**Exercise 2:** Data acquisition, use of numpy and Semi supervised learning

In this exercise you will experiment use of Pandas to read data from raw CSV file and will learn how to clean up missing values. Then you will experiment on how to apply semi-supervised learning methods to label and unlabeled data. Then you will train KNeighborsClassifier using newly labeled data.

We have provided the breast cancer tumor classification data in raw form. This data also available at <https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)>

1. Read the provided csv file using Pandas read\_csv API. Set the header and index columns appropriately
2. Replace ‘?’ with zero
3. Normalize the data using standard scaler
4. Review the label column, notice that labels are 2 and 4.
5. Use one hot encoding to convert this to binary classification
6. Divide the data in to 50% training set and 50% test set
7. Note This data is already pre labeled. However, for our exercise we will ignore labels of test set. Make a copy of the test label set and override it with -1. -1 indicate unlabeled data.
8. Combine the train and test data and create new data set
9. Combine training labels and test labels with -1 and create new label set
10. Go to scikit-learn API page and search for LabelSpreading and study the format and the usage pattern
11. Select knn as the kernel.
12. What value do you propose for n\_neighbors ?
13. Train the LabelSpreading with kernel=’knn’ and number of neighbors from 10.
14. Create array that contain newly assigned labels
15. Split the data in to 80% training set and 20% test set with the original data and newly derived data
16. Train KNeigborsClassifer with using training set from 13.
17. What kind of accuracy do you see in 14 ?

**from** sklearn.preprocessing **import** StandardScaler  
**from** sklearn.model\_selection **import** train\_test\_split  
**from** sklearn.neighbors **import** KNeighborsClassifier  
**from** sklearn.metrics **import** classification\_report  
**import** pandas **as** pd  
**import** numpy **as** np  
**from** sklearn.preprocessing **import** OneHotEncoder  
**from** sklearn.semi\_supervised **import** LabelPropagation, LabelSpreading  
  
col = [**'c1'**, **'c2'**, **'c3'**,**'c4'**, **'c5'**, **'c6'**, **'c7'**, **'c8'**, **'c9'**, **'c10'**, **'target'**]  
data = pd.read\_csv(**'c:\\workspace\\data.csv'**, header=**None**, names=col, index\_col=col[0])  
  
  
data = data.replace(to\_replace=**'?'**, value=0)  
X = data[col[1:10]].values  
X = np.asarray(X, dtype=**'float64'**)  
y = data[‘target’]. values  
y = np.asarray(y, dtype=**'int32'**)  
oE = OneHotEncoder(sparse=**False**)  
oE.fit(y.reshape(1,-1))  
y=oE.fit\_transform(y.reshape(y.shape[0],1))[:,0]  
*#y=oE.transform(y.reshape(1,-1))*s=StandardScaler()  
*#We are only transforming data not the labels*X1=s.fit\_transform(X)  
  
X1\_tr, X1\_tst, y\_tr, y\_tst = train\_test\_split(X1,y, shuffle=**True**, random\_state=32, test\_size=0.5)  
y\_tst\_unlabel = np.full(y\_tst.shape, -1, dtype=**'int32'**)  
X1\_new = np.concatenate((X1\_tr, X1\_tst), axis=0)  
y\_new = np.concatenate((y\_tr,y\_tst\_unlabel), axis=0)  
  
*#Label propogation bug, make n\_neighbors greater than default=9  
#https://github.com/scikit-learn/scikit-learn/issues/9292  
# Please comment or uncomment following APi as applicable  
#sms = LabelPropagation(kernel='knn', n\_neighbors=11, max\_iter=100000)*sms = LabelSpreading(kernel=**'knn'**, n\_neighbors=11, max\_iter=10000)  
  
sms.fit(X1\_new,y\_new)  
y\_derived = sms.transduction\_  
X1\_tr, X1\_tst, y\_tr, y\_tst = train\_test\_split(X1\_new,y\_derived, shuffle=**False**, random\_state=32, test\_size=0.8)  
*#There are 4 features*knn = KNeighborsClassifier(n\_neighbors=11)  
knn.fit(X1\_tr, y\_tr)  
sc=knn.score(X1\_tst, y\_tst)  
print(**'Score '**, sc)  
y\_pred = knn.predict(X1\_tst)  
c = classification\_report(y\_tst, y\_pred)  
  
print(**' Confusion Matrix '**, c )