

# What is Machine Learning (ML) ?

- prediction
- **informed** prediction
  - prediction is better than random guess
  - method: learn from existing data
  - goal: Generalization. Predicting on new, unseen data

# Where is ML used ?

Everywhere !!

- targeted advertising
  - Why does Facebook seem to know what I'm thinking ?
- spam detection
  - You are a winner !
- forecasting
  - Sales
  - Logistics
    - Where's my Uber ?
- anomaly detection
  - credit card fraud

## Uses in Finance


- model prices, risk
  - hedging
- Trading signals
- forecast sales
- predict defaults, pre-payments

# Not just numeric data !


- Images
  - Satellite:
    - Counting cars in a parking lot to forecast sales
    - How full is that oil tank ?
  - Did the CFO really mean what he said ? facial signals for confidence/evasiveness
- Text
  - Twitter sentiment as a signal ?
  - SEC filings
  - Derive industry groups by clustering press releases

# What you need to succeed ("Pre-requisites")


- An inquiring mind

-  Approach this topic like a Scientist  
Find a problem, gather data, formulate a hypothesis, test.  
Repeat.

- Technical skills

-  You are engineers !  
Solid programming skills.

- Some math/statistics

-  To be a successful data scientist, you need to understand the machinery.  
It is not enough to know an API.

- Self-motivation and energy
  - willingness to pick up tools/skills outside of lectures
  - You are engineers, nothing is too hard !

## Technical prerequisites

- Python
  - Object oriented (OO) Python
  - Numpy
  - Pandas
  - Matplotlib
- Some statistics (e.g., regression)
- Some math
  - comfort with Matrix/Vector notation

Fear not ! The last half of this lecture will be a whirlwind tour of the Python topics (except for OO)

# Textbooks

## Python Data Science Handbook

VanderPlas (<https://jakevdp.github.io/PythonDataScienceHandbook/>)

- Online !
- Solid foundation to acquire pre-requisites
  - Jupyter
  - Numpy
  - Pandas
  - Matplotlib
- Quick view of models



# Hands-On Machine Learning with Scikit-Learn and TensorFlow

Geron (<http://shop.oreilly.com/product/0636920052289.do>)

- Assumes you know the pre-requisites
- More detailed chapters on various models

# Notebooks

Both textbooks have code repositories on GitHub

- VanderPlas "book" is actually a notebook !

The real learning comes from active "doing" (play with notebooks) rather than passive "reading"

**Get the notebooks in the repos !**

- `mkdir ~/Notebooks; cd ~/Notebooks`
- `git clone https://github.com/jakevdp/PythonDataScienceHandbook.git`
- `git clone https://github.com/ageron/handson-ml.git`

# Accessing a repository on Github

If the `git` command is not available on your machine you can either

- install it via the command

```
conda install git
```

- Download the repo as a ZIP file. Visit the given URL
  - click on the button
  - Choose "Download ZIP"

## Get the lecture notebooks

- `git clone https://github.com/kenperry-public/ML_Fall_2019.git`
- periodically refresh: `git pull` from top level directory

# Machine Learning *using* Scikit-Learn (sklearn), TensorFlow (TF)

sklearn is a popular library for Machine Learning. We will be using it in the first part of the course. TensorFlow is another popular library that we will use in the second part of the course.

We are learning **Machine Learning**, not sklearn/TensorFlow !

Tools are a means, not an ends

- Goal is to understand ML independent of the toolset
- You can be an expert in sklearn/TensorFlow and still not understand ML

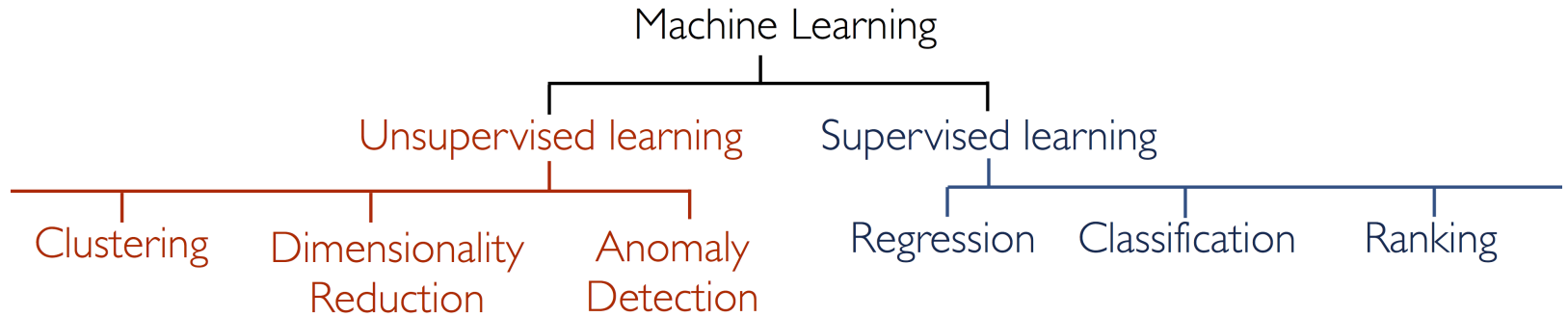
# Teaching method

**Iterative:** visit the problem many times, at increasing levels of focus

- top-down vs bottom-up
- Motivate: very high level view
  - know WHAT we are trying to achieve
- Understand: medium level view
- Deep understanding
  - math, statistics

Bonus: advice from a practioner

# **Landscape of ML**



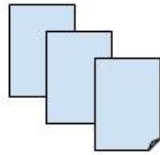
- Types of learning
- Types of targets
  - continous
  - discrete
- Types of features
  - numerical
  - categorical
  - text



# Challenges of ML

- You need data to train, often a lot of it
  - not always easy to get
    - supervised: needs to be labelled
  - quality issues
  - Is the training data representative of "the real world" for which you are designing ?
- Overfitting and Underfitting
  - Overfit: good training accuracy, poor generalization
  - Underfit: lost opportunity
- Engineering meaningful features is key
  - Data transformations
    - create features that aid prediction
    - art and science
  - Deep Learning may view feature engineering as part of the problem, not the solution !
- Testing and validation
  - an honest test uses held-out data
  - training data is a precious resource; painful to hold some out

# ML in one slide

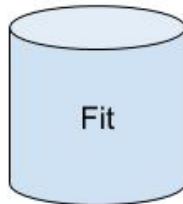


Observations:  
features  
features/labels

$x^1, x^2, \dots, x^N$

$y^1, y^2, \dots, y^N$

Scikit Learn



Create a predictor (estimator, hypothesis)  $y^* = h(x^*)$

- Predicts an outcome when presented with features  $x^*$
- Prediction based on the subset of  $x^1, x^2, \dots, x^N$
- That looks like  $x^*$

```
from sklearn import mod_class  
model = mod_class.Model()
```

```
model.fit(X,y)
```



Predict

```
y_star =  
model.predict(x_star)
```



In [1]: `print("Done")`

Done