Realizar um modelo de machine-learning, para analisar o texto de um e-mail e classificar entre spam ou não spam, a partir do processamento de linguagem natural.

In [1]:
```
### Findspark
import findspark
findspark.init()
```

In [2]:
```
# Imports
import pyspark
from pyspark import SparkContext
from pyspark.sql import SparkSession
from pyspark.ml import Pipeline
from pyspark.ml.feature import IDF, HashingTF, Tokenizer
from pyspark.ml.classification import NaiveBayes, NaiveBayesModel
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
```

Carga de Dados

In [3]:
```
# Spark Context
sc = SparkContext(appName = "ModeloSpam")
sc.setLogLevel("ERROR")
```

```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/02/06 15:07:03 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using
builtin-java classes where applicable
```

In [4]:
```
# Spark Session
spSession = SparkSession.builder.master("local").getOrCreate()
```

```
In [5]: # Carga dos dados gerando RDD com 2 partições
        spamRDD = sc.textFile("dados/dataset4.csv", 2)
        spamRDD.cache()
```

```
Out[5]: dados/dataset4.csv MapPartitionsRDD[1] at textFile at NativeMethodAccessorImpl.java:0
```

```
In [6]: spamRDD.collect()
```

```
Out[6]: ['ham,Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there
        got amore wat...,,,,,,,,,,',
         'ham,Ok lar... Joking wif u oni...,,,,,,,,,,,',
         'ham,U dun say so early hor... U c already then say...,,,,,,,,,,',
         "ham,Nah I don't think he goes to usf, he lives around here though,,,,,,,,,",
         'ham,Even my brother is not like to speak with me. They treat me like aids patent.,,,,,,,,,,',
         "ham,As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your ca
        llertune for all Callers. Press *9 to copy your friends Callertune,,,,,,,,,,",
         "ham,I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've crie
        d enough today.,,,,,,,,,",
         "ham,I've been searching for the right words to thank you for this breather. I promise i wont take
        your help for granted and will fulfil my promise. You have been wonderful and a blessing at all time
        s.,,,,,,,,,,",
         'ham,I HAVE A DATE ON SUNDAY WITH WILL!!,,,,,,,,,,',
         "ham,Oh k...i'm watching here:),,,,,,,,,,",
         'ham,Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.,,,,,,,,,,',
         'ham,Fine if that\x92s the way u feel. That\x92s the way its gota b,,,,,,,,,,',
         'ham,Is that seriously how you spell his name?,,,,,,,,,,',
```

Pré-Processamento

A partir dos dados, podemos constatar que os textos dos e-mails são compostos por frases separadas por ",". Então vamos realizar o experimento Bench Mark selecionando todo o texto até a primeira ",", pois acreditamos a partir da verificação dos dados, que há uma grande informação contida nas palavras até a primeira ",", e acreditamos que podemos obter um bom resultado com um menor gasto computacional.

In [7]:
```python
# Função de transformação
def TransformToVector(inputStr):

    # Separa as colunas
    attList = inputStr.split(",")

    # Ajusta target
    smsType = 0.0 if attList[0] == "ham" else 1.0

    return [smsType, attList[1]]
```

In [8]:
```python
# Aplica
spamRDD2 = spamRDD.map(TransformToVector)
```

In [9]:
```python
# Converte RDD em DF
spamDF = spSession.createDataFrame(spamRDD2, ["label", "message"])
spamDF.cache()
```

Out[9]:
```
DataFrame[label: double, message: string]
```

```
In [10]: spamDF.select("label", "message").show(truncate=False)
```

```
+-----+-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------+
|label|message
|
+-----+-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------+
|0.0  |Go until jurong point
|
|0.0  |Ok lar... Joking wif u oni...
|
|0.0  |U dun say so early hor... U c already then say...
|
|0.0  |Nah I don't think he goes to usf
|
|0.0  |Even my brother is not like to speak with me. They treat me like aids patent.
|
|0.0  |As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your calle
rtune for all Callers. Press *9 to copy your friends Callertune                                    |
|0.0  |I'm gonna be home soon and i don't want to talk about this stuff anymore tonight
|
|0.0  |I've been searching for the right words to thank you for this breather. I promise i wont take you
r help for granted and will fulfil my promise. You have been wonderful and a blessing at all times.|
|0.0  |I HAVE A DATE ON SUNDAY WITH WILL!!
|
|0.0  |Oh k...i'm watching here:)
|
|0.0  |Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.
|
|0.0  |Fine if that  s the way u feel. That  s the way its gota b
|
|0.0  |Is that seriously how you spell his name?
|
|0.0  |I‘m going to try for 2 months ha ha only joking
|
|0.0  |So ü pay first lar... Then when is da stock comin...
|
|0.0  |Aft i finish my lunch then i go str down lor. Ard 3 smth lor. U finish ur lunch already?
|
|0.0  |Fffffffff. Alright no way I can meet up with you sooner?
|
|0.0  |Just forced myself to eat a slice. I'm really not hungry tho. This sucks. Mark is getting worrie
```

```
d. He knows I'm sick when I turn down pizza. Lol                                                |
|0.0  |Lol your always so convincing.
|
|0.0  |Did you catch the bus ? Are you frying an egg ? Did you make a tea? Are you eating your mom's lef
t over dinner ? Do you feel my Love ?                                                          |
+-----+----------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------+
only showing top 20 rows
```

Processamento de Linguagem Natural

```
    Através de um tokenizador, vamos identificar cada palavra diferente e criar um token para cada
palavra. Depois, através de um termometro de frequência, verificaramos a frequência obtida de cada
palavra, e logo avaliamos a importância da palavra no conjunto do texto a partir do IDF.
```

In [11]:
```
# Tokenizador
tokenizador = Tokenizer(inputCol = "message", outputCol = "words")
```

In [12]:
```
# Term Frequency
term_frequency = HashingTF(inputCol = tokenizador.getOutputCol(), outputCol = "tempfeatures")
```

In [13]:
```
# IDF (Inverse Document Frequency)
inverse_tf = IDF(inputCol = term_frequency.getOutputCol(), outputCol = "features")
```

Machine Learning

```
    Para este problema de negócio, selecionamos o algorímo classificador Naive Bayes, que utiliza a
partir do príncipio da probabilidade de Bayes para encontrar o padrão entre os dados. É um ótimo
algorítimo para o processamento de linguagem natural.
```

Vamos também criar um Pipeline para manter sempre uma sequência organizada no processamento do modelo, onde os dados serão tokenizados, verificados sua frequência, nível de importância dentro do texto, e ai então o resultado será processado pelo algorítimo de machine-learning.

In [14]:
```
# Dados de Treino e Teste (75% para treino e 25% para teste)
(dados_treino, dados_teste) = spamDF.randomSplit([0.75, 0.25])
```

In [15]:
```
dados_treino.count()
```

Out[15]:
```
735
```

In [16]:
```
dados_teste.count()
```

Out[16]:
```
265
```

In [17]:
```
# Modelo com Naive Bayes
nbClassifier = NaiveBayes()
```

In [18]:
```
# Pipeline
pipeline = Pipeline(stages = [tokenizador, term_frequency, inverse_tf, nbClassifier])
```

In [19]:
```
# Treinamento com Pipeline
modelo = pipeline.fit(dados_treino)
```

In [20]:
```
# Previsões com dados de teste
previsoes = modelo.transform(dados_teste)
```

In [21]:
```python
previsoes
```

Out[21]:
```
DataFrame[label: double, message: string, words: array<string>, tempfeatures: vector, features: vector,
rawPrediction: vector, probability: vector, prediction: double]
```

In [22]:
```python
previsoes.select("label", "prediction", "probability").collect()
```

Out[22]:
```
[Row(label=0.0, prediction=0.0, probability=DenseVector([0.932, 0.068])),
 Row(label=0.0, prediction=1.0, probability=DenseVector([0.0, 1.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.5618, 0.4382])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=1.0, probability=DenseVector([0.2923, 0.7077])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.9993, 0.0007])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.9862, 0.0138])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.712, 0.288])),
 Row(label=0.0, prediction=1.0, probability=DenseVector([0.0842, 0.9158])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0]))
```

In [23]:
```python
# Avaliador da acurácia
avaliador = MulticlassClassificationEvaluator(predictionCol = "prediction",
                                              labelCol = "label",
                                              metricName = "accuracy")
```

In [24]:
```python
avaliador.evaluate(previsoes)
```

Out[24]:
```
0.9245283018867925
```

In [25]:
```
# Confusion Matrix
previsoes.groupBy("label","prediction").count().show()
```

```
+-----+----------+-----+
|label|prediction|count|
+-----+----------+-----+
|  1.0|       1.0|  122|
|  0.0|       1.0|   15|
|  1.0|       0.0|    5|
|  0.0|       0.0|  123|
+-----+----------+-----+
```

Nosso primeiro modelo apresentou um nível de 0.924 de acurácia, o que o torna um modelo com uma previsão forte, tendo uma alta probabilidade de acerto na sua classificação de e-mail entre Spam e Não Spam.

Agora vamos realizar um segundo modelo, onde vamos aumentar o número de palavras recebidas, fazendo com que o modelo leia uma parte maior do texto, até a segunda ",", e verificar se há mudanças significativas no nível de acurácia do modelo.

Modelo 2 - Analisando mais palavras.

Pré-Processamento

In [26]:
```
# Função de transformação
def TransformToVector2(inputStr):

    # Separa as colunas
    attList = inputStr.split(",")

    # Ajusta target
    smsType = 0.0 if attList[0] == "ham" else 1.0

    return [smsType, attList[1] + attList[2]]
```

In [27]:
```
# Aplica
spam2RDD = spamRDD.map(TransformToVector2)
```

In [28]:
```
# Converte RDD em DF
spam2DF = spSession.createDataFrame(spam2RDD, ["label", "message"])
spam2DF.cache()
```

Out[28]:
```
DataFrame[label: double, message: string]
```

```
In [29]: spam2DF.select("label", "message").show(truncate=False)
```

```
+-----+--------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------+
|label|message
|
+-----+--------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------+
|0.0  |Go until jurong point crazy.. Available only in bugis n great world la e buffet... Cine there got
amore wat...                                                                                        |
|0.0  |Ok lar... Joking wif u oni...
|
|0.0  |U dun say so early hor... U c already then say...
|
|0.0  |Nah I don't think he goes to usf he lives around here though
|
|0.0  |Even my brother is not like to speak with me. They treat me like aids patent.
|
|0.0  |As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your calle
rtune for all Callers. Press *9 to copy your friends Callertune                                      |
|0.0  |I'm gonna be home soon and i don't want to talk about this stuff anymore tonight k? I've cried en
ough today.                                                                                          |
|0.0  |I've been searching for the right words to thank you for this breather. I promise i wont take you
r help for granted and will fulfil my promise. You have been wonderful and a blessing at all times.|
|0.0  |I HAVE A DATE ON SUNDAY WITH WILL!!
|
|0.0  |Oh k...i'm watching here:)
|
|0.0  |Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.
|
|0.0  |Fine if that  s the way u feel. That  s the way its gota b
|
|0.0  |Is that seriously how you spell his name?
|
|0.0  |I‘m going to try for 2 months ha ha only joking
|
|0.0  |So ü pay first lar... Then when is da stock comin...
|
|0.0  |Aft i finish my lunch then i go str down lor. Ard 3 smth lor. U finish ur lunch already?
|
|0.0  |Fffffffff. Alright no way I can meet up with you sooner?
|
|0.0  |Just forced myself to eat a slice. I'm really not hungry tho. This sucks. Mark is getting worrie
```

```
d. He knows I'm sick when I turn down pizza. Lol                                            |
|0.0  |Lol your always so convincing.
|
|0.0  |Did you catch the bus ? Are you frying an egg ? Did you make a tea? Are you eating your mom's lef
t over dinner ? Do you feel my Love ?                                                       |
+-----+---------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------+
only showing top 20 rows
```

Processamento de Linguagem Natural

In [30]:
```
# Tokenizador
tokenizador2 = Tokenizer(inputCol = "message", outputCol = "words")
```

In [31]:
```
# Term Frequency
term_frequency2 = HashingTF(inputCol = tokenizador2.getOutputCol(), outputCol = "tempfeatures")
```

In [32]:
```
# IDF (Inverse Document Frequency)
inverse_tf2 = IDF(inputCol = term_frequency2.getOutputCol(), outputCol = "features")
```

Machine Learning

In [33]:
```
# Dados de Treino e Teste
(dados_treino, dados_teste) = spam2DF.randomSplit([0.75, 0.25])
```

In [34]:
```
dados_treino.count()
```

Out[34]:
```
764
```

In [35]:
```
dados_teste.count()
```
Out[35]:
```
236
```
In [36]:
```
# Pipeline
pipeline2 = Pipeline(stages = [tokenizador2, term_frequency2, inverse_tf2, nbClassifier])
```
In [37]:
```
# Treinamento com Pipeline
modelo2 = pipeline2.fit(dados_treino)
```
In [38]:
```
# Previsões com dados de teste
previsoes2 = modelo2.transform(dados_teste)
```
In [39]:
```
previsoes2
```
Out[39]:
```
DataFrame[label: double, message: string, words: array<string>, tempfeatures: vector, features: vector,
rawPrediction: vector, probability: vector, prediction: double]
```

In [40]:
```python
previsoes2.select("label", "prediction", "probability").collect()
```

Out[40]:
```
[Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=1.0, probability=DenseVector([0.0001, 0.9999])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.9714, 0.0286])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.9952, 0.0048])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=1.0, probability=DenseVector([0.4041, 0.5959])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.9483, 0.0517])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([1.0, 0.0])),
 Row(label=0.0, prediction=0.0, probability=DenseVector([0.9938, 0.0062])),
```

In [41]:
```python
# Avaliador da acurácia
avaliador.evaluate(previsoes2)
```

Out[41]:
```
0.9322033898305084
```

In [42]:
```
# Resumindo as previsões - Confusion Matrix
previsoes2.groupBy("label","prediction").count().show()
```

```
+-----+----------+-----+
|label|prediction|count|
+-----+----------+-----+
|  1.0|       1.0|  130|
|  0.0|       1.0|   13|
|  0.0|       0.0|   90|
|  1.0|       0.0|    3|
+-----+----------+-----+
```

O modelo 2 apresentou uma performance melhor, com a acurácia de 0.932, ligeiramente maior em relação ao modelo bench mark, o que pode repercutir em uma ligeira taxa de acerto maior nas suas classificações com esta massa de dados. Entretanto, ficam minhas orientações para possivelmente considerar do modelo 1 para produção. Pois a acurácia do modelo 2 pode ser ligeiramente maior, mas não tão significativa comparado a diferença da capacidade computacional que pode demandar o treinamento do modelo 2 frente ao modelo 1 no processamento de mais palavras, ao longo do tempo, dependendo do tamanho da massa de dados.

Também sugiro a criação de outros modelos, com diferentes combinações de hiperparâmetros, e/ou com outros algorítmos, buscando se possível o aumento da precisão do modelo na classificação de e-mails em spam ou não spam.