Πανεπιστήμιο Πατρών, Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

ΑΝΑΦΟΡΑ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΑΣΚΗΣΗΣ ΕΞΟΡΥΞΗΣ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΑΛΓΟΡΙΘΜΩΝ ΜΑΘΗΣΗΣ 2022-2023

1) Όνομα: Γεώργιος
2) Επώνυμο: Βέργος
3) Αριθμός Μητρώου: 1072604
4) Email: up1072604@upnet.gr
5) Έτος/Εξάμηνο: 4ο/8ο
6) Ημερομηνία: 1/9/2023

Πάτρα Σεπτέμβριος 2023

Καταγραφή του περιβάλλοντος υλοποίησης

Ως γλώσσα υλοποίησης χρησιμοποιώ την Python έκδοση **3.10.** Για την ανάπτυξη της άσκησης χρησιμοποιήθηκε το εργαλείο(IDE) PyCharm της JetBrains. Χρησιμοποιήθηκαν οι ακόλουθες βιβλιοθήκες της Python:

1) Pandas για τον χειρισμό των csv αρχείων καθώς και για τη επεξεργασία του συνόλου δεδομένων.
2) Numpy για μαθηματικούς υπολογισμούς. Αποτελεί απαιτούμενη βιβλιοθήκη για την sklearn.
3) Matplotlib για την απεικόνιση των δεδομένων και των αποτελεσμάτων επεξεργασίας αυτών. Πάνω σε αυτή έχει χτιστεί η sklearn.
4) seaborn για την απεικόνιση κάποιων συγκεκριμένων γραφικών παραστάσεων(όπως το μητρώο συσχετίσεων σε heatmap).
5) sklearn(scikit-learn) για την δημιουργία μοντέλων αλγορίθμων μηχανικής μάθησης(κατηγοριοποίηση, συσταδοποίηση, παλινδρόμηση)και την εκτέλεση τους πάνω στα δεδομένα.
6) SciPy για πράξεις μητρώων κ.ο.κ(γραμμική άλγεβρα). Αποτελεί απαιτούμενη βιβλιοθήκη για την sklearn.
7) IPython για την ωραιότερη απεικόνιση/εκτύπωση grouped by dataframes.
8) warnings για την απενεργοποίηση των warnings κατά την εκτέλεση των προγραμμάτων.
9) tensorflow για την δημιουργία και εκπαίδευση των RNNs.

Η εγκατάσταση των παραπάνω έγινε μέσω powershell(Περιβάλλον Windows 10) με την εκτέλεση της εντολής:
pip3 install <όνομα πακέτου>.

Παρακάτω επισυνάπτω τα στιγμιότυπα με τις βιβλιοθήκες κάθε ερωτήματος:

Ερώτημα 1

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import zscore
from IPython.display import display
from sklearn.cluster import KMeans
```

Ερώτημα 2

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import zscore
from IPython.display import display
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler #if we have big variance between elements standardscaling wouldnt work on small numbers
from sklearn.decomposition import PCA
import warnings
```

Ερώτημα 3

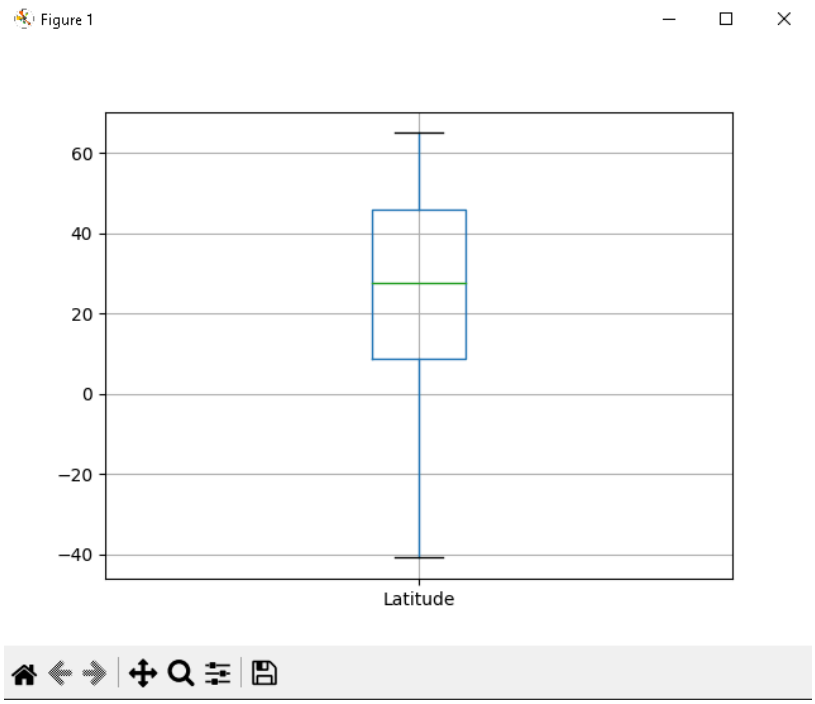
```python
from sklearn.svm import SVR
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import zscore
from IPython.display import display
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
import tensorflow as tf
#from tensorflow import keras
#from tensorflow.keras import layers
from datetime import timedelta,date,datetime
import warnings
```

Σύντομη περιγραφή της διαδικασίας υλοποίησης/Σχολιασμός των τελικών αποτελεσμάτων

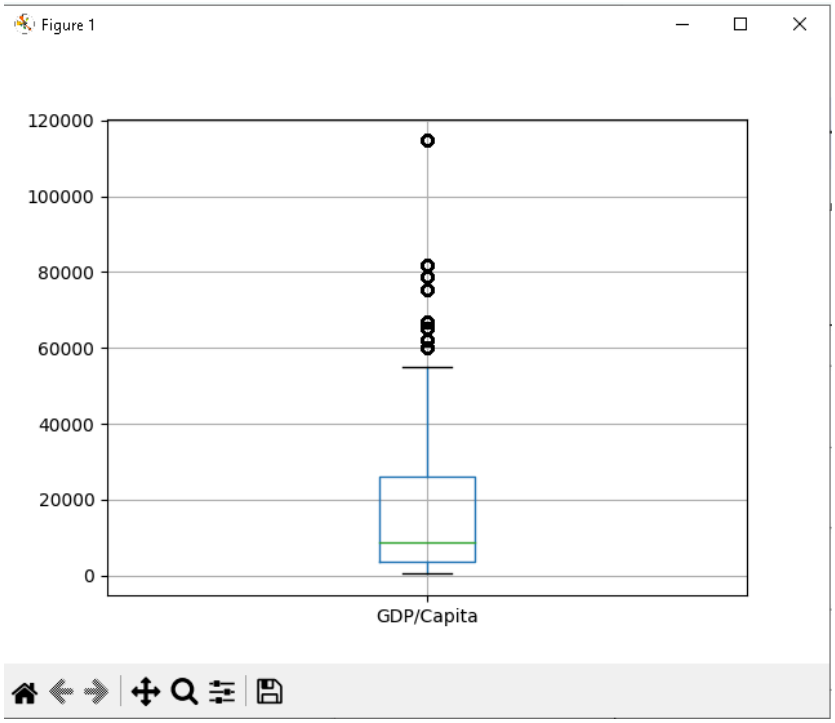
Ερώτημα 1

Για αρχή φορτώνω το σύνολο δεδομένων μέσω της **pandas**. Στη συνέχεια κάνω μία γενική περιγραφή του dataset και βρίσκω τις τιμές που απουσιάζουν μέσω των συναρτήσεων **info(), describe(), isnull(),sum()**. Έπειτα κάνω μία στατιστική περιγραφή των στηλών που μας ενδιαφέρουν και έχουν null τιμές όπως οι Daily Tests, Cases, Deaths μέσω της **describe()**. Ακόμη επιλεκτικά για ένα από αυτά τα 3 χαρακτηριστικά λ.χ τα Deaths κάνω στατιστική περιγραφή ανά ομάδες μέσω της **grouby('Entity')** ομαδοποιώντας δηλαδή τις εγγραφές ανά χώρα. Στη συνέχεια, αυτή τη φορά ομαδοποιώντας πάλι ανά χώρα τυπώνουμε τις ελλιπείς τιμές των 3 αυτών στηλών για κάθε χώρα ξεχωριστά. Σε αυτό χρησιμοποιήθηκαν και οι ανώνυμες συναρτήσεις **lambda** της python. Μιας και το σύνολο δεδομένων δεν περιλαμβάνει κατηγορικά δεδομένα ή διακριτά αριθμητικά δεδομένα δεν πραγματοποιώ κάποια ανάλογη διεργασία πάνω τους. Στη συνέχεια συμπληρώνω τις ελλιπείς τιμές μέσω της **fillna()** κάνονας **ffill()** και **bfill()** δηλαδή για κάθε γραμμή του χαρακτηριστικού που έχει ελλιπή τιμή συμπληρώνω με την προηγούμενη ή επόμενη της τιμή αντίστοιχα. Ακόμη βρίσκω το συνολικό αριθμό κρουσμάτων κάθε χώρας(το μέγιστο) ή αντίστοιχα το ποσοστό θνησιμότητας κάθε χώρας. Στη συνέχεια για τα χαρακτηριστικά που έχουν την ίδια τιμή στις εγγραφές κάθε χώρας όπως Langitude, Longitude ,...,Population κ.ο.κ τα απεικονίζω μέσω των boxplots της python.

Παράδειγμα για το longitude:

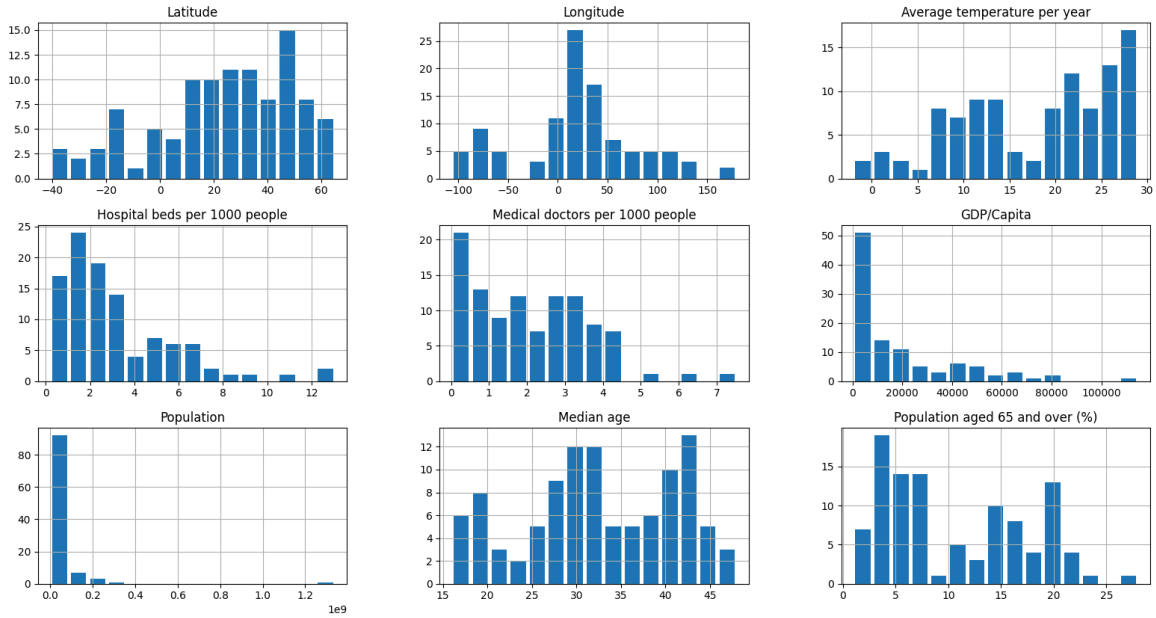


Ή το GDP/Capita:

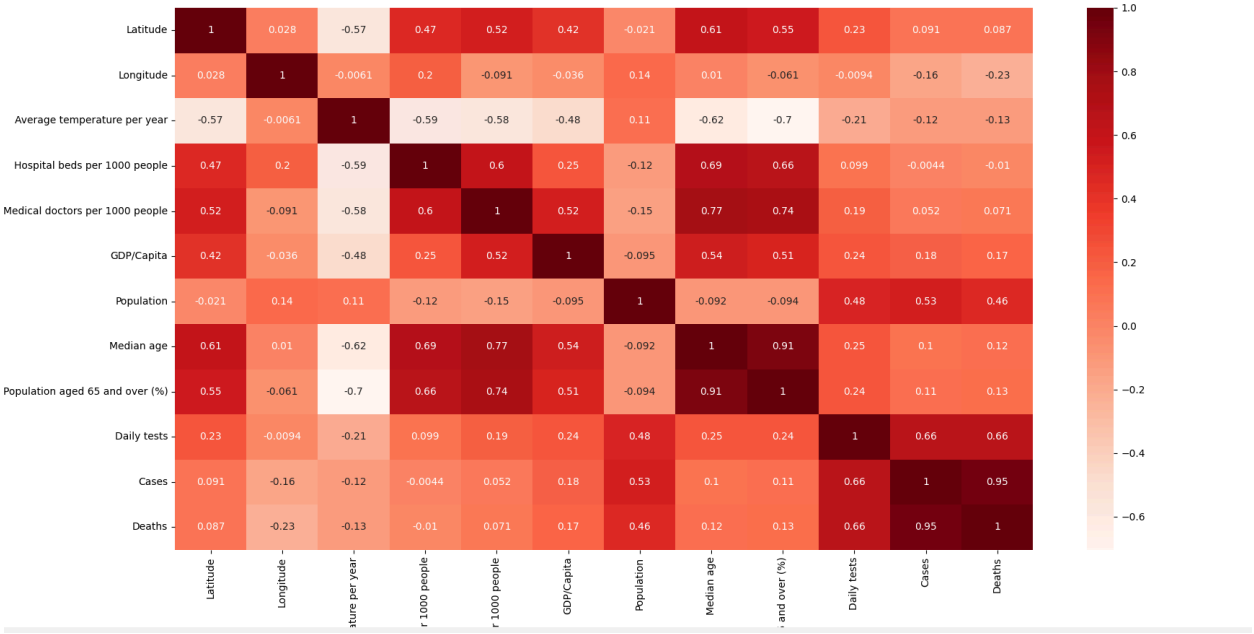


Όπου μεταξύ άλλων φαίνονται και οι outliers.

Έπειτα με τη χρήση ιστογραμμάτων δείχνω τη συχνότητα των τιμών των παραπάνω χαρακτηριστικών στο dataset:

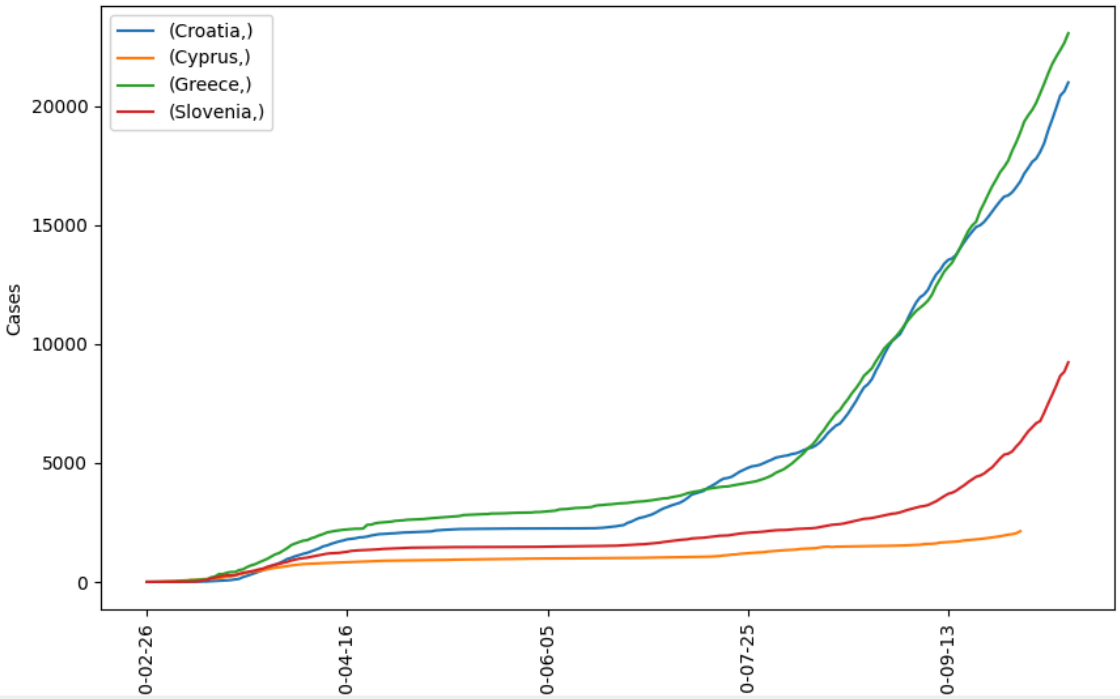


Ακόμη κατασκευάζω ένα μητρώο συσχετίσεων και το οπτικοποιώ πάνω σε ένα heatmap προκειμένου να δω τις συσχετίσεις όλων των χαρακτηριστικών μεταξύ τους μέσω της **heatmap()** της **seaborn** και της **corr()** της **pandas**:

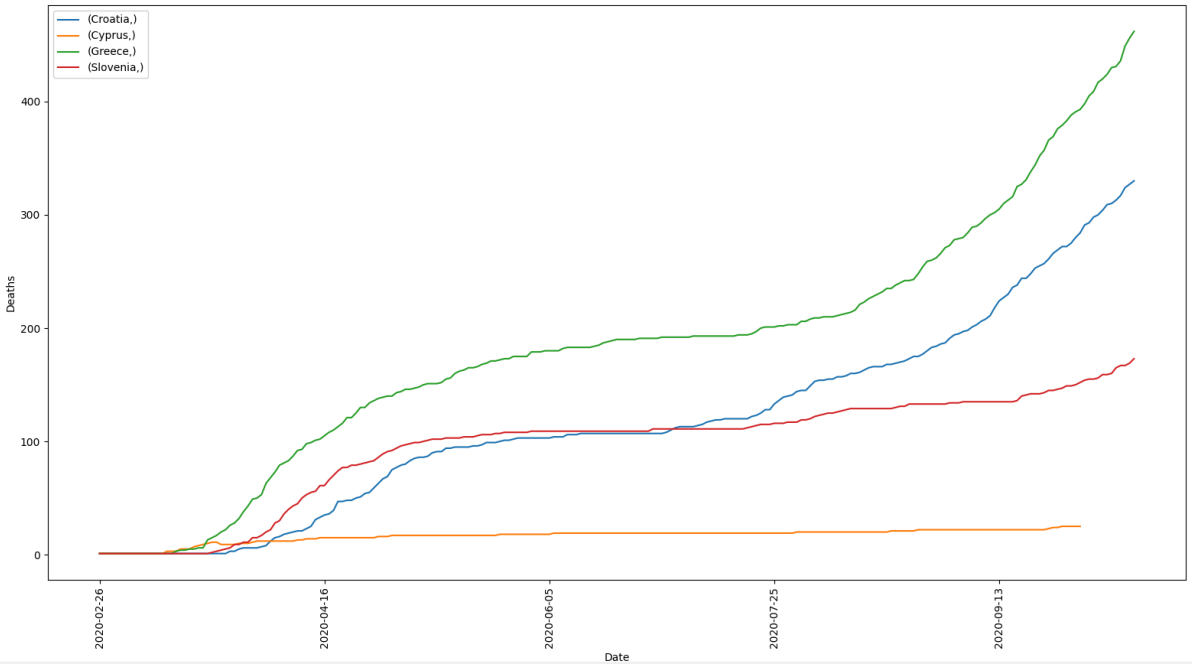


Παρατηρώ ότι υπάρχει ισχυρή θετική συσχέτιση μεταξύ των χαρακτηριστικών Cases και Deaths , median age και medical doctors per 1000 people, Population aged 65 and over(%) και medical doctors per 1000 people. Γενικά θεωρούμε ισχυρή θετική ή αρνητική συσχέτιση αυτή πού επιστρέφει τον συντελεστή συσχέτισης(relation coefficient) πάνω από 0.7 ή κάτω από -0.7 αντίστοιχα. Τα περισσότερα χαρακτηριστικά μεταξύ τους έχουν μέτριες και ασθενείς θετικές είτε αρνητικές συσχετίσεις.

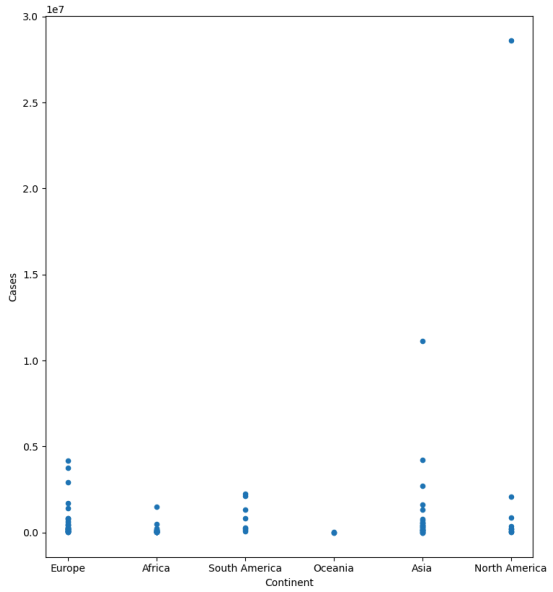
Για περαιτέρω οπτικοποίηση των δεδομένων, για επιλεγμένες χώρες δείχνω σε γραφικές παραστάσεις, την πορεία των Cases και Deaths μεταξύ ορισμένων από εμένα ημερομηνιών. Μέσω των κατάλληλων ορισμάτων στις συναρτήσεις **subplots** και **plot** των **pyplot** και **pandas** αντίστοιχα πετυχαίνω το παρακάτω αποτέλεσμα:

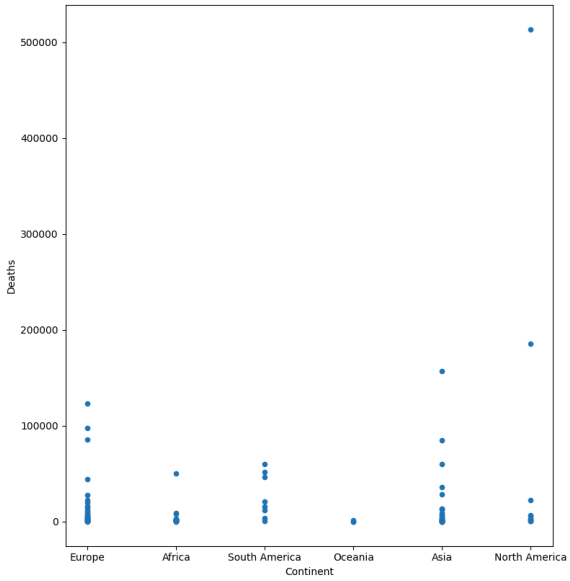


Και:

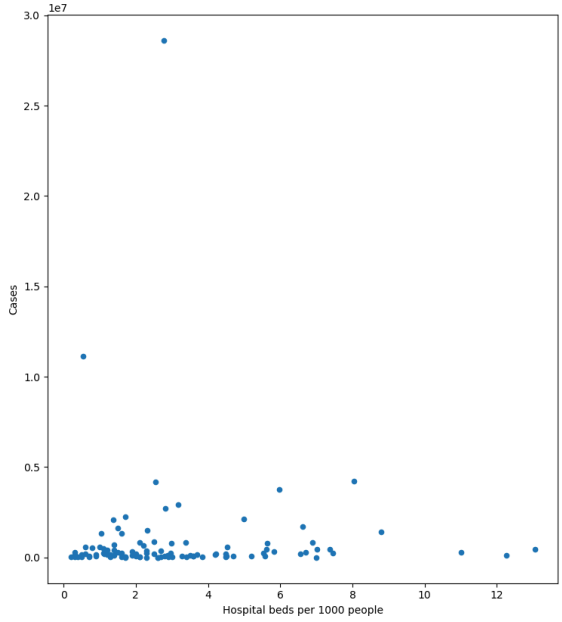


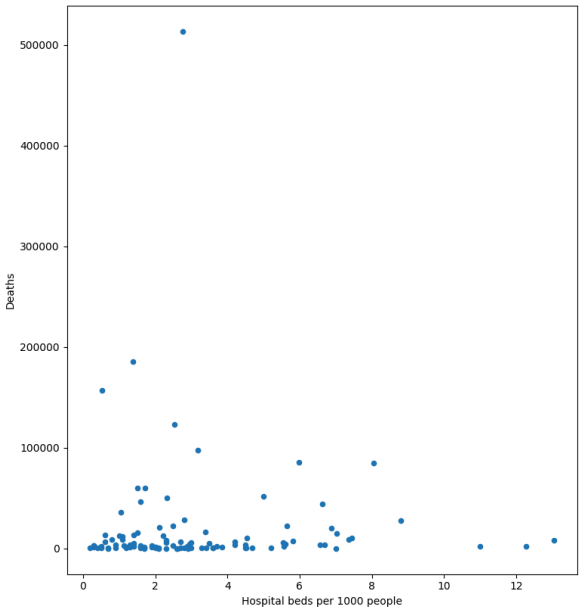
Τέλος μέσω των συναρτήσεων **subplots , plot, scatter** των **pyplot** και **pandas** αντίστοιχα δείχνω την κατανομή των cases και deaths ανά ήπειρο και τη γραφική τους σχέση με κάθε άλλο χαρακτηριστικό του dataset:


1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Cases
Europe
Africa
South America
Oceania
Asia
North America
Continent

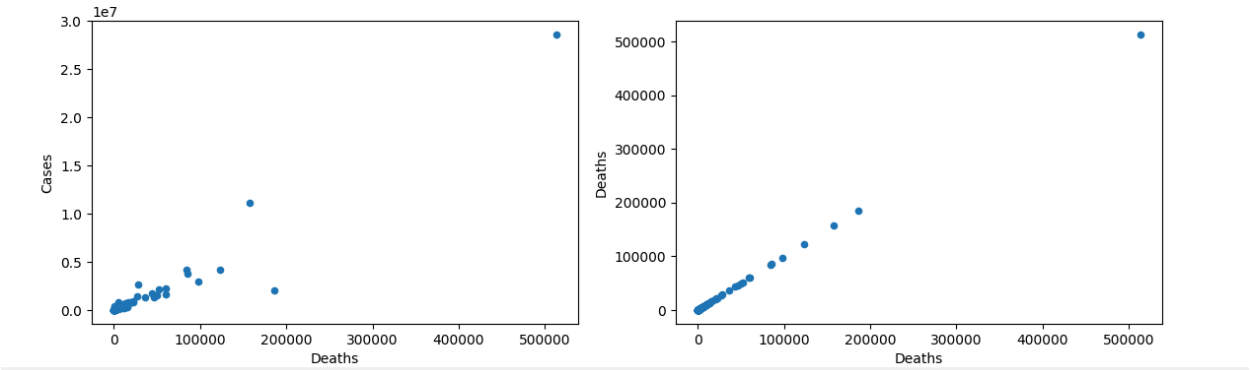

500000
400000
300000
200000
100000
0
Deaths
Europe
Africa
South America
Oceania
Asia
North America
Continent

και:


1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Cases
0
2
4
6
8
10
12
Hospital beds per 1000 people


500000
400000
300000
200000
100000
0
Deaths
0
2
4
6
8
10
12
Hospital beds per 1000 people

Και:

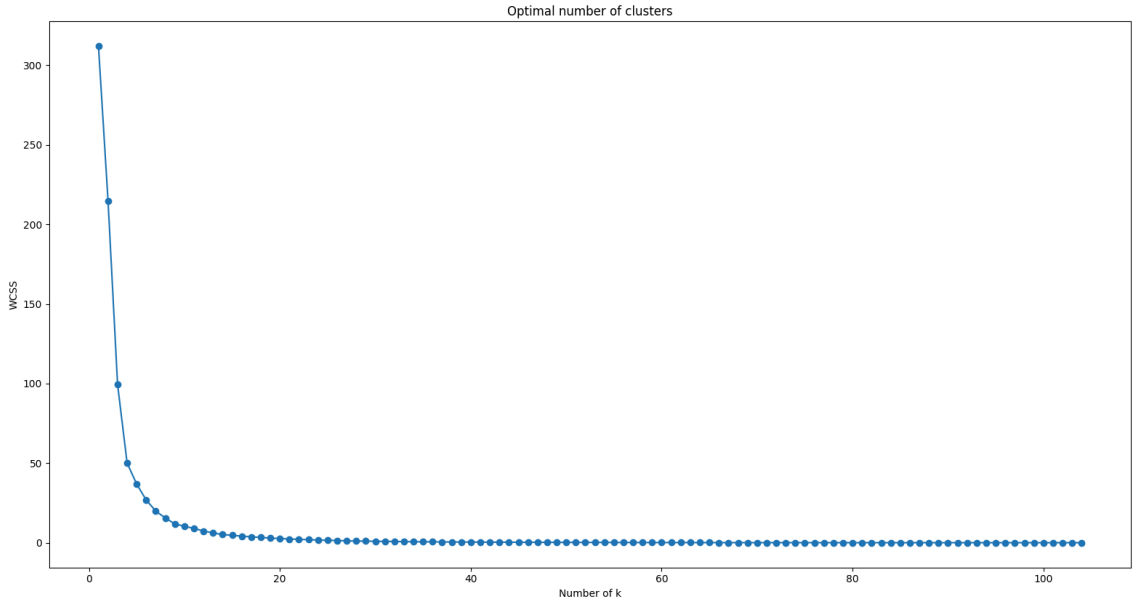


Περισσότερες λεπτομέρειες για τη λειτουργία του κώδικα που υλοποιεί τα παραπάνω, υπάρχουν στην τεκμηρίωση του σε σχόλια(documentation).

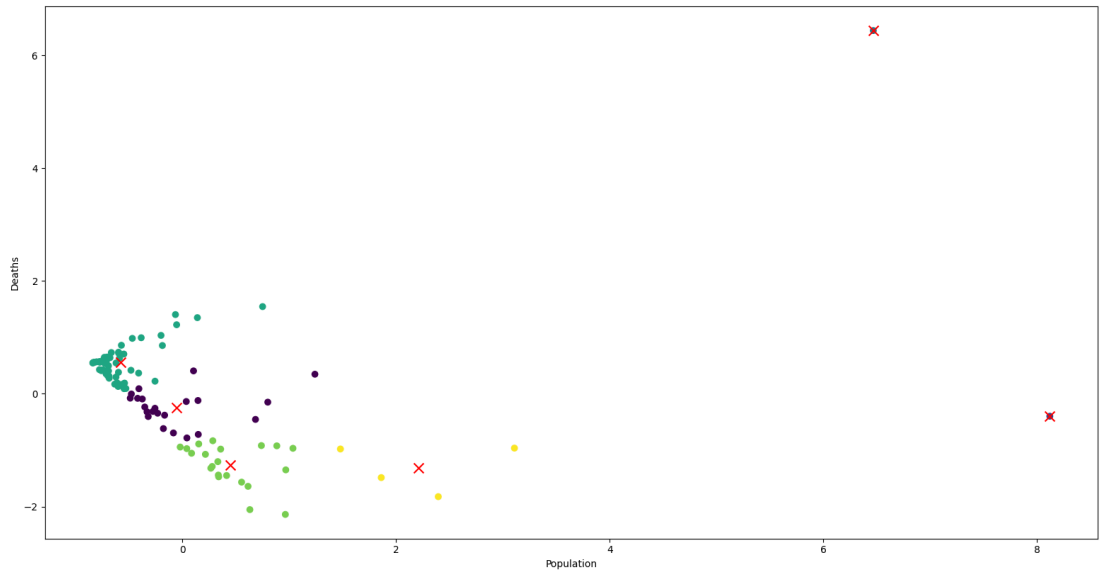
Ερώτημα 2

Για τη συσταδοποίηση των χωρών κάνω μία προεπεξεργασία πάνω στο σύνολο δεδομένων για αρχή. Συμπληρώνω τις ελλιπείς τιμές , αφαιρώ τις στήλες που δε με ενδιαφέρουν πέρα από το Population, Deaths, Cases.
Έπειτα ομαδοποιώ τις χώρες, με Cases/Deaths αυτή τη φορά το συνολικό τους αριθμό δηλαδή τον αριθμό(θανάτων, κρουσμάτων) της τελευταίας τους εγγραφής(τη μέγιστη τιμή δηλαδή). Οι στήλες Entity,Continent αφού ομαδοποιήσα το dataset βάσει αυτών χρησιμεύουν πια μόνο για δεικτοδότηση(indexing).
Κάνω την παραδοχή ότι το ποσοστό θνησιμότητας κάθε χώρας είναι το πηλίκο της διαίρεσης των συνολικών θανάτων (άρα και μέγιστο-τελικό) κάθε χώρας με τον πληθυσμό της. Δημιουργώ δηλαδή νέα attributes.
Για το ποσοστό θετικότητας παίρνω τον συνολικό αριθμό κρουσμάτων και τον διαιρώ με το σύνολο των τεστ που διεξήχθησαν. Για τον συνολικό αριθμό κρουσμάτων σε σχέση με τον πληθυσμό παίρνω τις στήλες Population, Cases και δημιουργώ τη νέα στήλη Cases/ Population.
Βρίσκοντας το βέλτιστο αριθμό των συστάδων μέσω της μεθόδου elbow, εκτελώντας KMeans πάνω στα μετασχηματισμένα δεδομένα(Standard Scaling για καλύτερη συσταδοποίηση) και απεικονίζοντας τα είτε στον 3D χώρο μέσω της **scatter3D** είτε στον 2D χώρο μέσω μείωσης της διαστατικότητας εφαρμόζοντας **principal component analysis** έχω τις ακόλουθες γραφικές παραστάσεις:

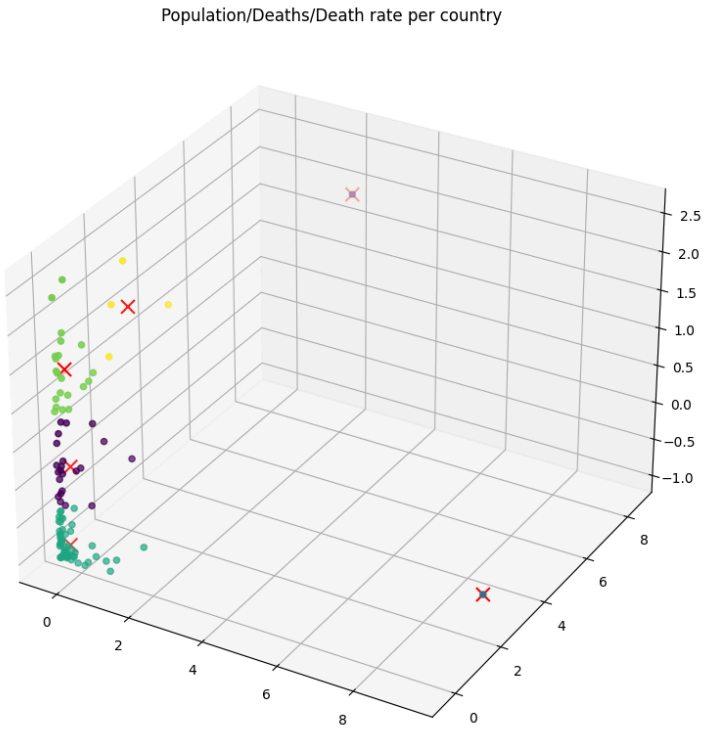
Elbow Method(Έχω 104 'σημεία' δηλαδή χώρες οπότε για τόσα θα τρέξω τη μέθοδο elbow):



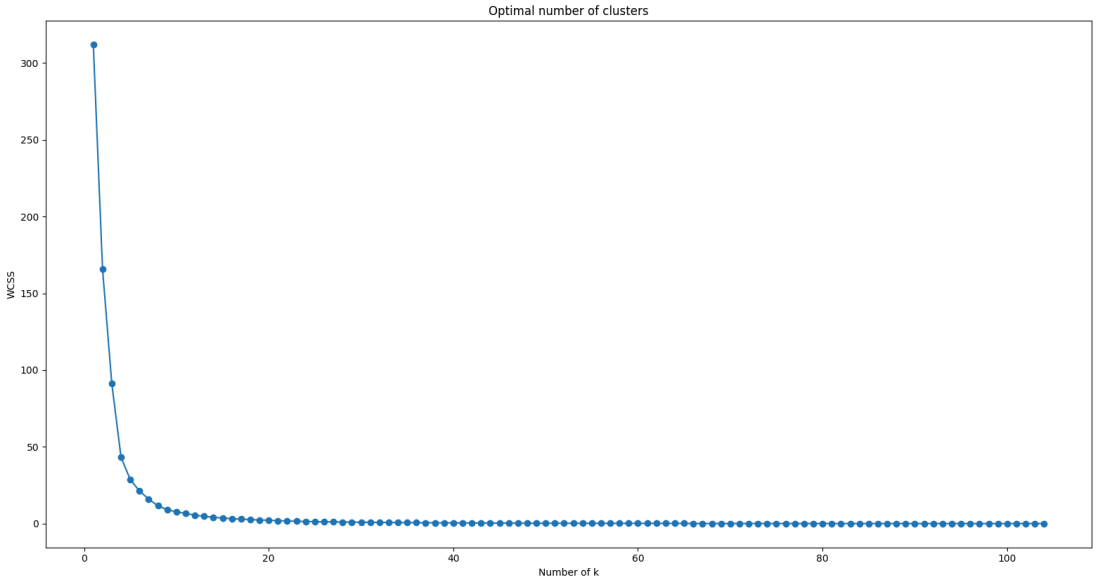
Παρατηρώ το βέλτιστο αριθμό για τις συστάδες να είναι 6.



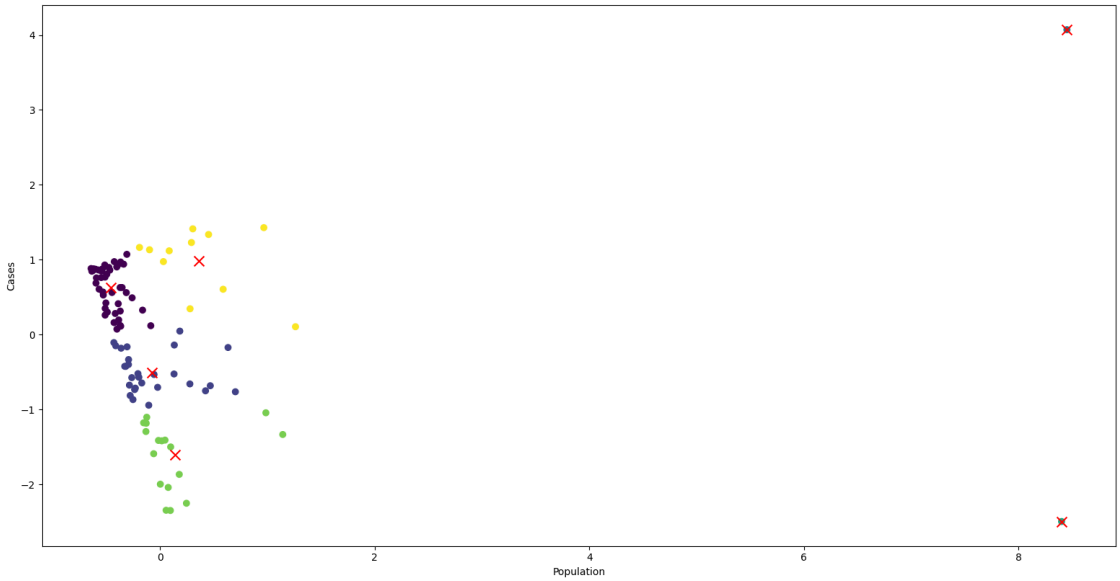
Και:



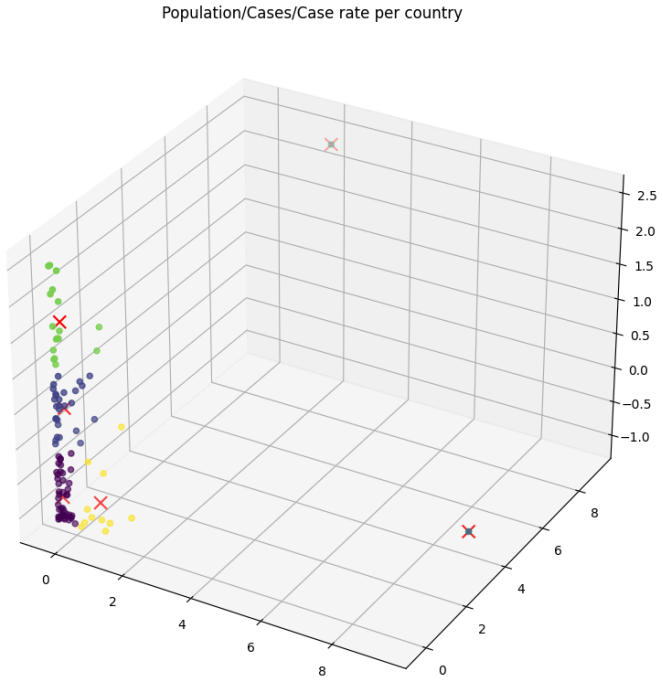
Για το ποσοστό θνησιμότητας και:



Με βέλτιστο k=6 και:

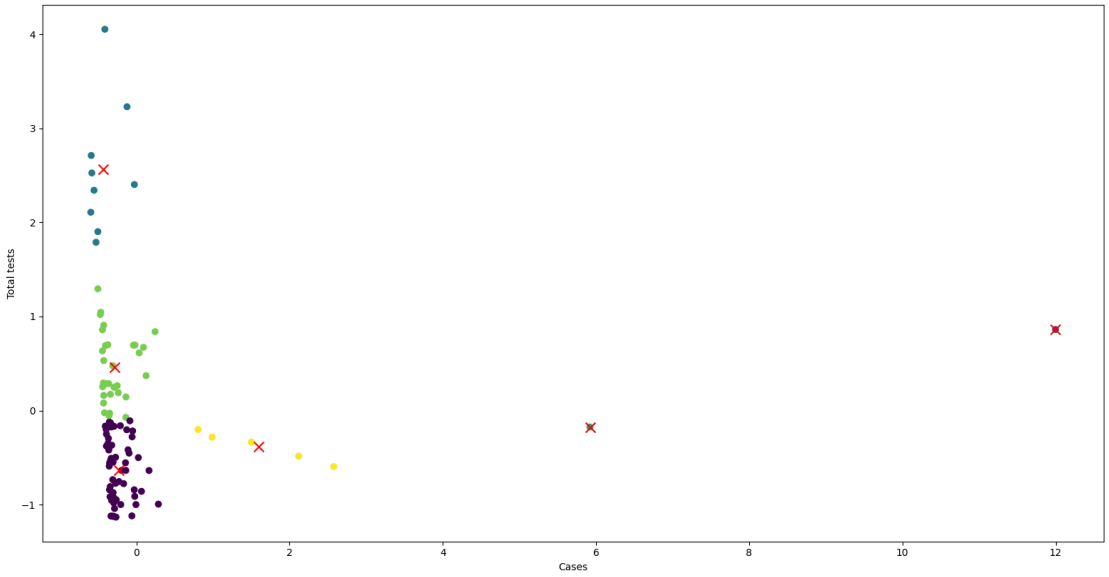


Και:

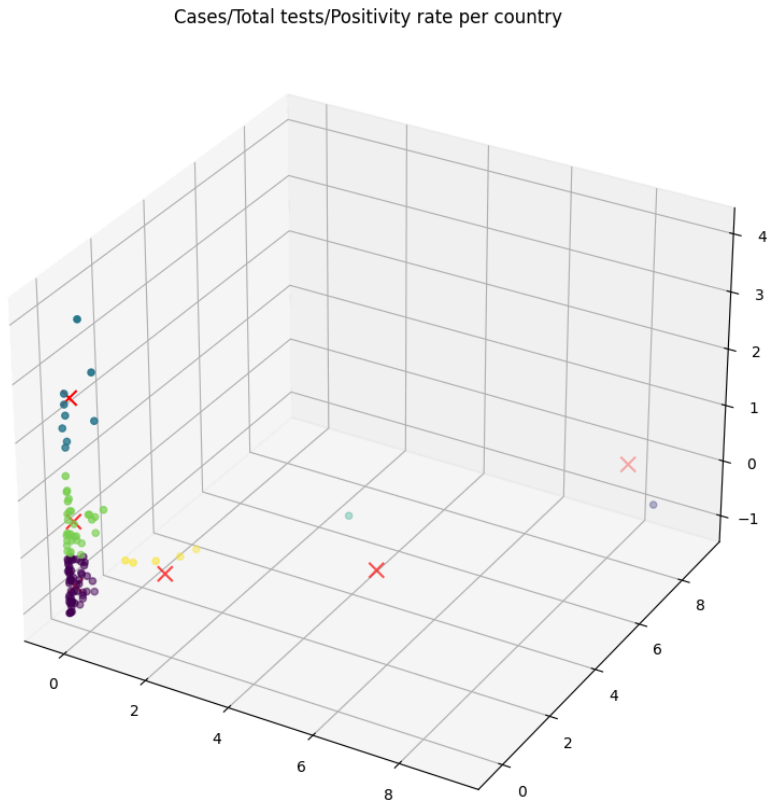


Για το συνολικό αριθμό κρουσμάτων σε σχέση με τον πληθυσμό.

Και :



Και:



Για το ποσοστό θετικότητας ορίζω ως πόσα από τα συνολικά τεστ βγήκαν θετικά δηλαδή συνολικά κρούσματα(Cases)/συνολικά test.

Βάσει και των δύο αυτών μετρικών (συνολικός αριθμός κρουσμάτων σε σχέση με τον πληθυσμό και θνησιμότητας) παρατηρώ δύο χώρες που ξεχωρίζουν αρνητικά(συγκεκριμένα πρόκειται για τις **Ηνωμένες Πολιτείες Αμερικής και την Ινδία**)και αποτελούν τους outliers(ακραίες περιπτώσεις). Οι δύο αυτές χώρες έχουν πολύ μεγαλύτερο πληθυσμό συγκριτικά με τις υπόλοιπες χώρες στο dataset και πολύ μεγαλύτερο αριθμό θανάτων/κρουσμάτων αλλά και συνολικό αριθμό τεστ συγκριτικά με τις άλλες χώρες(άρα και πολύ μεγαλύτερα ποσοστά θνησιμότητας, θετικότητας και συνολικά κρούσματα συγκριτικά με το πληθυσμό τους). Αυτά δηλαδή είναι τα κοινά τους χαρακτηριστικά. Εφόσον εντόπισα 6 ως βέλτιστο αριθμό clusters, αναγκαστικά τα δύο αυτά σημεία(USA,India) θα είναι δύο ξεχωριστές συστάδες.

Σημείωση: Για 'καλό' αριθμό συστάδων κάλλιστα θα μπορούσα να ορίσω και 5 συστάδες.

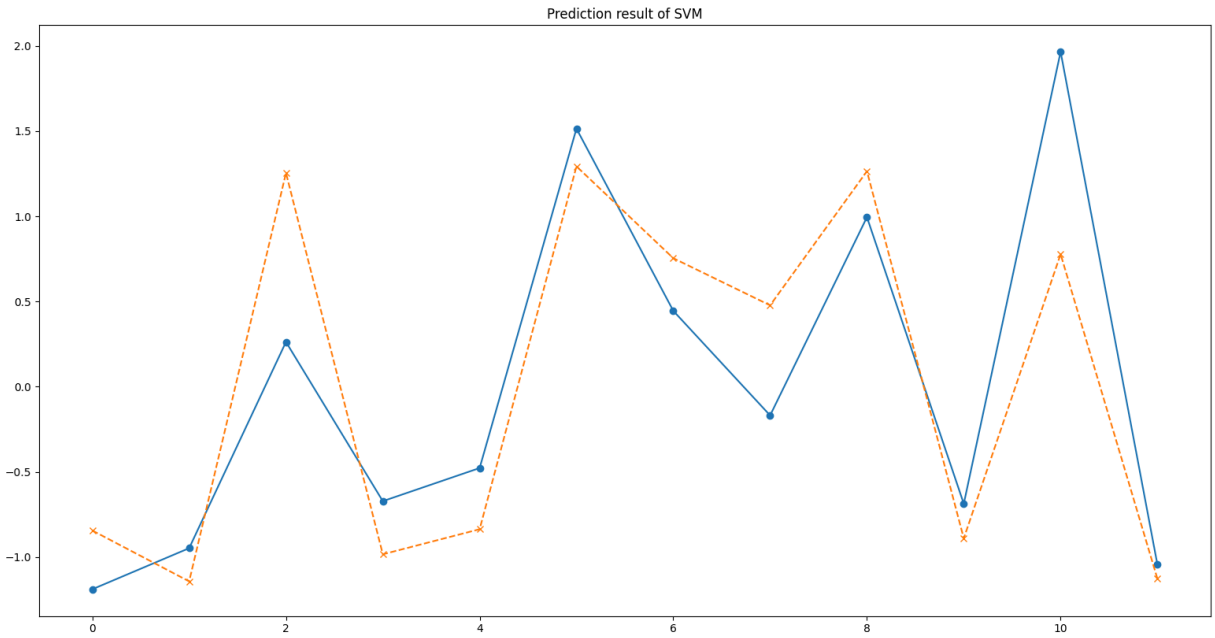
Ερώτημα 3

Για το τελευταίο ερώτημα όπως και με τα δύο προηγούμενα πάλι προ-επεξεργάζομαι τα δεδομένα συμπληρώνοντας τις τιμές που απουσιάζουν μέσω της **fillna()**. Έπειτα εφόσον η πρόβλεψη θέλω να γίνει για την ελλάδα φτιάχνω ένα Dataframe το οποίο περιλαμβάνει μόνο τις εγγραφές της ελλάδας μετά την 1/1/2021 μέσω των γραμμών:

```
df_greece = df_greece.drop(columns_to_drop,axis=1)
df_greece=df_greece[df_greece['Date']>= '2021-01-01'].
```

Ακόμη διαγράφω τις στήλες οι οποίες είναι ανεπιθύμητες για το regression που θα κάνω από το Entity μέχρι και το Population aged 65 and over (%).
Επίσης διαγράφω και το στοιχείο Deaths μιας και δεν βοηθά στον υπολογισμό του ποσοστού θετικότητας. Οπότε κρατάω μόνο το Population,Cases,Daily Tests και το Date. Στη συνέχεια για την πρόβλεψη των κρουσμάτων 3 μέρες μετά την τρέχουσα ημερομηνία δημιουργώ στις υπόλοιπες μεταβλητές 'καθυστερήσεις'(lags). Στην ουσία μέσω της **shift()** για τον υπολογισμό των κρουσμάτων 3 ημερών μετά μιας συγκεκριμένης ημερομηνίας χρησιμοποιώ τιμές των κρουσμάτων 1,2,3 μέρες πριν τη συγκεκριμένη ημερομηνία καθώς και των υπόλοιπων χαρακτηριστικών για να εκπαιδεύσω το μοντέλο.
Ακολούθως διασπάω το νέο dataframe σε δύο σύνολα X,Y όπου το X περιλαμβάνει τις ανεξάρτητες μεταβλητές(features) ενώ το Y τη μεταβλητή στόχος τα Cases(3 μέρες μετά). Έπειτα διασπάω τα δύο αυτά σύνολα σε training και testing sets το καθένα με αυθαίρετη αναλογία

80%-20%. Ακολούθως τα 4 αυτά σύνολα μετασχηματίζονται(**Standard Scaling είτε MinMax Scaling**) ώστε να έχει καλύτερη απόδοση ο αλγόριθμος παλινδρόμησης με **SVM.** Τα σύνολα Y_train, Y_test πρέπει να μετατραπούν σε δισδιάστατα arrays κάτι το οποίο κάνω μέσω της **reshape(-1,1).** Έπειτα κατασκευάζω ένα μοντέλο παλινδρόμησης με svm και μη γραμμικό πυρήνα **rbf** και το εκπαιδεύω μέσω των υποσυνόλων X_train,Y_train. Έπειτα μέσω της **predict()** η οποία παίρνει ως είσοδο το X_test (μετασχηματισμένο) προβλέπουμε την έξοδο δηλαδή την τιμή των Cases για 3 ημέρες μετά από κάθε ημερομηνία του dataset. Για να συγκρίνω τις αποκλίσεις πραγματικών τιμών κρουσμάτων (Cases) και αυτών που προέβλεψε το μοντέλο συγκρίνω το αποτέλεσμα της predict με το Y_test. Για την αξιολόγηση του μοντέλου χρησιμοποιώ την μετρική **r2_score**(πρόβλεψη τιμής,πραγματική τιμή). Για αυτή τη μετρική, τιμή στο 1.0 δηλώνει τέλεια πρόβλεψη ενώ στο 0.0 κακή πρόβλεψη(μικρή συσχέτιση) ενώ μπορεί να πάρει και αρνητικές τιμές(αυθαίρετα κακό μοντέλο). Μέσω των παραμέτρων του παλινδρομητή και αλλάζοντας το train-test split αλλάζει και η απόδοση της παλινδρόμησης.
Μία προσπάθεια παλινδρόμησης(ουσιαστικά κατηγοριοποίηση για συνεχείς μεταβλητές) :



Με r2_score :

```
R2 score is :
 0.7115174429165977
```

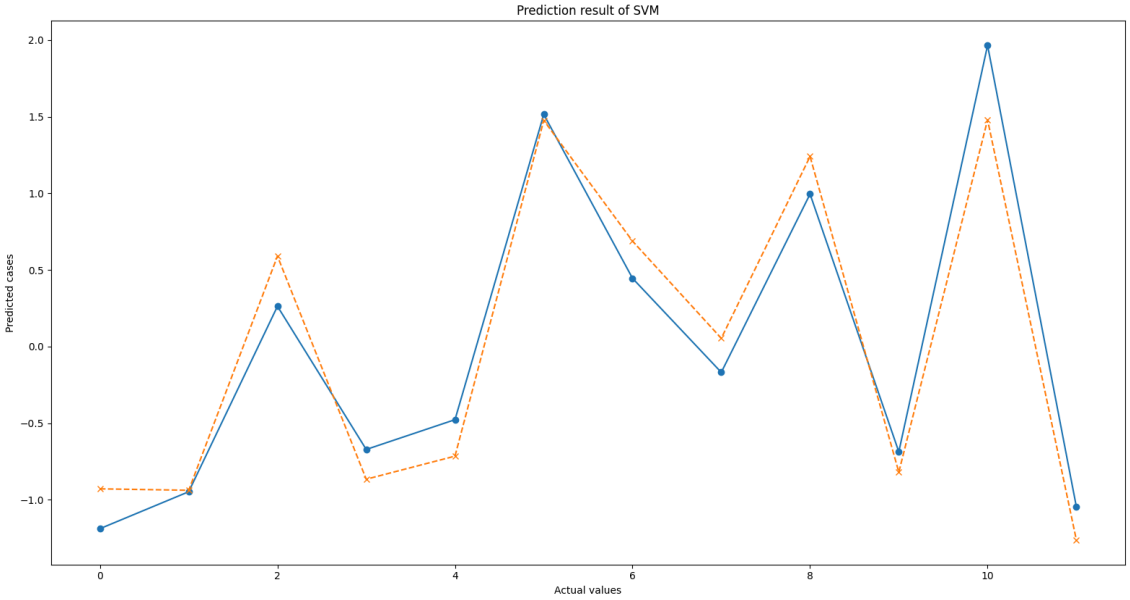
Το οποίο είναι ικανοποιητικό. Δηλαδή οι προβλέψεις μου ήταν αρκετά κοντά στις πραγματικές τιμές. Η παραπάνω παλινδρόμηση έγινε σε split 80-20 με πυρήνα(kernel) : rbf. Μία αλλαγή στον κώδικα η ακόλουθη:

```
svr_regressor.fit(X_train_scaled,Y_train_scaled)#vazo x_train,y_train-ekpaideyo to modelo -update: vazo tis scaled ekdoseis tous
```

Βελτιώνει ακόμα περισσότερο το r2_score σε :

```
R2 score is :
 0.9383178862917175
```

Και γραφικά:



Δε πρόλαβα να υλοποιήσω το RNN ωστόσο παρατίθεται στο αρχείο του κώδικα η βασική προετοιμασία των δεδομένων και η δημιουργία του μοντέλου του νευρωνικού δικτύου ως μία αρχική προσπάθεια.