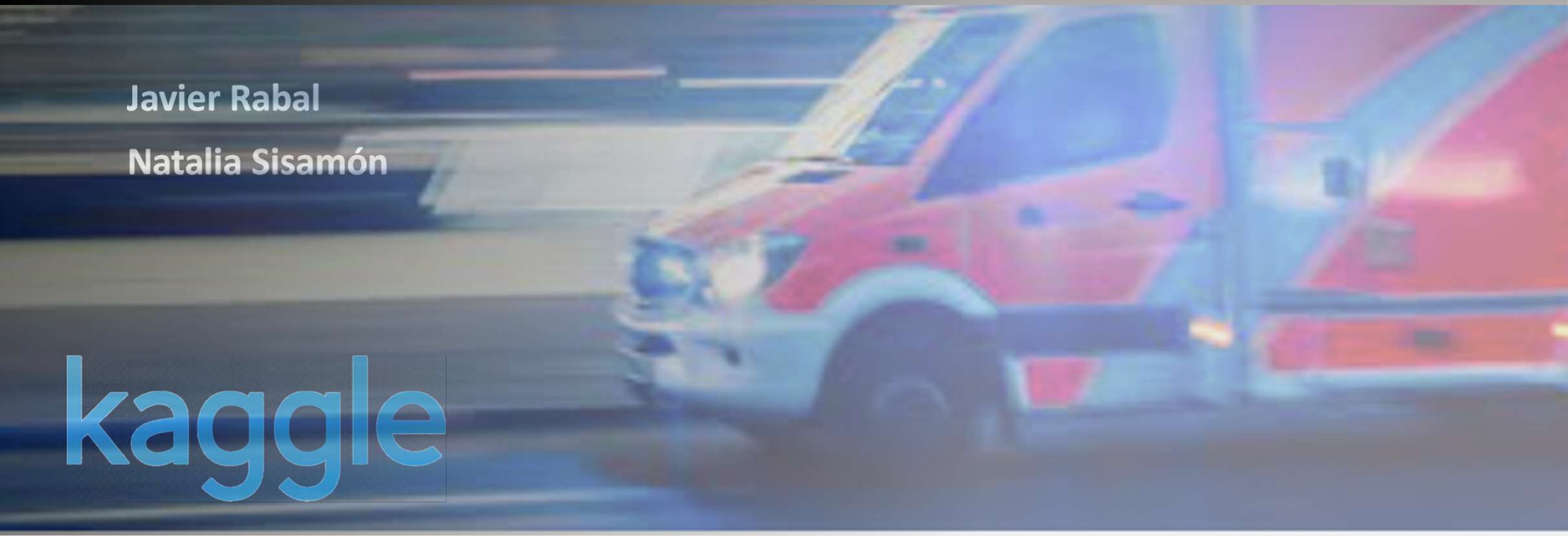
Natural Language Processing with Disaster Tweets

Javier Rabal

Natalia Sisamón



ÍNDICE

Fases de un proyecto de NLP

01 BUSINESS UNDERSTANDING

OBJETIVO

Predecir si un determinado tweet anuncia un desastre natural

KPI

F1 entre el valor predicho y la respuesta esperada

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

02 DATA PREPROCESSING

LIBRERIAS

- NLTK
- SpaCy
- Sklearn
- Texthero

02 DATA PREPROCESSING

Tokenizador

- From tensorflow.keras.preprocessing.test import Tokenizer
- From nltk.tokenize import RegexpTokenizer
- From nltk.tokenize import sent_tokenize
- From nltk.tokenize import Word_tokenize
- From Transformers import BertTokenizer

02 DATA PREPROCESSING

Clean text

Eliminar caracteres especiales:

- Def remove_special_caracteres (text):
  pattern = r'[^a-zA-Z]`
  text = re.sub(pattern, '', text)
  return text
- Def remove_emoji (text):
  emoji_pattern = re.compile(
  '['
  u'\U0001F600-\U0001F64F' # emoticons
  u'\U0001F300-\U0001F5FF' # symbols & pictographs
  u'\U0001F680-\U0001F6FF' # transport & map symbols
  u'\U0001F1E0-\U0001F1FF' # flags (iOS)
  u'\U00002702-\U000027B0'
  u'\U000024C2-\U0001F251'
  ']+',
  flags=re.UNICODE)
  return emoji_pattern.sub(r'', text)

02 DATA PREPROCESSING

Clean text

Eliminar caracteres especiales:

- ```
  Def remove_url (text):
    url = re.compile(r'https?://\S+|www\.\S+')
    return url.sub(r'', text)
  ```

Poner las frases en minusculas:

- `.lower()`
  ```

02 DATA PREPROCESSING

Clean text

Lematización:

- `From nltk.stem import WordNetLemmatizer`

Named entity recognition (NER) :

- `Spacy.load('en-core-web-sm')`

Eliminar stop-words:

- `From nltk.corpus.stopwords.words('english')`

02 DATA PREPROCESSING

Clean text

Hay algunas librerías que permiten hacer todo lo anterior en un solo paso :

- `From sklearn.feature_extracción.text import CountVectorizer:`
  - `CountVectorizer( lowercase=True, stop_words='english', tokenizer)`
- `From texthero import prepocessing as ppe:`
  - `pipeline= [ppe.fillna, ppe.lowercase, ppe.remove_puntuaciton, ppe.remove_whitespace, ppe.remove_urls, ppe.remove_stop_words, ppe.remove_digits]`
  - `texthero.clean(data,pipeline)`

03 MODEL BUILDING

Algoritmos de clasificación

- `from sklearn.naive_bayes import MultinomialNB`
- `from sklearn.naive_bayes import MultinomialNB`
- `from sklearn.naive_bayes import BernoulliNB`
- `from sklearn.linear_model import LogisticRegression`
- `from sklearn.linear_model import SGDClassifier`
- `from sklearn.ensemble import RandomForestClassifier`
- `from sklearn.svm import SVC`

```
from
nltk.classify.scikit
learn import
SklearnClassifier
```

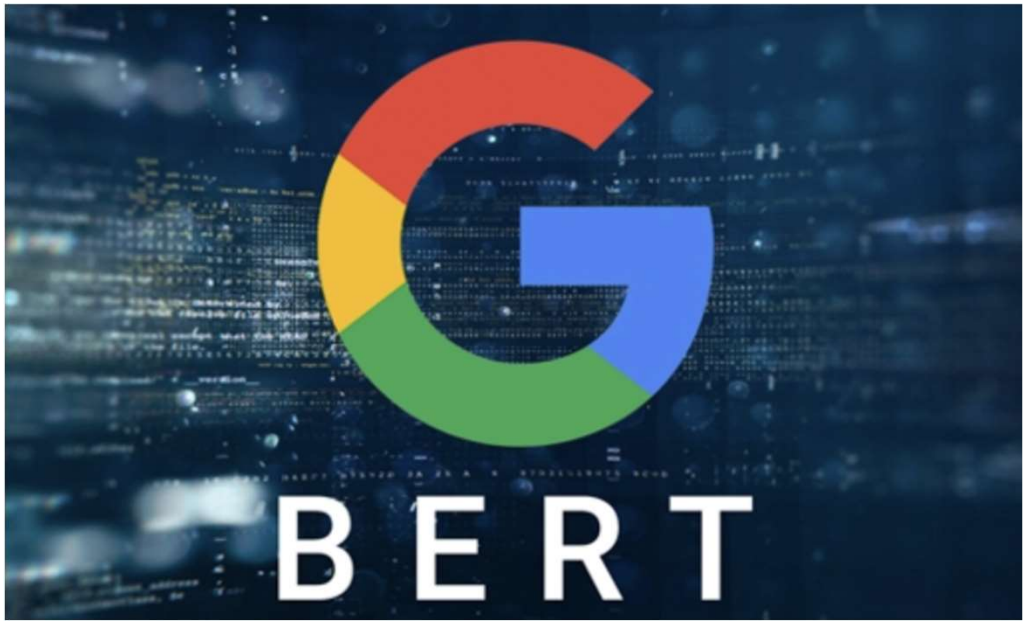
03 MODEL BUILDING

Redes neuronales

- Red LSTM
- Red GRU
- Red BERT
  - Base-case
  - Base-uncase
  - Large-uncase

03 MODEL BUILDING

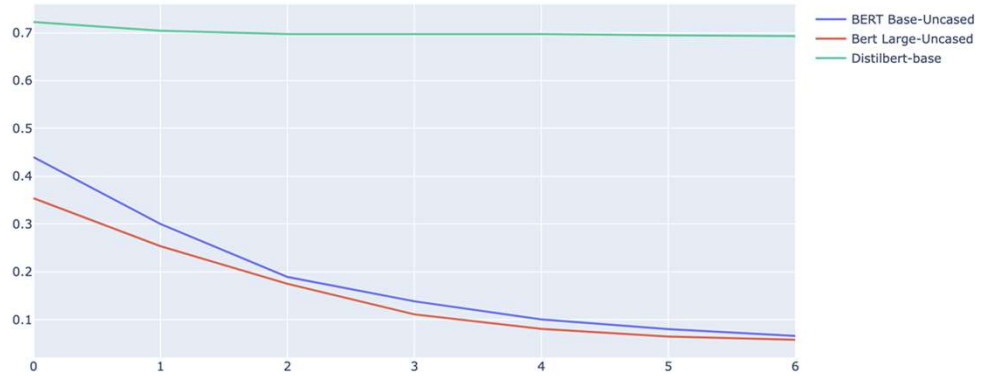
Mejor modelo: BERT



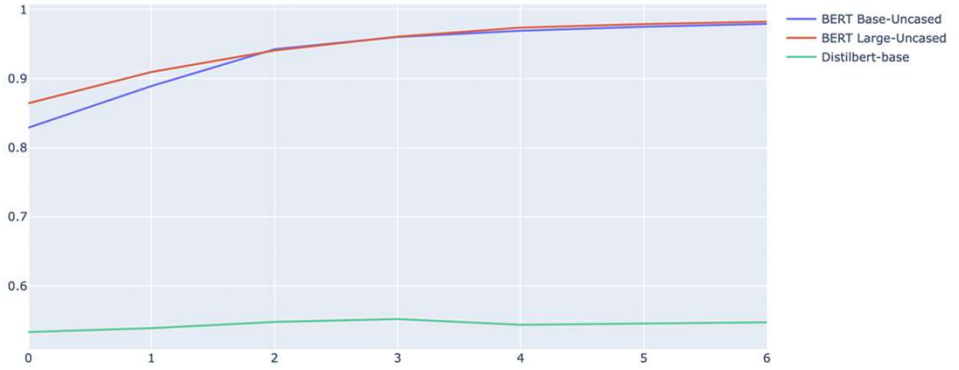
03 MODEL BUILDING

Mejor modelo: BERT

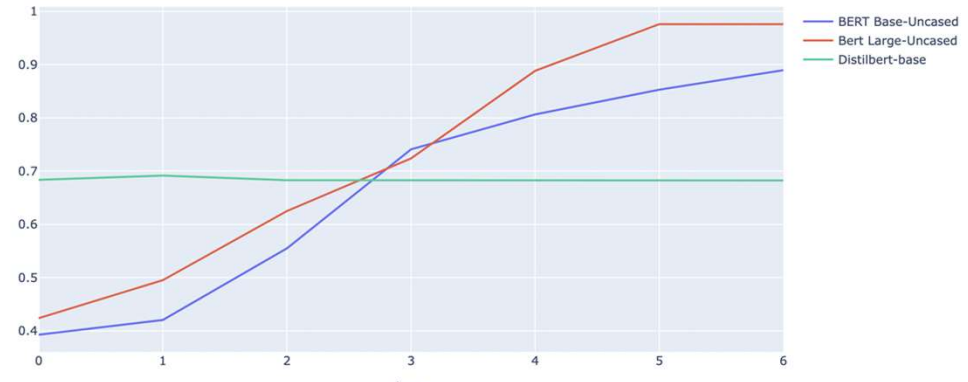
Loss segun Epoch Train



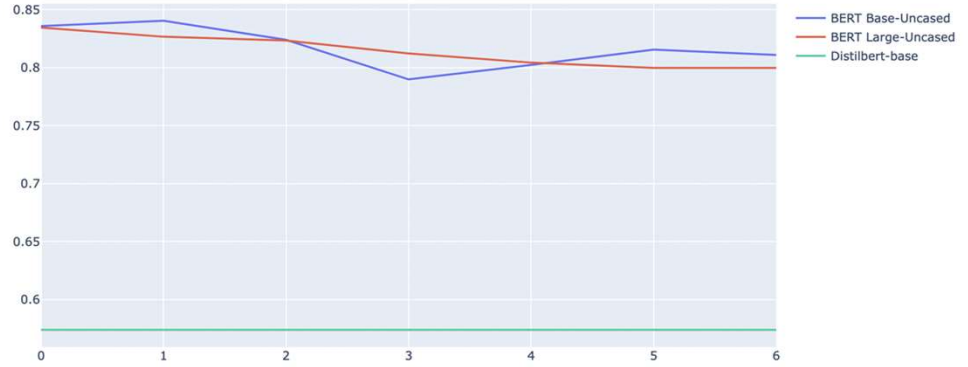
Accuracy segun Epoch Train



Loss segun Epoch Test



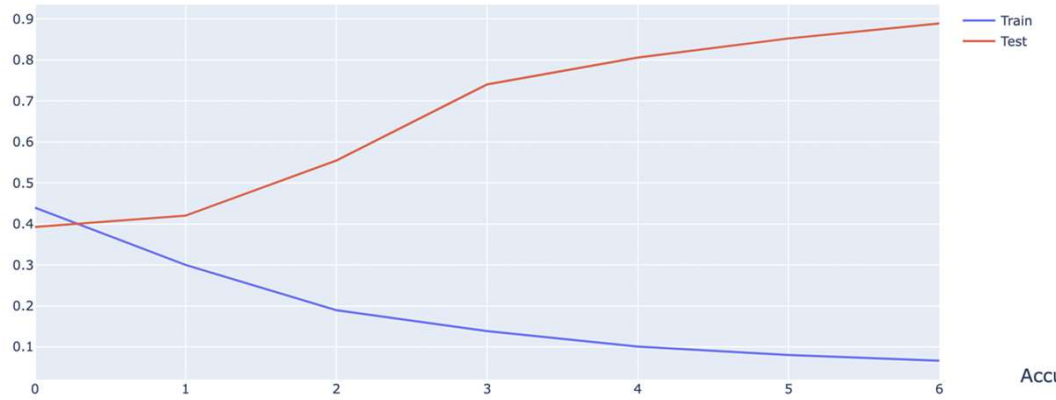
Accuracy segun Epoch Test



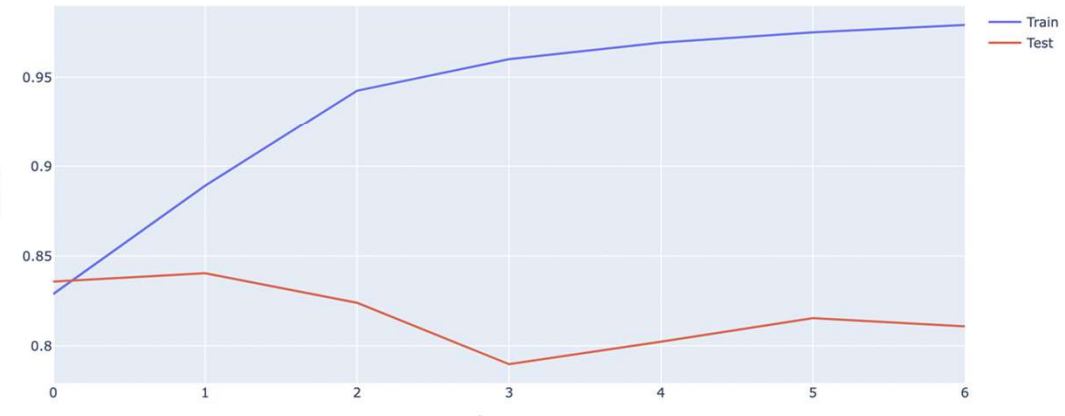
03 MODEL BUILDING

Mejor modelo: bert-base-uncased

Loss segun Epoch



Accuracy segun Epoch

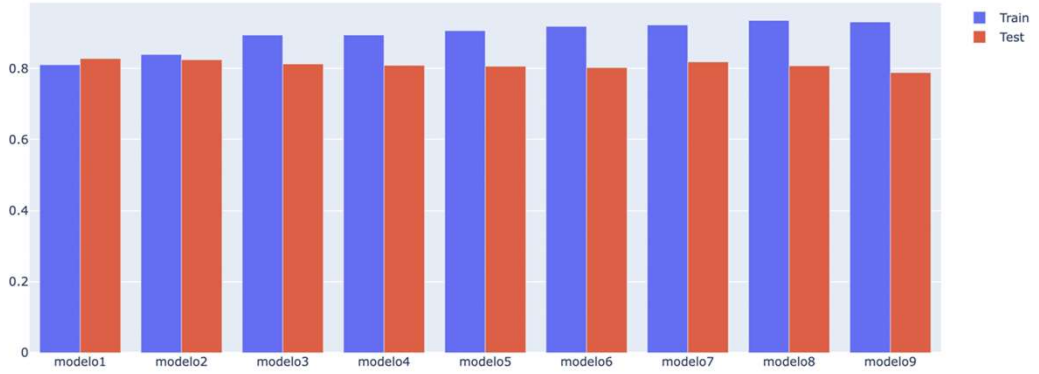


03 MODEL BUILDING

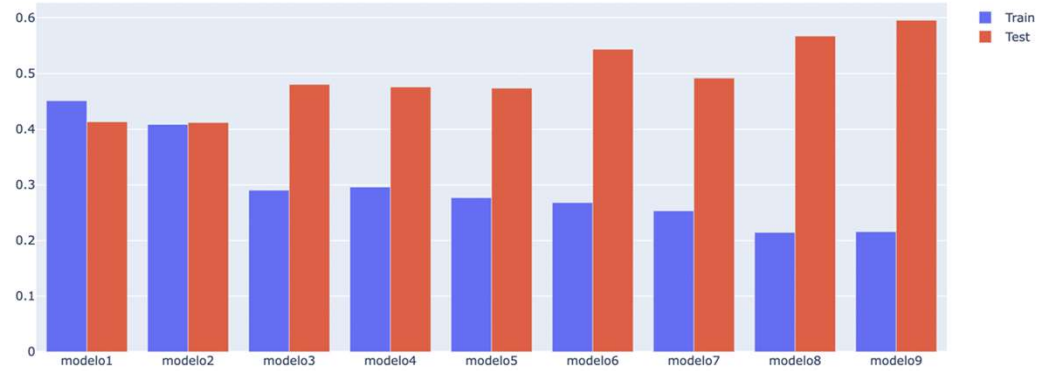
Mejor modelo: bert-base-uncased

NOMBRE	LEARNING_RATE	MAX_LEN	BACH_SIZE
Modelo1	0.00002	200	16
Modelo2	0.00003	200	16
Modelo3	0.000006	200	16
Modelo4	0.00002	80	16
Modelo5	0.00002	120	16
Modelo6	0.00002	160	16
Modelo7	0.00002	120	24
Modelo8	0.00002	120	32
Modelo9	0.00002	120	64

Accuracy de los diferentes modelos

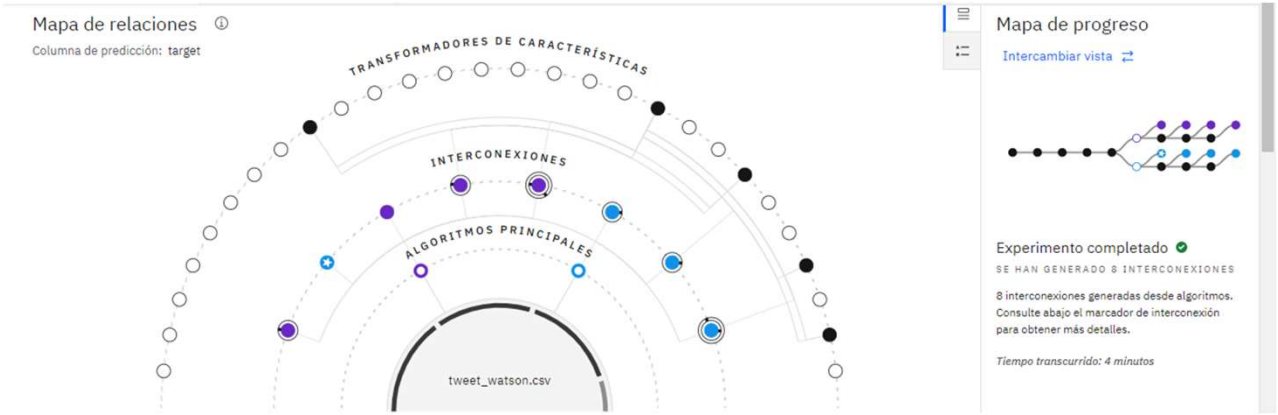


Loss de los diferentes modelos



SOLUCIONES CLOUD

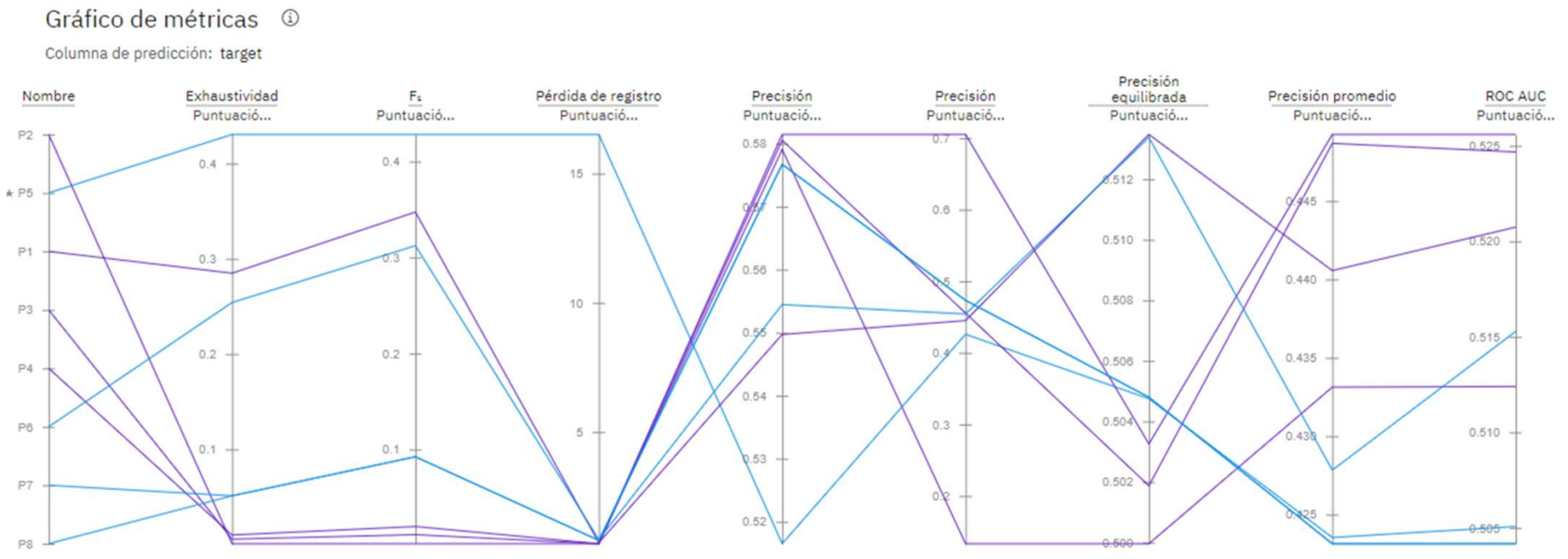
IBM WATSON STUDIO



	Clasificación	Nombre	Algoritmo	Precisión (Optimizado) Validación Cruzada	F_1 Validación Cruzada	Mejoras	Tiempo de creación
	6	4 interconexión	Clasificador XGB	0.581	0.020	HPO-1 FE HPO-2	00:00:39
	7	3 interconexión	Clasificador XGB	0.581	0.011	HPO-1 FE	00:00:29
	8	2 interconexión	Clasificador XGB	0.579	0.002	HPO-1	00:00:15
	4	7 interconexión	Clasificador de árbol de decisiones	0.577	0.092	HPO-1 FE	00:00:30
	5	8 interconexión	Clasificador de árbol de decisiones	0.577	0.092	HPO-1 FE HPO-2	00:00:10
	3	6 interconexión	Clasificador de árbol de decisiones	0.554	0.312	HPO-1	00:00:03
	2	1 interconexión	Clasificador XGB	0.550	0.347	*Ninguno*	00:00:01
★	1	5 interconexión	Clasificador de árbol de decisiones	0.517	0.428	*Ninguno*	00:00:01

IBM **Cloud**

IBM WATSON STUDIO



IBM Cloud

IBM WATSON STUDIO

- **NATURAL LANGUAGE UNDERSTANDING**

```
 8 from ibm_watson import NaturalLanguageUnderstandingV1
 9 from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
10 from ibm_watson.natural_language_understanding_v1 import Features, ClassificationsOptions
```

```
1 training_data_filename = 'train_aux.json'
2 with open(training_data_filename, 'rb') as file:
3     model = nlu.create_classifications_model(language='en', training_data=file,
4                                              training_data_content_type='application/json',
5                                              name='MyClassificationsModel',
6                                              model_version='1.0.1').get_result()
7     print("Created a NLU Classifications model:")
8     print(json.dumps(model, indent=4))
```

```
Information about the created NLU Classifications model:
{
    "name": "MyClassificationsModel",
    "user_metadata": null,
    "language": "en",
    "description": null,
    "model_version": "1.0.1",
    "version": "1.0.1",
    "workspace_id": null,
    "version_description": null,
    "status": "available",
    "notices": [],
    "model_id": "fa578cf9-173d-4ad5-b7cb-a0b4c7193ed8",
    "features": [
        "classifications"
    ],
    "created": "2022-06-16T09:48:33Z",
    "last_trained": "2022-06-16T09:48:33Z",
    "last_deployed": "2022-06-16T11:30:42Z"
}
```

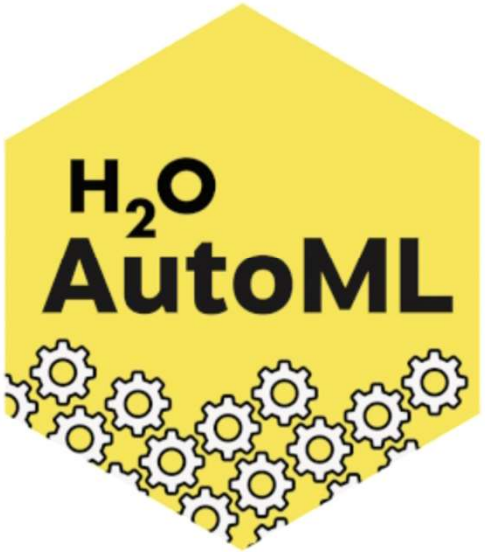
IBM Cloud

OTRAS SOLUCIONES

Amazon SageMaker

Google AI

LUIS



CLASIFICACIÓN

PUESTO	SCORE	JUGADORES TOTALES	PORCENTAJE
97	0.84033	957	10%