Naive Bayes, LDA, and Logistic Regression

courtesy of Prof. Marlin

The Classification Task

Definition: The Classification Task

Given a feature vector $\mathbf{x} \in \mathbb{R}^D$ that describes an object that belongs to one of C classes from the set $\mathcal{Y}$, predict which class the object belongs to.

The Classifier Learning Problem

Definition: Classifier Learning

Given a data set of example pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 : N\}$ where $\mathbf{x}_i \in \mathbb{R}^D$ is a feature vector and $y_i \in \mathcal{Y}$ is a class label, learn a function $f : \mathbb{R}^D \to \mathcal{Y}$ that accurately predicts the class label y for any feature vector $\mathbf{x}$.

Classification Error and Accuracy

Definition: Classification Error Rate

Given a data set of example pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 : N\}$ and a function $f : \mathbb{R}^D \to \mathcal{Y}$, the classification error rate of f on $\mathcal{D}$ is:

$$Err(f, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(y_i \neq f(\mathbf{x}_i))$$

Definition: Classification Accuracy Rate

Given a data set of example pairs $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 : N\}$ and a function $f : \mathbb{R}^D \to \mathcal{Y}$, the classification accuracy rate of f on $\mathcal{D}$ is:

$$Acc(f, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(y_i = f(\mathbf{x}_i))$$

Probabilistic Classification

- Suppose we know that the true probability of seeing a data case that belongs to class c is $P(Y = c) = \pi_c$ and the true probability density of seeing a data vector $\mathbf{x} \in \mathbb{R}^D$ that belongs to class c is $p(\mathbf{X} = \mathbf{x}|Y = c) = \phi_c(\mathbf{x})$.
- We can use this information to compute the probability that the class of any observation $\mathbf{x}$ is c. How?
- Bayes Rule: $P(Y = c|\mathbf{X} = \mathbf{x}) = \dfrac{\phi_c(\mathbf{x})\pi_c}{\sum_{c' \in \mathcal{Y}} \phi_{c'}(\mathbf{x})\pi_{c'}}$

The Bayes Optimal Classifier

- The Bayes Optimal Classifier uses the classification function:

$$\begin{aligned} f_B(\mathbf{x}) &= \arg\max_{c \in \mathcal{Y}} P(Y = c | \mathbf{X} = \mathbf{x}) \\ &= \arg\max_{c \in \mathcal{Y}} \phi_c(\mathbf{x}) \pi_c \end{aligned}$$

- The Bayes optimal classifier achieves the minimal possible expected error rate among all possible classifiers:

$$1 - \mathbb{E}_{P(\mathbf{X}=\mathbf{x})} \left[\max_{c \in \mathcal{Y}} P(Y = c | \mathbf{X} = \mathbf{x}) \right]$$

- This is a nice result, but is it useful in practice?

Naive Bayes

- The naive Bayes classifier approximates the Bayes optimal classifier using a simple form for the functions $\phi_c(\mathbf{x})$.
- This form assumes that all of the data dimensions are probabilisticaly independent given the value of the class variable:

$$\phi_c(\mathbf{x}) = p(\mathbf{X} = \mathbf{x}|Y = c) = \prod_{d=1}^{D} p(X_d = x_d|Y = c) = \prod_{d=1}^{D} \phi_{cd}(x_d)$$

- The general form for the classification function is:

$$f_{NB}(\mathbf{x}) = \arg\max_{c \in \mathcal{Y}} \pi_c \prod_{d=1}^{D} \phi_{cd}(x_d)$$

Class Conditional Distributions

- The functions $p(X_d = x_d|Y = c) = \phi_{cd}(x_d)$ are called *marginal class conditional distributions*.
- For real valued x_d, $p(X_d = x_d|Y = c)$ is typically modeled as a normal density $\phi_{cd}(x_d) = \mathcal{N}(x_d; \mu_{dc}, \sigma^2_{dc}) = \frac{1}{\sqrt{2\pi\sigma^2_{dc}}} \exp\left(-\frac{1}{2\sigma^2_{dc}}(x_d - \mu_{dc})^2\right)$.
- For binary valued x_d, $p(X_d = x_d|Y = c)$ is a Bernoulli distribution $\phi_{cd}(x_d) = \theta_{dc}^{x_d}(1 - \theta_{dc})^{(1-x_d)}$.
- For general categorical valued x_d, $p(X_d = x_d|Y = c)$ is a categorical distribution $\phi_{cd}(x_d) = \prod_{v \in \mathcal{X}_d} \theta_{vdc}^{[x_d = v]}$

Learning for Naive Bayes

- The class probabilities π_c and the parameters of the marginal class conditional distributions $\phi_{cd}(x_d)$ are learned by maximum likelihood on a sample of training data $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1:n\}$.

- This reduces to estimating them using class-conditional sample averages for most common distributions.

- Class probabilities: $\pi_c = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n}[y_i = c]$

- Normal: $\mu_{dc} = \frac{\sum_{i=1}^{n}[y_i = c]x_{di}}{\sum_{i=1}^{n}[y_i = c]}$ $\quad \sigma^2_{dc} = \frac{\sum_{i=1}^{n}[y_i = c](x_{di} - \mu_{dc})^2}{\sum_{i=1}^{n}[y_i = c]}$

- Bernoulli: $\theta_{dc} = \frac{\sum_{i=1}^{n}[y_i = c]x_{di}}{\sum_{i=1}^{n}[y_i = c]}$ Categorical: $\theta_{vdc} = \frac{\sum_{i=1}^{n}[y_i = c][x_{di} = v]}{\sum_{i=1}^{n}[y_i = c]}$

Geometric Interpretation

- Suppose we use normal distributions for the marginal class conditional distributions and we have a binary classification problem. What is the geometry of the decision boundary?

$$
\begin{aligned}
f_{NB}(\mathbf{x}) &= \underset{c \in \mathcal{Y}}{\arg\max}\, \pi_c \prod_{d=1}^{D} \phi_{cd}(x_d) \\
&= \underset{c \in \mathcal{Y}}{\arg\max}\, \log(\pi_c) + \sum_{d=1}^{D} \log(\phi_{cd}(x_d)) \\
&= \underset{c \in \mathcal{Y}}{\arg\max}\, \log(\pi_c) + \sum_{d=1}^{D} \left(-\frac{1}{2}\log(2\pi\sigma_{dc}^2) - \frac{1}{2\sigma_{dc}^2}(x_d - \mu_{dc})^2 \right)
\end{aligned}
$$

Geometric Interpretation

- The decision boundary consists of the set of points $\mathbf{x}$ where:

$$\log(\pi_0) + \sum_{d=1}^{D} \left(-\frac{1}{2} \log(2\pi\sigma_{d0}^2) - \frac{1}{2\sigma_{d0}^2}(x_d - \mu_{d0})^2 \right)$$
$$- \log(\pi_1) - \sum_{d=1}^{D} \left(-\frac{1}{2} \log(2\pi\sigma_{d1}^2) - \frac{1}{2\sigma_{d1}^2}(x_d - \mu_{d1})^2 \right) = 0$$

- It's easy to see that the decision boundary is a quadratic function of $\mathbf{x}$ with the form: $\sum_{d=1}^{D} \left(a_d x_d^2 + b_d x_d \right) + c = 0$.

- In the multi-class case, the decision boundary is piece-wise quadratic.

Trade-Offs

- Speed: Both learning and classification have very low computational complexity.
- Storage: The model only requires $O(DC)$ parameters. It achieves a large compression of the training data.
- Interpretability: The model has good interpretability since the parameters of $\phi_c(\mathbf{x})$ correspond to class conditional averages.
- Accuracy: The assumption of feature independence and canonical forms for class-conditional marginals will rarely be correct for real-world problems, leading to lower accuracy.
- Data: Some care is needed in the estimation of parameters in the discrete case when data is scarce.

Linear Discriminant Analysis

- Linear Discriminant Analysis (LDA) is a classification technique due to Fisher that dates back to the 1930's.
- It can be interpreted as a different approximation to the Bayes Optimal Classifier for real-valued data.
- Instead of a product of independent Normals like in Naive Bayes, LDA assumes the class-conditional densities are multivariate normal with a common covariance matrix:

$$\phi_c(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mu_c, \Sigma) = \frac{1}{|2\pi\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_c)^T \Sigma^{-1} (\mathbf{x} - \mu_c)\right)$$

- The classification function is $f_{LDA}(\mathbf{x}) = \arg\max_{c \in \mathcal{Y}} \phi_c(\mathbf{x}) \pi_c$

Learning for LDA

- As with Naive Bayes, LDA parameters are learned using maximum likelihood, which reduces to using sample estimates.
- Class probabilities: $\pi_c = \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} [y_i = c]$
- Class Means: $\mu_c = \frac{\sum_{i=1}^{n} [y_i = c] \mathbf{x}_i}{\sum_{i=1}^{n} [y_i = c]}$
- Shared Covariance: $\Sigma = \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} (\mathbf{x}_i - \mu_c)(\mathbf{x}_i - \mu_c)^T$

Geometric Interpretation

- What is the geometry of the decision boundary for LDA in the binary case?
- The decision boundary consists of the set of points $\mathbf{x}$ where:

$$\log(\pi_0) - \frac{1}{2}\log|2\pi\Sigma| - \frac{1}{2}(\mathbf{x}-\mu_0)^T\Sigma^{-1}(\mathbf{x}-\mu_0)$$
$$-\log(\pi_1) + \frac{1}{2}\log|2\pi\Sigma| + \frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma^{-1}(\mathbf{x}-\mu_1) = 0$$

- We can cancel a large number of terms because of the common covariance matrix and obtain the following result:

  $$\log(\pi_0)-\log(\pi_1)-0.5\mu_0^T\Sigma^{-1}\mu_0+0.5\mu_1^T\Sigma^{-1}\mu_1+(\mu_0-\mu_1)^T\Sigma^{-1}\mathbf{x} = 0$$

- This shows that the decision boundary is actually linear in $\mathbf{x}$.

Trade-Offs

- Speed: The quadratic dependence on D makes LDA slower than Naive Bayes by a factor of D during learning and classification.
- Storage: The model requires $O(D^2C)$ parameters. This can still represent a good compression of the data when $D << N$.
- Interpretability: The model has good interpretability since the mean parameters μ_c correspond to class conditional averages.
- Accuracy: The assumptions LDA makes will rarely be correct for real-world problems. However, the induced linear decision boundaries can often perform reasonably well.
- Data: LDA will generally need more data than NB since it needs to estimate the $O(D^2)$ parameters in the pooled covariance matrix Σ.

Generative vs Discriminative Classifiers

- The Bayes Optimal Classifier, Naive Bayes and LDA are said to be *generative* classifiers because they explicitly model the joint distribution $P(\mathbf{X}, Y)$ of the data vectors $\mathbf{x}$ and the labels y. Question: is this really necessary?

- No, to build a probabilistic classifier, all we really need to model is $P(Y|\mathbf{X})$.

- Classifiers based on directly estimating $P(Y|\mathbf{X})$ are called *descriminative* classifiers because they ignore the distribution of $\mathbf{x}$ and focus only on learning the distribution of y.

Logistic Regression

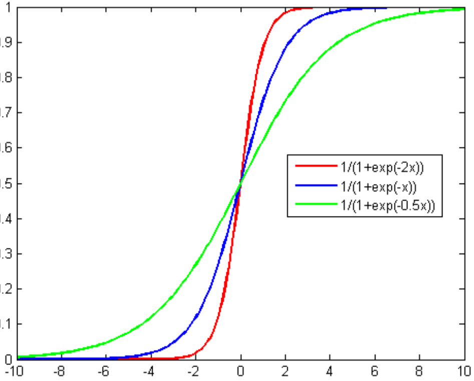
- Logistic Regression is a probabilistic discriminative classifier.
- In the binary case, it directly models the decision boundary using a linear function:

$$\log P(Y=1|\mathbf{x}) - \log P(Y=0|\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

$$P(Y=1|\mathbf{X}) = \frac{1}{1+\exp(-(\mathbf{w}^T\mathbf{x}+b))}$$

- The classification function is: $f_{LR}(\mathbf{x}) = \arg\max_{c\in\mathcal{Y}} P(Y=c|\mathbf{x})$

Logistic Function



Multiclass Logistic Regression

- Logistic regression can also be extended to the multiclass case:

$$P(Y = c|\mathbf{x}) = \frac{\exp(\mathbf{w}_c^T\mathbf{x} + b_c)}{\sum_{c'\in\mathcal{Y}}\exp(-(\mathbf{w}_{c'}^T\mathbf{x} + b_{c'}))}$$

- The classification function is still:
$f_{LR}(\mathbf{x}) = \arg\max_{c\in\mathcal{Y}} P(Y = c|\mathbf{x})$

Learning Logistic Regression

- The logistic regression model parameters $\theta = \{(\mathbf{w}_c, b_c), c \in \mathcal{Y}\}$ are selected to optimize the conditional log likelihood of the labels given a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 : n\}$:

$$\theta_* = \arg\max_{\theta} \mathcal{L}(\theta|\mathcal{D}) = \arg\max_{\theta} \sum_{i=1}^{n} \log P(Y = y_i|\mathbf{X} = \mathbf{x}_i)$$

- However, the function $\mathcal{L}(\theta|\mathcal{D})$ can not be maximized analytically. Learning the model parameters requires numerical optimization methods.

Geometry

- Logistic regression is explicitly designed to have a linear decision boundary in the binary case.
- In the multiclass case, the decision boundary is piece-wise linear.
- Logistic Regression has the same representational capacity as LDA.

Trade-Offs

- Speed: At classification time, LR is faster than NB and LDA. Learning LR requires numerical optimization, which will be slower than NB.
- Storage: The model requires $O(DC)$ parameters. The same order as Naive Bayes, but much less than LDA's $O(DC + D^2)$ when $C << D$.
- Interpretability: The "importance" of different feature variables x_d can be understood in terms of their weights w_{dc}.
- Accuracy: Tends to be better in high dimensions with limited data compared to LDA. Much worse than KNN in low dimensions with lots of data and non-linear decision boundaries.

Review/Preview/To Do

- **Review:** We've introduced the Bayes Optimal Classifier, Naive Bayes, LDA and Logistic Regression.
- **Preview:** Next class we'll cover meta-issues for classification.
- **To Do:** Homework 1 will be posted later tonight.