ITBA
Instituto Tecnológico de Buenos Aires

¿Cómo nombrar funciones?

82.18 – Procesamiento del Lenguaje Natural

13/11/2023

Agustin Benvenuto - 61448
Agustin Ezequiel Galarza - 61481
Bruno Ezequiel Soifer - 62423
Sol Alejandra Winkel - 62409

Agenda

01 **Introducción**
Sobre el TP 1. Objetivo

02 **Dataset**
Definición del corpus. Análisis exploratorio.

03 **Modelos y herramientas**
Explicación de Bert, Word 2 Vec y Code 2 Vec

04 **Resultados**
Armado de métricas. Análisis de resultados.

05 **Conclusiones**
Conclusiones. Limitaciones. Acciones hacia el futuro.

ITBA
Instituto Tecnológicc
de Buenos Aires

Introducción

Modificaciones respecto al TP 1

- Enfoque en la tarea de generar nombres de funciones
- Se definió utilizar un Transformer bidireccional (para analizar la totalidad del contexto)
- Modificación del corpus. Métodos en java.

Objetivo

El objetivo del trabajo es solucionar la problemática que se presenta al programar, que está asociada a elegir los nombres de las funciones. Se busca facilitar el proceso de elección del mismo mediante un modelo que devuelva posibles nombres que sean acordes a al cuerpo y al objetivo de la función.

ITBA
Instituto Tecnológicc
de Buenos Aires

Dataset

Armado del dataset

01

Archivos de clase .java

Se utilizaron muchos archivos de clase .java para iniciar el armado del dataset

02

Parser de java

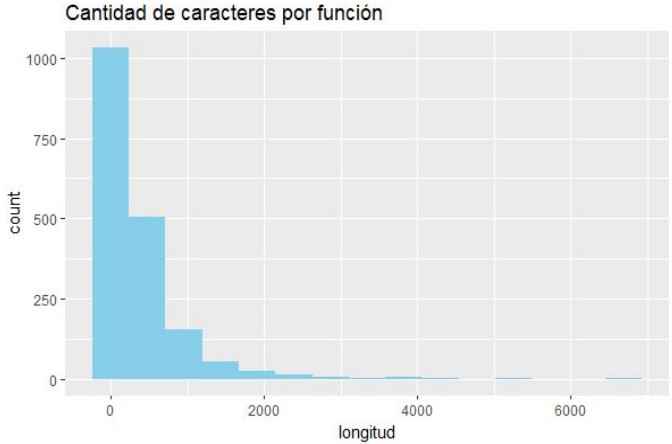
Se utilizó un parser de java para extraer las funciones de los archivos

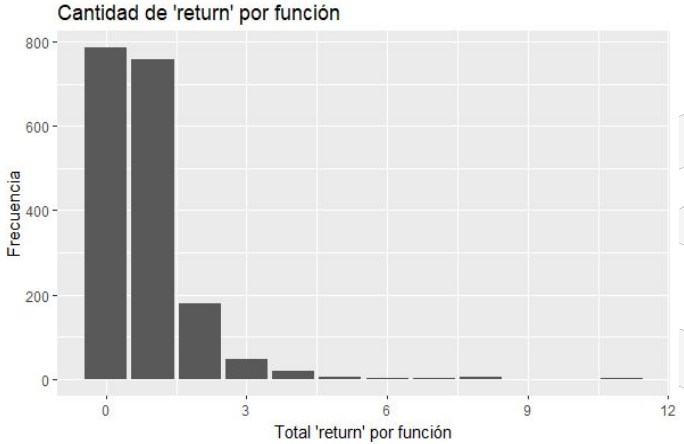
03

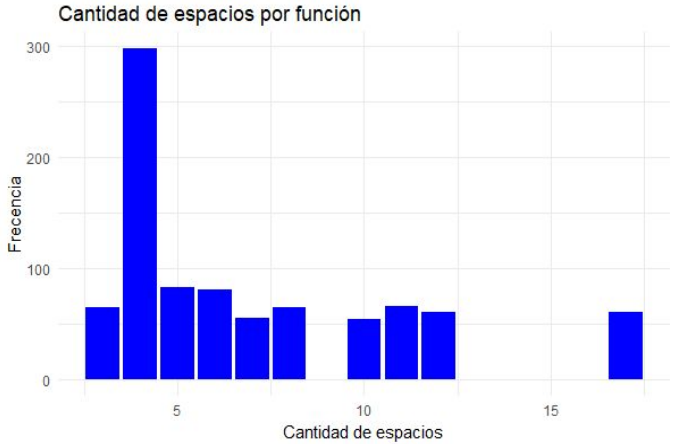
Armado de archivo

Armado del dataset con lo obtenido previamente

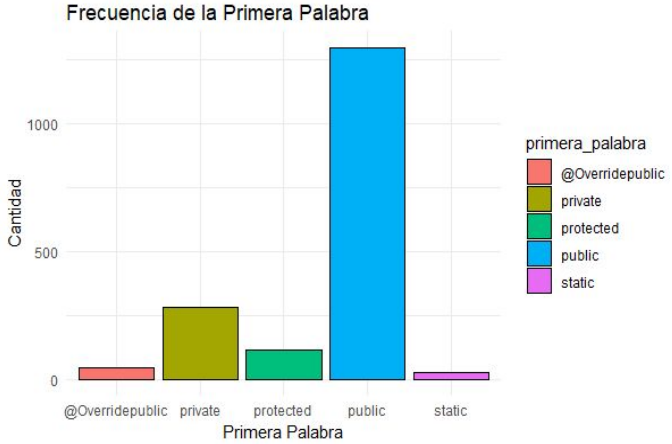
EDA

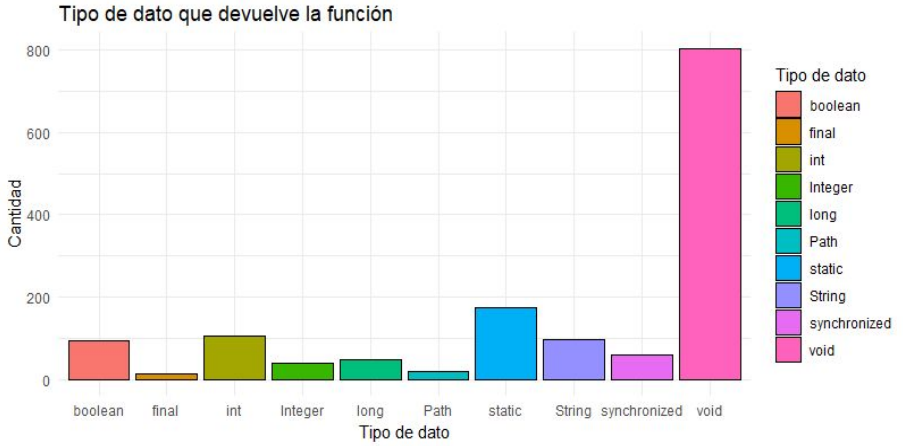

Cantidad de caracteres por función
count
1000
750
500
250
0
0
2000
4000
6000
longitud

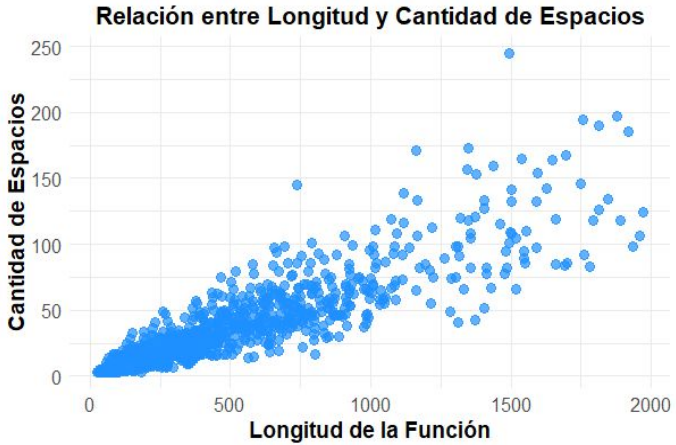

Cantidad de 'return' por función
Frecuencia
800
600
400
200
0
0
3
6
9
12
Total 'return' por función


Cantidad de espacios por función
Frecencia
300
200
100
0
5
10
15
Cantidad de espacios

EDA


Frecuencia de la Primera Palabra
Cantidad
1000
500
0
@Overridepublic
private
protected
public
static
Primera Palabra
primera_palabra
@Overridepublic
private
protected
public
static


Tipo de dato que devuelve la función
Cantidad
800
600
400
200
0
boolean
final
int
Integer
long
Path
static
String
synchronized
void
Tipo de dato
Tipo de dato
boolean
final
int
Integer
long
Path
static
String
synchronized
void


Relación entre Longitud y Cantidad de Espacios
Cantidad de Espacios
250
200
150
100
50
0
0
500
1000
1500
2000
Longitud de la Función

ITBA
Instituto Tecnológicc
de Buenos Aires

Modelos y herramientas

Modelos y herramientas

01 Bert

02 Word 2 Vec

03 Code 2 Vec

Bert

- Transformer bidireccional (utilizado para Masked Language Modeling)
- Inicio desde el modelo pre-entrenado bert-base-cased y realización de una adaptación de dominio, entrenándolo con el dataset (de métodos de Java)
- Para el entrenamiento: tokenización de entradas con el tokenizador del modelo. Agrupación en batches para entrenamiento y masking aleatorio.
- Validación de resultados: aplicando el mask token sólo al nombre de la función y evaluando su resultado.

Word 2 Vec

- Técnica de procesamiento que representa palabras como embeddings.
- Se utilizó para la evaluación de resultados de ambos modelos (propio y de referencia)
- Para la comparación: se separó el nombre de su formato camelcase a una lista de palabras *(por ejemplo ‘getName’ se transforma a [‘get’, ‘Name’])* y se utilizó word2vec para obtener un valor de similitud entre ambas listas.

Code 2 Vec

- Modelo de aprendizaje automático que se enfoca en entender el significado semántico de trozos de código.
- Funciona asignando vectores numéricos a secuencias de tokens en el código (permite capturar la relación entre los nombres de funciones y sus implementaciones)
- Colabora en tareas como buscar similitudes entre fragmentos de código, recomendaciones automáticas y comprensión de cómo se relacionan diferentes partes del código.
- Al ser el estado del arte, se utilizó como referencia para evaluar el desempeño de nuestro modelo.

ITBA
Instituto Tecnológicc
de Buenos Aires

Resultados

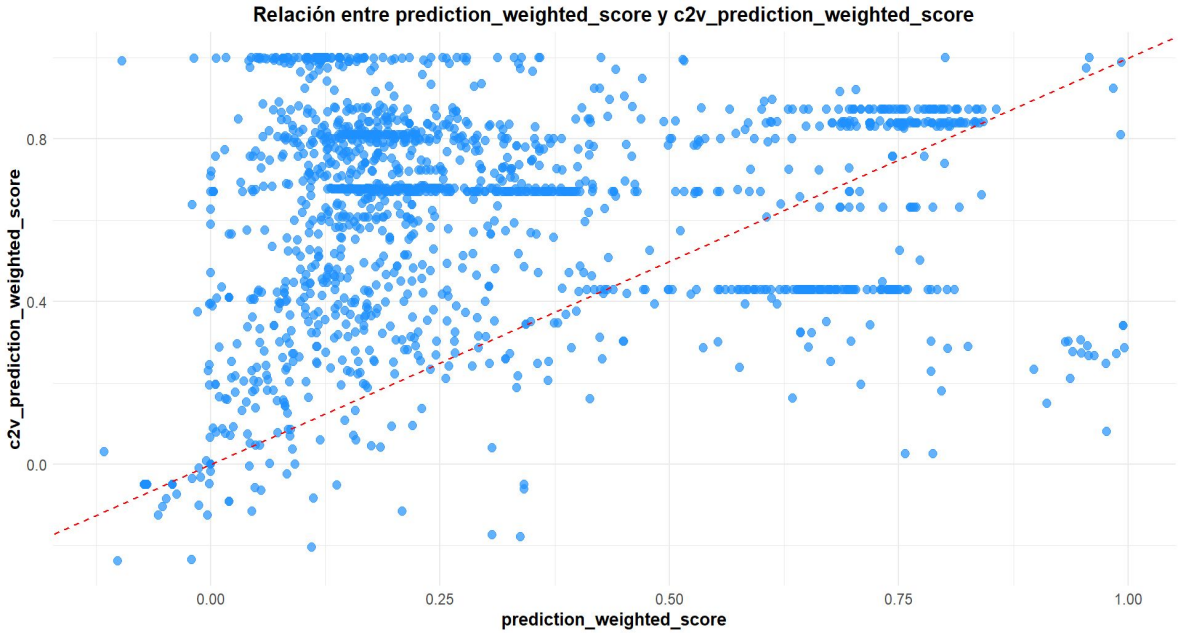
Armado de métricas

Analizando el problema, para determinar qué tan bueno es nuestro modelo, usamos las predicciones de **code2vec** para compararnos.

Como métricas importantes tomamos:

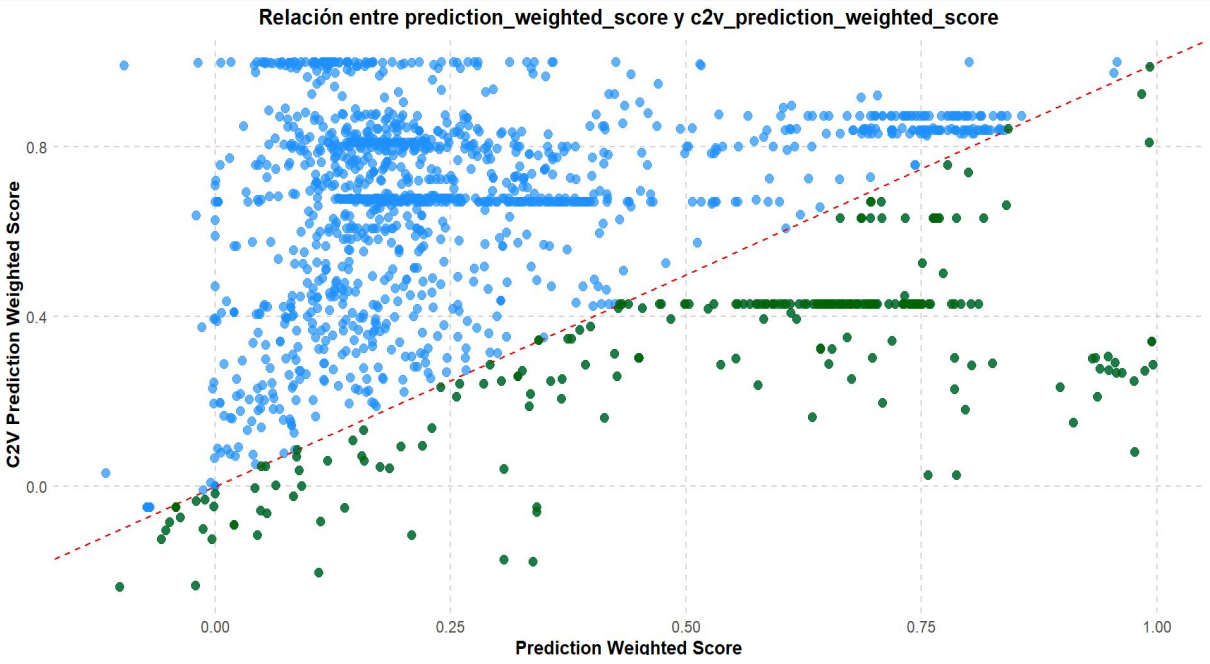
1. Cuantas predicciones estamos haciendo mejor
2. De las predicciones que dan mejor, cuanta diferencia hay con **code2vec**
3. De las predicciones que dan peor, cuanta diferencia hay con **code2vec**

Análisis de resultados



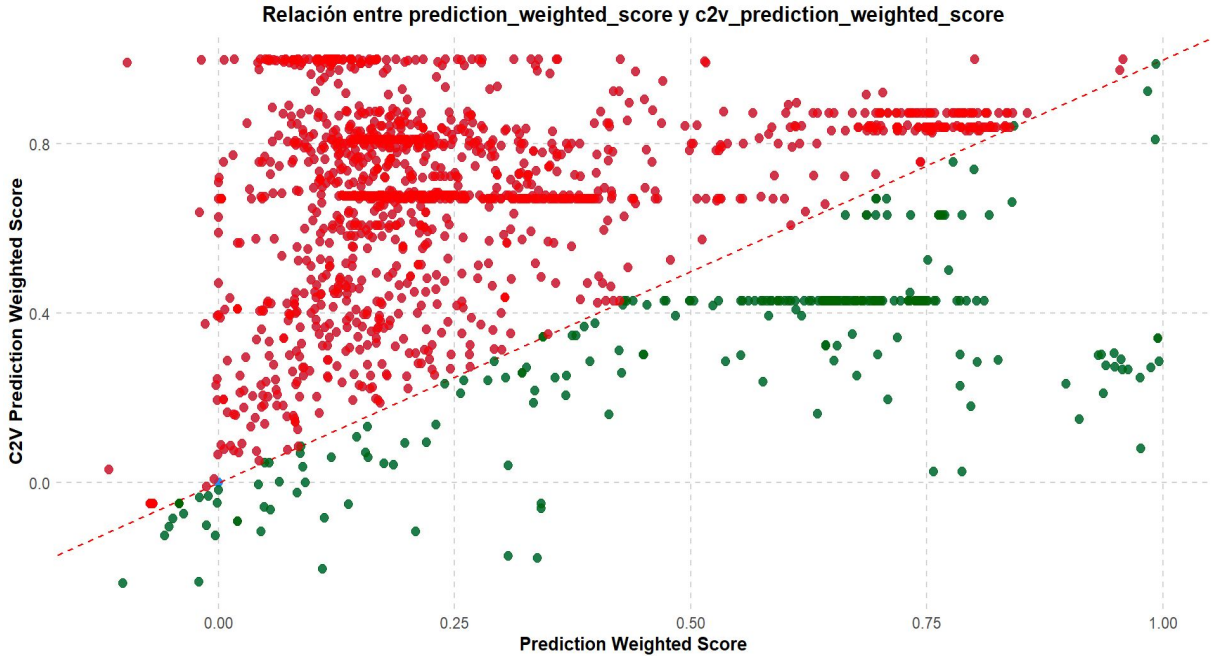
El 13,76% de los datos se encuentran bajo la curva

Análisis de resultados



En promedio, los puntos bajo la curva superan en 22,35 p.p a los datos sobre la curva

Análisis de resultados



En promedio, los puntos sobre la curva superan en 45,70 p.p a los datos bajo la curva

Análisis de resultados - Ejemplo

```
public void run() {
  setButtonDisabled(false);
  Gdx.app.log("NetAPITest", "HTTP requestcancelled");
  statusLabel.setText("HTTP request cancelled");
}
```

BERT

{'value': [***'run'***], 'probability': 0.568233847618103},
{'value': ['execute'], 'probability': 0.13582423329353333},
{'value': ['setup'], 'probability': 0.01746208593249321}

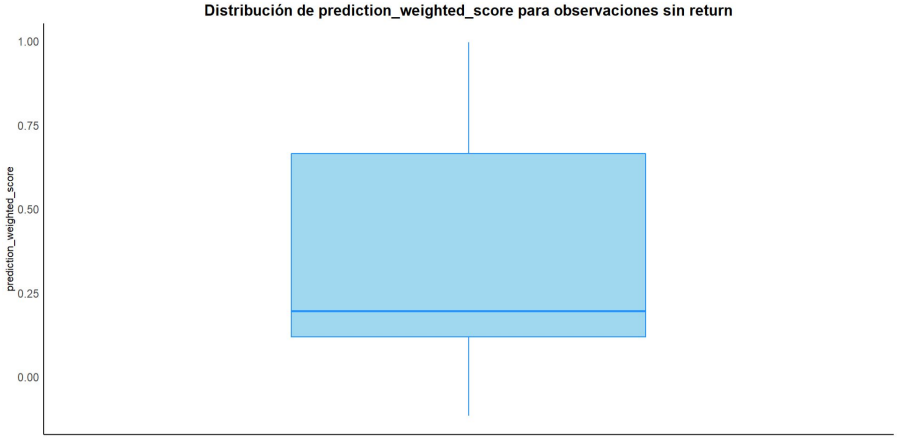
Score: 0.6344348

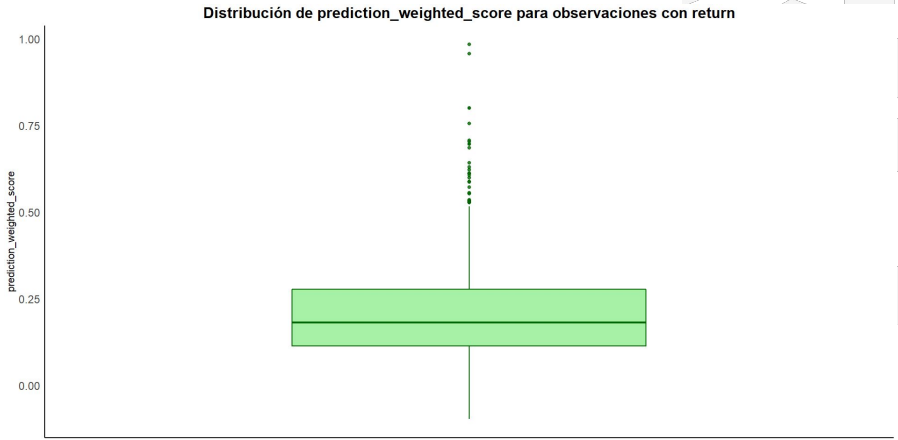
Code2vec

{'value': ['init'], 'probability': 0.22036169469356537},
{'value': ['refresh'], 'probability': 0.18205823004245758},
{'value': ['status'], 'probability': 0.14335161447525024}

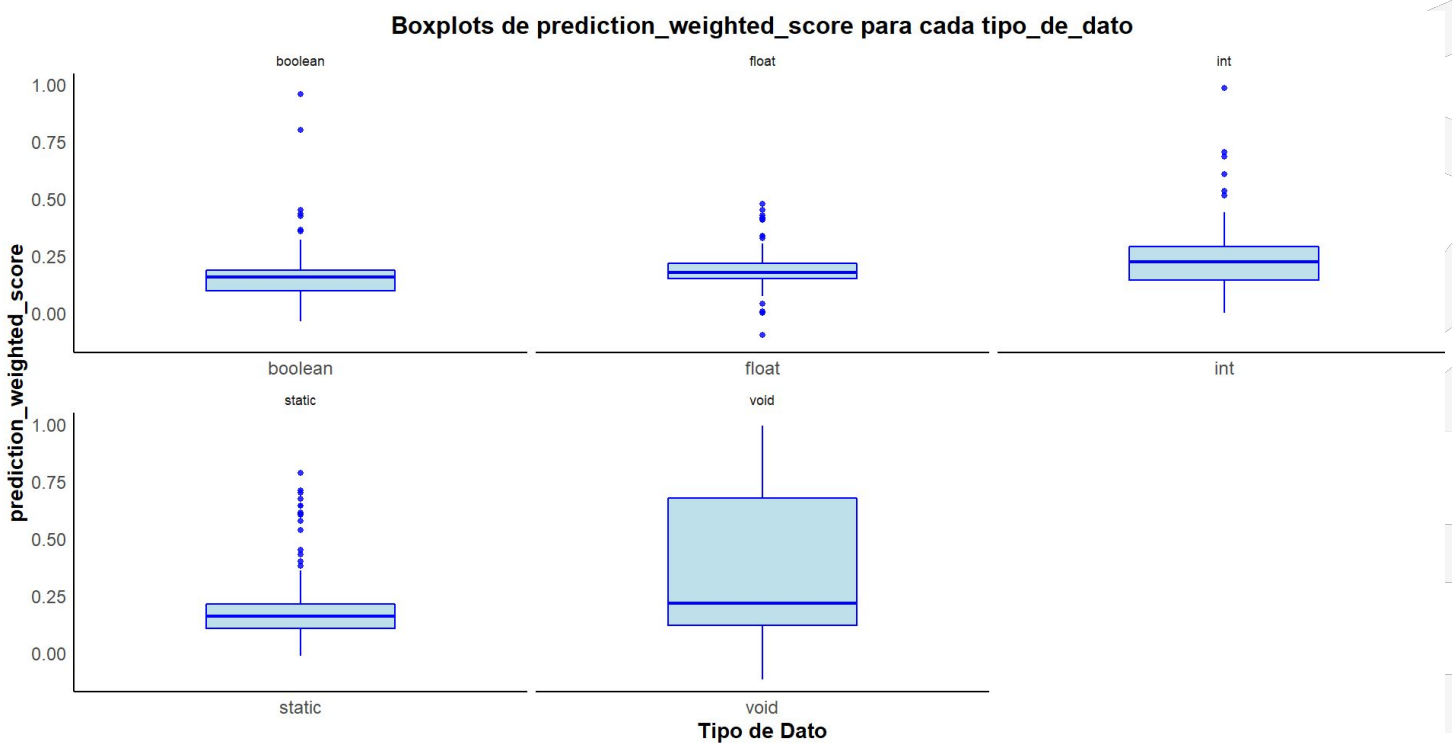
Score:0.1622433

Análisis de resultados





Análisis de resultados



ITBA
Instituto Tecnológicc
de Buenos Aires

Conclusiones

Conclusiones

- El modelo entrenado con BERT es capaz de devolver predicciones de nombres de función para una función dada, estando encaminado a los resultados de **code2vec** pero con un approach mucho más sencillo.
- El modelo funciona de forma mucho más consistente en las funciones que no retornan nada.

Desafíos y limitaciones actuales

- La perplexity actual de BERT (11.62), podría ser mejor.
- Con BERT solo se puede predecir una única palabra para el nombre de la función.
    - Para *countNumbers*, solo predice count o numbers

Mejoras futuras

- Probar diferentes tipos de entrenamiento (aplicar la mask sólo a los nombres de los métodos, entrenar los documentos por separado y no en batch, etc.)
- Entrenar con mayor cantidad de datos.
- Buscar una mejor performance (perplexity) para el fine-tunning.

ITBA
Instituto Tecnológicc
de Buenos Aires

¡Gracias!