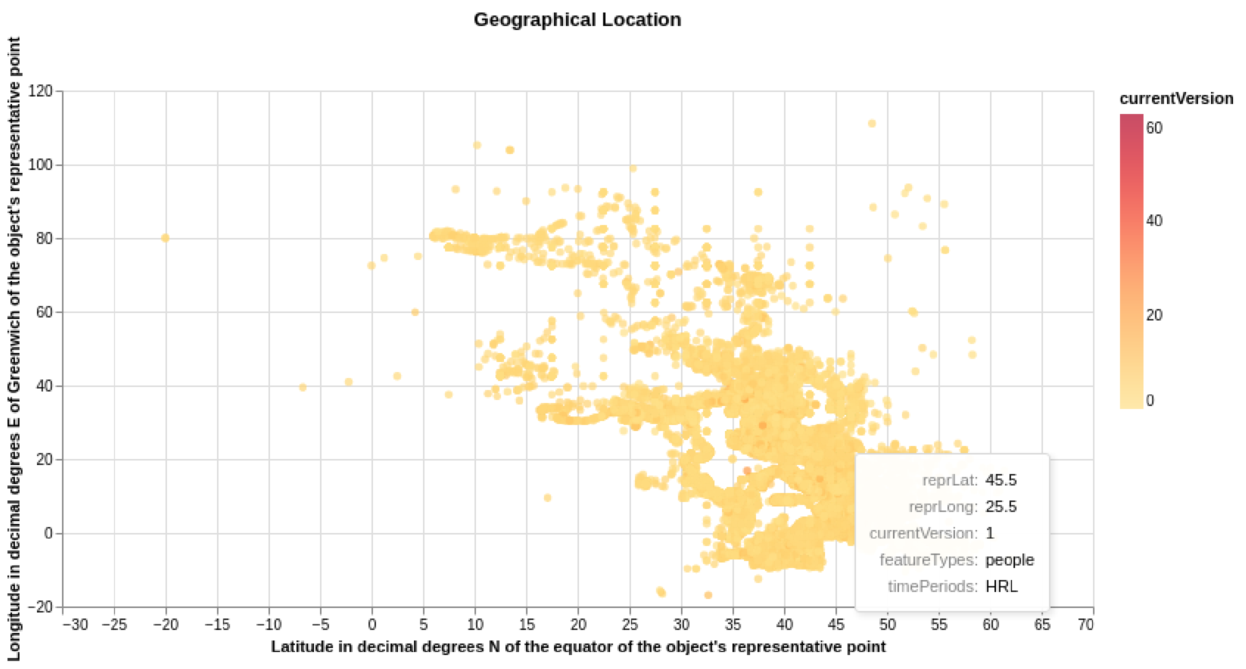
Visualization 1

Aim (aim): This visualization aims to find the object's geographical position by mapping its latitude and longitude along with the version. From that particular location of the object, infer what kind of feature type is existing. For eg: temple, settlement etc. Here in the visualization we could see most of the data points are on south-east part of the chart and there are outliers as well around the cluster.

Visual Design Type (vistype): Scatter Plot

Image:



Source Code

```python
# importing the necessary libraries altair and pandas for data visualization and manipulation
import altair as alt
import pandas as pd
alt.data_transformers.disable_max_rows()

# reading data from url and store it in places_data
places_data_url='https://raw.githubusercontent.com/SwanseaU-TTW/csc337_coursework1/master/pleiades-places-latest.csv'
places_data = pd.read_csv(places_data_url)

# creating a base structure for chart using the places_data
selector = alt.selection_single(empty='all', fields=['X','Y'])
base_main = alt.Chart(places_data, title='Geographical Location').properties(
    width=700,
    height=350
).add_selection(selector)

# creating a chart for the main design and assign to a variable
geo_points = base_main.mark_circle().encode(
    alt.X('reprLat',title="Latitude in decimal degrees N of the equator of the object's representative point",
          scale=alt.Scale(domain=(-30, 70))),
    alt.Y('reprLong',title="Longitude in decimal degrees E of Greenwich of the object's representative point",
          scale=alt.Scale(domain=(-20, 120))),
    alt.Color('currentVersion', scale=alt.Scale(scheme='yelloworangered')),
    tooltip=['reprLat', 'reprLong', 'currentVersion', 'featureTypes', 'timePeriods']).interactive()

# display visualization
geo_points
```

Visual Mappings (vismapping): The unique concept that generated from an intricate amount of data is to visualize the objects for their different versions along with geographical locations with the help of latitude and longitude. Hence the visualization is done by plotting the latitude on X axis and longitude on Y axis. That would give us the position and scatter plot is used for this concept to make a visualization. The next objective was to plot versions of each entry and we are able to make the color factor to define the versions. More instensity relates to latest version. Less intensity refers to older version. An interactive element is made by giving a tooltip for every points to display the details including feature types and timeperiod.

Data Preparation (dataprep): Data is taken from pleiades-places-latest.csv. This file has fields for latitude(reprLat) and longitude(reprLong) to plot the location and a field for storing version(currentVersion). For identifying feature type of that location, data was taken from featureTypes field and timePeriods field is used for plotting time period into the tooltip.

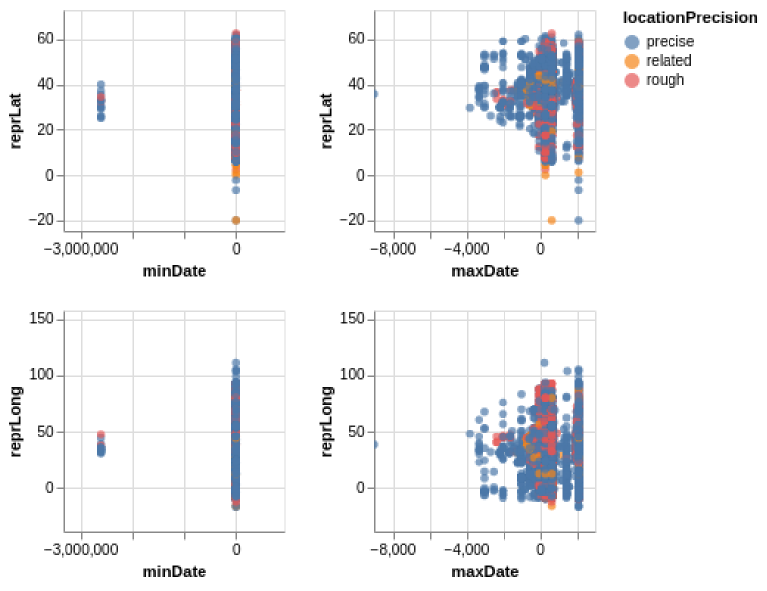
Improvements (improvements): Improvements could be done by plotting the geo cordinates to actual world maps and even the time periods and feature types could be specifically given. For the feature types it would be great to give a unique small icon for every points. For eg: if the feature type is a temple then a small icon of temple could be shown in the plot.

Visualization 2

Aim (aim): From this visualization the aim is to find out how precise the location was mapped in accordance with its geocontext location variables such as latitude and longitude with minimum and maximum date of objects to compare the location precisions with age. From this visualization we can see most of the minimum date is zero. The comparison of latitude and longitude with maximum date shows that precise locations are scattered more than rough and related locations.

Visual Design Type (vistype): Scatter Matrix

Image:



Source Code

```
# importing the necessary libraries altair and pandas for data visualization and manipulation
import altair as alt
import pandas as pd
from vega_datasets import data
alt.data_transformers.disable_max_rows()

# data = pd.read_csv('cw/csvdata/pleiades-places-latest.csv')
places_data_url = 'https://raw.githubusercontent.com/SwanseaU-TTW/csc337_coursework1/master/pleiades-places-latest.csv
places_data = pd.read_csv(places_data_url)

tp_arr = places_data['timePeriods'].unique()

alt.Chart(places_data).mark_circle().encode(
    alt.X(alt.repeat("column"), type='quantitative'),
    alt.Y(alt.repeat("row"), type='quantitative'),
    color='locationPrecision'
).properties(
    width=150,
    height=150
).repeat(
    row=['reprLat','reprLong'],
    column=['minDate','maxDate']
).interactive()
```

Visual Mappings (vismapping): The idea is to create a plot to compare multiple data fields to evaluate the location precision with minimum and maximum date. Thought process infered to make a scatter matrix to display various relationhip such as minimum date vs latitude, minimum date vs longitude, maximum date vs latitude and maximum date vs longitude. Along with this comparison, the location precision field is represented by color. This helps users to identify the location is whether precised or rough or related. The user can zoom in or zoom out each plot in the matrix. This is achieved by incorporating interactive method to chart. Altair's mark_circle method is used for creating the scatter plots and repeat method is used for creating matrix with tow and column.

Data Preparation (dataprep): For this visualization, the data preparation is pretty straight forward. We could get geocontext data from fields reprLat and reprLong. Other fields used are minDate, maxDate and locationPrecision. The data is taken from pleiades-places-latest.csv.

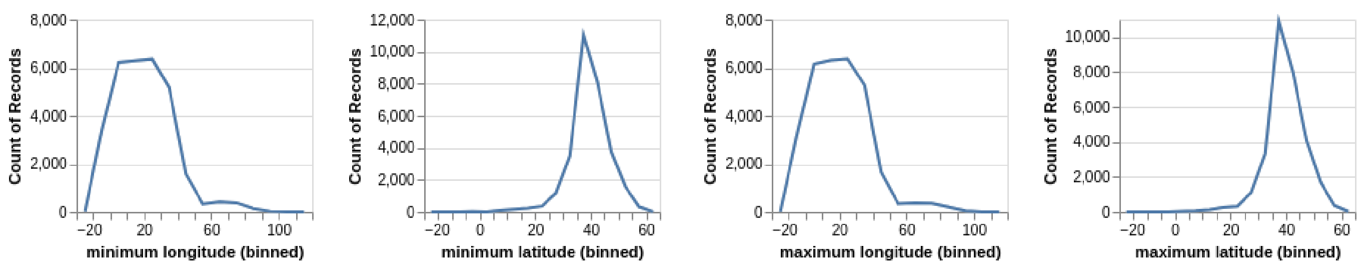
Improvements (improvements): Couple of improvements could be done is to make the plot more interactive by giving tooltips to each point or showing details of a particular data by clicking on itself showing different chart side by side. Including couple of data fields to row or column would make more comparisons.

Visualization 3

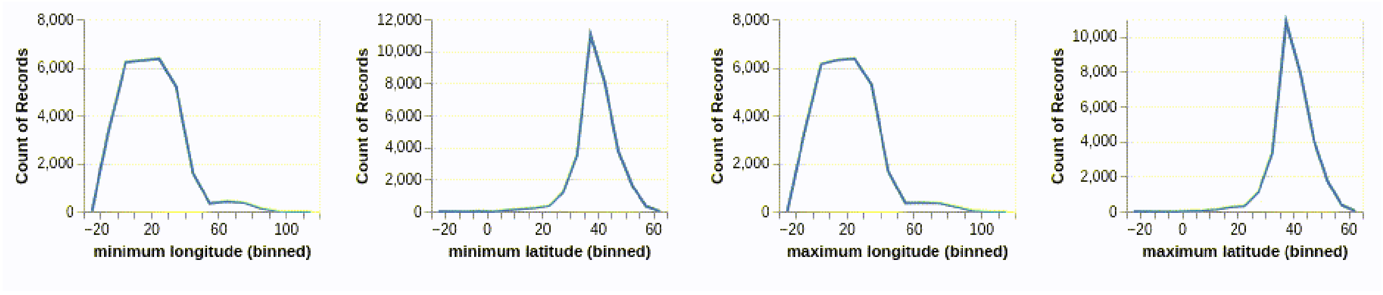
Aim (aim): From this visualization we could see that number of records for plotting a bounded area for pleiades in four differnt plots. Minimum latitude and minimum longitude which represent the bottom right corner of the bounded area has more records od data when longitude between -20 to 60. At the same time, minimum latitude has more records when the values in between 20 and 60, and has more than 10000 records at approximately 40. Maximum latitude and longitude have the similar range of records like in minimum.

Visual Design Type (vistype): Interactive Crossfilter

Image:



Interactivity



Source Code

```
# importing the necessary libraries altair and pandas for data visualization and manipulation
import altair as alt
import pandas as pd
from vega_datasets import data
alt.data_transformers.disable_max_rows()
```

```
# fetching the data from url
places_data_url = 'https://raw.githubusercontent.com/SwanseaU-TTW/csc337_coursework1/master/pleiades-places-latest.csv
places_data = pd.read_csv(places_data_url)
bbox = places_data['bbox'].str.split(",", n = 3, expand = True)
bbox.columns = ['minimum longitude','minimum latitude','maximum longitude','maximum latitude']

brush = alt.selection(type='interval', encodings=['x'])

# Define the base chart, with the common parts of the
# background and highlights
base = alt.Chart().mark_line().encode(
    x=alt.X(alt.repeat('column'), type='quantitative', bin=alt.Bin(maxbins=20)),
    y='count()'
).properties(
    width=160,
    height=130
)

# gray background with selection
background = base.encode(
    color=alt.value('#ddd')
).add_selection(brush)

# blue highlights on the transformed data
highlight = base.transform_filter(brush)

# layer the two charts & repeat
alt.layer(
    background,
    highlight,
    data=bbox
).repeat(column=['minimum longitude','minimum latitude','maximum longitude','maximum latitude'])
```

Visual Mappings (vismapping): The basic concept was to identify number of records and compare the occurrences of most two ends of the bounded area (bottom-right and top-left). We had the data from the field bbox. So for this concept we would need all four geo context location points and number of records present for each point. Altair is used for creating this visualization with the help of repeat functions we are able to make the four differnt plots to make an interactive crossfilter. Interactivity is added to chart for comparing number of records for each location coordinate. In the images above we could see plot and its interactivity in the second image. Y axis is taken for plotting the count of records and geo coordinate points are plotted on X axis. X axis values are binned for avoiding minor observation errors.

Data Preparation (dataprep): For this visualization data analysis and preparation had to be done. The required data is present in the column bbox in the file pleiades-places-latest.csv. The challenge was to separate values of bbox to different columns in a new dataframe so that we could plot directly from that dataframe. Pandas library functions are used for reading the data from url and python's split function is used for separating the content in this column. Analysing the data, we understood that the geographical coordinates are separated by comma. After splitting, appropriate column headings are given.

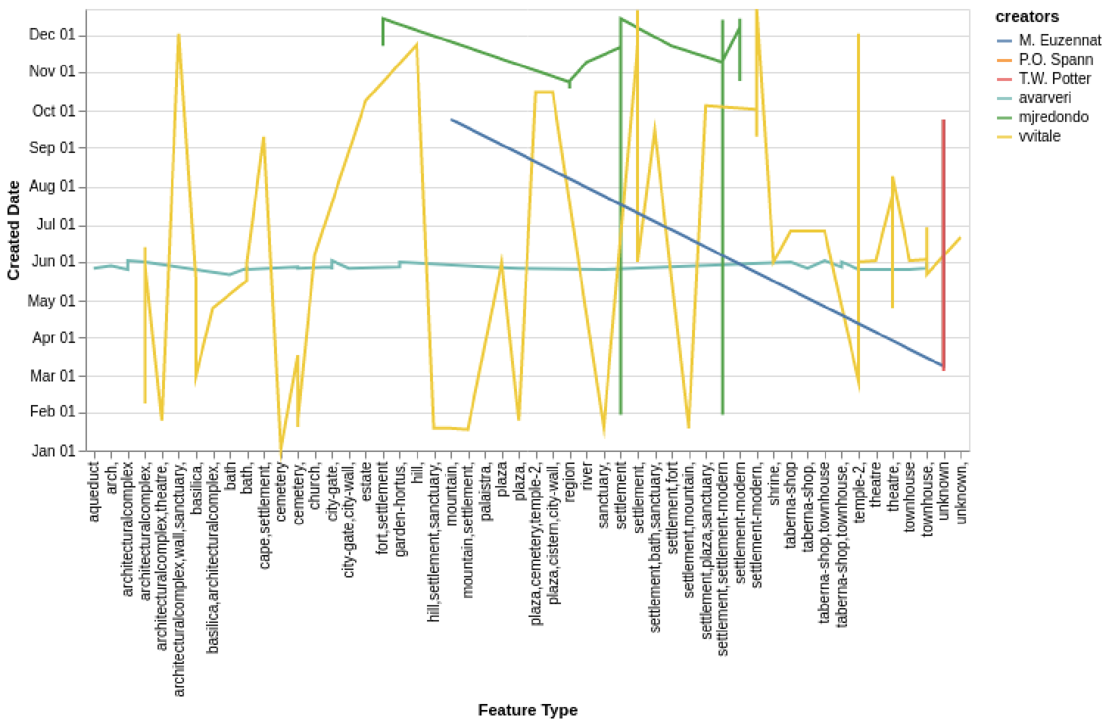
Improvements (improvements): Instead of record counts, there could be another values given so that comparison would be more specific related to the coulmns which is in the data file itself. For eg: Plotting latitude and longitude value of location along with bbox would give us the relation between bounded area and location and thereby conclude that it woud be a precise location.

Visualization 4

Aim (aim): The aim of this visualization is to compare and evaluate the feature types rgarding to created date of six known creators such as Avarveri, Mjredondo, Vvitale, P.O. Spann, M. Euzennat and T.W. Potter. From this visualization we could infer that Vvitale has created almost all kind of feature types and has great span of created date in a year as well. Avarveri has created mostly during June and he has got wide variety of feature types in his records. T.W. Potter's feature types are unknown. Mjredondo has created records in between March to October.

Visual Design Type (vistype): Parallel Coordinates

Image:



Source Code

```
# importing the necessary libraries altair and pandas for data visualization and manipulation
import altair as alt
import pandas as pd
from vega_datasets import data
alt.data_transformers.disable_max_rows()

# fetch data from url
locations_data_url = 'https://raw.githubusercontent.com/SwanseaU-TTW/csc337_coursework1/master/pleiades-locations-late
locations_data = pd.read_csv(locations_data_url)

# create new data frame
df2 = pd.DataFrame(locations_data)
df2_selected_columns = df2[['featureType','created','locationPrecision','timePeriods','creators','minDate','maxDate','
new_df2 = df2_selected_columns.copy()

# filter data for certain creators
creator_avarveri = new_df2.loc[new_df2['creators'].isin(['avarveri'])]
creator_mjredondo = new_df2.loc[new_df2['creators'].isin(['mjredondo'])]
creator_vvitale = new_df2.loc[new_df2['creators'].isin(['vvitale'])]
creator_spann = new_df2.loc[new_df2['creators'].isin(['P.O. Spann'])]
creator_euzennat = new_df2.loc[new_df2['creators'].isin(['M. Euzennat'])]
creator_potter = new_df2.loc[new_df2['creators'].isin(['T.W. Potter'])]


# concatenating above data to a single one
frames = [creator_avarveri, creator_mjredondo, creator_vvitale, creator_spann, creator_euzennat, creator_potter]
creators = pd.concat(frames)

# converting date to proper format
created_date=pd.to_datetime(creators['created'])
creators['created_date'] = pd.Series(created_date, index=creators.index)

# plot the chart
source = creators
alt.Chart(source).transform_window(
    index='count()'
).mark_line().encode(
    alt.X('featureType',title='Feature Type'),
    alt.Y('monthdate(created_date)',title='Created Date'),
    color='creators',
    opacity=alt.value(1)
).properties(width=600)
```

Visual Mappings (vismapping): The idea is to plot and find the comparison of certain known creators records according to their creation date in a year and evaluate the feature types each one of them created during a year. This concept was converted to a parallel coordinate chart with different months in a year in Y axis and feature types in X axis. The line segments are plotted with different colors to identify the creators. Since there are lot of different feature types, the X axis would be lengthier and hence the width is chosen as 600.

Data Preparation (dataprep): The data has been taken from pleiades-locations-latest.csv. From this file, a new data frame is created and filtered with fieds such as 'featureType','created','locationPrecision','timePeriods','creators','minDate','maxDate' and 'currentVersion'. Now the step was to filter the rows that needed for plotting the data. Six creators are chosen and filtered into different variables followed by concatenating these to a single variable 'creators'. Now the created date is converted to proper format and embed to the data frame. These were some essential data preparation done for this visualization.

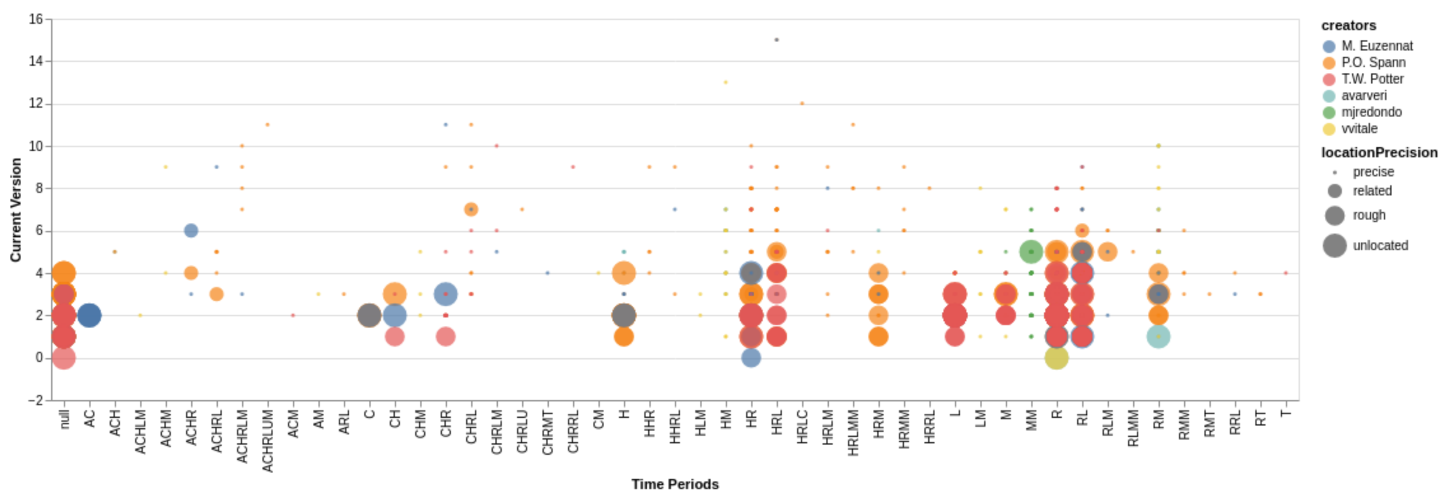
Improvements (improvements): There are some difficulties in analysing data from certain creators that it is not clear in this visualization and some feature types are repeating as well.

Visualization 5

Aim (aim): From this visualisation we could see various information regarding six known creators and their respective records in different time periods and versions. The precision of locations that they plotted are also communicating to the user. From the plot we could easily infer that most of the records are precised by its location. When the version increases the location is more precised. Vvitale has only one unlocated data and its in the version 0. According to the visualisation T.W. Potter has more number of unlocated data and when version increases the the data had been more precised.

Visual Design Type (vistype): Multi Feature Scatter Plot

Image:



Source Code

```
# importing the necessary libraries altair and pandas for data visualization and manipulation
import altair as alt
import pandas as pd
alt.data_transformers.disable_max_rows()

# fetch data from url
places_data_url = 'https://raw.githubusercontent.com/SwanseaU-TTW/csc337_coursework1/master/pleiades-places-latest.csv
places_data = pd.read_csv(places_data_url)

# creating new data frame and filter required data
df2 = pd.DataFrame(places_data)
df2_selected_columns = df2[['id','locationPrecision','timePeriods','creators','featureTypes','minDate','maxDate','curr
new_df2 = df2_selected_columns.copy()

creator_avarveri = new_df2.loc[new_df2['creators'].isin(['avarveri'])]
creator_mjredondo = new_df2.loc[new_df2['creators'].isin(['mjredondo'])]
creator_vvitale = new_df2.loc[new_df2['creators'].isin(['vvitale'])]
creator_spann = new_df2.loc[new_df2['creators'].isin(['P.O. Spann'])]
creator_euzennat = new_df2.loc[new_df2['creators'].isin(['M. Euzennat'])]
creator_potter = new_df2.loc[new_df2['creators'].isin(['T.W. Potter'])]
```

```
frames = [creator_avarveri, creator_mjredondo, creator_vvitale, creator_spann, creator_euzennat, creator_potter]
creators = pd.concat(frames)

source = creators

alt.Chart(source).mark_circle().encode(
    alt.X('timePeriods',title='Time Periods', scale=alt.Scale(zero=False)),
    alt.Y('currentVersion',title='Current Version', scale=alt.Scale(zero=False, padding=1)),
    color='creators',
    size='locationPrecision'
)
```

Visual Mappings (vismapping): The visualization is created by scatter plot, but with multiple features in it. The X and y axes are mapped to time periods and current version respectively. At the same time, size of the circle is mapped to location precision and color is mapped to creators. Since there are only six creators selected for plotting, there would be unique colors for each creator. Time Periods could be mapped to colors but since there are lot of different time periods available in data frame, we decided to plot time frame in X axis.

Data Preparation (dataprep): For plotting datas to this visualisation, we need four different kinds of data. Time periods and version data are pretty straight forward. For the creator data some filteration has been done to select six known creators from the whole data in pleiades-places-latest.csv. The selected creaors are Avarveri, Mjredondo, Vvitale, P.O. Spann, M. Euzennat and T.W. Potter. A new data frame is created and filtered the following columns: 'id', 'locationPrecision', 'timePeriods', 'creators', 'featureTypes', 'minDate', 'maxDate', 'currentVersion'. The next task was to filter the selected six creators and join them together. Six different data frames are created for creators and concat() method from pandas library was used for concatenating the frames together.

Improvements (improvements): Precise locations are too feable to notice. Some datas are overlapping and this makes it difficult to identify which creator's data is that point.