关联规则挖掘报告

王鹏　　　　2120171069

一、对数据集进行处理

本次作业选用数据集 NFL Play by Play 2009-2017 (v4)，利用 Python 中的 pandas 库进行 csv 数据文件的读取，对数据集进行处理，转换成适合关联规则挖掘的形式：

```
df = pd.read_csv('NFL Play by Play 2009-2017 (v4).csv', dtype=str)
for col in df:
    df_counts = df[col].value_counts()
    num = df_counts.max()
    name = df_counts.idxmax()
    if pd.isnull(name) or name == 'None' or num < len(df)/2  or num > len(df)*0.99:
        df.drop(col, axis=1, inplace=True)
df.fillna(method='ffill')
```

对于数据集中存在过多“NA”和“None”的字段进行剔除，众数小于文件长度一半的字段进行剔除，并且，对于大多数值都为同一个的字段（例如，对于 PlayAttempted 属性，它的值全为 1），我们认为其没有研究意义，应将其剔除。然后用 ffill 方法对缺失值进行简单的填充处理

二、找出频繁项集

Apriori 算法的两个输入参数分别是最小支持度和数据集。该算法首先会生成所有单个数据字段值的项集列表。接着扫描每行数据来查看哪些项集满足最小支持度要求，那些不满足最小支持度的集合会被去掉。然后，对剩下来的集合进行组合以生成包含两个元素的项集。接下来，再重新扫描每行记录，去掉不满足最小支持度的项集。该过程重复进行直到所有项集都被去掉。

2.1 生成候选项集

```python
# 生成初始候选频繁项集C1
def createC1(dataSet):
    C1 = []
    for transaction in dataSet:
        for item in transaction:
            if [item] not in C1:
                C1.append([item])
    C1.sort()
    return list(map(frozenset, C1))

def scanD(D, Ck, minSupport):
    ssCnt = {}
    for tid in D:
        for can in Ck:
            if can.issubset(tid):
                ssCnt[can] = ssCnt.get(can, 0) + 1
    numItems = len(D)
    retList = []
    supportData = {}
    for key in ssCnt:
        support = ssCnt[key] / numItems
        if support >= minSupport:
            retList.append(key)
            supportData[key] = support
    return retList, supportData
```

2.2 生成频繁项集

生成频繁项级算法的伪代码如下：

当集合中项的个数大于 0 时

构建一个 k 个项组成的候选项集的列表

检查数据以确认每个项集都是频繁的

保留频繁项集并构建 k+1 项组成的候选项集的列表

具体相关方法如下：

```python
def aprioriGen(Lk, k):
    retList = []
    lenLk = len(Lk)
    for i in range(lenLk):
        for j in range(i+1, lenLk):
            L1 = list(Lk[i]) ; L2 = list(Lk[j])
            L1.sort() ; L2.sort()
            if L1[:k-2] == L2[:k-2]:
                c = Lk[i] | Lk[j]
                if has_infrequent_subset(set(c), Lk): continue
                else: retList.append(c)
    return retList

def apriori(dataSet, minSupport=0.5):
    C1 = createC1(dataSet)
    D = list(map(set, dataSet))
    L1, supportData = scanD(D, C1, minSupport)
    L = [L1]
    k = 2
    while len(L[k-2]) > 0:
        Ck = aprioriGen(L[k-2], k)
        Lk, supK = scanD(D, Ck, minSupport)
        supportData.update(supK)
        L.append(Lk)
        k += 1
    return L, supportData
```

函数最后返回的是频繁项集列表以及每个频繁项的支持度。

三、导出关联规则，计算置信度和提升度

根据频繁集学习关联规则，针对规则右部的元素个数进行分级，导出关联规则，并计算关联规则的提升度和置信度：

```python
def generateRules(L, supportData, minConf=0.7):
    bigRuleList = []
    for i in range(1, len(L)):
        for freqSet in L[i]:
            H1 = [frozenset([item]) for item in freqSet]
            if i > 1:
                rulesFromConseq(freqSet, H1, supportData, bigRuleList, minConf)
            else:
                calcConf(freqSet, H1, supportData, bigRuleList, minConf)
    return bigRuleList
```

```python
def calcConf(freqSet, H, supportData, br1, minConf=0.7):
    prunedH = []
    for conseq in H:
        conf = supportData[freqSet] / supportData[freqSet-conseq]
        lift = conf / suppData[conseq]
        if conf >= minConf:
            print(freqSet-conseq, '-->', conseq, 'conf:', conf)
            print(freqSet-conseq, '-->', conseq, 'lift:', lift)
            br1.append((freqSet-conseq, conseq, conf))
            prunedH.append(conseq)
    return prunedH
```

四、规则挖掘结果及评价

下图为部分规则挖掘结果及使用 Lift 评价的结果：

```
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Post_3'}) conf: 0.9853377663032373
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Post_3'}) lift: 1.5144709225654265
frozenset({'AwayTimeouts_Remaining_Post_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) conf: 1.0
frozenset({'AwayTimeouts_Remaining_Post_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) lift: 1.5144709225654265
frozenset({'AwayTimeouts_Remaining_Post_3'}) --> frozenset({'posteam_timeouts_pre_3'}) conf: 0.8878558933526361
frozenset({'AwayTimeouts_Remaining_Post_3'}) --> frozenset({'posteam_timeouts_pre_3'}) lift: 1.3357104923343022
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Post_3'}) conf: 0.869033517470931
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Post_3'}) lift: 1.3357104923343022
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'posteam_timeouts_pre_3'}) conf: 0.8893330113857983
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'posteam_timeouts_pre_3'}) lift: 1.337932702120916
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) conf: 0.8834324133833715
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) lift: 1.337932702120916
frozenset({'AwayTimeouts_Remaining_Post_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) conf: 0.7685298286886236
frozenset({'AwayTimeouts_Remaining_Post_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) lift: 1.139562349238429
frozenset({'HomeTimeouts_Remaining_Pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Post_3'}) conf: 0.7414165587674761
frozenset({'HomeTimeouts_Remaining_Pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Post_3'}) lift: 1.139562349238429
frozenset({'HomeTimeouts_Remaining_Post_3'}) --> frozenset({'posteam_timeouts_pre_3'}) conf: 0.8687864108117281
frozenset({'HomeTimeouts_Remaining_Post_3'}) --> frozenset({'posteam_timeouts_pre_3'}) lift: 1.307021931382035
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Post_3'}) conf: 0.8689338839010602
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Post_3'}) lift: 1.307021931382035
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) conf: 0.7658017422314678
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) lift: 1.135517191202928
frozenset({'HomeTimeouts_Remaining_Pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) conf: 0.7497781398664475
frozenset({'HomeTimeouts_Remaining_Pre_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) lift: 1.135517191202928
frozenset({'HomeTimeouts_Remaining_Pre_3'}) --> frozenset({'posteam_timeouts_pre_3'}) conf: 0.8646689555843287
frozenset({'HomeTimeouts_Remaining_Pre_3'}) --> frozenset({'posteam_timeouts_pre_3'}) lift: 1.300827538586841
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) conf: 0.8772883432413383
frozenset({'posteam_timeouts_pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) lift: 1.3008275385868409
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Post_3'}) conf: 0.7575957948698898
frozenset({'AwayTimeouts_Remaining_Pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Post_3'}) lift: 1.1395508189556323
frozenset({'HomeTimeouts_Remaining_Post_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) conf: 0.7524415305546435
frozenset({'HomeTimeouts_Remaining_Post_3'}) --> frozenset({'AwayTimeouts_Remaining_Pre_3'}) lift: 1.1395508189556325
frozenset({'HomeTimeouts_Remaining_Post_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) conf: 1.0
frozenset({'HomeTimeouts_Remaining_Post_3'}) --> frozenset({'HomeTimeouts_Remaining_Pre_3'}) lift: 1.482782198815776
frozenset({'HomeTimeouts_Remaining_Pre_3'}) --> frozenset({'HomeTimeouts_Remaining_Post_3'}) conf: 0.9857827661957898
```

根据这些挖掘到的规则的置信度结果和提升度（Lift）结果，可知置信度较高，且提升度均大于 1，则可认为这些规则有用。