廣東工業大學

QG 中期考核详细报告书

题　　目　机器学习库的实现

学　　院　计算机学院

专　　业　计算机科学与技术

年级班别　19 级（1）班

学　　号　3119004757

学生姓名　许继元

2020 年 04 月 24 日

1、K-means 算法

此处使用 K-means 算法处理 Iris 数据集。

➢ 数据集的处理

1. 首先引入数据集
2. 接着去除数据集中表示分类结果的列
3. 然后把数据集转换为数组形式

```
1. data_X = pd.read_csv(r"iris.csv")
2. data_X = data_X.drop(data_X.columns[4], axis=1)
3. data_X = np.array(data_X)
```

➢ 算法步骤和思想

K-means 算法大致思想：对于给定的样本集，按照样本之间的距离大小，将样本集划分为 K 个簇。

实现步骤:

1. 随机选取 K 个初始点为质心（类别）
2. 遍历每一条数据，计算其与 K 个质心的距离
3. 选择与之距离最近的质心作为该数据所属的类别
4. 每个簇的质心更新为该簇所有点的平均值
5. 重复 2、3、4 步骤，直到代价函数收敛到最小值

➢ 算法实现结果评估

K-means 算法的代价函数为：

$$J\left(c^{(1)},\ldots,c^{(m)},\mu_1,\ldots,\mu_K\right)=\frac{1}{m}\sum_{i=1}^{m}\left\|X^{(i)}-\mu_c(i)\right\|^2$$

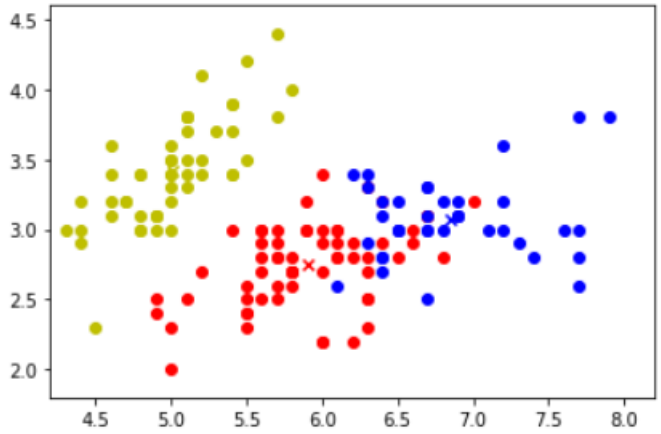
该代价函数为 SSE（误差平方和）

其中$X^{(i)}$是样本点，$c^{(i)}$是簇，$\mu_c(i)$是质心（簇的均值向量）。

我们的优化目标为最小化该代价函数。

在 Jupyter Notebook 运行并可视化如下：

```
误差平方和为:  61.16903343419394
```



根据结果可知误差平方和为 61.169 左右，还是蛮大的。

由此可见 K-means 算法还是存在一些不足之处，总结如下：

① 对初值敏感，对于不同的初始值，可能会导致不同结果。

② 通常会收敛于局部最小值。

➢ 优化之处：

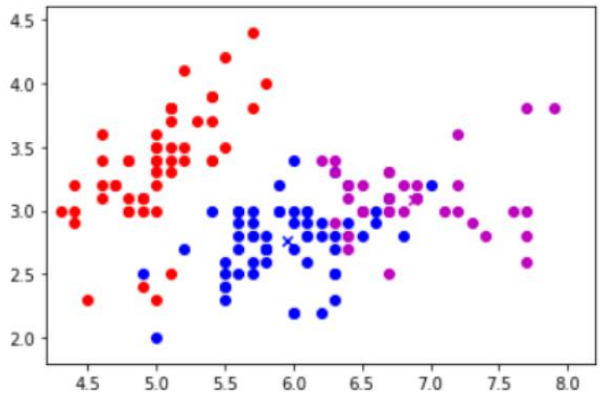
为了优化 K-means 算法，此处尝试使用 Bisecting K-means 算法，也就是二分 K-means 算法。

Bisecting K-means 算法 与 K-means 算法的不同之处：

Bisecting K-means 算法并不是一开始就随机选择 K 个中心点，而是先把所有点归为一个簇，然后将该簇一分为二。计算各个所得簇的代价函数（SSE），选择 SSE 最大的簇再进行划分以尽可能地减小误差，重复上述基于 SSE 划分过程，直到得到用户指定的簇数目为止。

再次运行，可视化结果如下：

```
误差平方和为:  42.02178663810265
```



误差平方和相对于 K-means 算法有所下降，可见 Bisecting K-means 算法在 K-means 算法的基础上提高了聚类的性能。

2、LinearRegression 算法

此处使用 LinearRegression 算法处理 housing 数据集。

➢ 数据集的处理

1. 首先引入数据集，同时使用正则表达式对空格符进行分割
2. 然后把数据集分为样本特征项和输出标记项
3. 接下来将数据转换为数组形式

```
data = pd.read_csv("housing.txt", sep="\s+", header=None)
X = np.array(data.iloc[:,[0,1,2,3,4,5,6,7,8,9,10,11,12]])
y = np.array(data.iloc[:,[13]])
y = y.reshape([506,])
```

➢ 算法步骤和思想

LinearRegression 算法的思路大致是，寻找一条直线，最大程度地拟合样本特征和样本输出标记之间的关系。

实现步骤:

1.通过最小二乘法训练 LinearRegression 模型
2.得出拟合数据的直线的参数

最小二乘法公式：

$$a=\frac{\sum_{i=1}^{m}(x^{(i)}-\bar{x})(y^{(i)}-\bar{y})}{\sum_{i=1}^{m}(x^{(i)}-\bar{x})^2} \qquad b=\bar{y}-a\bar{x}$$

➢ 算法实现结果评估

对拟合结果进行误差评估如下：

```
均方误差为: 18.82719744657374
均方根误差为: 4.339031855906769
平均绝对误差 3.1070116428840118
R Squared为: 0.76509066257533
```

➢ 不足之处：

可使用梯度下降算法训练 LinearRegression 模型。

3、LogisticRegression 算法

此处使用 LogisticRegression 算法处理 adult 数据集(未完成)和 Titanic 数据集。

➢ 数据集的处理

1. 首先引入数据集，给数据集添加列标签
2. 然后进行独热编码，缺失值填充等
3. 未完成

```
1. data = pd.read_csv("adult.csv", sep="\s+", header=None, names=["age","workcl
   ass", "fnlwgt", "education", "education_num", "maritial_status","occupation"
   , "relationship", "race", "sex", "capital_gain", "capital_loss", "hours_per_
   week", "native_country", "income"])
2. data = data.dropna(how='all')
3. data.loc[data['income'] == '>50K', 'income'] = 1
4. data.loc[data['income'] == '>50K.', 'income'] = 1
5. data.loc[data['income'] == '<=50K', 'income'] = 0
6. data.loc[data['income'] == '<=50K.', 'income'] = 0
```

发现特征工程难度不是很小，加上时间不太充足，故更换 Titanic 数据集。

1. 首先引入 Titanic 数据集，用.head 方法预览数据
2. 接下来进行缺失值填充和独热编码
3. 初步运行代码后准确率不高，尝试特征工程
4. 构造和删去了一些特征

➢ 算法步骤和思想

LogisticRegression 算法的大致思想：

逻辑回归算法是通过概率来进行预测的。假设 y 是想要的结果概率，而有 n 个因变量会影响该概率，将 n 个因变量表示为 x1，x2，...，xm，逻辑回归算法做的就是把这些因素对应的得分进行求和，和的结果越大表示可能性越大，反之越小。每个因变量都有对应的权值，表示为 θ1，θ2，...，θm，将因变量乘以相应的权值得到的加权和就是最后的得分，最后根据得分进行归类。

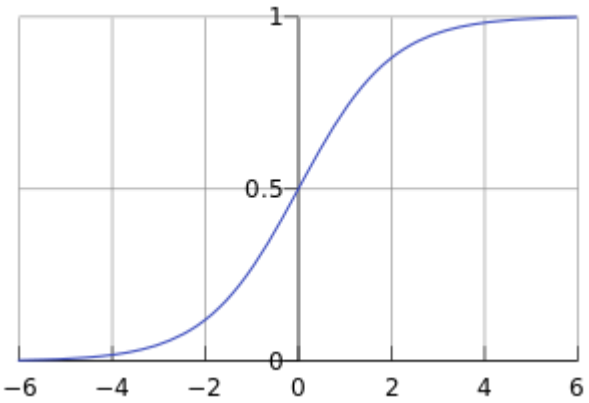
实现步骤:

1. 选择 Sigmoid 函数作为预测函数
2. 通过梯度下降法训练 LogisticRegression 模型
3. 对比预测结果和实际结果，计算准确率

Sigmoid 函数：

$$g(z)=\frac{1}{1+e^{-z}}$$

函数图像：



逻辑回归算法的损失函数：

$$Cost(h_\theta(x),y)=\begin{cases}-\log(h_\theta(x)) & if\ y=1\\ -\log(1-h_\theta(x)) & if\ y=0\end{cases}$$

$$J(\theta)=\frac{1}{m}\sum_{i=1}^{n}Cost(h_\theta(x^{(i)}),y^{(i)})=-\frac{1}{m}\sum_{i=1}^{n}\left(y^{(i)}\log h_\theta(x^{(i)})+(1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right)$$

它是基于极大似然估计推导得到的。

➢ 算法实现结果评估

对预测结果进行准确率计算如下：

```
准确率为: 75.843%
```

➢ 不足之处：

① 缺失数据可视化
② 误差分析不够

4、决策树算法

此处使用决策树算法处理 lenses 数据集。

➢ 数据集的处理

1. 首先引入数据集

2. 然后去除数据的空格符

➢ 算法步骤和思想

决策树算法思想：

根据决策树模型，从根节点开始，根据分裂的特征和阈值进行分裂（即决策），然后由判断结果决定进入哪个分支节点，直至到达叶节点处，得到结果。

实现步骤:

1. 根据信息增益的准则，筛选出跟分类结果相关性较高的特征，也就是分类能力较强的特征。
2. 从根节点开始，对每个节点计算所有特征的信息增益，选择信息增益最大的特征作为节点特征，根据该特征的不同取值建立子节点；然后对每个子节点使用相同的方式生成新的子节点，直到信息增益很小或者没有特征可以选择为止。
3. 主动去掉部分分支，防止过拟合。（未实现）

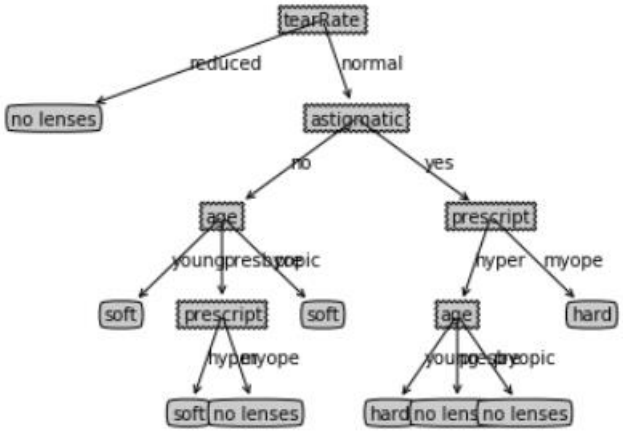
此处基于 ID3 算法进行最优决策：

ID3 算法的思路：信息增益最大化

信息增益 = 熵 - 条件熵

➢ 算法实现结果评估

参考于《机器学习实战》，运行结果如下：



{'tearRate': {'reduced': 'no lenses', 'normal': {'astigmatic': {'no': {'age': {'young': 'soft', 'presbyopic': {'prescript': {'hyper': 'soft', 'myope': 'no lenses'}}, 'pre': 'soft'}}, 'yes': {'prescript': {'hyper': {'age': {'young': 'hard', 'presbyopic': 'no lenses', 'pre': 'no lenses'}}, 'myope': 'hard'}}}}}}

➢ 不足之处：

① 可以使用 CART 算法和 C4.5 算法
② 尝试随机森林算法