Enron Submission Free-Response Questions

> 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”]

Introduction

The Enron fraud is the largest case of corporate fraud in American history. Founded in 1985, Enron Corporation went bankrupt by end of 2001 due to widespread corporate fraud and corruption. Before its fall, Fortune magazine had named Enron "America's most innovative company" for six consecutive years. So what happened? Who were the culprits?

In this project, I will play detective and build a classification algorithm to predict a person of interest identifier (POI) based on email and financial features in the combined dataset. A POI is anyone who has been indicted, settled without admitting the guilt and testified in exchange for immunity. We will check our predicted POI against actual POI in the dataset to evaluate our prediction.

Totally 145 data points in data set and have two class:POI/non POI. Two features in the data: 14 Finance and 6 Email.But there are many defect value in data set which like loan_advaces.I found two abnormal values: at least 5 millions bonus and over 1 million salary.the suspicious person to correspond these two value is LAY KENNETH L and SKILLING JEFFREY K.From the Enron background to realized these two person are core person in Enron.

Outlier investigation

Visualization is one of the most powerful tools for fingding outliers.Upon plotting salary against bonus,there is an outlier that pops out immediately-"TOTAL"(move the cursor and examine the points in scatterplot).Upon closer examination,I found on more entry which is not the name od a real person"THE TRAVEL AGENCY IN THE PARK".The entry is dropped from the dataset.The entried which have all the features as'NaN' are also dropped from the dataset.

> 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”]

First of all,I was cancel the total_payments and total_stock_value,because those are sum of other values.And other NAN I was choice to leave behind it maybe have other hinting information of values.

Base on those values I was created tow new value "from_this_person_to_poi_ratio"and"from_poi_to_this_person_ratio"which means"the ratio of suspicious person send POI mails to totally send mails"and "the ratio of suspicious person recived POI mails to totally recive mails".Because of send and recive mails ratio can show the suspicious person who are the POI.

After the treatment the values the evaluation result as below：

```
In [39]: %run poi_id.py
```

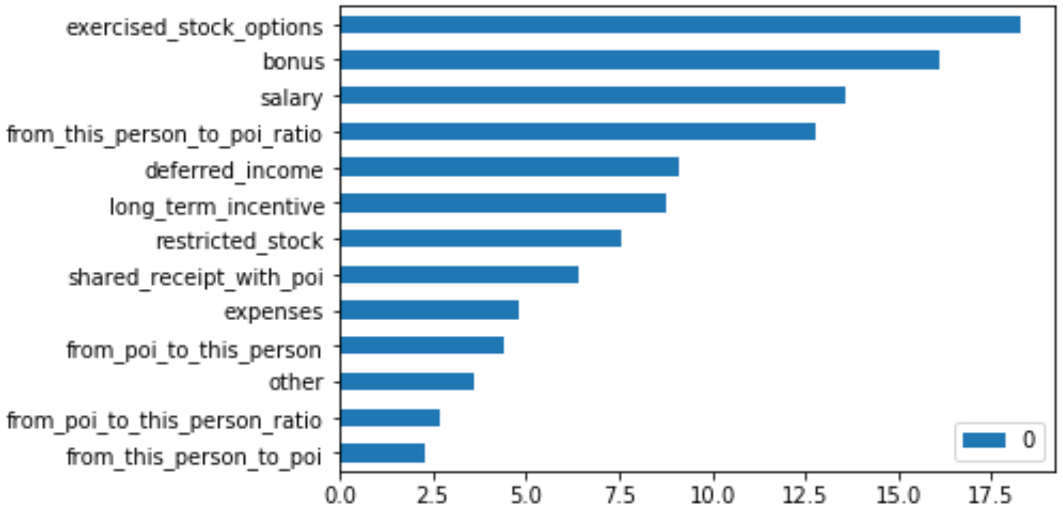
```
estimator: Pipeline(steps=[('pca', PCA(copy=True, iterated_power='au
to', n_components=8, random_state=42,
  svd_solver='auto', tol=0.0, whiten=False)), ('clf', GaussianNB(pri
ors=None))]), f1 score: 0.330611904762

estimator: Pipeline(steps=[('scaler', StandardScaler(copy=True, with
_mean=True, with_std=True)), ('pca', PCA(copy=True, iterated_power='
auto', n_components=4, random_state=42,
  svd_solver='auto', tol=0.0, whiten=False)), ('clf', SVC(C=1.0, cac
he_size=200, class_weight='balanced', coef0=0.0,
  decision_function_shape=None, degree=3, gamma=0.10000000000000001,
  kernel='sigmoid', max_iter=-1, probability=False, random_state=Non
e,
  shrinking=True, tol=0.001, verbose=False))]), f1 score: 0.49590033
3

estimator: Pipeline(steps=[('pca', PCA(copy=True, iterated_power='au
to', n_components=8, random_state=42,
  svd_solver='auto', tol=0.0, whiten=False)), ('clf', DecisionTreeCl
assifier(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=42, splitter='random'))]), f
1 score: 0.235250793651

estimator: Pipeline(steps=[('pca', PCA(copy=True, iterated_power='au
to', n_components=2, random_state=42,
  svd_solver='auto', tol=0.0, whiten=False)), ('clf', AdaBoostClassi
fier(algorithm='SAMME',
          base_estimator=DecisionTreeClassifier(class_weight=None, c
riterion='gini', max_depth=1,
            max_features...dom_state=None, splitter='best'),
          learning_rate=1.0, n_estimators=50, random_state=None))]),
f1 score: 0.259523809524


<matplotlib.figure.Figure at 0x7f5394ec92d0>
```

Properly scale features

Since we are going to perform a principal component analysis to reduce dimensionality later on,and many machine learning models ask for scaled features,a standardization of the features is going to be tested as the first step of our classification pipeline.if it improves the evaluation score of the model then the chosen final model will have this scaling step.

I have used SVM as my final algorithm that require feature scaling and calculate euclidean distance between points.

In [40]:
```
import operator
import pandas as pd
%matplotlib inline
selected_features = dict()
for f in features_list[1:]:
    selected_features[f] = feature_scores[f]
sorted_features = sorted(selected_features.items(), key=operator.itemg
etter(1), reverse=True)
indices = []
scores = []
for t in sorted_features:
    indices.append(t[0])
    scores.append(t[1])
df = pd.DataFrame(scores, index=indices)
df.plot(kind='barh').invert_yaxis()
```



Feature selection

I performed SelectKbest on remaining features and I removed features which had more than 50% the values missing(NaN),and I also remove the the lowest score 2.0 .From analysis graph,the score "from_this_person_to_poi_ratio" over 12.5 and the score "from_poi_to_this_person_ratio only 2.5. So the ratio of this person send email to POI has highline in here.

> 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 end up to using Support Vector Machines algorithm.which scored 0.496 on the nested cross-validation f1.The algorithms tested were: "Gaussian Naive-Bayes","Decision Tree Classifier"and "AdaBoost".

The socres obtained for them are as follows:

- Support Vector Machines 0.496
- Gussian Naive-Bayes 0.331
- AdaBoost 0.26
- Decision Tree 0.235

Although the other tested models scored better on other evaluation metrics,it is the nested cross-validation score that best depicts how the model generalizes on unseen data,therefore the Support Vector Machines was the chosen model.

> 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”]

A crucial part of selecting a machine learning algorithm is to adjust it's parameters in order to maximize the evaluation metrics.If the parameters are not properly tuned,the algorithm can underfit or overfit the data,hence producin suboptimal results.

From my choiced support vector machines algorithm.I used Pipeline+GridSearchCV to tuned the parameters.Used Pipeline to set up:at first,use standard feature scaling(SVM algorithm need) and PCA;Then through GridSearch CV and Param_grid parameters transmit parameter C,kernal,class_weight and gamma etc.different values.Lastly,according to GridSearch CV gave optimum parameter group.

> 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 in machine learning consists of evaluating a model using data that was not touched during the training process. A classic mistake is to ignore this rule, hence obtaining overly optimistic results due to overfitting the training data, but very poor performance on unseen data.

It is a good practice to separate data in three parts: training, cross-validation and test sets. The model is tuned to maximize the evaluation score on the cross-validation set, and then the final model efficiency is measured on the test set.

Firstly used GridSearchCV to validate the evaluate an algorithm,then used estimate_classifer function to calculate f1 score and choice the highest f1 score as final clf.

The estimate_classifier function principle: StratifiedShuffleSplit algorithm effectively uses a series of train/validation/test set splits, and through to changed sequence of parameters to validate and test.after that calling sklearn.metrics.f1_score to calculate f1 score.used f1 average to evaluating the performance of an algorithm.

> 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”

From my choice the Suppore Vector Machines algorithm.the best performance of parameter is C=1.0,class_weight ='balanced',kernel='sigmoid'and gamma =0.10000000000000001,PCA's n_components parameter = 4.

Precision = 0.331 which means there have 33.1% really POI in mark of POI person ;recall rate is 0.862 that means really POI person have 86.2% was corrected mark.

7, Reference

1.Enron Corporation (https://en.wikipedia.org/wiki/Enron)

2.Identify Fraud From Enron Data (https://arjan-hada.github.io/enron-fraud.html)

3.Sample Code (https://jefflirion.github.io/udacity/Intro_to_Machine_Learning/Lesson5.html)