In [94]:
```
from IPython.display import Image
import pandas as pd
from bokeh.io import output_notebook, show
from bokeh.plotting import figure
from bokeh.models import Span

output_notebook()
```

```
BokehJS 1.1.0 successfully loaded.
```
(https://bokeh.pydata.org)

Informe análisis Orange

Luego de realizar un análisis de los gráficos obtenidos en Microstrategy, se decidió analizar a través de diversos modelos el dataset. Para este caso, se utiliza la misma muestra que en el caso anterior.

Workflow de Orange

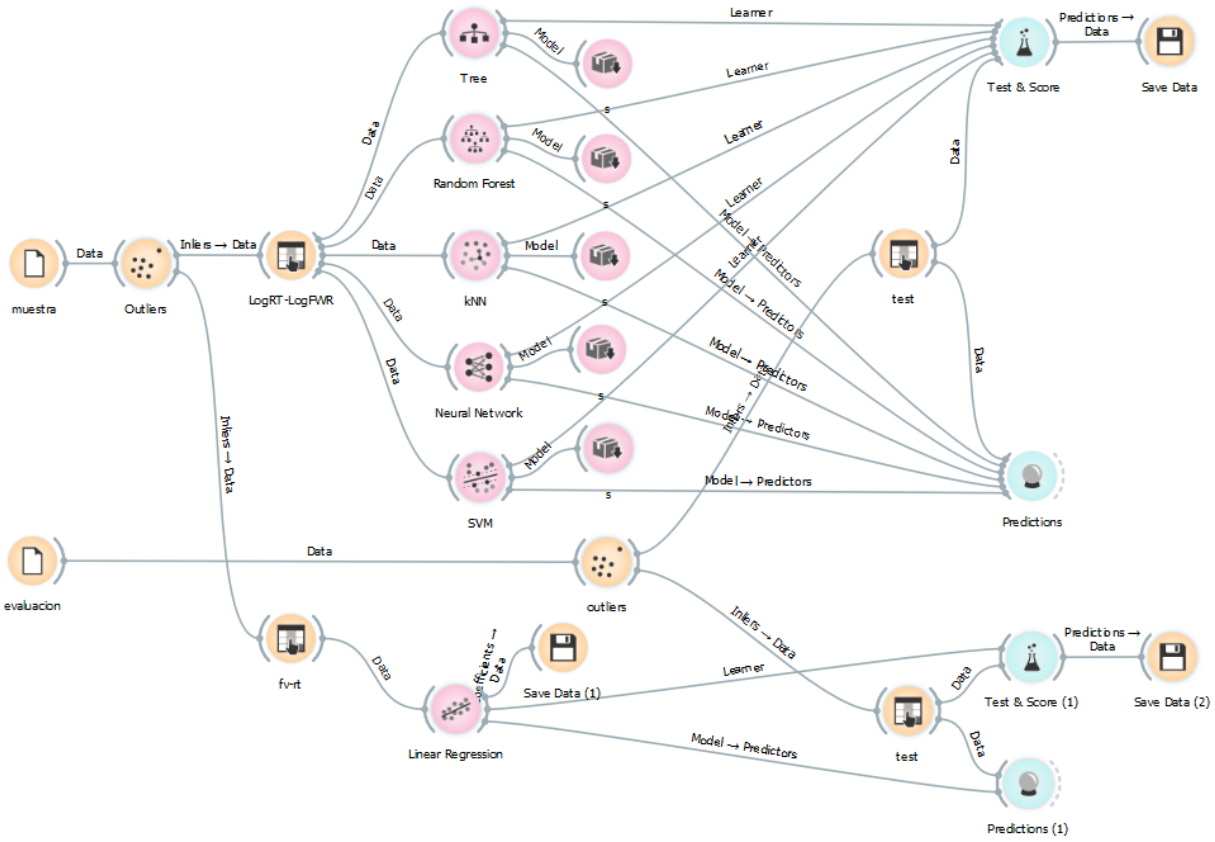
Para el estudio de los datos, se utilizaron los siguientes modelos:

- kNN
- SVM
- Random Forest
- Regresión lineal
- Arbol de desición
- Redes neuronales

El estudio realizado se basa en 2 análisis, el primero es la relación entre los diversos atributos numéricos y categóricos, esto para predecir el comportamiento de los RT. El segundo busca modelar el comportamiento de los FV en relación a los RT.

Para ambos casos de estudio se realizan 4 instancias distintas variando la forma en que se presentan los RT y FV del hilo:

1. Suma total del hilo.
2. Promedio de la suma total del hilo
3. Logaritmo de la suma total del hilo
4. Logaritmo del promedio de la suma total del hilo



Análisis RT-FV

En el análisis según gráficos, se observó una relación entre los RT y los FV del hilo, por lo que se busca modelar la relación entre estos, para esto se analizarán las relaciones RT-FV, log(RT)-Log(FV), promedioRT-promedioFV y Log(promedioRT)-Log(PromedioFV).

Como los hilos son de distinto largo, los hilos de mayor longitud tienen mayores probabilidades de sumar un número mayor que los de menor longitud, por lo que se busca excluir esa variable al calcular el promedio. Si es que el largo del hilo llegase a influir en la relación, la diferencia entre rt-fv y prt-fv

In [95]:
```
columnas = ["Relacion", "intercepto", "variable", "MSE", "RMSE", "MAE", "R2"]
rt_fv = ["RT vs FV", -12832.812328905835, 5.265690537263298, 677154880.129, 2602
lrt_lfv = ["log(RT) vs log(FV)", 0.5963644608531382, 0.9661846950338701, 0.100, (
prt_pfv = ["prom(RT) vs prom(FV)", -603.5396039332963, 4.4057284130596015, 25853
lprt_lpfv = ["log(prom(RT)) vs log(prom(FV))", 0.507194771931951, 0.984196470995(
lista_rtfv = [rt_fv, lrt_lfv, prt_pfv, lprt_lpfv]
df_rtfv = pd.DataFrame(lista_rtfv, columns = columnas)
df_rtfv
```

Out[95]:

	Relacion	intercepto	variable	MSE	RMSE	MAE	R2
0	RT vs FV	-12832.812329	5.265691	6.771549e+08	26022.200	11939.057	0.887
1	log(RT) vs log(FV)	0.596364	0.966185	1.000000e-01	0.316	0.226	0.869
2	prom(RT) vs prom(FV)	-603.539604	4.405728	2.585356e+06	1607.904	723.751	0.897
3	log(prom(RT)) vs log(prom(FV))	0.507195	0.984196	7.400000e-02	0.272	0.201	0.901

Finalmente, de estos resultados, se observa que la relación existe sin importar la forma en la que se modele, eso sí, se observa que el resultado mejora al considerar el promedio en vez del total y además en ambos casos, el modelo logarítmico entrega mejores resultados.

Finalmente se concluye que la relación entre RT y FV posee un comportamiento potencial, debido a esto, se estudian las relaciones entre los demás atributos y RT, ignorando FV.

Análisis de Modelos

Como se mencionó anteriormente, se buscará modelar el comportamiento entre los atributos del hilo y la cantidad de RT que recibe, por lo que primero se presentan las variables de estudio:

Los atributos a estudiar en este apartado serán los siguientes:

- Total Tweets: Largo del hilo. (número)
- Total HT: Cantidad total de hashtags usados en el hilo (número)
- Tópico: Número de tópico en el cual quedó clasificado (Categórico)
- Tupla Emoción: Tupla de primera y segunda emoción del hilo (categórico)
- Puntaje Sigmoídeo: Relación entre cantidad de seguidores y seguidos del autor (número)
- Log Followers: Logarítmo de la cantidad de seguidores del autor (número)
- Verificado: Si la identidad de la cuenta autora del hilo fue verificada por Twitter (categórico)

Además de los atributos ya expuestos, se poseen 4 variables objetivo, las cuales serán alternadas para estudiar cual posee mejor comportamiento en los diversos modelos:

- Total RT
- Promedio RT
- Log(Total RT)
- Log(Promedio RT)

Se eveeluarán los logaritmos de las variables debido a la distribución que presentan.

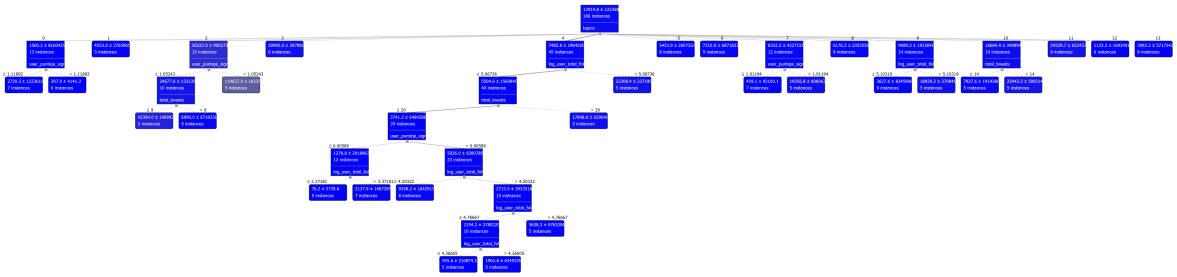
En todos los casos primero se analizará el arbol generado y luego los errores de los modelos generados.

Para los arboles generados, la codificación de colores indica con valores amarillos mejores resultados.

Total RT

Primero se estudia la suma total de los RT sin ningún ajuste, esto para obtener los valores base de comparación de los otros Objetivos.

Árbol de decisión



A simple vista se observa que el arbol de decisión no logró armar un conjunto de variables que entregue resultados destacables, lo que es un primer indicio sobre la calidad de los demás modelos en respecto al Objetivo utilizado.

In [96]:
```
columnas = ["Metodo", "MSE", "RMSE", "MAE", "R2"]
rt_neural = ["Redes Neuronales",551203245.773, 23477.718, 10731.293, -0.263]
rt_forest = ["Random Forest"   ,489847743.456, 22132.504, 10729.242, -0.122]
rt_svm    = ["SVM"             ,492463854.119, 22191.527, 10100.529, -0.128]
rt_tree   = ["Tree"            ,424930805.505, 20613.850, 11182.439,  0.026]
rt_knn    = ["kNN"             ,459045922.564, 21425.357, 10219.362, -0.052]
rt_lista_metodos = [rt_neural, rt_forest, rt_svm, rt_tree, rt_knn]
df_rt = pd.DataFrame(rt_lista_metodos, columns = columnas).set_index("Metodo")
df_rt
```

Out[96]:

Metodo	MSE	RMSE	MAE	R2
Redes Neuronales	5.512032e+08	23477.718	10731.293	-0.263
Random Forest	4.898477e+08	22132.504	10729.242	-0.122
SVM	4.924639e+08	22191.527	10100.529	-0.128
Tree	4.249308e+08	20613.850	11182.439	0.026
kNN	4.590459e+08	21425.357	10219.362	-0.052

Se observa a través de R2 que todos los modelos poseen pésimos resultados, por lo que se comprueba la noción entregada por el árbol.

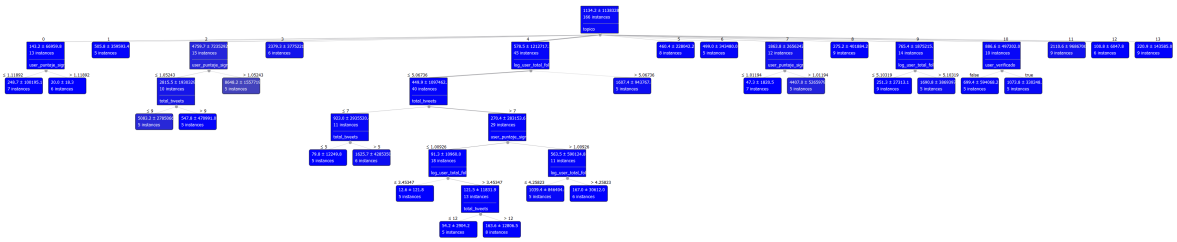
Promedio RT

El segundo Objetivo es el promedio de RT del hilo, este valor se calcula dividiendo el total de RT por el largo del hilo.

Se busca mejorar los modelos tratando de mejorar los intervalos entre los valores de RT, por lo que se promediará el valor total RT según el largo del hilo, pero se mantendrá el largo en los atributos utilizados para modelar.

En otras palabras, se busca que el valor utilizado como "éxito" no sea mayor por el simple hecho de que sea más largo el hilo.

Árbol de decisión



Se observa que al igual que el caso anterior, el arbol presenta pésimos resultados, por lo que se esperan malos resultados también en los demás modelos.

In [97]:
```
columnas = ["Metodo", "MSE", "RMSE", "MAE", "R2"]
prt_neural = ["Redes Neuronales", 3332334.231, 1825.468, 873.400, -0.277]
prt_forest = ["Random Forest"   , 2586659.886, 1608.310, 838.344, 0.009]
prt_svm    = ["SVM"             , 3081994.146, 1755.561, 854.727, -0.181]
prt_tree   = ["Tree"            , 2844566.569, 1686.584, 967.063, -0.090]
prt_knn    = ["kNN"             , 2526030.463, 1579.349, 751.189, 0.032]
prt_lista_metodos = [prt_neural, prt_forest, prt_svm, prt_tree, prt_knn]
df_prt = pd.DataFrame(prt_lista_metodos, columns = columnas).set_index("Metodo")
df_prt
```

Out[97]:

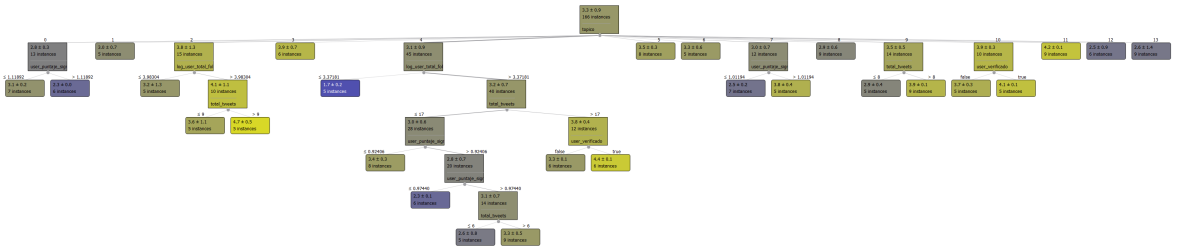
Metodo	MSE	RMSE	MAE	R2
Redes Neuronales	3332334.231	1825.468	873.400	-0.277
Random Forest	2586659.886	1608.310	838.344	0.009
SVM	3081994.146	1755.561	854.727	-0.181
Tree	2844566.569	1686.584	967.063	-0.090
kNN	2526030.463	1579.349	751.189	0.032

Al igual que en el caso anterior, se observa a través de R2 que los modelos son catastroficos.

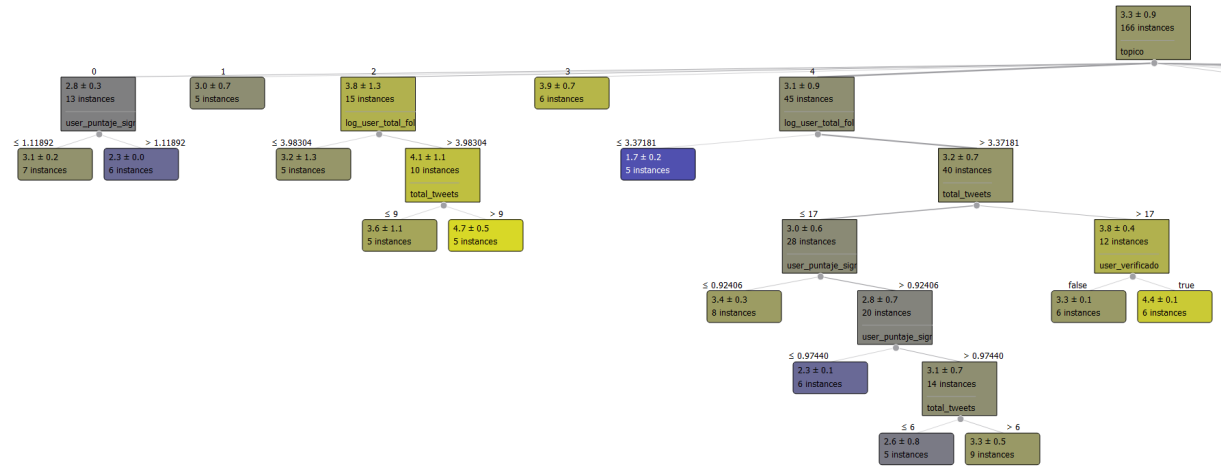
Logaritmo Total RT

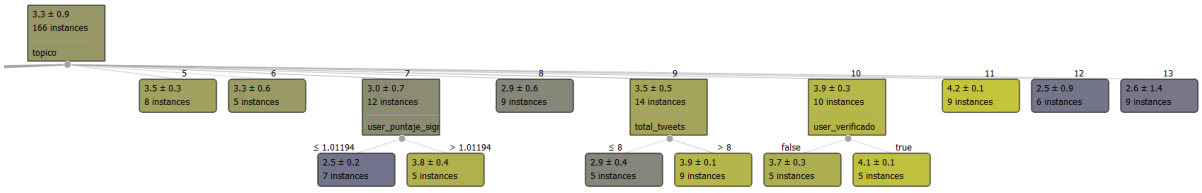
Tal como se observó en los gráficos BI, los totales RT poseen una distribución que dificulta el análisis con valores muy superiores, por lo que se busca normalizar esto a través del logaritmo.

Árbol de decisión



Se observan claras mejorías en comparación a los dos casos anteriores, por lo que se analizará en mayor detención.





Se observa que el atributo que más define el éxito de un hilo es la temática. Los atributos en el siguiente nivel del arbol son:

- Puntaje Sigmoídeo (Tópicos: 0, 7)
- Log_Followers (Tópicos: 2, 4)
- Total Tweets (Tópicos: 9)
- User Verificado (Tópicos: 10)

Para los tópicos con más niveles (2 y 4) se observan que ambos dividen el siguiente nivel por total de tweets.

De este arbol se observa que en todos los casos, la calidad del usuario es el valor más importante para el éxito del hilo luego del tópico. Sólo en el tópico 9 es más importante el largo que el usuario en sí.

In [98]:
```
columnas = ["Metodo", "MSE", "RMSE", "MAE", "R2"]
log_rt_neural = ["Redes Neuronales", 1.029, 1.014, 0.792, -0.195]
log_rt_forest = ["Random Forest"   , 0.624, 0.790, 0.575, 0.275]
log_rt_svm    = ["SVM"             , 0.577, 0.759, 0.575, 0.330]
log_rt_tree   = ["Tree"            , 0.746, 0.864, 0.638, 0.134]
log_rt_knn    = ["kNN"             , 0.686, 0.828, 0.659, 0.203]
log_rt_lista_metodos = [log_rt_neural, log_rt_forest, log_rt_svm, log_rt_tree, l
df_lrt = pd.DataFrame(log_rt_lista_metodos, columns = columnas).set_index("Metod
df_lrt
```

Out[98]:

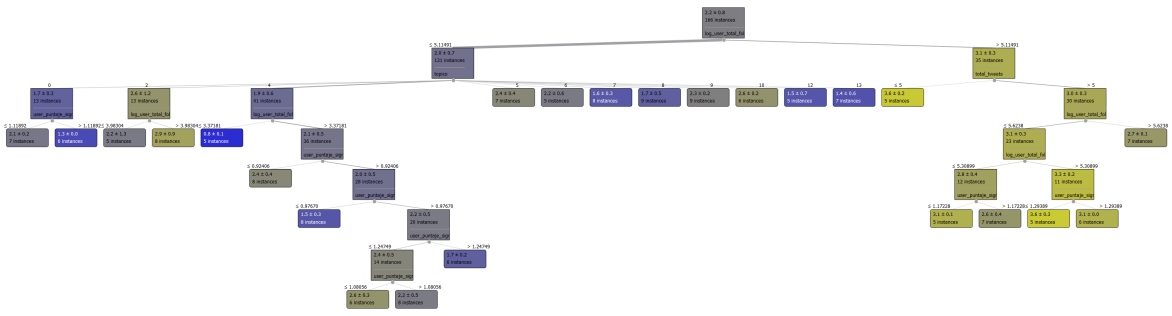
Metodo	MSE	RMSE	MAE	R2
Redes Neuronales	1.029	1.014	0.792	-0.195
Random Forest	0.624	0.790	0.575	0.275
SVM	0.577	0.759	0.575	0.330
Tree	0.746	0.864	0.638	0.134
kNN	0.686	0.828	0.659	0.203

Se observa a través de R2 que la calidad de la mayoría de los modelos subió en una cantidad no despreciable en comparación a los 2 apartados anteriores, pero aún son considerados de mala calidad.

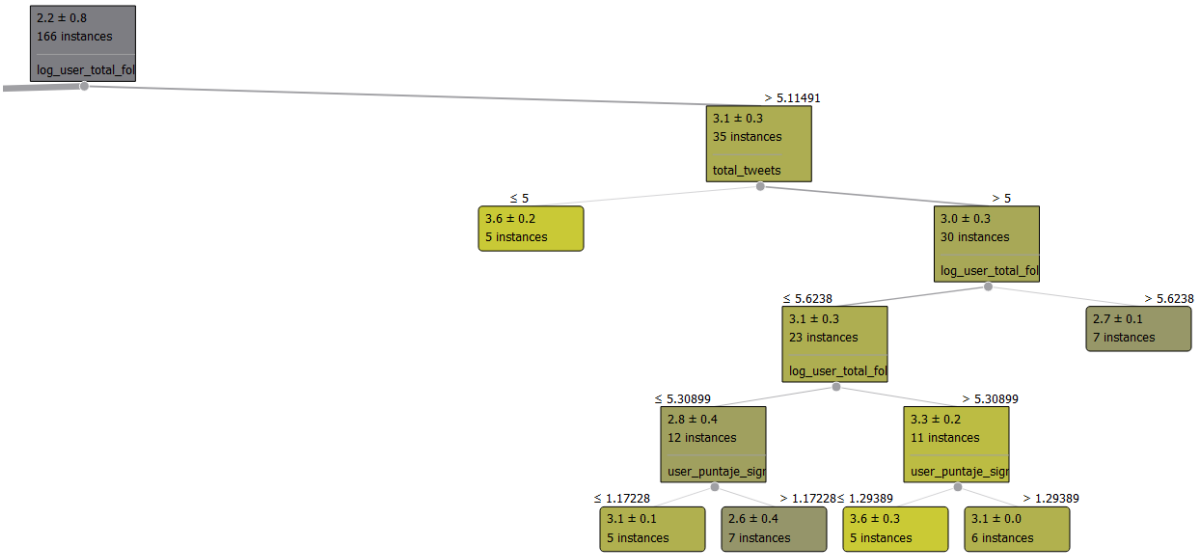
Logaritmo Promedio RT

Finalmente se busca analizar el efecto de normalizar el promedio de RT obtenidos por el hilo.

Árbol de decisión



Se observa que en este caso, el arbol de decisión inicia dividiendo por total de seguidores, lo que deja una rama buena y otra mala.



Se observa que la rama luego de dividirse por total de tweets, sigue dividiendose por caracteristicas del usuario. También se observa que los valores más altos de la variable objetivo son cercanos a 3.6, mientras que en el apartado anterior estos eran superiores.

In [99]:
```
columnas = ["Metodo", "MSE", "RMSE", "MAE", "R2"]
log_prt_neural = ["Redes Neuronales", 0.668, 0.817, 0.636, 0.135]
log_prt_forest = ["Random Forest"   , 0.615, 0.784, 0.609, 0.203]
log_prt_svm    = ["SVM"             , 0.504, 0.710, 0.571, 0.347]
log_prt_tree   = ["Tree"            , 0.771, 0.878, 0.674, 0.000]
log_prt_knn    = ["kNN"             , 0.620, 0.787, 0.644, 0.197]
log_prt_lista_metodos = [log_prt_neural, log_prt_forest, log_prt_svm, log_prt_tr
df_lprt = pd.DataFrame(log_prt_lista_metodos, columns = columnas).set_index("Met
df_lprt
```

Out[99]:

Metodo	MSE	RMSE	MAE	R2
Redes Neuronales	0.668	0.817	0.636	0.135
Random Forest	0.615	0.784	0.609	0.203
SVM	0.504	0.710	0.571	0.347
Tree	0.771	0.878	0.674	0.000
kNN	0.620	0.787	0.644	0.197

Se observa que los valores R2 son similares al caso anterior, pero el árbol presenta un resultado peor.

Análisis por modelos.

Luego de analizar los errores de los modelos en conjunto por variables objetivo, se presentan los errores de predicción según modelos

Redes neuronales

Se observa que las redes neuronales presentan pésimos resultados de predicción R2

In [106]:
```
all_lista = []
columnas = ["Objetivo", "MSE", "RMSE", "MAE", "R2"]
rt_neural[0] = "Total_RT_neural"
prt_neural[0] = "Prom_RT_neural"
log_rt_neural[0] = "Log_RT_neural"
log_prt_neural[0] = "Log_Prom_RT_neural"
lista_neural = [rt_neural, prt_neural, log_rt_neural, log_prt_neural]
all_lista.extend(lista_neural)
df_neural = pd.DataFrame(lista_neural, columns = columnas).set_index("Objetivo")
df_neural
```

Out[106]:

Objetivo	MSE	RMSE	MAE	R2
Total_RT_neural	5.512032e+08	23477.718	10731.293	-0.263
Prom_RT_neural	3.332334e+06	1825.468	873.400	-0.277
Log_RT_neural	1.029000e+00	1.014	0.792	-0.195
Log_Prom_RT_neural	6.680000e-01	0.817	0.636	0.135

Random Forest

In [107]:
```
columnas = ["Objetivo", "MSE", "RMSE", "MAE", "R2"]
rt_forest[0] = "Total_RT_forest"
prt_forest[0] = "Prom_RT_forest"
log_rt_forest[0] = "Log_RT_forest"
log_prt_forest[0] = "Log_Prom_RT_forest"
lista_forest = [rt_forest, prt_forest, log_rt_forest, log_prt_forest]
all_lista.extend(lista_forest)
df_forest = pd.DataFrame(lista_forest, columns = columnas).set_index("Objetivo")
df_forest
```

Out[107]:

Objetivo	MSE	RMSE	MAE	R2
Total_RT_forest	4.898477e+08	22132.504	10729.242	-0.122
Prom_RT_forest	2.586660e+06	1608.310	838.344	0.009
Log_RT_forest	6.240000e-01	0.790	0.575	0.275
Log_Prom_RT_forest	6.150000e-01	0.784	0.609	0.203

SVM

In [108]:
```
columnas = ["Objetivo", "MSE", "RMSE", "MAE", "R2"]
rt_svm[0] = "Total_RT_svm"
prt_svm[0] = "Prom_RT_svm"
log_rt_svm[0] = "Log_RT_svm"
log_prt_svm[0] = "Log_Prom_RT_svm"
lista_svm = [rt_svm, prt_svm, log_rt_svm, log_prt_svm]
all_lista.extend(lista_svm)
df_svm = pd.DataFrame(lista_svm, columns = columnas).set_index("Objetivo")
df_svm
```

Out[108]:

Objetivo	MSE	RMSE	MAE	R2
Total_RT_svm	4.924639e+08	22191.527	10100.529	-0.128
Prom_RT_svm	3.081994e+06	1755.561	854.727	-0.181
Log_RT_svm	5.770000e-01	0.759	0.575	0.330
Log_Prom_RT_svm	5.040000e-01	0.710	0.571	0.347

Árbol de decisión

In [109]:
```
columnas = ["Objetivo", "MSE", "RMSE", "MAE", "R2"]
rt_tree[0] = "Total_RT_tree"
prt_tree[0] = "Prom_RT_tree"
log_rt_tree[0] = "Log_RT_tree"
log_prt_tree[0] = "Log_Prom_RT_tree"
lista_tree = [rt_tree, prt_tree, log_rt_tree, log_prt_tree]
all_lista.extend(lista_tree)
df_tree = pd.DataFrame(lista_tree, columns = columnas).set_index("Objetivo")
df_tree
```

Out[109]:

Objetivo	MSE	RMSE	MAE	R2
Total_RT_tree	4.249308e+08	20613.850	11182.439	0.026
Prom_RT_tree	2.844567e+06	1686.584	967.063	-0.090
Log_RT_tree	7.460000e-01	0.864	0.638	0.134
Log_Prom_RT_tree	7.710000e-01	0.878	0.674	0.000

kNN

In [110]:
```
columnas = ["Objetivo", "MSE", "RMSE", "MAE", "R2"]
rt_knn[0] = "Total_RT_kNN"
prt_knn[0] = "Prom_RT_kNN"
log_rt_knn[0] = "Log_RT_kNN"
log_prt_knn[0] = "Log_Prom_RT_kNN"
lista_knn = [rt_knn, prt_knn, log_rt_knn, log_prt_knn]
all_lista.extend(lista_knn)
df_knn = pd.DataFrame(lista_knn, columns = columnas).set_index("Objetivo")
df_knn
```

Out[110]:

Objetivo	MSE	RMSE	MAE	R2
Total_RT_kNN	4.590459e+08	21425.357	10219.362	-0.052
Prom_RT_kNN	2.526030e+06	1579.349	751.189	0.032
Log_RT_kNN	6.860000e-01	0.828	0.659	0.203
Log_Prom_RT_kNN	6.200000e-01	0.787	0.644	0.197

Matriz de resultados

Se presenta una matriz resumen de todos los errores y modelos entrenados, las filas están agrupadas por Variable Objetivo

In [127]:
```
columnas = ["Metodo", "MSE", "RMSE", "MAE", "R2"]
df_all = pd.DataFrame(all_lista, columns = columnas).transpose().sort_values(by =
df_all.columns = df_all.iloc[0]
df_all = df_all.reindex(df_all.index.drop("Metodo")).transpose()
df_all
```

Out[127]:

Metodo	MSE	RMSE	MAE	R2
Log_Prom_RT_forest	0.615	0.784	0.609	0.203
Log_Prom_RT_kNN	0.62	0.787	0.644	0.197
Log_Prom_RT_neural	0.668	0.817	0.636	0.135
Log_Prom_RT_svm	0.504	0.71	0.571	0.347
Log_Prom_RT_tree	0.771	0.878	0.674	0
Log_RT_forest	0.624	0.79	0.575	0.275
Log_RT_kNN	0.686	0.828	0.659	0.203
Log_RT_neural	1.029	1.014	0.792	-0.195
Log_RT_svm	0.577	0.759	0.575	0.33
Log_RT_tree	0.746	0.864	0.638	0.134
Prom_RT_forest	2.58666e+06	1608.31	838.344	0.009
Prom_RT_kNN	2.52603e+06	1579.35	751.189	0.032
Prom_RT_neural	3.33233e+06	1825.47	873.4	-0.277
Prom_RT_svm	3.08199e+06	1755.56	854.727	-0.181
Prom_RT_tree	2.84457e+06	1686.58	967.063	-0.09
Total_RT_forest	4.89848e+08	22132.5	10729.2	-0.122
Total_RT_kNN	4.59046e+08	21425.4	10219.4	-0.052
Total_RT_neural	5.51203e+08	23477.7	10731.3	-0.263
Total_RT_svm	4.92464e+08	22191.5	10100.5	-0.128
Total_RT_tree	4.24931e+08	20613.8	11182.4	0.026

Ordenadas por valor R2

In [128]:
```
df_all.sort_values(by = "R2")
```

Out[128]:

Metodo	MSE	RMSE	MAE	R2
Prom_RT_neural	3.33233e+06	1825.47	873.4	-0.277
Total_RT_neural	5.51203e+08	23477.7	10731.3	-0.263
Log_RT_neural	1.029	1.014	0.792	-0.195
Prom_RT_svm	3.08199e+06	1755.56	854.727	-0.181
Total_RT_svm	4.92464e+08	22191.5	10100.5	-0.128
Total_RT_forest	4.89848e+08	22132.5	10729.2	-0.122
Prom_RT_tree	2.84457e+06	1686.58	967.063	-0.09
Total_RT_kNN	4.59046e+08	21425.4	10219.4	-0.052
Log_Prom_RT_tree	0.771	0.878	0.674	0
Prom_RT_forest	2.58666e+06	1608.31	838.344	0.009
Total_RT_tree	4.24931e+08	20613.8	11182.4	0.026
Prom_RT_kNN	2.52603e+06	1579.35	751.189	0.032
Log_RT_tree	0.746	0.864	0.638	0.134
Log_Prom_RT_neural	0.668	0.817	0.636	0.135
Log_Prom_RT_kNN	0.62	0.787	0.644	0.197
Log_RT_kNN	0.686	0.828	0.659	0.203
Log_Prom_RT_forest	0.615	0.784	0.609	0.203
Log_RT_forest	0.624	0.79	0.575	0.275
Log_RT_svm	0.577	0.759	0.575	0.33
Log_Prom_RT_svm	0.504	0.71	0.571	0.347

In [ ]:
```
```