Machine Learning

- ML := how to acquire a model from data / experience
  - Learning parameters (eg: probabilities)
  - Learning structure (eg: BN graphs)
  - Learning hidden concepts (eg: clustering)
- Given training data ≜ example data points and their actual values
- Goal ≜ build model based on training data that outputs results in a similar fashion

Types of Learning Problems

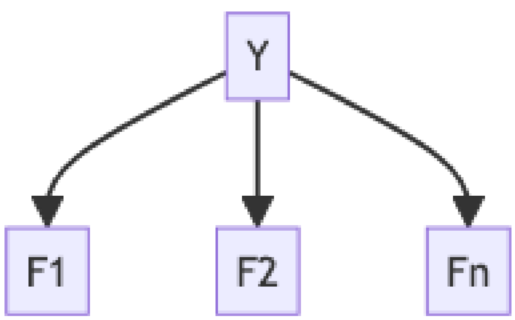
- Supervised learning ≜ correct answers for each training instance
  - Classification ≜ learning predictor with discrete outputs
  - Regression ≜ learning predictor with real-valued outputs
- Reinforcement learning ≜ reward sequence, no correct answers
- Unsupervised learning ≜ 'just make sense of the data'

Classification

- Dataset := each data point, x, is associated with some label (aka class), y
- Goal ≜ given inputs x, write alg to predict labels y
- Workflow:
  - x given
  - Extract features from input ≜ attributes of the input that characterize x and hopefully help with classification
  - Run some machine learning alg on the features
  - Output a predicted label y

Model-based Approach

- Build a model (eg: BN) where both the labels and features are random variables
- Instantiate any observed features
- Query for the distribution of the label conditioned on the features
- Challenges: how to structure BN / how to learn parameters



- Y := the label
- $\{F_1, F_2, ...F_n\}$:= the n features
- Prior := $\mathbb{P}(Y)$
- Generate a table per feature $\mathbb{P}(F_i|Y)$
  - Collectively called 'parameters'; denoted as θ
- Training ≜ use the training dataset to estimate the probability tables:
  1. Estimate $\mathbb{P}(Y)$ ≜ how often does each label occur
  2. Estimate $\mathbb{P}(F_i|Y)$ ≜ how does the label affect the feature

- Classification $\triangleq$ instantiate all features given the input features as evidence
  1. Derive features from sample
  2. Query for $\mathbb{P}(Y|F_1, F_2, ..., F_n) \triangleq$ probability of label, given all the input features
     - Use an inference alg (eg: variable elimination) to compute this

Naive Bayes

1. Extract features
2. Generate BN
3. Get joint probability of label and evidence for each label $\mathbb{P}(Y, f_1, ..., f_n)$
4. Sum to get probability of evidence
5. Normalize
6. Return the y with maximal probability
   - Can restrict to a confidence bound

- Good because returns exact confidence bound for every possible y
- Generally $\mathbb{P}(Y, F_1, ...F_n) = \mathbb{P}(Y) \prod_i \mathbb{P}(F_i|Y)$
  - Works because features are assumed to be independent
  - $n \times |F| \times |Y|$ parameters (linear in n)

Bag-of-words Model

- Each feature is identically distributed
- All features share the same conditional probs $\mathbb{P}(W|Y)$

Parameter Estimation

- Estimating the distribution of a random variable
- Elicitation $\triangleq$ ask a human
- Empirically $\triangleq$ use training data

Maximum Likelihood

- Chose the θ value that maximizes the probability of the observation
  - Find θ that maximizes $\mathbb{P}(\text{observation}|\theta)$
- eg:
  - Given $\mathbb{P}(\text{red}|\theta) = \theta$
  - $\mathbb{P}(\text{2 red and 1 blue}|\theta \text{ of beans are red}) = \mathbb{P}(\text{red}|\theta)\mathbb{P}(\text{red}|\theta)\mathbb{P}(\text{blue}|\theta) = \theta^2(1-\theta)$
  - Want to compute $\theta^* = \arg\max_\theta \theta^2(1-\theta)$
    - Find θ such that $\frac{d}{d\theta}\theta^2(1-\theta) = 0$
- Note: maximizing the likelihood is the same as maximizing the log-likelihood
  - $\arg\max_\theta f(\theta) = \arg\max_\theta \ln f(\theta)$
  - Makes derivatives much easier because it turns products into sums
- For naive BNs:
  - $\mathbb{P}(y) = \frac{\text{\# occurances of class } y}{\text{total \# of observations}}$
  - $\mathbb{P}(f|y) = \frac{\text{\# of occurences of feature } f \text{ and class } y}{\text{total \# of occurences of class } y}$