Project Overview

In this project, you will play detective, and put your machine learning skills to use by building an algorithm to identify Enron Employees who may have committed fraud based on the public Enron financial and email dataset.

Why this Project?

This project will teach you the end-to-end process of investigating data through a machine learning lens.

It will teach you how to extract/identify useful features that best represents your data, a few of the most commonly used machine learning algorithms today, and how to evaluate the performance of your machine learning algorithms.

What will I learn?

By the end of the project, you will be able to:

- Deal with an imperfect, real-world dataset
- Validate a machine learning result using test data
- Evaluate a machine learning result using quantitative metrics
- Create, select and transform features compare the performance of machine learning algorithms
- Tune machine learning algorithms for maximum performance
- Communicate your machine learning algorithm results clearly

Why is this Important to my Career?

Machine Learning is a first-class ticket to the most exciting careers in data analysis today.

As data sources proliferate along with the computing power to process them, going straight to the data is one of the most straightforward ways to quickly gain insights and make predictions.

Machine learning brings together computer science and statistics to harness that predictive power.

Project Introduction

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, there was a significant amount of typically confidential information entered into public record, including tens of thousands of emails and detailed financial data for top executives. In this project, you will play detective, and put your new skills to use by building a person of interest identifier based on financial and email data made public as a result of the Enron scandal. To assist you in your detective work, we've combined this data with a hand-generated list of persons of interest in the fraud case, which means individuals who were indicted, reached a settlement, or plea deal with the government, or testified in exchange for prosecution immunity.

Pep Talk

A note before you begin: the projects in the Intro to Machine Learning class were mostly designed to have lots of data points, give intuitive results, and otherwise behave nicely. This project is significantly tougher in that we're now using the real data, which can be messy and doesn't have as many data points as we usually hope for when doing machine learning. Don't get discouraged--imperfect data is something you need to be used to as a data analyst! If you encounter something you haven't seen before, take a step back and think about a smart way around. You can do it!

Resources Needed

You should have python and sklearn running on your computer, as well as the starter code (both python scripts and the Enron dataset) that you downloaded as part of the first mini-project in the Intro to Machine Learning course. The starter code can be found in the final_project directory of the codebase that you downloaded for use with the mini-projects. Some relevant files:

poi_id.py : starter code for the POI identifier, you will write your analysis here

final_project_dataset.pkl : the dataset for the project, more details below

tester.py : when you turn in your analysis for evaluation by a Udacity evaluator, you will submit the algorithm, dataset and list of features that you use (these are created automatically in poi_id.py). The evaluator will then use this code to test your result, to make sure we see performance that's similar to what you report. You don't need to do anything with this code, but we provide it for transparency and for your reference.

emails_by_address : this directory contains many text files, each of which contains all the messages to or from a particular email address. It is for your reference, if you want to create more advanced features based on the details of the emails dataset.

Steps to Success

We will provide you with starter code, that reads in the data, takes your features of choice, then puts them into a numpy array, which is the input form that most sklearn functions assume. Your job is to engineer the features, pick and tune an algorithm, test, and evaluate your identifier. Several of the mini-projects were designed with this final project in mind, so be on the lookout for ways to use the work you've already done.

The features in the data fall into three major types, namely financial features, email features and POI labels.

- financial features: ['salary', 'deferral_payments', 'total_payments', 'loan_advances', 'bonus', 'restricted_stock_deferred', 'deferred_income', 'total_stock_value', 'expenses', 'exercised_stock_options', 'other', 'long_term_incentive', 'restricted_stock', 'director_fees'] (all units are in US dollars)
- email features: ['to_messages', 'email_address', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi'] (units are generally number of emails messages; notable exception is 'email_address', which is a text string)
- POI label: ['poi'] (boolean, represented as integer)

You are encouraged to make, transform or rescale new features from the starter features. If you do this, you should store the new feature to my_dataset, and if you use the new feature in the final algorithm, you should also add the feature name to my_feature_list, so your coach can access it during testing. For a concrete example of a new feature that you could add to the dataset, refer to the lesson on Feature Selection.

Final Project Evaulation Instructions

When you're finished, your project will have 2 parts: the code/classifier you create and some written documentation of your work. Share your project with others and self-evaluate your project according to the rubric here (https://review.udacity.com/#!/projects/3174288624/rubric).

Before you start working on the project: Review the final project rubric carefully. Think about the following questions - How will you incorporate each of the rubric criterion into your project? Why are these aspects important? What is your strategy to ensure that your project “meets specifications” in the given criteria? Once you are convinced that you understand each part of the rubric, please start working on your project. Remember to refer to the rubric often to ensure that you are on the right track.

Items to include when sharing your work with others for feedback:

Code/Classifier

When making your classifier, you will create three pickle files (my_dataset.pkl, my_classifier.pkl, my_feature_list.pkl). The project evaluator will test these using the tester.py script. You are encouraged to use this script before checking to gauge if your performance is good enough. You should also include your modified poi_id.py file in case of any issues with running your code or to verify what is reported in your question responses (see next paragraph).

Documentation of Your Work

Document the work you've done by answering (in about a paragraph each) the questions found here (https://docs.google.com/document/d/1NDgi1PrNJP7WTbfSUuRUnz8yzs5nGVTSzpO7oeNTEWA/ec usp=sharing). You can write your answers in a PDF, Word document, text file, or similar format.

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

import sys
import pickle
sys.path.append("../tools/")

from feature_format import featureFormat, targetFeatureSplit
from tester import dump_classifier_and_data
```

```
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)
```

Task 1: Select what features you'll use.

features_list is a list of strings, each of which is a feature name.

The first feature must be "poi".

In [2]:
```
features_list = ['poi','salary']

### Load the dictionary containing the dataset
with open("final_project_dataset.pkl", "r") as data_file:
    data_dict = pickle.load(data_file)
```

Task 2: Remove outliers

Task 3: Create new feature(s)

Store to my_dataset for easy export below.

In [3]:
```
my_dataset = data_dict

### Extract features and labels from dataset for local testing
data = featureFormat(my_dataset, features_list, sort_keys = True)
labels, features = targetFeatureSplit(data)
```

Task 4: Try a varity of classifiers

Please name your classifier clf for easy export below.

Note that if you want to do PCA or other multi-stage operations, you'll need to use Pipelines.

For more info (http://scikit-learn.org/stable/modules/pipeline.html)

In [4]:
```
from sklearn.cross_validation import train_test_split
features_train, features_test, labels_train, labels_test = \
    train_test_split(features, labels, test_size=0.3, random_state=42)
from sklearn import svm
clf=svm.SVC(kernel="rbf")
clf.fit(features_train,labels_train)
clf.score(features_test,labels_test)
```

Out[4]:
```
0.6896551724137931
```

Task 5: Tune your classifier to achieve better than .3 precision and recall using our testing script.

Check the tester.py script in the final project folder for details on the evaluation method, especially the test_classifier function.

Because of the small size of the dataset, the script uses stratified shuffle split cross validation.

For more info (http://scikit-learn.org/stable/modules/generated/sklearn.cross_validation.StratifiedShuffleSplit.html)

In [ ]:
```
```

Task 6: Dump your classifier, dataset, and features_list so anyone can check your results.

You do not need to change anything below, but make sure that the version of poi_id.py that you submit can be run on its own and generates the necessary .pkl files for validating your results.

In [5]:
```
dump_classifier_and_data(clf, my_dataset, features_list)
```

In [ ]:
```
```