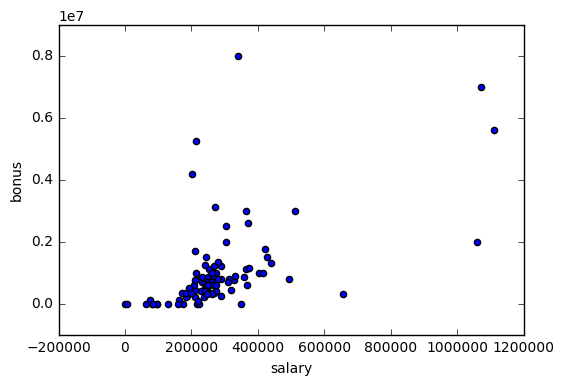
Jayden Yuen

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”]

***Intro***

The goal of this enron project is to use machine learning/algorithms and indentify persons of interest (POI) from the scandal.

The Enron dataset has over 600,000 emails by 158 employes that is acquired by the Federal Energy Regulatory Commision during its investigation after the company’s collpase. In contrast to 62,320 emails disclosed by the Clinton’s office in March 2015. This dataset would still consider one of the biggest public dataset.

Exploring the data by plotting salary and bonus ,and counting all the outliers.

Number of flagged Persons of Interest: 18

Number of people without POI flag: 128

146 executives in Enron Dataset

23 POI

95 have qualified salary

34 email found

outliers:

104 total index (not valid name)

101 travel agency in the park (not valid name)

Lockhart Eugene E (POI that has almost all missing data )

The outliers :

[('SKILLING JEFFREY K', 1111258), ('LAY KENNETH L', 1072321), ('FREVERT MARK A', 1060932), ('PICKERING MARK R', 655037)]

Removing outliers by .pop(‘total’,0)

.pop("THE TRAVEL AGENCY IN THE PARK", 0)

However out of 4 outlier,only 2 are POI.(SKILLING JEFFREY K LAY KENNETH L)

LOCKHART EUGENE E

{'salary': 'NaN',

'to\_messages': 'NaN',

'deferral\_payments': 'NaN',

'total\_payments': 'NaN',

'exercised\_stock\_options': 'NaN',

'bonus': 'NaN',

'restricted\_stock': 'NaN',

'shared\_receipt\_with\_poi': 'NaN',

'restricted\_stock\_deferred': 'NaN',

'total\_stock\_value': 'NaN',

'expenses': 'NaN',

'loan\_advances': 'NaN',

'from\_messages': 'NaN',

'other': 'NaN',

'from\_this\_person\_to\_poi': 'NaN',

'poi': False,

'director\_fees': 'NaN',

'deferred\_income': 'NaN',

'long\_term\_incentive': 'NaN',

'email\_address': 'NaN',

'from\_poi\_to\_this\_person': 'NaN'}

Therefore, LOCKHART EUGENE E are drop due to there is no data on it. In addition, TOTAL and THE TRAVEL AGENCY IN THE PART, are dropped due to both does not represent any indivudal.

Removing outliers by .pop(‘total’,0)

However out of 4 outlier,only 2 are POI.(SKILLING JEFFREY K LAY KENNETH L)

Features that contains NAN count

bonus 64

deferral\_payments 107

deferred\_income 97

director\_fees 129

email\_address 35

exercised\_stock\_options 44

expenses 51

from\_messages 60

from\_poi\_to\_this\_person 60

from\_this\_person\_to\_poi 60

loan\_advances 142

long\_term\_incentive 80

other 53

poi 0

restricted\_stock 36

restricted\_stock\_deferred 128

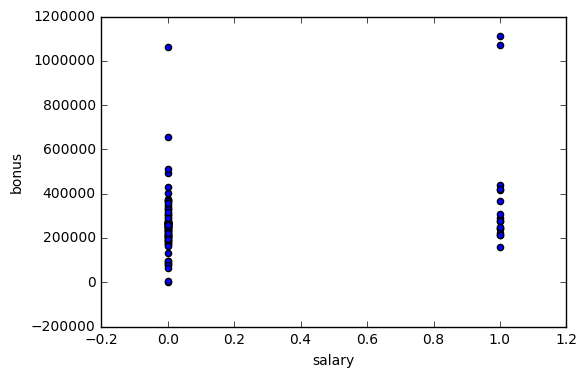
salary 51

shared\_receipt\_with\_poi 60

to\_messages 60

total\_payments 21

total\_stock\_value 20   
dtype: int64 (146, 21)

1. 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”, “properly scale features”, “intelligently select feature”]

New features are created:

"poi\_sender\_fract", "poi\_recipient\_fract"

Decision tree such as selectKbest are used for this dataset.

K-best features:

[(True, 'exercised\_stock\_options', 25.097541528735508),

(True, 'total\_stock\_value', 24.467654047526398),

(True, 'bonus', 21.060001707536582),

(True, 'salary', 18.575703268041824),

(True, 'poi\_recipient\_fract', 16.641707070469),

(False, 'deferred\_income', 11.595547659731226),

(False, 'long\_term\_incentive', 10.072454529369441),

(False, 'restricted\_stock', 9.346700791051429),

(False, 'total\_payments', 8.8667215371077752),

(False, 'shared\_receipt\_with\_poi', 8.7464855321290802),

(False, 'loan\_advances', 7.2427303965360172),

(False, 'expenses', 6.2342011405067357),

(False, 'poi\_sender\_fract', 3.2107619169667658),

(False, 'director\_fees', 2.1076559432760891),

(False, 'deferral\_payments', 0.21705893033950577),

(False, 'restricted\_stock\_deferred', 0.064984311723709984)]

As we goes on feature poi\_sender\_fract (0.14139568871)

and feature poi\_recipient\_fract (0.120901730257) are placed into 3rd and 4th. Therefore, it is a good indicator.

The accuracy is around 0.733 that was not quiet impressive.

The features that I consider would be all of those above, because the features after total\_stock\_value is 0. However, with such low precision and recall, I can either pick those that is above 0.1 or play around with different number of top features by comparing the final model’s performances. 2

I created new features such as fraction from or to POI email. With the scaling, I used MinMaxScaler due to the ratios of email and income (salary, stocks) are different.

Here are the before and after feature scoring of the new features:

Without new features created:

accuracy 0.866666666667 Decision tree algorithm time: 0.006 s

Feature Ranking:

1 feature salary (0.199959105222)

2 feature bonus (0.189313620185)

3 feature deferral\_payments (0.151335953699)

4 feature total\_payments (0.143762958004)

5 feature loan\_advances (0.128893662728)

6 feature restricted\_stock\_deferred (0.103660511824)

7 feature deferred\_income (0.0710441131494)

8 feature total\_stock\_value (0.012030075188)

With new features created:

accuracy 0.666666666667 Decision tree algorithm time: 0.002 s

Feature Ranking:

1 feature salary (0.328822055138)

2 feature bonus (0.197778963276)

3 feature deferral\_payments (0.149133225481)

4 feature poi\_sender\_fract (0.142291211901)

5 feature poi\_recipient\_fract (0.0748538011696)

6 feature total\_payments (0.0721804511278)

7 feature loan\_advances (0.0349402919062)

As we goes on feature poi\_sender\_fract (0.14139568871)

and feature poi\_recipient\_fract (0.120901730257) are placed into 3rd and 4th. Therefore, it is a good indicator.

The accuracy is around 0.733 that was not quiet impressive.

The features that I consider would be all of those above, because the features after total\_stock\_value is 0. However, with such low precision and recall, I can either pick those that is above 0.1 or play around with different number of top features by comparing the final model’s performances.

Also the accuracy jumps upto 0.866 that a lot more.

1. 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”]

GaussianN [ 0.82758621 0.86206897 0.82142857] accuracy 0.892857142857 NB algorithm time: 0.006 s

RandomForestClassifier [ 0.89655172 0.82758621 0.89285714] accu racy 0.892857142857 RF algorithm time: 26.237 s

AdaBoostClassifier [ 0.89655172 0.86206897 0.89285714] accuracy 0.892857142857 ABC algorithm time: 38.843 s

1. 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? (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 item: “tune the algorithm”]

With K-best, the algorithm evalues that after these 5 variables, it drops signifantly. Therefore, those are the features I picked.

[(True, 'salary', 25.097541528735508),

(True, 'deferral\_payments', 24.467654047526398),

(True, 'poi\_recipient\_fract', 21.060001707536582),

(True, 'poi\_sender\_fract', 18.575703268041824),

(True, 'expenses', 16.641707070469)

I recalled a phrase from deep learning course by Udacity that tuning parameters is like a pair of skinny jeans has to fit well on a person. Likewise, it is a final step in the process of applied machine learning before presenting results. Grid search, and random search, are the most common on.

accuracy before tuning 0.892857142857

Decision tree algorithm time: 0.002 s done in 0.002s

Validating algorithm: accuracy after tuning = 0.892857142857

precision = 0.5

recall = 0.666666666667

Wtih grid search there is an option to set number of features automatically and spliot sample from 2 – 10.

1. 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 item: “validation strategy”]

When perform supervised learning, 2 types of datasets are needed where data seperated from the whole set of data into 2 parts.There are traning and validation set. The training set is use to train the model and testing set is to test wether the model is validated or not.

Validation ensure the model performs well when the data does not appear on the training set.

Classic mistake can be overfitting, where the model is being too well trained however it performs poorly with the validation set.

Cross validation which is the method I used on this dataset, is a techique that shuffled to reduce bias and unbalance distrubtion.

1. 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”]

DecisionTreeClassifier(class\_weight=None,

criterion='gini',

max\_depth=None,

max\_features=None,

max\_leaf\_nodes=None,

min\_impurity\_split=1e-07,

min\_samples\_leaf=1,

min\_samples\_split=5,

min\_weight\_fraction\_leaf=0.0,

presort=False,

random\_state=None,

splitter='best')

Got a divide by zero when trying out:

DecisionTreeClassifier(class\_weight=None,

criterion='gini',

max\_depth=None,

max\_features=None, max\_leaf\_nodes=None,

min\_impurity\_split=1e-07,

min\_samples\_leaf=1,

min\_samples\_split=5,

min\_weight\_fraction\_leaf=0.0,

presort=False,

random\_state=None,

splitter='best')