Workflow

Data Exploration

The dataset consists of 145 data-points and 21 features. Out of the 145 data-points, 144 are used to record data about persons who used to work for Enron (each data-point refers to a person). The remaining data-point is used instead to sum-up the value of some feature across the other 144 data-points. This data-point was removed from the data-set, as not representing a real person.

Out of the 144 persons recorded in the data-set, 18 are POIs (Persons of Interest).

Out of the 21 features 18 are numeric.

Outliers Investigation

First we tried to see whether there are outliers in the dataset. The identification of outliers is subjective to some degree. To identify outliers in an automatic fashion we used the following approach.

1. For each numeric feature we considered all the values comprised between the lower and upper 2% (i.e. values higher than the 2 percentile and lower than the 98 percentile of the original range of values). We shall call the minimal and maximal value of this narrowed range *min_val* and *max_val*.
2. We computed the average distance of the values comprised between *min_val* and *max_val* included. We call such distance *avg_step*.
3. We defined an outlier as a value that satisfies the two following conditions:
    a. it is higher than *max_val* / lower than *min_val*;
    b. its distance from *max_val* / *min_val* is higher than *C* times *avg_step*.

By setting *C* = 30 we could identify 77 outliers. Of these 17 are POIs (94% of all the POIs) and 60 non-POIs (48% of all the non-POIs). We repeated this analysis for different values of the percentiles and *C*. The table below shows the results obtained.

lower perc.	upper perc.	C	N. of outliers	% of POIs in outliers	% non-POIs in outliers
1%	99%	20	89	94%	57%
2%	98%	20	91	100%	58%
3%	97%	20	104	100%	68%
1%	99%	25	63	83%	38%
2%	98%	25	77	94%	48%
3%	97%	25	92	100%	59%
1%	99%	30	53	78%	31%

2%	98%	30	60	83%	36%
3%	97%	30	92	89%	48%

This analysis shows that an outlier is more likely to be a POI than a non-POI. However the numbers show that a non-negligible portion of non-POI is also represented among the outliers. In the following table we show how much each numerical feature separates between POIs and non-POIs when looking for outliers. In the result shown below the percentiles are 1% and 99% and *C* was set to 30.

Feature	**N. of outliers**	**% of POIs in outliers**	**% non-POIs in outliers**
salary	8	17%	4%
deferral_payments	1	0%	1%
total_payments	19	17%	13%
loan_advances	2	6%	1%
bonus	6	6%	4%
restricted_stock_deferred	1	0%	1%
total_stock_value	14	22%	8%
expenses	20	28%	12%
exercised_stock_options	8	17%	4%
other	6	11%	3%
long_term_incentive	2	6%	1%
restricted_stock	12	22%	6%
to_messages	12	11%	8%
from_poi_to_this_person	10	11%	6%
from_messages	7	0%	6%
from_this_person_to_poi	3	6%	2%
shared_receipt_with_poi	19	44%	9%

In the table above we highlighted four of the numerical feature that separate better between POIs and non-POIs when looking for outliers. Such features should be considered in our final classification algorithm.

We stress that, apart from the data-point used by means of summary of the entire data-set (and that we previously removed), we saw no reason for removing any other data-point from the data-set. Outliers may indicating the presence of unlawful activities, and removing them could result in the loss of fundamental information.

Feature Selection and feature engineering

Adding new features

We engineered three new features that our intuition would consider fairly good indicators of whether someone is a POI or not. The new features are the following:

- **Feature 1** - **Definition**: Ratio 'total_payments' / 'salary'.

  Rationale: Someone whose total payments are disproportionately high with respect to their salary may be involved in unlawful financial activities.

- **Feature 2** - **Definition**: Ratio 'from_poi_to_this_person' / 'to_messages'.

  Rationale: A high ratio of emails received from POIs may suggest that the recipient is a POI him/herself.

- **Feature 3** - **Definition**: Ratio 'from_this_person_to_poi' / 'from_messages'.

  Rationale: A high ratio of emails sent to POIs may suggest that the sender is a POI him/herself.

Removing features

Because the number of POIs in the dataset is far lower than the number of non POIs, we considered for removal those features that have missing values ('NaN') in the POIs instances. The rationale is that a feature with missing values for the outnumbered class (POIs) will make the usable instances of that class decrease even further (with respect to the feature in question), hence increasing the bias of the model.

The table below shows, for each feature, the percentage of the missing values in the POIs and non-POIs classes. The highlighted rows correspond to the feature considered for removal.

Feature	**Percentage of missing values in...**	
	Non-POIs	**POIs**
salary	4.7%	0.0%
deferral_payments	10.2%	5.6%
total_payments	1.6%	0.0%
loan_advances	16.5%	11.1%
bonus	6.3%	0.0%
restricted_stock_deferred	15.0%	11.1%
deferred_income	12.6%	0.0%
total_stock_value	1.0%	0.0%
expenses	5.5%	0.0%
exercised_stock_options	3.9%	5.6%
other	6.3%	0.0%

long_term_incentive	10.2%	5.6%
restricted_stock	3.1%	0.0%
director_fees	16.5%	11.1%
to_messages	4.7%	0.0%
from_poi_to_this_person	4.7%	0.0%
from_messages	4.7%	0.0%
from_this_person_to_poi	4.7%	0.0%
shared_receipt_with_poi	4.7%	0.0%

We notice that three of the features considered for removal - namely 'loan_advances', 'exercised_stock_options' and 'long_term_incentive' - are among those that gave a better separation between POIs and non-POIs based on the outliers analysis. We then decided to remove the feature highlighted in the table above in the following two chunks, so as to assess their separate effect on the performance:

- Features considered for removal - **Group 1**
  - 'deferral_payments'
  - 'restricted_stock_deferred'
  - 'director_fees'
- Features considered for removal - **Group 2**
  - 'loan_advances'
  - 'exercised_stock_options'
  - 'long_term_incentive'.

We also considered for removal the set of those original features that we used to derive the new features. The rationale is that the new features are thought to contain the relevant information that was hidden in the original features used in their definition. In this case the original features mights not only be irrelevant for the classification task, but could also decrease the predictive power of the model when used together with the new features.

- Features considered for removal - **Group 3**
  - 'to_messages'
  - 'from_poi_to_this_person'
  - 'from_messages'
  - 'from_this_person_to_poi'
  - 'salary'
  - 'total_payments'

Performance of the classifier

Here we compare two algorithms: naive bayes networks and Decision Trees. In the tables

below we summarise the performance of these two classifiers in different condition, i.e. with removal or addition of specific (groups of) features.

Naive Bayes		**Added new features**							
		None	1	2	3	1, 2	1, 3	2, 3	1, 2, 3
Removed groups of features	None	P: 0.149 R: 0.842	P: 0.149 R: 0.842	P: 0.149 R: 0.842	P: 0.149 R: 0.842	P: 0.149 R: 0.842	P: 0.149 R: 0.842	P: 0.149 R: 0.842	P: 0.149 R: 0.842
	1	P: 0.217 R: 0.279	P: 0.210 R: 0.284	P: 0.217 R: 0.279	P: 0.220 R: 0.281	P: 0.210 R: 0.284	P: 0.212 R: 0.285	P: 0.221 R: 0.280	P: 0.212 R: 0.286
	2	P: 0.149 R: 0.848	P: 0.149 R: 0.847	P: 0.149 R: 0.848	P: 0.149 R: 0.848	P: 0.149 R: 0.847	P: 0.149 R: 0.847	P: 0.149 R: 0.848	P: 0.149 R: 0.847
	3	P: 0.154 R: 0.897	P: 0.154 R: 0.897	P: 0.154 R: 0.897	P: 0.154 R: 0.897	P: 0.154 R: 0.897	P: 0.154 R: 0.897	P: 0.154 R: 0.897	P: 0.154 R: 0.897
	1, 2	P: 0.365 R: 0.282	P: 0.308 R: 0.299	P: 0.359 R: 0.282	P: 0.357 R: 0.282	P: 0.300 R: 0.298	P: 0.298 R: 0.303	P: 0.349 R: 0.283	P: 0.286 R: 0.303
	1, 3	P: 0.246 R: 0.319	P: 0.239 R: 0.325	P: 0.246 R: 0.319	P: 0.246 R: 0.319	P: 0.240 R: 0.327	P: 0.240 R: 0.325	P: 0.246 R: 0.319	P: 0.239 R: 0.325
	2, 3	P: 0.154 R: 0.902	P: 0.154 R: 0.899	P: 0.154 R: 0.902	P: 0.154 R: 0.902	P: 0.154 R: 0.899	P: 0.154 R: 0.899	P: 0.154 R: 0.902	P: 0.154 R: 0.899
	1, 2, 3	P: 0.418 R: 0.300	**P: 0.322** **R: 0.322**	P: 0.410 R: 0.297	P: 0.386 R: 0.295	P: 0.311 R: 0.320	P: 0.303 R: 0.319	P: 0.373 R: 0.295	P: 0.292 R: 0.323

Decision Tree		**Added new features**							
		None	1	2	3	1, 2	1, 3	2, 3	1, 2, 3
Removed groups of features	None	P: 0.228 R: 0.218	P: 0.248 R: 0.249	P: 0.226 R: 0.215	P: 0.322 R: 0.311	P: 0.245 R: 0.247	P: 0.322 R: 0.317	P: 0.316 R: 0.303	P: 0.313 R: 0.314
	1	P: 0.224 R: 0.212	P: 0.256 R: 0.251	P: 0.225 R: 0.213	P: 0.320 R: 0.299	P: 0.248 R: 0.245	P: 0.316 R: 0.309	P: 0.318 R: 0.303	P: 0.314 R: 0.307
	2	P: 0.252 R: 0.235	P: 0.270 R: 0.266	P: 0.242 R: 0.232	P: 0.335 R: 0.320	P: 0.261 R: 0.264	P: 0.322 R: 0.311	P: 0.321 R: 0.307	P: 0.325 R: 0.314
	3	P: 0.245 R: 0.234	P: 0.275 R: 0.270	P: 0.234 R: 0.226	P: 0.348 R: 0.323	P: 0.267 R: 0.266	P: 0.354 R: 0.333	P: 0.344 R: 0.319	P: 0.326 R: 0.315
	1, 2	P: 0.245 R: 0.232	P: 0.271 R: 0.265	P: 0.234 R: 0.221	P: 0.329 R: 0.313	P: 0.262 R: 0.261	P: 0.320 R: 0.312	P: 0.315 R: 0.307	P: 0.321 R: 0.314
	1, 3	P: 0.229 R: 0.216	P: 0.251 R: 0.244	P: 0.224 R: 0.210	P: 0.346 R: 0.338	P: 0.248 R: 0.246	P: 0.318 R: 0.312	P: 0.344 R: 0.336	P: 0.320 R: 0.320
	2, 3	P: 0.259 R: 0.243	P: 0.287 R: 0.273	P: 0.250 R: 0.234	P: 0.359 R: 0.326	P: 0.276 R: 0.266	P: 0.350 R: 0.327	P: 0.352 R: 0.325	P: 0.338 R: 0.321
	1, 2, 3	P: 0.262 R: 0.245	P: 0.284 R: 0.265	P: 0.242 R: 0.226	P: 0.358 R: 0.342	P: 0.266 R: 0.259	P: 0.334 R: 0.320	P: 0.348 R: 0.330	P: 0.330 R: 0.321

The highlighted cells are those where both precision (P) and recall (R) are above 0.3, as requested in the project instructions. Both classifiers were used with default parameters. We notice that Decision Trees are better at this classification problem. In particular, precision and recall are both above 0.3 in a larger set of conditions. It appears that for the Decision Tree classifier the addition of 'Feature 3' (ratio between 'from_this_person_to_poi' and 'from_messages') plays a major role in obtaining acceptable performances (P, R >0.3).

The algorithm and the conditions that gave the best results are the following:

Classifier: Decision Tree

Conditions: Features added: Feature 3

Features removed: Group 1, Group 2 and Group 3.

Tune the algorithm

We now consider the classifier and the conditions that gave the best results in the previous section and try to further improve the performance of the algorithm by adjusting some of its parameters.

The three parameters investigated here are the following:

- **class_weight** - Value used: None (all classes are supposed to have weight one) or 'auto' (weights are inversely proportional to class frequencies).
- **max_depth** - int or None (default=None). The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure.
- **max_features** - The number of features to consider when looking for the best split.

When using class_weight = 'auto' we had in increase in precision (~15%), recall (~5%) and accuracy (~2%). We then kept the value of class_weight set to 'auto' and performed a scan on the other two parameters. The results are shown in the table below.

		max_depth					
		2	10	20	30	40	50
max_features	1	P: 0.275 R:0.681 A:0.718	P: 0.330 R: 0.292 A: 0.826	P: 0.330 R: 0.293 A: 0.826	P: 0.325 R: 0.282 A: 0.826	P: 0.339 R: 0.294 A: 0.830	P: 0.345 R: 0.304 A: 0.830
	3	P: 0.300 R: 0.663 A: 0.745	P: 0.359 R: 0.314 A: 0.834	P: 0.376 R: 0.319 A: 0.839	P: 0.369 R: 0.312 A: 0.837	P: 0.362 R: 0.308 A: 0.835	P: 0.388 R: 0.323 A: 0.842
	5	P: 0.330 R: 0.687 A: 0.77	P: 0.375 R: 0.337 A: 0.837	P: 0.374 R: 0.327 A: 0.837	P: 0.366 R: 0.313 A: 0.836	P: 0.371 R: 0.323 A: 0.837	P: 0.381 R: 0.332 A: 0.839
	7	P: 0.338 R: 0.680 A: 0.780	P: 0.396 R: 0.348 A: 0.841	P: 0.387 R: 0.340 A: 0.840	**P: 0.400** **R: 0.349** **A: 0.844**	P: 0.389 R: 0.341 A: 0.841	P: 0.397 R: 0.346 A: 0.843

Apart from one case (upper right cell), all the parameter values produce results that are acceptable, both in terms of precision (P) and recall (R). We remind that acceptable results are those where P and R are both greater than 0.3. There are different reasons why one may prefer to prioritize larger values of recall over larger values of precision, or vice versa. In this case we decided to chose that combination of parameter resulting in the highest accuracy (A). The cell corresponding to such combination of parameter values is highlighted in the table.

Validate and Evaluate

To choose the "best" algorithm for this classification problem we looked at precision, recall and accuracy.

- **Precision** is defined as the ratio between the true positive (elements in the dataset that have been *correctly* classified as positive) and the total number of elements classified as positive.
- **Recall** is defined as the ratio between the true positive and the total number of *real* positive in the dataset.
- **Accuracy** is defined as the ratio between the number of elements that are correctly classified (either as positive or negative) and the total number of elements in the dataset.

We stress again that there are different reasons why one may prefer to prioritize larger values of recall over larger values of precision, or vice versa. For example, if not jailing innocents is deemed more important that catching all the POIs, then we would be prioritizing larger values of recall. In this case we decided to go for the highest value of accuracy, provided that precision and recall are both larger than 0.3.

Because of the small size of the dataset, the classifier was validated using a stratified shuffle split cross validation. This validation method consists of splitting the data-set in randomized foldes where the percentage of samples for each class in the training-set and test-set is preserved as in the original data-set. In general, the reason why the data-set is split in a training-set and a test-set is to gain a non-biased (or less biased) idea of how the classifier really performs. The classifier is trained on the training-set, and its performances are checked against the test-set, which contains elements that were not used in the training process. This is a way to emulate a real situation, where our trained algorithm is used to classify elements whose class is unknown.