**INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY**

**A blue and white logo

Description automatically generated**

# 

# Clasificación automática de descripciones de carga contenerizada en códigos del Sistema Armonizado (HS) usando modelos de Procesamiento de Lenguaje Natural

**Prof. Titular: Dra. Grettel Barceló Alonso**

**Juan Pablo Acosta López (A01794035)**

**Oscar Villa Cárdenas (A01794052)**

**Genaro Rodriguez Vázquez (A01150931)**

**29 septiembre 2024**

**Índice**

[Clasificación automática de descripciones de carga contenerizada en códigos del Sistema Armonizado (HS) usando modelos de Procesamiento de Lenguaje Natural](#_heading=h.crzpksqdu7ux)

[Comparar el rendimiento de los modelos obtenidos](#_heading=h.j0432rcuia01)

[Google Flan](#_heading=h.rtb4tdoh4lhf)

[Pegasus](#_heading=h.ecyfbl66d9k0)

[Conclusiones](#_heading=h.xhep8orgpmfj)

[Referencias](#_heading=h.ud6c1uqe1vfv)

[(N.d.). Retrieved from https://www.dripcapital.com/hts-code/section/all](#_heading=h.yxuj5vdccito)

## Introducción

En este avance del proyecto, nos enfocamos en desarrollar una Large Language Model (LLM) capaz de predecir códigos HS (Harmonized System) a 2 y 4 dígitos a partir de descripciones de productos. El código HS es fundamental en el comercio internacional, y su predicción precisa automatiza y optimiza los procesos aduaneros y arancelarios.

La estrategia de este avance consiste en construir al menos seis modelos individuales, combinando sentence-transformers y modelos LLM, además de realizar ajustes en los hiperparámetros para ambos tipos de modelos. Estas combinaciones permitirán explorar diferentes enfoques para ver cuál ofrece el mejor rendimiento en la predicción de los códigos HS, basados en descripciones de productos.

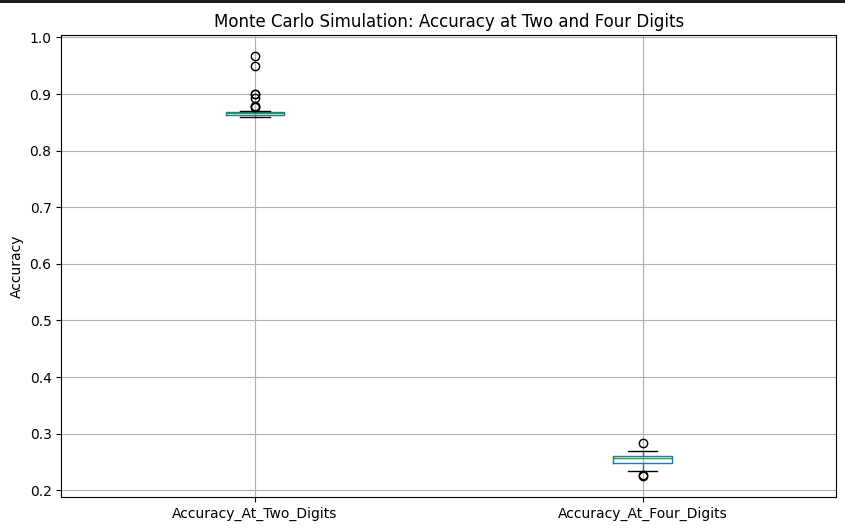
Para asegurar la robustez de los resultados, las validaciones se realizaron mediante simulaciones de Monte Carlo. Este enfoque incluyó la ejecución de 30 predicciones por modelo durante 50 iteraciones. Con estas corridas, se pudo obtener una evaluación más precisa del desempeño promedio de cada modelo, identificando los dos que mostraron el mejor rendimiento general. Después de esta selección, se ajustarán los hiperparámetros para maximizar aún más su precisión y confiabilidad, seleccionando finalmente el modelo que ofrezca los mejores resultados en la predicción de códigos HS.

## Comparar el rendimiento de los modelos obtenidos

### Llama 3.1 / all-mpnet-base-v2

### LLama 3.1 es una versión avanzada de la serie de modelos de lenguaje grande (LLM) desarrollados por Meta (anteriormente Facebook). Esta serie de modelos está diseñada para realizar tareas de procesamiento de lenguaje natural (NLP), como generación de texto, comprensión de lenguaje, traducción y más. LLama 3.1 mejora sobre sus versiones anteriores en términos de eficiencia, precisión y capacidad para manejar tareas complejas de texto con mayor coherencia y contexto.

### All-mpnet-base-v2 es un modelo de lenguaje desarrollado como parte de la familia MPNet, diseñado específicamente para tareas de representación de oraciones y texto. Es ampliamente utilizado para tareas de procesamiento de lenguaje natural (NLP) como clasificación de texto, búsqueda semántica, y recuperación de información, entre otras. Este modelo fue entrenado por Microsoft y forma parte de los modelos de sentence-transformers, que permiten convertir oraciones o textos en representaciones vectoriales (embeddings) de alta calidad para su uso en diversas aplicaciones de NLP.



**Precisión a dos dígitos**

* + La precisión parece estar centrada alrededor de **0.9**.
  + La caja es estrecha, lo que indica poca dispersión, es decir, los resultados están agrupados cerca del valor central.
  + Hay algunos **valores atípicos (outliers)** por encima de 0.9, lo que sugiere que en algunas simulaciones se alcanzó una precisión aún mayor.

**Precisión a cuatro dígitos**

* + La precisión media parece ser más baja, alrededor de **0.3**.
  + La caja es también bastante estrecha, lo que sugiere que los resultados en esta categoría son consistentes, aunque más bajos en comparación con los dos dígitos.
  + También se observan algunos **outliers** hacia abajo, lo que indica que en algunas simulaciones la precisión fue aún menor.

La **precisión a dos dígitos** es considerablemente mayor que la precisión a cuatro dígitos, con valores que rondan el 90%, mientras que la precisión a cuatro dígitos es mucho más baja, alrededor de un 30%.

Ambas categorías muestran valores atípicos, pero la dispersión general es pequeña, indicando que la mayoría de las simulaciones tuvieron resultados bastante consistentes dentro de cada grupo.

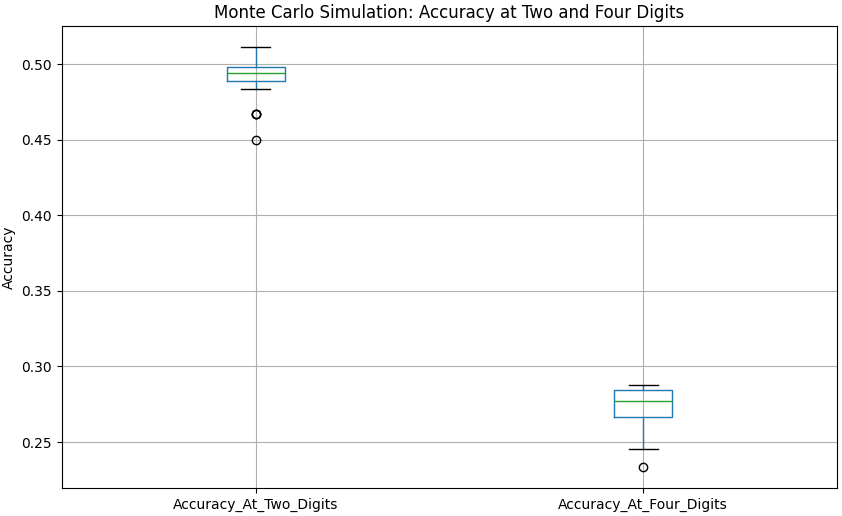
### GpT 4-o / text-embedding-3-large

**GPT-4** es un modelo de lenguaje avanzado desarrollado por OpenAI, y representa una evolución significativa en comparación con sus versiones anteriores. Se trata de un modelo de inteligencia artificial diseñado para comprender y generar texto en lenguaje natural con un nivel de coherencia, creatividad y precisión mucho mayor que sus predecesores. GPT-4 ha sido entrenado utilizando grandes cantidades de datos textuales y un enfoque de aprendizaje profundo, lo que le permite realizar tareas complejas de procesamiento de lenguaje natural, como generación de texto, traducción, resúmenes, entre otros.

Una de las mejoras clave de GPT-4 es su capacidad para manejar contextos más largos y mantener la coherencia en textos más extensos. Este avance lo hace más eficaz en aplicaciones donde es necesario procesar o generar grandes volúmenes de texto, manteniendo un flujo narrativo coherente. También puede comprender instrucciones más complejas y ofrecer respuestas más precisas y matizadas, lo que amplía su aplicabilidad a una gama más amplia de sectores, como la atención al cliente, la educación, la investigación y la escritura creativa.

**text-embedding-ada-003** es un modelo desarrollado por OpenAI específicamente para la creación de embeddings de texto de alta calidad. Los embeddings son representaciones numéricas de texto que capturan las características semánticas y contextuales de palabras, frases o documentos, lo que permite compararlos y realizar tareas de procesamiento de lenguaje natural (NLP) de manera eficiente. Este modelo es la tercera versión dentro de la familia de modelos de "Ada" y ofrece mejoras significativas en comparación con versiones anteriores.

**text-embedding-ada-003** se destaca por su capacidad para generar representaciones vectoriales que preservan las relaciones semánticas entre los textos. Esto lo convierte en una herramienta valiosa para una amplia gama de aplicaciones como la búsqueda semántica, la clasificación de texto, el análisis de sentimientos y el clustering. Al convertir oraciones o párrafos en vectores matemáticos, este modelo permite realizar comparaciones precisas basadas en el significado, mejorando la relevancia y la precisión en tareas de búsqueda y análisis.



**Precisión a dos dígitos:**

* + El valor central (mediana) de la precisión está cerca de **0.48**.
  + El rango intercuartil es pequeño, lo que indica que la mayoría de los valores de precisión están muy agrupados en torno a este valor.
  + Hay uno o dos **valores atípicos (outliers)** por debajo de **0.45**, indicando que algunas simulaciones tuvieron una precisión más baja.
  + Los valores máximos están cerca de **0.50**, lo que sugiere un límite superior de precisión en estas simulaciones.

**Precisión a cuatro dígitos:**

* + La precisión central está cerca de **0.28**.
  + La caja es estrecha, lo que indica poca dispersión en los resultados, lo que significa que la precisión a cuatro dígitos es relativamente consistente en las simulaciones.
  + Hay algunos **outliers** más bajos, con valores cercanos a **0.25**, indicando que en ciertas simulaciones la precisión fue aún más baja.
  + El límite superior está cerca de **0.30**, lo que marca un tope en las simulaciones a cuatro dígitos.

### 

La **precisión a dos dígitos** es significativamente mejor que la precisión a cuatro dígitos, con valores de alrededor de **0.48** frente a **0.28** en promedio.

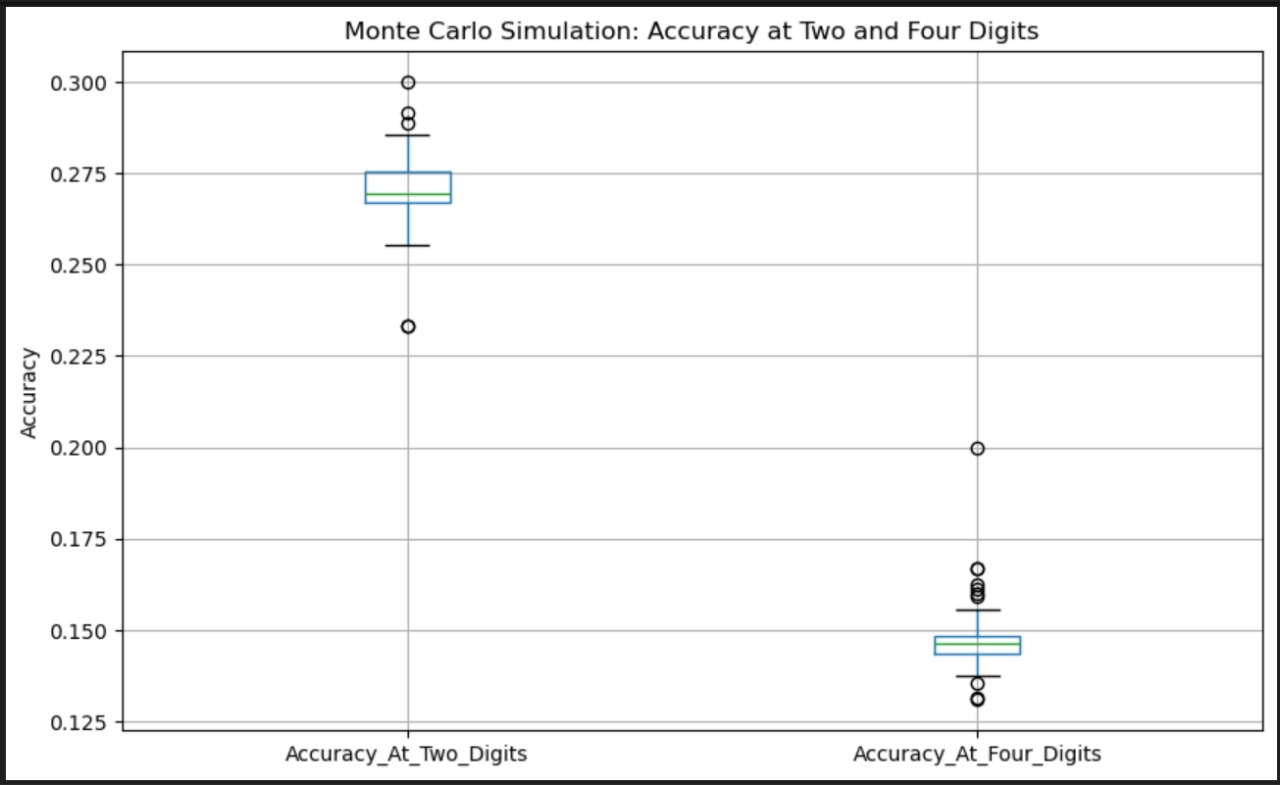
Ambas categorías presentan valores atípicos, pero la dispersión de la precisión en las simulaciones es baja en general.

A pesar de los valores atípicos, los resultados son consistentes en ambas categorías, siendo la precisión a dos dígitos casi el doble que la precisión a cuatro dígitos.

### GOOGLE FLAN T5 / LARGE TEXT-2-TEXT

Google FLAN-T5 Large es un modelo de generación de texto basado en la arquitectura Transformer, optimizado para tareas de texto a texto. Pertenece a la familia de modelos T5 (Text-to-Text Transfer Transformer), que convierte todas las tareas de procesamiento de lenguaje natural (NLP) en un formato de entrada y salida de texto. FLAN-T5 es una versión mejorada del modelo T5, entrenada con instrucciones explícitas (técnica de instruction tuning) para mejorar su capacidad en tareas de entendimiento y generación de lenguaje.

El modelo FLAN-T5 Large es especialmente eficaz en una amplia gama de tareas, como resumen de textos, traducción, respuesta a preguntas, clasificación, y muchas otras, gracias a su enfoque flexible de convertir cada problema en una tarea de generación de texto. Esta versión "Large" tiene un mayor número de parámetros en comparación con otras versiones (base o small), lo que le permite tener un rendimiento más robusto en tareas complejas, aunque también requiere más recursos computacionales.



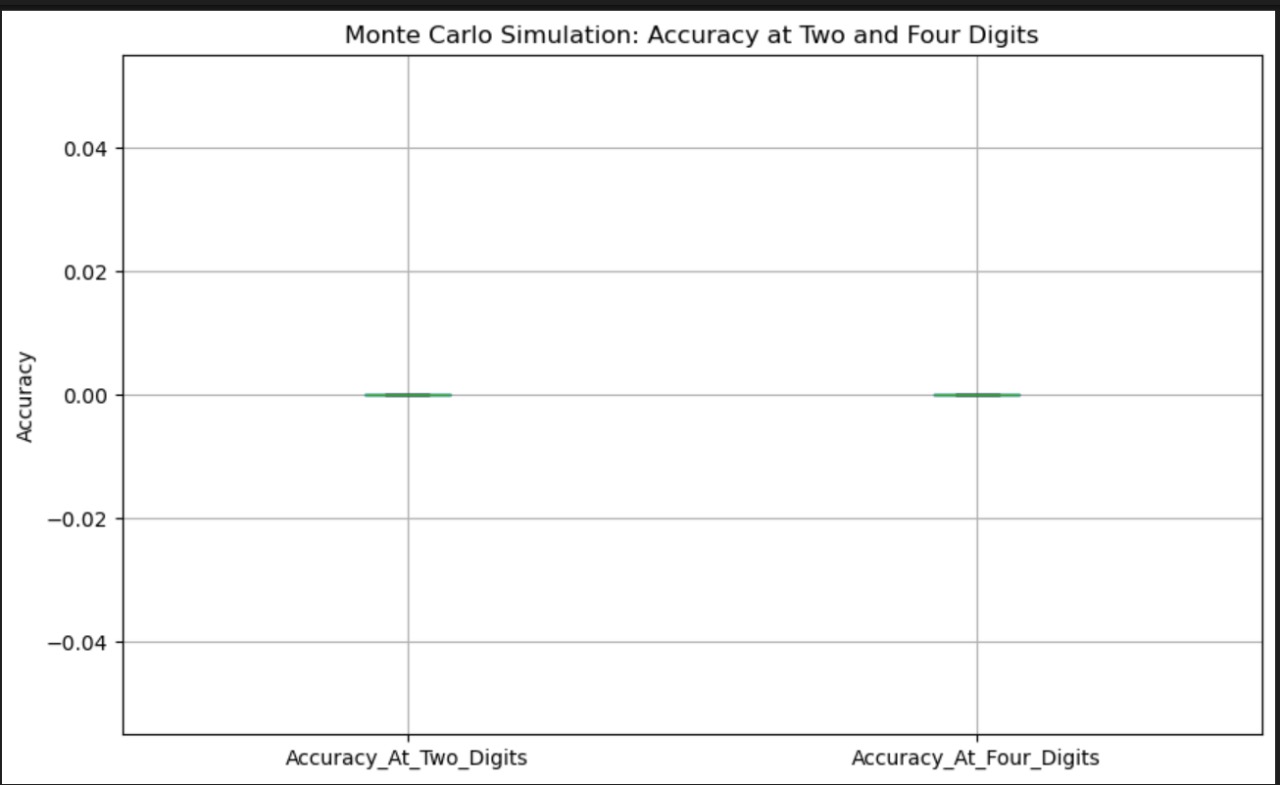
**Precisión a dos dígitos:** La precisión es mayor en general, con una mediana más alta y un rango intercuartílico más amplio. Sin embargo, también hay más valores atípicos, lo que sugiere que en algunos casos la precisión puede ser bastante baja.

**Precisión a cuatro dígitos:** La precisión es menor en promedio, con una mediana más baja y un rango intercuartílico más estrecho. Los valores atípicos también son menos frecuentes, lo que indica que la precisión es más consistente.

### GOOGLE PEGASUS / LARGE TEXT-2-TEXT

GOOGLE PEGASUS es un modelo de procesamiento de lenguaje natural diseñado específicamente para tareas de resumen de texto. Desarrollado por Google Research, PEGASUS (Pre-training with Extracted Gap-sentences for Abstractive Summarization Sequence) se entrena utilizando una técnica innovadora en la que ciertas oraciones clave se eliminan de un documento durante el preentrenamiento, y el modelo debe generar esas oraciones a partir del contexto restante. Esta estrategia permite que PEGASUS aprenda de manera más eficiente a generar resúmenes abstrayendo la información más importante de un texto extenso.

El modelo ha demostrado un rendimiento superior en una variedad de tareas de resumen, como la creación de resúmenes concisos para artículos de noticias, documentos científicos y contenido de sitios web. Gracias a su enfoque de entrenamiento único, PEGASUS es particularmente efectivo para generar resúmenes que mantienen la coherencia y la fidelidad con el contenido original.



**Precisión:** Parece que no pudo predecir ninguno a 2 o cuatro dígitos.

### MODELO GloVe (Global Vectors for Word Representation)

Es un modelo de aprendizaje de representaciones vectoriales para palabras, desarrollado por la Universidad de Stanford. Su objetivo principal es capturar las relaciones semánticas entre palabras mediante el análisis de la coocurrencia de palabras en grandes corpus de texto. Esto significa que GloVe genera vectores que representan palabras de manera que las palabras con significados similares estén más cercanas entre sí en el espacio vectorial.

En el problema de clasificación de códigos HS (Harmonized System), se busca asociar descripciones textuales de productos a códigos estandarizados que identifican estos productos. Esta tarea involucra comprender el lenguaje natural de las descripciones de los productos, lo que es un desafío debido a la variedad de términos, sinónimos y ambigüedades que pueden surgir. GloVe puede capturar relaciones semánticas entre palabras. En el contexto de los códigos HS, palabras o frases relacionadas con productos similares (por ejemplo, "cerdo vivo" y "animales vivos") deberían estar cerca en el espacio vectorial, lo que facilita la clasificación de productos en el código correcto.

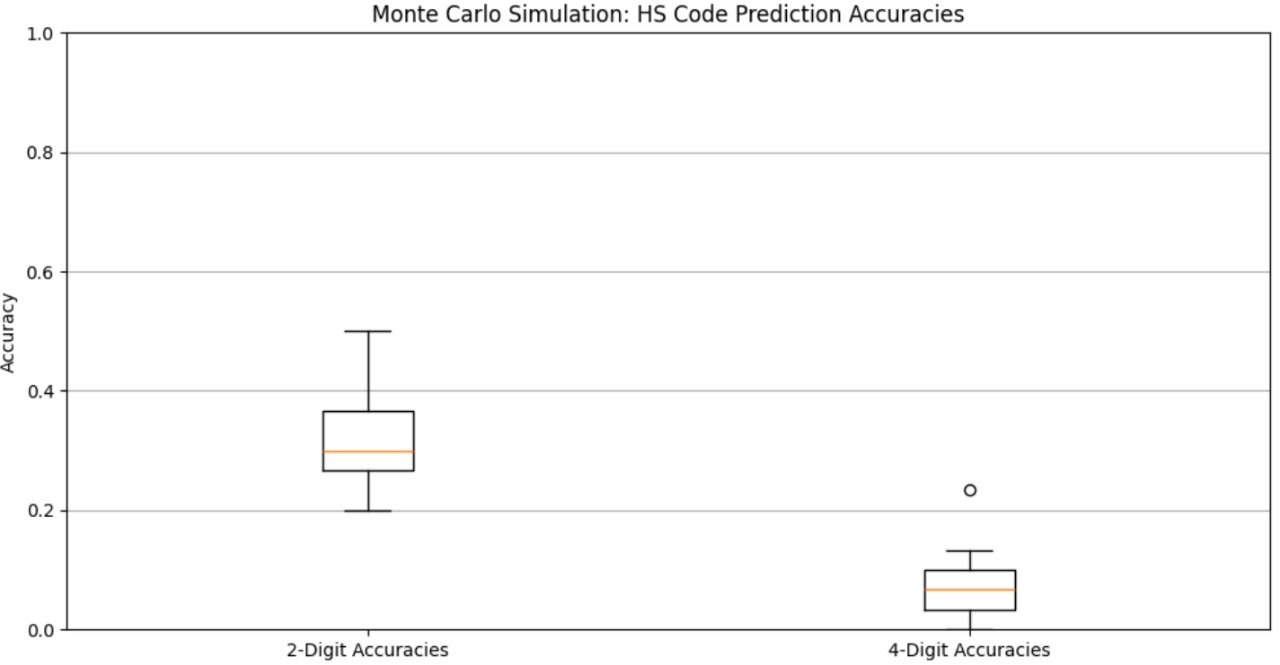
Se utilizaron 2 modelos pre-entrenados en corpus generales, ambos públicamente disponibles y soportados por GloVe:

**GloVe 6B 100D:**

Corpus: El número "6B" significa que este modelo fue entrenado con un corpus de 6 mil millones de tokens (palabras o grupos de palabras). Este corpus incluye datos provenientes de diversas fuentes, como Wikipedia y noticias.

Dimensiones: "100d" indica que los vectores de palabras generados tienen 100 dimensiones. Cada palabra se representa como un vector en un espacio de 100 números, lo que captura suficiente información semántica para tareas generales, pero con menos detalle que modelos de mayor dimensionalidad.

Ventajas: Este modelo es más liviano y rápido de utilizar, ideal para tareas donde no se necesita un análisis semántico tan profundo o cuando los recursos computacionales son limitados.

****

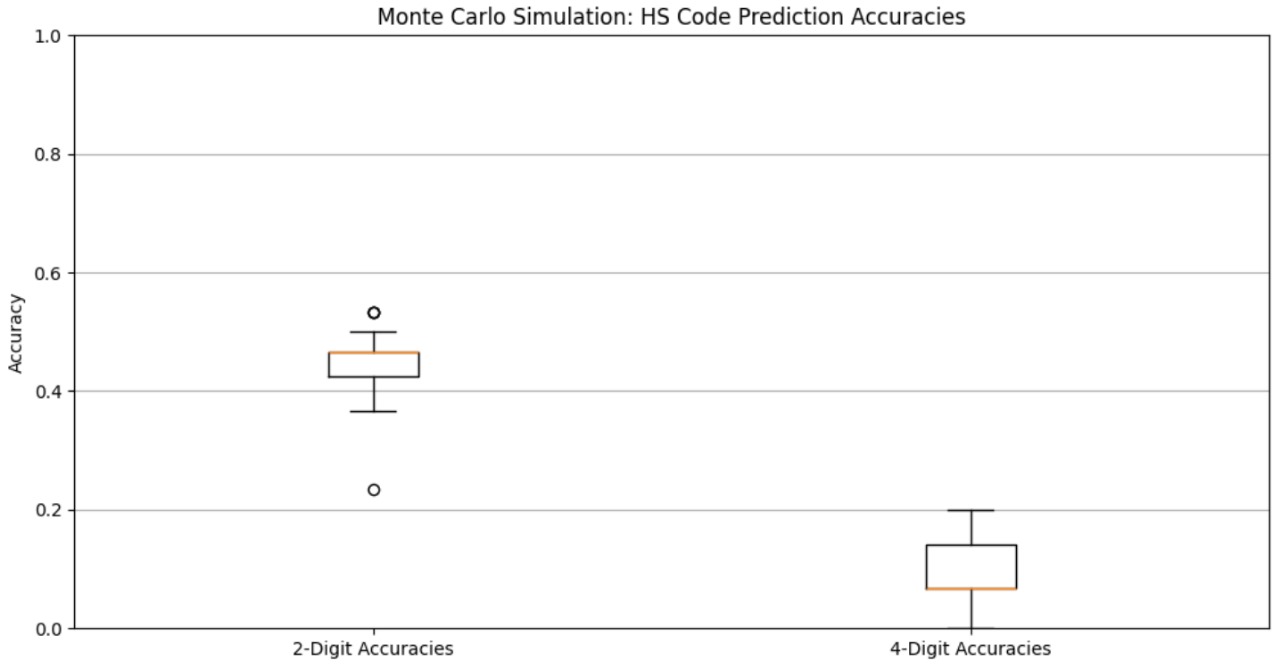
**GloVe 840B 300D:**

Corpus: "840B" significa que el modelo fue entrenado con un corpus mucho más grande, de 840 mil millones de tokens. Esto incluye una mayor variedad de textos y probablemente muchas más palabras y frases infrecuentes o especializadas.

Dimensiones: "300d" indica que los vectores de palabras tienen 300 dimensiones, lo que permite capturar detalles semánticos mucho más complejos y precisos que en el modelo de 100 dimensiones.

Ventajas: Este modelo es ideal para tareas más complejas o donde se requiere un análisis más detallado de las relaciones entre palabras, aunque requiere más capacidad de procesamiento y almacenamiento.

**Resultados:**

****

Comentarios modelo GloVe:

El modelo de 300 dimensiones (300D) fue superior al de 100 dimensiones (100D), lo cual tiene sentido, ya que el modelo de mayor dimensionalidad puede capturar más características semánticas complejas de las palabras. Al tener 300 dimensiones, el modelo es capaz de representar relaciones más detalladas entre las palabras, lo que mejora su capacidad para comprender descripciones de productos más complejas o especializadas, como es el caso de la clasificación de los códigos HS. Este resultado es consistente con lo esperado: mayor dimensionalidad implica una representación más rica y, por lo tanto, un mejor rendimiento en tareas complejas.

Por otro lado, en la primera iteración de los modelos, no se aplicó un pipeline de limpieza de datos adecuado, lo que resultó en un desempeño muy pobre. Los modelos GloVe, como cualquier modelo de aprendizaje basado en texto, dependen en gran medida de la calidad de los datos de entrada. Cuando las descripciones de los productos no fueron preprocesadas (por ejemplo, sin eliminar signos de puntuación, convertir el texto a minúsculas o eliminar caracteres innecesarios), los vectores generados por el modelo no pudieron capturar correctamente el significado subyacente de las palabras. Esto afectó negativamente la capacidad del modelo para predecir correctamente los códigos HS.

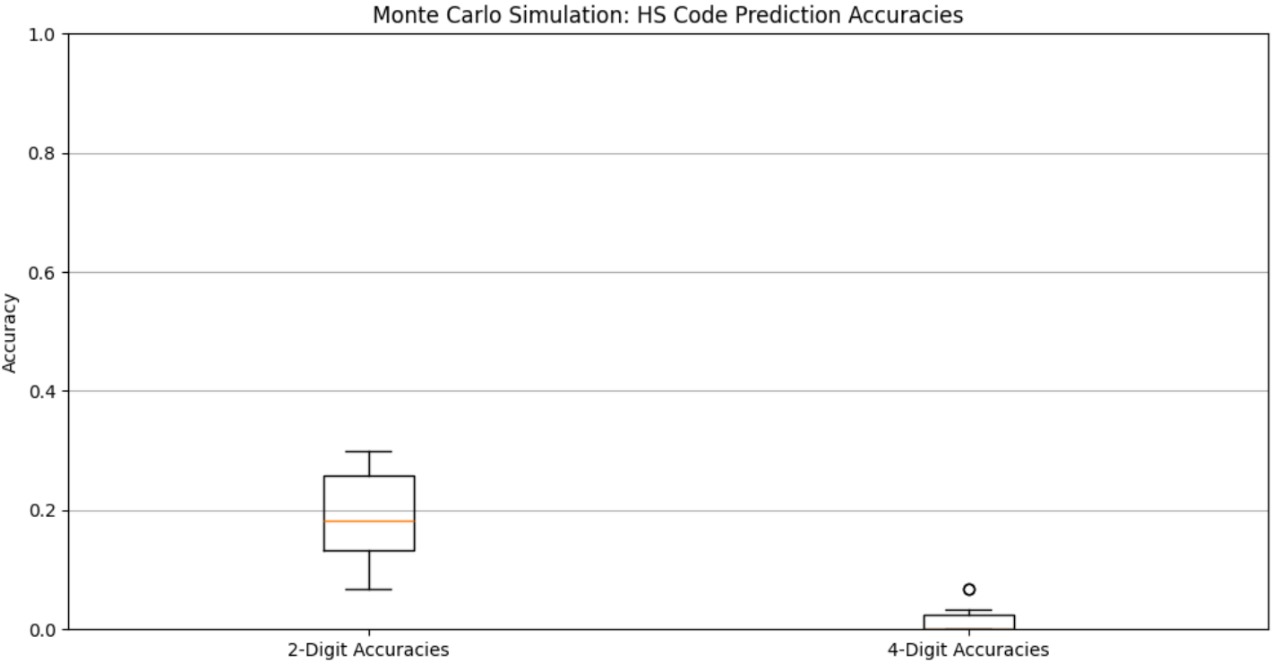
La limpieza de datos es fundamental para cualquier modelo basado en texto, ya que permite eliminar el ruido y garantizar que las palabras que representan el mismo concepto se traten de manera uniforme. Esto es especialmente relevante en modelos como GloVe, que calculan relaciones basadas en la co-ocurrencia de palabras. Sin un preprocesamiento adecuado, los modelos no podrán generar representaciones semánticas confiables, lo que subraya la importancia de preparar bien los datos antes de entrenar o aplicar modelos de lenguaje.

### MODELO Word2Vec

Word2Vec es un modelo de aprendizaje de representaciones vectoriales desarrollado por Google, diseñado para capturar relaciones semánticas entre palabras. Utiliza grandes corpus de texto para entrenar y generar vectores numéricos que representan palabras, de modo que palabras con significados similares estén cercanas entre sí en un espacio vectorial. El enfoque principal de Word2Vec es predecir palabras cercanas dentro de un contexto o ventana de palabras.

Al igual que otros modelos de vectores de palabras, Word2Vec ayuda a manejar la ambigüedad y la variedad de términos utilizados en las descripciones de productos, permitiendo agrupar términos relacionados en un mismo espacio semántico.

**Resultados:**

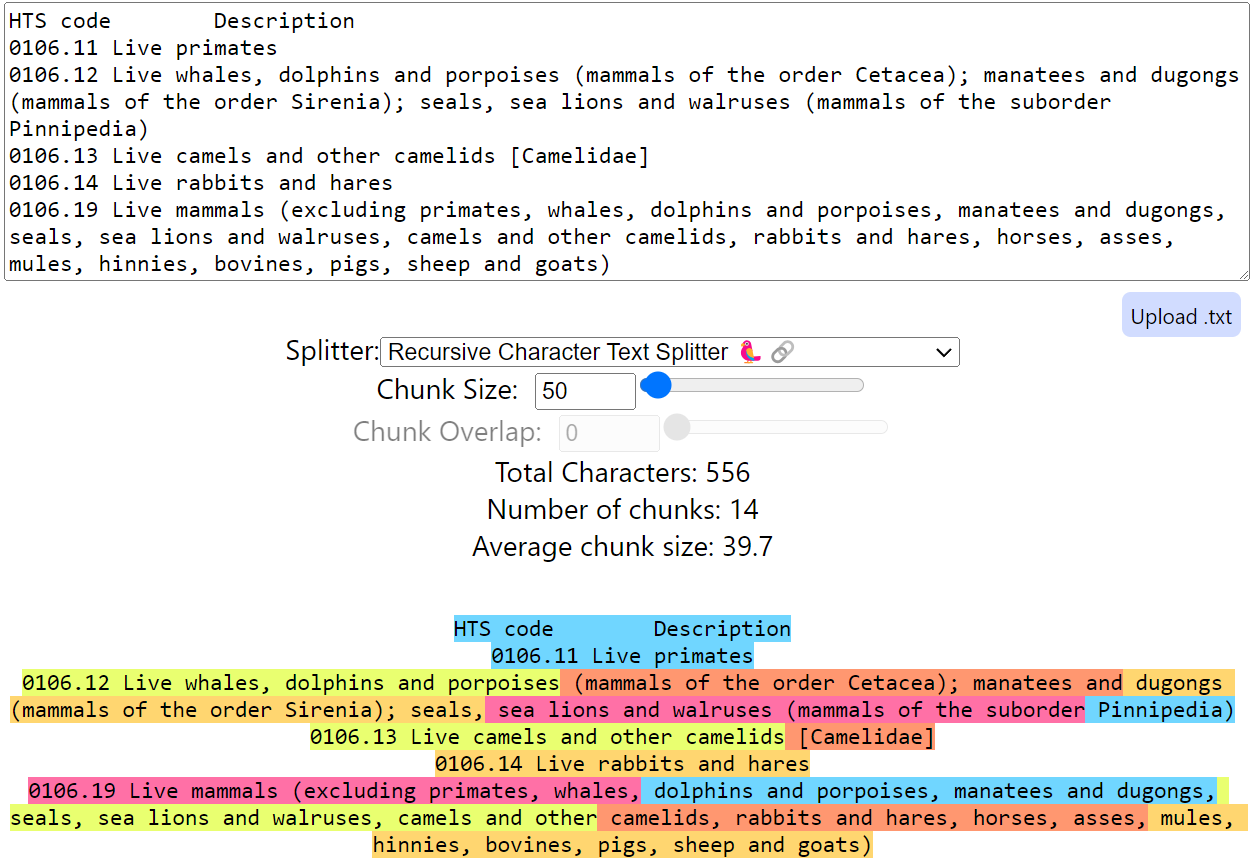


El modelo pre-entrenado utilizado para este ejercicio de Word2Vec es el modelo entrenado en el corpus de Google News, que contiene alrededor de 100 mil millones de palabras. Esto le permite capturar relaciones semánticas generales entre palabras comunes, especialmente aquellas usadas en noticias y textos populares. Sin embargo, para tareas especializadas como la clasificación de productos en códigos HS, este modelo puede no ser tan efectivo, ya que el vocabulario proveniente de noticias podría no representar con precisión los términos específicos utilizados en el comercio y la industria.

Por eso, se concluye que el rendimiento de Word2Vec en este caso podría ser inferior a modelos como GloVe, que usan corpus más amplios o especializados. Además, las representaciones generadas por modelos más recientes como los grandes modelos de lenguaje (LLMs) podrían ofrecer mejores resultados, al capturar relaciones semánticas más ricas y complejas que las de Word2Vec.

## Selección del mejor modelo

El modelo seleccionado fue: **Llama 3.1 / all-mpnet-base-v2**, donde uno de los hiperparametos clave fue el escoger un **chunk size** apropiado para la longitud de las descripciones de cada uno de los HS codes. Además se utilizó el tipo de divisor “**Recursive Character Text Splitter”** el cual suele ser útil cuando tienes texto largo y necesitas partirlo en fragmentos más pequeños para procesamiento. Este enfoque trata de dividir el texto sin interrumpir las oraciones o unidades de significado. Un ejemplo de cómo funciona este tipo de divisor para tokenización se puede visualizar de la siguiente manera:



Como técnica de preprocesamiento se limpia y normaliza texto en varios pasos para optimizar su uso en un modelo de lenguaje grande (LLM). Primero se convierte el texto a minúsculas para uniformidad, se eliminan caracteres no alfabéticos, y se tokeniza el texto en palabras individuales. Luego, se filtran las stopwords, que son palabras comunes sin mucho valor semántico, y se aplica lematización para reducir las palabras a su forma base, preservando el significado. Finalmente, las palabras resultantes se recomponen en una cadena limpia, dejando un texto simplificado, relevante y coherente, para ser usado en el modelo LLM Llama 3.1 y lograr predecir con mayor precisión el HS code correcto.

Los hiperparametros finales son:

index\_chunk\_size = 50

index\_sentence\_transformer = 'all-mpnet-base-v2'

index\_chunk\_technique = 'Recursive Character Text Splitter'

LLM\_model = 'llama3.1'

temperature\_parameter=0.25

retriever\_matches\_k = 3

El pipeline para pre-procesar descripciones antes de mandar a predicción a LLM::

def clean\_text(text):

text = text.lower() # Convert text to lowercase

text = re.sub(r'[^a-zA-Z\s]', '', text) # Remove non-alphanumeric characters (keep spaces)

words = word\_tokenize(text) # Tokenization

words = [word for word in words if word not in combined\_stopwords] # Remove stopwords

# Lemmatization to convert the received sentence to a meaningful sentence

lemmatizer = WordNetLemmatizer()

words = [lemmatizer.lemmatize(word) for word in words]

# Join words back into a sentence

meaningful\_text = ' '.join(words)

return meaningful\_text

## Conclusiones

Aunque es posible ofrecer una conclusión sobre el pre-procesamiento de los datos textuales (como tokenización, limpieza de texto, etc.), la metodología CRISP-ML está más orientada a proyectos de ciencia de datos tradicionales con modelos numéricos o categóricos. En los proyectos de LLM, la preparación de datos es diferente, ya que se centra más en la limpieza textual y menos en la transformación de variables numéricas. A su vez, es necesario utilizar una correcta arquitectura de transformación de sentencias dependiendo del caso de uso, que en este caso buscamos una mezcla entre clasificación, pero a su vez requerimos de similitud semántica.

Además, en el pipeline de limpieza propuesto, se incluyó la limpieza de texto (normalización, lematización, eliminación de stopwords), la traducción automática de los manifiestos a un idioma común y la extracción de tokens relevantes mediante análisis de frecuencias y técnicas de atención. Este enfoque no solo busca la clasificación correcta, sino también la captura de la similitud semántica, asegurando que descripciones textuales diferentes pero equivalentes se traten de forma coherente. Así, la metodología se adapta a la naturaleza específica de los modelos LLM, donde la comprensión semántica es clave para el éxito del proyecto.

## Referencias

Inmon, B. (2017). Turning text into gold: Taxonomies and textual analytics. Bradley Beach: Technics Publications

Manning, C., Raghavan, P., & Schütze, H. (2008). Introduction to information retrieval. Cambridge: Cambridge University Press.

Wilbur, W. J., & Sirotkin, K. (1992). The automatic identification of stop words. Journal of Information Science, 18(1), 45-55

#### (N.d.). Retrieved from <https://www.dripcapital.com/hts-code/section/all>