Final Activity

Today, we're going to try to predict whether a person's income is greater than of less than 50K a year, based on characteristics about that person. To make things more exciting, you can work in teams of two and the team that has the best-performing model will get a prize next week.

About the Dataset

This dataset was taken from UC Irvine's Machine Learning Repository and subsequently simplified. It contains data about ~30,000 adults in the US in 1994, and whether their income is above or below $50K/year. The data it contains is: **age (integer), workclass (string), education (string), marital status (string), occupation (string), race (string), sex (string), capital income (integer), hours per week worked (integer), native country (string)**:

- **workclass** contains the following values: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked
- **education** contains the following values: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool
- **marital status** contains the following values: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse
- **occupation** contains the following values: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces
- **race** contains the following values: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
- **sex** contains the following values: Female, Male
- **capital income** refers to money you make/lose from your possessions (i.e. real estate or stocks you own). It can be positive or negative.
- **native country** contains the following values: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands

Interacting with the Dataset

We have written Python code to allow you to easily interact with the database. Initialize it with the line "database = Database("Training")". You can then interact with the "database" variable as if it were a list (you can index into it, iterate over it in a loop, and get its `len`). Each entry in the database is a person object, and has the following properties:

- age
- workclass
- maritalStatus
- occupation
- race
- sex
- capitalIncome
- hoursPerWeek
- nativeCountry

For example, to access the 11th person's age, do: "database[10].age"

Guidelines for Writing the Function

You can write your function in any way you want. You can use decision stumps/trees, weighted majority algorithm, and/or k nearest neighbors. You can also use your intuition (i.e. if someone works less than a certain number of hours per week, automatically predict that they will make less than 50K).

Evaluation Technique

We will run all your functions in an overall weighted majority algorithm. This will have two outcomes: (1) as a class we will create a stronger learner comprised of all of your learners; and (2) we will be able to determine how accurate each learner is. The team with the best ***test set*** performance gets a prize.

Hints and notes

- Start by always predicting False. What's your model's accuracy on the training set? What about the dev set?
- The training data has 32561 data points. A decision tree or KNN algorithm will probably take a long time to run using all that data. What can you do to speed this process up?
- Our KNN and decision tree algorithms both only handle 2-dimensional data (i.e. two features). You'll have to pick the two features to use in the `loadDataFromDatabase` function, or you can modify the code to expand to more than 2 dimensions (not easy!).
    - We also only wrote code to handle features in the form of numbers. If you want to use one of the string features like "occupation," how would you modify the code to do so?
- If you want to use a KNN or decision tree, use the following functions:
    - loadDataForModel
    - decisionTreeSolutions.createDecisionTree
    - decisionTreeSolutions.getPrediction
    - knnSolutions.getPrediction
- We give you both a training and development set, yet we will ultimately be checking your model's accuracy on a test set. Make sure that you don't overfit!
- Think of the possible things you can "tune" in your model. (e.g. *k* for KNNs, maximum depth for DTs)