Enron Machine Learning Project:

1. **Summary:**

The Enron Corpus is a large database of 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. The goal of this project is to identify Person of Interest (POIs) in the Enron scandal based on email and financial data. We have 146 persons who are potential POIS each including 21 features.

- Total number of data points: 146
- Number of POIs in Dataset: 18
- Number of non_POIs in Dataset: 128
- Number of features used: 21

Something problematic with the above dataset is that we have 146 observations but only 18 identified as outliers which might affect our model training causing the model to get high number of false positives which will cause us to label POIs as non-POIs.

Outliers removed:

First we start with detecting the null values and we will drop the email_address column as it gives no value for our investigation and analysis.

Here are the columns with the missing values:

```
salary                     95 non-null float64
to_messages                86 non-null float64
deferral_payments          39 non-null float64
total_payments             125 non-null float64
exercised_stock_options    102 non-null float64
bonus                      82 non-null float64
restricted_stock           110 non-null float64
shared_receipt_with_poi    86 non-null float64
restricted_stock_deferred  18 non-null float64
total_stock_value          126 non-null float64
expenses                   95 non-null float64
loan_advances              4 non-null float64
from_messages              86 non-null float64
other                      93 non-null float64
from_this_person_to_poi    86 non-null float64
poi                        146 non-null bool
director_fees              17 non-null float64
deferred_income            49 non-null float64
long_term_incentive        66 non-null float64
from_poi_to_this_person    86 non-null float64
```

We can see that there are quite lots of nulls in columns like deferral_payments or laon_advances.

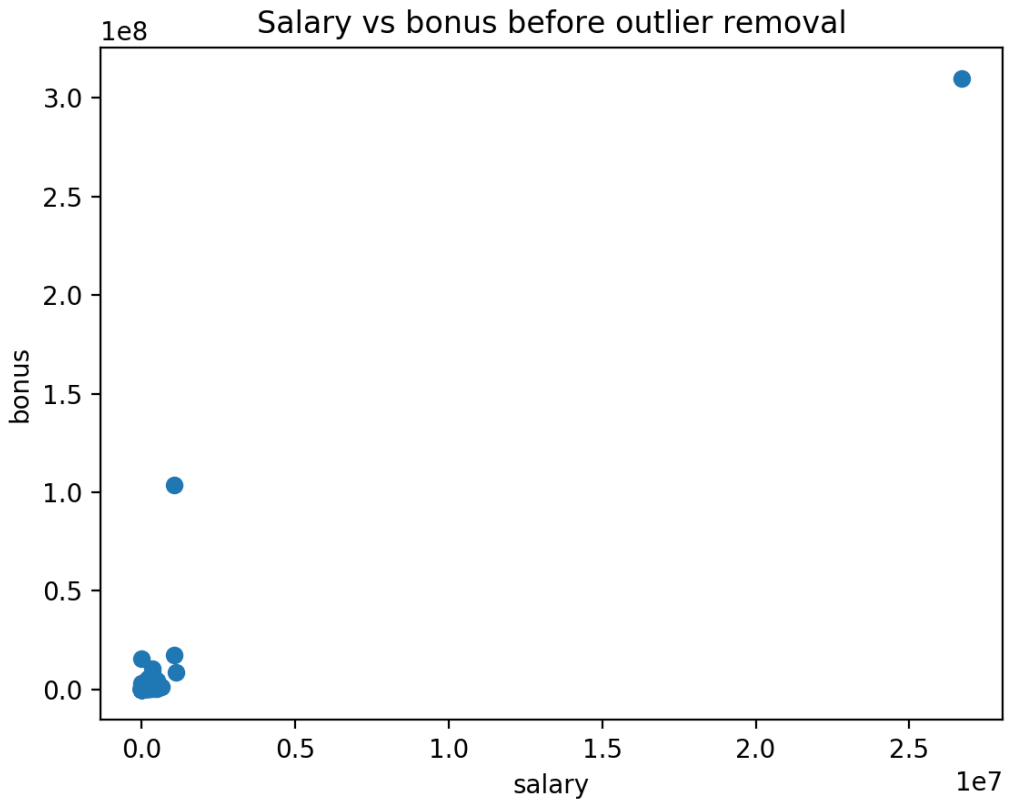
	salary	to_messages	deferral_payments	total_payments
count	95	86	39	125
mean	562194.2947	2073.860465	1642674.154	5081526.488
std	2716369.155	2582.700981	5161929.974	29061716.4
min	477	57	-102500	148
25%	211816	541.25	81573	394475
50%	259996	1211	227449	1101393
75%	312117	2634.75	1002671.5	2093263
max	26704229	15149	32083396	309886585
	exercised_stock_options	**bonus**	**restricted_stock**	**shared_receipt_with_poi**
count	102	82	110	86
mean	5987053.775	2374234.61	2321741.136	1176.465116
std	31062006.57	10713327.97	12518278.18	1178.317641
min	3285	70000	-2604490	2
25%	527886.25	431250	254018	249.75
50%	1310813.5	769375	451740	740.5
75%	2547724	1200000	1002369.75	1888.25
max	311764000	97343619	130322299	5521

	restricted_stock_deferred	total_stock_value	expenses	loan_advances
count	18	126	95	4
mean	166410.5556	6773957.452	108728.9158	41962500
std	4201494.315	38957772.73	533534.8141	47083208.7
min	-7576788	-44093	148	400000
25%	-389621.75	494510.25	22614	1600000
50%	-146975	1102872.5	46950	41762500
75%	-75009.75	2949846.75	79952.5	82125000
max	15456290	434509511	5235198	83925000
	from_messages	**other**	**from_this_person_to_poi**	**director_fees**
count	86	93	86	17
mean	608.7906977	919064.9677	41.23255814	166804.8824
std	1841.033949	4589252.908	100.0731114	319891.4097
min	12	2	0	3285
25%	22.75	1215	1	98784
50%	41	52382	8	108579
75%	145.5	362096	24.75	113784
max	14368	42667589	609	1398517
	deferred_income	**long_term_incentive**	**from_poi_to_this_person**	
count	49	66	86	
mean	-1140475.143	1470361.455	64.89534884	
std	4025406.379	5942759.316	86.97924419	
min	-27992891	69223	0	
25%	-694862	281250	10	
50%	-159792	442035	35	
75%	-38346	938672	72.25	
max	-833	48521928	528	

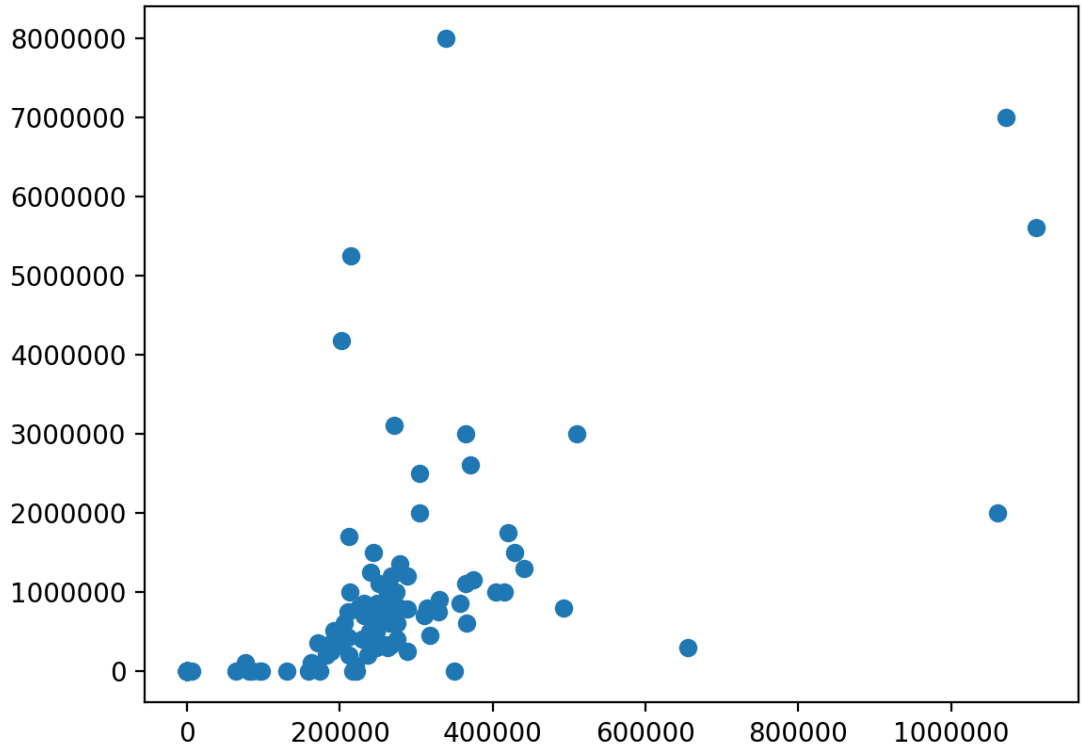
Looking at the description table of the features, we can see that there are some negative values in the minimum for restricted stocks and we can also see that there is someone with a total payment of 309 million dollars and even max number of email of 15,000 emails or a salary of almost 27 million.

Lets start cleaning the data set by replacing the null values, because if we drop the null values, we will end up with a very small dataset because this is a small dataset and to have a well trained model, the amount of data is has to be relatively high I will avoid to drop lots of the outliers mentioned above especially those who are POIs as they are very important for our training.

Now for checking for outliers, we can start with outliers by simply eyeballing them through visualizations. After plotting the salary and bonuses of every person we have the below plot which shows a huge outlier that surpasses everyone, after further investigation, we know it's the total row:



After removing the TOTAL row we can see the below scatter plot:



I also noticed a weird name which was definitely not a person under the name "THE TRAVEL AGENCY IN THE PARK" which was also removed.

Still looking at the scatter plot above, we can see outliers whom after investigation, for that we will use Tukey's IQR to detect who these outliers are for all the features and not just the salary and the bonus.

The top 10 outliers that are above or below the IQR are:

LAY KENNETH L	15
FREVERT MARK A	12
BELDEN TIMOTHY N	10
LAVORATO JOHN J	8
BAXTER JOHN C	8
SKILLING JEFFREY K	8
KEAN STEVEN J	7
WHALLEY LAWRENCE G	7
HAEDICKE MARK E	7
BUY RICHARD B	6

As mentioned earlier the number of persons that are already identified as POIs are quite low for us to remove any if they are outliers.

I will remove these from the top 5 and keep the rest:

FREVERT MARK A
LAVORATO JOHN J
BAXTER JOHN C

I didn't remove these as maybe they are the top POIs in the entire dataset:

LAY KENNETH L: Founder, CEO and chairman of Enron and also the CEO of Enron after Skilling resignation when the Enron stock price plummeted from 90$ in 2000 to 20$ in October 2001 and then bankruptcy in December of the dame year. Died in 2006 3 moths before his conviction.

SKILLING JEFFREY K: Worked for years at Enron as CEO of several of its subsidiaries, then on February 12, 2001, Skilling was named CEO of Enron, receiving $132 million during a single year he then resigned later that year. In 2006, he was convicted of federal felony charges relating to Enron's collapse and is currently serving 14 years of a 24-year, four-month prison sentence at the Federal Prison Camp (FPC) – Montgomery in Montgomery, Alabama.

2. Features Engineering:

Prior doing any feature reduction adding nay features that will hopefully strengthen our algorithm.
Since this is a labeled dataset, I selected four algorithms to test with,
Naïve Bayes,
Decision Tree
Random Forest
Logisitics Regression

The result we wanted was a non-ordered discrete output (POI/non-POI) thus Naïve Bayes was a good start, and I wanted to use another popular supervised algorithm so I went with the decision tree and an ensemble algorithm I chose the Random Forest.

List of initial selected features without removing any feature except the email addresses:

Classifier (23 features)	Accuracy	Precision	Recall	F1
Naïve Bayes	0.75	0.25	0.36	0.29
Decision Tree	0.83	0.38	0.29	0.333
Random Forest	0.85	0.52	0.18	0.26
Logisitics Regression	0.78	0.21	0.18	0.19

We can see that the best is the decision tree which almost passes the 0.3 threshold, next will be the Naïve Bayes but the rest are not very promising and not balanced.

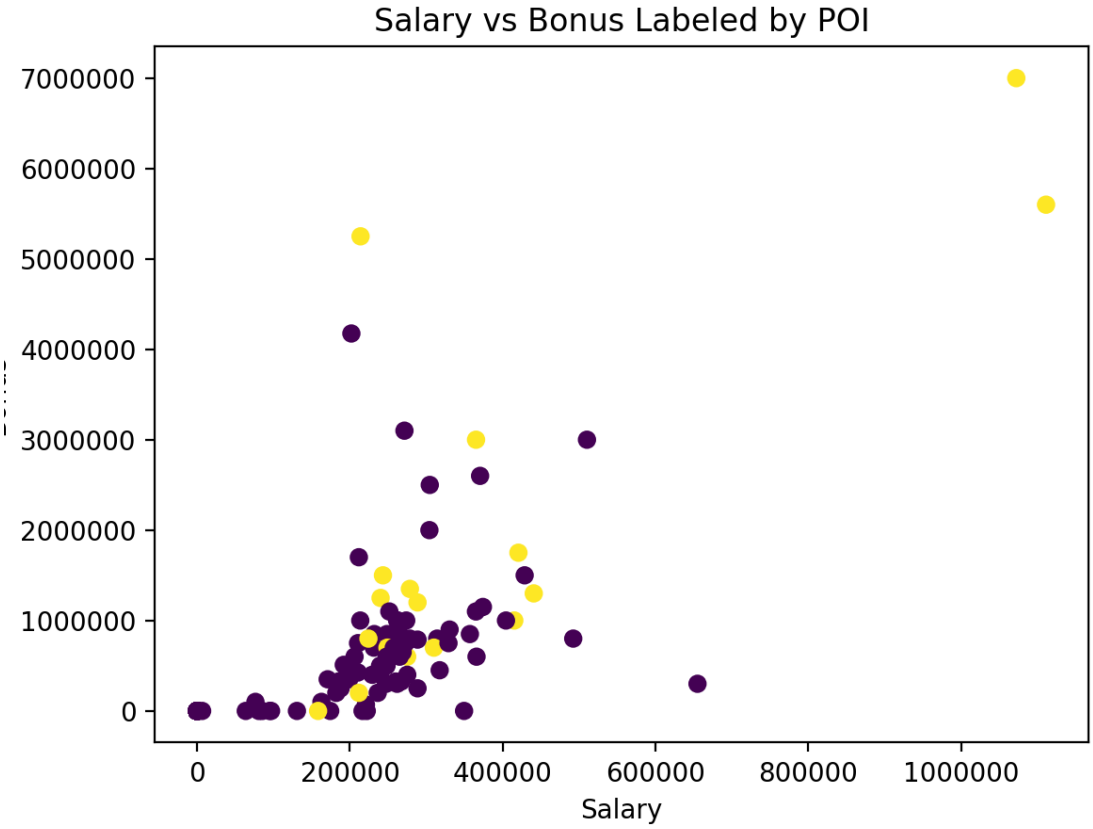
Now its time to create some new features that might assist in improving the algorithms. I have created the below:

List of new features added:

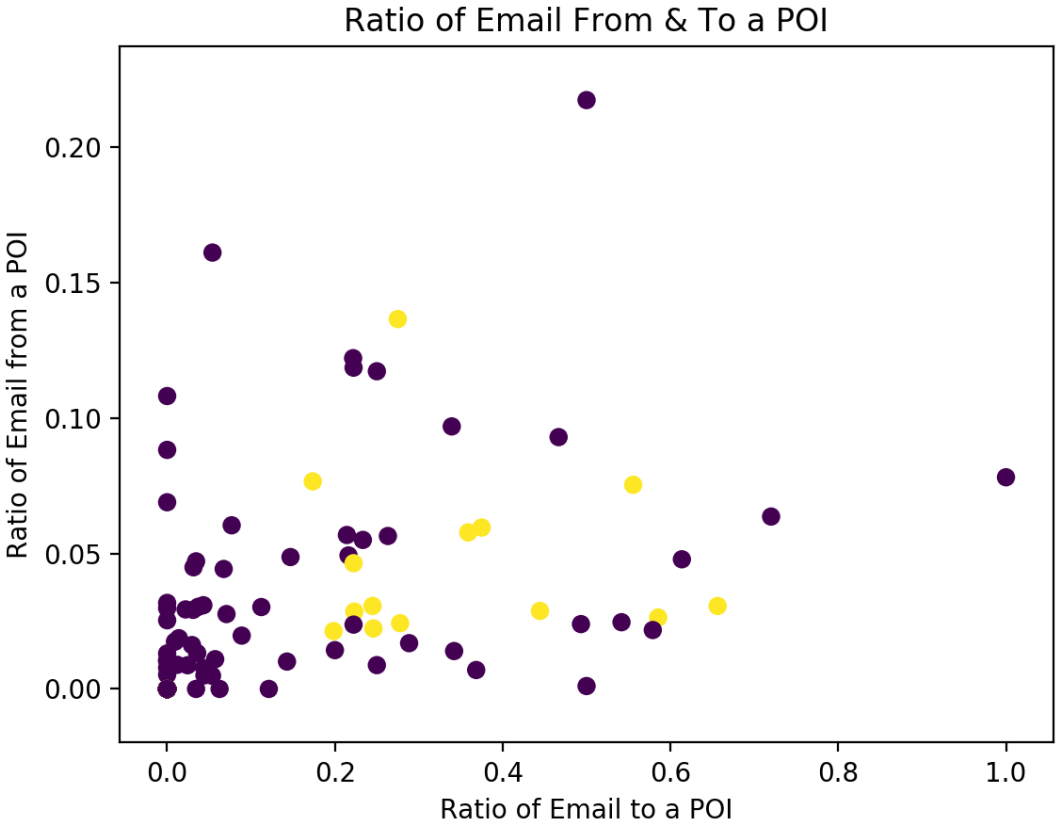
1. **Ratio of email to POI from the total number of messages**: It makes sense that if someone sends a high ratio of his emails to a POI will probably be a POI.
2. **Ratio of email from POI from the total number of messages**: This makes sense because if most of my emails are from a POI than probably I am a POI.
3. **Ratio of shared email with a POI**: If lots of a person email are CCed with a POI there is a probability he is a POI but also we know that this is not fair because if a person is a POI secretary then probably she is always shared in his emails but innocent of any wrong doing as was the case with Mrs. Paula H. Rieker.
4. **Ratio of bonus from salary**: This is useful because if the bonuses are huge and bigger than the salary this will flag a POI whom is getting illegal money in the shape of bonuses.
5. **Ratio of bonus from the total payments**: Similar to the above feature which indicates that the bigger the bonus from total payment the more this person is prone to be a POI.

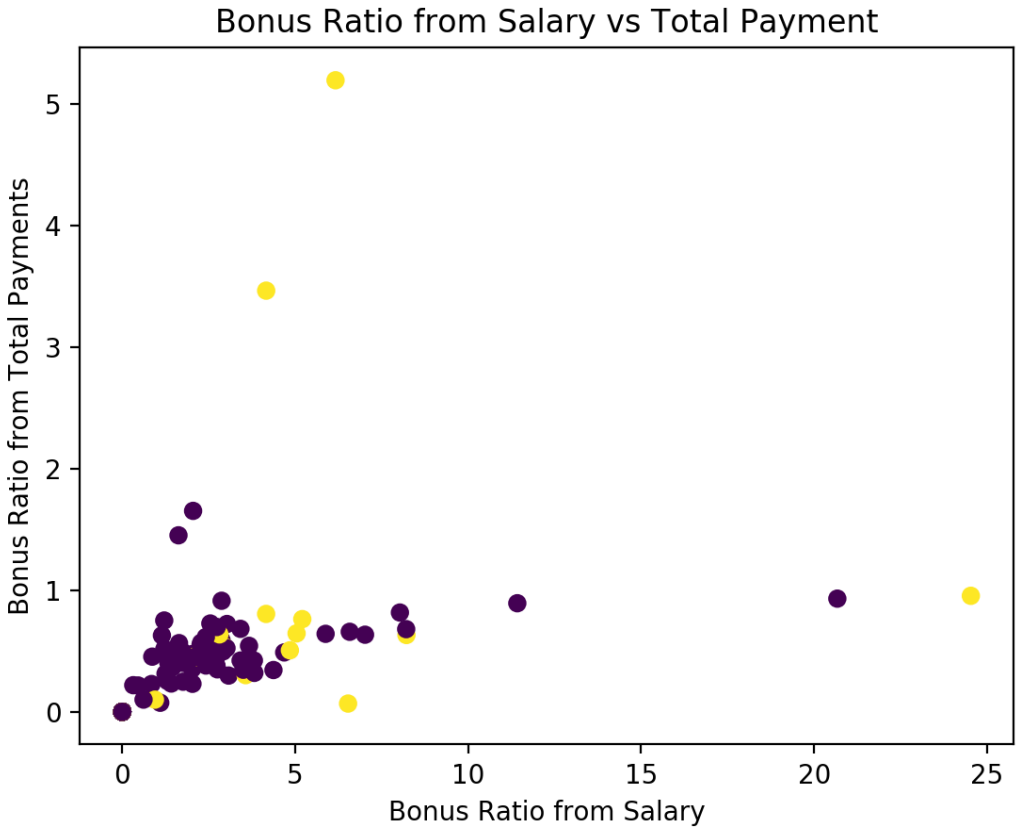
Visualizing some features:

Here is a plot some features labeled by POI (Yellow dots):



This plots tells a lot about POIs as we can see the 2 outliers on the right with high salaries and bonuses and even the point with the salary and bonus abnormally allocated.


Ratio of Email From & To a POI
Ratio of Email from a POI
0.00
0.05
0.10
0.15
0.20
0.0
0.2
0.4
0.6
0.8
1.0
Ratio of Email to a POI


Bonus Ratio from Salary vs Total Payment
Bonus Ratio from Total Payments
0
1
2
3
4
5
0
5
10
15
20
25
Bonus Ratio from Salary

Of the seven new created features above I will use four new features related to both emails and financial data:

- **Ratio of email from POI from the total number of messages**
- **Ratio of email to POI from the total number of messages**
- **Ratio of bonus from salary**
- **Ratio of bonus from the total payments**

Retest models with the 4 newly added features prior feature selection:

Classifier (24)	Accuracy	Precision	Recall	F1
Naïve Bayes	0.75	0.25	0.36	0.29
Decision Tree	0.82	0.36	0.3	0.33
Random Forest	0.86	0.57	0.18	0.28
Logisitics Regression	0.78	0.21	0.18	0.19

We can see little to no change in every of the algorithms retested with the new features, for that we need to do dome feature selection to only keep feature of value to the model.

Feature selection:
Before digging into the feature selection process I want to discuss that I didn't find it useful to do any scaling on the features because I am neither using SVM not K-Means clustering.

When using 100% of the features the f1 score was 0.33 for the decision tree, and in order to best estimate the best number of usable features, I will use the univariate feature selection, which treats each feature independently and asks how much power it gives you in classifying your data.

I will use SelectKBest from the Sklearn library because SelectKBest selects the K features that are most powerful then running SelectKBest with the GridSearchCV which will automatically iterate through the number of features to use the exact amount that will yield the highest score.

After pipeline with GridSearchCV with the best estimator, we derived the best number of features with the ultimate f1 score of 0.4 was achieved at k=13, here are the features with their scores:

salary score: 23.822124
bonus score: 33.226341
to_poi_ratio score: 16.469245
ratio_bonus_salary score: 16.323658
ratio_bonus_payments score: 20.507914
total_payments score: 9.520323
exercised_stock_options score: 26.584555
restricted_stock score: 10.131047
shared_receipt_with_poi score: 10.480691
total_stock_value score: 26.325073
deferred_income score: 17.145561
long_term_incentive score: 12.385712
from_poi_to_this_person score: 10.617155

After selecting the features I thought to apply PCA but the model was running quite fast and I believe the only way to better enhance its performance was to adjust the parameters of the decision tree classifier.

Here is the model before any Decision Tree parameter tuning:

Classifier	Accuracy	Precision	Recall	F1
Decision Tree (Before Feature Selection)	0.82	0.36	0.3	0.33
Decision Tree (After Feature Selection)	0.83	0.43	0.37	0.4

3. **Algorithm:**

Tuning the parameters of algorithm is simply the process of changing, testing and updating the parameters in order to get the right mix or settings where once completed, the parameters are optimized to produce the best results. After running the decision tree with the best number of features, and in hopes to achieve higher precision and recall and since the Decision Tree Classifier has around 13 parameters to be adjusted for achieving a more

accurate model, I will tune the parameters accordingly. I will use the param_grid in GridSearchCV to automate the process, here is the list of the parameters that will be adjusted:

- Min_Sample_Leaf
- Min_sample_Split
- Max_Depth
- criterion

GridSearch will be directed by cross-validation with 10 splits of the data, and the scoring criteria is specified as F1.

Here are the best parameters as per the GridSearchCV:

classify__min_samples_split': 2,
'classify__min_samples_leaf': 1,
'classify__criterion': 'gini',
'classify__max_depth': 4

These are the parameters I adjusted for the best result of my classifier:

Algorithm	Min_Sample_leaf	Min_Sample_Split	Max_Depth	Criterion	K
Decision Tree Classifier	1	2	4	gini	13

4. **Validation & Evaluation**:

Validation is the process of determining if the algorithm fits well outside data. For that we split the data into test and training data where we train our model on the training data and once we feel it is fit, we can test it on the test data.

A classical mistake will be to overfit the data on the training data set, that is why validating on a totally new dataset that is new to the model will help avoid such a mistake.

I used Cross validation train_test_split function to split the Enron data between training and testing with test size of 40% of the dataset (56 data points). After running the Decision Tree Classifier with the paramters tuning using GridSearcCV on the test data set it yielded the below evaluation performance:

Accuracy_score: 0.92
Precision: 0.57
Recall: 0.8
f1_score: 0.67

After running the above classifier on the real data in tester.py, it yielded results more than 0.3 on the evaluation metrics of recall and precision.

Classifier	Recall	Precision
Decision Tree Classifier	0.35	0.48

We can see that the metrics dropped especially the recall and I believe this is normal because with the small data set models tend to be overfitting that is why we see almost perfect (92%) accuracy.