机器学习大作业实验报告

1510590 姚睿 负责SELEC 算法（学姐让我把她学号删了可我还是不想删……）

1711290 李涵 负责baseline 特征选择 REML算法

一、问题描述

在现实生活当中，常常存在一个事物不仅仅有一个标签，即多标签的情况，我们采取多标签分类的方法。即有特征矩阵

$$X = \{x_1, x_2, \ldots\ldots, X_n\}$$

X是n×d的矩阵，以及标签矩阵

$$Y = \{y_1, y_2, \ldots\ldots, y_n\}$$

Y是n×L的矩阵，我们要找到一个映射方法，能够由X得到的预测矩阵尽可能地接近真实标签矩阵Y

二、解决方法

1.解决思路

（1）以SVC作为基分类器，利用Binary Relevance（它基本上把每个标签当作单独的一个类分类问题）的方式，建立n个分类器，对数据进行拟合，构建分类器

（2）利用Robust Extreme Multi-label Learning中的方法，求得映射矩阵W，构建分类器

（3）利用SLEEC算法，引入kNN分类器和聚类方法，高效解决大规模数据集的分类处理。

（4）通过特征选择CSFS来提高分类器的性能

2.基本理论

（1）在训练模型的时候，可能会造成过拟合，正则化是为了解决过拟合问题。采用正则化方法会自动削弱不重要的特征变量，自动从许多的特征变量中"提取"重要的特征变量，减小特征变量的数量级。其中正则化项可以分为L1、L2、L21。

L1正则化是指权值向量w中各个元素的绝对值之和，通常表示为||w||1，L1会趋向于产生少量的特征，而其他的特征都是0，因为最优的参数值很大概率出现在坐标轴上，这样就会导致某一维的权重为0 ，产生稀疏权重矩阵。

L2正则化是指权值向量w中各个元素的平方和然后再求平方根（Ridge回归的L2正则化项有平方符号），通常表示为||w||2， L2会选择更多的特征，最优的参数值很小概率出现在坐标轴上，因此每一维的参数都不会是0。当最小化||w||时，就会使每一项趋近于0。

L21范数是矩阵W每一行的l2范数之和，在最小化问题中，只有每一行的l2范数都最小总问题才最小。而每一个行范数取得最小的含义是，当行内尽可能多的元素为0时，约束才可能取得最小。而行内尽可能地取0意思是说行稀疏，综上可以这样解释，不同于l1范数（矩阵元素绝对值之和）的稀疏要求，l21范数还要求行稀疏。

（2）汉明损失，Hamming Loss反映了错误分类标签的分数

$$hloss(H) = \frac{1}{ml}\sum_{i=1}^{m}\sum_{j=1}^{l}[\![h_{ij} \neq y_{ij}]\!]$$

对数损失, 即对数似然损失(Log Loss)，用于评估分类器的概率输出

对数损失通过惩罚错误的分类，实现对分类器的准确度(Accuracy)的量化。最小化对数损失基本等价于最大化分类器的准确度。对数损失函数的计算公式如下:

$$L(Y, P(Y|X)) = -\log P(Y|X) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

其中, Y 为输出变量，X为输入变量，L 为损失函数， N为输入样本量，M为可能的类别数，yij 是一个二值指标，表示类别 j 是否是输入实例 xi 的真实类别， pij 为模型或分类器预测输入实例 xi 属于类别 j 的概率。

（3）多标签学习的关键是对标签间相关性进行建模，并将它们用于标签矢量预测。然而，在大多数现实世界的任务中，标签关系的先验知识通常是不可用的，因此有必要直接对标签关系进行建模。随着标签数量的不断增长，一些算法在计算上通常是不可行的。 基于嵌入的方法对于通过减少标签的有效数量来处理极端多标签学习问题是重要的。通常，他们假设标签矩阵是低秩，然后将标签向量投影到低维子空间。因此，不是直接学习预测每个示例的原始高维标签矢量，而是首先大大降低训练复杂度 。学习嵌入标签向量的预测器，然后采用解压缩操作将嵌入的标签向量提升回原始标签空间。 现有的嵌入方法已经开发了各种压缩和解压缩技术。

通过三步法解决了大量标签的分类问题。首先，随机变换用于将高维标签向量投影到低维空间；接下来，训练回归模型以预测变换后的标签向量的每个维度；最后，对于测试示例，其在低维空间中的预测标签向量被投射回原始标签空间。

但是存在尾标签（标签矩阵离群值or异常值），只发生几个例子，不能忽略。由于尾标签的的存在，违反了标签矩阵上的经典低秩假设。我们通过在REML中解决此问题。

Binary Relevance为每个标签独立训练分类器，训练预测成本会很高，我们假设标签矩阵是low rank的，来压缩标签向量。但是当存在tail labels的时候，会违反low rank假设，我们将tail labels视为异常值。我们将标签矩阵Y分解为Y_L和Y_S，其中Y_L是low rank的，Y_S是捕捉tail labels影响的稀疏部分。我们让Y_L=WX，Y_S=HX，通过使||Y-XW-XH||F最小来得到矩阵W和H，利用这两个矩阵，来预测标签Y。此外，还需要将W分解为U和V两个矩阵，进行求解。

（4） 即使对于小的低维度空间，嵌入式方法在训练和预测时也都很慢 。不同的嵌入式方法 差别在于压缩和解压方案，其能处理的分类尺度和能达到的准确率依然有待提升。这主要是由于存在数十万个“尾部”标签，这些标签各自出现在最多5个数据点，因此不能通过任何线性低维很好地近似。

为了解决这个问题，选择在特征空间使用kNN分类器而非其他压缩矩阵，因为kNN在数量极低的训练数据中优于其它判别方法，包括尾标签的情况。同时为了解决kNN在预测速度上的问题，SLEEC算法选择对整体的训练数据进行分类，避免出现预测结果不稳定或者准确率下降的情况，最后将随机生成的聚类的分类结果聚成一个模型。

$$\min_{V\in\mathbb{R}^{L\times d}} \|P_\Omega(Y^TY) - P_\Omega(X^TV^TVX)\|_F^2 + \lambda\|V\|_F^2 + \mu\|VX\|_1.$$

鉴于目标函数非凸且不可微分，也为了进一步优化对大规模数据的分类，选择分别计算特征Z和训练线性回归器V模型的优化.

a. ALS交替最小二乘法

交替最小二乘法是对最小二乘法处理多个变量时的扩展。基本原理是如果有两个变量需要确定，那ALS先固定第一个变量，然后求解第二个变量。之后固定第二个变量，求解第一个变量。如此交替迭代直至收敛或者达到最大迭代次数。利于矩阵缺失值处理。

b. 弹性网络回归

弹性网络回归是对岭回归和Lasso回归的融合，其惩罚项是对L1范数和L2范数的一个权衡。

（5）特征选择：

特征选择的方法：

- Filter：过滤法，按照发散性或者相关性对各个特征进行评分，设定阈值或者待选择阈值的个数，选择特征。
- Wrapper：包装法，根据目标函数（通常是预测效果评分），每次选择若干特征，或者排除若干特征。
- Embedded：嵌入法，先使用某些机器学习的算法和模型进行训练，得到各个特征的权值系数，根据系数从大到小选择特征。类似于Filter方法，但是是通过训练来确定特征的优劣。其中包括：基于L1的特征选择、随机稀疏模型、基于树的特征选择

  CSFS当中的特征选择方法：

  对于特征矩阵X，标签矩阵Y（有的有标签，有的无标签），我们假定F矩阵，对于有标签的，F对应的值就是Y对应的值，对于无标签的，F对应的值是预测标签。我们要找到一个f使得下边的函数成立

$$\min_{f,F_l=Y_l}\sum_{i=1}^{n}loss(f(x_i),f_i)+\mu\Omega(f),$$

其中Ω是范数

其在最小二乘法上的应用函数为：

$$\min_{W,F,\mathbf{b},F_l=Y_l}\sum_{i=1}^{n}s_i\|W^Tx_i+\mathbf{b}-f_i\|_2^2+\mu\|W\|_F^2,$$

其中si表示一个训练数据点的得分，1为有标签，0为无标签

为了使模型更加有效，我们在正则化项（1、2范数）上施加稀疏特征选择模型：

$$\min_{W,F,\mathbf{b},F_l=Y_l}\sum_{i=1}^{n}s_i\|W^Tx_i+\mathbf{b}-f_i\|_2^2+\mu\|W\|_{2,1}.$$

其中 0≤fi≤1

设定对角矩阵S，其中Sii=si，得到：

$$\min_{W,F,\mathbf{b},F_l=Y_l}Tr((X^TW+\mathbf{1}b^T-F)^TS(X^TW+\mathbf{1}b^T-F))$$
$$+\mu\|W\|_{2,1},$$

经过一系列的公式推导，我们可以得到：

$$XHSHX^TW + \mu DW = XHSHF,$$

W就是我们需要得到的特征选择矩阵，其中D一个对角矩阵，Dii为wi的2范数，所以

$$W = (XHSHX^T + \mu D)^{-1} XHSHF.$$

3.算法流程

（1）baseline：

a.分别调用回归模型的无正则化项、l1正则化项、l2正则化项的函数，构造分类器

b.计算准确率、召回率、对数损失和汉明损失

（2）Robust Extreme Multi-label Learning：

a. 随机初始化矩阵U，V，H

b.根据公式更新V

$$v_j^* = \left[\sum_{i=1}^{n}(x_iU)^T x_iU + \lambda_2 I\right]^{-1}\left[\sum_{i=1}^{n}\left(Y_{ij} - (XH)_{ij}\right)(x_iU)^T\right]$$

c.更新U

根据公式

$$u^* = \left[\sum_{i=1}^{n}\sum_{j=1}^{L}\widetilde{X}_{ij}(\widetilde{X}_{ij})^T + \lambda_2 I\right]^{-1}\left[\sum_{i=1}^{n}\sum_{j=1}^{L}\left(Y_{ij} - (XH)_{ij}\right)\widetilde{X}_{ij}\right]$$

更新u，其中u是U的向量化后的矩阵，X_ij~是Xi的转置乘Vj的转置

d.更新H，Z和μ

H的更新根据

$$\nabla_{H_j}\mathcal{J} = 2\Big(\sum_{i=1}^{n} x_i^T x_i + \lambda_1 I + \rho X^T X\Big)H_j - 2\Big(\sum_{i=1}^{n} x_i^T \widetilde{Y}_{ij} + \rho X^T(Z_j - \mu)\Big).$$

公式，令导等于0，求出当前最优的H的解

Z的更新根据公式

$$Z_j^* = soft(XH_j + \mu, \frac{\lambda_3}{\rho}), \tag{12}$$

where

$$soft(a, b) = sign(a)\max(|a| - b, 0). \tag{13}$$

μ的更新根据公式

$$\mu \leftarrow \mu + XH - Z.$$

e.重复bcd步骤，直到收敛，即下列函数的变化率小于0.001

$$\min_{U,V,H} \|Y - XUV - XH\|_F^2 + \lambda_1 \|H\|_F^2 + \lambda_2(\|U\|_F^2 + \|V\|_F^2) + \lambda_3 \|XH\|_1,$$

（3）SLEEC 算法：

训练算法：

a. 导入数据集X 并通过 k-means 算法将其分成$Q^1, \ldots, Q^C$集合

b. 每一个集合Q^j，对每个标签向量 $y_i \in Qj$ 用n 个近邻生成索引集Ω

c. 计算Z^j

$Z^j \leftarrow ALS(G, \Omega, \hat{L})$

d. 计算V^j

$V^j \leftarrow ElasticNet(X, Z, \lambda, \mu)$

e. 重新计算 Z^j

$Z^j = V^j X^j$

f. 输出 $\{(Q^1, V^1, Z^1), \ldots, (Q^C, V^C, Z^C)\}$

测试算法：

a. 导入训练集x，将其分到最近的集合Q_τ

b. 通过V^τx 计算 z

c. 计算z 在 Z^τ 空间最近的n个近邻N_z

d. 对于N_z中的点计算经验标签距离P_x

e. 选取P_x中最高的值作为预测标签y

（4）特征选择：

a. 输入features和labels以及迭代系数μ

b. 计算S矩阵，如果有标签，则Sii=1，无标签，Sii=0

c. 随机初始化W矩阵

d. 根据上述计算D矩阵，W t+1矩阵，

e. 计算b：$b_{t+1} = \frac{1}{m}\underset{\sim}{F}^T S\mathbf{1} - \frac{1}{m}\underset{\sim}{W}^T X S\mathbf{1}$；

f. 计算F t+1：$\widetilde{F}_{t+1} = X^T W + \mathbf{1}b^T$ 并调整F，如果F值大于1，则调整为1，若F值小于0，则调整为0

g. 重复d~f直到收敛

三、实验分析

1.实验数据

在本实验当中，我们采用 bibtex 数据集，一共有7395个样例，每个样例有1836个特征值，共有159个可能的标签。由于实验数据太大，所以我们将其随机打乱，取前30%来进行训练。

2.实验设计

（1）baseline：

a.导入数据集，并将数据集进行标准化，以便提高后期收敛速度

b.以RidgeCV为基分类器，用Binary Relevance的方法，为每一个标签类构建一个分类器，计算准确率、召回率、对数损失和汉明损失

c.以LassoCV为基分类器，用Binary Relevance的方法，为每一个标签类构建一个分类器，计算准确率、召回率、对数损失和汉明损失

d.L2,1正则化项的时候利用keras构建单层的神经网络，即线性回归，修改kernel_regularizer参数，进行自定义正则化项

e.以SGDClassifier为基分类器，用Binary Relevance的方法，损失函数分别为hinge和log，为每一个标签类构建一个分类器，计算准确率、召回率、对数损失和汉明损失

（2）Robust Extreme Multi-label Learning：

a.导入数据集，并将数据集进行标准化，以便提高后期收敛速度

b.设置参数k，λ1，λ2，λ3，ρ，随机初始化V，U，H，Z=XH，μ=1/ρ*Y

c.更新V

上文档中的公式的左半部分部分t1是一个k阶的单位矩阵的λ2倍加上Xi与U相乘的结果与其自身的转置相乘的总和；公式的右半部分t2是Yij与XHij的差值，与Xi与U相乘的结果的转置相乘的总和；将左半部分求逆，与右半部分相乘，即Vj的当前最优解。

d.更新U

对于i=range(n)，j=range(j)，求取Xi.T与V.T的乘积，将其向量化，与自身的转置相乘，得到一个1×1的矩阵，迭代相加，再加上1阶单位矩阵的λ2倍，作为公式的左半部分；对于i=range(n)，j=range(j)，Yij与XHij的差值与Xi.T与V.T的乘积的其向量化后的矩阵相乘，迭代相加，作为矩阵的右半部分；将左半部分求逆，与右半部分相乘，即U的向量化后的矩阵u的当前最优解，再调用reshape函数得到U矩阵。

e.更新Z、H、μ

对c×1的矩阵a和数值b，定义soft函数等于sign(a[i,0])*(max(abs(a[i,0])-b,0))。利用soft更新Z矩阵。

一个d阶的单位矩阵的λ1倍加上ρ+1倍的X的转置乘X，作为t1；Y-XUV作为t2；对t1求逆，乘以X的转置乘t2加上ρ倍的X(Z-μ)，即为H的当前最优解。

f.不断迭代cde步骤，并判断是否收敛，是否跳出循环。收敛函数如上文所述。

g.计算准确率和召回率

（3）SELEC 算法

a. 导入数据集，设置特征空间维度$\hat{L}$，kNN分类器的近邻值$\bar{n}$，目标聚类的数量 C ,以及正则化参数λ, μ

b. 用k-means 将训练集分为 C 个集合

c. 用kNN分类器代替解压缩矩阵，对于大量的x,预测的标签向量

$$\mathrm{y} = \sum_{i:V_{\mathrm{x}} \in KNN(V_x)} y_{\mathrm{i}}$$

生成最近邻的索引集合Ω

其中kNN 采用余弦相似度以更好的衡量标签向量方向上的特征差异。

$$sim(X,Y) = cos\theta = \frac{\vec{x} \cdot \vec{y}}{||x|| \cdot ||y||}$$

d. 交替最小平方法(Alternating least squares, ALS) 对kNN生成的 Ω 矩阵进行回归 生成 Z^j

e. 使用弹性网络回归算法求V^j，兼顾标签的稀疏特性同时保持正则化属性。

$$\min_{V \in R^{\hat{L} \times d}} ||Z - VX||_F^2 + \lambda ||V||_F^2 + \mu ||\mathrm{VX}||_1$$

f. 最后重新生成新的保留稀疏性的 Z^j 其中$Z^j = V^j X^j$ 并将每个集合的 特征 V, Z 聚集成一个模型输出。

$\{(Q^1, V^1, Z^1), \ldots, (Q^C, V^C, Z^C)\}$

不同的随机聚类集成一个模型这有助于解决聚类的不稳定性，并且仅通过训练和预测时间的线性增加显着提高预测准确性。

e. 对于测试集先划分属于的集合，并计算z,计算出在集合空间 Z^τ 中的临近点集合N_z，预测时加入权重 $wight = 1/distance$以提高准确率。

f. Z^τ中的点计算经验标签距离 P_x 并选择最高的几个值作为y的预测值。

g. 计算准确率

（4）特征选择：

首先要对读取到的数据进行随机打乱，再对features进行标准化处理，能够使得算法的收敛速度更快

设定迭代系数k，按照算法流程进行计算，最终得到F和W，要尝试不同的k，观察收敛速度，选取较优的k值

利用W计算2范式，根据设定的选取率（1/6，2/6，……，6/6）得到选择选取的特征的（利用headpq.nlargest函数）

对于监督学习，我们完全信任Y，所以用经过特征选择的X和Y对svm进行拟合，再对测试集进行测试；而对于半监督学习，有标签的我们用Y的值，没有标签的我们用F的值，来进行拟合以及测试。

由于这个是多标签学习，普通的svm并不能满足，所以需要调用sklearn.multiclass当中的Binary Relevance类来进行多标签学习。

在评估多标签学习和特征选择的性能好坏的时候，我们计算准确率和召回率来评判学习性能。

3.实验结果分析

（1）baseline：

l1：准确率：0.17670682730923695；召回率：0.32234432234432236

l2：准确率：0.5785123966942148；召回率：0.24504084014002334

l21：准确率：0.5722713864306784；召回率：0.24099378881987576

分析：L1正则化项小幅提高了准确率，召回率小幅降低，L2真这画像较大幅度提高了准确率，召回率极大降低。说明了正则化项的引入平衡了偏差与方差、拟合能力与泛化能力、经验风险（平均损失函数）与结构风险（损失函数+正则化项）。

以SGDClassifier作为基分类器，正则化项采取l2，看不同的损失函数对准确率的影响：

hinge：准确率：0.5936599423631124；召回率：0.2515262515262515

log：准确率：0.6182965299684543；召回率：0.23931623931623933

分析：Hinge Loss的值反映误分类标签数，这个值越小，则能对应召回率更高；Log Loss是对分类器的准确度的量化，这个值越小，则能对应准确率就会更高。

（2）Robust Extreme Multi-label Learning：

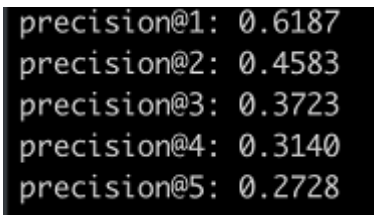
precision_score：0.7968217934165721

recall_score：0.24074074074074073

由实验结果可知，较以线性SVC为基分类器，用Binary Relevance的方法，准确率有了很大幅度的提高，召回率降低。说明了REML的方法对于尾标签能够有很好的处理，解决了尾标签对于低秩假设的破坏的问题。

（3）SLEEC:

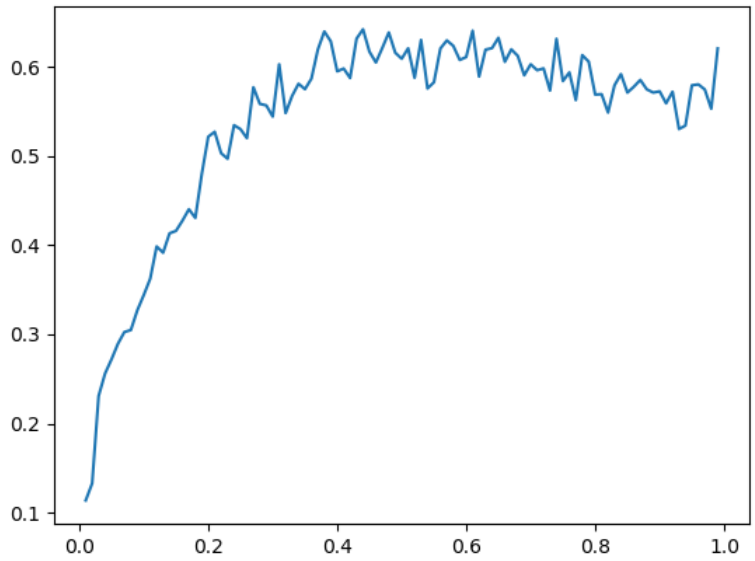
p@k 评分标签中前k个正确预测的分数



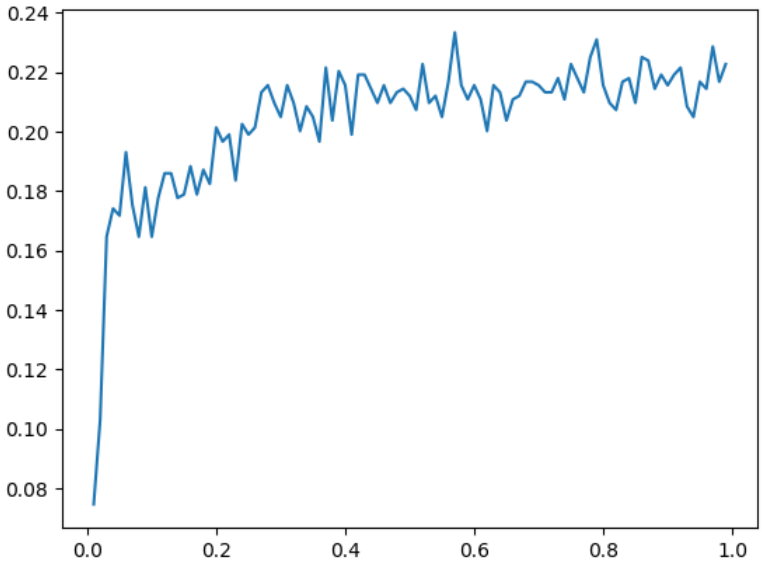
由实验结果可知，比起现行其它的嵌入式方法，对于极大标签量的数据集，SLEEC算法准确率有了显著提升。说明了对大规模多标签数据集，进行聚类并在在特征空间运用kNN算法确实是可以处理其的尾标问题，可以在其它问题上进行推广。

（4）特征选择的结果：

准确率随特征选择的比率的变化：



召回率随特征选择的比率的变化：



实验结果可以看到，准确率随着特征选择的比率大致有一个先增后减的趋势，但其中也有波动。实验结果表明，在进行特征选择之后，能够过滤掉一些无关特征或冗余特征，能够提高机器学习算法的性能，但是选择的特征过少，一些有用的特征值也被筛选掉，也会导致算法性能的下降，所以特征选区的比率就很重要。