30 DE ABRIL DE 2023

Departamento de Ingeniería de Sistemas y Computación
ISIS 3301 – Inteligencia de Negocios
Proyecto Analítica de textos – Parte 2

Estudiantes:

Julian Andres Mendez Melo	201920623	j.mendezm@uniandes.edu.co
María Alejandra Vargas Torres	201123148	ma.vargas73@uniandes.edu.co
Omar Esteban Vargas Salamanca	201921271	o.vargas@uniandes.edu.co

Tabla de contenido

1. Proceso de Automatización

1.1. Preparación de los datos

Para el proceso se propusieron dos formas en las que el usuario puede interactuar con la página para obtener el resultado de la reseña (ver sección 2). El primero es introducir solo 1 reseña, mientras que el segundo consiste en introducir un csv y retorna otro con la clasificación de cada una.

1.1.1. Limpieza y calidad

Utilizando lo realizado en la primera parte del proyecto y teniendo en cuenta la automatización del proceso se crearon funciones con cada uno de los pasos de la limpieza y que se asegurara así la calidad de los datos. El siguiente procedimiento describe el caso cuando se sube un archivo csv[1]:

a) Se transforma el archivo a un DataFrame utilizando la librería pandas, este se le pasa luego a una clase que está diseñada para realizar la limpieza y la transformación de los datos para aplicar los algoritmos.
b) Se realiza un perfilamiento de los datos, se obtienen las filas y columnas del archivo.
c) Se evalúa que no se tengan columnas sin reseñas o con espacios en blanco, si se presenta alguno de estos casos se elimina este valor.
   Función: `delete_null(self, df: pd.DataFrame)`
d) Se comprueba que no se tengan duplicados entre las reseñas, si se presenta más de una ocurrencia se deja solo la primera.
   Función: `duplicates(self, df: pd.DataFrame)`
e) Para reconocer el idioma se utiliza la librería langdetect, se utiliza la función detect en cada fila y su resultado se adjunta a una nueva columna que guarda los idiomas.
   Función: `df['Language'] = [detect(x) for x in df['review_es']]`
f) Con el resultado del literal anterior se eliminan los que no corresponden al idioma español, de forma que se cumpla con la restricción del diccionario dada por el negocio.
   Función: `deleteDiferents(self, df: pd.DataFrame)`

Para el caso en que el usuario introduce una reseña el paso ***e*** cambia ya que en la limpieza se identifica el idioma de la review y lo traduce a español apenas se introduce el texto. Además, un error que se presentaba al intentar realizar la predicción es cuando se tienen las comillas, por lo cual la función `clean_quotes()` las elimina.

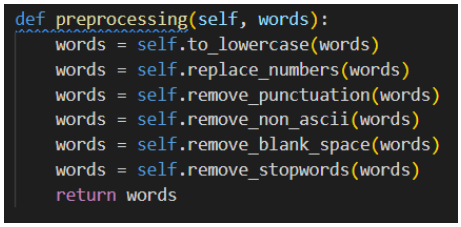
```
class EnglishTranslator:
    def __init__(self):

        self.translator2  = GoogleTranslator(source=

    def is_english(self, text):
        if detect(text) == 'en':
            return self.translator2.translate(text)
        else:
            return text

    def clean_quotes(self,text):

        texto_limpio = text.replace("'", '')
        texto_limpio = text.replace('"', '')
        return texto_limpio
```

Figura 1. Clase(Modulo) de limpieza de datos en el front previo al Post del archivo Traductor.py

[1] Las funciones que se referencian corresponden al archivo transformer.py

1.1.2. Preparación antes del algoritmo, transformación

Con la limpieza realizada se debe realizar la transformación de los datos para que los algoritmos funcionen correctamente, en las siguientes líneas de código se evidencia el llamado de las funciones de tokenización, el preprocesamiento que consiste en remover los caracteres que no son ASCII, pasar todo a minúsculas, quitar la puntuación, reemplazar los números por su equivalente textual, se remueven las *stop words* y finalmente si hay espacios en blanco se quitan de forma que no se afecte cuando se realice la lematización ya que este es el siguiente paso, se vuelven a unir las palabras y y debido a que en este proceso puede aparecer nuevamente algunas *stop words,* se vuelve a llamar a la función. Finalmente, de este se vuelven a unir las palabras con un `.join()` y se retorna el nuevo DataFrame. En la figura 2 se muestra el orden de estos pasos:



```
def preprocessing(self, words):
    words = self.to_lowercase(words)
    words = self.replace_numbers(words)
    words = self.remove_punctuation(words)
    words = self.remove_non_ascii(words)
    words = self.remove_blank_space(words)
    words = self.remove_stopwords(words)
    return words
```

Figura 2. Clase(Modulo) de limpieza de datos y preprocesamiento en la clase transformer.py (líneas 193-202)

El procedimiento anterior se guarda en un log de forma que si ocurre algún error se pueda identificar la parte específica en la que aparece para tener una corrección más rápida. A continuación, se muestra su implementación:

1.2. Persistencia del modelo analítico

Los Modelos Disponibles son:

- Reg. Logística
- Multinomial NB
- Red Neuronal

Para la generación de dichos modelos, en primer lugar, se realiza una búsqueda de hiperparametros acorde a los datos de prueba para optimizar las métricas de los modelos y después se persisten mediante un pipeline y se exportan con el uso de joblib. A continuación, se muestra su generación para los modelos de Reg. Logística, Multinomial NB y Red neuronal respectivamente.

Regresión logística

```
#busqueda de hiperparametros
lr = LogisticRegression()
param_grid = {'lr__solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag'],
              'lr__tol': [0.0001, 0.001, 0.01]}

grid_search_lr = GridSearchCV(pipeline_lr, param_grid, cv=5)
grid_search_lr.fit(X_train, y_train)
```

```
#Generacion Pipeline
pipeline_lr = Pipeline(steps=[('vectorize', TfidfVectorizer(max_features=3000)),
                        ('lr', LogisticRegression())])

# Pipeline resultante
best_model = grid_search_lr.best_estimator_
pipeline = best_model.fit(X_train,y_train)
pipeline
```

Multinomial

```
# Modelo Multinomial
pipeline_m = Pipeline(steps=[('vectorize', TfidfVectorizer()),
                        ('classifier', MultinomialNB())])
# Realizar búsqueda de hiperparámetros
parameters_m = {'classifier__alpha': np.linspace(1, 1.5, 100).tolist(),
                'classifier__fit_prior': [True, False]}
grid_search_m= GridSearchCV(pipeline_m, parameters_m, cv=20, scoring='accuracy',
                            n_jobs=-1)
grid_search_m.fit(X_train, y_train)

# Entrenar modelo
model_m  = grid_search_m.best_estimator_
m = model_m['classifier'].fit(train_X_Tfidf_m.toarray(), y_train)

# Generación pipeline
pipeline_m = Pipeline(steps=[('vectorize', TfidfVectorizer()),
                        ('classifier', m)])
```

Red Neuronal

```
# Modelo Red Neuronal
def keras_model(neurons1=1, neurons2=1, dropout_rate=0.0):
    model = Sequential()
    model.add(Dense(neurons1, input_dim=train_X_Tfidf_rn.shape[1], activation='relu'))
    model.add(Dropout(dropout_rate))
    model.add(Dense(neurons2, activation='relu'))
    model.add(Dropout(dropout_rate))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

# crear el modelo para GridSearchCV
model = keras_model()
```

```
rn = KerasClassifier(model=lambda: model, epochs=10, batch_size=32, verbose=0,
neurons1=1, neurons2=1, dropout_rate=0.0)


pipeline_rn = make_pipeline(TfidfVectorizer(max_features=10000),
                          KerasClassifier(model=keras_model, epochs=10, batch_size=32,
verbose=0, neurons1=1, neurons2=1, dropout_rate=0.0))
pipeline_rn


parameters_rn = {'kerasclassifier__neurons1': [4, 8, 16, 32, 64],
                 'kerasclassifier__neurons2': [4, 8, 16, 32, 64],
                 'kerasclassifier__dropout_rate': [0.1, 0.2, 0.3, 0.4, 0.5, 0.6]}
grid_search_rn= GridSearchCV(pipeline_rn, parameters_rn, cv=20, scoring='accuracy',
n_jobs=-1)
grid_search_rn.fit(X_train, y_train)
best_model = grid_search_rn.best_estimator_
pipeline = best_model.fit(X_train,y_train)
```

2. Proceso de despliegue

2.1. Aplicación construida

La aplicación está compuesta por un backend y un frontend totalmente independientes entre sí. Se requiere crear un ambiente virtual de Python, preferiblemente versión 3.9, e instalar las dependencias que se encuentran en el archivo requirements.txt [2]para el correcto funcionamiento de la aplicación.

En el backend se utilizó Django REST Framework, una herramienta escalable, integrada y flexible para realizar APIs de forma sencilla. También se utilizó una base de datos relacional MySQL para persistir la información de las migraciones realizadas en cada una de las clases de Django. El backend está compuesto por tres apps o clases: appResenia, que permite realizar todas las operaciones relacionadas con la predicción de una reseña dado un modelo; appReporte, que permite predecir un conjunto de reseñas dado un archivo de Excel y un modelo; y appUsuario, que permite a los usuarios registrarse en la aplicación. Cada clase tiene un archivo .utils donde se crean las diferentes funciones llamadas al momento de realizar las predicciones. Además, se creó una ruta static para persistir los archivos importantes como imágenes, pipelines y resultados.

En cuanto al frontend, también se utilizó Django, pero es independiente del API RESTful creado. Se hizo uso de librerías como Tailwind CSS, Flowbite, tw-elements y HTML para desplegar los diferentes estilos. Para lograr una interacción más amigable con el usuario, se usó JavaScript con el fin de tener un mejor control de los diferentes componentes que se muestran. Los templates de la aplicación se encuentran en la carpeta templates y los estilos en la carpeta static/CSS de Django. Esto mismo aplica a los múltiples módulos y archivos de configuración que se usaron de Node.js.

[2] https://github.com/Uniandes-byte/ProyectoUno/tree/main/Parte%202

La comunicación se realizó mediante múltiples fetchs desde el frontend con ayuda de JavaScript. Para ello, se proporcionaba un body con la información requerida por el API, y se recibía un JSON proveniente de Django REST Framework con la información esperada según la URL llamada. Una vez llegaba la información del backend, entonces se desplegaba en los diferentes componentes de cada template. Además, para verificar el correcto funcionamiento del API se hizo uso de Postman para realizar los request de cada método HTTP creado en el API.

La figura 4 muestra el diagrama con la posible navegabilidad que puede tener el usuario que utilice la aplicación web, y en la tabla 1 se tienen las vistas con sus funcionalidades y su descripción.

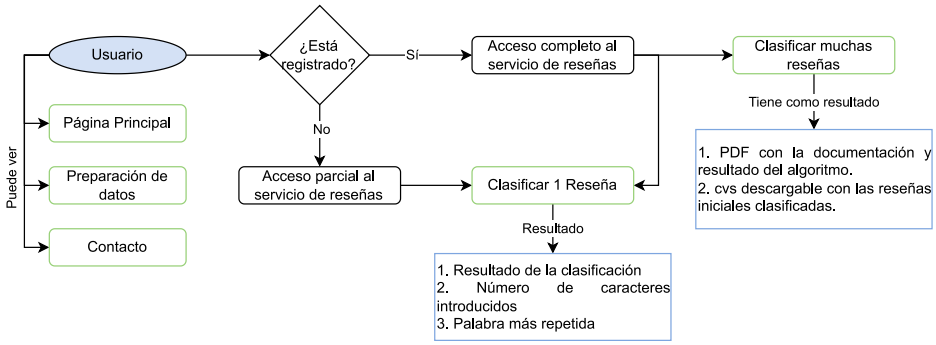


Figura 4. Recursos y navegabilidad que puede tener el usuario

Tabla 1. Vistas de la página web

Vista	Descripción
	Esta vista corresponde al inicio de la aplicación. En ella se pueden ver los modelos que se usan para el análisis de procesamiento de lenguaje natural junto con una breve descripción de cómo funcionan. Además, permite ver un video sobre ¿qué es NLP? Es importante, ya que permite darle al usuario una idea sobre de qué trata la aplicación y su propósito.

Figura 5. Página principal

Figura 6. Preparación de datos

Esta vista corresponde al proceso que se llevó a cabo para la preparación de los datos. En ella se puede ver una descripción detallada de los pasos que se tuvieron en cuenta para la limpieza, calidad y transformación de los mismos. Es importante debido a que le permite al usuario saber qué pasa detrás de la aplicación una vez se pasa una reseña o un dataset de reseñas.

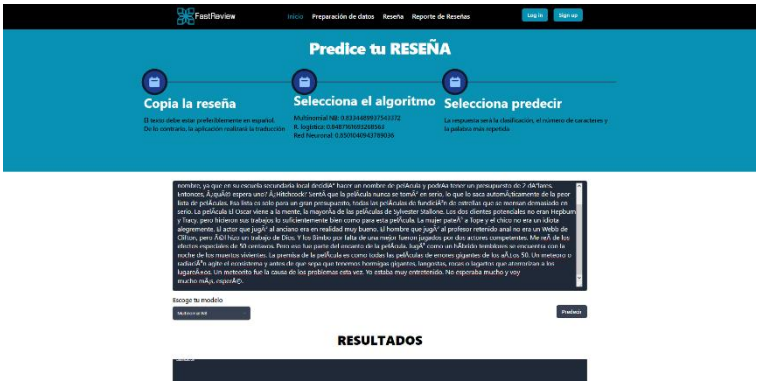


Figura 7. Reseña

Esta vista permite realizar predicciones de una sola reseña. En ella se puede ver primero los pasos a seguir para poder predecir una reseña satisfactoriamente dado un modelo. Su importancia recae en qué la respuesta de la predicción consta de la clasificación (positiva/negativa), número de caracteres y la palabra más repetida.

Esta vista permite realizar predicciones de un conjunto de reseñas. En ella se puede ver primero los pasos a seguir para poder predecir un conjunto de reseñas dado un archivo de Excel. Su importancia recae en qué la respuesta de la predicción consta de un reporte completo en donde se muestran características importantes del conjunto de datos y, además, provee un Excel con las reseñas y sus correspondientes predicciones.

Figura 8. Reporte Reseñas

2.2. Conexión entre la aplicación y el proceso de negocio

Entre los principales usuarios que se pueden beneficiar de la página se encuentran:

- Productoras de cines
- Páginas de reviews de películas
- Plataforma de películas (Ej: Netflix)

La principal utilidad reside en las vistas que procesan las reseñas donde se puede tanto cargar conjunto de datos individuales como varias cargadas en un csv con un formato especifico que tienen como resultado la clasificación en positivas o negativas. Adicional el número de caracteres y la palabra más repetida para cuando es una reseña. Para cuando se tienen más se le dan al usuario dos archivos un pdf con la descripción de los datos, una descripción de los modelos utilizados, las palabras que más afectan teniendo en cuenta el algoritmo, la precisión de este y una gráfica de la cantidad de reseñas positivas y negativas encontradas.

Para una productora de películas, una página web de este tipo puede ser útil para evaluar la recepción de las películas producidas. Los comentarios y reseñas de los usuarios pueden proporcionar información valiosa sobre los aspectos que los espectadores encuentran atractivos o desagradables en una película, lo que puede ayudar a la productora a mejorar la calidad de sus producciones futuras.

En el caso de un cine, una página web de calificación de películas puede ser beneficiosa para evaluar la calidad de las películas que se exhiben en sus pantallas y ajustar la programación de sus proyecciones en consecuencia. Si una película tiene malas críticas en la página web de calificación de películas, el cine puede decidir retirarla de la cartelera o programarla en horarios menos populares.

Para una plataforma de películas en línea tipo Netflix, una página web de calificación de películas puede ser útil para ayudar a los usuarios a elegir qué películas ver. Las calificaciones y reseñas de otros usuarios pueden servir como una guía para los usuarios que buscan nuevas películas para ver, y pueden ayudar a la plataforma a mejorar sus recomendaciones personalizadas para los usuarios.

En general, una página web de calificación de películas puede ser útil para una amplia gama de procesos de negocio en la industria del entretenimiento, desde la producción hasta la exhibición y distribución. Al proporcionar una forma fácil y accesible de evaluar la calidad de las películas, estas páginas web pueden ayudar a los negocios a mejorar su oferta y satisfacer mejor las necesidades de sus clientes.

Se debe tener en cuenta que el reporte que se genera cuando son muchas reseñas da un vistazo a las palabras que más afectan, es decir que si hay una mayoría de reseñas negativas se puede identificar con estos gráficos los objetos o situaciones que hacen que a los espectadores no les guste la película.

2.3. Importancia que tiene la aplicación

Es importante destacar que el análisis de opiniones y comentarios es una tarea esencial en cualquier negocio que busque entender la satisfacción y percepción de los clientes. En el caso de la industria cinematográfica, analizar las opiniones del público general es fundamental para entender la aceptación y popularidad de una película, teniendo su importancia en la toma de decisiones sobre su promoción y distribución.

"Un ejemplo de empresas que ejecutan este tipo de modelos es Cinelytic. Con un funcionamiento distinto, aunque un fondo parecido, la empresa opera, entre otras cosas, con resúmenes de películas y con todos los datos que generan los futuros lanzamientos –características clave, género, estrategia de lanzamiento, presupuesto, entre otras– para predecir el éxito de una producción y multiplicar por 20, según estiman, la velocidad a la que los profesionales pueden trabajar. Tobias Queisser, consejero delegado de la empresa, detalla el fondo de su idea: "Creo que el avance de Netflix y su uso de datos para la toma de decisiones marcó a la industria. Pensamos que la proporción ideal para la toma de decisiones es de alrededor del 60% o el 55% de instinto creativo y experiencia, y un 40% o un 45% de análisis de datos." [1]

3. Grupo de estadística

En la reunión que se tuvo se presentó la página web junto con las funcionalidades implementadas y los resultados que tendrían los usuarios utilizando la opción de subir una reseña o el archivo csv. Las preguntas esta vez estuvieron orientadas a cómo documentar de manera correcta la aplicación, si el enfoque escogido era de utilidad o no para una persona externa y si los resultados que se le presentaban (descarga de resultados, documentación y respuesta) tenían el formato adecuado. La reunión fue de gran utilidad.

Referencias

[1] E. Pais, «CincoDias45,» 25 Agosto 2019. [En línea]. Available: cincodias.elpais.com/cincodias/2019/08/23/fortunas/1566576551_385296.html.