INTELIGENCIA DE NEGOCIOS

ISIS3301

"Etapa 1. Analítica de textos"

Profesor: *Fabián Peña*

Grupo 5

Wyo Hann Chu Mendez – 202015066

Diana Alejandra Silva Álvarez – 201815366

Sofía Velásquez Marín – 202113334

Tabla de contenido

Entendimiento del negocio y enfoque analítico

Oportunidad / problema del negocio	El objetivo del proyecto es desarrollar un modelo de clasificación, con técnicas de aprendizaje automático, que permita relacionar de manera automática un texto según los ODS. Al igual que desarrollar una aplicación que facilite la interacción con el resultado de dicho modelo. Los criterios de éxito del proyecto son los siguientes: • El modelo debe ser capaz de clasificar los textos con una precisión de al menos el 95%. • La aplicación debe ser fácil de usar y permitir a los usuarios interactuar con los resultados del modelo de manera intuitiva. Según la ONU los objetivos de desarrollo sostenible (ODS) constituyen un llamamiento universal a la acción para poner fin a la pobreza, proteger el planeta y mejorar las vidas y las perspectivas de las personas en todo el mundo. Para el negocio es importante los ODS 3, 4 y 5. El 3ro hace referencia a garantizar una vida sana y promover el bienestar en todas las edades. El 4to hace referencia a Garantizar una educación inclusiva, equitativa y de calidad y promover oportunidades de aprendizaje durante toda la vida para todos. Y el 5to hace referencia a lograr la igualdad de género. El impacto en Colombia por el hecho de ser abordados es ayudar a comprender mejor dichas problemáticas y el modelo a realizar puede ser utilizado para identificar problemas y sus tendencias, y priorizar las que sean más comunes.
Enfoque analítico	Para poder hacer el modelo debemos seguir una serie de pasos para poder encontrar el mejor modelo. Primero, se limpian los datos para que no queden con inconsistencias. Una vez tenemos los datos limpios, procedemos a escoger que algoritmos utilizar. Para poder comparar y escoger el mejor modelo, se van a desarrollar 3 Learning Language Models: BOW (con un RandomForestClassifier), TD-IDF (con RandomForestClassifier) y el tercer modelo lo haremos con Word2Vec (usando AverageWordsVectors).
Organización y rol dentro de ella que se beneficia con la	La ONU tiene una agencia especializada de las Naciones Unidas llamada Fondo de Población de las Naciones Unidas la cual se beneficiaria de la oportunidad debido a que busca identificar problemas y evaluar soluciones actuales, relacionando la información con los diferentes ODS.

oportunidad definida	
Contacto con el experto al proyecto	Se contactó con el experto y se gestionó una reunión para el 19 de octubre.

Entendimiento y preparación de los datos

Los datos que se van a recibir para el manejo de nuestro problema vendrán en formato EXCEL, por ello se hará uso de la librería Openpyxl para procesar estos datos.

Para el entrenamiento y preparación de los datos se hicieron varias evaluaciones generales para poder detectar aquellos factores que tendrá el mayor impacto negativo dentro de nuestro modelo.

Los que más se destacaron y su tratamiento con solución son:

- Errores de conversión para caracteres con tildes o puntuaciones del idioma español (encoding ANSI), frente al formato de entrada de los datos (UTF-8)

ANSI	UTF-8
Á	Ã□
á	Ã¡
É	Ã‰
é	Ã©
Í	Ã□
í	Ã
Ó	Ã“
ó	Ã³
Ú	Ãš
ú	Ãº
Ñ	Ã‘
ñ	Ã±
¿	Â¿

Para este caso, en la imagen de arriba, podemos ver la diferenciación y referencia a ciertos caracteres manejados para el idioma español que debemos usar en nuestro modelo.
Por ello, se hace uso de la librería de Python FTFY (fixes text for you) para solucionar estos caracteres inválidos para el modelo y análisis. Con el objetivo de que a la hora de vectorizar o relacionar palabras se tenga un mayor % en cuenta.

- Idiomas diferentes a los tratados en los datos a analizar

Dentro de nuestro modelo, se hace la aclaración de que se manejaran todas nuestras entradas en español, por ello, se hizo un análisis con respecto a los idiomas manejados dentro de los datos, con el objetivo de removerlos y evitar encuentros de palabras no validas que puedan reducir la efectividad del modelo.

- Remover signos y puntuaciones dentro de los textos

Como nuestros algoritmos harán conteos y vectorizaciones para hacer los cálculos y algoritmos, se procede a retirar los signos de puntuación del texto, para mejorar el tiempo de ejecución y no se tomen estos mismo como puntos de referencia para la clasificación de los textos.

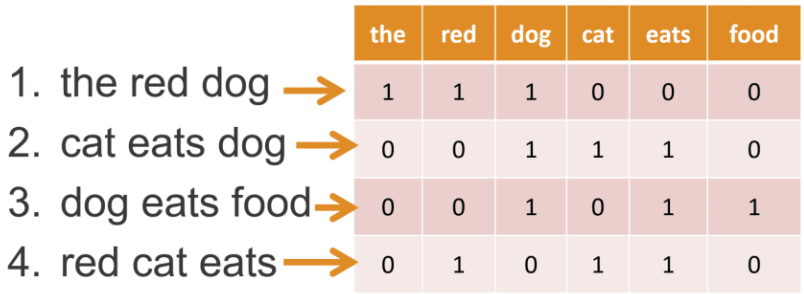
- importación de librerías y funciones

Ya que se va a trabajar datos para análisis con Learning Language Models, se va a hacer uso principalmente de scikit learn, pandas y librerias básicas.

Pero debido a la necesidad de otro modelo y fallas en los datos a nivel de UTF-8, se harán uso también de las librerias fity, unidecode, y gensim. Esta última nos ayudará a trabajar con el ultimo algoritmo a trabajar, el cual pedirá algunos datos adicionales de tokens para trabajar.

Modelado y evaluación

Modelo 1 BOW (Bag of Words) (Sofía Velásquez Marín – 202113334 5 horas)



	the	red	dog	cat	eats	food
1. the red dog →	1	1	1	0	0	0
2. cat eats dog →	0	0	1	1	1	0
3. dog eats food →	0	0	1	0	1	1
4. red cat eats →	0	1	0	1	1	0

El Bag of Words (BoW) es una técnica de procesamiento de texto utilizada en procesamiento de lenguaje natural (NLP) y minería de texto. La técnica consiste en representar un documento de texto como un conjunto (bag) de palabras, ignorando el orden y la estructura gramatical de las palabras en el texto.

En el modelo BoW, se crea un vocabulario de todas las palabras únicas en un conjunto de documentos de texto, y cada documento se representa como un vector de tamaño igual al tamaño del vocabulario. Cada posición en el vector representa una palabra en el vocabulario, y el valor en la posición indica la frecuencia de la palabra en el documento.

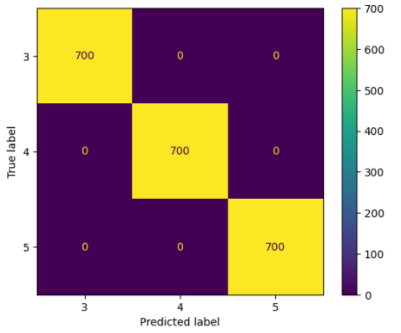
Por ejemplo, si el vocabulario contiene las palabras “gato”, “perro” y “juguete”, y un documento tiene una frecuencia de dos para la palabra “gato”, una frecuencia de tres para la palabra “perro” y una frecuencia de cero para la palabra “juguete”, entonces el vector de representación BoW para este documento sería [2,3,0]

Básicamente como se representó en la imagen de la parte superior, se crea una matriz por cada palabra que se contiene en el texto y se hace un conteo por cada una de ellas.

Para nuestro caso hicimos uso de un random forest (42), para *fittear* nuestros datos ya limpiados y preparados anteriormente. Para los cuales tuvimos los siguientes resultados de entrenamiento y prueba.

```
ConfusionMatrixDisplay.from_predictions(y_train, y_train_bow_predict)
```

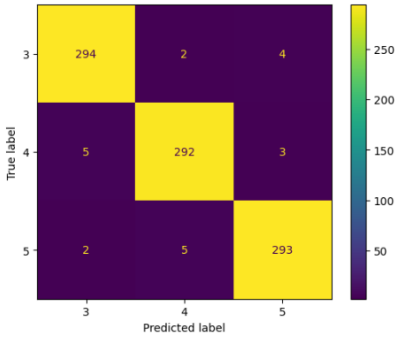
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fdf8d9181c0>
```



Matriz de confusión para el train

```
ConfusionMatrixDisplay.from_predictions(y_test, y_test_bow_predict)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fdf8cd21b70>
```



Matriz de confusión para el test

```
[2788] print("Precision:", precision_score(y_train, y_train_bow_predict, average='micro'))
       print("Recall:", recall_score(y_train, y_train_bow_predict,average='micro'))
       print("F1:", f1_score(y_train, y_train_bow_predict,average='micro'))
       print("Accuracy:", accuracy_score(y_train, y_train_bow_predict))
```

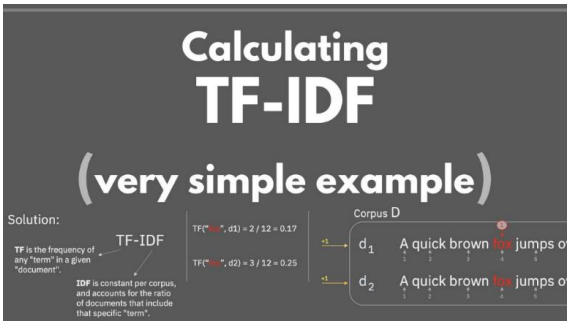
```
Precision: 1.0
Recall: 1.0
F1: 1.0
Accuracy: 1.0
```

```
print("Precision:", precision_score(y_test, y_test_bow_predict, average='micro'))
print("Recall:", recall_score(y_test, y_test_bow_predict, average='micro'))
print("F1:", f1_score(y_test, y_test_bow_predict, average='micro'))
print("Accuracy:", accuracy_score(y_test, y_test_bow_predict))
```

```
Precision: 0.9766666666666667
Recall: 0.9766666666666667
F1: 0.9766666666666667
Accuracy: 0.9766666666666667
```

Lo cual nos arroja unos resultados bastante buenos, aunque no es el mejor algoritmo desarrollado.

Modelo 2 TD-IDF (Wyo Hann Chu – 202015066 5 horas)



TF-IDF (Term Frequency-Inverse Document Frequency) es una medida utilizada en el procesamiento del lenguaje natural (NLP) para evaluar la importancia de una palabra en un documento dentro de una colección de documentos.

La frecuencia de término (TF) se refiere a la frecuencia con la que una palabra aparece en un documento. La frecuencia de documento inversa (IDF) se refiere a la frecuencia con la que una palabra aparece en la colección de documentos. Una palabra con un alto TF-IDF es aquella que aparece con frecuencia en el documento,

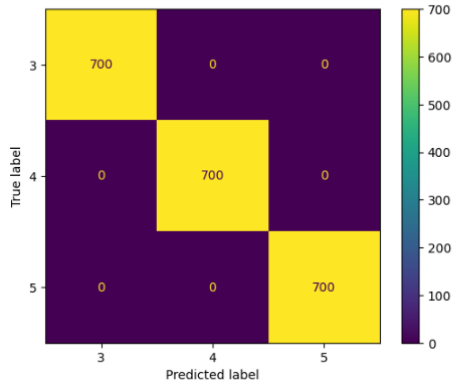
pero no aparece con frecuencia en la colección de documentos. Esto sugiere que la palabra es importante para el documento en particular y puede ser útil para distinguirlo de otros documentos.

En este caso como venimos trabajando, la idea es que este modelo tome parte de los datos para próximamente identificar las clases de los ODS.

Para nuestro caso de desarrollo una implementación de este algoritmo con Random Forest (3), para fittear los datos y hacer las estimaciones de este, con el cual se desarrolla más adelante un pipeline para generar los resultados de la mano de un search RandomizedSearchCV (43).

```
ConfusionMatrixDisplay.from_predictions(y_train, y_train_tfidf_predict)
```

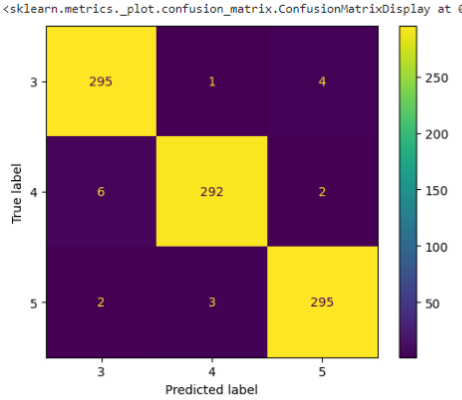
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fdf993d4550>
```



Matriz de confusión para el train

```
ConfusionMatrixDisplay.from_predictions(y_test, y_test_tfidf_predict)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fdf91ecf7f0>
```



Matriz de confusión para el test

```
[7] print("Precision:", precision_score(y_train, y_train_tfidf_predict, average='micro'))
    print("Recall:", recall_score(y_train, y_train_tfidf_predict, average='micro'))
    print("F1:", f1_score(y_train, y_train_tfidf_predict, average='micro'))
    print("Accuracy:", accuracy_score(y_train, y_train_tfidf_predict))
```

```
Precision: 1.0
Recall: 1.0
F1: 1.0
Accuracy: 1.0
```

```
[ ] print("Precision:", precision_score(y_test, y_test_tfidf_predict, average='micro'))
    print("Recall:", recall_score(y_test, y_test_tfidf_predict, average='micro'))
    print("F1:", f1_score(y_test, y_test_tfidf_predict, average='micro'))
    print("Accuracy:", accuracy_score(y_test, y_test_tfidf_predict))
```

```
Precision: 0.98
Recall: 0.98
F1: 0.98
Accuracy: 0.98
```

Para este modelo nos generó el mejor resultado, que más adelante en la pestaña "resultados" se hará más indagación en el mismo con su proceso en el pipeline.

Modelo 3 Word2Vec (Diana Alejandra Silva Álvarez – 201815366 4 horas)

Word2Vec es un algoritmo de aprendizaje automático que se utiliza para crear representaciones vectoriales de palabras. Estas representaciones vectoriales capturan las relaciones semánticas y sintácticas entre las palabras, y se pueden utilizar para una amplia gama de tareas de procesamiento del lenguaje natural (NLP), como la búsqueda semántica, la recomendación de productos y el análisis de sentimientos.

Word2Vec funciona entrenando un modelo de lenguaje neuronal en un gran corpus de texto. El modelo aprende a predecir la probabilidad de que una palabra aparezca en un contexto dado. Por ejemplo, el modelo podría aprender a predecir que la palabra "perro" es más probable que aparezca en el contexto de las palabras "animal" y "mascota" que en el contexto de las palabras " coche " y "ordenador"

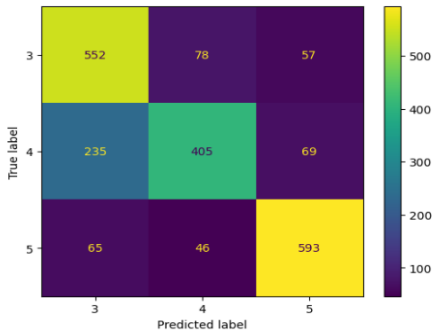
Este será nuestro último modelo por probar con la información sobre los ODS. **Téngase en cuenta que los dos algoritmos anteriores son mucho mejores y tienen mayor trabajo y eficiencia esperada para estos Learning Language Model**

Por ello mismo debido al manejo del *tokenizer* especializado del modelo, se crea la columna para que el algoritmo pueda trabajar con los datos tippeados de la forma en la que lo necesita.

A continuación, la información del análisis realizado y proceso con Word2Vec, trabajado con un average_word_vectors.

```
ConfusionMatrixDisplay.from_predictions(y_train_, predictions1)
```

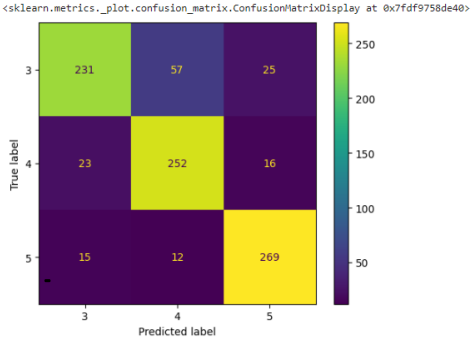
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fdf8bc67e20>
```



Matriz de confusión para el train

```
ConfusionMatrixDisplay.from_predictions(y_test_, predictions)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7fdf9758de40>
```



Matriz de confusión para el test

```
print("Precision:", precision_score(y_train_, predictions1, average='micro'))
print("Recall:", recall_score(y_train_, predictions1, average='micro'))
print("F1:", f1_score(y_train_, predictions1, average='micro'))
print("Accuracy:", accuracy_score(y_train_, predictions1))
```

```
Precision: 0.7380952380952381
Recall: 0.7380952380952381
F1: 0.7380952380952381
Accuracy: 0.7380952380952381
```

```
print("Precision:", precision_score(y_test_, predictions, average='micro'))
print("Recall:", recall_score(y_test_, predictions, average='micro'))
print("F1:", f1_score(y_test_, predictions, average='micro'))
print("Accuracy:", accuracy_score(y_test_, predictions))
```

```
Precision: 0.8355555555555556
Recall: 0.8355555555555556
F1: 0.8355555555555556
Accuracy: 0.8355555555555556
```

Para este algoritmo no se tienen muy buenos resultados con los comparados anteriormente, se tomó como opción ya que queríamos traer también el uso de otras librerías y formatos de Language Learning Models a nuestro problema y tener un espectro más amplio de opciones de resolución que al final del día optimicen los resultados del problema.

Resultados

Se arrojo como resultado que para nuestro problema de análisis de textos relacionado con los ODS 3,4 y 5 de la Agenda 2030 será nuestro algoritmo de ***TD-IDF***. Dándonos una precisión para los datos de entrenamiento y prueba de 0.99907619076191 y 0.9811111111111112 respectivamente.

Para este acá podemos ver el proceso del Aleatorizador aplicado para el entrenamiento, en el cual se realizaron 15 iteraciones para el desarrollo y vectorización.

Como se dijo anteriormente se hace la implementación del pipeline con este algoritmo, para los cuales se usa un randomizador semilla 49, con el cual tiene un mejor rendimiento. Con el diccionario de parámetros se realiza el search para poder obtener los resultados para el entrenamiento y test ya implementado.

```python
class Preprocessing  (BaseEstimator, TransformerMixin):
  def fit(self, X, y=None):
      return self

  def remove_punctuation(self, text):
      # Use ftfy to fix text and remove punctuation points using regular expression
      cleaned_text = ftfy.fix_text(text)
      cleaned_text = re.sub(r'[^\w\s]', '', cleaned_text)
      return cleaned_text

  def transform (self, X):
      X['Textos_espanol'] = X['Textos_espanol'].apply(ftfy.fix_text)
      X['Textos_espanol'] = X['Textos_espanol'].apply(lambda x: x.lower())
      # Apply remove_punctuation function to 'Textos_espanol' column
      X['Textos_espanol'] = X['Textos_espanol'].apply(self.remove_punctuation)
      X["Textos_espanol"] = X["Textos_espanol"].astype(str)

      return X["Textos_espanol"]
```

```python
pipeline = Pipeline(steps = [
    ("preprocessing", Preprocessing()),
    ("vectorizer", CountVectorizer(tokenizer=word_tokenize, stop_words=stop_words, lowercase=True)),
    ("classifier", RandomForestClassifier(random_state=49))
])
```

```python
param_grid = {
    "vectorizer": [CountVectorizer(tokenizer=word_tokenize, stop_words=stop_words), TfidfVectorizer(tokenizer=word_tokenize, stop_words=stop_words)],
    "vectorizer__lowercase": [True, False],
    "classifier__n_estimators": [50, 100],
    "classifier__criterion": ['gini', 'entropy'],
    "classifier__max_depth": [25, 50, 75, 100]
}
```

```python
search = RandomizedSearchCV(pipeline, param_grid, n_iter=15, scoring=["precision", "recall", "f1"], refit="f1", cv=3, return_train_score=True, verbose=1, random_state=43)
```

```python
print("Precision:", precision_score(y_train, y_train_search_predict, average='micro'))
print("Recall:", recall_score(y_train, y_train_search_predict, average='micro'))
print("F1:", f1_score(y_train, y_train_search_predict, average='micro'))
print("Accuracy:", accuracy_score(y_train, y_train_search_predict))
```

```
Precision: 0.9990476190476191
Recall: 0.9990476190476191
F1: 0.9990476190476191
Accuracy: 0.9990476190476191
```

```python
print("Precision:", precision_score(y_test, y_test_search_predict, average='micro'))
print("Recall:", recall_score(y_test, y_test_search_predict, average='micro'))
print("F1:", f1_score(y_test, y_test_search_predict, average='micro'))
print("Accuracy:", accuracy_score(y_test, y_test_search_predict))
```

```
Precision: 0.9811111111111112
Recall: 0.9811111111111112
F1: 0.9811111111111112
Accuracy: 0.9811111111111112
```

Con esto concluimos el desarrollo y entrenamiento de nuestro LLM para el análisis de los ODM 3, 4 y 5 para el análisis de sus respectivos textos. Como final de este proceso, se hará la aplicación de nuestro modelo sobre los datos sin etiquetar para ver la verdadera eficacia de nuestro algoritmo.

Mapa de actores

Rol dentro de la empresa	Tipo de actor	Beneficio	Riesgo
Personas Enfermas	Beneficiado	Gracias a las conclusiones de los análisis de datos, será posible brindarles una mejor atención y prevención frente a inconvenientes de salud que se presenten	Riesgo de que las decisiones médicas basadas en análisis incorrectos puedan afectar negativamente la salud de las personas enfermas.

Mujeres	Beneficiado	Gracias al análisis de los datos, se generarán herramientas y habilidades que puedan usar las mujeres para tener la posibilidad de romper esas barreras de desigualdad, y que se empoderen frente a su género.	Posibilidad de que las intervenciones diseñadas no sean efectivas, lo que podría perpetuar la desigualdad de género.
Estudiantes	Beneficiado	Con el análisis frente a los inconvenientes y barreras que tienen frente a la educación, se generaran procesos y actividades que brinden a los estudiantes mayor acceso a la información y que tengan un mejor proceso pedagógico para su crecimiento como persona.	Riesgo de que las mejoras educativas propuestas no tengan el impacto deseado en los estudiantes, lo que podría no abordar adecuadamente los problemas educativos.
Asamblea Agenda 2030	Usuario-cliente	Acceso a información y análisis que respalden la toma de decisiones alineadas con los Objetivos de Desarrollo Sostenible.	Fallo en la toma de decisiones basadas en datos incorrectos, lo que podría llevar a una implementación ineficaz de la Agenda 2030.
Cumbre ODS	Usuario-cliente	Datos y análisis que respalden la planificación y evaluación de la Cumbre de Objetivos de Desarrollo Sostenible.	Riesgo de que los resultados de la Cumbre no reflejen con precisión la realidad de los Objetivos de Desarrollo Sostenible.
Fondo monetario ONU	Financiador	Mayor confianza en cómo sus donaciones se utilizan para abordar desafíos locales y mejorar la calidad de vida de la población.	En caso de que el modelo no funcione es dinero mal invertido y pudo dejarse de hacer un proyecto con mayor impacto y viabilidad.
ONU TIC	Proveedor	Garantiza el cumplimiento de estándares de calidad de los modelos desarrollados.	Riesgo de incumplimiento del proveedor, problemas técnicos, demoras en

			la entrega de servicios.

Trabajo en equipo

- Líder de proyecto: Wyo Hann Chu, estuvo a cargo de la gestión del proyecto, definición de tareas, reparto de responsabilidades, chequeo de trabajo realizado, bloqueantes y tiempo de gestión de la entrega. Si no hay consenso sobre algunas decisiones, tiene la última palabra.
- Líder de negocio: Sofía Velásquez, fue responsable de velar por resolver el problema o la oportunidad identificada y alinearse con la estrategia del negocio. Contactó al grupo de expertos de estadística para determinar la fecha de reunión para revisar los resultados de esta etapa e iniciar el trabajo de la etapa 2.
- Líder de datos: Diana Silva, encargada de gestionar los datos que se van a usar en el proyecto y de las asignaciones de tareas sobre datos.
- Líder de analítica: Diana Silva, encargada de gestionar las tareas de analítica. Encargada de verificar que los entregables cumplen con los estándares de análisis y tener “mejor modelo” según las restricciones existentes.

Los algoritmos trabajados por cada uno y las horas dedicadas son especificados anteriormente. Sobre el trabajo realizado el porcentaje de tiempo dedicado, de 100 puntos se otorgan 35 puntos a Wyo Hann Chu, 35 a Sofía Velásquez y 30 a Diana Silva. Se definieron reuniones semanales para chequear avances y bloqueantes.