Pattern Recognition

Lecture 16. Linear Discriminant Functions: Logistic Regression Classifier

Dr. Shanshan ZHAO

shanshan.zhao@xjtlu.edu.cn

School of AI and Advanced Computing

Xi'an Jiaotong-Liverpool University

Academic Year 2021-2022

Table of Contents

Plan

- week5 day1
  - practical use of Classifiers : perception, MSE
  - Logistic Regression Classifier
- week5 day2: SVM
- week5 day3: Dimensionality reduction and feature selection
- week5 day4: Clustering
- week6 day1: Algorithm independent topics: feature normalization, cross validation, etc.

Perception

Perception criterion function

$$J_p(a) = \sum_{y \in Y_M} (-a^T y)$$

The gradient decent batch update rule for $J_p(a)$ is

$$a^{(k+1)} = a^{(k)} + \eta^{(k)} \sum_{y \in Y_M} y \tag{1}$$

```
from sklearn.datasets import load_digits
from sklearn.linear_model import Perceptron
X, y = load_digits(return_X_y=True)
clf = Perceptron(tol=1e-3, random_state=0)
clf.fit(X, y)
print(clf.score(X, y))
```

code/perception.py

sample codes

- https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html#sklearn.linear_model.Perceptron
- https://scikit-learn.org/stable/auto_examples/linear_model/plot_sgd_comparison.html#sphx-glr-auto-examples-linear-model-plot-sgd-comparison-py
- https://scikit-learn.org/stable/auto_examples/text/plot_document_classification_20newsgroups.html#sphx-glr-auto-examples-text-plot-document-classification-20newsgroups-py

MSE

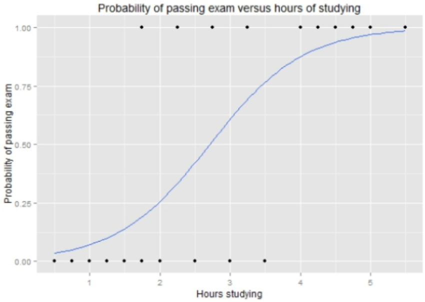
How to use MSE methods to classify samples?

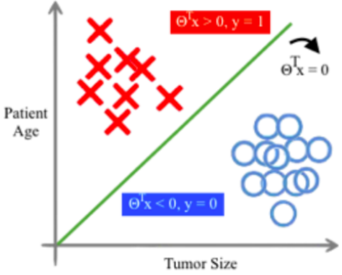
Linear classifiers	
`linear_model.LogisticRegression`([penalty, ...])	Logistic Regression (aka logit, MaxEnt) classifier.
`linear_model.LogisticRegressionCV`(*[, Cs, ...])	Logistic Regression CV (aka logit, MaxEnt) classifier.
`linear_model.PassiveAggressiveClassifier`(*)	Passive Aggressive Classifier.
`linear_model.Perceptron`(*[, penalty, alpha, ...])	Linear perceptron classifier.
`linear_model.RidgeClassifier`([alpha, ...])	Classifier using Ridge regression.
`linear_model.RidgeClassifierCV`([alphas, ...])	Ridge classifier with built-in cross-validation.
`linear_model.SGDClassifier`([loss, penalty, ...])	Linear classifiers (SVM, logistic regression, etc.) with SGD training.
`linear_model.SGDOneClassSVM`([nu, ...])	Solves linear One-Class SVM using Stochastic Gradient Descent.

`https://scikit-learn.org/stable/modules/linear_model.html`

Logistic Regression Classifier

Using a Linear regression equation to produce discrete binary output.





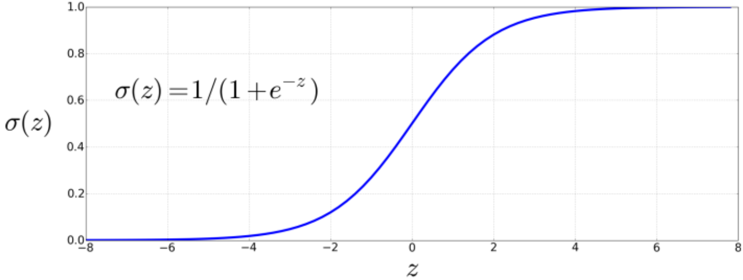
Components of Logistic Regression Classifier

- A feature representation of the input.
- A classification function that computes $\hat{y}$, the estimated class.
- An objective function for learning.
- An algorithm for optimizing the objective function.

Decision/Activation Function

$$z = w \cdot x + b$$

The **sigmoid function** $\sigma(z) = \frac{1}{1+e^{-z}}$ takes a real value and maps it to the range $[0, 1]$. It is nearly linear around 0 but outlier values get squashed toward 0 or 1.



$$\text{decision}(x) = \begin{cases} 1 & \text{if } P(y=1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Example: sentiment classification

Let's have an example. Suppose we are doing binary sentiment classification on movie review text, and we would like to know whether to assign the sentiment class $+$ or $-$ to a review document *doc*. We'll represent each input observation by the 6 features $x_1 \ldots x_6$ of the input shown in the following table; Fig. 5.2 shows the features in a sample mini test document.

Var	Definition	Value in Fig. 5.2
x_1	count(positive lexicon words $\in$ doc)	3
x_2	count(negative lexicon words $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(66) = 4.19$

Example: sentiment classification(cont.)

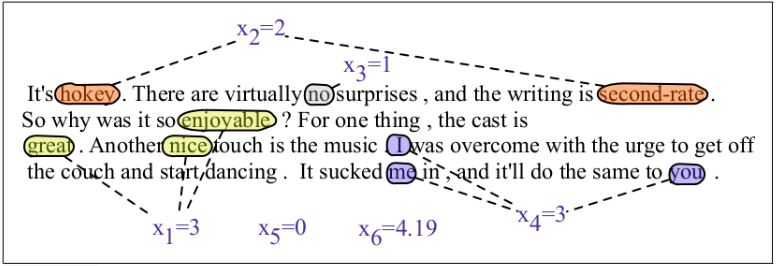


Figure 5.2 A sample mini test document showing the extracted features in the vector *x*.

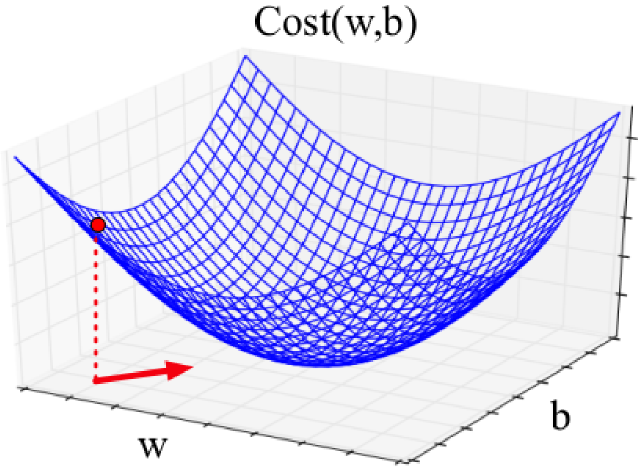
Cross Entropy Loss Function

We need a loss function that expresses, for an observation x, how close the classifier output ($\hat{y} = \sigma(w \cdot x + b)$) is to the correct output (y, which is0 or 1). We'll call this:

$$L(\hat{y}, y) = \text{How much } \hat{y} \text{ differs from the true } y$$

cross-entropy loss ?

Gradient Descent



Logistic regression has two phases:

- **training:** we train the system (specifically the weights w and b) using **stochastic gradient descent** and the cross-entropy loss.
- **test:** Given a test example x we compute p(y|x) and return the higher probability label y = 1 or y = 0.

Reference I

Thank You !

$\mathcal{Q}$ & $\mathcal{A}$