

Kede Ma



Slide template by courtesy of Benjamin M. Marlin

Classification with Generative Models

- Steps to build a classifier
 - 1 Collect training data (features \mathbf{x} and class labels y)
 - 2 Learn **class-conditional distribution**, $p(\mathbf{x}|y)$
 - 3 Use Bayes' rule to calculate class probability, $p(y|\mathbf{x})$
- **Note:** The data is used to learn the class-conditional distribution; the classifier is secondary
 - Density estimation is an **"ill-posed" problem** - which density to use? how much data is needed?
- Advice from Vladimir Vapnik (inventor of SVM)

When solving a problem, try to **avoid solving a more general problem as an intermediate step**

- Discriminative solution
 - Solve for the classifier $p(y|\mathbf{x})$ directly!

- **“Discriminative”** - learn to directly discriminate the classes apart using the features
- **“Generative”** - learn models of how the features are generated from different classes

■ C

- C

- Equivalently, $y = \text{sign}(f(\mathbf{x}))$

- The linear classifier separates the feature space into 2 half-spaces
 - Each corresponds to feature values belonging to Class +1 and Class -1
 - The class boundary is normal (orthogonal or perpendicular) to \mathbf{w}
 - Also called the **separating hyperplane**
- Example: $\mathbf{w} = [2, 1]^T, b = 0$



Kede Ma
CS5489 Lecture 2.2

- City University of Hong Kong (Dongguan)

Learning the Classifier

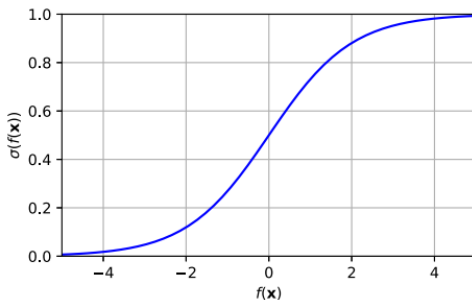
- How to set the classifier parameters (\mathbf{w}, b) ?
 - Learn them from training data!
- Classifiers differ in the objectives used to learn the parameters (\mathbf{w}, b)
 - We will look at **two examples:**
 - Logistic regression
 - Support vector machine (SVM)

为什么是并列关系？？

- Logistic regression takes a **probabilistic approach** 概率方法

- Sigmoid function maps from real number to interval $[0, 1]$

- $\sigma(z) = \frac{1}{1+e^{-z}}$, for $z \in \mathbb{R}$



Logistic Regression

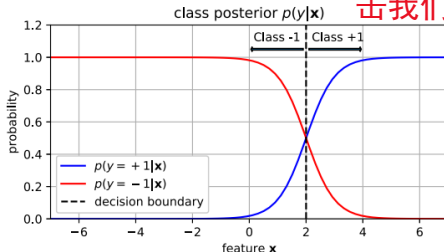
- Given a feature vector \mathbf{x} , the probability of a class is

- $p(y = +1|\mathbf{x}) = \sigma(f(\mathbf{x}))$
- $p(y = -1|\mathbf{x}) = 1 - \sigma(f(\mathbf{x})) = \sigma(-f(\mathbf{x}))$
- Equivalently, $p(y|\mathbf{x}) = \sigma(yf(\mathbf{x}))$

- Note: here we are directly modeling the **class posterior probability!**

- Not the class-conditional $p(\mathbf{x}|y)$

对后验概率建模，而不是
对类条件概率建模，更直
击我们的目标



Learning the Parameters

- Given the data set $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)}), i = 1, \dots, M\}$, learn the function parameters (\mathbf{w}, b) **using MLE**
- Maximize the conditional log likelihood of the data \mathcal{D} :

$$\begin{aligned}(\mathbf{w}^*, b^*) &= \arg \max_{\mathbf{w}, b} \frac{1}{M} \sum_{i=1}^M \log p(y^{(i)} | \mathbf{x}^{(i)}, \mathbf{w}; b) \quad \text{类条件概率} \\ &= \arg \max_{\mathbf{w}, b} -\frac{1}{M} \sum_{i=1}^M \log \left(1 + \exp \left(-y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) \right) \right)\end{aligned}$$

- To prevent **overfitting**, add a prior distribution on \mathbf{w}
 - Assume Gaussian distribution on $p(\mathbf{w})$ with variance $C/2$

$$p(\mathbf{w}) \propto \exp\left(-\frac{1}{C} \mathbf{w}^T \mathbf{w}\right)$$

Learning the Parameters

■ Now maximize

$$(\mathbf{w}^*, b^*) = \arg \max_{\mathbf{w}, b} \frac{1}{M} \sum_{i=1}^M \log p(\mathbf{w} | \mathbf{y}^{(i)}, \mathbf{x}^{(i)}; b)$$

带*表示最优参值

$$= \arg \max_{\mathbf{w}, b} \frac{1}{M} \sum_{i=1}^M \log \frac{p(\mathbf{w}, y^{(i)} | \mathbf{x}^{(i)}; b)}{p(y^{(i)} | \mathbf{x}^{(i)})}$$

$$= \arg \max_{\mathbf{w}, b} \frac{1}{M} \sum_{i=1}^M \log \frac{p(\mathbf{w}) p(y^{(i)} | \mathbf{x}^{(i)}, \mathbf{w}; b)}{p(y^{(i)} | \mathbf{x}^{(i)})}$$

$$= \arg \max_{\mathbf{w}, b} \log p(\mathbf{w}) + \frac{1}{M} \sum_{i=1}^M \log p(y^{(i)} | \mathbf{x}^{(i)}, \mathbf{w}; b)$$

$\log p(\mathbf{w})$ 是权重 \mathbf{w} 的对数先验概率，通常假设为高斯分布，用于正则化。

Learning the Parameters

- Equivalently,

$$(\mathbf{w}^*, b^*) = \arg \min_{\mathbf{w}, b} \frac{1}{C} \mathbf{w}^T \mathbf{w} + \frac{1}{M} \sum_{i=1}^M \log \left(1 + \exp \left(-y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) \right) \right)$$

- The first term is the **regularization term**
 - Note: $\mathbf{w}^T \mathbf{w} = \sum_{j=1}^N w_j^2$ 模型参数
 - Penalty term keeps **entries** in \mathbf{w} from getting too large
 - C is the regularization **hyperparameter**
 - Larger C values allow large values in \mathbf{w}
 - Smaller C values discourage large values in \mathbf{w}

Learning the Parameters

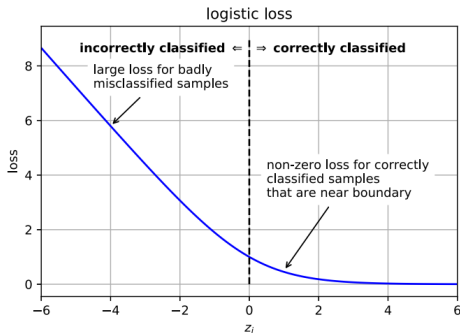
$$(\mathbf{w}^*, b^*) = \arg \min_{\mathbf{w}, b} \frac{1}{C} \mathbf{w}^T \mathbf{w} + \frac{1}{M} \sum_{i=1}^M \log \left(1 + \exp \left(-y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) \right) \right)$$

- The second term is the **data-fit term**
 - Wants to make the parameters (\mathbf{w}, b) to well fit the data
 - Defining $z^{(i)} = y^{(i)} f(\mathbf{x}^{(i)})$, we have the following interesting observation
 - $z^{(i)} > 0$, when sample $\mathbf{x}^{(i)}$ is classified correctly
 - $z^{(i)} < 0$, when sample $\mathbf{x}^{(i)}$ is classified incorrectly
 - $z^{(i)} = 0$, when sample $\mathbf{x}^{(i)}$ is on classifier boundary

Logistic Loss Function

■ Definition

$$L(z) = \log(1 + \exp(-z))$$



Learning the Parameters

$$(\mathbf{w}^*, b^*) = \arg \min_{\mathbf{w}, b} \underbrace{\frac{1}{C} \mathbf{w}^T \mathbf{w} + \frac{1}{M} \sum_{i=1}^M \log \left(1 + \exp \left(-y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) \right) \right)}_{\ell(\mathbf{w}, b)}$$

- No **closed-form solution** (i.e., no simple one-line equation to solve for (\mathbf{w}, b))
 - Use an **iterative** optimization algorithm to find the optimal solution
 - E.g., **gradient descent** - step downhill in each iteration
 - $\mathbf{w} \leftarrow \mathbf{w} - \eta \frac{\partial \ell}{\partial \mathbf{w}}$
 - ℓ is the **objective function**
 - η is the **learning rate** (how far to step in each iteration)

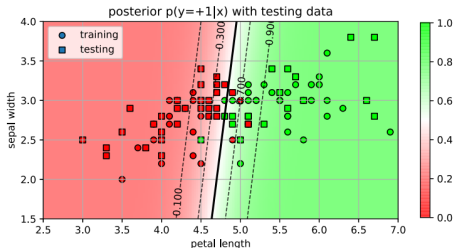
Example: Iris Data

■ Equation:

$$f(\mathbf{x}) = 4.87 \times \text{petal_length} - 0.62 \times \text{sepal_width} - 21.68$$

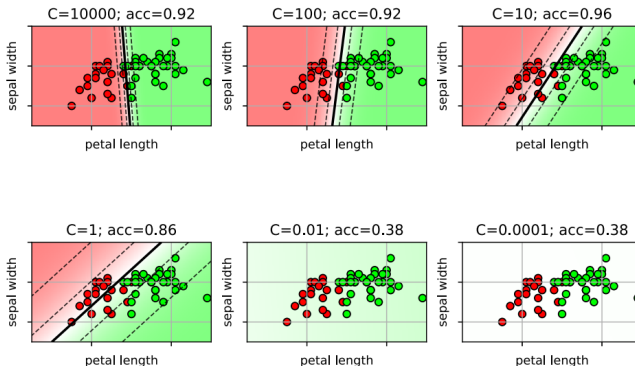
■ Interpretation: 解释

- Large petal length makes $f(\mathbf{x})$ positive, so large petal length is associated with Class +1



Selecting the Regularization Hyperparameter

- The regularization hyperparameter C has a big effect on the decision boundary and the accuracy
- How to select the value of C ?

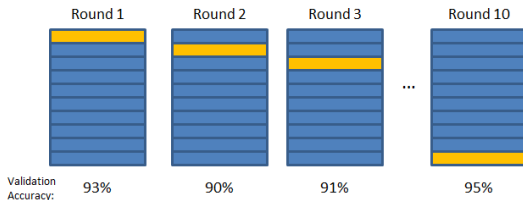


Cross-Validation

- Use **cross-validation** on **training** set to select the best value of C
- Run many experiments on the training set to see which parameters work on different versions of the data
- **Partition** the data into folds of training and validation data
- Try a range of C values on each fold (as validation set)
- Pick the value that works best over all folds

动词时, 分割

Validation Set
Training Set



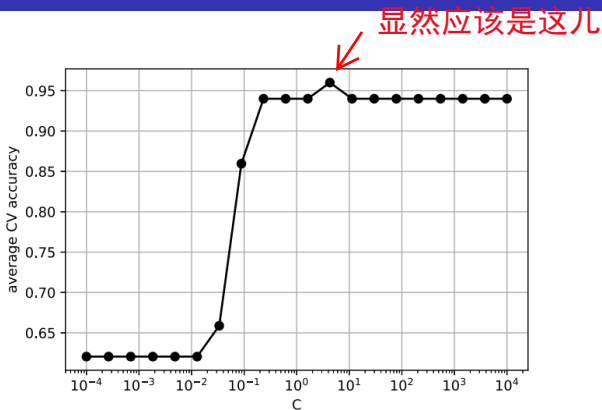
Final Accuracy = Average(Round 1, Round 2, ...)

Cross-Validation

■ Procedure

- 1 Select a range of C values to try
- 2 Repeat K rounds
 - 1 Split the training set into training data and validation data
 - 2 Learn a classifier for each value of C
 - 3 Record the accuracy on the validation data for each C
- 3 Select the value of that has the highest average accuracy over all K rounds
- 4 Retrain the classifier using all data and the selected C

Which C to Select?

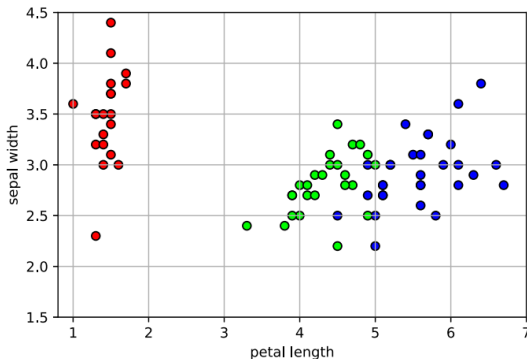


Multiclass Classification

- So far, we have only learned a classifier for 2 classes $\{-1, 1\}$
 - Called a **binary classifier**
- For more than 2 classes, split the problem into several binary classifier problems
- **One-vs-rest** 可以翻译成“一对多”，对应意，多意
 - *Training*: for each class, train a classifier for that class versus the other classes
 - For example, if there are 3 classes, then train 3 binary classifiers: 1 vs $\{2, 3\}$; 2 vs $\{1, 3\}$; 3 vs $\{1, 2\}$
 - *Prediction*: calculate probability for each binary classifier. Select the class with the highest probability

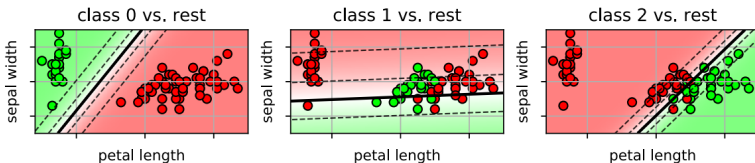
Example on 3-Class Iris Data

■ $\mathcal{Y} = \{\text{“setosa”, “versicolor”, “virginica”}\}$

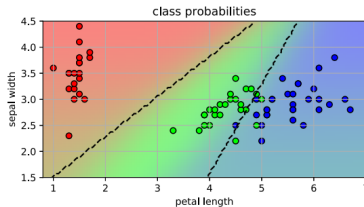


Example on 3-Class Iris Data

■ The individual 1-vs-rest binary classifiers



■ The final classifier, combining all 1-vs-rest classifiers



Multiclass Logistic Regression

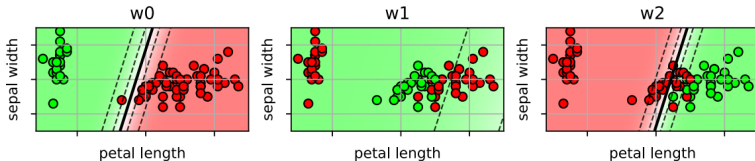
- Another way to get a multi-class classifier is to **define a multi-class objective**
 - One weight vector \mathbf{w}_c for each Class c
 - We omit the bias b_c for each class for notation simplicity
- Define probabilities with **softmax** function
 - Analogous to sigmoid function for binary logistic regression

$$p(y = c|\mathbf{x}) = \frac{\exp(\mathbf{w}_c^T \mathbf{x})}{\exp(\mathbf{w}_1^T \mathbf{x}) + \cdots + \exp(\mathbf{w}_C^T \mathbf{x})}$$

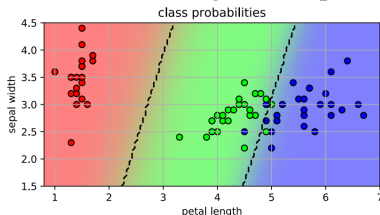
- The class with the largest response of $\mathbf{w}_c^T \mathbf{x}$ will have the highest probability
- Estimate the $\{\mathbf{w}_c\}_{c=1}^C$ parameters using MLE as before

Multiclass Logistic Regression on 3-Class Iris Data

- The “binary” classifiers based on individual weight vectors



- Individual weight vectors work together to partition the space



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**

分段线性决策边界

Logistic Regression Versus NB and LDA

- Speed: Learning logistic regression requires iterative numerical optimization, which will be slower than NB and LDA
- Storage: The model requires $O(N)$ parameters, the same order as NB, but much less than LDA's $O(N^2)$
- Interpretability: The “importance” of feature x_j can be understood in terms of the corresponding learned weight w_j

Logistic Regression Summary

■ Classifier:

- Linear function $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$
- Given a feature vector \mathbf{x} , the probability of a class is
 - $p(y = +1|\mathbf{x}) = \sigma(f(\mathbf{x})), p(y = -1|\mathbf{x}) = 1 - \sigma(f(\mathbf{x}))$
 - Sigmoid function: $\sigma(z) = \frac{1}{1+e^{-z}}$
- Logistic loss function: $L(z) = \log(1 + \exp(-z))$ z 就是代入 $f(\mathbf{x})$

■ Training:

- Maximize the likelihood of the training data
- Use regularization to prevent overfitting
 - Use cross-validation to pick the regularization hyperparameter C

■ Classification:

- Given a new sample \mathbf{x}^* , pick class with the highest $p(y|\mathbf{x}^*)$

$$y^* = \begin{cases} +1, & \frac{p(y=+1|\mathbf{x}^*)}{p(y=-1|\mathbf{x}^*)} > 1 \\ -1, & \text{otherwise} \end{cases} \quad \text{or } y^* = \begin{cases} +1, & f(\mathbf{x}^*) > 0 \\ -1, & \text{otherwise} \end{cases} = \text{sign}(f(\mathbf{x}^*))$$