Proyecto BI

Etapa 1

INTELIGENCIA DE NEGOCIOS

Haydemar Núñez

María Isabel Carrascal Cruz

Luimarco Carrascal Diaz

15 de octubre de 2023

1. (10%)Entendimiento del negocio y enfoque analítico.

Oportunidad/problema Negocio	En el proyecto asignado, se abordan los Objetivos de Desarrollo Sostenible (ODS) 6, 7 y 16, que se refieren a agua limpia y saneamiento, energía asequible y no contaminante, y paz, justicia e instituciones sólidas, respectivamente. La relevancia de estos ODS en el contexto de Colombia se relaciona con la mejora de la calidad de vida, la sostenibilidad ambiental y la promoción de la paz y la justicia en el país.
Enfoque analítico (Descripción del requerimiento desde el punto de vista de aprendizaje automático) e incluya las técnicas y algoritmos que propone utilizar.	Para abordar la clasificación de textos relacionados con los ODS 6, 7 y 16, se propone utilizar técnicas de aprendizaje automático basadas en - Random Forest puede manejar datos de alta dimensión, lo que es común en el procesamiento de texto donde cada palabra o n-grama puede considerarse como una característica. - Naive Bayes es conocido por su simplicidad y eficiencia, siendo una opción rápida para la clasificación de texto. Aunque asume independencia entre las características, en la práctica ha demostrado ser efectivo en tareas de clasificación de texto, donde esta independencia no siempre se cumple. - SVM: Es eficaz tanto en clasificación binaria como multiclase, lo que lo hace adecuado para predecir varios SDG.
Organización y rol dentro de ella que se beneficia con la oportunidad definida	La organización que se beneficia con esta oportunidad es el Fondo de Poblaciones de las Naciones Unidas (UNFPA). La implementación de este proyecto contribuirá a los esfuerzos de UNFPA para evaluar y comprender mejor la percepción de la población local sobre cuestiones relacionadas con los ODS 6, 7 y 16. Esto permitirá a UNFPA tomar decisiones más informadas y estratégicas en su trabajo.
Contacto con experto externo al proyecto	Se ha establecido contacto con los estudiantes del curso de estadística para colaborar en la Etapa 2 del proyecto. Los correos electrónicos de los estudiantes son ma.suarez2@uniandes.edu.co. La fecha de reunión con estos expertos externos está programada para el 12 de octubre y la comunicación se llevará a cabo a través de Zoom.

(20%) Entendimiento y preparación de los datos.

Procesamiento de datos

Limpieza y Tokenización:

Se eliminaron etiquetas y caracteres especiales.

Se convirtió el texto a minúsculas y se tokenizó.

Se eliminaron las stopwords y los tokens repetidos.

Normalización y Conversión:

Se eliminaron caracteres no ASCII.

Se convirtieron números a su representación textual.

Se realizó una segunda ronda de eliminación de puntuación.

Stemming y Lemmatization:

Se aplicó stemming (reducción a la raíz) y lemmatization (reducción a la forma base) a los tokens utilizando las bibliotecas spaCy y NLTK.

Entrenamiento Word2Vec:

Se entrenó un modelo Word2Vec con los textos procesados para obtener representaciones vectoriales de las palabras.

Vectorización de Documentos:

Se creó una función para obtener un vector por documento, promediando los vectores de las palabras que lo componen.

Se aplicó la vectorización a cada documento en los dataframes.

Preparación de Datos para el Modelo:

Se estructuraron los datos en matrices de características (X) y etiquetas (y).

Se dividió el dataset en conjuntos de entrenamiento y prueba para evaluar posteriormente el desempeño del modelo.

(25%) Modelado y evaluación.

Algoritmo 1

RandomForest, implementado por Isabel Carrascal

```
525] from sklearn.model_selection import train_test_split

    X = np.array(textos_df['doc_vector'].tolist())
    y = textos_df['sdg']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Hiperparametros:

Se intentó utilizar GridSearchCV, pero el tiempo de ejecución era demasiado alto

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier

# Crear el modelo
rf = RandomForestClassifier(random_state=42)

# Definir la grilla de hiperparámetros
param_grid = {
    'n_estimators': [100, 200, 300, 400, 500],
    'max_depth': [10, 20, 30, 40, 50, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['auto', 'sqrt']
}

# Crear el objeto GridSearchCV
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5, n_jobs=-1, verbose=2)

# Ajustar el modelo a los datos
grid_search.fit(X_train, y_train)

# Obtener los mejores hiperparámetros encontrados
print(grid_search.best_params_)

Fitting 5 folds for each of 540 candidates, totalling 2700 fits
```

Por lo que se utilizó un RnaodmizedSearchCV, lo que redujo considerablemente el tiempo de ejecución

```
param_distributions = {
    'n_estimators': [100, 200, 300, 400, 500],
    'max_depth': [10, 20, 30, 40, 50, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['auto', 'sqrt']
}

# Crear el objeto RandomizedSearchCV
random_search = RandomizedSearchCV(estimator=rf, param_distributions=param_distributions, n_iter=10, cv=5, n_jobs=-1, verbose=2, random

# Ajustar el modelo a los datos
random_search.fit(X_train, y_train)

# Obtener los mejores hiperparámetros encontrados
print(random_search.best_params_)

Fitting 5 folds for each of 10 candidates, totalling 50 fits
/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_forest.py:424: FutureWarning: `max_features='auto'` has been deprecated in 1.
  warn(
{'n_estimators': 500, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_features': 'auto', 'max_depth': None}
```

Dio como resultados los hiperparametros más optimos:

{'n_estimators': 500, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_features': 'auto', 'max_depth': None}

Ejecución del RandomForestClassifier con los parametros por default

```
[526] from sklearn.ensemble import RandomForestClassifier

    clf = RandomForestClassifier(random_state=42)
    clf.fit(X_train, y_train)

▾ RandomForestClassifier
RandomForestClassifier(random_state=42)
```

Algoritmo 2

Naive Bayes, implementado por Luimarco Carrascal

Hiperparametros: en este caso se hizo una prueba implementando el algoritmo con el hiperparametro más óptimo, y dio un resultado muy similar al de dejarlo por default.

Algoritmo 3

SVM implementado por Isabel Carrascal

```
from sklearn.svm import SVC

# Crear el modelo
svm_model = SVC(kernel='linear', random_state=42)

# Entrenar el modelo
svm_model.fit(X_train, y_train)

# Predecir las etiquetas de los datos de prueba
y_pred_svm = svm_model.predict(X_test)

# Evaluar el modelo
print("Reporte de clasificación para SVM:\n", classification_report(y_test, y_pred_svm))
```

```
Reporte de clasificación para SVM:
              precision    recall  f1-score   support

           6       0.96      0.91      0.93       217
           7       0.84      0.83      0.83       197
          16       0.83      0.88      0.85       186

    accuracy                           0.88       600
   macro avg       0.87      0.87      0.87       600
weighted avg       0.88      0.88      0.88       600
```

(15%) Resultados.

Random Forest implementado por Isabel Carrascal

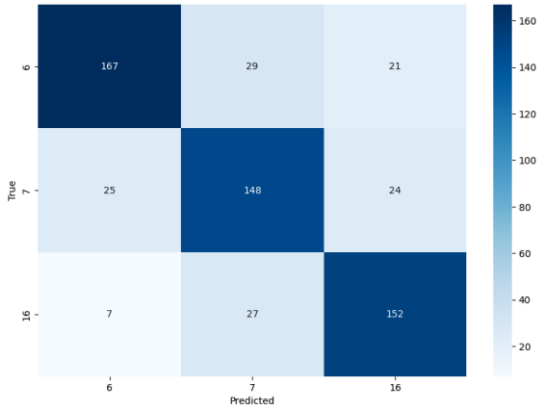
Métricas:

Después de encontrar los hiperparametros más optimos:

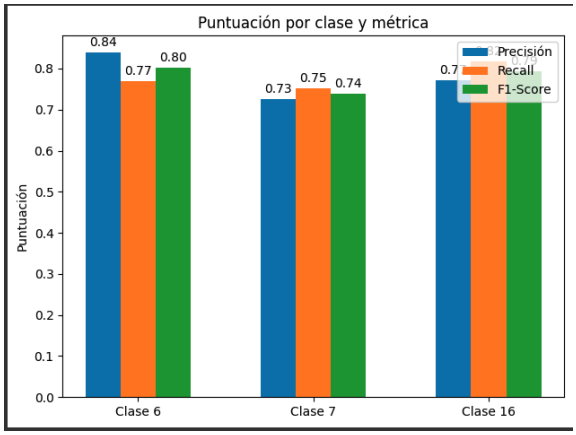
```
/usr/local/lib/python3.10/dist-packages/sklearn/ensemble/_forest.py:42
  warn(
              precision    recall  f1-score   support

           6       0.84      0.77      0.80       217
           7       0.73      0.75      0.74       197
          16       0.77      0.82      0.79       186

    accuracy                           0.78       600
   macro avg       0.78      0.78      0.78       600
weighted avg       0.78      0.78      0.78       600
```

Si se compara con el modelo anterior vemos que si bien en accuracy y recall, no es significativa, sin embargo al ser mayores se utilizó para evaluar los resultados:

Matriz de Confusión:



Métricas:



Random Forest:

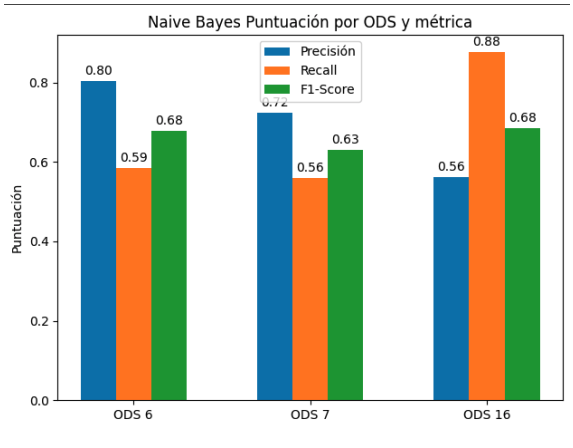
Con una precisión y recall del 78%, Random Forest proporciona una clasificación bastante equilibrada entre los tres ODS. Sin embargo, el ODS 6 tiene la mayor precisión del 84%, lo que indica una buena capacidad para identificar correctamente este ODS en particular.

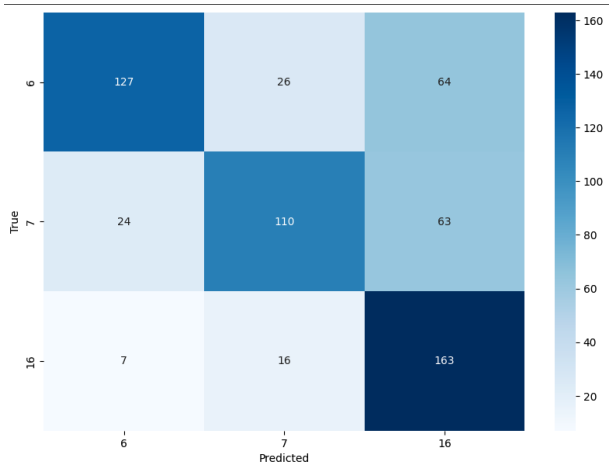
Esto sugiere que Random Forest podría ser una herramienta confiable para la clasificación inicial, pero podría necesitar refinamiento para mejorar la clasificación de los ODS 7 y 16.

Naive Bayes implementado por Luimarco Carrascal

```
[558] from sklearn.naive_bayes import GaussianNB
      from sklearn.metrics import classification_report

      # Crear el modelo
      nb_model = GaussianNB()

      # Entrenar el modelo
      nb_model.fit(X_train, y_train)

      # Predecir las etiquetas de los datos de prueba
      y_pred_nb = nb_model.predict(X_test)

      # Evaluar el modelo
      print("Reporte de clasificación para Naive Bayes:\n", classification_report(y_test, y_pred_nb))
```

```
Reporte de clasificación para Naive Bayes:
              precision    recall  f1-score   support

           6       0.80      0.59      0.68       217
           7       0.72      0.56      0.63       197
          16       0.56      0.88      0.68       186

    accuracy                           0.67       600
   macro avg       0.70      0.67      0.66       600
weighted avg       0.70      0.67      0.66       600
```





Naive Bayes tiene una precisión y recall general más baja del 67%. No obstante, tiene un recall alto del 88% para el ODS 16, lo que indica una buena sensibilidad hacia este objetivo, aunque la precisión es solo del 56%.

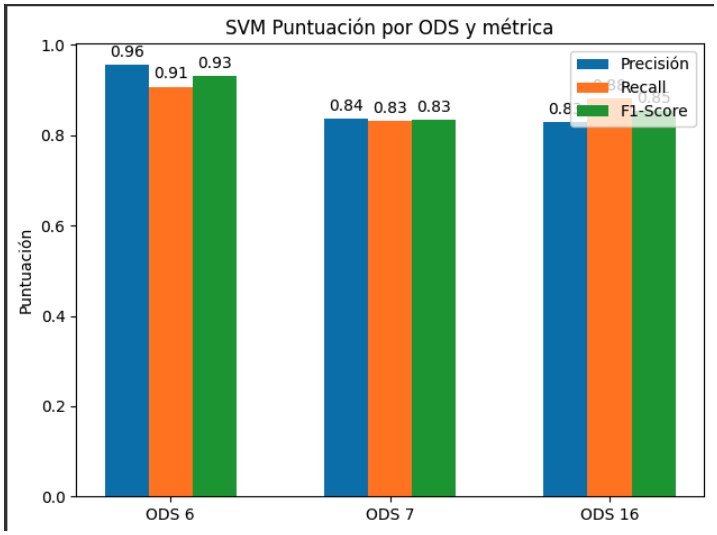
La baja precisión sugiere que Naive Bayes podría generar varios falsos positivos, lo que podría requerir una revisión manual adicional. Sin embargo, su alta sensibilidad hacia el ODS 16 podría ser útil en ciertos escenarios.

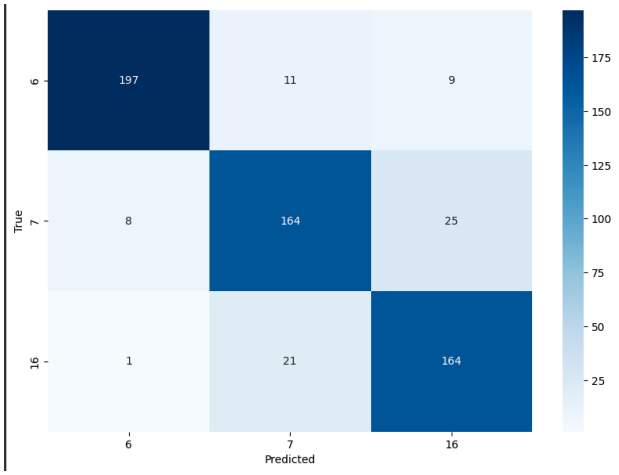
SVM (Support Vector Machine):

```python
from sklearn.svm import SVC

# Crear el modelo
svm_model = SVC(kernel='linear', random_state=42)

# Entrenar el modelo
svm_model.fit(X_train, y_train)

# Predecir las etiquetas de los datos de prueba
y_pred_svm = svm_model.predict(X_test)

# Evaluar el modelo
print("Reporte de clasificación para SVM:\n", classification_report(y_test, y_pred_svm))
```

```
Reporte de clasificación para SVM:
              precision    recall  f1-score   support

           6       0.96      0.91      0.93       217
           7       0.84      0.83      0.83       197
          16       0.83      0.88      0.85       186

    accuracy                           0.88       600
   macro avg       0.87      0.87      0.87       600
weighted avg       0.88      0.88      0.88       600
```

SVM (Support Vector Machine)





SVM muestra el rendimiento más alto con una precisión y recall del 88%. Es especialmente robusto en la clasificación del ODS 6 con una precisión del 96%.

Este rendimiento sugiere que SVM podría ser el modelo más confiable para la clasificación automática de textos según los ODS, permitiendo una identificación precisa y una reducción en la necesidad de revisión manual.

Recomendaciones y Conclusiones:

SVM parece ser el modelo más prometedor para este proyecto, con una alta precisión y recall en la clasificación de textos según los ODS. Podría ser la opción principal para implementar en el sistema de clasificación automática.

Random Forest también muestra un rendimiento decente y podría beneficiarse de una optimización adicional de hiperparámetros o de una revisión de los datos de entrada para mejorar su rendimiento, especialmente en los ODS 7 y 16.

Naive Bayes, aunque tiene un rendimiento inferior, podría ser útil en escenarios donde el recall es más crítico, especialmente para el ODS 16. Sin embargo, la baja precisión indica que podría no ser adecuado para una clasificación precisa sin revisión manual adicional.

La implementación de SVM podría ayudar significativamente a UNFPA en la clasificación automática de textos, facilitando la identificación de temas críticos relacionados con los ODS y permitiendo una evaluación más eficiente de las políticas públicas y las intervenciones relacionadas.

(10%) Mapa de actores relacionado con un producto de datos creado con el modelo analítico construido.

Rol dentro de la Organización	Tipo de Actor	Beneficio	Riesgo
Dirección de Análisis de Políticas Públicas	Usuario-Cliente	Acceso a un análisis automatizado y estructurado de la información textual, facilitando la evaluación de políticas públicas en relación con los ODS.	Interpretación incorrecta de los resultados debido a una clasificación errónea, lo que podría llevar a decisiones políticas desinformadas.
Departamento de Finanzas	Financiador	Mejora de la eficiencia en el análisis de datos, permitiendo una asignación de recursos más informada para proyectos que apoyan los ODS.	Inversión en un modelo que podría no proporcionar los insights esperados, desviando fondos de otros proyectos potencialmente impactantes.
Departamento de TI	Proveedor	Oportunidad para desarrollar y mantener una	Manejo incorrecto de los datos o fallos en la

		plataforma analítica avanzada, promoviendo la innovación tecnológica dentro de la organización.	seguridad de la plataforma que podrían comprometer la privacidad y la integridad de los datos.
Equipos de Trabajo en ODS	Beneficiado	Obtención de insights valiosos de las opiniones y feedback de los habitantes locales, enriqueciendo el entendimiento y la planificación en torno a los ODS.	Dependencia excesiva en el modelo automatizado podría ignorar aspectos cualitativos críticos o insights que requieran una interpretación humana más profunda.
Equipos de Investigación de la Universidad de los Andes	Colaborador	Oportunidad para aplicar y evaluar estrategias de clasificación de textos en un escenario real, contribuyendo al avance de la investigación en NLP y aprendizaje automático.	Resultados insatisfactorios del modelo podrían afectar la reputación y la confianza en futuras colaboraciones entre la universidad y organizaciones externas.

Trabajo en equipo

Líder de Proyecto (Luimarco Carrascal):

- Planificación y Gestión del Proyecto:
- Definir el alcance, los objetivos, los entregables y los plazos del proyecto.
- Desarrollar y mantener un plan de proyecto detallado, incluyendo la asignación de recursos y el cronograma de actividades.
- Comunicación y Colaboración:
- Actuar como punto de contacto principal entre los stakeholders, los equipos de proyecto y los colaboradores externos.

Líder de Negocio (Luimarco Carrascal):

- Recopilar y analizar los requerimientos de negocio relacionados con el modelo de clasificación y su aplicación en el apoyo a los ODS.
- Trabajo en conjunto de la revisión y preparación de datos sea coherente con el negocio
- Implementación de 1 algoritmo del modelo Naive Bayes
- Revisión de resultados del modelo implementado

Líder de Datos (Isabel Carrascal):

- Gestión y Preparación de Datos:
- Supervisar la recolección, limpieza y transformación de los datos textuales que serán utilizados para entrenar y validar el modelo.

Líder de Analítica (Isabel Carrascal):

- Desarrollo y Optimización de 2 algoritmos del modelo
- Análisis de Resultados y Reporting de los algoritmos utilizados
- Analizar y presentar los resultados obtenidos por el modelo, proporcionando insights y recomendaciones basadas en los datos.

12%) Sustentación y evaluación del aporte individual

Para el archivo de predicción de los datos no etiquetados, se uitlizo una técnica de aprendizaje semi-supervisado.

Se hizo la exportación del archivo .csv y se adjunto en la carpeta del proyecto enviada