**Data Preparation**

We started with analyzing the UFO sightings data. The most important column identified was the location as it was be used to get the latitude and longitude using geocode API. We observed following nature of the location field in the data which were later exploited to decode the ISO region for the location.

1. Total number of sightings are 61067. Number of non-US sightings based on a cursory analysis is 9019. Out of these there are around 21k unique locations.
2. Most of the US sightings followed the notation of (county\_name, state\_code) For example: - Santa Cruz, CA.
3. Some US sightings are mostly random and has Freeway/Expressway names
4. There is no similar notation used in non-US sightings except for the countries Canada, Australia and some others. These countries have the country name in the location string.

**Finding Nearest Airport**

We used the following airport dataset**: -** <http://ourairports.com/data/airports.csv>

This file contains around 53k airports with the longitude, latitudes, iso\_country and iso\_region. First thing we had to do was get the latitudes and longitudes for the UFO sightings dataset and then compare it to the airports dataset to get the closest airport. The first instinct was to compare all the locations with all the airports. But soon we could see that it will be too many comparisons. Then we found that as most of the sightings are in US and there are around 22k airports corresponding to US region, we need to have a better strategy for reducing number of comparisons. So, we came up with the following strategy.

1. For all the US sightings, we will assume that it could be related to an airport in that state or the neighboring state. For this we created a key-value pair data structure which had every state’s neighboring state. Using the state codes of the neighboring states we formed ISO region codes such as US-CA (for USA California). We used these codes to narrow down on the list of airports to be compared to the UFO sighting location.
2. For non-US sightings we assume that the sightings will be related to the airports in that country alone. While getting the latitude and longitudes from geocode API, we also fire the reverse query to capture the ISO country code of that location. This helps reducing the number of comparisons in non-US sightings.

**Issues faced while getting longitude and latitudes from geocoding API**

1. The OpenMaps geocode API would throw Too many requests error if overwhelmed with too many geocode requests. For this we had to store all the unique locations in pyMongo DB and then call the geocode API with a sleep of 1 second.
2. Some cities in US share names with cities in Canada and Mexico for such cases to get best results we had to include the name of country in the location query string.
3. Same city names across different states in USA. For these we had to include the state name in the location query string.