EECS 461 / ECE 523 - MACHINE LEARNING
Assignment 3

Preparing Data

Scikit-Learn provides many helper functions to download popular datasets. MNIST is one of them. In this assignment, you will use the MNIST dataset. First 60000 instances will form your training set, next 10000 instances will be used to create the validation and test sets. Training, validation and test sets are provided in skeleton_function.py file.

Voting Classifiers (40 points)

a. (10 points) Create a Random Forest classifier with parameters random_state=0. Train the classifier using the training set. Save your classifier in pickle format as *RFClassifier.pkl*.

b. (10 points) Create an Extra-Trees with parameters random_state=0. Train the classifier using the training set. Save your classifier in pickle format as *ETClassifier.pkl*.

c. (10 points) Combine Random Forest and Extra-Trees classifiers into an ensemble classifier using a soft Voting classifier. Save your trained classifier in pickle format as *SoftEnsembleClassifier.pkl*. You may use the Scikit-Learn's VotingClassifier.
Note: In the ensemble classifier, Random Forest and Extra-Trees classifiers are new classifiers, not the classifiers from part a and b.

d. (10 points) How much better does the ensemble classifier perform compared to the individual classifiers? Use **test set** to measure **accuracy score** of each classifier and return them in a single list. Save your result in pickle format as *part_d.pkl*.
The order of the classifiers is Random Forest, Extra-Trees and the ensemble classifier. Return your results according to that order.
For example, if Random Forest, Extra-Trees and the ensemble classifier has the accuracy score of 0.6, 0.65 and 0.8 respectively, the result will be [0.6, 0.65, 0.8].

Stacking (50 points)

e. (20 points) Run the individual classifiers (Random Forest and Extra-Trees mentioned in part a and b) to make **probabilistic predictions** on the **validation set** and create a new training set with the resulting predictions: each training instance is a vector containing the set of probabilistic predictions from all your classifiers for an image, and the target is the image's class. Save the new training set into a pickle file as *part_e.pkl*.

For instance, if you are going to predict the image of number 9, firstly, you will predict the probability vector of the image with your individual classifiers' *predict_proba()* function.
Lets say RandomForest classifier's probabilistic prediction output is:

[0. 0. 0.1 0. 0.2 0. 0. 0. 0.1 0.6],

and Extra-Trees classifier's probabilistic prediction output is:

[0. 0. 0.1 0.1 0.2 0. 0.2 0. 0. 0.4].

The new representation of the image will be:

[0. 0. 0.1 0. 0.2 0. 0. 0. 0.1 0.6 0. 0. 0.1 0.1 0.2 0. 0.2 0. 0. 0.4].

The new training set will consist of new representation of validation set instances (keep the same order as in the validation set).

f. (10 points) Train a new Random Forest Classifier (set random_state to 0) with the new training set you created in part e. Save your classifier in pickle format as *Blender.pkl*.

Congratulations, you have just trained a blender, and together with the classifiers they form a Stacking ensemble!

g. (20 points) Evaluate the ensemble on the test set: for each image in the test set, make predictions with all your classifiers, then feed the predictions to the blender to get the ensemble's predictions. Report the **test accuracy** of the stacking ensemble in pickle format as *part_g.pkl*.

Saving Results

In order to save your answers in pickle format, you will use the scikit-learn's joblib as follows:

```
clf = anyModel()
from sklearn.externals import joblib
joblib.dump(clf, 'filename.pkl')
```

Assignment Submission (10 points)

At the end, you have 9 pickle files. Put only these files into a folder and name the folder as **<Your first name>_<Your last name>_Assignment3**.

- Don't compress the file, just submit as it is.
- Don't use Turkish characters or space in the folder name.

(5 points) Proper folder name.

(5 points) Proper format of files & file names.