Bernoulli Distribution

$p(x) = \text{Ber}(x|\lambda)$ $\quad Pr(x) = \lambda^x(1-\lambda)^{1-x}$

Categorical Distribution

$p(x) = \text{Cat}(x|\lambda)$ $\quad Pr(x = \mathbf{e}_k) = \prod_{j=1}^{K} \lambda_j^{\mathbf{e}_{kj}} = \lambda_k$

$$y_{MAP} = \underset{y}{\text{argmax}}\, p(y)p(x|y)$$

$$p(y \mid x) = \frac{p(x|y)p(y)}{p(x)} \propto p(x|y)p(y)$$

- Prior $p(y)$ is constant (uniform) $\Longrightarrow$ maximum likelihood (ML) estimate

$$y_{MAP} = \underset{y}{\text{argmax}}\, p(x|y) \triangleq y_{ML}$$

$$p(D|\theta) = \text{Bin}(N_1|N,\theta) = \binom{N}{N_1}\theta^{N_1}(1-\theta)^{N_0} \quad \text{(binomial likelihood)}$$

$$p(\theta|a,b) = \text{Beta}(\theta|a,b) = \frac{1}{B(a,b)}\theta^{a-1}(1-\theta)^{b-1}$$

For $\text{Beta}(x|c,d)$, mode $= \frac{c-1}{c+d-2}$

Posterior $p(\theta|D) = \text{Beta}(\theta|N_1+a, N_0+b)$

Maximum-A-Posteriori (MAP) estimate

$$\hat{\theta}_{MAP} = \underset{\theta}{\text{argmax}}\, p(\theta|D) = \frac{N_1+a-1}{N+a+b-2}$$

Maximum-likelihood (ML) estimate

$$\hat{\theta}_{ML} = \frac{N_1}{N} \text{ by setting a = b = 1 (uniform prior)}$$

For $\text{Dir}(x|\gamma = \gamma_1, \cdots, \gamma_K)$, mode $x_k = \frac{\gamma_k - 1}{\sum_k \gamma_k - K}$

Posterior $p(\theta|D) = \text{Dir}(\theta|N_1+\alpha_1, \cdots, N_K+\alpha_K)$

Maximum-A-Posteriori (MAP) estimate $\hat{\theta}_{MAP} = \text{argmax}_\theta\, p(\theta|D)$

$$\hat{\theta}_k^{MAP} = \frac{N_k+\alpha_k-1}{N+\sum_k \alpha_k - K}$$

Maximum-likelihood (ML) estimate

$$\hat{\theta}_k^{ML} = \frac{N_k}{N} \text{ by setting } \alpha_k = 1 \text{ (uniform prior)}$$

- Strategy 1 (Maximum likelihood)
  - Step 1: Estimate $\boldsymbol{\theta}_{ML} = \text{argmax}_{\boldsymbol{\theta}}\, p(\text{ D} \mid \boldsymbol{\theta})$
  - Step 2: Plug in $\boldsymbol{\theta}_{ML}$ into p(x, y | $\boldsymbol{\theta}_{ML}$) and find MAP estimate of y
- Strategy 2 (Maximum-A-Posteriori)
  - Step 1: Estimate $\boldsymbol{\theta}_{MAP} = \text{argmax}_{\boldsymbol{\theta}}\, p(\boldsymbol{\theta} \mid \text{D})$
  - Step 2: Plug in $\boldsymbol{\theta}_{MAP}$ into p(x, y | $\boldsymbol{\theta}_{MAP}$) and find MAP estimate of y

Strategy 1 (Maximum likelihood)

- Step 1: Estimate $\boldsymbol{\theta}_{ML} = \text{argmax}_{\boldsymbol{\theta}}\, p(\text{ D} \mid \boldsymbol{\theta})$
- Step 2: To predict label $\tilde{y}$ of test data $\tilde{x}$, plug in $\boldsymbol{\theta}_{ML}$ into posterior $p(\tilde{y} \mid \tilde{x}, \boldsymbol{\theta}) \propto p(\tilde{y} \mid \lambda_{ML})\, p(\tilde{x} \mid \tilde{y}, \eta_{ML})$ and find MAP estimate of $\tilde{y}$

Strategy 2 (Maximum-A-Posteriori)

- Step 1: Estimate $\boldsymbol{\theta}_{MAP} = \text{argmax}_{\boldsymbol{\theta}}\, p(\boldsymbol{\theta} \mid \text{D})$
- Step 2: To predict label $\tilde{y}$ of test data $\tilde{x}$, plug in $\boldsymbol{\theta}_{MAP}$ into posterior $p(\tilde{y} \mid \tilde{x}, \boldsymbol{\theta}) \propto p(\tilde{y} \mid \lambda_{MAP})\, p(\tilde{x} \mid \tilde{y}, \eta_{MAP})$ and find MAP estimate of $\tilde{y}$

$$g = \frac{d}{dw}NLL(w) = \sum_{i=1}^{N}(\mu_i - y_i)x_i = X^T(\mu - y)$$

$$H = \frac{d}{dw}g(w)^T = \sum_{i=1}^{N}\mu_i(1-\mu_i)x_i x_i^T = X^T S X,$$

Univariate Gaussian Distribution

$Pr(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-0.5(x-\mu)^2/\sigma^2\right]$ $\quad p(x) = \mathcal{N}(x|\mu,\sigma^2)$

$$\underset{\mu,\sigma^2}{\text{argmax}} \sum_{n=1}^{N}\left[-\frac{(x_n-\mu)^2}{2\sigma^2} - \log\sqrt{2\pi\sigma^2}\right] \qquad \hat{\mu} = \frac{1}{N}\sum_{n=1}^{N}x_n \qquad \hat{\sigma}^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n-\mu)^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n-\hat{\mu})^2$$

$$g_{reg}(\mathbf{w}) = g(\mathbf{w}) + \lambda\mathbf{w}$$

$$H_{reg}(\mathbf{w}) = H(\mathbf{w}) + \lambda I,$$

λ are parameters of class prior: $p(y \mid \boldsymbol{\theta}) = p(y \mid \lambda)$

η are parameters of feature likelihood: $p(x \mid y, \boldsymbol{\theta}) = p(x \mid y, \eta)$

$$\log \lambda_c^{MAP} \prod_{j=1}^{D} p(\tilde{x}_j|\eta_{jc}^{MAP}) = \log \lambda_c^{MAP} + \sum_{j=1}^{D} \log p(\tilde{x}_j|\eta_{jc}^{MAP}) \qquad p(x) = \frac{k(x)/N}{V}$$

$$NLL(w) = -\sum_{i=1}^{N}\log p(y_i|x_i,w) = -\sum_{i=1}^{N}[y_i \log\mu_i + (1-y_i)\log(1-\mu_i)]$$

$$NLL_{reg}(\mathbf{w}) = NLL(\mathbf{w}) + \tfrac{1}{2}\lambda\mathbf{w}^T\mathbf{w}$$

If $\mathbf{w}$ is big, then $\frac{1}{2}\lambda\mathbf{w}^T\mathbf{w}$ is big, so $NLL_{reg}(\mathbf{w})$ is big. Since we are minimizing $NLL_{reg}(\mathbf{w})$, this means big $\mathbf{w}$ is discouraged

Consider a 3-class naive Bayes classifier with one binary feature and one Gaussian feature. More specifically, class label y follows a categorical distribution parametrized by π, i.e., $p(y = c) = \pi_c$. The first feature x_1 is binary and follows a Bernoulli distribution: $p(x_1|y = c) = \text{Bernoulli}(x_1|\theta_c)$. The second feature x_2 is univariate Gaussian: $p(x_2|y = c) = \mathcal{N}(x_2|\mu_c, \sigma_c^2)$. Let $\pi = [0.5\ \ 0.25\ \ 0.25]$, $\theta = [0.5\ \ 0.5\ \ 0.5]$, $\mu = [-1\ \ 0\ \ 1]$ and $\sigma^2 = [1\ \ 1\ \ 1]$.

(i) Compute $p(y|x_1 = 0)$. Note that result is a vector of length 3 that sums to 1.

(ii) Compute $p(y|x_2 = 0)$. Note that result is a vector of length 3 that sums to 1.

(iii) Compute $p(y|x_1 = 0, x_2 = 0)$. Note that result is a vector of length 3 that sums to 1.

(i)

$$\begin{aligned} p(y|x_1 = 0) &= \frac{p(y)p(x_1 = 0|y)}{p(x_1 = 0)} \\ &\propto p(y)p(x_1 = 0|y) \\ &\propto p(y) \qquad \text{because feature is uninformative.} \\ &= [0.5\ \ 0.25\ \ 0.25] \end{aligned}$$

x_1 is uninformative because $\theta = [0.5\ \ 0.5\ \ 0.5]$, i.e., the probability of getting a head is the same for all classes. Note that x_1 will also be uninformative if $\theta = [0.4\ \ 0.4\ \ 0.4]$.

(ii)

$$\begin{aligned} p(y|x_2 = 0) &= \frac{p(y)p(x_2 = 0|y)}{p(x_2 = 0)} \\ &\propto p(y)p(x_2 = 0|y) \\ &= \pi_y \frac{1}{\sqrt{2\pi\sigma_y^2}} e^{-\frac{1}{2\sigma_y^2}(0-\mu_y)^2} \\ &\propto \pi_y e^{-0.5\mu_y^2} \\ &= [0.5e^{-0.5}\ \ 0.25e^{0}\ \ 0.25e^{-0.5}] \\ &= [0.3033\ \ 0.25\ \ 0.1516] \end{aligned}$$

Therefore

$$\begin{aligned} p(y|x_2 = 0) &= [0.3033\ \ 0.25\ \ 0.1516]/(0.3033 + 0.25 + 0.1516) \\ &= [0.4302\ \ 0.3547\ \ 0.2151] \end{aligned}$$

(iii)

$$\begin{aligned} p(y|x_1 = 0, x_2 = 0) &= p(y|x_2 = 0) \qquad \text{because } x_1 \text{ is uninformative.} \\ &= [0.4302\ \ 0.3547\ \ 0.2151] \end{aligned}$$

Q1: Parzen's Window

Consider data samples x_1, x_2, x_3, x_4 to be $1, 3, 4, 10$. Using the Gaussian Parzen's window: $\frac{1}{\sqrt{2\pi h^2}} e^{-\frac{x^2}{2h^2}}$, what is the Parzen's window estimate of $p_h(x)$ at $x = 2$ and $x = 5$ for $h = 1$?

Q2: KNN

Consider training data $x_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $x_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, $x_3 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$, $x_4 = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$ with corresponding class labels $y_1 = 0, y_2 = 0, y_3 = 1, y_4 = 1$. What is the 3-NN estimate of the class label posterior probabilities of datapoints $x_5 = \begin{pmatrix} 0 \\ 0.5 \end{pmatrix}$ and $x_6 = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$, where the distance metric used is the Euclidean distance? What are the MAP classifications of data points x_5 and x_6?

a. Now consider the following prior, that believes the coin is fair, or is slightly biased towards tails:

$$p(\theta) = \begin{cases} 0.5 & \text{if } \theta = 0.5 \\ 0.5 & \text{if } \theta = 0.4 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Derive the MAP estimate under the prior as a function of N_1 and N.

a. The posterior of the Bernoulli

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

if $\theta = 0.5$,

$$\begin{aligned} & p(\mathcal{D}|\theta)p(\theta) = 0.5^{N+1} \\ \implies & \log p(\mathcal{D}|\theta)p(\theta) = (N+1)\log 0.5 \end{aligned}$$

if $\theta = 0.4$,

$$\begin{aligned} & p(\mathcal{D}|\theta)p(\theta) = 0.4^{N_1} 0.6^{N-N_1} 0.5 \\ \implies & \log p(\mathcal{D}|\theta)p(\theta) = N_1 \log 0.4 + (N - N_1)\log 0.6 + \log 0.5 \end{aligned}$$

Consider a binary classification problem of predicting binary class y from features x. The cost of wrong prediction is \$6 and the cost of correct prediction is 0. Suppose the cost of asking a human to perform the manual classification is \$2. Therefore for a particular x, there are three possible decisions: (1) decision α_0 predicts y to be 0, (2) decision α_1 predicts y to be 1 and (3) decision α_h requires a human to perform the manual classification. Let $p_1 = p(y = 1|x)$

(i) Assume the human is 100% accurate and suppose $p_1 = 0.4$, what should our decision be to minimize expected loss?

(ii) Assume the human is 100% accurate and suppose $p_1 = 0.1$, what should our decision be to minimize expected loss?

(iii) Assume the human is 100% accurate. What is the general decision rule (as a function of p_1) in order to minimize expected loss?

(iv) Assume the human is only 95% accurate. What is the general decision rule (as a function of p_1) in order to minimize expected loss?

- (i) Here are the expected loss

$$\begin{aligned} R(\alpha_0) &= 0.4 * 6 = 2.4 \\ R(\alpha_1) &= 0.6 * 6 = 3.6 \\ R(\alpha_h) &= 2 \end{aligned}$$

Therefore we should choose α_h

- (ii) Here are the expected loss

$$\begin{aligned} R(\alpha_0) &= 0.1 * 6 = 0.6 \\ R(\alpha_1) &= 0.9 * 6 = 5.4 \\ R(\alpha_h) &= 2 \end{aligned}$$

Therefore we should choose α_0

- (iii) The general expected loss is given by

$$\begin{aligned} R(\alpha_0) &= 6p_1 \\ R(\alpha_1) &= 6(1 - p_1) \\ R(\alpha_h) &= 2 \end{aligned}$$

We should choose α_0 if $R(\alpha_0) < R(\alpha_1) \implies p_1 < 0.5$ **and** $R(\alpha_0) < R(\alpha_h) \implies p_1 < 1/3$
We should choose α_1 if $R(\alpha_1) < R(\alpha_0) \implies p_1 > 0.5$ and $R(\alpha_1) < R(\alpha_h) \implies p_1 > 2/3$
Therefore we should choose α_0 if $p_1 < 1/3$, α_1 if $p_1 > 2/3$ and α_h otherwise.

- (iv) If the human is correct 95% of the time, then the general expected cost of α_h is $0.95 \times 2 + 0.05 \times 8 = 2.3$
Therefore, we should choose α_0 if $p_1 < 0.5$ and $R(\alpha_0) < R(\alpha_h) \implies p_1 < 2.3/6 = 0.383$
And we should choose α_1 if $p_1 > 0.5$ and $R(\alpha_1) < R(\alpha_h) \implies 6(1-p_1) < 2.3 = 1 - 0.38 = 0.617$
Therefore we should choose α_0 if $p_1 < 0.383$, α_1 if $p_1 > 0.617$ and α_h otherwise.

Q1: Parzen's Window

- $p_h(x) = \frac{1}{4}\sum_{n=1}^{4} \frac{1}{\sqrt{2\pi}} e^{-\frac{(x_n - x)^2}{2}}$
- Plugging x = 2 and the data, we get

$$\begin{aligned} p_1(x = 2) &= \frac{1}{4\sqrt{2\pi}} \left[e^{-\frac{(2-1)^2}{2}} + e^{-\frac{(2-3)^2}{2}} + e^{-\frac{(2-4)^2}{2}} + e^{-\frac{(2-10)^2}{2}} \right] \\ &= \frac{1}{4\sqrt{2\pi}} \left[e^{-0.5} + e^{-0.5} + e^{-2} + e^{-32} \right] \\ &= 0.134483 \end{aligned}$$

- Plugging x = 5 and the data we get

$$\begin{aligned} p_1(x = 5) &= \frac{1}{4\sqrt{2\pi}} \left[e^{-\frac{(5-1)^2}{2}} + e^{-\frac{(5-3)^2}{2}} + e^{-\frac{(5-4)^2}{2}} + e^{-\frac{(5-10)^2}{2}} \right] \\ &= \frac{1}{4\sqrt{2\pi}} \left[e^{-8} + e^{-2} + e^{-0.5} + e^{-12.5} \right] \\ &= 0.0740 \end{aligned}$$

Q2: KNN

- The 3 closest datapoints for x_5 are x_1, x_2, and x_3.
- Therefore $p(y = 1|x) = 1/3$ and $p(y = 0|x) = 2/3$
- Therefore the datapoint should be classified as class 0
- The 3 closest datapoints for x_6 are x_2 (or x_1), x_3, and x_4.
- Therefore $p(y = 1|x) = 2/3$ and $p(y = 0|x) = 1/3$ (Note that x_1 and x_2 are equidistant, so I am also ok with $p(y = 1|x) = p(y = 0|x) = 1/2$)
- Therefore the datapoint should be classified as class 1

For 0.5 to win out over 0.4,

$$\begin{aligned} & (N+1)\log 0.5 > N_1 \log 0.4 + (N - N_1)\log 0.6 + \log 0.5 \\ \implies & N \log \frac{0.5}{0.6} > N_1 \log \frac{0.4}{0.6} \\ \implies & \frac{N_1}{N} > \frac{\log 5/6}{\log 2/3} = \frac{\log 1.2}{\log 1.5} = 0.4497 \text{ because } \log 2/3 \text{ is negative} \end{aligned}$$

Therefore, we have

$$\hat{\theta}_{MAP} = \begin{cases} 0.4 & \text{if } \frac{N_1}{N} < \frac{\log 1.2}{\log 1.5} \\ 0.5 & \text{if } \frac{N_1}{N} > \frac{\log 1.2}{\log 1.5} \end{cases}$$

Note that N_1/N can never be exactly equal to $\frac{\log 1.2}{\log 1.5}$ because $\frac{\log 1.2}{\log 1.5}$ is irrational.

Ensemble Classifiers

- Basic idea: Build different "experts" and let them vote
- Advantages:
  - Improve predictive performance
  - Different types of classifiers can be directly included
  - Easy to implement
  - Not too much parameter tuning
- Disadvantages:
  - The combined classifier is not transparent (black box)
  - Not a compact representation

Bagging = Bootstrap Aggregating

- Training
  - Given a dataset S, at each iteration i, a training set S_i is sampled with replacement from S (i.e. bootstraping)
  - A classifier C_i is learned for each S_i
- Classification: given an unseen sample X
  - Each classifier C_i returns its class prediction
  - The bagged classifier H counts the votes and assigns the class with the most votes to X

Bagging

- Bagging works because it reduces variance by voting/averaging
  - Usually, the more classifiers the better
- Problem: we only have one dataset
- Solution: generate new ones of size n by bootstrapping, i.e. sampling with replacement
- Can help a lot if data is noisy

- *Bagging* = Bootstrap Aggregating
- Reweighing of the learning sets is done by drawing at random with replacement from the learning sets
- Predictors are aggregated by voting
- Main idea: train a strong classifier by combining weak classifiers
  - Practically useful
  - Theoretically interesting

Advantages of Random Forests

- Very high accuracy – not easily surpassed by other algorithms
- Efficient on large datasets
- Can handle thousands of input variables without variable deletion
- Effective method for estimating missing data, also maintains accuracy when a large proportion of the data are missing
- Robust to label noise
- Can be used in clustering, locating outliers and semi-supervised learning

Classifier Cascade (Viola-Jones)

- For real problems results are only as good as the features used.
  - This is the main piece of ad-hoc (or domain) knowledge
- Rather than the pixels, use a very large set of simple functions
  - Sensitive to edges and other critical features of the image
  - Computed at multiple scales
- Introduce a threshold to yield binary features
  - Binary features seem to work better in practice
  - In general, convert continuous features to binary by quantizing

Supervised vs. Unsupervised Learning

- Up to now we considered **supervised learning** scenarios, where we are given:
  1. samples $x_1, \ldots, x_n$
  2. class labels for all samples
  - This is also called learning ***with*** teacher, since the correct answer (the true class) is provided
- Here, we consider **unsupervised learning** scenarios, where we are only given:
  1. Only samples $x_1, \ldots, x_n$
  - This is also called learning ***without*** teacher, since the correct answer is not provided
  - Do not split data into training and test sets

- Parametric Approach
  - Assume parametric ***distribution*** of data
  - Estimate parameters of this distribution
    - Expectation Maximization
- Non-Parametric Approach
  - Group the data into ***clusters***, each cluster (hopefully) says something about classes present in the data

- What is a good clustering?
  1. internal distances should be small
  2. external should be large

Why Unsupervised Learning?

- Unsupervised learning is harder
  - How do we know if results are meaningful? No answer (labels) is available
    - Let the experts look at the results (external evaluation)
    - Define an objective function on clustering (internal evaluation)
- We nevertheless need it because
  1. Labeling large datasets is very costly (speech recognition, object detection in images)
     - Sometimes can label only a few examples by hand
  2. May have no idea what/how many classes there are (data mining)
  3. May want to use clustering to gain some insight into the structure of the data before designing a classifier

K-means Clustering

- Finding the optimum of J_{SSE} is NP-hard
- In practice, k-means clustering usually performs well
- It can be very efficient
- Its solution can be used as a starting point for other clustering algorithms
- Hundreds of papers on variants and improvements of k-means clustering are published every year

Hierarchical Clustering

- Generates minimum spanning tree
- Encourages growth of elongated clusters
- Disadvantage: very sensitive to noise

Clustering Summary

- Clustering (nonparametric learning) is useful for discovering inherent structure in data
- Clustering is immensely useful in different fields
- Clustering comes naturally to humans (in up to 3 dimensions), but not so to computers
- It is very easy to design a clustering algorithm, but it is very hard to make theoretical claims on performance
- General purpose clustering is unlikely to exist; for best results, clustering should be tuned to application at hand