Actividad | K-Means

Integrantes

Alumno: **Erick de Jesus Hernández Cerecedo**
Matricula: **A01066428**

Información del Curso

Nombre: **Ciencia y analítica de datos**
Profesor: **Jobish Vallikavungal Devassia**

Fechas: **Martes 9 de noviembre de 2022**

In [1]:
```
# Requisitos iniciales
! pip install qeds fiona geopandas xgboost gensim folium pyLDAvis descartes
```

```
Collecting qeds
  Downloading qeds-0.7.0.tar.gz (24 kB)
  Preparing metadata (setup.py) ... done
Collecting fiona
  Downloading Fiona-1.8.22-cp310-cp310-macosx_10_10_x86_64.whl (26.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 26.5/26.5 MB 189.0 kB/s eta 0:00:0000:0100:
04
Collecting geopandas
  Downloading geopandas-0.12.1-py3-none-any.whl (1.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.1/1.1 MB 240.1 kB/s eta 0:00:0000:0100:0
1
Collecting xgboost
  Downloading xgboost-1.7.1-py3-none-macosx_10_15_x86_64.macosx_11_0_x86_64.macosx_12_0_
x86_64.whl (1.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 259.5 kB/s eta 0:00:0000:0100:0
1
Collecting gensim
  Downloading gensim-4.2.0-cp310-cp310-macosx_10_9_universal2.whl (24.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.4/24.4 MB 49.0 kB/s eta 0:00:0000:0100:
08
Collecting folium
  Downloading folium-0.13.0-py2.py3-none-any.whl (96 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 96.5/96.5 kB 24.1 kB/s eta 0:00:00a 0:00:0
1
Collecting pyLDAvis
  Downloading pyLDAvis-3.3.1.tar.gz (1.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.7/1.7 MB 39.9 kB/s eta 0:00:0000:0100:02
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Installing backend dependencies ... done
  Preparing metadata (pyproject.toml) ... done
Collecting descartes
  Downloading descartes-1.1.0-py3-none-any.whl (5.8 kB)
Requirement already satisfied: pandas in /Library/Frameworks/Python.framework/Versions/
3.10/lib/python3.10/site-packages (from qeds) (1.4.2)
Requirement already satisfied: requests in /Library/Frameworks/Python.framework/Version
s/3.10/lib/python3.10/site-packages (from qeds) (2.28.1)
Collecting quandl
  Downloading Quandl-3.7.0-py2.py3-none-any.whl (26 kB)
Requirement already satisfied: scipy in /Library/Frameworks/Python.framework/Versions/3.
10/lib/python3.10/site-packages (from qeds) (1.9.1)
```

```
Requirement already satisfied: numpy in /Library/Frameworks/Python.framework/Versions/3.
10/lib/python3.10/site-packages (from qeds) (1.22.4)
Collecting quantecon
  Downloading quantecon-0.5.3-py3-none-any.whl (179 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 179.5/179.5 kB 48.7 kB/s eta 0:00:00a 0:00:0
1
Requirement already satisfied: matplotlib in /Library/Frameworks/Python.framework/Versio
ns/3.10/lib/python3.10/site-packages (from qeds) (3.6.0)
Collecting pyarrow
  Downloading pyarrow-10.0.0-cp310-cp310-macosx_10_14_x86_64.whl (24.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 137.6 kB/s eta 0:00:0000:0100:
06
Collecting openpyxl
  Downloading openpyxl-3.0.10-py2.py3-none-any.whl (242 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 242.1/242.1 kB 286.7 kB/s eta 0:00:00a 0:00:0
1
Collecting plotly
  Downloading plotly-5.11.0-py2.py3-none-any.whl (15.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 15.3/15.3 MB 81.7 kB/s eta 0:00:0000:0100:
06
Collecting pandas_datareader
  Downloading pandas_datareader-0.10.0-py3-none-any.whl (109 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 109.5/109.5 kB 596.3 kB/s eta 0:00:00a 0:00:0
1
Requirement already satisfied: scikit-learn in /Library/Frameworks/Python.framework/Vers
ions/3.10/lib/python3.10/site-packages (from qeds) (1.1.2)
Requirement already satisfied: seaborn in /Library/Frameworks/Python.framework/Versions/
3.10/lib/python3.10/site-packages (from qeds) (0.12.0)
Collecting statsmodels
  Downloading statsmodels-0.13.5-cp310-cp310-macosx_10_9_x86_64.whl (9.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9.7/9.7 MB 136.3 kB/s eta 0:00:0000:0100:0
2
Collecting click>=4.0
  Downloading click-8.1.3-py3-none-any.whl (96 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 96.6/96.6 kB 569.2 kB/s eta 0:00:00a 0:00:0
1
Collecting click-plugins>=1.0
  Downloading click_plugins-1.1.1-py2.py3-none-any.whl (7.5 kB)
Collecting munch
  Downloading munch-2.5.0-py2.py3-none-any.whl (10 kB)
Requirement already satisfied: six>=1.7 in /Library/Frameworks/Python.framework/Version
s/3.10/lib/python3.10/site-packages (from fiona) (1.16.0)
Requirement already satisfied: attrs>=17 in /Library/Frameworks/Python.framework/Version
s/3.10/lib/python3.10/site-packages (from fiona) (22.1.0)
Collecting cligj>=0.5
  Downloading cligj-0.7.2-py3-none-any.whl (7.1 kB)
Requirement already satisfied: setuptools in /Library/Frameworks/Python.framework/Versio
ns/3.10/lib/python3.10/site-packages (from fiona) (58.1.0)
Requirement already satisfied: certifi in /Library/Frameworks/Python.framework/Versions/
3.10/lib/python3.10/site-packages (from fiona) (2022.9.24)
Requirement already satisfied: packaging in /Users/hunter/Library/Python/3.10/lib/pytho
n/site-packages (from geopandas) (21.3)
Collecting pyproj>=2.6.1.post1
  Downloading pyproj-3.4.0-cp310-cp310-macosx_10_9_x86_64.whl (8.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.0/8.0 MB 37.9 kB/s eta 0:00:0000:0100:07
m
Collecting shapely>=1.7
  Downloading Shapely-1.8.5.post1-cp310-cp310-macosx_10_9_x86_64.whl (1.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.2/1.2 MB 36.4 kB/s eta 0:00:0000:0100:02
Collecting smart-open>=1.8.1
  Downloading smart_open-6.2.0-py3-none-any.whl (58 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.6/58.6 kB 286.2 kB/s eta 0:00:00a 0:00:0
1
Collecting branca>=0.3.0
  Downloading branca-0.6.0-py3-none-any.whl (24 kB)
Requirement already satisfied: jinja2>=2.9 in /Library/Frameworks/Python.framework/Versi
```

```
ons/3.10/lib/python3.10/site-packages (from folium) (3.1.2)
Collecting sklearn
  Downloading sklearn-0.0.post1.tar.gz (3.6 kB)
  Preparing metadata (setup.py) ... done
Collecting numexpr
  Downloading numexpr-2.8.4-cp310-cp310-macosx_10_9_x86_64.whl (99 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100.0/100.0 kB 335.8 kB/s eta 0:00:00a 0:00:0
1
Collecting future
  Downloading future-0.18.2.tar.gz (829 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 829.2/829.2 kB 460.6 kB/s eta 0:00:0000:0100:
01
  Preparing metadata (setup.py) ... done
Collecting funcy
  Downloading funcy-1.17-py2.py3-none-any.whl (33 kB)
Requirement already satisfied: joblib in /Library/Frameworks/Python.framework/Versions/
3.10/lib/python3.10/site-packages (from pyLDAvis) (1.2.0)
Requirement already satisfied: MarkupSafe>=2.0 in /Library/Frameworks/Python.framework/V
ersions/3.10/lib/python3.10/site-packages (from jinja2>=2.9->folium) (2.1.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /Library/Frameworks/Python.fram
ework/Versions/3.10/lib/python3.10/site-packages (from pandas->qeds) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /Library/Frameworks/Python.framework/Vers
ions/3.10/lib/python3.10/site-packages (from pandas->qeds) (2022.1)
Requirement already satisfied: pillow>=6.2.0 in /Library/Frameworks/Python.framework/Ver
sions/3.10/lib/python3.10/site-packages (from matplotlib->qeds) (9.2.0)
Requirement already satisfied: contourpy>=1.0.1 in /Library/Frameworks/Python.framework/
Versions/3.10/lib/python3.10/site-packages (from matplotlib->qeds) (1.0.5)
Requirement already satisfied: kiwisolver>=1.0.1 in /Library/Frameworks/Python.framewor
k/Versions/3.10/lib/python3.10/site-packages (from matplotlib->qeds) (1.4.4)
Requirement already satisfied: cycler>=0.10 in /Library/Frameworks/Python.framework/Vers
ions/3.10/lib/python3.10/site-packages (from matplotlib->qeds) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /Library/Frameworks/Python.framewor
k/Versions/3.10/lib/python3.10/site-packages (from matplotlib->qeds) (4.37.3)
Requirement already satisfied: pyparsing>=2.2.1 in /Users/hunter/Library/Python/3.10/li
b/python/site-packages (from matplotlib->qeds) (3.0.9)
Collecting et-xmlfile
  Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Requirement already satisfied: lxml in /Library/Frameworks/Python.framework/Versions/3.1
0/lib/python3.10/site-packages (from pandas_datareader->qeds) (4.9.1)
Requirement already satisfied: charset-normalizer<3,>=2 in /Library/Frameworks/Python.fr
amework/Versions/3.10/lib/python3.10/site-packages (from requests->qeds) (2.1.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /Library/Frameworks/Python.frame
work/Versions/3.10/lib/python3.10/site-packages (from requests->qeds) (1.26.12)
Requirement already satisfied: idna<4,>=2.5 in /Library/Frameworks/Python.framework/Vers
ions/3.10/lib/python3.10/site-packages (from requests->qeds) (3.4)
Collecting tenacity>=6.2.0
  Downloading tenacity-8.1.0-py3-none-any.whl (23 kB)
Collecting more-itertools
  Downloading more_itertools-9.0.0-py3-none-any.whl (52 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 52.8/52.8 kB 553.9 kB/s eta 0:00:00 0:00:01
Collecting inflection>=0.3.1
  Downloading inflection-0.5.1-py2.py3-none-any.whl (9.5 kB)
Collecting numba
  Downloading numba-0.56.4-cp310-cp310-macosx_10_14_x86_64.whl (2.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.4/2.4 MB 462.8 kB/s eta 0:00:0000:0100:0
1
Collecting sympy
  Downloading sympy-1.11.1-py3-none-any.whl (6.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.5/6.5 MB 480.0 kB/s eta 0:00:0000:0100:0
1
Requirement already satisfied: threadpoolctl>=2.0.0 in /Library/Frameworks/Python.framew
ork/Versions/3.10/lib/python3.10/site-packages (from scikit-learn->qeds) (3.1.0)
Collecting patsy>=0.5.2
  Downloading patsy-0.5.3-py2.py3-none-any.whl (233 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 233.8/233.8 kB 409.6 kB/s eta 0:00:00a 0:00:0
1
```

```
Collecting llvmlite<0.40,>=0.39.0dev0
  Downloading llvmlite-0.39.1-cp310-cp310-macosx_10_9_x86_64.whl (25.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 25.5/25.5 MB 561.8 kB/s eta 0:00:0000:0100:
02
Collecting mpmath>=0.19
  Downloading mpmath-1.2.1-py3-none-any.whl (532 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 532.6/532.6 kB 652.6 kB/s eta 0:00:0000:0100:
01
Building wheels for collected packages: qeds, pyLDAvis, future, sklearn
  Building wheel for qeds (setup.py) ... done
  Created wheel for qeds: filename=qeds-0.7.0-py3-none-any.whl size=27813 sha256=b62e11e
2dd251a00bd1799f2c0ccb8b17294099a07807b8a07f41cdbd9e13b10
  Stored in directory: /Users/hunter/Library/Caches/pip/wheels/c5/f4/a5/e6fb5a90bb74190e
a193a19e666199e16fb8fef65a15babf3d
  Building wheel for pyLDAvis (pyproject.toml) ... done
  Created wheel for pyLDAvis: filename=pyLDAvis-3.3.1-py2.py3-none-any.whl size=136882 s
ha256=2416f6c6b05e909bbfb55a3ae1f76cf7a744075c9965dafa6db8e7e5bb6bbc06
  Stored in directory: /Users/hunter/Library/Caches/pip/wheels/f0/26/9e/a6d11f6155723bb4
7e608728aac50d0f0fa0e87226659bd5d8
  Building wheel for future (setup.py) ... done
  Created wheel for future: filename=future-0.18.2-py3-none-any.whl size=491070 sha256=0
dfd1e786d63cd4a23b2f869526e5732329d00faf323a3cf37c49c4e0412c77d
  Stored in directory: /Users/hunter/Library/Caches/pip/wheels/dc/16/09/eb08b4e34e6b638f
113d2018cf0b22de1d8dca22a3a71873f7
  Building wheel for sklearn (setup.py) ... done
  Created wheel for sklearn: filename=sklearn-0.0.post1-py3-none-any.whl size=2344 sha25
6=a3c8496f5abf431fc267bf348c8a019b750a77e71a38c913c339c7d772d6c78b
  Stored in directory: /Users/hunter/Library/Caches/pip/wheels/db/9f/0b/772886b624f84c13
8a5febb6966c89d374ab58c62bd65d109e
Successfully built qeds pyLDAvis future sklearn
Installing collected packages: sklearn, mpmath, funcy, tenacity, sympy, smart-open, shap
ely, pyproj, pyarrow, patsy, numexpr, munch, more-itertools, llvmlite, inflection, futur
e, et-xmlfile, click, xgboost, plotly, openpyxl, numba, gensim, cligj, click-plugins, br
anca, statsmodels, quantecon, quandl, pyLDAvis, pandas_datareader, folium, fiona, descar
tes, qeds, geopandas
Successfully installed branca-0.6.0 click-8.1.3 click-plugins-1.1.1 cligj-0.7.2 descarte
s-1.1.0 et-xmlfile-1.1.0 fiona-1.8.22 folium-0.13.0 funcy-1.17 future-0.18.2 gensim-4.2.
0 geopandas-0.12.1 inflection-0.5.1 llvmlite-0.39.1 more-itertools-9.0.0 mpmath-1.2.1 mu
nch-2.5.0 numba-0.56.4 numexpr-2.8.4 openpyxl-3.0.10 pandas_datareader-0.10.0 patsy-0.5.
3 plotly-5.11.0 pyLDAvis-3.3.1 pyarrow-10.0.0 pyproj-3.4.0 qeds-0.7.0 quandl-3.7.0 quant
econ-0.5.3 shapely-1.8.5.post1 sklearn-0.0.post1 smart-open-6.2.0 statsmodels-0.13.5 sym
py-1.11.1 tenacity-8.1.0 xgboost-1.7.1

[notice] A new release of pip available: 22.3 -> 22.3.1
[notice] To update, run: pip install --upgrade pip
```

In [1]:

```
# Importar librerias
import pandas as pd
import numpy as np
from tqdm import tqdm
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import geopandas


import ssl

try:
    _create_unverified_https_context = ssl._create_unverified_context
except AttributeError:
    # Legacy Python that doesn't verify HTTPS certificates by default
    pass
else:
```

```
    # Handle target environment that doesn't support HTTPS verification
    ssl._create_default_https_context = _create_unverified_https_context
```

In [2]:
```
# Consumo de datos
url="https://raw.githubusercontent.com/marypazrf/bdd/main/target-locations.csv"
df=pd.read_csv(url)
df.head()
```

Out[2]:

	name	latitude	longitude	address	phone	website
0	Alabaster	33.224225	-86.804174	250 S Colonial Dr, Alabaster, AL 35007-4657	205-564-2608	https://www.target.com/sl/alabaster/2276
1	Bessemer	33.334550	-86.989778	4889 Promenade Pkwy, Bessemer, AL 35022-7305	205-565-3760	https://www.target.com/sl/bessemer/2375
2	Daphne	30.602875	-87.895932	1698 US Highway 98, Daphne, AL 36526-4252	251-621-3540	https://www.target.com/sl/daphne/1274
3	Decatur	34.560148	-86.971559	1235 Point Mallard Pkwy SE, Decatur, AL 35601-...	256-898-3036	https://www.target.com/sl/decatur/2084
4	Dothan	31.266061	-85.446422	4601 Montgomery Hwy, Dothan, AL 36303-1522	334-340-1112	https://www.target.com/sl/dothan/1468

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1839 entries, 0 to 1838
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   name       1839 non-null   object 
 1   latitude   1839 non-null   float64
 2   longitude  1839 non-null   float64
 3   address    1839 non-null   object 
 4   phone      1839 non-null   object 
 5   website    1839 non-null   object 
dtypes: float64(2), object(4)
memory usage: 86.3+ KB
```

Definición de Latitud y Longitud

Latitud Es la distancia en grados, minutos y segundos que hay con respecto al paralelo principal, que es el ecuador (0°). La latitud puede ser norte y sur.

Longitud: Es la distancia en grados, minutos y segundos que hay con respecto al meridiano principal, que es el meridiano de Greenwich (0°).La longitud puede ser este y oeste.

In [4]:
```
latlong=df[["latitude","longitude"]]
latlong
```

Out[4]:

	latitude	longitude
0	33.224225	-86.804174
1	33.334550	-86.989778
2	30.602875	-87.895932
3	34.560148	-86.971559

4	31.266061	-85.446422
...	...	...
1834	43.034293	-88.176840
1835	42.989604	-88.259806
1836	42.846799	-106.264166
1837	41.162019	-104.800048
1838	43.469617	-110.789456

1839 rows × 2 columns

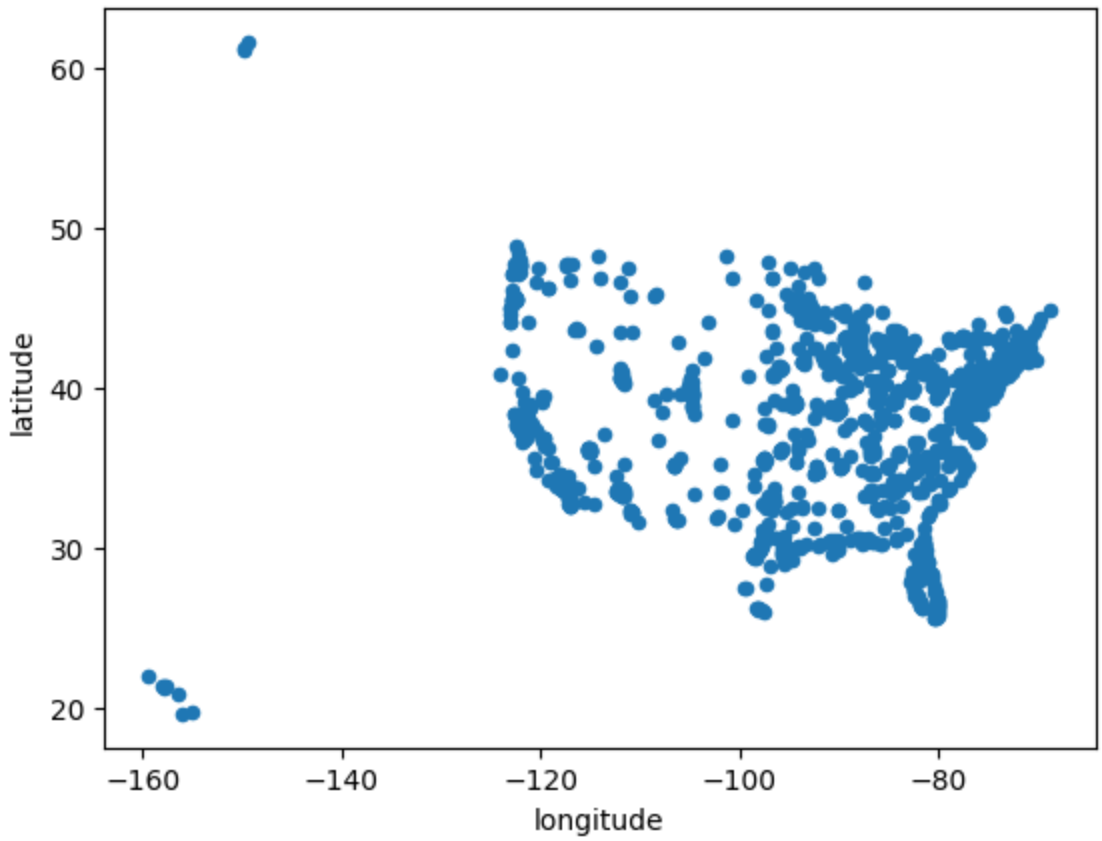
¡Visualizemos los datos!, para empezar a notar algún patron.

A simple vista pudieramos pensar que tenemos algunos datos atípicos u outliers, pero .... no es así, simplemente esta grafica no nos está dando toda la información.

In [5]:
```
#extrae los datos interesantes
latlong.plot.scatter( "longitude","latitude")
```

```
/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-packages/pandas/plotting/_matplotlib/core.py:1114: UserWarning: No data for colormapping provided via 'c'. Parameters 'cmap' will be ignored
  scatter = ax.scatter(
```

Out[5]:
```
<AxesSubplot: xlabel='longitude', ylabel='latitude'>
```



In [6]:
```
latlong.describe()
```

Out[6]:

	latitude	longitude
count	1839.000000	1839.000000
mean	37.791238	-91.986881

std	5.272299	16.108046
min	19.647855	-159.376962
25%	33.882605	-98.268828
50%	38.955432	-87.746346
75%	41.658341	-80.084833
max	61.577919	-68.742331

Para entender un poco más, nos auxiliaremos de una librería para graficar datos geográficos. Esto nos ayudara a tener un mejor entendimiento de ellos.

In [6]:
```
import geopandas as gpd
import matplotlib.pyplot as plt
import pandas as pd

from shapely.geometry import Point

%matplotlib inline
# activate plot theme
import qeds
qeds.themes.mpl_style();
```

In [7]:
```
df["Coordinates"] = list(zip(df.longitude, df.latitude))
df["Coordinates"] = df["Coordinates"].apply(Point)
df.head()
```

Out[7]:

	name	latitude	longitude	address	phone	website	
0	Alabaster	33.224225	-86.804174	250 S Colonial Dr, Alabaster, AL 35007-4657	205-564-2608	https://www.target.com/sl/alabaster/2276	(-86.8041
1	Bessemer	33.334550	-86.989778	4889 Promenade Pkwy, Bessemer, AL 35022-7305	205-565-3760	https://www.target.com/sl/bessemer/2375	(-86.9897
2	Daphne	30.602875	-87.895932	1698 US Highway 98, Daphne, AL 36526-4252	251-621-3540	https://www.target.com/sl/daphne/1274	(-87.8959
3	Decatur	34.560148	-86.971559	1235 Point Mallard Pkwy SE, Decatur, AL 35601-...	256-898-3036	https://www.target.com/sl/decatur/2084	POINT (
4	Dothan	31.266061	-85.446422	4601 Montgomery Hwy, Dothan, AL 36303-1522	334-340-1112	https://www.target.com/sl/dothan/1468	POINT (-

In [8]:
```
gdf = gpd.GeoDataFrame(df, geometry="Coordinates")
gdf.head()
```

Out[8]:

	name	latitude	longitude	address	phone	website	Coordinat
0	Alabaster	33.224225	-86.804174	250 S Colonial Dr, Alabaster, AL 35007-4657	205-564-2608	https://www.target.com/sl/alabaster/2276	POI (-86.804 33.2242
1	Bessemer	33.334550	-86.989778	4889 Promenade Pkwy, Bessemer, AL 35022-7305	205-565-3760	https://www.target.com/sl/bessemer/2375	POI (-86.989 33.3345
2	Daphne	30.602875	-87.895932	1698 US Highway 98, Daphne, AL 36526-4252	251-621-3540	https://www.target.com/sl/daphne/1274	POI (-87.895 30.6028
3	Decatur	34.560148	-86.971559	1235 Point Mallard Pkwy SE, Decatur, AL 35601-...	256-898-3036	https://www.target.com/sl/decatur/2084	POI (-86.971 34.5601
4	Dothan	31.266061	-85.446422	4601 Montgomery Hwy, Dothan, AL 36303-1522	334-340-1112	https://www.target.com/sl/dothan/1468	POI (-85.446 31.2660

In [9]:
```
#mapa

world = gpd.read_file(gpd.datasets.get_path("naturalearth_lowres"))
world = world.set_index("iso_a3")

world.head()
```

Out[9]:

iso_a3	pop_est	continent	name	gdp_md_est	geometry
FJI	889953.0	Oceania	Fiji	5496	MULTIPOLYGON (((180.00000 -16.06713, 180.00000...
TZA	58005463.0	Africa	Tanzania	63177	POLYGON ((33.90371 -0.95000, 34.07262 -1.05982...
ESH	603253.0	Africa	W. Sahara	907	POLYGON ((-8.66559 27.65643, -8.66512 27.58948...
CAN	37589262.0	North America	Canada	1736425	MULTIPOLYGON (((-122.84000 49.00000, -122.9742...
USA	328239523.0	North America	United States of America	21433226	MULTIPOLYGON (((-122.84000 49.00000, -120.0000...

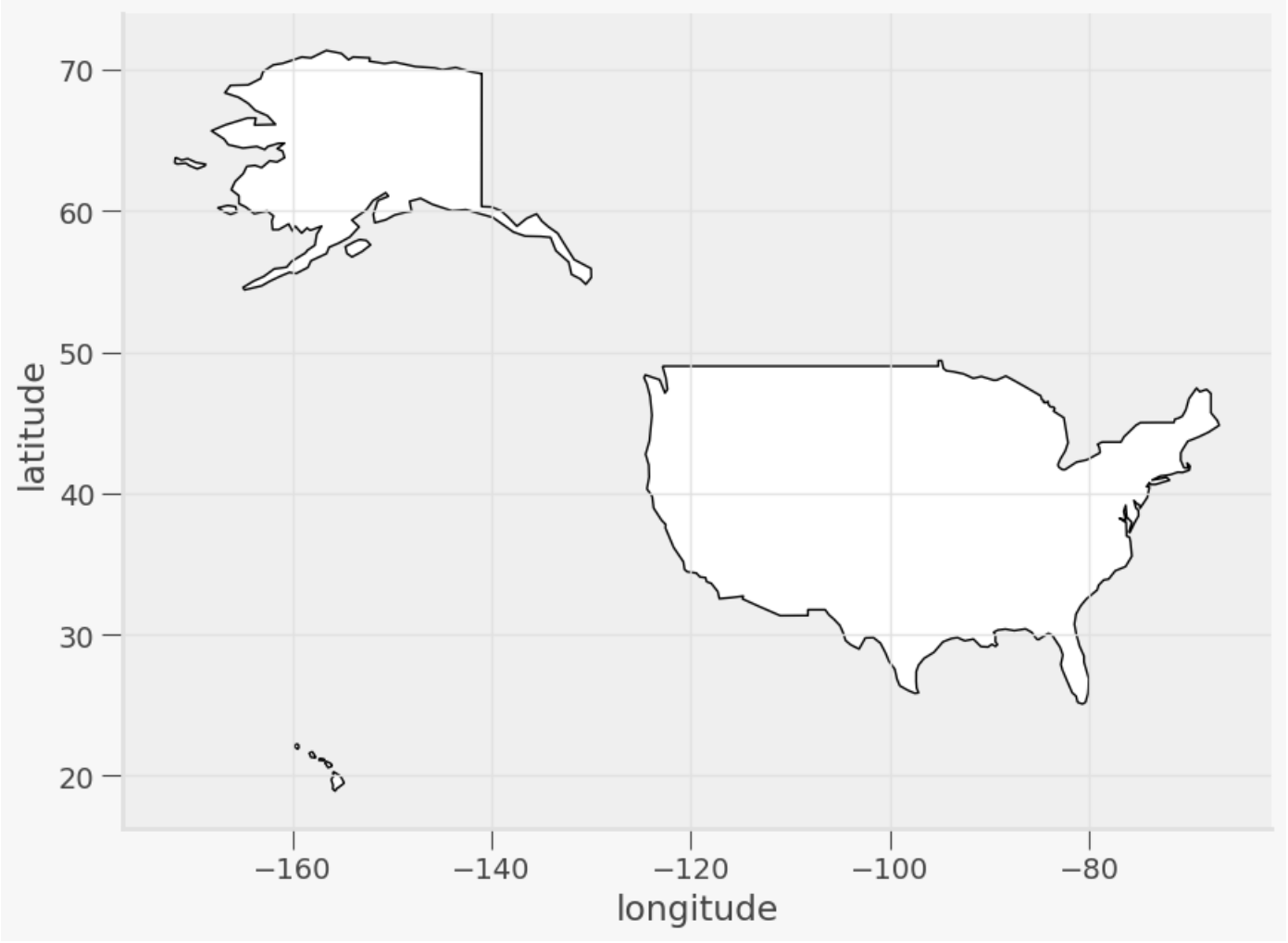
In [10]:
```
#graficar el mapa
world.name.unique()
```

Out[10]:
```
array(['Fiji', 'Tanzania', 'W. Sahara', 'Canada',
       'United States of America', 'Kazakhstan', 'Uzbekistan',
       'Papua New Guinea', 'Indonesia', 'Argentina', 'Chile',
       'Dem. Rep. Congo', 'Somalia', 'Kenya', 'Sudan', 'Chad', 'Haiti',
       'Dominican Rep.', 'Russia', 'Bahamas', 'Falkland Is.', 'Norway',
       'Greenland', 'Fr. S. Antarctic Lands', 'Timor-Leste',
```

```
       'South Africa', 'Lesotho', 'Mexico', 'Uruguay', 'Brazil',
       'Bolivia', 'Peru', 'Colombia', 'Panama', 'Costa Rica', 'Nicaragua',
       'Honduras', 'El Salvador', 'Guatemala', 'Belize', 'Venezuela',
       'Guyana', 'Suriname', 'France', 'Ecuador', 'Puerto Rico',
       'Jamaica', 'Cuba', 'Zimbabwe', 'Botswana', 'Namibia', 'Senegal',
       'Mali', 'Mauritania', 'Benin', 'Niger', 'Nigeria', 'Cameroon',
       'Togo', 'Ghana', "Côte d'Ivoire", 'Guinea', 'Guinea-Bissau',
       'Liberia', 'Sierra Leone', 'Burkina Faso', 'Central African Rep.',
       'Congo', 'Gabon', 'Eq. Guinea', 'Zambia', 'Malawi', 'Mozambique',
       'eSwatini', 'Angola', 'Burundi', 'Israel', 'Lebanon', 'Madagascar',
       'Palestine', 'Gambia', 'Tunisia', 'Algeria', 'Jordan',
       'United Arab Emirates', 'Qatar', 'Kuwait', 'Iraq', 'Oman',
       'Vanuatu', 'Cambodia', 'Thailand', 'Laos', 'Myanmar', 'Vietnam',
       'North Korea', 'South Korea', 'Mongolia', 'India', 'Bangladesh',
       'Bhutan', 'Nepal', 'Pakistan', 'Afghanistan', 'Tajikistan',
       'Kyrgyzstan', 'Turkmenistan', 'Iran', 'Syria', 'Armenia', 'Sweden',
       'Belarus', 'Ukraine', 'Poland', 'Austria', 'Hungary', 'Moldova',
       'Romania', 'Lithuania', 'Latvia', 'Estonia', 'Germany', 'Bulgaria',
       'Greece', 'Turkey', 'Albania', 'Croatia', 'Switzerland',
       'Luxembourg', 'Belgium', 'Netherlands', 'Portugal', 'Spain',
       'Ireland', 'New Caledonia', 'Solomon Is.', 'New Zealand',
       'Australia', 'Sri Lanka', 'China', 'Taiwan', 'Italy', 'Denmark',
       'United Kingdom', 'Iceland', 'Azerbaijan', 'Georgia',
       'Philippines', 'Malaysia', 'Brunei', 'Slovenia', 'Finland',
       'Slovakia', 'Czechia', 'Eritrea', 'Japan', 'Paraguay', 'Yemen',
       'Saudi Arabia', 'Antarctica', 'N. Cyprus', 'Cyprus', 'Morocco',
       'Egypt', 'Libya', 'Ethiopia', 'Djibouti', 'Somaliland', 'Uganda',
       'Rwanda', 'Bosnia and Herz.', 'North Macedonia', 'Serbia',
       'Montenegro', 'Kosovo', 'Trinidad and Tobago', 'S. Sudan'],
      dtype=object)
```

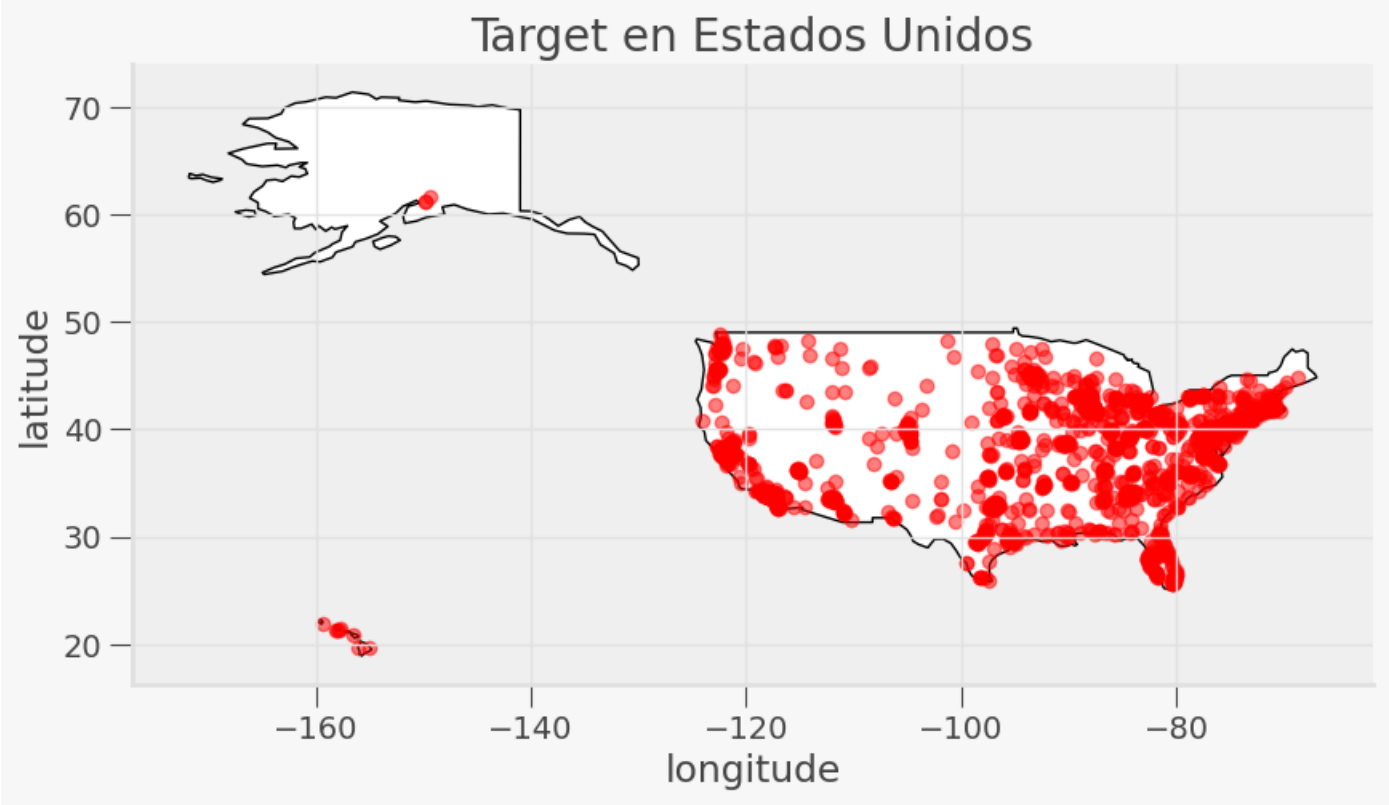
In [12]:

```
import matplotlib.pyplot as plt
plt.rcParams["font.family"] = "sans-serif"

fig, gax = plt.subplots(figsize=(10,10))

# By only plotting rows in which the continent is 'South America' we only plot SA.
world.query("name == 'United States of America'").plot(ax=gax, edgecolor='black',color='

# By the way, if you haven't read the book 'longitude' by Dava Sobel, you should...
gax.set_xlabel('longitude')
gax.set_ylabel('latitude')

gax.spines['top'].set_visible(False)
gax.spines['right'].set_visible(False)
```



In [13]:

```
# Step 3: Plot the cities onto the map
# We mostly use the code from before --- we still want the country borders plotted --- a
# add a command to plot the cities
fig, gax = plt.subplots(figsize=(10,10))

# By only plotting rows in which the continent is 'South America' we only plot, well,
# South America.
world.query("name == 'United States of America'").plot(ax = gax, edgecolor='black', colo

# This plot the cities. It's the same syntax, but we are plotting from a different GeoDa
# I want the cities as pale red dots.
gdf.plot(ax=gax, color='red', alpha = 0.5)

gax.set_xlabel('longitude')
gax.set_ylabel('latitude')
gax.set_title('Target en Estados Unidos')

gax.spines['top'].set_visible(False)
gax.spines['right'].set_visible(False)

plt.show()
```



¿qué tal ahora?, tiene mayor sentido verdad, entonces los datos lejanos no eran atípicos, de aquí la importancia de ver los datos con el tipo de gráfica correcta.

Ahora sí, implementa K means a los datos de latitud y longitud :) y encuentra donde colocar los almacenes.

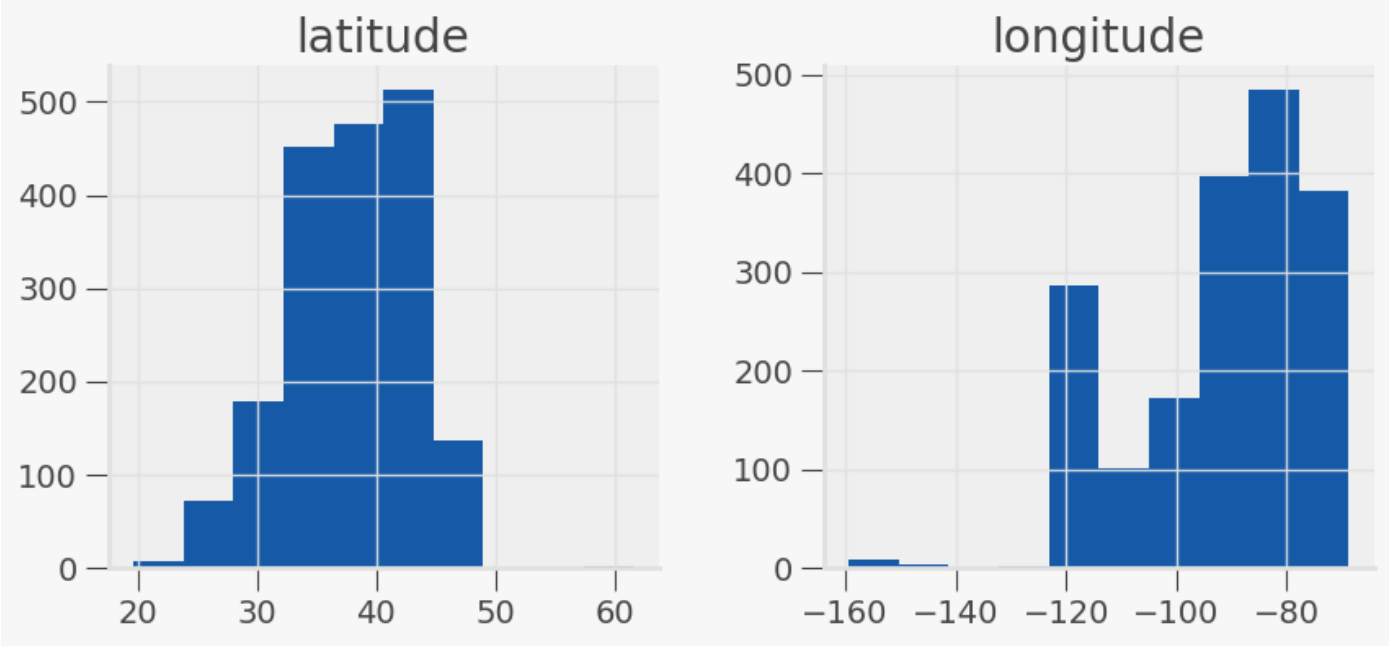
Nota: si te llama la atención implementar alguna otra visualización con otra librería, lo puedes hacer, no hay restricciones.

1. Encuentra el numero ideal de almacenes, justifica tu respuesta:

- Encuentra las latitudes y longitudes de los almacenes ¿qué ciudad es? ¿a cuantas tiendas va surtir?, ¿sabes a que distancia estará?, ¿Cómo elegiste el número de almacenes?, justifica tu respuesta técnicamente.

In [28]:
```
df.hist(figsize=(10, 4))
```

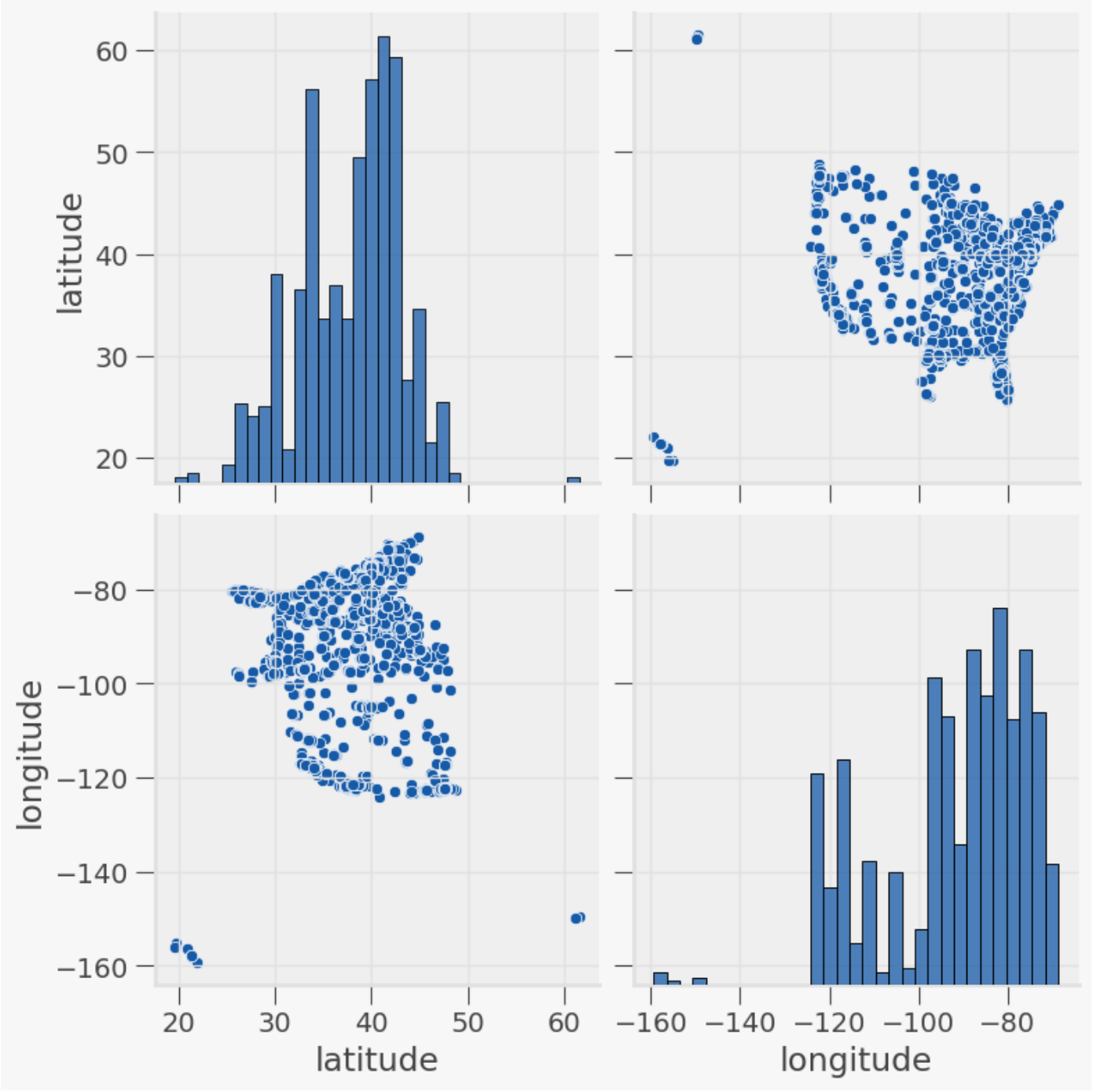
Out[28]:
```
array([[<AxesSubplot: title={'center': 'latitude'}>,
        <AxesSubplot: title={'center': 'longitude'}>]], dtype=object)
```



In [29]:
```
import seaborn as sb
sb.pairplot(df.dropna(),size=4,kind='scatter')
```

```
/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-packages/seaborn/
axisgrid.py:2095: UserWarning: The `size` parameter has been renamed to `height`; please
update your code.
  warnings.warn(msg, UserWarning)
```

Out[29]:
```
<seaborn.axisgrid.PairGrid at 0x13c20b310>
```



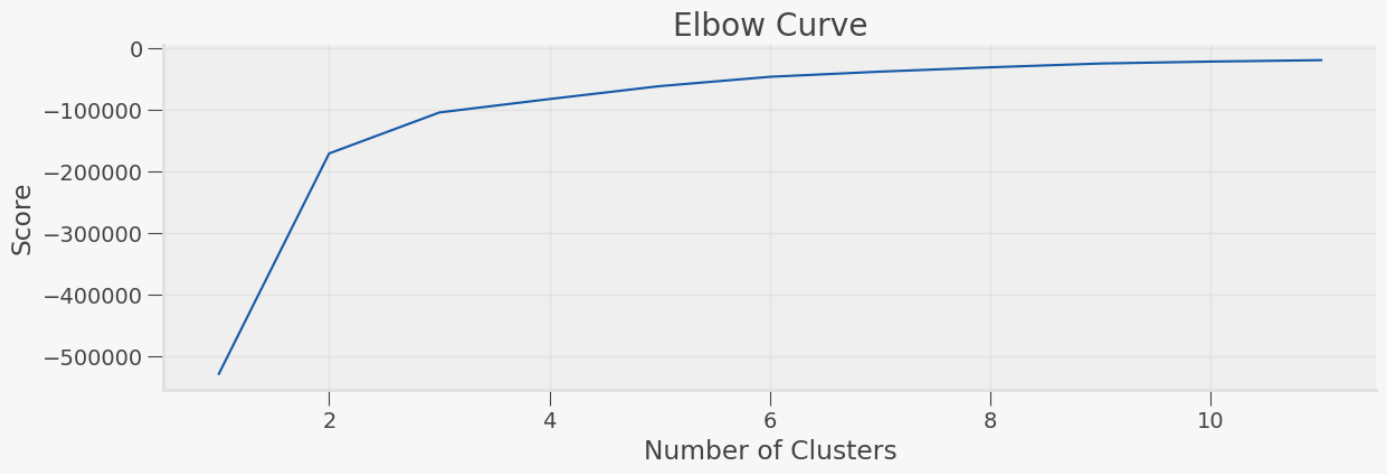
In [42]:

```
from sklearn.cluster import KMeans

# Elegimos un rango de puntos que seran evaluados
nclusters = range(1,12) # arbitrariamente decidimos que el número de clusters, es decir,

# Modelos KMeans con diferentes Clusters
kmeans = [KMeans(n_clusters=i) for i in nclusters]
print("Kmeans: ", kmeans)

# Generamos un score para cada cluster con los datos de entrada X
score = [kmeans[i].fit(latlong).score(latlong) for i in range(len(kmeans))]
print("\nScores: ", score)
```

```
Kmeans:  [KMeans(n_clusters=1), KMeans(n_clusters=2), KMeans(n_clusters=3), KMeans(n_clu
sters=4), KMeans(n_clusters=5), KMeans(n_clusters=6), KMeans(n_clusters=7), KMeans(), KM
eans(n_clusters=9), KMeans(n_clusters=10), KMeans(n_clusters=11)]

Scores:  [-527995.4430694163, -171146.62599564387, -104758.59758525781, -82987.937724474
7, -62084.01203320784, -46975.760861283256, -38595.75033719001, -31578.070195780958, -25
438.164852358885, -22300.9843379875, -20011.81961958983]
```

In [43]:
```
# Grafica "Curva codo"
plt.figure(figsize=(14, 4))
plt.plot(nclusters, score)
plt.xlabel('Number of Clusters')
plt.ylabel('Score')
plt.title('Elbow Curve')

# Numero de clusters ideal = 3
```

Out[43]:
```
Text(0.5, 1.0, 'Elbow Curve')
```



Adicionalmente, en el notebook notaras que al inicio exploramos los datos y los graficamos de manera simple, después nos auxiliamos de una librería de datos geográficos.

¿qué librerías nos pueden ayudar a graficar este tipo de datos? ¿Consideras importante que se grafique en un mapa?, ¿por qué? Agrega las conclusiones

In [58]:
```
# Creamos los clusters
kmeans = KMeans(n_clusters = 3)

# Entrenamos el modelo
kmeans.fit(latlong)

# Centroides
centroids = kmeans.cluster_centers_
print(centroids)
```

```
[[ 3.79858610e+01 -9.33153536e+01  2.00000000e+00]
 [ 3.74873420e+01 -1.18624473e+02  1.00000000e+00]
 [ 3.77849149e+01 -7.85610278e+01  3.10862447e-15]]
```

In [73]:
```
# Definimos las entradas en X
X = df[["longitude","latitude"]]

kmeans = KMeans(n_clusters=3).fit(X)
labels = kmeans.predict(X)

# Obtenemos los Centroides
centroides = kmeans.cluster_centers_
centroides
```

Out[73]:
```
array([[-118.62447332,   37.48734203],
       [ -78.56990807,   37.789554  ],
```

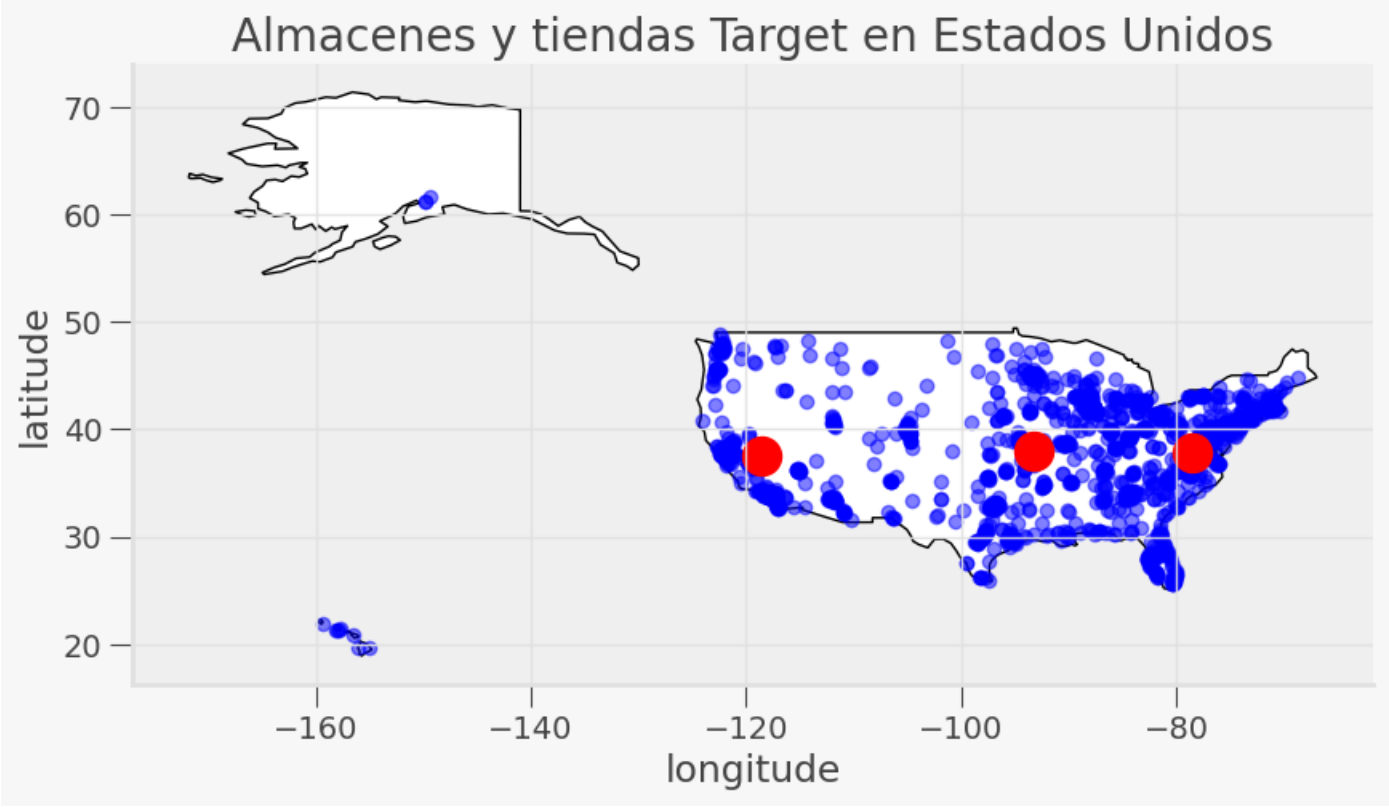
```
       [ -93.3271723 ,   37.98006261]])
```

In [74]:
```
# Agrupamos las cordenas en una sola columna para graficar corrctamente
cordenadas = pd.DataFrame(centroides)
cordenadas["Coordenadas"] = list(zip(cordenadas[0], cordenadas[1]))
cordenadas["Coordenadas"] = cordenadas["Coordenadas"].apply(Point)

# Convertimos el mapa a uno de tipo GeoDataFrame
geopanda_df= gpd.GeoDataFrame(cordenadas, geometry="Coordenadas")
geopanda_df
```

Out[74]:

	0	1	Coordenadas
0	-118.624473	37.487342	POINT (-118.62447 37.48734)
1	-78.569908	37.789554	POINT (-78.56991 37.78955)
2	-93.327172	37.980063	POINT (-93.32717 37.98006)

In [75]:
```
# Graficamos los clusters
fig, gax = plt.subplots(figsize=(10,10))

# By only plotting rows in which the continent is 'South America' we only plot, well,
# South America.
world.query("name == 'United States of America'").plot(ax = gax, edgecolor='black', colo

# This plot the cities. It's the same syntax, but we are plotting from a different GeoDa
# I want the cities as pale red dots.
gdf.plot(ax=gax, color='blue', alpha = 0.5) #Aqui grafica los datos originales
geopanda_df.plot(ax=gax, color='red', alpha = 1, markersize = 300) #Aqui grafica los dat

#De aqui para abajo es puro ploting busines
gax.set_xlabel('longitude')
gax.set_ylabel('latitude')
gax.set_title('Almacenes y tiendas Target en Estados Unidos')

gax.spines['top'].set_visible(False)
gax.spines['right'].set_visible(False)

plt.show()
```



In [76]:
```
# Cuantas tiendas tiene cada cluster
latlong['kmeans'] = kmeans.labels_
latlong.loc[:, 'kmeans'].value_counts()
```

```
/var/folders/6k/743ttrkx5pqf092zt1_xq4dr0000gn/T/ipykernel_4637/913729421.py:2: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  latlong['kmeans'] = kmeans.labels_
```

Out[76]:
```
1    826
2    628
0    385
Name: kmeans, dtype: int64
```

In [77]:
```
AlmacenA = str(geopanda_df[1][0]) + ", " + str(geopanda_df[0][0])
print('Coordenadas almacén A: ', AlmacenA)
AlmacenB = str(geopanda_df[1][1]) + ", " + str(geopanda_df[0][1])
print('Coordenadas de almacén B: ' , AlmacenB)
AlmacenC = str(geopanda_df[1][2]) + ", " + str(geopanda_df[0][2])
print('Coordenadas de almacén C:',AlmacenC)
```

```
Coordenadas almacén A:  37.48734203064935, -118.62447331844157
Coordenadas de almacén B:  37.789554004474006, -78.56990807484885
Coordenadas de almacén C: 37.98006260590112, -93.32717230430622
```