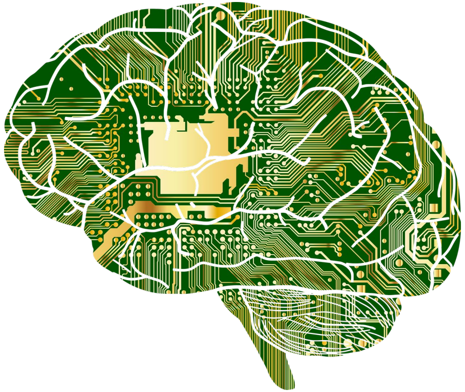
Machine Learning



What is Machine Learning

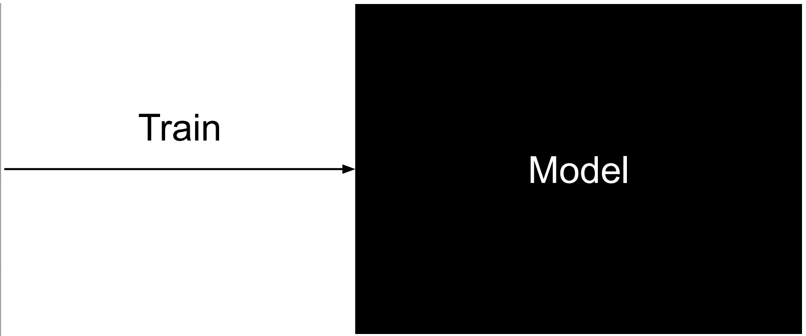
- Algorithms that enable machines to learn on data to provide predictions on unseen data
  - Supervised Learning
- Machine Learning is not Artificial Intelligence
  - It is one of the fields in AI
- Exploratory Analysis is still important!
  - Usual the first step before deep diving into ML algorithms

Machine Learning

- Efficient at learning patterns from data and linking them with labels
- Requires labeled data
    - For some problems it is hard/expensive or even impossible to get such data
- Often hard to understand the logic behind it
    - White Box model - model that is easy to understand and reason about
    - Black Box model - model that is hard to comprehend

Machine Learning

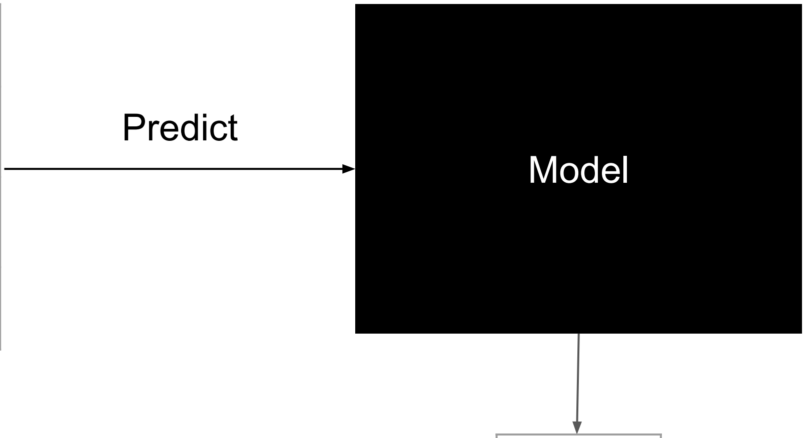
Pclass	Sex	Age	...	Survived
3	male	22	...	No
1	female	38	...	Yes
...	...	...	...	...



Supervised Learning

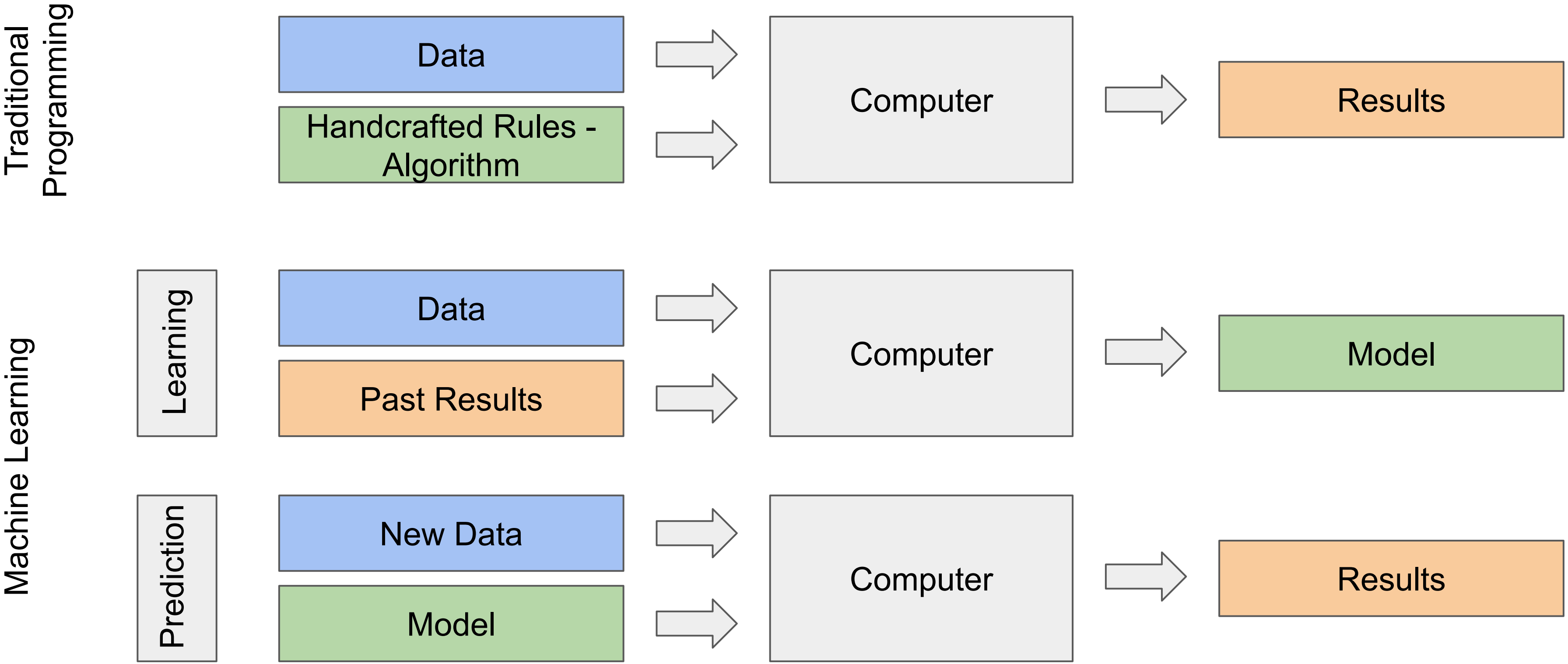
- The model can be applied to unseen data to predict label

Pclass	Sex	Age	...	Survived
1	female	50	...	?
2	male	30	...	?
...	...	...	...	...



No
No
...

Machine Learning vs. Programming



Python for ML

- Pandas
  - Data Analysis and Transformations
- Matplotlib
  - Data Visualization
- Numpy
  - Library for scientific computing on multi-dimensional matrices
- Scikit Learn
  - A Machine Learning Library

Machine Learning Process

- Define Problem
  - The hardest part
- Get data
  - Often connected with problem definition step
- **Prepare data**
  - Most tedious and time consuming, involves exploratory analysis
- **Run modeling**
  - Usually using a ML pipeline
- **Select best model**
  - The best is defined differently depending on the problem definition and business goals
- Iterate!

Data Preparation

Data Preparation

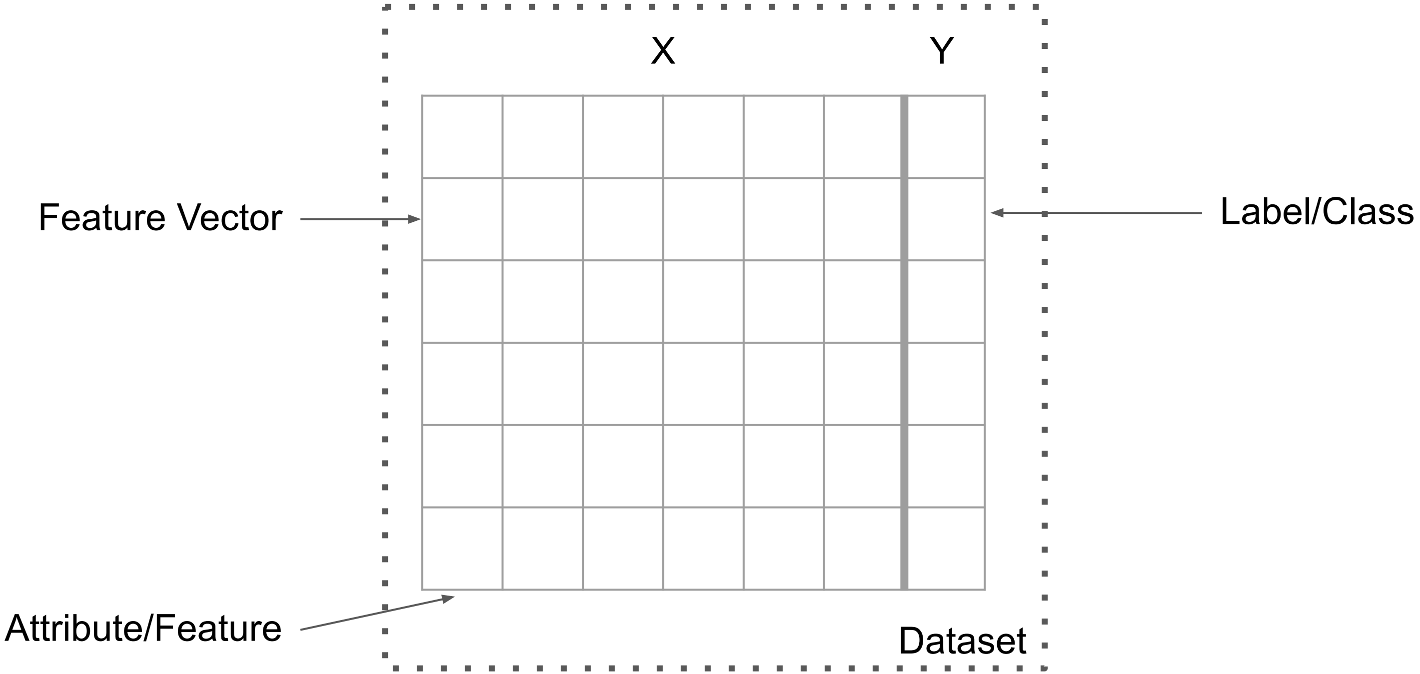
- Raw data can come in various formats
    - Database, Json, CSV, Excel, etc.
- Data for ML should be in tabular format (pandas/numpy)
- Each row is a feature vector
    - Each feature can be seen as a dimension
    - Each instance is a point in in n-dimensional space
- ML models build decision boundaries that separate points from different classes
- Curse of dimensionality
    - As dimensions increase so does volume
    - Data becomes sparse
    - Amount of data needed grows exponentially vs growth of dimensions

Terminology

- Feature/Attribute
  - A single variable (binary, nominal, numerical)
- Instance/Feature vector
  - One entity described by features
- Label/Class/Target Variable
  - An extra information that categorizes/classifies a given instance
- Dataset
  - Collection instances

Terminology

- Typically a list of feature vectors (matrix) is called X
- The corresponding vector of class labels is called Y



Datasets

- Typically there are two datasets in ML:
  - Training - Data used to train the model
  - Validation/Test - Data used to validate the model (not used in training)
- The training dataset has to represent real world
  - Model will only be trained for data that it sees,
  - Model will likely fail if unseen/real world data looks different than training data
- Validation data should not be related to training data
  - Keep samples dissimilar
  - Reflect how model will be used
  - E.g. if data is timed use older samples for training and newer for validation

Data Preprocessing - Feature Scaling

- Normalization
  - x -> (x - min)/(max - min)
  - Keeps data in 0-1 range
  - Sensitive to outliers
- Standardization
  - x -> (x - mean)/std_dev
  - Shifts mean to 0 and std_dev to 1
  - Values are not bounded
- It's good to scale the features so they have similar magnitude

Data Preprocessing - One Hot Encoding

- A lot of algorithms cannot handle categorical variables
- To this end categorical variables are encoded as binary representation
  - Create new features 1 for each of possible values
  - Fill with 1 or 0 depending whether original value correspond to new feature or not

color
red
green
blue

→

color_red	color_green	color_blue
1	0	0
0	1	0
0	0	1

Data Preprocessing - Discretization

- Converts numerical feature into categorical
  - Note: poor support for categorical features in sklearn
- Equal width
  - Each bin have equal width (sensitive to outliers, may produce empty bins)
- Equal Frequency
  - Each bin has same amount of instances (irregular shape)
- Supervised, e.g. Information Entropy Maximization
  - Selects bin automatically to maximize separation between classes

Data Preprocessing - Outliers removal

- Outliers often cause problems with ML algorithms
- Often is good to visualize data and do sanity check
  - Maybe data is bounded already
- Set threshold for outliers at top and/or bottom
  - n^{th} percentile (but check if data is not bounded, e.g. always greater than 0)
  - 4-5 standard deviations from mean (if data is normally distributed)
- Try to determine where do outliers come from
  - Human input error, malfunctioning sensors, etc.
- Handle outliers:
  - remove whole instance - but what to do when outlier appears in unseen data
  - replace value with max/min threshold - be sure to do such preprocessing to unseen data as well

Data Preprocessing - Missing Values

- Data may often miss some values (single or multiple features)
- Determine cause and if it could be biased
  - Hard to collect, lost, people will not provide information in survey, etc.
- Check how often values are missing
- Deal with missing values:
  - If a given feature has strong impact but is often missing potentially have model trained with and without a feature and use depending on feature availability
  - Replace missing value with avg or median value
  - Remove feature if a lot of data is missing (try with replacement first)
  - If categorical include a new category - N/A
  - Model missing values based on other features - increases complexity
    - E.g. Titanic dataset, if missing gender or age use mrs/miss/mr/master in name

Data Processing - Features Engineering

- Often done as part of data collection
  - Brainstorming what features may be related to the task and collecting them
- Some features may be engineered from features provided in dataset
  - E.g. divide distance traveled by trip length in hours to get avg speed
- Use domain specific knowledge, examples:
  - Body Mass Index = weight / height_in_meters^2
  - Use indices that describe physicochemical and biochemical properties of amino acids
  - Convex hull of units in a strategy game
  - Preprocessing of images so that individual objects are extracted (removes background noise)

Machine Learning Evaluation

Confusion Matrix

- Matrix that hold counts of instances depending on original label and the predicted label

	Predicted class is 1	Predicted class is 0
Original class is 1	True Positive (TP)	False Negative (FN)
Original class is 0	False Positive (FP)	True Negative (TN)

Accuracy

- The most known metric
- TP + TN / (TP + FP + TN + FN)
- or Correct_predictions / All_instances
- Is accuracy of 95% good?
  - Depends on base distribution, accuracy is not good measure for imbalanced datasets
  - Consider that 99% of instance may belong to one class, 95% is then less than naive predictor who always assigns more frequently occurring class
  - But if majority of instances that belong to the other class is within those 5% that may still be a great predictor!

Beyond Accuracy

% of a given class correctly identified:
True Positive Rate (Sensitivity): TP/(TP + FN) (also known as Recall)
True Negative Rate (Specificity): TN/(TN + FP)

% of predicted that are correct:
Positive Predictive Value: TP/(TP + FP) (also know as Precision)
Negative Predictive Value: TN/(TN + FN)

Matthews correlation coefficient (MCC):

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Probabilities

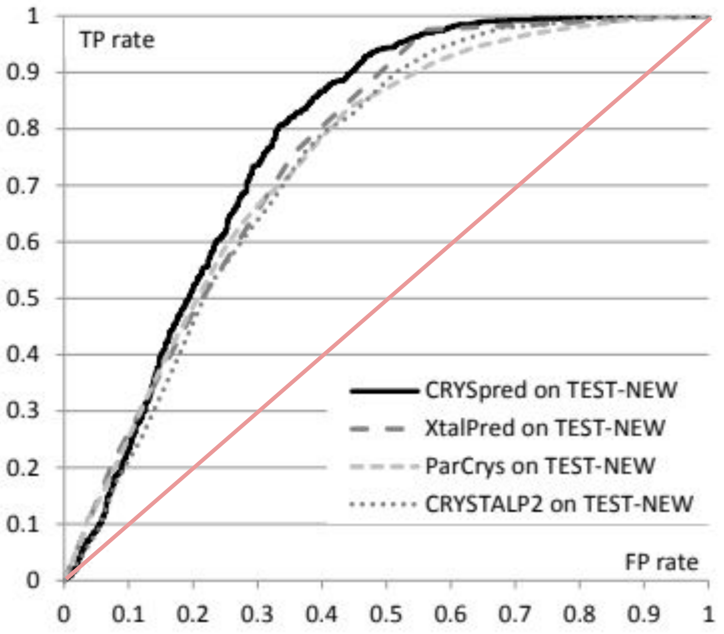
- Usually ML model provides probability of a class belonging to a given class
- That enables selecting cut off threshold for classification
  - Default is 50%
  - But we can change it to optimize for a given metric (Sensitivity, Specificity, MCC, etc.)
- What is more important depends on the problem
  - E.g. when performing medical pre screening checks you don’t want to miss any sick patients to send for more accurate (and expensive) tests

Probabilities

- Since we have actual labels we can validate predicted probabilities
- Cross entropy loss (log loss)
  - 0 is the perfect score
  - This score is hard to compare across different datasets so it’s typically used to build the model
- Point-biserial correlation coefficient

ROC

- Receiver operating characteristic
  - Puts threshold at each possible probability value
  - Calculate TPR and FPR (FP/N) for that threshold
  - Draw the resulting line
- ROC enables to visualize prediction properties for different thresholds
  - Point (0,1) - Perfect score
  - Line from (0,0) to (1,1) - random predictions
- AUC - Area Under Curve, common metric to summarize ROC



Beyond 2 labels...

- Accuracy (Correct_Predictions/All)
- Observed % (TC_K / C_K)
- Predicted % (TC_K / All_Predicted_As_C_K)
- R_K - generalized MCC for K classes
- Log-loss works for 2 or more classes
- ROC is hard to generalize for more than two classes

Machine Learning

Building model

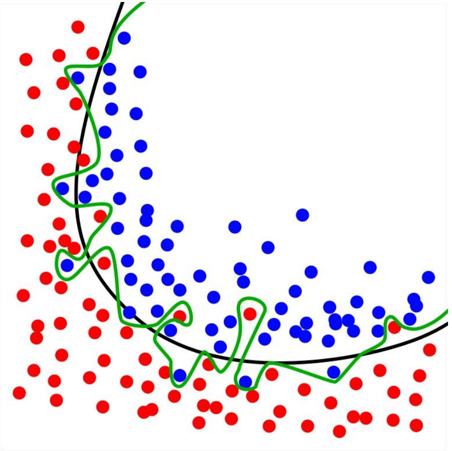
Hyperparameter search

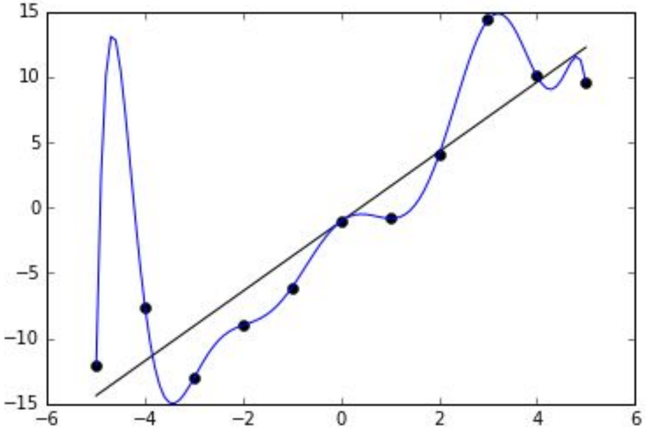
- Most algorithms have parameters that guide how the algorithms work
- They have to be tuned to specific problems/data
- Typically a grid search is performed where different sets of parameters are tried
    - Model with a given set of parameters is trained and evaluated, using training set
    - The set with highest score is selected
    - There are also more complex search approaches used
- But you have to be careful when you selecting hyperparameters to make sure model still works well on unseen data

(More on hyperparameters on Sunday)

Overfitting

- Model works perfectly on Training Data, but has lower performance on unseen data
- ML techniques aim at building generalized models
    - Regularization - Allow for error on training data
    - Cross Validation - Split training dataset and use splits to choose best parameters for a model
    - Validation Dataset - Use unseen data to evaluate model





K-fold Cross Validation

- We have one training set, but we can divide it into subsets/folds
- For example for 5-fold cross validation: divide data into 5 equal folds

1	2	3	4	5

- Use one fold as validation and combine remaining and use as training

Test	Training	Training	Training	Training

K-fold Cross Validation - cont'd

- Repeat to end up with 5 pairs of training-test data:

Test	Training	Training	Training	Training
Training	Test	Training	Training	Training
Training	Training	Test	Training	Training
Training	Training	Training	Test	Training
Training	Training	Training	Training	Test

- Thanks to Cross Validation set of parameters is evaluated over multiple datasets making the evaluation more robust
    - It's easy to overfit to one dataset, hard to overfit to multiple

K-fold Cross Validation

- The final model should be trained on the best set of parameters using whole dataset
  - You can also have k models and average predictions over them
- The reported classifier performance should be an average over all folds
  - Typically sum of all predictions vs avg of scores
- CV can also be used to evaluate model if there is not a lot of data
- How many folds to use?
  - Typical applications are 5 or 10, the larger data the smaller k
  - Jackknife - uses as many folds as there are instances
  - The more folds the longer it will take to train the model
- If you have huge dataset it may be OK not to use CV
  - Common in deep learning

Feature Selection

- Features have a big impact on a model performance
  - A lot of them will be redundant or irrelevant
- It's common to include feature selection as part of model training
- Find a subset of features that works best
  - There is 2^n combinations of features - impossible to check all of them in reasonable time
  - To this end heuristic search algorithms are used to explore features space to find a good subset (but not guaranteed to find the best)
  - Typically each subset is evaluated on training set
- Multiple benefits:
  - Simplified models
  - Shorter training time
  - Minimized chance of overfitting

(More details in Advanced ML class)