Question 1

In [29]:
```
import pandas as pd
from sklearn import tree
from sklearn.metrics import cohen_kappa_score
```

In [30]:
```
PATH_TO_FINENAME = 'BDEPennW1.csv'
data = pd.read_csv(PATH_TO_FINENAME)
data.head()
```

Out[30]:

	UNIQUEID	SCHOOL	Class	GRADE	CODER	STUDENTID	Gender	OBSNUM	totalobs-forsession	Activity	ONTASK	TRANSITIONS	NumACTIVITIE
0	14400	B	T9Q	0	Z	600865	0	1	0	Wholecarpet	Y	3	4
1	14401	B	T9Q	0	Z	596466	0	1	1	Wholecarpet	Y	3	4
2	14402	B	T9Q	0	Z	616590	0	1	2	Wholecarpet	Y	3	4
3	14403	B	T9Q	0	Z	734358	1	1	3	Wholecarpet	Y	3	4
4	14404	B	T9Q	0	Z	826308	1	1	4	Wholecarpet	Y	3	4

In [31]:
```
data.dtypes
```

Out[31]:
```
UNIQUEID                 int64
SCHOOL                  object
Class                   object
GRADE                    int64
CODER                   object
STUDENTID                int64
Gender                   int64
OBSNUM                   int64
totalobs-forsession      int64
Activity                object
ONTASK                  object
TRANSITIONS              int64
NumACTIVITIES            int64
FORMATchanges            int64
NumFORMATS               int64
Obsv/act               float64
Transitions/Durations  float64
Total Time               int64
dtype: object
```

In [32]:
```
data_dum = pd.get_dummies(data, columns = ['SCHOOL','Class', 'CODER ', 'Activity'])
data_dum.head()
```

Out[32]:

	UNIQUEID	GRADE	STUDENTID	Gender	OBSNUM	totalobs-forsession	ONTASK	TRANSITIONS	NumACTIVITIES	FORMATchanges	...	Class_T9U	Cla
0	14400	0	600865	0	1	0	Y	3	4	1	...	0	0
1	14401	0	596466	0	1	1	Y	3	4	1	...	0	0
2	14402	0	616590	0	1	2	Y	3	4	1	...	0	0
3	14403	0	734358	1	1	3	Y	3	4	1	...	0	0
4	14404	0	826308	1	1	4	Y	3	4	1	...	0	0

5 rows × 49 columns

In [33]:
```
y = data['ONTASK']
```

In [34]:
```
X = data_dum.drop(columns = ['ONTASK'], axis = 1)
```

We don't need ONTASK when predicting the label.

In [35]:
```
clf = tree.DecisionTreeClassifier(min_samples_split = 10)
clf.fit(X, y)
```

Out[35]:
```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=10,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

No, kappa can not be computed now.

In [36]:
```
predictions = clf.predict(X)
```

In [37]:
```
kappa = cohen_kappa_score(y, predictions)
kappa
```

Out[37]:
```
0.7287148733570503
```