机器学习（L08）：有监督分类

授课教师：王贝伦 / 助教：张嘉琦，黄旭，谈笑，徐浩卿

在结束了监督学习中回归问题的学习之后，我们将翻开新的篇章——分类问题。分类问题同样是监督学习的一个核心问题。在监督学习中，当输出变量 Y 取有限个离散值的时候，预测问题便成为分类问题。这时，输入变量 X 可以是离散的，也可以是连续的。监督学习从数据中学习一个分类模型或分类决策函数，称为分类器（classifier）。根据分类的类别个数不同，我们有二分类问题，也有多分类问题以及其他经典问题，我们将在下面的内容中逐一介绍。

1 二分类（Binary Classification）

二分类问题指输出变量 Y 值域中仅含两个离散值，在二分类的所有分类器中，感知机（Perception），即二类分类的线性分类模型最为人们所熟知。

1.1 感知机（Perception）

定义：感知机

假设输入空间（特征空间）是 $\mathcal{X} \in \boldsymbol{R}^n$，输出空间是 $\mathcal{Y} = \{+1, -1\}$。输入 $x \in \mathcal{X}$ 表示实例的特征向量，对应于输入空间（特征空间）的点；输出 $y \in \mathcal{Y}$ 表示实例的类别。由输入空间到输出空间的如下函数：

$$f(x) = sign(w \cdot x + b) \tag{1}$$

称为感知机。其中，w 和 b 为感知机模型参数，$w \in \boldsymbol{R}^n$ 叫作权值（weight）或权值向量（weight vector），$b \in \boldsymbol{R}$ 叫作偏置（bias），$w \cdot b$ 表示 w 和 b 的内积。sign 是符号函数，即

$$sign(x) = \begin{cases} +1, x \geq 0 \\ -1, x < 0 \end{cases} \tag{2}$$

在假设训练数据集是线性可分的前提下，感知机学习的目的是求得一个能够将训练集正实例点完全正确分开的分离超平面。为了找出这样的超平面，即确认感知机的模型参数 w，b，需要确定一个学习策略，即定义损失函数并将损失函数极小化。

在这种情况下，我们很自然的一个想法是，将误分类的点的总数作为损失函数。但是，这样的损失函数并不是参数 w 和 b 的可导函数，不易优化。所以感知机所采用的的损失函数是误分类点到超平面 $\mathcal{S}$ 的总距离。

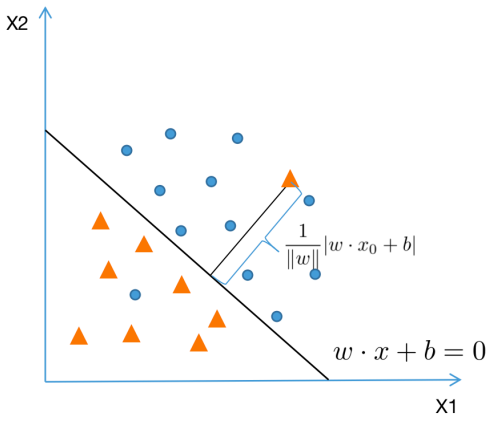

Figure 1: 感知机模型

由图我们可以写出输入空间中任一点 x_0 到超平面 $\mathcal{S}$ 的距离：

$$\frac{1}{\|w\|}|w \cdot x_0 + b| \tag{3}$$

在一般情况下，如果 $\|w\|$ 不加以说明，一般默认是 L_2 范数。

对于误分类的数据 (x_i, y_i) 来说，因为当 $w \cdot x_i + b > 0$ 时，$y_i = -1$；而当 $w \cdot x_i + b < 0$ 时，$y_i = +1$，所以

$$-y_i(w \cdot x_i + b) > 0 \tag{4}$$

假设超平面 $\mathcal{S}$ 的误分类点集合 $\mathcal{M}$，那么所有误分类点到超平面 $\mathcal{S}$ 的总距离为

$$-\frac{1}{\|w\|}\sum_{x_i \in \mathcal{M}} y_i(w \cdot x_i + b) \tag{5}$$

$\frac{1}{\|w\|}$ 对优化损失函数没有影响，可以不予考虑，遂我们得到了感知机学习的损失函数，即

$$L(w, b) = -\sum_{x_i \in \mathcal{M}} y_i(w \cdot x_i + b) \tag{6}$$

显然，我们得到的损失函数是非负的。如果没有误分类点，那么损失函数的值就是 0。在有误分类时，损失函数是参数 w 和 b 的线性函数，此时的损失函数是连续可导的。

1.2 感知机学习算法

感知机学习的具体算法包括两种形式，即原始形式和对偶形式。在这里我们仅介绍原始形式，因感知机学习常被用来在线学习，所以我们采用之前学到的 SGD 算法进行优化。

给定一个训练数据集

$$T = (x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$$

其中，$x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}, i = 1, 2, ..., N$，我们要解决的优化问题为

$$\min_{w,b} -\sum_{x_i \in \mathcal{M}} y_i(w \cdot x_i + b) \tag{7}$$

© 2020 本课件仅限东南大学机器学习课程使用，版权属于相关作者，严禁传播。

假设误分类点结合 $\mathcal{M}$ 是固定的，那么损失函数 $L(w,b)$ 的梯度由

$$\nabla_w L(w,b) = -\sum_{x_i \in \mathcal{M}} y_i x_i \tag{8}$$

$$\nabla_b L(w,b) = -\sum_{x_i \in \mathcal{M}} y_i \tag{9}$$

给出。这样我们可以写出感知机学习算法的原始形式的伪代码。

Algorithm 1: 感知机学习算法的原始形式

输入：**训练数据集** $T = (x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$，**其中** $x_i \in \mathcal{X} = \boldsymbol{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}$，$i = 1, 2, ..., N$；**学习率** $\alpha (0 < \alpha \leq 1)$；**选取初值** w_0, b_0；
for $i \leftarrow 1$ *to* N **do**
  在训练集中选取数据 (x_i, y_i)；
  if $y_i(w \cdot x_i + b) \leq 0$ **then**
    $w \leftarrow w + \alpha y_i x_i$
    $b \leftarrow b + \alpha y_i$
  end
  if 训练集中没有误分类点 **then**
    break;
  end
end
输出：**预测系数** w，b。

2 多分类（Multi-class Classification）

多分类问题是指输出变量 Y 可取的离散值达到三个及三个以上，这是它与二分类的主要区别。

文本分类（Text Categorization）就是多分类应用的一个例子。文本分类根据文本的特征将其划分到已有的类中。输入的是文本的特征向量，输出的是文本的类别。通常把文本中的单词定义为特征，每个单词对应一个特征。**单词的特征可以是二值的，如果单词在文本中出现则取值为 1，否则为 0。单词的特征也可以是多值的，表示单词在文本中出现的频率**。并以此来构成输入的文本特征向量。直观地，如果“股票”“银行”“货币”这些词出现很多，这个文本很有可能属于经济类；如果“网球”“比赛”“运动员”这些词频繁出现，那这个文本很可能属于体育类。

3 层次分类 (Hierarchical Classification)

层次分类同样是解决多类分类问题的一个常用方法。

关于层次分类器，其定义一般包括类别层次和分类器两部分。类别层次决定多类问题的分解，层级分类器在类别层次的基本上进行组织。类别层次一般使用“树”或者“有向无环图”结构存储。而在层级分类器构建方面，一般有两种形式，第一种形式如图 2，从根节点起，每个中间节点都对应一个分类器，每个分类器所解决的分类问题由当前节点的“类别划分冶决定，分类时是否走向一个节点由其父节点所处的分类器决定。而对图 3 对应的分类器来说，除根节点外，每个节点都对应一个分类器，每个分类器所解决的分类问题由其父节点的“类别划分冶决定，分类时是否走向一个节点由这个节点所拥有的分类器结果及其兄弟节点所拥有的分类器结果共同决定。

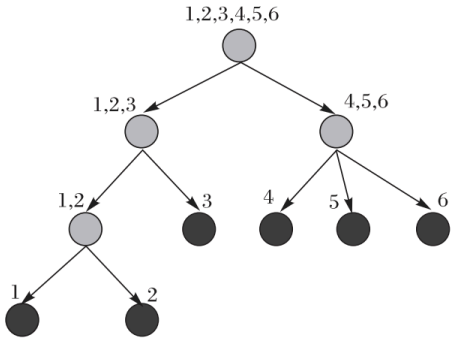

Figure 2: 树形结构

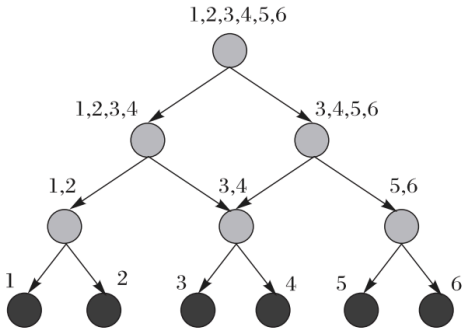

Figure 3: DAG 结构

4 多标签分类（Multi-label Classification）

在机器学习中，多标签分类与多分类（Multi-class Classification）密切相关，在多分类中每个实例只有一个标签，但是在多标签分类中，每个实例可以被分得多个标签，它由两个以上的单标签问题组成。在实际应用中，多标签问题可以用来解决图像分类、物体识别的问题。事实上，现实生活中的一张图片中往往包含多个类别的物体。对于多标签问题，我们主要的两个思路，一个是将问题转为多个单标签分类问题，还有一个是其他适应性算法，例如 Multi-label Decision 和 Ranking SVM。

在图像识别中有一种被称为“视觉词袋（bag of visual words）”的方法。视觉词袋的主要思想是将图像表示为一组特征。特征包含“关键点（keypoints）”和“描述符 (despriptors)”。关键点是图像中较为突出，更有特色的点，因此，无论图像旋转缩小还是放大，关键点始终相同。之所以取名叫做视觉词袋，也是因为在图像识别中，这些关键点就相当于在文本分类中“单词”一样。我们用关键点和描述符构建出了词汇表，并将每个图像表示为图像中特征的频率直方图。根据得到的频率直方图，我们可以找到相似的图像或者预测图像类别。

© 2020 本课件仅限东南大学机器学习课程使用，版权属于相关作者，严禁传播。

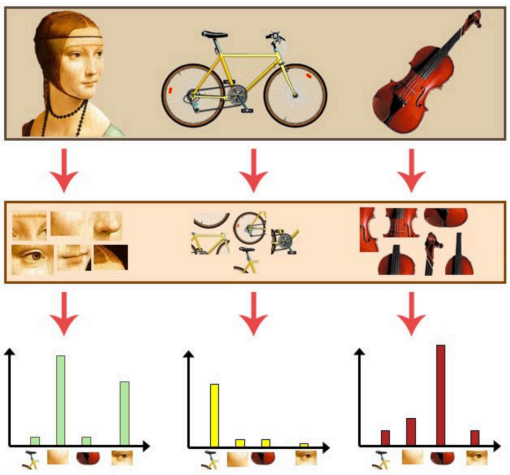

Figure 4: 视觉词袋

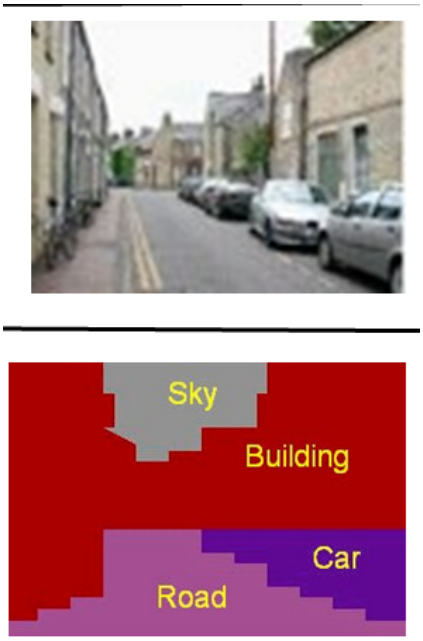

Figure 5: 图片元素识别任务。

5 结构化输出 (structuralOutput)

有一些分类问题还具有结构化的输出。在这类问题中，我们需要预测的不再是连续或离散的值，而是具有结构的“物体”。在实际应用中我们可能会碰到各种各样的结构化输出。在这里我们只讨论其中常见的一部分。

5.1 序列化输出

序列化输出通常出现在一些具有序列化输入的应用中。比如在自然语言处理领域中，为一句话中的词语确定词性是一件很基础也很必要的任务。利用序列化的输入（即一句话），一个机器学习分类模型应该为这句话中的每一个词语打上表示词性的标签，输出由标签组成的序列，这类问题被称为词性标注问题。假设我们有一个句子

我	爱	中国
代词	动词	名词

我们需要正确地为每一个词标注词性。因为一句话的结构或者顺序会影响到词性的标注，所以显然输出序列中每个标签是会相互影响的。具有序列化输出的问题需要我们设计出的机器学习模型具有挖掘序列结构的能力。

5.2 网格输出

网格式输出经常出现在图像分析和视频分析相关应用中。比如，在图像分析中，一个经典的问题是对图像中的不同元素进行识别与划分。图5就是这种问题的一个例子。对于一张图片，我们需要监测出“建筑”，“路”，还有“汽车”等区域。由于图片是网格化的数据，这类问题对应的输出也是网格化的。

5.3 关系型数据

6 有监督分类

在机器学习领域中，通常我们会使用三种方法解决有监督问题：判别方法，生成方法，和基于样本的分类器。在本节讲义中我们只对这几种方法作简单的介绍。详细的模型与算法解释会在之后的课程中给出。

6.1 判别法

判别法直接预测出不同类数据的边界。利用这个边界我们能够很容易地将数据划分到不同类中。比如，在图6中，红色与蓝色的点表示两类不同数据。判别法就是通过模型预测出这两类数据的边界，即图中黑色虚线部分。有了数据边界，当我们有新的数据时，我们就可以根据新数据在数据边界的哪一边来对数据进行分类。有很多种模型都是利用判别法来进行分类的，比如 Logistic 回归和感知向量机（SVM）。之后的课程中我们会对这些方法进行详细解释。

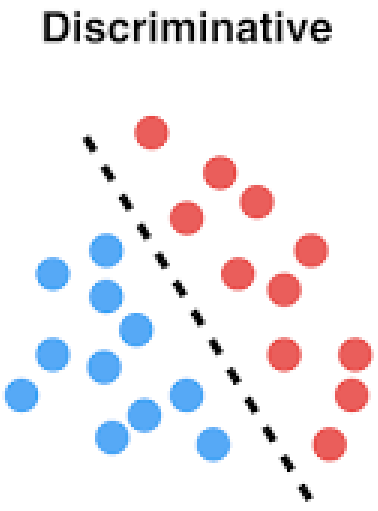

Figure 6: 判别法示例。

© 2020 本课件仅限东南大学机器学习课程使用，版权属于相关作者，严禁传播。

6.2 生成法

生成法与判别法不同，生成法假设不同类的数据样本产生于不同的概率分布。当我们需要分类一个新的数据时，我们需要看这个数据更有可能是由哪个概率分布产生的。我们从概率的角度来简单理解一下生成法。假设我们有数据 X 和对应的类别标签 Y。我们共有两类数据，所以每个标签的取值为 $Y_i = 0$ 或 $Y_i = 1$。生成法假设不同类的样本生成于不同的概率分布 $P(Y = 0|X)$ 或者 $P(Y = 1|X)$。我们需要先根据数据来预测这两种分布，然后对于新的数据 $\tilde{x}$，我们比较两个概率的大小。如果 $P(Y = 0|\tilde{x})$ 更大，则新的数据属于 $Y = 0$ 这一类，否则属于 $Y = 1$ 这一类。

图7是一个二分类的例子。我们假设不同类的样本产生于不同的概率分布（红色的椭圆区域和蓝色的椭圆区域）。对于一个新的数据，我们需要计算这个数据落在不同区域的概率，数据就属于概率较高的那一类。朴素贝叶斯方法和贝叶斯网络就属于这一类分类方法。

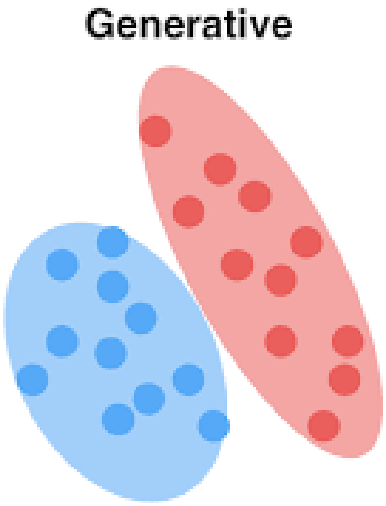

Figure 7: 生成法示例。

6.3 基于样本的分类器

基于样本的分类器直接利用样本数据来进行分类。一个最简单的方法就是，针对一个新的样本数据 A，我们在已有的数据中找到一个离新数据最近的那个数据 B，那么我们就认为数据 A 和 B 属于同一类。衡量样本的距离可以根据数据类型不同来具体选择，最普通的一种算法就是几何距离。由于每次只考虑一个最近点可能会存在误差，我们可以使用更好地算法：K-最近邻算法。K-最近邻算法对每个新数据检索到 K 个最近的数据点，然后综合考虑这些数据的标签来对新数据进行分类。图8是使用 K-最近邻算法的一个例子，我们选择 $K = 5$。可以看到在 5 个离新数据最近的数据中，属于类别 B 的样本最多，那么我们可以预测新样本数据的类别标签就是 B。严格来说，这一类算法其实并不属于机器学习算法。以 K-最近邻算法为例，我们并不是在学习一个模型，而是利用已有的数据来完成分类。这本质上不是一个“学习”过程，而是一个“数据检索”过程。然而，这些算法有时会有令人惊讶的分类效果，而且算法的成本较低。所以这些算法在一些应用也会有广泛应用。

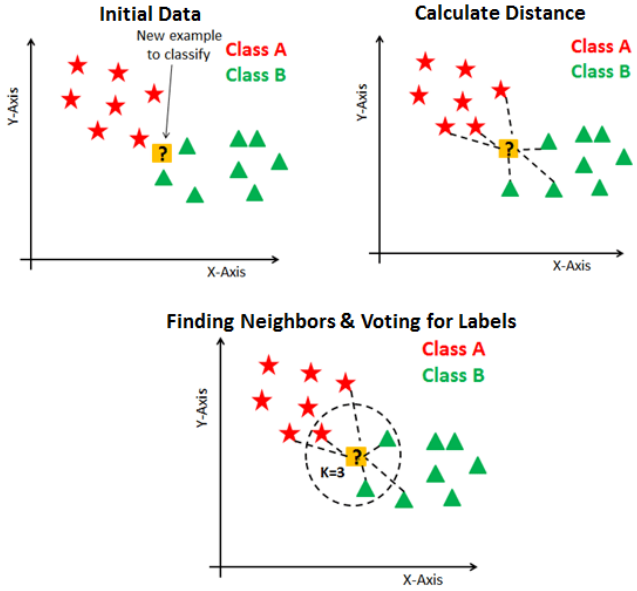

Figure 8: K-最近邻算法示例。

7 分类效果指标

最后我们还需要介绍一下在分类问题中，评估分类效果常用的几个指标。这里我们以二分类问题（及只有两个类别，这里我们定义为正类和负类）为例。之后的课程中我们也是以二分类问题为主来介绍。

对于一个样本数据，假设我们用 y^* 表示这个数据真正的类别标签，而用 $\widehat{y}$ 表示我们预测的样本标签。那么预测结果可能存在四种可能

	$y^* = +$	$y^* = -$
$\widehat{y} = +$	TP	FP
$\widehat{y} = -$	FN	TN

其中：

- **TP**：表示真正例（true positive），即数据真实的标签和预测的标签都是 $+$。
- **FP**：表示假正例（false positive），即我们将一个本应该属于负类的样本错误的划分成了正类。
- **FN**：表示假负例（false negative），即我们将一个本应该属于正类的样本错误的划分成了负类。
- **TN**：表示真负例（true negative），即我们正确预测了一个属于负类的数据。。

对于多个数据，我们通常会判断每个数据对应上述四种情况的哪一种，然后统计在整个数据集中 TP，FP，TN 以及 FN 的数量。可以看到，这里我们不仅考虑了正确分类的情况，也考虑了分类错误的情况。

下一步，我们就可以用这些指标来计算预测结果的准确度了，具体有以下几个常用的算法，每种计算方法关注的重点也不同：

- **准度（accuracy）**：$\frac{TP+TN}{TP+TN+FP+FN}$。体现整体的预测准确度。

© 2020 本课件仅限东南大学机器学习课程使用，版权属于相关作者，严禁传播。

- **精度（precision）**：$\frac{TP}{TP+FP}$。对于预测为正类的数据的预测准确度。
- **回归率（recall）**：$\frac{TP}{TP+FN}$。对于真正为正类的数据的预测准确度。
- **特异性（specificity）**：$\frac{TN}{TN+FP}$。对于真正为负类的数据的预测错误率。
- **F1 值（F1 score）**：$\frac{2TP}{2TP+FP+FN}$。对于不平衡类别的预测衡量方法。

引用 [5] 中给出了其他很多计算分类预测结果准确性的方法。

7.1 编程实现

下面我们用 Python 实现一下上述的几种预测准确度计算方法。

```python
import numpy as np

# 随机生成标签与预测结果。这里数据没有任何关联性，只是为了展示计算方法而生成的。
N = 100
Y = np.random.uniform(0, 1, size = (N,))
pred_Y = np.random.uniform(0, 1, size = (N, ))
# 对生成的数据进行四舍五入。这是为了只得到两种类标签，即0和1。
# 在之后，我们将1看作正类，而0看作负类。
Y = np.round(Y)
pred_Y = np.round(pred_Y)

# 计算TP, FP, TN和FN
def countNum(Y, pred_Y):
    TP = np.sum(
        np.logical_and(pred_Y == 1, Y == 1)
    )
    FP = np.sum(
        np.logical_and(pred_Y == 1, Y == 0)
    )
    TN = np.sum(
        np.logical_and(pred_Y == 0, Y == 0)
    )
    FN = np.sum(
        np.logical_and(pred_Y == 0, Y == 1)
    )
    return (TP, FP, TN, FN)

TP, FP, TN, FN = countNum(Y, pred_Y)
# 计算准确率（accuracy）
accuracy = (TP + TN) / (TP + TN + FP + FN)
# 计算精度（precision）
precision = TP / (TP + FP)
# 计算回归率（recall）
recall = TP / (TP + FN)
# 计算特异性（specificity）
specificity = TN / (TN + FP)
# 计算F1值（F1 score）
F1 = (2 * TP) / (2 * TP + FP + FN)
# 输出计算结果
print("Accuracy为：{}".format(accuracy))
print("Precision为：{}".format(precision))
print("Recall为：{}".format(recall))
print("Specificity为：{}".format(specificity))
print("F1 score为：{}".format(F1))

# Output：
#    Accuracy为：0.45
#    Precision为：0.4423076923076923
#    Recall为：0.46938775510204084
#    Specificity为：0.43137254901960786
#    F1 score为：0.45544554455445546
```

因为我们的数据是随机生成的，这里计算出的准确度没有任何实际意义。

引用

[1] Structured Prediction: https://en.wikipedia.org/wiki/Structured_prediction.

[2] Discriminative Model: https://en.wikipedia.org/wiki/Discriminative_model.

[3] Generative Model: https://en.wikipedia.org/wiki/Generative_model.

[4] K Nearest Neighbor Algorithm: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.

[5] Precision and Recall: https://en.wikipedia.org/wiki/Precision_and_recall.

[6] Hang Li "Statistical Learning"

[7] Yeting Lu & Jianfeng Lu & Jingyu Yang from NUST "Summary of Hierarchical Classification Method"

[8] about BOVW:https://towardsdatascience.com/bag-of-visual-words-in-a-nutshell-9ceea97ce0fb

© 2020 本课件仅限东南大学机器学习课程使用，版权属于相关作者，严禁传播。