spam_classification_binary

April 24, 2018

```
In [81]: import pandas as pd
         from sklearn.linear_model import SGDClassifier, LogisticRegression
         from sklearn.model_selection import train_test_split
```

1, 0. | spam, non-spam classes

word_freq_make: continuous. word_freq_address: continuous. word_freq_all: continuous. word_freq_3d: continuous. word_freq_our: continuous. word_freq_over: continuous. word_freq_remove: continuous. word_freq_internet: continuous. word_freq_order: continuous. word_freq_mail: continuous. word_freq_receive: continuous. word_freq_will: continuous. word_freq_people: continuous. word_freq_report: continuous. word_freq_addresses: continuous. word_freq_free: continuous. word_freq_business: continuous. word_freq_email: continuous. word_freq_you: continuous. word_freq_credit: continuous. word_freq_your: continuous. word_freq_font: continuous. word_freq_000: continuous. word_freq_money: continuous. word_freq_hp: continuous. word_freq_hpl: continuous. word_freq_george: continuous. word_freq_650: continuous. word_freq_lab: continuous. word_freq_labs: continuous. word_freq_telnet: continuous. word_freq_857: continuous. word_freq_data: continuous. word_freq_415: continuous. word_freq_85: continuous. word_freq_technology: continuous. word_freq_1999: continuous. word_freq_parts: continuous. word_freq_pm: continuous. word_freq_direct: continuous. word_freq_cs: continuous. word_freq_meeting: continuous. word_freq_original: continuous. word_freq_project: continuous. word_freq_re: continuous. word_freq_edu: continuous. word_freq_table: continuous. word_freq_conference: continuous. char_freq_;: continuous. char_freq_(: continuous. char_freq_[: continuous. char_freq_!: continuous. char_freq_$: continuous. char_freq_#: continuous. capital_run_length_average: continuous. capital_run_length_longest: continuous. capital_run_length_total: continuous.

```
In [137]: df = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/spambase/s
          df.head()
```

```
Out[137]:       0  0.64  0.64.1  0.1  0.32   0.2   0.3   0.4   0.5   0.6 ...   0.40  \
          0  0.21  0.28    0.50  0.0  0.14  0.28  0.21  0.07  0.00  0.94 ...   0.00
          1  0.06  0.00    0.71  0.0  1.23  0.19  0.19  0.12  0.64  0.25 ...   0.01
          2  0.00  0.00    0.00  0.0  0.63  0.00  0.31  0.63  0.31  0.63 ...   0.00
          3  0.00  0.00    0.00  0.0  0.63  0.00  0.31  0.63  0.31  0.63 ...   0.00
          4  0.00  0.00    0.00  0.0  1.85  0.00  0.00  1.85  0.00  0.00 ...   0.00

              0.41  0.42  0.778   0.43   0.44  3.756   61   278  1
          0  0.132   0.0  0.372  0.180  0.048  5.114  101  1028  1
          1  0.143   0.0  0.276  0.184  0.010  9.821  485  2259  1
```

```
2  0.137   0.0  0.137  0.000  0.000  3.537   40   191  1
3  0.135   0.0  0.135  0.000  0.000  3.537   40   191  1
4  0.223   0.0  0.000  0.000  0.000  3.000   15    54  1

[5 rows x 58 columns]
```

```
In [138]: x = df[list(df.columns)[:-1]]
          y = df['1']
          df['1'].value_counts()
```

```
Out[138]: 0    2788
          1    1812
          Name: 1, dtype: int64
```

```
In [139]: # df['category'] = ('NSP','SP')[bool(df['1'].eq(0).all())]
          df.tail()
          # df.category.value_counts()
```

```
Out[139]:          0  0.64  0.64.1  0.1  0.32   0.2  0.3  0.4  0.5  0.6 ...   0.40  \
          4595  0.31   0.0    0.62  0.0  0.00  0.31  0.0  0.0  0.0  0.0 ...  0.000
          4596  0.00   0.0    0.00  0.0  0.00  0.00  0.0  0.0  0.0  0.0 ...  0.000
          4597  0.30   0.0    0.30  0.0  0.00  0.00  0.0  0.0  0.0  0.0 ...  0.102
          4598  0.96   0.0    0.00  0.0  0.32  0.00  0.0  0.0  0.0  0.0 ...  0.000
          4599  0.00   0.0    0.65  0.0  0.00  0.00  0.0  0.0  0.0  0.0 ...  0.000

                 0.41  0.42  0.778  0.43  0.44  3.756  61  278  1
          4595  0.232   0.0  0.000   0.0   0.0  1.142   3   88  0
          4596  0.000   0.0  0.353   0.0   0.0  1.555   4   14  0
          4597  0.718   0.0  0.000   0.0   0.0  1.404   6  118  0
          4598  0.057   0.0  0.000   0.0   0.0  1.147   5   78  0
          4599  0.000   0.0  0.125   0.0   0.0  1.250   5   40  0

          [5 rows x 58 columns]
```

```
In [140]: x_train,x_test,y_train,y_test=train_test_split(x,y)
```

```
In [141]: # model = SGDClassifier()
          model = LogisticRegression()
          model.fit(x_train,y_train)
```

```
Out[141]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                    intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
                    penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
                    verbose=0, warm_start=False)
```

```
In [142]: model.score(x_test,y_test)
```

```
Out[142]: 0.92
```

```
In [153]: 'Spam',len(df[df['1']==1]),'Not Spam',len(df[df['1']==0])
```

```
Out[153]: ('Spam', 1812, 'Not Spam', 2788)
```