实验一 分类技术---二分网络上的链路预测

学号：19049100002　　姓名:张泽群　　任课老师：马小科

1. 实验内容

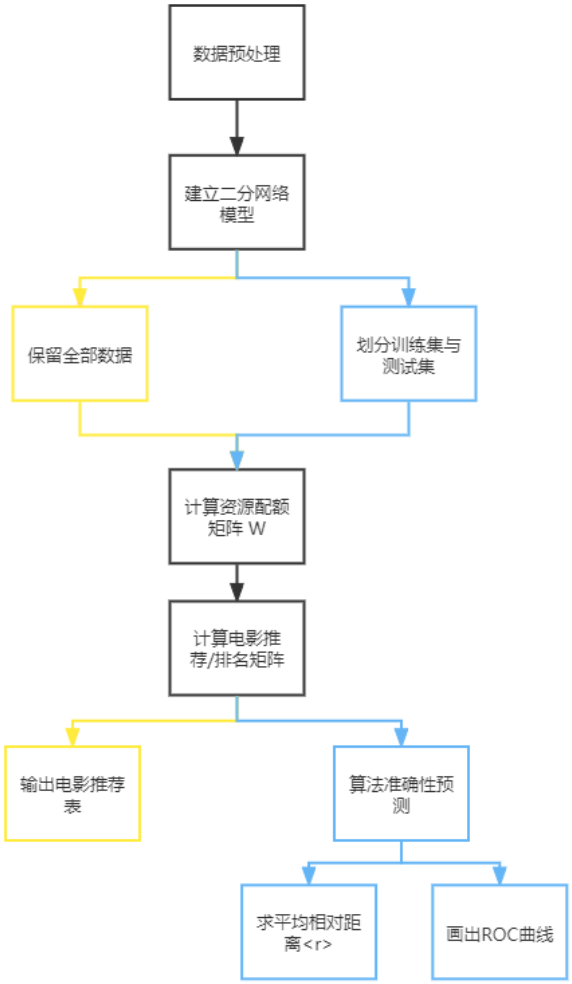
基于网络结构的链路预测算法被广泛的应用于信息推荐系统中。算法不考虑用户和产品的内容特征，把它们看成抽象的节点，利用用户对产品的选择关系构建二部图。为用户评估它从未关注过的产品，预测用户潜在的消费倾向。

本实验使用MovieLens所提供的数据集：ml-latest-small.zip，其中包括700个用户对9000部电影的100000条评价。

本实验首先根据提供的数据集，利用基于网络结构的链路预测算法，对用户的打分数据构造用户-电影邻接矩阵，计算资源配额矩阵W并以此求出针对每个用户的电影推荐排名，将排名靠前的电影推荐给用户。然后，将用户-电影二部图中的边随机分为90%的训练集和10%的测试集，对算法的准确性进行预测，包括求出平均相对位置<r>以及画出ROC曲线。

2. 分析及设计

流程图如下所示：



2.1 数据的选取与预处理

由于该算法不考虑用户和产品的内容特征，而是把它们看成抽象的节点，利用用户对产品的选择关系构建二部图，因此我们在数据选择中不需要知道用户和电影的具体特征（例如类型等），因此我们在数据的选取上只选用ml-1m文件夹中的'ratings.dat',获取用户对电影的选择关系。

由于matlab无法读取混合数据，我们利用记事本功能，替换数据中的分隔符:: 为 空格。

2.2 建立二分网络模型

这里我们构建用户-电影的二部图，根据喜爱度高低，以3为阈值将高喜爱度的电影和低喜爱度及未评价的电影分为两类，以此构造0/1矩阵。

2.3 计算资源配额矩阵

资源配额矩阵W中的元素w_ij表示产品j愿意分配给产品i的资源配额，这个抽象的能力可以看做位于相关产品上的某种可分的资源 ——拥有资源的产品会把更多的资源交给自己更青睐的产品。

实际上就是两电影被同一用户同时喜爱的比例的平均值，这可以反应两电影的喜爱关联度。

$$w_{ij} = \frac{1}{k_j} \sum_{l=1}^{m} \frac{a_{il} a_{jl}}{k_l}$$

其中,a_il,a_jl,表示邻接矩阵上的值, k_j表示产品j的度（被多少用户评价过），k_l表示用户l的度（用户选择过多少产品）。

2.4 计算电影推荐评分/排名矩阵

我们设定目标用户的资源分配矢量f，初始时，将用户选择过的电影对应项资源设置为1，其他为0，得到初始n维0/1向量。所有用户的资源分配矢量f的集合为A，则最终的资源分配矩阵如下:

$$A' = WA$$

矩阵中每一行中的值越大就说明该用户越喜欢(这些产品在那些已经被选择过的产品心目中总的分量最重)该电影。

这其实可以视作电影推荐评分矩阵，将该矩阵按行排序获得推荐电影的按序排名。

2.5 算法准确性预测

第一步是随机划分90%训练集和10%测试集，这里按照所给步骤是对于二部图中的边进行划分，也就是说是对高喜爱的用户-电影关系划分，而不是对所有数据集进行划分，我认为这一点有待商榷。

第二步是根据实验步骤中求出计算平均相对位置< r >,其中越精确的算法，给出的< r >越小量化评价算法的精确度。

由于构造二部分图和计算W矩阵时，只有训练集可以使用.在没有其他已经条件的前提下，只能假设用户已经选择过的产品是他喜欢的，因此，一个好的算法应该要把训练集中已知的用户喜欢的产品排在比较靠前的位置。对于任意一个用户 i，假设他有 Li 个产品是没有选择过的，那么算法会给出这 Li 个产品一个按照喜好程度的排序(最终资源数量相同的产品被赋予一个随机的序号)。如果在测试集中 i 选择了产品 j(这同时意味着 j 不会出现在训练集中, 因此是算法中 Li 个没有选择的产品之一), 而 j 被算法排在 第R_ij位，那么认为(i , j)的相对位置是

$$r_{ij} = \frac{R_{ij}}{L_i}$$

第三步是画出ROC曲线，我们将电影的总数×变化的阈值参数作为阈值，以此对比预测结果和测试集。

3. 详细实现

实验使用的编程语言为*matlab*，编程环境为*MATLAB R2021b*。

3.1 数据提取

利用load()函数。

```
load('ratings.dat');

rate_Number = length(ratings);      % 评价条数
users_Number = max(ratings(:,1));   % 用户个数
movies_Number = max(ratings(:,2)); % 电影个数
```

3.2 建立二分网络模型(邻接矩阵)

```
%% 1.构建'用户-电影'二分网络模型

A = zeros(users_Number,movies_Number);    % 二部图的邻接矩阵

k_User = zeros(6040,1); % 用户的度
k_Moive = zeros(3952,1);% 电影的度

count_Edge = 0;                    % 边的条数
edges = zeros(rate_Number,2);    % 边的两点信息集合
```

```
for i=1:rate_Number                    %计算边的条数，构造二部图邻接矩阵，计算用户度，
电影度
    if(ratings(i,3)>3)
        count_Edge = count_Edge+1;
        A(ratings(i,1),ratings(i,2))=1;        %构建邻接矩阵
        edges(count_Edge,1:2) = ratings(i,1:2);
        k_User(ratings(i,1),1) = k_User(ratings(i,1),1)+1;
        k_Moive(ratings(i,2),1) = k_Moive(ratings(i,2),1)+1;
    end
end
```

3.3 计算资源配额矩阵

这里即利用公式计算W，

```
%% 2.计算资源配额矩阵
W = zeros(movies_Number,movies_Number); % 利用公式计算
for i=1:movies_Number
    for j=1:movies_Number
        sum = 0;
        for l=1:users_Number
            if k_User(l,1) ~= 0
            sum = sum + A(l,i)*A(l,j)/k_User(l,1);   %除以用户的度
            end
        end
        W(i,j)=sum/k_Moive(j,1);    % 除以电影的度
        if isnan(W(i,j))      % 将nan的置0
            W(i,j) = 0;
        end
    end
end
```

k_j表示产品j的度（被多少用户评价过），k_l表示用户l的度（用户选择过多少产品）。

3.4 计算电影推荐评分/排名矩阵

这里我们直接利用矩阵，对所有用户的资源分配矢量f进行操作。

```
A_grade = A * W;      % 对应项评分矩阵

A_grade_sort = zeros(users_Number,movies_Number);   % 排序后的对应项评分矩阵

A_favor = zeros(users_Number,5);

for i=1:users_Number
    [A_grade_sort(i,:),I] = sort(A_grade(i,:),'descend');
    A_favor(i,1:5) = I(1,1:5); % 为用户推荐评分最高的5个电影
end

xlswrite('recmdMovies.xls', A_favor);    % 保存为xls
```

3.5 随机划分训练集与测试集

将二部图中的边随机分为两部分，其中90%归为训练集，10%归为测试集。（...为无关、省略部分）

```
count_Edge = 0;                    % 边的条数

edges = zeros(rate_Number,2);    % 边的两点信息集合

for i=1:rate_Number                %计算边的条数
    if(ratings(i,3)>3)
        count_Edge = count_Edge+1;
        edges(count_Edge,1:2) = ratings(i,1:2);
    end
end

random_Edge = randperm(count_Edge);      % 将边的序号随机排列

train_Set =   edges(random_Edge(1:round(0.9*count_Edge)),:);     % 获得训练集，90%的随机序号
...
test_Set  =  edges(random_Edge(round(0.9*count_Edge)+1:count_Edge),:);
 %获得测试集，剩余的10%
...
```

3.6 算法准确性预测_计算平均相对位置< r >

```
% 4.算法预测准确性预测

% 求R(i,j)，即A_train_grade_rank（i,j)，这里首先降序排列评分矩阵，并将电影的对应排名赋给对应序号
for i=1:users_Number
    [A_train_grade_sort(i,:),I] = sort(A_train_grade(i,:),'descend');
    for j=1:movies_Number
        A_train_grade_rank(i,I(j))=j;
```

```
    end
end
% 求L(i)，用于i有Li个产品是未选择
L = zeros(users_Number,1);
for i=1:users_Number
    L(i) = movies_Number - k_User(i);
end
% 求 r(i,j) = R(i,j)/L(i)
r = zeros(users_Number,movies_Number);

for i=1:users_Number
    for j=1:movies_Number
        if A_test(i,j)==1
            r(i,j) = A_train_grade_rank(i,j)/L(i);
        end
    end
end
% 对所有用户的r_ij求平均值
r_average = mean(r(:));
disp(r_average);
```

3.7 算法准确性预测_画出ROC曲线

```
%% 5.画出ROC曲线

TPR =[];
FPR =[];

for threshold =0:0.001:1
    A_isRight = double(A_train_grade_rank < (movies_Number *
threshold));
    % 0，1矩阵，排名在threshold*movie_size之前的为靠前=1，否则=0

    % 若一个电影获得过评价，且排名靠前（即，预测正确），则试其为真阳性
    A_TP = A_test .* A_isRight;
    TP = sum(A_TP(:));
    % 若一个电影未获得过评价，且排名靠前（即，预测错误），则试其为假阳性
    A_FP = (1 - A_test) .* A_isRight;
    FP = sum(A_FP(:));

    TPR = [TPR,TP/test_Number];
    FPR = [FPR,FP/(users_Number*movies_Number-test_Number)];
end
%画图
figure('NumberTitle', 'off','Name','ROC曲线_19049100002_张泽群')
plot(FPR,TPR,'-ro','LineWidth',1,'MarkerSize',1,"Color",'b');
xlabel('FPR');
ylabel('TPR');
title('ROC Curve');
```

```
% 计算包围面积,返回auc
auc = trapz(FPR,TPR);
disp(auc);
```

4. 实验结果

注：多次实验可过可见 实验结果图 文件夹

① 平均相对位置**< r >**以及**ROC**曲线的**auc**值

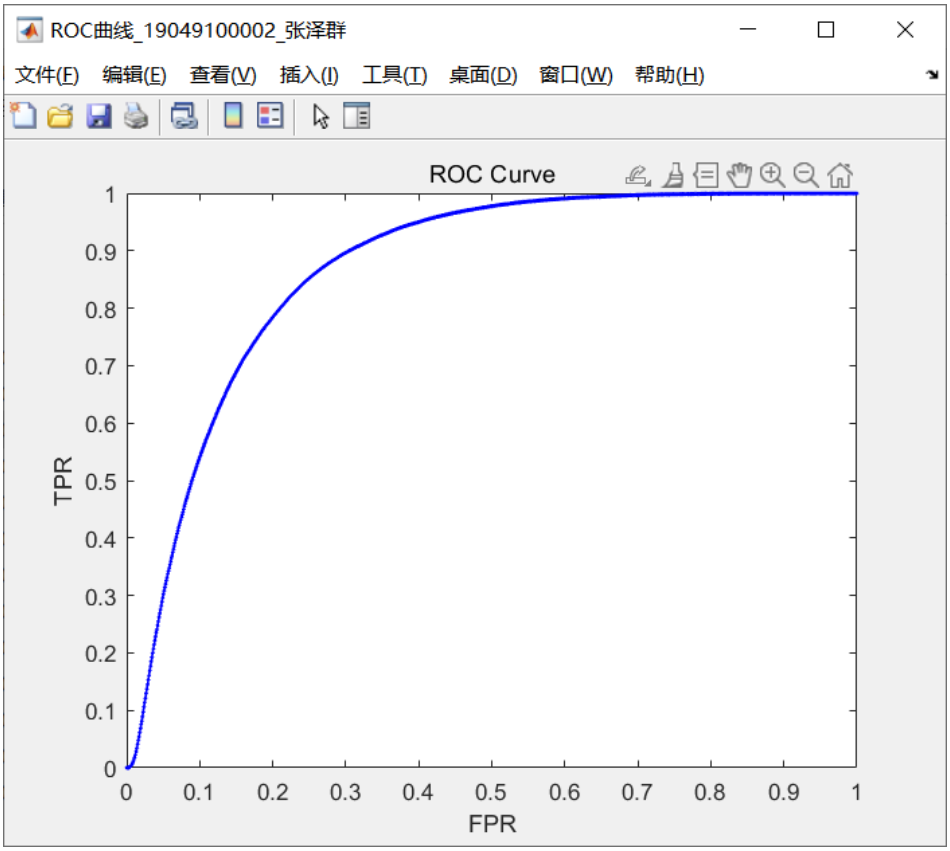
```
    3.4373e-04

    0.8672

fx >>
```

由于测试集训练集的划分具有随机性，根据多次的实验的验证,< r >值大致在0.00034左右，auc值大致在0.87左右

② **ROC**曲线



总结：完成设计方案内容，通过调试，整体完成度较高。

5. 心得体会

通过本次实验的研究学习，我了解到了一种较为简单的用户-产品推荐算法，对基于网络结构的链路预测算法有了进一步的了解，同时用程序实现分类算法，实现ROC曲线图让我对课本上的知识也有了亲身实践，理解更为深入，受益良多。

对于实验我也有些疑问，为何要对二部图中的边进行划分，即对高喜爱的用户-电影关系划分，而不是对所有数据集进行划分，我认为这一点有待商榷。

本程序也存在不足之处，训练的时间大概在10分钟左右、较为缓慢，另外ROC曲线的auc值也没有非常优秀，能否找到一种更快捷和准确的方式是改进的方向。

6. 源程序与文件附录

1. **task1_1.m**（主要用于获取整个数据集的用户电影推荐）
2. **task1_2.m**（主要用于算法准确性预测）
3. **recmdMovies.xls**（程序1运行得出排名前五的电影推荐表）
4. **test2_data.mat**（程序2运行得出的数据，导入该数据即可复现出实验结果图中的ROC曲线）
5. 剩余实验结果截图均在'实验结果图'文件夹中