杨启凡2016213623 刘佳军2016213624

实验一

# 概述

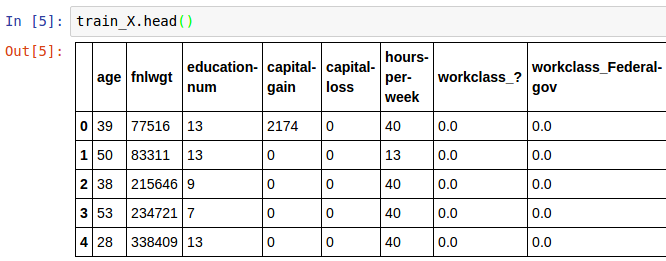
本实验中，我们首先对给定的人口普查数据做因子化处理，并对缺失的属性单独划分维度；我们首先实现了两个基本的分类器：K近邻和朴素贝叶斯，并与Sklearn中实现的K近邻、朴素贝叶斯、决策树、集成学习的AdaBoost和简易的MLP神经网络从准确率、F-score等方面进行了对比评估，还尝试了xgboost(Gradient Boosting)的开源实现，最终了xgboost达到了对测试集87.8%的分类准确度。

# 特征工程

人口普查数据集合共有14个维度的特征，其中age, fnlwgt, education-num, capital-gain, capital-loss, hours-per-week, income为连续的数值型属性，其余为枚举类型属性。

在训练与测试集中，workclass, occupation, native-country属性中有缺失值，workclass有2799人缺失，occupation有2809人缺失，native-country有857人缺失，由此看来，缺失的属性和人数较少，并且缺失的属性均为枚举类型，因此不能取平均值等进行填充，所以考虑将缺失的属性单列枚举类型为“缺失”。

分类器一般难以处理枚举类型，所以我们将枚举类型的维度因子化处理，例如education可能有“Assoc-voc”, “Some-college”等属性，将每个唯一的枚举类型单独作为一个维度，如果某个instance在这个是这个枚举类型，则记为1，否则记为0。使用Python Pandas库对特征因子化处理后的结果如下所示：



可以看到，保留了数值类型属性不变，将枚举类型属性因子化，原来没个维度的缺失值单独作为一类属性，共计107维特征。训练集32561人，测试集16281人。

其中值得一提的是本二类分类问题中，正负类样本并不均衡，测试集合中类别0占了76%，所以分类器准确率应该以76%为基线。

# 分类器

两个分类器实现在yqflearn.py文件中的KNN类和NativeBayes中，实现了基本的Sklearn BaseEstimator的接口，因此可以方便地嵌入Sklearn框架进行算法的验证。

## K近邻

通过距离函数计算每个测试样本到所有训练样本的距离，取距离最小的前K个近邻居，投票多数决定测试样本的类别。可以调整的参数有：

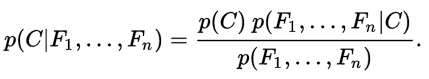
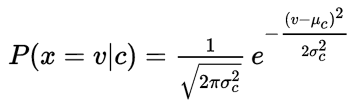
* Dis\_func，距离函数。可以取sklearn.neighbors中的距离函数，如euclidean，manhattan，hamming等；
* Scale，是否将所有维度的范围都按照训练集缩放到[-1, 1]区间，以消除某个维度较大取值范围对其他维度较小取值范围的影响；
* K，参考的邻居数量，默认为1，也就是最近邻居；

算法简单流程：

* 训练阶段，将样本的特征X和类标y存储下来，如果需要scale，则将训练样本的每一维度scale到[-1, 1]区间。
* 预测阶段，如果需要scale，则将预测样本先按相同的规则进行scale处理，然后对于每一个预测样本，计算它到所有训练样本的距离，取距离最小的前K个样本，按照类标投票，取计数最大的类标返回。

## 朴素贝叶斯

基于样本个维度之间相互独立的朴素假设，使用(1)公式计算测试样本属于每个类别的概率，取概率最大的类作为该样本的类标。

(1) (2)

本次实验特征均处理为数值型，所以需要使用高斯朴素贝叶斯，假设连续的数值为高斯分布。

* 训练阶段，首先计算出每一类的所占比例，记为先验概率，然后对数据根据类别分类，然后计算每个类别中x的均值和方差
* 预测阶段，使用(2)公式，计算出预测样本属于每一类的概率，取最大概率类标

## 其他分类器

决策树：使用sklearn.tree.DecisionTreeClassifier

Adaboost：使用sklearn.ensemble.AdaBoostClassifier

MLP神经网络：使用sklearn.neural\_network.MLPClassifier

Xgboost：使用Python版本Xgboost实现库

# 评估与分析

实验环境为Python 3.4

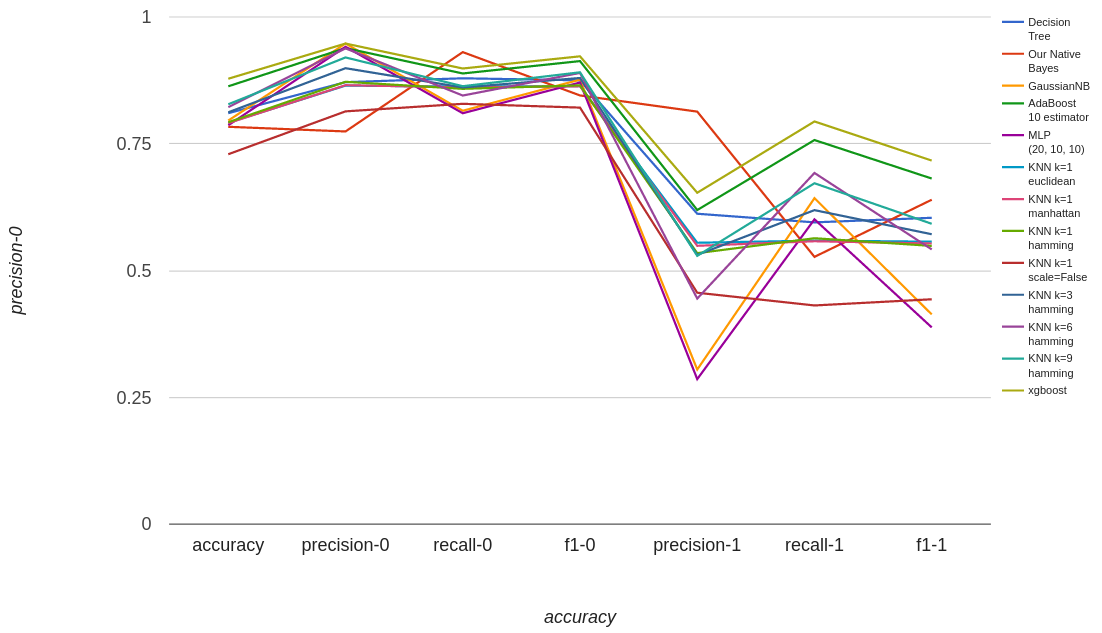
Jupyter Notebook 4.2.3

本实验是一个数据挖掘项目，使用Jupyter Notebook方便进行数据的互动和画图，使用了Sklearn的机器学习模型和评估框架，使用了Pandas进行数据处理，使用Seaborn可视化。

实验的原始数据和数据处理代码可以查看程序输出 results/task-1-output.html。

### 实验结果

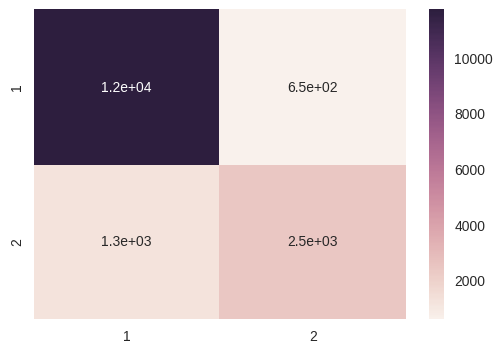
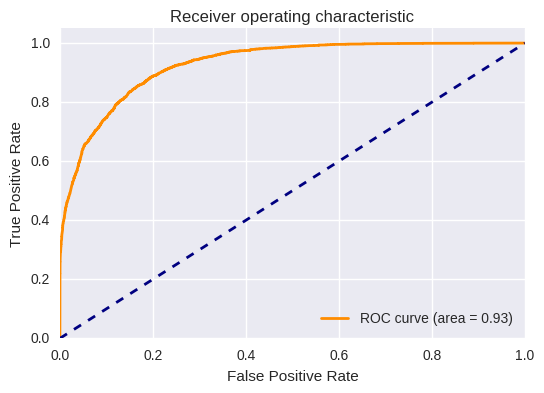
使用32561人x107维度特征做训练集对不同的模型和参数组合训练，16281人x107维度做测试集，使用accuracy和两类的precision、recall、f1等指数进行评估。



各分类器的指数比较

可以从上图中看出，表现最好的两个分类器是Xgboost和Adaboost，而Xgboost的准确率达到了87.8%，并且在所有指数上都比Adaboost好一些。Bayes方法虽然准确度和精度不高，但召回率相对比较高。KNN算法中，使用hamming距离表现最好，不用scale的话KNN算法准确度较低，逐步增加K的值，准确率有一定的上升，但class1的精度有所降低。

### Xgboost的更多评估指数



**ROC 混淆矩阵**

可从混淆矩阵观察到，由于0类样本更多，分类器更能分对0类，并且把更多的1类错分为0类。

实验二

# 实验目的

对任务二中的两组数据dataset1,dataset2分别使用两种不同类别的聚类算法（K-mean和DBSCAN）进行聚类，通过调节不同参数，分析不同参数下聚类的效果。

# 实验环境

操作系统:windows8

CPU：2.5 GHz Intel Core i7

内存：16 GB 1600 MHz DDR3

编程语言：python3.4

# 算法描述

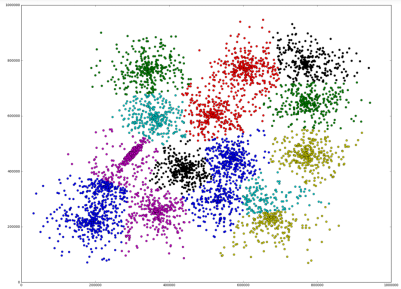
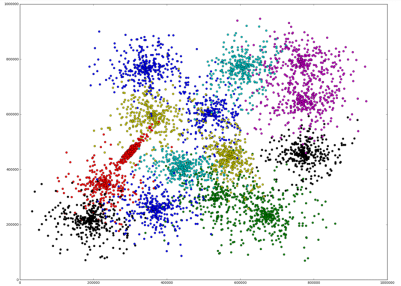
## K-mean

1. 设置参数K
2. 随机选择K个点作为K个簇的中心
3. 根据欧式距离，将所有点分配到K个簇
4. 对每个簇，更新均值
5. 重复（3）-（4）直到每个簇没有变化
6. 计算purity和F-score

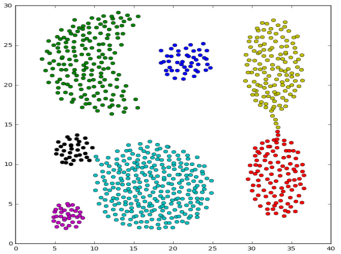
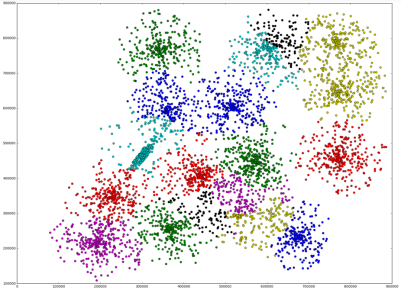
## DBSCAN

1. 设置参数Eps和minPts
2. 根据参数求出所有的core\_points
3. 根据所求出的core\_points，求出border\_points
4. 舍弃掉所有的noise\_points
5. 将相连的core\_point聚到一个类中，并将其相连的border\_point也放入该类中
6. 计算purity和F-score

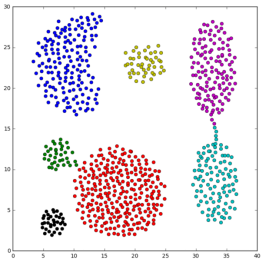
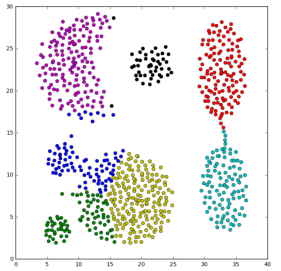
## 聚类效果



dataset1 ground truth图 dataset1\_knn=15聚类图



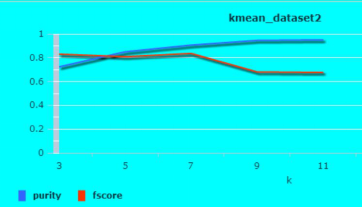
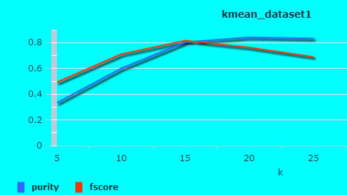
Dataset1\_dbscan聚类图 dataset2 ground truth图



dataset2\_knn=7聚类图 dataset2\_dbscan聚类图

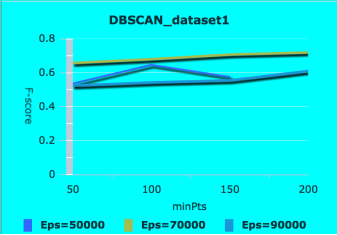
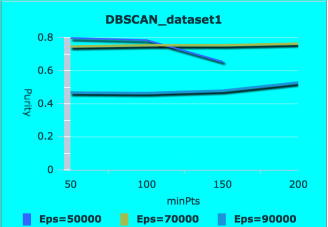
## 评价指标

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| kmean-dataset1 | | | kmean\_dataset2 | | |
| K | Purity | F-score | K | Purity | F-score |
| 5 | 0.333 | 0.4926 | 3 | 0.724 | 0.831 |
| 10 | 0.5992 | 0.71 | 5 | 0.85 | 0.809 |
| 15 | 0.804 | 0.816 | 7 | 0.907 | 0.837 |
| 20 | 0.843 | 0.763 | 9 | 0.946 | 0.682 |
| 25 | 0.8326 | 0.69 | 11 | 0.951 | 0.677 |



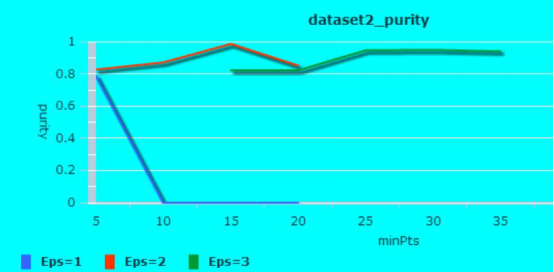
Purity与F-score图

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DBSCAN\_dataset1 Purity | | | | | DBSCAN\_dataset1 F-score | | | | |
| Eps/minPts | 50 | 100 | 150 | 200 | Eps | 50 | 100 | 150 | 200 |
| 50000 | 0.798 | 0.786 | 0.6554 |  | 50000 | 0.537 | 0.647 | 0.5758 |  |
| 70000 | 0.7468 | 0.754 | 0.7532 | 0.764 | 70000 | 0.658 | 0.680 | 0.7072 | 0.720 |
| 90000 | 0.468 | 0.465 | 0.479 | 0.529 | 90000 | 0.524 | 0.543 | 0.555 | 0.610 |



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Eps/minPts | 5 | 10 | 15 | 20 | 25 | 30 | 35 |
| 1 | 0.788 | 0 | 0 | 0 |  |  |  |
| 2 | 0.827 | 0.871 | 0.986 | 0.851 |  |  |  |
| 3 |  |  | 0.822 | 0.822 | 0.946 | 0.950 | 0.940 |

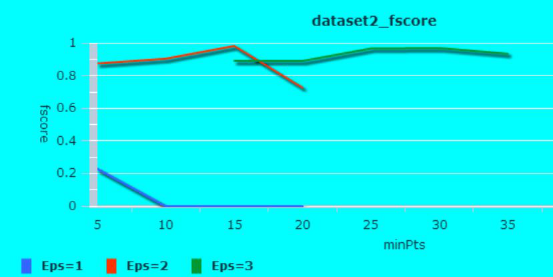
DBSCAN\_dataset2 Purity表



Purity图

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Eps/minPts | 5 | 10 | 15 | 20 | 25 | 30 | 35 |
| 1 | 0.228 | 0 | 0 | 0 |  |  |  |
| 2 | 0.874 | 0.903 | 0.981 | 0.725 |  |  |  |
| 3 |  |  | 0.889 | 0.889 | 0.966 | 0.969 | 0.934 |

F-score表



F-score图

## 结果分析

purity反应了聚类结果中每个簇的“纯度”（每个组中主类所占的比重），但是当把实际的一个类分成了两个类时，纯度一样会很高，F-Score在这种情况下得分则会较低，因此我们在评价聚类效果时，希望purity和F-score能同时达到较好的效果。

1. 由之前的结果可以看出，dataset1使用k-mean方法时，k取值在15-20之间时聚类的效果较好，dataset2由于数据集较小，真是的类别也只有7个，因此聚类的效果也显示在k取7的时候效果较好（虽然k取9以上purity很高，但是F-score会很低，这是因为将一个类分成了多个类）
2. dataset1由于数据较多，相同类中元素距离较远，因此在使用DBSCAN时，参数不是很好确定，仅就进行过的实验数据而言，虽然Eps=50000,minPts=50时purity虽然较高，但是F-score很低，因此不是最佳选择，而当Eps=70000，minPts=150-200时，在纯度能达到较高的水平时，F-score也能达到可接受的水平。当dataset2使用DBSCAN时，由于数据规模较小，数值较小，因此参数比较好选择，所以聚类的效果很好，在Eps=2，minPts=15时，purity和F-score值都达到了0.98以上，说明聚类基本实现了对数据的正确聚集。
3. 对于dataset1，K-mean方法的整体聚类效果要好于DBSCAN，这可能是由于数据规模较大，数据本身的数值离散度也较大，因此DBSCAN的参数选择较困难（这也是K-mean相对于DBSCAN的一个优点）。对于dataset2，DBSCAN的方法明显优于K-mean，DBSCAN进行适当的参数选择，很容易就能使purity和F-score同时达到95%以上

实验三

# 概述

本问题是通过对可穿戴设备的传感器的三维时序数据进行挖掘，对用户的活动进行分类。该问题可用于实时预测使用者当前的活动状态，或是收集数据后离线分析。其中实时预测的难度更高，但也更有实用意义，这样设备可以在数据收集端就可以分析用户的活动，并实时做出反馈。

因此，我们更关注“观察用户几秒钟后，预测这几秒用户最可能的活动是什么”这类问题。我们在数据分析、预处理的基础上，采取了数值统计信息、信号频谱这两种特征提取方法，并使用了三种分类器来解决用短时间内数据分类用户活动的问题。

实验证明，对于第一个子问题，我们的最优方法在所有数据集上10折交叉验证有72.94%的准确度(72.09%加入DWT特征)，5折交叉验证有69.00%的准确度，使用前12个数据集预测后三个数据集的样本达到了72.92%(加入DWT特征)；对于第三个子问题，通过前12个用户的数据预测后3个用户的每个时间点的类标，我们达到了平均70%(71.39%加入DWT特征)的准确率，并且对于每个一小时的数据，平均预测耗时为20秒(15秒加入DWT)，可以在数据收集端完成实时地活动分类。除此之外，我们还尝试了卷积和循环神经网络在本问题上的应用，结合未处理的XYZ信号和数值统计特征，也达到了5折交叉验证63%左右。

# 问题定义

**原始问题一**

原问题定义中样本的维度为3，时间跨度为k，而k是未定义。我们取k为300-600，即约为6-12秒的数据，因此我们对于每一个原始数据中长段的活动，切分为小段的、拥有一致类标的样本，并基于这些样本进行学习和分类。

**原始问题二**

本问题可以基于问题三“多活动类别识别做出”，即如果我们已经能做到对每个时间点较为精确地打点，活动发生转换的时间点非常容易求出，在本次实验中，我们集中精力追求问题三的准确度。

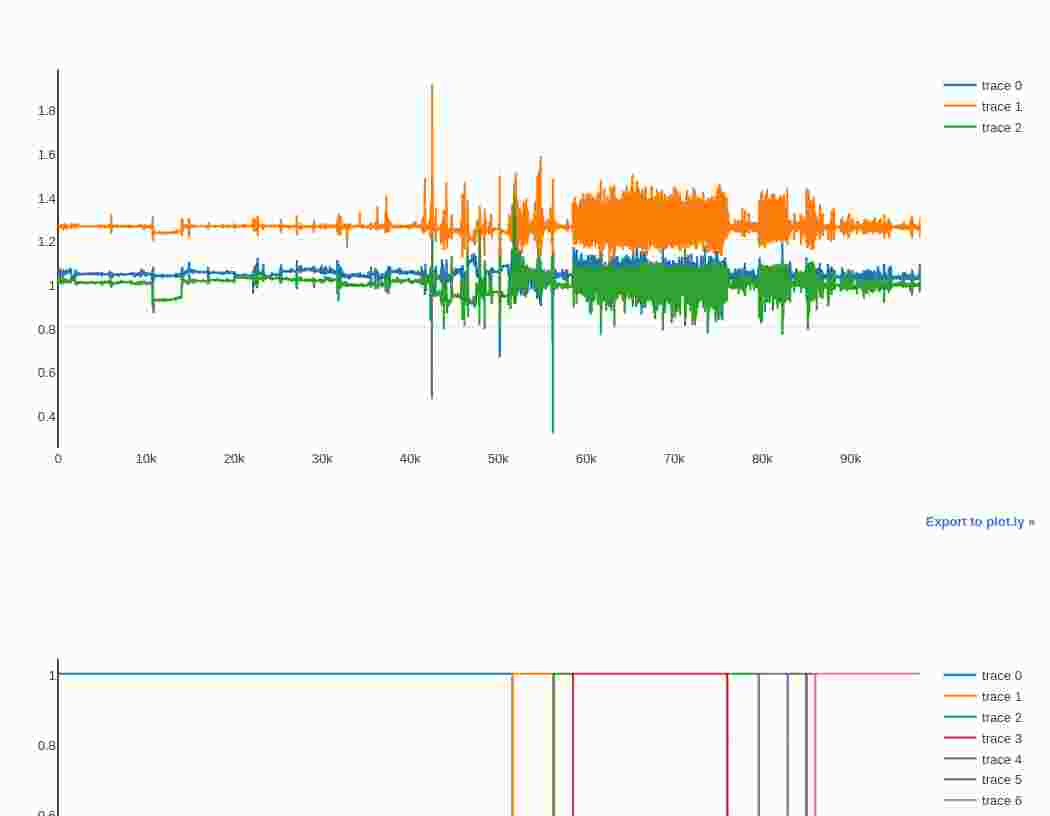
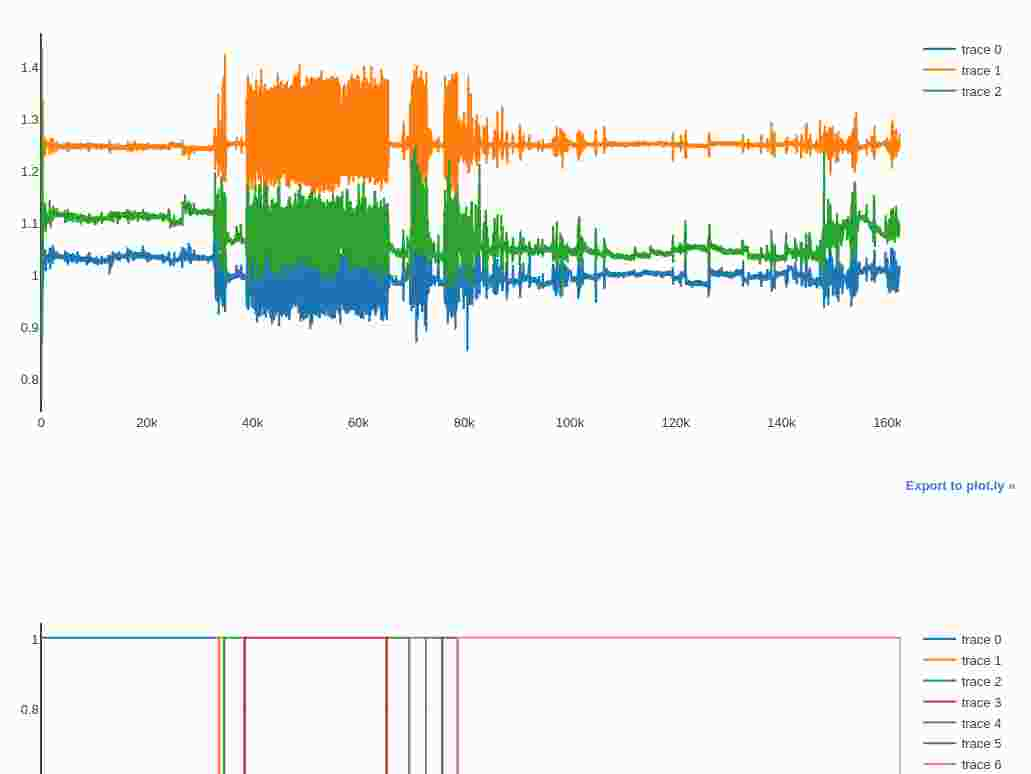
**原始问题三**

我们沿着问题一的思路，首先将训练集切分为6-12秒小样本进行训练，再对未标记的数据集，即整个一小时的活动数据，采用60帧数据（约1秒）的滑动窗口，用训练后的分类器不断对滑动的数据进行类别标注。

# 数据分析

我们将XYZ和Label数据可视化，对数据有了以下的分析结果：

* 某些活动之间存在明显区别，而某些活动间区别较小。从图1a中，可以明显观察出1与3,4类，4与7类直接有明显的差异，比如1非常平稳，4波动剧烈，而7平稳中有一些波动，但从图3观察，3,4两类在局部放大图中比较相似，有一些微小的模式差异。总体看来，信号呈现出多种频率波组合的特性。
* 对于一个用户，每一类的活动时间并不均衡。观察用户1可以发现，1,4,7三类活动占据了大部分时间段，而有一些活动持续的时间可能不足几分钟，样本的数量差异可能会引发分类器更倾向于将所有数据划分到大类中，造成小类的分类效果较差。
* 不同用户的活动数据存在差异，比如类别的分布情况和每一类的模式。从图1看出，用户1的活动比较剧烈，且其第7个活动占时最长，而用户15活动比较平缓，其第1个活动耗时最长，并且在第一个活动中也存在一些突发的剧烈运动，可能噪音比较大。
* 第2,4,6,8,9,13,15用户的XYZ数据和Label直接存在一个时间差，并不是完美对齐的。如图3所示，存在偏差后，一些持续时间短的活动可能类标大部分都打错了。



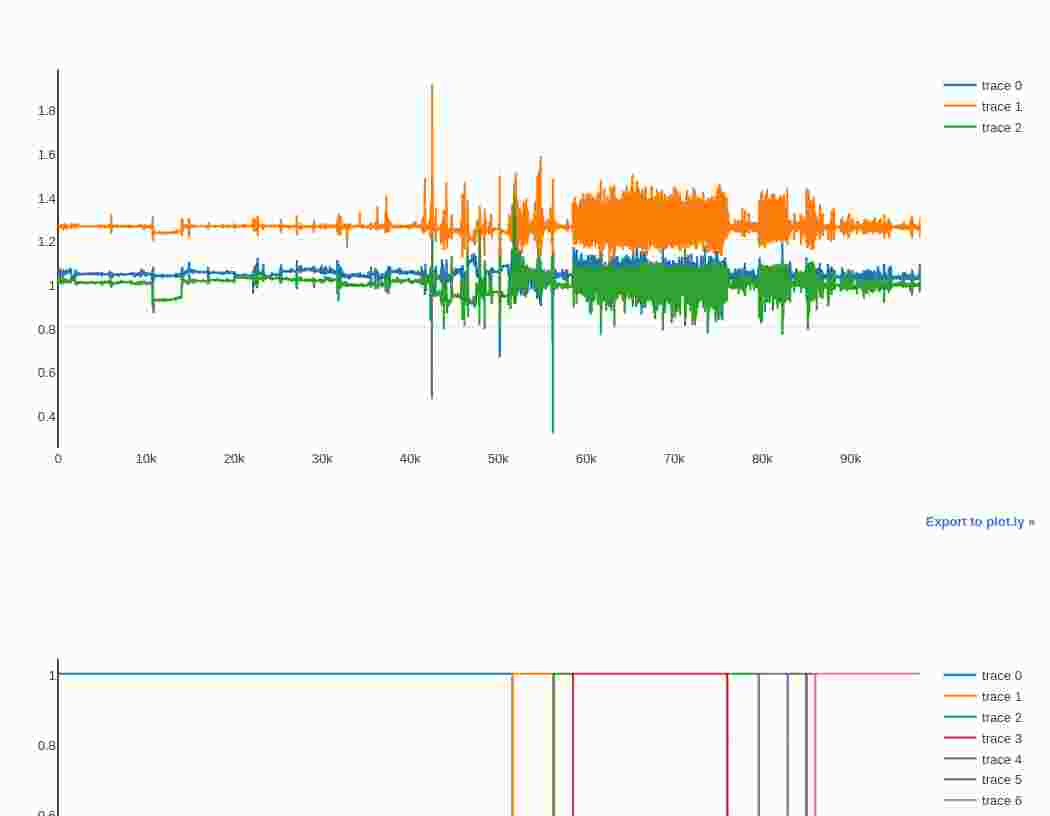
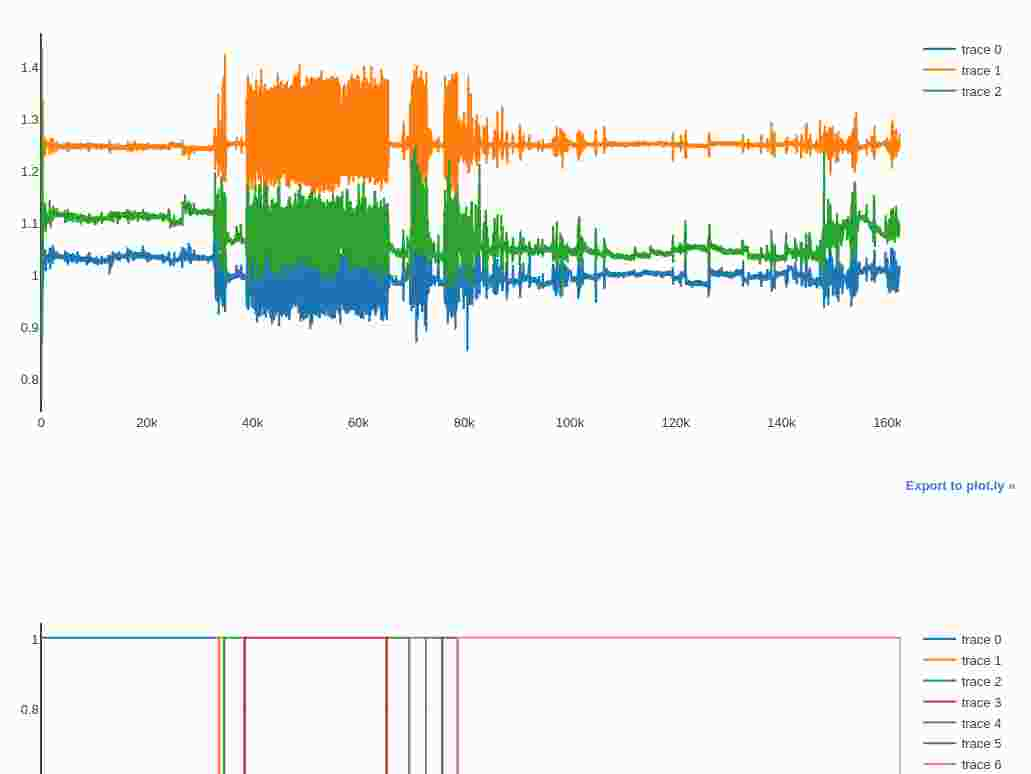
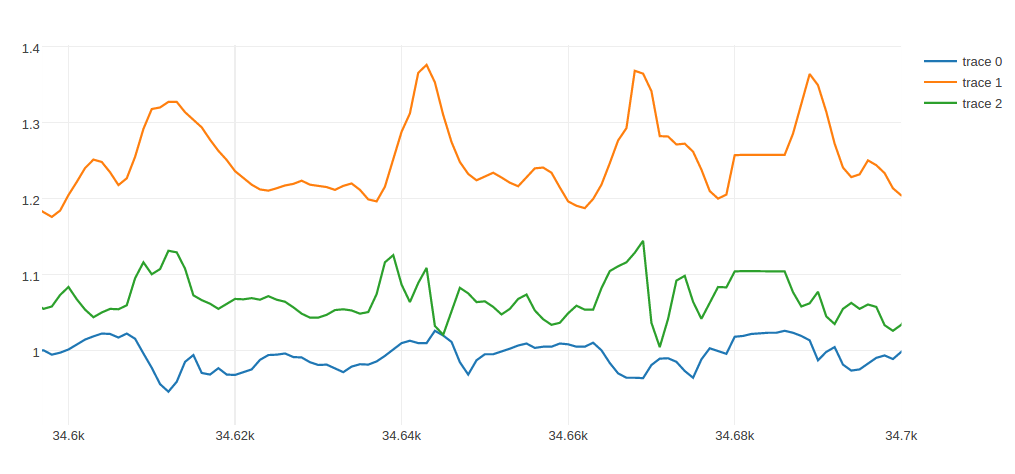
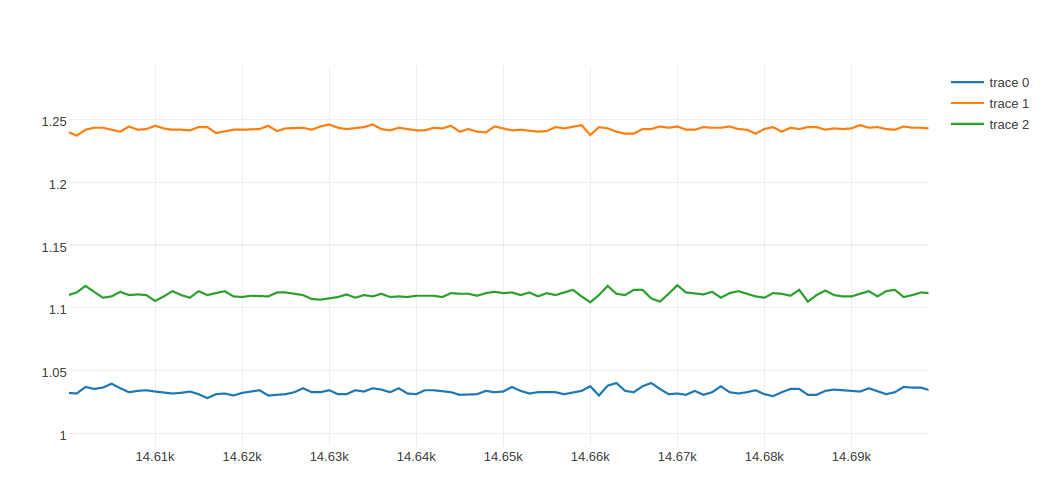


图1：数据集1(左a)、数据集15(右b)的缩放后XYZ与Label随时间变化图



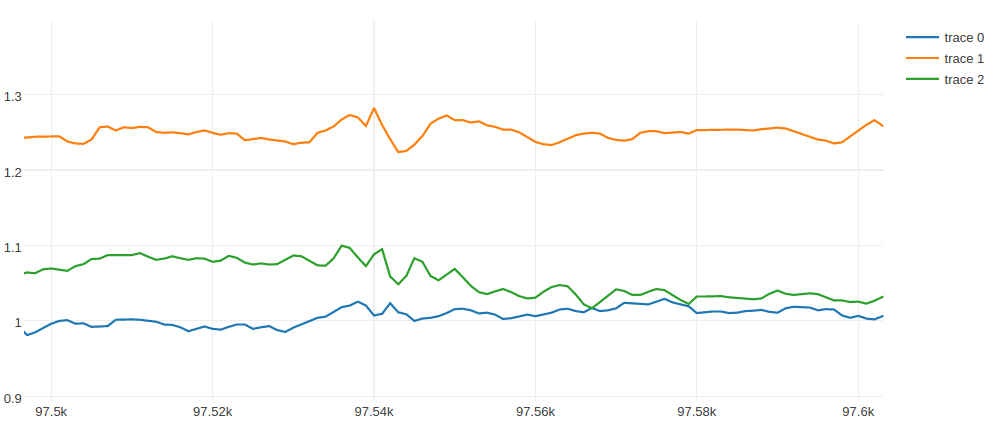
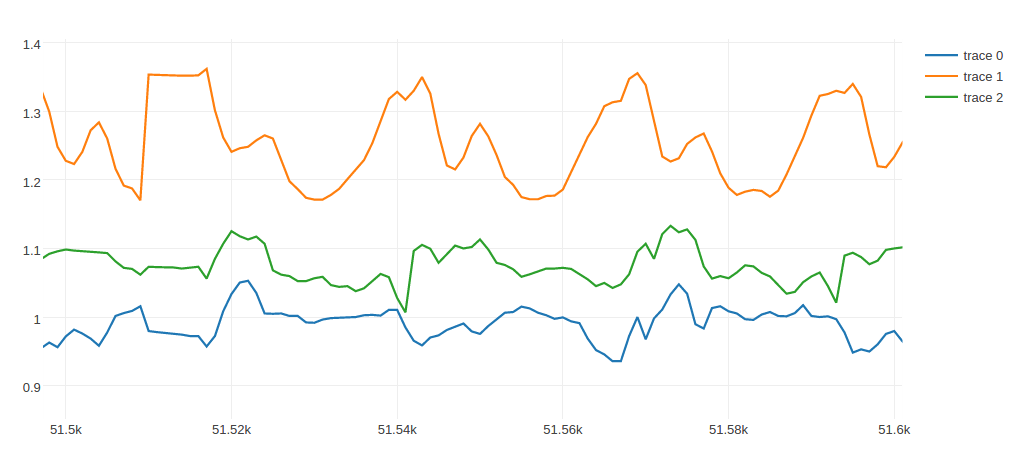


图2(左)：数据集7的缩放后XYZ与Label随时间变化图

图3(右)：数据集1不同活动的100数据点图，分别为活动1,3,4,7

# 数据预处理

## 数据矫正

以图2为例，通过观察我们发现，XYZ信号在开始时有较为异常的波动，并且后续的数据均比label“迟”一些，我们尝试将XYZ的头部3000多个数据噪声删去，并删去label尾部多余的3000多点数据。人工可观察得出矫正后的数据，活动转换点和label的转换点吻合度很高。实验证明，在未矫正数据的情况下，问题一中全部数据集10折交叉验证只有66.76%，低于矫正后72.94%的准确率。

## 数据缩放

观察原始数据后发现，不同使用者的XYZ信号的基线不一致，这可能影响分类器对不同用户的数据的泛化性能，在综合比较了传统的均一化等操作后，采取了使用原有信号数据与三维中最小的一个平均值的比值作为缩放处理后的信号。通过这一操作，可以将不同用户未校准的信号统一到一个近似的数值范围区间内，同时也能保持原有信号的特性。

## 低通滤波

低通滤波器（Low-pass filter）容许低频信号通过，但减弱（或减少）频率高于截止频率的信号的通过。考虑到人体活动的频率不可能过高，采样率52Hz表示通过FFT变换后的数据，只能分析出0-26Hz的活动频率，但人的活动频率可能不会超过10Hz，高频的波动可能是传感器抖动或是噪音干扰，因此，将原始数据通过一个低通滤波器，过滤高频干扰。

图4显示了原始信号和通过一个6Hz/10级的低通滤波器的信号，可以明显观察出在处理后的信号最大程度地保留了信号波形的大致形状，但过滤掉了高频抖动。在实际实验中，我们发现截止频率限制在3Hz下效果更好一些。

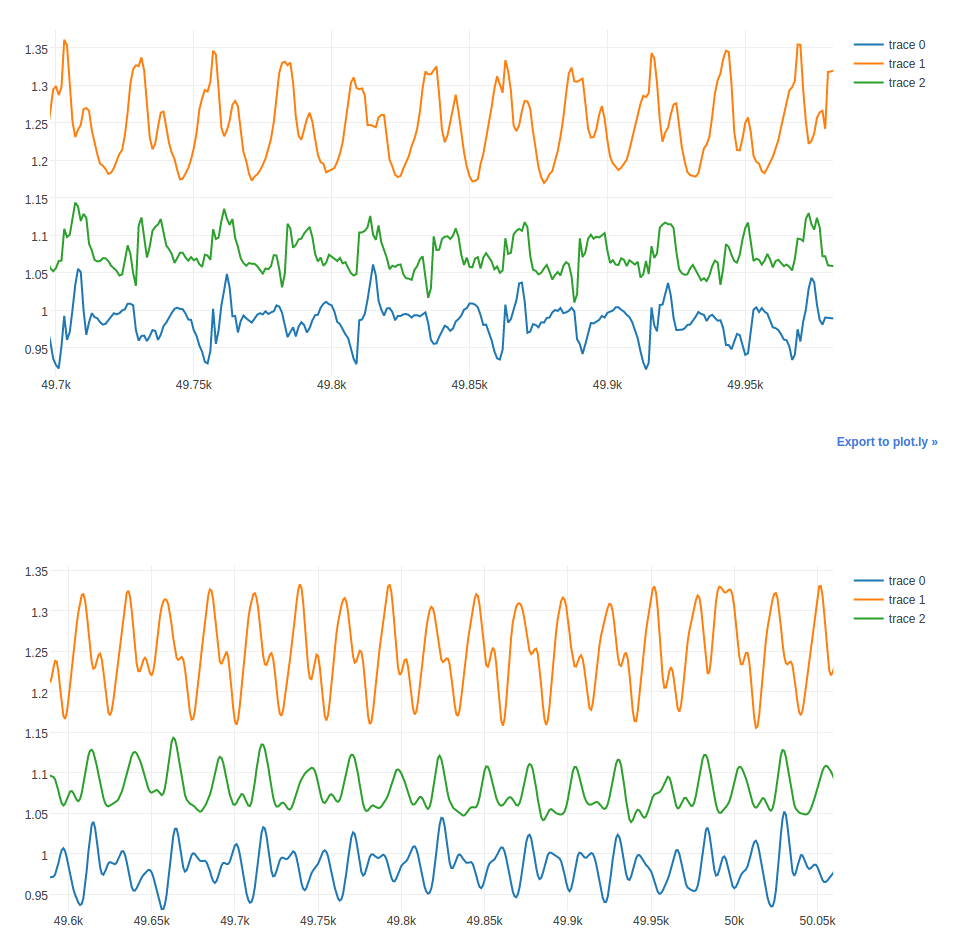
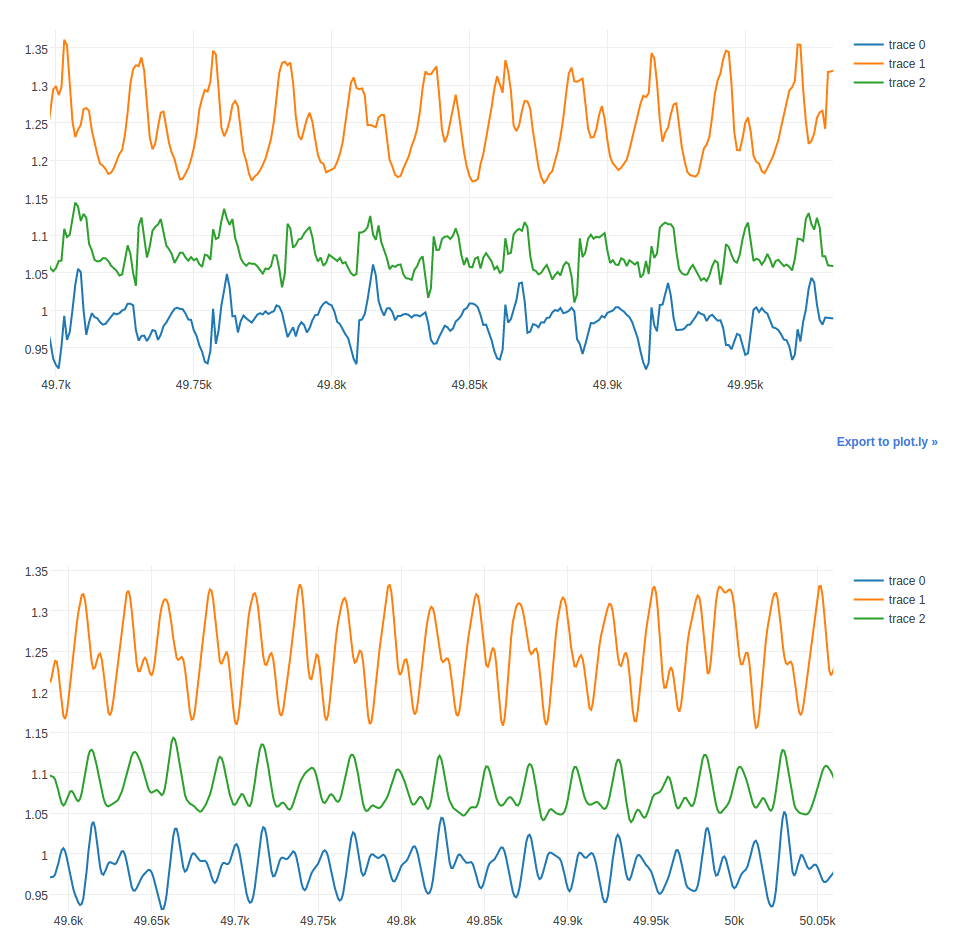


图4：未经过低通滤波器的原始信号，经过6Hz/10级低通滤波器的信号

# 分类方法

## 基于信号频谱和统计特征，使用集成学习分类

### 特征提取

**快速傅里叶变换**

傅里叶分析将信号从原始域（通常是时间或空间）转换到频域的表示或者逆过来转换。对于XYZ的传感器的时域数据，我们观察出这些信号数据呈现出明显的周期性，也就是可能是多种不同频率的波的叠加，因此使用快速傅里叶变换向频域转换。考虑到人体活动通常也是具有周期性的特点，比如走路的身体周期性摆动等，人体的一个活动可能可以以多种不同频率的波来进行近似表示。将600个点的信号数据归一化后，进行快速傅里叶变换，取XYZ每一维频谱图上的前60个数据点的均一化结果，可以较好地刻画不同活动的特征。

**离散小波变换**

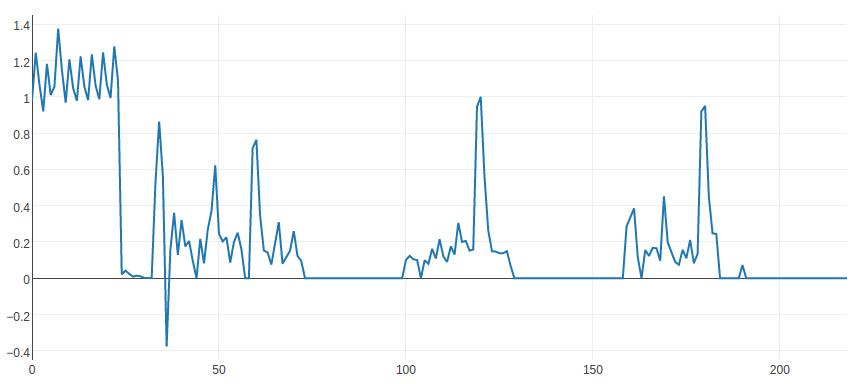
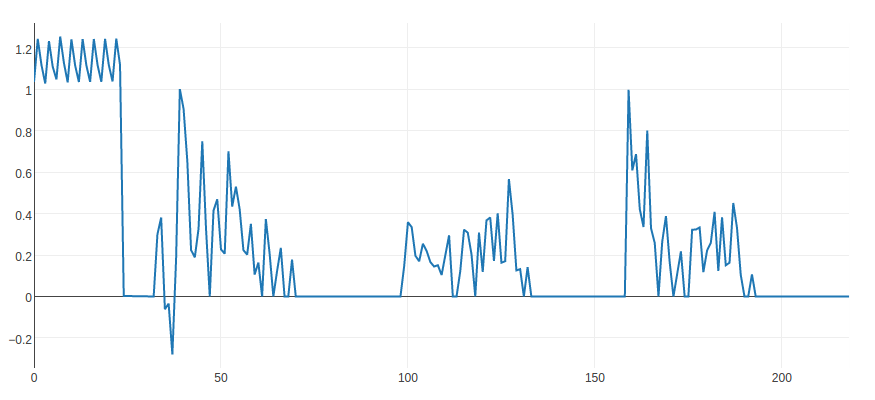
离散小波变换此处不加说明，我们使用了每个样本，三轴信号经过12Hz滤波器后，rbio3.3的小波前30位变换结果作为特征。

**统计信息**

对于每一个600个数据点的采样，我们还提取了以下的数据统计特征：三轴平均数，三轴最大值，三轴最小值，三轴的四分位数，三轴的标准差，三轴的平均逐点的绝对差（前后点对间差的绝对值的平均值），X-Y,Y-Z,Z-X信号间的相关系数（corr），三轴的无偏差标准误差（sem），三轴的无偏斜值（skew），三轴的均方根（RMS），三轴的直方分布数据的均一化。

**特征合并**

上述特征向量的维度为3\*12+3\*10+3\*60+3\*30=246(276)维数据，将上述特征合并到一起，形成了最终输入分类器的特征向量。如图5所示，每一个特征向量的前段为统计信息特征，后部分为快速傅里叶变换变化得到的XYZ的局部频谱信息。肉眼观察可以发现，3,4活动的统计特征和频域特征和1,7有较为明显的不同，从频域信息来看，3,4活动在较高的频率出现尖峰，并有几个辅助的频率峰值，而1,7活动的频率主要集中在低频，并且能量强度分布比较均衡。这和3,4活动更加剧烈，因此人体活动存在着更高频率摆动的事实是吻合的。



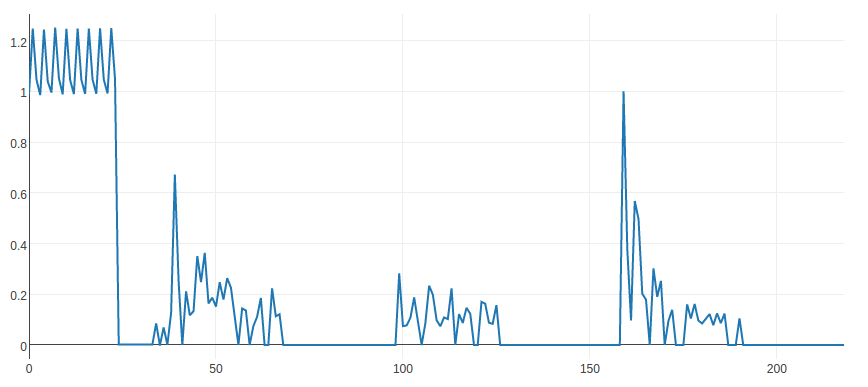
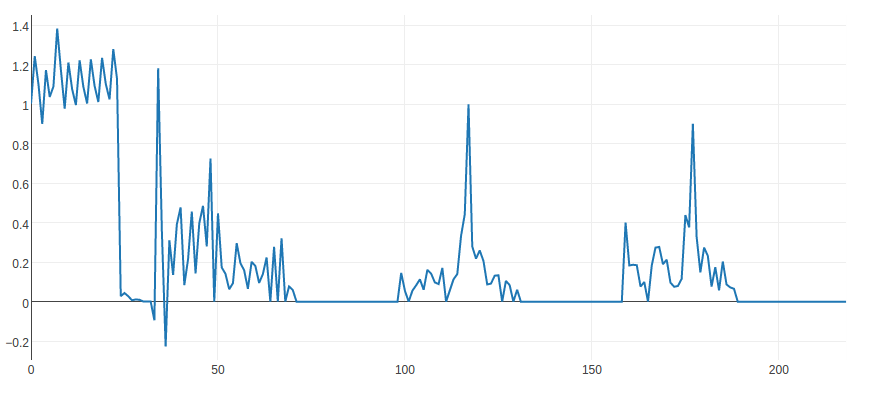


图5：可视化活动1,3,4,7的特征向量（未包含DWT）

### 分类

在尝试了Adaboost，XgBoost等集成学习方法后，随机森林分类器表现较好，可调参数多并且可并行学习。接下来解释如何使用该分类器解决问题一和三。

**单样本分类（问题一）：**是一个标准的分类问题，使用Sklearn提供的训练、测试和交叉验证框架即可。

**活动数据逐点标记（问题三）：**由于问题一获得的分类器是以600个数据点提取样本进行标注的，因此将问题一的分类器运用到问题三的逐点标注，需要使用一个更小的滑动窗口才能获得更好的效果，具体步骤如下：

* 确定样本的数据点数为600，以75个数据点为滑动步长，原有的一个样本被切分为8份
* 对于数据集的连续序列，从起点开始，每次读头向后600个数据点，提取特征进行类别的预测，并记录下每次预测每一类的可能性，而不是记录下最可能的一类
* 滑动读头75个数据，提取600个数据点的特征进行预测，对于靠近读头的75个点，依据他们标注的8次类别的概率结果，取概率最大的一类作为这75个数据点的类标
* 在标注完所有数据后，人工平滑Label序列，例如以600个数据点以下的Label段为噪音，使用前一段Label值进行覆盖

## 基于滤波后的信号，使用卷积+循环神经网络分类

近年来，卷积和循环神经网络在人体活动分类上有了更多的应用。论文[1]使用了卷积与循环神经网络的叠加识别人体活动，论文[2]单独使用循环神经网络，并可以以高准确率、低延时预测6类人体活动，并且不需要进行样本的窗口化裁切。鉴于时间原因，我们还没有对文章详细研究，但这可以作为未来提升准确率的另一种思路。我们使用Keras和Tenserflow试验了简单的卷积+循环神经网络，虽然分类效果不如人工提取特征与随机森林分类器的组合，但神经网络“端到端”的特性值得一提，即我们输入的是仅仅经过滤波、分段、均一化的XYZ信号，而并没有特征提取阶段，特征提取由神经网络的训练过程中，在卷积核参数、LSTM存储和网络的参数中被自动学习出来。

**预处理**

对于本问题，我们首先对数据集做如下处理：缩放到设备的基准线、经过6Hz/10级的低通滤波器、按照300个数据点进行切分、生成每个样本的维度为(300, 3)，即按照时间线有300个数据点，每个数据点有XYZ三个维度，(300, 3)也为神经网络的输入层。

为了进一步加强神经网络的分类能力，我们也可以对于300个时间帧的数据提取一系列的数值统计特征，与上述提到的XYZ信号一起，使用两个神经网络的输入层和隐藏结点分别处理，最后将两个网络的输入连接到统一的隐藏层后训练输出。实验证明上述结合的方法，可以将5折交叉验证的准确率由55%左右提升至63%左右。

**表1：针对原始信号输入的网络设计**

|  |  |  |
| --- | --- | --- |
| 层类型 | 需要学习的参数数量 | 输出维度 |
| 卷积层，128个一维卷积核，每个长度为3 | 1280 | (300, 128) |
| 一维MaxPooling，长度为2 | 0 | (150, 128) |
| 卷积层，128个一维卷积核，每个长度为3 | 49280 | (150, 128) |
| 一维MaxPooling，长度为2 | 0 | (75, 128) |
| 卷积层，128个一维卷积核，每个长度为3 | 49280 | (75, 128) |
| 一维MaxPooling，长度为2 | 0 | (37, 128) |
| LSTM单元，共96个 | 86400 | 96 |
| 稠密结点，共64个 | 6208 | 64 |
| 稠密结点，共48个 | 3120 | 48 |
| 稠密结点，共32个 | 1568 | 32 |
| 稠密结点，共7个 | 231 | 7 |

**表2：针对统计信息输入的网络设计**

|  |  |  |
| --- | --- | --- |
| 层类型 | 需要学习的参数数量 | 输出维度 |
| 稠密结点，共32个 | 1088 | 32 |
| 稠密结点，共24个 | 792 | 24 |
| 稠密结点，共24个 | 600 | 24 |
| 稠密结点，共7个 | 399 | 7 |

针对上述两个网络输入，最后使用7个稠密结点构成的输出层连接后输出。

# 评估与分析

## 实验环境与源码地址

CPU：Intel Core i7 6700HQ @2.6Ghz up to 3.5Ghz

Mem：2x8G @2133Mhz

Python 3.4

本实验使用Jupyter Notebook 4.2.3方便进行数据的互动和画图，使用了Sklearn、Keras的机器学习模型和评估框架，Pandas进行数据处理，Seaborn和Plotly可视化。

**实验的原始数据和数据处理代码**

基于信号频谱和统计特征，使用集成学习分类：result/3-3-using-seg-fft-and-summary-to-solve-3-final.html

基于信号频谱、DWT和统计特征：result/3-3-using-seg-fft-dwt-and-summary-to-solve-3-final.html

基于信号频谱，使用卷积神经网络分类：result/3-1-use-small-seg-rnn-solve-1-final.html

## 单一活动类别识别

**评估标准**

* 使用所有数据集进行5/10倍交叉验证，查看准确率和混淆矩阵，其中5折交叉验证大致相当于使用12个人预测3个人
* 使用前12个数据集预测后3个数据集，查看准确率和混淆矩阵

**基于信号频谱和统计特征，使用集成学习分类**

参数：随机森林参数为400个弱分类器，使用信息熵为学习标准，限制子树最大深度为16层。

结果：对所有的数据集做5折交叉验证 69.00%，对所有的数据集做10折交叉验证 72.94%，使用前12个数据集预测后3个数据集71.75%。任务一的10折交叉验证混淆矩阵、和测试集的混淆矩阵如图6所示。

**基于信号频谱，DWT和统计特征，使用集成学习分类**

参数：随机森林参数为600个弱分类器，使用信息熵为学习标准，限制子树最大深度为14层。

结果：对所有的数据集做5折交叉验证 69.00%，对所有的数据集做10折交叉验证 72.09%，使用前12个数据集预测后3个数据集72.91%

**基于滤波后的信号，使用卷积+循环神经网络分类**

参数：网络结构设计如表1和表2所示。

结果：对所有的数据集做5折交叉验证 63%，使用前12个数据集预测后3个数据集64.90%。任务一的10折交叉验证混淆矩阵、和测试集的混淆矩阵如图7所示。

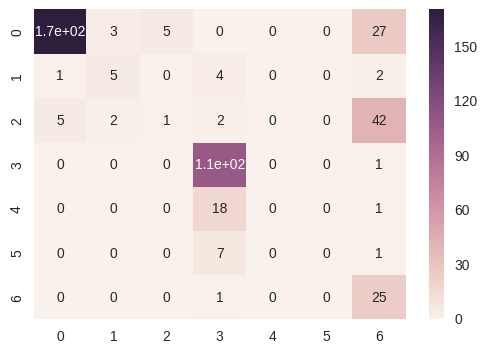
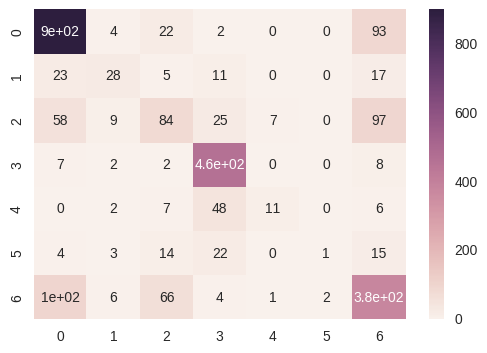


图6：使用随机森林，任务一的10折交叉验证混淆矩阵、和测试集的混淆矩阵

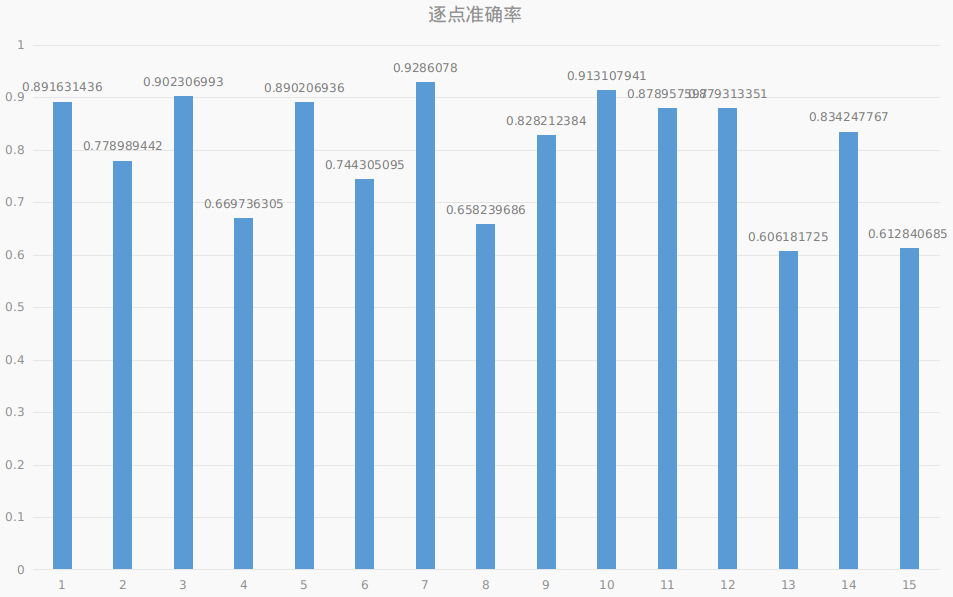
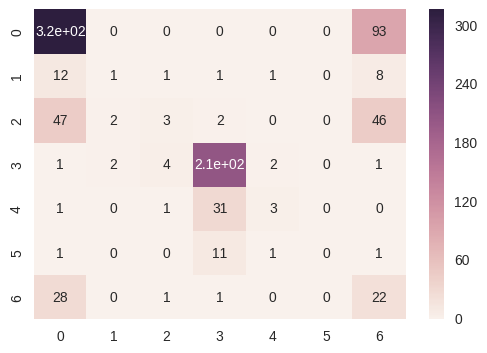
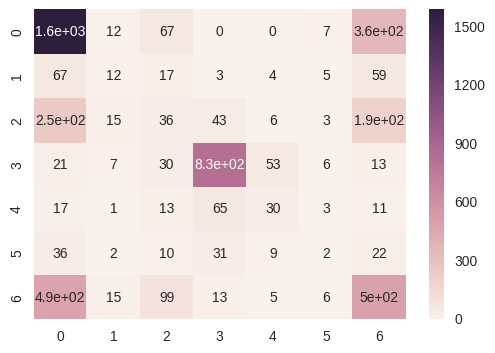


图7：使用神经网络，任务一的5折交叉验证混淆矩阵、和测试集的混淆矩阵

图7（右）：任务三各个数据集上的逐点标注准确度，前12个为训练集，后3个为测试集

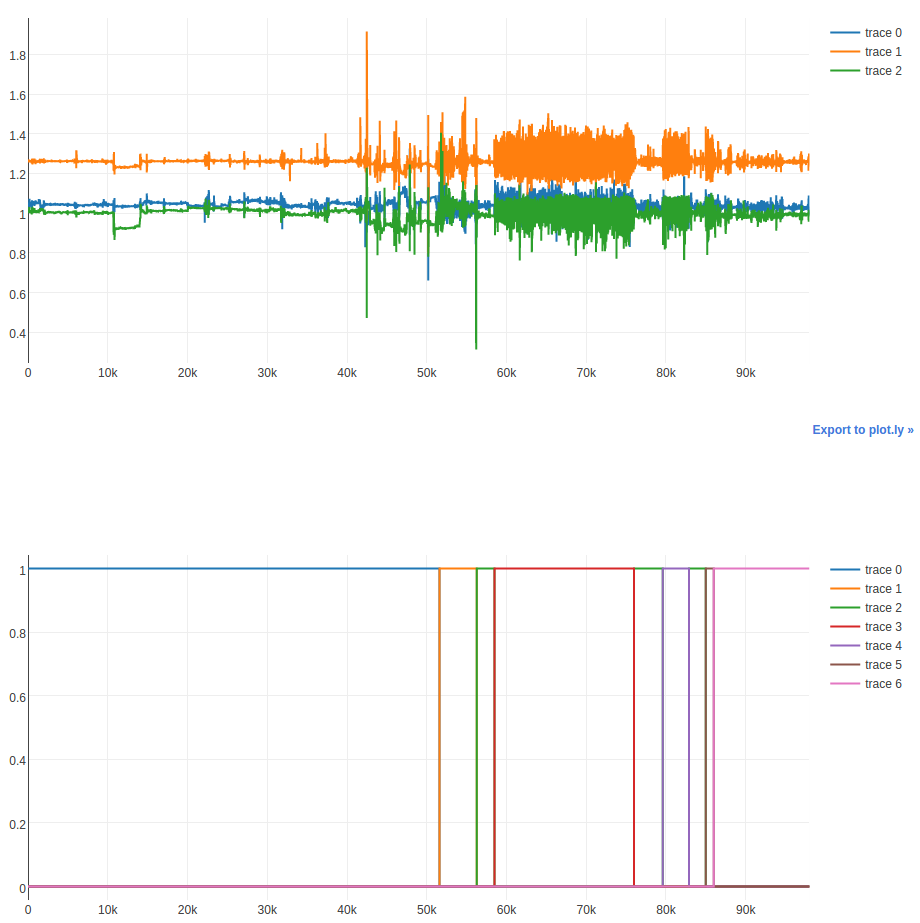
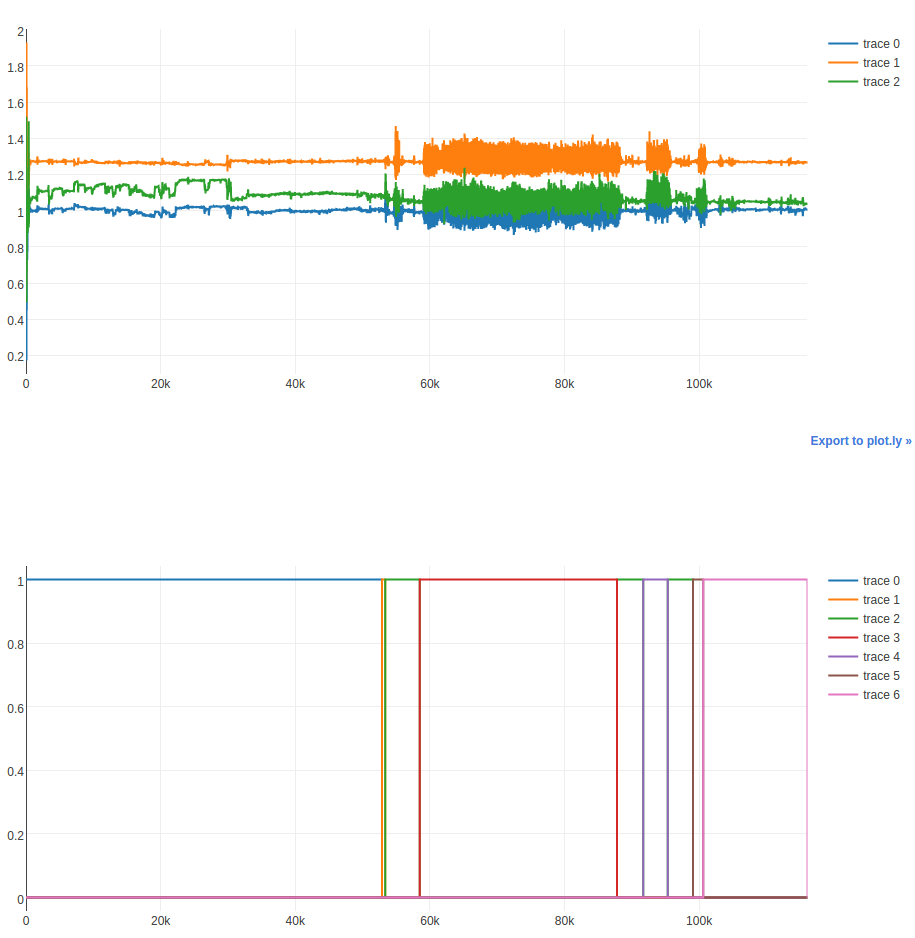
## 活动转换点检测

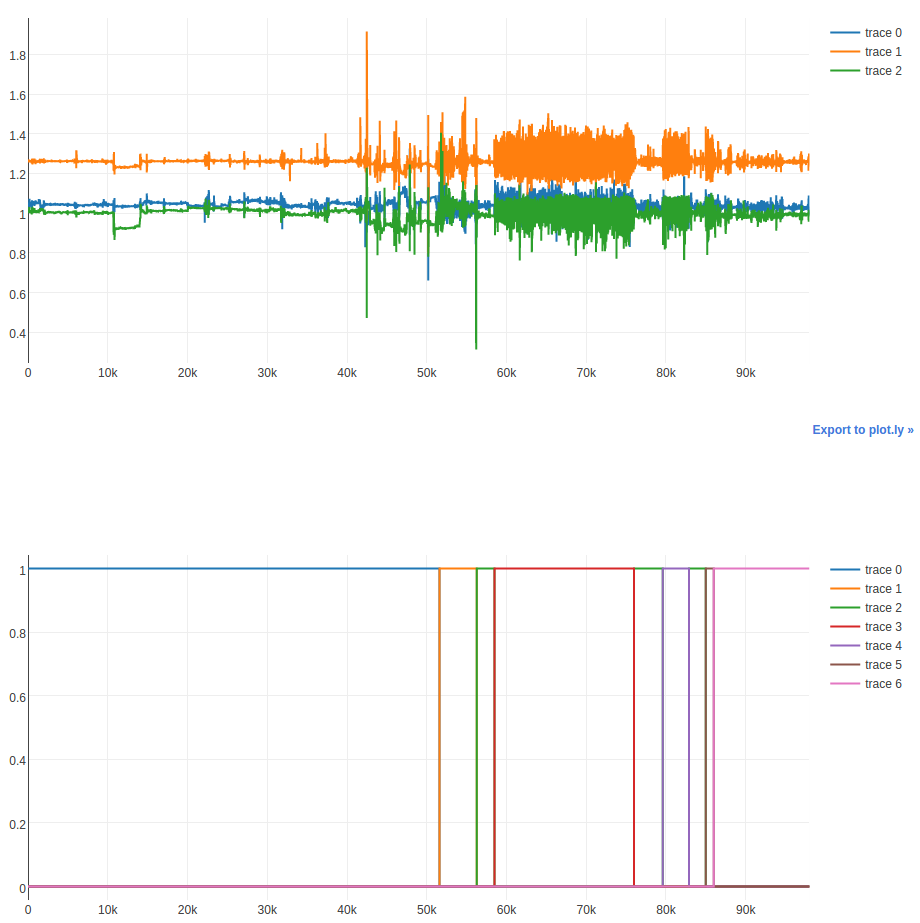
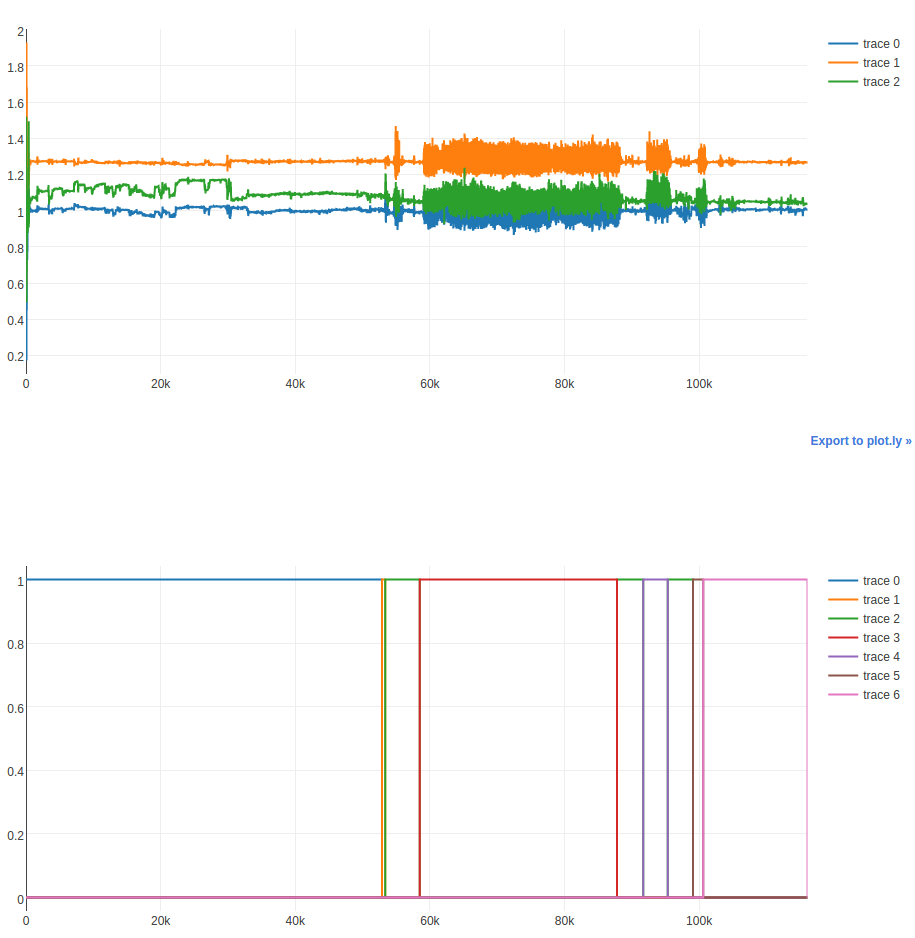
*可以使用第三问的预测结果做活动的转换点检测。*

## 多活动类别识别

指标：使用前12个数据集，标注后3个数据集的所有数据点，与真实结果相比的准确度。

结果：基于信号频谱和统计特征，对后三个数据集的逐点预测准确度分别为60.62%,83.42%,61.28%，平均约为70%。基于信号频谱，DWT和统计特征，对后三个数据集的逐点预测准确度分别为60.62%,83.37%,64.51%，平均约为71.39%。





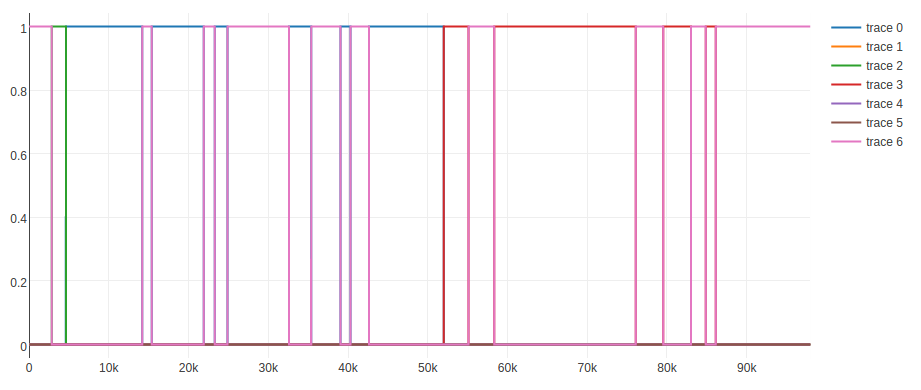
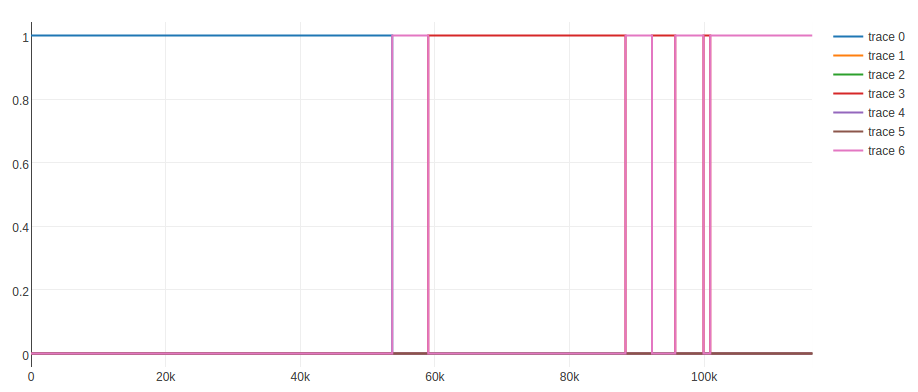


图8：任务三中两个测试集的逐点标注与正确结果的对比分析

通过图8，可发现左图的数据集分类器的逐点标注和正确结果比较接近，其中1,4,7三个大类均几乎分类正确，虽然对于一些样本较小的活动分类效果不佳，也达到了83%的正确率，而且直观来看，活动的转换点的标注也正确或是十分接近正确答案。而右图的用户只达到了61%的准确度，其中4和7类基本分类正确，1类被频繁地误认为7类，造成了不少错误的活动转换点，其他活动转换点虽然检测正确，但是小类的分类准确度依然较低。

# 参考文献

[1] Ordóñez, Francisco Javier, and Daniel Roggen. "Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition." Sensors 16.1 (2016): 115.

[2] Inoue, Masaya, Sozo Inoue, and Takeshi Nishida. "Deep Recurrent Neural Network for Mobile Human Activity Recognition with High Throughput." arXiv preprint arXiv:1611.03607 (2016).

# 总结

本次实验中，我们试图通过学习和分类短时间片内的样本，来统一性地解决本实验的三个问题。我们首先使用矫正、均一化等多种预处理方法，提取了数值统计特征、频谱特征和XYZ三维裸信号三类特征，使用随机森林分类器和神经网络验证训练和分类结果，实验证明最优的分类器5折交叉验证接近70%的准确率，对测试集合有71%的分类准确度。接下来提出了如何通过短时间片样本来实现问题三，即对每个时间点进行标注，实验发现我们对后三个使用者的逐点标注正确率达到了71.39%左右。

感谢助教批阅。