Tecnológico de Monterrey

Campus Querétaro, Querétaro

Reto Datos

Inteligencia artificial avanzada para la ciencia de datos II | TC3007

Profesor:

Ismael Solis Moreno

Alumnos:

Vacas Saturno Saturnitas 🐮🪐

Kevin Alejandro Ramírez Luna		A01711063
Diego Antonio García Padilla		A01710777
José Eduardo Viveros Escamilla		A01710605
Fidel Alexander Bonilla Montalvo		A01798199
Guadalupe Paulina López Cuevas		A01701095
Ángel Mauricio Ramírez Herrera		A01710158
Cristian Chávez Guía		A01710680

19 oct 2025

Índice

Introducción

El uso de la inteligencia artificial se ha vuelto crucial en la estrategia comercial de múltiples industrias, y la industria ganadera no es la excepción. En este reto, por medio de la creación e implementación de un modelo de machine learning realizaremos un producto el cual sea capaz de brindar un beneficio a nuestro socio formador, el CAETEC.

El origen de los datos con lo que estaremos trabajando para este proyecto es generado por medio de los robots de DeLaval, los cuales son robots de ordeño de las vacas que permiten que las vacas puedan irse a ordeñar solas sin necesidad de intervención humana. El sistema tiene una gran cantidad de datos referentes a todo el hato, puesto que cada vaca tiene sus propios registros y su información referente a su ordeño y producción de leche.

En este caso nos enfocaremos en realizar un modelo el cual ayude a la toma de decisiones para saber el mejor momento de mandar a secar a una vaca, esto significa que nuestro modelo será capaz de identificar si una vaca ya debe irse a descansar y dejar de ser ordeñada antes de que dé a luz. Este proceso de identificar a las vacas para mandarlas a secar y monitorearlas, lo realiza el personal del CAETEC el cual es un proceso tedioso y conforme crece al hato más complejo, ya que monitorear más de 100 vacas ya es complicado, y más teniendo en cuenta que todo el trabajo relacionado con las vacas es muy demandante. Por lo que queremos ayudar a automatizar este proceso y ayudar en la toma de decisiones de si mandar o no a secar a una vaca.

Herramientas y tecnologías

Para este proyecto, estamos utilizando la metodología CRISP-DM la cual es el estándar más utilizado en proyectos para ciencia de datos, machine learning y analitica avanzada.

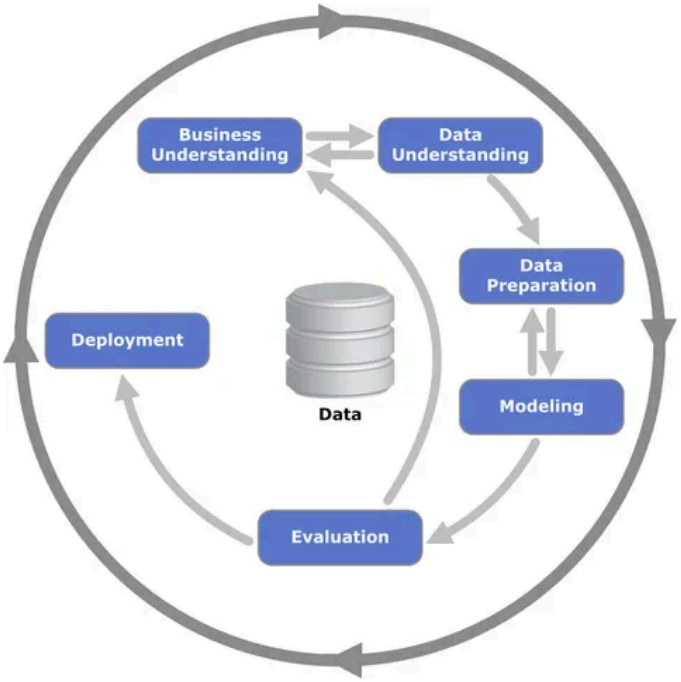

Figura 1. Fases de CRISP-DM

CRISP-DM se divide en 6 fases principales para la producción de conocimiento. En los siguientes apartados estaremos relatando cada una de las diversas tecnologías y herramientas que se usaron por cada fase.
Ver documentación completa

Business Understanding

Para business understanding estuvimos usando la plataforma de Notion para la administración, gestión y documentación del proyecto. Así mismo, de manera complementaria, usamos como servicio de almacenamiento en la nube este Google Drive. Los datos obtenidos para esta fase fueron los que inicialmente nos brindó el socio formador en las primeras semanas de inicio del semestre, sin embargo con el paso del tiempo fuimos obteniendo más datos por medio de visitas al rancho del CAETEC. Estos datos, fueron almacenados en nuestro drive personal Vacas el cual fue nuestro método de almacenamiento para todos los archivos referentes a las vacas que recibimos del CAETEC.

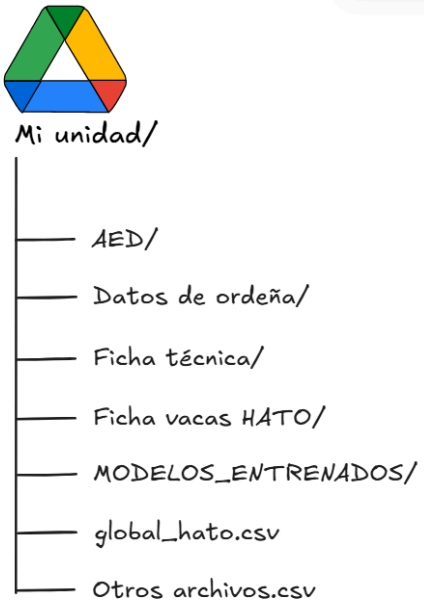

Figura 2. Diagrama de almacenamiento de los datos del CAETEC

Data Understanding

En el data understanding el objetivo de esta etapa es explorar los datos, observar cómo ha sido su comportamiento a lo largo del tiempo, cómo están distribuidos, valores nulos o faltantes, si es que tienen alguna distribución en específico y si existen relaciones fuertes entre los atributos. De igual forma con el entendimiento de los datos podemos saber si es que son útiles para el objetivo del proyecto o si son atributos relevantes para las predicciones del modelo. Para ello realizamos distintas tareas como lo son la colecta de nuestros datos, la descripción de nuestros datos, la exploración de los mismos y verificar su calidad.

A continuación se detallan las librerías que estuvimos usando en la parte de exploración:

- https://colab.research.google.com/ - Colab es el entorno inicial y útil para la ejecución ya que con este entorno podemos hacer uso de todas las librerías necesarias sin tener que instalarlas de manera local
- https://pandas.pydata.org/ - Esta es una librería que usaremos con más recurrencia debido a que es útil para la exploración de dataframes de los datos; así como el merge de dataframes.
- https://numpy.org/es/ - Esta es otra de las librerías que es útil para el uso de funciones matemáticas para la búsqueda de atributos u operaciones en los datos.
- https://matplotlib.org/ - Esta librería nos ayuda a la visualización y graficación de datos para encontrar correlaciones significativas y ver el comportamiento de los datos.
- https://seaborn.pydata.org/ -Esta libreria de igual forma sirve para la visualización de manera un poco más avanzada.

La manipulación de los datos para su exploración está contenida en los siguientes notebooks, los cuales con las herramientas anteriormente mencionadas se realizó toda esta fase tan importante.

Nombre del archivo	Enlace
ID Vaca(1204-8794)	Vacas_Data Exploration.ipynb
Reporte	Reporte_DataUnderstanding.ipynb
Inventario	intentario_data_exploration.ipynb
Patadas	Patadas_Data Exploration
Registro Ordeña	Modelo Multi Step
Registro ordeño Hato	Exploración Registro Ordeño Hato.ipy…
Fichas vacas Hato	Fichas_Vacas_Hato_DataUnderstand…
Global Hato	Global_HATO_DataUnderstanding.ip…

Data Preparation

Para el proceso de los datos antes de pasar al modelado, realizamos varios notebooks en Google Colab. En los cuales realizamos múltiples pipelines con el propósito de crear nuevos dataframes, los cuales utilizamos para entrenar a nuestros modelos. Dentro de estos pipelines realizamos distintas operaciones a nuestros datos como lo son la selección de nuestros datos de interés, la limpieza de nuestros datos, crear atributos derivados si es que pueden servir para nuestro modelo, combinar distintos datos de diferentes fuentes y reformatear nuestros datos si es necesario. Todos estas acciones contribuyen con el llegar a la siguiente fase con datos sólidos y que aporten a los modelos.

Mis archivos derivados fueron guardados en mi carpeta principal donde tengo todos mis demás archivos. Vacas

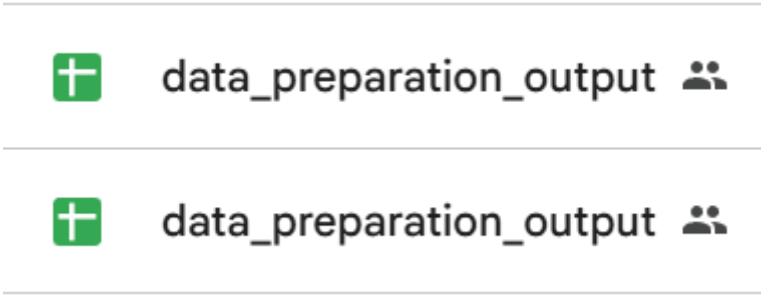

Figura 3. Archivos csv usados en la parte de modelado

Las herramientas utilizadas fueron:

- https://colab.research.google.com/ -Entorno inicial y útil para la ejecución de todas nuestras librerías
- https://pandas.pydata.org/ - Útil para la exploración de dataframes de los datos; así como el merge de dataframes.
- https://numpy.org/es/ - Útil para el uso de funciones matemáticas para la búsqueda de atributos en los datos.
- https://matplotlib.org/ -Visualización y graficación de datos para encontrar correlaciones significativas.

Estas tecnologías permiten una exploración de datos eficiente, reproducible y exhaustiva. NumPy y Pandas proporcionan la base computacional y de manipulación, mientras que Matplotlib y Seaborn ofrecen un espectro completo de visualización, desde lo básico y personalizable hasta lo estadístico y de alto nivel.

Cabe recalcar que como entorno para la creación de los códigos Python se usó Google Colab por su facilidad y rápida conexión con Google Drive (lugar en donde están almacenados los archivos.csv).

Además a diferencia de la utilización de notebooks con el entorno de Anaconda, Visual Studio Code o Pycharm, nos permite tenerlo de manera remota para que en cualquier momento, otro miembro pueda ingresar en cualquier momento y realizar sus propias modificaciones.

Nuestras modificaciones se pueden encontrar en los siguientes notebooks:

Nombre del archivo	Enlace
Registro ordeño para series de tiempo	Modelo Multi Step
Patadas	Patadas_Data Exploration
Inventario	intentario_data_exploration.ipynb
Reporte	Reporte_DataUnderstanding.ipynb
ID Vaca(1204-8794)	Vacas_Data Exploration.ipynb
Fichas vacas Hato	Fichas_Vacas_Hato_DataUnderstand…
Registro ordeño Hato	Exploración Registro Ordeño Hato.ipy…
Global Hato	Global_HATO_DataUnderstanding.ip…

Figura 4. Archivos usado en Data Preparation

Modeling

Durante la fase de modelado, utilizamos Matplotlib para la visualización de datos y Keras, tensorflow para el desarrollo e implementación de nuestros modelos. Esta combinación nos permitirá construir desde modelos básicos hasta arquitecturas avanzadas de Deep Learning, asegurando que cumplan con los requisitos del proyecto. Asimismo, aprovecharemos las capacidades de estas librerías para realizar el ajuste de hiperparámetros, arquitecturas base y las métricas de evaluación necesarias, lo que las convierte en herramientas fundamentales para esta etapa.

Los detalles del modelado en cada fase se documentó en el Notio. Ver en:
DOCUMENTACIÓN CRISP-DM

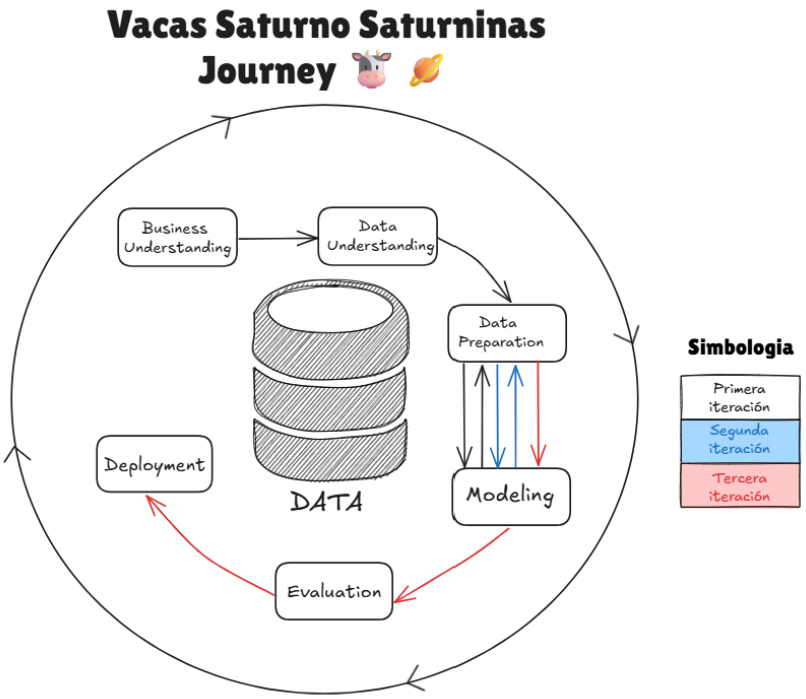

Figura 5. Recorrido de VSS a través de CRISP-DM

Las librerías a usar fueron:

- https://keras.io/ - Libreria de Python usada para la implementación de modelos de Machine Learning y Deep Learning
- https://scikit-learn.org/stable/ - Libreria de Python usada para la implementación de modelos de Machine Learning y Deep Learning a menor escala que Keras
- https://pandas.pydata.org/ - Útil para la exploración de dataframes de los datos; así como el merge de dataframes.

- https://numpy.org/es/ - Útil para el uso de funciones matemáticas para la búsqueda de atributos en los datos.
- https://matplotlib.org/ - Esta librería nos ayuda a la visualización y graficación de datos para encontrar correlaciones significativas y ver el comportamiento de los datos.
- https://seaborn.pydata.org/ - Esta librería de igual forma sirve para la visualización de manera un poco más avanzada.

Después de definir nuestro entorno de trabajo, realizamos un total de 3 iteraciones para buscar el mejor enfoque que cumpliera nuestros objetivos de negocio y minería de datos. Los archivos que se muestran a continuación corresponden a la 3ra iteración realizada, la cual cumplio con lo previamente mencionado.

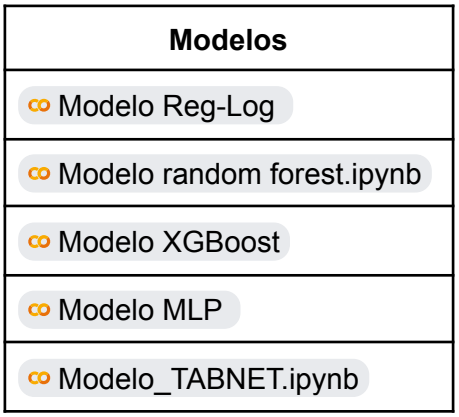

Figura 5. Modelos generados en la 3ra iteración

Un ejemplo de las métricas obtenidas de nuestro modelo es la matriz de confusión, con la cual al comparar con otros modelos y analizando algunas otras métricas importantes como el f1-recall.

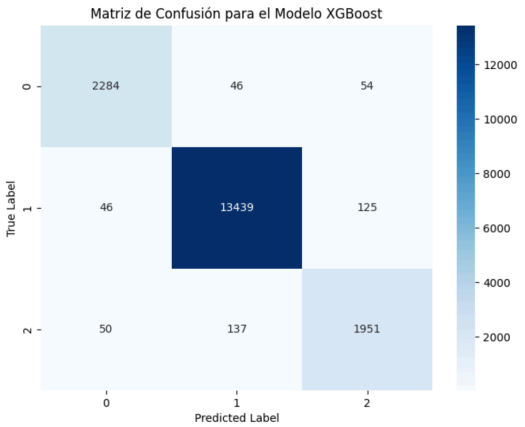

Figura 6. Matriz de confusión

```
Accuracy del Modelo XGBoost: 0.9747

Classification Report para el Modelo XGBoost:
              precision    recall  f1-score   support

           0       0.96      0.96      0.96      2384
           1       0.99      0.99      0.99     13610
           2       0.92      0.91      0.91      2138

    accuracy                           0.97     18132
   macro avg       0.95      0.95      0.95     18132
weighted avg       0.97      0.97      0.97     18132


Accuracy en entrenamiento: 0.9787
Accuracy en prueba: 0.9747
```

Figura 7. Reporte de métricas

Al comparar nuestros resultados de los diversos modelos determinamos que nuestro modelo más óptimo era el XGBoost.

Evaluation

La evaluación del modelo es una etapa fundamental dentro del proceso de análisis y desarrollo de modelos predictivos o de aprendizaje automático. Su propósito es verificar si nuestro modelo cumple con nuestros criterios de negocios, nuestros objetivos y si es que cumple con nuestras expectativas. Para ello hicimos la comparación entre nuestro distintos modelos y al tener nuestro modelo elegido, revisamos el proceso con el cual se llevó a ese resultados para asegurarnos de la calidad, persistencia y confiabilidad requerida. Determinamos los siguientes pasos, o acciones posibles que podríamos realizar y por último tomamos nuestras decisiones finales. En este proceso ya no utilizamos herramientas tecnológicas como en anteriores fases, sino que evaluamos el proceso realizado, nuestros criterios y los resultados obtenidos.

Deployment

En esta última fase de nuestro proyecto, este proyecto tienen como fundamental el uso de la inteligencia artificial, sin embargo, para nuestros usuarios ellos no son personas técnicas, ni tienen un conocimiento tecnológico avanzado, por lo que hay que desplegar nuestro resultado para que pueda ser utilizado y visualizadas las predicciones realizadas por nuestro modelo. Sin embargo también como este es un proyecto con alcance escolar o de aprendizaje, por lo que de momento creamos una demo la cual construimos como se muestra a continuación:

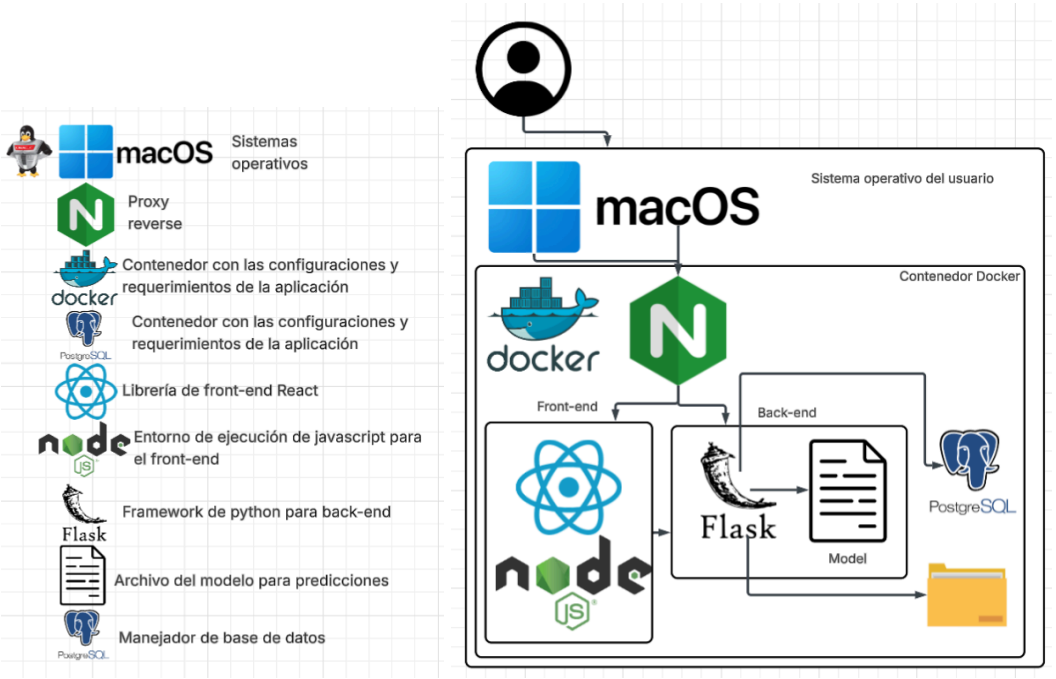

Figura 8. Arquitectura actual

En esta demo nuestro usuario o quienes lo van a probar, lo podrán utilizar de manera local, por medio de un contenedor de docker el cuál, contiene todos los requerimientos que ocupa nuestra aplicación para ser ejecutada. Dentro de nuestro contenedor, la aplicación se divide en 2, front-end y back-end. El Nginx es nuestro balanceador de carga, el cual distribuye las peticiones enviadas a la aplicación para mejorar el rendimiento y la fiabilidad de la aplicación. Por parte del front end, nuestro usuario podrá ver toda la interfaz e interactuar con nuestro sistema, para correr nuestro front end utilizamos lo que es Node Js el cuál es un entorno de ejecución de javascript el cual nos permitirá poder correr todo el front end. Ahora algo fundamental es nuestro backend, este se comunicara directamente con el front end, ya que todas las peticiones, consultas de información y subida de archivos, se manejan por medio del back end el cual es el encargado de realizar toda la logica detras de nuestra aplicación, ya que a su vez también el backend tiene la responsabilidad de comunicarse con nuestra base de datos, la cual utiliza postgreSQL como manejador de base de datos en la cual se encuentra la información de nuestros usuarios y la información de las vacas. En el diagrama el folder que aparece representa el volumen del contenedor, donde alojare los archivos csv, del archivo que subirá nuestro usuario y con base en el registraremos esa información de las vacas en la base de datos y también con ello realizaremos las predicciones.

El ciclo de vida de los datos es el siguiente:

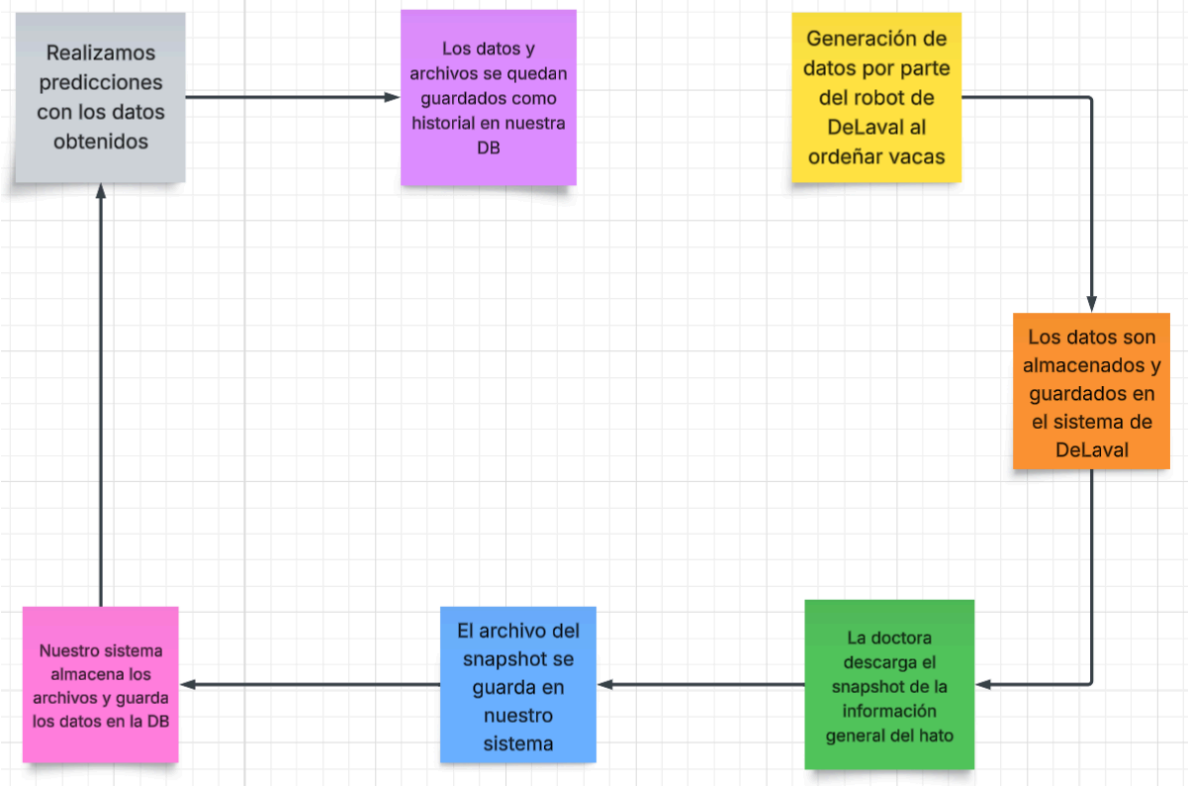

Figura 9. Ciclo de vida de los datos

Todos los datos se quedan almacenados en nuestro sistema(en la DB) y la carpeta designada para los archivos csv, esto para mantener el historial de las predicciones. Como mencionamos antes, este proyecto tiene un alcance escolar, sin embargo, sabemos que tiene potencial para poder escalar y llegar a desplegarse.

Para estas nuevas propuestas en el caso de que si se llegue a utilizar o sea del interés del socio formador implementarlo, primero tenemos que conocer las limitaciones técnicas que tenemos.

- Se tienen que descargar archivos csv: los datos tienen que ser pasados del sistema de DeLaval al nuestro. Esto ya que no tenemos acceso a ninguna API para consultar los datos.
- Se requiere un sistema nuevo(el nuestro): el sistema de DeLaval en cuanto a funcionamiento está muy completo, sin embargo, para cosas más específicas como integración de inteligencia artificial, nosotros no podemos modificar su sistema privado para realizar modificaciones por lo que es necesario hacer un sistema propio para utilizar nuestros modelos.

Ya teniendo el contexto de nuestras limitaciones, hicimos otra propuesta de arquitectura la cuál es más robusta.

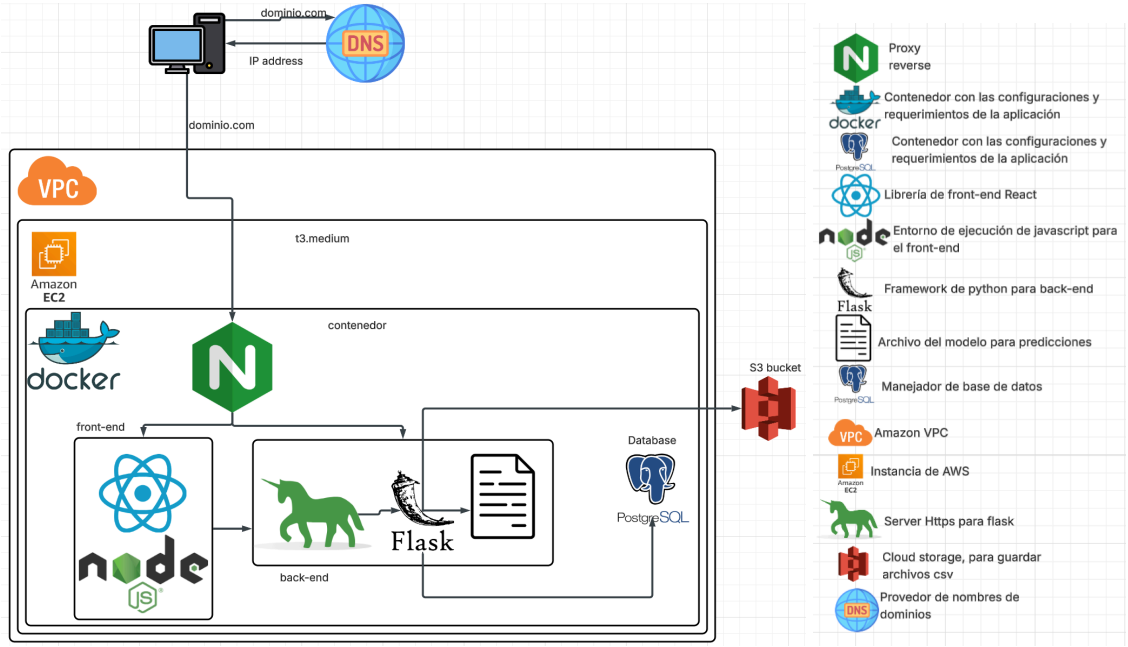

Figura 9. Diagrama de despliegue

En nuestra nueva propuesta de la arquitectura para el despliegue de la aplicación, mucho mayor, donde la aplicación se accede por medio de internet, utilizando una ip y un dominio de la aplicación. El servicio de VPC lo utilizaremos para poder limitar el acceso solo a IPs autorizadas o que sean del caetec. Por otro lado, el nginx se mantiene igual al la arquitectura de nuestro demo, al igual que el front-end. En cuanto al backend utilizamos gunicorn el cual es un server http que se utiliza para producción y funciona con flask. Por otro lado en lugar de tener los archivos csv que subirá el usuario en una carpeta de manera local, ahora serán guardadas en el servicio de S3 de amazon para no llenar el espacio de almacenamiento disponible de la instancia, por ende nos conviene mejor utilizar mejor este servicio y así utilizamos el espacio disponible únicamente para mantener la aplicación corriendo.

Información adicional de algunas tecnologías y servicios que utilizaremos para el almacenamiento de nuestros datos:

- AWS S3 servicio de almacenamiento en la nube que almacena datos como "objetos" en contenedores llamados "buckets".
- PostgreSQL es un sistema de gestión de bases de datos relacionales de código abierto.
- Docker es una plataforma de código abierto que automatiza el despliegue, la ejecución y la gestión de aplicaciones en contenedores.

Modelo entidad relación

Para nuestro modelo entidad relación tanto de la aplicación demo, como para la propuesta de implementación lo diseñamos de la siguiente manera:

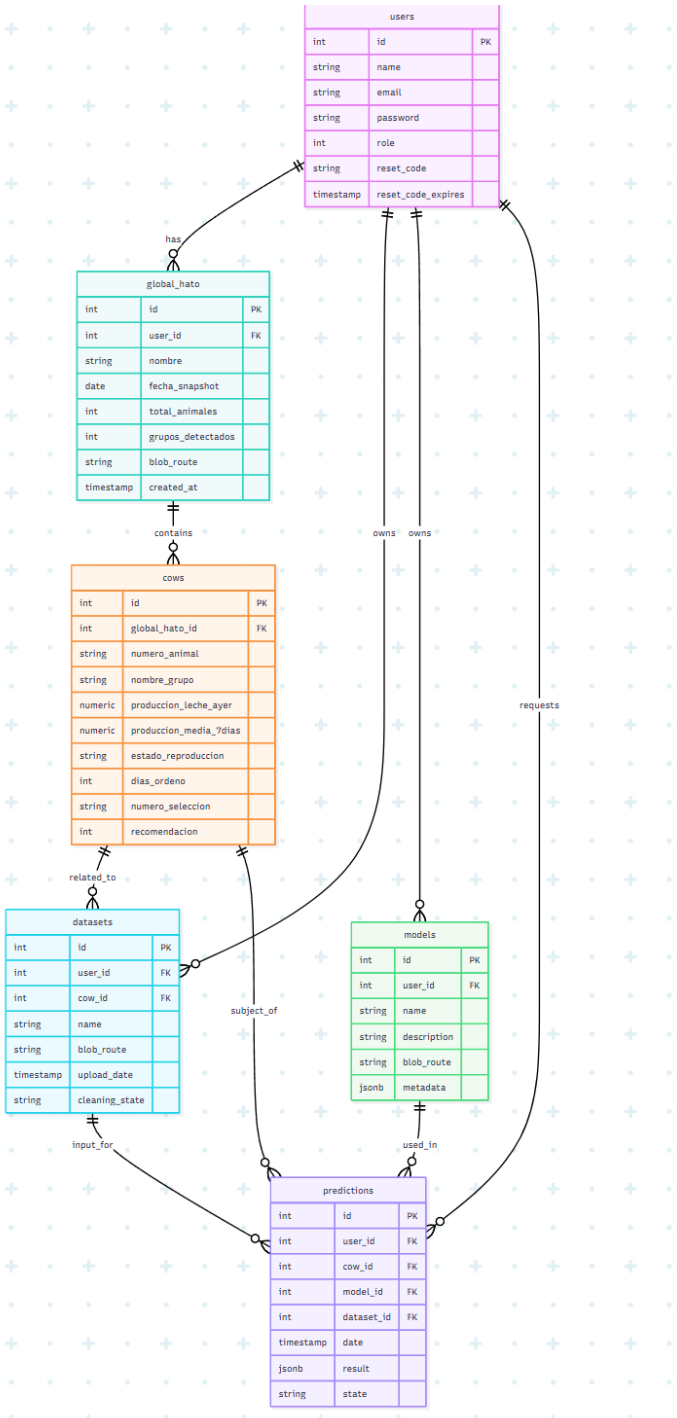

Figura 10. Modelo entidad relación

Como se puede observar en nuestro modelo tenemos múltiples entidades como lo son nuestras vacas, nuestros dataset, predicciones, modelos y usuarios. Cada uno de estos elementos son necesarios para el correcto funcionamiento y almacenamiento de nuestros datos, ya que en el caso de nuestros usuarios es necesario tener restricciones para usuarios no autorizados por nuestro sistema. En cuanto el resto de entidades, las vacas, los datasets y las predicciones están altamente relacionadas puesto que a partir de la subida del archivo csv, se guardan los datos de las vacas en nuestra base de datos, se genera un dataset del mismo, el cual será utilizado para realizar las predicciones por nuestro modelo, es decir necesitamos guardar también esas predicciones realizadas que nos dio de output el modelo.

Primer aproximación del reto

Antes de tener detallado nuestro modelo, y de realizar las múltiples iteración que realizamos en concorde a la metodología de CRISP DM realizamos lo siguiente:

Para el análisis exploratorio de los registros dados por CAETEC, se inició con cuatro archivos principales que contenían la información base:

- ID Vaca.csv (con registros desde la vaca 1204 hasta la 8794)
- patadas.csv
- inventario.csv
- reporte.csv.

Estos archivos sirvieron como punto de partida para establecer un proceso estandarizado de limpieza y transformación de datos.

Ahora, a modo de listado, en la siguiente sección se detalla lo que se puede observar en cada csv.

1. El proceso comenzó con la carga del dataset principal (descargado de manera local) en el entorno de Google Colab, donde se montó la unidad de Drive y se accedió al archivo CSV correspondiente al análisis. Una vez cargado el dataset, se procedió a verificar sus dimensiones originales, identificando el número total de registros y variables disponibles para su registro.

2. La siguiente etapa consistió en una evaluación exhaustiva de la calidad de los datos. Se realizó una búsqueda de registros duplicados, contabilizando aquellos que aparecían repetidos en el dataset. Para, posteriormente, examinar la presencia de valores nulos por cada columna, creando un inventario completo de los datos faltantes en el conjunto original.
    a. Dada la importancia de contar con datos completos para el análisis, se estableció un criterio de limpieza basado en el porcentaje de valores nulos. Se

identificaron aquellas columnas que presentaban más del 50% de datos faltantes y se procedió a eliminarlas del dataset, conservando únicamente las variables con suficiente información para ser útiles en el análisis posterior.

b. Adicionalmente, se detectaron columnas específicas que contenían exclusivamente valores cero, las cuales no aportan información relevante para el modelo. Estas columnas fueron eliminadas explícitamente, resultando en un dataset más significativo y cómodo para nuestras implementaciones futuras.

3. Una vez completada la limpieza básica, se inició la fase de exploración mediante visualizaciones. Se generaron histogramas y gráficos de barras para todas las variables restantes, excluyendo aquellas relacionadas con fechas e identificadores. Esto permitió observar las distribuciones de los datos y detectar patrones o anomalías en las variables numéricas y categóricas en contraste al dataframe.

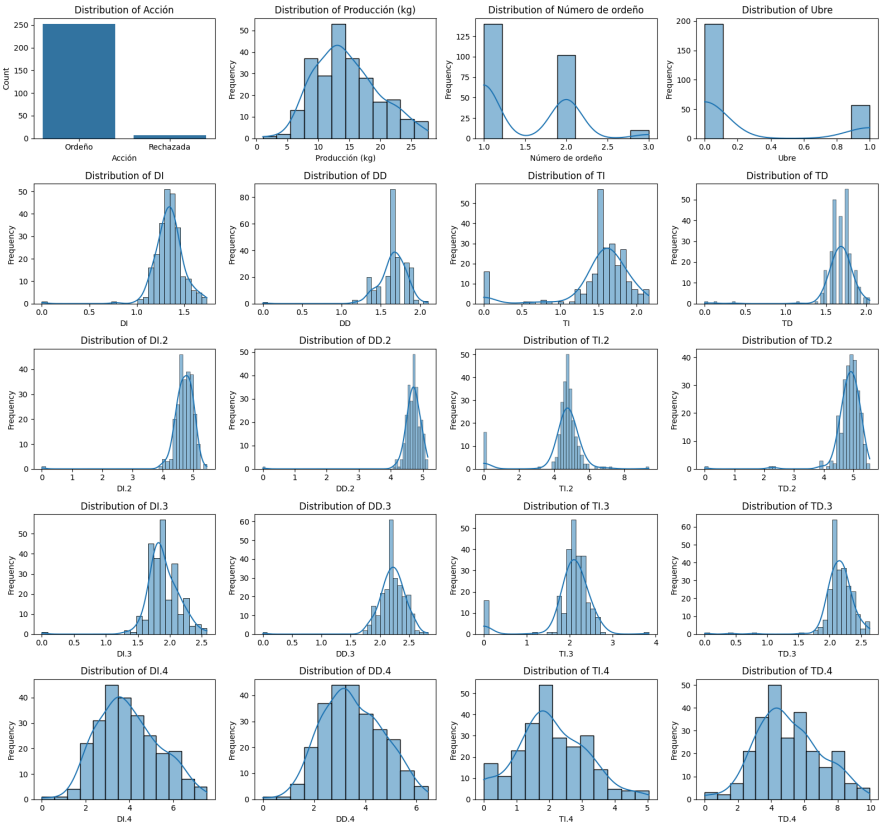

Figura 11. Histogramas generados para el csv de vacas

4. El análisis de valores faltantes continuó con la creación de un mapa de calor que mostraba visualmente la distribución de los datos nulos a lo largo del dataset. Se calcularon los porcentajes de valores faltantes por columna, priorizando aquellas variables que requerirían estrategias de imputación más elaboradas.

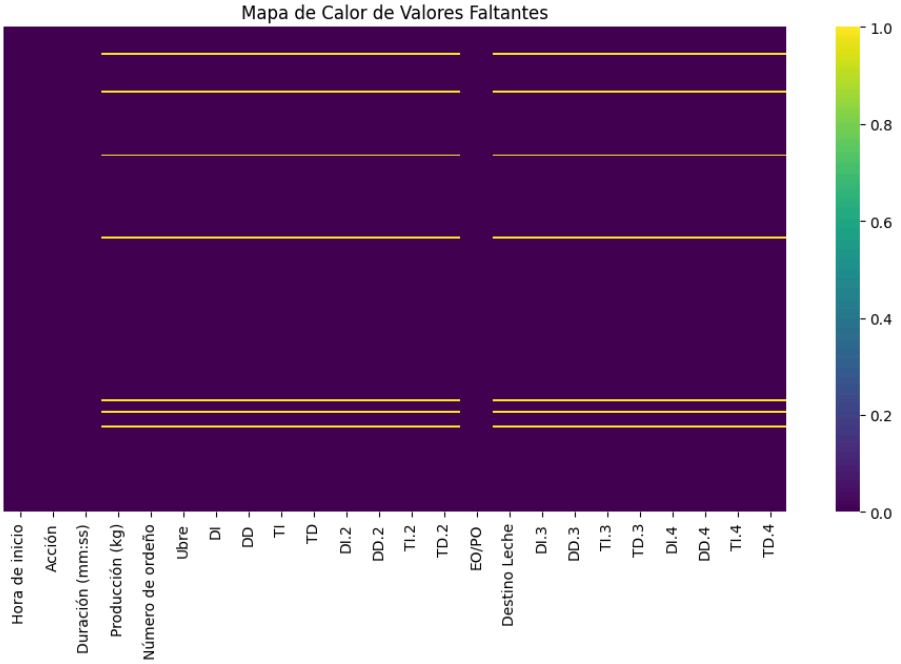

Figura 12. Mapa de calor generado para el csv de vacas

5. Para comprender mejor las características del dataset, se generaron estadísticas descriptivas completas que incluían medidas de tendencia central y dispersión para variables numéricas, así como distribuciones de frecuencia para variables categóricas.

6. Finalmente, se realizó un análisis de correlaciones entre las variables numéricas mediante un mapa de calor, identificando las relaciones más fuertes tanto positivas como negativas. Este análisis proporcionó insights valiosos sobre la estructura subyacente de los datos y las posibles interdependencias entre variables para la consideración en el modelo de Machine Learning.

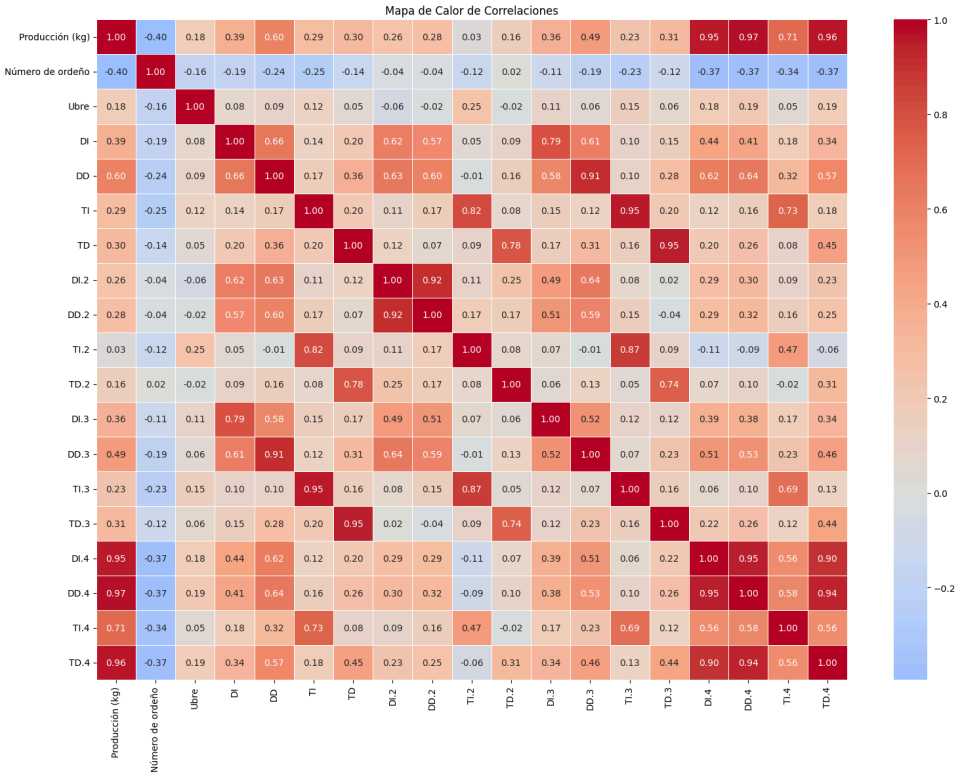

Figura 9. Mapa de correlaciones generado para el csv de vacas

Este proceso sistemático de limpieza y exploración sentará las bases para etapas posteriores de imputación y modelado, asegurando que los análisis se realizan sobre datos de calidad y con una comprensión profunda de sus características fundamentales.

Los archivos de collab se listan a continuación:

- Vacas Data Exploration.ipynb
- reporte_DataUnderstanding.ipynb
- Patadas_Data exploration
- Datos de ordeña - Todas las vacas
- events.ipynb

Para más detalle de los datos, se recomienda checar el enlace al Notion del equipo:
Notion - Data Description

En el siguiente Google Colab es una versión preliminar para la división de datos de Test y Train del .csv de los de registros de ordeña de las 44 vacas, la segmentación de los datos la hicimos a través de KFolds, escogiendo el mejor comportamiento de 4 algoritmos (Random Forest, Ridge, Linear Regression, Lasso), el proceso se documentó en el archivo siguiente:

edve_Datos de ordeña - Todas las vacas

¿Es necesario usar Big Data?

Los datos de las vacas comprenden alrededor de 50,000 registros, y el archivo pesa aproximadamente 7 MB. Dado que estos registros abarcan los últimos 3 años, esto significa que, incluso si el número de registros creciera durante los próximos 10 años, el tamaño estimado del archivo sería de aproximadamente 23 MB, con un total de entre 166,670 y 170,000 registros. Por esta razón, seguimos hablando de Small Data, cuyo rango abarca desde miles hasta cientos de miles de registros.

El concepto de Big Data se refiere a millones, miles de millones o incluso petabytes de datos. Por lo tanto, un archivo CSV con 50,000 registros es fácilmente manejable. Esto implica que, incluso trabajando con herramientas como Google Colab, que proporciona hasta 12 GB de RAM, podemos utilizar pandas u otras librerías tradicionales de Python sin problema. En otras palabras, no necesitamos usar una "bazuca para matar una hormiga".

Implementar tecnologías como Hadoop o Spark, con procesamiento distribuido y clústeres de servidores, agregaría una complejidad innecesaria, además de costos adicionales, tiempo extra de desarrollo y mayor mantenimiento, recursos que actualmente tenemos limitados como equipo.

Las ventajas de trabajar con este volumen de datos incluyen que la limpieza se realiza en segundos y que el entrenamiento de modelos de machine learning es mucho más rápido. En conclusión, podemos manejar los datos eficientemente utilizando las librerías tradicionales del stack de Python.

Según la definición de Coursera:
"Small data, as you might guess, comprises data sets small enough for human comprehension and analysis. It concerns identifying precise causations within an isolated ecosystem and is often used to address immediate needs or answer specific questions."

Esto quiere decir que nuestro proyecto cumple exactamente con las características de Small Data: trabajamos con un dataset comprensible, buscamos identificar causaciones específicas (la relación entre el periodo de secado y la producción de leche), operamos en un ecosistema aislado osea el rancho del CAETEC, y respondemos una pregunta específica (cuándo es el momento óptimo para secar cada vaca). Además, según las "3 V's" que definen Big Data (Volumen, Velocidad y Variedad), nuestro proyecto no cumple con ninguna: tenemos bajo volumen (7 MB vs terabytes), baja velocidad (datos históricos de 3 años vs generación continua en tiempo real), y baja variedad (archivos CSV estructurados vs múltiples fuentes no estructuradas). Por lo tanto, no solo es innecesario utilizar Big Data, sino que sería

contraproducente, ya que las herramientas de Small Data nos permiten obtener insights más rápidamente, con menor costo.

Cambios realizados

Anteriormente se describió la fase de exploración que realizamos en un principio sin embargo al paso del tiempo realizamos no solo un cambios en la exploración de datos, si no que además cambiamos la problemática que íbamos a abordar, lo que llevo a recolectar nuevos datos, realizar más iteraciones de algunas de las fases de CRISP DM. Toda la nuevo documentación, cambios realizados, links y recursos adicionales se encuentran en la siguiente carpeta:

DOCUMENTACIÓN CRISP-DM

Aquí se podrá visualizar toda la evolución del proyecto hasta llegar a los resultados.

Conclusión

El proyecto al estar altamente relacionado con ciencias de datos e inteligencia artificial, fue de suma importancia utilizar herramientas adecuadas para todo el manejo de los datos, así mismo en la parte del deployment el diseño del ciclo de vida de los datos, los diagramas de despliegue y el seguimiento de la información fue una labor que requería del conocimiento adquirido referente a big data y cloud computing. Está claro que se pudo haber llegado a utilizar big data y que ahora más que nunca, es importante utilizar las tecnologías adecuadas para la generación de grandes volúmenes, nuestra propuesta a futuro podría implementar esta aproximación de big data, sin embargo para el alcance actual del proyecto y en el caso que se llegue a implementar no consideramos tan necesario hacer uso de estas herramientas. Sin embargo para escalar este proyecto a no solo un rancho, si no múltiples ranchos, con múltiples hatos, y teniendo una mucho mayor cantidad de información para el entrenamiento de nuestro modelos, las herramientas implementadas cambiaran y ahora sí tendríamos que utilizar big data sin duda.