PASOS PARA LA REALIZACIÓN DEL PROYECTO

Archivo que resume toda la estrategia de solución planteada, los resultados obtenidos y las conclusiones a las que se logró llegar.

1. Base de Datos:

Se utilizó la base de datos utilizada para el proyecto 2.

2. Modelos de aprendizaje:

Para el análisis de la base de datos: Appliances Energy Prediction, que trata sobre la predicción del consumo de energía era importante seleccionar algoritmos que manejaran bien problemas de regresión, por esto, elegimos Random Forest Regressor (RFR) y Support Vector Regressor (SVR) quienes también tienen capacidades para capturar tanto relaciones simples como complejas entre las características.
Adicionalmente, se realizó una búsqueda de hiperparámetros adecuados para cada algoritmo, para esto, se utilizó GridSearchCV.

Para RFR:

- ***n_estimators*:** Número de árboles en el bosque: Se probó con valores comunes como 100, 200 y 500.
- ***max_depth*:** Profundidad máxima de cada árbol: Se probó con valores 10, 20 y 30, ya que suelen ser adecuados para evitar sobreajuste sin hacer el modelo demasiado simple.
- ***min_samples_split*:** Número mínimo de muestras necesarias para dividir un nodo: Valores comunes como 2, 5, y 10. Para reducir el sobreajuste.
- ***min_samples_leaf*:** Número mínimo de muestras que debe tener una hoja del árbol: usamos valores comunes como 1, 2 y 4.

Para SVR:

- ***kernel***: El kernel determina cómo se proyectan los datos en un espacio de alta dimensión. utilizamos: **linear**(Si los datos son linealmente separables)**, poly**(Si se requieren transformaciones polinómicas) y **rbf**(Se usa para datos no lineales)

- ***C*:** Parámetro de regularización. Se probaron valores como 0.1, 1, 10, 100, que cubren un rango de error amplio para asegurar que el modelo se ajuste adecuadamente.
- ***gamma*:** Controla la forma del kernel para los modelos **poly y rbf** o las no lineales. Se usaron valores estándar.

Búsqueda de parámetros:

```
grid_svr = GridSearchCV(estimator=svr_modelo, param_grid=parametros_svr, cv=5,
scoring='r2', n_jobs=-1)
grid_svr.fit(X_train, y_train)
```

Con este código buscamos los hiperparámetros para cada modelo.

3. Realizar las curvas de aprendizaje:

Primero se creó una función para calcular métricas como R^2, MAE, MSE y RMSE que luego se utilizarían para crear las curvas de aprendizaje.

Visualización: Se creó un gráfico que permite comparar los valores reales y los predichos de cada modelo.

Para SVR:
Creamos un diagrama de dispersión donde: ***y_test*** son los valores reales (eje X) y ***svr_pred*** son los valores predichos por el modelo SVR (eje Y). Aquí cada punto representa una predicción del modelo para una muestra. Luego se dibuja una línea roja (***'--r'***) que representa la diagonal ideal donde valores reales = valores predichos. Esta línea ayuda a identificar cuánto se desvían las predicciones del modelo respecto a los valores reales.

Para RFR:
Creamos una diagrama para graficar las desviaciones estándar
plt.plot: grafica las curvas

- Curva de entrenamiento: Tamaño del conjunto de entrenamiento (***train_sizes***) vs error promedio en entrenamiento (***train_mean***).

- Curva de validación: Tamaño del conjunto de entrenamiento vs error promedio en validación (***val_mean***).

4. Realizar una evaluación diagnóstica:

grid_svr.predict(X_train): Utiliza el modelo entrenado (con los mejores hiperparámetros encontrados mediante GridSearchCV) para predecir los valores en el conjunto de entrenamiento (***X_train***).

grid_svr.predict(X_test): Realiza predicciones sobre el conjunto de prueba (`X_test`).

El objetivo es comparar cómo se desempeña el modelo en ambos conjuntos:

- **Entrenamiento**: Indica qué tan bien el modelo ajusta los datos con los que fue entrenado.
- **Prueba**: Indica qué tan bien el modelo generaliza a datos nuevos, es decir, qué tan buenas son sus predicciones para datos que no vio durante el entrenamiento.

Luego, se calculan las métricas para el conjunto de entrenamiento y el de prueba para identificar si hay grandes discrepancias entre el desempeño en ambos conjuntos

r2_score:

- Calcula el coeficiente de determinación (R2R^2R2), que mide qué tan bien el modelo explica la variabilidad de los datos.
- Valores cercanos a 1 indican un buen ajuste; valores bajos indican que el modelo no está capturando bien la relación entre las características y la variable objetivo.

Se calcula la diferencia del r2 entre los dos conjuntos:

- Si esta diferencia es grande, puede ser un indicador de **overfitting**.
- Si ambos valores son bajos, puede ser un indicador de **bias**.

Se hace el diagnóstico con "***ifs"***, analizando los valores r2 entre los dos conjuntos.

5. Compara el desempeño de los dos modelos:

Lo primero que se debe considerar es que ya se hayan guardado las predicciones para ambos modelos. Nosotros guardamos ambas en un datasheet:

grid_rf (Random Forest) y ***grid_svr*** (SVR)
X_test es el conjunto de datos de prueba.
rf_pred y ***svr_pred*** son las predicciones respectivas de ambos modelos.

Estrategia de promediado simple

Creamos un ensamble que combina las predicciones de ambos modelos tomando el promedio simple de las predicciones y lo promediamos para:

- Ayuda a reducir el error individual de cada modelo.
- Si ambos modelos tienen fortalezas y debilidades en diferentes partes del conjunto de datos, el promedio puede equilibrarlas.

Función para evaluar el desempeño del modelo

Creamos una función para calcular las siguiente métricas:

- **MAE**: Error promedio absoluto (qué tan lejos están las predicciones de los valores reales, en promedio).
- **MSE**: Penaliza errores grandes, ya que los eleva al cuadrado.
- **RMSE**: Similar a MSE, pero está en las mismas unidades que la variable objetivo.
- **R^2**: Indica qué proporción de la variabilidad de los datos es explicada por el modelo (1 = ajuste perfecto, 0 = sin relación).

Evaluación de los modelos:

- Para cada modelo (**RF**, **SVR**, y el ensemble), se evalúa el desempeño usando las predicciones y los valores reales (***y_test***).
- Las métricas calculadas son almacenadas en variables específicas para cada modelo.

Creación de un DataFrame con los resultados

- Se crea un diccionario con los modelos y sus métricas:
    - ***Model***: Los nombres de los modelos evaluados.
    - ***R2 Square***: Valores de R^2 para cada modelo.
    - ***MAE***, ***MSE***, y ***RMSE***: Métricas de error.
- El diccionario se convierte en un DataFrame con **pandas** para facilitar la visualización y manipulación.

Establecer el índice y graficar los resultados

- ***set_index***: Convierte la columna "Model" en el índice del DataFrame para organizar los datos.
- ***plot(kind='barh')***: Crea un gráfico de barras horizontal para comparar los valores de R^2 entre los modelos.
- ***color***: Se usa una paleta de colores personalizada para distinguir los modelos.
- **figsize**: Ajusta el tamaño del gráfico (12x8 pulgadas).

Mostrar los resultados en formato tabular

Imprimimos la tabla completa con las métricas de cada modelo para un análisis más detallado.

6. Resultados:

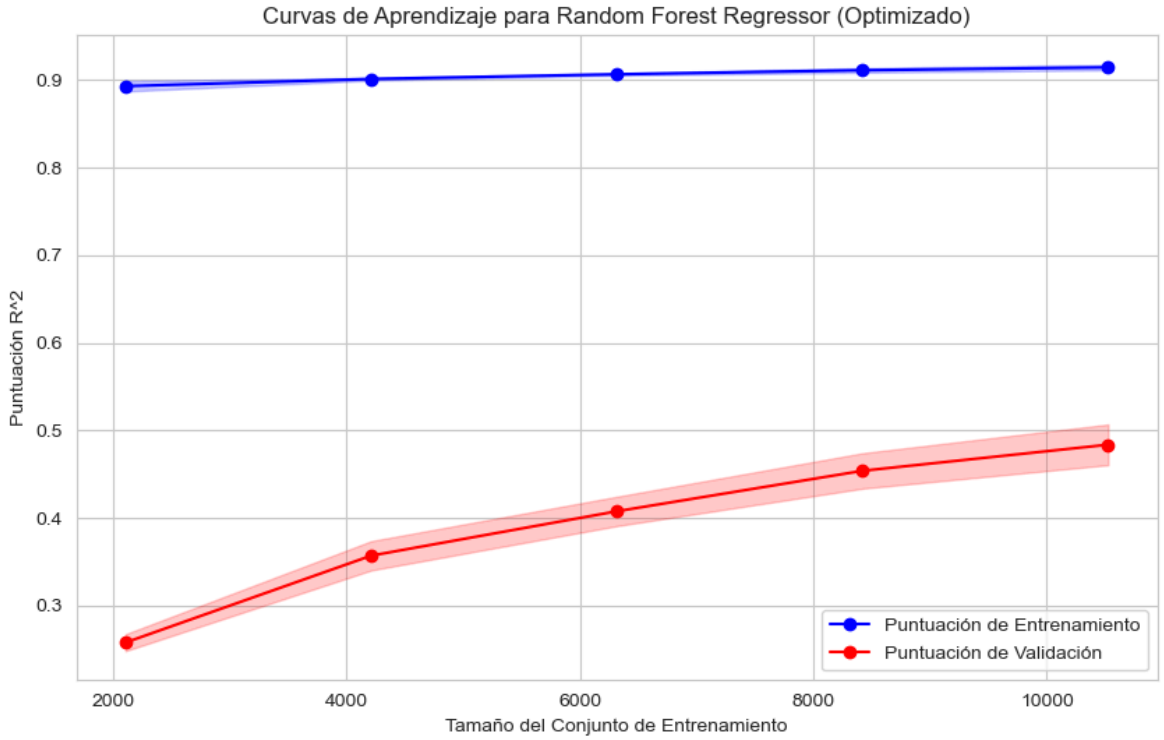
Random Forest Regressor:

En primer lugar, él tamaño de la prueba fue el siguiente usados para este modelo fueron los siguientes:

- Tamaño del conjunto de entrenamiento: 15788
- Tamaño del conjunto de prueba: 3947

Se utilizaron datos de 27 columnas distintas de características comparándolas con la variable "Appliances" que es la cantidad de energía gastada en Wh.

Los resultados del método Random Forest mostraron no ser tan efectiva con los hiperparametros y el tamaño de datos que se usaron. Esto se puede evidenciar en la siguiente gráfica:



A pesar que en el código se estén usando los hiperparametros hallados, los cuales son los siguientes:

- max_depth: 20
- max_features: sqrt
- min_samples_leaf: 1
- min_samples_split: 2
- n_estimators: 100

Los resultados del random forest regressor fueron los siguientes>

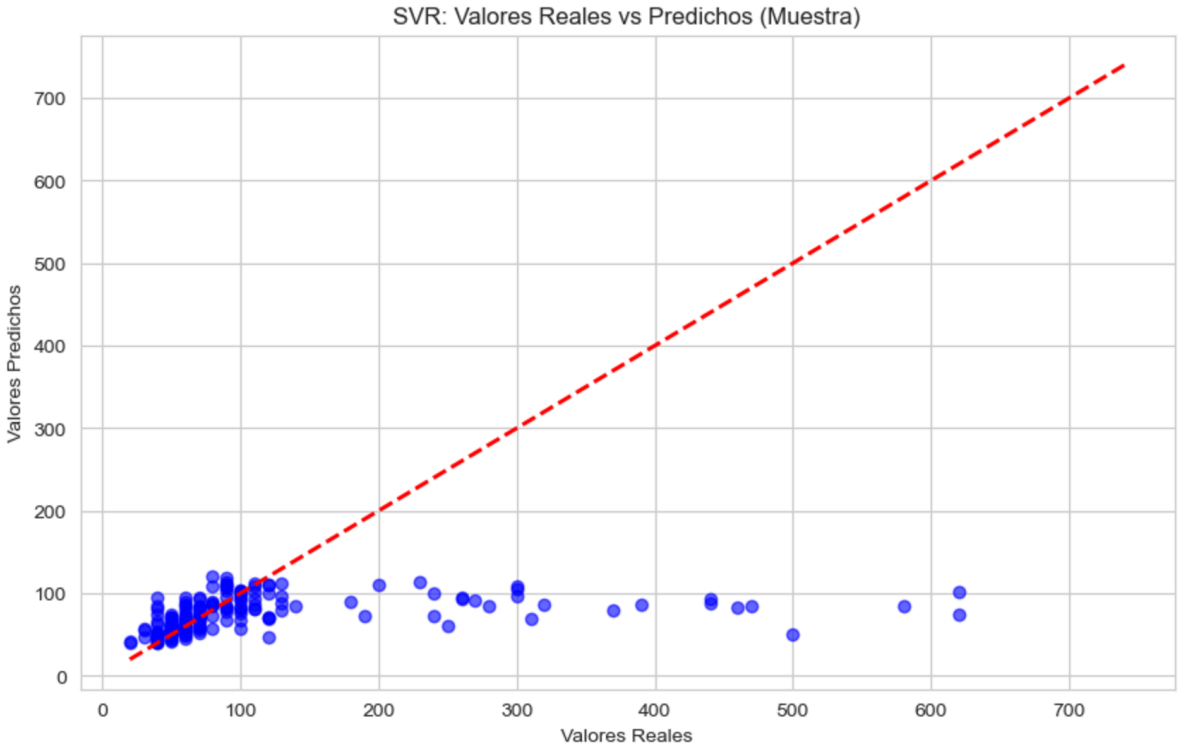
- Desempeño en el conjunto de entrenamiento:
    - MAE: 12.71
    - MSE: 712.80
    - RMSE: 26.70
    - R^2: 0.93

- Desempeño en el conjunto de prueba:
    - MAE: 32.89
    - MSE: 4678.15
    - RMSE: 68.40
    - R^2: 0.53

Y la diferencia en R^2 (Entrenamiento - Prueba) fue de 0.40.

Support Vector Regressor:

Desempeño de SVR:
MAE: 44.75
MSE: 10361.70
RMSE: 101.79
R^2: 0.03



La gráfica es un diagrama de dispersión, en donde la línea roja representa la diagonal ideal donde valores reales = valores predichos. Aquí podemos observar que las predicciones no tienen una desviación muy grande en los primeros valores, pero aproximadamente desde el 200 la desviación va aumentando en conforme aumentan los valores.

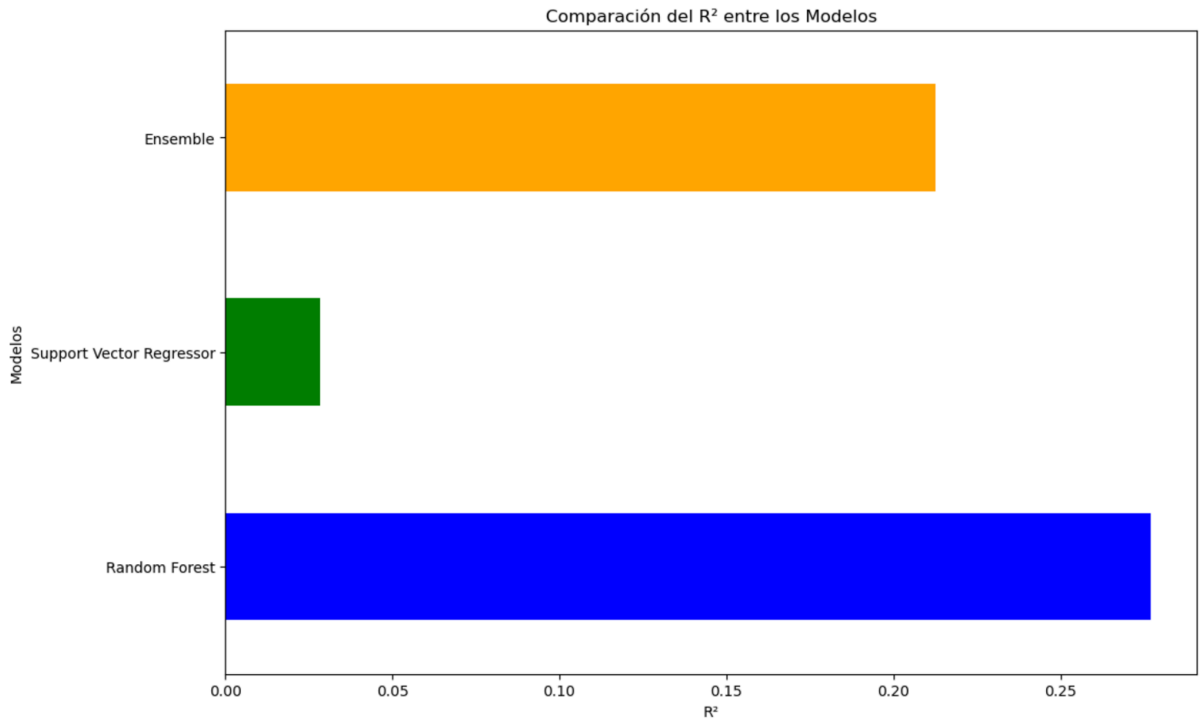
- Desempeño en el conjunto de entrenamiento:
    - MAE: 39.25
    - MSE: 9221.00
    - RMSE: 96.03
    - R^2: 0.05

- Desempeño en el conjunto de prueba:
    - MAE: 44.75
    - MSE: 10361.70
    - RMSE: 101.79
    - R^2: 0.03

Diferencia en R^2 (Entrenamiento - Prueba): 0.02

Los valores de R^2 mostraron ser bajos, lo que indica posibles bias. Esto sugiere que el modelo es incapaz de capturar relaciones significativas entre las variables independientes y la variable objetivo.

<u>Para la Comparación:</u>



La gráfica compara los valores de R2 de los dos modelos.

Model	R2 Square	MAE	MSE	RMSE
Random Forest	0.276934	47.558293	7709.722058	87.805023
Support Vector Regressor	0.028215	44.746752	10361.698715	101.792430
Ensemble	0.212521	44.120931	8396.533326	91.632600

Esta tabla muestra las métricas en los dos modelos.

[ ]:

7. Conclusiones:

- De la curva de aprendizaje del random forest regressor, podemos concluir que hay un overfitting en los datos porque hay una diferencia de rendimiento en el entrenamiento y en la prueba muy grande, lo que quiere decir que el modelo está siendo sobreentrenado y es muy específico para los datos que tenemos.

- Esto también lo podemos verificar con los resultados del R^2. Que cambian mucho del entrenamiento a la prueba (un 0.4). Además, el MAE y el RMSE son

relativamente bajos en el entrenamiento, sugiriendo un buen ajuste en los datos de entrenamiento. Pero luego de la prueba, incrementan bastante, sugiriendo que en la prueba se cometieron más errores y que como se propuso el modelo inicialmente no está siendo tan efectivo.

- SVR (Support Vector Regressor) es un modelo poderoso para regresión, pero su desempeño depende en gran medida de una correcta selección de hiperparámetros, por ende el modelo SVR no es el más adecuado para estos datos

- En la gráfica de dispersión, se observa que las predicciones están razonablemente cerca de los valores reales en rangos bajos, pero la dispersión aumenta significativamente a partir de valores mayores a 200. Lo que indica que el modelo SVR tiene dificultades para predecir valores extremos o atípicos. Es posible que no esté capturando bien la complejidad de los datos.

- El bajo valor de R2 tanto en entrenamiento (0.05) como en prueba (0.03) indica que el modelo Support Vector Regressor no está capturando de manera adecuada las relaciones entre las variables de entrada y la variable objetivo.

- La diferencia pequeña entre R2R^2R2 de entrenamiento y prueba (0.02) del modelo SVR muestra que no hay overfitting

- Random Forest mostró ser el mejor modelo debido a su mayor R2 y menores errores. Parece ser más adecuado para los datos.

- El ensemble no supera al Random Forest individualmente, aunque logra un MAE más bajo. Esto sugiere que el SVR está afectando negativamente la combinación.