**CRISP-DM: Clasificación de SPAM del SMS usando ML**

## **Comprensión del Negocio**

Un ejemplo ilustrativo de la metodología vamos a considerar el caso de una organización telefónica especializada en SMS que pretende mejorar la precisión en el mensaje que recibe sus usuarios, si es un tipo SPAM o no.

Para ello se procederá a realizar una captura contextos del mensaje y se medirá de forma automatizada una serie de atributos sobre el contexto que nos permita clasificar el contexto si es **SPAM** (mensaje de promoción) o **HAM** (mensaje enviado por otro usuario/compañía).

El **objetivo final** es establecer un sistema automatizado que permita clasificar automáticamente un mensaje del usuario y complemente el contexto, mejorando la precisión mensaje; Usando los métodos **Naive Bayesian**, **SVM** y **Random Forest Classifier**

Este proyecto se utilizamos la herramienta **Jupyter-Notebook Python**, las librerías que necesitan son **Sklearn**, **Gensim**, y **Pandas**

## **Comprensión y Análisis de Calidad de los datos**

Usamos el dato público del [conjunto de datos del SPAM en SMS](https://archive.ics.uci.edu/ml/datasets/sms+spam+collection), que no es puramente limpio. El dato consiste en dos columnas una de la clasificación (SPAM o HAM) y el otro del contexto.

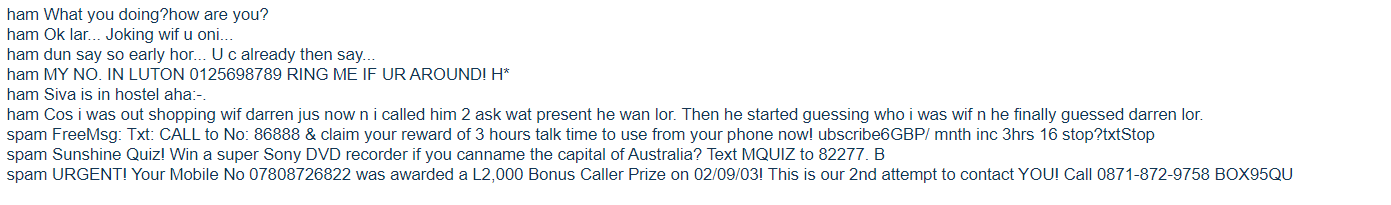


Ilustración 1 ejemplos del dato

Se procede a realizar un análisis estadístico básico. A continuación, se muestra el cálculo de la media, la desviación estándar y los cuartiles para las primeras nueve variables numéricas.

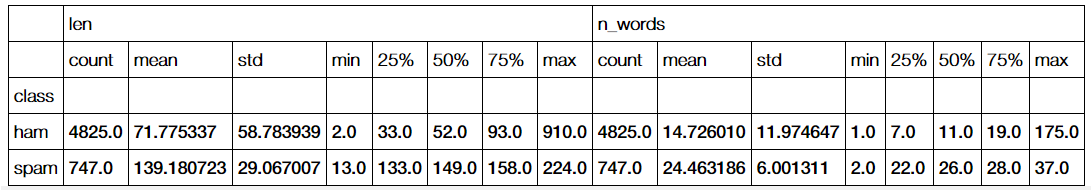


Ilustración 2 Análisis estadístico básico

Usando **Deep Learning**

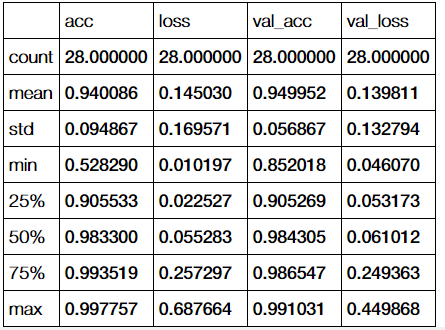


Ilustración 3 Análisis estadístico básico (Deep Learning)

## **Preparación de los datos**

Como ejemplo de exploración visual usamos grafico del Función de distribución (FDA) y Diagrama de caja (boxplot) para demostrar la distribución del SMS.

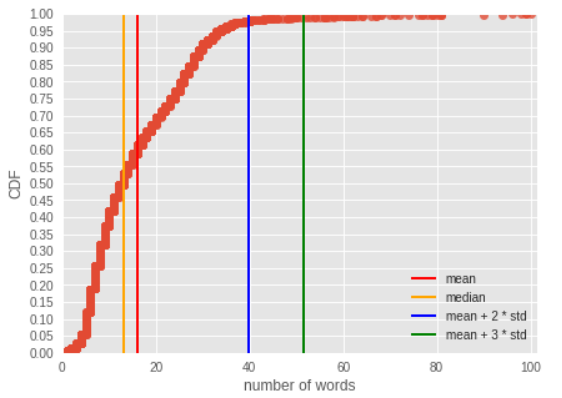


Ilustración 4 números de palabras

Según la ilustración 4:

* 50% del corpus consta de contextos que tienen menos de 13 palabras.
* 90% del corpus consta de contextos que tienen menos de aproximadamente 32 palabras.

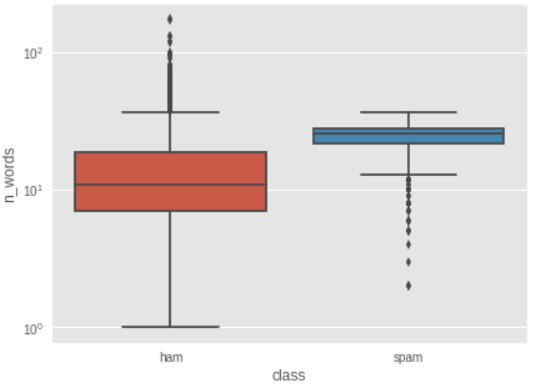


Ilustración 5 clasificación del SMS por números de palabras

Según ilustración 5:

* Las clases (HAM, SPAM) tienen una distribución identificada de forma única, claramente.

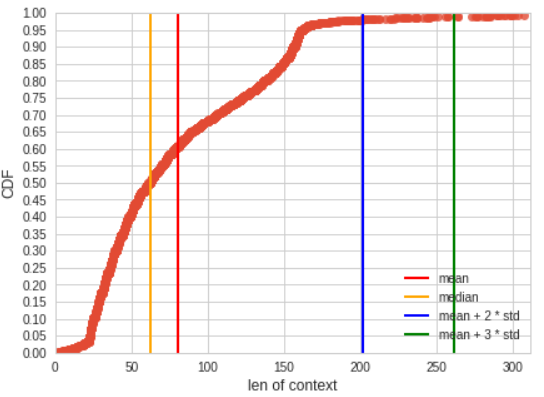


Ilustración 6 Longitudes del contexto

Según ilustración 6:

* El 50% del corpus consta de contextos cuya longitud es inferior a aproximadamente 62.
* El 90% del corpus consta de contextos cuya longitud es inferior a aproximadamente 155.

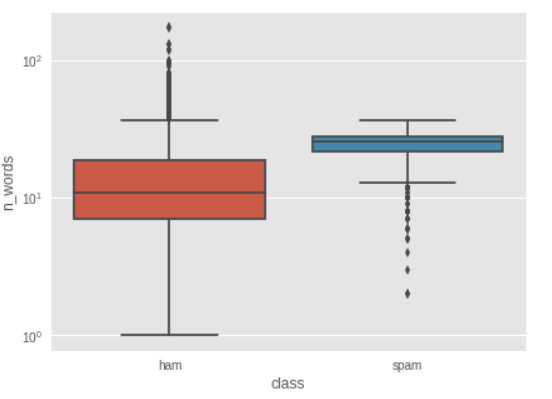


Ilustración 7 clasificación del SMS por longitudes de contexto

Según ilustración 7:

* Las clases (HAM, SPAM) tienen una distribución identificada de forma única, claramente.

## Modelado

Nos encontramos ante un problema de clasificación. Aplicaremos un método de aprendizaje supervisado, concretamente el clasificador **GridSearchCV**, para usar los hiper parámetros óptimos posteriormente en la etapa de prueba.

Después de seleccionar los hiper parámetros óptimos, cada modelo correspondiente a cada tubería se guardó en formato binario como un archivo **Pickle**. Para probar cualquier instancia en cualquier modelo, esos archivos binarios se pueden usar después de aplicar la deserialización, fácilmente

## **Predicción y Evaluación del Modelo**

Verificamos nuestra evaluación de desempeño utilizando una matriz de confusión. Usamos diferentes métricas, como **precisión**, **recuperación**, **puntaje f1** y **exactitud**.

También nos ocupamos del número de Verdadero Positivo (TP), Verdadero Negativo (TN), Falso Positivo (FP), Falso Negativo (FN) en la matriz de confusión.

NB está dando los mejores resultados de acuerdo con la métrica de precisión en las etiquetas de spam además de los datos de prueba. Porque su SMS de HAM no está etiquetado como SPAM en absoluto. La clasificación está funcionando completamente bien si nos enfocamos en el número de FP (HAM vs. SAPM). Nadie no quiere encontrar el SMS relevante (HAM) en su casilla de correo no deseado. Por lo tanto, la precisión del spam es importante para ese caso porque tiene la mejor métrica de precisión (1.00).

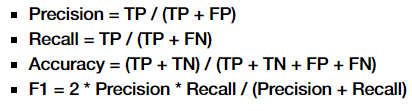


Ilustración 8 Formularios para SPAM vs HAM

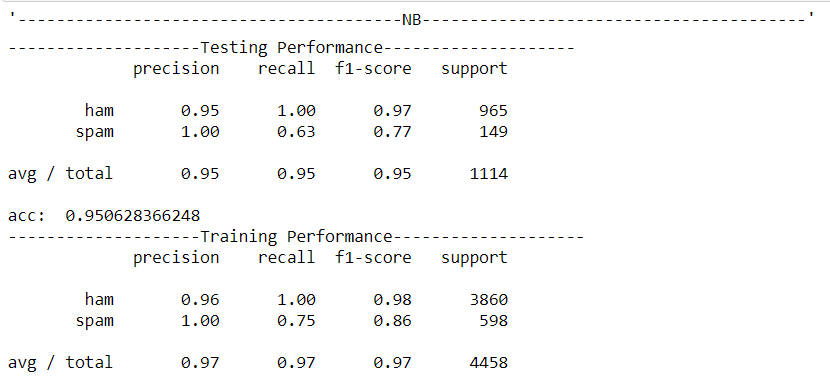


Ilustración 9 Evaluación por Naive Bayes classifiers

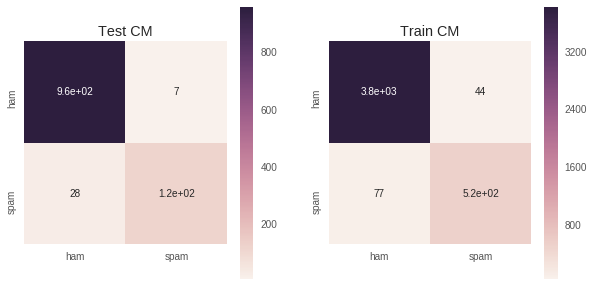


Ilustración 10 Matriz de confusión para NB

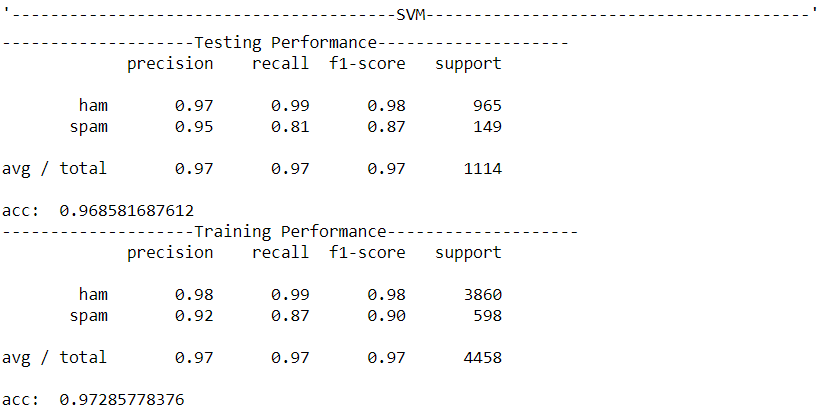


Ilustración 11 Evaluación por Máquinas de vectores de soporte

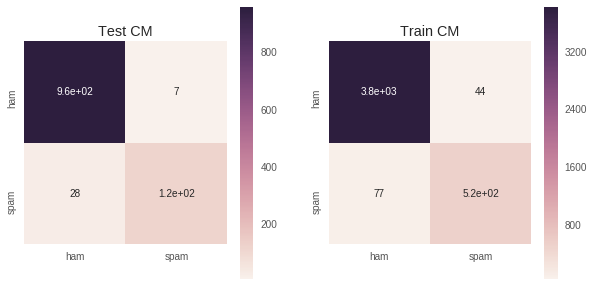


Ilustración 12 Matriz de confusión para SVM

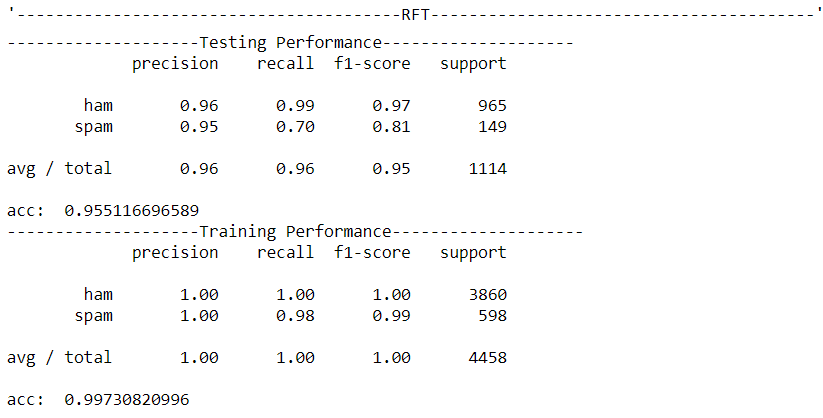


Ilustración 13 Evaluación por Random Forest

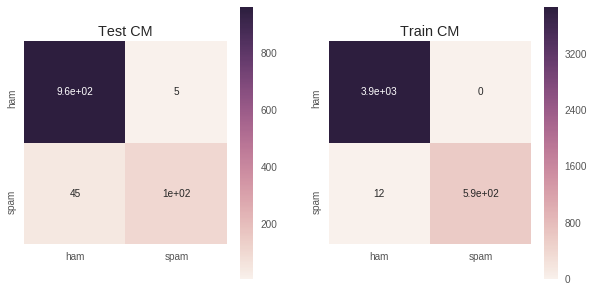


Ilustración 14 Matriz de confusión para Random Forest

## Despliegue e implementación

Para despliegue solo necesita entra **Jupyter-Notebook Python** y luego subir el archivo *model.ipynb*, luego de eso le generaría un archivo **Pickle**, y del **dataset**

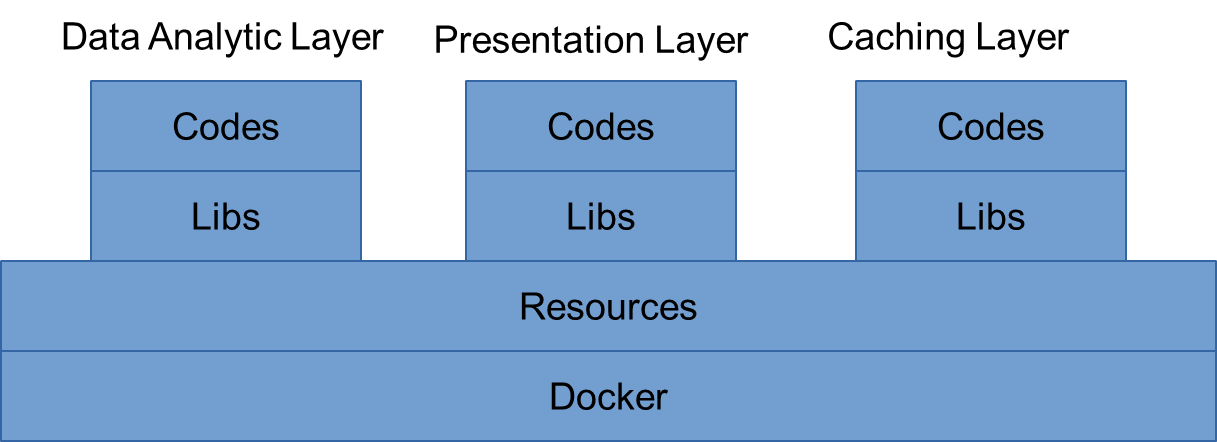


Ilustración 15 Arquitectura del proyecto

## **Gestión del Modelo**

El modelo se genera por cada modificación en repositorio ya que es un proyecto **Open Source**, todas las personas tienen permiso de contribuirlo.

## **Referencia**

[Repositorio en Github](https://github.com/erdiolmezogullari/ml-spam-sms-classification)