CS 189 Introduction to Machine Learning
Fall 2017 HW12

This homework is due **Tuesday, November 21 at 10pm.**

1 Decision Trees Appendix

You do not need to submit anything for this question! It is just to help you with the previous question.

1. Categorical variables. Most of the data you've dealt with so far has been continuous-valued. Many features in this dataset represent types/categories. There are many possibilities to deal with categorical variables:

   (a) (Easy) In the feature extraction phase, map categories to binary variables. For example suppose feature 2 takes on three possible values: 'TA', 'lecturer', and 'professor'. In the data matrix, these categories would be mapped to three binary variables. These would be columns 2, 3, and 4 of the data matrix. Column 2 would be a boolean feature $\{0,1\}$ representing the TA category, and so on. In other words, 'TA' is represented by $[1,0,0]$, 'lecturer' is represented by $[0,1,0]$, and 'professor' is represented by $[0,0,1]$. Note that this expands the number of columns in your data matrix. This is called "vectorizing," or "one-hot encoding" the categorical feature.

   (b) (Hard, but more generalizable) Keep the categories as strings or map the categories to indices (e.g. 'TA', 'lecturer', 'professor' get mapped to $0,1,2$). Then implement functionality in decision trees to determine split rules based on the subsets of categorical variables that maximizes information gain. You cannot treat these as normal continuous-valued features because ordering has no meaning for these categories (the fact that $0 < 1 < 2$ has no significance when $0,1,2$ are discrete categories).

   Here is a list of the field names for the categorical variables:

   ```
   [workclass, education, marital-status,
     occupation, relationship, race, sex, native-country]
   ```

2. Missing values. Some data points are missing features. In the `csv` files, this is represented by the value '`?`'. You have three approaches:

   (a) (Easiest) If a data point is missing some features, remove it from the data matrix (**this is good as a start but you must submit something more sophisticated**).

   (b) (Easy) Infer the value of the feature from all the other values of that feature (e.g. fill it in with the mean, median, or mode of the feature).

(c) (Hard, but more powerful). Use kNN to impute feature values based on the nearest neighbors of a data point. You will need to define in your distance metric what the distance to a missing value is.

(d) (Hardest, but more powerful) Implement within your decision tree functionality to handle missing feature values based on the current node. There are many ways this can be done. You might infer missing values based on the mean/median/mode of the feature values of data points sorted to the current node. Another possibility is assigning probabilities to each possible value of the missing feature, then sorting fractional (weighted) data points to each child (you would need to associate each data point with a weight in the tree).

It is recommended you use the following classes to write, read, and process data:

```
csv.DictReader
sklearn.feature_extraction.DictVectorizer(vectorizing
                                          categorical variables)
    (There's also sklearn.preprocessingOneHotEncoder,
    but it's much less clean)
sklearn.preprocessing.LabelEncoder
    (if you choose to discretize but not vectorize categorical
      variables)
sklearn.preprocessing.Imputer
    (for inferring missing feature values in the preprocessing phase)
```

If you use `csv.DictReader`, it will automatically parse out the header line in the `csv` file (first line of the file) and assign values to fields in a dictionary. This can then be consumed by `DictVectorizer` to binarize categorical variables.

To speed up your work, you might want to store your cleaned features in a file, so that you don't need to preprocess every time you run your code.