Universidad de Los Andes

Curso *Proyecto aplicado de analítica*- Anexo Técnico
Grupo 11

Lina Marcela Ladino Solis, Juan Felipe Manjarres Mur,
Mayra Alejandra Neisa Valero
Mayo– 2024

Introducción

En el marco del proyecto destinado a desarrollar una herramienta de análisis del comportamiento del consumo de energía de los clientes no regulados de Electro Dunas, se presenta el siguiente anexo técnico. Este documento complementario al manual de usuario pretende dar una visión detallada de los elementos técnicos fundamentales que respaldan el funcionamiento y la implementación de la herramienta.

El proyecto surge en respuesta a la necesidad de la compañía y sus inversionistas de comprender mejor el comportamiento del consumo de energía en el segmento de clientes no regulados, cuyos niveles de consumo anual superan los 2,5 MW en cada punto de suministro. Con un enfoque en la identificación de posibles anomalías, se propone una solución que combine análisis de datos históricos con técnicas avanzadas de modelado y visualización.

Este anexo técnico se estructura de acuerdo con la rúbrica establecida, que incluye la entrega de varios elementos clave:

1. Diagrama esquemático propuesto: se presenta una representación visual del diseño y la arquitectura de la herramienta, destacando los componentes principales y las interacciones entre ellos.
2. Reporte técnico de experimentos: se detallan los procedimientos experimentales llevados a cabo durante el desarrollo y la evaluación de la herramienta, incluyendo la selección de datos, las metodologías de análisis y los resultados obtenidos.
3. Rúbrica de evaluación diligenciada (tabla de requerimientos): se incluye una evaluación de la herramienta según los criterios establecidos en la rúbrica, destacando sus fortalezas, áreas de mejora y cumplimiento de los objetivos del proyecto.
4. Archivos de código desarrollados: se proporciona acceso a los archivos de código fuente utilizados para la implementación del prototipo, así como para la realización de pruebas y validación de resultados. Los archivos están en un repositorio de GitHub y están documentados para facilitar su comprensión y reproducción.

En este anexo técnico se busca ofrecer una perspectiva detallada del proceso de desarrollo y despliegue de la herramienta de análisis de consumo de energía, contribuyendo a que los usuarios finales en Electro Dunas la comprendan y utilicen eficazmente.

1. **Diagrama Esquemático Propuesto**

A continuación, se puede observar el diagrama esquemático que se definió para el proyecto (Para ver el Diagrama en su tamaño real ir a: Diagrama Esquemático)

2. Reporte Técnico de Experimentos

A lo largo de esta sección se detallará el proceso metodológico utilizado para la construcción del prototipo cubriendo desde la preparación de datos hasta la evaluación de modelos, destacando la integración coherente de modelos descriptivos y la detección de anomalías.

Acorde a los requerimientos del cliente, se deben plantear dos escenarios de análisis, por un lado, un análisis descriptivo para reconocer patrones de comportamiento de cada cliente y, por otro lado, un análisis de anomalías, donde se buscarán puntos que no coincidan con el comportamiento usual de consumo de energía de los clientes.

Para el desarrollo del trabajo se cuenta con información de 30 clientes en dos tablas:

- Tabla con la medición de 4 variables: energía activa, energía reactiva, voltaje_FA y voltaje_FC. Las mediciones se registran con un intervalo de una hora, las 24 horas del día.
- Tabla con la información del sector al que pertenece cada cliente. Se tienen 7 diferentes sectores, que cubren tratamiento y distribución de agua, cultivos y tratamiento de alimentos, tratamiento y venta de metales.

2.1. Preprocesamiento de Datos

En esta fase, se realizan tareas para limpiar, transformar y preparar los datos de manera que sean adecuados para su uso en los modelos de aprendizaje no supervisados seleccionados. La información disponible tiene las siguientes características:

1. Por reserva de la información se tiene solamente un identificador del cliente y no su nombre o información adicional.
2. Se tienen registros de 4 variables, energía activa, energía reactiva, voltaje_FA y voltaje_FC. Estas medidas están en una estructura de serie de tiempo, con medidas cada hora en una ventana desde el 1 de enero de 2021 a las 00:00 horas hasta el 1 de abril de 2023 a las 00:00 horas.

3. Se encuentra que para todos los registros por fecha de cada cliente se tienen las medidas de las 4 variables, sin embargo, no todos los clientes tienen registro en la misma ventana de tiempo.

Completitud

En el periodo comprendido entre 2021 y 2023, se dispone de información para los 30 clientes documentados en el archivo sector_economico_clientes. La cantidad de clientes con datos registrados se mantiene constante en 30 durante el año 2021. Para el año 2022 reportan los 30 clientes, pero entre enero y agosto, dejaron de reportar 15 clientes. Finalmente, en el año 2023, cinco clientes reportaron hasta el 24 de marzo; lo que deja como saldo, que 10 clientes tienen información completa para todo el periodo.

Tabla de Completitud

Ítem	2021	2022	2023
Cantidad de clientes	30	30	15

Consistencia

Se ha confirmado que los 30 clientes que disponen de información detallada sobre su consumo de energía están debidamente asignados a un sector económico. Este proceso de verificación garantiza la coherencia entre la información de consumo y la clasificación correspondiente al sector económico de cada cliente.

Durante la revisión de los datos, se verifico también que la totalidad de los valores deberían ser positivos. Sin embargo, se identificaron valores negativos en la variable Active_energy. Con base en esto, se tomará una decisión estratégica sobre cómo abordar esta inconsistencia.

```
Cantidad total de valores negativos en la columna Voltaje_FA: 0
Cantidad total de valores negativos en la columna Voltaje_FC: 0
Cantidad total de valores negativos en la columna Active_energy: 505
Cantidad total de valores negativos en la columna Reactive_energy: 0
```

Ilustración consistencia

Claridad

Aunque la identificación del cliente se presenta de manera anónima por razones de privacidad, se mantiene una consistencia en las relaciones y la combinación de la información a través del modelo.

Formato

Se ha verificado que el formato de la fecha se asigna correctamente como dato "fecha". Además, las variables relacionadas con energía (Active_energy y Reactive_energy) y voltajes (Voltaje_FA y Voltaje_FC) se han confirmado como datos numéricos.

Limpieza de datos

Al evaluar la calidad de la información, se procede a hacer transformaciones a los datos y a elegir que variables se usarán para cada modelo.

- En la variable energía activa se encuentran valores menores a cero, lo cual puede indicar un mal registro desde el medidor o al guardar el dato en la base de datos. Se hace una imputación de estos valores a cero, pero se mantienen en la base.
- Con base en el campo de fecha se procede generar las columnas de Año, Mes, Trimestre, Dia, Hora y Dia_Semana

Con la limpieza de datos realizada, se considera que la información está lista para usarse en los modelos con ajustes específicos necesarios dependiendo del modelo que se aplique.

2.2. Análisis Descriptivo

Nota: Solo se incluyen graficas de algunos clientes para ejemplificar los análisis realizados.

Se recuerda que un interés de la compañía es conocer el comportamiento de consumo de sus clientes y por parte del equipo se espera poder identificar si hay comportamientos constantes o picos de consumo en diferentes franjas o agrupaciones (por ejemplo, hora o día).

Centrados en el consumo, se revisan en conjunto las medidas de energía activa y reactiva. De unas medidas generales, sin discriminar por cliente, se encuentra que la energía activa suele tomar valores más altos que la energía reactiva.

Tabla de medidas centrales de energía activa y reactiva

	Energía activa	Energía reactiva
Mínimo	0	0
Primer cuartil	0,2428	0,1128
Mediana	0,8108	0,3806
Media	1,4729	0,8731
Tercer cuartil	1,9925	1,2228
Máximo	14,6226	11,1351

En una primera exploración se hace uso de gráficos de boxplot, teniendo en cuenta que en donde se tengan cajas muy amplias o con muchos puntos outliers, puede ser más complejo definir o asumir que existe un patron.

Por hora del día

Se identifican casos como los siguientes:

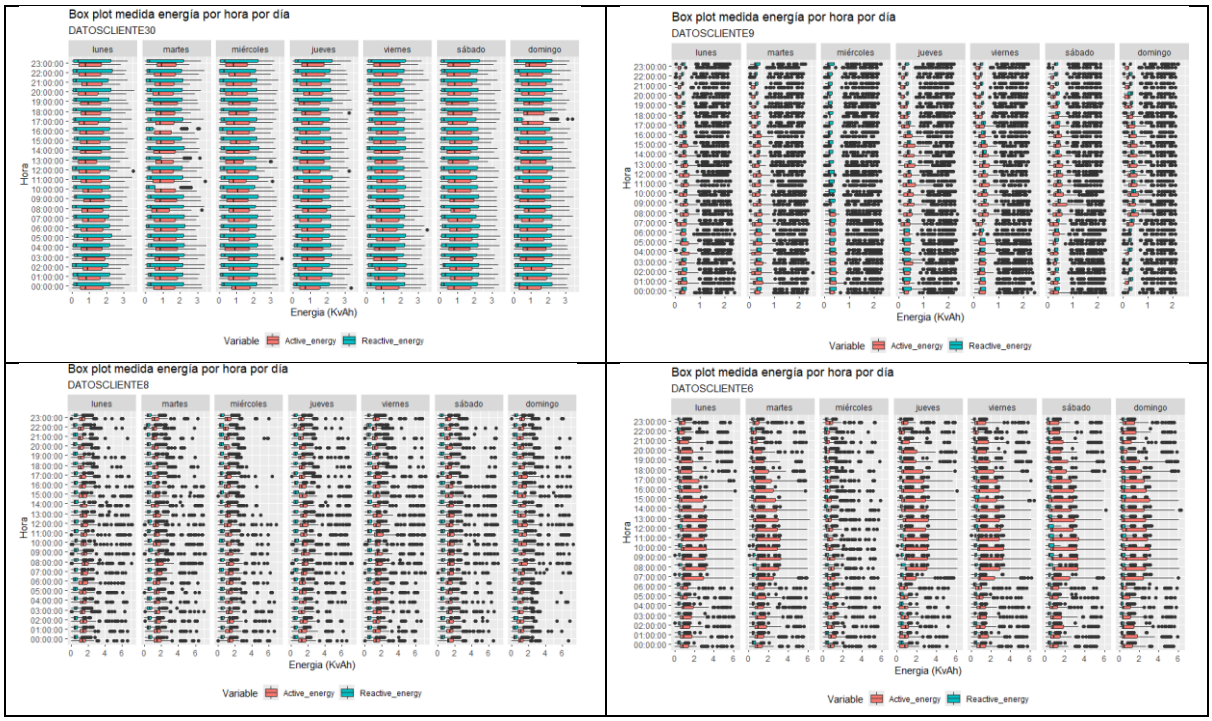
- Comportamiento relativamente constante, cajas del mismo tamaño, donde las medidas de energía activa son más altas que las de la reactiva (por ejemplo, el cliente 8).
- Comportamiento relativamente constante, sin picos a lo largo del día, las medidas de energía activa y reactiva se solapan y la energía reactiva toma valores más altos (por ejemplo, cliente 12).
- Picos en ciertos momentos del día, puede ser en distintas horas (por ejemplo, clientes 16 o 19) o diferenciando día y noche (por ejemplo, cliente 17).

Gráficos de tendencia variables de energía por hora del día

En este análisis por hora, resalta que los clientes 16, 17, 18, 19 y 20, que pertenecen al sector *Captación, tratamiento y distribución de agua*, tienen un comportamiento particular diferenciando por momentos del día.

Por día de la semana

Habiendo observado que el comportamiento por hora puede variar, se revisa en conjunto con el día de la semana. Similar al caso anterior, se pueden identificar casos en que el comportamiento parece estable (o sin una tendencia especifica) a través de los días (por ejemplo, cliente 30) y otros que tienen comportamientos particulares en ciertos días de la semana (por ejemplo, cliente 6).



Gráficos de tendencia variables de energía por hora del día a través de los días de la semana

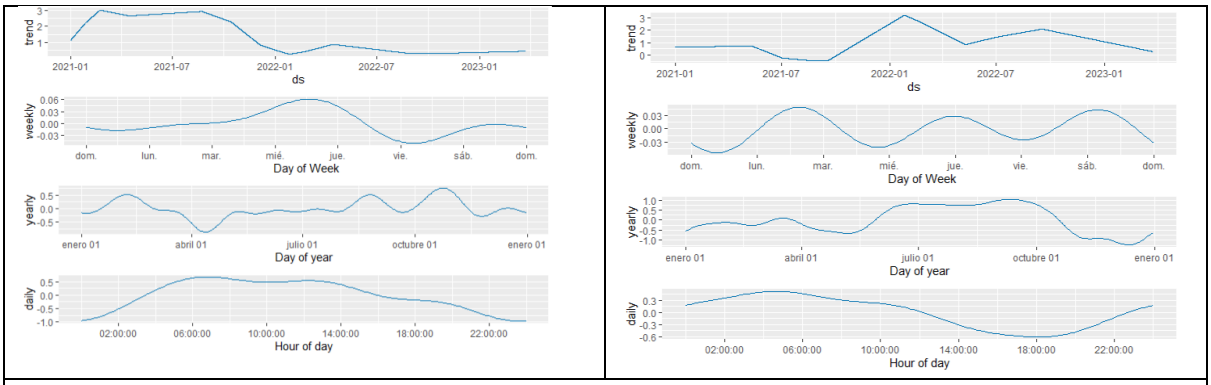
Análisis de series de tiempo

Por otro lado, en la exploración del comportamiento del consumo, se recurre al análisis de series de tiempo, que estudia la relación entre datos registrados en el tiempo, buscando tendencias (por ejemplo, un valor que siempre va en aumento) o si hay estacionalidad (los valores tienen el mismo comportamiento cada semana o todos los años se identifica un incremento en los meses de una determinada estación), todo esto bajo el supuesto de que siempre se tendrá un factor aleatorio o de ruido. Se hace análisis de la variable de energía activa.

Por ejemplo, para el cliente 16, en el análisis del boxplot se había intuido un consumo mayor en las horas diurnas, con la descomposición de la serie de tiempo (imagen a continuación) se podría confirmar esta suposición, viendo que empieza un incremento aproximadamente a las 6 am y empieza a caer aproximadamente después de las 4 de la tarde. También se podría decir que parece haber un mayor consumo entre los martes y jueves.

Como otro ejemplo, para el cliente 20, no parece identificarse alguna tendencia por el día de la semana y en cuanto al consumo diario se podría interpretar que hay un consumo menor hacia las 6 de la tarde.

Cliente 16	Cliente 20



En este tipo de grafica se puede ver la tendencia general de la variable (trend) y, si las hay, tendencias por día en la semana (weekly), por día en el año (yearly) y por hora cada día (daily).

Gráficos de análisis serie temporal para variable energía activa

2.3. Modelos de aprendizaje no supervisado para detección de anomalías

Para abordar la detección de anomalías en los datos energéticos con un enfoque no supervisado, se han seleccionado tres modelos distintos: *Isolation Forest, K-Means y One-Class Support Vector Machine (OCSVM)*. Cada uno de estos modelos ofrece enfoques únicos y ventajas particulares que los hacen adecuados para identificar patrones anómalos en conjuntos de datos de alta dimensionalidad y complejidad (Himeur et al., 2021) A continuación, se presentan y justifican estas elecciones, destacando las características clave de cada modelo y su idoneidad para el contexto específico de este proyecto.

Isolation Forest

Isolation Forest es un algoritmo de Machine Learning de Aprendizaje no Supervisado para la detección de anomalías. Como su nombre lo indica, Isolation Forest es un método de conjunto. En otras palabras, utiliza el promedio de las predicciones de varios árboles de decisión al asignar la puntuación de anomalía final a un punto de datos determinado. A diferencia de otros algoritmos de detección de anomalías, que primero definen lo que es "normal" y luego informan cualquier otra cosa como anómala, Isolation Forest intenta aislar los puntos de datos anómalos desde el principio (Isolation Forest. Isolation Forest is an unsupervised… | by Cory Maklin | Medium, s. f.).

Este algoritmo fue seleccionado debido a su uso en varias oportunidades para la detección de anomalías en el consumo de energía y su bajo costo computacional (Mendes et al., 2023) esta ventaja junto con su uso común en la detección de anomalías de varias industrias y su habilidad de trabajar con datos en series de tiempo lo volvió un modelo ideal para detectar las anomalías que se están presentando en el consumo de energía no regulado.

K-Means

K-Means Clustering es un algoritmo de aprendizaje no supervisado que agrupa el conjunto de datos sin etiquetar en diferentes grupos. Aquí K define la cantidad de clústeres predefinidos que deben crearse en el proceso. Es un algoritmo basado en centroides, donde cada grupo está asociado con un centroide. El objetivo principal de este algoritmo es minimizar la suma de distancias entre el punto de datos y sus correspondientes grupos (K-Means Clustering Algorithm - Javatpoint, s. f.).

Este algoritmo se usa en la detección de anomalías usando la distancia entre los centroides. Se determina un threshold y todos aquellos datos que se encuentren con una distancia superior al mismo pueden ser clasificados como datos anómalos (Harnessing the Power of K-Means for Anomaly Detection | by Tommaso Romani | Medium, s. f.).

Se decide usar este algoritmo por su simplicidad y eficacia en identificar clústeres de datos, lo que ayuda a identificar patrones normales de comportamiento del consumo de energía. Además de lo anterior está el hecho de que se ha aplicado en múltiples ocasiones en la industria para la detección de datos anómalos (Himeur et al., 2021).

One-Class Support Vector Machine (OCSVM)

El *One-Class Support Vector Machine (OCSVM)* es una técnica de machine learning de Aprendizaje no Supervisado utilizada para la detección de anomalías en conjuntos de datos no etiquetados. A diferencia de los algoritmos de clasificación tradicionales que requieren ejemplos de dos o más clases, *OCSVM* se entrena solo con ejemplos de la clase de interés, que en este caso serían los datos energéticos considerados normales. El algoritmo construye un hiperplano en un espacio dimensional elevado para encapsular la región que contiene la mayoría de los datos normales. Luego, clasifica como anomalías aquellos puntos que caen fuera de esta región encapsulada (Himeur et al., 2021).

La inclusión de *OCSVM* se justifica por su capacidad para encontrar la región de datos más densa y distinguir entre puntos normales y anómalos en un espacio multidimensional. Esto lo hace particularmente adecuado para detectar anomalías en conjuntos de datos energéticos, donde es crucial identificar desviaciones significativas del comportamiento esperado.

2.4. Implementación de modelos

Isolation Forest

En una etapa inicial se corrieron cuatro modelos evaluando diferentes agrupaciones de los datos dependiendo su temporalidad de medición:

- Datos agrupados por Año y mes por cliente
- Datos agrupados por Año, mes, día por cliente
- Datos agrupados por Año, mes, día, hora por cliente
- Datos agrupados por Año, mes, día, hora y día de la semana por cliente. (Ver anexos)

Según los resultados de los modelos y los hallazgos de los análisis descriptivos, no es relevante para desarrollar la calibración de parámetros con uno basado en datos agrupados. Por lo anterior se realizaron 5 modelos en búsqueda de mejores parámetros. En el modelo se utilizaron las 4 variables de energía además de la división de la variable de fecha en año, mes, día y hora**.**

- Isolation Forest con los parámetros por Default:

```python
# Configurando el Isolation Forest Agrupación Año Mes dia y hora
clf5 = IsolationForest(n_estimators=100, max_samples='auto', contamination='auto', \
                       max_features=1.0, bootstrap=False, n_jobs= None, random_state=None, verbose=0)

# Entrenar el Isolation Forest
clf5.fit(datos_3)

# Predecir si una observación es una anomalía. -1 indica anomalía, 1 indica normal.
preds5 = clf5.predict(datos_3)

# Añadir las predicciones al DataFrame
datos_3['anomaly'] = preds5

anomalies_5 = datos_3[datos_3['anomaly'] == -1]
print(anomalies_5)
```

✓ 5.2s Python

```
    fuente  Year  Month  Day  Hour  Active_energy  Reactive_energy  \
```

El modelo con los parámetros por defecto encuentra que 127.134 observaciones son anomalías de 463.425 registros de la base inicial, lo que equivale al 27,4% de anomalías.

1. n_estimators: igual a 100
2. max_samples: entre 256 y n_samples
3. Contamination: 27.4%
4. Max_features: 1
5. Bootstrap: false
6. N_jobs: None
7. Random_state: None
8. Verbose= 0

- Isolation Forest con contaminación de 4%

```python
# Configurando el Isolation Forest Agrupación Año Mes dia y hora
clf6 = IsolationForest(n_estimators=100, max_samples='auto', contamination=float(.04), \
                       max_features=1.0, bootstrap=False, n_jobs= None, random_state=None, verbose=0)

# Entrenar el Isolation Forest
clf6.fit(datos_3)

# Predecir si una observación es una anomalía. -1 indica anomalía, 1 indica normal.
preds6 = clf6.predict(datos_3)

# Añadir las predicciones al DataFrame
datos_3['anomaly'] = preds6

anomalies_6 = datos_3[datos_3['anomaly'] == -1]
print(anomalies_6)
```

✓ 6.3s

El modelo con una contaminación del 4% y el resto de parámetros con su valor por defecto encuentra que 18.537 observaciones son anomalías de 463.425 registros de la base inicial, lo que equivale al 4% de anomalías.

1. n_estimators: igual a 100
2. max_samples: entre 256 y n_samples
3. Contamination:4%

4. Max_features: 1
5. Bootstrap: false
6. N_jobs: None
7. Random_state: None
8. Verbose= 0

- Isolation Forest con contaminación de 50%

```
# Configurando el Isolation Forest Agrupación Año Mes dia y hora
clf7 = IsolationForest(n_estimators=100, max_samples='auto', contamination=float(.5), \
                       max_features=1.0, bootstrap=False, n_jobs= None, random_state=None, verbose=0)

# Entrenar el Isolation Forest
clf7.fit(datos_3)

# Predecir si una observación es una anomalía. -1 indica anomalía, 1 indica normal.
preds7 = clf7.predict(datos_3)

# Añadir las predicciones al DataFrame
datos_3['anomaly'] = preds7

anomalies_7 = datos_3[datos_3['anomaly'] == -1]
print(anomalies_7)
```

✓ 7.5s

El modelo con una contaminación del 50% y el resto de los parámetros con su valor por defecto encuentra que 231.707 observaciones son anomalías de 463.425 registros de la base inicial, lo que equivale al 50% de anomalías.

9. n_estimators: igual a 100
10. max_samples: entre 256 y n_samples
11. Contamination:50%
12. Max_features: 1
13. Bootstrap: false
14. N_jobs: None
15. Random_state: None
16. Verbose= 0

- Isolation Forest con contaminación de 12%, random_state: 42 y n_jobs: -1

```
# Configurando el Isolation Forest Agrupación Año Mes dia y hora
clf3 = IsolationForest(n_estimators=100, max_samples='auto', contamination=float(.12), \
                       max_features=1.0, bootstrap=False, n_jobs=-1, random_state=42, verbose=0)

# Entrenar el Isolation Forest
clf3.fit(datos_3)

# Predecir si una observación es una anomalía. -1 indica anomalía, 1 indica normal.
preds3 = clf3.predict(datos_3)

# Añadir las predicciones al DataFrame
datos_3['anomaly'] = preds3

anomalies_3 = datos_3[datos_3['anomaly'] == -1]
print(anomalies_3)
```

✓ 6.4s

El modelo con una contaminación del 12% y el resto de los parámetros con su valor por defecto encuentra que 55.610 observaciones son anomalías de 463.425 registros de la base inicial.

17. n_estimators: igual a 100
18. max_samples: entre 256 y n_samples
19. Contamination: 12%
20. Max_features: 1
21. Bootstrap: false
22. N_jobs: -1
23. Random_state: 42
24. Verbose= 0

- Isolation Forest con contaminación de 4%, random_state: 42 y n_jobs: -1

```
# Configurando el Isolation Forest Agrupación Año Mes dia y hora
clf8 = IsolationForest(n_estimators=100, max_samples='auto', contamination=float(.04), \
                       max_features=1.0, bootstrap=False, n_jobs=-1, random_state=42, verbose=0)

# Entrenar el Isolation Forest
clf8.fit(datos_3)

# Predecir si una observación es una anomalía. -1 indica anomalía, 1 indica normal.
preds8 = clf8.predict(datos_3)

# Añadir las predicciones al DataFrame
datos_3['anomaly'] = preds8

anomalies_8 = datos_3[datos_3['anomaly'] == -1]
print(anomalies_8)
✓ 6.8s
```

El modelo con una contaminación del 4% y el resto de los parámetros con su valor por defecto encuentra que 18.537 observaciones son anomalías de 463.425 registros de la base inicial, lo que equivale al 4% de anomalías.

25. n_estimators: igual a 100
26. max_samples: entre 256 y n_samples
27. Contamination: 4%
28. Max_features: 1
29. Bootstrap: false
30. N_jobs: -1
31. Random_state: 42
32. Verbose= 0

K-Means

En el caso de K-means, solo se hará uso de la información de Active_energy ya que los otros datos no son necesarios para el modelo, además la información se modificará en su estructura para que pueda ser aplicada.

```
## Transformarción del data frame para darle uso en K-Means
df_cluster= df_final
X = df_cluster['Active_energy'].values.reshape(-1, 1)  # Convertirlo a una matriz de una sola columna
print(X)
```

[7] ✓ 0.0s

```
[[0.35784098]
 [0.37226424]
 [1.04468683]
 ...
 [0.231     ]
 [0.15029583]
 [0.11671427]]
```

Ilustración

modificación datos para uso en el modelo de K-Means

Hecho esto se procede a identificar el número de clústeres óptimos para agrupar los datos.

```
## Encontrar el K - Optimo

## Se toma una muestra del total

# Reducción del número de muestras
df_sample = df_cluster.sample(frac=0.2, random_state=123)

# Se estandarizan los datos
X = df_sample['Active_energy'].values.reshape(-1, 1)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

from sklearn.metrics import silhouette_score


varianza_intra_cluster = []
silhouettes = []
Y = {}  # Diccionario para almacenar asignaciones de clústeres
```

```
for k in range(1, 7):
    kmeans = KMeans(n_clusters=k, random_state=123, n_init=10)
    kmeans.fit(X_scaled)
    varianza_intra_cluster.append(kmeans.inertia_)

    try:
        silhouette = silhouette_score(X_scaled, kmeans.labels_)
    except:
        silhouette = 0

    silhouettes.append(silhouette)
    Y[k] = kmeans.labels_

# Imprimir los resultados para k=6
k = 6
print(f"Clusters: {k}, Inercia: {varianza_intra_cluster[k-1]}, Silhouette: {silhouettes[k-1]}")
```

✓ 10m 8.3s

```
Clusters: 6, Inercia: 3590.6520802664563, Silhouette: 0.5947307699845402
```

Ilustración

cálculo de K Optimos

Como se pude observar en la imagen se procede a tomar una muestra del total de los datos ya procesados y se realiza un cálculo de su inercia y sillhouete score (que ayudan a determinar cuál es el número de clústeres que mejor categoriza la información). Con esto se verifica cual es el valor de K óptimo para la información.

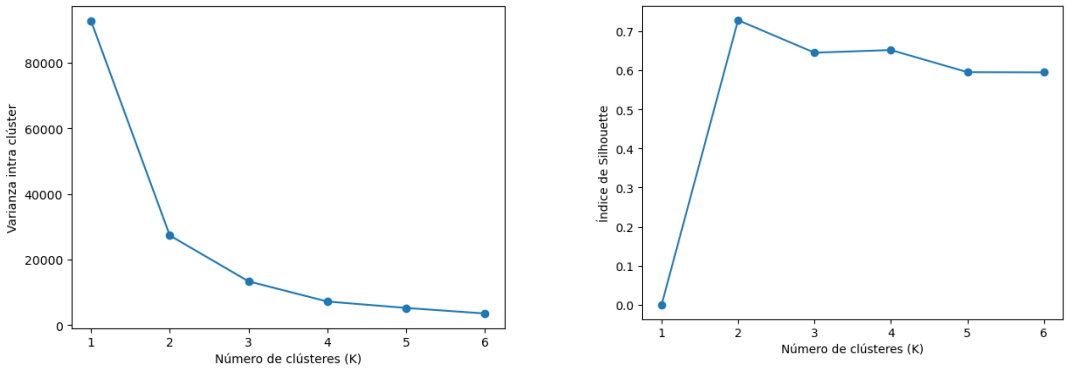


Ilustración Graficas de varianza intra-clúster e índice de Sillhouette

Como se puede observar en la gráfica el valor óptimo de clústeres para el modelo es 2. Definido esto se procede a correr el modelo. Hecho esto se genera un threshold, en este caso de 95, para identificar las anomalías que se han presentado en el consumo de energía de clientes no regulados.

```
df_cluster= df_cluster
X = df_cluster['Active_energy'].values.reshape(-1, 1)  # Convertirlo a una matriz de una sola columna

## Estandarizar los datos
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Aplicar K-means
k = 2  # Número de clusters optimo
kmeans = KMeans(n_clusters=k, random_state=42)
kmeans.fit(X_scaled)

## Obtener las etiquetas de los clusters y los centros de los clusters
cluster_labels = kmeans.labels_
cluster_centers = kmeans.cluster_centers_

## Calcular la distancia de cada punto al centro del cluster asignado
distances = [np.linalg.norm(x - cluster_centers[cluster]) for x, cluster in zip(X_scaled, cluster_labels)]

## Identificar valores anómalos
## Definir un umbral para determinar qué puntos se consideran anómalos
threshold = np.percentile(distances, 95)  # Por ejemplo, tomar el percentil 95 de las distancias
anomalies_indices = np.where(distances > threshold)[0]

## Obtener los valores anómalos y sus índices
anomalies_values = X[anomalies_indices]
anomalies_indices = df_cluster.index[anomalies_indices]
```

Ilustración código del modelo

Con base en este se identifican 23.171 anomalías.

```
## Mostrar el resultado
print("Total de anomalías en la columna 'anomalias':")
print(total_anomalias)
```

✓ 0.0s

```
Total de anomalías en la columna 'anomalias':
0    440244
1     23171
Name: anomalias, dtype: int64
```

Ilustración número de anomalías identificadas

One-Class Support Vector Machine (OCSVM)

Para el modelo de OCSVM se realiza el entrenamiento con la información disponible dando uso únicamente a la información de la variable de Active_energy

```
import numpy as np
import pandas as pd
from sklearn.svm import OneClassSVM
from sklearn.preprocessing import StandardScaler

df_cluster= df_cluster
X = df_cluster['Active_energy'].values.reshape(-1, 1)  # Convertirlo a una matriz de una sola columna

## Estandarizar los datos
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

## Entrenar el modelo OCSVM
ocsvm = OneClassSVM(nu=0.01)  # Valor para detectar anomalias
ocsvm.fit(X_scaled)
```

15] ✓ 26m 36.9s

```
OneClassSVM
OneClassSVM(nu=0.01)
```

Ilustración entrenamiento del modelo OCSVM

Una vez realizado el entrenamiento se procede a identificar aquellos consumos anormales

```
## Identificar anomalías
anomaly_mask = ocsvm.predict(X_scaled) == -1  # -1 indica anomalía
df_cluster['anomalias_ocsvm'] = anomaly_mask.astype(int)

## Contar el total de anomalías en la columna 'anomalias_ocsvm'
total_anomalias_ocsvm = df_cluster['anomalias_ocsvm'].sum()

## Mostrar el total de anomalías
print("Total de anomalías detectadas por OCSVM:", total_anomalias_ocsvm)
```

16] ✓ 2m 10.9s

```
Total de anomalías detectadas por OCSVM: 6237
```

Ilustración cálculo de anomalías

Como se observa con este método, se identificaron 6237 anomalías.

2.5. Resultados

Para analizar los resultados obtenidos de los diferentes modelos se realizó una marcación de lo que se puede llegar a considerar como anomalía bajo la aplicación de la carta de control de calidad *seis sigma*. Este método estadístico toma la desviación estándar de los datos y se determinan limites aceptables de funcionamiento esperado del proceso.

Para el caso particular de Electro Dunas no se definieron límites a nivel global de la compañía, si no se realizó una agrupación por cliente esto con el fin de no mezclar comportamientos típicos de cada cliente. Es importante aclarar, que este valor se toma como referencia para la aproximación al problema, pero en un próximo escenario de evolución del análisis sería ideal poder validar los datos con el cliente para tener una marca previa identificando anomalías bajo su criterio y concepto del negocio.

```python
def identify_outliers(group, threshold=3):
    mean = group['Active_energy'].mean()
    std_dev = group['Active_energy'].std()
    group['Outlier_Active_energy'] = group['Active_energy'].apply(lambda x: 1 if np.abs((x - mean) / std_dev) > threshold else 0)

    mean = group['Reactive_energy'].mean()
    std_dev = group['Reactive_energy'].std()
    group['Outlier_Reactive_energy'] = group['Reactive_energy'].apply(lambda x: 1 if np.abs((x - mean) / std_dev) > threshold else 0)

    mean = group['Voltaje_FA'].mean()
    std_dev = group['Voltaje_FA'].std()
    group['Outlier_Voltaje_FA'] = group['Voltaje_FA'].apply(lambda x: 1 if np.abs((x - mean) / std_dev) > threshold else 0)

    mean = group['Voltaje_FC'].mean()
    std_dev = group['Voltaje_FC'].std()
    group['Outlier_Voltaje_FC'] = group['Voltaje_FC'].apply(lambda x: 1 if np.abs((x - mean) / std_dev) > threshold else 0)

    return group

# Apply the function to each group
outlier_df_AM = SB_Agrupado_AM.groupby('fuente').apply(identify_outliers)
```

Tablas cruzadas resultados modelos

De acuerdo con lo descrito anteriormente se realizan tablas cruzadas de los resultados de las anomalías identificadas por la aplicación del criterio predefinido (desviaciones estándar) y los resultados de clasificación de los modelos implementados (1 es anomalía, 0 no es anomalía).

Modelo	**Anomalía 6 sigma**	**Modelo IF**	
		1	**0**
TODO DEFAULT	1	7786	9866
	0	119348	326415
Contaminación 0.04 random state default	1	1560	16092
	0	16977	428786
Contaminación 0.5 random state default	1	11016	6636
	0	220691	225072
Contaminación 0.12 random state 42	1	5636	12016
	0	49974	395789
Contaminación 0.04 random state 42	1	2117	15535
	0	16420	429343

Se realiza este mismo conteo para los modelos realizados de K-mean y OCSVM:

Modelo K-Means				Modelo OCSVM		
	1	0			1	0
1	3227	14425		1	980	16672

0	19944	425819		0	5257	440506

	TODO DEFAULT	Contaminación 0.04 random state default	Contaminación 0.5 random state default	Contaminación 0.12 random state 42	Contaminación 0.04 random state 42
Cant. Registros	463425	463425	463425	463425	463425
Cant. anomalías IF	127,134	18,537	231,707	55,610	18,537
% Isolation Forest	27.4%	4%	50.0%	12.0%	4.0%
Cant. anomalías Criterio	18,352	18,352	18,352	18,352	18,352
%Criterio	4.0%	4.0%	4.0%	4.0%	4.0%
COMUNES/Anomalías IF	6.1%	8.4%	4.8%	10.1%	11.4%
COMUNES/criterio	42%	9%	60%	31.9%	12%

Aunque en cantidad neta, por el parámetro de contaminación, el modelo de *Isolation Forest* con contaminación del 50% es el que predice una mayor cantidad de anomalías, teniendo en cuenta que se hace el contraste con las anomalías marcadas por el criterio de desviación estándar, se encuentra que el modelo de IF con un parámetro de contaminación del 4% es el que tiene una mayor cantidad de aciertos en común (11.4%).

2.6. Implementación para el prototipo

Para implementar este modelo a la herramienta primero se procederá a preparar los datos realizando las transformaciones necesarias, como se mostró anteriormente, para los datos puedan ser usados en el modelo. Una vez hecho esto se procederá a entrenar el modelo de *Isolation Forest* con estos datos identificando aquellos datos anormales y marcándolos en las gráficas de la herramienta. Una vez la herramienta esté en servicio, se tendrá una carpeta especifica donde se pondrá la información organizada en la misma estructura con la que se entregó una vez puesta allí, se procederá a la transformación y el modelo sacando la información para que pueda usarse en la herramienta (Para más información ver el manual de usuario).

3. Rubrica de evaluación diligenciada

A continuación, se puede observar la rúbrica de evaluación diligenciada:

Aspecto	Requerimiento	Prueba prevista	Criterio o métrica de evaluación	Descripción cumplimiento, ampliación detalle y consideraciones
Negocio				
R1	Visualizar el comportamiento histórico de los clientes, incluyendo información descriptiva.	Validación con stakeholder (o los encargados del curso)	Cumple o no cumple	Se presenta una sección que contiene dos graficas que presentan las series históricas de las 4 variables (se agrupan por un lado la energía activa y reactiva y por otro lado los voltajes), con medidas descriptivas de la población total y por cada cliente. En esta visual se puede hacer filtros por cliente, sector económico, año, mes y día. Los descriptivos de la población se actualizan según los filtros seleccionados (ver anexo imagen 1). Se presenta una sección para cada variable con tres gráficos para cada cliente: el grafico 1 muestra la tendencia general de comportamiento de la variable en una línea suavizada lo que lo diferencia de la sección anterior y los gráficos 2 y 3 muestran la tendencia de consumo por día de la semana y hora del día y permite identificar si hay algún comportamiento particular en estos intervalos. Se

				puede hacer filtro por cliente o por sector económico (ver anexo imagen 2). Sobre este procesamiento para obtener las tendencias, se hace una estimación por cliente y variable, por lo que puede tomar un tiempo en ejecutar, no es automático (con el tamaño actual puede tomar alrededor de 30 minutos).
R2	Visualización de consumos anómalos	Validación con stakeholder (o los encargados del curso)	Cumple o no cumple	Se tiene una sección que muestra sobre la serie original los puntos que se identificaron como anómalos con el modelo de *isolation forest*. Un color más oscuro representa una criticidad más alta (ver anexo imagen 3).
Desempeño				
R3	Se debe encontrar una medida de criticidad de tal manera que se marquen solo los valores que sean relevantes en el contexto.	Hacer pruebas en distintos puntos de corte, dependiendo del modelo, para definir un valor.	Cumple o no cumple	Con el modelo de isolation forest, se obtiene un puntaje (*anomaly score*) que da el punto de corte para definir las anomalías y además funciona como medida de la criticidad. Este puntaje arroja valores entre positivos y negativos, los puntos con un puntaje menor a –0.6 se clasifican como anómalos y de estos se consideran como más críticos los del último cuartil (puntajes más bajos en la serie).
R4	Elegir el mejor modelo para detectar las anomalías en el consumo de energía.	Se evaluarán los diferentes modelos a través de herramientas gráficas y análisis de los resultados eligiendo el que mejor represente las anomalías	Cumple o no cumple	Se probaron los modelos *K-means*, *One-Class Support Vector Machine* e *Isolation Forest*. Al tener un problema no supervisado, se toma como referencia los puntos marcados por medio de un modelo de control de calidad, carta *seis sigma*. Se hizo una comparación de las anomalías identificadas por los distintos modelos contra estos puntos marcados y se elige para la aplicación el modelo *isolation forest*, al ser el que tenía una mayor tasa de puntos identificados en común con los puntos marcados con la carta *seis sigma*. El modelo se eligió teniendo en cuenta los datos actuales, en caso de incluirse nuevos datos sería recomendable hacer una nueva iteración para mantener resultados confiables en el contexto del problema.
Funcional				
R5	Visualización amigable con el usuario	Validación con stakeholder (o los encargados del curso)	Cumple o no cumple	Se mantiene constante el uso de un solo tipo de graficas (línea-serie de tiempo) esperando que la familiarización sea más sencilla y la lectura sea más rápida.

				Se usan colores distintos para cada variable y para marcar las anomalías, de tal manera que se puedan identificar y diferenciar. Se mantienen sencillas las instrucciones en el tablero, para no tener una cantidad extensa de texto que podría dificultar la interacción, pero se incluye un manual detallado para el usuario con los detalles de uso.
R6	Mostrar información clara y comprensible	Validación con stakeholder (o los encargados del curso)	Cumple o no cumple	Se tienen varias secciones que permiten enfocar la información y consultar según la necesidad (histórico, tendencias y anomalías). Se incluyen títulos, etiquetas en las gráficas y las unidades de las variables. Según la sección consultada, se incluyen diferentes filtros que permiten consultar información por cliente, sector económico o momento de tiempo (año, día). Estos filtros son útiles para comparaciones.
R7	Uso de software y librerías de manejo libre	Validación	Cumple o no cumple	Para el análisis descriptivo y el modelado de anomalías se usa Python, un software de libre acceso. Para la visualización, se usó PowerBI que es de uso libre con una licencia de Microsoft. Con el tamaño actual de problema, no se consideró necesario hacer uso de herramientas externas para el almacenamiento, todo se puede ejecutar a nivel local.

Anexos tabla de requerimientos

Imagen 1: consumo histórico con datos descriptivos

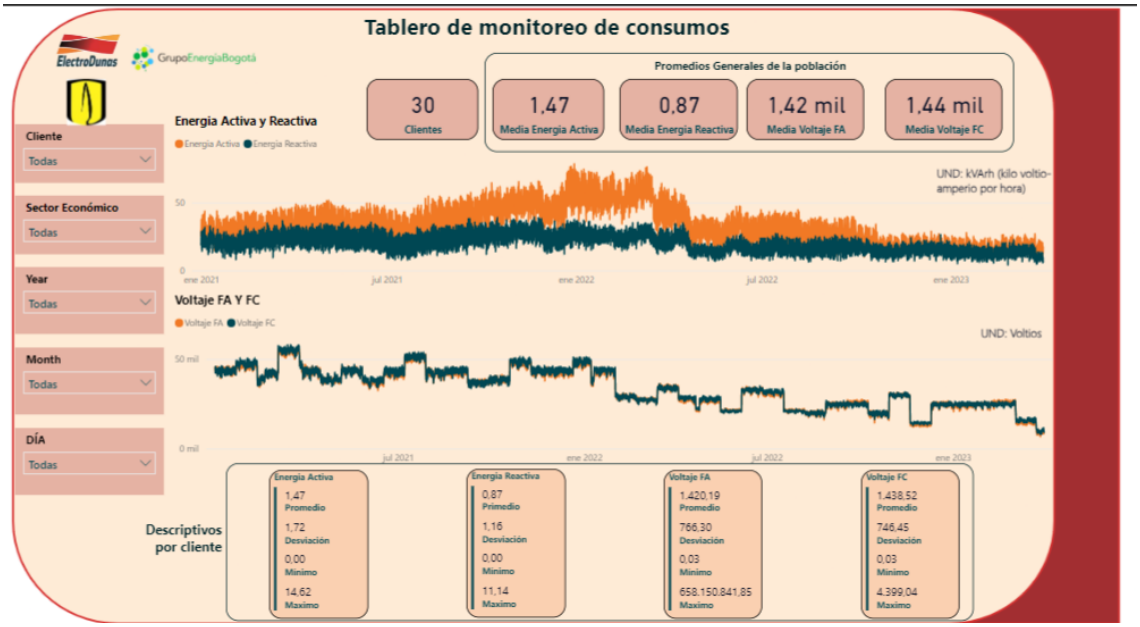


Imagen 2: tendencia de variable y patrones por día y hora

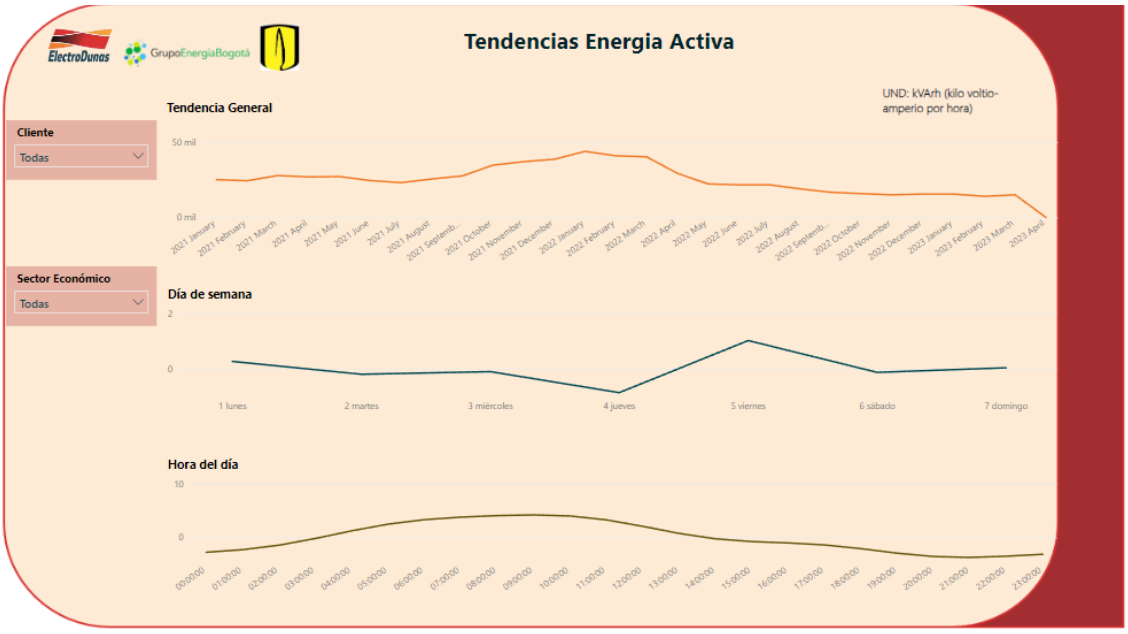
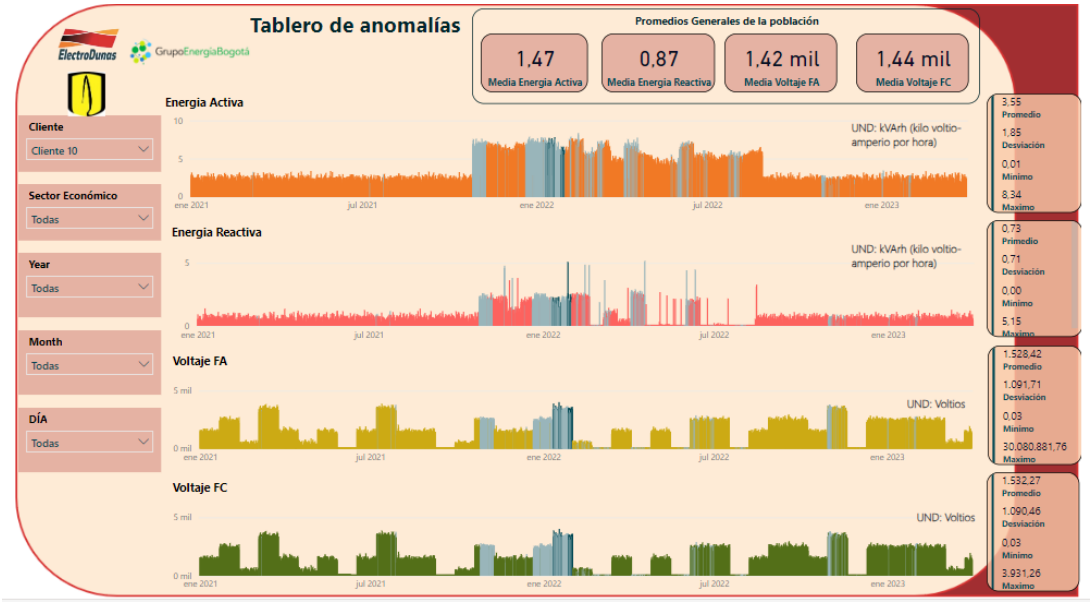


Imagen 3: visualización anomalías



4. Archivos de Código Desarrollado

Los archivos de código desarrollado se pueden encontrar en el repositorio de github a través del siguiente enlace: [Proyecto_Grado_MIAD]

En este repositorio se encuentran todos los notebooks donde se desarrolló la evaluación de los diferentes modelos junto con un archivo en el que se explica cómo están organizados las carpetas y los archivos para facilitar la búsqueda de la información (archivo readme).

La presentación con el demo del prototipo actual se encuentra en el siguiente enlace https://www.youtube.com/watch?v=RyAid_J0ERA&feature=youtu.be y el prototipo en Power BI puede ser visualizado en el siguiente enlace https://app.powerbi.com/view?r=eyJrIjoiNzgxNGQ1NWItOWVmMC00YWNjLWIyN2QtMDA0NjAwZWY0NmUwIiwidCI6ImQ2NDUyMzRkLTI2OGEtNGVhYS05NWEwLWFlOTFmOTRkY2Q4MSIsImMiOjR9