UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS
MAESTRÍA EN CIENCIA DE DATOS

APRENDIZAJE AUTOMÁTICO

Profesor: Jose Anastacio Hernandez Saldaña

Proyecto Final

Dora Alicia Guevara Villalpando
Matrícula: 1551003
Grupo: 003

28 de julio de 2024

UANL
UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FCFM
FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

Proyecto Final

Introducción.

El presente documento presenta los resultados obtenidos durante el proceso de realizar la Tarea 3 de Aprendizaje Automático. Esta tarea consiste en realizar el modelo de clasificación a una base de datos de nuestro interés.

Contexto.

Utilizando la base de conjunto de datos proporcionada encuentra un modelo de clasificación con sus parámetros utilizando cross validation con el criterio ROC_auc que te de un valor mayor al 0.75 en el conjunto de validación y prueba.

Desarrollo.

La base de datos cuenta con 9,239 filas y 10 columnas:

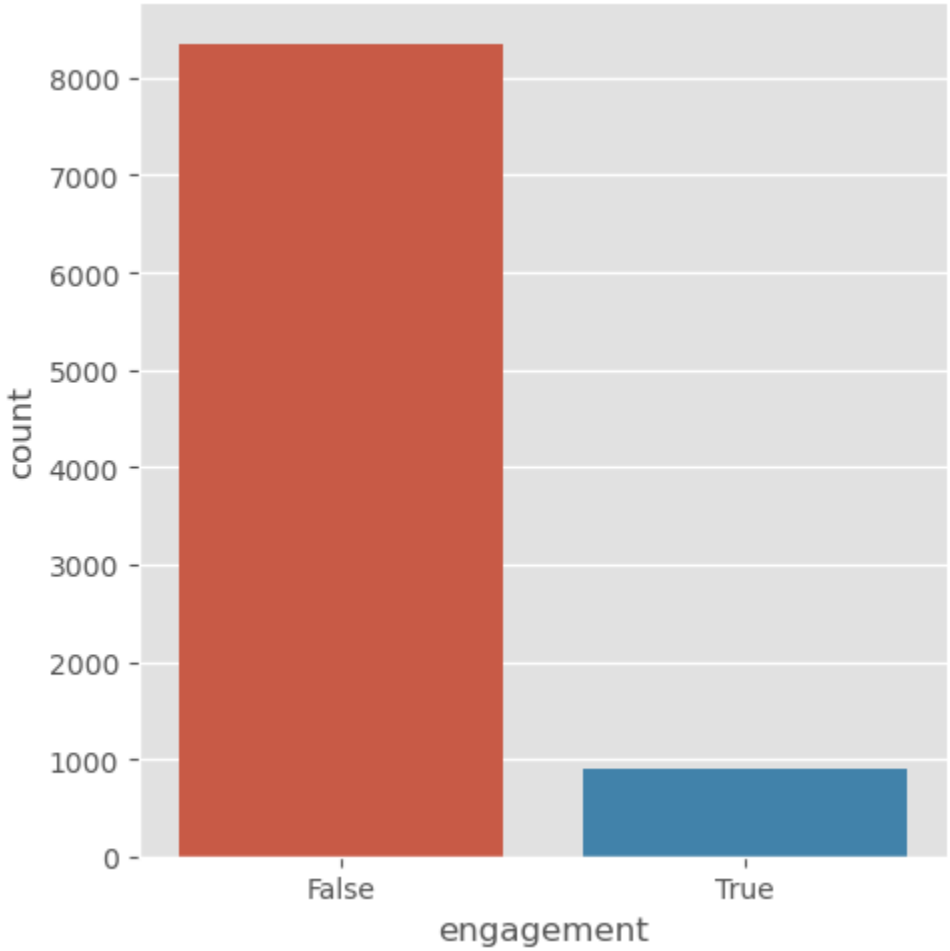
id	title_word_count	document_entropy	freshness	easiness	fraction_stopword_presence	normalization_rate	speaker_speed	silent_period_rate	engagement
1	9	7.753995	16310	75.583936	0.553664	0.034049	2.997753	0.0	True
2	6	8.305269	15410	86.870523	0.584498	0.018763	2.635789	0.0	False
3	3	7.965583	15680	81.915968	0.605685	0.030720	2.538095	0.0	False
4	9	8.142877	15610	80.148937	0.593664	0.016873	2.259055	0.0	False
5	9	8.161250	14920	76.907549	0.581637	0.023412	2.420000	0.0	False

Se analizó la base de datos para detectar si tiene valores nulos, dando como resultado que no se cuentan con valores nulos, por lo que se puede proceder con los análisis.

```
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   id                         9239 non-null   int64
 1   title_word_count           9239 non-null   int64
 2   document_entropy           9239 non-null   float64
 3   freshness                  9239 non-null   int64
 4   easiness                   9239 non-null   float64
 5   fraction_stopword_presence 9239 non-null   float64
 6   normalization_rate         9239 non-null   float64
 7   speaker_speed              9239 non-null   float64
 8   silent_period_rate         9239 non-null   float64
 9   engagement                 9239 non-null   bool
```

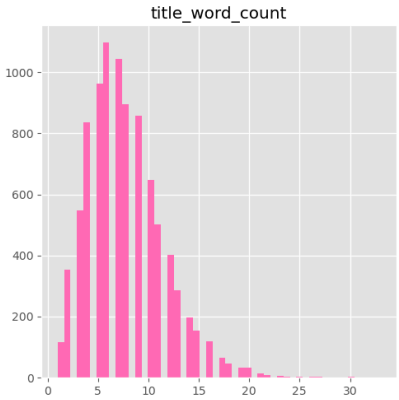
Variable Objetivo.

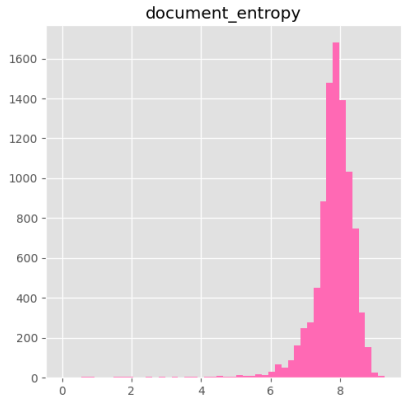
La variable objetivo es **engagement** y tiene la siguiente distribución:

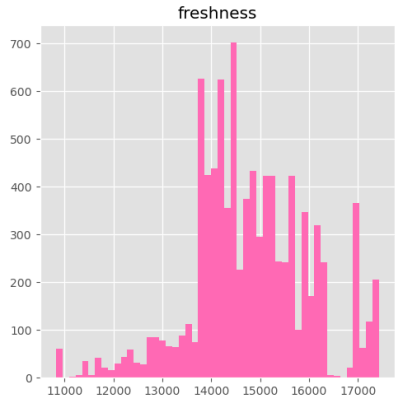


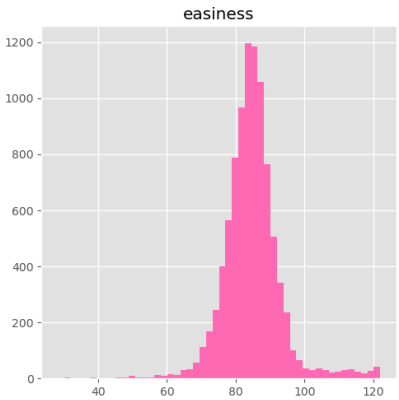
Variables cuantitativas.

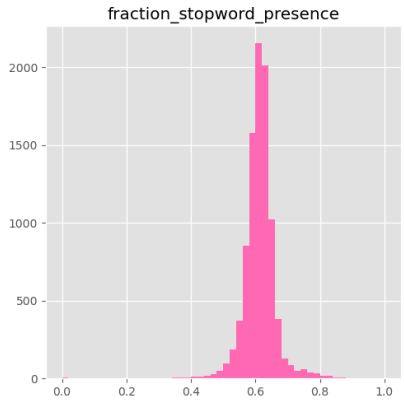
Graficamos las variables para revisar si se puede deducir características o patrones de estas:

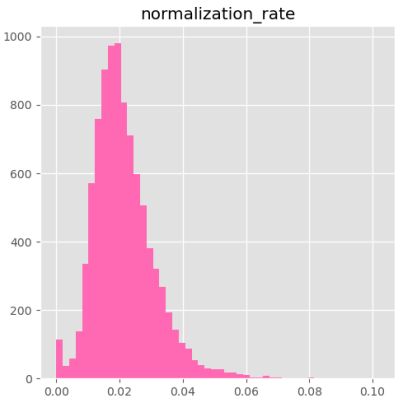


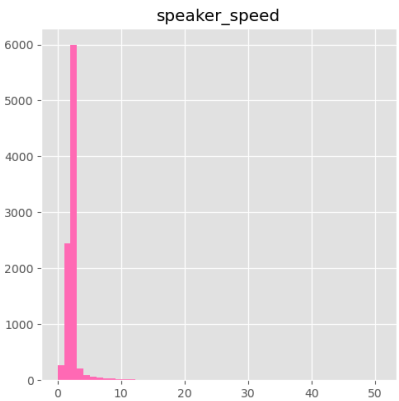


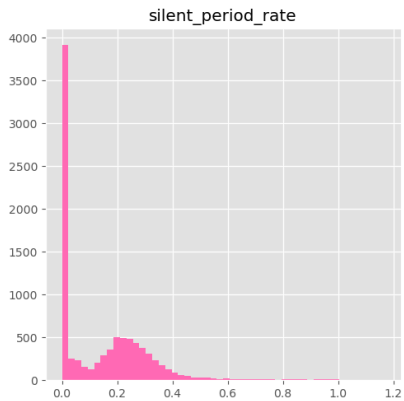






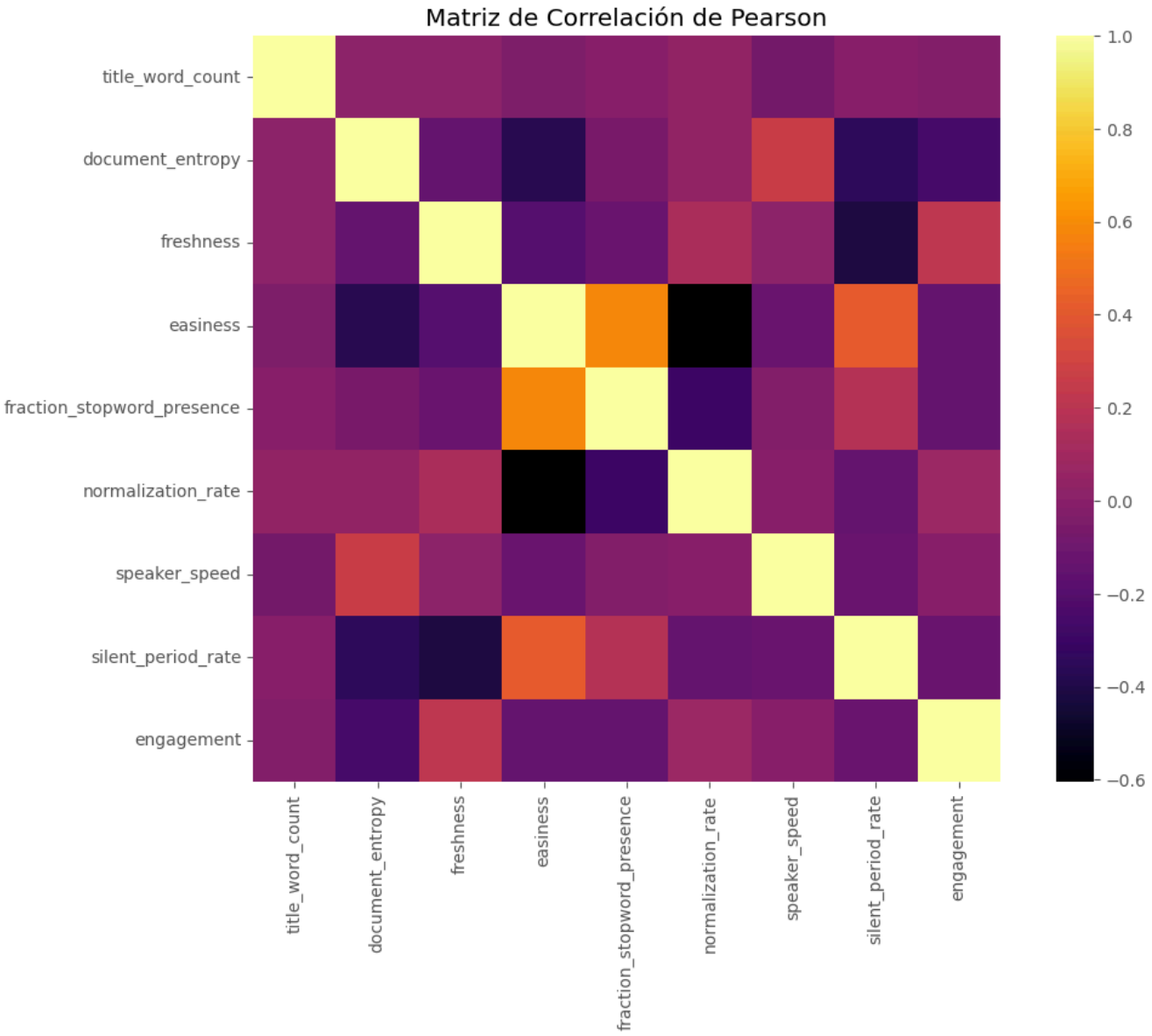






Correlación.

Obtenemos el mapa de calor de la correlación de Pearson para identificar las variables que tengan mayor relación entre ellas.



Las variables con mayor relación entre ellas son:

- *easiness* con *normalization_rate* con una correlación negativa,
- *easiness* también se relaciona con *fraction_stopword_presence* y *silent_period_rate* de forma positiva.

Análisis del modelo.

Preparamos la base de datos para poder dividir la base en dos conjuntos: train y test.

```
# Variables numéricas. En este modelo representan las variables independientes.

X = df[['title_word_count', 'document_entropy', 'freshness', 'easiness', 'fraction_stopword_presence',
     'normalization_rate', 'speaker_speed', 'silent_period_rate']]


# Variable Categórica. La que queremos predecir.

y = df[['engagement']]
```

Escogemos el 80% de los datos para entrenar el modelo (train) y el 20% restante para probarlo (test).

Decision Tree Classifier

Primero se analizó el árbol de decisiones:

```
model = DecisionTreeClassifier()
```

```
model.fit(X_train, y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
# Generar la predicción

y_pred = model.predict(X_test)
```

```
# Evaluar con las métricas referenciales

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy DATOS ORIGINALES: %.2f%%" % (accuracy * 100.0))
```

```
Accuracy DATOS ORIGINALES: 89.99%
```

El modelo que se obtiene tiene un accuracy del 89.99%, el cuál se considera bueno.

Adicional, para este modelo se realizó una validación cruzada con K-Folds, el anterior se ejecutó 10 veces obteniendo los siguientes porcentajes de ***accuracy***:

No.	Accuracy
1	91.45%
2	87.88%
3	89.29%
4	88.85%
5	89.50%
6	90.04%
7	88.74%
8	89.50%
9	89.17%
10	88.62%

En la tabla anterior se observa que todos los modelos que se ejecutaron tienen un accuracy de más de 87%, por lo que estos modelos se consideran buenos y, en el caso del primero, tiene un accuracy de casi el 91% por lo que se considera excelente.

Logistic Regression

La regresión logística tiene como objetivo resolver problemas de clasificación y lo hace prediciendo resultados categóricos, a diferencia de la regresión lineal que predice un resultado continuo.

En el caso analizado hay dos resultados, lo que se denomina binomial.

```
logr = LogisticRegression()
```

```
logr.fit(X_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
y_pred2 = logr.predict(X_test)
```

```
accuracy = accuracy_score(y_test, y_pred2)
print("Accuracy DATOS ORIGINALES: %.2f%%" % (accuracy * 100.0))
```

```
Accuracy DATOS ORIGINALES: 90.69%
```

Se puede observar que el modelo tiene un accuracy de casi el 91%; procedemos a obtener el **ROC_auc score**. Tomemos en cuenta lo siguiente:

- La curva ROC muestra el rendimiento de un clasificador binario con diferentes umbrales de decisión. Representa gráficamente la tasa de verdaderos positivos frente a la tasa de falsos positivos.
- La puntuación **ROC_auc** es el área bajo la curva ROC. Resume la capacidad de un modelo para generar puntuaciones relativas para discriminar entre instancias positivas o negativas en todos los umbrales de clasificación.
- La puntuación **ROC_auc** varía de 0 a 1 , donde 0.5 indica una conjetura aleatoria y 1 indica un rendimiento perfecto.

Al analizar el modelo de regresión logística obtuvimos un accuracy del 90.69%, el cual se considera bueno. A continuación se mostrará el valor del ROC_auc de este modelo:

```
roc_auc_score(y_test, logr.predict_proba(X_test)[:, 1])
roc_auc_score(y_test, logr.decision_function(X_test))
```

```
0.847451556647509
```

Tenemos un valor de ROC_auc = 0.85.

De forma gráfica: