Prova Precipitacions Barcelona

September 29, 2021

Aram Puig Capdevila, Data Science - IT Academy
14 de julio del 2021

0.0.1 Objetivo: ¿Cuándo se prevén más lluvias?

Hacer un análisis y recomendación sobre las precipitaciones de la ciudad de Barcelona basados en sus datos históricos.

0.0.2 Guión:

1. Obtener los datos necesarios y estructurarlos (https://opendata-ajuntament.barcelona.cat/data/ca/dataset/precipitacio-hist-bcn/resource/5da03f48-020e-4f46-9199-a919feac2034?inner_span=True) i https://www.meteo.cat/wpweb/climatologia/serveis-i-dades-climatiques/serie-climatica-historica-de-barcelona/.
2. Guardar los datos de manera estructurada en formato SQL.
3. Realizar un plan sobre (máximo una página) sobre el tipo de análisis a realizar y cómo se presentarán los resultados.
4. Utilizar PANDAS, NUMPY, MATPLOTLIB (u otras librerías) junto a un cuaderno JUPYTER para la presentación del trabajo realizado.
5. Subir el resultado final a un repositorio privado de Github y compartirlo una vez finalizado con el tutor.

Tiempo estimado: 2 días.

0.0.3 Debe teber:

- El plan sobre el análisis a realizar.
- Incluir un análisis supervisado y uno no supervisado.

0.0.4 A considerar:

- Intentar no atascarse en ningún punto. Se evaluará el alcance del proyecto de manera global.
- No se busca la perfección en la ejecución del ejercicio, pero obtener información sobre las limitaciones y conocimientos de los candidatos para ofrecer feedback.

[1]:
```
# Librerias básicas
import pandas as pd
```

```
import numpy as np
import re
from datetime import datetime
# Librerias visualización
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
import seaborn as sns
from pandas_profiling import ProfileReport

#Otras
import warnings

pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
pd.set_option('display.expand_frame_repr', False)

warnings.filterwarnings('ignore')
```

[2]:
```
df = pd.read_csv('precipitacionsbarcelonadesde1786_format_long.csv')
```

[3]:
```
df
```

[3]:
```
        Any  Mes  Desc_Mes  Precipitacions
0      1786    1     Gener            32.8
1      1786    2    Febrer            28.4
2      1786    3      Març            84.4
3      1786    4     Abril            42.3
4      1786    5      Maig             8.5
…     …  …       …              …
2815   2020    8     Agost            12.4
2816   2020    9  Setembre            60.2
2817   2020   10   Octubre            23.1
2818   2020   11  Novembre            52.5
2819   2020   12  Desembre            41.5

[2820 rows x 4 columns]
```

0.1 1. Introducción

Las precipitaciones varían de acuerdo a ciertos ciclos temporales determinados por los movimientos de rotación y traslación terrestres y por la localización astronómica o geográfica del lugar de que se trate. Esos ciclos pueden ser: diarios, mensuales o estacionales o en ciclos anuales, en efecto, siempre hay meses en que las precipitaciones son mayores que en otros.

Para poder evaluar correctamente las características objetivas del clima, en el cual la precipitación, y en especial la lluvia, desempeña un papel muy importante, las precipitaciones mensuales deben haber sido observadas por un período de por lo menos 20 a 30 años, lo que se llama un período de

observación largo.

La variación estacional de las precipitaciones, en especial de la lluvia, define el año hidrológico. Este da inicio en el mes siguiente al de menor precipitación media de largo período.

Fuentes: 1. https://es.wikipedia.org/wiki/Precipitaci%C3%B3n_(meteorolog%C3%ADa)#Origen_de_la_

0.1.1 1.1 Características del clima de Barcelona y sus precipicationes

La ciudad de Barcelona posee un clima mediterráneo con influencias marítimas. De acuerdo con los criterios de la clasificación de Köppen-Geiger la ciudad se encuentra próxima a una zona de transición del clima subtropical de veranos secos y calurosos Csa (clima mediterráneo) a un clima subtropical húmedo Cfa.

Una de las singularidades del clima mediterráneo asociadas al ciclo del agua es que la estación más húmeda es el otoño. Esto se debe a la inestabilidad generada por las diferencias de temperatura entre la superficie del mar Mediterráneo y la atmósfera. durante el otoño, las aguas del mar que se han calentado durante todo el verano, entran en contacto con el aire frío. Esta diferencia térmica es la responsable de los chubascos intensos que en pocas horas dejan grandes cantidades de agua.

Los recurrentes episodios de sequía y de lluvias intensas son un hecho natural y típico de nuestra climatología mediterránea. En el futuro, según las proyecciones climáticas se prevé que el efecto que tendrá el cambio climático sobre el ciclo del agua en general, hará más acusada esta problemática de episodios extremos.

La precipitación media anual en Barcelona se sitúa en torno a los 600 mm, con un máximo de precipitaciones de fin de verano y principio de otoño (llegando a superar los 90 mm de media en octubre), que es originado a menudo por el fenómeno conocido como gota fría, que ha llegado a superar en numerosas ocasiones los 100 mm en un día.

La gota fría, DANA (depresión aislada en niveles altos) o baja segregada, es un fenómeno meteorológico anual que suele coincidir con el inicio del otoño y la primavera en el Mediterráneo occidental. Se experimenta particularmente en España y más concretamente a lo largo de la costa este y las islas Baleares, aunque sus efectos pueden sentirse en zonas interiores también. A grandes rasgos, la gota fría es el resultado de un frente de aire polar frío (corriente en chorro) que avanza lentamente sobre Europa occidental a gran altura (normalmente 5-9 km) y que, al chocar con el aire más cálido y húmedo del Mar Mediterráneo, genera fuertes y dañinas tormentas.

Por el contrario, el mínimo se produce al comienzo del verano, llegando a la media algo por encima de los 20 mm en julio. La humedad media anual es alta debido a las condiciones marítimas de la ciudad, situándose entre 69 y 70 % y variando poco a lo largo del año.

Fuentes: 1. https://es.wikipedia.org/wiki/Barcelona#Clima
2. https://www.barcelona.cat/barcelona-pel-clima/sites/default/files/documents/cap08_
3. https://es.wikipedia.org/wiki/Gota_fr%C3%ADa

0.1.2 1.2 Efectos del cambio climático

El Grupo Intergubernamental de Expertos sobre Cambio Climático (más conocido por sus siglas en inglés, IPCC) es una entidad científica creada en 1988 por la Organización Meteorológica Mundial (OMM) y el Programa de las Naciones Unidas para el Medio Ambiente (PNUMA). El IPCC concluye que el calentamiento del sistema climático es inequívoco observado cambios en el sistema

climático que no tienen precedente, tanto si se comparan con registros históricos observacionales, que datan de mediados del siglo XIX, como si se comparan con registros paleoclimáticos referidos a los últimos milenios. Las observaciones permiten afirmar que la atmósfera y los océanos se han calentado, que la cantidad y extensión de las masas de hielo y nieve han disminuido, que el nivel del mar ha subido y que las concentraciones de gases de efecto invernadero han aumentafo (IPCC 2013).

La temperatura media global muestra un incremento de 0,85ºC (entre 0,65 y 1,06ºC) en el período 1880-2012. Cada una de las últimas tres décadas ha sido más calida que todas las anteriores desde 1850, siendo la primera década del siglo XXI la más cálida de todas. Las tendencias en periodos cortos (entre 10 y 15 años) están muy afectadas por la variabilidad natural, tal y como sucede, por ejemplo, en los últimos 15 años, en los que la tasa de calentamiento ha sido inferior a la media registrada desde 1951.

La precipitación ha aumentado en las zonas terrestres de latitudes medias del hemisferio norte desde 1950. Se han observado cambios en los episodios extemos desde 1950. El número de días y noches frías ha disminuido y el número de días y noches cálidas ha aumentado a nivel global (IPCC 2013).

[…] Unas emisiones iguales a las tasas actuales o superiores inducirán cambios en todos los componentes del sistema climático, algunos de ellos sin precedentes en cientos o miles de años. Los cambios tendrán lugar en todas las regiones del globo, incluyendo cambios en la tierra y en el océano, en el ciclo del agua, en la criosfera, en el nivel del mar, en algunos episodios extremos y en la acidez de los océanos. Muchos de estos cambios persistirán durante muchos siglos. La limitación del cambio climático requerirá reducciones substanciales y sostenidas de las emisiones de CO2 (IPCC 2013).

[…] En la región Mediterránea tendrá lugar un incremento de temperatura superior a la media global, más pronunciado en los meses estivales que en los invernales. Para el escenario RCP 8.5 y para finales del siglo XXI, la región Mediterránea experimentará incrementos medios de temperatura de 3,8ºC y de 6,0ºC en los meses invernales y estivales, respectivamente, y reducciones medias de precipitación de 12% y 24% en los meses invernales y estivales , respectivamente. Habrá un aumento de los extremos relacionados con las precipitaciones de origen tormentoso (IPCC 2013).

Los Escenarios RCP (del inglés, Representative Concentration Pathways) son cuatro escenarios de emisiones sobre la evolución estimada de la emisión y concentración de gases de efecto invernadero a la atmósfera durante el siglo XXI, establecidos por el IPCC para la elaboración de su quinto informe de evaluación.

Los cuatro escenarios RCP son: RCP2.6, RCP4.5, RCP6.0 y RCP8.5, cuyo nombre se basa en el posible rango de valores de forzamiento radiativo alcanzado en 2100 (2,6; 4,5 ; 6,0 y 8,5 W / m2, respectivamente). El escenario RCP8.5 sigue el rango más alto de emisiones de gases de efecto invernadero, con concentraciones que crecen rápidamente

`Fuentes: 1. https://www.miteco.gob.es/es/cambio-climatico/temas/impactos-vulnerabilidad-y-adapt`

0.1.3 1.3 Proyecciones de las precipitaciones en Barcelona

El estudio realizado por el Servicio Meteorológico de Cataluña (SMC, 2015) sobre la generación de escenarios climáticos futuros regionalizados para el área metropolitana de Barcelona en una primera fase que incluye el periodo 1971-2050, concluye que para las próximas cuatro décadas,

independientemente del escenario de emisiones, se proyecta por toda el área metropolitana un aumento de temperatura alrededor de 1ºC (horquilla de 0.8 a 1.1ºC) respecto al valor actual; situándose 2050 entre 1 y 2ºC por encima de la media del periodo 1971-2000 . este incremento afecta tanto a las temperaturas media como la máxima y la mínima, siendo sensiblemente mayor en los valores mínimos. Estacionalmente, el mayor incremento tendrá lugar en verano.

Respecto a la precipitación, las simulaciones muestran un rango de variación más amplio, con una tendencia menos clara, sobre todo a escala estacional . A grandes rasgos, se espera una ligera disminución que podría alcanzar los 55 mm, pero con una alta variabilidad interanual. Habrán pocos cambios o un aumento moderado en invierno y en verano, y una disminución marcada en primavera. Los extremos de precipitación aumentarán de manera apreciable, esperándose episodios de lluvias intensas.

`Fuentes: 1. https://www.amb.cat/es/web/ecologia/actualitat/publicacions/detall/-/publicacio/ef`

0.1.4 1.4 Unidades de medida y historial de registros

La unidad de medida que usaremos para las precipitaciones acumuladas mensuales son los milímetros (mm). En el caso del agua, existe una correspondecia entre 1 mm y 1 l/m2, porque un litro en un cubo de un metro de ancho y un metro de largo ocupa en volumen exactamente un milímetro de altura. Un cubo de 1 metro cúbico (un metro de alto, uno de ancho y uno de alto) tiene una capacidad de 1000 litros. Si un metro de altura son 1000 milímetros (mm), entonces 1 mm corresponderá, por tanto, a un litro.

```
1 mm = 1 l/m2
```

Las precipitaciones en forma de nieve también se expresan en cm (centímetros) en lugar de milímetros:

```
1 cm nieve =  10 mm nieve
```

Se considera que, aproximadamente, el espesor de la nieve es 10 veces el del agua. Por tanto, 1 cm de nieve equivaldría aproximadamente a 1 mm de agua, aunque esto puede variar de 0,5 a 2 mm en función de la densidad de la nieve.

```
1 cm nieve   1 mm lluvia
```

Historial de registros Tenemos los datos de las precipitaciones acomuladas (mm) en la ciudad de Barcelona para cada mes desde enero de 1786 hasta diciembre de 2020.

[4]:
```
df.isna().sum()
```

[4]:
```
Any              0
Mes              0
Desc_Mes         0
Precipitacions   0
dtype: int64
```

0.1.5 1.5 Estructuración de los datos

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2820 entries, 0 to 2819
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Any             2820 non-null   int64
 1   Mes             2820 non-null   int64
 2   Desc_Mes        2820 non-null   object
 3   Precipitacions  2820 non-null   float64
dtypes: float64(1), int64(2), object(1)
memory usage: 88.2+ KB
```

Añadimos la variable día con todos los valores en el día 1, para poder convertir las columnas de Año y Mes a formato 'datetime'. NO significa que los datos fueran recogidos al día uno de cada mes, es sólo para tener esta parte de la fecha. Pasamos las tres variables de fecha a string para juntarlas en una sola columna y a continuación pasarla a datetime. Así podremos trabajar con time series.

[6]:
```
a = [1 for x in range(len(df))]
```

[7]:
```
df['Dia'] = a
```

[8]:
```
df['Dia'] = df['Dia'].astype(str)
```

[9]:
```
df['Any'] = df.Any.apply(str)
```

[10]:
```
df['Mes'] = df.Mes.apply(str)
```

[11]:
```
df['Date'] = df['Any'] + '-' + df['Mes'] + '-' + df['Dia']
```

[12]:
```
df['Date']
```

[12]:
```
0        1786-1-1
1        1786-2-1
2        1786-3-1
3        1786-4-1
4        1786-5-1
           ...
2815     2020-8-1
2816     2020-9-1
2817    2020-10-1
2818    2020-11-1
2819    2020-12-1
Name: Date, Length: 2820, dtype: object
```

[13]:
```
pd.to_datetime(df['Date'])
```

[13]:
```
0       1786-01-01
1       1786-02-01
2       1786-03-01
3       1786-04-01
4       1786-05-01
           ...
2815    2020-08-01
2816    2020-09-01
2817    2020-10-01
2818    2020-11-01
2819    2020-12-01
Name: Date, Length: 2820, dtype: datetime64[ns]
```

[14]:
```
df.drop(['Dia'], 1, inplace=True)
```

[15]:
```
df = df[['Date', 'Any', 'Mes', 'Desc_Mes', 'Precipitacions']]
```

[16]:
```
df.set_index('Date', inplace=True)
```

[17]:
```
df['Any'] = df['Any'].astype(int)
df['Mes'] = df['Mes'].astype(int)
```

[18]:
```
df.to_csv('data_precipitacions.csv')
```

0.2 2. Conexión a MySql

Pasamos los datos a MySql, creando la base de datos relacional y las tablas correspondientes, e inyectamos los datos directamente desde el Workbench. Nos conectamos a ella mediante python y accedemos a los datos.

[19]:
```
import mysql.connector
from getpass import getpass
from mysql.connector import connect, Error
```

[20]:
```
# connection.close()
```

[23]:
```
connection = connect(
        host="localhost",
        user=input("Enter username: "),
        password=getpass("Enter password: "),
        database = 'precipitacions'
    )
```

```
Enter username: root
Enter password: ········
```

[24]:
```
cursor = connection.cursor()
```

Comprobamos que funcionen la connexión y los datos cargados a MySQL. Los insertamos en un dataframe:

[25]:
```
select_clients_query = "SELECT * FROM hist_bcn"

cursor.execute(select_clients_query)
result = cursor.fetchall()
```

[26]:
```
result[:5]
```

[26]:
```
[(1, '1786', '1', 'Gener', '32.8'),
 (2, '1786', '2', 'Febrer', '28.4'),
 (3, '1786', '3', 'MarÃ§', '84.4'),
 (4, '1786', '4', 'Abril', '42.3'),
 (5, '1786', '5', 'Maig', '8.5')]
```

[28]:
```
columns = 'ind', 'Any', 'Mes', 'Mes_desc', 'Precipitacions'
```

[29]:
```
sql_df = pd.DataFrame(result, columns=columns)
```

[30]:
```
sql_df
```

[30]:

	ind	Any	Mes	Mes_desc	Precipitacions
0	1	1786	1	Gener	32.8
1	2	1786	2	Febrer	28.4
2	3	1786	3	MarÃ§	84.4
3	4	1786	4	Abril	42.3
4	5	1786	5	Maig	8.5
…	…	…	..	…	…
2815	2816	2020	8	Agost	12.4
2816	2817	2020	9	Setembre	60.2
2817	2818	2020	10	Octubre	23.1
2818	2819	2020	11	Novembre	52.5
2819	2820	2020	12	Desembre	41.5

[2820 rows x 5 columns]

0.3 3. Exploración

[31]:
```
profile = ProfileReport(df, title="Pandas Profiling Report", explorative=True)
```

[28]:
```
profile
```

```
Summarize dataset:   0%|          | 0/18 [00:00<?, ?it/s]

Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]

Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

```
<IPython.core.display.HTML object>
```

[28]:

0.3.1 3.1 Outliers

[32]:
```
from scipy import stats
from scipy.stats import iqr
```

[33]:
```
df.drop('Desc_Mes', 1, inplace=True)
```

[34]:
```
df.skew()
```

[34]:
```
Any              0.000000
Mes              0.000000
Precipitacions    1.871138
dtype: float64
```

[35]:
```
z = np.abs(stats.zscore(df))
print(z)
```

```
[[1.72469599 1.59325501 0.36676043]
 [1.72469599 1.30357228 0.46388918]
 [1.72469599 1.01388955 0.77229486]
 …
 [1.72469599 1.01388955 0.58088516]
 [1.72469599 1.30357228 0.06811145]
 [1.72469599 1.59325501 0.17471041]]
```

[36]:
```
outliers = np.where(z>3)
```

[37]:
```
outliers
```

[37]:
```
(array([  10,   20,   50,   57,   94,  296,  692,  704,  764,  808,  860,
         920,  969, 1041, 1076, 1340, 1344, 1391, 1461, 1545, 1593, 1630,
        1713, 1875, 1895, 1897, 1989, 2084, 2157, 2231, 2287, 2296, 2316,
        2325, 2374, 2421, 2492, 2504, 2710, 2793, 2794, 2811], dtype=int64),
 array([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
        2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2],
       dtype=int64))
```

[38]:
```
outliers[0]
```

[38]:
```
array([  10,   20,   50,   57,   94,  296,  692,  704,  764,  808,  860,
         920,  969, 1041, 1076, 1340, 1344, 1391, 1461, 1545, 1593, 1630,
        1713, 1875, 1895, 1897, 1989, 2084, 2157, 2231, 2287, 2296, 2316,
        2325, 2374, 2421, 2492, 2504, 2710, 2793, 2794, 2811], dtype=int64)
```

[39]:
```
len(outliers[0])
```

[39]: 42

De momento sólo los identificamos, aún no decidimos que hacer con ellos

0.3.2 3.2 Estudio de 1786 a 1990

[40]:
```
df = pd.read_csv('data_precipitacions.csv')
```

[41]:
```
df.set_index('Date', inplace=True)
```

[42]:
```
mean_ppl = df['Precipitacions'].mean()
```

[43]:
```
mean_ppl
```

[43]: 49.41450354609924

[44]:
```
df[:1990-1-1]['Precipitacions'].mean()
```

[44]: 48.201710261569495

[45]:
```
x = round(df.groupby(by='Mes')['Precipitacions'].mean().sum(),2)
print('La media histórica (1786-2020) anual de precipitaciones (mm) es igual a:
 ↪', x)
```

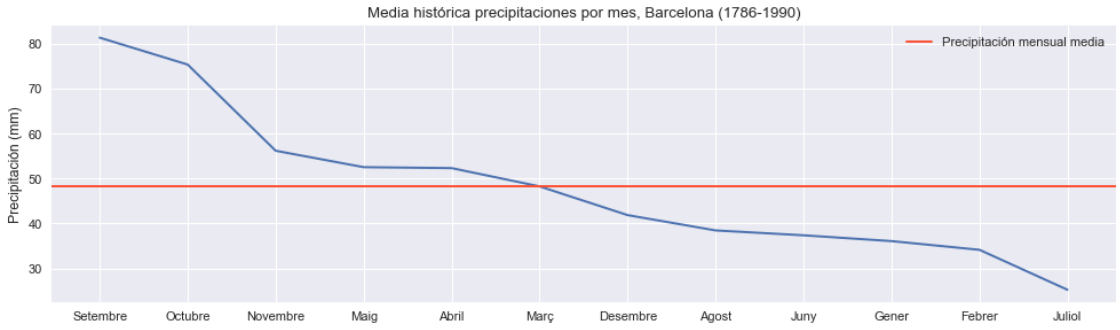
```
La media histórica (1786-2020) anual de precipitaciones (mm) es igual a: 592.97
```

[46]:
```
hist_prep = round(df[:1990-1-1].groupby(by='Mes')['Precipitacions'].mean().
 ↪sort_values(ascending=False),2)
hist_prep
```

[46]:
```
Mes
9     81.31
10    75.32
11    56.15
5     52.50
4     52.29
3     48.22
12    41.84
8     38.43
6     37.34
1     36.05
2     34.11
7     25.23
Name: Precipitacions, dtype: float64
```

[47]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)
```

```
plt.plot(df[:1990-1-1].groupby(by='Desc_Mes')['Precipitacions'].mean().
 ↪sort_values(ascending=False))
plt.axhline(df[:1990-1-1]['Precipitacions'].mean(), color ='#fc4f30',␣
 ↪label='Precipitación mensual media')

plt.title('Media histórica precipitaciones por mes, Barcelona (1786-1990)')
# plt.xlabel('Mes (numérico)')
plt.ylabel('Precipitación (mm)')

plt.legend()

plt.show()
```



[48]:
```
df['Any'] = df['Any'].astype(int)
```

0.3.3 3.3 Estudio de 1990 a 2020

Creamos un df derivado del primero que incluya sólo los datos de los últimos 30 años (1990-2020) para comparar las medias hitóricas con las 'actuales'.

[49]:
```
df30 = df[df['Any'] > 1989]
```

[50]:
```
x = round(df30.groupby(by='Mes')['Precipitacions'].mean().sum(),2)
print('La media histórica (1990-2020) anual de precipitaciones (mm) es igual a:
 ↪', x)
```

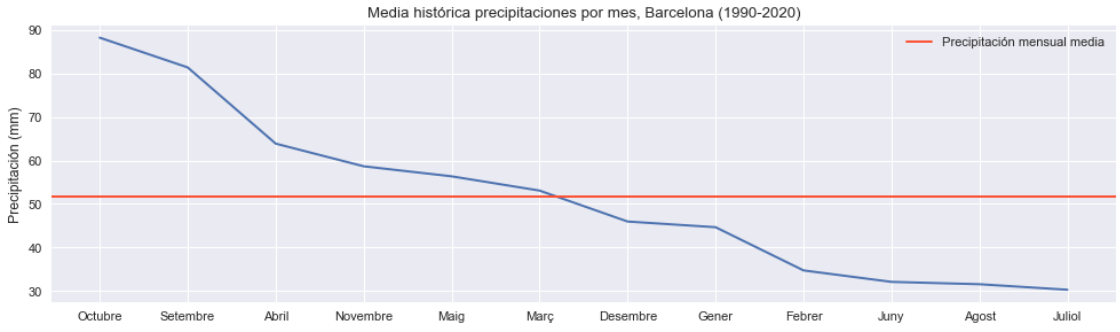
```
La media histórica (1990-2020) anual de precipitaciones (mm) es igual a: 620.6
```

[51]:
```
prep90 = round(df30.groupby(by='Mes')['Precipitacions'].mean().
 ↪sort_values(ascending=False),2)
prep90
```

[51]:
```
Mes
10    88.25
9     81.40
```

```
4     63.87
11    58.65
5     56.34
3     53.05
12    45.94
1     44.63
2     34.68
6     32.05
8     31.51
7     30.23
Name: Precipitacions, dtype: float64
```

[52]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)

plt.plot(df30.groupby(by='Desc_Mes')['Precipitacions'].mean().
 ↪sort_values(ascending=False))

plt.title('Media histórica precipitaciones por mes, Barcelona (1990-2020)')

plt.axhline(df30['Precipitacions'].mean(), color ='#fc4f30',␣
 ↪label='Precipitación mensual media')

plt.ylabel('Precipitación (mm)')

plt.legend()

plt.show()
```



Observamos que entre la media más alta de los últimos 30 años (octubre, 88.25mm) y la más baja (julio, 30.23mm) hay una diferencia de 58.02mm. En octubre hay tres veces las precipitaciones de julio, aproximadamente. Le siguen septiembre, con 81.40mm y abril, ya con cierta diferencia, 63.87mm.

A simple vista podemos dividir las temperaturas medias de los último treinta años en tres grupos:

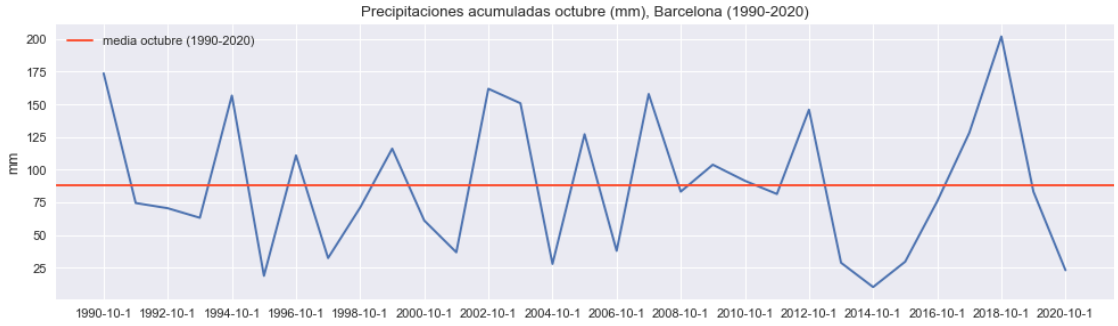
```
                                                1990-2020    1786-1990

Grupo 1, máximas precipitaciones:

                                  - Octubre:      88.25        77.80 (2)
                                  - Septiembre:   81.40        78.65 (1)

Grupo 2, precipitaciones medianas:

                                  - Abril:        63.87        52.30 (5)
                                  - Noviembre:    58.65        58.32 (3)
                                  - Mayo:         56.34        53.68 (4)
                                  - Marzo:        53.05        48.32 (6)
                                  - Diciembre:    45.94        42.98 (7)
                                  - Enero:        44.63        37.23 (10)

Grupo 3, precipitacioines bajas:

                                  - Febrero:      34.68        34.86 (11)
                                  - Junio:        32.05        38.28 (9)
                                  - Agosto:       31.51        40.51 (8)
                                  - Julio:        30.23        25.73 (12)
```

3.3.1 Precipitaciones octubre 1990-2020

[53]:
```
df30['Precipitacions'][df['Mes']==10].mean()
```

[53]:
```
88.25161290322579
```

[54]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)

plt.plot(df30['Precipitacions'][df['Mes']==10])
plt.axhline(df30['Precipitacions'][df['Mes']==10].mean(), color ='#fc4f30',␣
 ↪label='media octubre (1990-2020)')

plt.title('Precipitaciones acumuladas octubre (mm), Barcelona (1990-2020)')
plt.ylabel('mm');

plt.legend()

plt.xticks(np.arange(0, 31, 2))

plt.show()
```

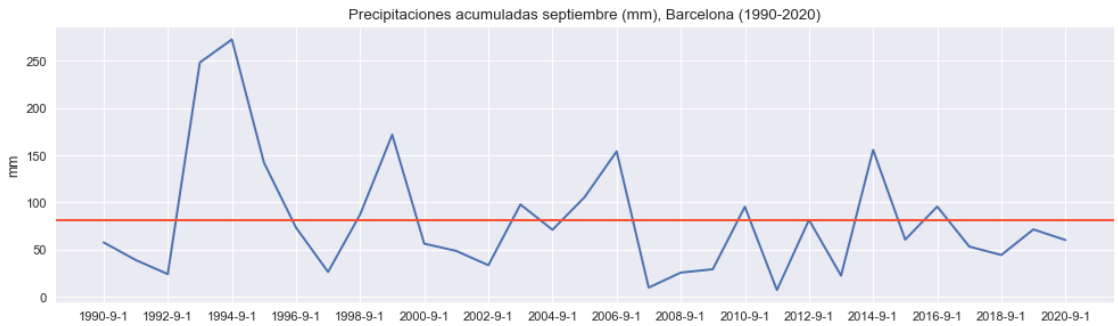


Con los datos de octubre, actualmente el mes con la media de precipitaciones más elevadas, podemos ver una gran variabilidad interanual, habiendo caído el año pasado (2020) tan solo 23.1 mm acumulados y el 2018, 201.9mm.

A parte de su alta variabilidad, podemos apreciar cierto con un año de grandes lluvias y después uno o dos años con pocas precipitacions. No es exacto, pero si hay una especie de variabilidad interanual cíclia. Como veremos a continuación, este patrón aparece con mayor o menor variabilidad en el resto de meses lluviosos. Viendo las sumas anuales, el fenómeno es el mismo, por este motivo hablamos de años lluvios o de sequía.

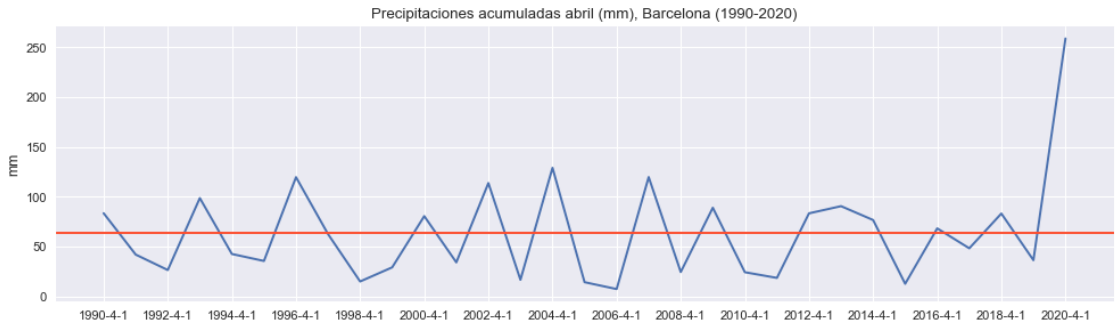
3.3.2 Precipitaciones septiembre 1990-2020

[55]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)

plt.plot(df30['Precipitacions'][df['Mes']==9])
plt.axhline(df30['Precipitacions'][df['Mes']==9].mean(), color ='#fc4f30',␣
↪label='media septiembre (1990-2020)')

plt.title('Precipitaciones acumuladas septiembre (mm), Barcelona (1990-2020)')
plt.ylabel('mm');

plt.xticks(np.arange(0, 31, 2))

plt.show()
```



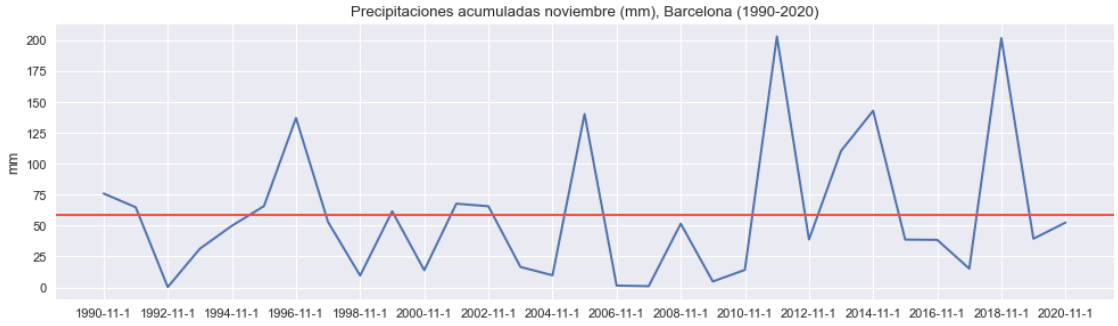
3.3.3 Precipitaciones abril 1990-2020

[56]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)

plt.plot(df30['Precipitacions'][df['Mes']==4])
plt.axhline(df30['Precipitacions'][df['Mes']==4].mean(), color ='#fc4f30',␣
↪label='media abril (1990-2020)')

plt.title('Precipitaciones acumuladas abril (mm), Barcelona (1990-2020)')
plt.ylabel('mm');

plt.xticks(np.arange(0, 31, 2))

plt.show()
```



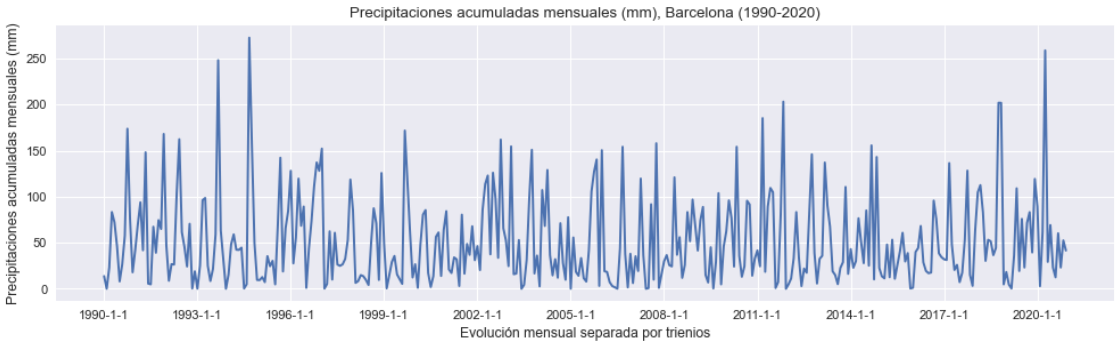
3.3.4 Precipitaciones noviembre 1990-2020

[57]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)

plt.plot(df30['Precipitacions'][df['Mes']==11])
plt.axhline(df30['Precipitacions'][df['Mes']==11].mean(), color ='#fc4f30',␣
↪label='media noviembre (1990-2020)')

plt.title('Precipitaciones acumuladas noviembre (mm), Barcelona (1990-2020)')
plt.ylabel('mm');

plt.xticks(np.arange(0, 31, 2))

plt.show()
```



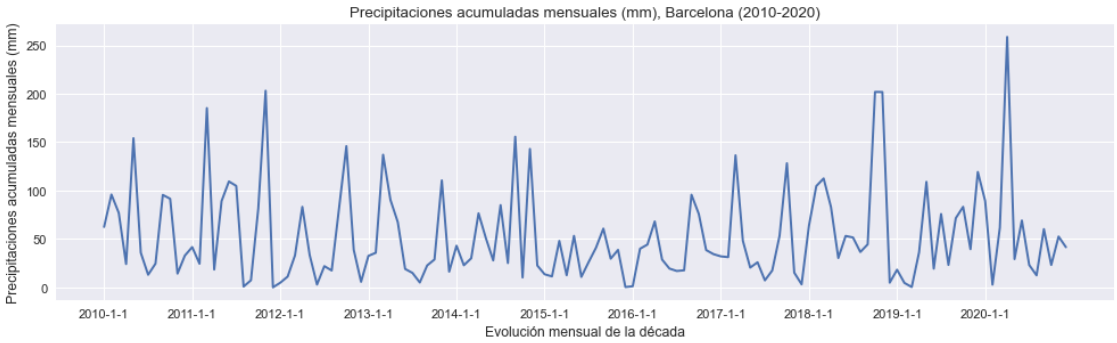
3.3.5 Variación interanual

[58]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)

plt.plot(df30.index, df30['Precipitacions'])

plt.title('Precipitaciones acumuladas mensuales (mm), Barcelona (1990-2020)')
plt.ylabel('Precipitaciones acumuladas mensuales (mm)');
plt.xlabel('Evolución mensual separada por trienios')

plt.xticks(np.arange(0, 396, 36))

plt.show()
```



[59]:
```
plt.style.use('seaborn')
figure(figsize=(15, 4), dpi=80)

plt.plot(df['2010-1-1':].index, df['2010-1-1':]['Precipitacions'])

plt.title('Precipitaciones acumuladas mensuales (mm), Barcelona (2010-2020)')
plt.ylabel('Precipitaciones acumuladas mensuales (mm)');
```

```
plt.xlabel('Evolución mensual de la década')

plt.xticks(np.arange(0, 132, 12))

plt.show()
```



0.4 4. Clusters No-Supervisados

[60]:
```
#Librerias para clustering no-supervisado

from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.cluster import KMeans
from sklearn import metrics
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import AgglomerativeClustering
import scipy.cluster.hierarchy as sch
from sklearn.preprocessing import normalize
from kneed import DataGenerator, KneeLocator
from sklearn.cluster import DBSCAN
from sklearn.mixture import GaussianMixture
from sklearn.metrics import adjusted_rand_score
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder
```

[61]:
```
df = pd.read_csv('precipitacionsbarcelonadesde1786_format_long.csv')
```

[62]:
```
a = [1 for x in range(len(df))]

df['Dia'] = a

df['Dia'] = df['Dia'].astype(str)

df['Any'] = df.Any.apply(str)
```

```
df['Mes'] = df.Mes.apply(str)

df['Date'] = df['Any'] + '-' + df['Mes'] + '-' + df['Dia']

df['Date']

pd.to_datetime(df['Date'])

df.drop(['Dia'], 1, inplace=True)

df = df[['Date', 'Any', 'Mes', 'Desc_Mes', 'Precipitacions']]

df.set_index('Date', inplace=True)
```

[63]:
```
df['Mes'] = df['Mes'].astype(int)
df['Any'] = df['Any'].astype(int)
```

[64]:
```
df.reset_index(inplace=True)
```

[65]:
```
df.drop(['Desc_Mes', 'Date'], 1, inplace=True)
```

[66]:
```
df
```

[66]:

	Any	Mes	Precipitacions
0	1786	1	32.8
1	1786	2	28.4
2	1786	3	84.4
3	1786	4	42.3
4	1786	5	8.5
…	…	…	…
2815	2020	8	12.4
2816	2020	9	60.2
2817	2020	10	23.1
2818	2020	11	52.5
2819	2020	12	41.5

[2820 rows x 3 columns]

0.4.1 4.1 Hopkins Statistics

To understand if the dataset can be clustered, we will use the Hopkins statistic, which tests the spatial randomness of the data and indicates the cluster tendency or how well the data can be clustered. It calculates the probability that a given data is generated by a uniform distribution (Alboukadel Kassambara, n.d.). The inference is as follows for a data of dimensions 'd':

- If the value is around 0.5 or lesser, the data is uniformly distributed and hence it is unlikely to have statistically significant clusters.

- If the value is between {0.7, …, 0.99}, it has a high tendency to cluster and therefore likely to have statistically significant clusters.

---

[67]:
```
from sklearn.neighbors import NearestNeighbors
from random import sample
from numpy.random import uniform
import numpy as np
from math import isnan

def hopkins(X):
    d = X.shape[1]
    #d = len(vars) # columns
    n = len(X) # rows
    m = int(0.1 * n) # heuristic from article [1]
    nbrs = NearestNeighbors(n_neighbors=1).fit(X.values)

    rand_X = sample(range(0, n, 1), m)

    ujd = []
    wjd = []
    for j in range(0, m):
        u_dist, _ = nbrs.kneighbors(uniform(np.amin(X,axis=0),np.
↪amax(X,axis=0),d).reshape(1, -1), 2, return_distance=True)
        ujd.append(u_dist[0][1])
        w_dist, _ = nbrs.kneighbors(X.iloc[rand_X[j]].values.reshape(1, -1), 2,␣
↪return_distance=True)
        wjd.append(w_dist[0][1])

    H = sum(ujd) / (sum(ujd) + sum(wjd))
    if isnan(H):
        print(ujd, wjd)
        H = 0

    return H
```

[68]:
```
round(hopkins(df), 5)
```

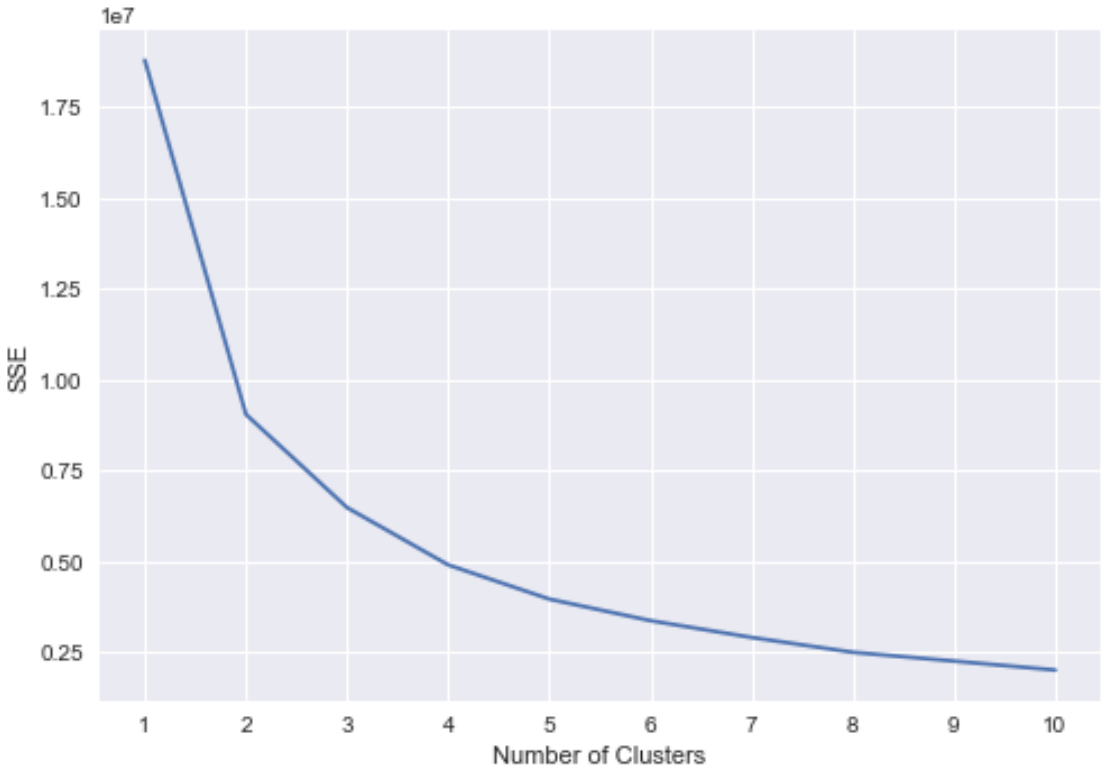
[68]:
```
0.84838
```

El cálculo de Hopkins Statistics nos devuelve 0.86129, un valor entre 0.7 y 0.99, por lo cual en principio nuestros datos son clusterizables.

0.4.2 4.2 Elbow Method for K-Means

[69]:
```
kmeans_kwargs = {
    "init": "random",
    "n_init": 10,
    "max_iter": 300,
    "random_state": 42,
}

# A list holds the SSE values for each k
sse = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, **kmeans_kwargs)
    kmeans.fit(df)
    sse.append(kmeans.inertia_)
```

[70]:
```
plt.style.use("seaborn")
plt.plot(range(1, 11), sse)
plt.xticks(range(1, 11))
plt.xlabel("Number of Clusters")
plt.ylabel("SSE")
plt.show()
```



[71]:
```
kl = KneeLocator(
    range(1, 11), sse, curve="convex", direction="decreasing"
)

kl.elbow
```

[71]: 3

0.4.3 4.3 Silhouette coefficient en vez de SSE

[72]:
```
# A list holds the silhouette coefficients for each k
silhouette_coefficients = []

# Notice you start at 2 clusters for silhouette coefficient
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, **kmeans_kwargs)
    kmeans.fit(df)
    score = metrics.silhouette_score(df, kmeans.labels_, sample_size = 1000)
    silhouette_coefficients.append(score)
```

[73]:
```
plt.style.use("seaborn")
plt.plot(range(2, 11), silhouette_coefficients)
plt.xticks(range(2, 11))
plt.xlabel("Number of Clusters")
plt.ylabel("Silhouette Coefficient")
plt.show()
```

0.4.4 4.4 Agrupación

[74]:
```
kmeans = KMeans(init="random",
       n_clusters=3,
       n_init=10,
     max_iter=300,
    random_state= 0)
```

[75]:
```
kmeans.fit(df)
```

[75]:
```
KMeans(init='random', n_clusters=3, random_state=0)
```

[76]:
```
# The lowest SSE value
kmeans.inertia_
```

[76]:
```
6481130.369840048
```

[77]:
```
# Locations of the centroid
kmeans.cluster_centers_
```

[77]:
```
array([[1841.31965944,    6.40712074,   39.95162539],
       [1962.62340967,    6.35114504,   34.36064461],
       [1929.91977077,    7.34670487,  135.30143266]])
```

[78]:
```
# The number of iterations required to converge
kmeans.n_iter_
```

[78]:
```
14
```

[79]:
```
labels = kmeans.labels_
```

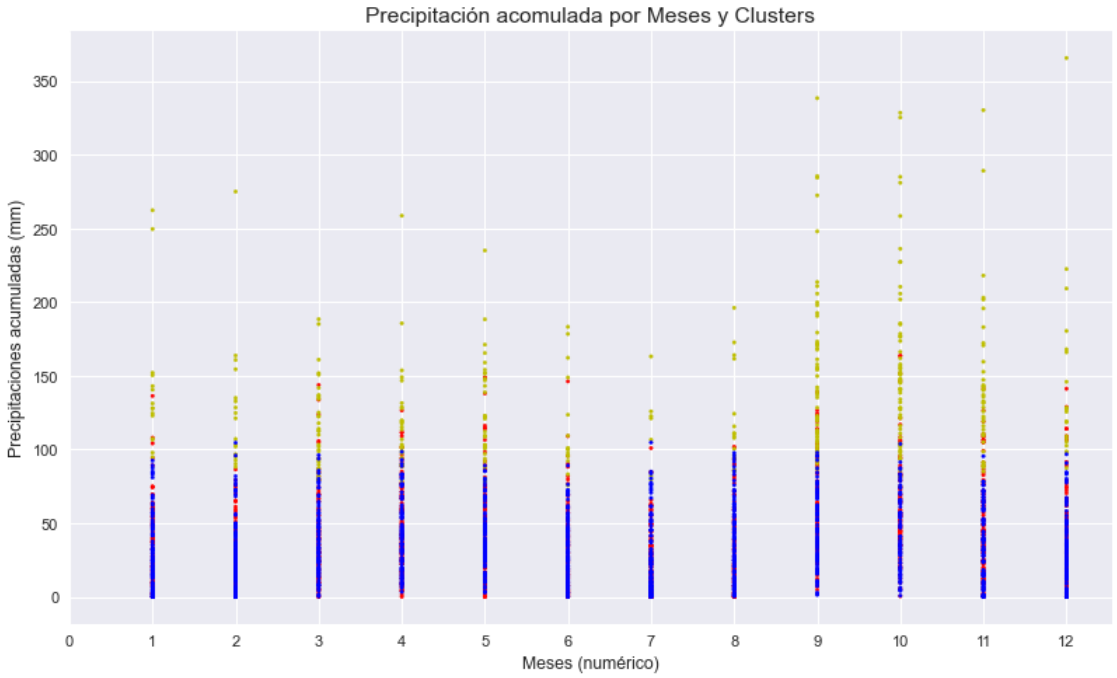
[80]:
```
labels
```

[80]:
```
array([0, 0, 0, …, 1, 1, 1])
```

Hacemos un scatter plot diferenciando por cluster de las precipitaciones por meses:

[81]:
```
centroids = kmeans.cluster_centers_
```

[82]:
```
plt.style.use('seaborn')
figure(figsize=(12, 7), dpi=80)

colors = ['r','b','y','g','c','m']
plt.scatter(df['Mes'], df['Precipitacions'], color=[colors[l_] for l_ in␣
 ↪labels], label=labels, s=5)

plt.title('Precipitación acomulada por Meses y Clusters', size=14)

plt.xticks(np.arange(0, 13))
plt.xlabel('Meses (numérico)')
plt.ylabel('Precipitaciones acumuladas (mm)')
# plt.legend(['Cluster 0', 'Cluster 1', 'Cluster 2'])

plt.show()
```



Los puntos Rojos pertenecen al Cluster 0, los Azules al Cluster 1 y los Amarillos al Cluster 2. Viendo esta visualización podemos intuir que la cantidad de precipitaciones ha jugado un papel importante en la división de los clusters. Pero vamos a asegurarnos y ver como han quedado divididos los meses y años con el análisis estadístico de los clusters.

0.4.5 4.5 Estudio estadístico de los clusters

[83]:
```
labels = pd.Series(labels)
```

[84]:
```
df['Labels'] = labels
```

[85]:
```
df['Labels'].value_counts(normalize=True)*100
```

[85]:
```
0    45.815603
1    41.879433
2    12.304965
Name: Labels, dtype: float64
```

Número d'observacions assignades a cada cluster:

```
Cluster 0:    1292    45.82%
Cluster 1:    1181    42.88%
Cluster 2:    347     12.30%
```

Agrupamos por clusters

[86]:
```
label_grp = df.groupby(['Labels'])
```

4.5.1 Precipitaciones

[87]:
```
df['Precipitacions'].mean()
```

[87]:
```
49.41450354609924
```

[88]:
```
label_grp['Precipitacions'].describe()
```

[88]:

Labels	count	mean	std	min	25%	50%	75%	max
0	1292.0	39.951625	31.275677	0.0	14.8	35.0	58.35	163.6
1	1181.0	34.452667	24.648468	0.0	14.2	30.4	51.40	104.8
2	347.0	135.570029	49.610654	75.0	101.0	122.7	154.10	365.8

Podemos observar, a parte de la disparidad de registros en cada cluster, una desigualdad en las medias de precipitaciones. El primer (0) y el segundo (1) cluster tienen unas medias parecidas (39,95 y 34,45mm respectivamente), aunque inferiores a la media de la población (49,41). El tercer cluster (2) se desmarca, con una media de 135,57mm y una desviación estandar de 49,61.

Es significativo observar que mientras el mínimo de los dos primeros clusters es 0, el del tercero es 75. Los máximos siguen un patron parecido: (1) 163,6 (2) 104,8 (3) 365,8.

Vamos a investigar como han quedado repartidos los meses y años para cada uno.

4.5.2 Meses En el df original los valores de los meses son proporcionalmente iguales (=0,83333). A continuación podemos ver como han quedado repoartidos por Clusters:

[89]:
```
round(label_grp['Mes'].value_counts(normalize=True)*100,2)
```

[89]:

Labels	Mes	
0	12	8.82
	1	8.75
	7	8.75
	2	8.67
	5	8.67
	8	8.59
	6	8.51
	4	8.44
	3	8.36
	11	7.97
	10	7.59
	9	6.89
1	7	9.74
	6	9.31
	8	9.14
	1	9.06
	2	8.98
	12	8.55

```
         4       8.30
         3       7.96
         5       7.71
         11      7.62
         9       7.28
         10      6.35
2        10     17.87
         9      17.29
         11     12.10
         3       9.51
         5       9.22
         4       8.07
         12      5.76
         2       4.90
         8       4.61
         1       4.32
         6       4.32
         7       2.02
Name: Mes, dtype: float64
```

Conociendo como se reparten las cantidades de precipitaciones anuales por meses podemos inferir qué lógica ha seguido el proceso de clustering no-supervisado.

Viendo la repartición proporcional de meses por cluster sabemos que si la aparición de ese mes en el cluster es superior a 8,33%, estará por encima de la media de la población, mientras que si es inferior estará por debajo.

En el primer cluster(0) los valores estan repartidos casi equitativamente, destacan por debajo de la media:

```
Septiembre: 6,88%
Octubre:    7,58%
Noviembre:  7,97%
```

Y por encima, ningún mes aparece más de un 0.5% que la media de la población (estan por debajo de 8,83%). Los mas presentes son:

```
Diciembre:  8,82%
Enero:      8,75%
Julio:      8,75%
```

Conc. Vemos como los meses con menos presencia coinciden con los meses históricos de más precipitaciones.

En el segundo cluster(1) hay una mayor variabilidad en la presencia proporcional de cada mes, tanto por encima como por debajo de la media. Destacan por debajo:

```
Octubre:    6,35%
Septiembre: 7,28%
Noviembre:  7,62%
```

Y por encima de la media:

```
Julio:      9.74%
Junio:      9.31%
Agosto:     9.14%
```

Conc. En el caso del segundo cluster vemos que la distribución de los meses es parecida a la del primero, pero acentuando su caso. Los meses que menos aparecen corresponden también a los más lluviosos históricamente y los que más aparecen, a los menos lluviosos.

En el tercer cluster(2) las distribuciones son totalmente dispares, teniendo meses que aparecen hasta ocho veces más que otros (Octubre/Julio). Tenemos a octubre (17.87%) y septiembre (17.29%), que doblan la media, y noviembre significativament por encima (12.10%).

El menos representado es julio (2.02%), seguido por junio, enero, agosto y febrero (todos alrededor del 4%).

Conc. En el tercer cluster se observa una tendencia clara a reunir aquellos registros de los meses que están asociados históricamente con la mayor cantidad de precipitaciones.

4.5.1 Años

[90]:
```
len(df['Any'].unique())
```

[90]:
```
235
```

La media de la población en este caso es igual a 1/número de años distintos = 1/235 = 0,00425 = 0,42%

[91]:
```
label_grp['Any'].describe()
```

[91]:

Labels	count	mean	std	min	25%	50%	75%	max
0	1292.0	1841.319659	32.945499	1786.0	1813.0	1840.0	1869.0	1902.0
1	1181.0	1962.616427	34.023512	1901.0	1934.0	1963.0	1992.0	2020.0
2	347.0	1929.755043	49.542362	1786.0	1897.0	1926.0	1969.0	2020.0

[92]:
```
round(label_grp['Any'].value_counts(normalize=True)*100,2)[2].unique()
```

[92]:
```
array([1.44, 1.15, 0.86, 0.58, 0.29])
```

[93]:
```
round(label_grp['Any'].value_counts(normalize=True)*100,2)[2].value_counts()
```

[93]:
```
0.29    56
0.58    52
0.86    33
1.15    12
1.44     8
Name: Any, dtype: int64
```

[94]:
```
grp_144 = [1901, 1906, 1907, 1921, 1923, 1943, 1971, 1996]
```

[95]:
```
df[df['Any'] == 1901].describe()
```

[95]:
```
              Any         Mes  Precipitacions      Labels
count    12.0  12.000000       12.000000  12.000000
mean   1901.0   6.500000       85.883333   1.000000
std       0.0   3.605551       69.310642   0.953463
min    1901.0   1.000000        9.200000   0.000000
25%    1901.0   3.750000       31.000000   0.000000
50%    1901.0   6.500000       65.850000   1.000000
75%    1901.0   9.250000      130.725000   2.000000
max    1901.0  12.000000      209.300000   2.000000
```

[96]:
```
df.isin({'Any': [grp_144]}).value_counts()
```

[96]:
```
Any    Mes    Precipitacions  Labels
False  False  False           False     2820
dtype: int64
```

[99]:
```
round(label_grp['Any'].value_counts(normalize=True)*100,2)[2].head()
```

[99]:
```
Any
1901    1.44
1906    1.44
1907    1.44
1921    1.44
1923    1.44
Name: Any, dtype: float64
```

[100]:
```
round(label_grp['Any'].value_counts(normalize=True)*100,2)[2].keys().
↪sort_values()
```

[100]:
```
Int64Index([1786, 1787, 1790, 1793, 1810, 1811, 1820, 1840, 1843, 1844,
            …
            2008, 2010, 2011, 2012, 2013, 2014, 2017, 2018, 2019, 2020],
dtype='int64', name='Any', length=161)
```

[101]:
```
len((label_grp['Any'].value_counts(normalize=True)*100)[0].keys())
```

[101]:
```
117
```

[102]:
```
len((label_grp['Any'].value_counts(normalize=True)*100)[1].keys())
```

[102]:
```
120
```

[103]:
```
len((label_grp['Any'].value_counts(normalize=True)*100)[2].keys())
```

[103]:
```
161
```

Esta variable es más dificil de describir porqué tiene 235 valores distintos. Del mismo modo que antes, compararemos su proporción en cada cluster con la media de la población. Podemos ver que en los 3 casos ocurre un fenomeno similar. La mayoría de los años aparecidos en cada cluster

tiene una proporción superior o muy superior a la de la media. Este hecho nos indica que es muy probable que no aparezcan registros de todos los años en cada cluster es decir, que los años han quedado repartidos por clusters.

Años distintos por cluster (sobre 235):

```
Cluster 0:    117
Cluster 1:    120
Cluster 2:    161
```

Ordenando por fecha ascendiente vemos en qué rango estñan los años de cada cluster:

```
Cluster 0:    1786 - 1902
Cluster 1:    1901 - 2020
Cluster 2:    1786 - 2020
```

El tercer cluster incluye todos los años pero tiene una distribución particular de estos. Los años quedan agrupados en 5 tipos de distribuciones: 1.44, 1.15, 0.86, 0.58, 0.29. Recordamos que la media era 0,42%. Tenemos cuatro grupos que aparecen por encima la media, destacando por encima el de 1.44. Hay un grupo por debajo la media, 0.29.

```
Tipo    años distintos

0.29    56
0.58    52
0.86    33
1.15    12
1.44     8
```

0.4.6 4.6 Conclusiones estudio de clusters

Después de analizar los valores de cada una de las variables en cada cluster podemos inferir que el algoritmo no-supervisado ha dividio:

```
1. Por año: en el caso del primer cluster(0) y segundo cluster(1) hay una clara división en los

2. en el caso del tercer cluster, teniendo en cuenta la variable 'Año', hemos encontrado otro p

3. Por meses: mientras el primer y segundo cluster han mantenido una proporción de meses simila

4. Por precipitación: siguiendo la misma lógica, el primer y segundo cluster han tenido una med

5. Por último, el número de registros: los dos primeros clusters han reunido el 88.7% de los re
```

Cluster 0: cluster con 1292 (45.82%) registros, con datos del 1786 al 1902 y una distribución similar de los meses. Media de precipitaciones 39.95mm con desviación típica de 31.28.

Cluster 1: cluster con 1181 (42.88%) registros, datos del 1901 al 2020 y una distribución de los meses mñas acentuada hacía aquellos asociados a menos precipitaciones. Media de precipitaciones 34.45mm y una desviación típica de 24.65.

Cluster 2: cluster con 347 (12.30%) registros, datos del 1786 al 2020, con una distribución de meses totalmente inclinada hacía los meses asociados con más precipitaciones.

0.5 5. Modelo supervisado

[104]:
```
# Librerias regresión lineal
from sklearn import preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
import xgboost as xgb
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import AdaBoostRegressor
from sklearn.metrics import r2_score
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import GridSearchCV
from sklearn import model_selection

import statsmodels.api as sm
import math, datetime
import random
```

0.5.1 5.1 Modelo elegido y justificación

El modelo supervisado que se ajusta más al problema en cuestión (determinar la cantidad de precipitaciones acumuladas en cada mes en mm) es el de regresión lineal ya que nos permite trabajar y obtener datos numéricos continuos. Probaremos diferentes algorítmos de regresión lineal para quedarnos con el que mejor resultados nos proporcione.

Finalmente los algoritmos elegidos han sido LinearRegression y MLPRegressor por tener de media el r2 score más elevando, usando K-Fold model selection para comprobarlo.

0.5.2 5.1 Feature selection

Tras ver los resultados iniciales con todos los datos históricos y obtener un R2 score de -0.02, hemos intentado obtener nuevas variables para mejorar los modelos. Hemos añadido los labels resultantes del K-Means - cosa que ha mejorado el score hasta 0.4 - y después hemos añadido tambien las temperaturas máximas y mínimas en la ciudad por cada mes.

Los datos de temperatura se remontaban hasta el 1950 por lo que hemos cortado el dataset histórico por esta fecha.

[105]:
```
columns= ['Year', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12']
```

[106]:
```
temp_max = pd.read_csv('temp_max_anual.txt', delimiter = '\t', names=columns)
```

[107]:
```
temp_min = pd.read_csv('temp_min_anual.txt', delimiter = '\t', names=columns)
```

[108]:
```
temp_max.reset_index(inplace=True)
```

[109]:
```
max_list = []
for i in temp_max.index:
    x = temp_max.loc[i]
    for i in range (2,14):
        y = x[i]
        max_list.append(y)
```

[110]:
```
min_list = []
for i in temp_min.index:
    x = temp_min.loc[i]
    for i in range (1,13):
        y = x[i]
        min_list.append(y)
```

Nuevo df:

[111]:
```
df2 = df[df['Any'] > 1949]
```

[112]:
```
df2['max_temp'] = max_list
```

[113]:
```
df2['min_temp'] = min_list
```

[114]:
```
df2
```

[114]:

	Any	Mes	Precipitacions	Labels	max_temp	min_temp
1968	1950	1	7.3	1	10.9	5.4
1969	1950	2	4.8	1	13.3	6.5
1970	1950	3	91.5	2	14.8	7.8
1971	1950	4	66.8	1	15.4	7.6
1972	1950	5	47.1	1	20.0	12.9
…	…	…	…	…	…	…
2815	2020	8	12.4	1	29.9	21.1
2816	2020	9	60.2	1	26.1	17.4
2817	2020	10	23.1	1	20.7	12.0
2818	2020	11	52.5	1	17.9	11.4
2819	2020	12	41.5	1	12.4	6.2

[852 rows x 6 columns]

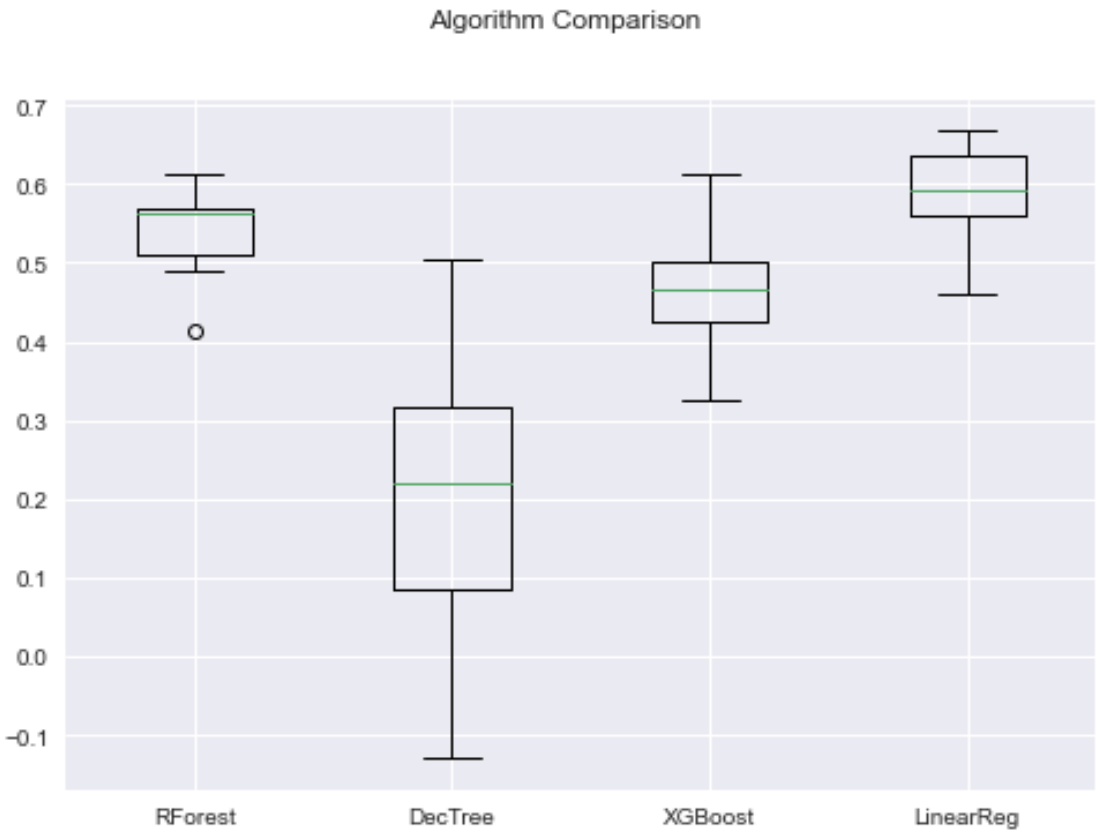
[115]:
```
forecast_col ='Precipitacions'
```

[116]:
```
X = df2.drop([forecast_col],1)
y = df2[forecast_col]
```

0.5.3 5.2 Model selection

[117]:
```
models = [('RForest', RandomForestRegressor()),
          ('DecTree', DecisionTreeRegressor()),
          ('XGBoost', xgb.XGBRegressor()),
          ('LinearReg', LinearRegression())]


results = []
names = []

for name, model in models:
    kfold = model_selection.KFold(n_splits=10, random_state=2, shuffle = True)
    cv_results = model_selection.cross_val_score(model, X, y, cv=kfold)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)



fig = plt.figure()
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()
```

```
RForest: 0.543114 (0.057645)
DecTree: 0.195874 (0.181715)
XGBoost: 0.468808 (0.082404)
LinearReg: 0.587405 (0.063472)
```



0.5.4 5.3 Linear Regression

[118]:
```
len(df2)
```

[118]: 852

[119]:
```
forecast_col ='Precipitacions'
```

[120]:
```
X = df2.drop([forecast_col],1)
y = df2[forecast_col]
```

[121]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

[122]:
```
steps = [('scaler', StandardScaler()), ('rf', LinearRegression())]
pipeline = Pipeline(steps)
```

[123]:
```
parameters = {}
```

[124]:
```
grid = GridSearchCV(pipeline, param_grid=parameters, cv=5)
```

[125]:
```
grid.fit(X_train, y_train)
print ("score = %3.4f" %(grid.score(X_test,y_test)))
print (grid.best_params_)
```

```
score = 0.6623
{}
```

[126]:
```
lr_pred = grid.predict(X_test)
```

[164]:
```
df_lr=pd.DataFrame({'Actual':y_test, 'Predicted':lr_pred})
df_lr[:10]
```

[164]:

	Actual	Predicted
1990	88.5	32.573583
2041	17.4	139.723966
2261	39.3	43.443573
2683	0.4	147.712848
2201	26.6	39.537782
2701	24.4	34.730412
2282	10.2	45.124738
2063	35.3	42.293652
2266	4.3	36.972625
2519	84.4	27.963137

[128]:
```
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, lr_pred).
 ↪round(2))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, lr_pred).
 ↪round(2))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,␣
 ↪lr_pred)).round(2))
```

```
Mean Absolute Error: 23.66
Mean Squared Error: 855.84
Root Mean Squared Error: 29.25
```

0.5.5 5.4 Random Forest Regressor

[129]:
```
forecast_col ='Precipitacions'
```

[130]:
```
X = df2.drop([forecast_col],1)
y = df2[forecast_col]
```

[131]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

[132]:
```
steps = [('scaler', StandardScaler()), ('rf', RandomForestRegressor())]
pipeline = Pipeline(steps)
```

[133]:
```
parameters = {'rf__n_estimators':[100, 500, 700], 'rf__max_features':['auto',
'sqrt', 'log2']}
```

[134]:
```
grid = GridSearchCV(pipeline, param_grid=parameters, cv=5)
```

[135]:
```
grid.fit(X_train, y_train)
print ("score = %3.4f" %(grid.score(X_test,y_test)))
print (grid.best_params_)
```

```
score = 0.5571
{'rf__max_features': 'log2', 'rf__n_estimators': 100}
```

[136]:
```
rf_pred = grid.predict(X_test)
```

[153]:
```
df_rf=pd.DataFrame({'Actual':y_test, 'Predicted':rf_pred})
df_rf[:10]
```

[153]:

	Actual	Predicted
1990	88.5	146.037
2041	17.4	58.685
2261	39.3	21.786
2683	0.4	151.571
2201	26.6	32.710
2701	24.4	30.853
2282	10.2	29.998
2063	35.3	121.351
2266	4.3	43.467
2519	84.4	20.891

[138]:
```
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, rf_pred).
↪round(2))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, rf_pred).
↪round(2))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,␣
↪rf_pred)).round(2))
```

```
Mean Absolute Error: 23.82
Mean Squared Error: 948.44
Root Mean Squared Error: 30.8
```

0.5.6 5.4 Neural network

Añadimos variables para intentar mejorar el r2 score de la regresión. Añadimos la temperatura máxima y mínima mensual de Barcelona des de 1950. El df que pasaremos al algoritmo será desde el 1950 a 2020 para no tener nulls.

[139]:
```
forecast_col ='Precipitacions'
```

[140]:
```
X = df2.drop([forecast_col],1)
y = df2[forecast_col]
```

[141]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

[142]:
```
scaler = StandardScaler()
scaler.fit(X_train)
StandardScaler(copy=True, with_mean=True, with_std=True)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

[143]:
```
mlp = MLPRegressor(hidden_layer_sizes=(6,6,6),max_iter=500)
```

[144]:
```
mlp.fit(X_train,y_train)
```

[144]:
```
MLPRegressor(hidden_layer_sizes=(6, 6, 6), max_iter=500)
```

[145]:
```
mlp.score(X_train,y_train)
```

[145]:
```
0.6130842436614741
```

[146]:
```
mlp_pred = mlp.predict(X_test)
```

[150]:
```
df_mlp=pd.DataFrame({'Actual':y_test, 'Predicted':mlp_pred})
df_mlp[:10]
```

[150]:

	Actual	Predicted
1990	88.5	142.947632
2041	17.4	53.033183
2261	39.3	32.491560
2683	0.4	28.965057
2201	26.6	35.724590
2701	24.4	37.210239
2282	10.2	28.454192
2063	35.3	47.709205
2266	4.3	38.692868
2519	84.4	32.948265

[154]:
```
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, mlp_pred).
 ↪round(2))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, mlp_pred).
 ↪round(2))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,␣
 ↪mlp_pred)).round(2))
```

```
Mean Absolute Error: 24.57
Mean Squared Error: 922.53
Root Mean Squared Error: 30.37
```

5.4.1 Hyperparameter tunning

[155]:
```
mlp = MLPRegressor(max_iter=500)
parameter_space = {
    'hidden_layer_sizes': [(50,50,50), (50,100,50), (100,)],
    'activation': ['tanh', 'relu'],
    'solver': ['sgd', 'adam'],
    'alpha': [0.0001, 0.05],
    'learning_rate': ['constant','adaptive'],
}
```

[156]:
```
clf = GridSearchCV(mlp, parameter_space, n_jobs=-1, cv=3)
clf.fit(X_train, y_train)
```

[156]:
```
GridSearchCV(cv=3, estimator=MLPRegressor(max_iter=500), n_jobs=-1,
             param_grid={'activation': ['tanh', 'relu'],
                         'alpha': [0.0001, 0.05],
                         'hidden_layer_sizes': [(50, 50, 50), (50, 100, 50),
                                                (100,)],
                         'learning_rate': ['constant', 'adaptive'],
                         'solver': ['sgd', 'adam']})
```

[157]:
```
# Best parameter set
print('Best parameters found:\n', clf.best_params_)
```

```
Best parameters found:
 {'activation': 'relu', 'alpha': 0.0001, 'hidden_layer_sizes': (100,),
'learning_rate': 'adaptive', 'solver': 'sgd'}
```

[158]:
```
# All results
means = clf.cv_results_['mean_test_score']
stds = clf.cv_results_['std_test_score']
for mean, std, params in zip(means, stds, clf.cv_results_['params']):
    print("%0.3f (+/-%0.03f) for %r" % (mean, std * 2, params))
```

```
0.595 (+/-0.111) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'constant', 'solver':
'sgd'}
0.158 (+/-0.066) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'constant', 'solver':
'adam'}
0.573 (+/-0.155) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'adaptive', 'solver':
'sgd'}
0.178 (+/-0.049) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'adaptive', 'solver':
'adam'}
0.595 (+/-0.107) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'constant', 'solver':
```

```
'sgd'}
0.185 (+/-0.030) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'constant', 'solver':
'adam'}
0.583 (+/-0.107) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'adaptive', 'solver':
'sgd'}
0.160 (+/-0.057) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'adaptive', 'solver':
'adam'}
0.578 (+/-0.116) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'constant', 'solver': 'sgd'}
0.562 (+/-0.100) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'constant', 'solver': 'adam'}
0.576 (+/-0.127) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'adaptive', 'solver': 'sgd'}
0.563 (+/-0.105) for {'activation': 'tanh', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'adaptive', 'solver': 'adam'}
0.592 (+/-0.134) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'constant', 'solver': 'sgd'}
0.156 (+/-0.066) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'constant', 'solver': 'adam'}
0.601 (+/-0.075) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'adaptive', 'solver': 'sgd'}
0.159 (+/-0.035) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'adaptive', 'solver': 'adam'}
0.601 (+/-0.098) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'constant', 'solver': 'sgd'}
0.174 (+/-0.044) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'constant', 'solver': 'adam'}
0.420 (+/-0.599) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'adaptive', 'solver': 'sgd'}
0.178 (+/-0.047) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'adaptive', 'solver': 'adam'}
0.578 (+/-0.127) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'constant', 'solver': 'sgd'}
0.562 (+/-0.102) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'constant', 'solver': 'adam'}
0.573 (+/-0.129) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'adaptive', 'solver': 'sgd'}
0.563 (+/-0.105) for {'activation': 'tanh', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'adaptive', 'solver': 'adam'}
0.209 (+/-1.039) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'constant', 'solver':
'sgd'}
0.584 (+/-0.107) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'constant', 'solver':
'adam'}
```

```
0.581 (+/-0.106) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'adaptive', 'solver':
'sgd'}
0.583 (+/-0.119) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 50, 50), 'learning_rate': 'adaptive', 'solver':
'adam'}
0.571 (+/-0.113) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'constant', 'solver':
'sgd'}
0.592 (+/-0.103) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'constant', 'solver':
'adam'}
0.561 (+/-0.101) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'adaptive', 'solver':
'sgd'}
0.592 (+/-0.112) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (50, 100, 50), 'learning_rate': 'adaptive', 'solver':
'adam'}
0.598 (+/-0.108) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'constant', 'solver': 'sgd'}
0.592 (+/-0.105) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'constant', 'solver': 'adam'}
0.611 (+/-0.102) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'adaptive', 'solver': 'sgd'}
0.590 (+/-0.107) for {'activation': 'relu', 'alpha': 0.0001,
'hidden_layer_sizes': (100,), 'learning_rate': 'adaptive', 'solver': 'adam'}
0.553 (+/-0.162) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'constant', 'solver': 'sgd'}
0.585 (+/-0.094) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'constant', 'solver': 'adam'}
0.575 (+/-0.134) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'adaptive', 'solver': 'sgd'}
0.592 (+/-0.098) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 50, 50), 'learning_rate': 'adaptive', 'solver': 'adam'}
0.558 (+/-0.112) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'constant', 'solver': 'sgd'}
0.582 (+/-0.113) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'constant', 'solver': 'adam'}
0.559 (+/-0.103) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'adaptive', 'solver': 'sgd'}
0.586 (+/-0.096) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(50, 100, 50), 'learning_rate': 'adaptive', 'solver': 'adam'}
0.589 (+/-0.104) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'constant', 'solver': 'sgd'}
0.590 (+/-0.105) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'constant', 'solver': 'adam'}
0.600 (+/-0.116) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'adaptive', 'solver': 'sgd'}
```

```
0.590 (+/-0.102) for {'activation': 'relu', 'alpha': 0.05, 'hidden_layer_sizes':
(100,), 'learning_rate': 'adaptive', 'solver': 'adam'}
```

Los mejores resultados del hyperparameter tunning no son diferentes de los que ya teníamos.

0.5.7 5.5 Scores

[159]:
```
columns = ['model', 'R2_score', 'MAE', 'MSE', 'RMSE']
```

[160]:
```
data = {'model':['LinearRegression', 'RandomForestRegressor', 'MLPRegressor'],
        'R2_score':[0.6187, 0.6115, 0.6286],
        'MAE':[26.05,24.69,23.08],
        'MSE':[1262.32,1153.99,908.52],
        'RMSE':[35.53,33.97,30.14]}
```

[161]:
```
results = pd.DataFrame(data)
```

[162]:
```
results.set_index('model', inplace=True)
```

[163]:
```
results
```

[163]:

model	R2_score	MAE	MSE	RMSE
LinearRegression	0.6187	26.05	1262.32	35.53
RandomForestRegressor	0.6115	24.69	1153.99	33.97
MLPRegressor	0.6286	23.08	908.52	30.14

0.6 6. Conclusiones

El primer hecho para evaluar las estimaciones de precipitaciones en un futuro es el estudio de los ciclos pasados. Para ello hemos investigado sobre el clima de Barcelona, sus variaciones históricas y las influencias - más recientes - del cambio climático. Para aproximar-nos al futuro hemos estudiado los datos dividiéndolos en dos grupos: de 1786-1990 y 1990-2020.

Tanto en el primer período largo de tiempo cómo en el segundo, y considerando las características del clima de Barcelona, los meses más lluviosos han sido y serán octubre, septiembre, noviembre y abril

Creemos que el ejercicio requiere una hipótesis más definida y un período de predicción determinado. Aún para el caso y sin hipótesis definida, trataremos de determinar cuándo se prevén más lluvias durante los próximos 3 años. Dado que el ejercicio de regresión lineal no ha sido un gran éxito no podemos apoyarnos en sus datos para las predicciones. No obstante, gracias a otros trabajos parecidos, a la información de expertos y nuestra própia exploración trataremos de llegar a algunas conclusiones:

1. Tomando como referentes de clima actual y próximo los datos de los último teinta años, poder

2. El análisis de los clusters del modelo no-supervisado nos ha acercado a los datos y nos ha

3. El modelo supervisado deja mucho que desear con el MAE más bajo de 23.08, un margen de erro

Para incrementar las predicciones de precipitaciones recomendamos seguir el artículo “Rainfall forecasting model using machine learning methods: Case study Terengganu, Malaysia”, donde llegano a conseguir un R2 de 0.999 en la predicción de precipitaciones con modelos supervisados de regresión lineal.

Nos habría gustado profundizar en el estudio de los outliers, así como en el de períodos más reducidos de tiempo, cada 3 o cinco años y el estudio de meses por separado. Hubiéramos probado otros algoritmos de clusterización y habríamos trabajado más el feature engineering: obtención de nuevas variables y datos para aumentar el R2 score.

Fuentes: 1. https://www.sciencedirect.com/science/article/pii/S2090447920302069