

DAJER

David Muñoz del Valle

Artur Vargas Carrión

Juan Manuel Valenzuela González

Eduardo González Bautista

Rubén Oliva Zamora



DAJER



DESCRIPCIÓN DEL PROBLEMA

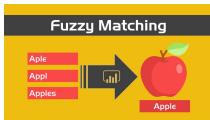


DESCRIPCIÓN DEL PROBLEMA



CÓMO LO HEMOS HECHO: 1^a FASE

- Fuzzy matching



- Oracle Cloud + SQL



Resultado de la Consulta						
	ID_EMBALE	NOMBRE_EMBALES	AMBITO_NOMBRE	NOMBRE_EMPAREJ	DEMARC_EMPAREJ	CODIGO_EM
50		124	CAMPORREDONDO	DUERO	CAMPORREDONDO	DUERO
51		126	CASTRO	DUERO	CASTRO	DUERO
52		127	LINARES DEL ARROYO	DUERO	LINARES DEL ARROYO	DUERO
53		129	RICOBAYO	DUERO	RICOBAYO	DUERO
54		133	NAVACERRADA	TAJO	NAVACERRADA	TAJO
55		135	SALOR	TAJO	SALOR	TAJO
56		138	LA PEDRERA	SEGURA	PEDRERA, LA	SEGURA
57		141	REGAJO	JÚCAR	REGAJO, EL	JÚCAR
58		142	CAMARASA	EBRO	CAMARASA	EBRO
59		143	EBRO	EBRO	EBRO	EBRO
60		145	SALLENTE	EBRO	SALLENTE	EBRO
61		149	ULLIVARRI	EBRO	ULLIVARRI	EBRO
62		157	FRESNEDA	GUADALQUIVIR	FRESNEDA, LA	GUADALQUIVIR
63		159	GUADALMENA	GUADALQUIVIR	GUADALMENA	GUADALQUIVIR
64		160	HUESNA	GUADALQUIVIR	HUESNA	GUADALQUIVIR
65		162	PUENTE NUEVO	GUADALQUIVIR	PUENTE NUEVO	GUADALQUIVIR

```
# Iterar por cada fila de la tabla 'embalses'
for index, row_embalses in embalses.iterrows():
    nombre_embalses = row_embalses['EMBALSE_NOMBRE']
    ambito_nombre = row_embalses['AMBITO_NOMBRE']
    id_embalse = row_embalses['ID'] # Obtener el ID del embalse

    # Realizar fuzzy matching entre 'AMBITO_NOMBRE' y 'DEMARC' en la tabla 'lista'
    demarc_matches = difflib.get_close_matches(ambito_nombre, lista['DEMARC'], n=1, cutoff=0.9)

    if demarc_matches:
        # Filtrar la tabla 'lista' con el resultado del fuzzy matching en 'DEMARC'
        lista_filtrada = lista[lista['DEMARC'] == demarc_matches[0]]

        # Realizar fuzzy matching entre los nombres de embalses en las filas coincidentes
        embalse_matches = difflib.get_close_matches(nombre_embalses, lista_filtrada['NOMBRE'], n=1, cutoff=0.6)

        # Almacenar el nombre original, el ambito, el ID y el emparejado (si existe)
        if embalse_matches:
            resultados.append({
                'ID_Embalse': id_embalse, # Añadir ID del embalse
                'Nombre_Embalses': nombre_embalses,
                'Ambito_Nombre': ambito_nombre,
                'Nombre_Emparejado': embalse_matches[0],
                'Demarc_Emparejado': demarc_matches[0]
            })
        else:
            resultados.append({
                'ID_Embalse': id_embalse, # Añadir ID del embalse
                'Nombre_Embalses': nombre_embalses,
                'Ambito_Nombre': ambito_nombre,
                'Nombre_Emparejado': None,
                'Demarc_Emparejado': demarc_matches[0]
            })
    else:
        resultados.append({
            'ID_Embalse': id_embalse, # Añadir ID del embalse
            'Nombre_Embalses': nombre_embalses,
            'Ambito_Nombre': ambito_nombre,
            'Nombre_Emparejado': None,
            'Demarc_Emparejado': None
        })

# Convertir los resultados en un DataFrame
resultados_df = pd.DataFrame(resultados)
```

CÓMO LO HEMOS HECHO: 2^a FASE

- Servidor externo en Hostinger (jajerje.fun)
- API Rest 
- HTML (Hugo), CSS y JavaScript



Mapa Interactivo de Embalses

Haga clic en el mapa para seleccionar un punto. Luego, introduzca el radio (en kilómetros) para mostrar los embalses dentro de ese radio.

Radio (km): Buscar Embalses Usar mi ubicación



```
[{"items": [ {"fecha": "2024-11-01T00:00:00Z", "prediccion_agua": 51.946671951645, "id_embalse": 18, "agua_total": 60, "prediccion_sequia": "False", "codigo_embalse": 1320004 }, {"fecha": "2024-12-01T00:00:00Z", "prediccion_agua": 51.9446177756994, "id_embalse": 18, "agua_total": 60, "prediccion_sequia": "False", "codigo_embalse": 1320004 }, {"fecha": "2025-01-01T00:00:00Z", "prediccion_agua": 51.9424951272222, "id_embalse": 18, "agua_total": 60, "prediccion_sequia": "False", "codigo_embalse": 1320004 }, {"fecha": "2025-02-01T00:00:00Z", "prediccion_agua": 51.9403724787451, "id_embalse": 18, "agua_total": 60, "prediccion_sequia": "False", "codigo_embalse": 1320004 }, {"fecha": "2025-03-01T00:00:00Z", "prediccion_agua": 51.9384552478625, "id_embalse": 18, "agua_total": 60, "prediccion_sequia": "False", "codigo_embalse": 1320004 }, {"fecha": "2025-04-01T00:00:00Z", "prediccion_agua": 51.936325993853, "id_embalse": 18, "agua_total": 60, "prediccion_sequia": "False", "codigo_embalse": 1320004 }, {"fecha": "2025-05-01T00:00:00Z", "prediccion_agua": 51.9342784234396, "id_embalse": 18, "agua_total": 60, "prediccion_sequia": "False", "codigo_embalse": 1320004 }], [{"label": "Leaflet © OpenStreetMap contributors"}]
```

CÓMO LO HEMOS HECHO: 3^a FASE



CÓMO LO HEMOS HECHO: 4^a FASE

- Regresión lineal

```
# Crear el modelo de regresión lineal
model = LinearRegression()

# Ajustar el modelo a los datos históricos
model.fit(X, y)

# Predecir los próximos 12 meses
last_date = df_embalse.index[-1]
future_dates = pd.date_range(last_date, periods=13, freq='MS')[1:] # Próximos 12 meses

# Convertir las fechas futuras a formato numérico
future_dates_num = np.array([convertir_fecha_a_num(fecha) for fecha in future_dates]).reshape(-1, 1)

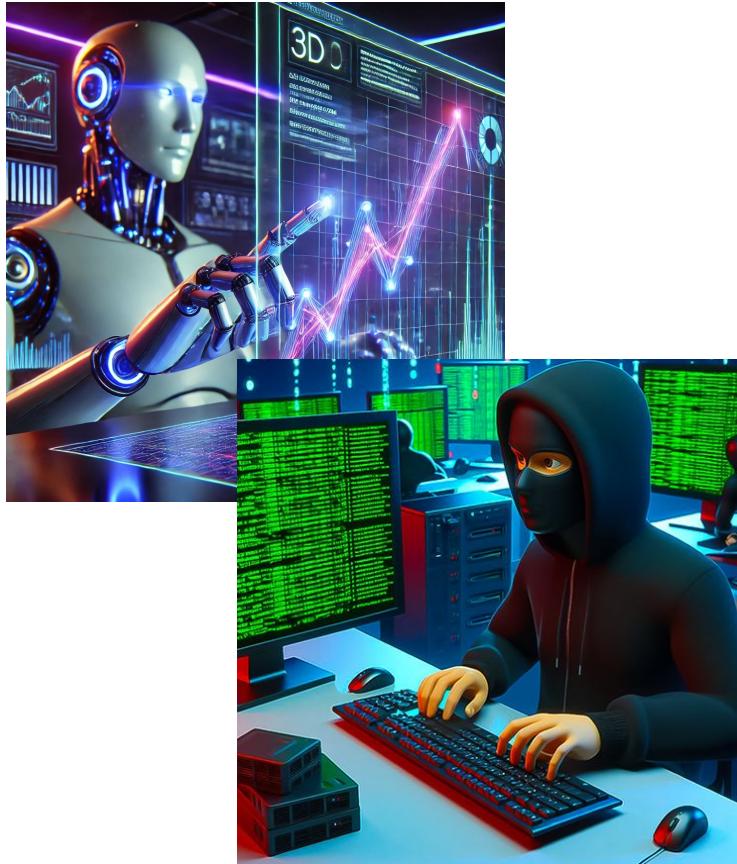
# Predecir la cantidad de agua para las fechas futuras
forecast = model.predict(future_dates_num)

# Crear un DataFrame con las predicciones para este embalse
forecast_df = pd.DataFrame({
    'FECHA': future_dates,
    'PREDICCIÓN_AQUA': forecast,
    'ID_EMBALESE': embalse_id,
    'AGUA_TOTAL': df_embalse['AGUA_TOTAL'].iloc[0] # Suponemos que AGUA_TOTAL no cambia
})

# Añadir una columna booleana que indica si el agua predicha baja del 25% de la capacidad total
forecast_df['BAJO_25_PORCIENTO'] = forecast_df['PREDICCIÓN_AQUA'] < (0.25 * forecast_df['AGUA_TOTAL'])

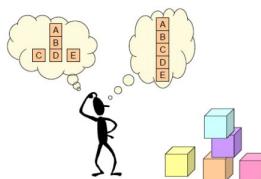
# Añadir las predicciones al DataFrame total
predicciones_totales = pd.concat([predicciones_totales, forecast_df])

# Guardar el nuevo archivo CSV con la columna adicional
predicciones_totales.to_csv('predicciones_embalses_con_bajo_25_por_ciento.csv', index=False)
```



CÓMO LO HEMOS HECHO: 5^a FASE

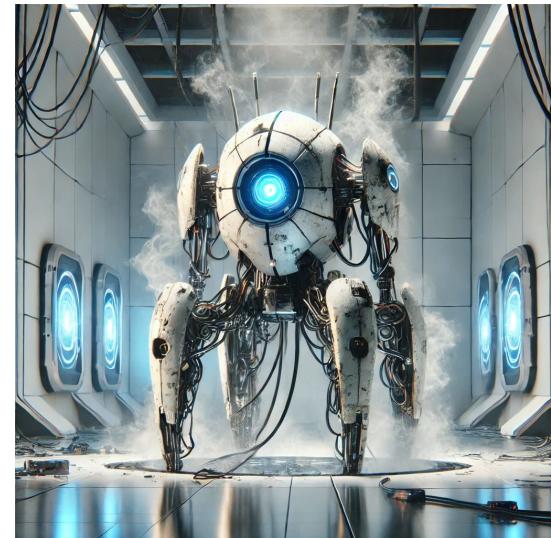
- Versión muy preliminar



- Algoritmo sencillo:

sequia = contenido < 0.25*capacidad_maxima

- Planteamiento no implementado de expansión



PRODUCTO FINAL



o dirígete a
<https://jajerje.fun/es/>



POSIBLES EXPANSIONES

- Refinamiento del algoritmo de **predicción**

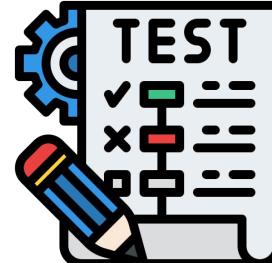


- **¿Escalabilidad y rendimiento?**

- Una página más **accesible**



- **Test unitarios**
- Limpieza código (**code review**)



LECCIONES APRENDIDAS



Las herramientas IA
son muy potentes



Cambia a un plan superior

Personal Empresa

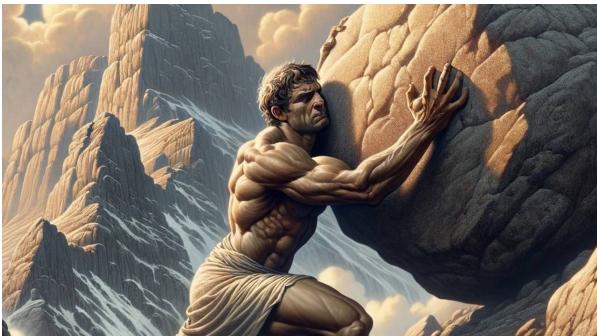
Plus

\$ 20 USD/
mes

Mejora tu productividad con un acceso ampliado

Tu plan actual

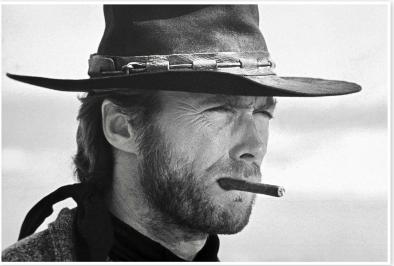
- ✓ Todo lo que está incluido en la versión gratuita
- ✓ Acceso a OpenAI o1-preview, OpenAI o1-mini
- ✓ Acceso a GPT-4o, GPT-4o mini, GPT-4
- ✓ Acceso a análisis de datos, cargas de archivos,



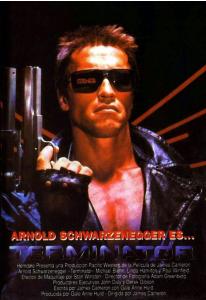
Nunca es tarde para sacar
el proyecto adelante 😐



LECCIONES APRENDIDAS



Es fundamental elegir un
buen equipo



LECCIONES APRENDIDAS

Las cosas bien hechas...

Si algo se quiere hacer bien siempre ocupará más tiempo del que aparentaba inicialmente...

yo al terminar esto



Aprender a buscar la información

MUCHAS GRACIAS POR SU ATENCIÓN

Partners



Mercedes-Benz
Group Services Madrid



Patrocinadores

ORACLE **Google**



UNIVERSIDAD
DE MÁLAGA



Cátedra de Gestión para el desarrollo Tecnológico FINTECH

