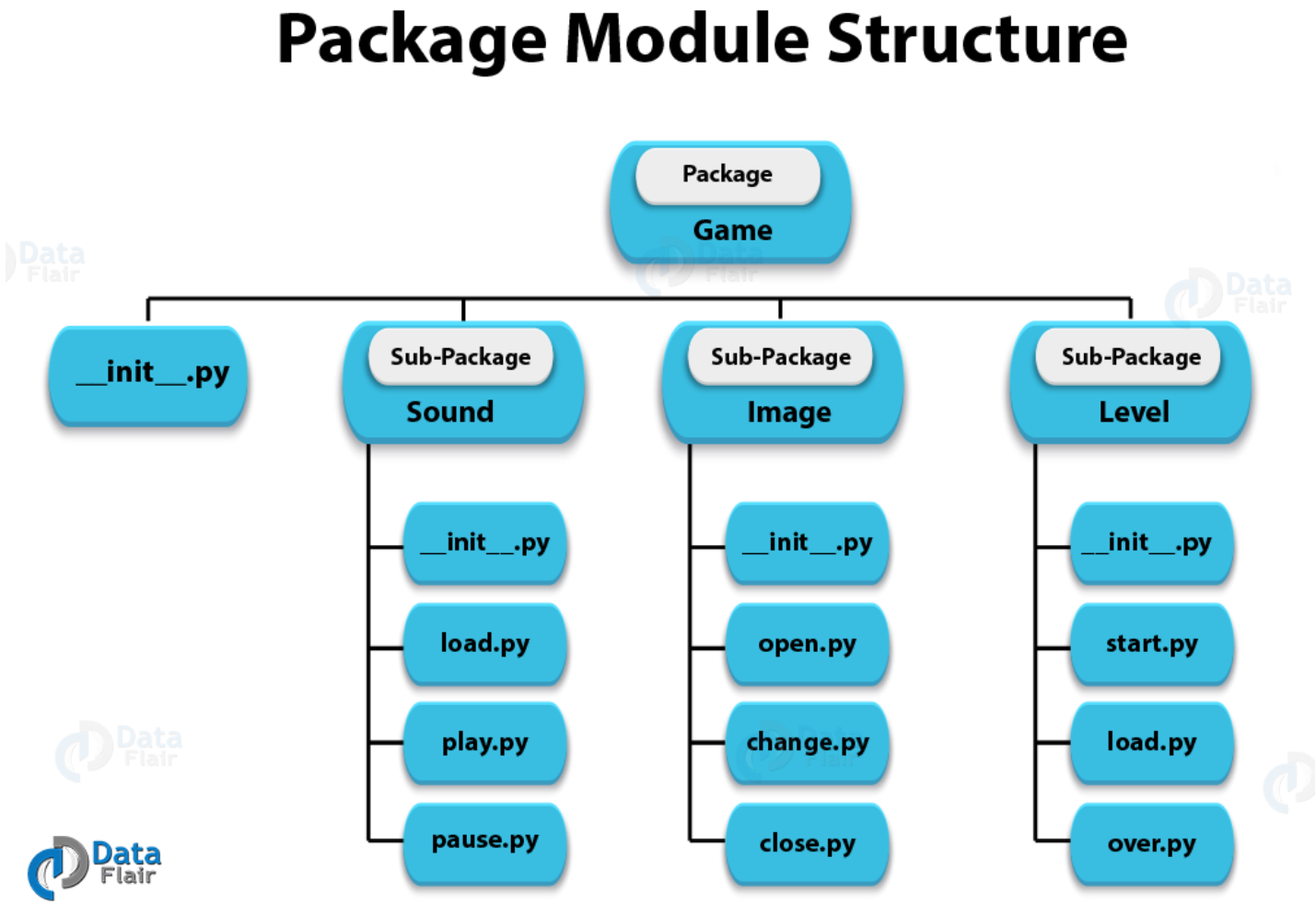


Package concepts



- Package module import
  - “from simple_package import a”
  - “a.do_stuff(‘some argument’)”
- Package function import
  - “from simple_package.a import do_stuff”
  - “do_stuff(‘some_argument’)”
- Package constants
  - “import simple_package; simple_package.CONSTANT”
  - “from simple_package import CONSTANT”

Growing packages



Coding example in an IDE

References

- Free Intro to Python Course | Udacity | Udacity Free Courses
- Why Learn Python - Top 10 Reasons to Learn Python in 2021 | upGrad blog
- Python Tutorial (w3schools.com)
- https://www.python-course.eu/python3_packages.php
- https://projectmanagementshop.eu/products/change-managers-toolbox/
- https://data-flair.training/blogs/python-packages/

AI Fundamentals

Alejandro Buendia
Jaya Mathew
Sahitya Mantravadi

Overview

Topics for the day

- Introduction to AI
- Probability and Statistics
- Linear Regression
- Binary Classification
- Q&A

Speakers

- Alejandro Buendia – Introduction to AI, Probability and Statistics
- Jaya Mathew – Linear Regression
- Sahitya Mantravadi – Binary Classification

Introduction to AI

* Some graphics courtesy of Nakul Verma (COMS 4771, Columbia University)

What is AI and machine learning?

- Artificial intelligence (AI): Broadly refers to machines that can do tasks “intelligently.”
- Machine learning: Study of making machines learn a concept **from data** without having to explicitly program it.
- Machine learning can be thought of as a subset of AI.
- AI/machine learning involves creating algorithms that can:
  - Learn from a set of input data to make predictions
  - Find interesting patterns in data

Why AI/machine learning?

- Often we can't write down a simple set of rules to solve complicated real-world problems.
- For example, how would we write python code to distinguish between these two faces?

- Other times, we don't know what explicit task we want to solve given some data as a starting point.

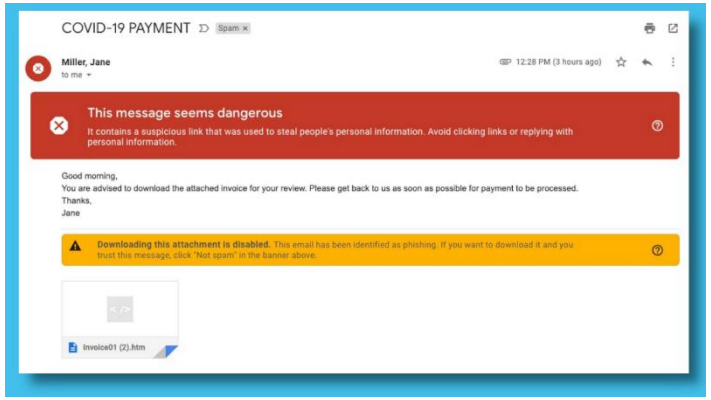
Machine Learning: The basics

Let’s look at three prediction problems:

- Handwritten character recognition

→ {0, 1, 2, …, 9}

- Spam filtering



→ {spam, not spam}

- Object recognition

→ {building, tree, car, road, sky, …}

Machine Learning: The basics (cont'd)

What do these problems have in common? We have…

Input:  $= \vec{x} = \begin{pmatrix} x^{(1)} \\ \vdots \\ x^{(d)} \end{pmatrix} \in \mathcal{X}$ 

An algorithm: f

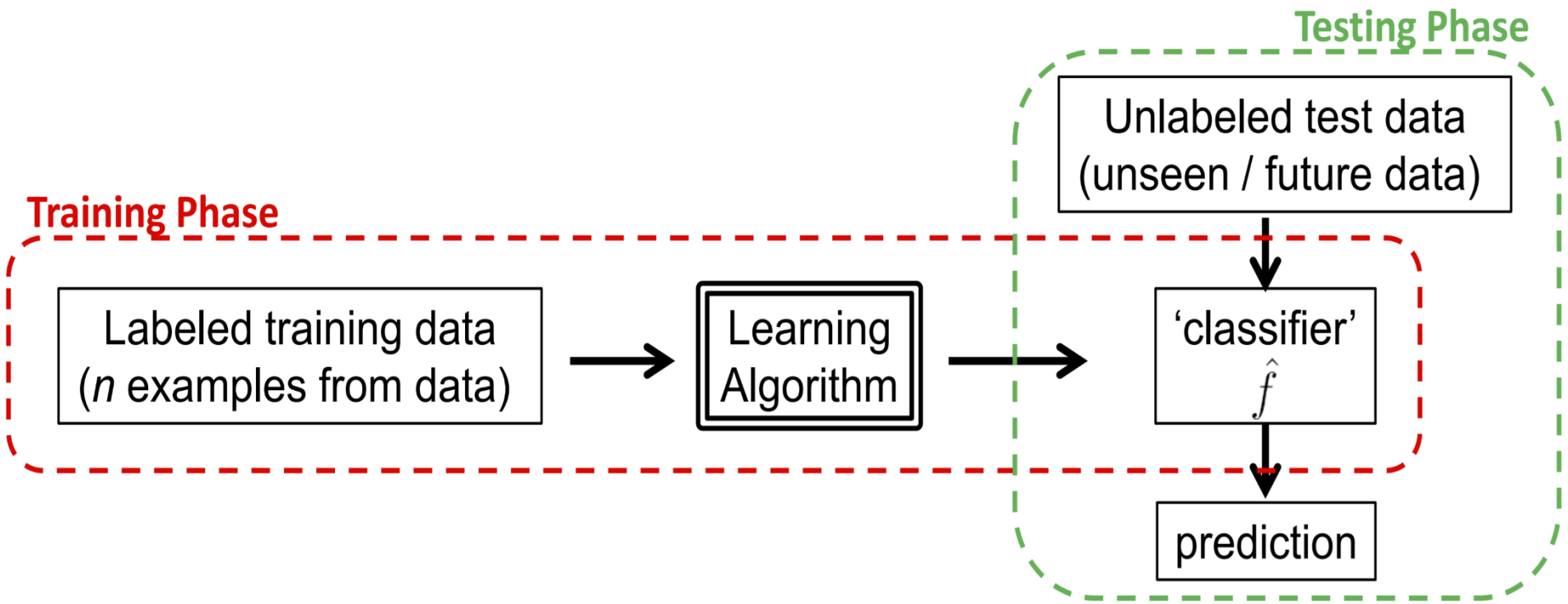
Output: 5 $= y \in \mathcal{Y}$ { 0, 1, 2, …, 9 }

We want the machine to **learn** some function (algorithm) $f: X \to Y$ which maps the input set to the output set.

Machine Learning: The basics (cont'd)

- In **supervised** machine learning, we have a bunch of input and output pairs that we use to **train** the machine: $(x_1, y_1), (x_2, y_2), (x_3, y_3)$
- Assumption: There is a function (algorithm) $f: X \rightarrow Y$ such that $f(x_1) = y_1, f(x_2) = y_2$, and so on, for **all** of the data points.
- f is the "real" answer (algorithm) that we want the machine to learn.
- Using *n* data points, we want to find an approximation $\hat{f}$ for the "real" answer f.
- Goal: The approximation $\hat{f}$ that the machine learns gives mostly correct predictions on unseen data.

The machine learning workflow



AI problems in the real world

AI is everywhere!

- Search engines and content filtering: Google, Bing
- Product recommendation systems: Netflix, Amazon
- Speech recognition: Alexa, Siri
- Finance and predicting stocks: Goldman Sachs, Morgan Stanley
- Face or object recognition: Facebook

Now that we understand the problem setup, *how* do we actually have the machine learn the model $\hat{f}$?

We'll look at a few different approaches.

Probability and Statistics

What is a random variable?

- A **random variable** X is a variable whose values are outcomes of a random phenomenon.
- **Discrete** random variables take on a countable number of distinct values. For example, the value of a coin flip $X =$ {heads, tails}.
- **Continuous** random variables take on one of an infinite number of possible values. For example, *X* could represent height and take on any positive value: 5.5 feet, 5.7 feet, etc.

How do we compute probabilities?

- The **probability** that a random variable *X* takes on a particular value *A* describes how likely *X* will be equal to *A*, written as $P(X = A)$.
- Probability is a value between 0 and 1, where 0 means the event never happens and 1 means the event always happens.
- For example, $P(X = \text{heads}) = \frac{1}{2}$ and $P(X = \text{tails}) = \frac{1}{2}$ if *X* describes a regular, fair coin. We compute this by taking the outcome of interest over all possible outcomes:

$$P(X = \text{heads}) = \frac{|\text{ heads }|}{|\{\text{heads, tails}\}|} = \frac{1}{2}$$

Conditional probabilities

- We can also look at **conditional probability**, which is the probability of one event *A* happening given that another event *B* happened, or $P(A \mid B)$.
- Often $P(A \mid B)$ is different from $P(A)$. For example, $P(\text{snow today} \mid \text{winter})$ is likely much different than $P(\text{snow today})$.
- We compute conditional probability as:

$$P(A \mid B) = \frac{P(A \text{ and } B)}{P(B)}$$

Joint probability that both A and B happen

Probability that B happens

Bayes' rule

1. Using the previous definitions for conditional probabilities:

$$P(A \mid B) = \frac{P(A \text{ and } B)}{P(B)} \qquad P(B \mid A) = \frac{P(A \text{ and } B)}{P(A)}$$

2. Rewrite as:

$$P(A \text{ and } B) = P(A \mid B)\, P(B) \qquad P(A \text{ and } B) = P(B \mid A)\, P(A)$$

3. Set right hand sides equal to each other:

$$P(A \mid B)\, P(B) = P(B \mid A)\, P(A)$$

4. Divide by $P(B)$:

$$\boxed{P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}}$$

Bayes' rule

Bayes' rule in action…

Suppose that a particular cancer test is:

1. 97% correct for catching positive cases (3% of time gives a negative result when it should have been positive)
2. 95% correct for catching negative cases (5% of time gives a positive result when it should have been negative)

0.5% of the general population has this type of cancer. If a randomly sampled person tests positive (+), what is the probability that they actually have cancer?

$$\mathrm{P}(\text{cancer} \mid +) = \frac{\mathrm{P}(+ \mid \text{cancer})\ \mathrm{P}(\text{cancer})}{\mathrm{P}(+)}$$

$$= \frac{\mathrm{P}(+ \mid \text{cancer})\ \mathrm{P}(\text{cancer})}{\mathrm{P}(+ \mid \text{cancer})\ \mathrm{P}(\text{cancer}) + \mathrm{P}(+ \mid \text{no cancer})\ \mathrm{P}(\text{no cancer})}$$

$$= \frac{0.97 * 0.005}{0.97 * 0.005 + 0.05 * (1 - 0.005)} = 0.089 = \boxed{8.9\%}$$

Why is Bayes' rule important?

We see a surprising result in the previous example: there's only a 8.9% chance that someone who tests positive actually has cancer!

This is because we incorporated the **prior** information that the prevalence of cancer is very low in the population, leading to a low probability.

Our first AI model: Naïve Bayes classifier

- We can apply Bayes' theorem to create an AI algorithm that predicts whether an incoming email is spam or not!
- Assume we have a dataset of 8 "not spam" emails and 4 "spam" emails which we found in our inbox.
- We went through and counted the number of times the following words appeared in each email type overall: {dear, friend, money}

	Not Spam (8 emails)	Spam (4 emails)
Dear	8	2
Friend	5	1
Money	1	4

Naïve Bayes classifier (Example 1)

- We receive a new email that reads “dear friend.” We want to determine whether this email is spam or not spam using an AI classifier.
- Let’s say that we are trying to predict variable *y*, which can be “spam” or “not spam.”
- We are interested in these two conditional probabilities:

$$\mathrm{P}(\mathrm{y} = \mathrm{spam} \mid \mathrm{email} = \text{"dear friend"})$$
$$\mathrm{P}(\mathrm{y} = \mathrm{not\ spam} \mid \mathrm{email} = \text{"dear friend"})$$

- Whichever probability is higher will tell us our prediction for the email!

Naïve Bayes classifier (Example 1)

We can use Bayes' rule!

$$P(\text{spam} \mid \text{"dear friend"}) = \frac{P(\text{"dear friend"} \mid \text{spam})\, P(\text{spam})}{P(\text{"dear friend"})}$$

$$= \frac{P(\text{"dear"} \mid \text{spam})\, P(\text{"friend"} \mid \text{spam})\, P(\text{spam})}{P(\text{"dear friend"})} = \frac{\frac{2}{7} * \frac{1}{7} * \frac{4}{12}}{P(\text{"dear friend"})} \propto 0.014$$

$$P(\text{not spam} \mid \text{"dear friend"}) = \frac{P(\text{"dear friend"} \mid \text{not spam})\, P(\text{not spam})}{P(\text{"dear friend"})}$$

$$= \frac{P(\text{"dear"} \mid \text{not spam})\, P(\text{"friend"} \mid \text{not spam})\, P(\text{not spam})}{P(\text{"dear friend"})} = \frac{\frac{8}{14} * \frac{5}{14} * \frac{8}{12}}{P(\text{"dear friend"})} \propto \mathbf{0.136}$$

	Not Spam (8 emails)	Spam (4 emails)
Dear	8	2
Friend	5	1
Money	1	4

We conclude that the email is **not spam** since:

$P(\text{not spam} \mid \text{"dear friend"}) > P(\text{spam} \mid \text{"dear friend"})$

Naïve Bayes classifier (Example 2)

Now the text in the email reads “money friend”:

$$P(\text{spam} \mid \text{"money friend"}) = \frac{P(\text{"money friend"} \mid \text{spam})\, P(\text{spam})}{P(\text{"money friend"})}$$

$$= \frac{P(\text{"money"} \mid \text{spam})\, P(\text{"friend"} \mid \text{spam})\, P(\text{spam})}{P(\text{"money friend"})} = \frac{\frac{4}{7} * \frac{1}{7} * \frac{4}{12}}{P(\text{"money friend"})} \propto \mathbf{0.027}$$

$$P(\text{not spam} \mid \text{"money friend"}) = \frac{P(\text{"money friend"} \mid \text{not spam})\, P(\text{not spam})}{P(\text{"money friend"})}$$

$$= \frac{P(\text{"money"} \mid \text{not spam})\, P(\text{"friend"} \mid \text{not spam})\, P(\text{not spam})}{P(\text{"money friend"})} = \frac{\frac{1}{14} * \frac{5}{14} * \frac{8}{12}}{P(\text{"money friend"})} \propto 0.017$$

	Not Spam (8 emails)	Spam (4 emails)
Dear	8	2
Friend	5	1
Money	1	4

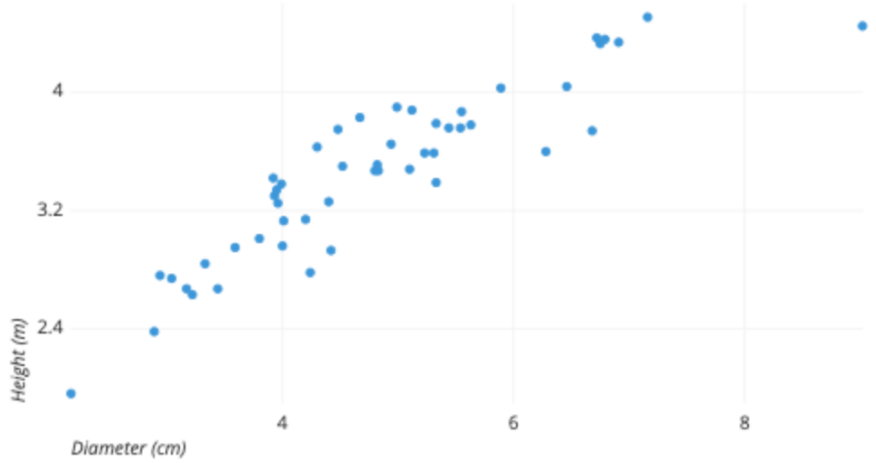
We conclude that the email is **spam** since:

$P(\text{spam} \mid \text{“money friend”}) > P(\text{not spam} \mid \text{“money friend”})$

Linear Regression

What is a scatter plot?

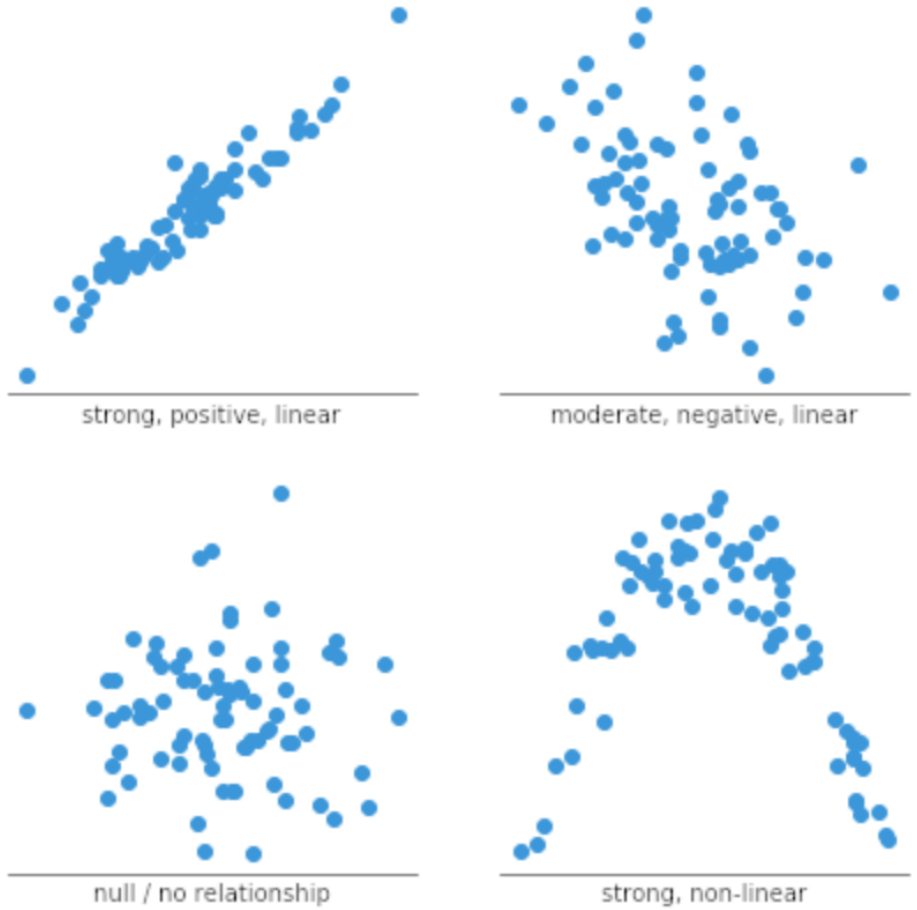
- A scatter plot (aka scatter chart, scatter graph) uses dots to represent values for two different numeric variables.
- The position of each dot on the horizontal and vertical axis indicates values for an individual data point.



[Scatter Plots | A Complete Guide to Scatter Plots (chartio.com)]

Let's look at some scatter plots

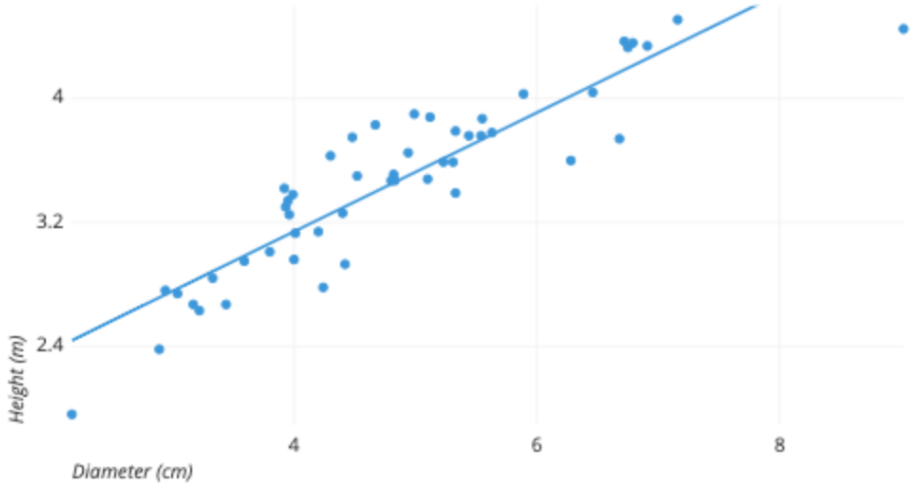
- Scatter plots are used to observe relationships between variables.
- In the plots, as you can see some of them seem to show some relationship between the variables



[Scatter Plots | A Complete Guide to Scatter Plots (chartio.com)]

Let's add a trend line to the scatterplot

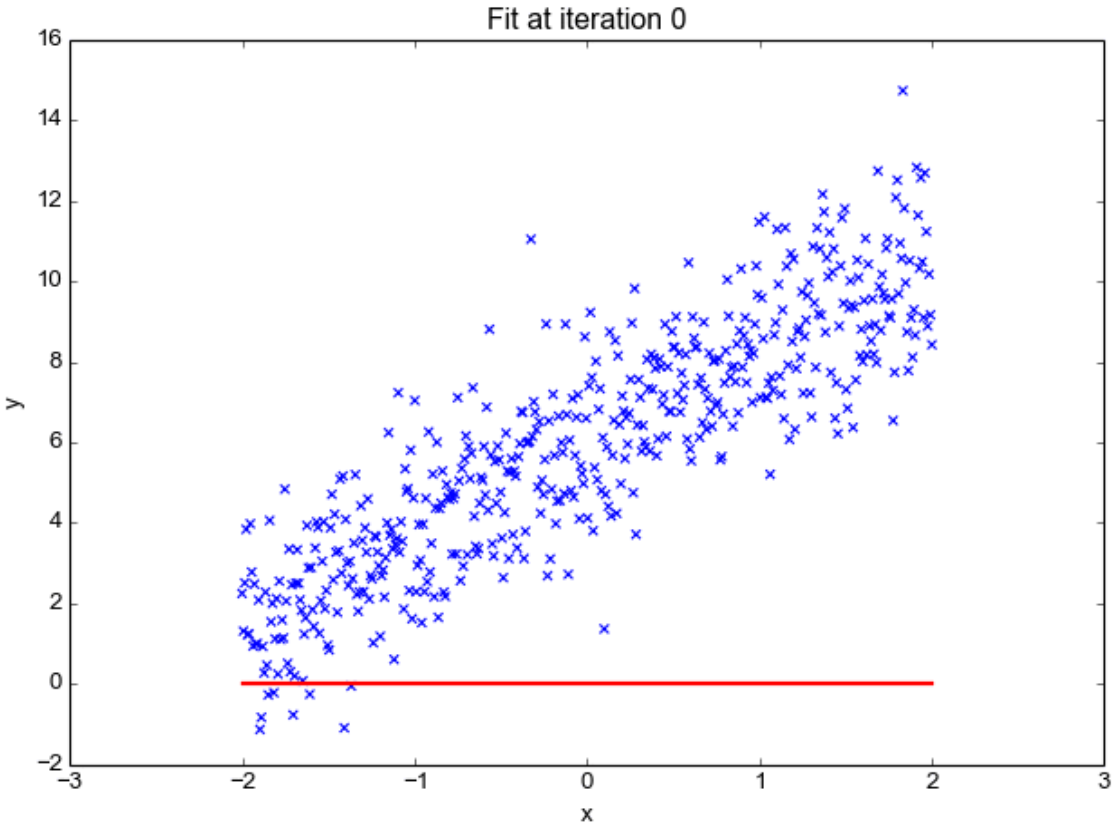
- When a scatter plot is used to look at a predictive or correlational relationship between variables, it is common to add a trend line to the plot showing the mathematically best fit to the data.
- This can provide an additional signal as to how strong the relationship between the two variables is, and if there are any unusual points that are affecting the computation of the trend line.



[Scatter Plots | A Complete Guide to Scatter Plots (chartio.com)]

Let's visualize Linear Regression

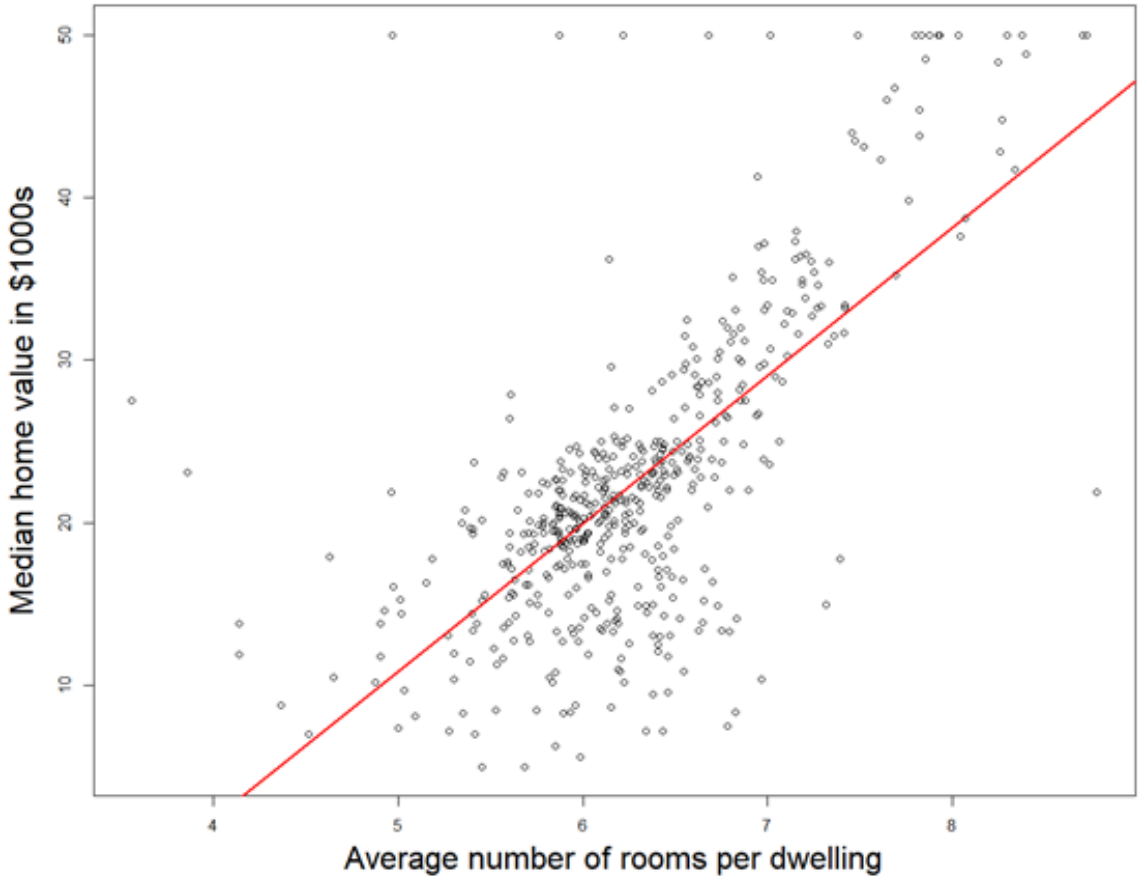
- Linear regression is a common statistical method, which has been adopted in machine learning and enhanced with many new methods for fitting the line and measuring error.



Linear Regression - PRIMO.ai

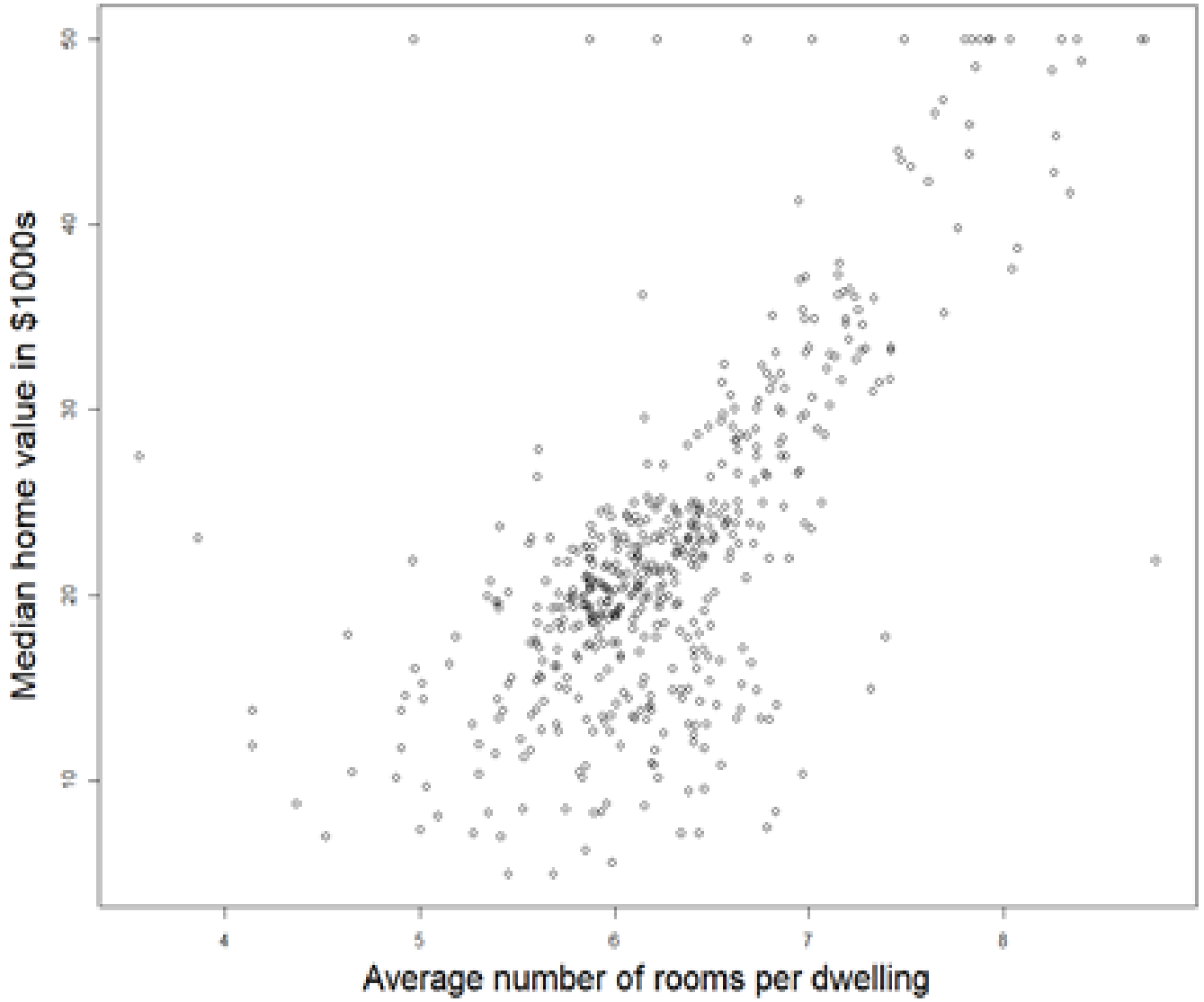
Formal definition of Regression

- **Regression analysis** is a set of statistical processes for estimating the relationships among variables.
- It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between an outcome and one or more predictors.



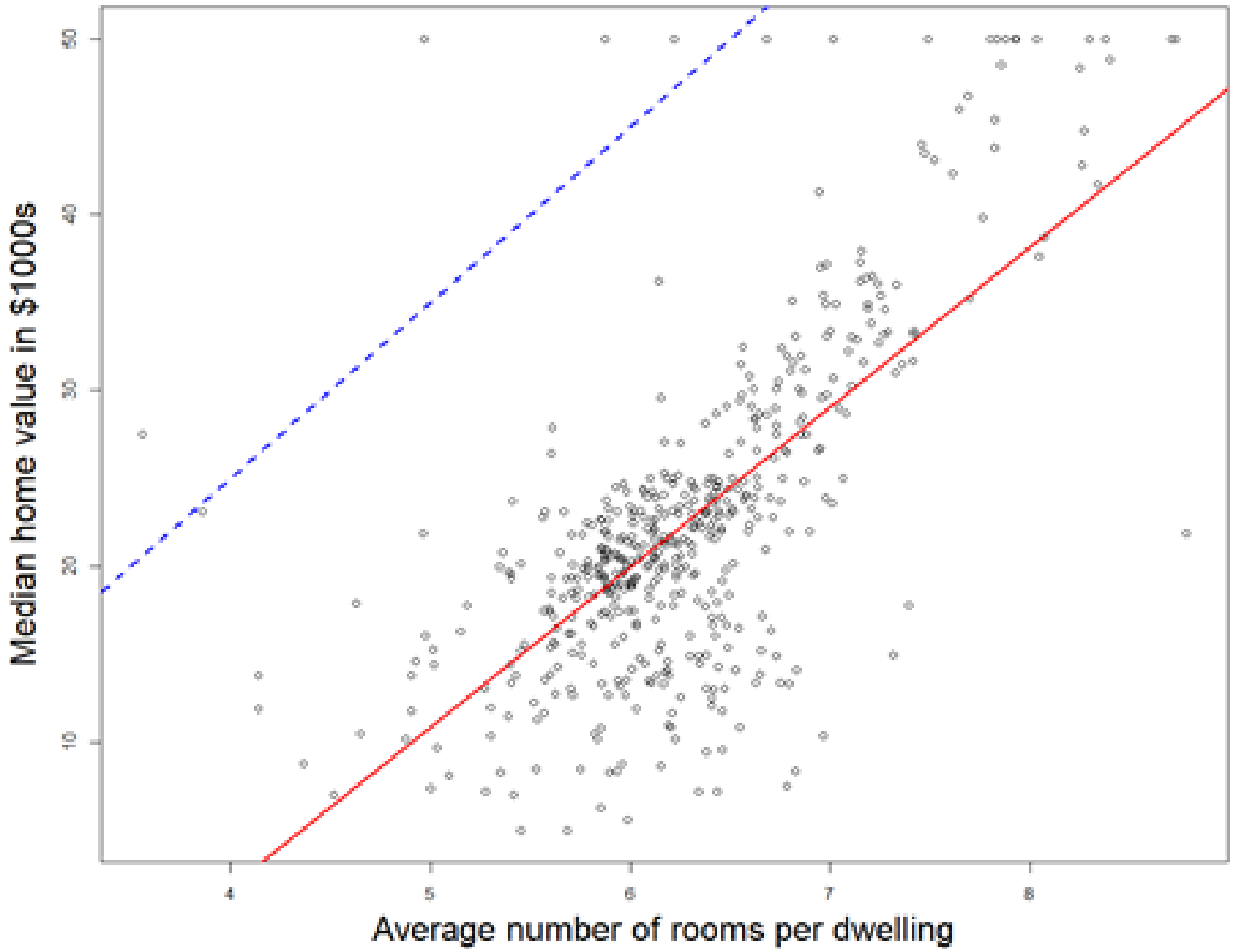
Simple Linear Regression

- Predict a quantitative outcome Y on the basis of a single predictor variable X.
- Maybe you are familiar with this: "Y = mx + b".
- Mathematically:
  $E[Y|X] = \beta 0 + \beta 1X$ or $Y = \beta 0 + \beta 1X + \varepsilon$
- Median home value = $\beta 0 + \beta 1$*Average number of rooms + ε.



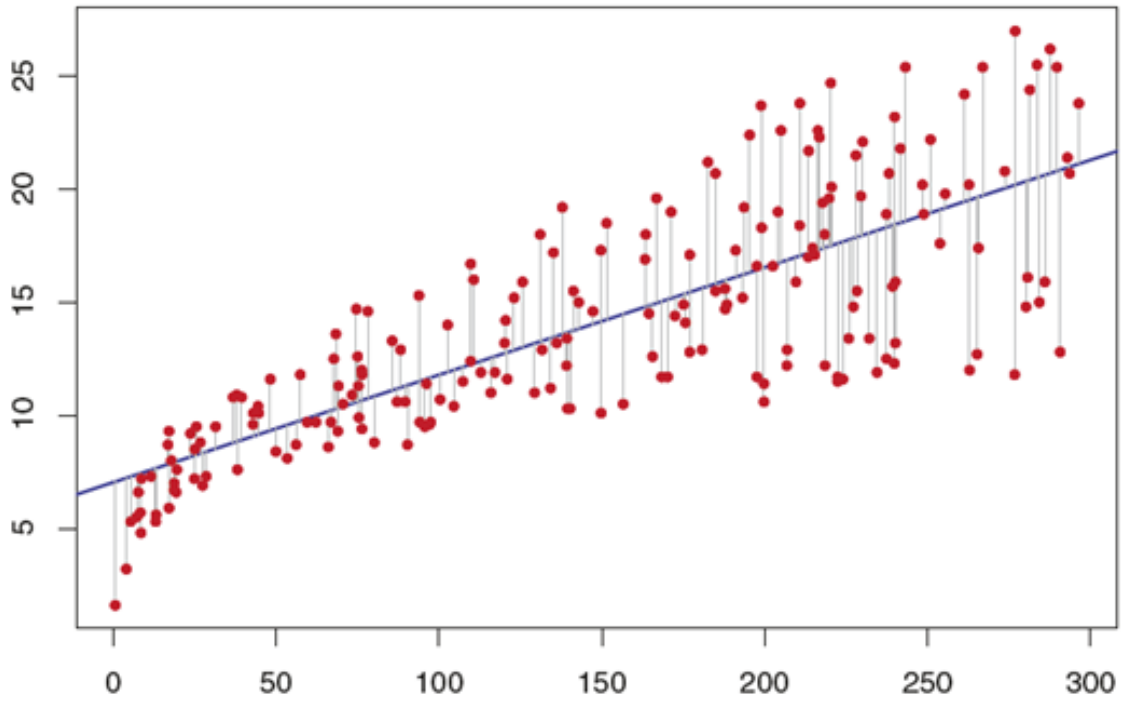
Estimate the coefficients

- In our model $Y = \beta 0 + \beta 1X + \varepsilon$
  Here
  $\beta 0$: “intercept”, the expected value of Y when X = 0.
  $\beta 1$: “slope”, the average increase in Y associated with a one-unit increase in X.
  ε: “error term” or “residual”.
- Which line fits the data better, blue or red?
  We need to calculate $\beta 0$ and $\beta 1$, so that the resulting line is as close as possible to these data points.



Estimate the coefficients

- There are a number of ways to define closeness. The most common approach involves minimizing the “residual sum of squares (RSS)”.
- Intuition: minimize the total squared vertical distance of each data point to a potential linear line.
- RSS = vertical distance1^2 + vertical distance2^2 + …
- RSS = (y1 - β0 - β1X1)^2 + (y2 - β0 - β1X2)^2 + …



R or Python can be used to do this!

- Our linear model: E[Median home value] = -34.7 + 9.1 × Average number of rooms.
- Interpretation: associated with a one-unit increase in average number of rooms, the median home value is estimated to increase by $9,100.

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -34.671      2.650  -13.08   <2e-16 ***
rm             9.102      0.419   21.72   <2e-16 ***
```

Goodness of fit: R-square

- R-squared statistic: measures the proportion of variability in Y that can be explained using X.
- It ranges from 0 to 1 (can be negative in some rare cases).
- Higher is better, but it can be challenging to determine what is good.
- In our case, R-squared = 0.48.

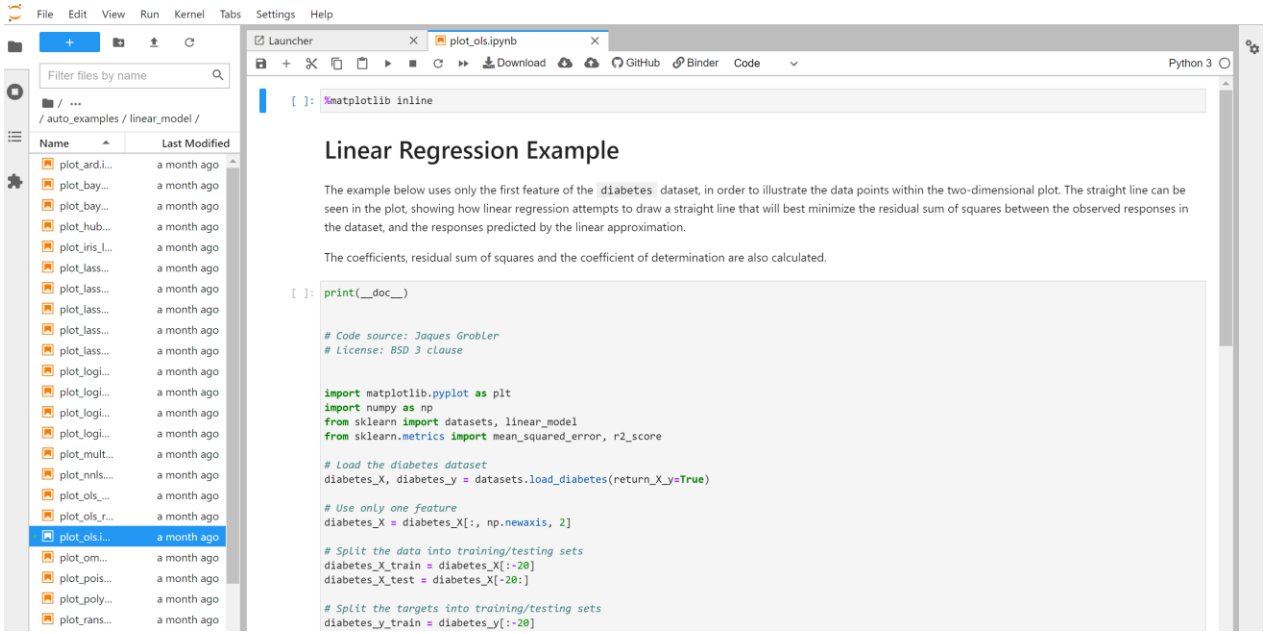
```
> fit1 <- lm(medv ~ rm, data = Boston)
> summary(fit1)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -34.671      2.650  -13.08   <2e-16 ***
rm             9.102      0.419   21.72   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 6.616 on 504 degrees of freedom
Multiple R-squared:  0.4835,    Adjusted R-squared:  0.4825
F-statistic: 471.8 on 1 and 504 DF,  p-value: < 2.2e-16
```

Assumptions when building a model

- Linearity: is the relationship between outcome and predictors really linear?
- Normality: does the error term follow a Normal distribution?
- Equal Variance: does the error term have equal variance?
- Independence of observations: does your observations intendent from each other? Eg. can we assume one observation is independent from another observation?

Try it out!

- [Linear Regression Example — scikit-learn 0.24.2 documentation]



Classification

Data for Making Predictions

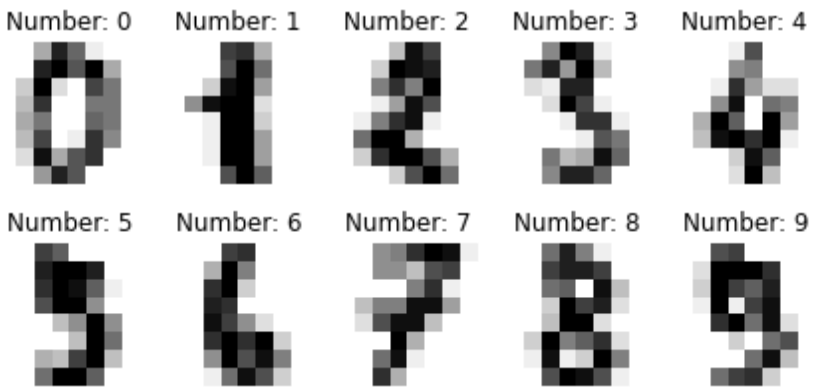
- When using statistics or machine learning to make predictions, we allow the model to learn based on labeled examples.
- A **label** is what we're predicting
- A **feature** is an input variable
- A **labeled example** includes feature(s) and the label.
- A dataset consists of many labeled examples.

Preparation of Data

- Try to get all numeric features on the same scale (usually, scale to between 0 and 1)
  - For example, compare the scale of house prices with the scale of square footage? ($100,000 vs 1200 sqft)
- Map categorical features to integers
  - For example, {apartment, condo, house} -> {0, 1, 2}
- Reserve some data to evaluate the model – called a train/test split
  - Model sees train data during training phase
  - Model is evaluated on test data after training – to show performance on unseen data

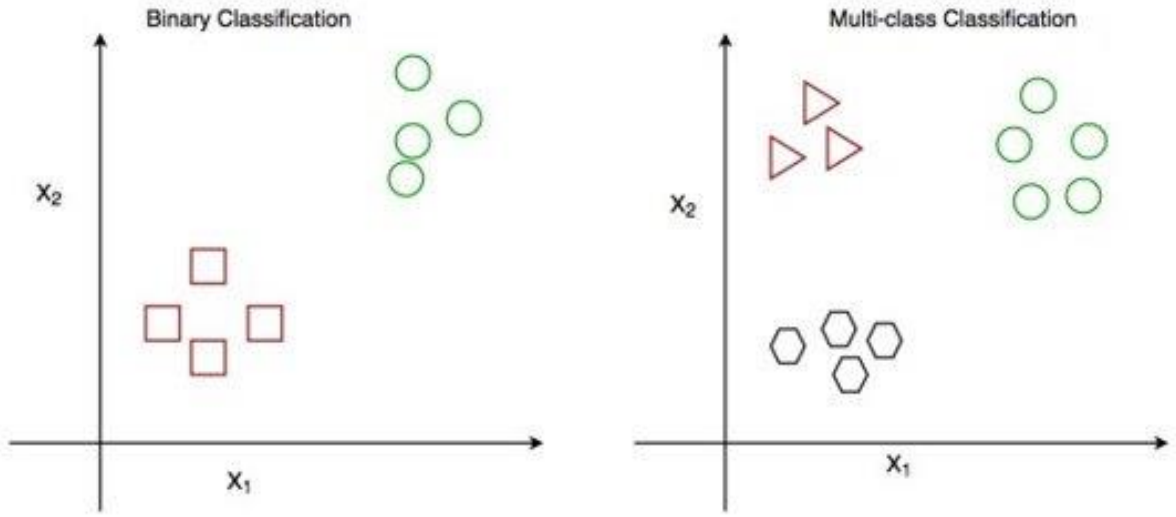
Defining the Classification Problem

- Learn how to make predictions from past examples
- What are the predictions? Finite set of class labels or categories
  - Given medical history, does an individual have an illness or not? (2 classes)
  - Given past weather information, will the weather tomorrow be sunny, rainy, or cloudy? (3 classes)
  - For each image, what digit 0-9 is represented? (10 classes)



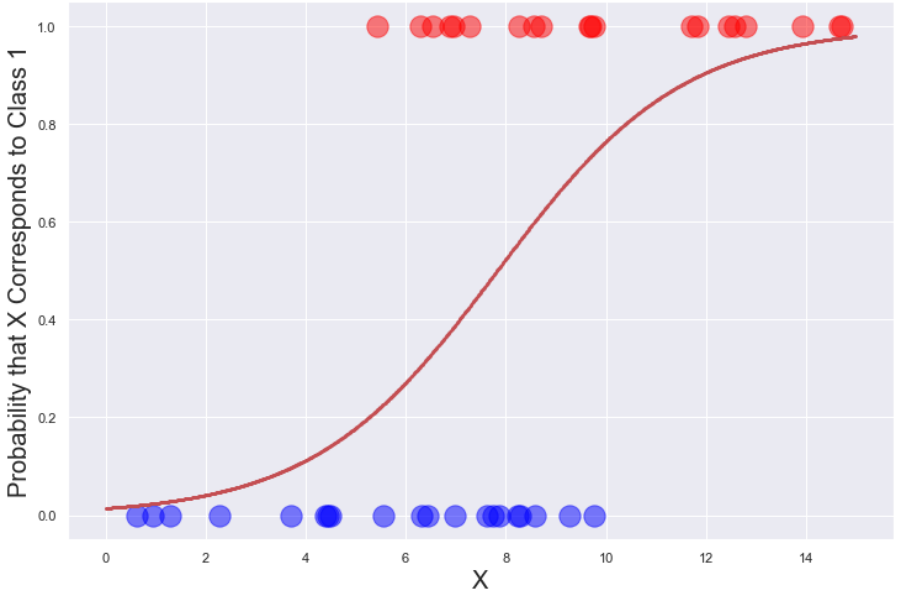
Binary Classification

- Classification where two classes are being predicted:
  - Is this email SPAM or NOT SPAM?
  - Is this sentence POSITIVE or NEGATIVE?
  - Given medical history, does an individual HAVE AN ILLNESS or NOT?



Binary Classification Approach

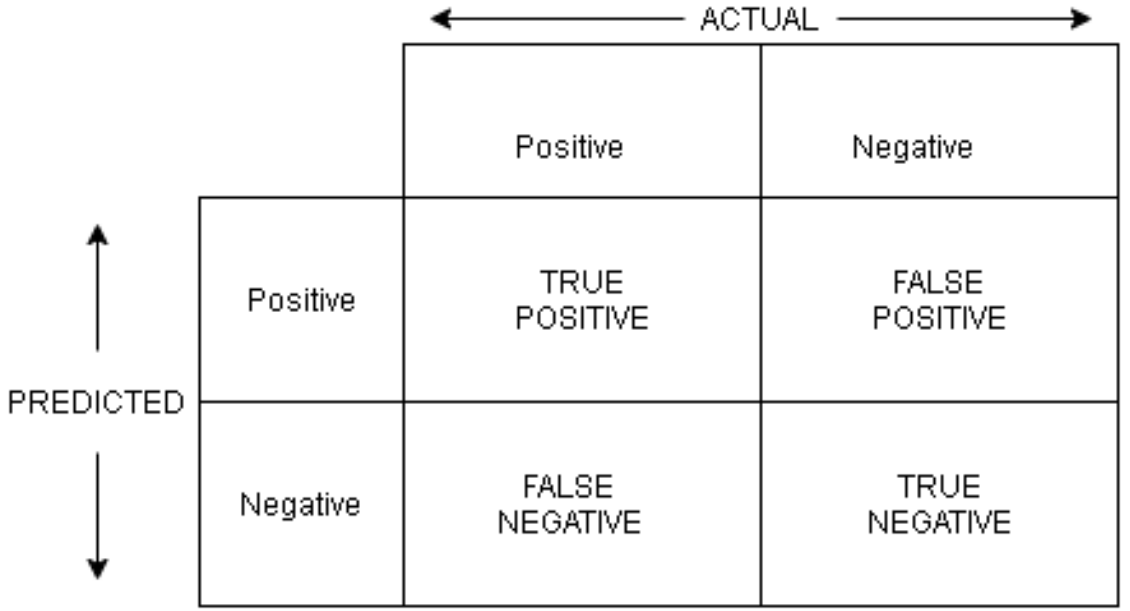
- Map one class to 0 (NOT SPAM) and the other class to 1 (SPAM)
- Model produces a score between 0 and 1
  - If this score is <= 0.5, classify example as NOT SPAM
  - If this score is > 0.5, classify example as SPAM
- Different methods for model to produce this score are different machine learning algorithms and approaches



Evaluating a Binary Classification Model

How do we know if our model is performing well?

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$



Confusion Matrix –
allows you to vizualize performance

Try it out!

Google CoLab Notebook

AI in Industry

Overview

Topics for the day

- Introduction to AI in Industry
- Natrual Language Processing
- Computer Vision
- Q&A

Speakers

- Daisy Deng– Natural Language Processing
- Max Kaznady – Computer Vision

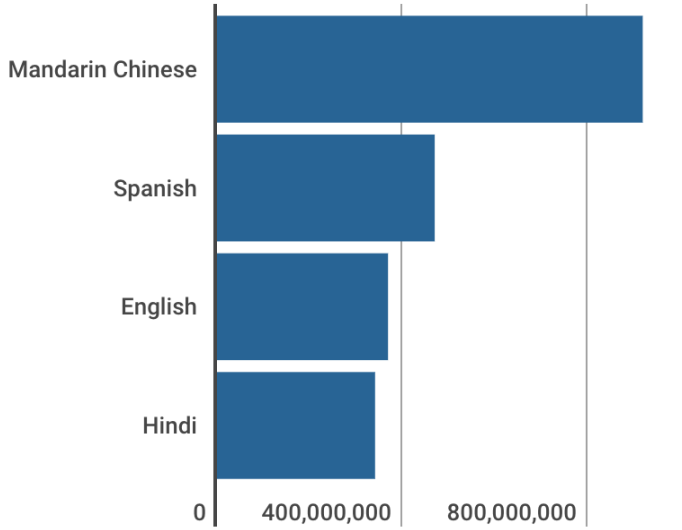
AI by Industry

https://gabi.sphera.com/industries/

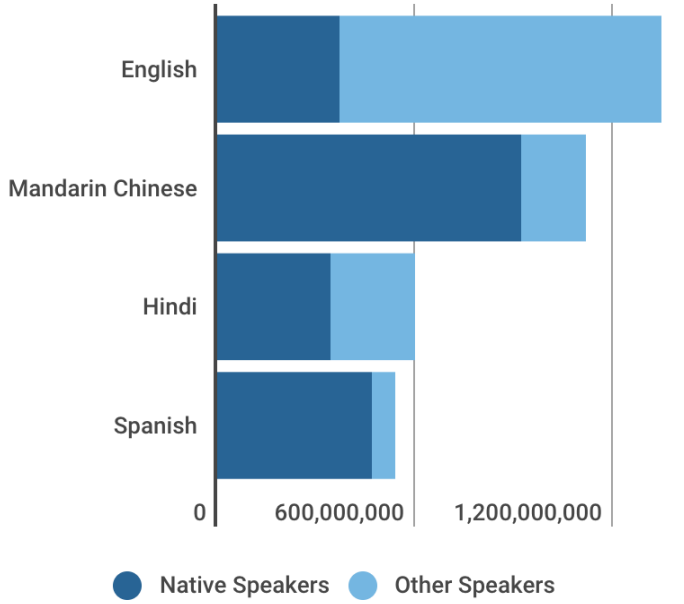
Natural Language Processing

What's the most spoken language?

Languages with the most native speakers



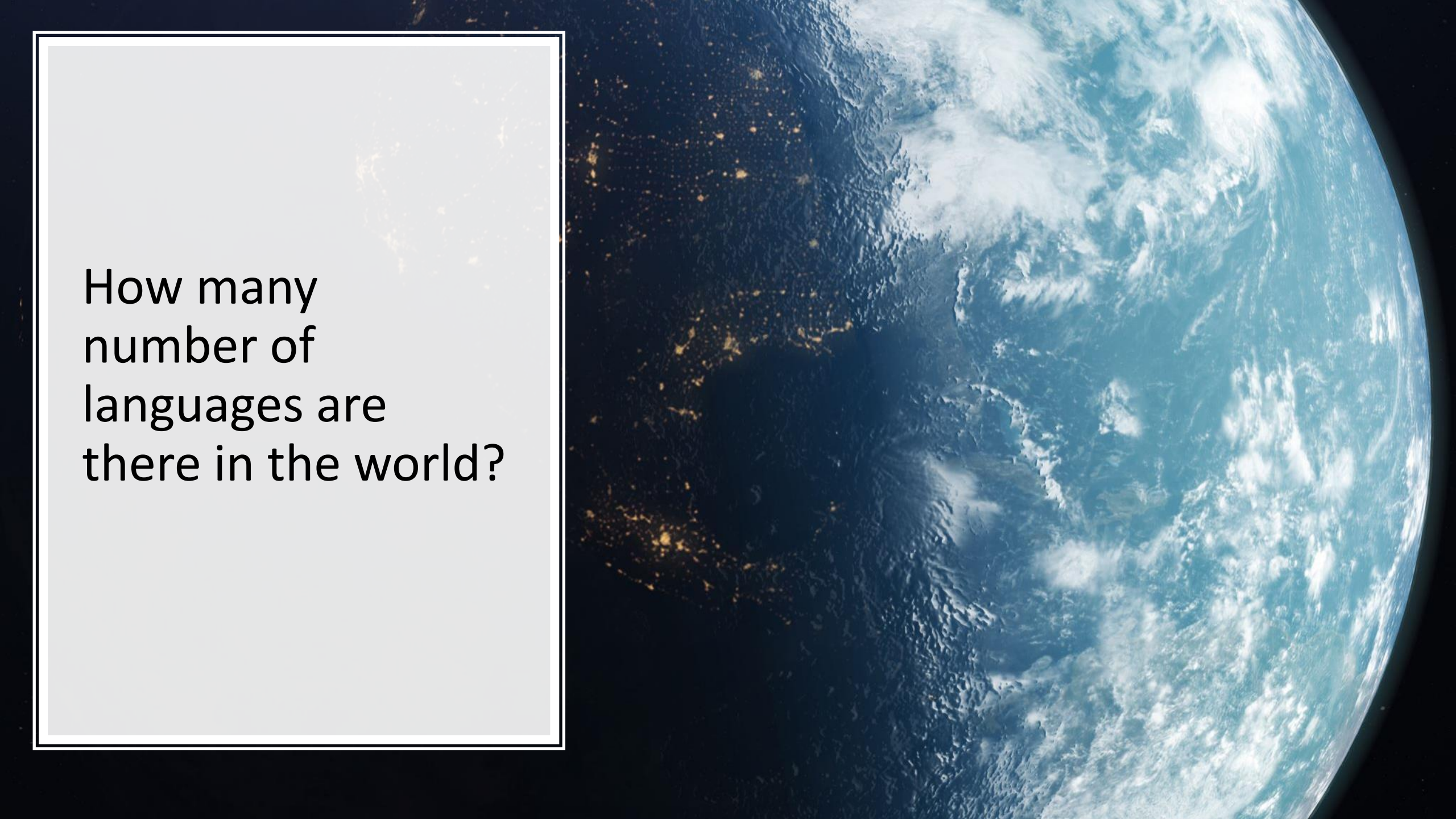
Languages with the most speakers



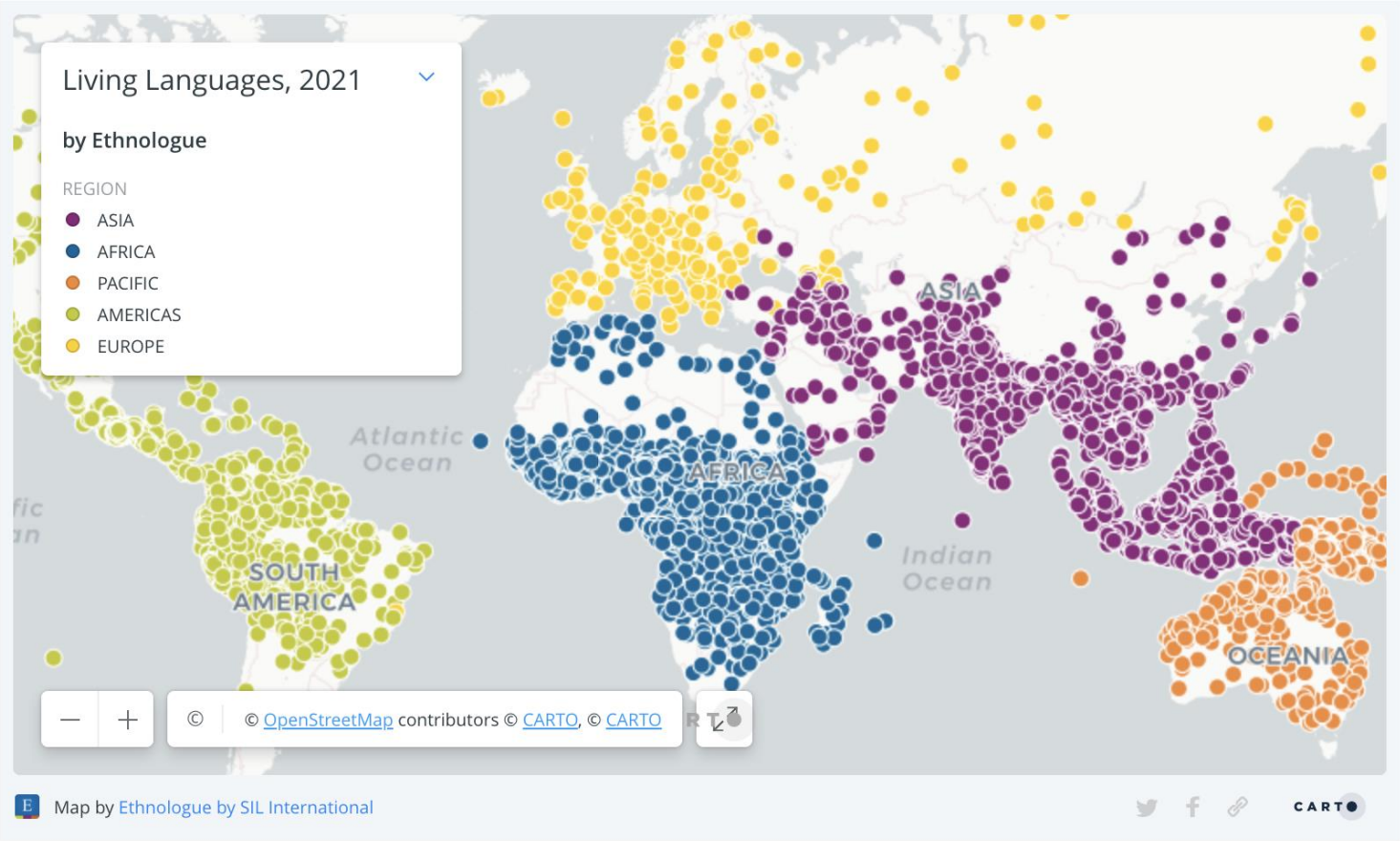
Mandarin Chinese is the largest language in the

https://www.ethnologue.com/guides/most-spoken-languages

How many number of languages are there in the world?



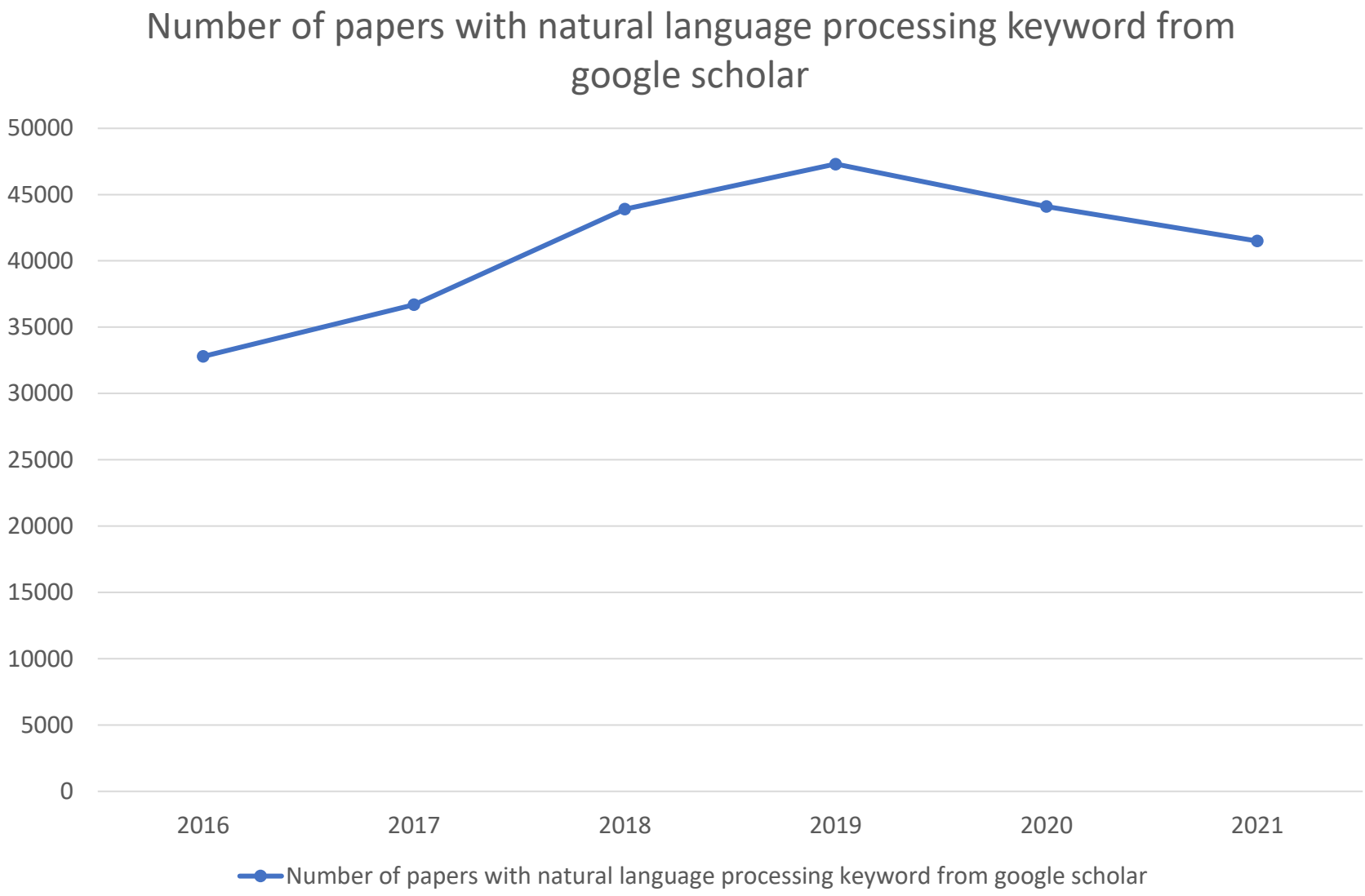
The number of natural languages in the world



4,065 have a developed writing system.

7,139 languages are spoken today

https://www.ethnologue.com/guides/how-many-languages
https://www.ethnologue.com/browse/countries


Number of papers with natural language processing keyword from google scholar
50000
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
2016
2017
2018
2019
2020
2021
Number of papers with natural language processing keyword from google scholar

What is **Natural Language Processing** NLP

Wiki definition

- subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language.
- Extract meaningful information from natural language input
- Produce useful natural language output based in the input

Modern Language applications

- Email categorization (spam, promotion, primary etc)
- Search engine (Bing, Google …)
- Chat bot (MSFT Cortana, Google Assistant, Apple Siri)
- Translation systems (Bing Microsoft Translator, Google Translate)
- Brand sentiment monitoring
- Customer review analysis
- News Digest
- Summarize for a 2nd grader (https://beta.openai.com/examples/default-summarize)

- …

NLP Applications

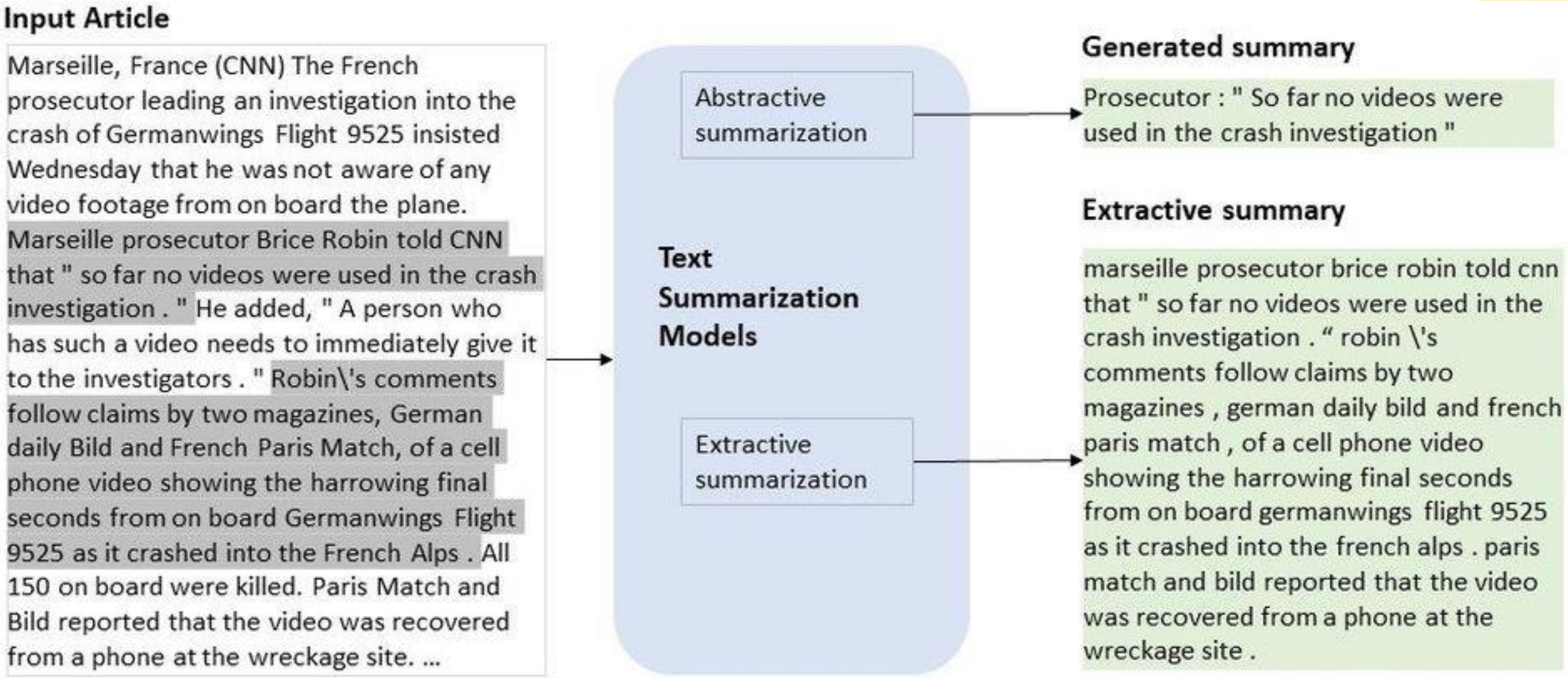
- Classification:
  - Best burger ever.
  - Service is disappointing.

- Named entity recognition
  - Mary have a doctor's appointment at 11AM EDT in xxx Massachusetts Ave, Cambridge.

  Entities: Person, event, time, location

- Translation:
  - Input: 我的名字叫黛西. (Simplified Chinese)
  - Output: My name is Daisy (English)

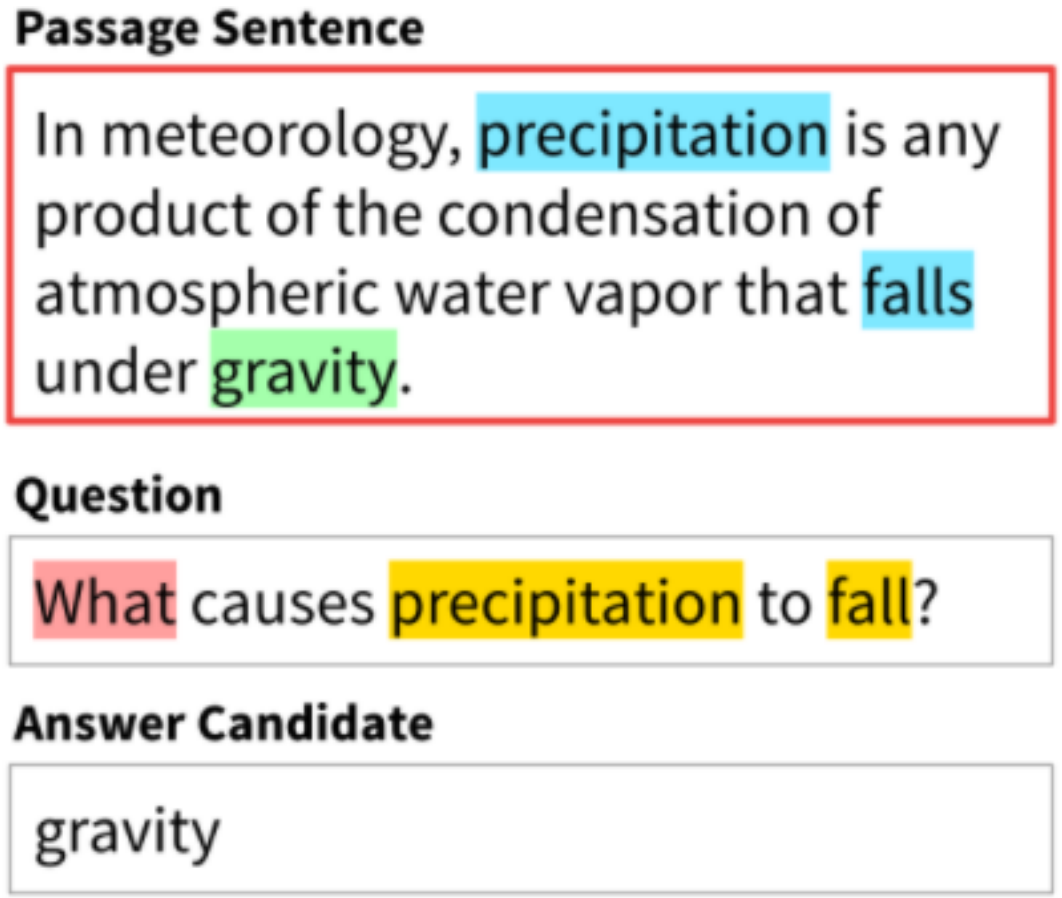
NLP Applications (Summarization)



NLP Applications (Continued)

- Question and Answering (extractive and abstractive)
  - Context:
  - Question
  - Answer

- More….



https://rajpurkar.github.io/mlx/qa-and-squad/

NLP Applications Pop Quiz

On a piece of tech/finance news, you want to find all the companies the news mentioned, what application would you use to solve the problem?

1. **Classification**
2. **Question and Answering**
3. **Summarization**
4. **Named Entity Recognition**

NLP Linguistic Tasks

- Tokenization: split sentences to words
  - “this is a simple sentence” -> [“this”, “is”, “a”, “simple”, “sentence”]
- Stop words removal: remove high-frequency words with little or no semantic value
  - “This is a simple sentence” -> “This simple sentence”
- Stemming: get the root form of a word
  - Fish, fishing, fishes -> fish
  - Better -> better
- Lemmatization: remove inflectional endings, return the base dictionary form of a word
  - Gone, going, went -> go
  - Good, better, best-> good

NLP Linguistic Tasks

- Part-of-speech Tagging (POS)
  - This is a simple sentence -> DT VBS DT JJ NN
- Dependency parsing
  - Syntax structure
- Constituency Parsing
  - Phrase structure

Movie Review Sentiment Analysis

Task: Given a piece of movie review text, determine if the review is positive or negative.

What NLP application would you use?

Dataset: Large Movie Review Dataset

- **https://ai.stanford.edu/~amaas/data/sentiment/**
- highly polar movie reviews
- 25,000 for training
- 25,000 for testing.

Data Cleaning/Text preprocessing

- Remove irrelevant parts.
- Tokenize the input
- Remove Stop Words
- Lemmatization

Convert text to features in classification model

- Question: you have learned classification models in the previous session. How would you translate the text into the numerical inputs for a classification model?

Split text to words and counting words!
Text vectorization: given a dictionary, one-hot encoding

N-gram model

- Multiple word is considered as a unit for data processing

- S1 = ”Natural language processing is interesting."
  S2 = ”Learning a programming language is interesting”

Uni-gram: Vocabulary = set(“a”, “interesting”, “is”, “natural”, “language”, “learning”, “processing”, ”programming”)

- Size of vocabulary is 8
- S1 = [0, 1, 1, 1, 1, 0, 1, 0]
- S2 = [1, 1, 1, 0, 1, 1, 0, 1]

N-gram model (Continued)

- "I love vanilla, but I hate chocolate"
- "I love chocolate, but I hate vanilla"

With Uni-gram model, order info is completely discarded

```
vectorizer = CountVectorizer(ngram_range=(1,1))
sentence = "I love chocolate, but I hate vanilla"
X = vectorizer.fit_transform([sentence])
print(f"features for \"{sentence}\" is {vectorizer.get_feature_names()} ")
sentence = "I love vanilla, but I hate chocolate"
X = vectorizer.fit_transform([sentence])
print(f"features for \"{sentence}\" is {vectorizer.get_feature_names()} ")
```

```
features for "I love chocolate, but I hate vanilla" is ['but', 'chocolate', 'hate', 'love', 'vanilla']
features for "I love vanilla, but I hate chocolate" is ['but', 'chocolate', 'hate', 'love', 'vanilla']
```

N-gram model (continued)

- S1 = ”Natural language processing is interesting."
  S2 = ”Learning a programming language is interesting”

Bi-gram: Vocabulary = set(“natural language”, “language processing”, “processing is”, “is interesting”, “learning a”, “a programming”, “programming language”, “language is”)

- Size of vocabulary is 8

- S1 = [1, 1, 1, 1, 0, 0, 0, 0]
- S2 = [0, 0, 0, 1, 1, 1, 1, 1]

N-gram model (Continued)

```
vectorizer = CountVectorizer(ngram_range=(2,2))
sentence = "I love chocolate, but I hate vanilla"
X = vectorizer.fit_transform([sentence])
print(f"features for \"{sentence}\" is {vectorizer.get_feature_names()} ")
sentence = "I love vanilla, but I hate chocolate"
X = vectorizer.fit_transform([sentence])
print(f"features for \"{sentence}\" is {vectorizer.get_feature_names()} ")
```

```
features for "I love chocolate, but I hate vanilla" is ['but hate', 'chocolate but', 'hate vanilla', 'love chocolat
e']
features for "I love vanilla, but I hate chocolate" is ['but hate', 'hate chocolate', 'love vanilla', 'vanilla but']
```

With Bi-gram model, some sequence info is kept

Document Frequency

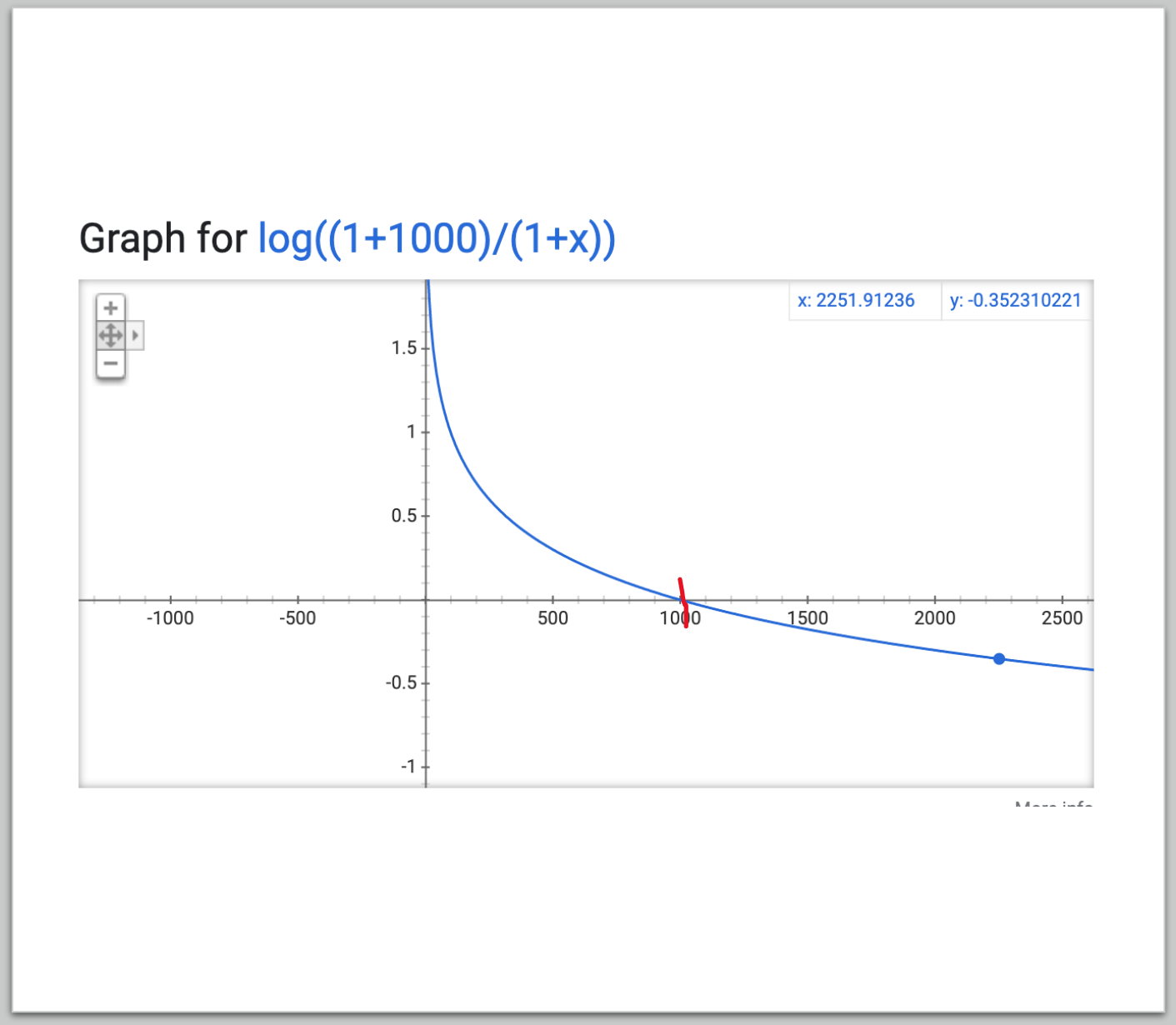
- Common words for movie review: director, movie, story, film ….
- Words that differentiate: worst, best, wonderful, awful

- The **document frequency** of a word is equal to the number of documents which contain the word divided by the total number of documents.
  - Total 100 reviews
  - document frequency of word “movie” might be close to 1
  - Document frequency of word “awful” probably much less than 1

Inverse Document Frequency

$$idf(w, D) = \log(\frac{1 + |D|}{1 + df(w, D)}$$

- Question: for words that are not occurred in the document, it would have a very high IDF. How can we mitigate this?



TF-IDF

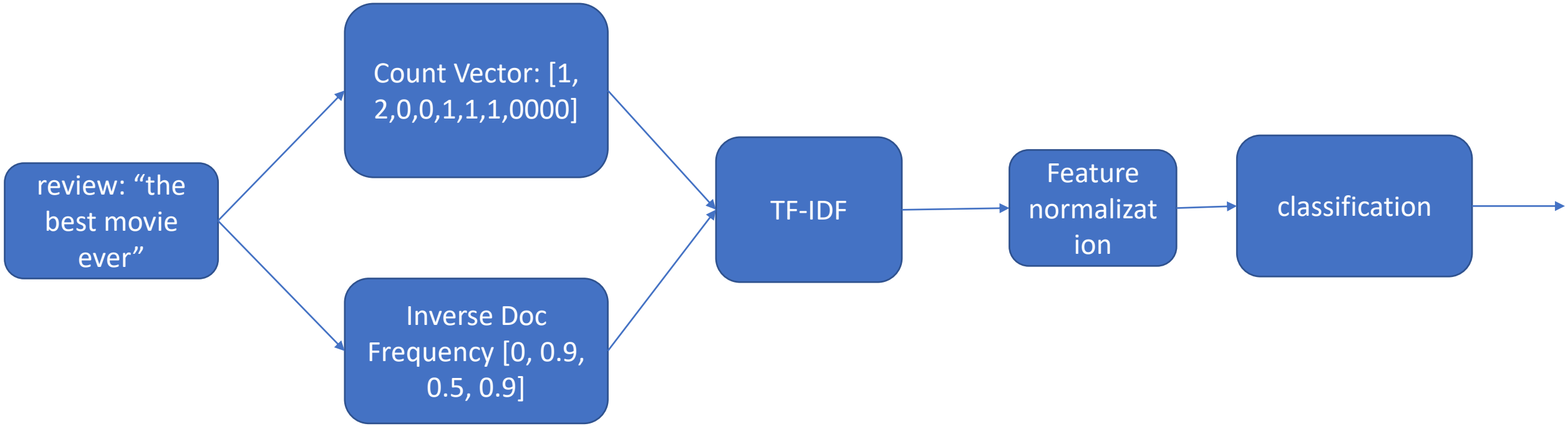
Term Frequency – Inverse Document Frequency

$$tf(w,d) = f_d(w)\text{: frequency of } w \text{ in document } d$$

$$idf(w,D) = \log(\frac{1+|D|}{1+df(w,D)}$$

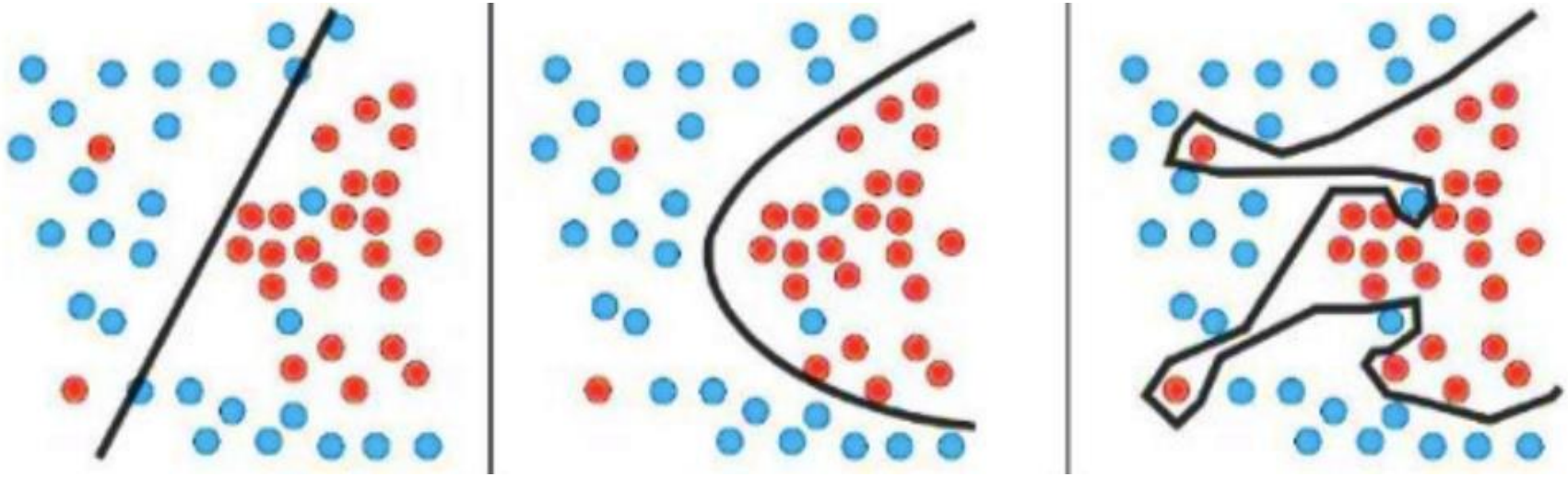
$$tfidf(w,d,D) = tf(w,d) \times idf(w,D)$$

Putting it together



What N should be in N-gram model?

Model Complexity vs Model Quality

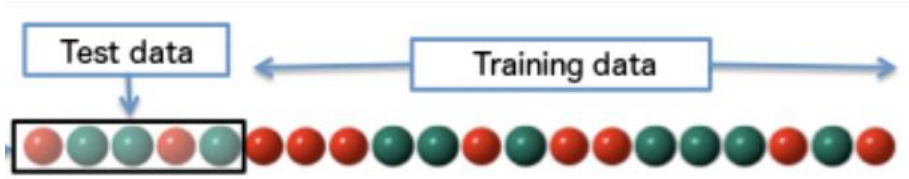


https://towardsdatascience.com/techniques-for-handling-underfitting-and-overfitting-in-machine-learning-348daa2380b9
https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229

Measure Model Quality/Performance

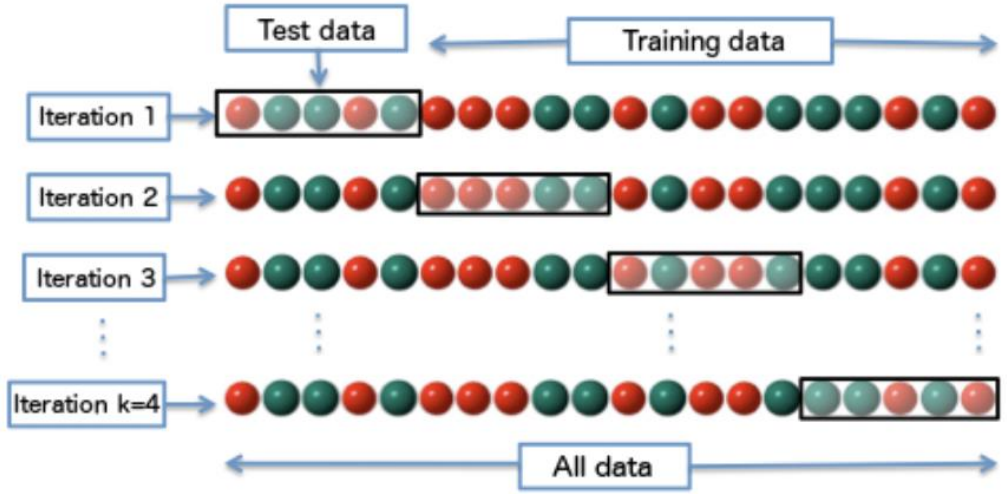
Is Bi-Gram better than Uni-Gram?

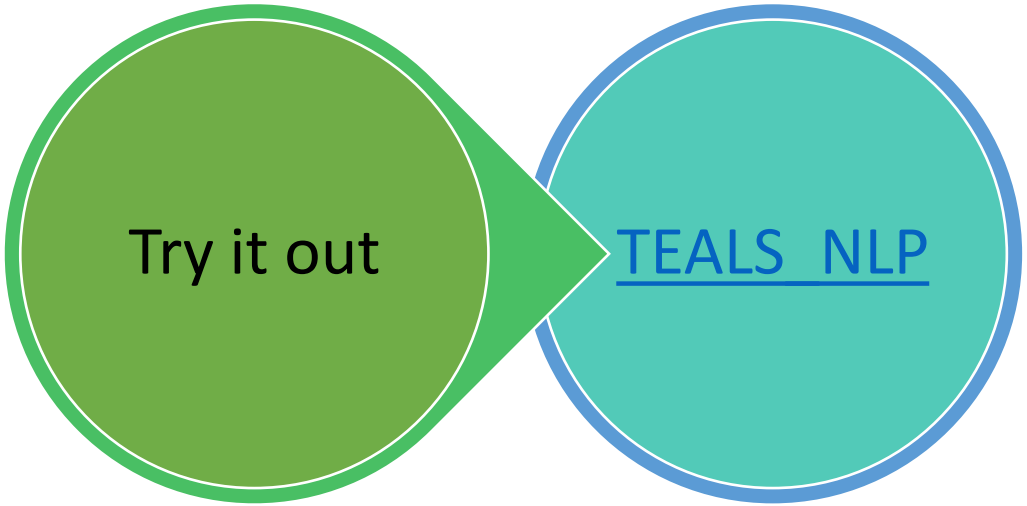
Train/test split



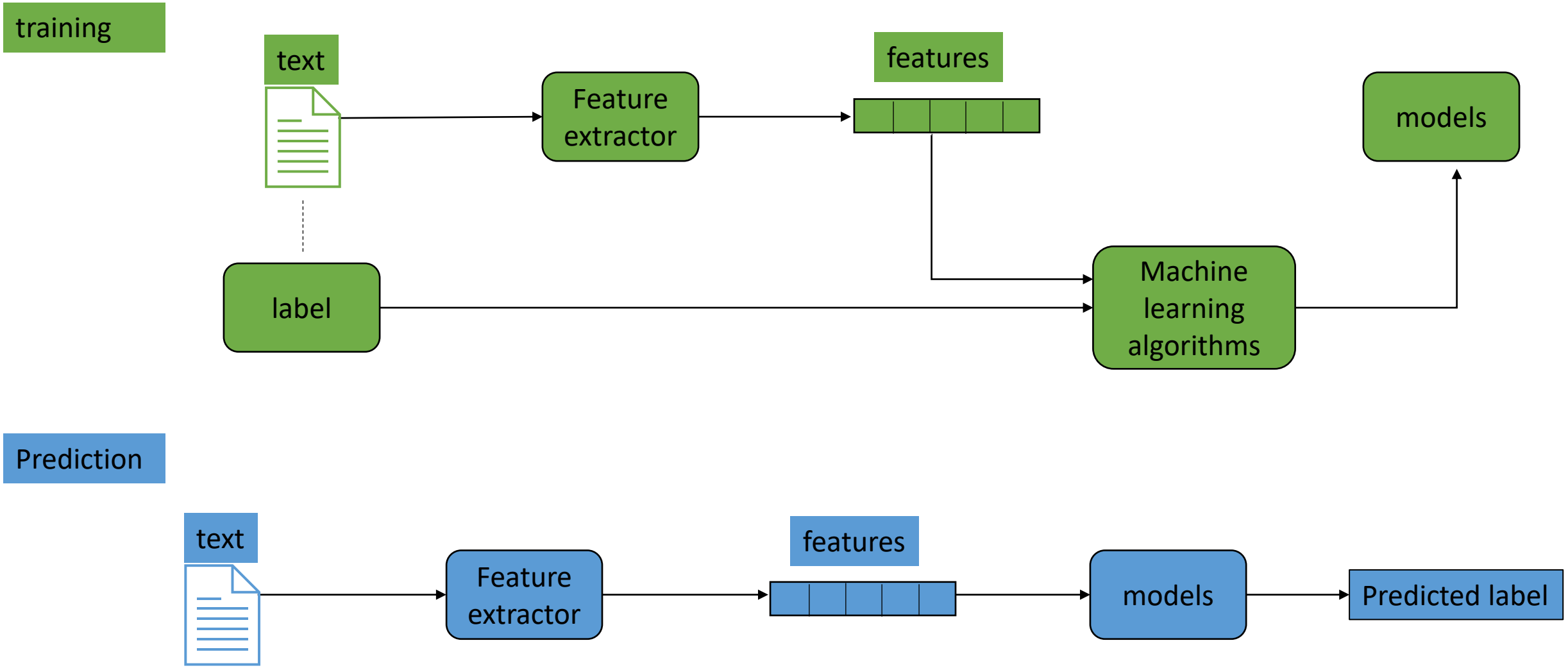
Can the Bi-gram model perform consistently well on unseen data?

Cross validation



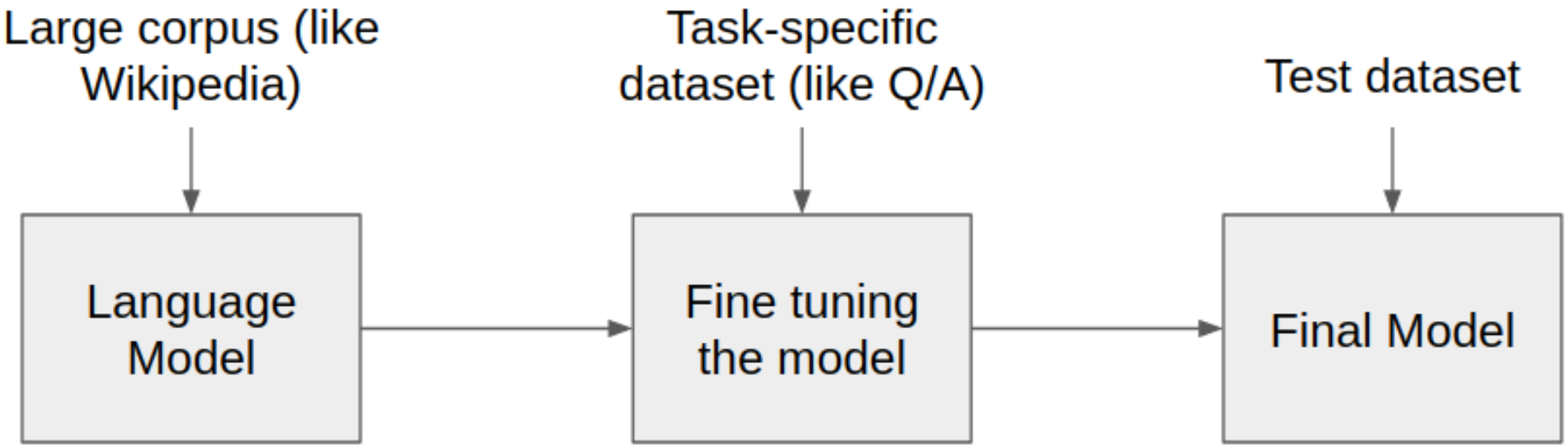

Try it out
TEALS NLP

NLP for Classification



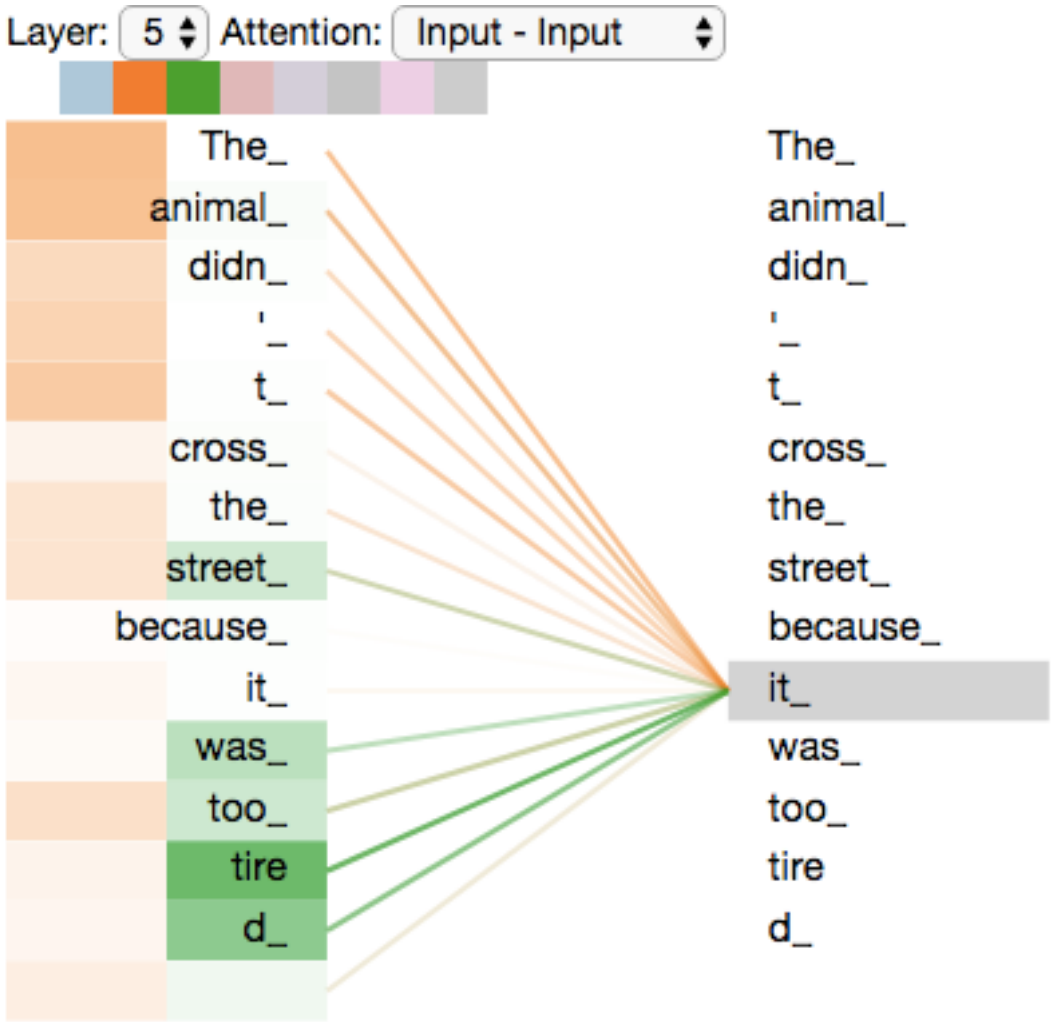
https://monkeylearn.com/natural-language-processing/

Transfer Learning



https://towardsdatascience.com/pre-trained-language-models-simplified-b8ec80c62217

BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers)



BERT (Cont'd)

- Encode information with longer dependencies from both directions
- **340 million parameters**
- More than 100 natural languages support
- Much better performance with much less labeled data

- https://www.kaggle.com/atulanandjha/bert-testing-on-imdb-dataset-extensive-tutorial

GPT-3

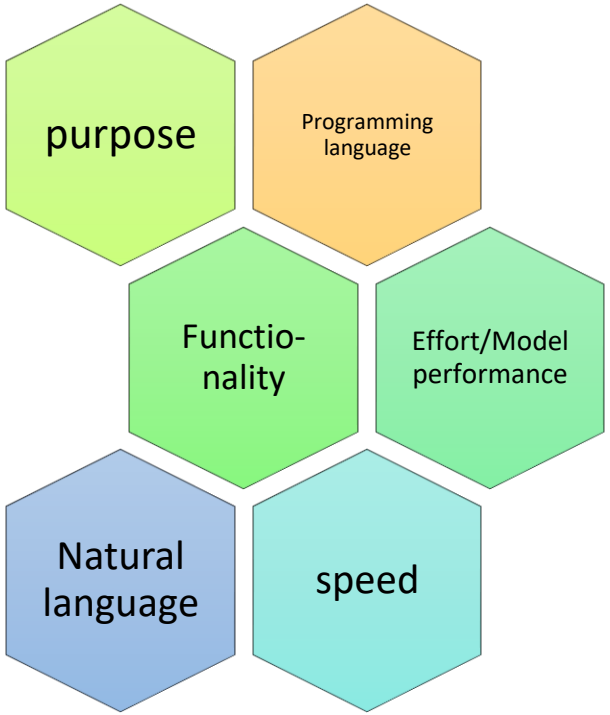
- **Generative Pre-trained Transformer 3**
- **175 billion parameters**

```
Q: What is your favorite animal?
A: My favorite animal is a dog.

Q: Why?
A: Because dogs are loyal and friendly.

Q: What are two reasons that a dog might be in a bad mood?
A: Two reasons that a dog might be in a bad mood are if it is hungry or if it is
hot.

Q: How many eyes does a giraffe have?
A: A giraffe has two eyes.

Q: How many legs does a frog have?
A: A frog has four legs.

Q: Are there any animals with three legs?
A: No, there are no animals with three legs.

Q: Why don't animals have three legs?
A: Animals don't have three legs because they would fall over.
```

https://lacker.io/ai/2020/07/06/giving-gpt-3-a-turing-test.html

Useful python packages for NLP

- NLTK
- Spacy
- CoreNLP
- Huggingface



https://analyticsindiamag.com/top-10-python-nlp-libraries-for-2019/

Computer Vision

Computer Vision Tasks

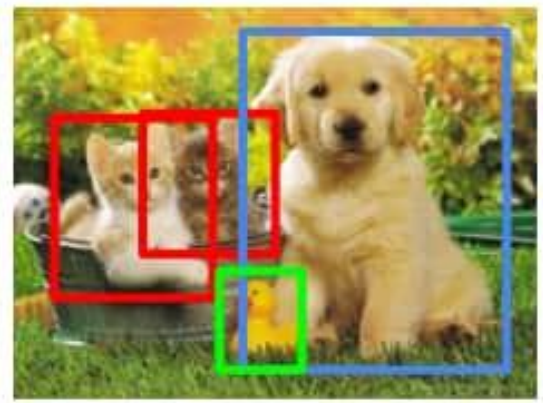
Classification

CAT

Classification + Localization

CAT

Object Detection



CAT, DOG, DUCK

Instance Segmentation

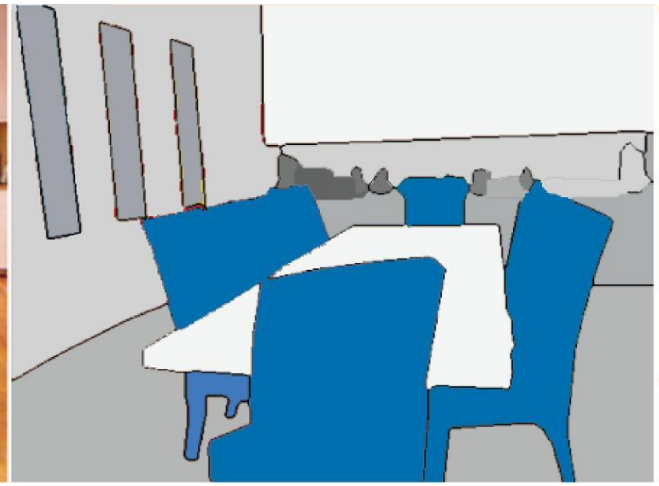
CAT, DOG, DUCK

Single object

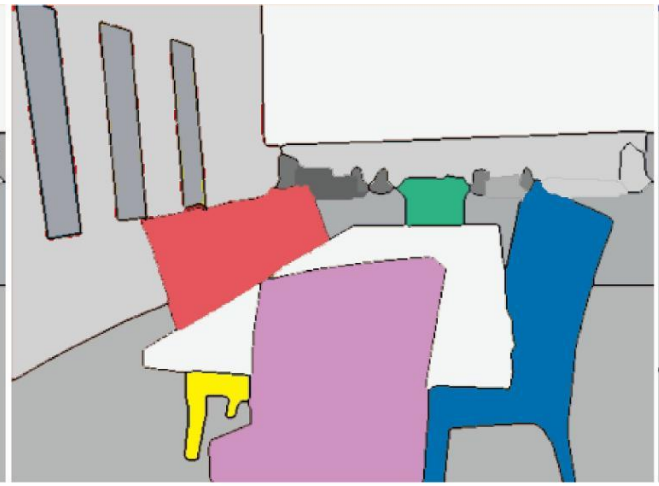
Multiple objects

Image Segmentation flavors

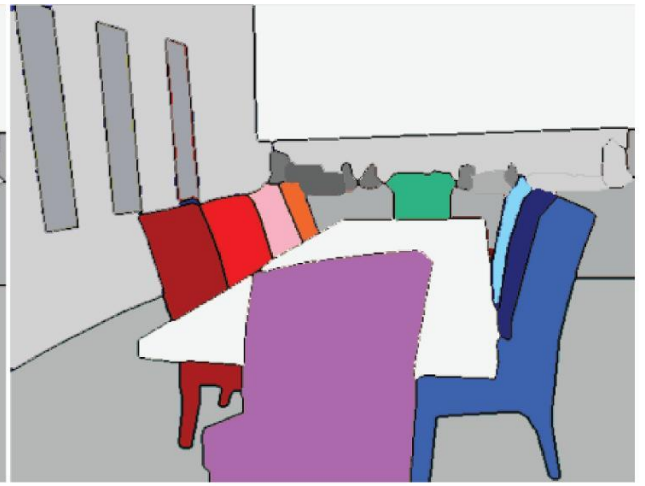
Input Image



Semantic Segmentation



Boundary Segmentation



Semantic Instance Segmentation

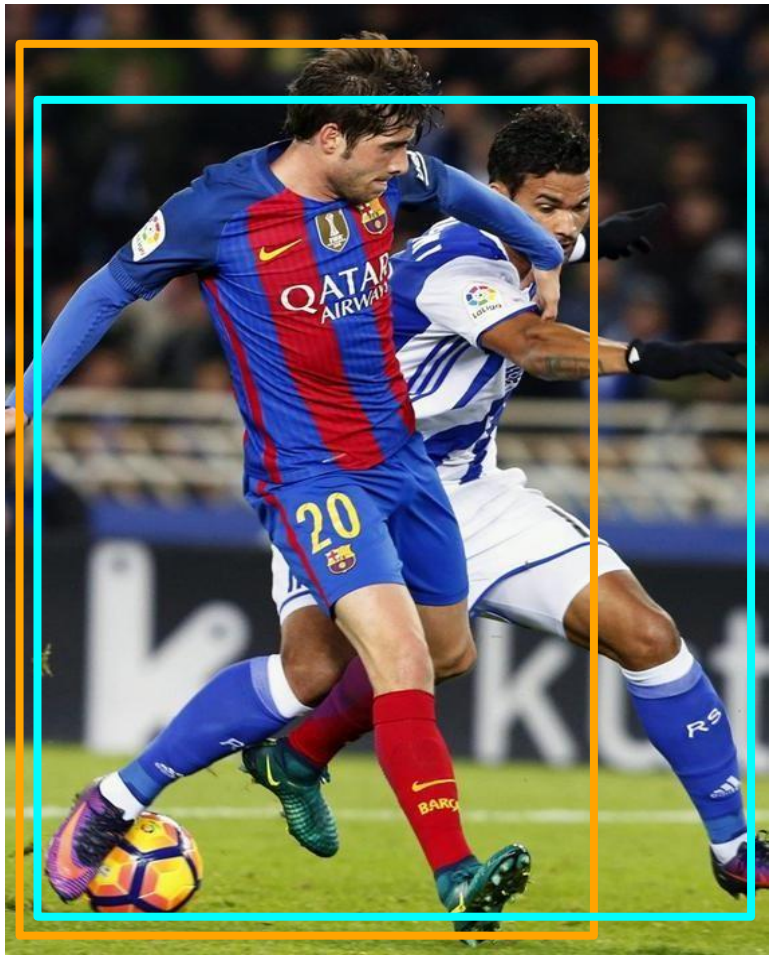
Image Masking

Boundary matters a lot in segmentation

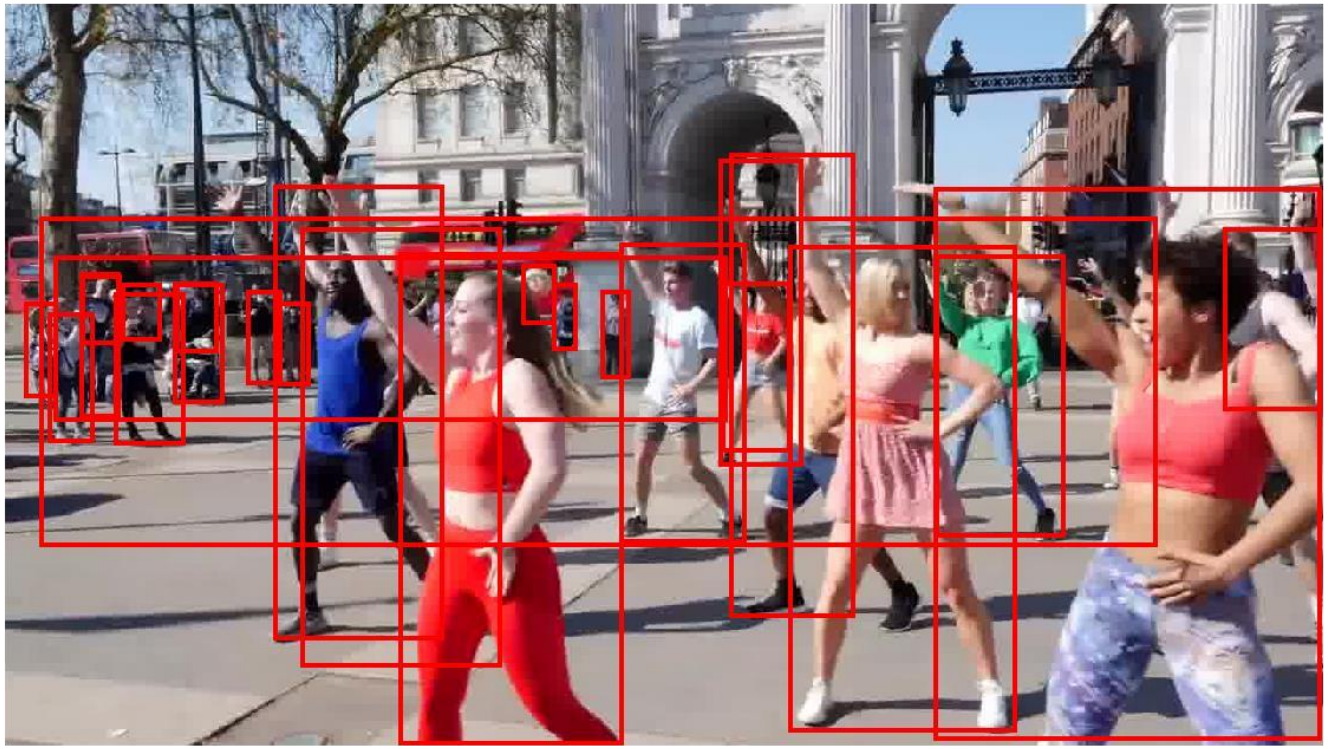
[GitHub - MVIG-SJTU/AlphaPose: Real-Time and Accurate Full-Body Multi-Person Pose Estimation&Tracking System]

CrowdPose - Challenges

Input Human Proposals

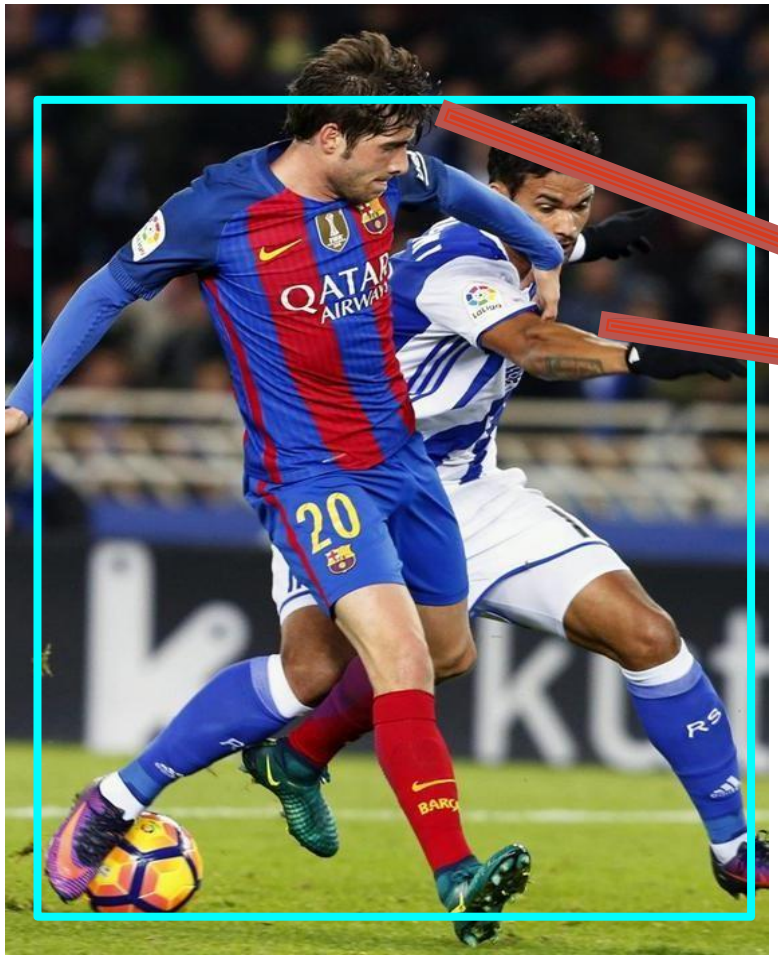


- One Bounding Box contains **multi person**.
- Human Detection fails in crowded scenes.



CrowdPose – Challenges

Input Human Proposals



- It is hard to identify which person this bounding box belongs to.

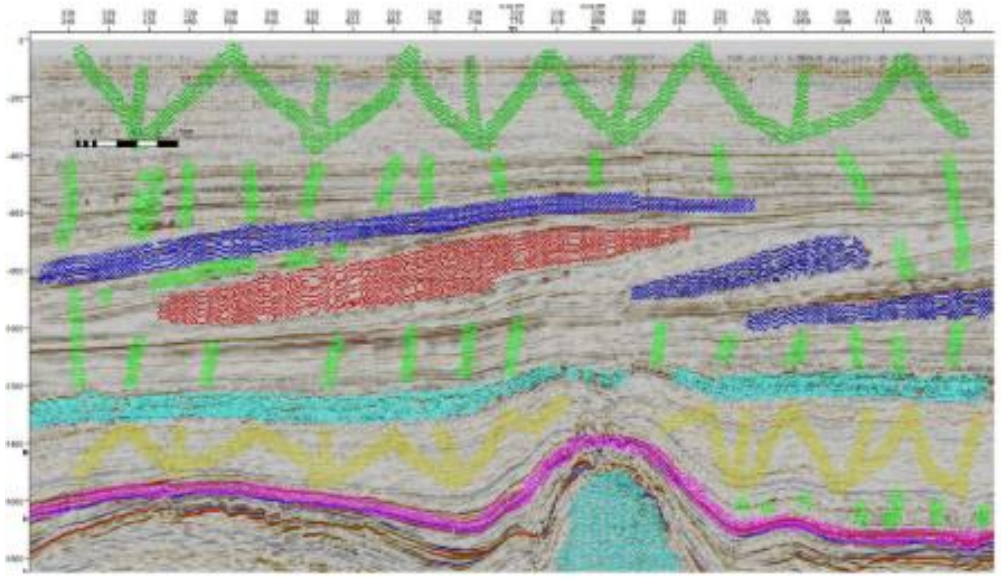
Infra-red image inspection

https://www.photonics.com/Articles/Chevron_Turns_to_Microsoft_Custom_Analytics_to/a66948

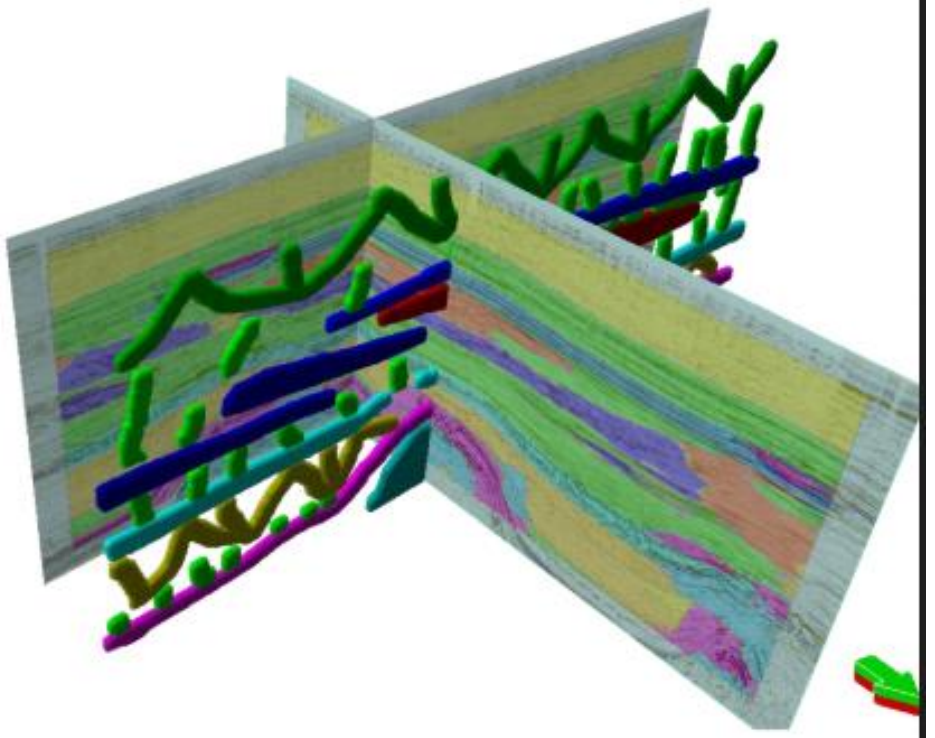
Seismic Data (3D)

Multi facies classification (F3 dataset)

One annotated inline with 9 facies classes



Auto-classified cube with 9 facies classes

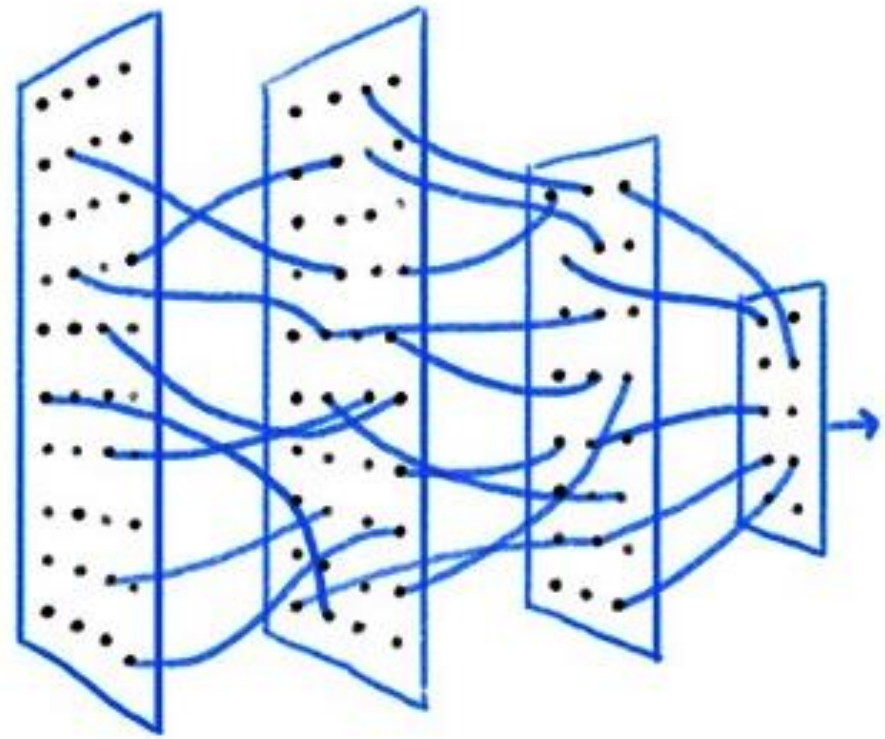


Data provided by MalenoV project:
https://github.com/bolgebrygg/MalenoV

https://github.com/microsoft/seismic-deeplearning

Cat

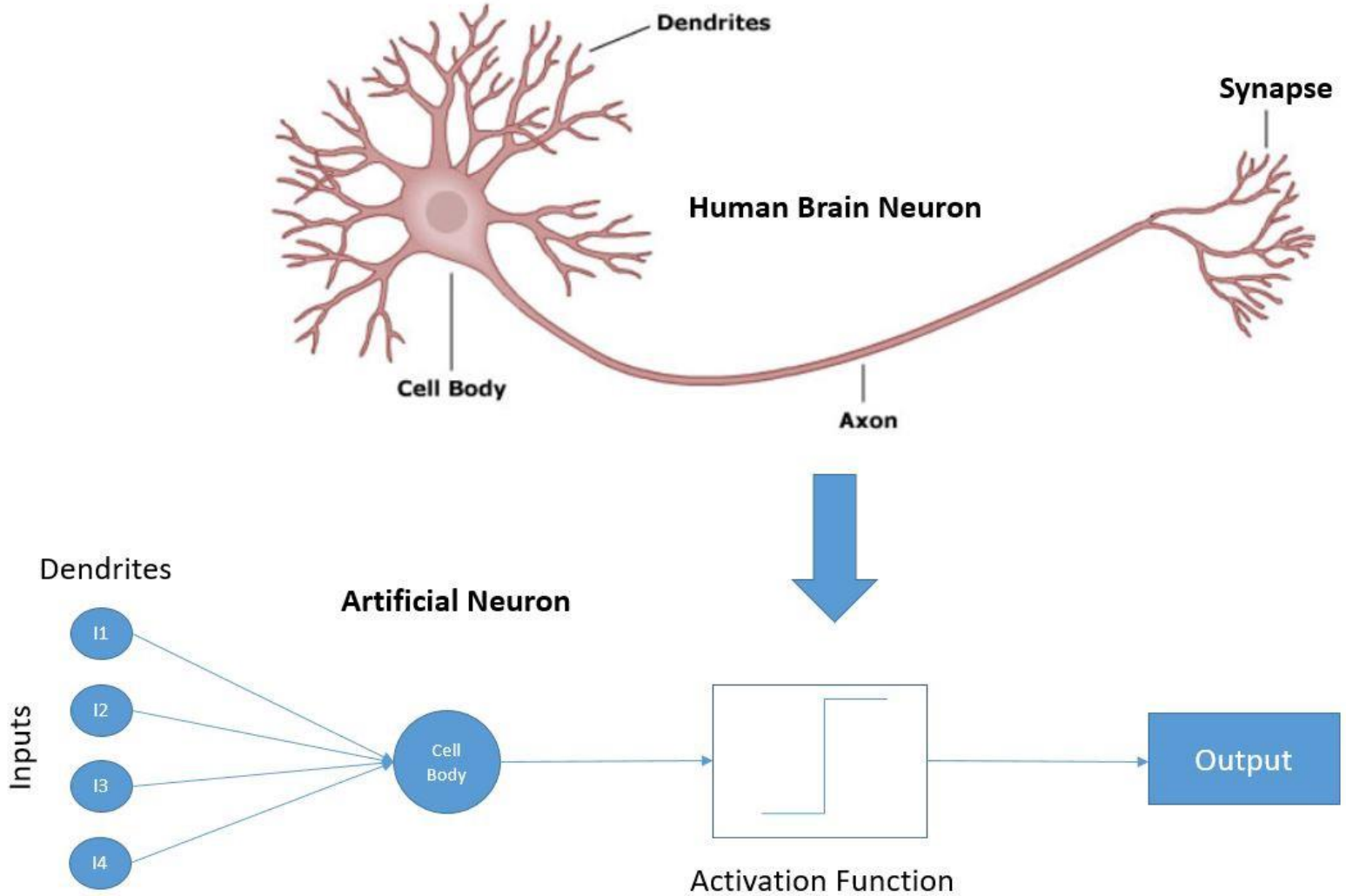
Dog



Output

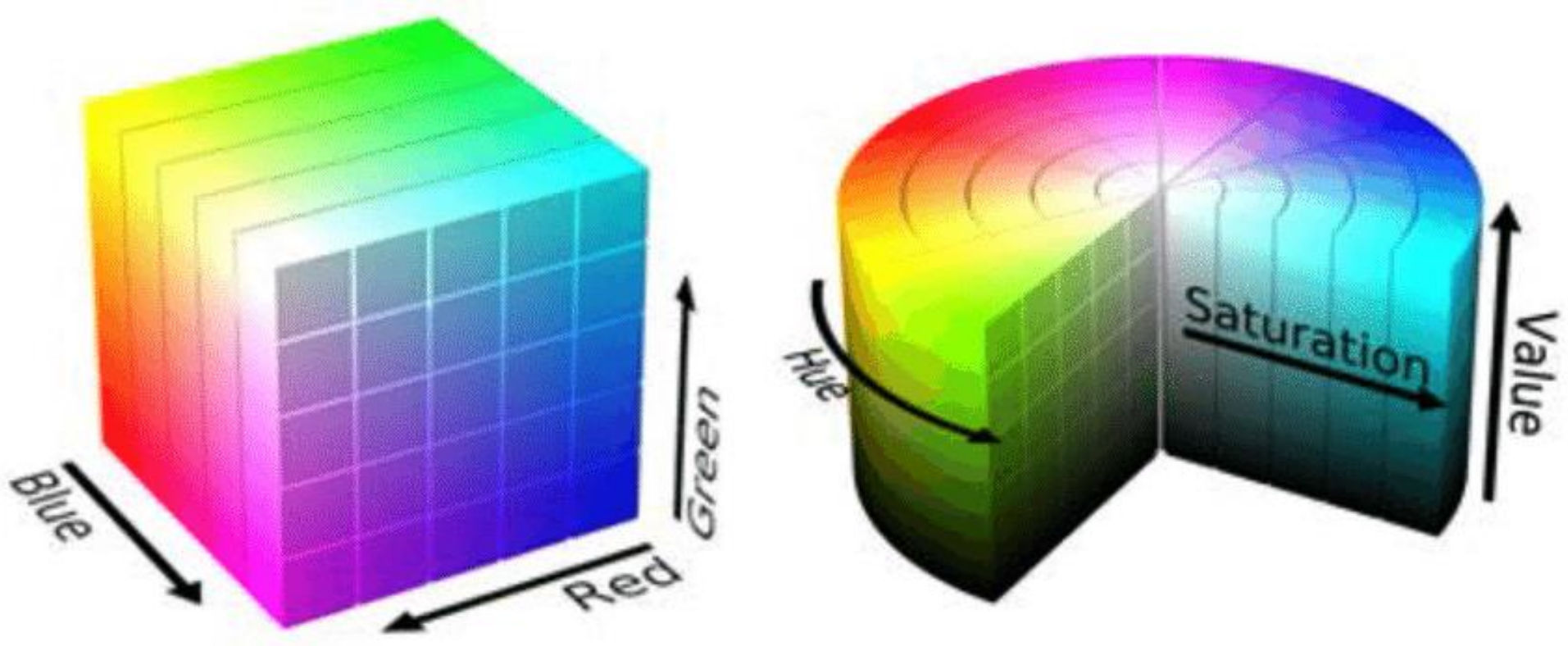
Cat

Why “neural” networks

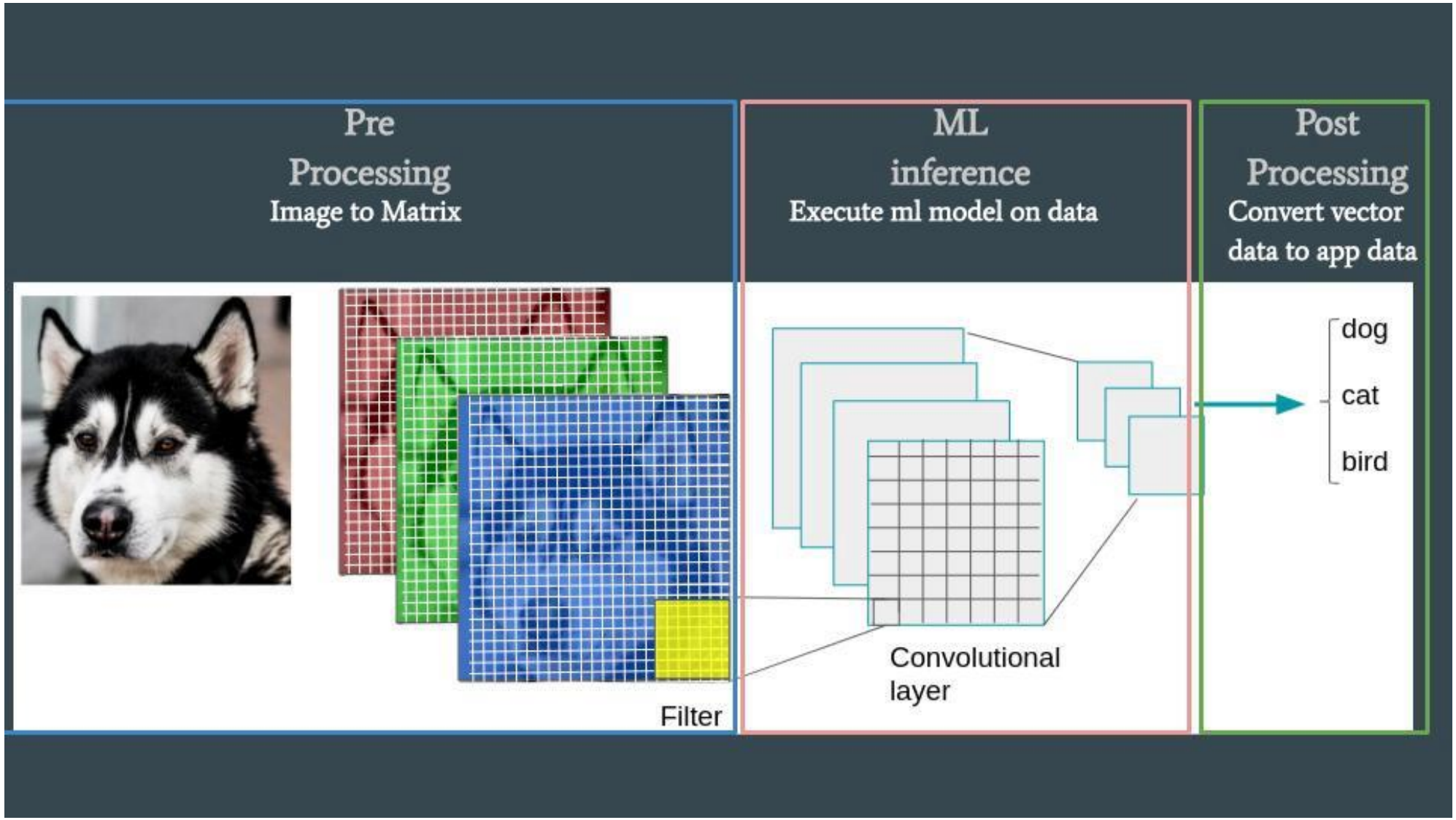


https://medium.com/@eshant.sah/deep-learning-in-simple-words-d6e027468836

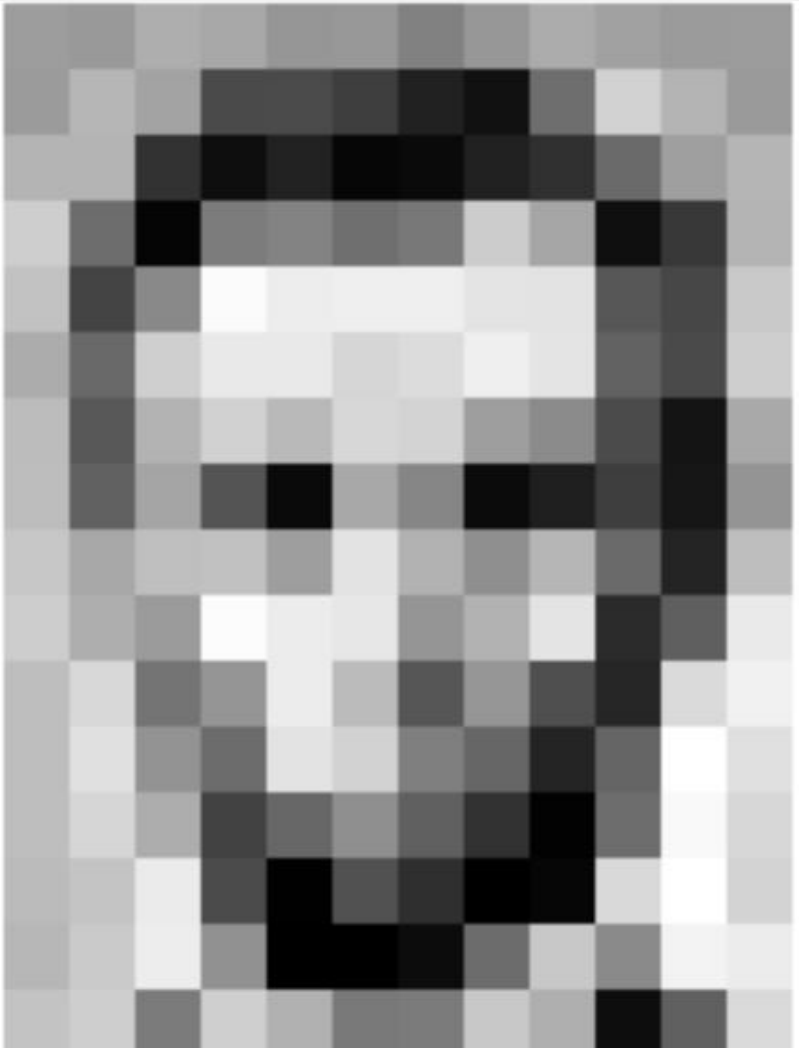
Color Space Representation



RGB image representation

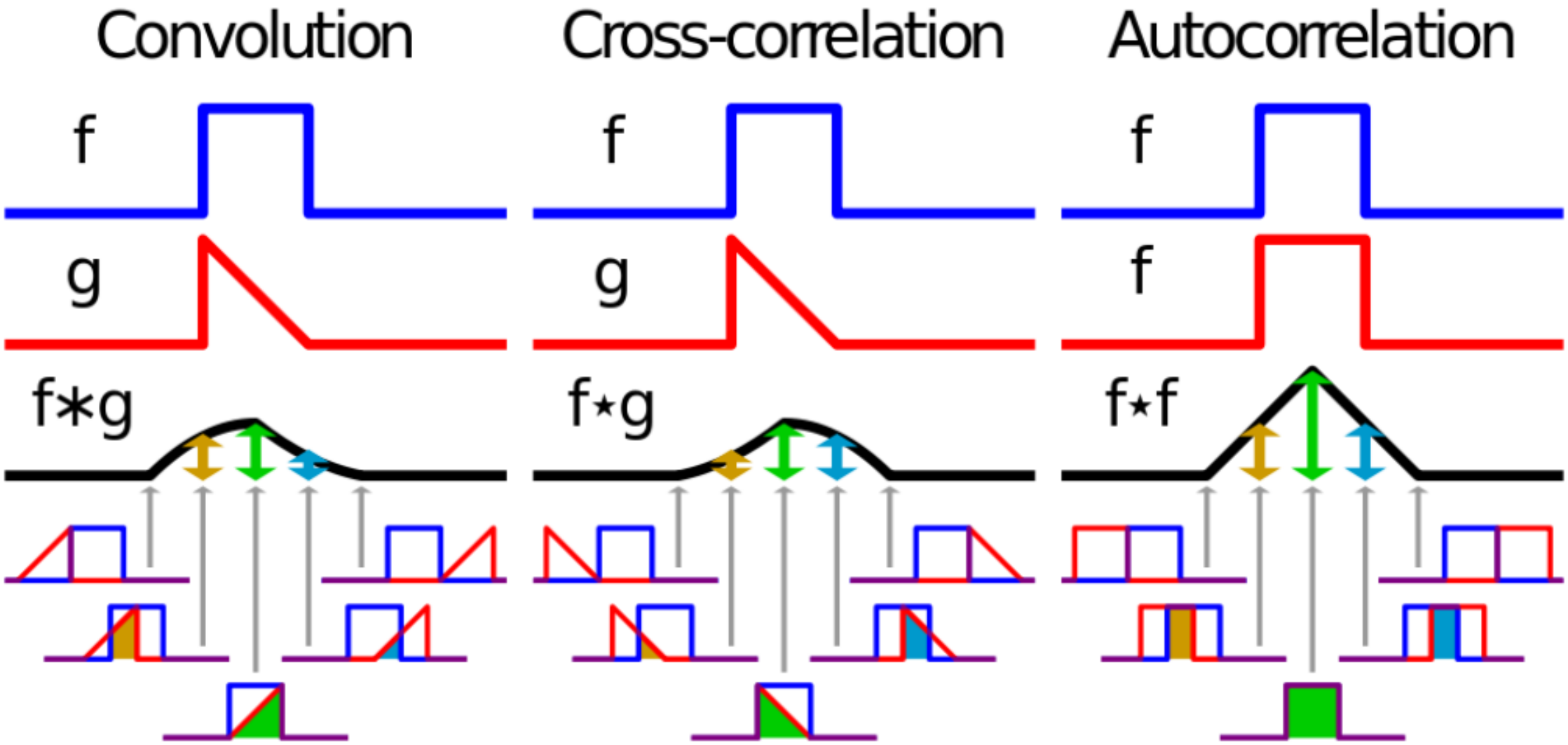


Grayscale Image Representation



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218



Convolution and Cross-Correlation will just flip the weight kernel for NNs!!

Image from http://commons.wikimedia.org/wiki/File:Comparison_convolution_correlation.svg

CNN recap

$$y_i = \frac{e^{x_i}}{\sum_k e^{x_k}}$$

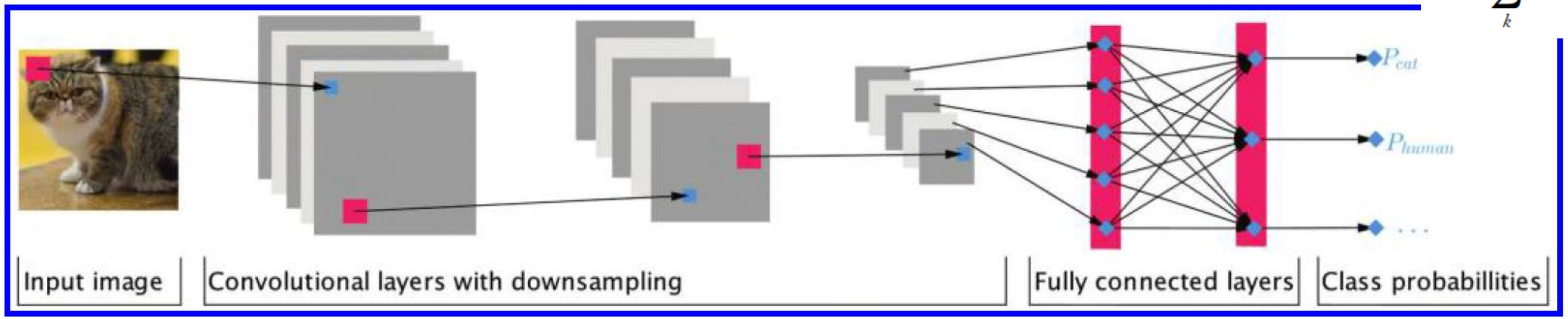


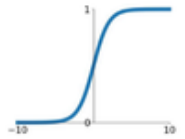
Figure 3. A CNN applied for image classification. The input is an image, and the output is a probability vector in which each element contains the estimated probability of the image containing an object of the given class.

$$X_{i+1}(x, y, j) = f\left(\sum_{k=1}^{C_{i-1}} X_i(x, y, k) * W_{i,j}(x, y, k)\right)$$

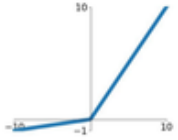
Activation function f can be linear or non-linear
C is the channel dimension
Zero padding is used to resize image to common dimension

Activation Functions

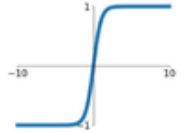
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$



Leaky ReLU
$\max(0.1x, x)$

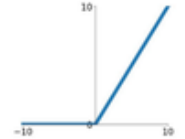


tanh
$\tanh(x)$



Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ReLU
$\max(0, x)$



ELU
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

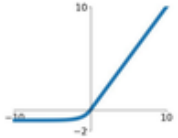


Image Classification demo

https://github.com/microsoft/computervision-recipes/blob/master/scenarios/classification/01_training_introduction.ipynb

- More CV scenarios can be found at
  - https://github.com/microsoft/computervision-recipes/tree/master/scenarios

References

- https://www.themtank.org/a-year-in-computer-vision
- https://www.analyticsvidhya.com/blog/2020/09/18-open-source-computer-vision-projects-beginners/
- https://www.kaggle.com/hrmello/intro-to-image-processing-colorspaces
- https://becominghuman.ai/real-computer-vision-for-mobile-and-embedded-part-3-postprocessing-96966f707fbd?gi=375e9364dfe8