María Carolina Gutiérrez Cód. 201922402
Sara Torres Benavides Cód. 201923245
Sofía Quiroga Robles Cód. 201913999
Luis Carlos Rodríguez Cód. 201920638

Problem Set 4: Predicting Tweets

i. Introducción

Partiendo de la idea de que las palabras que usan las personas revelan mucho más que solo lo que quieren decir, la motivación de este trabajo es que las palabras pueden revelar la identidad de una persona. El objetivo es ajustar un modelo que pueda predecir quién es el autor o autora de un tweet, clasificándolo en una de las tres categorías: Claudia López, Álvaro Uribe y Gustavo Petro. El modelo se entrenó usando una base de datos que contiene tweets de estas tres figuras políticas de Colombia y usando el método de redes neuronales. Twitter se ha convertido en una plataforma imprescindible para las figuras políticas alrededor del mundo: es un medio en el que cualquier persona con una cuenta te puede encontrar, puedes compartir ideas de forma rápida y concisa y alcanzar a personas que incluso no te están siguiendo. Cada una de estas figuras públicas cuenta con millones de seguidores en la aplicación y tuitean por lo menos una vez al día. En la actualidad Twitter es una línea directa entre la figura política y los usuarios de esta plataforma, puede comunicar lo que desee y recibir feedback en likes o retweets e incluso respuestas. Por lo tanto, se considera importante analizar cuáles son las palabras que usan estas tres personas en sus tweets, cuáles son los temas de los que más tuitean y encontrar una forma de identificar quién escribió un tweet determinado.

Con la base de datos se entrenaron varios modelos de Deep Learning usando redes neuronales con múltiples capas. El conjunto de datos de entrenamiento fue procesado antes de ajustar los modelos, se llevaron a cabo procesos de limpieza de datos como la eliminación de stopwords para facilitar el uso de esta base. Para predecir quién publicó un tweet, se entrenaron modelos de redes neuronales variando las funciones de activación, el número de neuronas y otros parámetros utilizados en el ajuste, entrenamiento y compilación del modelo. Finalmente, se escogió la especificación con el valor más alto de accuracy al probar el modelo en la competencia de Kaggle.

ii. Datos

Base de datos

La base de datos utilizada para predecir a quién pertenece un tweet es la suministrada dentro de la competencia de Kaggle. La cual, tiene una muestra con diferentes comentarios publicados en las cuentas de Twitter de tres políticos colombianos importantes: Claudia López, Gustavo Petro y Álvaro Uribe. Asimismo, la base de datos descargada de la plataforma Kaggle, ya estaba dividida entre las bases de entrenamiento y prueba. De esta forma, la base de datos de entrenamiento contiene la información de 9349 tweets, con la información del autor y un identificador único por cada comentario en Twitter. Por otro lado, la base de prueba presenta la información completa de 1500 tweets, así como el identificador único por cada uno de estos.

Al revisar la base de datos, es posible evidenciar que el texto de los comentarios no es uniforme, esto dado principalmente por algunas palabras mal escritas, uso de tildes, signos de puntuación, uso de números o emojis. No obstante, se considera que dentro de las bases de datos suministradas se encuentra información relevante y útil para poder plantear un modelo de predicción del autor entre los políticos Claudia López, Gustavo Petro y Álvaro Uribe para los tweets publicados en dicha red social.

Limpieza de la base de datos

Las bases de datos obtenidas se encontraban en formato de CSV “Comma Separated Values”, el cual es muy útil para hacer análisis de datos en el programa de R, de forma que solo es necesario

cargar las bases de datos. Así, se guardan las bases como un "dataframe", tipo de formato especial de R que facilita el análisis de datos, permitiendo el manejo de estos entre filas, columnas y observaciones.

Por otra parte, se limpia el texto de los tweets de las dos bases de datos suministradas, para tener la base más homogénea. De esta forma, se convierten todos los textos de tweets en letras minúsculas, se eliminan las tildes, números y signos de puntuación. Además, se reemplazan todos los caracteres especiales no alfanuméricos por espacios, y finalmente se eliminan los dobles espacios dentro de cada texto. Esto se logra fácilmente utilizando diversos comandos dentro del paquete de r de "tidytext" y "stringi". De otro lado, para tener un mejor manejo sobre todos los textos, se tokeniza por palabra cada tweet y se guarda dentro de un nuevo date frame "words_test" y words_train" para cada base de datos respectivamente. Con esta nueva información, se eliminan los stopwords que se encuentran en común con las palabras de las listas de "snowball", "stopwords-iso" y "nltk", previamente unificadas y modificadas para eliminar dentro listas las tildes, y de esta manera, ser congruentes con las bases suministradas de entrenamiento y prueba previamente limpiadas.

De otro modo, se convierte en una variable categórica la variable suministrada de name, la cual establece el dueño del tweet dentro de la base de entrenamiento, de forma que si el político es Claudia López la variable tomará ahora el valor de 1, si es Gustavo Petro tomará el valor de 2, y tomará el valor de 3 en caso contrario, es decir, si el político es Álvaro Uribe. Por otra parte, debido a que muchas palabras en el español tienen el mismo significado, se lematizan todas las palabras del texto suministrado en ambas bases, de forma que se reduce a su raíz cada una de las palabras. Para lograr lo anterior, se utiliza el modelo de udpipe "spanish-gsd-ud-2.5-191206.udpipe" y se dejan solo las palabras únicas dentro de todas las palabras de los tweets. De esta forma, una vez lematizadas todas las palabras de interés, se tiene en cuenta los casos en que el modelo udpipe no encuentra la raíz de una palabra, de forma que se considera dentro del modelo la palabra original utilizada en el tweet. Con esta información recopilada, se vuelven a unir todas las palabras individuales lematizadas a nivel de los tweets suministrados utilizando la variable de id.

Por último, se decide crear una matriz TF-IDF "Frecuencia del Término - Frecuencia Inversa de los Documentos", esto con el fin de analizar las palabras con más importancia dentro del texto, en comparación con todos los documentos disponibles, y así lograr otorgarle un peso a cada palabra por tweet. Para lograr lo anterior, se crea un corpus y se utiliza el comendo "TermDocumentMatrix" para crear la matriz. Lo anterior, se realiza para las dos bases suministradas de entrenamiento y de prueba.

Estadísticas descriptivas

Como en este caso la idea consistía en predecir la pertenencia de un tweet, a partir de tweets pasados, las variables de la base de datos eran las palabras lematizadas. Como se mencionó anteriormente, la construcción de la base de datos tuvo en cuenta las palabras más importantes entre los tweets para poder hacer una reducción de la muestra sin pérdida de generalidad. Estas incluyen los ítems encontrados tanto como en texto y hashtags. Con esto en mente, se comenzó por construir una nube de palabras para identificar la relevancia de estas dentro de la muestra. A continuación, se presenta la nube de palabras para cada una de las bases de datos (test y train).

Figura 1. Nube de palabras para la base de test de datos.

Como se observa, los tres políticos se distinguen por escribir acerca del país, Colombia, de la capital, Bogotá, al igual que de la situación política, social y económica del entorno. Como los datos son tomados de años anteriores, existe una alta prevalencia de los temas relacionados con la pandemia y el confinamiento, como por ejemplo las palabras de salud, vacuna, COVID, entre otras. Sin embargo, a plena vista es un poco difícil definir a qué político le corresponde cada palabra, pues el momento de la crisis sanitaria da cabida a que se comenten temas similares de cada uno.

Para comparar estos resultados, a continuación, se presenta la nube de palabras encontrada en la base de datos del train de datos.

Figura 2. Nube de palabras para la base de entrenamiento

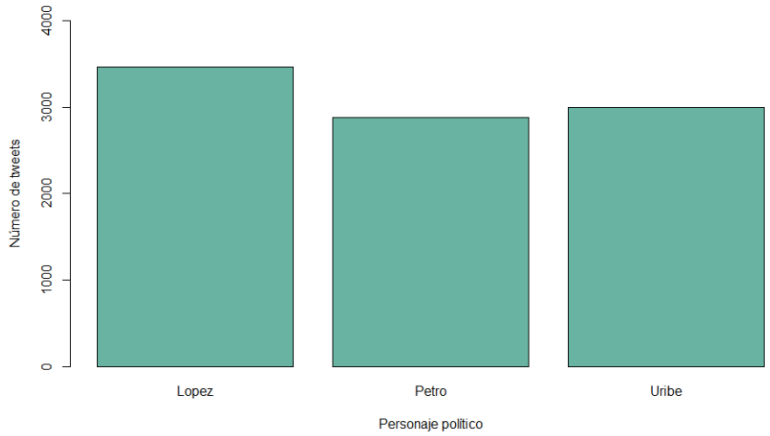
De la figura 2 se evidencia, nuevamente, que los tweets de los tres personajes están relacionados con la capital, el país, y aspectos económicos, sociales y políticos. Entre estos se encuentran palabras como salud, covid, vacunación, derechos, entre otras. A comparación de la figura de la nube de palabras para los datos de test, se encuentra que las palabras parecen ser las mismas o muy similares y también están ordenadas en orden parecido. Palabras como Bogotá, Colombia, gobierno y jóvenes parecen permanecer en la misma jerarquía. Nuevamente, es complejo determinar de

forma visual la pertenencia de nuevos tweets. Por ejemplo, policiabogota, puede ser comúnmente utilizada por la alcaldesa Claudia López, por temas de seguridad y porque la conciernen directamente a ella al ser el lugar de mandato; no obstante, el expresidente Uribe, puede que haga referencia a esta como una necesidad para apaciguar los problemas de seguridad o como instrumento para acallar las protestas y daños sociales, como el uso del ESMAD.

Como es complejo predecir visualmente, es necesario construir un modelo de predicción. No obstante, para construir una base homogénea entre el test y train, es importante tener en cuenta las palabras menos repetidas, o con menor peso, para que el modelo tenga una buena capacidad predictiva.

Por otro lado, es importante tener una noción de la cantidad de tweets que publica cada uno de los políticos, pues de esta forma el modelo podrá tener la frecuencia en cuenta. A continuación, se presentan estas estadísticas.

Gráfica 1. Número de tweets según personaje político



La gráfica ilustra que, de la base de datos, la alcaldesa tweetea más que el expresidente Uribe y que el presidente Petro. Esto puede ser congruente con la realidad, primero porque López es más joven que Uribe y Petro, por lo cual puede estar más actualizada con las redes sociales, y segundo, porque siendo alcaldesa, debe comunicarse abiertamente con la ciudadanía para dar señales de credibilidad, determinación y para informar sobre los planes actuales o futuros que conciernen a los bogotanos. Además, puede que López no esté tan ocupada como el presidente Petro para twittear en su tiempo libre. El segundo con mayor frecuencia para twittear es el expresidente Uribe. Es curioso pensar que este escribe más por redes sociales que el actual presidente a partir de un argumento de edad, pues Uribe tiene más años que Petro. Sin embargo, como ambos políticos son de ideologías y partidos contrarios, y el que está en el cargo de presidencia es Petro, Uribe puede escribir más tweets al ser la oposición y estar pendiente para publicar noticias devastadoras o problemas sociales que parecen ser puestos en segundo plano. En este sentido, siendo Uribe la oposición, Twitter puede ser un instrumento para sacar a flote los problemas que el actual gobierno no ha podido resolver o incluso que ha pasado a segundo plano con el fin de generar revuelta a través de este medio y contrarrestar el populismo del actual presidente.

iii. Modelo final y resultados

Modelo final

El modelo final utilizado fue un modelo de Redes Neuronales con 50 entradas de datos, 4 neuronas, y 3 salidas. El accuracy final del modelo predicho tuvo un valor de 0.47.

Las redes neuronales son técnicas de Deeplearning. Este tipo de técnicas consisten en entrenar algoritmo para que reconozca características específicas de los datos de entrada. De esta forma, adquieren la capacidad de reconocer patrones que utiliza para predecir, clasificar o generar datos. Las redes neuronales son una de las principales técnicas de Deeplearning. Esta consiste en utilizar regresiones no lineales con nodos (neuronas) de transformación no lineales. Al hacer esto, se logra hacer que un modelo lineal sea mejor al predecir un modelo que tienen distribuciones no lineales. Estas neuronas están divididas por capaz, cada capa esta interconectada con las otras capaz. Por lo que todas las observaciones pasan por todas las neuronas, generando la no linealidad en el modelo. Luego de pasar por todos las capaz, se obtienen los valores generados por las neuronas y se usan para realizar la regresión lineal.
Los pasos para obtener una red neuronal son: Elegir el número de observaciones de entrada, el número de capaz, la cantidad de neuronas por capa, y el número de opciones de salida.

En el modelo final, la red neuronal es una red que tiene 50 observaciones de entrada, las cuales son las 50 palabras más utilizadas en los tweets y sus pesos en el n-esimo tweet. Luego, se encuentra una primera capa con una profundidad de 4 neuronas, con función de activación "ReLu". Por último, se encuentra un output de 3 categorías, en la cual, se usa la función "Softmax" para las opciones de salida, las cuales son los 3 políticos de la muestra: López, Petro y Uribe, y están relacionadas con el n-esimo tweet. En resumen, el modelo toma la n-esima observación, revisa cuales palabras tienen peso y el valor de este, luego le pasa estos valores a las 4 neuronas, las cuales transforman los pesos con la función ReLu. Estos nuevos valores de los pesos se pasan a la capa de salida, donde pasan por la función de activación softmax, por último, el resultado se asocia con el político de esa observación. De esta forma se van asociando las palabras a los políticos y el modelo empieza a aprender las características de estos.

La función de activación "ReLu" consiste en tomar el máximo entre el elemento x y 0. De esta forma si el valor al pasar por la neurona es negativo, se vuelve 0, y si es positivo toma el valor de x, después la transformación de la neurona. La razón de escoger esta función es porque es la función más utilizada debido a su simplicidad y buen desempeño en la mayoría de los modelos. La función de activación "softmax" consiste en tomar el valor de entrada y lo transforma en una probabilidad. De esta forma, se escoge esta función dado que es la mejor para predecir problemas de clasificación múltiple, como en este caso, para predecir a los 3 políticos dueños de un tweet dado.

Para poder correr el modelo, se necesita compilar para colocar más parámetros que se utilizaran al momento de entrenar. Primero, el optimizador dicta como se actualiza el modelo según la función de perdida y los datos que se están analizando. El método del optimizador es el de "Adam" debido a su fama, que ajusta la tasa de aprendizaje de forma individual para cada parámetro en función de la estimación de primer y segundo momento de los gradientes. Segundo, la función de perdida, la cual mide la precisión del modelo durante el entrenamiento. El método escogido fue el de "categorical crossentropy" debido a que es una función especializada en problemas de clasificación múltiple. Por último, la métrica, la cual va analizando los resultados del modelo y los va guiando hacia la optimización de un parámetro de rendimiento. En este caso se utilizado la métrica "Accuracy", la cual busca maximizar el número de predicciones correctas sobre el total de predicciones.

Por último, para entrenar al modelo, se utilizó una fórmula de 50 epochs y 50 batch_size. Estos parámetros significan que, al momento de entrenar el modelo, se utilizaron todas las 50 observaciones (batch_size), durante 50 ciclos (epochs). La razón de escoger tantos ciclos es para que el modelo puede definir un buen peso e importancia de los nodos.

Comparación con otros modelos

Antes de escoger el modelo final, se llevó a cabo varios intentos para entrenar el mejor modelo. En primer lugar, una de las especificaciones que se intentó fue una red de tres capas con funciones de activación ReLu en las capas ocultas y una función Softmax en la capa de salida. Se usó la misma función de pérdida que en el modelo final, considerando que la variable a predecir es categórica de más de dos clases. En contraste con el modelo final, se usaron diferentes parámetros para su entrenamiento, tanto el número de ciclo como el número de observaciones que se usaron para cada uno fue menor que en el modelo final (cinco ciclos y 16 observaciones). Sin embargo, la medida de accuracy para esta especificación no es muy alejada del resultado del modelo final (0.39), posiblemente por que cuenta con las mismas funciones de activación en la red neuronal.

Otro modelo que se especificó fue una red neuronal más compleja. Esta red tenía una entrada de 500 observaciones, con 5 capaz ocultas, con 500, 300, 100, 50, y 10 neuronas en cada capa respectivamente. Además, para evitar overfitting, se tenía los siguientes porcentajes de perdida: 0.5, 0.3 y 0.1. También se utilizó ReLu en todas las capas de neuronas. Además, se utilizó los mismos parámetros de compilación que el modelo final. En el entrenamiento se utilizaron 100 observaciones durante 15 ciclos. A pesar de ser un modelo más complejo, su accuracy se alejó del accuracy del modelo final (0.343). Esto se pudo haber debido a que, al complejizar mucho el modelo, se pudo haber sufrido de overfitting, por lo que el accuracy en muestra es bueno, pero en predicción resulto ser malo.

Luego de ver que los modelos muy complejos no son buenos para predecir, se decidió crear un modelo más sencillo, por lo que se corrió un modelo con 300 observaciones de entrada, una capa con 20 neuronas, función de activación ReLu y función de salida Softmax. Se utilizaron los mismos parámetros de compilación. Al momento de entrenar el modelo, se utilizaron 50 ciclos y 15 observaciones en cada ciclo. El valor de accuracy de este modelo fue de 0.42. Este resultado nos indica que simplificar el modelo aumenta su capacidad predictiva, probablemente debido a la disminución del overfitting.

Siguiendo la corriente planteada por los anteriores modelos de que mayor simplicidad mayor accuracy, se creó un modelo más simple que el anterior. El modelo consta de 100 observaciones, con una sola capa de 10 neuronas con función ReLu y una función de salida Softmax. Se utilizaron los mismos parámetros de compilación. Al momento del entrenamiento del modelo. Se utilizaron 50 ciclos y 100 observaciones, por lo que se utilizó la totalidad de observaciones del modelo. Al hacer esto, se esperaba que la capacidad predictiva del modelo aumentara. Lo cual se logró ya que el valor del accuracy fue de 0.45. Este resultado siguió con la tendencia de disminuir el overfitting para aumentar el accuracy.

iv. Conclusiones

Finalmente, se encuentra que la mejor predicción se obtuvo bajo una red neuronal con 50 observaciones de entrada, una única capa escondida con 4 neuronas, y una variable de resultado de tres niveles, una por cada político. Sin embargo, dada la baja exactitud de este modelo, se realizaron diversas redes neuronales con diversos hiperparametros. De esta forma, se encuentra un segundo modelo con una red neuronal con 16 observaciones de entrada, con tres capas ocultas con cinco neuronas, e igualmente una variable de resultado de tres niveles. Por otra parte, se desarrolla un tercer modelo de redes neuronales más complejo, utilizando 500 observaciones de entrada, con cinco capas escondidas, dentro de las cuales de toman diversos inputs. Es así como, se plantean 500, 300, 100, 50, y 10 neuronas en cada capa respectivamente. Además, para evitar el overfitting de modelos anteriores, se plantean los siguientes porcentajes de pérdida 0.5, 0.3 y 0.1, respectivamente. Por otra parte, se desarrolla un último modelo más sencillo con 300 observaciones de entrada, una única capa oculta con 20 neuronas, y una capa de salida con tres categorías respectivamente. Posteriormente, se desarrolla un último modelo con 100 observaciones de entrada, y una capa escondida con 10 neuronas, para terminar en un output

nuevamente de tres categorías. Es importante resaltar que todos los modelos utilizados se plantearon bajo la misma función de activación ReLu entre capas, y función de activación Softmax para la capa de salida.

En definitiva, este trabajo consistió en predecir tweets fuera de muestra que correspondían a los personajes políticos Gustavo Petro, Claudia López y Álvaro Uribe. Los modelos predictivos estuvieron basados en la metodología de DeepLearning para lograr predecir a través de datos en forma de texto. En particular, los modelos utilizados no tuvieron una buena predicción, medida a través del score del accuracy en Kaggle; principalmente, hubo problemas de sobreajuste dentro del entrenamiento. De esta forma, no se logró encontrar una combinación adecuada y óptima con buenos resultados predictivos sobre la pertenencia de cada tweet a los tres políticos establecidos anteriormente. Con los problemas anteriores, se intentó corregirlos a partir de diversos cambios en los hiperparámetros y/o con ayudas como Dropout dentro de los diferentes modelos. No obstante, las predicciones logradas continuaron teniendo baja predicción fuera de muestra, con cambios poco significativos dentro del accuracy en Kaggle. Dado lo anterior, se recomienda en futuras ocasiones utilizar diferentes funciones de activación, como lo es la del coseno, además de revisar uno a uno cada uno de los parámetros para determinar con cual configuración dentro de estos modelos es posible llegar a la mejor predicción, con la mejor métrica de Accuracy.

v. Anexos

Enlace al repositorio de GitHub: https://github.com/SofiaQuiroga/Repositorio_Taller4_BDML