Informe final Datathon AI4ES

Categoría: Futuras promesas

Participante: *Raúl Coterillo*

Email contacto: **raul.coterillo@mailbox.org**

Bilbao, 23/11/2023

Contenido

1 Enfoque

1.1 Reto abordado y principales desafíos

Se ha tratado el reto B del datathon, que consiste en la predicción del consumo energético de los dos edificios de oficinas de CTIC, localizadas en Asturias (Gijón y Peón). Se dispone de los datos de consumo horario de ambos edificios durante 22 meses desde 2021 a 2023, con el objetivo de predecir el consumo energético de los dos meses siguientes.

Aunque el hecho de que se trate de una oficina en contexto laboral facilita el problema, las fluctuaciones de consumo eléctrico en edificios individuales siguen siendo bastante altas y requiere de métodos avanzados para su estimación.

1.2 Gestión de datos y pre-procesamiento

Todos los datos se han tratado con la librería *pandas* de Python antes de ser utilizados para entrenar los modelos. Aparte de los datos proporcionados, se han tenido en cuenta variables exógenas adicionales que se consideraron de utilidad para la predicción. En concreto:

- **Temperaturas máximas y mínimas:** Se han obtenido las temperaturas máximas y mínimas de 2021 a 2023 de la estación meteorológica del campus de Gijón, ya que era la más cercana a ambas instalaciones. Esto se hizo parseando el HTML de la página web "datosclima.es" (enlace), puesto que no se encontró otro tipo de datasets. Los registros faltantes se interpolan con el valor medio de los datos más próximos.
- **Festivos nacionales, autonómicos y locales:** Se ha descargado un dataset (enlace) del portal de datos públicos "datos.gob.es" con los festivos de todos los municipios de Asturias de 2021 a 2023. De esta manera, el modelo de cada centro tiene en cuenta los festivos locales del mismo.

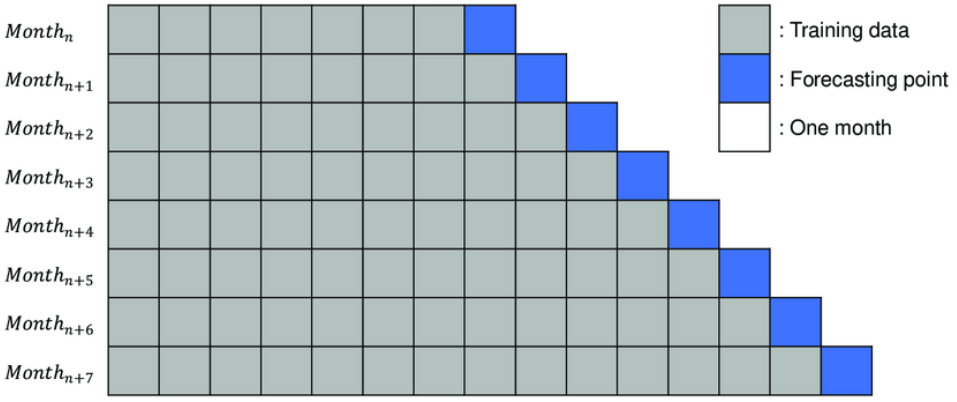
El hecho de que los datos de consumo tengan un carácter temporal hace que sea bastante sencillo añadir este tipo de variables. En total, las variables consideradas han sido:

- Hora y dia de la semana
- Temperatura máximas y mínimas
- Horario laboral (estimado de 7:00 a 19:00)
- Día laboral, festivo y/o fin de semana
- Consumos mínimos, medios y máximos diarios

En los datos también se observaron dos puntos de consumo nulo, que se han supuesto corresponden con anomalías y por tanto se han imputado con el valor medio de los puntos adyacentes para evitar distorsionar estadísticas como el MAPE.

1.3 Estrategia de particionado de datos

En la selección modelo se ha utilizado la estrategia *ExpandingWindowSplitter* de la librería *sktime*, dividiendo los datos en tres particiones distintas para la comparativa. El diagrama muestra la estrategia de particionado tomando un horizonte de predicción de un mes:



Para la optimización de hiper parámetros de ha utilizado un K-fold con 5 particiones, ya que se ha considerado suficiente dadas las características del modelo elegido.

2 Desarrollo

2.1 Descripción del algoritmo

Finalmente, se ha utilizado el módulo de series temporales de la librería PyCaret, que implementa un gran número de modelos de regresión de series temporales. Esta librería permite comprobar el funcionamiento de un gran número de modelos en el dataset objetivo de manera rápida, facilitando una elección adecuada del algoritmo final. Estos son los resultados de dicha comparativa para el dataset de centro de Gijón:

	Model	MASE	RMSSE	MAE	RMSE	MAPE	SMAPE	R2	TT (Sec)
lightgbm_cds_dt	Light Gradient Boosting w/ Cond. Deseasonalize & Detrending	0.2992	0.3130	4.5017	6.7882	0.1271	0.1154	0.7078	35.1800
rf_cds_dt	Random Forest w/ Cond. Deseasonalize & Detrending	0.3024	0.3118	4.5479	6.7581	0.1284	0.1172	0.7135	55.0000
et_cds_dt	Extra Trees w/ Cond. Deseasonalize & Detrending	0.3034	0.3175	4.5621	6.8822	0.1272	0.1161	0.6977	54.6133
gbr_cds_dt	Gradient Boosting w/ Cond. Deseasonalize & Detrending	0.3290	0.3229	4.9473	6.9998	0.1456	0.1313	0.6859	33.3767
dt_cds_dt	Decision Tree w/ Cond. Deseasonalize & Detrending	0.3590	0.3888	5.3939	8.4204	0.1494	0.1371	0.5785	31.5400
huber_cds_dt	Huber w/ Cond. Deseasonalize & Detrending	0.4785	0.4128	7.1952	8.9446	0.2216	0.2084	0.5351	32.4667
br_cds_dt	Bayesian Ridge w/ Cond. Deseasonalize & Detrending	0.5019	0.4061	7.5550	8.8067	0.2542	0.2381	0.5295	32.0433
ridge_cds_dt	Ridge w/ Cond. Deseasonalize & Detrending	0.5020	0.4062	7.5564	8.8082	0.2543	0.2382	0.5293	31.8133
lr_cds_dt	Linear w/ Cond. Deseasonalize & Detrending	0.5021	0.4062	7.5571	8.8089	0.2543	0.2383	0.5292	31.3733
ada_cds_dt	AdaBoost w/ Cond. Deseasonalize & Detrending	0.5402	0.4561	8.1591	9.9137	0.2811	0.2319	0.3381	34.9833
knn_cds_dt	K Neighbors w/ Cond. Deseasonalize & Detrending	0.6262	0.6543	9.4056	14.1825	0.2817	0.2414	-0.2173	36.3500
theta	Theta Forecaster	0.6404	0.6909	9.6246	14.9644	0.2204	0.2626	-0.2718	0.1467
naive	Naive Forecaster	0.6411	0.6926	9.6342	15.0013	0.2201	0.2629	-0.2791	0.0767
snaive	Seasonal Naive Forecaster	0.7098	0.6904	10.6756	14.9558	0.2832	0.2937	-0.2725	0.0900
en_cds_dt	Elastic Net w/ Cond. Deseasonalize & Detrending	0.7516	0.6049	11.3149	13.1126	0.3460	0.3191	0.0133	31.3467
ets	ETS	0.7523	0.7371	11.2938	15.9525	0.2840	0.3221	-0.4635	0.5867
llar_cds_dt	Lasso Least Angular Regressor w/ Cond. Deseasonalize & Detrending	0.7749	0.6218	11.6653	13.4782	0.3565	0.3286	-0.0418	31.9133
lasso_cds_dt	Lasso w/ Cond. Deseasonalize & Detrending	0.7749	0.6218	11.6653	13.4781	0.3565	0.3286	-0.0418	31.9867
stlf	STLF	0.7764	0.6499	11.6958	14.0862	0.3538	0.3270	-0.1443	0.1333
polytrend	Polynomial Trend Forecaster	0.7784	0.6228	11.7182	13.4987	0.3589	0.3301	-0.0443	0.0433
omp_cds_dt	Orthogonal Matching Pursuit w/ Cond. Deseasonalize & Detrending	0.7829	0.6241	11.7844	13.5294	0.3613	0.3319	-0.0530	32.1500
exp_smooth	Exponential Smoothing	0.7965	0.8179	11.9822	17.7299	0.2797	0.3646	-0.7789	1.4733
grand_means	Grand Means Forecaster	0.7978	0.6245	12.0051	13.5375	0.3729	0.3383	-0.0523	0.0633
croston	Croston	0.8197	0.6594	12.3531	14.2982	0.3904	0.3451	-0.1991	0.0600

En base a esta comparación, se escogió el modelo 'lightgbm_cds_dt' para la predicción final, que se basa en un modelo de regresión LightGBM (enlace) con modificaciones de tendencias y estacionalidad.

Sin embargo, debido a fallos de código (bugs) de la librería PyCaret, ha sido imposible exportar el modelo para realizar las predicciones del reto. Es por esto que se ha utilizado la librería LightGBM base y se ha entrenado un modelo de regresión que utiliza las variables exógenas de cada hora para predecir el valor de consumo.

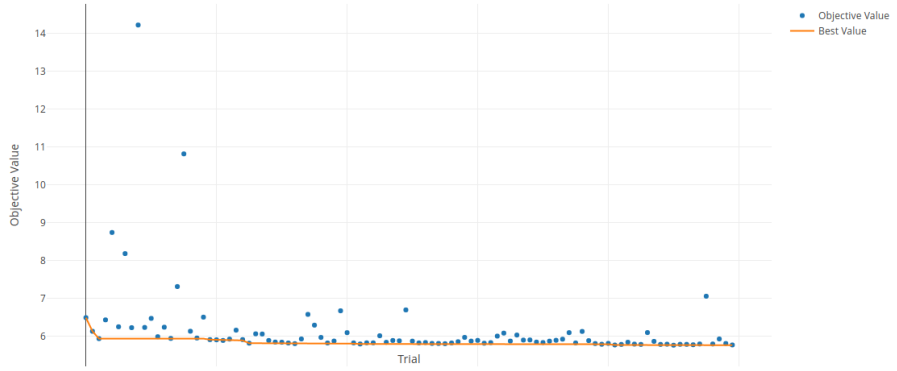
Cabe destacar que se realizaron pruebas con otros algoritmos más orientados a la predicción del consumo, como el presentado en el artículo "Probabilistic Load Forecasting based on Adaptive Online Learning" (enlace), pero no se llegó a alcanzar resultados satisfactorios con el horizonte de predicción a dos meses.

2.2 Entrenamiento

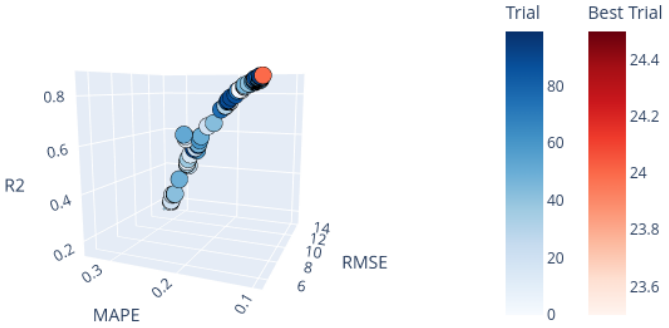
Para la optimización de hiper parámetros de modelo se ha utilizado la librería *optuna,* que permite realizar esta labor de manera sencilla. Se adjunta el bucle de optimización utilizado, incluyendo la división estratificada de los datos en entreno y validación:

```python
from sklearn.metrics import mean_squared_error, mean_absolute_percentage_error, r2_score
from sklearn.model_selection import KFold
import lightgbm as lgb
import numpy as np
import optuna

df_train.dropna(inplace=True)
X = df_train.drop("cons", axis=1).values
y = df_train[["cons"]].values
n_splits = 5

def objective(trial):
    params = {
        "objective": "regression",
        "metric": "rmse",
        "n_estimators": trial.suggest_int("n_estimators", 200, 1000, log=True),
        "verbosity": -1,
        "bagging_freq": 1,
        "learning_rate": trial.suggest_float("learning_rate", 1e-3, 0.1, log=True),
        "num_leaves": trial.suggest_int("num_leaves", 2, 2**10),
        "subsample": trial.suggest_float("subsample", 0.05, 1.0),
        "colsample_bytree": trial.suggest_float("colsample_bytree", 0.05, 1.0),
        "min_data_in_leaf": trial.suggest_int("min_data_in_leaf", 1, 100)
    }
    kf = KFold(n_splits=n_splits)
    rmse, mape, r2 = np.zeros(n_splits), np.zeros(n_splits), np.zeros(n_splits)
    for i, (train, test) in enumerate(kf.split(np.arange(X.shape[0]))):
        X_train, X_val, y_train, y_val = X[train], X[test], y[train], y[test]
        model = lgb.LGBMRegressor(**params)
        model.fit(X_train, np.squeeze(y_train))
        predictions = model.predict(X_val)
        rmse[i] = mean_squared_error(np.squeeze(y_val), predictions, squared=False)
        mape[i] = mean_absolute_percentage_error(np.squeeze(y_val), predictions)
        r2[i] = r2_score(np.squeeze(y_val), predictions)

    return rmse.mean(), mape.mean(), r2.mean()
```

Se adjuntan las gráficas de la optimización, durante 100 pasos, del modelo del dataset de Gijón. La izquierda muestra el progreso del parámetro objetivo (RMSE), mientras que abajo se sitúa el frente de Pareto entre las distintas métricas utilizadas.



Pareto-front Plot

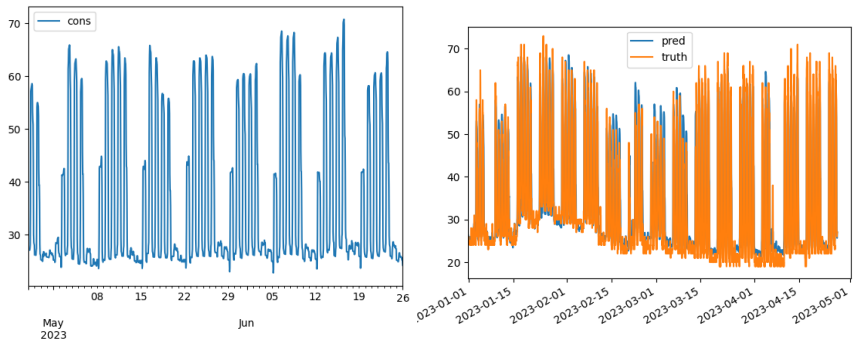


3 Resultados

En la tabla se muestran los resultados de los entrenamientos. Para cada dataset se muestra la raíz del error medio cuadrático (RMSE), el error medio absoluto (MAPE) y la puntuación R2 de un regresor que predice el valor del punto anterior (Dummy), los datos medios de la optimización con la cross-validation (CV) y la auto-regresión de los datos de entrenamiento.

	Gijón			Peón		
	Dummy	CV	AR	Dummy	CV	AR
RMSE	5.93	5.30	3.79	0.49	0.73	0.36
MAPE	0.08	0.09	0.07	0.2	0.56	0.19
R2	0.85	0.87	0.94	0.8	0.42	0.88

Finalmente, se adjunta una gráficas de la predicción objetivo (izquierda) y la autoregresión (derecha) para el dataset de Gijón. Las predicciones se adjuntan en dos ficheros ".csv" distintos, uno con el formato del PDF y otro con el formato de los datasets proporcionados.



4 Conclusiones

Se ha entrenado un modelo de LightGBM para la predicción del consumo energético de los dos centros de CTIC en Asturias, utilizando variables temporales (dia de la semana, mes, día laborable/festivo, horario laboral/no laboral, etc.), estadísticas de consumo previas y temperaturas como variables.

Si bien el modelo en sí (LightGBM) es bastante estándar en competiciones de este tipo, cabe destacar la optimización de hiper parámetros realizada con el sistema *optuna*, así como la (infructuosa aunque útil) utilización del software PyCaret para la comparativa de modelos.

A nivel personal, nunca había lidiado con un problema de regresión de series temporales per sé. Tampoco había obtenido nunca información de datasets públicos para este tipo de actividades. Si bien los festivos fueron relativamente fáciles de adquirir con una simple búsqueda en Internet, las temperaturas diarias presentaron una complicación mayor, por lo que acabé recurriendo a *parsear* las tablas HTML de una web con pandas.

Creo que el enfoque es fácilmente escalable a más volúmenes de datos (el entrenamiento lleva menos de medio segundo en un portátil), y su implantación es bastante sencilla. El modelo como tal no es generalizable puesto que está bastante adaptado a predecir un edificio de oficina, pero la metodología podría ser extendida obviando algunas particularidades. Las estadísticas de consumo diario se pueden obtener fácilmente de registros pasados, y la temperatura se puede tomar como la temperatura esperada ese día en base a valores pasados.

Si cambiasen significativamente los hábitos de consumo del edificio (mayor número de empleados, nuevo sistema de AC, etc.) el enfoque podría dejar de ser válido. En dicho caso se podrían implementar estadísticas de consumo de media móvil por día de la semana para tener esto en cuenta.