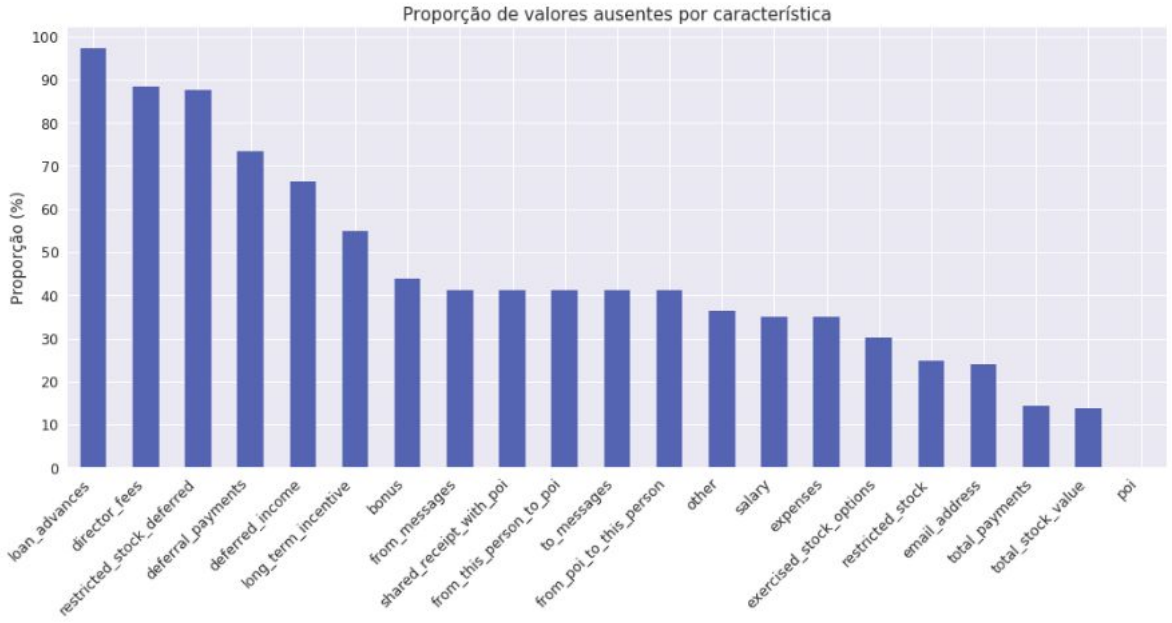
Perguntas de resposta livre da Submissão da Enron

1. Resuma para nós o objetivo deste projeto e como o aprendizado de máquina é útil na tentativa de realizá-lo. Como parte de sua resposta, dê um panorama sobre o conjunto de dados e como ele pode ser usado para responder à pergunta do projeto. Houve algum valor discrepante nos dados quando você os obteve e como você lidou com eles?[rubricas relevantes: “exploração de dados”, “investigação atípica”]

O objetivo deste projeto é, com base em um conjunto de informações financeiras e de e-mails de funcionários e acionistas, criar um algoritmo de aprendizado de máquina capaz de identificar pessoas que provavelmente tiveram envolvimento no escândalo de corrupção ocorrido na Enron. Os algoritmos de aprendizado de máquina podem ser de grande ajuda na busca de padrões que são de difícil identificação.
O conjunto de dados contém 145 registros e 20 características. Algumas pessoas já foram identificadas como participantes do escândalo. Para estas pessoas, a característica chamada POI (Person of interest) está preenchida com o valor 1 que indica “sim”. Para as demais pessoas, o conteúdo está com 0 que indica “não”.
O conjunto de dados contém informações financeiras como salário, bônus, despesas, valor em ações - dos funcionários, de membros da diretoria e de acionistas. Possui também informações sobre a quantidade de e-mails enviados, recebidos de pessoas que foram identificadas como POI.
A ideia principal é usar os algoritmos de aprendizado de máquina e treiná-lo para identificar os padrões das pessoas envolvidas no escândalo (POI = 1) onde um modelo será criado e testado. Ao final iremos usar métodos de avaliação para sabermos como o modelo está se saindo.
A quantidade de POIs na base de dados é 18 e a quantidade de não POIs é de 127.
Ao realizar uma análise dos valores, foi identificado um desvio muito acima do normal para o salário e bônus. Investigando estes valores, foi constatado que como este conjunto de dados é uma cópia de um relatório gerado na investigação, ele também contém uma linha que é a somatória dos valores, ou seja, a linha do “**TOTAL**”. Para prosseguir com os trabalhos, foi realizada a exclusão desta linha. Foi encontrado também o funcionário “**LOCKHART EUGENE E**” sem qualquer informação preenchida. Este também foi considerado como um outlier e excluído do conjunto de dados.

Foram encontrados diversos valores ausentes. Como podemos ver no gráfico abaixo, os dados com maior número de informações ausentes realmente são dados específicos para poucas pessoas, como empréstimos, valores em dinheiro ou ações para membros do conselho não funcionários, etc. No processo de separação dos dados entre features e rótulos para posterior utilização pelos algoritmos de machine learning, os valores ausentes foram preenchidos com zero.

Gráfico com a proporção de características com valores nulos:



2. Quais recursos você usou no seu identificador de POI e que processo de seleção você usou para selecioná-los? Você teve que fazer algum escalonamento? Por que ou por que não? Como parte da tarefa, você deve tentar projetar seu próprio recurso que não vem pronto no conjunto de dados - explique qual recurso você tentou fazer e a lógica por trás dele. (Você não precisa necessariamente usá-lo na análise final, apenas o engenheire e teste.) Em sua etapa de seleção de recursos, se você usou um algoritmo como uma árvore de decisão, forneça também as importâncias de recursos dos recursos que você usa, e se você usou uma função de seleção de recursos automatizada como o SelectKBest, por favor, informe as pontuações dos recursos e as razões para a sua escolha de valores de parâmetros. [rubricas relevantes: “criar novos recursos”, “selecionar recursos de maneira inteligente”, “dimensionar recursos corretamente”]

Foi realizado uma análise utilizando os seguintes métodos de seleção de características:

- Importância das características
- RFECV (Recursive Feature Elimination with Cross-Validation)
- Seleção Univaridada (`SelectKBest`)

Para realização da seleção univariada, foi necessário a utilização de escalonamento de recursos através da biblioteca MinMaxScaler, pois os cálculos estatísticos realizados não aceitam valores negativos. Outra biblioteca de pré-processamento que foi utilizada foi a StandardScaler, a utilização de uma biblioteca ou outra ficou a cargo da biblioteca `GridSearchCV`. Falarei mais a respeito logo abaixo.

Foram adicionados seis novos recursos/características:

1. total_general: soma total de todos os valores da pessoa (total_payments + total_stock_value).
2. perc_from_poi: percentual de mensagens recebidas de um POI em relação a quantidade total

3. perc_shared_poi: : percentual de mensagens em compartilhadas com um POI em relação a quantidade total
4. perc_to_poi: percentual de mensagens enviadas a um POI em relação a quantidade total
5. perc_total_payments: percentual do total de pagamentos em relação à soma total
6. perc_total_stock_value: percentual de valores com ações em relação à soma total

A criação das novas características tiveram um impacto bastante positivo, veja abaixo que das 10 principais features, 4 são novas features criadas (tanto para o método estatístico CHI2 quanto para o ANOVA) :

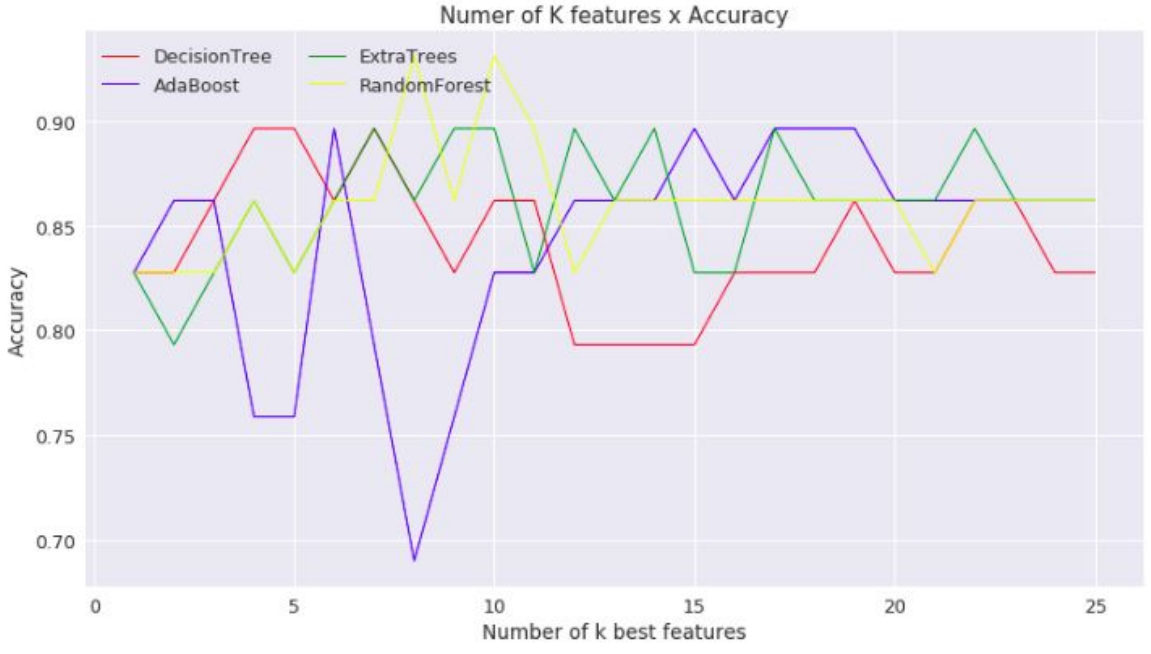
Feature Selection - CHI

	Features	CHI
9	loan_advances	6.95
0	bonus	6.08
22	perc_to_poi	4.44
4	exercised_stock_options	3.91
21	perc_shared_poi	3.65
17	total_stock_value	3.58
24	perc_total_stock_value	3.58
23	perc_total_payments	3.54
18	total_payments	3.54
19	total_general	3.38

Feature Selection - ANOVA

	Features	ANOVA
0	bonus	23.93
14	salary	19.07
22	perc_to_poi	15.82
17	total_stock_value	15.44
24	perc_total_stock_value	15.44
4	exercised_stock_options	14.87
19	total_general	13.58
2	deferred_income	12.77
10	long_term_incentive	12.64
15	shared_receipt_with_poi	11.63

Gráfico da acurácia com o respectivo número de features utilizando o méteodo `SelectKBest`:



Alguns algoritimos de aprendizado de máquina tiveram acurácia semelhante, como Extratrees e RandonForest, no entanto, alguns algoritmos tiveram grande divergência, como AdaBoost e DecisionTree.
Ao final, escolhi utilizar o método de Seleção Univariada (`SelectKBest`) com escolha automática de melhor número de características (k) através da biblioteca `GridSearchCV` que é uma biblioteca para realização de uma busca exaustiva identificando o melhor parâmetro.

3. Qual algoritmo você acabou usando? Que outro (s) você tentou? Como o desempenho do modelo diferiu entre os algoritmos? [rubrica de rubrica relevante: “escolha um algoritmo”]

Utilizei os algoritmos DecisionTree, Naive Bayes (Gaussian), ExtraTrees, LogisticRegression, AdaBoost e RandonForest.

Segue abaixo a tabela com as pontuações finais dos algoritmos. As linhas destacadas em azul são os algoritmos com as melhores pontuações - precision e recall >= 0,3 conforme exigência do projeto):

Modelo	Sequência	Acurácia	Precision	Recall	F1
AdaBoost	1	0.85	0.40	0.30	0.34
AdaBoost	2	0.85	0.44	0.36	0.40
AdaBoost	3	0.86	0.44	0.26	0.33
DecisionTree	1	0.80	0.23	0.21	0.22
DecisionTree	2	0.81	0.26	0.25	0.26
DecisionTree	3	0.79	0.36	0.76	0.49
ExtraTrees	1	0.87	0.52	0.19	0.28
ExtraTrees	2	0.86	0.44	0.15	0.22
ExtraTrees	3	0.78	0.32	0.57	0.41
GaussianNaiveBayes	1	0.75	0.24	0.40	0.30
GaussianNaiveBayes	2	0.80	0.28	0.34	0.31
GaussianNaiveBayes	3	0.84	0.38	0.27	0.32
LogisticRegression	1	0.78	0.18	0.17	0.17
LogisticRegression	2	0.79	0.19	0.18	0.19
LogisticRegression	3	0.74	0.24	0.43	0.31
RandomForest	1	0.86	0.38	0.11	0.18
RandomForest	2	0.86	0.40	0.13	0.20
RandomForest	3	0.85	0.36	0.16	0.22

A execução dos algoritmos foi realizada em três etapas identificadas acima na coluna “Sequência”. Na Sequência 1, a execução foi realizada com os dados após a limpeza. Na Sequência 2, foram adicionados novas características. Na Sequência 3, realizamos a otimização de recursos.

Para alguns algoritmos, a acurácia pouco mudou ao executarmos as sequências descritas acima. Um exemplo são os algoritmo RandomForest, AdaBoost e Decision Tree. No entanto, o foco não estava na acurácia e sim em outras métricas de avaliação. Para estas

métricas, houve uma mudança significativa conforme realizarmos a execução através das sequências. Iremos ver detalhes nas próximas questões.

4. O que significa ajustar os parâmetros de um algoritmo e o que pode acontecer se você não fizer isso bem? Como você ajustou os parâmetros do seu algoritmo particular? Quais parâmetros você ajustou? (Alguns algoritmos não possuem parâmetros que você precisa ajustar - se este for o caso daquele que você escolheu, identifique e explique brevemente como você teria feito isso para o modelo que não foi sua escolha final ou um modelo diferente que faz utilizar o ajuste de parâmetros, por exemplo, um classificador da árvore de decisão). [rubricas relevantes: “discutir o ajuste dos parâmetros”, “ajustar o algoritmo”]

Cada conjunto de dados contém realidades distintas, por isso os algoritmos permitem a realização de ajustes através de parâmetros afim de melhorar o resultado final do modelo.
Se não for feito de maneira adequada, pode piorar os resultados ou até mesmo a performance de execução do algoritmo.

Para realizar o ajuste dos algoritmos, também utilizei a biblioteca `GridSearchCV`, conforme dito anteriormente é uma biblioteca para realização de uma busca exaustiva identificando o melhor parâmetro. Verifiquei na documentação dos algoritmos, e informei alguns valores possíveis como parâmetro, e a partir daí o próprio `GridSearchCV` realiza a identificação e utilização do melhor parâmetro.

Segue os melhores parâmetros e quantidades de features identificadas para o algoritmo DecisionTree:

```
grid_search.best_params_:
{'classifier__class_weight': 'balanced', 'classifier__criterion': 'gini',
'classifier__max_depth': 3, 'classifier__min_samples_leaf': 16,
'classifier__min_samples_split': 3, 'classifier__splitter': 'random', 'scaler':
StandardScaler(copy=True, with_mean=True, with_std=True), 'selector__k': 18}
```

5. O que é validação e qual é um erro clássico que você pode cometer se errar? Como você validou sua análise? [rubricas relevantes: “discutir validação”, “estratégia de validação”]

A validação é o processo realizado para confirmar se o algoritmo realmente está fazendo o que você quer que ele faça. Existem várias formas, mas basicamente o conjunto de dados é dividido em duas partes, uma parte é utilizada para treinar e outra para testar o algoritmo e sabermos o quanto o algoritmo está acertando.

Algo que pode ser muito comum de ocorrer é obter um alto resultado da acurácia e achar que está tudo ok (também chamado de overfitting). No entanto, devemos tomar cuidado e certificar que as classes estão balanceadas, caso contrário, é possível que a classe minoritária seja classificada incorretamente.

Como temos poucos dados para a classe 1 - POI, o algoritmo não será tão bem treinado. Para melhorar o funcionamento dos algoritmos e evitar problemas, utilizei a função `StratifiedShuffleSplit` para criar uma validação cruzada estratificada com parâmetro `cv = 10`. Este parâmetro é responsável pela divisão do conjunto de dados em **n** partes fazendo com que o nosso conjunto seja dividindo visando preservar a porcentagem de amostras

para cada classe, evitando o possíveis problemas com desbalanceamento das classes. Neste projeto realizei o treinamento do modelo com 80% dos dados, no entanto, para a realização dos testes, utilizarei o programa tester.py disponibilizado pela Udacity, que será o programa utilizado para avaliação final deste projeto.

6. Dê pelo menos duas métricas de avaliação e seu desempenho médio para cada uma delas. Explique uma interpretação de suas métricas que diz algo que seja compreensível ao homem sobre o desempenho do seu algoritmo. [rubrica de rubrica relevante: "uso de métricas de avaliação"]

Uma etapa essencial no processo é a avaliação dos resultados gerados pelo algoritmo. É através da avaliação que poderemos quantificar a qualidade das predições e sabermos se estamos atingindo ou não os objetivos do projeto.

A melhor acurácia foi atingida através do algoritmo Extratrees (0,87), no entanto, esta métrica não o foco principal de nossa avaliação.

Para isto, iremos analisar as métricas de avaliação chamadas **precision** e **recall**, e foi por esse motivo que o algoritmo escolhido foi o **DecisionTree**. Lembrando que a pontuação exigida para este estudo foi de 0,30 para precision e recall.

PRECISION

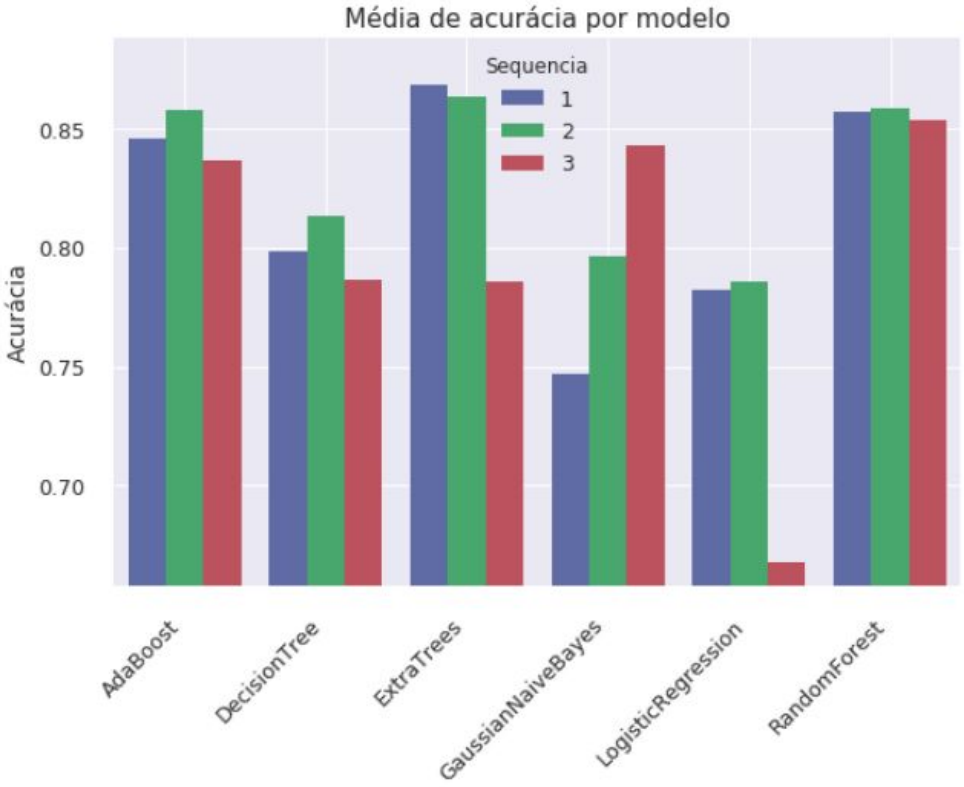
Ter uma alta taxa para a **precision**, não significa que o algoritmo identificou a maioria dos POIs corretamente, mas sim que quando o algoritmo prediz que trata-se de um "1 - POI", é porque há uma grande probabilidade de realmente ele seja um "1 - POI".
No nosso caso a métrica **precision** teve uma taxa de acerto de **0,36**.

RECALL

Uma alta taxa para **recall**, indica que não faço predições desta classe para outras classes. Focando na pontuação da classe "1 - POI" significaria dizer que se a pessoa for um "1 - POI", dificilmente o algoritmo iria predizer que trata-se de um "0 - não-POI".
A pontuação média para a métrica **recall** foi de **0.76**

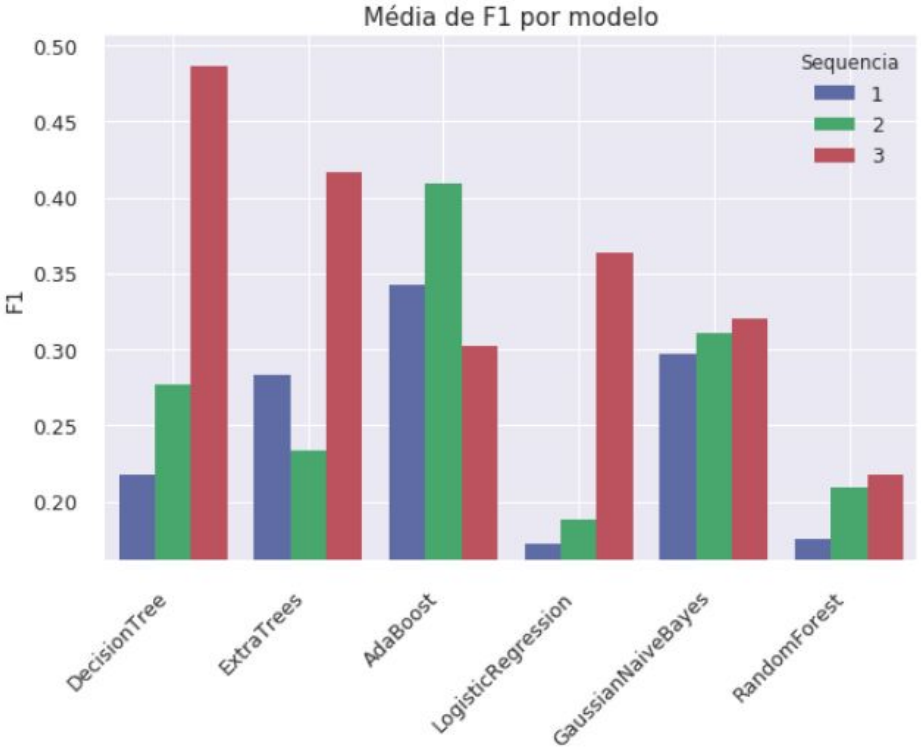
A execução do GridSearchCV na busca pelos melhores parâmetros, foi executado com parâmetro **scoring = F1**, que é uma média harmônica entre as métricas precision e recall.

- Segue o gráfico médias de acurácia por modelo:



Como podemos ver acima, com relação a acurácia os passos realizados não tiveram tanto impacto sobre alguns algoritmos, como AdaBoost, DecisionTree e RandomForest. Já para Naive Bayes houve um ganho considerável, enquanto que para ExtraTrees e Regressão Logística, a acurácia caiu bastante. Mas como o nosso foco principal é em relação a Precision e Recall, iremos analisar o gráfico abaixo com a média da métrica F1 por modelo.

- Segue o gráfico com média de pontuações para a métrica F1:



Pode-se notar no gráfico acima a importância da otimização de parâmetros para melhoria da métrica (Consequentemente Precision e Recall)

Segue abaixo os quatro modelos que obtiveram pontuações de Precision e Recall acima de 0,30 que é um dos requisitos deste projeto:

Modelo	Sequencia	Acurácia	Precision	Recall	F1
DecisionTree	3	0.79	0.36	0.76	0.49
ExtraTrees	3	0.78	0.32	0.57	0.41
AdaBoost	2	0.85	0.44	0.36	0.40
AdaBoost	1	0.85	0.40	0.30	0.34

CONCLUSÃO:

Logicamente seria ótimo se as duas métricas (tanto precision quanto recall) tivessem taxas altas, mas tudo dependerá muito do assunto, do problema e do objetivo do estudo.

Dado a natureza do problema e a quantidade de casos que estamos trabalhando, acredito que duas estratégias poderiam ser utilizadas:

1. Trabalhar para obter **uma taxa alta de recall** para a classe "1 - POI" e se possível uma taxa média para precision. Isso indica que quando o algoritmo está analisando uma pessoa que não está envolvida no caso de fraude, geralmente ele o algoritmo não prediz que ela está envolvida.
   Ao final, teríamos uma lista de possíveis pessoas que estão envolvidas no caso, e a partir daí, iniciaria uma investigação a procura de provas apurando quem realmente está ou não envolvida.
   A ideia principal é separar um grupo de pessoas, e por mais que saibamos da possibilidade de ter pessoas neste grupo que não estão envolvidas no escândalo, sabemos que a grande maioria das pessoas envolvidas (os chamados POIs) estão contidos neste grupo. O trabalho então é levantar provas para identificá-las.

2. Outra estratégia que poderia ser escolhida é ter uma **alta taxa para precision** para a classe 1-POI. Desta maneira podemos até estar deixando muitas pessoas envolvidas com o escândalo passarem, no entanto, as que o algoritmo identificar que são POI, é porque a chance é realmente muito grande. E a partir daí traçar uma estratégia e fazer uma investigação nestas pessoas para conseguir chegar às demais pessoas envolvidas (Ex.: estratégias como delação premiada).
   Pode ser uma boa estratégia em casos onde o custo de investigação é alto.

Entre as duas estratégias acima, eu escolheria a primeira (recall), pois a partir de um número de pessoas identificadas, realizaria uma investigação neste conjunto selecionado para encontrar evidências se a pessoa teve ou não envolvimento na fraude. Neste sentido, o algoritmo obteve uma boa pontuação para a métrica Recall.

O conjunto de dados que trabalhamos é muito pequeno e há um desequilíbrio muito grande entre as classes (0 - não-POI e 1 - POI), contudo com a aplicação de técnicas para limpeza de dados, seleção e criação de características, validação cruzada, otimização de parâmetros dos algoritmos e testes com utilização de várias métricas para avaliação, acredito que os resultados encontrados foram bons.