Machine Learning: Intro

CPSC 501: Advanced Programming Techniques
Winter 2025

Jonathan Hudson, Ph.D
Assistant Professor (Teaching)
Department of Computer Science
University of Calgary

Friday, February 21, 2025

UNIVERSITY OF CALGARY

Artificial Intelligence

UNIVERSITY OF CALGARY

So, what is Artificial Intelligence (AI)

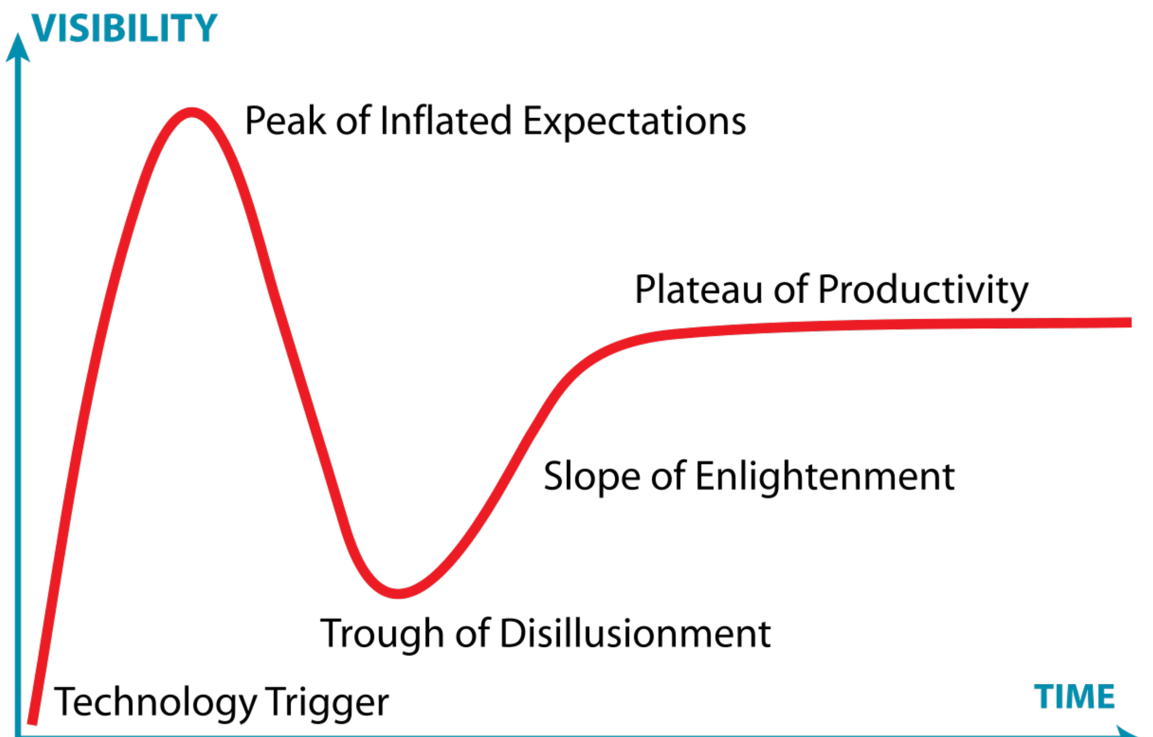
One Definition of AI:

AI deals with the development of systems either displaying a behavior humans associate with intelligence or solving a problem humans think only an intelligent being can solve.

- subjective definition
- changes over time!

AI: Moving Target

- **Lots of things you take for granted were called AI once.**
- Alexa/Siri/etc., google maps, biometrics, google search, automatic translation, natural language understanding, handwriting recognition, …
- We'll talk about the history of AI,
  - but in short, the **trough of disillusionment** in technology development is historic key



AI vs AGI

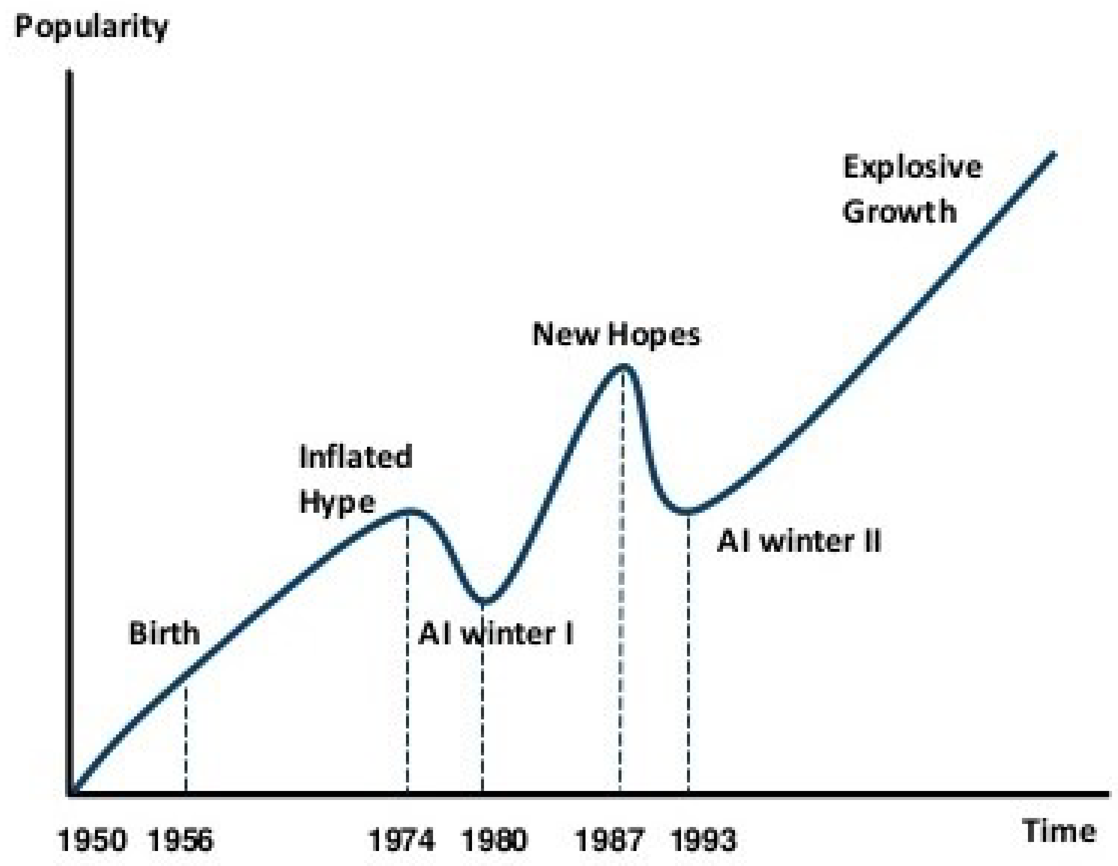
- **Artificial general intelligence** (**AGI**) is a **hypothetical** form of artificial intelligence (**AI**) capable of understanding, learning, and applying knowledge.
  - Like human cognitive abilities.
  - AGI is often the layman (non-professional) definition of AI

- Professional definition of AI includes **AGI** as well as the **currently existing specialized AI** systems.

- AGI, if it is created, would be able to **adapt** and **perform tasks not explicitly programmed** for, like human intelligence.

Artificial Intelligence (AI) in Computer Science

AI HAS A LONG HISTORY OF BEING "THE NEXT BIG THING"...



Timeline of AI Development

- **1950s-1960s**: First AI boom - the age of reasoning, prototype AI developed
- **1970s**: AI winter I
- **1980s-1990s**: Second AI boom: the age of Knowledge representation (appearance of expert systems capable of reproducing human decision-making)
- **1990s**: AI winter II
- **1997**: Deep Blue beats Gary Kasparov
- **2006**: University of Toronto develops Deep Learning
- **2011**: IBM's Watson won Jeopardy
- **2016**: Go software based on Deep Learning beats world's champions

UNIVERSITY OF CALGARY

What was that Turing Test at the start?

- **‘Does a machine exhibit behaviour which is indistinguishable from that of a human?’**
- Place a human interrogator is in a room with two computer terminals
  - One terminal is connected to another person
  - The other terminal is connected to an artificial intelligence
- The interrogator asks questions by typing them into the terminals
- If the interrogator cannot reliably determine which terminal connects to the human and which connects to the AI then the AI passes the test
  - The goal of the AI is not correct answers –it’s answers that resemble those of a person (Eugene Goostman!)
- Based on work by Alan Turing in the 1940s and 1950s
  - ‘The Imitation Game’, ‘Codebreaker’
  - **Father of Computer Science** (mathematician)

UNIVERSITY OF CALGARY

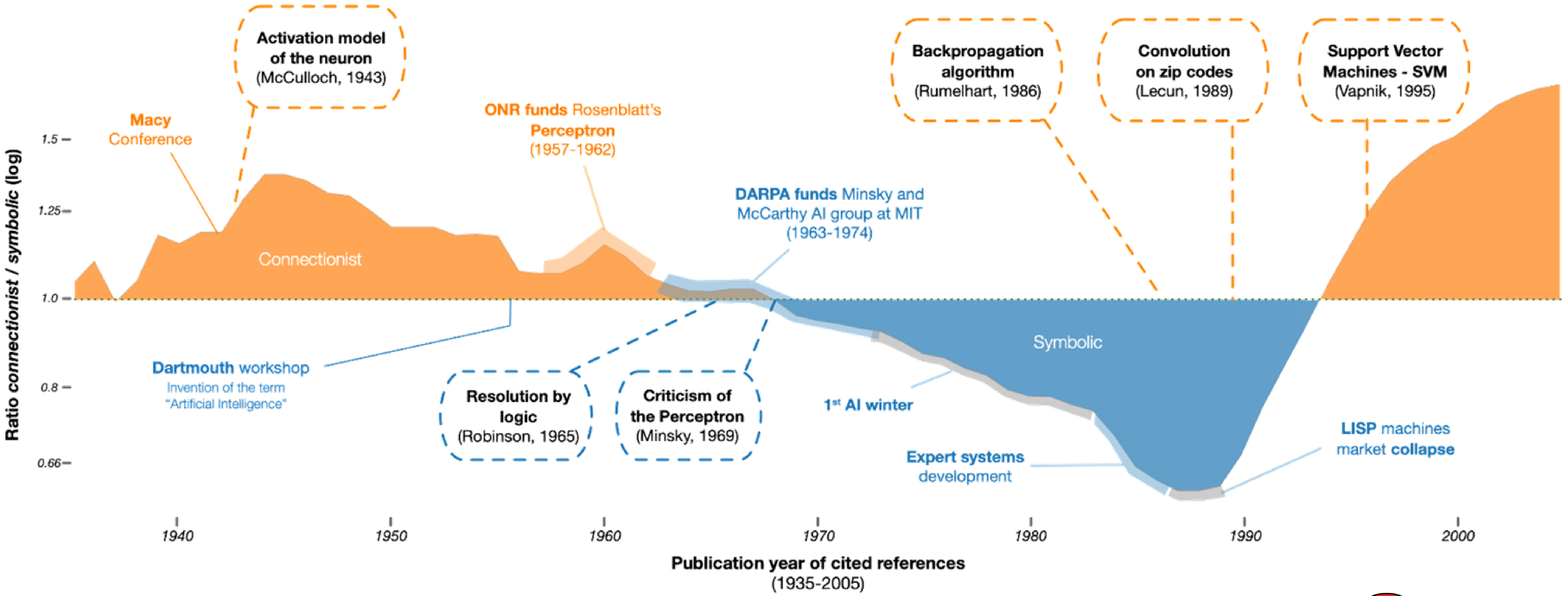
Connectionist vs Symbolic AI

- Connectionist → Make something simple, like a digital brain cell, connect them together with some simple network structures, intelligence falls out? Profit

- Symbolic AI → examine physical/animal/human world, find/create abstract structures and processes, apply to application problem that fits

- Differences:
  - rely on emergence of capability (conn.) or bake it in (symb)
  - flexible to a degree (conn.) or often can't think beyond what is baked in (symb)
  - great with learning patterns (conn.) or better at less common (symb.)

Connectionist vs Symbolic AI



Cardon, D., Cointet, J.-P. & Mazieres, A. (2018). Neurons spike back. The invention of inductive machines and the artificial intelligence controversy. *Réseaux*, 36(211), 173-220.

Current

After second AI Winter computer science made a lot of progress on taking the specialized system on the symbolic side and generalizing their ideas

On the other side our computing power reached the point that connectionist became powerful again

Current

- Popular areas: Deep learning (neural networks), machine learning (AI for finding patterns), digital assistants (connect symbolic systems), service architectures, internet of things, self-automation, etc.
- Things likely less close than you are sold: universal self-driving cars (visual identification has a lots of weaknesses), machine intelligence (?), true generic systems (we’ve generalized but there’s usually a lot more specificity under the hood than you think)
- Machine Learning: In many ways is used as a modern term for AI by many. Really just means machine changes/adds knowledge with less human influence.
  - in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Machine Learning

UNIVERSITY OF CALGARY

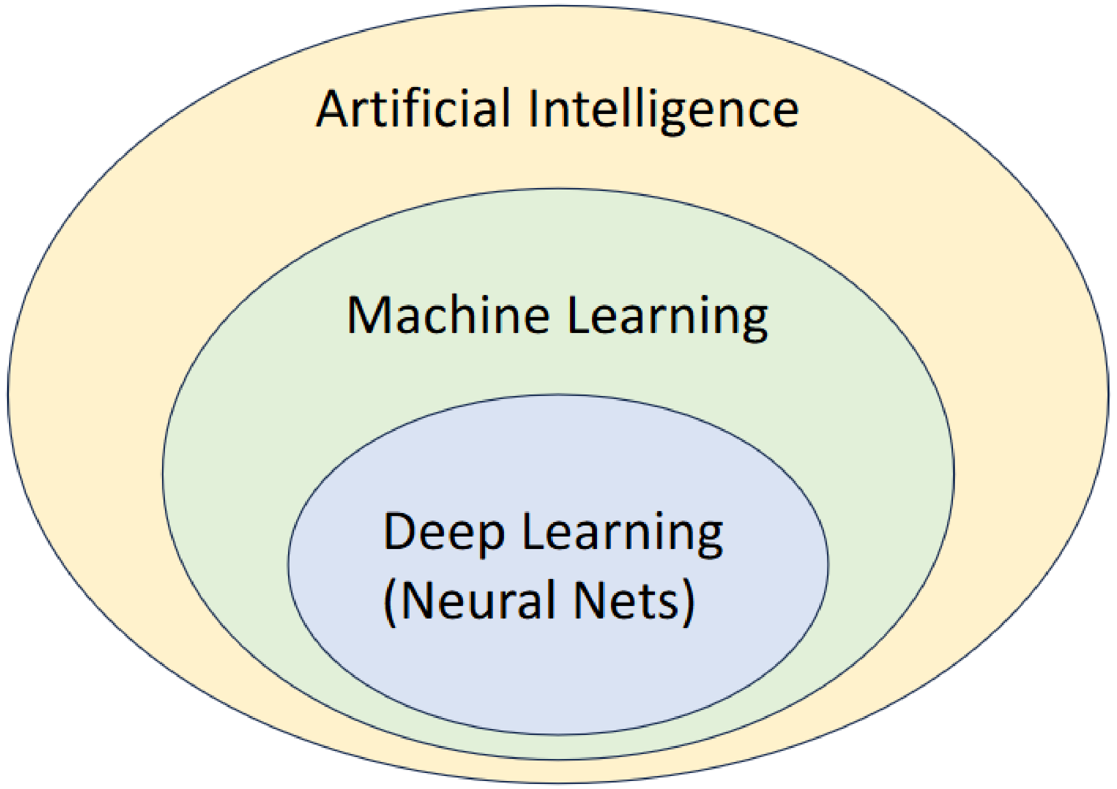
Machine Learning

- Around concepts around since Optical Character recognition research (1914!)
- Major prominence in 1990s with spam filters

- Machine learning is about **learning patterns from data** in order to make useful **predictions**.

- Good at seeing patterns in images, classifying things with many parts (text), filtering by pattern, chat systems, modeling complex things like, voice recognition, etc.

Relationships



Machine Learning and Humans

- Machine learning is built on the **premise of induction**, i.e. that data we have seen in the past can be used to predict future data.
- Machine learning is **not a magic solution** that solves all problems!
- Even the best ML approaches require a “smart human” to set them up.
  - A human needs to choose
    1. Approach & Assumptions
    2. Model (type of solution you would expect to see)
    3. Data source, encoding
    4. Training algorithm and Evaluation method

Categories

1. **Supervised Learning:** Machine is given guidance
   - Common examples are classifying something under labels, or finding parameters for a model that best matches data
   - e.g. (x, y), where x is an input value and y is the "correct" or "target" output
     - y is also sometimes called a label
2. **Unsupervised Learning:** Machine is left unguided
   - has to decide most everything (often minor guidance like # of things to find)
   - e.g. clustering (determine sub-groups that are 'similar')
3. **Reinforcement Learning:** Future actions responses guide what is elevated/added as learned knowledge, and what is reduced or lost

- Of course there are hybrid variants (semi-supervised)

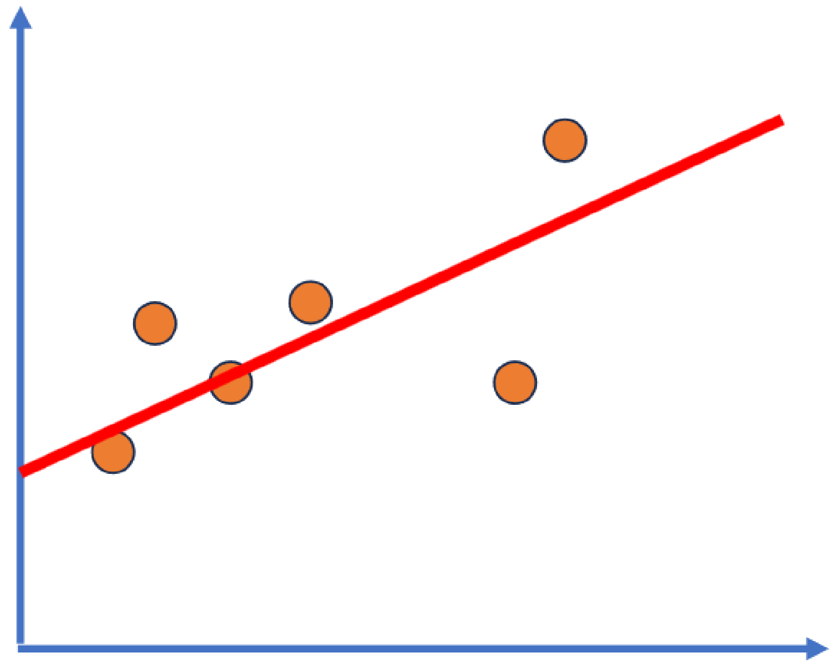
Design

1. Study the problem you are trying to solve (What?)
2. Choose a model class, hyperparameters (How?)
3. Prepare data (Do.)
4. Run learning algorithm to train the model (Do.)
5. Evaluate trained model (Did it work?)

UNIVERSITY OF CALGARY

Design: Model

- A model in machine learning is a hypothesis used for making predictions about data.
- By choosing a model class, you are selecting a space of possible solutions for your learning algorithm to explore.
- e.g. linear regression: $y = mx + b$
  - I have input variable x and output target y (I need learning what m/b match best)



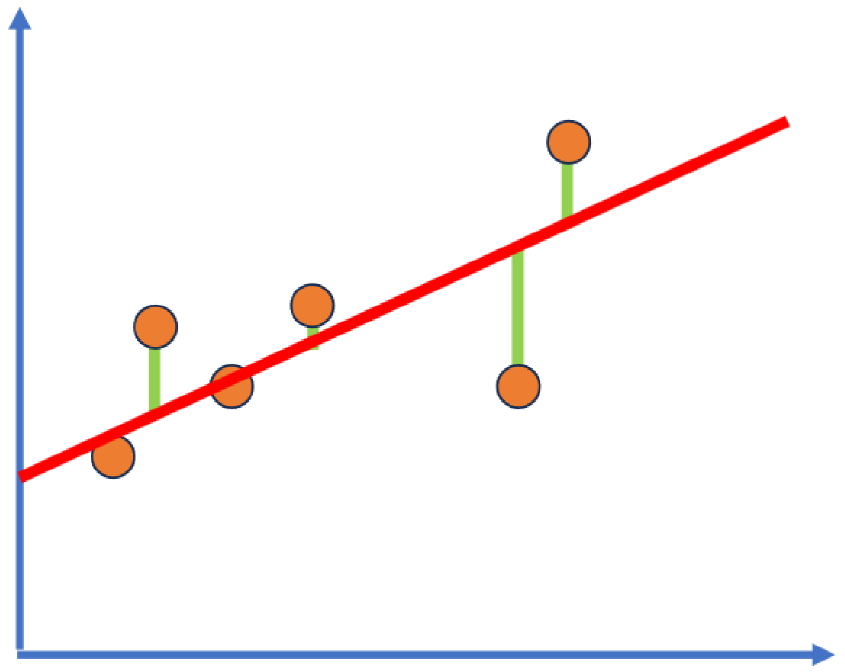
Loss Function

- How good our model is.
- Measures for each data point x we have, how close the model y was to the actual y

- Ex. Least Squared

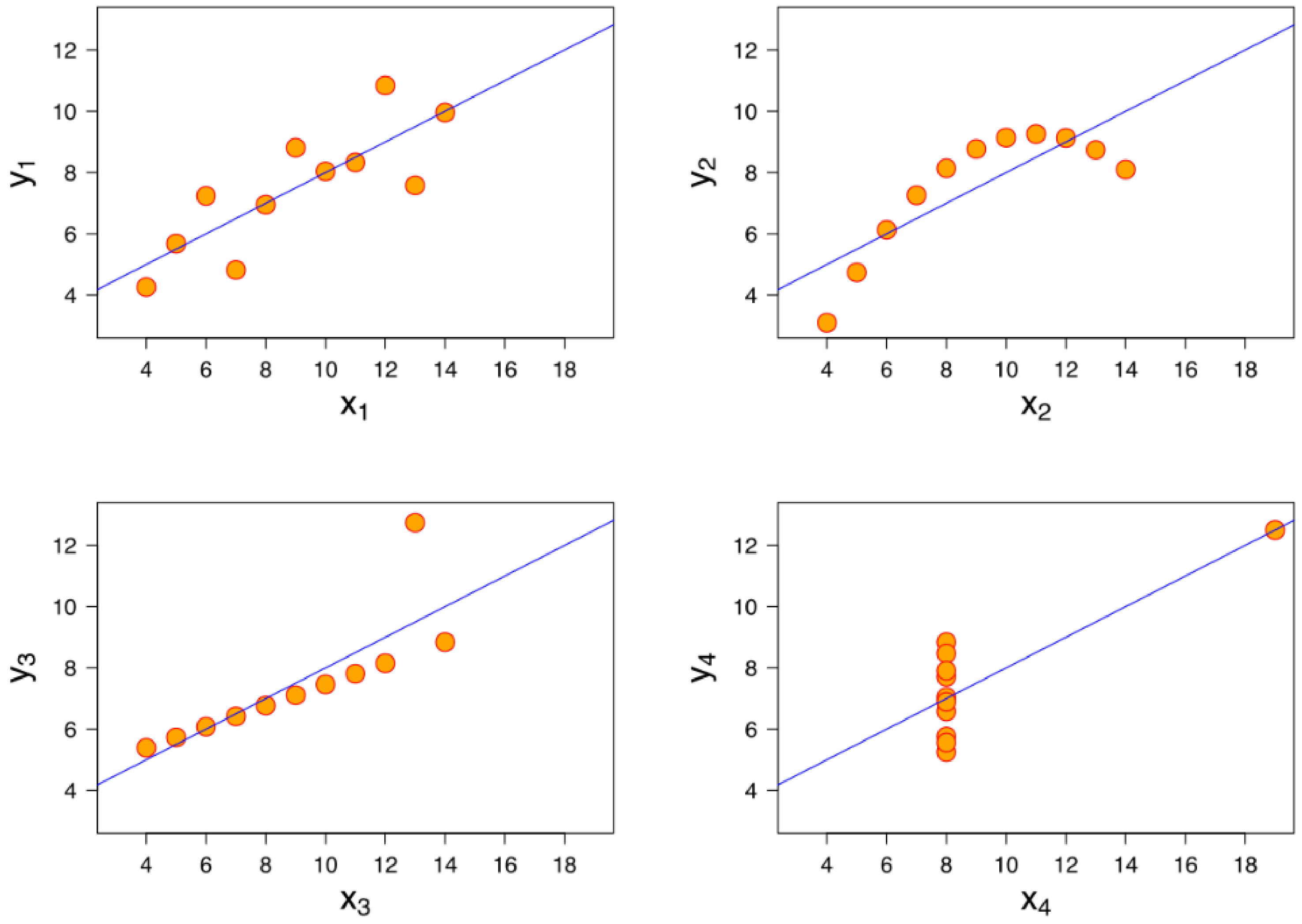
$$\min \sum (y_{actual} - y_{model})^2$$

$$\min \sum \left(y_{actual} - (mx_{input} + b)\right)^2$$



Challenges

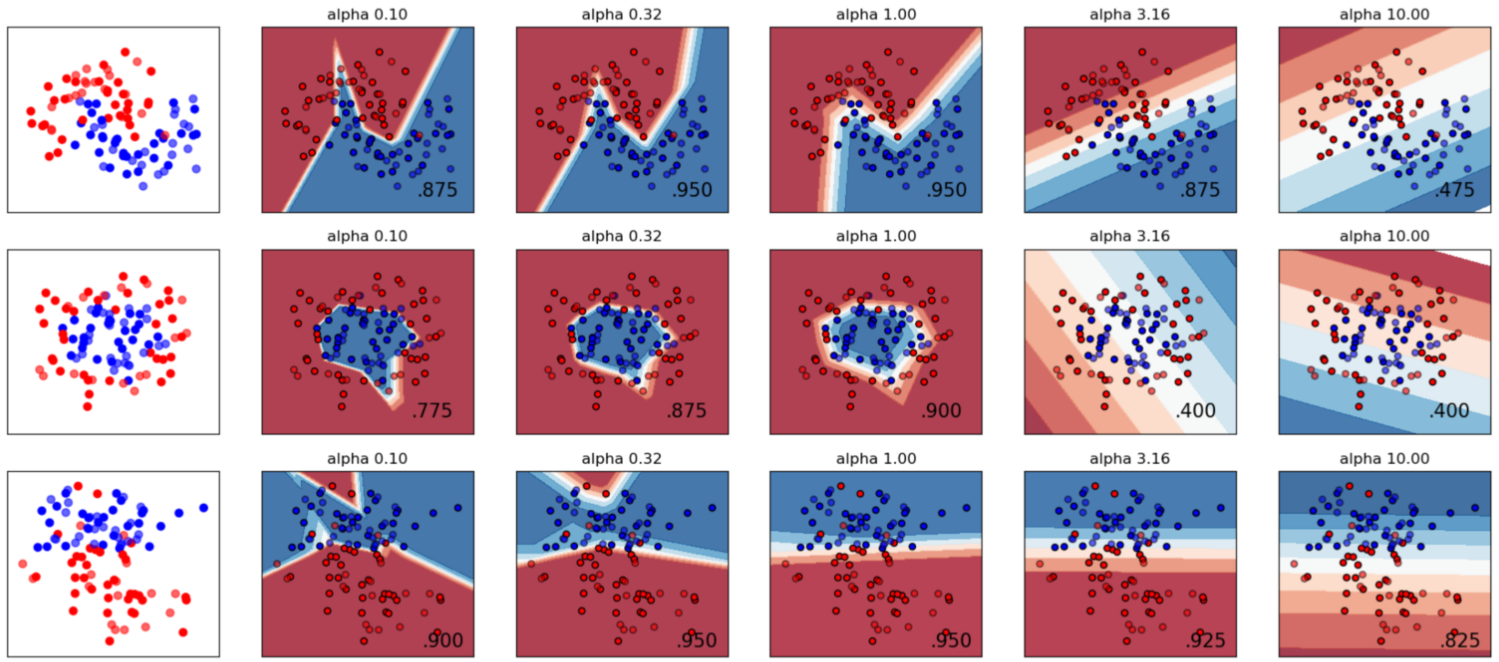
- The four data sets have identical statistics
  - Same mean, and variance
- You might produce same model m/b for all 4 inputs
- But we might disagree that that model will actually predict future y points on the line for new x values as we desire



https://en.wikipedia.org/wiki/Anscombe%27s_quartet

Fitting and Regularization

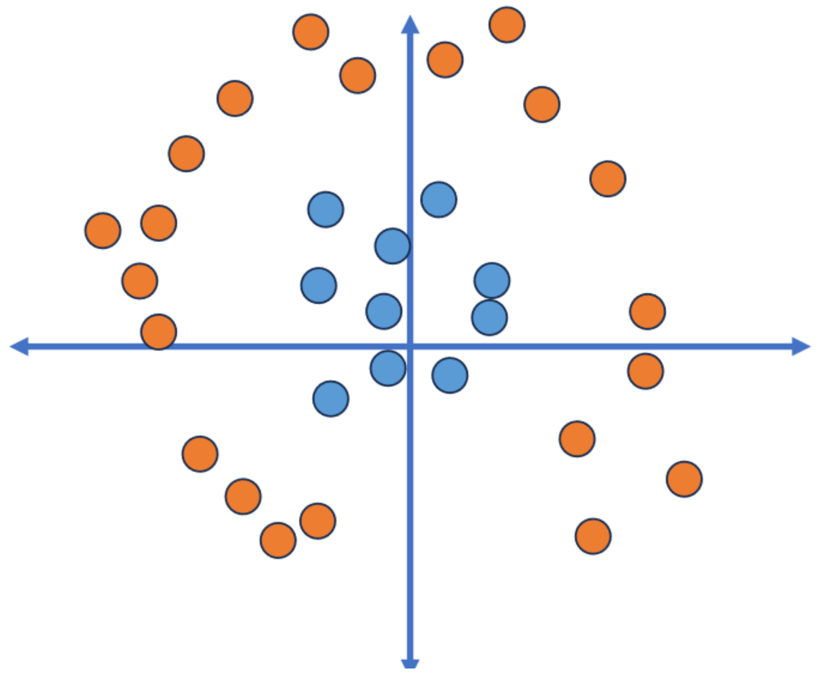
- Under-fitting -> we don't have enough training data so model can't find a pattern and is bad at unknown future input
- Over-fitting -> the model matches data exactly, but is bad with unknown future input
  - Regularization is a technique that allows error in the model while fitting so that it can be more 'wrong' but also more general for future prediction

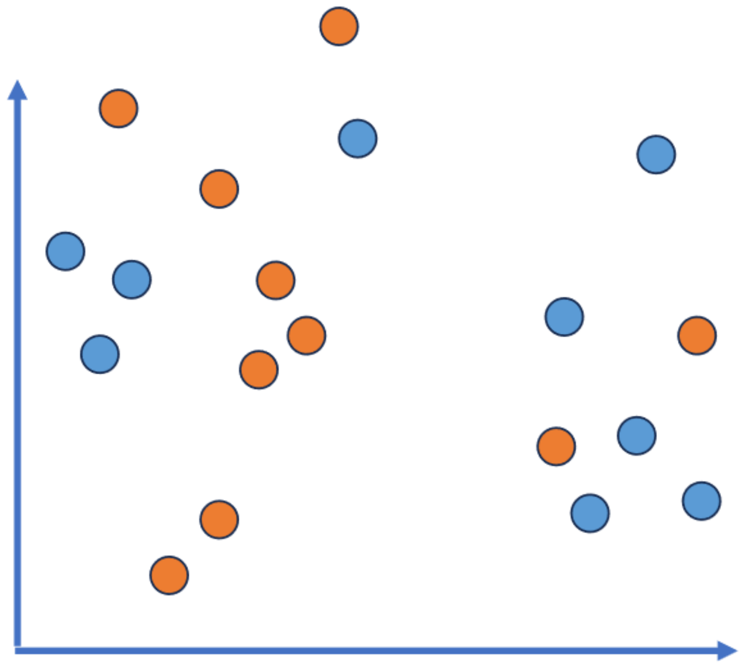


Neural networks often use drop-out for overfitting.
(Only use X% of your brain each time!)

Challenges

- Some data sets don’t have natural linear…
- For some like the top-right it could be pre-processed into linear space
  - using polar coordinate system
- Some like bottom-right may have no reasonable re-mapping
- For shapes like the first, we could also make our function more complex by making the model non-linear (Neural networks often chosen here)
  - http://playground.tensorflow.org

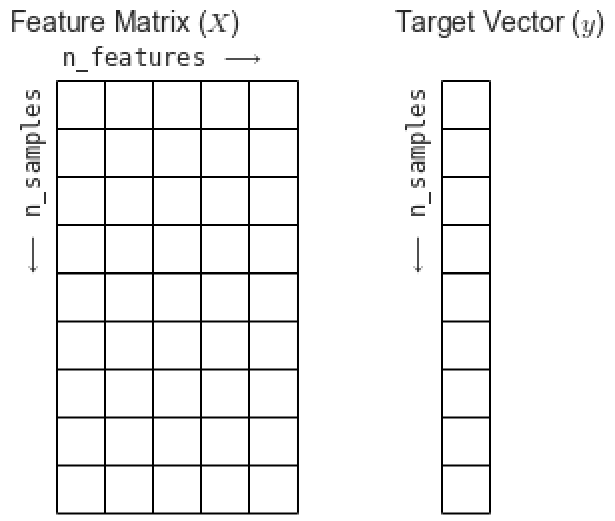




Scikit

scikit-learn

- scikit-learn is one of most common machine learning libraries
- Contains many, many boilerplate algorithms that take a couple lines of code of plug into data science operation
- Also a lot of syntax is mappable between different models to the point very different structural ones can often be swapped with a simple copy paste of one name
- Most models work on this structure
- Have array of data
  - split into input on left and output on right
- Apply model generally to find a relationship (left -> right)

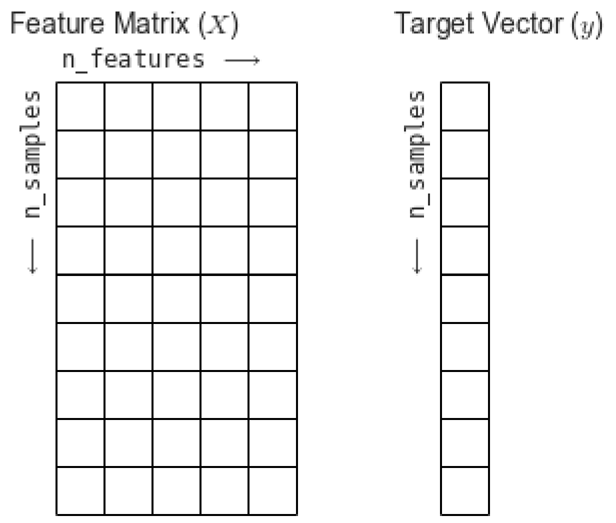


Design

- Consistency: common interface
- Inspection: expose parameters as public attributes
- Limited object hierarchy: only use standard arrays, numpy, pandas
- Composition: make use of sequence of standard algorithms, instead of making groupings
- Sensible defaults: have common defaults so only advanced usage needs setup

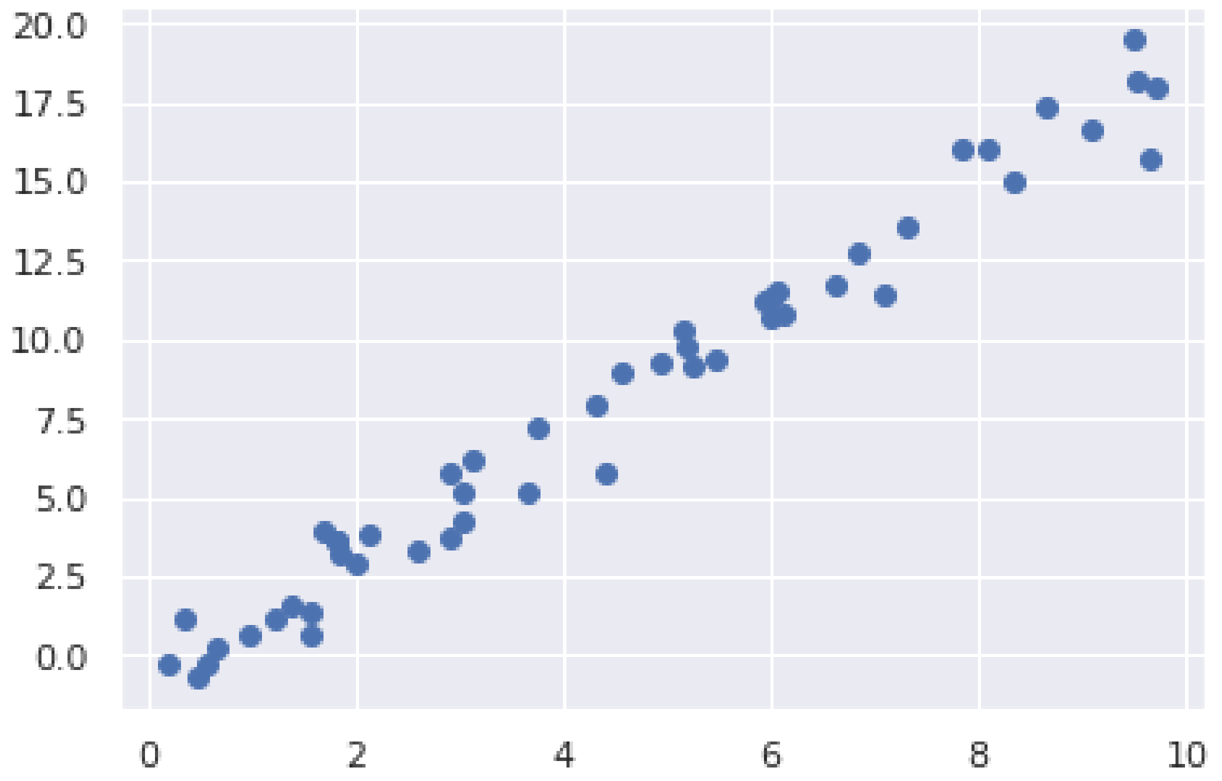
Process of Estimator API

- Choose model and import
- Choose parameters
- Arrange data into feature/target
- Call fit() on the estimator model
- Apply model to new data
  - predict for supervised learning
  - predict/transform for unsupervised learning



Example linear regression

```
import matplotlib.pyplot as plt
import numpy as np

rng = np.random.RandomState(42)
x = 10 * rng.rand(50)
y = 2 * x - 1 + rng.randn(50)
plt.scatter(x, y);
```

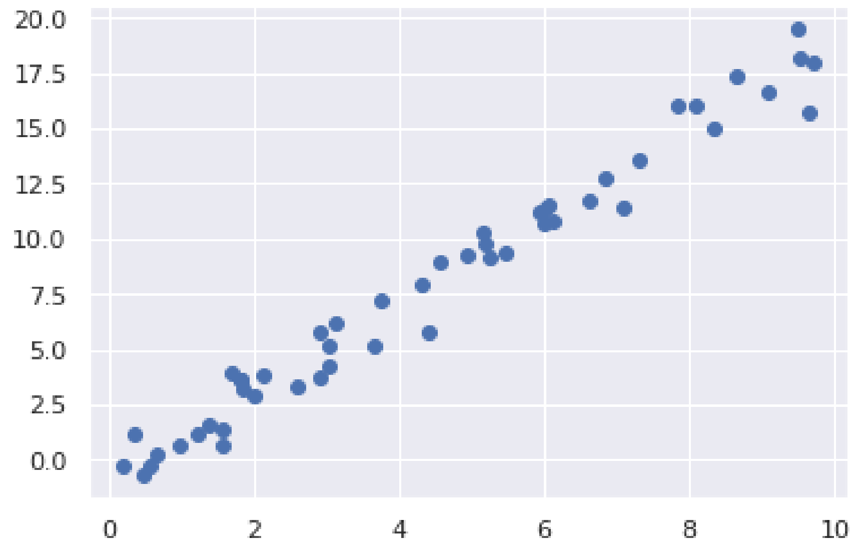


Example linear regression

Our model is a line of best fit

Used to predict future values

from sklearn.linear_model import LinearRegression



Example hyper parameters (not necessary)

1. fit for the offset (i.e., y-intercept)?
2. model to be normalized?
3. preprocess our features to add model flexibility?
4. degree of regularization?
5. how many model components?

Example linear regression

Our model is a line of best fit

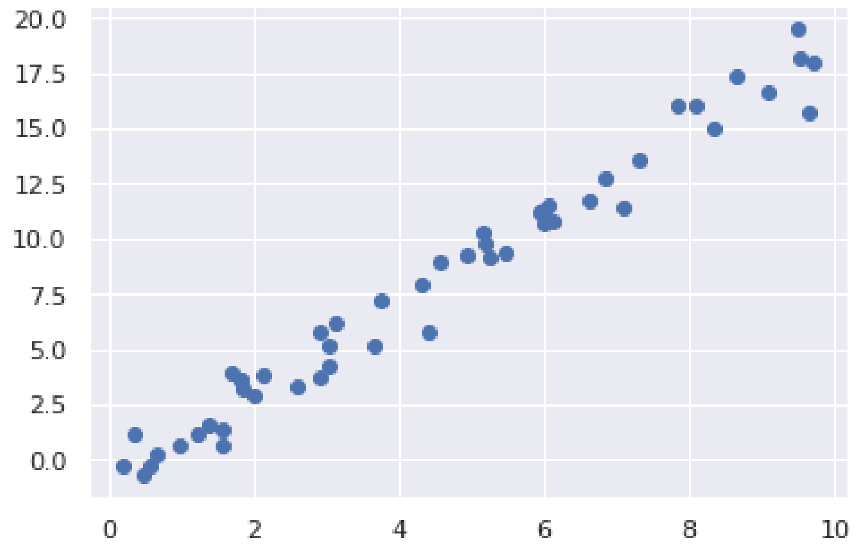
Used to predict future values

from sklearn.linear_model import LinearRegression

model = LinearRegression(fit_intercept=True)

or (defaults will be filled for us like or we can put them in)

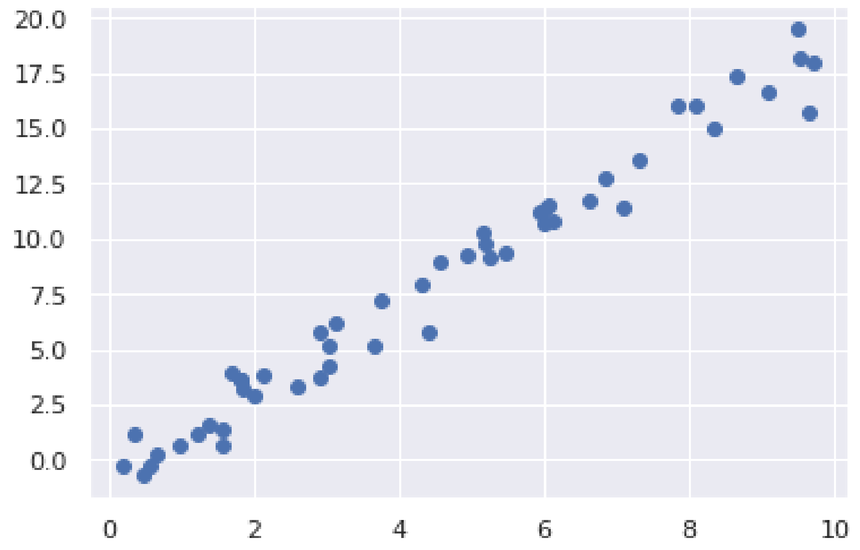
model = LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)



UNIVERSITY OF CALGARY

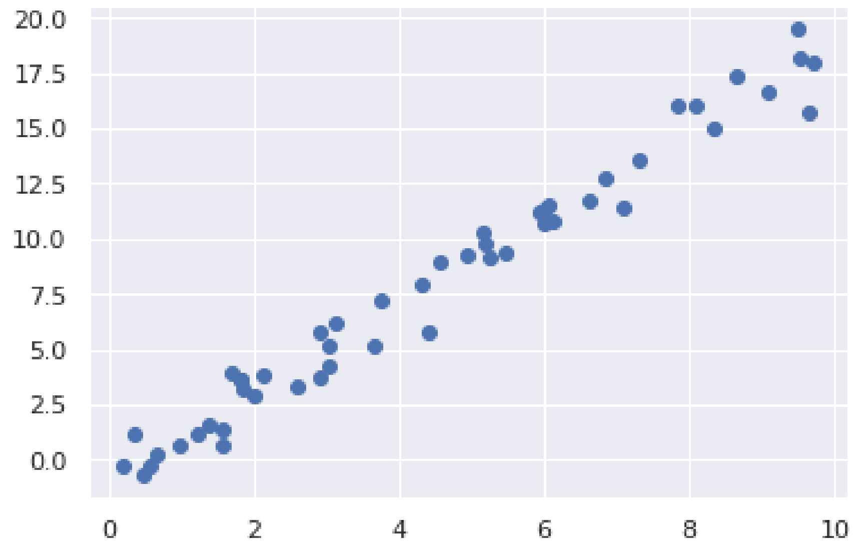
Example linear regression

```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)

#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
```



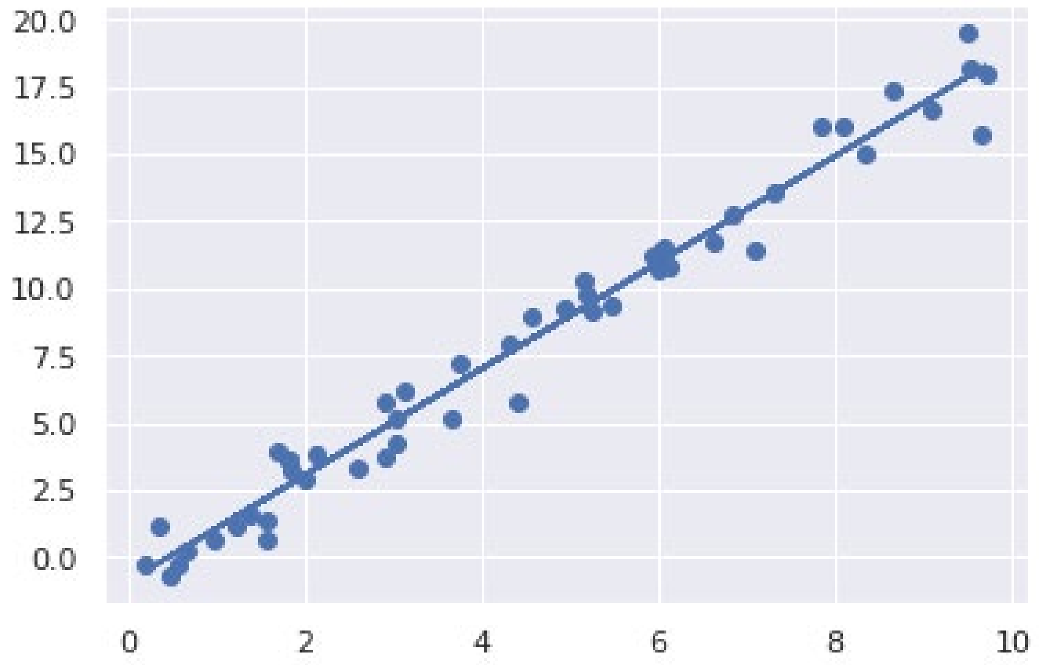
Example linear regression

```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)


#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
print(f"y = {float(model.intercept_):.2f} + x * {float(model.coef_[0]):.2f}")
```



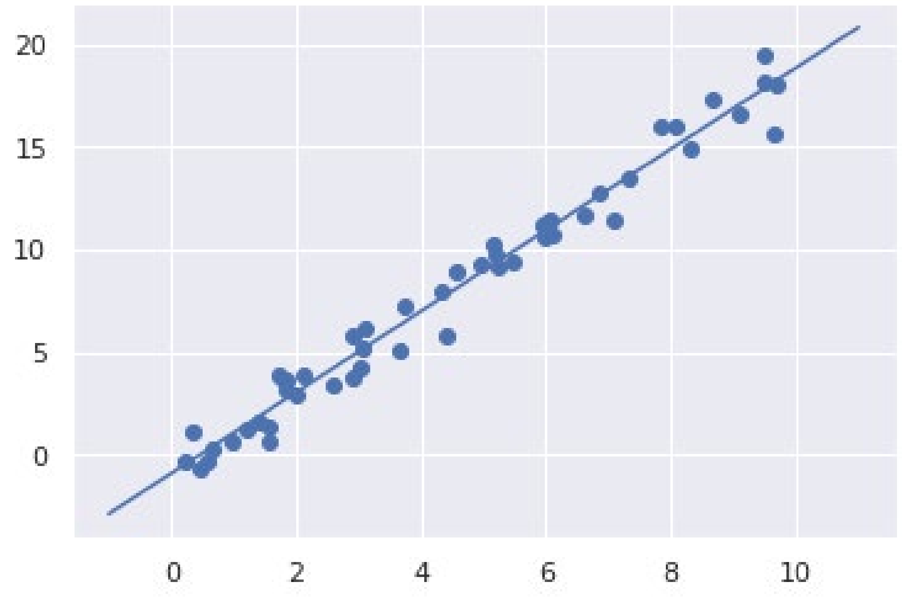
Example linear regression

```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)


#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
print(f"y = {float(model.intercept_):.2f} + x * {float(model.coef_[0]):.2f}")
plt.scatter(x, y);
plt.plot(x, float(model.intercept_)+float(model.coef_[0])*x )
```



Example linear regression

```
from sklearn.linear_model import LinearRegression
model = LinearRegression(fit_intercept=True)


#transform X into 2D array
X = x[:, np.newaxis]
#now fit
model.fit(X, y)
plt.scatter(x, y);
xfit = np.linspace(-1, 11) [:, np.newaxis]     #Make 2d array across range -1 to 11
yfit = model.predict(Xfit)                     #predict it
plt.plot(xfit, yfit);                          #plot it
```



Available Models

https://scikit-learn.org/stable/user_guide.html

1. Supervised learning

1.1. Linear Models

1.2. Linear and Quadratic Discriminant Analysis

1.3. Kernel ridge regression

1.4. Support Vector Machines

1.5. Stochastic Gradient Descent

1.6. Nearest Neighbors

1.7. Gaussian Processes

1.8. Cross decomposition

1.9. Naive Bayes

1.10. Decision Trees

1.11. Ensemble methods

1.12. Multiclass and multioutput algorithms

1.13. Feature selection

1.14. Semi-supervised learning

1.15. Isotonic regression

1.16. Probability calibration

1.17. Neural network models (supervised)

Available Models

https://scikit-learn.org/stable/user_guide.html

2. Unsupervised learning

2.1. Gaussian mixture models

2.2. Manifold learning

2.3. Clustering

2.4. Biclustering

2.5. Decomposing signals in components (matrix factorization problems)

2.6. Covariance estimation

2.7. Novelty and Outlier Detection

2.8. Density Estimation

2.9. Neural network models (unsupervised)

7. Dataset loading utilities (learning helpers)

7.1. Toy datasets

7.2. Real world datasets

7.3. Generated datasets

7.4. Loading other datasets

Onward to … neural networks

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~jwhudson/