第二节课笔记

https://github.com/cystanford/Recommended_System 老师的GitHub

第二节课笔记

课堂笔记	第二次反馈

- 课堂笔记

先用传统机器学习给出一个baseline
传统机器学习和深度学习：深度学习是由多层复杂t特征提取获取权重

基于标签的召回=基于标签的用户画像

#几个训练自己代码能力的网站：
ACM Online Judge
LeetCode
codewars

数据模型：
Kaggle、天池、DataCastle

对数据集进行处理：
小样本采样和特征工程

按照数据流处理阶段划分
收集原始数据：前端埋点、后端脚本（日志）
算法层将原始数据打上标签（偏好）
业务指导

K-means
Step1：选取N个中心点
Step2：将每个点按距离分配到最近的类的中心点，然后重新计算中心点
重复Step2直到中心点不在发生变化或达到最大次数
两点之间距离的定义：

- 欧氏距离：两点之间的

记录讲义内容

- 分类回归算法：
逻辑回归，线性回归，决策树，LDA，朴素贝叶斯，SVM，KNN，AdaBoost，XGBoost

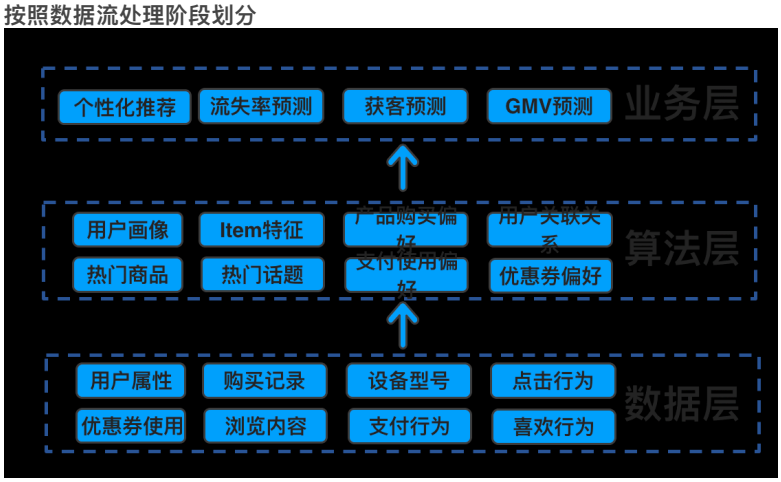
- 聚类算法：
K-Means，EM聚类，Mean-Shift，DBSCAN，层次聚类

- 推荐算法：
基于标签推荐：SimpleTagBased，NormTagBased，TagBased-TFIDF
基于内容的推荐
基于协同过滤的推荐：User-CF, Item-CF

用户标签都有哪些维度
八字原则：用户消费行为分析
用户标签：性别、年龄、地域、收入、学历、职业等
消费标签：消费习惯、购买意向、是否对促销敏感
行为标签：时间段、频次、时长、收藏、点击、喜欢、评分
（**User Behavior**可以分成**Explicit Behavior**和**Implicit Behavior**）
内容分析：对用户平时浏览的内容进行分析，比如体育、游戏、八卦

用户画像的准则
Step1、统一标识
用户唯一标识是整个用户画像的核心
Step2、给用户打标签
用户标签的**4**个维度
Step3、基于标签指导业务
业务赋能的**3**个阶段

按照数据流处理阶段划分



标签从何而来
典型的方式有：

距离

- 曼哈顿距离：纵坐标+横坐标
- 切比雪夫距离：max{纵坐标、横坐标}
- 余弦距离：vec(x)vec(y)/|vec(x)*||vec(y)|

Z-score = (样本值-均值)/标准差
样本和平均值差了多少标准差

fit、fit_transform的区别

聚类是无监督的学习，具体含义需要我们指定
什么时候使用聚类：

- 缺乏足够的先验知识
- 人工打标签太贵

混淆矩阵：

	预测类别		
实际类别		Yes	No
	Yes	TP	FN
	No	FP	TN

准确率 accuracy = (TP+TN)/(TP+FP+TN+FN)
召回率 recall = TP/(TP+FN)
精确率 precision = TP/(TP+FP)
F值 = (\alpha^2 + 1)precision*recall/(\alpha^2 (precision+recall))

SimpleTagBased算法
NormTagBased算法：对score进行归一化
TagBased-TFIDF算法：IDF = log{文档次数/（单词出现但是文档数+1）}

IDF：Inverse Document Frequency

pip install加速
pip国内的镜像：
阿里云
http://mirrors.aliyun.com/pypi/simple/
中国科技大学
https://pypi.mirrors.ustc.edu.cn/simple/
豆瓣(douban)
http://pypi.douban.com/simple/
清华大学
https://pypi.tuna.tsinghua.edu.cn/simple/
中国科学技术大学
http://pypi.mirrors.ustc.edu.cn/simple/
使用方法：
pip install tensorflow -i https://pypi.tuna.tsinghua.edu.cn/simple

TPOT：基于**Python**的**AutoML**工具

- PGC：专家生产
- UGC：普通生产

标签是对高维事物的抽象（降维）
聚类算法：K-Means，EM聚类，Mean-Shift，DBSCAN，层次聚类

数据规范化的方式：

- **Min-max**规范化

将原始数据投射到指定的空间**[min,max]**
新数值 = （原数值-极小值）/ (极大值 - 极小值)
当**min=0, max=1**时，为**[0,1]**规范化
sklearn中的**MinMaxScaler**

- **Z-Score**规范化

将原始数据转换为正态分布的形式
新数值 = （原数值 - 均值）/ 标准差
sklearn中的**preprocessing.scale()**

- 小数定标规范化

通过移动小数点的位置来进行规范化
使用**numpy**

阿特曼Z-score模型
公开上市交易的制造业公司的破产指数模型：
Z =1.2X1 + 1.4X2 +3.3X3 + 0.6X4 + 0.999X5
X1 = 净营运资本 / 总资产 = （流动资产— 流动负债） / 总资产
X2 = 留存收益 / 总资产
X3 =息税前收益/ 总资产 = （利润总额+财务费用） / 总资产
X4 = 优先股和普通股市值 /总负债= （股票市值 * 股票总数） / 总负债
X5 = 销售额 / 总资产
判断准则：**Z<1.8,**破产区；**1.8≤Z<2.99,**灰色区；**2.99<Z,**安全区

数据规范化的方式：

- 小数定标规范化

通过移动小数点的位置来进行规范化
比如**A**的取值范围 **[-9999,666]**
A的新数值=原数值**/10000**。

SimpleTagBased算法

- 统计每个用户的常用标签
- 对每个标签，统计被打过这个标签次数最多的商品
- 对于一个用户，找到他常用的标签，然后找到具有这些标签的最热门物品推荐给他
- 用户u对商品i的兴趣：score（u，i）=、sum_t{user_tags[u,t]*tag_item(t,i)」

数据结构定义：

- 用户打标签记录： records[i] = {user, item, tag}
- 用户打过的标签：user_tags[u][t]
- 用户打过标签的商品：user_items[u][i]
- 打上某标签的商品：tag_items[t][i]
- 某标签使用过的用户：tags_users[t][u]

	预测类别		
实际类别		Yes	No
	Yes	TP	FN
	No	FP	TN

TPOT
https://github.com/EpistasisLab/tpot
（6.2K）
TPOT可以解决：特征选择，模型选择，但不包括数据清洗
处理小规模数据非常快，大规模数据非常慢。可以先抽样小部分，使用**TPOT**

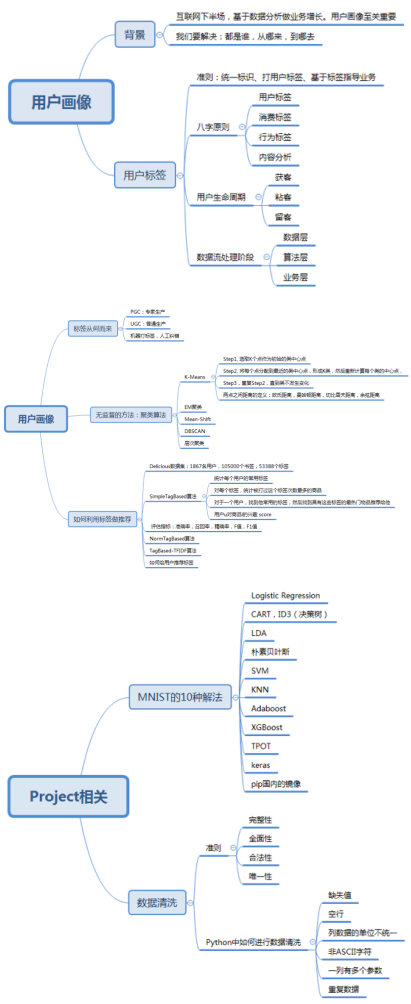
Google Cloud AutoML
华为ModelArts

好的数据：完全合一
完整性，全面性（单位、字段与数值不符），合法性（数据类型内容大小），唯一性（是否有重复）

Get dummies??

获取特征的重要性：feature_importance = model.coef_[0]

机器之心



如何给用户推荐标签
当用户u给物品i打标签时，可以给用户推荐和物品i相关的标签，方法如下：

- 方法1：给用户u推荐整个系统最热门的标签
- 方法2：给用户u推荐物品i上最热门的标签
- 方法3：给用户u推荐他自己经常使用的标签
- 将方法2和3进行加权融合，生成最终的标签推荐结果

基于内容的推荐系统架构

- 物品表示 Item Representation：

为每个item抽取出features

- 特征学习Profile Learning：

利用一个用户过去喜欢（不喜欢）的item的特征数据，来学习该用户的喜好特征（profile）；

- 生成推荐列表Recommendation Generation：

通过用户profile与候选item的特征，推荐相关性最大的item。

Summary

- **聚类是一种降维方式，距离的定义**
- **定义用户画像的维度：用户消费行为内容**
- **围绕用户生命周期开展业务：获客粘客留客**
- **数据处理层次：数据源-算法层-业务层**
- **标签是一种抽象能力，通过用户画像进行profile learning，同时对item提取标签，从而完成基于标签的召回**
- **标签照会简单计算，属于召回的一种侧率**

Excel数据统计：

- **mysql-for-excel**

https://dev.mysql.com/downloads/windows/excel/

- **mysql-connector-odbc**

https://dev.mysql.com/downloads/connector/odbc/

缺失值：删除、均值、高频
均值：df['Age'].fillna(df['Age'].mean(), inplace=True)
删除： df.dropna(how='all',inplace=True)

```
# 删除非 ASCII 字符
df['name'].replace({r'[^\x00-\x7F]+':''}, regex=True, inplace=True)
统一单位：
# 获取 weight 数据列中单位为 lbs 的数据
rows_with_lbs = df['weight'].str.contains('lbs').fillna(False)
# 将 lbs转换为 kgs, 2.2lbs=1kgs
for i,lbs_row in df[rows_with_lbs].iterrows():
# 截取从头开始到倒数第三个字符之前，即去掉lbs。
weight = int(float(lbs_row['weight'][:-3])/2.2)
df.at[i,'weight'] = '{}kgs'.format(weight)
```

apply lambda？？

一列有多个参数（可选）
可以将**Name**分成**last name + first name**也可以进行保留。

```
# 切分名字，删除源数据列
df[['first_name','last_name']] = df['name'].str.split(expand=True)
df.drop('name', axis=1, inplace=True)
```

默认采用的空格进行分割，相当于**df['name'].str.split(' ', expand=True)**

```
# 删除重复数据行
df.drop_duplicates(['first_name','last_name'],inplace=True)
```

数据探索：
print(train_data.info())
数据探索：
print(train_data.describe(include=['O']))
查看离散数据类型的分布

特征选择：

```
# 特征选择
features = ['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked']
train_features = train_data[features]
train_labels = train_data['Survived']
test_features = test_data[features]
dvec=DictVectorizer(sparse=False)
train_features=dvec.fit_transform(train_features.to_dict(orient='record'))
print(dvec.feature_names_)
```

总结Summary

- 记录作业内容

1. Thinking1：如何使用用户标签来指导业务（如何提升业务）
2. Thinking2：如果给你一堆用户数据，没有打标签。你该如何处理（如何打标签）
3. Thinking3：准确率和精确率有何不同（评估指标）
4. Thinking4：如果你使用大众点评，想要给某个餐厅打标签。这时系统可以自动提示一些标签，你会如何设计（标签推荐）
5. Thinking5：我们今天使用了10种方式来解MNIST，这些方法有何不同？你还有其他方法来解决MNIST识别问题么（分类方法）
6. Action1：针对Delicious数据集，对SimpleTagBased算法进行改进（使用NormTagBased、TagBased-TFIDF、TagBased-TFIDF++算法）**Delicious数据集：1867名用户，105000个书签，53388个标签 https://grouplens.org/datasets/hetrec-2011/ 格式：userID bookmarkID tagID timestamp**
7. Action2：对Titanic数据进行清洗，使用之前介绍过的10种模型中的至少2种（包括TPOT）
8. **MNIST的十种解法**
9. **Project：对Steam-200K 数据进行数据清洗**
10. NBA球员数据分析 **NBA**球员数据表：**https://www.kaggle.com/edgarhuichen/espn-nba-players-data**