1- A Enron foi uma empresa de distribuição de energia e gás e chegou a ser a sétima maior empresa do Estados Unidos, mas após várias fraudes na sua corporação a empresa entrou em colapso. O governo americano abriu dezenas de investigações e identificou que a empresa havia manipulado os seus balanços, caracterizando assim fraude contábil, assim, muitos dados que normalmente são confidenciais, tornaram-se públicos. Dentre eles um relatório com dados financeiros de funcionários e membros do conselho e também milhares de arquivos de e-mails.

O objetivo deste projeto é utilizar técnicas de aprendizado de máquina (machine learning) que consiga aprender com os dados disponibilizados e realizar predições indicando se o funcionário é ou não um POI (Person of interest— Pessoa de interesse).

O conjunto de dados disponível possue 20 features (atributos) do tipo financeiros e do tipo e-mail, e um label se é POI ou não. Como features iniciais para a análise foram utilizadas todas do tipo financeiro e na de e-mail só a feature 'email_address' não foi utilizada, por ser uma string que para essa finalidade não possui uma informação relevante.

A análise inicial mostra que existe 146 amostras, sendo 18 POIs e 128 não-POIs. Em relação a análise de features com muitos valores faltantes, foi realizado o cálculo da proporção de valores faltantes em relação ao valor total das features, essa análise pode ser vista na Figura 1.

É possível notar através da figura que existe features com mais de 80% dos seus valores nulos, onde, provavelmente, elas não possuem informações relevantes para o treiamento.

Figura 1- Proporção valores faltantes no conjunto de dados

2 - A próxima análise feita é para retornar amostras com muitos valores faltantes. Na Figura 2 é possível notar que existe 3 amostras com mais de 80% de suas features com valor nulo. Essas amostras são:

['WODRASKA JOHN',

'LOCKHART EUGENE E',

'THE TRAVEL AGENCY IN THE PARK']

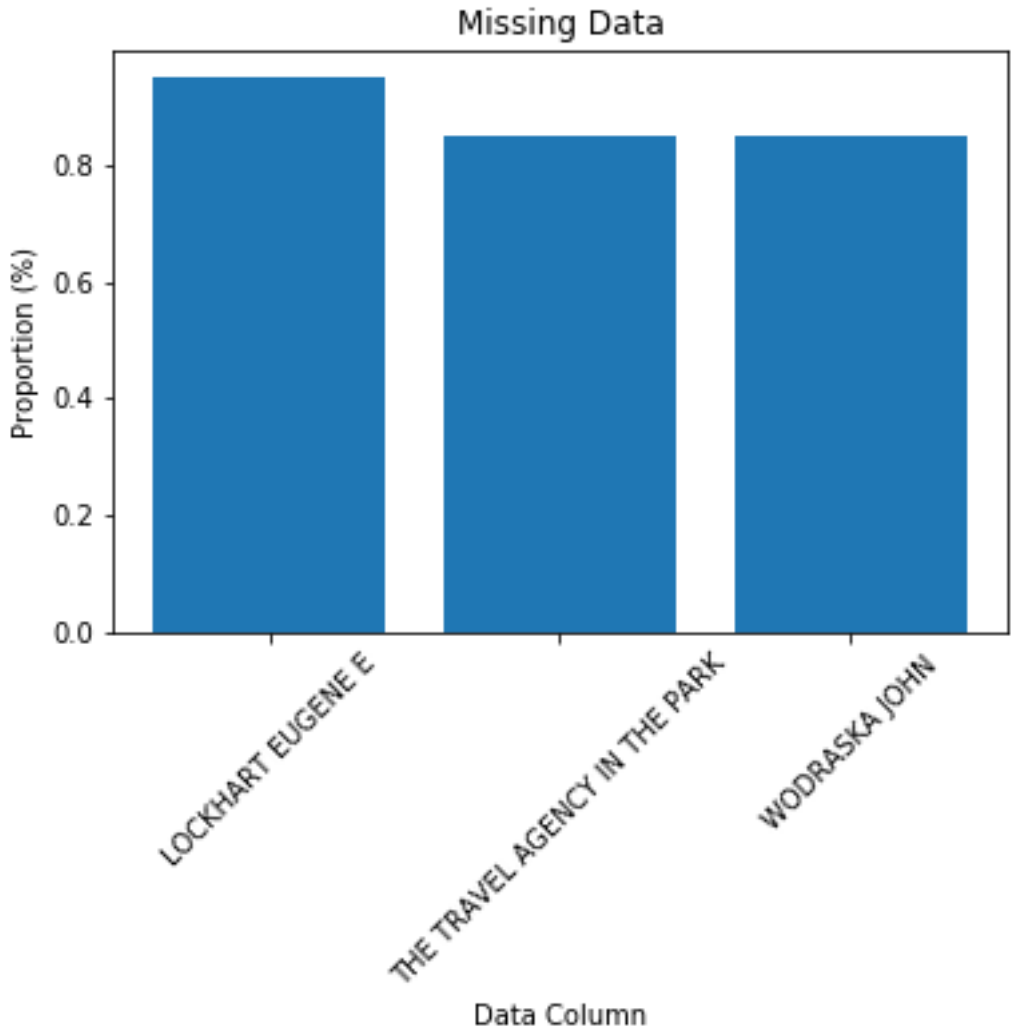

Figura 2- Amostras com mais valores nulos

Já fazendo uma outra análise visual nos dados, é possível notar que o valor de uma amostra está bem desproporcional dos valores das demais, tanto no salário como no bônus, como mostra as Figuras 3 e 4.

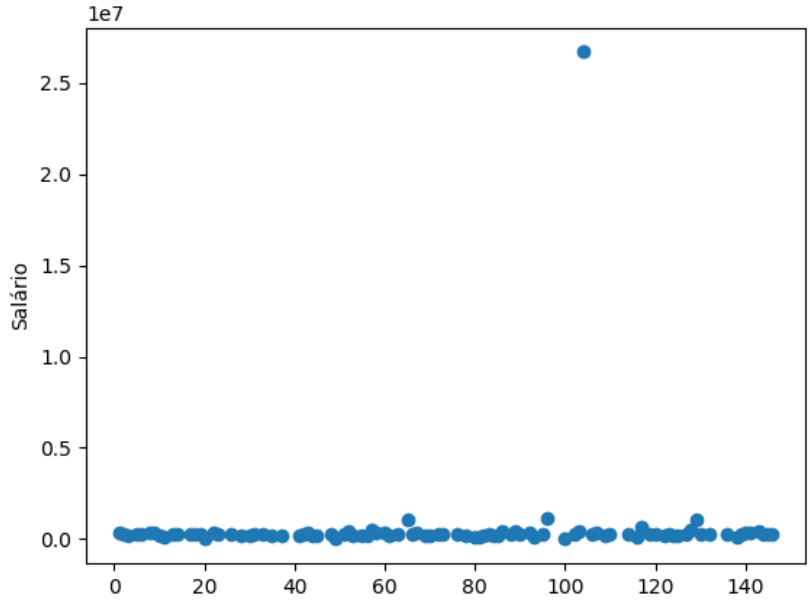

Figura 3- Salários dos amostras do conjunto

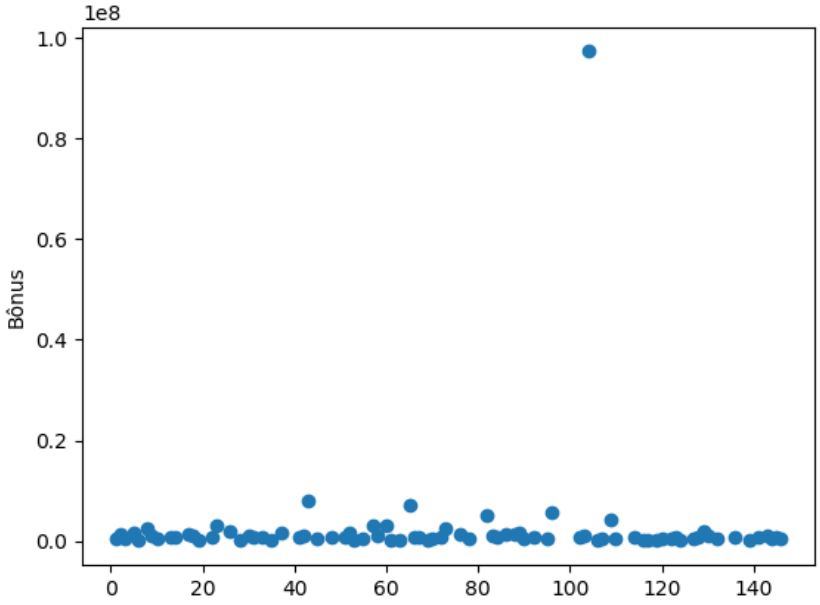

Figura 4-- Bônus dos amostras do conjunto

Essa amostra é ‘TOTAL’ e ela não se trata da informação de um funcionário, e sim de uma linha contendo o valor total de todas as colunas. Portanto é um Outliers e será removido junto com as três amostras anteriores.

2 – Foram criadas 3 novas features:

fraction_from_poi: percentual de mensagens recebidas de um POI em relação ao total recebida.

fraction_to_poi: percentual de mensagens enviadas para um POI em relação ao total enviada.

fraction_total_stock_value: percentual de ações da amostra em relação ao total de ações.

Assim, ficou um total de 23 features no conjunto de dados. A tabela a seguir mostra o impacto das novas features para o modelo final.

RandomForestClassifier	Precision	Recall
Sem novas features	0.5	0.5
Com novas Features	0.6666666666666666	0.5

Neste caso, eu estou considerando todas as features e todas as features com as novas features, sem a seleção das melhores.

Para selecionar as melhores features para o treinamento, com as novas features também incluídas, foi utilizado o método SelecktBest(). Primeiro, foi ordenado com esse método as features que possuíam um maior valor do score. A nova ordem para a lista ‘features_list’ ficou:

['poi', 'exercised_stock_options', 'total_stock_value', 'fraction_total_stock_value', 'bonus', 'salary', 'fraction_to_poi', 'deferred_income', 'long_term_incentive', 'restricted_stock', 'total_payments', 'shared_receipt_with_poi', 'loan_advances', 'expenses', 'from_poi_to_this_person', 'other', 'fraction_from_poi', 'from_this_person_to_poi', 'director_fees', 'to_messages', 'deferral_payments', 'from_messages', 'restricted_stock_deferred']

Assim, também possível notar que as features criadas possuem mais valor significativo para o treinamento do que algumas do conjunto original. O próximo passo, foi criado um laço loop para verificar a eficiência do modelo para diferentes valores de features (K) dessa nova lista, utilizando também o método GridSearch e StandardScaler (normaliza os dados) que retornam os melhores

parâmetros a cada novo valor de k. O resultado obtido pode ser visto na Figura 5.

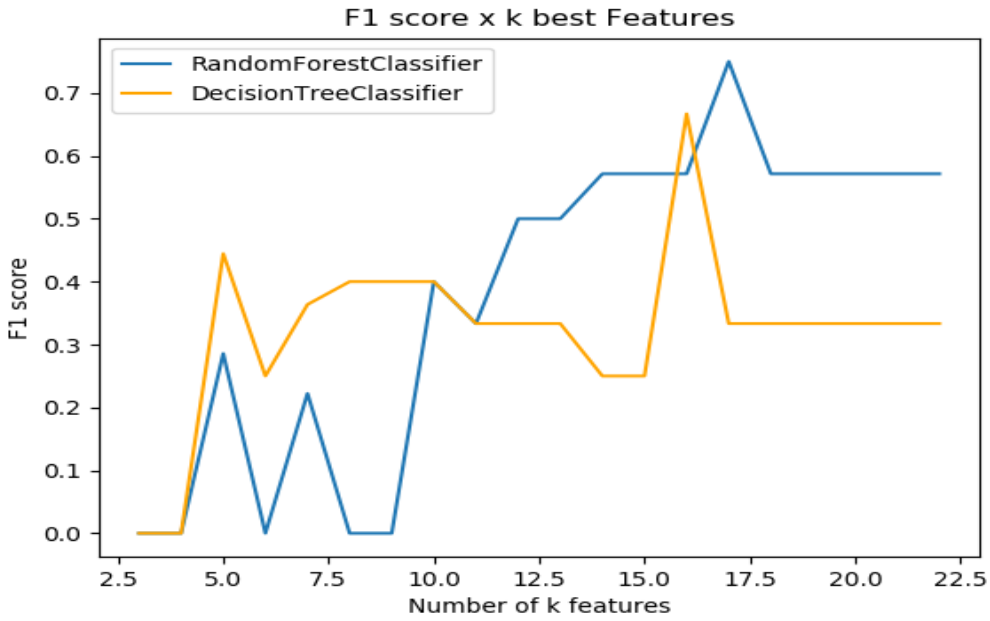

Figura 5- Número dos K melhores features

Neste caso, o modelo RandomForestClassifier apresentou um melhor para o F-score quando k=17.

3 – Os dois algoritmos implementados foram RandomForestClassifier e DecisionTreeClassifier, sendo que o primeiro teve o melhor desempenho, como visto na Figura 5. Na tabela a seguir também pode ser visto o melhor desempenho dos modelos, comprovando que a RandomForestClassifier teve um melhor desempenho.

	Accuracy	**Prediction**	**Recall**
RandomForestClassifier	0.9310344827586207	0.75	0.75
DecisionTreeClassifier	0.896551724137931	0.6	075

4 – A importância de ajustar os parâmetros de um algoritmo é porque esse ajuste retorna a melhor performance do algoritmo para o conjunto de dados utilizado, por causa da descoberta dos melhores parâmetros para a tarefa. Para os dois algoritmos utilizados neste projeto, foi implementado o GridSearchCV para a

escolha dos melhores parâmetros. No RandomForestClassifier os parâmetros ajustados foram:

```
parameters = {
                "criterion": ['entropy', 'gini'],
                "n_estimators": [25, 50, 75],
               "bootstrap": [False, True],
                "max_depth": [3, 5, 10],
                "max_features": ['auto', 0.1, 0.2]
                 }
```

Sendo que os melhores parâmetros foram:

{'bootstrap': False, 'criterion': gini, 'max_features': 'auto', 'n_estimators': 25}

Já em DecisionTreeClassifier o GridSearch foi realizado com os seguintes parâmentros:

```
parameters = {'min_samples_split' : range(10,500,20),
              "criterion": ["gini", "entropy"]}
```

Sendo que os selecionados foram:

{'criterion': entropy, 'min_samples_split': 14}

5 - A validação é importante por permitir a análise se o treinamento do método utilizado foi eficaz e, assim, permite que se utilize o método com uma base de dados independente.

Na validação divide-se o conjunto de dados em treinamento e teste. Para o projeto, foi utilizado o método train_test_split, que divide os dados em treinamento e teste, sendo utilizado, 80% para treinamento e 20% para teste.

6 – As métricas de avaliação utilizadas foram Precision, Recall e Accuracy. A métrica Precision é definida por:

$$\frac{Verdadeiros\ Positivos\ (TP)}{Verdadeiros\ Positivos\ (TP) + Falsos\ Positivos\ (FP)}$$

Que significa que o valor do Precision é o quanto que é confiável na predição quando o algoritmo afirma que o resultado é um determinado valor.

A métrica Recall é definida por:

$$\frac{Verdadeiros\ Positivos\ (TP)}{Verdadeiros\ Positivos\ (TP) + Falsos\ Negativos\ (FN)}$$

Que significa que se o valor do Recall for alto, as predições das classes serão feitas de forma exatas. Já a Accuracy é definida por:

$$\frac{Verdadeiros\ Positivos\ (TP) + Verdadeiros\ Negativos\ (TN)}{Verdadeiros\ Positivos\ (TP) + Verdadeiros\ Negativos\ (TN) + Falsos\ Negativos\ (FN) + Falsos\ Positivos\ (FP)}$$

Que seria uma porcentagem de quanto o classificador está correto.

Com essa métricas é possível dizer qual modelo teve o melhor desempenho, comparando os melhores resultados, como na tabela a seguir. Neste caso o algoritmo de mechine learning final foi o RandomForestClassifier.

	Accuracy	**Prediction**	**Recall**
RandomForestClassifier	0.9310344827586207	0.75	0.75
DecisionTreeClassifier	0.896551724137931	0.6	075

OBSERVAÇÃO: Quando você executar o meu código poi_id.py, você terá as informações das métricas do RandomForestClassifier iguais a da tabela acima. Eu passo esses mesmos parâmetros e dados para testar no arquivo tester.py, mas há uma queda de eficiência mesmo, e não consigo entender porque, já que são os mesmos parâmetros e conjunto. Executei algumas vezes o gridSearch e esses foram os melhores parâmetros. Eu utilizei 80% dos dados para treinamento e 20% para teste, talvez valores diferentes desses pode estar diminuindo a eficiência.