SEng 474 / CSc 578D
Data Mining – Spring 2016
Assignment 1

Due: February 3rd

Hand in a paper copy for questions 1 to 3 in class.
Submit code for question 4 through ConneX before 11:55pm.

Different marking schemes will be used for undergrad (SEng 474) and grad (CSc 578D) students.
Undergrad students do not have to answer the grad questions.

All code questions use the python sckit-learn library. You might have to install it, along with the NumPy and SciPy libraries on your own computer. Alternatively, you can work in the lab.
http://scikit-learn.org/stable/install.html
http://www.numpy.org/

1.
a) (SEng 474: 20 points; CSc 578D: 15 points)
Construct the root and the first level of a decision tree for the contact lenses data using the ID3 algorithm. Show the details of your construction and all your calculations; no points will be given for solutions only.

b) (CSc 578D: 5 points)
Using the tree.DecisionTreeClassifier module from python's sckit-learn, fit a tree using the contact-lenses data. Compare the entropy values obtain in part a) with the ones calculated by the sklearn.tree module. Explain in detail why the trees are not the same.

note: You can import the data directly from the 'contact-lenses.arff' file using the *Arff2Skl()* converter from *util2.py* provided with this assignment, using these lines of code:

```
from util2 import Arff2Skl

cvt = Arff2Skl('contact-lenses.arff')
label = cvt.meta.names()[-1]
X, y = cvt.transform(label)
```

http://scikit-learn.org/stable/modules/classes.html#module-sklearn.tree

2. (SEng 474: 20 points; CSc 578D: 20 points)
Construct two rules using the PRISM algorithm for the weather data. Show the details of your construction and all your calculations; no points will be given for solutions only.

3. (SEng 474: 20 points; CSc 578D: 20 points)
Calculate the probabilities needed for Naïve Bayes using the contact lens dataset.
Classify: "*prepresbyopic, hypermetrope, yes, reduced, ?*" using your calculated probabilities.

4. a) (SEng 474: 40 points; CSc 578D: 30 points)
Implement the PRISM algorithm, as described in class, and verify the results you obtained for question 2, running your code with the weather.arff file.

Open the zip file attached to the assignment on connex. In it you will find incomplete code in *prism.incomplete.py*. Rename your file *prism.py* and fill the required method so that your code opens any .arff file given as an argument, and constructs the set of rules resulting from the PRISM algorithm.

You are given the *util2.py* file to deal with the conversion from. arff to a compatible format of the dataset. Make sure to have the *util2.py* file in your working directory.

Scripts will be used to verify your code, so make sure that you read the specifications on the expected input and output carefully.

Usage:
```
python prism.py *.arff
```

Ouput:
the result of:
```
print rules
```

For example: If we were to run your file on the contact-lensesTest data:

```
python prism.py contact-lensesTest.arff
```

and we would expect the output to be:

```
[{'soft': [('astigmatism', 'no'), ('tear-prod-rate', 'normal')]}, {'hard':
[('astigmatism', 'yes'), ('spectacle-prescrip', 'myope'), ('tear-prod-rate',
'normal')]}, {'hard': [('age', 'young'), ('spectacle-prescrip',
'hypermetrope'), ('astigmatism', 'yes')]}, {'none': [('tear-prod-rate',
'reduced')]}, {'none': [('spectacle-prescrip', 'hypermetrope'), ('age',
'presbyopic')]}, {'none': [('age', 'pre-presbyopic'), ('spectacle-prescrip',
'hypermetrope')]}]
```

You are provided with a rules printing function for debugging.
****** Make sure to comment it out before you submit your code.**

For this particular example, the debugging printing function would print:

```
IF astigmatism = no
        AND tear-prod-rate = normal
```

```
       THEN contact-lenses = soft

IF astigmatism = yes
       AND spectacle-prescrip = myope
       AND tear-prod-rate = normal
       THEN contact-lenses = hard

IF age = young
       AND spectacle-prescrip = hypermetrope
       AND astigmatism = yes
       THEN contact-lenses = hard

IF tear-prod-rate = reduced
       THEN contact-lenses = none

IF spectacle-prescrip = hypermetrope
       AND age = presbyopic
       THEN contact-lenses = none

IF age = pre-presbyopic
       AND spectacle-prescrip = hypermetrope
       THEN contact-lenses = none
```

b) (CSc 578D: 10 points)
Implement the `predict()` method in *prism.incomplete.py*.

Test your method using the provided *test_data.pck*, after training your algorithm with the weather data.

Output:
The output has to be a list of classification of each instance of the test_data.
For example, in case of the contact-lenses dataset *p* should looks like:

```
p = ['none', 'none', 'soft', ... , 'hard', ...]
```

The expected final output is the result of:
`print p`