Regresión Logística: Detencion de SPAM

En este ejercicio se muestran los fundamentos de la regresión logística planteando uno de los primeros problemas que fueron solucionados mediante el uso de teécnicas de Machine Learning: La detención de SPAM.

Enunciado del ejercico.

SE propone la construcción de un sistema de aprendizaje automáticos capaz de predecir si un correo determinado se corresponde con un correo SPAM o no, para ello se utilizara el siguiente DataSet:

2007 TREC Public Spam Corpus

The corpus trec07p contains 75,419 messages:

```
25220 ham
50199 spam
```

These messages constitute all the messages delivered to a particular server between these dates:

```
Sun, 8 Apr 2007 13:07:21 -0400
Fri, 6 Jul 2007 07:04:53 -0400
```

In [1]:
```python
# En esta clase se facilita el procesamiento de correos electronicos que poseen código HTML

from html.parser import HTMLParser

class MLStripper(HTMLParser):
    def __init__(self):
        self.reset()
        self.strict = False
        self.convert_charrefs = True
        self.fed = []

    def handle_data(self, d):
        self.fed.append(d)

    def get_data(self):
        return ''.join(self.fed)
```

In [8]:
```python
# Esta función se encarga de eliminar los tags HTML que se encuentren en el texto dehtml los correos electrónicos
def strip_tags(html):
    s = MLStripper()
    s.feed(html)
    return s.get_data()
```

In [9]:
```python
# Ejemplo de eliminacion de los tags HTML de un texto
t = '<tr><td align="left"><ahref="../../issues/51/16.html#article">Phrack World News</a><td>'
strip_tags(t)
```

Out[9]:
```
'Phrack World News'
```

Ademas de eliminar los posibles tags HTML que se encuentra en el correo electronico deben realizarse otras acciones para evitar que los mensajes contengan ruido inecesario. Entre ellas se encuentran la eliminación de os signos de puntuación, eliminación de los posibles campos de correo electronico que no sean relevantes o eliminación de los afijos de una palabra manteniendo únicamente la raíz de la misma (steming). La clase que se muestra a continuación realiza estas transformaciones.

In [10]:

```
import email
import string
import nltk

class Parser:

    def __init__(self):
        self.stemmer =nltk.PorterStemmer()
        self.stopwords= set(nltk.corpus.stopwords.words('english'))
        self.punctuation = list(string.punctuation)

    def parse(self,email_path):
        """Parse an email."""
        with open(email_path,errors = 'ignore') as e:
            msg = email.message_from_file(e)
        return None if not msg else self.get_email_content(msg)

    def get_email_content(self,msg):
        """Extract the email content."""
        subject = self.tokenize(msg['Subject']) if msg['Subject'] else []
        body= self.get_email_body(msg.get_payload(),
                                  msg.get_content_type())
        content_type = msg.get_content_type()
        #Return the content of the email
        return {"Subject": subject,
                "body":body,
                "content_type": content_type}

    def get_email_body(self,payload,content_type):
        """Extract the body of the email."""
        body= []
        if type(payload) is str and content_type == 'text/plain':
            return self.tokenize(payload)
        elif type(payload) is str and content_type == 'text/html':
            return self.tokenize(strip_tags(payload))
        elif type(payload) is list:
            for p in payload:
                body += self.get_email_body(p.get_payload(),
                                            p.get_content_type())
        return body
```

```
        self.punctuation = list(string.punctuation)

    def parse(self,email_path):
        """Parse an email."""
        with open(email_path,errors = 'ignore') as e:
            msg = email.message_from_file(e)
        return None if not msg else self.get_email_content(msg)

    def get_email_content(self,msg):
        """Extract the email content."""
        subject = self.tokenize(msg['Subject']) if msg['Subject'] else []
        body= self.get_email_body(msg.get_payload(),
                                  msg.get_content_type())
        content_type = msg.get_content_type()
        #Return the content of the email
        return {"Subject": subject,
                "body":body,
                "content_type": content_type}

    def get_email_body(self,payload,content_type):
        """Extract the body of the email."""
        body= []
        if type(payload) is str and content_type == 'text/plain':
            return self.tokenize(payload)
        elif type(payload) is str and content_type == 'text/html':
            return self.tokenize(strip_tags(payload))
        elif type(payload) is list:
            for p in payload:
                body += self.get_email_body(p.get_payload(),
                                            p.get_content_type())
        return body

    def tokenize(self, text):
        """Transform a text string in tokens. Perfomr two main actions,
        claen the punctuation symbols and do stemming of tehe text."""
        for c in self.punctuation:
            text= text.replace(c,"")
        text = text.replace("/t","")
        text = text.replace("/n","")
        tokens =list(filter(None, text.split(" ")))
        #Stemming of the tokens
        return [self.stemmer.stem(w) for w in tokens if w not in self.stopwords]
```

Lectura de un correo en formato .raw

In [11]:
```
inmail = open("datasets/datasets/trec07p/data/inmail.1").read()
print(inmail)
```

```
From RickyAmes@aol.com  Sun Apr  8 13:07:32 2007
Return-Path: <RickyAmes@aol.com>
Received: from 129.97.78.23 ([211.202.101.74])
        by speedy.uwaterloo.ca (8.12.8/8.12.5) with SMTP id l38H7G0I003017;
        Sun, 8 Apr 2007 13:07:21 -0400
Received: from 0.144.152.6 by 211.202.101.74; Sun, 08 Apr 2007 19:04:48 +0100
Message-ID: <WYADCKPDFWWTWTXNFVUE@yahoo.com>
From: "Tomas Jacobs" <RickyAmes@aol.com>
Reply-To: "Tomas Jacobs" <RickyAmes@aol.com>
To: the00@speedy.uwaterloo.ca
Subject: Generic Cialis, branded quality@
Date: Sun, 08 Apr 2007 21:00:48 +0300
X-Mailer: Microsoft Outlook Express 6.00.2600.0000
MIME-Version: 1.0
Content-Type: multipart/alternative;
        boundary="--889648405160655728 6"
X-Priority: 3
X-MSMail-Priority: Normal
Status: RO
Content-Length: 988
Lines: 24

----889648405160655728 6
Content-Type: text/html;
Content-Transfer-Encoding: 7Bit

<html>
<body bgcolor="#ffffff">
<div style="border-color: #00FFFF; border-right-width: 0px; border-bottom-width: 0px; margin-bottom: 0px;" align="ce
nter">
<table style="border: 1px; border-style: solid; border-color:#000000;" cellpadding="5" cellspacing="0" bgcolor="#CCF
FAA">
<tr>
<td style="border: 0px; border-bottom: 1px; border-style: solid; border-color:#000000;">
<center>
Do you feel the pressure to perform and not rising to the occasion??<br>
FAA">
<tr>
<td style="border: 0px; border-bottom: 1px; border-style: solid; border-color:#000000;">
<center>
Do you feel the pressure to perform and not rising to the occasion??<br>
</center>
</td></tr><tr>
<td bgcolor=#FFFF33 style="border: 0px; border-bottom: 1px; border-style: solid; border-color:#000000;">
<center>

<b><a href='http://excoriationtuh.com/?lzmfnrdkleks'>Try <span>V</span><span>ia<span></span>gr<span>a</span>.....</a
></b></center>
</td></tr><td><center>your anxiety will be a thing of the past and you will<br>
be back to your old self.
</center></td></tr></table></div></body></html>


----889648405160655728 6--
```

Parsing del correo electrónico

In [12]:
```
p = Parser()
p.parse("datasets/datasets/trec07p/data/inmail.1")
```

Out[12]:
```
{'Subject': ['gener', 'ciali', 'brand', 'qualiti'],
 'body': ['\n\n\n\n\n\n\ndo',
  'feel',
  'pressur',
  'perform',
  'rise',
  'occasion\n\n\n\n\n\ntri',
  'viagra\nyour',
  'anxieti',
  'thing',
  'past',
  'will\nb',
  'back',
  'old',
  'self\n\n\n'],
 'content_type': 'multipart/alternative'}
```