Data Exploration and outliers removal (Question 1)

Data Exploration

The goal of this project is to identify fraud from the Enron email and financial data. It contains information about 146 persons with 21 features for each of them. One of these features is POI (person of interest); indicating whether or not the person (or the data point) is POI. The dataset contains 18 POIs.

I will use data mining to build and test a classifier based on this dataset to help identifying POIs. The data is incomplete (there are 35 existing POIs but only 18 of them appear in our data set), but for now I will try to build a model from what I have.

Outliers removal

I first removed the persons with "NaN" or 0 as value of all features (CHAN RONNIE and LOCKHART EUGENE E).

"THE TRAVEL AGENCY IN THE PARK" and "TOTAL" do not look like valid person names so I also removed the corresponding data points.

I ended up keeping just 142 data points.

Feature Selection/Engineering (Question 2)

I started by using an extra tree classifier to build a model for predicting POIs with all the features, and I kept only the 10 most important ones. That is:

1. exercised_stock_options
2. expenses
3. other
4. from_messages
5. from_this_person_to_poi
6. from_poi_to_this_person
7. to_messages
8. total_payments
9. bonus
10. total_stock_value

I then computed two new features as follows:

- exchanged_messages= from_messages+ to_messages
- exchanged_messages_with_poi= from_this_person_to_poi+ from_poi_to_this_person

This helped me simplify data dimensionality by managing just exchanges instead of sent and received mails.

The new features had a good effect on the final model performances, taking precision/recall from 0.35253/0.37650 to 0.36439/ 0.39500. Following (Table 1) are the features I ended up using with their importance

Table 1 : Features importance

Feature	Importance
exercised_stock_options	0.305
expenses	0.066
other	0.056
total_payments	0.136
bonus	0.120
total_stock_value	0.182
exchaned_messages	0.081
exchaned_messages_with_poi	0.054

I did not have to do any scaling because the algorithms I tried did not require scaling.

Pick and Tune an Algorithm (Questions 3 and 4)

Pick an Algorithm (Question 3)

Following are the classifiers I first tried with their performances

Table 2 : Classifiers performances

Classifier	Precision	Recall
ExtraTreesClassifier	0.603	0.251
GaussianNB	0.405	0.258
RandomForestClassifier	0.459	0.170

None the classifiers meet performances requirements for the moment. GaussianNB is not parameterized so it cannot be tuned. ExtraTreesClassifier seems to perform better than RandomForestClassifier so I will try to tune it to achieve better performances.

Tune an Algorithm (Question 4)

Tuning an algorithm means iterating through the process of changing its parameters and testing until we reach satisfying performances. I you don’t do that, you might not reach the best performances possible given your algorithm and the data at hand.

I tuned up my **ExtraTreesClassifier** by using a **GridSearchCV** with the following tuned parameters:

- n_estimators : 1,5,10,15,20
- max_features : sqrt,log2,None
- min_samples_split : 2-11
- criterion : gini, entropy

I used **StratifiedShuffleSplit** for cross validation ("cv" parameter of **GridSearchCV**) because of the small size of the data set.

Since the model produced had good prediction values and bad recall, I tried to pass **recall** as the "scoring" parameter and it came out with acceptable values for both metrics.

The best estimator used the following values: criterion='gini', max_features=None,

min_samples_split=2, n_estimators=1

Validate and Evaluate (Questions 5 and 6)

Validation (Question 5)

Validation is the process of testing our model on a dataset (test data) different from the one it was trained on (training dataset). If you do it wrong, you might end up with a model that seems to be good because it performs well on the training data (**overfitting**), but which actually performs bad if we apply it on different data.

As mentioned earlier, I used **StratifiedShuffleSplit** as the "cv" parameter of **GridSearchCV** to validate my model.

Evaluation metrics (Question 6)

I reached the following performances:

Precision: **0.3264.** This means that if my model predicts a data point as POI, it will be true in **32.64%** of cases.

Recall: 0.3465. This means that if a data point is POI, my model will predict it in **34.65%** of cases.