CentroSur
e-ISSN: 2600-5743
Social Science Journal

Centro Sur.
Social Science Journal
2021 – E1
http://centrosureditorial.com/index.php/revista
eISSN: 2600-5743
revistacentrosur@gmail.com

Análisis comparativo de modelos de aprendizaje automático para la detección de mensajes considerados como grooming online

Comparative analysis of machine learning models for the detection of messages considered as online grooming

Byron Oviedo Bayas [1]
Paúl Romero Alvarado[2]
José Montes Toapanta[3]
Luis Sánchez Zamora[4]

Resumen

Este articulo presenta un estudio comparativo entre algoritmos de aprendizaje automático para realizar la clasificación automática de texto considerado como grooming, para ello se utilizan distintas librerías del lenguaje de programación Python que permiten realizar el análisis y tratamiento del conjunto de datos, para el posterior entrenamiento de un modelo predictivo con estos mismos datos divididos en un grupo de entrenamiento y otro de validación con el fin de comprobar la precisión del modelo de clasificación. El principal objetivo de este estudio es determinar que algoritmo presenta un mejor rendimiento para la clasificación de este tipo de texto, comparando la eficacia de los algoritmos con el conjunto de datos mediante diferentes métricas de evaluación independientes. Como resultado de la comparativa entre los algoritmos de clasificación de texto se concluye que el modelo de Máquinas de Vectores de Soporte muestra un mejor rendimiento en cuanto a las métricas obtenidas obteniendo una precisión del 90%, además ofrece una mayor precisión para la clasificación de texto incluso con una alta dimensión de datos de entrenamiento y sin consumir muchos recursos de memoria.

[1] Facultad de Ciencias de la Ingeniería - Universidad Técnica Estatal de Quevedo, https://orcid.org/0000-0002-5366-5917, boviedo@uteq.edu.ec, Quevedo - Ecuador

[2] Facultad de Ciencias de la Ingeniería - Universidad Técnica Estatal de Quevedo, https://orcid.org/0000-0001-6548-9685, paul.romero2017@uteq.edu.ec, Quevedo - Ecuador

[3] Facultad de Ciencias de la Ingeniería - Universidad Técnica Estatal de Quevedo, https://orcid.org/0000-0003-4290-9338, jose.montes2017@uteq.edu.ec, Quevedo - Ecuador

[4] Facultad de Ciencias de la Ingeniería - Universidad Técnica Estatal de Quevedo, https://orcid.org/0000-0002-1445-2783, luis.sanchez2017@uteq.edu.ec, Quevedo -

Palabras Clave: Aprendizaje automático, algoritmos de clasificación, acoso sexual en línea, procesamiento de datos, Python.

Abstract

This article presents a comparative study between machine learning algorithms to perform the automatic classification of text considered as grooming, for which different libraries of the Python programming language are used that allow the analysis and treatment of the data set, for the subsequent training of a predictive model with these same data divided into a training group and a validation group to check the accuracy of the classification model. The main objective of this study is to determine which algorithm has a better performance for the classification of this type of text, comparing the efficiency of the algorithms with the data set through different independent evaluation metrics. As a result of the comparison between the text classification algorithms, it is concluded that the Support Vector Machines model shows better performance in terms of the metrics obtained, obtaining an accuracy of 90%, and also offers greater accuracy for text classification. even with a high dimension of training data and without consuming many memory resources.

Key words: Machine learning, classification algorithms, online grooming, data processing, Python.

Introducción

Las Tecnologías de la Información y Comunicación (TIC) se han aplicado en distintos ámbitos con el objetivo de lograr un beneficio para los usuarios en muchos aspectos de la vida cotidiana, sin embargo, existen peligros inherentes a la misma, las cuales afectan a la seguridad de los usuarios en Internet, siendo vulnerables a muchas amenazas que incluyen el ciberbullying, sexting y grooming las cuales tienen como principales víctimas a los menores de edad con el fin de ganarse su confianza para conseguir imágenes eróticas de la víctima y satisfacer sus necesidades sexuales [1].

El incremento de los dispositivos móviles inteligentes en las últimas décadas, han hecho que el acceso a Internet sea mucho más sencillo, por lo que hace complicado a los padres vigilar la actividad que realizan sus hijos en línea. El uso malintencionado de las TIC por parte de los ciberdelincuentes siempre va a existir, por lo tanto, los menores de edad que navegan libremente por Internet sin la supervisión de su tutor responsable corren el riesgo de ser damnificados de estos atacantes y sus oscuras intenciones. Por consiguiente, si no se toma las medidas

necesarias para la prevención de estos actos, puede desencadenar muchos problemas dentro de la familia e incluso en la sociedad.

El procesamiento del lenguaje natural y el aprendizaje automático son solo dos ramas más de lo que viene siendo la inteligencia artificial. Tal vez suene novedoso, pero lleva más tiempo de lo que pensamos, como ejemplo de este podemos pensar en el famosa prueba de Turing; el cual en resumen consistía en probar la capacidad de una máquina para hacer ostensible un comportamiento inteligente análogo al de una persona. Conforme el avance tecnológico acrecienta se amplían los métodos para el análisis de textos con un poder de procesamiento del lenguaje cada vez mayor, logrando desarrollar algoritmos de inteligencia artificial para diversas tareas de clasificación [2], [3].

Algunas de las aplicaciones que giran entorno a este campo son: autocompletados, generación automática de comentarios, asistentes virtuales, sugerencia de contenido, etc. Investigaciones similares realizadas anteriormente se encuentra la de [4] en esta tesis se desarrolló un complemento para el navegador de Google Chrome que toma los mensajes recibidos de la aplicación de Facebook y los envía al clasificador el cual verificará si el contenido del mensaje posee o no grooming. La precisión promedio del modelo desarrollado en este trabajo es del 94%, considerada como satisfactoria. Además, se concluyó que la extracción de datos o información personal de los sitios web en tiempo real es una técnica de hacking ético, por lo cual para obtener las conversaciones del menor se debe tener autorización por parte del tutor de este.

Otra similar es la de [5] en este trabajo de investigación tiene como objetivo la detección inmediata de ciberacoso y brindar protección al usuario comunicando a un tutor responsable para que no ocurran desenlaces fatales. Utilizando algoritmos para como el Análisis Semántico Latente para el análisis de la información de redes sociales del adolescente y determinar si es víctima de ciberacoso, generando una alerta en caso de que sea así, de modo que pueda tomar las acciones necesarias para salvaguardar su integridad y bienestar.

Este comportamiento pedófilo en internet presenta un crecimiento peligroso para la infancia. Los gobiernos en su interés por controlar este delito han usado muchas estrategias, dentro de las cuales se destacan dos: la primera es usar páginas ficticias que hacen el papel de señuelo para detectar usuarios pedófilos, y la segunda utilizar agentes de policía, que haciéndose pasar por niños logran atraer y engañar a los pedófilos para después arrestarlos, pero estas soluciones requieren

de recursos económicos y mucho personal calificado, aun así no son del todo eficientes, ya que este tipo de delincuentes pueden crear nuevos métodos para lograr sus objetivos [3].

Tras analizar la gravedad de las consecuencias que puede causar el grooming, se realiza la propuesta de este proyecto el cual consiste en desarrollar una aplicación informática, utilizando el lenguaje de programación Python por su sencillez y escalabilidad, que permita la detección de contenido que sea considerado como grooming en aplicaciones de mensajería instantánea, usando técnicas de procesamiento de lenguaje natural (NLP) combinado con algoritmos de Machine Learning (ML) para la clasificación automática del texto.

Materiales y métodos

El presente estudio se realizó bajo un tipo de investigación descriptiva, debido que, para iniciar con el desarrollo de este, primero se realizó un análisis exhaustivo de la información pertinente, recopilado de artículos científicos, libros, trabajos realizados y demás, para posteriormente determinar los objetivos puntuales, las limitaciones y resultados que se puedan llegar a obtener.

El desarrollo del proyecto está dividido en tres fases sustanciales:

Fase 1: Estudio e investigación

En esta fase se centró en recolectar a través de fuentes confiables toda la información necesaria para determinar el estado del arte de lo que se plantea realizar. En base a esta información se identificarán los requerimientos funcionales y las tecnologías necesarias para la correcta elaboración del proyecto. Para llevar a cabo el cumplimiento de esta fase se llevaron a cabo las siguientes actividades:

- Estudio del tema;
- Análisis de la problemática;
- Elección de tecnologías.

Fase 2: Documentación

Teniendo en cuenta la información recopilada de la fase anterior se procede a estructurar las ideas en la redacción del documento. Además, se elaboraron los diferentes diagramas y esquemas en los que se plasma gráficamente la forma en que funciona el sistema.

- Redacción del documento;
- Conclusiones y recomendaciones;
- Correcciones finales.

Fase 3: Desarrollo del software

De acuerdo con el diseño del sistema realizado en la fase anterior, se inicia el desarrollo del software teniendo en cuenta los requerimientos y estructura del mismo definido anteriormente. En esta fase se realiza la codificación en un lenguaje de programación un modelo clasificador de texto mediante técnicas de Machine Learning siguiendo los procesos definidos en los diagramas para cumplir con el objetivo definido.

- Análisis de datos;
- Diseño de modelo;
- Elección del mejor modelo;
- Pruebas y ajustes.

Las herramientas y tecnologías que se utilizaron son las siguientes:

Anaconda: IDE para el desarrollo de aprendizaje automático con Python.

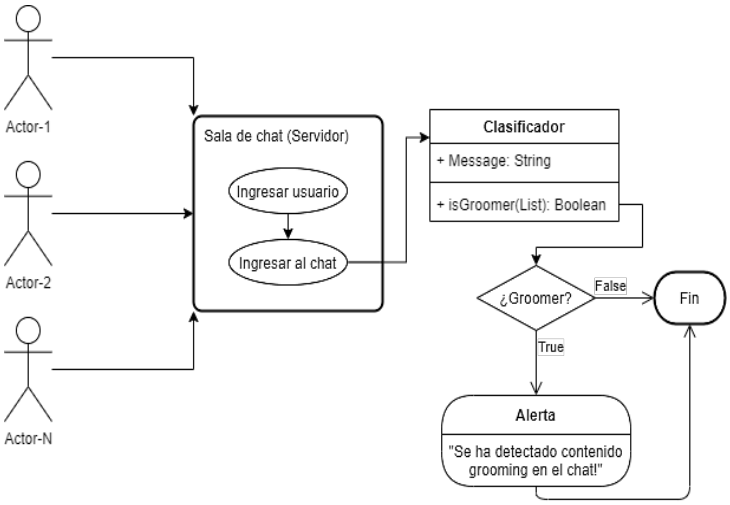
NLTK: librería para las técnicas de procesamiento del lenguaje natural.

Scikit-learn: Librería para los algoritmos de aprendizaje automático.

Flask-SocketIO: Framework para el desarrollo de la aplicación web y envío de mensajes.

Como se mencionó anteriormente, se optó por desarrollar un sitio web que permita la comunicación entre dos o más usuarios, una sala de chat, la misma que servirá para poder comprobar el funcionamiento del sistema clasificador de textos, siguiendo la estructura como se muestra en la figura 1.

Figura 1. Esquema funcional del sistema.



Recolección de datos

Para iniciar la construcción del modelo de clasificación primeramente se debe tener preparado un conjunto de datos, el cual se ingresa al modelo clasificador para entrenarlo, es decir, que pueda aprender de ejemplos del pasado [6]. Hay que tener en cuenta que, en la adquisición de datos, la selección de características y el etiquetado de los datos de entrenamiento constituyen la tarea más minuciosa para obtener un modelo de predicción competente.

Para el proceso de recolección de los datos, se codificó un script en Python para la descarga de 215 conversaciones disponibles en el sitio web www.justicedperverted.com, esta organización colabora con el estudio de conversaciones entre pedófilos mediante la publicación continua de conversaciones reales en su plataforma web, con el propósito de erradicar a estos depredadores en línea exponiendo sus conversaciones.

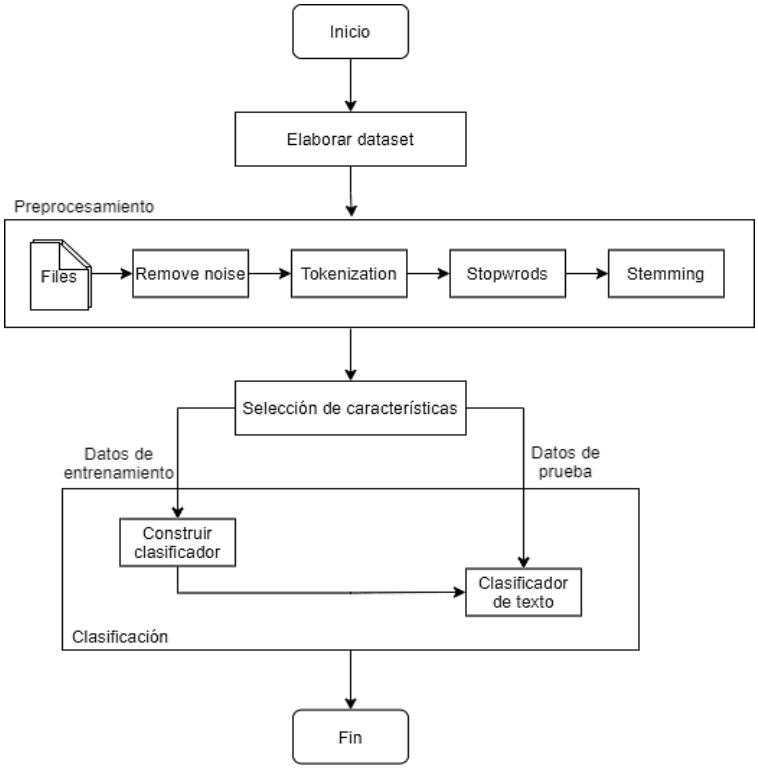
Preprocesamiento de datos

En comparación con el uso de un conjunto de datos completo, eliminar el ruido en los datos no solo mejora el rendimiento del algoritmo, sino que también ayuda a aumentar la confianza y obtener mejores resultados de este, con el fin de reducir tiempo y complejidad al entrenar el modelo, para hacerlo más fácil de interpretar y reducir el sobreajuste. Durante la etapa de preprocesamiento se descartó cualquier texto que no aportaría información al proceso de clasificación, tales como registros de la hora y fecha de la conversación, anotaciones del chat y nombres de usuario, dejando simplemente los mensajes del usuario catalogado como groomer, los cuales pasarán luego por una etapa de filtrado para eliminar caracteres especiales, caracteres

numéricos, stopwords y demás datos innecesarios para la clasificación, puesto que las conversaciones no siguen reglas gramaticales, por lo que además de ser informales y gramaticalmente desestructuradas, los errores ortográficos frecuentes y el uso común de acrónimos y emoticonos son comunes. Esto hace que la detección se vuelva un desafío debido a los falsos positivos que puede generar el clasificador. Por esta razón, se requirió de un arduo trabajo de purificación del texto.

La figura 2 muestra el orden que se siguió para la construcción del modelo de clasificación de texto.

Figura 2. Procedimiento del trabajo.



Teniendo listo los datos pertenecientes a la clase groomer con un total de 11500 mensajes de texto, se busca otro conjunto de datos de conversaciones de tipo no groomer, donde obtuvimos un total de 7800 mensajes. Luego formamos un solo dataset con ambos archivos de diferentes clases (1: groomer y 0: no groomer) obteniendo un total de 19300 mensajes para proceder con la clasificación.

Figura 3. Archivo no estructurado de un chat.

```
er1ca_leah_cx (07/08/14 10:27:24 PM): hey were back finally
jackjohnsons7 (07/08/14 10:28:08 PM): you want this cock baby
er1ca_leah_cx (07/08/14 10:28:48 PM): huh
jackjohnsons7 (07/08/14 10:29:04 PM): you want to watch me jack off
er1ca_leah_cx (07/08/14 10:29:10 PM): well I wanted 2 come online so u stop complaining
jackjohnsons7 (07/08/14 10:30:20 PM): you like my big hard cock baby
er1ca_leah_cx (07/08/14 10:30:33 PM): lol yea looks big
jackjohnsons7 (07/08/14 10:30:55 PM): how do you want me to stroke it
er1ca_leah_cx (07/08/14 10:31:21 PM):
jackjohnsons7 (07/08/14 10:31:35 PM): you want this cum dont you
```

Figura 4. Archivo estructurado de un chat.

```
Text,Groom
peace and love sexy,1
i will have to get directions from you before i would come down,1
i will talk to you later though okay sexy,1
im sure your you are best will be good enough,1
im gettin excited just thinking about it,1
well your you are exciting me,1
abraxis is the latin god of balance its from a book,1
i do not think i spelt that right,1
hello there my sweet,1
do you dress sexy sometimes,1
```

Siguiendo con la etapa de procesamiento, es fundamental realizar un análisis de la frecuencia de las palabras para evitar la existencia de palabras no relacionadas con el contexto de la conversación.

En las figuras 5 y 6 se muestran nubes de palabras, donde las palabras con mayor tamaño indican una mayor frecuencia en el documento y las de menor tamaño lo contrario, las cuales corresponden a su forma antes y después de la etapa de procesamiento respectivamente.

Figura 5. Texto sin procesar.

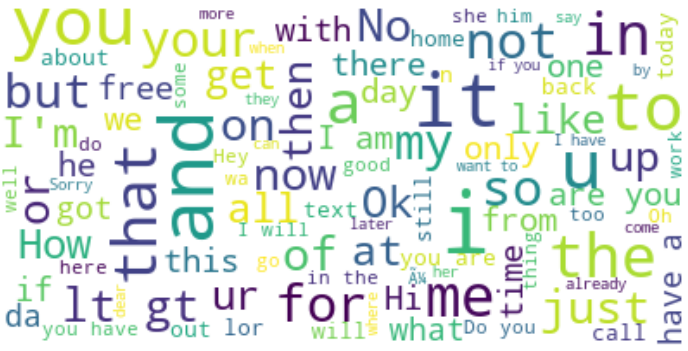
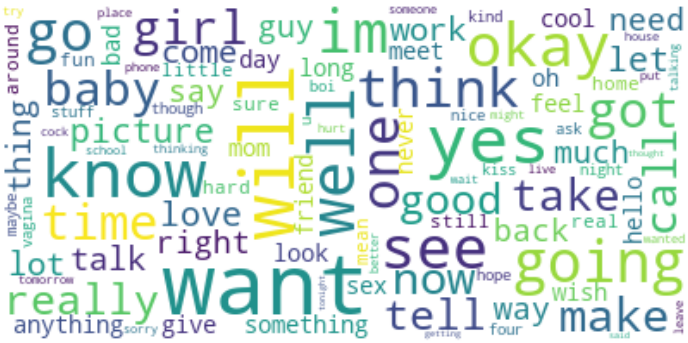


Figura 6. Texto procesado.



Luego de haber realizado la etapa de preprocesamiento del conjunto de datos elaborado, empezamos con la preparación del modelo clasificador, para ello, se analizaron tres de los algoritmos de aprendizaje supervisado con mejores cualidades para la clasificación de texto en el aprendizaje automático, siendo estos Naive Bayes, Máquinas de Vectores de Soporte y el algoritmo de Bosques Aleatorios.

Estos algoritmos de aprendizaje automático toman como entrada vectores numéricos para el entrenamiento y la clasificación de estos datos, entonces, con el conjunto de datos se realiza un proceso de normalización para preparar estos datos antes de ingresar al clasificador, con el objetivo de transformar estos datos categóricos a numéricos, sin distorsionar ni perder la información [7].

Selección de características

Para vectorizar el texto se puede hacer uso de vectores de palabras, donde cada palabra del corpus se asigna a un vector de alta dimensión, las palabras que aparecen en el mismo contexto suelen tener vectores de palabras muy similares, lo que ayuda a que los clasificadores sean mejores para generalizar [8].

El algoritmo que se utilizó para la vectorización fue TF-IDF, el cual consiste en medir la frecuencia de un término o frase de un determinado documento en una colección de documentos, donde la frecuencia del término TF, para el término i en el documento j se define cómo:

$$TF(i,j) = \frac{frecuencia\ de\ i\ en\ documento\ j}{palabras\ totales\ en\ documento}$$

Y la frecuencia inversa del documento IDF está definida como:

$$IDF(i) = \log_2\left(\frac{documentos\ totales}{documentos\ con\ el\ término\ i}\right)$$

Finalmente, la medida TF-IDF se obtiene multiplicando TF con IDF.

$$TF\text{-}IDF(i,j) = TF(i,j) * IDF(i)$$

Considerando el término ocurrencia del número de documentos, las ponderaciones TF-IDF de palabras ridículamente comunes son bajas, obteniendo entonces las características más importantes del documento en todo el conjunto de datos. Este proceso de normalización contribuyó a la optimización del modelo en el cálculo y procesamiento que conlleva su entrenamiento.

Naive Bayes

Es uno de los modelos de clasificación más usados y sencillos, este se basa en dos supuestos: uno es que cada propiedad es condicionalmente independiente de una propiedad específica de la clase, y la otra es que todos los atributos afectan a dicha clase. El teorema de Bayes establece la siguiente relación, dada una variable de clase y y vector de características dependientes x_1 mediante x_n [9]:

$$P(y|x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n|y)}{P(x_1, \dots, x_n)}$$

Usando el supuesto ingenuo de independencia condicional de que:

$$(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y)$$

Para todo i, esta relación se simplifica:

$$P(y|x_1, \dots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_1, \dots, x_n|y)}{P(x_1, \dots, x_n)}$$

Support Vector Machines

Es un poderoso y versátil algoritmo de aprendizaje automático, muy usado en problemas de clasificación binaria, particularmente adecuado para la clasificación de un conjunto de datos complejo de pequeño a mediano tamaño[10].

Una máquina de vectores de soporte construye uno o un conjunto de hiperplanos en un espacio dimensional enorme para realizar la clasificación. El hiperplano logra una buena separación con la distancia máxima al punto de datos de entrenamiento más cercano en cualquier clase, cuanto mayor sea el margen de separación, menor será el error de generalización del clasificador [9].

Para el entrenamiento, dado m vectores de entrada al modelo, la función objetivo es maximizar [11]:

$$W(\propto) = \sum_{i=1}^{m} \propto_i - \frac{1}{2}\sum_{j=1}^{m} \propto_i \propto_j y_i y_j (x_i \cdot x_j) \;\; \propto_i \geq 0, \sum_{i=1}^{m} \propto_i y_i = 0$$

Los puntos de datos con valores alfa superiores a 0 son vectores de soporte y afectan el comportamiento del hiperplano divisor.

Random Forest

Es otro poderoso modelo de aprendizaje automático formado por muchos árboles de decisión, otro algoritmo de aprendizaje. Random Forest entrena a cada árbol en un conjunto ligeramente diferente, dividiendo los nodos en cada árbol considerando un pequeño número de características. Las predicciones del bosque aleatorio se realizan promediando cada predicción de cada árbol individual [10].

La importancia total de las características de cada árbol se calcula y se divide por el número total de árboles.

$$RF(i) = \frac{\sum_{i=1}^{T} \widehat{f_{ij}}}{T}$$

Donde $\widehat{f_{ij}}$ es la importancia de la característica normalizada para i en el árbol j, y T correspondiente al número total de árboles. Usando el bosque aleatorio para la clasificación, cada árbol proporciona una estimación de probabilidad de la etiqueta de clase y las probabilidades se promedian sobre el árbol n para predecir el rendimiento más alto en la etiqueta de clase [10].

Scikit-Learn

Todos estos algoritmos mencionados los podemos encontrar en la librería Scikit-Learn de Python, la cual ofrece una manera sencilla de trabajar con estructuras de datos y el aprendizaje automático, por lo que es adecuada para crear modelos predictivos, como es en este caso para la clasificación del texto.

Lo siguiente es dividir el conjunto de datos en dos partes, una para el entrenamiento, es decir la información de la que va a aprender el clasificador, y la de prueba, con la que el modelo va a tratar de predecir y verificar si se logró un buen trabajo al clasificar el texto. Es recomendable dividir los datos en una proporción del 70-80% para alimentar el modelo y 20-30% para prueba, esto va a depender de la cantidad de datos que se tenga.

Resultados

Luego de realizar el entrenamiento de los modelos de clasificación, es obtienen los resultados de cada una de sus métricas de evaluación. Una buena forma de evaluar los modelos es usar la técnica de validación cruzada, en nuestro caso usamos el modelo K-Fold, el cual consiste en dividir el conjunto de entrenamiento en K partes o

pliegues, comúnmente en 5 o 10 pliegues. Para luego realizar la predicción y evaluar cada pliegue utilizando un modelo entrenado [12].

Como resultado de las 5 iteraciones, obtenemos un vector con la puntuación de la precisión para cada iteración del conjunto de datos de entrenamiento, las cuales poseen una precisión aproximadamente similar indicando que el modelo está funcionando correctamente.

Posteriormente promediamos estos valores y los dejaremos para cada clasificador de la siguiente manera: 87.58% para Naive Bayes, 88.28% para SVM y 87.09% para el algoritmo Bosques Aleatorios, esto indica que tan bien están funcionando los algoritmos e intuitivamente se deduce que el modelo con mejor desempeño es el SVM.

Es una tabla que describe gráficamente el desempeño de un modelo de clasificación en un conjunto de datos de prueba cuyos valores verdaderos son conocidos [6]. Esta matriz nos permite visualizar el nivel de acierto del modelo, mostrando cuántos casos se clasificaron correcta o incorrectamente.

- Verdadero positivo (True Positive, TP): número de clasificaciones correctas en la clase positiva, dato real 1 y predicción 1.
- Verdadero negativo (True Negative, TN): número de clasificaciones correctas en la clase negativa, dato real 0 y predicción 0.
- Falso negativo (False Negative, FN): número de clasificaciones incorrectas de la clase positiva clasificada como negativa, dato real 1 y predicción 0.
- Falso positivo (False Positive, FP): número de clasificaciones incorrectas de la clase negativa clasificada como positiva, dato real 0 y predicción 1.

En las figuras 7 a 9 se presentan las matrices de confusión correspondientes a cada modelo entrenado.

Figura 7. Matriz de confusión para el clasificador Naive Bayes.

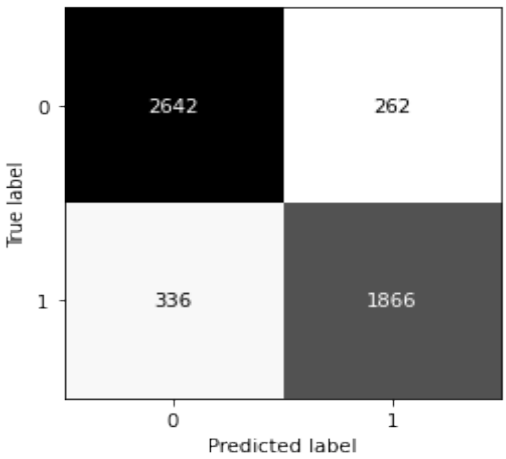


Figura 8. Matriz de confusión para el clasificador SVM.

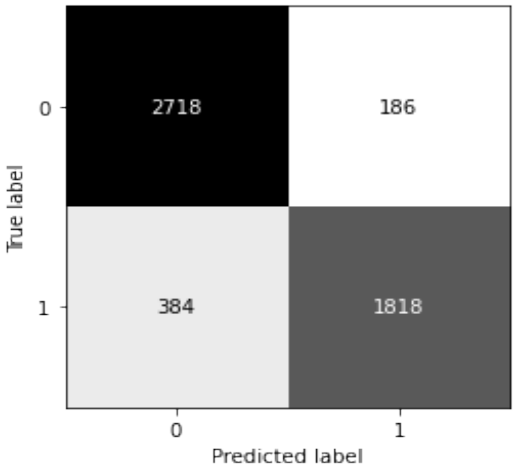
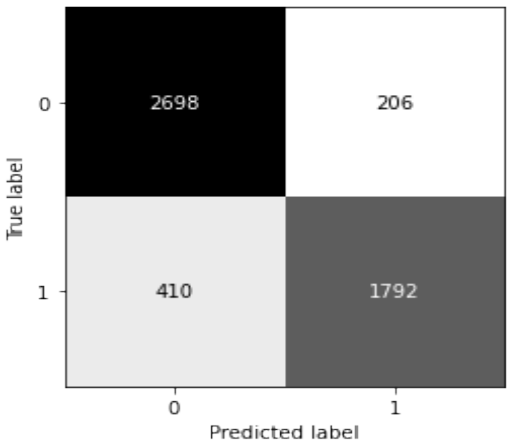


Figura 9. Matriz de confusión para el clasificador Random Forest.



Como se observa en las tres figuras, cada una presenta valores desiguales, y haciendo el análisis de la diagonal principal la figura 8 referente al clasificador SVM presenta mejores resultados en comparación a las otras dos.

Las métricas para realizar la evaluación de un modelo dependen principalmente del tipo de problema a resolver, en este sentido, consideraremos las métricas específicas para problemas de clasificación.

Exactitud: Es la relación entre las predicciones correctas y el número total de predicciones.

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN}$$

Precisión: Mide el rendimiento relacionado con las tasas de verdaderos positivos y negativos.

$$precision = \frac{TP}{TP + FP}$$

Sensibilidad: Es la relación entre las predicciones positivas correctas y el número total de predicciones positivas.

$$recall = \frac{TP}{TP + FN}$$

Especificidad: Es la relación entre las predicciones negativas correctas y el número total de negativos.

$$espicificity = \frac{TN}{TN + FP}$$

Puntaje F1: Es la medida armónica entre la sensibilidad y la precisión, una puntuación alta indica un mejor desempeño del modelo.

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

Del mismo modo, utilizamos las funciones para calcular las métricas de evaluación para cada modelo, las mismas que se presentan en la tabla 1:

Tabla 1. *Resultados de la evaluación para cada clasificador.*

Algoritmo	Precisión	Sensibilidad	Puntaje F1	Error absoluto medio
NB	87.69%	84.74%	86.19%	11.71%
SVM	90.72%	82.56%	86.45%	11.16%
RF	89.69%	81.38%	85.35%	12.06%

Estos resultados que se muestran en la tabla 1 confirman mediante la puntuación F1 que el modelo de clasificación con mejor desempeño es el de Máquinas de Vectores de Soporte, con un puntaje algo superior en comparación a los algoritmos Naive Bayes y Bosques Aleatorios. Además, el resultado del error absoluto medio, el cual es el promedio de la diferencia entre el valor real y el predicho, mientras menor sea indica que el modelo tiene un buen ajuste ante la precisión con la que el modelo predice y el modelo SVM tiene el menor puntaje.

Una curva ROC (Receiver Operating Characteristc) mide el rendimiento respecto a la tasa de falsos positivos y verdaderos positivos, esta métrica nos dice que tan bien distingue el modelo entre dos clases [13]. También podemos usar el área bajo la curva (AUC) para comparar los clasificadores, un clasificador ideal tendrá un ROC-AUC igual a 1, mientras que un clasificador aleatorio tendrá una puntuación igual a 0.5 [10].

Por último, obtenemos el puntaje ROC-AUC y presentamos las gráficas de la curva para cada algoritmo clasificador:

Figura 10. Puntaje y curva ROC-AUC para el clasificador Naive Bayes.

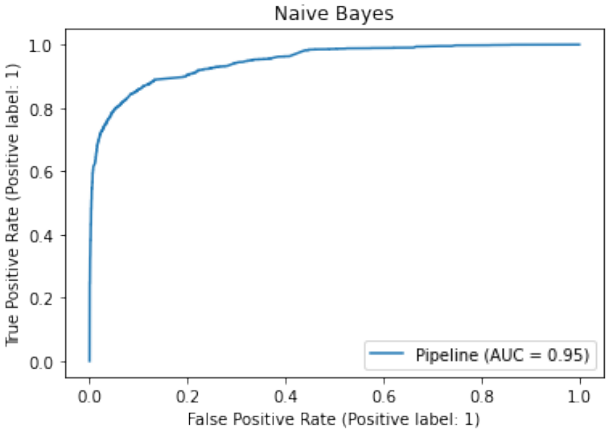


Figura 11. Puntaje y curva ROC-AUC para el clasificador SVM.

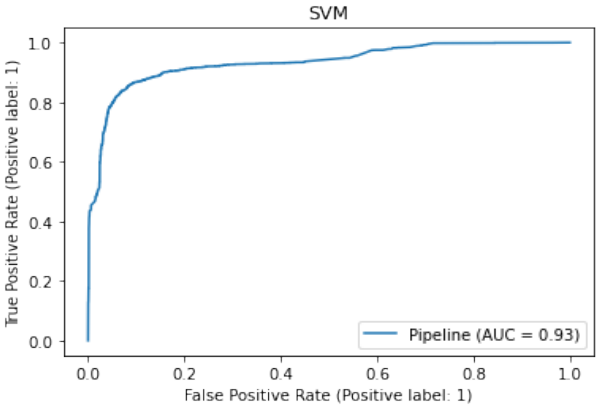
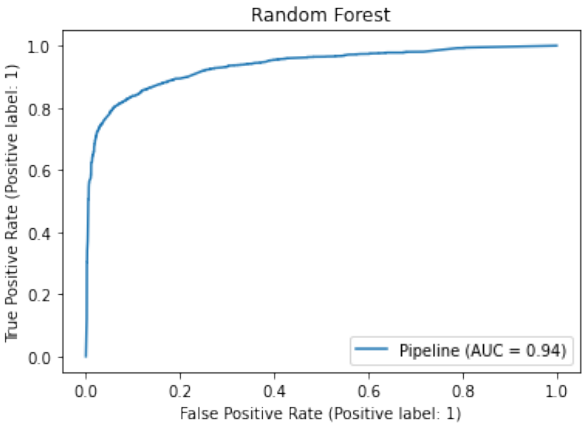


Figura 12. Puntaje y curva ROC-AUC para el clasificador Bosques Aleatorios.



En las figuras 10 a 12 se muestra la curva y la puntuación de la clasificación para cada modelo, mostrando una gran eficacia el modelo Naive Bayes en la clasificación

correcta del texto con un total del 95%, seguido de Bosques Aleatorios con un 94% y SVM con 93%.

Llegando a este punto, finalizamos con la evaluación de los modelos y se puede decir que los algoritmos Naive Bayes y Bosques Aleatorios han tenido una gran eficacia para la clasificación del texto. Sin embargo, el algoritmo de Máquinas de Vectores de Soporte fue el que más destacó durante el proceso de evaluación mostrando mejores resultados de eficiencia.

Para concluir en la elección del modelo de clasificación, se procede a probar individualmente cada clasificador con un texto de prueba para observar cual logra clasificar correctamente el mensaje que se le introduce.

Tabla 2. *Prueba de clasificación con modelo Naive Bayes.*

Etiqueta	Mensaje	Predicción	Confianza
0	I'm going to play basketball in the park	0	60.2%
1	I unchap your brassiere and let it fall to the floor	1	85.4%

Tabla 3. *Prueba de clasificación con modelo SVM.*

Etiqueta	Mensaje	Predicción	Confianza
0	I'm going to play basketball in the park	0	62.5%
1	I unchap your brassiere and let it fall to the floor	1	98.7%

Tabla 4. *Prueba de clasificación con modelo Random Forest.*

Etiqueta	Mensaje	Predicción	Confianza
0	I'm going to play basketball in the park	0	51%
1	I unchap your brassiere and let it fall to the floor	1	79%

Se comprobó el nivel de precisión de cada algoritmo al clasificar un mensaje positivo, es decir perteneciente a la clase Groomer, teniendo resultados muy favorables en cada uno. De igual forma, el algoritmo SVM es el que presenta mayor confianza, es decir, mayor probabilidad de que el mensaje pertenezca a una clase, la cual está 98.73% seguro de que el mensaje corresponde a la clase Groomer, mientras que el algoritmo Naive Bayes predice una probabilidad de 85.44% y Bosques Aleatorios un 79%.

Para probar el funcionamiento del modelo de clasificación de texto, se elaboró una aplicación web básica de mensajería. El funcionamiento de esta aplicación está basado totalmente en el protocolo Socket.IO, el cual permitió una comunicación bidireccional basada en eventos en tiempo real entre clientes y un servidor.

Tal como se muestra en la figura 13, se diseñó una página para que el usuario ingrese un nombre al momento de entrar a la sala de chat del sitio web.

Figura 13. Página de ingreso a la sala de chat.

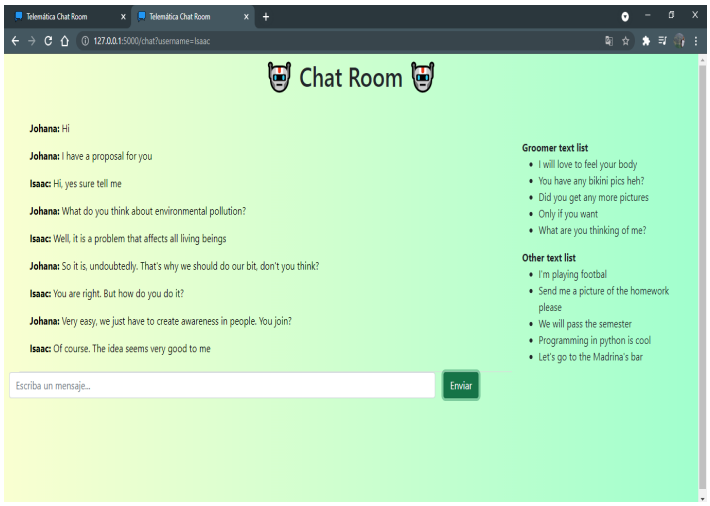
Una vez que el usuario ingresa a la sala de chat puede chatear con cualquier otro usuario que se encuentre en la sala. Dentro de la sala el modelo de clasificación creado en secciones anteriores tomará como entrada cada mensaje que reciba el servidor desde el cliente, durante este análisis se consideró utilizar como función la predicción probabilística, ya que esta nos devuelve un valor entre 0.0 y 1.0, podemos asegurarnos de que si la probabilidad de que pertenezca o no a la clase Groomer con un valor mayor a 0.8 y así evitar falsos positivos al momento de generar la alerta.

Finalmente, se elaboró un total de 4 experimentos para observar en cuantos de estos escenarios de conversaciones logra clasificar correctamente el sistema.

Escenario 1 – Conversación con tema del medio ambiente

Para el primer escenario se probó el clasificador con una muestra de un chat sobre la contaminación del medio ambiente.

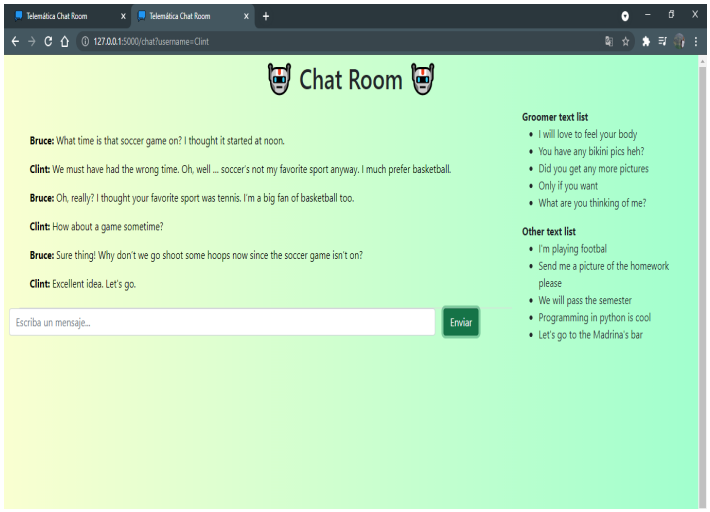
Figura 14. Chat escenario 1.



Escenario 2 – Conversación con tema de deportes

En el escenario 2, se tiene un chat sobre temas de deportes.

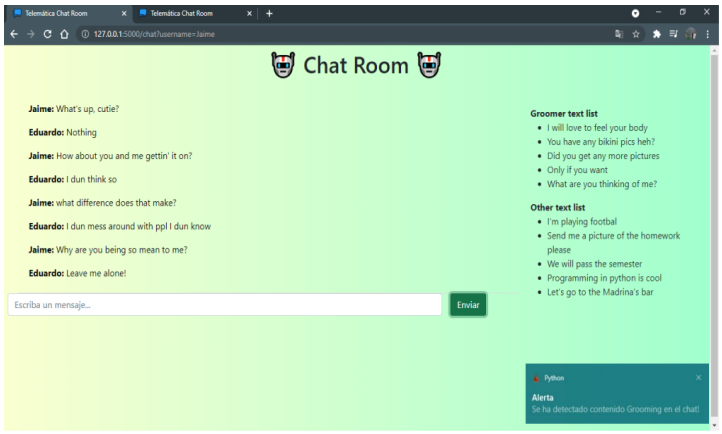
Figura 15. Chat escenario 2.



Escenario 3 – Conversación con tema de acosamiento

Para este caso, debido al tipo de mensajes, el clasificador debería de considéralos y generar una alerta de igual manera, tal como se observa en la figura 16.

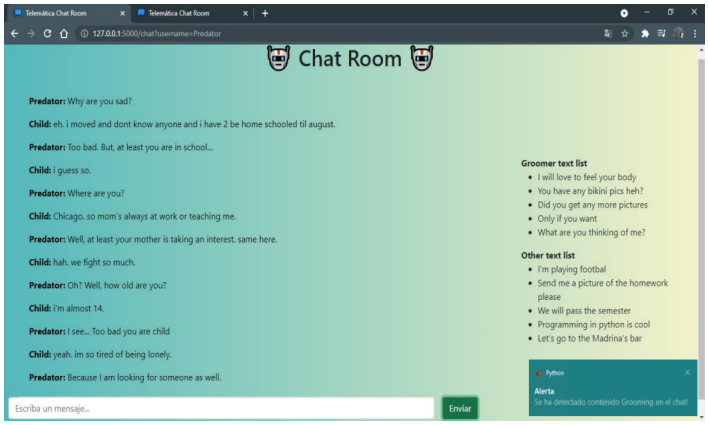
Figura 16. Escenario chat 3.



Escenario 4 – Conversación con tema pedófilo

Para el último caso disponemos de una muestra de un chat entre un pedófilo y una pseudo-victima, una persona que posa como cebo para la conversación.

Figura 17. Chat escenario 4.



El clasificador tomó individualmente cada mensaje para el respectivo análisis, por lo que para medir el rendimiento del clasificador ante este y los demás escenarios su utilizó la siguiente ecuación.

$$rendimiento = \frac{mensajes\ clasificados}{cl.correcta + cl.incorrecta}$$

La siguiente tabla muestra los resultados de clasificación para cada mensaje de cada escenario. La columna mensajes clasificados muestra la cantidad de mensajes que clasificó como grooming el modelo y la siguiente columna muestra cantidad de mensajes de tipo grooming que existen en ese chat. Por lo tanto, la primera fila quiere

decir lo siguiente: En el escenario uno el modelo clasificó incorrectamente 4 mensajes ya que existen 0 mensajes de tipo grooming en el chat, y la última columna muestra la precisión del modelo con respecto a la clasificación en cada escenario.

Tabla 5. *Resultados de precisión en cada escenario.*

Escenario	Mensajes clasificados	Mensajes clasificar	Mensajes totales	Rendimiento
1	4	0	9	55%
2	1	0	6	83.33%
3	4	4	8	100%
4	8	7	13	87.5%

Discusión

Se desarrolló y entrenó un modelo clasificador de textos el mismo que se implementó en una sala de chat, este programa permite la clasificación automática de los mensajes de texto de una conversación y determinar si posee una intención maliciosa con fines sexuales, acorde a los objetivos que se plantearon.

De acuerdo a otras investigaciones relacionadas como [14] y [15] los resultados en cuanto a precisión del modelo son de 98% y 97% respectivamente. Mientras que para nuestro modelo con el algoritmo SVM se logró una precisión del 90.72%, esto puede deberse al recorte del conjunto de datos que se realizó para balancear los datos o en el proceso de selección de características, de cualquier forma, este resultado es bastante bueno para nuestro clasificador.

En cuanto al análisis del rendimiento en general del clasificador, se puede observar los resultados obtenidos en el tabla 5, logrando una gran precisión cuando el tema de la conversación va por un sentido grooming o relacionado como es en el caso de los escenarios 3 y 4. Sin embargo, para el caso 1 no se obtuvo muy buenos resultados en la clasificación, una forma de solucionar o mejorar este proceso de clasificación podría ser almacenar los mensajes en una cola para luego obtener un promedio referencial del modelo para ir clasificando según el contexto de la conversación y así evitar generar falsas alarmas.

Como trabajo futuro se pretende conseguir o crear un conjunto de datos en idioma español para realizar la clasificación de los mensajes de texto entre personas hispanohablantes, puesto que en países de Latinoamérica es donde se registran mayores casos de actividades pedofilias en Internet, y Ecuador no es la excepción. Además, de buscar una manera de integrar el software a una aplicación de mensajería real que se utilice habitualmente.

Conclusiones

Después de estudiar algunos de los algoritmos de clasificación de texto en Machine Learning, se eligieron tres para evaluar su desempeño y probar su eficacia ante la clasificación de algunos mensajes. De igual manera, se utilizaron técnicas del procesamiento del lenguaje natural como la tokenización y el stemming durante la etapa de preprocesamiento de datos, la cual fue una de las partes más delicadas durante el desarrollo del proyecto.

Entre los tres algoritmos de clasificación, se confirmó el modelo de Máquinas de Vectores de Soporte como el mejor modelo para la clasificación de textos, mostrando los mejores resultados, tal como se presenta en la tabulación de datos de la tabla 1.

Se logró diseñar un sitio web como plataforma de sala de chat con el fin de comprobar el funcionamiento el modelo clasificador en un entorno casi real, tomando como entrada los mensajes que se envían los usuarios de la sala y analizándolos, para posteriormente emitir una alerta que actúa como un control parental cuando el modelo clasifique uno de los mensajes como grooming.

Referencias

[1] I. Larrakoetxea Cañasveras, «Identificación del Grooming», p. 93, 2017.

[2] Y. Goldberg, «A Primer on Neural Network Models for Natural Language Processing», arXiv:1510.00726 [cs], oct. 2015, doi: https://doi.org/10.48550/arXiv.1510.00726.

[3] E. D. Torrez Torrez, «Sistema inteligente para la detección de conversaciones con posible contenido pedofilico basado en redes neuronales», p. 123, 2018.

[4] C. D. Oña Salazar y J. G. Proaño Chaviznan, «Implementación de un prototipo de control parental enfocado en la detección inteligente de ataques de grooming», Escuela Politécnicaa Nacional, p. 93, 2020.

[5] M. J. Peláez Curillo y P. P. Bermeo Aguaysa, «Análisis, diseño e implementación de una aplicación inteligente basada en procesamiento de lenguaje natural para la prevención de presunto acoso cibernético usando información de redes sociales», Universidad Politécnica Salesiana Sede Cuenca, p. 288, 2021.

[6] A. Bosch Rué, J. Casas Roma, y T. Lozano Bagén, Deep learning principios y fundamentos. 2020. Accedido: 20 de enero de 2022. [En línea]. Disponible en: http://www.digitaliapublishing.com/a/69970/

[7] A. D. F. Rus y R. M. C. Salas, «Construcción de modelos de clasificación automática para la detección de acoso», Universidad Autónoma de Madrid, p. 63, 2018.

[8] D. Freberg, Evaluating Statistical MachineLearning and Deep Learning Algorithms for Anomaly Detection in Chat Messages. 2018. Accedido: 1 de abril de 2022. [En línea]. Disponible en: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-235957

[9] F. Pedregosa et al., «Scikit-learn: Machine Learning in Python», Journal of Machine Learning Research, vol. 12, n.o 85, pp. 2825-2830, 2011.

[10] A. Géron, Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems, Second edition. Beijing [China] ; Sebastopol, CA: O'Reilly Media, Inc, 2019.

[11] K. S. Parikh y T. P. Shah, «Support Vector Machine – A Large Margin Classifier to Diagnose Skin Illnesses», Procedia Technology, vol. 23, pp. 369-375, ene. 2016, doi: 10.1016/j.protcy.2016.03.039.

[12] M. Gori, Machine learning: a constraint-based approach. Cambridge, MA: Morgan Kaufmann Publishers, an imprint of Elsevier, 2018.

[13] J. M. Rodriguez Rama, «Aplicación de técnicas de machine learning a la detección de ataques», p. 52, jun. 2018.

[14] A. Beltrán Gómez y S. Ordoñez Salinas, «Sistema inteligente para la detección de diálogos con posibles contenidos pedofílicos∗ Intelligent System for Detecting Dialogues with Possible Pedophilic Contents Système intelligent pour la détection de dialogues avec possibles contenus pédophiles», Revista Virtual Universidad Católica del Norte, n.o 42, pp. 164-181, 2014.

[15] P. Zambrano et al., «Technical Mapping of the Grooming Anatomy Using Machine Learning Paradigms: An Information Security Approach», IEEE Access, vol. 7, pp. 142129-142146, 2019, doi: 10.1109/ACCESS.2019.2942805.