ML Model Fitment

Concepts

- Training set $\triangleq$ the data used to train the model
- Held-out set $\triangleq$ the data used to train the hyperparameters
- Test set $\triangleq$ the data set used to evaluate the accuracy of the model
- Features $\triangleq$ attribute-value pairs which characterize each input

Overfitting

- Fitting the training data very closely, but not generalizing well on test data
- Relative frequency parameters will overfit the training data
  - Doesn't account for values not in training set
  - Can't go around giving unseen events zero probability
- Solution: need to smooth or regularize the estimates

Smoothing

- Laplace estimate $\triangleq$ pretend you saw every outcome once more than you actually did
  - $c(x) \coloneqq |x|$
  - $k \coloneqq$ strength of the prior (hyperparameter)
  - $P_{LAP,k}(x) = \frac{c(x)+k}{\sum_x [c(x)+1]} = \frac{c(x)+k}{N+|X|}$
- Learn parameters from training data
- Tune hyperparameters on different data
  - Value of k on the 'held-out' data (extracted from the training data) that maximizes accuracy

Underfitting

- Fits the training set poorly

Baselines

- Very simple 'straw man' procedure
- Help determine how hard the task is
- Help know what a 'good' accuracy is

Weak Baseline

- Most frequent label classifier
- Gives all test instances whatever label was most common in the training set
- Accuracy might be very high if the problem is skewed

Perceptrons (Classification)

- Inputs are feature values
- Each feature has a weight
- Sum is the activation

$$\text{activation}_w(x) = \sum_i w_i f_i(x) = wf(x)$$

- If the activation is positive, output +1, if it is negative, output -1

Weights

- Binary case $\triangleq$ compare features to a weight vector
- Learning $\triangleq$ figure out the weight vector from examples
- Vectors on side of hyperplane orthogonal to $\vec{w}$ are positive; those on other side of orthogonal hyperplane are negative
  - Generally the vectors we care about are $f(x_i) \forall i \in n$ (all weight vectors are hyperplanes)
  - One side corresponds to $Y = +1$, the other corresponds to $Y = -1$

Weight Updates (Learning)

- Start with weights $w = 0$
- For each training instance $f(x), y^*$:
  - $y^* :=$ correct mapping of x
  2. Classify with current weights
     -
       $$y = \begin{cases} +1, w \cdot f(x) \geq 0 \\ -1, w \cdot f(x) < 0 \end{cases}$$
     - If correct ($y = y^*$), don't change anything
     - If wrong: adjust the weight vector by adding or subtracting the feature vector (subtract if $y^* = -1$): $w = w + y^* f$
- Eventually converges with separable data