Data Mining Project #2

Objective

The purpose of this project is to implement different classifiers to predict whether a person's income is more than $50k or not. The project consists of multiple components that involve data preprocessing, training classification models, assessing the performance of the classifier, and analyzing some of the models that were estimated. It is recommended to use Python to implement the algorithms. If you wish to use another language, you must ask for permission first. This project was designed for teams of 2 or 3 students.

This project can also be considered a small competition among the teams. Teams will need to build a number of different models, and for their best model, submit their predictions on the test set. The teams that will be at the top of the ranking will receive extra points in the end! You can see the leaderboard in this link: leaderboard.

Dataset

The dataset is derived from the Census income data. You will be provided with two files with input data (*adult.input*) and test data (*adult.test*). For the test data, you won't be given the true class labels. You need to employ an evaluation methodology (split the data into training and validation tests or use a k-fold validation) to train the classifiers using some part of the data and evaluate your models to perform data selection using the rest of the data. You will build the models, and you will have two opportunities to submit your predictions before the final submission to get the performance of your models in the test set. The provided files do not have header files, but the attribute values are comma-separated. The columns/attributes included in the data files are as follows:

1. age : The age of the individual.
2. type_employer : The type of the employer that the individual has.
3. fnlwgt : The number of people the census takers believe that observation represents.
4. education : The highest level of education achieved for that individual.
5. education_num : Highest level of education in numerical form.
6. marital : Marital status of the individual.
7. occupation : The occupation of the individual.
8. relationship : The family relationship of the individual.
9. race : The descriptions of the individual's race.
10. sex : Biological Sex.
11. capital_gain : Capital gains recorded.
12. capital_loss : Capital Losses recorded.
13. hr_per_week : Hours worked per week.
14. country : Country of origin for person.
15. **income** : Whether or not the person makes more than $50,000 per year income. **Class label.**

e.g.
51, Private, 259323, Bachelors, 13, Married-civ-spouse, Exec-managerial, Husband, White, Male, 0, 0, 50, United-States, >50K

Classification approach

You need to test three (for graduate students) or two (for undergraduate students) different approaches: 1) decision trees, 2) perceptron, 3) multilayer neural network. You can use pre-built functions on the scikit-learn package. The following table shows the hyper-parameters that you can play with and select the model with the best performance (i.e., **lowest misclassification error**). Each approach will be a different file with the following names respectively:

```
dt_<teamID>.py
perc_<teamID>.py
nn_<teamID>.py
```

Hyperparameters that you need to tune for the different methods and some values to try.

Classification method	Hyper parameters	Values to try
Decision Tree sklearn.tree.DecisionTreeClassifier	*criterion*	gini, entropy
	min_leaf_size (if there are <= min_leaf_size instances in a node, don't split, make it a leaf node)	>4
Percepton sklearn.linear_model.Perceptron	*max_iter* (maximum number of epochs)	>1
	eta0 (learning rate)	[0.1, 1, 1.2]
Multilayer neural network sklearn.neural_network.MLPClassifier	*hidden_layer_sizes* (tuple, the ith element represents the number of neurons in the ith hidden layer).	(2,) -> for 1 hidden layer*, (2,2) -> for 2 hidden layers*
	learning_rate_init (learning rate)	[0.1, 0.01, 0.001]

* Use at least two neurons/nodes per layer. You can also try values greater than 2.

Each code file needs to have the following parts:

1) **Pre-processing**

You need to read the two files provided and perform the following actions:

1. remove instances that have any missing values
2. remove attributes: fnlwgt, education, relationship
3. transform the dataset into a numerical one with the necessary binarization and normalization. All attributes should be binary (0/1) or in the range [0,1]. You can transform some categorical attributes as follows:
    - 3.1. binarize the following attributes:
        - capital-gain (yes: >0, no: =0)
        - capital-loss (yes: >0, no: =0)
        - native country (United-States, other)
    - 3.2. discretize continuous attributes into asymmetric binary attributes:
        - age: divide values of age to 4 levels: young (<=25), adult ([26,45]), senior ([46,65]), and old ([66,90]).
        - hours-per-week: divide the values into 3 levels: part-time (<40), full-time (=40), over-time (>40)

3.3. merge attribute values together and create asymmetric binary attributes:

- workclass: create the following 3 attributes: gov (Federal-gov, Local-gov, State-gov), Not-working (Without-pay, Never-worked), Private, Self-employed (Self-emp-inc, Self-emp-not-inc)
- marital-status: create the following 3 attributes: Married (Married-AF-spouse, Married-civ-spouse), Never-married, Not-married (Married-spouse-absent, Separated, Divorced, Widowed)
- occupation: create the following 5 attributes: Exec-managerial, Prof-specialty, Other (Tech-support, Adm-clerical, Priv-house-serv, Protective-serv, Armed-Forces, Other-service), ManualWork (Craft-repair, Farming-fishing, Handlers-cleaners, Machine-op-inspct, Transport-moving), Sales.

In the end of this part, you need to output some statistics about the data. More specifically you need to show:

```
Input data: (xx, yy)
Test data: (zz, ww)
```

where xx (zz) and yy (ww) is the size and the dimensionality of the data in the input data (test data), respectively.

2) **Training & Model Selection**

In this part, you need to set up your evaluation process. You need to split the input data into a training set and a validation set. **Remember to set the seed of the random number generator into your team's ID.** [e.g. `random.seed(0)`] Just to be sure, set the corresponding arguments on the models you build as well, [e.g., `random_state=teamID`]. You will build a model for each combination of hyperparameter values that you want to try based on the training set, and you will make predictions (and compute the misclassification error) for the training set, the validation set and the test set. For each classification approach, you will choose the model that exhibits the least misclassification error on the validation set.

Note that for each hyperparameter, you need to examine/test at least **3** values.

3) **Model Evaluation**

In the end, you need to evaluate the best solution that you have acquired. You need to print out some evaluation metrics. More specifically, print the accuracy in the validation set in one line and the output produced when calling the following function: `sklearn.metrics.classification_report`. Also, save the predictions for the test set in a file `pred_<dt/perc/nn>_<teamID>.csv`.

The file will have one prediction per line (either **>50K** or **<=50K**), and the order of the predictions needs to match the order of the data instances in the test file provided. In other words, the predicted label in the 10th line needs to correspond to the 10th instance in the test file. This is the file that you need to send to us in order to place your team on the leaderboard before the submission deadline. Each team can submit the prediction file for each classification method 4 times before the submission deadline. We will check the accuracy of your predictions and update your position on the leaderboard!

Deliverables:

A. The three (or two for undergrads) code files, named as indicated above. Note that you should leave in comments any values that you will try for the hyperparameters.
B. Three (or two for undergrads) prediction files, named as indicated above.
C. PDF Report.

Report:

Q0) Mention the members of your team and how they contributed to the project. Also mention if you collaborated with another team, and if yes, mention which one.

Q1) For each attribute, discuss the following:

1. Type of attribute, and possible attribute values (or min-max values that it takes).
2. How did your team handle it in data preprocessing?

Q2) How did you perform model selection? Are there any additional design choices that you made?

Q3) For each classification approach, discuss the following:

1. Which values did you try for the different hyperparameters?
2. Which combination of hyperparameters had the best performance?
3. Which class is harder to predict?

Q4) Create the following three plots (or two, for the undergrad teams). Plot the training and validation misclassification error occurred for different combination of hyperparameters. Discuss what you can see in the plots.

1. For the decision trees: for the criterion that gave you the best performance, show the performance for at least three values of min_leaf_size that you tried.
2. For the perceptron: for the criterion that gave you the best performance, show the performance for at least three values of min_leaf_size that you tried.
3. For the neural networks: for learning_rate_init=0.001 and when you used 1 hidden layer, show the performance for at least three values of number of nodes/neurons that you tried.