Assignment 1

CSE 343/543 : Machine Learning **Due: 11:59PM, Aug. 28,2017**

Note: Please complete programming as well as theory components

Problem Statement

You are given three small datasets:

- (part_A) A dataset with each datum being a 784 dimensional vector, and each datum has one of 10 labels (in 0-9)
- (part_B) A dataset, with each datum being a 2048 dimensional vector, and each datum having one of 2 labels (0 or 1)
- (part_C) A dataset, with each datum being 2 dimensional, and each datum having one of 2 labels (0 or 1)

Programming Assignment [130 marks]

1. Visualise the three datasets. For this, you'll need load the h5 files and use *t-SNE* to plot the data. Use *sklearn's* implementation of *t-SNE*. You are free to use any parameters so that the plots make sense.

2. For the second part you need to use GaussianNB, LogisticRegression & DecisionTreeClassifier in *sklearn* and train them on the training h5 files.
    a. For each model use grid search (to be written by you, not using *sklearn*'s implementation) on the parameters to find the optimal parameters.
    b. Use k-fold cross validation (to be written by you, not using *sklearn*'s implementation) to evaluate the accuracy of each parameter in the grid (Value of k is to be determined by you)
    c. Plot the validation accuracy vs the parameters in the grid, and save your plots (one for each dataset) in the Plots folder, each in a different plot. Note that plots are to be made for all models across all datasets, so 9 plot files.
    d. Save the best model (for each dataset) in the Weights folder. You can serialize the model in any way you want (preferred: sklearn's joblib function to save models as pickled files). Load the saved model using *predict.py* to predict the results on unlabeled data (again, for each dataset).

3. For this part, you need to implement all three models on your own and then repeat the above question. You have to fill the Python classes given in the *Models* folder. Make sure your classes are compatible with *sklearn*'s classes. You should only change the import statements by importing your own model classes instead of *sklearn's* and the code should work. **Using *sklearn* for this part is strictly not allowed (not even importing it); only *scipy* and *numpy* can be used for this part.**

Template Details

A code template has been provided to you. You are expected to write implementations for all of the functions written in these files. The structure is:

- train.py
- predict.py
- tests.py
- visualize.py
- Weights/
- Plots/
- Data/
    - partA_train.h5
    - partB_train.h5
    - partC_train.h5
- Models/
    - GaussianNB.py
    - LogisticRegression.py
    - DecisionTreeClassifier.py

1. The basis for picking a specific model for the dataset has to be given, *i.e.*, the report must include graphs, along with appropriate explanations. These graphs are validation accuracy v/s hyper-parameters (used in grid-search).
2. Any submission with changes to the directory structure in the given template will not be evaluated.
3. No other imports other than the ones already defined in the files may be used (apart from *os*). Any code that fails to run because of any other added import/missing package will not be evaluated.
4. As announced on Backpack, python 2.7 is to be used.
5. Comments are expected to be added in code for all non-trivial part of code written by you.

6. Example command for training gaussian model for partA:

```
python train.py --model_name GaussianNB --weights_path Weights/save_model_A
--train_data Data/partA_train.h5 --plots_save_dir Plots/
```

Theory Questions [30 marks + 10 marks bonus]

1. The minima of a given function may be found using its first order derivative and equating it to zero (and second order derivative > 0, etc). Consider the case of a simple linear regression model. Why don't we then, in all cases, simply find the minima of this function using a similar approach, instead of using gradient descent which is obviously slower.
2. How is machine learning different from function approximation? Would the two be the same if we had all the possible the data that the model is expected to ever see?
3. You are given a function that maps a set of 2D points to another set of 2D points. The function is given by :

   $y \ = \ (1/\lambda)R^3x \ + \ B$ where $x \subseteq R^2 \; and \; y \subseteq R^2$

   $B \ = \ [\,a \quad b\,]^T$

   $$R = \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix}$$

   You are given a set of n data points in X and Y where $X \subseteq R^{nx2}$ and $Y \subseteq R^{nx2}$. Find θ, λ , a and b in terms of X and Y such that the squared *L2* distance between the ground truth data points Y and the predicted data points Y' is minimized. Find a closed form solution. Can you also use gradient descent to solve the above problem?
4. Let x_1, x_2, ..., x_n be i.i.d. data from a uniform distribution over the disc of radius θ in R^2, i.e., $x_i \in R^2$ and

   $$p(x;\theta) = \begin{cases} \frac{1}{\pi\theta^2}, & \| x \| \leq \theta \\ 0, & otherwise \end{cases}$$

   where $\| x \| = \sqrt{x_1{}^2 + x_2{}^2}$. What is the maximum likelihood estimate of θ.
5. [Bonus] In Q3, assume you have a lot of outliers (about 50% of them) . Write an algorithm (pseudo code) that could learn the parameters with the noisy data.