The Machine Learning Landscape

Supervised learning: Training sets include desired solutions, called labels
Regression: Given predictors, predict a target numeric value
Attribute: Data type (e.g. 'mileage')
Feature: Depends on context but usually an attribute plus its value (e.g. 20% chance of being spam)

Most important supervised learning algos covered:
- kNearest Neighbors
- Linear Regression
- Logistic Regression

- Support for Vector Machines (SVMs)
- Decision trees + random forests
- Neural networks (some nn's are unsupervised)

Unsupervised Learning: Training data is unlabeled
- Clustering
  - K-Means
  - DBSCAN
  - Heirarchial Cluster Analysis (HCA)
- Anomaly detection and novelty detection
  - One-class SVM
  - Isolation forest
- Visualization and dimensionality reduction
  - Principal component analysis (PCA)
  - Kernal PCA
  - Locally Linear Embedding (LLE)
  - t-Distributed Stochastic Neighbor Embedding (t-SNE)
- Association rule learning
  - Apriori
  - Eclat

Dimensionality reduction: Simplify data without losing too much information

Feature extraction: Combine several correlated features into one

Anomaly/Novelty detection: Detect for unique instances of data
- Training instances must be very clean

Association rule learning: Find data regularly clumped together
? Similair to feature extraction? ← No because these are features worthy of testing for rather than closely correlated data

Semi-Supervised Learning: Partially labeled data
- Cluster data then require values for each cluster
- Usually combinations of supervised and unsupervised learning

Reinforcement Learning:
Agent: Learning system (observes, interacts, and is rewarded)
Policy: Agent must develop a policy to get the most reward over time

Batch vs. Online Learning:
Batch: System must be trained using all available data (cannot be trained incrementally)
Offline Learning: When a strategy is used in production but cannot learn from its decisions
- New data must be clumped with the old and everything must be rerun. This can be optimized easily but is time consuming and computationally demanding
Online: Train system incrementally by feeding it data instances sequentially either individually or in mini-batches (small groups)
- Learning is fast and cheap
- Great for data recieved at a continuous flow (stock prices) that needs to react rapidly and autonomously
Learning rate: The speed a strategy adapts to new data and forgets old data
- Sometimes paired with anomoly detection algorithms to look for poor training data/performance

Instance vs Model-Based Learning: How well a system can make predictions on completely new data (generalized)
Instance-Based Learning: Hardcoded (labeled) data; finds new data by comparing/searching for similar values
Model-Based Learning: Create models using correlated attributes to classify data in the future using the model
? Can a mix of both be used?
Model selection: Deciding how to best model the type of relationship observed (eg. linear model)
Linear model: $f(x) = \theta_0 + \theta_1 x$

Model parameters: The parameters used in your model
- What param vals will result in the best performance?

Performance measure:
- Utility Function: Measures how good a model is
- Cost Function: Measures how bad a model is
  - Linear regression problems typically use a cost function that measures distance between a linear model's predictions and training examples (minimize the distance/cost)

Training: Applying performance measure to the model with the data to find optimal model parameters

Example 1.1: Training + Running a linear model using scikit-learn
- Chart life satisfaction/GDP per capita and predict future value
- Uses linear regression which is a model-based learning strategy. Could have also used K-nearest neighbors regression which would be instance-based learning

Things that go wrong:

Bad Data
- Not enough data: ML applications require absurd amounts of data in a logarithmic trend.
- Nonrepresentitive training data: Train with similar data to that which will be generalized
- Sampling bias: Data should not be filtered in any way as this could skew results
- Poor Quality data: Noisy data produces wild results and incomplete data must be treated with care
- Irrelevant Features: Training on irrelevant features should be avoided through feature selection where the most relevant information is used and the rest are removed or merged

- Overfitting training data: Building out a model that fits too closely to training data will not generalize well
- Regularization: Adding data, removing noise, and simplifying a model are all examples of regularizing data to help prevent overfitting

Bad Algorithm

Degrees of Freedom: Ways in which a model can change (free variables)

Hyperparameter: Parameter of learning algorithm
- Can modify regularization strictness

Underfitting Training Data: Occurs when model is too simple to understand the underlying structure of the data

Fix by...
- More powerful model
- Better features
- Reduce constraints on the model

Testing and Validating:
- Data should be split between training data and testing data

Overfitting: Occurs when training error is low and generalization is poor

Underfitting: Occurs when training error is high and generalization is poor

Hyperparameter Tuning and Model Selection:
- When adjusting hyper parameters or selecting a model, holdout data should be used to ensure the hyper parameters/model have not been overfitted themselves

Cross-Validation: Averaging out the results of a model tested against many smaller validation sets. This works well but takes a while

Data Mismatch: Occurs when two seemingly similair data sources do not mix well
- Can be tested for by holding out training data as test data
  - If tests poorly, overfit data
  - If tests well, data mismatch

No Free Lunch Theorem: Assumptions should (probably) be made about the data to reach optimal strategy