









- 
 Back-End(<https://imasters.com.br/back-end>)
  - 
 Mobile(<https://imasters.com.br/mobile>)
  - 
 Front End(<https://imasters.com.br/front-end>)
  - 
 DevSecOps(<https://imasters.com.br/devsecops>)
  - 
 Design & UX(<https://imasters.com.br/design-ux>)
  - 
 Data(<https://imasters.com.br/data>)
  - 
 APIs e Microsserviços(<https://imasters.com.br/apis-microsservicos>)
  - 
 IoT e Makers(<https://imasters.com.br/iot-makers>)

## PATROCINADORES:


BACK-END




23 NOV, 2016

## Preveno evasão (churning) usando scikit-learn

100 visualizações


 (<https://www.facebook.com/sharer?u=https://imasters.com.br/back-end/preveno-evasio-churning-usando-scikit-learn>)


 (<https://twitter.com/share?url=https://imasters.com.br/back-end/preveno-evasio-churning-usando-scikit-learn>)


 (<https://www.linkedin.com/shareArticle?url=https://imasters.com.br/back-end/preveno-evasio-churning-usando-scikit-learn>)

COMPARTILHE!

LUIZ FELIPE MENDES  
 ([HTTPS://IMASTERS.COM.BR/PERFIL/LUIZFELIPEMENDES](https://imasters.com.br/perfil/luizfelipemendes))  
 Tem 2 artigos publicados com 1012 visualizações desde 2016



PUBLICIDADE

LUIZ FELIPE MENDES ([HTTPS://IMASTERS.COM.BR/PERFIL/LUIZFELIPEMENDES](https://imasters.com.br/perfil/luizfelipemendes))2 

É Data Scientist e co-founder da Hekima, empresa de Big Data e Inteligência Artificial. Formado em Ciência da Computação pela UFMG, trabalha com aprendizado de máquina e análise de dados. Cinéfilo em construção e grande apreciador de sushis. Não confia em quem não gosta de cachorros.

LEIA MAIS ([HTTPS://IMASTERS.COM.BR/PERFIL/LUIZFELIPEMENDES](https://imasters.com.br/perfil/luizfelipemendes))

23 NOV, 2016

Preveno evasão (churning) usando scikit-learn (<https://imasters.com.br/back-end/preveno-evasio-churning-usando-scikit-learn>)

**O** lá a todos novamente!

Hoje farei uma introdução à biblioteca [scikit-learn](http://scikit-learn.org/stable/) (<http://scikit-learn.org/stable/>) de Python. Se você procurar no Google “machine learning library”, imagino que esse vai ser o primeiro resultado. E não é a toa, o scikit é fácil de usar, extramamente completo e possui muito material na internet.

Para exemplificar os principais pontos da biblioteca, faremos uma análise de evasão (churning).

Mas, “Senhor Mendes”, o que é churning?

## Churning

A palavra churning em inglês possui vários significados: agitar leite em uma máquina para produzir manteiga e mover algo com muita força são algumas delas.

Mas a definição que queremos hoje é a de **evasão** – no caso, evasão de clientes. É bem simples: uma empresa tem (normalmente) muitos consumidores; por algum motivo, alguns deles querem abandonar a marca e/ou parar de utilizar o produto. Esses clientes estão evadindo, o que normalmente gera uma diminuição no lucro da empresa (e ninguém gosta de perder dinheiro, não é mesmo?).

G Suite P  
Entre em

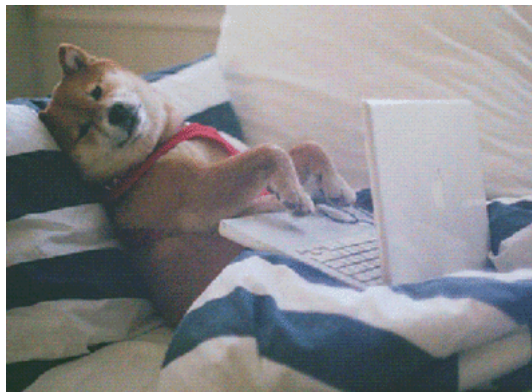
Anúncio U:  
do Gmail, I  
gsuite.google.

Saber ma

O objetivo de uma análise de evasão é tentar entender os motivos dessa saída do cliente. Falamos um pouco sobre isso no artigo sobre análise de sobrevivência (<https://imasters.com.br/desenvolvimento/analise-de-dados-desenvolvimento/vamos-falar-sobre-analise-de-sobrevivencia-survival-analysis/>).

Hoje tentaremos prever quais clientes abandonarão a empresa/produto. Tendo essas informações em mãos, a empresa poderia entrar em contato com esses clientes de forma proativa, oferecendo um novo plano ou um BB-8 de controle remoto (<http://store.sphero.com/products/bb-8-by-sphero>) (quem não quer um BB-8 de controle remoto?).

## Hora de programar



([https://imasters.com.br/?attachment\\_id=108261](https://imasters.com.br/?attachment_id=108261))

**Dados:** Vamos utilizar uma base de dados de churning comum na internet (você pode baixar o CSV (<https://s3.amazonaws.com/zahpee-public/churn.csv>)).

Novamente, estou utilizando o Jupyter Notebook (<https://ipython.org/notebook.html>) (antigo IPython Notebook) para escrever o artigo e fazer toda a programação. Você pode baixar o arquivo aqui (<https://s3.amazonaws.com/zahpee-public/Churning+-+Parte+1.ipynb>) e rodar em seu computador.

```
1 from __future__ import division
2 import pandas as pd
3 import numpy as np
4
5 # Primeiro, vamos ler o csv usando a biblioteca pandas.
6 df_churn = pd.read_csv('churn.csv')
7
8 # Agora vamos ver como se parecem esses dados.
9 df_churn.head()
```

```
<code class="language-python" data-lang="python"><span class="n">df_churn</span><span class="o">.</span><span class="n">head</code>
```

	State	Account Length	Area Code	Phone	Int'l Plan	VMail Plan	VMail Message	Day Mins	Day Calls	Day Charge	...	Eve Calls	Eve Charge	Night Calls	Night Charge
0	KS	128	415	382-4657	no	yes	25	265.1	110	45.07	...	99	16.78	244.7	91
1	OH	107	415	371-7191	no	yes	26	161.6	123	27.47	...	103	16.62	254.4	103
2	NJ	137	415	358-1921	no	no	0	243.4	114	41.38	...	110	10.30	162.6	104
3	OH	84	408	375-9999	yes	no	0	299.4	71	50.90	...	88	5.26	196.9	89
4	OK	75	415	330-6626	yes	no	0	166.7	113	28.34	...	122	12.61	186.9	121

### Kettlebell Emborrachado

R\$ 499,99 R\$ 1.299

Netshoes

Utilizamos no código acima a biblioteca [pandas](http://pandas.pydata.org/) (<http://pandas.pydata.org/>), lemos o CSV e passamos para um dataframe.

### Pré-processamento dos dados

Antes de trabalhar com essa matriz, vamos limpar dados redundantes e não discriminantes.

Os campos de “Phone” e “Area Code” podem ser removidos da matriz.

```
1 | df_churn.drop(['Area Code', 'Phone'], axis=1, inplace=True)
```

Agora, vamos separar a resposta em um vetor. Além disso, vamos transformar o “True” para 1 e o “False” para 0.

Depois disso, vamos remover a resposta do dataframe com os dados e visualizar o vetor de resposta e a tabela resultante. Essa separação é necessária para se adequar a interface dos modelos de predição do Scikit.

```
1 | # adiciona uma nova coluna chamada "Churn" com valores booleanos
2 | df_churn['Churn'] = df_churn['Churn?'] == 'True.'
3 |
4 | # Vamos criar um vetor de resposta y transformando os booleanos em 0 e 1
5 | y = df_churn['Churn'].as_matrix().astype(np.int)
6 |
7 | # agora vamos remover as colunas Churn e Churn? de nosso dataframe
8 | df_churn.drop(['Churn', 'Churn?'], axis=1, inplace=True)
9 |
10 | # Vamos ver como estão nossos dados
11 | print str(y) # deve ser composto de 0s e 1s
12 | df_churn.head()
```

```
1 | [0 0 0 ..., 0 0 0]
```

	State	Account Length	Int'l Plan	VMail Plan	VMail Message	Day Mins	Day Calls	Day Charge	Eve Mins	Eve Calls	Eve Charge	Night Mins	Night Calls
0	KS	128	no	yes	25	265.1	110	45.07	197.4	99	16.78	244.7	91
1	OH	107	no	yes	26	161.6	123	27.47	195.5	103	16.62	254.4	103
2	NJ	137	no	no	0	243.4	114	41.38	121.2	110	10.30	162.6	104
3	OH	84	yes	no	0	299.4	71	50.90	61.9	88	5.26	196.9	89
4	OK	75	yes	no	0	166.7	113	28.34	148.3	122	12.61	186.9	121

A biblioteca do scikit trabalha exclusivamente com atributos numéricos. Logo, é necessário transformar labels e campos \*booleanos em números. No caso, vamos repetir o processo do campo “Churn” para “Int'l Plan” e “VMail Plan” e transformar “yes” em 1 e “no” em 0.

```

1 | yes_or_no = ["Int'l Plan", "VMail Plan"]
2 | df_churn[yes_or_no] = df_churn[yes_or_no] == 'yes'
3 | df_churn[yes_or_no] = df_churn[yes_or_no].astype(np.int)
4 | df_churn.head()

```

	State	Account Length	Int'l Plan	VMail Plan	VMail Message	Day Mins	Day Calls	Day Charge	Eve Mins	Eve Calls	Eve Charge	Night Mins	Night Calls
0	KS	128	0	1	25	265.1	110	45.07	197.4	99	16.78	244.7	91
1	OH	107	0	1	26	161.6	123	27.47	195.5	103	16.62	254.4	103
2	NJ	137	0	0	0	243.4	114	41.38	121.2	110	10.30	162.6	104
3	OH	84	1	0	0	299.4	71	50.90	61.9	88	5.26	196.9	89
4	OK	75	1	0	0	166.7	113	28.34	148.3	122	12.61	186.9	121

Agora, vamos lidar com o campo "State". Ele possui valores em formato de string que transformaremos em números. Para isso podemos usar o `LabelEncoder` (<http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html>) da biblioteca `sklearn`.

```

1 | from sklearn import preprocessing
2 |
3 | # primeiro vamos criar um label encoder
4 | le_state = preprocessing.LabelEncoder()
5 |
6 | # agora vamos passar a coluna State
7 | le_state.fit(df_churn['State'])
8 |
9 | print 'Labels'
10 | print str(list(le_state.classes_))
11 |
12 | df_churn['State'] = le_state.transform(df_churn['State'])
13 |
14 | # podemos também sair do valor numerico e chegar na label
15 | print '\nNúmero para label'
16 | print(list(le_state.inverse_transform([16, 35, 31])))
17 |
18 | df_churn.head()

```

```

1 | Labels
2 | ['AK', 'AL', 'AR', 'AZ', 'CA', 'CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID', 'IL', 'IN', 'KS', 'KY', 'LA', 'MA', 'MD',
3 |
4 | Número para label
5 | ['KS', 'OH', 'NJ']

```

	State	Account Length	Int'l Plan	VMail Plan	VMail Message	Day Mins	Day Calls	Day Charge	Eve Mins	Eve Calls	Eve Charge	Night Mins	Night Calls
0	16	128	0	1	25	265.1	110	45.07	197.4	99	16.78	244.7	91
1	35	107	0	1	26	161.6	123	27.47	195.5	103	16.62	254.4	103
2	31	137	0	0	0	243.4	114	41.38	121.2	110	10.30	162.6	104
3	35	84	1	0	0	299.4	71	50.90	61.9	88	5.26	196.9	89
4	36	75	1	0	0	166.7	113	28.34	148.3	122	12.61	186.9	121

Vamos, então, dar uma olhada em como os valores dessa tabela se comportam?

## 2 Clássicos por R\$14

McDonald's



Primeiro, veremos a média, mediana, variância e o desvio padrão de alguns campos.

```
1 # Primeiro vamos pegar os nomes das colunas
2 col_names = ['Account Length', 'Day Mins', 'Day Calls', 'Day Charge', 'Intl Calls', 'CustServ Calls']
3
4 # Agora vamos mostrar a média
5 for name in col_names:
6     print name
7     print 'Média:' + str(df_churn[name].mean())
8     print 'Mediana:' + str(df_churn[name].median())
9     print 'Variância:' + str(df_churn[name].var())
10    print 'Desvio Padrão:' + str(df_churn[name].std())
11    print '\n'
```

```
1 Account Length
2 Média:101.064806481
3 Mediana:101.0
4 Variância:1585.80012059
5 Desvio Padrão:39.8221059286
6
7
8 Day Mins
9 Média:179.77509751
10 Mediana:179.4
11 Variância:2966.69648652
12 Desvio Padrão:54.4673892024
13
14
15 Day Calls
16 Média:100.435643564
17 Mediana:101.0
18 Variância:402.76814092
19 Desvio Padrão:20.0690842073
20
21
22 Day Charge
23 Média:30.5623072307
24 Mediana:30.5
25 Variância:85.7371282585
26 Desvio Padrão:9.25943455393
27
28
29 Intl Calls
30 Média:4.47944794479
31 Mediana:4.0
32 Variância:6.05757568554
33 Desvio Padrão:2.46121427055
34
35
36 CustServ Calls
37 Média:1.56285628563
38 Mediana:1.0
39 Variância:1.73051668912
40 Desvio Padrão:1.31549104487
```

E analisaremos um pouco as entradas booleanas.

```

1 | bool_fields = ['Int'l Plan', 'VMail Plan']
2 |
3 | # Agora vamos mostrar a somatoria
4 | print 'Total de usuários: ' + str(len(df_churn['VMail Plan']))
5 | print '\n'
6 | for name in bool_fields:
7 |     print name
8 |     print 'Soma: ' + str(df_churn[name].sum())
9 |     print 'Percentual: ' + str(df_churn[name].sum()/len(df_churn[name]))
10 |    print '\n'

```

```

1 | Total de usuários: 3333
2 |
3 |
4 | Int'l Plan
5 | Soma: 323
6 | Percentual: 0.0969096909691
7 |
8 |
9 | VMail Plan
10 | Soma: 922
11 | Percentual: 0.276627662766

```

Vamos olhar outro dado importante: quantos casos de churning temos nesses 3333 usuários.

```

1 | print('Número total de usuários: {}'.format(y.shape[0]))
2 | print('Quantidade de churn: {}'.format(y.sum()))
3 | print('Percentual de churn: {}'.format(y.sum()/y.shape[0]))

```

```

1 | Número total de usuários: 3333
2 | Quantidade de churn: 483
3 | Percentual de churn: 0.144914491449

```

Temos, então, 14.4% de casos de churning dentro dos nossos 3333 usuários observados, o que mostra um desbalanceamento de classes.

O próximo passo no tratamento/processamento da entrada é normalizar os seus valores entre aproximadamente -1.0 e 1.0. Para isso, utilizaremos o `StandardScaler` (<http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html#sklearn.preprocessing.StandardScaler>).

```

1 | from sklearn.preprocessing import StandardScaler
2 | scaler = StandardScaler()
3 |
4 | print 'X antes de dimensionar\n'
5 | print str(df_churn.head())
6 |
7 | X = scaler.fit_transform(df_churn)
8 |
9 | print '\nValores depois do StandardScaler\n'
10 | print str(X)

```

```

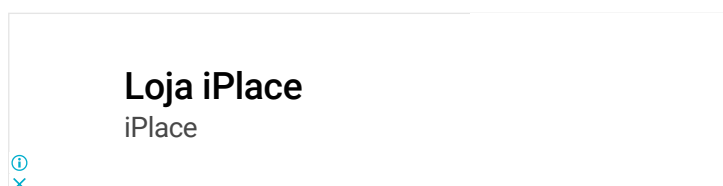
1 X antes de dimensionar
2
3 State Account Length Int'l Plan VMail Plan VMail Message Day Mins \
4 0 16 128 0 1 25 265.1
5 1 35 107 0 1 26 161.6
6 2 31 137 0 0 0 243.4
7 3 35 84 1 0 0 299.4
8 4 36 75 1 0 0 166.7
9
10 Day Calls Day Charge Eve Mins Eve Calls Eve Charge Night Mins \
11 0 110 45.07 197.4 99 16.78 244.7
12 1 123 27.47 195.5 103 16.62 254.4
13 2 114 41.38 121.2 110 10.30 162.6
14 3 71 50.90 61.9 88 5.26 196.9
15 4 113 28.34 148.3 122 12.61 186.9
16
17 Night Calls Night Charge Intl Mins Intl Calls Intl Charge \
18 0 91 11.01 10.0 3 2.70
19 1 103 11.45 13.7 3 3.70
20 2 104 7.32 12.2 5 3.29
21 3 89 8.86 6.6 7 1.78
22 4 121 8.41 10.1 3 2.73
23
24 CustServ Calls
25 0 1
26 1 1
27 2 0
28 3 2
29 4 3
30
31 Valores depois do StandardScaler
32
33 [[-0.6786493 0.67648946 -0.32758048 ..., -0.60119509 -0.0856905
34 -0.42793202]
35 [ 0.6031696 0.14906505 -0.32758048 ..., -0.60119509 1.2411686
36 -0.42793202]
37 [ 0.33331299 0.9025285 -0.32758048 ..., 0.21153386 0.69715637
38 -1.1882185 ]
39 ...,
40 [ 0.87302621 -1.83505538 -0.32758048 ..., 0.61789834 1.3871231
41 0.33235445]
42 [-1.35329082 2.08295458 3.05268496 ..., 2.24335625 -1.87695028
43 0.33235445]
44 [ 1.07541867 -0.67974475 -0.32758048 ..., -0.19483061 1.2411686
45 -1.1882185 ]]

```

## Avaliando o modelo (matriz de confusão e métricas de sucesso)

Agora vamos definir nossa função para avaliar o modelo com um cross-validation ([https://en.wikipedia.org/wiki/Cross-validation\\_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics))). Nesse caso, vou utilizar o StratifiedKFold ([http://scikit-learn.org/stable/modules/generated/sklearn.cross\\_validation.StratifiedKFold.html](http://scikit-learn.org/stable/modules/generated/sklearn.cross_validation.StratifiedKFold.html)), já que existe um desbalanceamento de classes.

O StratifiedKFold mantém o percentual de cada classe nos folds gerados, impedindo que tenhamos pouquíssimos ou nenhum caso de evasão em algum dos folds.



Além disso, vamos criar uma função que desenha a [matriz de confusão](https://en.wikipedia.org/wiki/Confusion_matrix) ([https://en.wikipedia.org/wiki/Confusion\\_matrix](https://en.wikipedia.org/wiki/Confusion_matrix)).

```
1 from sklearn import cross_validation
2
3 # função que realiza a divisão de folds e retorna o y previsto
4 def stratified_cv(X, y, clf_class, shuffle=True, n_folds=10, **kwargs):
5     stratified_k_fold = cross_validation.StratifiedKFold(y, n_folds=n_folds, shuffle=shuffle, random_state=12)
6     y_pred = y.copy()
7     for ii, jj in stratified_k_fold:
8         X_train, X_test = X[ii], X[jj]
9         y_train = y[ii]
10        clf = clf_class(**kwargs)
11        clf.fit(X_train, y_train)
12        y_pred[jj] = clf.predict(X_test)
13    return y_pred
```

```
1 import pylab as pl
2 import matplotlib.pyplot as plt
3
4 %matplotlib inline
5
6 def plot_confusion_matrix(cm, title='Confusion matrix', cmap=plt.cm.Blues):
7     plt.imshow(cm, interpolation='nearest', cmap=cmap)
8     plt.title(title)
9     plt.colorbar()
10    plt.tight_layout()
11    plt.ylabel('True label')
12    plt.xlabel('Predicted label')
13    plt.show
```

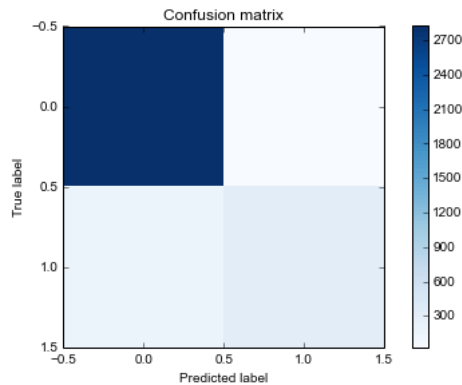
```
1 from sklearn.ensemble import RandomForestClassifier as RF
2 from sklearn.metrics import confusion_matrix
3 from sklearn.metrics import accuracy_score
4 from sklearn.metrics import f1_score
5
6 def test_classifier(X, y, print_cm, classifier, **kwargs):
7     # testando com Random Forest
8     y_pred = stratified_cv(X, y, classifier, **kwargs)
9
10    # Acurácia
11    print 'Acurácia do modelo: ' + str(accuracy_score(y, y_pred))
12    print 'F1 do modelo: ' + str(f1_score(y, y_pred))
13
14    if print_cm:
15        # adicionando resultado na matriz de confusão
16        cm = confusion_matrix(y, y_pred)
17        print '\nMatriz de confusão'
18        print str(cm)
19        plot_confusion_matrix(cm)
20
21        cm_normalized = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
22        print('\nMatriz de confusão normalizada')
23        plt.figure()
24        plot_confusion_matrix(cm_normalized, title='Normalized confusion matrix')
25        print(cm_normalized)
26
27
28
29 test_classifier(X, y, True, RF, random_state=12)
```



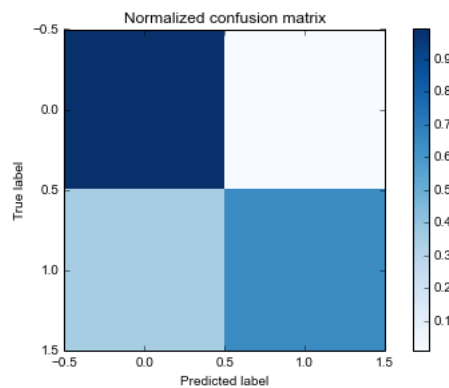
```

1 | Acurácia do modelo: 0.940894089409
2 | F1 do modelo: 0.760048721072
3 |
4 | Matriz de confusão
5 | [[2824  26]
6 |  [ 171 312]]
7 |
8 | Matriz de confusão normalizada
9 | [[ 0.99087719  0.00912281]
10 |  [ 0.35403727  0.64596273]]

```



([https://imasters.com.br/?attachment\\_id=108262](https://imasters.com.br/?attachment_id=108262))



([https://imasters.com.br/?attachment\\_id=108263](https://imasters.com.br/?attachment_id=108263))

Primeiramente, preciso dizer que estamos utilizando o algoritmo Random Forests (<http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>), que utiliza várias árvores de decisão para fazer a classificação. Passamos para ele a matriz X com as features e o vetor y de respostas.



([https://imasters.com.br/?attachment\\_id=108264](https://imasters.com.br/?attachment_id=108264))

Como estamos considerando aqui o problema como uma classificação binária, precisamos escolher uma métrica de sucesso condizente. No caso, vamos mostrar tanto a acurácia ([https://en.wikipedia.org/wiki/Accuracy\\_and\\_precision](https://en.wikipedia.org/wiki/Accuracy_and_precision)), que é simples de entender e explicar, quanto a métrica f1 ([https://en.wikipedia.org/wiki/F1\\_score](https://en.wikipedia.org/wiki/F1_score)), que é mais apropriada para avaliar o modelo quando existe um desbalanceamento de classes.

Utilizamos também a matriz de confusão normalizada para melhor visualizar o quão bem o modelo prevê o churning.

Agora vamos listar a importância das features.

Fonte: Feature importances with forests of trees ([http://scikit-learn.org/stable/auto\\_examples/ensemble/plot\\_forest\\_importances.html](http://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html))

## Importância de features

A importância de uma feature mostra o quão relevante ela foi considerada na criação do modelo. Essa informação pode ser utilizada para selecionar as melhores features para um modelo (Feature Selection ([https://en.wikipedia.org/wiki/Feature\\_selection](https://en.wikipedia.org/wiki/Feature_selection))).

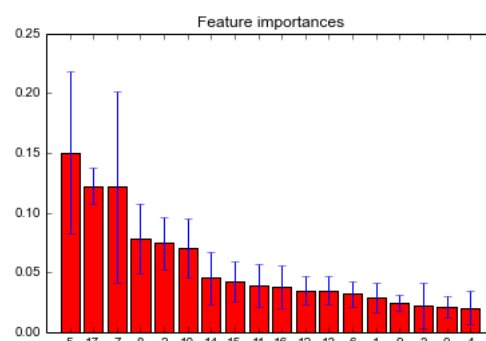
```
1 from sklearn.datasets import make_classification
2 from sklearn.ensemble import RandomForestClassifier
3
4
5 def feature_importance(X,y):
6     # Build a forest and compute the feature importances
7     forest = RandomForestClassifier(random_state=12)
8     forest.fit(X, y)
9     importances = forest.feature_importances_
10    std = np.std([tree.feature_importances_ for tree in forest.estimators_],
11                axis=0)
12    indices = np.argsort(importances)[::-1]
13
14    # Print the feature ranking
15    print("Feature ranking:")
16
17    for f in range(X.shape[1]):
18        print("%d. feature %d (%f)" % (f + 1, indices[f], importances[indices[f]]))
19
20    # Plot the feature importances of the forest
21    plt.figure()
22    plt.title("Feature importances")
23    plt.bar(range(X.shape[1]), importances[indices],
24            color="r", yerr=std[indices], align="center")
25    plt.xticks(range(X.shape[1]), indices)
26    plt.xlim([-1, X.shape[1]])
27    plt.show()
```

```
1 i = 0
2 for feature in df_churn.columns.tolist():
3     print str(i) + ' - ' +feature
4     i += 1
5
6 print '\n'
7
8 feature_importance(X,y)
```

```

1 0 - State
2 1 - Account Length
3 2 - Int'l Plan
4 3 - VMail Plan
5 4 - VMail Message
6 5 - Day Mins
7 6 - Day Calls
8 7 - Day Charge
9 8 - Eve Mins
10 9 - Eve Calls
11 10 - Eve Charge
12 11 - Night Mins
13 12 - Night Calls
14 13 - Night Charge
15 14 - Intl Mins
16 15 - Intl Calls
17 16 - Intl Charge
18 17 - CustServ Calls
19
20
21 Feature ranking:
22 1. feature 5 (0.150194)
23 2. feature 17 (0.122422)
24 3. feature 7 (0.121427)
25 4. feature 8 (0.078148)
26 5. feature 2 (0.074473)
27 6. feature 10 (0.070833)
28 7. feature 14 (0.045192)
29 8. feature 15 (0.042706)
30 9. feature 11 (0.038932)
31 10. feature 16 (0.037543)
32 11. feature 12 (0.035036)
33 12. feature 13 (0.034994)
34 13. feature 6 (0.031734)
35 14. feature 1 (0.028708)
36 15. feature 9 (0.024622)
37 16. feature 3 (0.022160)
38 17. feature 0 (0.020662)
39 18. feature 4 (0.020216)

```



([https://imasters.com.br/?attachment\\_id=108265](https://imasters.com.br/?attachment_id=108265))

Podemos ver por esses dados que as features mais importantes são:

- Day Charge
- Day Mins
- Customer Service Calls

Isso faz sentido, uma vez que um usuário que utiliza pouco e/ou faz muitas ligações para o SAC tem uma chance maior de cancelar seu plano.

Vamos agora variar alguns hyper-parâmetros do classificador?



Chegou i

Anúncio Ch  
cartões cor

SumUp

Confira A

Vamos utilizar aqui o [GridSearchCV](http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.GridSearchCV.html) ([http://scikit-learn.org/stable/modules/generated/sklearn.grid\\_search.GridSearchCV.html](http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.GridSearchCV.html)). Ele analisa todas as combinações possíveis e escolhe a melhor baseada na métrica desejada.

```

1 from sklearn.grid_search import GridSearchCV
2 from sklearn.cross_validation import train_test_split
3 from sklearn.metrics import classification_report
4
5 def gridSearch(X,y,classififer, tuned_parameters):
6     scores = ['f1', 'accuracy']
7
8     X_train, X_test, y_train, y_test = train_test_split(
9         X, y, test_size=0.5, random_state=0)
10
11     for score in scores:
12         print("# Otimização hyper-parametros para %s" % score)
13
14         clf = GridSearchCV(classififer, tuned_parameters, cv=5,
15                             scoring=score)
16         clf.fit(X_train, y_train)
17
18         print("Melhores parâmetros encontrados no conjunto de treino:")
19         print
20         print(clf.best_params_)
21         print
22         print("Resultado do grid:")
23         print
24         for params, mean_score, scores in clf.grid_scores_:
25             print("%0.3f (+/-%0.03f) for %r"
26                   % (mean_score, scores.std() * 2, params))
27         print
28
29         print("Detalhamento:")
30         print
31         print("O modelo é treinado com o conjunto de treino")
32         print("Os resultados finais são do conjunto de teste")
33         print
34         y_true, y_pred = y_test, clf.predict(X_test)
35         print(classification_report(y_true, y_pred))
36         print
37
38
39 tuned_parameters = [{'n_estimators': [10,100,250], 'max_features': ['auto', 'sqrt', 'log2', None]]}
40
41 # Buckle up, this may take a while
42 gridSearch(X,y,RandomForestClassifier(random_state=12),tuned_parameters)

```

```

1 # Otimização hyper-params para f1
2 Melhores parâmetros encontrados no conjunto de treino:
3
4 {'max_features': 'auto', 'n_estimators': 250}
5
6 Resultado do grid:
7
8 0.687 (+/-0.081) for {'max_features': 'auto', 'n_estimators': 10}
9 0.806 (+/-0.062) for {'max_features': 'auto', 'n_estimators': 100}
10 0.815 (+/-0.054) for {'max_features': 'auto', 'n_estimators': 250}
11 0.687 (+/-0.081) for {'max_features': 'sqrt', 'n_estimators': 10}
12 0.806 (+/-0.062) for {'max_features': 'sqrt', 'n_estimators': 100}
13 0.815 (+/-0.054) for {'max_features': 'sqrt', 'n_estimators': 250}
14 0.687 (+/-0.081) for {'max_features': 'log2', 'n_estimators': 10}
15 0.806 (+/-0.062) for {'max_features': 'log2', 'n_estimators': 100}
16 0.815 (+/-0.054) for {'max_features': 'log2', 'n_estimators': 250}
17 0.751 (+/-0.109) for {'max_features': None, 'n_estimators': 10}
18 0.787 (+/-0.068) for {'max_features': None, 'n_estimators': 100}
19 0.793 (+/-0.069) for {'max_features': None, 'n_estimators': 250}
20
21 Detalhamento:
22
23 O modelo é treinado com o conjunto de treino
24 Os resultados finais são do conjunto de teste
25
26          precision    recall  f1-score   support
27
28         0         0.96      0.98      0.97     1422
29         1         0.87      0.73      0.80      245
30
31 avg / total         0.94      0.94      0.94     1667
32
33
34 # Otimização hyper-params para accuracy
35 Melhores parâmetros encontrados no conjunto de treino:
36
37 {'max_features': 'auto', 'n_estimators': 250}
38
39 Resultado do grid:
40
41 0.927 (+/-0.018) for {'max_features': 'auto', 'n_estimators': 10}
42 0.951 (+/-0.014) for {'max_features': 'auto', 'n_estimators': 100}
43 0.953 (+/-0.011) for {'max_features': 'auto', 'n_estimators': 250}
44 0.927 (+/-0.018) for {'max_features': 'sqrt', 'n_estimators': 10}
45 0.951 (+/-0.014) for {'max_features': 'sqrt', 'n_estimators': 100}
46 0.953 (+/-0.011) for {'max_features': 'sqrt', 'n_estimators': 250}
47 0.927 (+/-0.018) for {'max_features': 'log2', 'n_estimators': 10}
48 0.951 (+/-0.014) for {'max_features': 'log2', 'n_estimators': 100}
49 0.953 (+/-0.011) for {'max_features': 'log2', 'n_estimators': 250}
50 0.935 (+/-0.028) for {'max_features': None, 'n_estimators': 10}
51 0.943 (+/-0.019) for {'max_features': None, 'n_estimators': 100}
52 0.945 (+/-0.019) for {'max_features': None, 'n_estimators': 250}
53
54 Detalhamento:
55
56 O modelo é treinado com o conjunto de treino
57 Os resultados finais são do conjunto de teste
58
59          precision    recall  f1-score   support
60
61         0         0.96      0.98      0.97     1422
62         1         0.87      0.73      0.80      245
63
64 avg / total         0.94      0.94      0.94     1667

```

Lembrando que o GridSearchCV utilizado aqui pode demorar muito, dependendo do número de hyper-parâmetros, dado que ele testará todas as combinações possíveis. Existem outras maneiras de variar esses hyper-parâmetros; para aprender mais sobre isso visite a página relativa a [GridSearch](http://scikit-learn.org/stable/modules/grid_search.html) ([http://scikit-learn.org/stable/modules/grid\\_search.html](http://scikit-learn.org/stable/modules/grid_search.html)).

Podemos ver, depois de todos esses testes, que o melhor resultado encontrado foi com os seguintes parâmetros do classificador:

```
{'max_features': 'auto', 'n_estimators': 250}
```

Vamos agora verificar como fica nossa matriz de confusão com esses parâmetros.

```
1 | test_classifier(X,y,False,RandomForestClassifier,max_features='auto', n_estimators= 250,random_state=12)
```

```
1 | Acurácia do modelo: 0.954695469547
2 | F1 do modelo: 0.825433526012
```

Podemos ver que houve uma pequena melhora dos resultados variando os hyper-parâmetros.

**Antigos:**

→ Acurácia do modelo: 0.940894089409

→ F1 do modelo: 0.760048721072

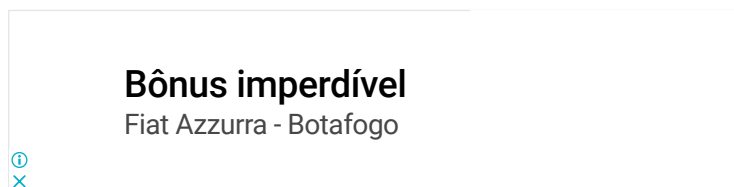
**Novos:**

→ Acurácia do modelo: 0.954695469547

→ F1 do modelo: 0.825433526012

Para finalizar nosso exemplo, vamos tentar gerar mais features?

Vamos utilizar [PolynomialFeatures](http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html#sklearn.preprocessing.PolynomialFeatures) (<http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html#sklearn.preprocessing.PolynomialFeatures>). Com ele conseguimos gerar mais colunas em nossa tabela X, gerando features baseados nos dados existentes.



No caso, se você possui duas features [a,b] essa operação irá gerar [1, a, b, a^2, ab, b^2] – isto é, vai combinar as features com elas mesmas e com as outras.

```
1 | from sklearn.preprocessing import PolynomialFeatures
2 |
3 | print 'Número de features: ' + str(X.shape[1])
4 | poly = PolynomialFeatures(2)
5 | X_poly = poly.fit_transform(X)
6 | print '\nNúmero de features depois da transformação: ' + str(X_poly.shape[1])
```

```
1 | Número de features: 18
2 |
3 | Número de features depois da transformação: 190
```

Saímos de 18 features para 190!

Outra opção é não gerar as features que são combinações delas mesmas, passando o parâmetro `interaction_only` como `true`. Dessa forma, apenas as combinações de features serão geradas.

Exemplo: [1,a,b,ab]

```
1 | poly_less = PolynomialFeatures(interaction_only=True)
2 | X_poly_less = poly_less.fit_transform(X)
3 |
4 | print '\nNúmero de features com interaction_only: ' + str(X_poly_less.shape[1])
```

<code class="language-" data-lang="">Número de features com interaction\_only: 172</code>

Vamos testar nosso classificador agora com 190 features.

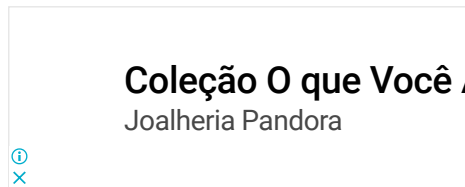
```
1 | test_classifier(X_poly,y,False,RandomForestClassifier,max_features='auto', n_estimators= 250, n_jobs=-1, random_state=12)
```

```
1 | Acurácia do modelo: 0.952895289529
```

```
2 | F1 do modelo: 0.818497109827
```

Uai (sim, sou mineiro), temos mais features e o valor da acurácia piorou?

Provavelmente, isso aconteceu pois fizemos o GridSearch para as 18 features iniciais ou as novas entradas adicionam ruído.



Vamos testar agora um GridSearch nesse hiper-parâmetro para nossas 190 features.

```
1 | tuned_parameters = [{'max_features':['auto','sqrt','log2', None]]
2 |
3 |
4 | # Buckle up, this REALLY may take a while
   gridSearch(X_poly,y,RandomForestClassifier(n_estimators= 250, n_jobs=-1),tuned_parameters)
```

```

1 # Otimização hyper-parâmetros para f1
2 Melhores parâmetros encontrados no conjunto de treino:
3
4 {'max_features': None}
5
6 Resultado do grid:
7
8 0.798 (+/-0.063) for {'max_features': 'auto'}
9 0.783 (+/-0.045) for {'max_features': 'sqrt'}
10 0.693 (+/-0.094) for {'max_features': 'log2'}
11 0.803 (+/-0.064) for {'max_features': None}

```

12 Detalhamento:

13 O modelo é treinado com o conjunto de treino  
 14 Os resultados finais são do conjunto de teste

	precision	recall	f1-score	support
0	0.96	0.97	0.97	1422
1	0.84	0.75	0.79	245
avg / total	0.94	0.94	0.94	1667

```

25
26 # Otimização hyper-parâmetros para accuracy
27 Melhores parâmetros encontrados no conjunto de treino:
28
29 {'max_features': 'sqrt'}
30
31 Resultado do grid:
32
33 0.947 (+/-0.016) for {'max_features': 'auto'}
34 0.950 (+/-0.016) for {'max_features': 'sqrt'}
35 0.928 (+/-0.012) for {'max_features': 'log2'}
36 0.950 (+/-0.013) for {'max_features': None}

```

37 Detalhamento:

38 O modelo é treinado com o conjunto de treino  
 39 Os resultados finais são do conjunto de teste

	precision	recall	f1-score	support
0	0.96	0.98	0.97	1422
1	0.88	0.73	0.80	245
avg / total	0.94	0.95	0.94	1667

```

1 test_classifier(X_poly,y,False,RandomForestClassifier,max_features='sqrt', n_estimators= 250, n_jobs=-1, random_state=12)

```

```

1 Acurácia do modelo: 0.952895289529
2 F1 do modelo: 0.818497109827

```

Demorou MUITO para rodar e mesmo assim não melhoramos o resultado. Talvez alguma dessas 172 features seja interessante, mas parece que o ruído que elas geram não melhora o resultado.

## Conclusão

Nesse artigo, passamos por praticamente todas as etapas de um projeto de ciência de dados: limpamos, pré-processamos e criamos novos dados. Também fizemos um gridsearch para achar os melhores hiper-parâmetros e verificamos os resultados vendo as métricas (acurácia e f1), além de verificar a matriz de confusão gerada.



Logicamente, em um projeto real de churning existem outras complicações. Normalmente os dados são por tempo, e no CSV utilizado aqui, foram apenas os dados relativos a um mês. Além disso, em projetos de evasão temos que levar em conta o negócio para encontrar as melhores métricas de sucesso e o melhor algoritmo. Pode ser que você tenha que retornar uma lista de tamanho definido com os clientes com maior probabilidade de saída, mas com o ferramental apresentado nesse post é possível atacar esses problemas.

Espero que o artigo tenha sido interessante e os ajude a usar o Scikit.

Abraços e até a próxima!

\*\*\*


Artigo publicado originalmente em: <http://developers.hekima.com/machine%20learning/python/2016/05/17/churn-prediction/>  
(<http://developers.hekima.com/machine%20learning/python/2016/05/17/churn-prediction/>)



De 0 a 10, o quanto você recomendaria este artigo para um amigo?



#### ARTIGOS PUBLICADOS POR ESTE AUTOR

 LUIZ FELIPE MENDES ([HTTPS://IMASTERS.COM.BR/PERFIL/LUIZFELIPEMENDES](https://imasters.com.br/perfil/luizfelipecmdes))  
23 NOV, 2016



Prevenção de evasão (churning) usando scikit-learn (<https://imasters.com.br/back-end/prevencao-evacao-churning-usando-scikit-learn>)



SAIBA MAIS  
([HTTPS://IMASTERS.COM.BR/PERFIL/LUIZFELIPEMENDES](https://imasters.com.br/perfil/luizfelipecmdes))

Luiz Felipe Mendes

✉ (<mailto:luiz.mendes@hekima.com>)

 2 Artigo(s)

É Data Scientist e co-founder da Hekima, empresa de Big Data e Inteligência Artificial. Formado em Ciência da Computação pela UFMG, trabalha com aprendizado de máquina e análise de dados. Cinéfilo em construção e grande apreciador de sushis. Não confia em quem não gosta de cachorros.

0 comentários

Classificar por Mais antigos



Adicione um comentário...

[Plugin de comentários do Facebook](#)

Este projeto é oferecido pelas empresas



(<http://bit.ly/2sB2idH>)



([http://impulso.network/?utm\\_source=imasters&utm\\_medium=site&utm\\_campaign=footer](http://impulso.network/?utm_source=imasters&utm_medium=site&utm_campaign=footer))

Este projeto é mantido e patrocinado pelas empresas

<http://www.alura.com.br><https://www.cielo.com.br/e-commerce/>[http://www.dialhost.com.br?  
utm\\_campaign=PatrocinioiMasters&utm\\_sour](http://www.dialhost.com.br?utm_campaign=PatrocinioiMasters&utm_sour)<https://fiap.me/2C8SbRO>[https://www.hostgator.com.br/?  
utm\\_source=imasters&utm\\_medium=logo&utm\\_campaign=hostgator&utm\\_term=hospec](https://www.hostgator.com.br/?utm_source=imasters&utm_medium=logo&utm_campaign=hostgator&utm_term=hospec)<https://www.idexo.com.br/>  
utm\_campaign=hostgator&utm\_term=hospec<https://www.impacta.com.br/><http://www.ingrammicro.com.br/isv/>[https://king.host/?  
utm\\_source=parceiros&utm\\_medium=&utm  
2019&utm\\_campaign=](https://king.host/?utm_source=parceiros&utm_medium=&utm_campaign=2019&utm_campaign=)<http://lambda3.com.br/>[https://www.locaweb.com.br/?  
utm\\_campaign=eventos&utm\\_source=imasters&utm\\_medium=site&utm\\_content=locaweb](https://www.locaweb.com.br/?utm_campaign=eventos&utm_source=imasters&utm_medium=site&utm_content=locaweb)<http://bit.ly/2BXbpvx>  
utm\_content=locaweb<https://www.userede.com.br/>[https://www.schoolofnet.com/cursos/gratuitos/?  
utm\\_source=imasters&utm\\_medium=patrocinio&utm\\_campaign=institucional\\_patrocinio  
institucional&utm\\_content=institucional\\_patrocinio\\_imasters\\_link-  
institucional](https://www.schoolofnet.com/cursos/gratuitos/?utm_source=imasters&utm_medium=patrocinio&utm_campaign=institucional_patrocinio&utm_content=institucional_patrocinio_imasters_link-institucional)<https://developers.totvs.com/>[http://clicklogger.rm.uol.com.br/?  
prd=16&grp=src:34;chn:118;cpg:imasters2019;&msr=Cliques%20de%20Origem:1&oper=11&redir=https://uolhost.uol](http://clicklogger.rm.uol.com.br/?prd=16&grp=src:34;chn:118;cpg:imasters2019;&msr=Cliques%20de%20Origem:1&oper=11&redir=https://uolhost.uol)<http://www.zarpsystem.com.br/>

Este projeto é apoiado pelas empresas

<https://imasters.tech/><http://www.w3c.br>ASSINE NOSSA  
NewsletterFique em dia com as novidades do iMasters! Assine nossa newsletter e receba  
conteúdos especiais curados por nossa equipe

Qual é o seu e-mail?

ASSINAR

[SOBRE O IMASTERS \(HTTPS://IMASTERS.COM.BR/P/SOBRE-O-IMASTERS\)](https://imasters.com.br/p/sobre-o-imasters)

[POLÍTICA DE PRIVACIDADE \(HTTPS://IMASTERS.COM.BR/P/POLITICA-DE-PRIVACIDADE\)](https://imasters.com.br/p/politica-de-privacidade)

[FALE CONOSCO \(HTTPS://IMASTERS.COM.BR/FALE-CONOSCO/\)](https://imasters.com.br/faq)

[QUERO SER AUTOR \(HTTPS://IMASTERS.COM.BR/P/QUERO-SER-AUTOR\)](https://imasters.com.br/p/quero-ser-autor)

[FÓRUM \(HTTPS://FORUM.IMASTERS.COM.BR/\)](https://forum.imasters.com.br/)