Answers to the questions!

1. O projeto consiste em identificar causes de fraude na empresa *Enron*. As pessoas de interesse (POI) são identificadas com o label 1, e as pessoas que não são de interesse são identificadas com o label 0 (non-POI). O aprendizado de máquina (machine learning) é usado para encontrar um modelo baseado em alguma técnica (por exemplo, Random Forest, Support Vector Machine) e fazer ajustes nos dados de treino. Em seguida, o modelo que foi ajustado nos dados de treino é aplicado aos dados de teste, e, baseado em alguma métrica, é averiguado se tal modelo também descreve adequadamente os dados de teste. No conjunto de dados estão dadas algumas informações sobre os indivíduos, como bônus, salário, total de pagamentos que são utilizadas para predizer quais seriam os potenciais POI's. No processo de teste de modelos são removidos os pontos "fora da curva" (outliers) pegando os 10% dos pontos que estão mais distantes da média, com base em uma função custo que é a diferença entre a predição (predição através de uma regressão linear dos dados) e o valor do ponto, e eliminando-os do conjunto de dados. A remoção dos outliers tende a melhorar significativamente o ajuste dos modelos aos dados.
Outras informações relevantes:
1) Número total de data-points: 146
2) Número de características usadas (por otimização): 3
3) Existem características com muitos valores faltando: as características que tinham muitos valores faltando foram retiradas da features_list inicial! Foram retiradas as características com mais de **60%** de valores faltantes.

2. Eu utilizei o método SelectKBest por ser o mais apropriado, uma vez que o conjunto de dados é "pequeno". Acabei utlizando os seguintes features, como resultado da otimização do número de features: ['bonus', 'exercised_stock_options', 'shared_receipt_with_poi'], como resultado da seleção mostrada abaixo. Usei como métricas para fazer a comparação recall e precision. Note que o melhor número de features fica para os modelos com número de features 3 e 5, mas o número de features igual a 3 apresenta melhores valores para as métricas acurácia, f1 e f2.

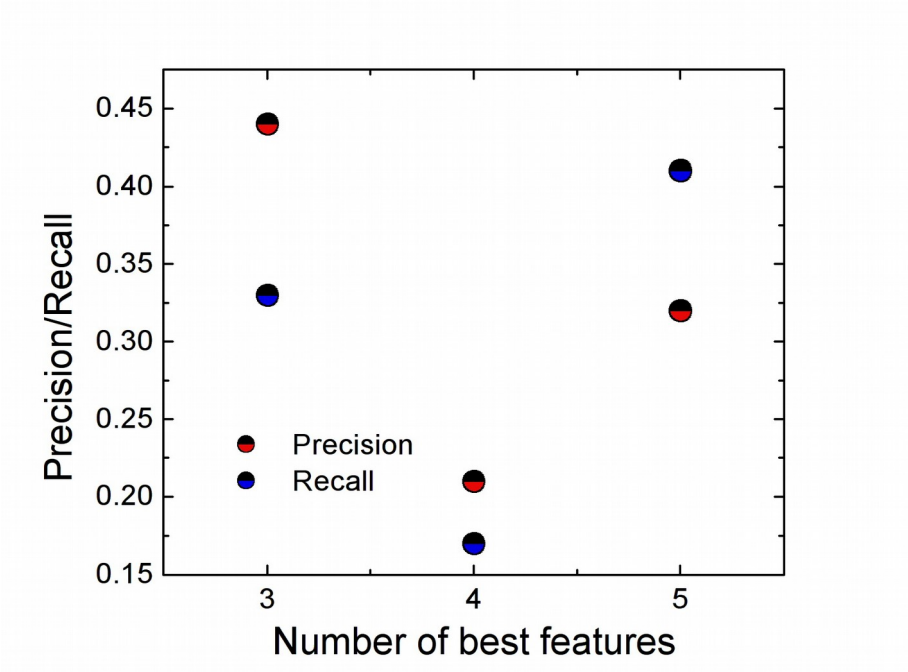

Fig. 1. Seleção do melhor número de atributos que foi encontrado utilizando o SelectKBest e variando o número de features a ser selecionado.

Eu optei por fazer *scaling* necessariamente, uma vez que scaling é mandatório para alguns algoritmos de machine learning, e nos que não são obrigatórios a aplicação de scaling não faz diferença. Em seguida, na lista acima de features eu redefini os features dividindo pelo valor respectivo de 'exercised_stock_options' no dicionário, exceto para o próprio feature ‘exercised_stock_options’. Isso define uma taxa e uma relação de quão signicativo é o 'exercised_stock_options' (que é um atributo que com certeza está correlacionado com fraude de um determinado indivíduo, uma vez que está relacionada com as ações que um indivíduo tem) frente a determinado atributo. A seleção de features com SelectKBest foi feita com base em f_classif conforme mostrado na Fig. 1.

Na tabela abaixo é mostrado uma comparação entre rodar RF com e sem os novos features

Tabela mostrando RF com ou sem as novas features para o número otimizado de features determinado anteriormente

Modelo	Precision	Recall
RF – sem novas features	0.16	0.25
RF – com novas features	0.45	0.33

3. Eu utilizei Decision Tree (DT), Support Vector Machine (SVM) e Gaussian Naive-Bayes (GNB) como modelos possíveis. Os modelos GNB e SVM tiveram desempenho signicativamente inferior aos demais, mesmo alterando vários parâmetros. O modelo DT foi o que teve melhor desempenho.

Tabela mostrando a comparação dos modelos

Modelo	Precision	Recall
DT	0.45	0.33
SVM	0.50	0.06
GNB	0.32	0.21

Note que SVM, apesar de apresentar uma boa precisão, ele tem um péssimo recall!

4. Fazer “fine-tuning” do modelo significa encontrar os melhores parâmetros que o modelo aceita que apresentam o melhor valor de uma métrica, por exemplo, “recall”. Pode-se seguramente utilizar GridSearchCV para esse fim. Se você não fizer isso o “fine-tuning” corretamente, você não encontrará um modelo que funcione corretamente nos dados de treino e teste. Para o DT, os parâmetros foram: ‘criterion’ que é o critério que mede a qualidade do “split”; 'min_samples_leaf' que é o número mínimo de amostras requisitado para criar um nodo na árvore; “max_depth” que é máxima profundidade da árvore; 'min_samples_split' que é o mínimo número de amostras para cada split; “max_features” que é o número de features a considerar quando estiver procurando pelo melhor “split”; “max_leaf_nodes” que é um parâmetro que cresce a árvore com “max_leaf_nodes” na melhor maneira de primeira. Os demais parâmetros continuaram como o previsto pelo “default” de tais.

5 . Validação cruzada (cross-validation) é uma forma de avaliar a qualidade ou habilidade do seu modelo usando um dataset independente. É uma maneira de testar a generalização do modelo. Daí surge o motivo de separar em dataset de treino e testes. Nesse método é conveniente separar o conjunto de dados em partes de teste e treino, mas tomando o cuidado de que os resultados são influenciados pela “fatia” que é tomada como conjunto de teste (o conjunto de treino é o complemento do conjunto de teste com respeito ao conjunto total). Então, divide-se o conjunto de dados total em *k* dobras (folds) e o processo de divisão em treino e teste é efetuado, e no final a média é computada. Um problema clássico de validação é quando temos os dados de treino e teste similares, levando a um modelo que pode apresentar um significativo overfitting. Para contornar isso deve-se utilizar um número razoável de dobras.

6. Trabalhamos com métricas de avaliação denominadas “recall” e “precision”. A definição de recall é:

recall = TP/(TP+FN),

e a de precision:

precision = TP/(TP+FP),

onde TP é o número total de verdadeiros positivos, FN é o número total de falsos negativos e FP é o número total de falsos positivos. É importante basear-se em duas métricas, porque pode acontecer de uma métrica, por exemplo recall, ter valor próximo a 1, o que significa que temos poucos falsos negativos, mas a precisão ser baixa porque temos um número relativamente grande de falsos positivos.

Ambas as métricas tem valor maior do que 0.3 nos dados de teste para o melhor com o algortimo de DT! Isso é bom! Desses valores de recall e precision podemos afirmar que estamos com um modelo razoável para a detecção de POI’s!