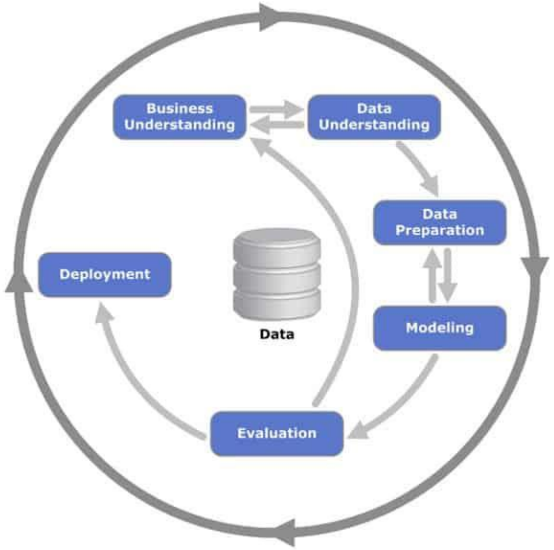
MSDS600 W3: Machine Learning

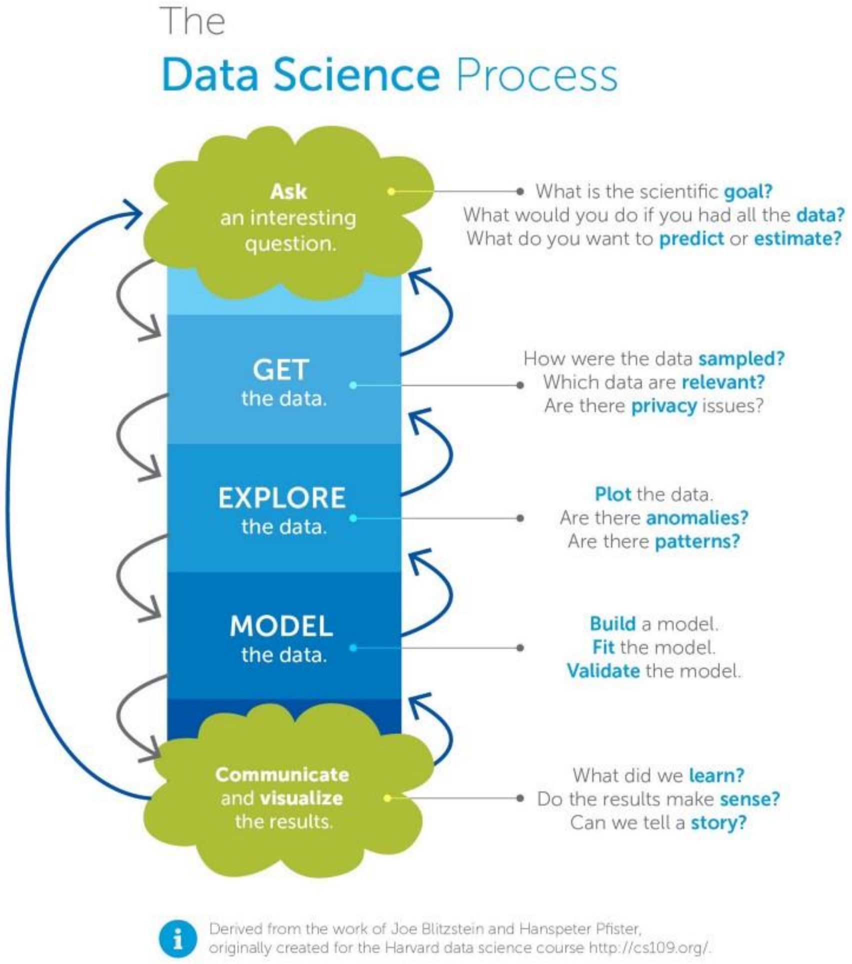
REGIS UNIVERSITY

The data science process

CRISP-DM, TDSP, others

Polls show 25-75% of time is spent cleaning and preparing data. Many data scientists report they even spend up to 90% of their time cleaning/preparing data. Some of this work is being moved into data engineering jobs.





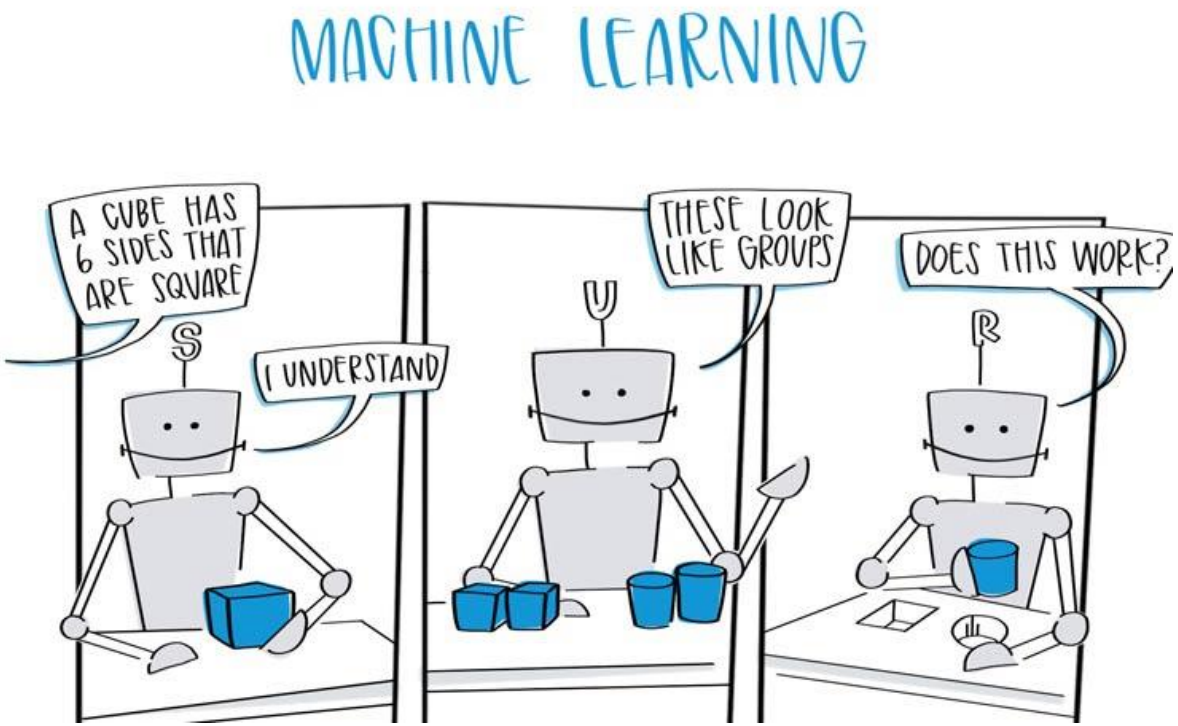
http://www.datascience-pm.com/

This week’s content

- Describe the different types of machine learning algorithms and how AI and ML relate.
- Implement machine learning algorithms using Python.
- Discuss ethical concerns of machine learning and AI.
- Discuss overfitting, underfitting, and the bias-variance tradeoff.

Machine learning

- Computer algorithms that learn patterns from data
- Three major types:
  - Supervised
  - Unsupervised
  - Reinforcement learning
- Semi-supervised combines unsupervised and supervised
- ML uses math and statistics to learn patterns



https://www.ceralytics.com/3-types-of-machine-learning/

The 2 main types of ML

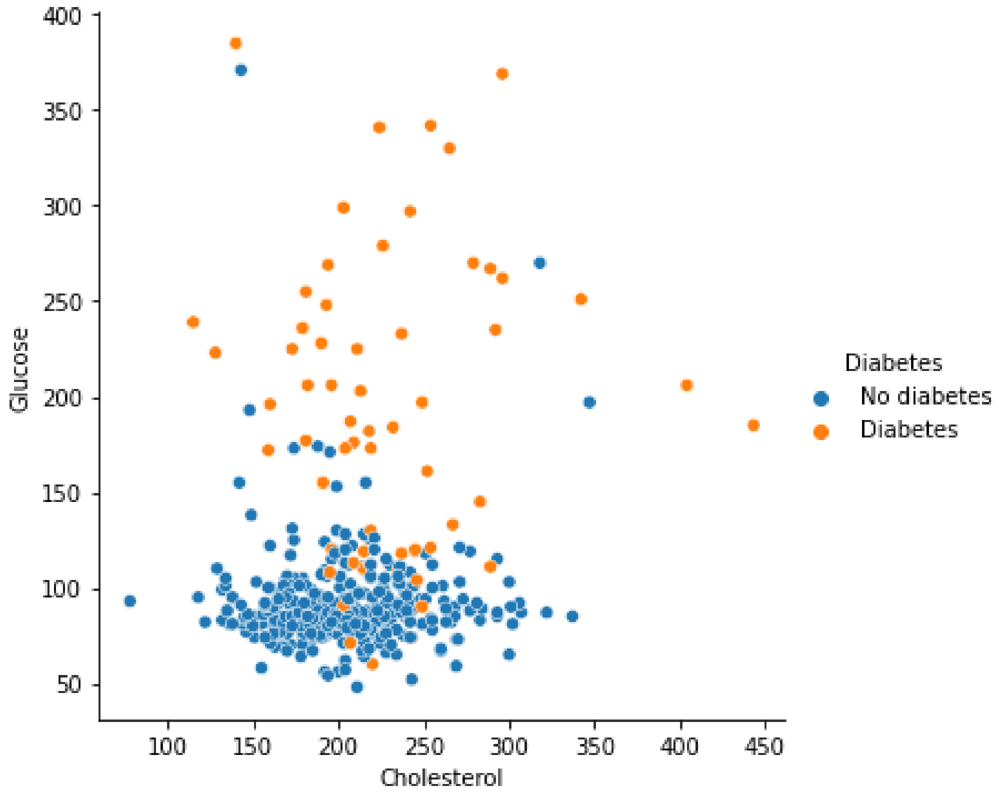
Supervised

- Features (inputs) and targets (outputs, labels)
- Regression: predict continuous numeric values (e.g. temperature, sales next month)
- Classification
    - Binary: 0 and 1, such as diabetes / no diabetes
    - Multi-class: any number of classes (e.g. dog breeds)
    - Multi-label: multi-class, but each datapoint can have multiple classes (e.g. topics for news articles)
- Logistic and linear regression
- Decision trees, random forests, boosting algorithms
- SVMs, neural networks, others

Unsupervised

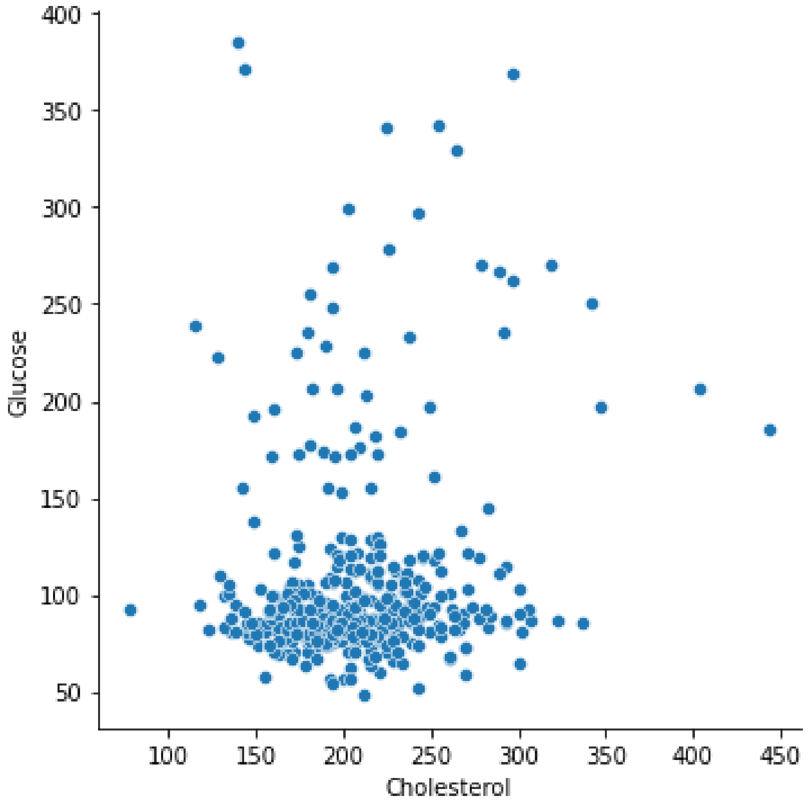
- No labels, only features
- Uses algorithms, math, and stats to group data
- Mostly clustering algorithms, but also includes topic modeling and dimensionality reduction techniques (e.g. PCA, SVD)
- Discovers how data naturally groups or clusters
- K-means clustering, DBSCAN, hierarchical clustering
- Topic modeling: LDA, LSA/LSI

Supervised



- Features/inputs and outputs/targets/labels

Unsupervised



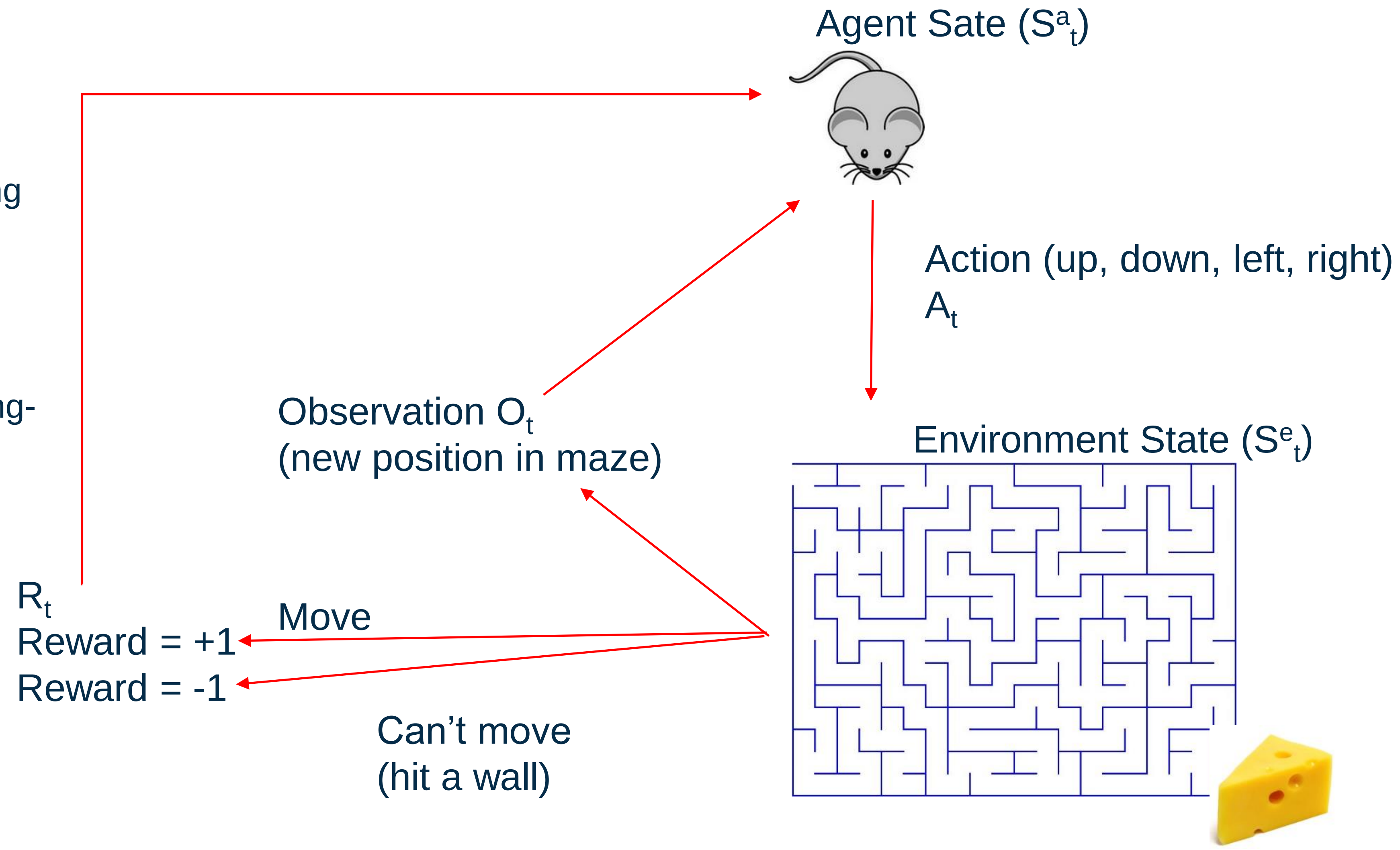
- No labels

Reinforcement

System learns from environment interaction

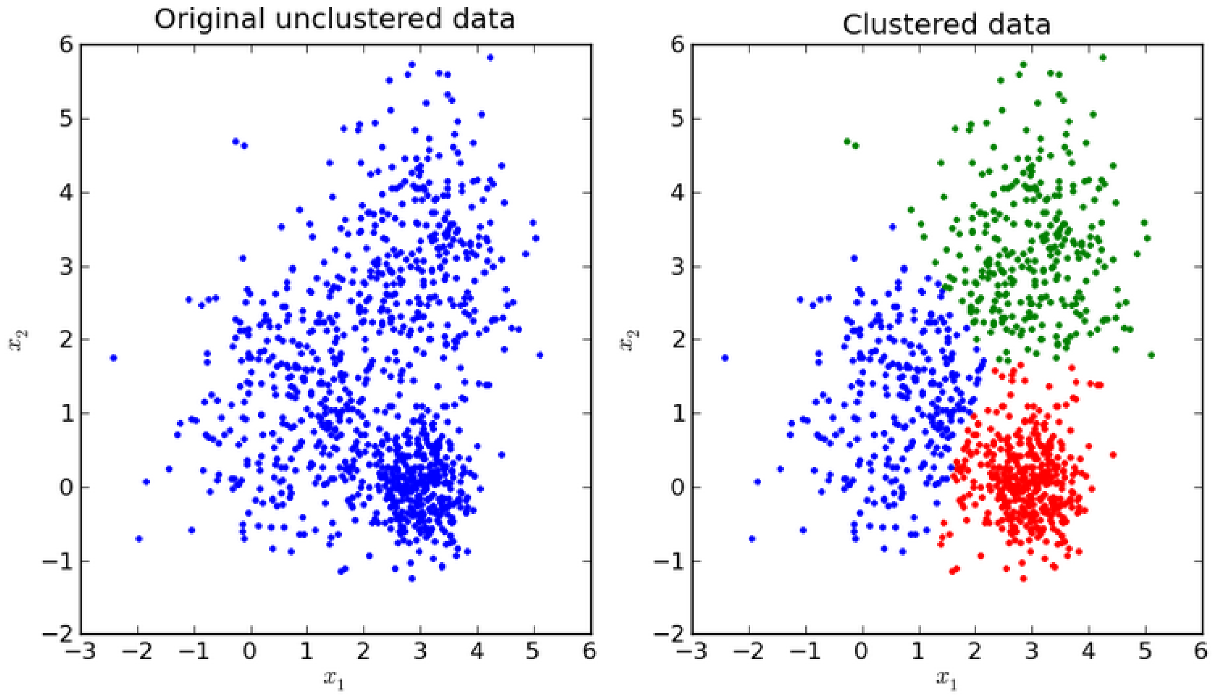
Improving elevator performance using reinforcement learning

https://papers.nips.cc/paper/1073-improving-elevator-performance-using-reinforcement-learning.pdf



Clustering and semi-supervised learning

- Groups data using math and stats
- Can use clustering with labels for some data generate more label of un-labeled data
- Many other methods are possible



http://trendsofcode.net/kmeans/

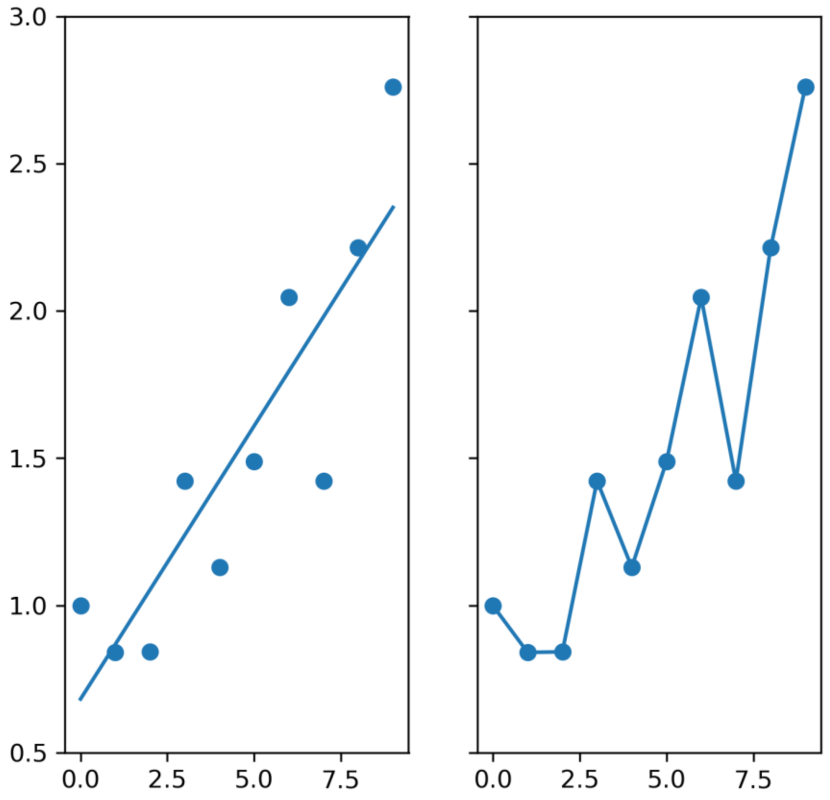
Regression and the bias-variance tradeoff

Linear fit

$y = mx + b$

Polynomial fit

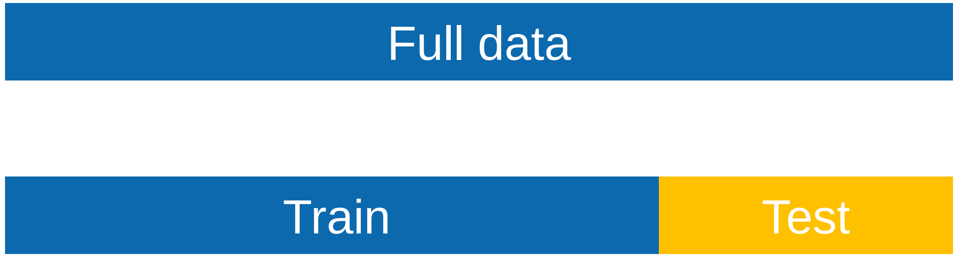
$y = m_1x + m_2x^2 + m_3x^3 + m_4x^4 + b$



- Overfitting – model is too complex and fits to noise in the data
  - High scores on training data, low scores on unseen data
  - Example: high-degree polynomial fit shown on the right
- Underfitting – model is not complex enough and has low accuracy
  - Low scores on seen and unseen data
- High bias – model has large errors on most datapoints
  - Ex: linear fit shown on the left
- High variance – model has high variance in prediction values across the sample space
  - Overfitting – low error on training data, big error on new data
- ML models have a bias-variance tradeoff – as we decrease one, we increase the other

Train/test splits

- We fit our model using training data
- Evaluate performance (e.g. accuracy) on both train and test data
- If train scores are high and test scores are much lower, it’s a sign of overfitting
- A more advanced version of this is cross-validation
  - Break up data into *n* train/test sets, run *n* fits and *n* evaluations of performance
- Ideally break up data randomly, unless there are timing issues (e.g. timeseries dataset)



Logistic regression for classification

- Predicts binary outcomes based on numeric input data
- P is probability of an outcome, like having diabetes (1) vs not having diabetes (0)
- log(p / (1-p) is the logit, or log-odds
  - Linearly related to coefficients (betas, B_1)
  - A unit change (+1) in a feature (x) changes the log-odds by a factor of *B*
  - Can be written with log (logarithm) and ln (natural logarithm)
- Uppercase B or beta and X are matrix notation lowercase are individual
  - X = [x_1, x_2, x_3…]
  - x = [1.1, 2.2, 1.0, 1.5…]

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$

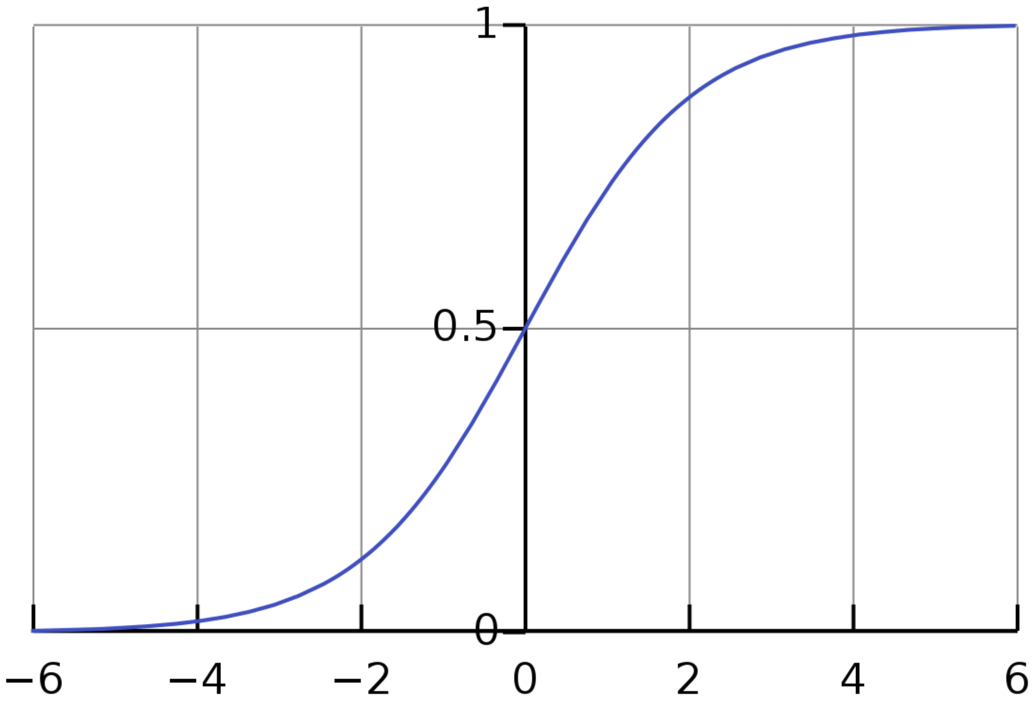
$$\frac{p(X)}{1-p(X)} = \exp(\beta_0 + \beta_1 X)$$

$$p = \frac{1}{1+e^{-(b_0+b_1x_1+b_2x_2+\cdots+b_px_p)}}$$

- https://medium.com/@mallrishabh52/logistic-regression-12fe50ffacb3
- https://www.saedsayad.com/logistic_regression.htm

Logistic regression

- Coefficients are fit with an iterative algorithm
  - Initialize coefficients
  - Make predictions with data
  - Calculate predictions, compare with ground truth
  - Change coefficients to bring predictions closer to truth
  - Iterate until prediction-truth difference stops changing much
- The logit with a single feature is the sigmoid curve
- Great for simple problems and as a first try with classification (especially binary classification)
- Assumes:
  - Linear relationships between log odds and features
  - Independent samples (each datapoint doesn’t effect another one)
  - A few others
- Can also use logistic regression for multi-class regression



https://en.wikipedia.org/wiki/Sigmoid_function

Machine learning in Python

Many Python packages available:

- scikit-learn (sklearn) and related scikit packages (e.g. scikit-optimize and many more)
- H2O
- pyspark (for big data)
- statsmodels (more classic stats methods)
- Neural network libraries (tensorflow/keras, pytorch, etc)
- autoML: pycaret, TPOT, etc
- More…

We will use sklearn this week for logistic regression on our binary diabetes/no-diabetes and churn/no-churn datasets.