M212-Ufo

April 12, 2024

[26]:
```
import pandas as pd
import seaborn as sns
sns.set_theme()

import numpy as np

import matplotlib.pyplot as plt
import plotly.express as px

import math
from scipy import stats

from sklearn import cluster
from sklearn.preprocessing import MinMaxScaler
```

UFO dataset downloaded from: https://github.com/rfordatascience/tidytuesday/tree/2e9bd5a67e09b14d01f616b0 06-25

1 Initial data exploration

Let's start by looking at some of the data in the UFO dataset.

[47]:
```
df = pd.read_csv("ufo_sightings.csv")
df = df[["date_time", "ufo_shape", "encounter_length", "latitude", "longitude"]]

df = df.dropna()
df.head()
```

[47]:

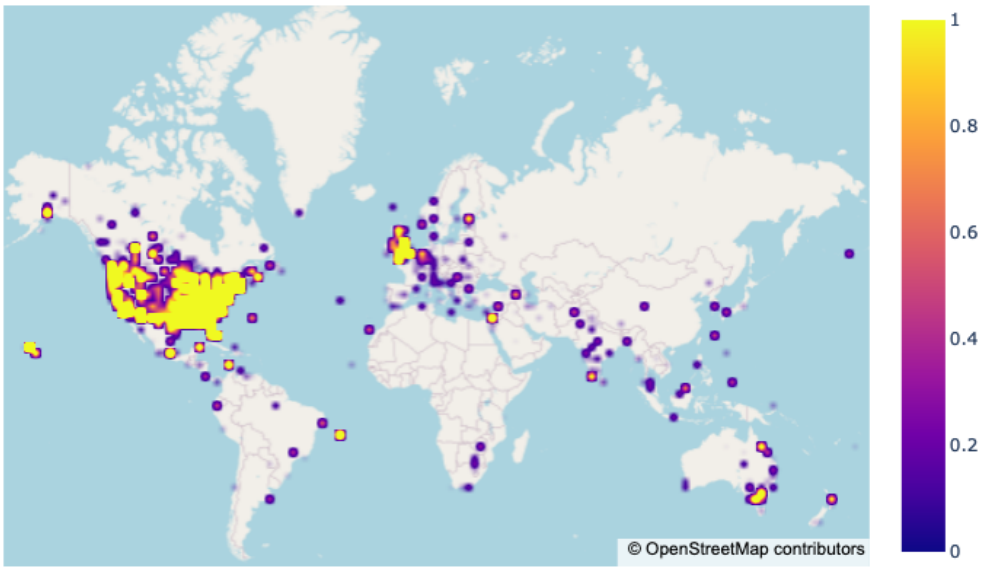
	date_time	ufo_shape	encounter_length	latitude	longitude
0	10/10/1949 20:30	cylinder	2700.0	29.883056	-97.941111
1	10/10/1949 21:00	light	7200.0	29.384210	-98.581082
2	10/10/1955 17:00	circle	20.0	53.200000	-2.916667
3	10/10/1956 21:00	circle	20.0	28.978333	-96.645833
4	10/10/1960 20:00	light	900.0	21.418056	-157.803611

1.1 Question

Where are the locations of the UFO sightings in this dataset? How might the dataset be biased by location? What are three possible reasons for this bias?

[89]:
```
# The original dataset included states and cities, but those won't apply␣
 ↪outside the US
# Instead, let's graph the latitude and longitudes on a map

fig = px.density_mapbox(df, lat='latitude', lon='longitude', radius=1,
                        mapbox_style="open-street-map")
fig
```



1.2 Question

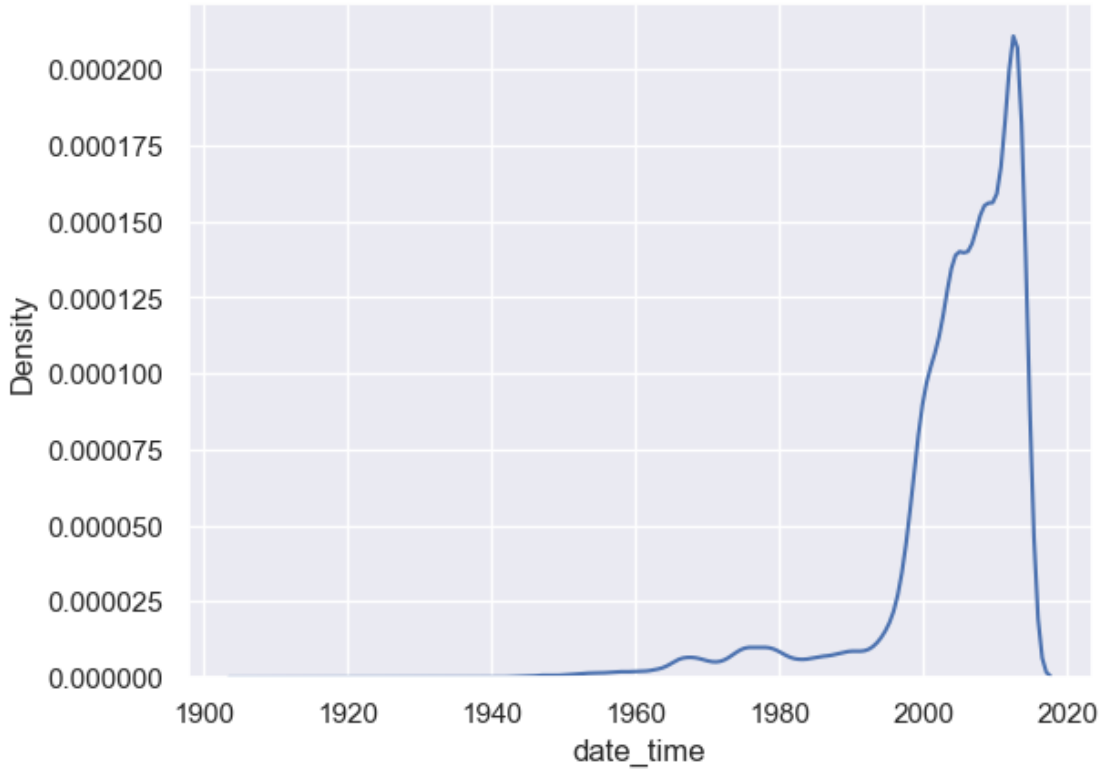
Visualize the distribution of UFO sighting timestamps in this dataset. We will also do some dataset cleaning while we're here.

[48]:
```
def custom_to_datetime(date):
    """This dataset has the nonstandard time 24:00.
    We will treat this as the start of the next day"""
    # If the time is 24, set it to 0 and increment day by 1
    if date[-5:] == '24:00':
        return pd.to_datetime(date[:-5]) + pd.Timedelta(days=1)
    else:
        return pd.to_datetime(date)
```

```
df['date_time'] = df['date_time'].apply(custom_to_datetime)

# There are some extreme outliers in encounter length - 10e8 seconds is␣
 ↪unreasonable
# Since the data are not normally distributed, let's strip out observations
# where the encounter length is greater than the 99% percentile
q = df["encounter_length"].quantile(0.99)
df = df[df["encounter_length"] < q]

sns.kdeplot(df['date_time'])
plt.show()
```



[51]:
```
df.to_csv("ufo_sightings_cleaned.csv")
```

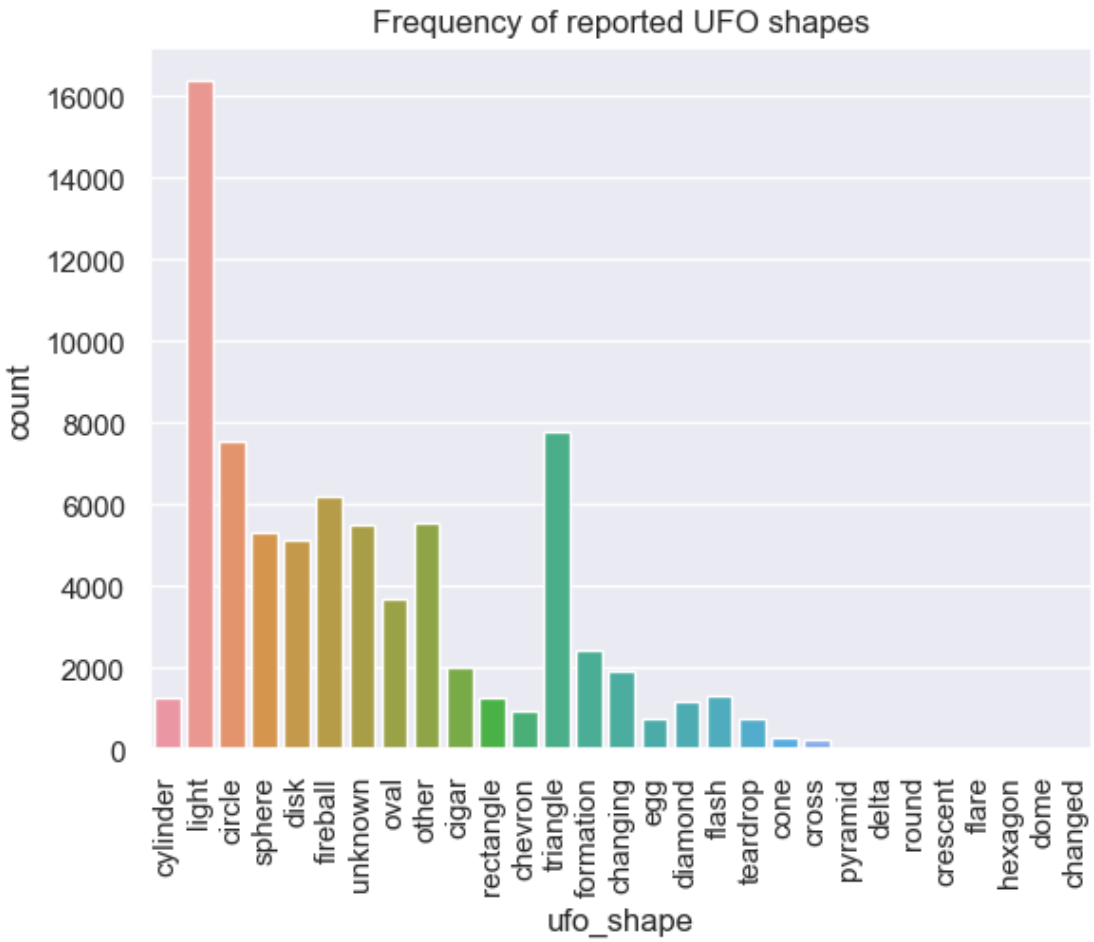
1.3 Question

Hypothesis: some UFO shapes are significantly more commonly seen than others. This is best answered by a chi-squared test.

[49]:
```
g = sns.countplot(data=df, x="ufo_shape")
g.set_xticklabels(g.get_xticklabels(), rotation=90)
```

```
g.set_title('Frequency of reported UFO shapes')
```

[49]: Text(0.5, 1.0, 'Frequency of reported UFO shapes')



[50]:
```
shape_counts = df.groupby("ufo_shape").size().reset_index(name="count")
n = shape_counts["count"].sum()
shape_counts["expected_count"] = n/len(shape_counts)

chi_square_test_statistic, p_value = stats.
 ↪chisquare(shape_counts["count"],shape_counts["expected_count"])
cramer_v = math.sqrt(chi_square_test_statistic / (2.0*n))
print("chi_square statistic:" + str(chi_square_test_statistic))
print("p: " + str(p_value))
```

```
chi_square statistic:137864.1885882323
p: 0.0
```

1.4 Question

Find a way to categorize the UFO sightings into groups. Hint: this will likely be an unsupervised method because we are trying to find a pattern in the data without any ground truth labeling.

[52]:
```
# We're going to use K-means clustering.
# We have to select numeric attributes to cluster on.
# Since one of them will be the datetime of the sighting,
# let's convert the date time to an integer timestamp.
df["time_numeric"] = df["date_time"].apply(lambda x: x.value)
df_numeric = df[["time_numeric", "encounter_length", "latitude", "longitude"]]
```

[53]:
```
# There are some extreme outliers in encounter length - 10e8 seconds is␣
 ↪unreasonable
# Since the data are not normally distributed, let's strip out observations
# where the encounter length is greater than the 99% percentile
q = df_numeric["encounter_length"].quantile(0.99)
df_outlier = df_numeric[df_numeric["encounter_length"] < q]
```

[54]:
```
# K-means clustering assumes each dimension is equally important.
# Right now, the scale of the timestamp dimension is far larger than all others,
# so its contribution will overwhelm the encounter length or location.
# To fix this, let's scale all columns to be 0-1 (i.e. all columns have equal␣
 ↪weight).
scaler = MinMaxScaler()

df_scaled = scaler.fit_transform(df_outlier.to_numpy())
df_scaled = pd.DataFrame(df_scaled, columns=df_outlier.columns)

df_scaled.head()
```

[54]:

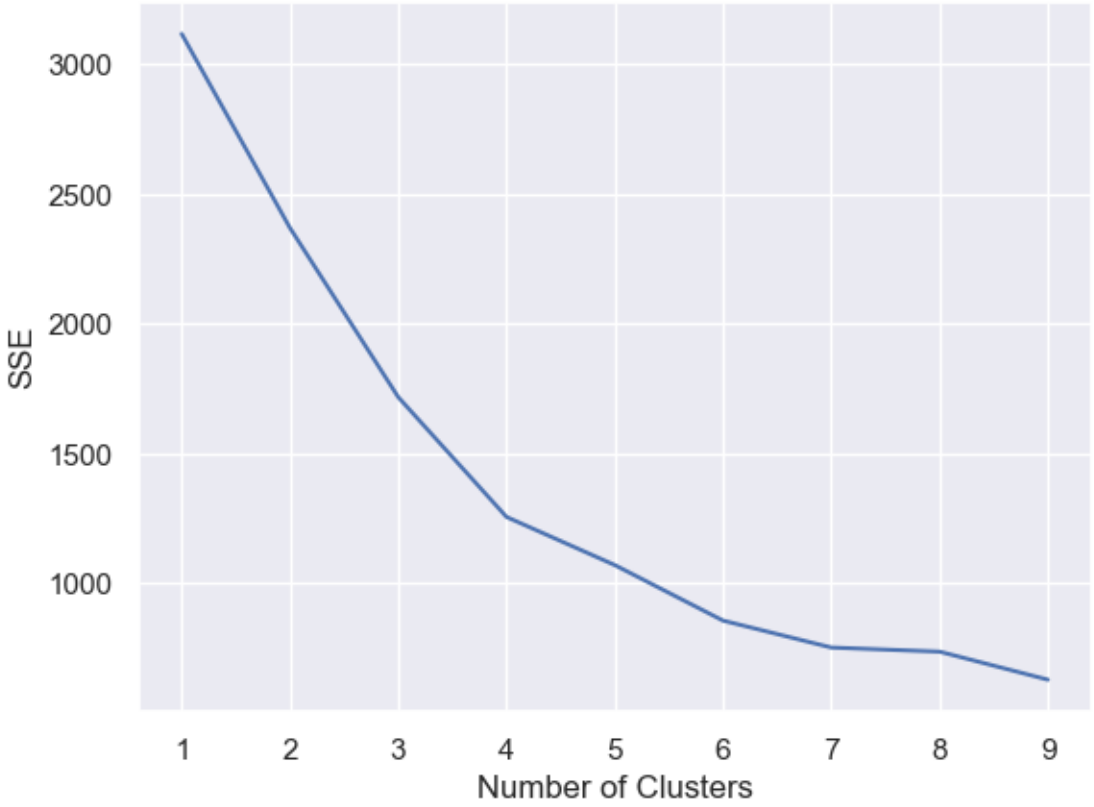
	time_numeric	encounter_length	latitude	longitude
0	0.381154	0.381356	0.724761	0.221675
1	0.438638	0.002825	0.874649	0.489275
2	0.448245	0.002825	0.718945	0.225323
3	0.486578	0.127119	0.670346	0.053096
4	0.496154	0.042373	0.767907	0.266035

[55]:
```
# An important question: how many clusters should we select?
# One way to choose is to try several cluster values and look for the "elbow"␣
 ↪joint.
# This joint is where adding new clusters no longer reduces the sum of squared␣
 ↪errors.
numClusters = range(1,10)
SSE = []
for k in numClusters:
    k_means = cluster.KMeans(n_clusters=k, n_init='auto')
    k_means.fit(df_scaled)
```

```
    SSE.append(k_means.inertia_)

plt.plot(numClusters, SSE)
plt.xlabel('Number of Clusters')
plt.ylabel('SSE')
```

[55]: Text(0, 0.5, 'SSE')



[56]:
```
# Looking at the above graph, it seems that 4, 6, or 8 are reasonable values.
# Let's use 4 for ease of interpretation. Random_state is set to 1 for␣
 ↪reproducibility.
k_means = cluster.KMeans(n_clusters=4, random_state=1, n_init='auto')
k_means.fit(df_scaled)

centroids = k_means.cluster_centers_
pd.DataFrame(centroids,columns=df_scaled.columns)
```

[56]:

	time_numeric	encounter_length	latitude	longitude
0	0.915552	0.394490	0.780511	0.239090
1	0.930824	0.037328	0.784245	0.241471
2	0.897108	0.062310	0.498983	0.795863

```
3      0.626995          0.063854  0.781652   0.247137
```

We've found 4 clusters. Roughly, they correspond to: 1. Very short, recent American sightings. 2. Very short, recent European sightings. 3. Very short, old American sightings. 4. Long, recent American sightings.

How much do we believe these clusters are "real?" There is a medium elbow joint in the cluster graph above, which suggests K-means is OK but not great for clustering this data. The actual clusters are reasonable in the sense that we would expect American and European data to be distinct clusters under any reasonable metric, and it also makes sense there is a difference between short and long sightings or recent or past sightings.

If we were only to look at this clustering, we see that the long sightings tend to occur recently and in the US, as opposed to far in the past in Europe. However, that might just be an artifact of the way the data were collected, which are consistent with an online English survey based on the location of respondents.

[ ]: