Ex4 record and report

王敏行 id：2018012386 wangmx18@mails.tsinghua.edu.cn

Part 8: Feature selection

这里使用 sklearn.feature_selection.VarianceThreshold 函数，实现特征选择。

考虑一个特征的分类能力，如果某一特征是单值的，其方差就较小，那么其包含的信息就少，难以用于分类。VarianceThreshold 函数会计算每个特征的的方差，并根据输入参数 threshold，去除方差小于 threshold 的特征。

这里采用线性 SVM（sklearn.svm.LinearSVC）作为分类模型，参数选择为 Ex3 中摸索的最优参数（C=0.001）。分别用原始数据（108 个 feature）和过滤的数据（67 个 feature）在 trainset1 上进行训练，以 testset1 上的准确率测试其表现。

本实验设置 threshold 为概率为 0.8 的伯努利分布的方差 $\mathrm{var} = p(1-p) = 0.16$。对 108 个特征进行阈值截断后，高于阈值的还有 67 个特征。训练结果表示模型不收敛。具体结果如下：

```
origin data train acc:0.78900 validation acc:0.78304
filtered data train acc:0.77940 validation acc:0.77484
```

尝试多次，取不同的 random_state，发现对于训练集的过滤与否对于线性 svm 的影响不显著。

Part 9: Lasso regression

Lasso 回归的优化目标是找到带有一次正则项的代价函数的最小值：

$$J(w) = \frac{1}{2 * n_{samples}} * \| y - Xw \|_2^2 + \alpha \| w \|_1$$

最终优化得到的 w 是一个 1*dim_feature 的矩阵，抛去其中是 0 的项，剩下的非零项可作为特征用作为下由分类器的输入。

本实验中，108 维的数据保留了 29 维。尝试将这 29 维数据用于 linearSVM 的训练、测试，结果如下：

```
iterations:7 Lasso selected data
train acc:0.73180 validation acc:0.73382
```

显然，引入 Lasso regression 加快了 SVM 模型收敛的速度，但是分类的效果比原始数据要差。

还可以用 Lasso 函数本身的打分功能，返回每个样本的分类概率，再人为设定截止概率（这里设为 0.5），完成二分类。结果稳定，且效果优于 Lasso 降维再用 SVM 分类的分类流程。

```
Lasso regression classifier
train acc:0.75560 validation acc:0.75479
```

Part 10: random forest

利用 sklearn.ensemble.RandomForestClassifier 函数，进行随机森林的分类。考虑到这是一个二分类问题，选择决策树的数量 n_estimators 为 100，为了避免严重的过拟合问题，限制最大叶子数目 max_leaf_nodes 为 10。

分别用原始数据和 Lasso 数据对模型进行训练，并用对应的测试集数据测试。结果分别如下：

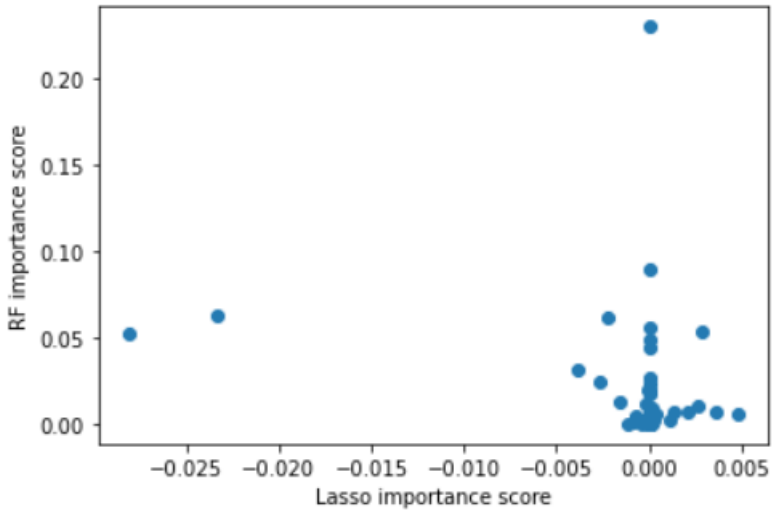
```
RF on raw data
train acc:0.79940 validation acc:0.79125

RF on Lasso-decomposed data
train acc:0.77260 validation acc:0.77940
```

二者的表现差异不明显。

利用 RandomForestClassifier 自带的 feature_importances_方法，给每一个 feature 的重要程度进行打分。打分结果见 notebook 输出。

可以比较 RF 的打分结果和 lasso 的打分结果，如下图所示。很遗憾，二者没有显著的相关性。



随机森林部分参考了：

[How to Calculate Feature Importance With Python (machinelearningmastery.com)]