机器学习课程实验报告：监督机器学习算法及其应用

213106000057 蒋效会

1、实验目的

实现 Logistic 回归等监督机器学习算法，并将之应用于文本情感分析问题。

2、实现环境

实验主要以 matlab2012 作为实验环境，编程语言为 matlab。

硬件环境为普通 PC 机，CPU 为 Intel I3，内存 4G。操作系统为 Windows 8。

3、实验内容

本次实验实现了以下内容：

1 Logistic 回归算法

2 SVM 算法（直接调用）

3 Naïve Bayes 算法

4 NBEM（无监督学习）

5 PCA（主成分分析）

6 k-fold 交叉验证

7 TF, BOOL, TF-IDF 单词加权模型

对文本情感分类问题，所用文本表示模型为Bag-of-words(BOW)模型亦即 Vector Space Model (VSM)。词加权方法主要使用 BOOL 模型，另外还测试了词频模型（TF）、词频-逆文档频率模型（TF-IDF）。特征提取利用信息增益率对单词的分类相关性进行排序，选取一定数量的（130 个）增益率最高的高区分度词汇作为特征进行训练和分类。算法性能的评价使用了 holdout 验证和 k-fold 交叉验证。

4、实验背景与数据

随着互联网的发展，出现了大量观点评论数据。 随着互联网的发展，出现了大量观点评论数据。 他人的想法对于我们做出决策有着很重要的作用。情感文本分类是挖掘出文本中表达的情感信息（褒或贬， Positive or Negative），帮助我们决策。

所用数据集是携程网提取的宾馆评论文本数据（已分词）， 其中包含 2000 篇褒类 （positive）评论 和 2000 篇贬类 （negative）评论 。数据集文件夹 dataset 里，包含两个文件，每类别一个文件， 文件名为对应的类别标签，每个文件各包含 2000 篇评论 文本 （以<text> < \text>为分隔符）。

4、实验过程与程序说明

4.1 文本预处理

首先从原始数据文件读取评论文本，结果为一系统单词的列表；然后以<text></text>为分隔符号，把这些文本分组，保存为 cell 数组，每组为一个样本。用到的函数为：

- out = readText(fn): fn 为文件名，out 为输出的二维 cell 数组，内容为单词字符串

然后生成词袋（词汇表），将所有单词形成一个集合并去重。所用函数为：

- bag = genBag(t)：t 为正类和负类的 cell 数组，bag 为所有非重复单词构成的 cell 数组

第三步，将单词数组转化为特征向量。先查找每个样本的所有单词在词汇表 bag 中的索引，然后对重复的单词进行计数，得到 TF 模型的特征向量构成的矩阵；将词频数大于 0 的置为 1，否则为 0，得到 BOOL 模型的特征矩阵。用到的函数为：

- idx = indexText(s, bag)：bag 为词汇表，s 为样本的单词 cell 数组，输出 idx 为每个单词的索引
- f = toFeature(idx, len)：idx 为索引向量，len 为词汇表长度，输出 f 为 TF 向量，该函数对一个本进行转换
- fea = toFeatures(idxs, len)：idx 为索引矩阵，len 为词汇表长度，输出 fea 为 TF 矩阵

实验中发现，使用 TF 模型和 BOOL 模型对分类结果影响差别不大，BOOL 相对稍好，所以后面的实验中主要采用了 BOOL 模型。

4.2 特征提取和降维

4.2.1 特征提取

上一步生成的词汇表包含 15846 个词，生成的特征矩阵 Xorigin 为 4000 行 15846 列，包括正负样本，每行为一个样本，第列代表一个特征词。显然这个矩阵特征维度太大，必须对它进行特征提取或降维。特征提取使用了信息增益率（Information gain ratio，IGR）作为特征（单词）对分类的重要性的评价准则：

$$IGR(t_i) = \frac{IG(t_i)}{H(t_i)}$$

其中 $IG(t_i)$为第 i 个特征 t_i 的信息增益（Information gain），$H(t_i)$ 为 t_i 的信息熵：

$$IG(t_i) = \{-\sum_{j=1}^{c} P(c_j)\log P(c_j)\} + \{P(t_i)[\sum_{j=1}^{c} P(c_j \mid t_i)\log P(c_j \mid t_i)] + P(\bar{t_i})[\sum_{j=1}^{c} P(c_j \mid \bar{t_i})\log P(c_j \mid \bar{t_i})]\}$$

$$H(t_i) = -\sum P(t_i)\log P(t_i)$$

计算信息增益率用到的函数为：

- e = infoEntropy(x, isbool)：x 为要计算的特征向量或目标向量（列向量），isbool 参数表示数据是否 BOOL 类型
- r = infoGain(X, y, j)：X 为特征矩阵，y 为目标向量（类标签），j 指定对哪个特征求 IGR，j=0 表示求所有列的 IGR

  求出每一列对应的特征对应的 IGR 得到一个向量 ig，对它按从大

到小排序，：

```
[ig1 idxf] = sort(ig, 'descend');
```

然后选择 IGR 最大的若干列作为分类特征：

```
nSel = 130;
selfea = idxf(1:nSel)';
X = Xorigin(:, selfea);
```

选择的特征对应的词汇为（只列出 IGR 值最大的一部分）：

'不错'，'差'，'恶劣'，'一流'，'答应'，'结算'，'上当'，'冷冰冰'，'笑'，'折腾'，'奉劝'，'旅馆'，'烂'…

这些词从语义上来看，确实对情感分类具有很高的区分度，说明用 IGR 作为特征选择依据具有较强的合理性。实验时尝试了采用信息增益（IG）作为排序依据，但选出的特征词含有大量没有区分度的词，识别结果很差，因此改用信息增益率为排序依据。

实验过程中发现，选择的特征词越多，分类精度越高，但综合考虑算法运行时间和性能，选择了 130 个特征进行分类。

上面得到的矩阵 X 即为分类用到的特征矩阵，类标签为 y，(X, y) 即样本集，下面为进行对样本集进行划分，比例为训练集占 70%共 2800 个，测试集占 30%共 1200 个：

```
part = cvpartition(y, 'holdout', 0.3);
istrain = training(part); istest = test(part);
Xtr = X(istrain, :); ytr = y(istrain);
Xte = X(istest, :); yte = y(istest);
```

后面的算法默认都采用此配置的数据集进行测试。

4.2.1 数据降维

上述特征提取过程实际上就实现了数据维度的缩减，除此之外，还尝试了使用主成分分析方法（PCA）进行降维。PCA 实现直接调用了 matlab 工具箱的 pcacov 函数：

```
covx = cov(X);
[COEFF, latent, explained] = pcacov(covx);
X1 = X * COEFF;
covX1 = cov(X1)
dg1 = diag(covX1);
sum(dg1(1:34)) / sum(dg1)
```

经过变换以后的特征矩阵为 X1，它的协方差矩阵为对角阵，观察发现协方差矩阵的前 34 个特征向量大概占 80.67%的比例，故选择 X1 的前 34 列作为特征进行分类，采用 Logistic 方法：

```
[theta, cost] = logisticR(X2(istrain), ytr, lambda);
p = logisticRpredict(theta, X2(istest));
```

得到精度为 79%；而总耗时仅 0.14 秒，相比直接用 130 个特征训练 Logistic 回归的用时 21.65 秒，时间加快了约 150 倍；若直接采用原始特征的前 34 个，得到的精度只有 73.91%；有此可见使用 PCA 对数据进行降维可以有效的缩短算法运行时间，提高分类精度。当然，实际使用中选择多少个特征进行降维，降维以后选择多少个特征，还

需要进一步研究。

4.3 训练和分类

4.3.1 Logistic 回归

模型训练：

```
theta= logisticR(Xtr, ytr, lambda);
```

其中 lambda 为正则化因子，返回值 theta 为模型参数。

分类：

```
p = logisticRpredict(theta, Xte);
```

返回值 p 为分类结果。将之与类标签向量 yte 作比较就得到分类精度：

```
mean(double(p == yte)) * 100
```

实验中选择 130 个特征，正则化因子为 lambda=0 时，得到测试分类精度为 83.25%；lambda=0.05 时，精度为 83.17%；lambda=0.1 时，精度为 83.08%；lambda=0.5 时，精度为 82.92%；由此可见，对该问题正则化会降低测试精度。但从训练时间上来看，采取适量正则化会显著加快算法收敛速度。

4.3.2 Naïve Bayes

由于数据为 BOOL 类型，所以算法采取 Multi-variate Bernoulli event 模型。

模型训练：

```
nb = mynaivebayes(Xtr, ytr, 0);
```

返回结果为结构体对象，包含算法训练得到的两组参数 pi 和 theta。第 3 个参数表示是否进行 Laplace 平滑。

分类：

```
[p err] = mynaivebayesPredict(nb, Xte, yte);
```

实验中选择 130 个特征，采用 Laplace 平滑，得到分类精度为 80 %；不使用 Laplace 平滑，得到分类精度为 80.25 %。

4.3.3 SVM

SVM 算法的实现中直接调用了 matlab 工具箱的 svmtrain/svmclassify 函数。代码如下：

```
svmStruct = svmtrain(double(X), y, 'Kernel_Function', …
    'rbf', 'autoscale', 'false');
p = svmclassify(svmStruct, double(Xte));
```

参数的选择：

因为此例数据为 BOOL，不适合进行缩放，故 autoscale 参数必须设为 false；Kernel_Function 参数的选择对结果影响不大；优化方法 method 选择 SMO 速度最快，QP 最慢，但对分类结果没有影响。

实验中选择 130 个特征，svmtrain 得到的分类精度为 82.08 %，支持向量个数 1199。

4.3.4 EMNB

为了和监督学习方法作对比，本实验还实现了无监督学习算法EMNB（Expectation Maximization for Naïve Bayes）：

```
nb = myemnb(X, 100);
[prob, pred] = myemnbPredict(nb, X);
```

训练和分类都使用全部样本。myemnb 为训练过程，迭代 100 次，返回 nb 结构体（和 Naïve Bayes 相同）。分类结果从正反两方面比较，取正确率较高者。最后得到的正确率为 73.3%。另外测试发现迭代次数大于 10 时，再增大并不能提高识别精度，但结果更加稳定，因为训练算法使用了随机初始化（为了打破对称性），训练次数增多，可以消除随机性的影响。

4.4 算法性能评价

对算法性能的评价，除了以上采用 holdout 验证的方法，将训练数据和测试数据分开，用测试数据评价分类性能之外，还实现了 k-fold 交叉验证方法，代码为：

```
k = 5;
cp = cvpartition(y, 'k', k);
f = @(xtrain, ytrain, xtest)…
 (logisticRpredict(logisticR(xtrain, ytrain, lambda), xtest));
err = crossvalid(f, X, y, k, cp);
```

这段代码对 Logistic 回归（lambda=0）进行 k=5 的 k-fold 交叉验证实验，其中函数 cvpartition 为 matlab 工具箱函数，是对数据进行分组，得到划分对象 cp；crossvalid 实现 k-fold 交叉验证并返回平均错误率。对该算法得到的平均错误率为 82.85%，耗时 125.66 秒。

5、实验结论

各种算法和实验方案的识别精度和训练耗时汇总如下表：

表-1 识别精度和训练时间

单词加权	BOOL				TF		TF-IDF
识别算法	测试识别精度（%）	训练时间（秒）	5-fold CV 精度（%）	5-fold CV 用时（秒）	测试识别精度（%）	训练时间（秒）	测试识别精度（%）
Logistic	83.25	21.65	82.85	125.66			82.75
Logistic + lambda=0.1	83.08	5.84	82.52	28.958	79.5	33.14	
Logistic + PCA	79	0.037	81.4	8.936	72.7	0.047	79.08
Naïve Bayes	80.25	0.003	80.52	0.12	77.08	0.073	
Naïve Bayes + smoothing	80	0.003	80.18	0.083			61.75
SVM	82.08	0.92	81.38	5.645	77.67	0.993	82.83
EMNB	73.3	12.3			77.55	10.385	67.08

从上述实验过程和结果来看，对本实验指定的宾馆评价文本情感分类这个问题，Logistic 回归算法效果较好，但训练时间比较长；SVM 次之，但速度很快；其次为 Naïve Bayes。参数调优方面，Logistic 回归不使用正则化（lambda=0）较好，但训练时间显著变大；Navie Bayes 也是不使用正则化（Laplace 平滑）效果较好。单词加权模型方面，TF 模型较 BOOL 稍差，使用 TF 得到的识别精度对大部分算法相比 BOOL 都要低 3~4 个百分点，但对无监督学习的 EMNB 来说，TF 模型得到的识别率较高；使用 TF-IDF 模型，几个连续型算法（Logistic 回归，SVM）都能得到较好的结果，获得和 BOOL 模型差不多的精度；而只针对离散型数据实现的算法（Naïve Bayes 和 EMNB）得到

的结果都比较差；把 TF-IDF 型的数据当作连续数据来处理，进行数值正规化（feature normalizing）以后效果有所改善。综合来看，对文本情感分类，带正则化的 Logistic 回归和 SVM 是较好的选择，Naïve Bayes 在训练时间方面有较大优势，适合数据样本规模较大的情形。对于特征维度特别大的情况，需要采取 PCA 等方法降维以后再进行处理。