Marko Horvat

Identity fraud from Enron email

The purpose of this project is to use machine learning to identify Enron employees who have committed fraud based on the public Enron financial and email dataset.

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, a significant amount of typically confidential information entered into the 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.

1. Summarize for us the goal of this project and how machine learning is useful in trying to accomplish it. As part of your answer, give some background on the dataset and how it can be used to answer the project question. Were there any outliers in the data when you got it, and how did you handle those? [relevant rubric items: “data exploration”, “outlier investigation”]

The dataset contains 146 users. 18 of them are considered POIs and128 are non-POIs. We can use machine learning to answer questions like "is it possible to identify patterns in the sent emails?”. With usage of clustering we will be able to identify who is a member of the board of directors, and who is just an employee for example.

The dataset has 21 features:

POI_label = ['poi']

features_list = ['poi','salary', 'expenses', 'total_stock_value', 'bonus', 'from_poi_to_this_person', 'shared_receipt_with_poi'] # You will need to use more features

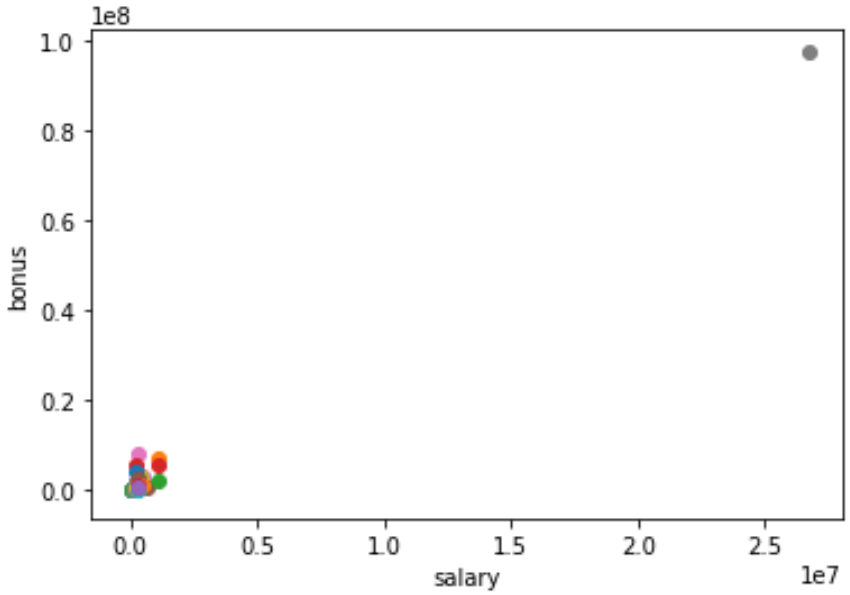
financial_features = ['salary', 'deferral_payments', 'total_payments', 'loan_advances', 'bonus', 'restricted_stock_deferred', 'deferred_income', 'total_stock_value', 'expenses', 'exercised_stock_options', 'other', 'long_term_incentive', 'restricted_stock', 'director_fees']

email_features = ['to_messages', 'email_address', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi']

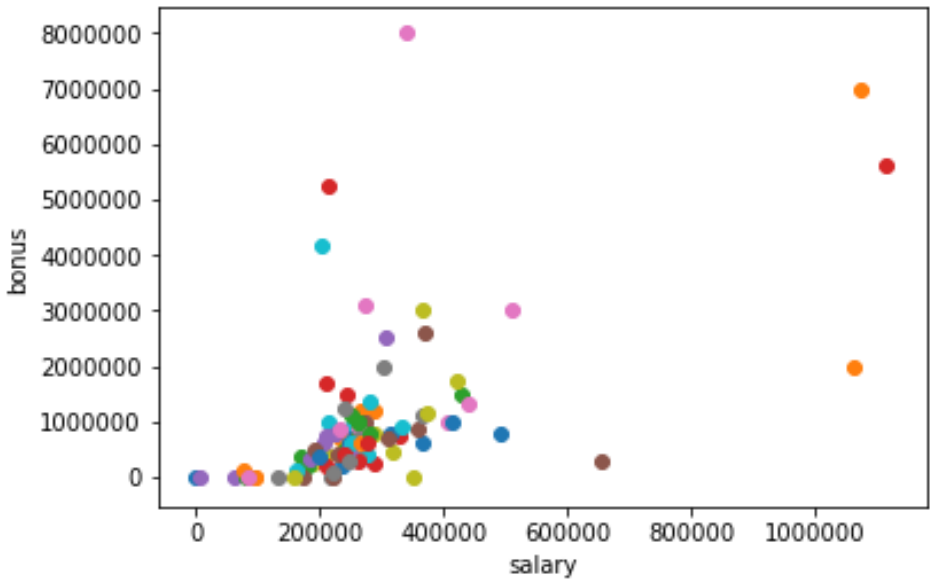
Some features have missing values (NaN).

Below is a list and a count of missing values:
{'salary': 51,
'to_messages': 60,
'deferral_payments': 107,
'total_payments': 21,
'exercised_stock_options': 44,
'bonus': 64,
'director_fees': 129,
'restricted_stock_deferred': 128,
'from_messages': 60,
'total_stock_value': 20,
'expenses': 51,
'from_poi_to_this_person': 60,
'loan_advances': 142,
'email_address': 35,
'other': 53,
'from_this_person_to_poi': 60,
'poi': 0,
'deferred_income': 97,
'shared_receipt_with_poi': 60,
'restricted_stock': 36,
'long_term_incentive': 80}

Plot below, bonus vs. salary, shows an outlier representing the 'TOTAL' column. Another one labelled TRAVEL AGENCY IN THE PARK is also removed as well. Also we removed LOCKHART EUGENE E since all the data related to it are NaN.



The scatterplot below shows dataset after removing the outlier.



2. What features did you end up using in your POI identifier, and what selection process did you use to pick them? Did you have to do any scaling? Why or why not? As part of the assignment, you should attempt to engineer your own feature that does not come ready-made in the dataset -- explain what feature you tried to make, and the rationale behind it. (You do not necessarily have to use it in the final analysis, only engineer and test it.) In your feature selection step, if you used an algorithm like a decision tree, please also give the feature importances of the features that you use, and if you used an automated feature selection function like SelectKBest, please report the feature scores and reasons for your choice of parameter values. [relevant rubric items: “create new features”, “intelligently select features”, “properly scale features”]

I created a new feature which I added to my original features_list. The feature name is fraction_poi. Feature represents the ratio of the messages from POI to this person against all the messages sent to this person.

Univariate feature selection works by selecting the best features based on univariate statistical tests. It can be seen as a preprocessing step to an estimator. Scikit-learn exposes feature selection routines as objects that implement the transform method.

VarianceThreshold function is used to removes all features with variance below 80%.

To decide which features is the best to use, I used an automated feature selection function SelectKBest, which selects the K features that are most powerful. f_classif compute the ANOVA F-value for the provided sample. Below are scores of all the features.

[('exercised_stock_options', 25.097541528735491),
('total_stock_value', 24.467654047526398),
('bonus', 21.060001707536571),
('salary', 18.575703268041785),
('deferred_income', 11.595547659730601),
('long_term_incentive', 10.072454529369441),
('restricted_stock', 9.3467007910514877),

('total_payments', 8.8667215371077717),
('shared_receipt_with_poi', 8.7464855321290802),
('loan_advances', 7.2427303965360181),
('expenses', 6.2342011405067401),
('from_poi_to_this_person', 5.3449415231473374),
('other', 4.204970858301416),
('fraction_poi', 3.2107619169667441),
('from_this_person_to_poi', 2.4265081272428781),
('director_fees', 2.1076559432760908),
('to_messages', 1.6988243485808501),
('deferral_payments', 0.2170589303395084),
('from_messages', 0.16416449823428736),
('restricted_stock_deferred', 0.06498431172371151)]

In first submission I took the first 10 features (k = 10) along with POI as they obtained the highest scores from SelectKBest results. My engineered feature received a mid-low score so I did not include it in my final features. On suggestion of reviewer I analysed data little bit more to see how Accuracy, Precision and Recall changes on data with and without new feature along with change of k. Based on table below I can concluded that newly created feature lower the scoring and therefore will not be included in final analysis. Also based on the scoring below I chose k to be 8.

k	Accuracy (kbest)	Precision (kbest)	Recall (kbest)	Accuracy (kbest+new)	Precision (kbest+new)	Recall (kbest+new)
10	0.816590909091	0.344198729411	0.343277777778	0.816136363636	0.339672322775	0.334253968254
9	0.8475	0.389051587302	0.328547619048	0.845	0.382904761905	0.324880952381
8	0.85	0.400371933622	0.325103174603	0.8475	0.393804473304	0.325992063492
7	0.854761904762	0.432977633478	0.373191558442	0.845238095238	0.409030747031	0.377556277056
6	0.858809523781	0.457334776335	0.379957792208	0.850238095238	0.425247474747	0.378746753247

['poi', 'exercised_stock_options', 'total_stock_value', 'bonus', 'salary', 'deferred_income', 'long_term_incentive', 'restricted_stock', 'total_payments']

Selected features had different units and some of them have a big values and therefore they needed to be transform. I used MinMaxScaler to scale selected features to a range between 0 and 1.

3. What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms? [relevant rubric item: “pick an algorithm”]

I tested three different algorithms and used Naive Bayes as final one as it scored the highest evaluation metrics. Other algorithms are SVM and Random Forest.

Algorithm	Accuracy	Precision	Recall
Naive Bayes	0.85	0.400371933622	0.325103174603
Support Vector Machines	0.868409090909	0.205833333333	0.0765396825397
Decision Tree	0.803636363636	0.241963647464	0.277226190476

4. What does it mean to tune the parameters of an algorithm, and what can happen if you don't do this well? How did you tune the parameters of your particular algorithm? What parameters did you tune? (Some algorithms do not have parameters that you need to tune -- if this is the case for the one you picked, identify and briefly explain how you would have done it for the model that was not your final choice or a different model that does utilize parameter tuning, e.g. a decision tree classifier). [relevant rubric items: "discuss parameter tuning", "tune the algorithm"]

Tuning is essentially selecting the best parameters for an algorithm to optimize its performance given a working environment such as hardware, specific workloads, etc. And tuning in machine learning is an automated process for doing this. Failure in choosing the right parameters may lead to the low prediction power such as low accuracy or/and low precision.

I used GridSearchCV function to get the best parameters for each of algorithm I used:

Naive Bayes: no need to specify any parameter

SVM: kernel = 'linear', C = 1, gamma = 1,

Decision Tree: splitter = random, criterion = gini,

5. What is validation, and what's a classic mistake you can make if you do it wrong? How did you validate your analysis? [relevant rubric items: "discuss validation", "validation strategy"]

Validation is a strategy to separate data into training and testing dataset. This allows to train model on a training set and then test the given model on an independent test data. This method will reduce the problem of over fitting the model to the dataset which on which was trained initially. It we only tested on the model on training dataset that would be bad because we would not know how well our model would performed on a new dataset. While validating test set, if we get poor results it would mean that there is a chance of overfitting our model to the training data.

To avoid overfitting mistake I applied cross validation technique to split the data into training data and test data 100 times, calculate the accuracy, precision, and recall of each iteration.

6. Give at least 2 evaluation metrics and your average performance for each of them. Explain an interpretation of your metrics that says something human-understandable about your algorithm's performance. [relevant rubric item: "usage of evaluation metrics"]

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. It's calculated as True Positive/(True Positive + False Negative). In our case proportion of the correct prediction of all the people who are predicted to be POI. A recall of 0.325 means that in 100 true POI, 33 POI are correctly classified.

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. It's calculated as True Positive/(True Positive + False Positive). In our case the proportion of the POI the model can detect of all the POIs. A precision of 0.4 means that in total 100 persons classified as POI, 40 persons are actually POI.

Naive Bayes algorithm has average precision of 0.400371933622 and average recall of 0.325103174603

7. References:

https://www.programiz.com/python-programming/methods/built-in/sorted

http://www.pythoncentral.io/how-to-sort-a-list-tuple-or-object-with-sorted-in-python/

http://scikit-learn.org/stable/modules/cross_validation.html

http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html

https://stats.stackexchange.com/questions/19048/what-is-the-difference-between-test-set-and-validation-set

https://machinelearningmastery.com/difference-test-validation-datasets/

http://scikit-learn.org/stable/modules/tree.html

https://stackoverflow.com/questions/22903267/what-is-tuning-in-machine-learning

http://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

http://scikit-learn.org/stable/modules/feature_selection.html

http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.VarianceThreshold.html