Introduction to Machine Learning (89511)
Exercise 1

Due: March 28th, 2018 (no extensions)

1 Submission Instructions:

1. **Theoretical Part**

   (a) The main objective of this part is to practice the terms, concepts and different bounds for algorithms in the online prediction framework.

   (b) If needed, make sure to explain or present an example to any given answer to assure full understanding of your idea.

   (c) Write your full name and ID at the top of your solution.

   (d) Submission: please submit a PDF version named `Ex_1_theoretical.pdf` via the Submit system.

2. **Practical Part**

   (a) The main objective of this part is to assure you know some basic and necessary methods in python to give a very basic and preliminary taste of a machine learning model.

   (b) You are not allowed to use any machine learning packages or tools (e.g. scikit-learn, PyBrain, PyML, etc.).

   (c) You are allowed to use numpy package

   (d) Use Python 2.7

   (e) In order to submit your solution - please submit the relevant files to the corresponding assignment on the Submit system.

   Your files should include:

   i. A text file called `details.txt` with your full name (in the first line) and ID (in the second line).

   ii. `boolean_conj_predictor.py` - Your code file.

Good Luck!

2 Questions:

2.1 Theoretical Part

In this part you will compare between two strategies for the task of *online boolean conjunctions prediction*. The goal of this task is to predict the boolean conjunction that correctly represents the training set, where a *conjunction* is any product ($\wedge$) of literals (e.g., $\bar{x}_1 \wedge x_2$). The *hypothesis class* is defined as a set of all possible conjunctions over d variables (i.e., d is the number of variables in the boolean conjunction statement).

A conjunction containing both versions of a literal (atomic and its negation) is interpreted as the *all-negative hypothesis*. Note that the *hypothesis class* includes the *all-negative hypothesis*.

Example: Given a training example: (0, 1, 1) and its tag: 1, its predicted boolean conjunction could be: $\bar{x}_1 \wedge x_2 \wedge x_3$ (because obviously: $\bar{0} \wedge 1 \wedge 1 = 1$). The hypothesis class would be all of the possible conjunctions of x_1, x_2 and x_3.

1. **General definition of the problem:**

   Recall that our goal is to predict the correct boolean conjunction.

   (a) Describe the instance domain (X) and the label set (Y).

   (b) Present the entire hypothesis class for *d=2*.

   (c) Suggest an equation that describes the size of the hypothesis class for any given *d*.

   (d) Suppose that the true conjunction is $\bar{x_1} \wedge x_2 \wedge x_3$:

      i. Can the following example exist in the training set: ((1,0,1,1),0)? If not, suggest a correction.

      ii. Can the following examples co-exist in the training set: ((0,1,1,0),1), ((0,1,1,1),1)? If not, suggest a correction.

2. **The Consistency Algorithm.** We will now present a prediction strategy to find the hypothesis that correctly represents the dataset. In general, in this approach we create an initial hypothesis, which will be updated in every iteration, so it would fit the environment. Given that the initial hypothesis is the *all-negative hypothesis*, this prediction strategy is

described as follows:

```
1  h^0 = all_negative_hypothesis
2  for instance t in examples do
3      h^t = h^{t-1}
4      if y^t = 1 ∧ ŷ^t = 0 (our hypothesis is no good any more) then
5          for index i in instance t do
6              if x_i^t = 1 then
7                  remove x̄_i from h^t
8              end
9              if x_i^t = 0 then
10                 remove x_i from h^t
11             end
12         end
13     end
14     if y^t = 0 then
15         leave h^t as is
16     end
17 end
```

Example:
Suppose your training set consists of the following two instances ($d = 2$):
Instance 1: ((1,0),1) Instance 2: ((0,0),0)
The initial hypothesis would be: $h^0 = x_1 \wedge \bar{x}_1 \wedge x_2 \wedge \bar{x}_2$.
We will now iterate over the instances. Take the first training instance in our example; its given tag (y^t) is 1. However, if we would use our hypothesis to predict its tag ($\hat{y}^t$; $\hat{y}$ indicates the tag that the hypothesis predicts), we would get: $1 \wedge \bar{1} \wedge 0 \wedge \bar{0} = 0$. Thus, $y^t = 1 \wedge \hat{y}^t = 0$, so we must enter the *if* statement in line 3.
We will now iterate over the variables in our example.

- The first variable, i.e. $x_{i=1}^{t=1}$, is equal to 1. Therefore, according to line 6, we must remove $\bar{x}_1$ from our hypothesis.
- The second variable, i.e. $x_{i=2}^{t=1}$, is equal to 0. Therefore, according to line 9, we must remove x_2 from our hypothesis.

Thus, we are left with our new hypothesis for this iteration: $h^1 = x_1 \wedge \bar{x}_2$

In the next iteration, using the second training instance, you will find that $y^t = 0$, and therefore the hypothesis remains as is.

Questions:

(a) Does this algorithm implement the ERM principle?

(b) Denote by $M(a)$ the number of mistakes made by algorithm a. Prove that the number of mistakes made by the algorithm is: $M(a) \leq d + 1$. (Hints: (1) use induction to

prove $M(a) \leq 2d$, (2) Infer the desired bound by observing the outcome of the first iteration in the algorithm).

(c) What is the run-time per iteration?

2.2 Practical Part

In this part you will implement the *Consistency Algorithm* for the task of online boolean conjunction prediction. Given training examples and their classification, stored in a text file, and using the implemented algorithm you are expected to predict the boolean conjunction correctly.

1. Your main file should be called `boolean_conj_predictor.py`
2. The input text file path will be given as an input parameter (e.g. `python boolean_conj_predictor.py trainingData\example1.txt`)
3. The following format will be used in the training file:

   Each line represents a single example and classification, and is constructed by values 0,1 separated by white spaces.

   For example, say d = 3:

0 1 1 1		$((0,1,1),1) \in X \times Y$
1 0 0 1	is equivalent to	$((1,0,0),1) \in X \times Y$
1 1 1 1		$((1,1,1),1) \in X \times Y$

4. Load the file and separate the training data into two containers:

   (a) An X matrix such that its columns correspond to the examples of the training data.

   (b) A Y vector such that its values correspond to the classification (the last column in the raw data).

   Loading the data from the files can be done using:
   ```
   import numpy as np
   training_examples = np.loadtxt(file_path)
   ```
5. Implement the *Consistency Algorithm.*
6. Your algorithm's final predicted boolean conjunction should be written to a file called `output.txt`. The following format should be used:

   In a single line separated by commas (',') you should write a list of the *existing* literals in the boolean conjunction - ordered by their index. Literals in the negation form should be written using the 'not(x_i)' notation.

   For example, say d=5 and the true boolean conjunction is: $x_1 \wedge \bar{x}_2 \wedge \bar{x}_4 \wedge x_5$ (notice that x_3 is missing). Your algorithm should write to the file at the end of its run: **x1,not(x2),not(x4),x5**

7. Don't forget to submit the additional `details.txt` file, as mentioned in the submission instructions.

Tips:

- To ease the data creation process - create a separate script in the following format:
  1. Randomly generate a boolean conjunction.
  2. For every training example in the training set, initialize the conjunction accordingly so the outcome would be the classification of the example.
  3. Write the examples and their classification to the training file.

  This is not mandatory or part of the exercise submission.

- Remember to create enough examples to assure that the desired conjunction is fully described, otherwise the answer may be ambiguous.