In [1]:
```
#!/usr/bin/python


"""
    Starter code for the validation mini-project.
    The first step toward building your POI identifier!

    Start by loading/formatting the data

    After that, it's not our code anymore--it's yours!
"""

import pickle
import sys
sys.path.append("../tools/")
from feature_format import featureFormat, targetFeatureSplit

data_dict = pickle.load(open("../final_project/final_project_dataset.pkl", "r") )

### first element is our labels, any added elements are predictor
### features. Keep this the same for the mini-project, but you'll
### have a different feature list when you do the final project.
features_list = ["poi", "salary"]

data = featureFormat(data_dict, features_list)
labels, features = targetFeatureSplit(data)



### it's all yours from here forward!
```

You'll start by building the simplest imaginable (unvalidated) POI identifier. The starter code (validation/validate_poi.py) for this lesson is pretty bare--all it does is read in the data, and format it into lists of labels and features. Create a decision tree classifier (just use the default parameters), train it on all the data (you will fix this in the next part!), and print out the accuracy. THIS IS AN OVERFIT TREE, DO NOT TRUST THIS NUMBER!

Nonetheless, what's the accuracy?

From Python 3.3 forward, a change to the order in which dictionary keys are processed was made such that the orders are randomized each time the code is run. This will cause some compatibility problems with the graders and project code, which were run under Python 2.7. To correct for this, add the following argument to the featureFormat call on line 25 of validate_poi.py:

sort_keys = '../tools/python2_lesson13_keys.pkl'

This will open up a file in the tools folder with the Python 2 key order.

Note: If you are not getting the results expected by the grader, then you may want to check the file tools/feature_format.py. Due to changes in the final project, some file changes have affected the numbers output on this assignment as written. Check that you have the most recent version of the file from the repository, such that the featureFormat has a default parameter for sort_keys = False and that keys = dictionary.keys() results.

In [3]:
```
from sklearn.tree import DecisionTreeClassifier as dtc

clf = dtc().fit(features,labels)
print "Accuracy:",clf.score(features,labels)
```

```
Score: 0.9894736842105263
```

Now you'll add in training and testing, so that you get a trustworthy accuracy number. Use the train_test_split validation available in sklearn.cross_validation; hold out 30% of the data for testing and set the random_state parameter to 42 (random_state controls which points go into the training set and which are used for testing; setting it to 42 means we know exactly which events are in which set, and can check the results you get).

What's your updated accuracy?

In [4]:
```
from sklearn import cross_validation

features_train, features_test, labels_train, labels_test = cross_validation.train
clf = dtc().fit(features_train, labels_train)
print "Accuracy: ",clf.score(features_test,labels_test)
```

```
Accuracy:  0.7241379310344828

C:\Users\Andrew\Anaconda3\envs\conda2\lib\site-packages\sklearn\cross_validatio
n.py:41: DeprecationWarning: This module was deprecated in version 0.18 in favo
r of the model_selection module into which all the refactored classes and funct
ions are moved. Also note that the interface of the new CV iterators are differ
ent from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

In [ ]:
```

```