Udacity Data Analyst Nanodegree

Introduction To Machine Learning

Project - Identify Fraud from Enron Email

Author – Sourabh Chakraborty

Date – 09/Dec/2017

Contents

1. Project Overview
2. Dataset Overview
3. Analysis
4. Feature Engineering
5. Model Building & Results
6. Validation
7. Project Files
8. Conclusion

1. Project Overview

Enron Corporation was an American energy, commodities, and services company based in Houston, Texas. Enron employed approximately 20,000 staff and was one of the world's major electricity, natural gas, communications, and pulp and paper companies, with claimed revenues of nearly $101 billion during 2000. Fortune named Enron "America's Most Innovative Company" for six consecutive years.

At the end of 2001, it was revealed that its reported financial condition was sustained by institutionalized, systematic, and creatively planned accounting fraud, known since as the Enron scandal. Enron has since become a well-known example of willful corporate fraud and corruption. Accounting fraud perpetrated by top executives resulted in one of the largest bankruptcies in U.S. History.

It's very much surprising that over 600,000 emails generated by 158 employees of the Enron Corporation and acquired by the Federal Energy Regulatory Commission during its investigation after the company's collapse. This is referred as the "Enron Corpus".

The goal of this project is to utilize the Enron corpus dataset to build a machine-learning algorithm to identify the employees who might have committed the fraud.

2. Dataset Overview

The Enron corpus dataset containing emails data about 150 top management employees available at https://www.cs.cmu.edu/~./enron/.

The dataset is divided into 150 named directories and one directory represents an employee. There are sub-directories for different stuffs like inbox, sent items, conversations etc.

In the final project, a summary dataset was provided containing 3 categories of data as POI Label, Financial data & Email data as pickle dump.

There are 146 data records and 22 features as mentioned below (not including attribute Name, Email Address, and POI). Among 146 people there are 18 people who were marked as POI.

FINANCIAL	**STOCK**	**EMAIL**
SALARY	EXERCISED_STOCK_OPTIONS	TO_MESSAGES
DEFERRAL_PAYMENTS	RESTRICTED_STOCK	FROM_THIS_PERSON_TO_POI
TOTAL_PAYMENTS	RESTRICTED_STOCK_DEFERRED	FROM_POI_TO_THIS_PERSON
BONUS	TOTAL_STOCK_VALUE	FROM_MESSAGES
SHARED_RECEIPT_WITH_POI		
EXPENSES		
LOAN_ADVANCES		
OTHER		
DIRECTOR_FEES		
DEFERRED_INCOME		
LONG_TERM_INCENTIVE		

3. Analysis

Question: 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 goal of this project is to build a prediction model using the financial & email summary features to identify the probable POIs.

Machine learning algorithms will learn the data features to train a model that will be able to classify whether someone belongs to POI class or not. This is not a single operation instead a series of steps involved to scale & identify the best features across different class of algorithms with respective parameters being tuned.

The dataset is a summary of financial & email statistics of a person including important details like his/her salary, payments, loans, stocks, email exchanges etc.

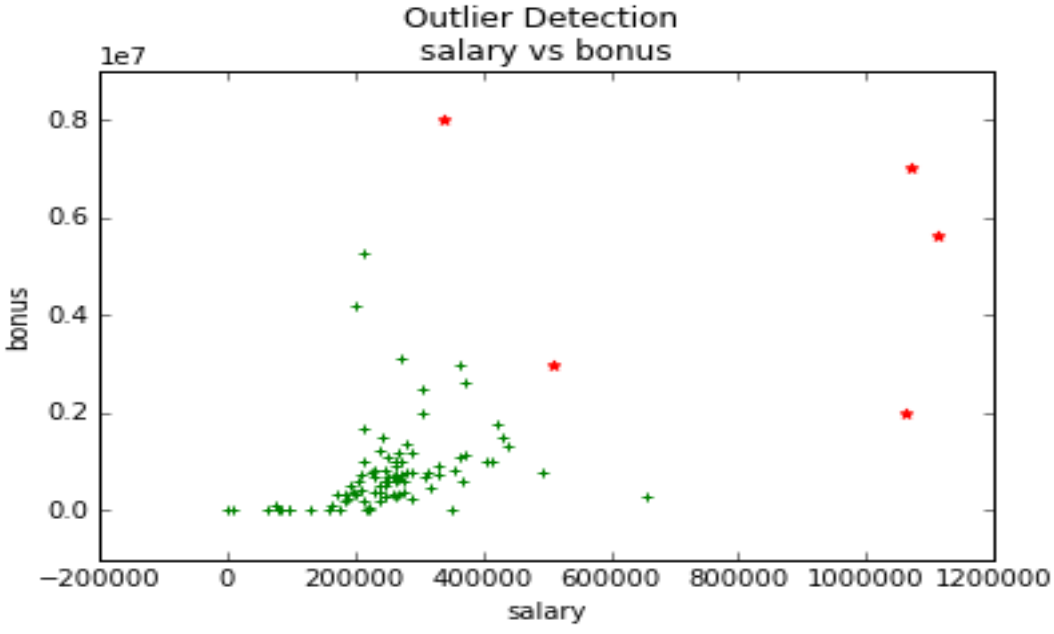
Feature relations & Outliers

There were 2 records named "TOTAL", "LOCKHART EUGENE E", and "THE TRAVEL AGENCY IN THE PARK" that were be marked as invalid. TOTAL is simply the sum of all financial features and "LOCKHART EUGENE E" does not have any meaningful information, as all the data items are NaNs.

I did some analysis on the relationships among the features.

Salary Vs. Bonus
The following people are identified as outliers:

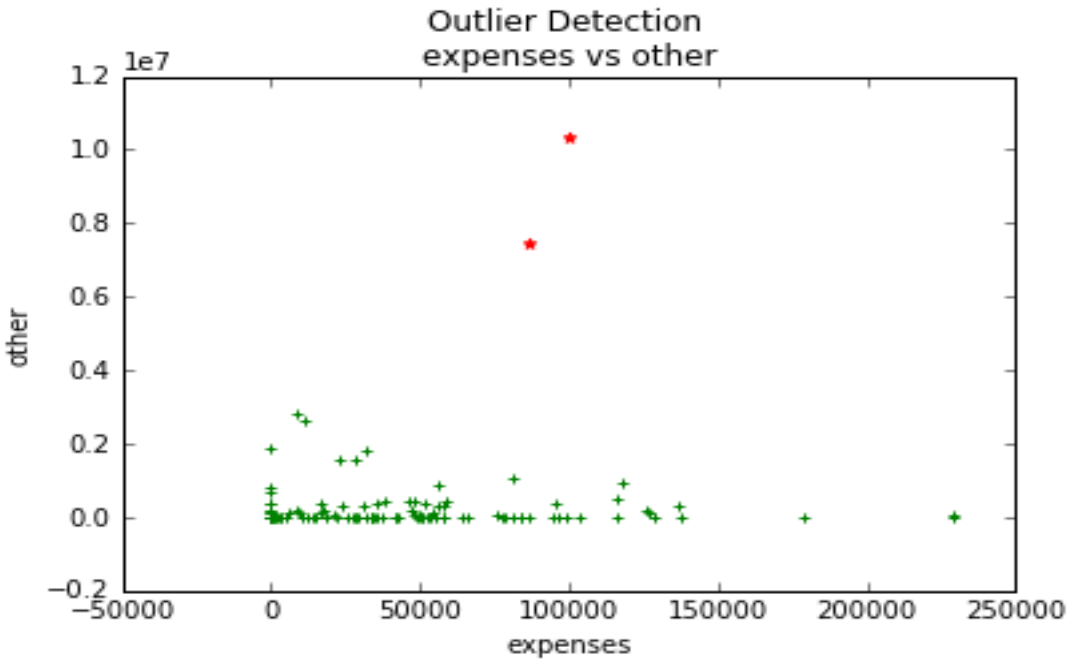
- LAVORATO JOHN J
- WHALLEY LAWRENCE G
- LAY KENNETH L
- SKILLING JEFFREY K
- FREVERT MARK A



Expenses vs. others

The following people are identified as outliers:

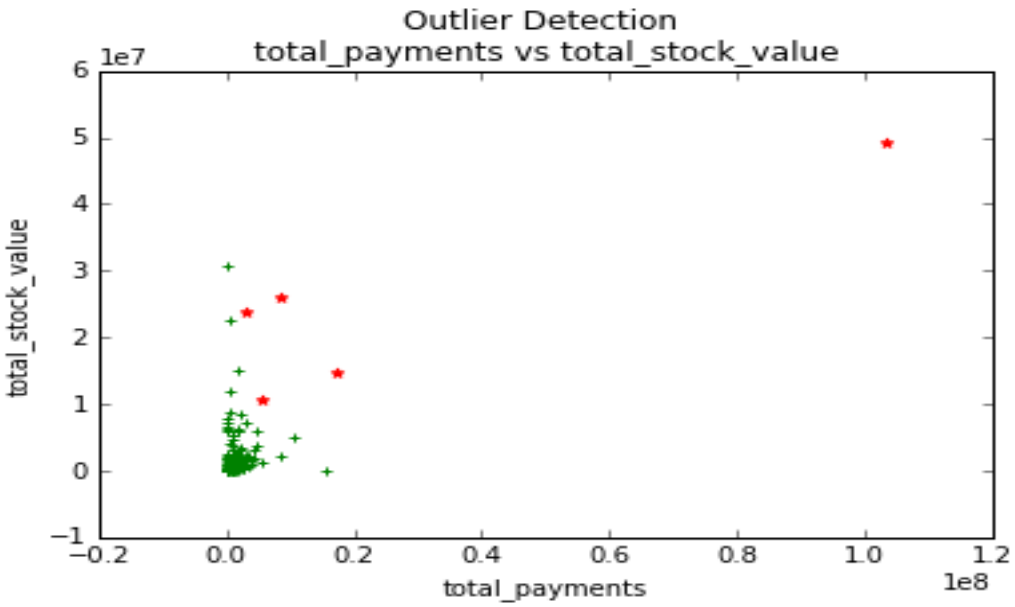
LAY KENNETH L
FREVERT MARK A



Total Payments vs. Total Stock value

The following people are identified as outliers:

BAXTER JOHN C
LAY KENNETH L
SKILLING JEFFREY K
FREVERT MARK A
PAI LOU L



There was one person common as outlier "LAY KENNETH L" and he happened to be the CEO and Chairman of Enron Corporation. I did not remove the outliers as they could have some different features from others and could be crucial to find out the POIs.

4. Feature Engineering

Question: 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"]

The dataset had 22 attributes for a person. Person's Name and email address cannot be considered a feature with importance. We removed these 2 features from the analysis. POI was the target/label. We were left with 19 features to work with.

New Features

I added 3 new features as mentioned below. I believe the below features helped the model to learn how total payment; stock to payment ratio and email involvement with POI helped it distinct between a POI and non-POI.

Again there might be a person who has a huge number of emails compared to someone has less emails, and chances are there that the email with poi will be higher for him. But that does not always imply the person is a suspected POI. The EMAIL_PCT_WITH_POI will help the model to understand how much of a person's emails involve a POI.

Feature Name	Calculation
COST_TO_COMPANY	Total Payment + Total Stock
STOCK_TO_PAYMENT_RATIO	Total Stock / Total Payment
EMAIL_PCT_WITH_POI	Email involving POI / Total Emails

I used the features in some combination and selected the best set of combination based on the overall results by the base models. I could see the results are almost same.

	Metric	Gaussian NB	SVM	Decision Tree	KNN
ALL Features including new ones	Score	0.4	0.9333	0.8	1
	Accuracy	0.5759		0.7992	0.8831
	Precision	0.2086		0.2336	0.7217
	Recall	0.781		0.222	0.201
Some Basic Features including new ones	Score	0.7333	0.9333	0.8666	1
	Accuracy	0.7148		0.804	0.884
	Precision	0.2519		0.2609	0.7329
	Recall	0.5785		0.2565	0.2045
Selected Features without new ones	Score	0.9333	0.9333	0.7333	1
	Accuracy	0.7947		0.7962	0.8854
	Precision	0.2734		0.2401	0.7473
	Recall	0.3255		0.244	0.213

The following features were used in the final model as I find them enough to generate best results. Rest of the features were not used in final model as those did not help to get a better result instead using all the features increased the run time exponentially.

BASIC FEATURES for final model	NEW FEATURES (Not Used in Final Model)
SALARY	COST_TO_COMPANY
BONUS	STOCK_TO_PAYMENT_RATIO
DIRECTOR_FEES	EMAIL_PCT_WITH_POI
OTHER	
TOTAL_PAYMENTS	
TOTAL_STOCK_VALUE	
EXERCISED_STOCK_OPTIONS	
RESTRICTED_STOCK	

The restricted stock deferred feature was not used as most of the cases it was cancelled by restricted stock feature. The exercised stock option and total stock value were included as restricted stock is sometimes 0. Below screenshot will show the relationships among the stock values.

email_address	str	1	phillip.allen@enron.com
email_pct_with_poi	float	1	0.3
exercised_stock_options	int	1	1729541
expenses	int	1	13868
from_messages	int	1	2195
from_poi_to_this_person	int	1	47
from_this_person_to_poi	int	1	65
loan_advances	int	1	0
long_term_incentive	int	1	304805
other	int	1	152
poi	bool	1	False
restricted_stock	int	1	126027
restricted_stock_deferred	int	1	-126027
salary	int	1	201955
shared_receipt_with_poi	int	1	1407
stock_to_payment_ratio	float	1	0.39
to_messages	int	1	2902
total_payments	int	1	4484442
total_stock_value	int	1	1729541

Feature Scaling

A pipeline was created that utilized "**MinMaxScalar**" for scaling the features within a range of 0-1. This scaling was required as PCA was used and it generally works very well when features are scaled properly.

To select best features the pipeline made use of "**SelectKBest**" to select k-best features. Different values of "k" were supplied to the pipeline to find the best value of K. As PCA was used for final model so **the K-best selection was not used in the final model**. PCA was used to reduce dimensionality and to generate the best set of components. The number of components was passed to the pipeline ranged from 3 – 8.

At the heart the pipeline contained a classification algorithm that used the components generated by PCA. I tried different set of algorithms and found K-Nearest Neighbor to provide the most optimal result.

All these pipeline options were passed to it as a set of estimators with respective parameter set for each estimator.

I used "**GridSearchCV**" to use the pipeline to find out the most optimal set of parameters for best result.

5. Model Building & Results

Question: 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 tried a number of algorithms and can categorize them into following.

- *Base Models – Naïve Bayes (Gaussian), SVC, KNN, Decision Tree etc.*
- *Advanced Models – Uses Ensemble models (Random Forest, ExtraTree, AdaBoost etc.)*
- *Pipeline models – GridSearchCV with proper pipelines*

There were some differences with base models and find the KNN to produce best score for a subset of base features as mentioned earlier. Then I tried some advanced models involving ensemble methods. Those models did not yield an optimal set of scores for the metrics like precision, recall, F-scores combined together.

Later I tried to build pipelines with feature scaling enabled, PCA for dimension reduction, and classifier with parameter sets to intelligently select the best set of combination using GridSearchCV model with Stratified Shuffle split cross validator with 1000 splits chosen at random. This improved the different metrics score and could achieve a decent result.

Question: 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"]

Each algorithm has its own set of parameters used to create decision boundaries. Any change in a parameter may produce different classification boundary and predictions.

Parameters can be adjusted so that the model can generate optimal results. Along with the prediction score there are other scores like precision & recall that need to be optimized. Thus a model can provide an optimal solution.

Sometimes the model can score high but other scores like recall, accuracy etc. may be very low. In those cases a high number of FALSE POSITIVES and FALSE NEGETIVES can be encountered. This is why we need to optimize a solution that will help achieving overall accuracy.

Base models were tested with a variety of parameters to understand how they were performing on the unseen data. It was found that the K-Nearest Neighbor to yield the best score, yet somewhat unwanted precision & recall values were encountered.

Similarly advanced models using ensemble methods were tested with a variety of different parameters but could not get expected results with them.

The summary of results provided below.

Base Model Results

Model	Score	Accuracy	Precision	Recall	F1	F2
Gaussian Naïve Bayes	0.9333	0.7148	0.2519	0.5785	0.351	0.4594
SVM	0.9333	NA	NA	NA	NA	NA
Decision Tree	0.8	0.8034	0.26	0.257	0.2584	0.2575
K Nearest Neighbor	1	0.884	0.7329	0.2045	0.3197	0.2389

Advanced Model Results

Model	Score	Accuracy	Precision	Recall	F1	F2
Random Forest	0.8666	0.8554	0.3669	0.1165	0.1768	0.1349
Extra Tree Classifier	0.8683	NA	NA	NA	NA	NA
Bagging Classifier	0.9333	0.8668	1	0.0015	0.003	0.0018
Ada Boost	0.866	0.8395	0.3056	0.16	0.21	0.1768

Tuning

Finally pipelines were created on the base classifiers to understand what are the features that will yield the most optimal results. Pipeline was created using estimators as **MaxMinScalar** for feature scaling, **SelectKBest** for feature selection, **PCA** for dimensionality reduction, and **KNearestNeighbor** as classifier (however not used in final model). Estimators were provided with a set of input parameters to find out the best parameter combination. It was found the KNN pipeline model produced the most optimal solution with decent score, accuracy, precision, and recall values.

GridSearchCV Model Results

Best Model	Score	Accuracy	Precision	Recall	F1	F2
Final KNN Model	1.0	0.83547	0.39126	0.42100	0.40559	0.41470

Best Estimator Parameters

Purpose	Name	Parameter Name	Parameter Value Range	Best Estimator Parameter Value
Feature Scaling	MaxMinScalar	NA	NA	feature_range=(0, 1)
Reduce Dimensionality	PCA	n_components	3 to 8	7
		whiten	TRUE, FALSE	FALSE
Classifier	KNeighborsClassifier	n_neighbors	1, 2, 3, 4, 5, 8, 10	1
		algorithm	kd_tree', 'ball_tree'	ball_tree
		leaf_size	1, 2, 3, 4, 5, 10	1

6. Validation

Question: 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 the process of verifying the output. For machine learning models the output is set of predictions and scores. As this is a supervised classification problem the test set can be prepared to run the prediction model and the predicted labels can be validated against the test set.

A classic mistake is to over fit the model that can produce excellent predictions on the training data but it fails to generate decent predictions on the unseen data. Over fitting were eliminated using cross-validation techniques.

StratifiedShuffleSplit cross-validation was used to validate the results. By default, ***GridSearchCV*** does a 3-Fold cross validation. As the target label is binary class the final model used a ***StratifiedShuffleSplit*** validator.

In case of ***StratifiedShuffleSplit*** the data is randomly shuffled and split every time. In final model the data is split 1000 times. Different test sets can have overlapping elements. This makes the data splits more balanced. Then classifier runs on each split, and finally aggregates results to produce the final set of predictions.

Due to the class imbalance problem, it is preferred to use a stratified shuffle split instead. This ensures that an equal ratio of POIs to non-POIs is found in the training and test sets.

Question: 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"]

There were a number of evaluation metrics that were considered for performance by the model. One of the goals of this project was to get precision & recall scores more than 0.3. A test classifier was provided to test the classifier for calculating all these metrics. The test classifier module follows the similar approach of performing a K-Fold (K is 1000) Stratified Shuffle splits for cross validation.

Using the KNN classifier the final **GridSearchCV** model was able to achieve the below scores.

Best Score	Accuracy	Precision	Recall	F1	F2
1.0	0.83547	0.39126	0.42100	0.40559	0.41470

Best Score – Mean cross-validated score of the best estimator
Accuracy – How much of predicted results are correct
Precision – Positive predictive value, i.e., how much of positive predictions are correct.

Precision = True Positives / (True Positives + False Positives)

Recall - percent of true positives belong to the positive class

Recall = True Positives / (True Positives + False Negatives)

F Scores – Harmonic mean of precision and recall

The general formula for positive real β is:

$$F_\beta = (1+\beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}.$$

The precision score tells us that around 60% non-POIs were marked as POIs. From the recall score we can say around 40% of actual POIs were identified correctly, and rest 60% of POIs were marked as non-POI.

7. Project Files

The below table list all the input & source code files with a short description.

File Name	Directory	Description
final_project_dataset.pkl	final_project/	This file contains all the 146 top employees data.
poi_id.py	final_project/	The main project file that runs all the steps.
tester.py	final_project/	Provided by Udacity team and used to perform cross-validation on the classifiers for scores.
data_operations.py	final_project/utils/	This python script does data cleaning activity and adds new features to the dataset.
poi_outliers.py	final_project/utils/	This python script does the outlier identification and plots them in RED.
basic_models.py	final_project/utils/	This python script contains all base models tested as part of this project.
advanced_models.py	final_project/utils/	This python script contains all the advanced models build using ensemble methods.
pipeline_models.py	final_project/utils/	This python script contains all the intelligent models that used RandomizedSearchCV or GridSearchCV to find out best set of features, parameter values for a classifier.

8. Conclusion

The dataset might look smaller but it was a complex and interesting one. The dataset could add few more features like the employees association years with Enron, location, etc. I really enjoyed working with it, as I was able to apply my understanding in much detail and faced a number of challenges that really helped me to learn a number of things.