Naive Bayes Classifier

Spam E-mail Detection

25 de agosto de 2025

Situación problema

El problema consiste en clasificar automáticamente emails como spam o no spam y poder alivianar la bandeja de entrada de usuarios de correo electrónico, usando clasificadores ingenuos de Bayes.

Figura: Correo spam físico

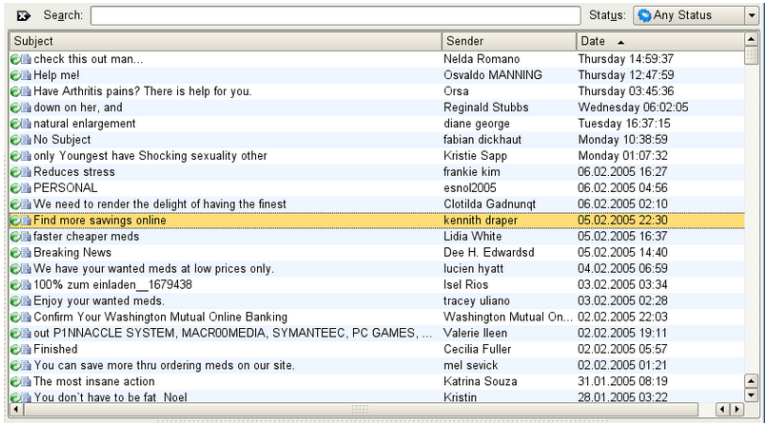

Figura: Correo spam electrónico

Preprocesamiento

El conjunto de datos trabajado puede encontrarse en el siguiente hipervínculo: **Mail Dataset** Para el procesamiento se realizó lo siguiente

- Limpieza del detaset
  1. Eliminación de columnas innecesarias.
  2. Renombrado de columnas.
  3. Codificación de etiquetas.
- Preprocesamiento de texto (NLP)
  1. Estandarizar a minúsculas los strings `.lower()`.
  2. Remoción de caracteres especiales.
  3. Remoción de stopwords.
  4. Stemming y lemmatization.
- División de datos para entrenamiento y evaluación.

Funciones implementadas

Se construyeron las siguientes funciones como bloques en el diseño del clasificador.

- flatten_list (`list[list] -> list`):
  Esa función toma una lista de palabras y las aplana en una sola lista concatenando todas las palabras de todos los emails.
- bag_of_words (`pd.Dataframe -> dict`):
  Retorna la tabla de frecuencias para las palabras de un conjunto de mails.
- probability_words (`pd.Dataframe -> dict`): Retorna la tabla de frecuencias relativas para las palabras de un conjunto de mails.
- clasiffy_email (`list -> int`): returna entre `1, 0` para clasificar un email en spam

Construcción de Bolsa de Palabras

Algorithm 1 Construcción de Bolsa de Palabras

Require: corpus: conjunto de emails tokenizados

Ensure: diccionario $\{palabra : frecuencia\}$

```
lista_palabras ← ∅
for cada email en corpus do
    for cada palabra en email do
        agregar palabra a lista_palabras
        agrega 1 al contador de la palabra
    end for
end for
calcular frecuencia de cada palabra única en lista_palabras
return diccionario con palabras y sus frecuencias
```

Cálculo de Probabilidades de Palabras

Algorithm 2 Probabilidades de Palabras

Require: df: conjunto de emails (spam o no-spam) procesados

Ensure: diccionario $\{palabra : probabilidad\}$

1: $lista\_palabras \leftarrow$ aplanar todas las palabras de df
2: $tamaño\_total \leftarrow$ longitud de $lista\_palabras$
3: $frecuencias \leftarrow$ bag_of_words(df)
4: $probabilidades \leftarrow \emptyset$
5: **for** cada $(palabra, frecuencia)$ en $frecuencias$ **do**
6: $\quad probabilidades[palabra] \leftarrow \frac{frecuencia}{tamaño\_total}$
7: **end for**
8: **return** $probabilidades$

Clasificador de correos

Algorithm 3 Clasificar correo

Require: email: lista de palabras procesadas del correo
Require: p_{spam}: probabilidad a priori de spam
Require: p_{no_spam}: probabilidad a priori de no-spam
Ensure: 1 si el email es spam, 0 en caso contrario

1: $spam\_prob \leftarrow \log(p_{spam})$
2: $not\_spam\_prob \leftarrow \log(p_{no\_spam})$
3: **for** cada palabra en email **do**
4: $spam\_word\_prob \leftarrow probability\_spam\_words.get(palabra, 10^{-6})$
5: $not\_spam\_word\_prob \leftarrow probability\_ham\_words.get(palabra, 10^{-6})$
6: $spam\_prob \leftarrow spam\_prob + \log(spam\_word\_prob)$
7: $not\_spam\_prob \leftarrow not\_spam\_prob + \log(not\_spam\_word\_prob)$
8: **end for**
9: **return** 1 if $spam\_prob > not\_spam\_prob$ else 0

Funcionamiento del clasificador

Para implementar la solución computacionalmente, se utilizó el suavizamiento de Laplace, que asigna una probabilidad muy pequeña ($1e^{-6}$), pero no cero, a las palabras no encontradas en la bolsa de palabras. De esta manera, se evitan ceros cuando hay una palabra no vista antes.
Además, trabajamos en el espacio logarítmico para evitar el underflow numérico y convertir productos en sumas.

$$\log\Big(P(c)\prod_{i=1}^{n}P(w_i \mid c)\Big) \;=\; \log P(c) \;+\; \sum_{i=1}^{n}\log P(w_i \mid c)$$

Funcionamiento del clasificador

Por último, usamos un treshold (o umbral) de 0.5 para disminuir la cantidad de falsos positivos.

Evaluación del modelo

Al construir el modelo clasificador Naive Bayes y entrenarlo se empleó para la evaluación del desempeño la matriz de confusión y las 4 métricas derivadas de la matriz, **accuracy, precision, recall, F1 score**

Aplicación

Estas fueron las métricas finales del modelo:

	Predicted	
	positives	**negatives**
actual positives	131	13
actual negatives	5	965
Metrics		
Accuracy	0.983842	
Precision	0.963235	
Recall	0.909722	
F1 Score	0.935714	

Ejemplos

idx	email (tokens)	spam	prediction
3	[u, dun, say, earli, hor, u, c, alreadi, say]	0	0
4	[nah, think, goe, usf, live, around, though]	0	0
7	[per, request, mell, mell, oru, minnaminingt...]	0	0
8	[winner, valu, network, custom, select, receiv...]	1	1
10	[gon, na, home, soon, want, talk, stuff, anymo...]	0	0
5551	[wen, get, spiritu, deep, great]	0	0
5555	[yeh, indian, nice, tho, kane, bit, shud, go, ...]	0	0

Generación de mensajes

A partir del modelo anterior, se generaron mensajes de spam y no spam con palabras aleatorias que son probables de encontrar en ambas bags of words. El resultado fue el siguiente:

Clase	`Mensaje (tokens)`
No spam	`sleep sm salam wait gim ah love hand librari thought usc hungri lot watch sian hundr what cheap mornin school`
Spam	`ur 1 xxx today com frnd 18p winner activ 69911 4 stop life get mind 2 capit 2optout xma phone`

Cuadro: Ejemplos de mensajes clasificados por el modelo.

Conclusiones

- El clasificador Naive Bayes resultó ser altamente efectivo para identificar correos spam con alta **precisión** y **recall**.
- El preprocesamiento de los emails, desde la limpieza, tokenizacion, el uso de stopwords y stemming fueron clave para la implementación y éxito del modelo.
- Para poder generar un email coherente se tendría que implementar procesos extra para aprender la relación de cada palabra con las demás, es decir, un bloque de atención, haciéndolo más parecido a un modelo de arquitectura transformer. En este caso unicamente se generaron aleatoriamente palabras de la clase spam o no spam sin ninguna relación una con la otra.
- Se experimentó con técnicas para mejorar el modelo como el suavizamiento de laplace y probabilidad logaritmica.