* **Explore spam email dataset**
* **sklearn CountVectorizer**
* **Bayes Classifiers**
* **sklearn pipeline**

**import** pandas **as** pd

df **=** pd**.**read\_csv("spam.csv")

df**.**head()

|  | **Category** | **Message** |
| --- | --- | --- |
| **0** | ham | Go until jurong point, crazy.. Available only ... |
| **1** | ham | Ok lar... Joking wif u oni... |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... |
| **3** | ham | U dun say so early hor... U c already then say... |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... |

df**.**groupby('Category')**.**describe()

|  | **Message** | | | |
| --- | --- | --- | --- | --- |
|  | **count** | **unique** | **top** | **freq** |
| **Category** |  |  |  |  |
| **ham** | 4825 | 4516 | Sorry, I'll call later | 30 |
| **spam** | 747 | 641 | Please call our customer service representativ... | 4 |
|  |  |  |  |  |

Cree columna ‘spam’ 0 o 1 dependiendo de category si tiene o no la palabra spam

df['spam']**=**df['Category']**.**apply(**lambda** x: 1 **if** x**==**'spam' **else** 0)

df**.**head()

|  | **Category** | **Message** | **spam** |
| --- | --- | --- | --- |
| **0** | ham | Go until jurong point, crazy.. Available only ... | 0 |
| **1** | ham | Ok lar... Joking wif u oni... | 0 |
| **2** | spam | Free entry in 2 a wkly comp to win FA Cup fina... | 1 |
| **3** | ham | U dun say so early hor... U c already then say... | 0 |
| **4** | ham | Nah I don't think he goes to usf, he lives aro... | 0 |

**from** sklearn.model\_selection **import** train\_test\_split

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(df**.**Message,df**.**spam)

**from** sklearn.feature\_extraction.text **import** CountVectorizer

v **=** CountVectorizer()

X\_train\_count **=** v**.**fit\_transform(X\_train**.**values)

X\_train\_count**.**toarray()[:2]

array([[0, 0, 0, ..., 0, 0, 0],

[0, 0, 0, ..., 0, 0, 0]], dtype=int64)

**from** sklearn.naive\_bayes **import** MultinomialNB

model **=** MultinomialNB()

model**.**fit(X\_train\_count,y\_train)

MultinomialNB(alpha=1.0, class\_prior=None, fit\_prior=True)

emails **=** [

'Hey mohan, can we get together to watch footbal game tomorrow?',

'Upto 20% discount on parking, exclusive offer just for you. Dont miss this reward!'

]

emails\_count **=** v**.**transform(emails)

model**.**predict(emails\_count)

Output

array([0, 1], dtype=int64)

**X\_test\_count** **=** v**.**transform(X\_test)

model**.**score(**X\_test\_count**, y\_test)

0.98277

**Sklearn Pipeline**

**============**

**from** sklearn.pipeline **import** Pipeline

clf **=** Pipeline([

('vectorizer', CountVectorizer()),

('nb', MultinomialNB())

])

clf**.**fit(X\_train, y\_train)

clf**.**score(X\_test,y\_test)

0.982770

clf**.**predict(emails)

array([0, 1], dtype=int6