Project: Identifying Fraud in Enron Email

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"]*

O objetivo do projeto é uso de *machine learning* para identificação de POI (*persons of interest*) a partir de dados financeiros e de e-mails da Enron. A Enron era uma companhia de energia que se envolveu em fraudes financeiras, e o conjunto de dados de seus e-mails se tornou público. No projeto, POI são funcionários que participaram das fraudes. O uso de machine learning é interessante pois podemos encontrar padrões não óbvios em conjuntos de dados multivariados e complexos.

Inicialmente, os dados foram explorados usando a biblioteca *pandas*. Foram observadas 144 entradas correspondendo a funcionários da Enron e mais duas outras inseridas incorretamente nos dados: "THE TRAVEL AGENCY IN THE PARK" sem dados e não é funcionário, e "TOTAL" que também se configura um **outlier** em dados financeiros. Estas entradas foram removidas. Das 144 restantes, apenas 18 são POI (12,5%). Este desbalanço entre classes merece atenção na hora da criação e avaliação dos algoritmos. Por exemplo, o uso de uma métrica como acurácia pode ser enganosa, uma vez que uma solução trivial como "nenhum funcionário é POI" resultaria em valor elevado de 87,5% de acurácia.

Outra característica notável do conjunto de dados é o grande número de valores ausentes em diversas características, conforme visualizado na *Figura 1*.

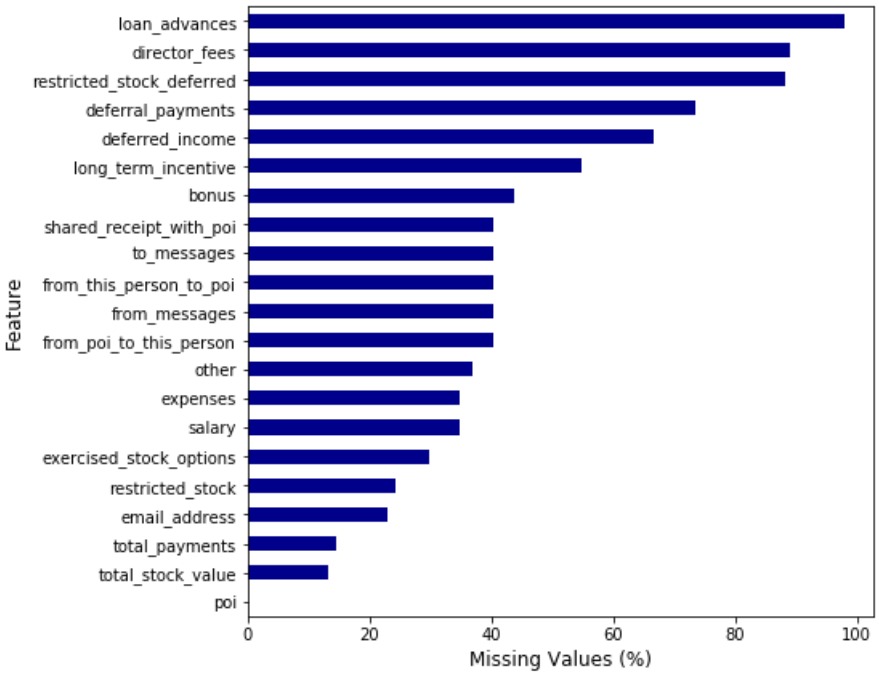

Figura 1. Porcentagem de valores faltantes para as características do conjunto de dados.

Veremos que no processo de seleção de características descrito abaixo, as que contêm maior proporção de valores faltantes tendem a ser menos importantes.

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 importance 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"]*

As 10 características utilizadas no identificador foram:

```
'salary', 'bonus', 'total_stock_value', 'expenses',
'exercised_stock_options', 'other', 'restricted_stock',
'shared_receipt_with_poi', 'fraction_to_poi', fraction_from_poi'
```

Duas novas características foram criadas. Como o número absoluto de e-mails em geral não deve ser um bom preditor para fraudadores, características envolvendo números relativos foram criadas: `fraction_to_poi` e `fraction_from_poi`. A primeira indica, dentre todos os e-mails enviados pelo funcionário, qual fração foi para POIs, e a segunda, dentre todos os e-mails recebidos, qual fração foram de POIs.

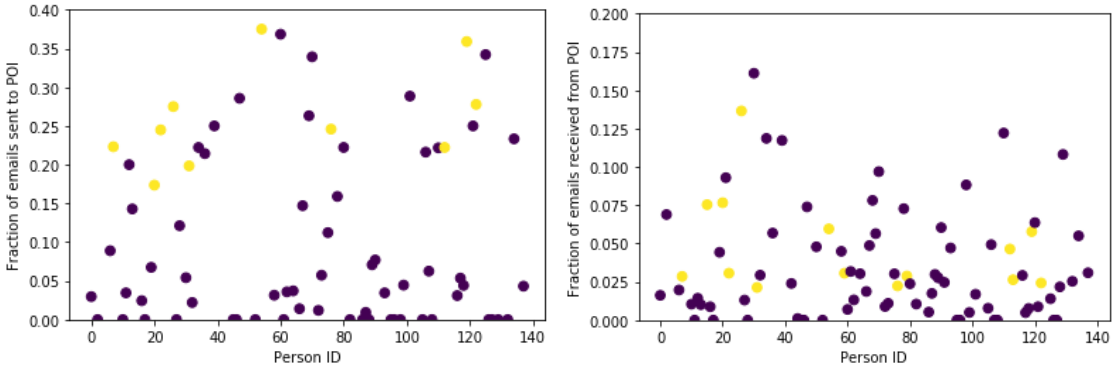

Figura 2. Visualização das duas características criadas. POI em amarelo.

Vemos que, principalmente para `fraction_to_poi` (fração de e-mails que foi enviada para POIs), uma separação relativamente boa existe entre classes. Isto se confirmará na análise da importância desta variável.

Das características iniciais, 3 foram removidas de imediato: `email_adress` pois não carrega nenhuma informação sobre participação em fraudes, e `restricted_stock_deferred` e `director_fees` que não contêm valores para a classe de interesse (POI).

A seleção de características prosseguiu com auxílio de eliminação recursiva com validação cruzada *5-fold*, utilizando a função "RFECV" e *feature_importances_* do algoritmo RandomForest. Utilizando *F1-score* de cada iteração como métrica, observou-

se que a performance degradou significativamente quando mais de 10 características foram utilizadas. Deste modo, numa abordagem conservadora, as características selecionadas foram as 10 mais importantes (classificadas pelo RandomForest).

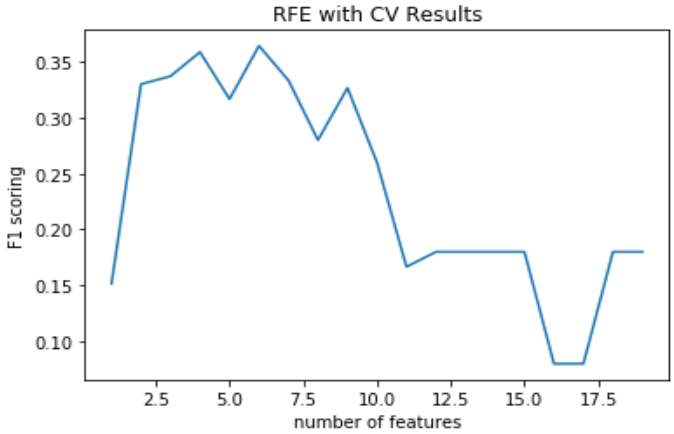

Figura 3. F1-score de validação cruzada 5-fold em função do número de variáveis. Extraído do resultado do algoritmo RFECV, que a cada iteração remove a "pior variável" (classificadas por "*feature importance*").

As duas características criadas ficaram entre as dez mais importantes e, portanto, foram utilizadas no classificador final. Além disso, quando a análise final foi realizada **sem** as *features* criadas, a performance piorou:

Tabela 1. Comparação de performance para modelo com e sem as características criadas.

MODELO	*PRECISION*	*RECALL*
*Random Forest **com** novas features*	0,53	0,45
*Random Forest **sem** novas features*	0,38	0,37

Escalonamento de Características: Escalonamento de características foi utilizado apenas quando o algoritmo de SVM (*support vector machine*) foi testado, pois os algoritmos de Random Forest não se baseiam em distâncias no espaço das características. O escalonamento prévio antes do SVM foi introduzido via "StandardScaler" na função "Pipeline".

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"]*

Dois algoritmos foram testados: *SVM* e *Random Forest*. O último apresentou melhores resultados e foi utilizado. O significado das métricas é discutido ao fim deste documento.

Tabela 2. Métricas de avaliação na análise final para os algoritmos testados.

ALGORITMO	*PRECISION*	*RECALL*	*F1-SCORE*
Random Forest	0,53	0,45	0,49
SVM	0,25	0,89	0,39

Embora SVM tenha bons resultados em *recall*, resultados de *precision* foram insuficientes, não atingindo a meta e resultando em menor *F1-score*.

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"]*

O afinamento (*tuning*) do algoritmo é a otimização dos parâmetros visando melhores predições. É uma etapa importante, pois com parâmetros não-ideais, a performance do classificador/regressor torna-se ruim.

O afinamento foi realizado através de "GridSearchCV": Para *Random Forest*, um hiper-parâmetro importante é o que controla o número mínimo de amostras/observações para que um novo nó seja criado nas árvores ("min_sample_split"), indiretamente controlando a profundidade. Valores na faixa de 2 a 20 foram investigados, e os melhores resultados (via validação cruzada) foram obtidos com o uso de 12. O grid de hiper-parâmetros para *SVM* foi bidimensional, variando a constante *C* e o *kernel*. Os melhores resultados foram obtidos para C=0.1 e *kernel*="rbf".

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"]*

Validação é a etapa em que o desempenho de um modelo/algoritmo é verificado em um conjunto de dados "novo". No caso específico, o algoritmo de classificação é testado com um conjunto de observações diferente do utilizado no processo de ajuste (*fit*). A validação correta evita a ocorrência de um erro clássico, o sobreajuste. Isto ocorre quando parâmetros escolhidos geram resultados que refletem muito bem os dados de ajuste, mas falham em generalizar para novas observações.

Validação cruzada *5-fold* foi realizada (na escolha dos melhores hiper-parâmetros). Métricas de avaliação resultantes deste processo de validação foram extraídas para o melhor classificador (melhor *F1-score*). A performance final também foi avaliada num processo similar de validação cruzada, mas utilizando "StratifiedShuffleSplit", que realiza divisões treino-teste estratificadas e "embaralhadas".
Uma divisão estratificada é interessante para conjuntos de dados desbalanceados, e significa que cada subconjunto criado manterá a proporção dos rótulos, no caso específico, a proporção POI/não-POI. Além disso, o "shuffle" permite criação de um grande número de possíveis divisões via aleatorização de entradas a cada iteração. Este grande número de versões treino-teste diminui as chances de o classificador escolhido ter boa performance devido ao acaso (um corte específico nos dados).

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"]*

Para o conjunto de dados e a investigação realizada, duas métricas importantes são *precision* e *recall*. Seus valores e definições são dados a seguir:

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} = \mathbf{0.45}$$

O *recall* é avaliado dentro do subconjunto de interesse: é a fração destes elementos que foi corretamente classificada. De outro modo, um valor de recall 0.45 indica que, caso um funcionário seja fraudador, o algoritmo tem probabilidade 0.45 de classificá-lo corretamente. Ou seja, é a taxa de positivos verdadeiros (*true positive rate*) no subconjunto de fraudadores.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} = \mathbf{0.53}$$

Precision, por outro lado, é avaliada dentro do subconjunto que foi classificado como de interesse. Trata-se da fração corretamente classificada destes. Com *precision* 0.53, para cada 100 funcionários classificados como poi/fraudadores, o algoritmo estará correto 53 vezes, em média.