

Tópicos Avanzados de IA: Redes complejas y aprendizaje computacional.

Alumno: Ohtli Gerardo Quiroz Sánchez
Prof.: Dr. Matías Alvarado Mentado.

1 Objetivo

En este trabajo se plantea una red compleja de las estadísticas operativas selectas de la base de datos de Petróleos Mexicanos (Pemex). Las estadísticas operativas van desde la producción hasta la comercialización interior y exterior de productos como petrolíferos, petroquímicos, gas natural, etc. A partir de la cual se busca encontrar las redes de mundo pequeño (small-world en inglés). Por lo tanto, se hará uso de un método ingenioso como GENIE3 usada en la inferencia de redes reguladoras desde los datos de expresión genética [2]. Este método usa bosques aleatorios como aprendizaje automático para calcular la importancia de una instancia en la predicción de uno de los objetivos [2], por lo que esta importancia se usará como indicador para la construcción de un enlace.

2 Teoría

Una red de $|N| = n$ nodos y $|M| = m$ aristas, $G(N, M)$, puede representarse de varios modos entre los cuales está la representación gráfica y la matriz de adyacencia $A = (a_{ij})$ donde

$$a_{ij} = \begin{cases} 1, & \text{si hay una arista entre el nodo } i \text{ y } j \\ 0, & \text{en otro caso} \end{cases}$$

En una red podemos encontrar varias propiedades que caracterizan, uno de ellos son el coeficiente de agrupamiento y la longitud entre dos nodos.

2.1 Coeficiente de Agrupamiento

Consideremos las rutas de longitud 2. Supongamos la ruta formada por una triada de nodos uvw donde tenemos que el nodo v es adyacente a los nodos u y w . Si u y w son adyacentes, uvw forman un triángulo. El coeficiente de

agrupamiento está dada por

$$C = \frac{3(\text{número de triángulos})}{(\text{número de triadas})} \quad (1)$$

es la proporción de triángulos en una red. Un valor del coeficiente va de 0 a 1, si el coeficiente es 0 significa que no hay triángulos en la red; si el coeficiente es 1, cada nodo forma un triángulo con los demás.

2.2 Modelo de Grafo aleatorio de Erdos-Rényi

Sea $\mathcal{G}_{n,m}$ el conjunto de todos los grafos con n nodos y exactamente m aristas, $0 \leq m \leq \binom{n}{2}$. Para todo $G \in \mathcal{G}_{n,m}$,

$$P(\mathbb{G}_{n,m} = G) = \binom{\binom{n}{2}}{m}^{-1}$$

donde la probabilidad de un grafo cualquiera con n nodos y m aristas es el inverso de todas las posibles combinaciones de m aristas de $\binom{n}{2}$ posibles aristas.

2.3 Mundo pequeño

Una red que presentan la propiedad de mundo pequeño tienden a tener un coeficiente de agrupamiento alto y un promedio de longitud mínima L .

Definición 1 La red G se dice que es una red de mundo pequeño si $L_g \geq L_{rand}$ y $C_g^{\Delta} \gg C_{rand}^{\Delta}$ [1].

Sean

$$\gamma_g = \frac{C_g}{C_{rand}} \quad (2)$$

y

$$\lambda_g = \frac{L_g}{L_{rand}} \quad (3)$$

donde C_g y L_g son el coeficiente de agrupamiento y la longitud promedio de nuestra red, respectivamente. Las variables C_{rand} y L_{rand} corresponden al grafo aleatorio. La métrica de mundo pequeño (small-world-ness en inglés) S está dado por

$$S = \frac{\gamma_g}{\lambda_g} \quad (4)$$

Definición 2 Una red se dice que es de mundo pequeño si $S > 1$. [1].

De esta forma podemos saber con mayor precisión si una red presenta la propiedad de mundo pequeño.

2.4 Métricas de desempeño

EN este trabajo se usará dos métricas para la evaluación del desempeño de los modelos de aprendizaje automático con regresión, en nuestro caso el Random Forest, que son R^2 , el error cuadrático medio (RMSE por las siglas en inglés de Root Mean Square Error) y Puntuación fuera de la bolsa (OOB por las siglas en inglés out-of-bag):

- El valor R^2 , que va de 0 a 1, representa la proporción de la varianza en la variable dependiente (objetivo) que es predecible a partir de las variables independientes (características) del modelo [3]. Un valor R-cuadrado más alto sugiere que el modelo se ajusta bien a los datos.

$$R^2 = 1 - (SRC/STC) \quad (5)$$

donde

- SRC (Suma de residuos al cuadrado): es la suma de los cuadrados de los residuos. Los residuos son la diferencia entre los valores reales de la variable dependiente y los valores predichos por el modelo de regresión [4].
- STC (Suma total de cuadrados): es la suma de las diferencias entre los valores reales de la variable dependiente y el valor medio de la variable dependiente, todos elevados al cuadrado [4].
- El RMSE es un estimador que mide el promedio de los errores al cuadrado. Un valor RMSE alto sugiere que las predicciones del modelo están bastante lejos de los valores reales en promedio [3].
- OOB estima el rendimiento de generalización del modelo [3]. Indica el rendimiento del modelo utilizando muestras no incluidas en el proceso de agregación bootstrap (el re-muestreo con reemplazo) [3].

3 Metodología

En este trabajo se intenta inferir una red compleja en base a un método ingenioso conocido como GENIE3. Este método ajusta un modelo de bosques aleatorios para cada variable. Posteriormente se infiere una red en base en la importancia que tiene una variable (nodo) en la predicción de otro tomando esto como indicador para crear un enlace o arista entre estas [2]. La propuesta para el conjunto de datos de las operaciones selectas de PEMEX es el siguiente:

- Cada fila es un nodo (Hidrocarburos líquidos, petróleo crudo, complejos petroquímicos, etc.)
- Para cada nodo i , se entrena un **Random Forest** como aprendizaje automático para predecir su serie mensual usando, en el mismo mes, las series del resto de nodos.

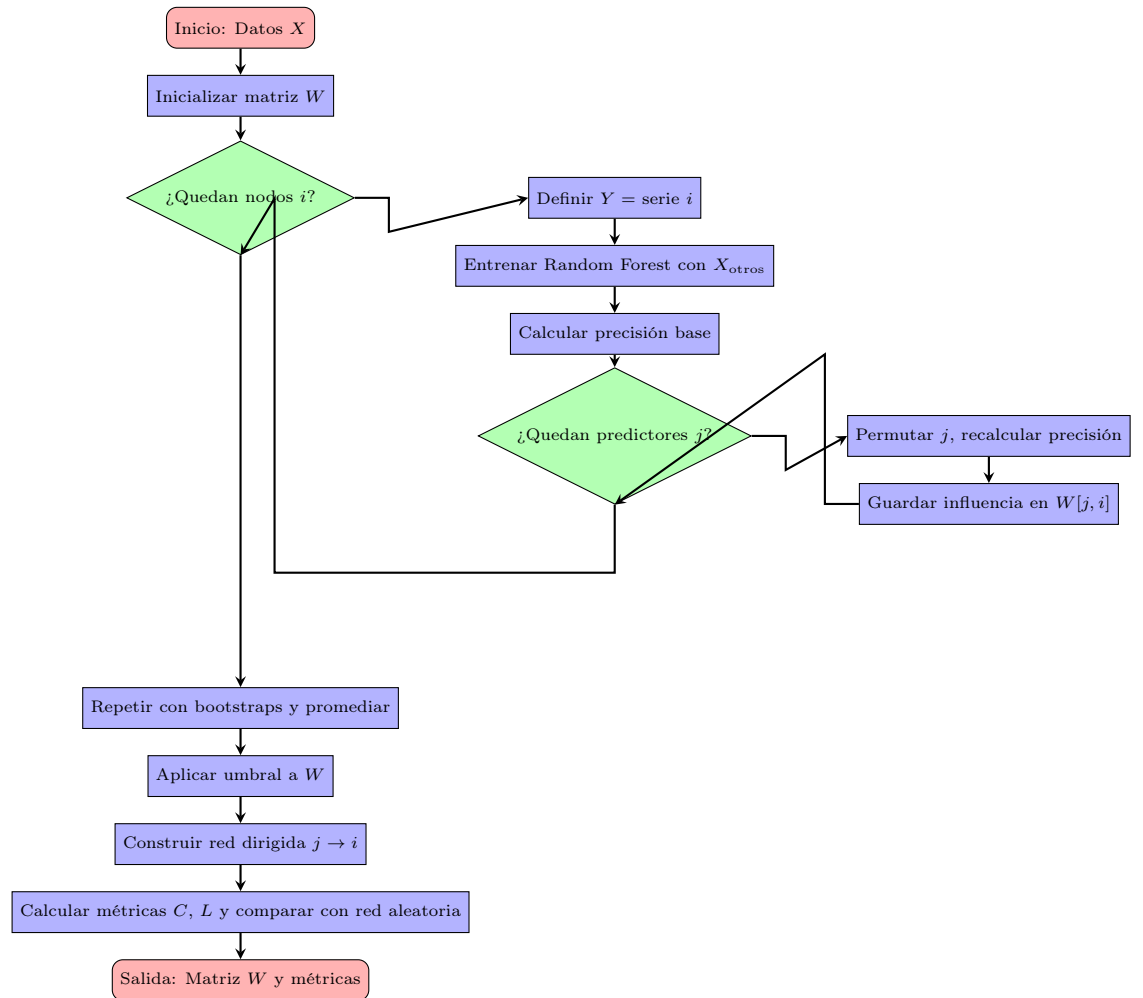
- Usa el método **importancia por permutación** (permutation importance) para medir la influencia de un nodo sobre otro:
Una vez entrenado el modelo *Random Forest* de un nodo i (nodo objetivo):
 - Primero se mide qué tan bien predice el modelo.
 - Luego, se toma la serie mensual de un nodo j (por ejemplo, "volumen de ventas de gasolina magna") y se **desordena al azar**.
 - Si al hacer esto el modelo de i pierde precisión, significa que la información de j era importante para predecir a i .
 - Cuanto más empeore el modelo al desordenar a j , mayor es la influencia de j sobre i .

Se repite este procedimiento para todos los nodos j , y se obtiene incluso una medida de influencia dirigida. El resultado final es una **matriz de influencias \mathbf{W}** que nos dice quién influye sobre quién y con qué fuerza

- **Umbralización.** Se aplica un umbral a \mathbf{W} en base al percentil 85, y otro para 90 para decidir qué influencias son fuertes para convertirse en aristas dentro de un grafo.
- **Calcula la métrica de mundo pequeño (Ecuación 4).** Si $S > 1$, entonces la red construida es de mundo pequeño.

Algorithm 1 Construcción de la red de influencias

- 1: **Entrada:** Datos X con T meses y N variables (producción, ventas, volumen, etc.), percentil p
 - 2: Inicializar matriz W de tamaño $N \times N$ en ceros
 - 3: **for** cada variable $i = 1..N$ **do**
 - 4: Definir objetivo $Y =$ variable i
 - 5: Definir predictores $X_{\text{otros}} =$ todas las variables $j \neq i$
 - 6: Entrenar un modelo **Random Forest** para predecir Y usando X_{otros}
 - 7: Calcular precisión base del modelo.
 - 8: **for** cada predictor $j \in X_{\text{otros}}$ **do**
 - 9: Guardar los valores reales de la variable j
 - 10: Desordenar aleatoriamente los valores de j (romper relación con Y)
 - 11: Recalcular la precisión del modelo
 - 12: Calcular caída en precisión = influencia de j sobre i
 - 13: Guardar este valor en $W[j, i]$
 - 14: **end for**
 - 15: **end for**
 - 16: Repetir los pasos anteriores en varios subconjuntos (bootstraps)
 - 17: Promediar resultados para estabilizar W
 - 18: Aplicar umbral a W en base al percentil p
 - 19: Construir red dirigida/no dirigida: Aristas con $W[j, i] > 0$
 - 20: Calcular métricas de la red: Nodos = variables Coeficiente de agrupamiento C Longitud media de camino L Comparar con red aleatoria Erdos Rényi de igual tamaño Si $S > 1$ (Ecuación 4), la red es de tipo “mundo pequeño”
 - 21: **Salida:** Matriz de influencias W , red compleja y valor de S (métrica de small-worldness (4))
-



3.1 Código

```

1 import pandas as pd
2 import numpy as np
3 from sklearn.ensemble import RandomForestRegressor
4 from sklearn.inspection import permutation_importance
5 from sklearn.utils import resample
6 import networkx as nx
7
8 # ===== CONFIGURACIÓN =====
9 CSV_PATH = "Data/Estadísticas_operativas_seleccionadas_Mayor.csv"
10 # <-- tu archivo (filas = productos, columnas = meses)
11 index_col_name = 0 # si la primera columna contiene el nombre del
12 # producto
13 n_estimators = 500
14 random_state = 50

```

```

13 n_boot = 30 # número de bootstraps temporales
14 top_percent = 85 # percentil de corte para aristas (mantener las m
    ás fuertes)
15
16 # ===== CARGA DE DATOS =====
17 df = pd.read_csv(CSV_PATH, encoding="latin-1", index_col=
    index_col_name)
18
19 # Normaliza cada fila (producto) con z-score
20 df_z = df.T.apply(lambda s: (s - s.mean()) / (s.std(ddof=1) + 1e-9)
    ).T
21
22 series_names = df_z.index.tolist()
23 n = df_z.shape[0]
24
25 # ===== MATRIZ DE INFLUENCIAS (RF + BOOTSTRAP)
    =====
26 W = pd.DataFrame(0.0, index=series_names, columns=series_names) #
    fuente -> objetivo
27
28 rng = np.random.RandomState(random_state)
29 for target in series_names:
30     y = df_z.loc[target, :].values
31     # predictores: todas las demás series
32     X = df_z.drop(index=target).T.values
33     feat_names = df_z.drop(index=target).index.tolist()
34
35     imp_accum = np.zeros(len(feat_names), dtype=float)
36
37     for b in range(n_boot):
38         # bootstrap por columnas (meses)
39         idx = resample(np.arange(len(y)), replace=True,
            random_state=rng.randint(0, 1_000_000))
40         Xb = X[idx, :]
41         yb = y[idx]
42
43         rf = RandomForestRegressor(
44             n_estimators=n_estimators,
45             random_state=rng.randint(0, 1_000_000),
46             n_jobs=-1,
47             max_features="sqrt"
48         )
49         rf.fit(Xb, yb)
50
51         # importancia por permutación
52         pi = permutation_importance(rf, Xb, yb, n_repeats=5,
            random_state=rng.randint(0, 1
53             _000_000), n_jobs=-1)
54         imp_accum += np.maximum(0, pi.importances_mean)
55
56     imp_mean = imp_accum / n_boot
57     if imp_mean.max() > 0:
58         imp_mean = imp_mean / imp_mean.max()
59
60     for k, src in enumerate(feat_names):
61         W.loc[src, target] = imp_mean[k]
62

```

```

63 W.to_csv('matrizInfluencias.csv', index=True) #Guardamos los datos
64
65 # ===== CONSTRUCCIÓN DE LA RED =====
66 '''
67 thr es el umbral de corte que selecciona solo las conexiones más
68 fuertes (por percentil), para que la red no se llene de ruido
69 '''
70 thr = np.percentile(W.values[W.values > 0], top_percent) if (W.
    values > 0).sum() > 0 else 1.0
71
72 A = (W >= thr).astype(int) # adyacencia binaria por percentil
73 G = nx.from_pandas_adjacency(pd.DataFrame(A, index=series_names,
    columns=series_names),
74                             create_using=nx.DiGraph)
75
76 # Convertir a no dirigido para small-world clásico
77 G_und = G.to_undirected()
78 G_und.remove_nodes_from(list(nx.isolates(G_und))) # elimina nodos
    aislados
79
80 # ===== FUNCIÓN SMALL-WORLD =====
81 def small_world(Gu):
82     if Gu.number_of_nodes() < 3 or Gu.number_of_edges() == 0:
83         return np.nan, np.nan, np.nan, np.nan
84     # componente gigante
85     components = list(nx.connected_components(Gu))
86     GC = Gu.subgraph(max(components, key=len)).copy()
87
88     C = nx.transitivity(GC) # clustering global
89     L = nx.average_shortest_path_length(GC)
90
91     # Grafo aleatorio ER con mismo n y p
92     n_nodes = GC.number_of_nodes()
93     m_edges = GC.number_of_edges()
94     p = (2 * m_edges) / (n_nodes * (n_nodes - 1))
95     ER = nx.gnp_random_graph(n_nodes, p, seed=123)
96     Cr = nx.transitivity(ER)
97     Lr = nx.average_shortest_path_length(ER)
98
99     S = (C/Cr)/(L/Lr) if (Cr > 0 and Lr > 0) else np.nan
100     return C, L, Cr, Lr, S
101
102 # ===== RESULTADOS =====
103 C, L, Cr, Lr, S = small_world(G_und)
104 print(f"Nodos: {G_und.number_of_nodes()}, Aristas: {G_und.
    number_of_edges()}")
105 print(f"C={C:.3f}, L={L:.3f}, C_rand={Cr:.3f}, L_rand={Lr:.3f}, S={
    S:.3f}")
106
107 if S > 1:
108     print("La red muestra propiedades de mundo pequeño.")
109 else:
110     print("No se detectan propiedades claras de mundo pequeño.")
111
112

```

Listing 1: Función Reference Set Update

4 Datos

Los datos se obtuvieron de las estadísticas operativas selectas de la base de datos de Petróleos Mexicanos (Pemex) (página <https://ebdi.pemex.com/bdi/>). Algunos de los datos que fueron removidos fueron porque tenían demasiados datos faltantes (valores N/D equivalentemente a valores nan), estos fueron exportaciones Petroquímicos (Mt) (MMUS), Exportaciones Petroquímicos (Mt) (volumen) y Istmo US\$/b. En la Figura (1) se puede apreciar las graficas de estas instancias que no tienen valores faltantes.

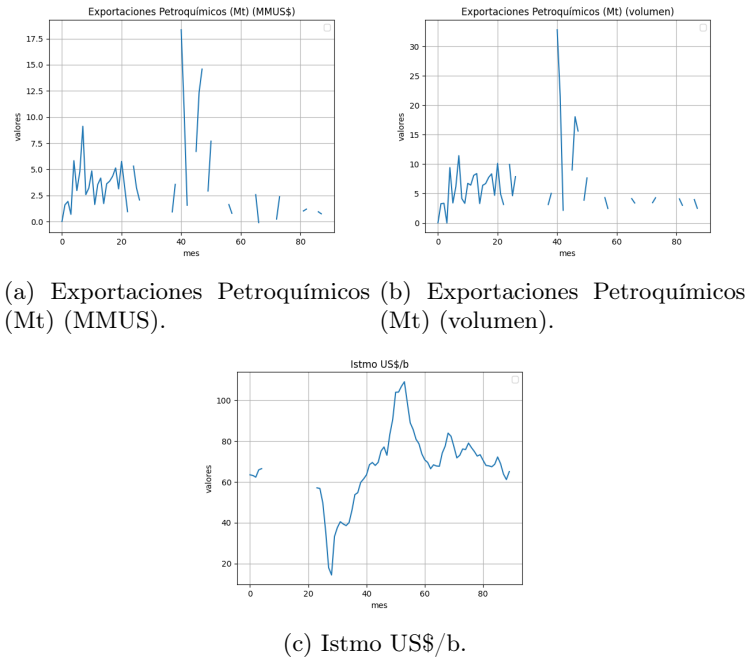


Figure 1: Valores faltantes.

5 resultados

En la ejecución del programa se estableció la siguiente configuración:

1. $n_estimators = 500$ (número de árboles)
2. $random_state = 50$ (semilla)
3. $n_boot = 30$ número de bootstraps (re-muestreo con reemplazo)
4. $top_percent =$ percentil 85 y 90 de corte para aristas (mantener las más fuertes)

Percentil	Nodos	Aristas	C	L	C_{rand}	L_{rand}	S
85	53	319	0.422	1.910	0.229	1.787	1.728
90	53	221	0.362	2.231	0.172	2.037	1.926

Table 1: Resultados de ambas redes complejas. Ambas presentan la propiedad de mundo pequeño.

Al terminar la ejecución se guardará la matriz de influencias W para que no halla necesidad de tener que ejecutar el programa de nuevo debido a su costo computacional.

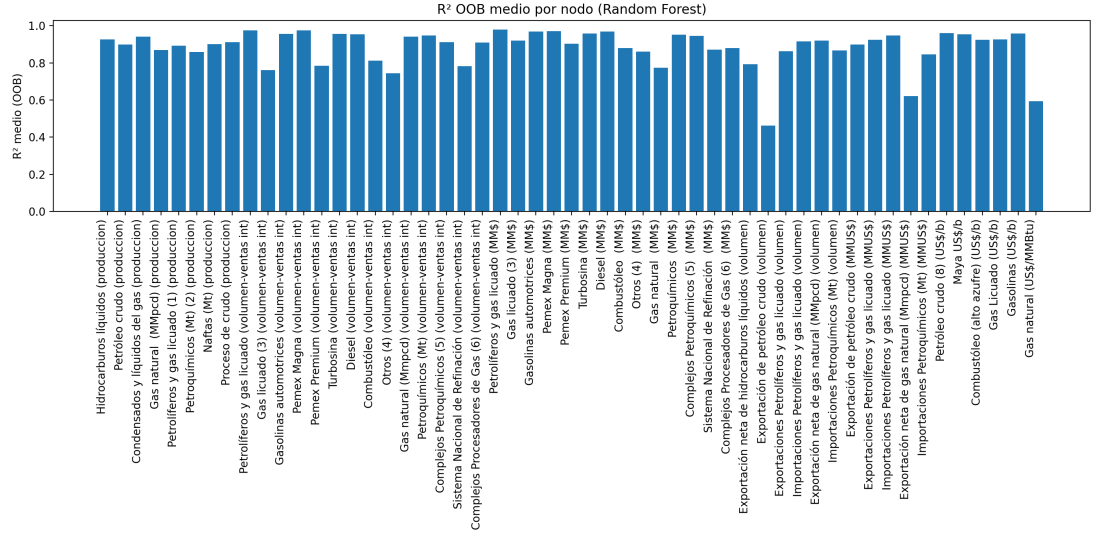
Se realizó la evaluación del desempeño de los modelos random forest para estimar cada nodo de la red compleja de PEMEX. Los resultados muestran , véase la Figura (2 a) que la mayoría de los nodos presentan valores de R^2 promedio cercanos a 1, lo que nos indica que los modelos se ajustan bien a los datos. Sin embargo, hay nodos como exportaciones de petroleo crudo (volumen) bajos con una puntuación de 0.4 como se ve en la gráfica. Por otro lado, en cuanto al error cuadrático medio, véase la Figura (2 b) por lo general se mantuvo por debajo de 0.5. Si los valores estuvieran cercanos a 1 nuestro modelo no estaría prediciendo bien si tomamos en cuenta que normalizamos los datos con z-score (puntuación z).

De esta forma, podemos observar que nuestro modelo de Random Forest puede captar patrones relevantes en la mayoría de las variables lo que respalda el uso de la importancia por permutación.

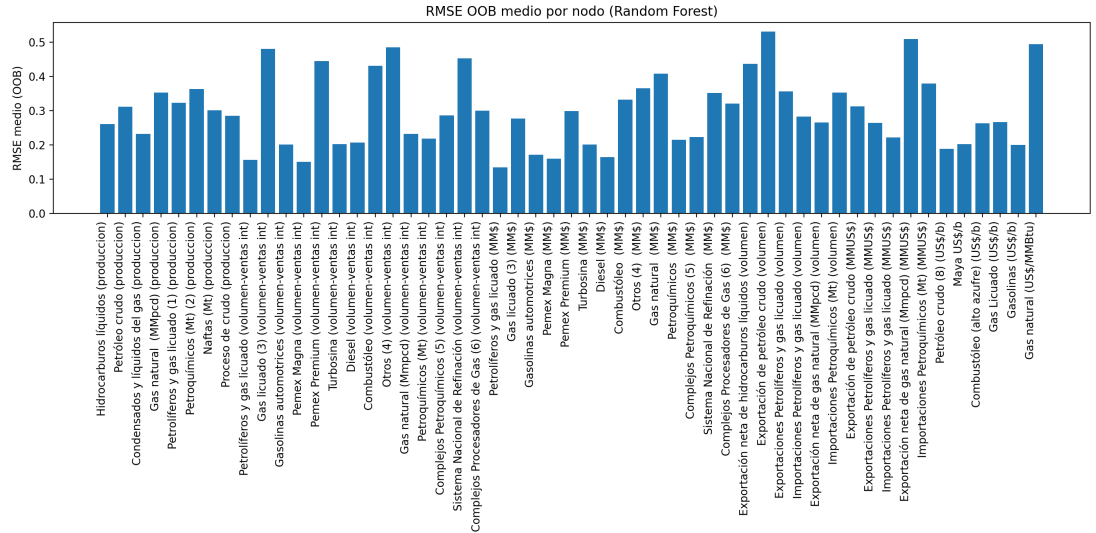
En conjunto, estos resultados permiten concluir que, aunque existen nodos con baja capacidad predictiva, el Random Forest capta patrones relevantes en la mayoría de las series, lo que respalda el uso de la importancia por permutación como medida robusta de influencia para la construcción de la red.

En la Figura (3) muestro el mapa de calor de la matriz de influencias, es decir, qué tanto cada variable (ej. exportación, ventas internas, etc.) contribuye a explicar a las demás en el modelo Random Forest. A partir de esta matriz se decide crear dos umbrales a partir del cual vamos a quitar las aristas que tengan un valor menor al percentil 85 y al percentil 90. Posteriormente procedemos a crear la red compleja que se muestra en las Figuras (4) y (5).

Los resultados de la red compleja creada a partir del umbral con percentil 85 en la matriz de influencias son 53 nodos y 319 aristas donde presentó un valor en la métrica de mundo pequeño S igual a 1.728, por lo que se considera de mundo pequeño (véase la Ecuación 4). En el percentil 90, disminuyeron aún más el número de aristas hasta 221 con un aumento en S de 1.926 por lo que también es de mundo pequeño. En la Tabla (1) se resumen estos valores.



(a) R (OOB).



(b) $RMSE$ (OOB).

Figure 2: Gráfica de desempeño de random forest.

References

- [1] Mark D. Humphries and Kevin Gurney. “Network ‘Small-World-Ness’: A Quantitative Method for Determining Canonical Network Equivalence”. In:

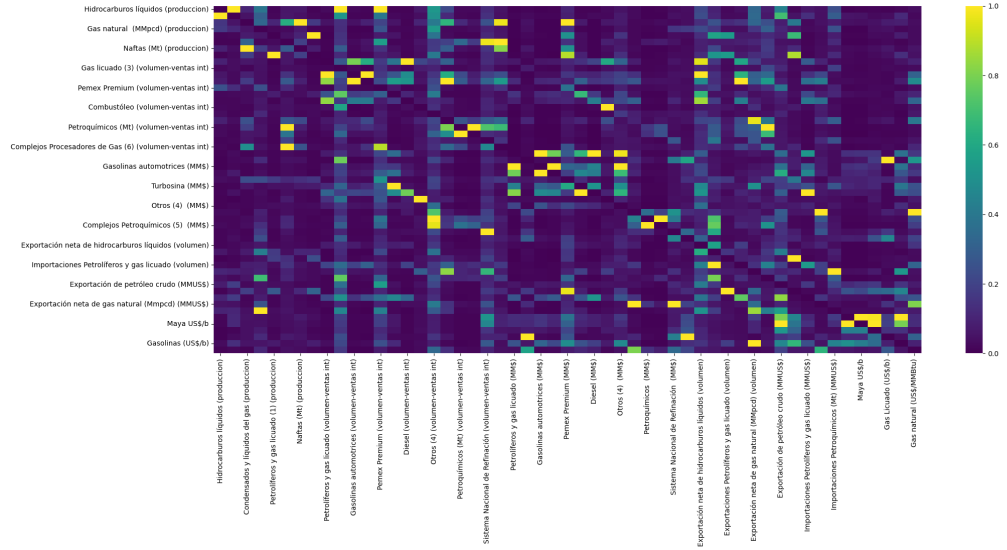


Figure 3: Mapa de calor de la matriz de influencias

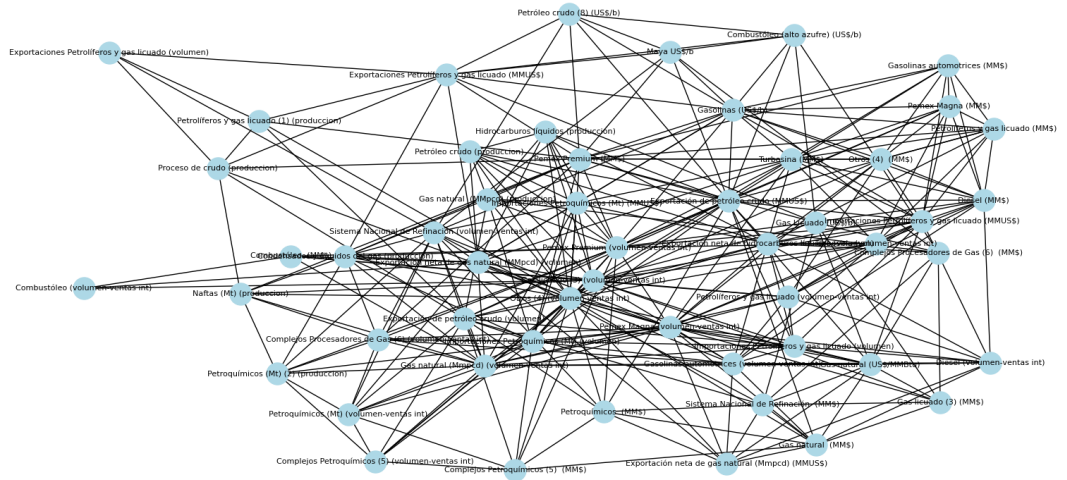


Figure 4: Red compleja con el percentil 85

