

PROGETTO DATA SCIENCE

ANALISI DATI COVID COREA

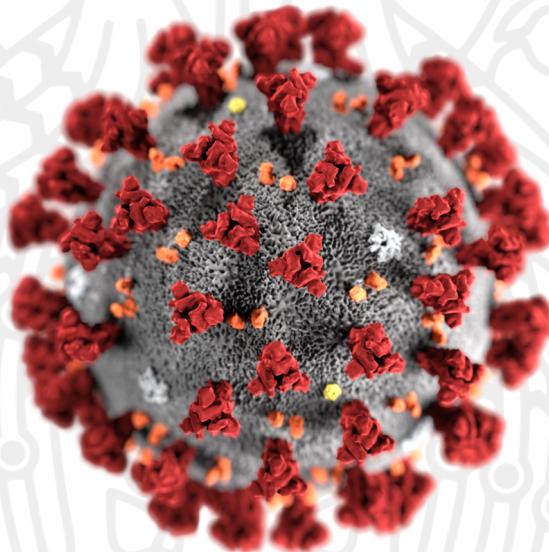
Professore:

Domenico Ursino

Dipartimento di Ingegneria dell'Informazione

Corso: Data Science

Università Politecnica delle Marche, Ancona



Di:

Claudio Menotta

Mattia Ippoliti

Italo Cervigni

Indice

1	Introduzione	1
1.1	Storia del Covid-19 in Corea del Sud	1
1.2	Introduzione a Pandas e Seaborn	3
1.3	Introduzione al dataset	3
1.4	Struttura del dataset	5
1.5	ETL	5
1.5.1	Case	5
1.5.2	PatientInfo	6
1.5.3	Time	10
1.5.4	Time_Age	11
1.5.5	Time_Gender	12
1.5.6	Time_Province	13
1.5.7	Region	13
1.5.8	Weather	15
1.5.9	Search_Trend	16
1.5.10	Seoul_Floating	17
1.5.11	Policy	18
2	Analisi Descrittiva	20
2.1	Grafici sull'epidemia in Corea	20

INDICE

2.2	Analisi sulle relazioni tra le variabili	33
3	Sklearn	38
3.1	Introduzione al tool	38
3.2	Divisioni in cluster	38
3.3	Classificazione	45
4	Statsmodels	53
4.1	Introduzione alla libreria	53
4.2	Analisi temporali e predizioni	53
4.2.1	Andamento dei test	54
4.2.2	Andamento degli spostamenti	61

1. Introduzione

Il progetto su cui si è incentrata questa relazione consiste nell’analisi di un dataset relativo ai dati covid in Sud Corea nel periodo che va dalla registrazione del primo caso (20 Gennaio 2020) alla fine di Giugno 2020.

1.1 Storia del Covid-19 in Corea del Sud

Per contenere l’epidemia di Covid-19 la Corea del Sud ha fatto ricorso alla tecnologia. Sulla base di esperienze precedenti, le autorità coreane hanno usato strumenti tecnologici innovativi per individuare le persone che erano state esposte al virus e che avrebbero potuto essere state infettate.

La Corea del Sud ha infatti già affrontato un’epidemia simile a quella del Covid-19 nel 2015, quando il paese è stato colpito da un altro coronavirus, che provoca una malattia chiamata Mers, o Middle east respiratory syndrome (i due virus sono simili tra loro, ma non uguali). Il paese asiatico aveva così già sviluppato conoscenze, sistemi e normative per affrontare la diffusione di un virus nuovo, i cui effetti clinici sono ignoti, di cui non si conoscono le caratteristiche epidemiologiche e per il quale non esiste un farmaco specifico.

Nel caso del Covid-19 si è cercato di individuare tutte le persone potenzialmente infettate. Il primo passo è stato intervistare il paziente e determinarne gli spostamenti nel periodo precedente alla scoperta dell’infezione. Il metodo tradizionale consiste in un’intervista. Ma in Corea del Sud i dati così raccolti sono stati verificati e integrati con quelli ottenuti con altri sistemi.

Gps e carte di credito, per esempio, sono stati consultati gli archivi degli accessi agli ambulatori e alle farmacie, dove le persone in possesso dell’assicurazione sanitaria la-

CAPITOLO 1: INTRODUZIONE



Figura 1.1: Foto scattata a Seoul durante la pandemia

sciano traccia del loro passaggio. Il percorso del paziente è stato verificato anche con i dati gps del telefono cellulare, a disposizione delle autorità di polizia. Sono stati inoltre usati gli archivi delle carte di credito, conservati dagli enti finanziari, e le registrazioni delle videocamere di sorveglianza.

Con questi strumenti sono stati ricostruiti gli spostamenti dei singoli pazienti. È stato possibile individuare le persone con cui erano entrati in contatto, il tipo di contatto e anche lo stato di salute del paziente, per esempio se tossiva. Le persone potenzialmente esposte al contagio sono state contattate e sono state stabilite misure di quarantena. Alcuni siti sono stati isolati.

I motivi principali del successo del metodo coreano per la lotta al virus sono stati:

- Trasparenza: Le capacità di test e diagnostica della Corea sono eccezionali e forniscono risultati in poche ore. Esegue oltre 20.000 test al giorno e rilascia i dati istantaneamente.
- Screening e quarantena robusti: La Corea mappa e traccia in modo aggressivo i casi infetti. Le auto-quarantine sono rigorosamente monitorate e applicate: i trasgressori rischiano l'irregolarità e le multe.
- Tecnologia: La Corea utilizza la tecnologia IT e siti di test drive-through per ridurre al minimo i contatti. Le app mobili tengono traccia delle condizioni e dei

movimenti delle persone infette.

- Controllo rigoroso: La Corea proibisce alle persone che sono state in contatto con casi confermati di uscire dalla Corea durante il periodo di auto-quarantena di 14 giorni.
- Trattamento: La Corea fornisce cure mediche avanzate ai pazienti confermati. Il trattamento è gratuito sia per i coreani che per i cittadini stranieri.

1.2 Introduzione a Pandas e Seaborn

Pandas è una libreria che fornisce strutture dati e strumenti di analisi dei dati ad alte prestazioni e di facile utilizzo per il linguaggio di programmazione Python. Bisogna immaginare pandas come la struttura o l'organizzazione di tutti i dati su cui si andrà a lavorare. Esso è costruito su NumPy ed è destinato ad integrarsi bene in un ambiente di calcolo scientifico con molte altre librerie di terze parti.

Pandas funziona attraverso due strutture dati: Series (1-dimensionale) e DataFrame (2-dimensionale). Un DataFrame può essere immaginato come un contenitore per le serie ed una Series è un contenitore per gli scalari.

Al contrario per interpretare i dati, bisogna costruire dei grafici. In quest'ottica, organizzare i dati con pandas è molto utile perché possiamo sfruttare Seaborn. Seaborn fornisce un'interfaccia di alto livello a Matplotlib, dove i punti di forza di Seaborn sono il rendere attraenti i grafici, visualizzare le distribuzioni in modo facile e flessibile e una visualizzazione delle informazioni da matrici e DataFrame.

Tuttavia, Seaborn è un'aggiunta, non un sostituto di Matplotlib. Ci sono alcuni accorgimenti nella costruzione di grafici che richiedono ancora Matplotlib.

1.3 Introduzione al dataset

Il dataset conta dei seguenti 10 diversi file .csv:

CAPITOLO 1: INTRODUZIONE

COVID-19 COREA	
Caratteristiche	Descrizione
Case	Dati sui casi di infezione da COVID-19 in Corea del Sud
PatientInfo	Dati epidemiologici di pazienti COVID-19 in Corea del Sud
TimeProvince	Dati di serie temporali dello stato COVID-19 in termini di provincia in Corea del Sud
TimeGender	Dati di serie temporali dello stato COVID-19 in termini di sesso in Corea del Sud
TimeAge	Dati di serie temporali dello stato COVID-19 in termini di età in Corea del Sud
Region	Posizione e dati statistici delle regioni della Corea del Sud
Weather	Dati del meteo nelle regioni della Corea del Sud
SearchTrend	Dati di tendenza delle parole chiave cercate in NAVER, uno dei più grandi siti di ricerca della Corea del Sud

CAPITOLO 1: INTRODUZIONE

SeoulFloating	Dati sulla popolazione in movimento a Seoul, Corea del Sud (da SK Telecom Big Data Hub)
Policy	Dati della politica del governo per COVID-19 in Corea del Sud

1.4 Struttura del dataset

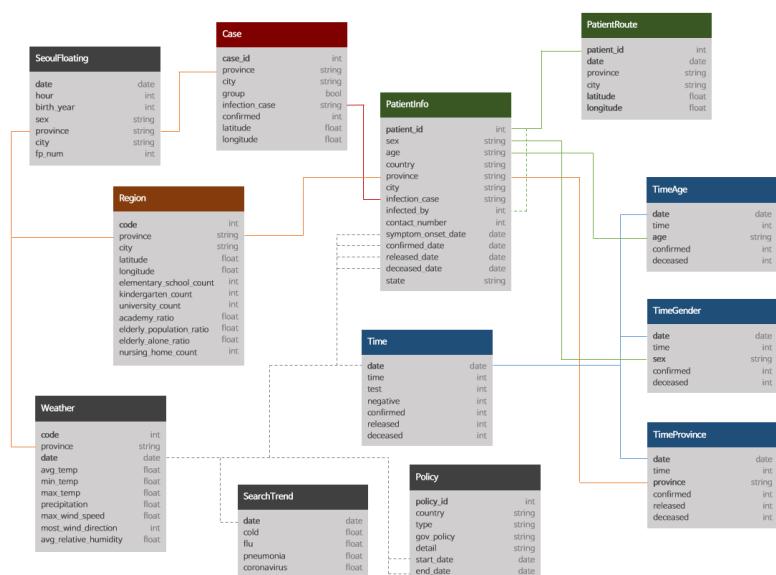


Figura 1.2: Il colore significa che hanno proprietà simili. Se una linea è collegata tra le colonne, significa che i valori delle colonne sono parzialmente condivisi. Le linee tratteggiate indicano una scarsa rilevanza

1.5 ETL

1.5.1 Case

All'intero di questo file .csv si trovano i dati di casi di infezione da COVID-19 in Corea del Sud.

Nello studiare tale tabella, si è andati inizialmente a cambiare dei valori all'interno delle colonne *"latitude"* e *"longitude"* che presentavano dei caratteri *"-"* per indicare come

CAPITOLO 1: INTRODUZIONE

1. case_id: l'ID del caso di infezione

- case_id = region_code + case_number
- province: Città Speciale / Città Metropolitana / Provincia (-do)
- city Città (-si) / Paese (-gun) / Distretto (-gu)
- group: TRUE: infezione di gruppo / FALSE: non gruppo
 - Se il valore è "TRUE" in questa colonna, il valore di "infezioni_casi" indica il nome del gruppo.
 - I valori denominati "contatto con il paziente", "afflusso dall'estero" e "ecc." Non sono infezioni di gruppo.
 - Il valore "from another city" significa che dove è iniziata l'infezione di gruppo è un'altra città.
- caso_infezione: il caso di infezione (il nome del gruppo o altri casi)
 - Il valore "afflusso dall'estero" significa che l'infezione proviene da un altro paese.
 - Il valore "ecc." Include i casi individuali, i casi in cui la classificazione della pertinenza è in corso dopo l'indagine e i casi sotto inchiesta.
- confirmed: il numero accumulato dei confermati
- latitude: la latitudine del gruppo
- longitude: la longitudine del gruppo

il dato non fosse presente. Tali stringhe sono state sostituite con valori NaN grazie alla libreria *numpy*. Gli unici valori nulli del file riguardano infatti tali due colonne che presentavano 109 valori nulli. Essendo la tabella molto piccola (174 colonne) si è preferito non andare ad eliminare tali valori nulli per non snaturare la tabella stessa. L'ultima operazione fatta è stata quella di convertire tali due colonne da stringa a float attraverso il comando *astype*. Il risultato finale è il seguente:

case_id	province	city	group	infection_case	confirmed	latitude	longitude
1000001	Seoul	Yongsan-gu	True	Itaewon Clubs	139	37.538621	126.992652
1000002	Seoul	Gwanak-gu	True	Richway	119	37.48208	126.901384
1000003	Seoul	Guro-gu	True	Guro-gu Call Center	95	37.508163	126.884387
1000004	Seoul	Yangcheon-gu	True	Yangcheon Table Tennis Club	43	37.546061	126.874209
1000005	Seoul	Dobong-gu	True	Day Care Center	43	37.679422	127.044374
1000006	Seoul	Guro-gu	True	Manmin Central Church	41	37.481059	126.894343
1000007	Seoul	from other city	True	SMR Newly Planted Churches Group	36	NaN	NaN
1000008	Seoul	Dongdaemun-gu	True	Dongan Church	17	37.592888	127.056766
1000009	Seoul	from other city	True	Coupan Logistics Center	25	NaN	NaN
1000010	Seoul	Gwanak-gu	True	Wangsung Church	30	37.481735	126.930121

1.5.2 PatientInfo

All'interno di questa tabella si trovano i dati relativi ai pazienti che hanno riscontrato il virus in Corea del Sud:

CAPITOLO 1: INTRODUZIONE

1. Patient_id: l'ID del paziente

- Pati_id = codice_regionale + numero_paziente
- Esistono due tipi di numero_paziente 1) local_num: il numero fornito dal governo locale. 2) global_num: il numero fornito dal KCDC.
- sesso: il sesso del paziente
- età: l'età del paziente
 - 0 : 0 ~ 9
 - 10 : 10 ~ 19 ...
 - 90: 90 ~ 99
 - 100 : 100 ~ 109
- paese: il paese del paziente
- provincia: la provincia del paziente
- città: la città del paziente
- Infezione_caso: il caso di infezione
- Infetto_by: l'ID di chi ha infettato il paziente
 - Questa colonna fa riferimento alla colonna "Patient_id".
- contact_number: il numero di contatti con le persone
- sintom_onset_date: la data di insorgenza dei sintomi
- Confirm_date: la data di conferma
- Release_date: la data di rilascio
- deceased_date: la data del decesso
- state: isolato / rilasciato / deceduto
 - isolato: essere isolato in ospedale
 - rilasciato: essere dimesso dall'ospedale
 - defunto: essere deceduto

La tabella in questione presenta tutti i dati in forma di stringhe ed è la seguente:

patient_id	sex	age	country	province	city	infection_case	infected_by	contact_number	symptom_onset_date	confirmed_date	released_date	deceased_date	state
1000000001	male	50s	Korea	Seoul	Gangseo-gu	overseas inflow	NaN	75	2020-01-22	2020-01-23	2020-02-05	NaN	released
1000000002	male	30s	Korea	Seoul	Jungnang-gu	overseas inflow	NaN	31	NaN	2020-01-30	2020-03-02	NaN	released
1000000003	male	50s	Korea	Seoul	Jongno-gu	contact with patient	2002000001	17	NaN	2020-01-30	2020-02-19	NaN	released
1000000004	male	20s	Korea	Seoul	Mapo-gu	overseas inflow	NaN	9	2020-01-26	2020-01-30	2020-02-15	NaN	released
1000000005	female	20s	Korea	Seoul	Seongbuk-gu	contact with patient	1000000002	2	NaN	2020-01-31	2020-02-24	NaN	released

Quello che si è andato a fare allora è stato innanzitutto un casting da stringa a intero per la colonna *age*, dopo aver rimosso la 's' a fine stringa.

Eseguendo un'istruzione *info* si nota come all'interno di questa tabella, al contrario della precedente, ci siano molti valori che sono nulli.

Per quanto riguarda i valori mancanti nella colonna *age* si è deciso di farne una **media** e di andarli a riempire, nel caso la cella fosse vuota, con tale valore.

Al contrario le città mancanti vengono riempite con il valore della **moda** della città per la provincia in questione. Per farlo è bastato crearsi inizialmente un vettore binario, con valori true per le righe che hanno il valore della città mancante, e una lista delle

CAPITOLO 1: INTRODUZIONE

#	Column	Non-Null Count	Dtype
0	sex	4043 non-null	object
1	age	3785 non-null	float64
2	country	5165 non-null	object
3	province	5165 non-null	object
4	city	5071 non-null	object
5	infection_case	4246 non-null	object
6	infected_by	1346 non-null	object
7	contact_number	791 non-null	object
8	symptom_onset_date	690 non-null	object
9	confirmed_date	5162 non-null	object
10	released_date	1587 non-null	object
11	deceased_date	66 non-null	object
12	state	5165 non-null	object

varie province. Si crea così un ciclo for che scorre tutte le provincie all'interno della lista creata e successivamente, attraverso un altro ciclo for interno al primo, si scorrono tutte le righe del dataset che hanno come valore sulla colonna *province* il valore della provincia corrente, e per tali righe si sostituisce il valore null nella colonna *city* con la città **moda** per quella provincia.

Listing 1.1: Moda delle Province

```
p_city = p_info.city.isnull()
#vettore con true e false delle varie citta mancanti dal dataset

lista_prov = p_info[p_info['city'].isnull()].province.unique().tolist()
#creo una lista con le varie province

for prov in lista_prov:
    if prov=='Gwangju':
        #non abbiamo valori di citta per questa provincia,
        #perciò non potremo farne la media
        moda_prov='Gwangju'
    else :
        moda_prov1=p_info[(p_info['province']==prov)].city.mode()
        #moda delle varie province
        moda_prov=moda_prov1.loc[0]

    i=0
    for cond in p_info['province']==prov:
        if cond:
            if (p_city[p_city.index[i]]).any():
                p_info.city[p_info.index[i]]=moda_prov
            #riempio quella citta mancante con la moda per la provincia
```

i=i+1

Per quanto riguarda le colonne vuote della *infection_case*, le si sono andate a sostituire attraverso una *fillna* con la stringa *unknown*.

Similmente per quanto riguarda le colonne vuote della *infection_by*, le si sono andate a sostituire attraverso una *fillna* con la stringa *patient not found*.

Al contrario per la colonna *contact_number* si è optato per una **drop** di tale colonna dato che i valori presenti nel dataset erano troppo pochi da poter dare un significato effettivo a tale informazione.

Per quanto riguarda i valori mancanti nella colonna *released* si è deciso di studiare inanzitutto la media dei giorni passati in ospedale dopo il giorno che il paziente ha contratto il virus. Una volta ottenuta tale media si calcola la data di rilascio come la data di conferma che il paziente ha contratto il virus sommata con la media dei giorni di rilascio, solo per le righe che contengono il valore *released* nella colonna "State", che non ha valori nulli. In totale i valori presenti all'interno della colonna '*released_date*' sono 2937 che coincidono con il numero di valori '*released*' all' interno della colonna "State".

Listing 1.2: Data di rilascio

```
dateFormatter = "%Y-%m-%d"
p_released= p_info.released_date.isnull()

somma=timedelta(0)
k=0

for i in p_info.index:
    if not p_released[i].any():
        data1 = datetime.strptime(p_info.released_date[i], dateFormatter)
        data2 = datetime.strptime(p_info.confirmed_date[i], dateFormatter)
        diff = data1-data2
        somma = somma + diff
        k = k+1

media_rilascio = somma/k
media_rilascio_giorni = timedelta(media_rilascio.days)

p_state = (p_info.state == 'released')
#abbiamo riempito solo le righe 'released'
```

```

for i in p_info.index:
    if p_state[i].any():
        if p_released[i].any():
            data_confirmed = datetime.strptime(p_info.confirmed_date[i], dateFormatter)
            p_info.released_date[i] = data_confirmed + media_rilascio_giorni

```

Per i valori mancanti nella data di morte si è utilizzato lo stesso ragionamento fatto per la data di rilascio, andando a studiare il tempo trascorso tra la data di morte in media dei pazienti dopo che gli è stato diagnosticato il virus. Per farlo si vanno a riempire solo le righe che hanno il valore *deceased* nella colonna "State"

Infine per le tre colonne che riguardano le date di rilascio, di conferma e di morte e di avvertimento dei primi sintomi, le si andrà a trasformare in date temporali attraverso il comando *pd.to_datetime*. Se si esegue il comando info avremo che il file csv finale avrà una forma di questo tipo:

#	Column	Non-Null Count	Dtype
0	sex	4043 non-null	object
1	age	5165 non-null	int64
2	country	5165 non-null	object
3	province	5165 non-null	object
4	city	5165 non-null	object
5	infection_case	5165 non-null	object
6	infected_by	5165 non-null	object
7	symptom_onset_date	689 non-null	datetime64[ns]
8	confirmed_date	5165 non-null	datetime64[ns]
9	released_date	2937 non-null	datetime64[ns]
10	deceased_date	78 non-null	datetime64[ns]
11	state	5165 non-null	object

1.5.3 Time

Nella seguente tabella sono conservati i dati della serie temporale dello stato di contagio, morti e tamponi in Corea del Sud ed è strutturato in questo modo:

- data: AAAA-MM-GG
- time: Time (0 = AM 12:00 / 16 = PM 04:00)

|| L'ora in cui KCDC mette a disposizione le informazioni è stata modificata dalle 04:00 alle 12:00 dal 2 marzo.

- test: il numero di test accumulati

|| Un test è una diagnosi di infezione.

- negativo: il numero accumulato di risultati negativi
- confermato: il numero di risultati positivi accumulati
- rilasciato: il numero accumulato di rilasci
- deceduto: il numero accumulato di decessi

e se si esegue il comando **head**, estraendo la testa del dataset, quello che uscirà sarà:

	date	time	test	negative	confirmed	released	deceased
0	2020-01-20	16	1	0	1	0	0
1	2020-01-21	16	1	0	1	0	0
2	2020-01-22	16	4	3	1	0	0
3	2020-01-23	16	22	21	1	0	0
4	2020-01-24	16	27	25	2	0	0

La tabella in questione non ha valori NaN perciò l'unica operazione fatta è il casting della colonna *date*, attraverso la funzione *pd.to_datetime*, in data temporale invece che stringa, mentre il resto dei valori è di tipo intero, come si osserva se si esegue il comando *info*:

#	Column	Non-Null Count	Dtype
0	date	163 non-null	datetime64[ns]
1	time	163 non-null	int64
2	test	163 non-null	int64
3	negative	163 non-null	int64
4	confirmed	163 non-null	int64
5	released	163 non-null	int64
6	deceased	163 non-null	int64

1.5.4 Time_Age

Tale file è utile nello studio delle serie temporali di dati sullo stato del COVID-19 in termini di età in Corea del Sud ed è strutturato in questa maniera:

- data: AAAA-MM-GG
- time: Time
- età: l'età dei pazienti
- confermato: il numero accumulato dei confermati
- defunto: il numero accumulato dei defunti

e se si esegue il comando **head**, estraendo la testa del dataset, quello che uscirà sarà:

	date	time	age	confirmed	deceased
0	2020-03-02	0	0s	32	0
1	2020-03-02	0	10s	169	0
2	2020-03-02	0	20s	1235	0
3	2020-03-02	0	30s	506	1
4	2020-03-02	0	40s	633	1

Le operazioni da fare in tale dataset sono due: la prima è un casting della colonna *date* da stringa a dato temporale e la seconda operazione è la rimozione della lettera "s" nella *age* e la conversione di quest'ultimo a tipo intero. Facendo una *info* si avrà:

#	Column	Non-Null Count	Dtype
0	date	1089 non-null	datetime64[ns]
1	time	1089 non-null	int64
2	age	1089 non-null	int64
3	confirmed	1089 non-null	int64
4	deceased	1089 non-null	int64

1.5.5 Time_Gender

Tale file è utile nello studio delle serie temporali di dati sullo stato del COVID-19 in termini di sesso in Corea del Sud ed è strutturato in questa maniera:

- data: AAAA-MM-GG
- time: Time
- sesso: il sesso dei pazienti
- confermato: il numero accumulato dei confermati
- defunto: il numero accumulato dei defunti

e se si esegue il comando **head**, estraendo la testa del dataset, quello che uscirà sarà:

	date	time	sex	confirmed	deceased
0	2020-03-02	0	male	1591	13
1	2020-03-02	0	female	2621	9
2	2020-03-03	0	male	1810	16
3	2020-03-03	0	female	3002	12
4	2020-03-04	0	male	1996	20

Anche in questo caso l'unica operazione da effettuare è un casting della *date* da stringa a dato temporale e facendo una *info* si avrà:

#	Column	Non-Null Count	Dtype
0	date	242 non-null	datetime64[ns]
1	time	242 non-null	int64
2	sex	242 non-null	object
3	confirmed	242 non-null	int64
4	deceased	242 non-null	int64

1.5.6 Time_Province

L'ultima tabella sui dati temporali è quella della Time Province che indica una serie di dati sullo stato del COVID-19 in termini di provincia in Corea del Sud.

	date	time	province	confirmed	released	deceased
0	2020-01-20	16	Seoul	0	0	0
1	2020-01-20	16	Busan	0	0	0
2	2020-01-20	16	Daegu	0	0	0
3	2020-01-20	16	Incheon	1	0	0
4	2020-01-20	16	Gwangju	0	0	0

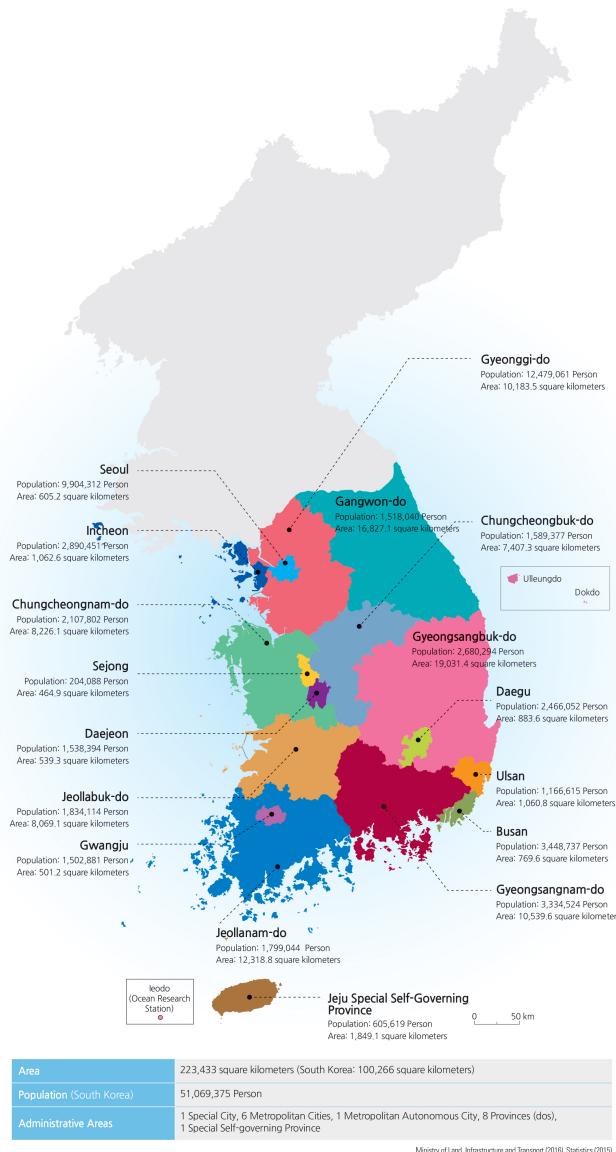
Anche in questo caso l'unica operazione da effettuare è un casting della *date* da stringa a dato temporale.

1.5.7 Region

All'interno di tale file si trova la posizione in termini di latitudine e longitudine e alcuni dati statistici delle regioni della Corea del Sud. In particolare si andrà a dividere i distretti amministrativi del Sud Corea in due gruppi:

- **Livello Superiore (provinciale):**
 - **Città Speciale:** Seoul
 - **Città Metropolitane:** Busan / Daegu / Daejeon / Gwangju / Incheon / Ulsan
 - **Province (-do):** Gyeonggi-do / Gangwon-do / Chungcheongbuk-do / Chungcheongnam-do / Jeollabuk-do / Jeollanam-do / Gyeongsangbuk-do / Gyeongsangnam-do
- **Livello Inferiore (livello municipale):**
 - **Città (-si)**
 - **Paesi (-gun)**
 - **Distretti (-gu)**

CAPITOLO 1: INTRODUZIONE



La struttura della tabella è la seguente:

- codice: il codice della regione
- provincia: Città Speciale / Città Metropolitana / Provincia (-do)
- città: Città (-si) / Paese (-gun) / Distretto (-gu)
- latitudine: la latitudine della visita
- longitudine: la longitudine della visita
- elementary_school_count: il numero di scuole elementari
- kindergarten_count: il numero di asili
- university_count: il numero di università
- academy_ratio: il rapporto delle accademie
- old_population_ratio: il rapporto tra la popolazione anziana
- senior_alone_ratio: il rapporto tra le famiglie anziane che vivono sole
- nursing_home_count: il numero di case di cura

CAPITOLO 1: INTRODUZIONE

e se si esegue il comando `head` si ottiene:

code	province	city	latitude	longitude	elementary_school_count	kindergarten_count	university_count	academy_ratio	elderly_population_ratio	elderly_alone_ratio	nursing_home_count
10000	Seoul	Seoul	37.566953	126.977977	607	830	48	1.44	15.38	5.8	22739
10010	Seoul	Gangnam-gu	37.518421	127.047222	33	38	0	4.18	13.17	4.3	3088
10020	Seoul	Gangdong-gu	37.530492	127.123837	27	32	0	1.54	14.55	5.4	1023
10030	Seoul	Gangbuk-gu	37.639938	127.025508	14	21	0	0.67	19.49	8.5	628
10040	Seoul	Gangseo-gu	37.551166	126.849506	36	56	1	1.17	14.39	5.7	1080

Dato che in questo caso il dataset è senza valori NaN non sarà necessario fare alcuna operazione di ETL, facendo una `info` si vede infatti:

#	Column	Non-Null Count	Dtype
0	province	244 non-null	object
1	city	244 non-null	object
2	latitude	244 non-null	float64
3	longitude	244 non-null	float64
4	elementary_school_count	244 non-null	int64
5	kindergarten_count	244 non-null	int64
6	university_count	244 non-null	int64
7	academy_ratio	244 non-null	float64
8	elderly_population_ratio	244 non-null	float64
9	elderly_alone_ratio	244 non-null	float64
10	nursing_home_count	244 non-null	int64

1.5.8 Weather

La seguente tabella riguarda i dati meteo delle regioni della Corea del Sud. Potrebbe sembrare inutile studiare tali dati, tuttavia si è dimostrato che non solo le condizioni di inquinamento e quindi lo smog ad alta concentrazione di polveri sottili possano aver accresciuto il rischio di contagio e favorito un decorso più rapido della malattia, ma anche quelle metereologiche possono influenzare l'andamento della curva dei contagi.

Ma in che modo le piogge possono influire sulle epidemie? Le precipitazioni e più precisamente l'elevata umidità rallentano la circolazione di agenti patogeni come quello dell'influenza nell'ambiente, ovviamente in luoghi chiusi è comunque facile che una persona infetta trasmetta il patogeno. Molti dei virus che infettano le vie respiratorie, seguono una stagionalità, ovvero i casi di infezione diminuiscono con l'arrivo dell'estate e il cambiamento delle temperature. Il dataset è formato dalle seguenti colonne:

CAPITOLO 1: INTRODUZIONE

- codice: il codice della regione
- provincia: Città Speciale / Città Metropolitana / Provincia (-do)
- data: AAAA-MM-GG
- avg_temp: la temperatura media
- min_temp: la temperatura più bassa
- max_temp: la temperatura più alta
- precipitazione: la precipitazione giornaliera
- max_wind_speed: la velocità massima del vento
- most_wind_direction: la direzione del vento più frequente
- avg_relative_humidity: l'umidità relativa media

e se si esegue il comando `head` si ottiene:

	code	province	date	avg_temp	min_temp	max_temp	precipitation	max_wind_speed	most_wind_direction	avg_relative_humidity
0	10000	Seoul	2016-01-01	1.2	-3.3	4.0	0.0	3.5	90.0	73.0
1	11000	Busan	2016-01-01	5.3	1.1	10.9	0.0	7.4	340.0	52.1
2	12000	Daegu	2016-01-01	1.7	-4.0	8.0	0.0	3.7	270.0	70.5
3	13000	Gwangju	2016-01-01	3.2	-1.5	8.1	0.0	2.7	230.0	73.1
4	14000	Incheon	2016-01-01	3.1	-0.4	5.7	0.0	5.3	180.0	83.9

In questo caso, dopo la prima solita operazione di casting della colonna `date` da stringa a data si riempiono i valori mancanti della colonna `avg_temp` del dataset con la media dei valori calcolati per quella determinata provincia. Stesso procedimento sarà eseguito per le colonne `min_temp`, `max_temp`, `max_wind_speed`, `most_wind_direction` e `avg_relative_humidity`. In questo modo facendo una `info` si nota come il dataset è completo (assenza di valori NaN):

#	Column	Non-Null Count	Dtype
0	code	26271	non-null int64
1	province	26271	non-null object
2	date	26271	non-null datetime64[ns]
3	avg_temp	26271	non-null float64
4	min_temp	26271	non-null float64
5	max_temp	26271	non-null float64
6	precipitation	26271	non-null float64
7	max_wind_speed	26271	non-null float64
8	most_wind_direction	26271	non-null float64
9	avg_relative_humidity	26271	non-null float64

1.5.9 Search_Trend

Tale dataset va a studiare la percentuale delle ricerche della parola "coronavirus" nel corso degli anni all'interno del motore di ricerca sud coreano NAVER. Il dataset parte dall'anno 2016 fino ad arrivare a Giugno del 2020. Infatti il coronavirus non nasce nel

CAPITOLO 1: INTRODUZIONE

2019 con la sua variante COVID-19 bensì era già presente negli anni passi nella sua variante MERS-Cov.

La MERS (acronimo di Middle East Respiratory Syndrome in inglese) o sindrome respiratoria mediorientale da coronavirus (conosciuta anche come influenza cammello) è una patologia causata dal coronavirus MERS-CoV. All'interno del motore di ricerca Coreano tale parola era già cercata dal 2016 poiché (come già accennato nel paragrafo 1.1) la Corea aveva affrontato tale epidemia nel 2015. Interessante è studiare come cambia nel corso dei giorni tale ricerca nel motore browser della Corea, in base all'aumento o alla diminuzione dei casi e se si riesce a trovare qualche correlazione. Il dataset è strutturato con le seguenti colonne:

- date: AAAA-MM-GG
- cold: il volume di ricerca di "raffreddore" in lingua coreana
- flu: il volume di ricerca di "flu" in lingua coreana
- pneumonia: il volume di ricerca di "polmonite" in lingua coreana
- coronavirus: il volume di ricerca del "coronavirus" in lingua coreana

e se si esegue il comando *head* si ottiene:

	date	cold	flu	pneumonia	coronavirus
0	2016-01-01	0.11663	0.05590	0.15726	0.00736
1	2016-01-02	0.13372	0.17135	0.20826	0.00890
2	2016-01-03	0.14917	0.22317	0.19326	0.00845
3	2016-01-04	0.17463	0.18626	0.29008	0.01145
4	2016-01-05	0.17226	0.15072	0.24562	0.01381

Anche in questo caso non sono necessarie operazioni di ETL dato che il dataset non presenta valori NaN.

1.5.10 Seoul_Floating

Nella seguente tabella sono riportati i dati sulla popolazione fluttuante a Seoul, provincia della Corea del Sud (da SK Telecom Big Data Hub). Per popolazione fluttuante si intende la popolazione che in un determinato giorno ad una determinata ora circolava in quella zona della città. Questo è possibile dato che il governo Coreano utilizza tali mezzi tecnologici per combattere il crimine e che si rilevano utili anche per un analisi

CAPITOLO 1: INTRODUZIONE

approfondita dei dati per queste situazioni "fuori dal comune", andando però a minare sulla privacy dei singoli individui. In particolare il dataset presenta tali colonne:

- date: AAAA-MM-GG
- hour: ora
- birth_year: l'anno di nascita della popolazione fluttuante
- sex: il sesso della popolazione fluttuante
- province: Città Speciale / Città Metropolitana / Provincia (-do)
- city: Città (-si) / Paese (-gun) / Distretto (-gu)
- fp_num: il numero di popolazione fluttuante

e se si esegue il comando *head* si ottiene:

	date	hour	birth_year	sex	province	city	fp_num
0	2020-01-01	0	20	female	Seoul	Dobong-gu	19140
1	2020-01-01	0	20	male	Seoul	Dobong-gu	19950
2	2020-01-01	0	20	female	Seoul	Dongdaemun-gu	25450
3	2020-01-01	0	20	male	Seoul	Dongdaemun-gu	27050
4	2020-01-01	0	20	female	Seoul	Dongjag-gu	28880

In questo caso l'unica operazione da effettuare sarà un casting da stringa a data temporale della colonna *date*. A questo punto il file csv avrà una struttura di questo tipo:

#	Column	Non-Null Count	Dtype
0	date	1084800 non-null	datetime64[ns]
1	hour	1084800 non-null	int64
2	birth_year	1084800 non-null	int64
3	sex	1084800 non-null	object
4	province	1084800 non-null	object
5	city	1084800 non-null	object
6	fp num	1084800 non-null	int64

1.5.11 Policy

Tale dataset è molto interessante dato che fornisce le date di inizio e di fine con i dettagli dei vari provvedimenti effettuati dal governo Coreano per la lotta al Covid-19, dove le colonne sono:

CAPITOLO 1: INTRODUZIONE

- policy_id: l'ID della policy
- paese: il paese che ha implementato la politica
- tipo: il tipo di polizza
- gov_policy: la politica del governo
- dettaglio: il dettaglio della polizza
- data_inizio: la data di inizio della polizza
- end_date: la data di fine della polizza

e se si esegue il comando `head` si ottiene:

	country	type	gov_policy	detail	start_date	end_date
policy_id						
1	Korea	Alert	Infectious Disease Alert Level	Level 1 (Blue)	2020-01-03	2020-01-19
2	Korea	Alert	Infectious Disease Alert Level	Level 2 (Yellow)	2020-01-20	2020-01-27
3	Korea	Alert	Infectious Disease Alert Level	Level 3 (Orange)	2020-01-28	2020-02-22
4	Korea	Alert	Infectious Disease Alert Level	Level 4 (Red)	2020-02-23	NaN
5	Korea	Immigration	Special Immigration Procedure	from China	2020-02-04	NaN

In questo dataset ci sono alcuni dati mancanti nella colonna `detail` riguardante i dettagli dei vari provvedimenti, che si andranno a riempire attraverso una `fillna` con la voce *not known*. Infine come al solito si farà un casting da stringa a data temporale delle colonne `start_date` e `end_date`, riempendo i valori mancanti della colonna `end_date` (che indicano che il provvedimento è ancora in vigore) con l'ultima data presente sul dataset. Facendo una `info` si potrà vedere come nel dataset non sono presenti valori `NaN`.

#	Column	Non-Null Count	Dtype
0	country	61 non-null	object
1	type	61 non-null	object
2	gov_policy	61 non-null	object
3	detail	61 non-null	object
4	start_date	61 non-null	datetime64[ns]
5	end_date	61 non-null	datetime64[ns]

2. Analisi Descrittiva

Dopo una prima parte di ETL necessaria per rendere i vari dataset utilizzabili per estrarne informazioni, si inizia con la vera e propria parte di analisi. Inizialmente ci si concentrerà soprattutto su un’analisi puramente descrittiva e diagnostica per poi passare, successivamente, ad un’analisi predittiva e prescrittiva dell’andamento e delle conseguenze della pandemia di Covid-19 in Corea del Sud.

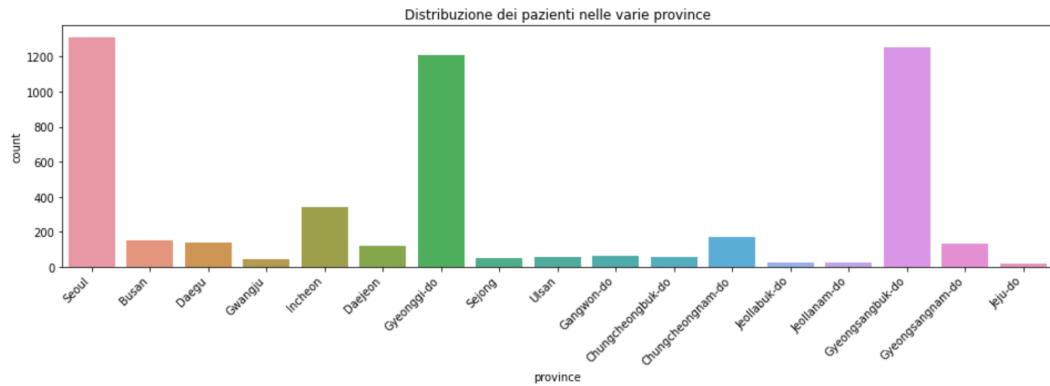
Per prima cosa ci si è voluti concentrare sull’inizio della pandemia, visualizzando la prima data in cui si sono registrati contagi nel Paese, ovvero il 20 gennaio 2020. Per farlo si è utilizzato il dataset “time”, in particolare accedendo al primo termine della colonna “date”, visto che le date sono ordinate in senso cronologico e quindi il primo elemento corrisponde al giorno in cui si è avuto il primo contagio.

2.1 Grafici sull’epidemia in Corea

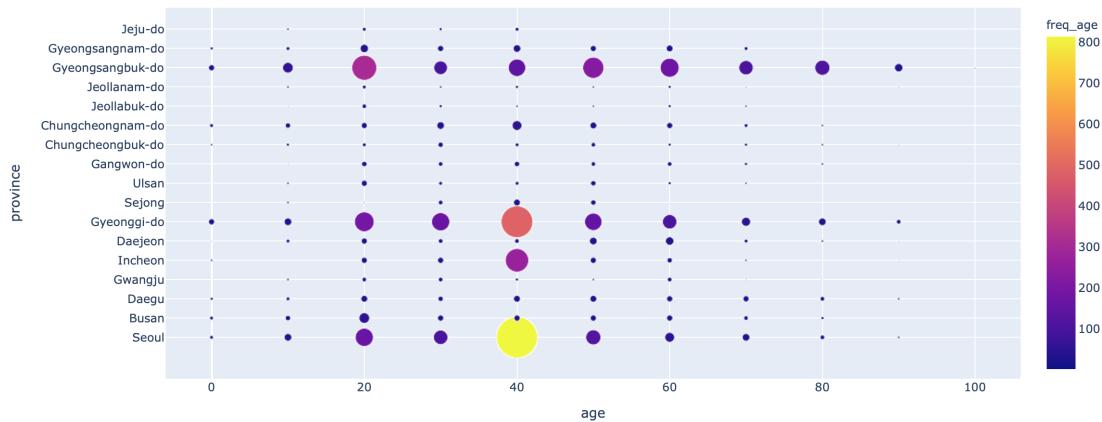
In questo istogramma si è voluto visualizzare quali province della Corea del Sud sono state maggiormente colpite dalla pandemia. Si nota immediatamente che le tre province più colpite sono state Seoul, Gyeonggi-do e Gyeongsangbuk-do, che sono anche tra le province con più abitanti. In particolare è Seoul che ha il triste primato di aver avuto più contagi, con oltre 1300 casi, essendo la provincia della Corea del Sud con maggior densità di abitanti. Per i resto delle province si è avuto un numero molto inferiore di contagi fortunatamente, tutti sotto i 400 casi. Per realizzare questo istogramma sono stati presi i dati dal dataset con i dati dei pazienti, cioè “p-info”, e in particolare la sua colonna “province”.

In questo scatterplot sono visualizzati i contagi che si sono avuti per fascia d’età e per provincia, quindi ogni bolla rappresenta gli individui di quella provincia e di quell’età

CAPITOLO 2: ANALISI DESCRITTIVA



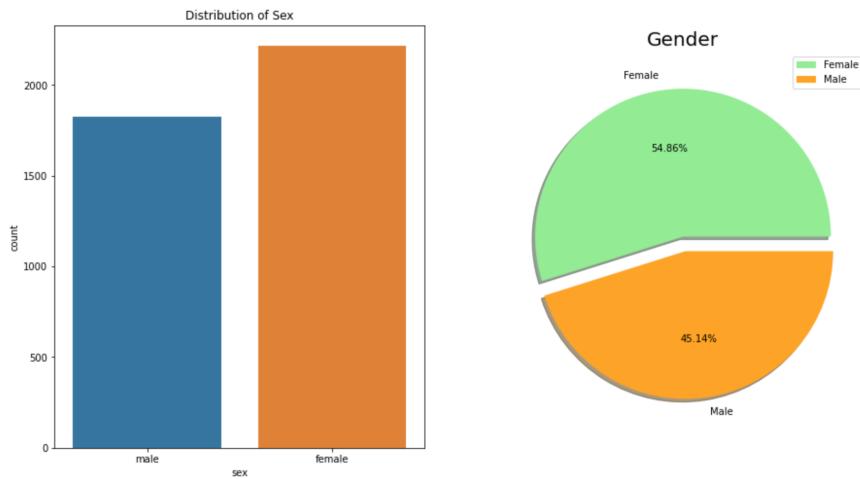
e la dimensione delle bolle è proporzionale al numero di persone totale con quei valori. Il colore delle bolle è basato su un gradiente in base al numero totale di casi di interesse. Concentrandoci sulle tre province coreane più colpite dalla pandemia, cioè Seoul, Gyeonggi-do e Gyeongsangbuk-do, si può notare che le prime due confermano i dati nazionali, cioè che si sono avuti più contagi tra i quarantenni e in generale nelle fasce d'età intermedie, invece si differenzia un po' Gyeonsangbuk-do in cui si è avuta la maggior parte dei contagi tra i ventenni (309 casi). Il grafico è stato ottenuto aggiungendo al dataset “p-info” una colonna contenente per ogni riga il numero totale di contagi delle persone aventi stessa età e stessa provincia del paziente a cui corrisponde la riga, e tale valore è stato usato per determinare la grandezza e il colore di ciascuna bolla.



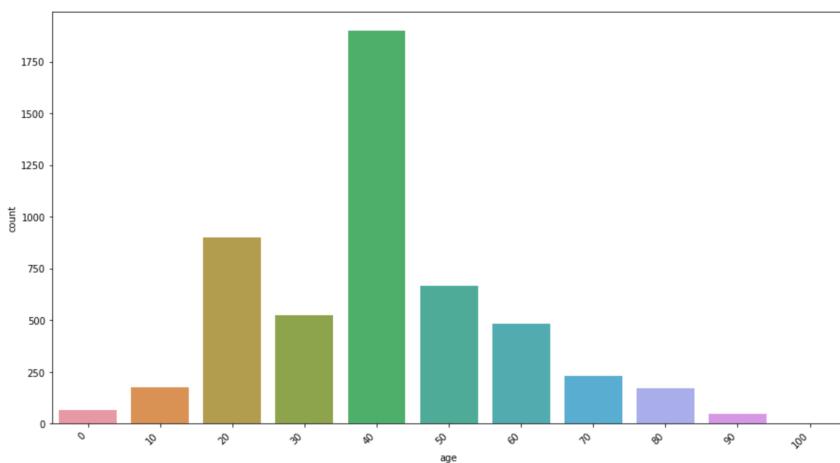
Si hanno successivamente un istogramma e un diagramma a torta in cui si mostra la distribuzione dei pazienti a livello di sesso. Si osserva dall'istogramma che si hanno avuti più contagi tra le donne, con oltre 2200 casi, rispetto a quelli tra gli uomini che sono stati circa 1800. Questi numeri corrispondono in percentuale a poco meno del 55% di donne e poco più del 45% di uomini, come si nota dal diagramma a torta a fianco. Per realizzare questi due grafici è stato usato sempre il dataset con le informazioni dei

CAPITOLO 2: ANALISI DESCRITTIVA

pazienti, cioè “p-info”, in particolare la sua colonna “sex”.

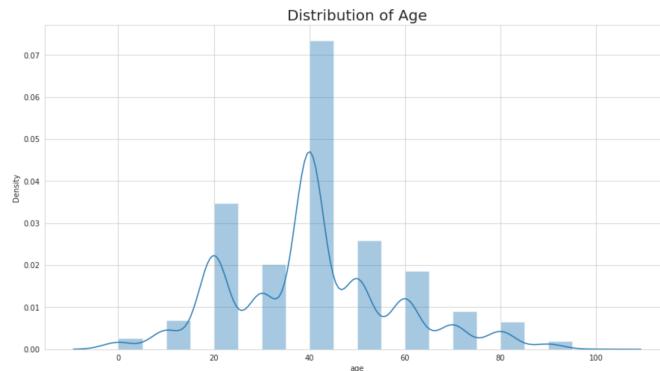


In questo ulteriore istogramma vengono mostrate le fasce d’età più colpite dal Covid-19 in Corea del Sud, si osserva che sono state colpite maggiormente le fasce d’età intermedie, in particolare i quarantenni con più di 1800 casi seguiti a distanza dai ventenni con circa 800 casi. Questo è dovuto al fatto che le persone appartenenti a queste fasce d’età tendono più a uscire per motivi di lavoro o studio e ad avere contatti sociali, al contrario dei più anziani che invece tendono più a rimanere a casa essendo per lo più persone in pensione (oltre al timore di essere contagiati che ha spinto molti anziani a restare a casa). Per realizzare questo istogramma sono stati presi i dati dal dataset con i dati dei pazienti, cioè “p-info”, e in particolare la sua colonna “age”.

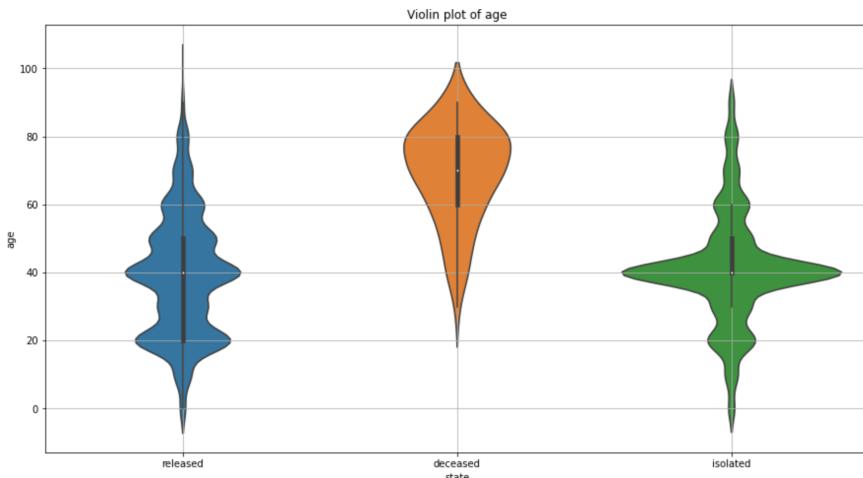


In questo distplot è stata rappresentata nuovamente la distribuzione dei contagi nelle varie fasce d’età, tuttavia non più in termini di valore assoluto, ma di densità, infatti è anche riportata una curva che mostra la densità di distribuzione che si ha per i vari

valori. Anche per realizzare questo distplot sono stati presi i dati dal dataset con i dati dei pazienti, cioè “p-info”, e in particolare la sua colonna “age”.



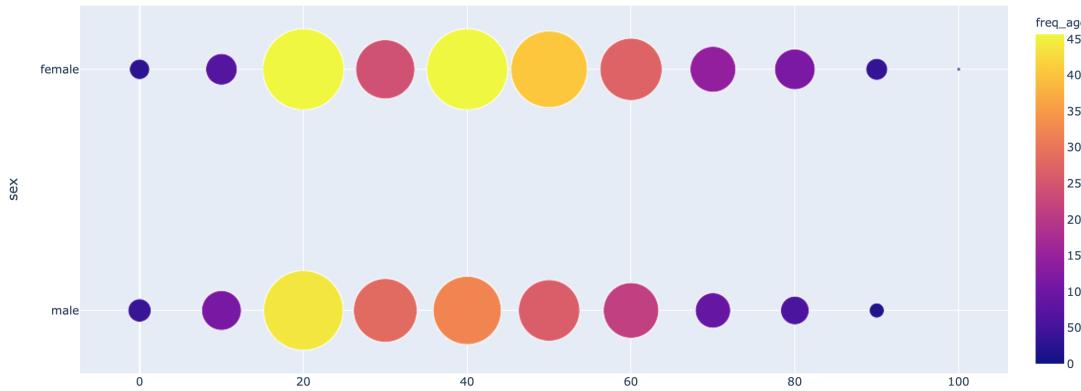
Questo **volinplot** mostra le distribuzioni dei pazienti deceduti, rilasciati e isolati tra le varie fasce d'età, indicando per ognuna delle distribuzioni i valori corrispondenti ai quartili e alle mediane. Come si può osservare la mediana dei deceduti corrisponde ai settant'anni, mentre per gli isolati e i rilasciati si ha in corrispondenza dei quarant'anni. Ciò è facile da spiegare perché la mortalità del virus è maggiore per le persone più anziane mentre, come visto in precedenza, le persone più colpite sono state quelle di quarant'anni, che tuttavia, essendo più giovani, hanno maggiore resistenza al virus. Per realizzare questo violinplot è stato preso il dataset con i dati dei pazienti, cioè “p-info”, e in particolare le sue colonne “released”, “deceased” e “isolated”.



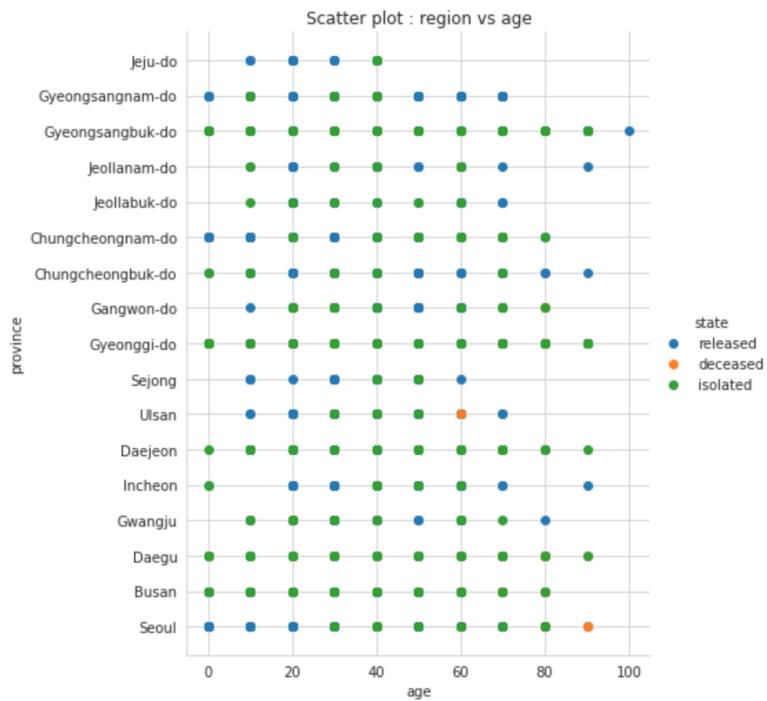
In questo scatterplot invece si è voluto mostrare quanti casi si hanno avuti in base al sesso e alla fascia d'età, quindi ogni bolla rappresenta gli individui di quel sesso e di quell'età e la dimensione delle bolle è proporzionale al numero di persone totale con quei valori. Il colore delle bolle è basato su un gradiente in base al numero totale di casi di interesse. Si osserva che tra le femmine le fasce di età più contagiate sono state quelle

CAPITOLO 2: ANALISI DESCRITTIVA

delle quarantenni e delle ventenni, entrambe con 457 casi totali, mentre tra i maschi sono i ventenni con un totale di 440 casi. Il grafico è stato ottenuto aggiungendo al dataset “p-info” una colonna contenente per ogni riga il numero totale di contagi delle persone aventi stesso sesso e stessa età del paziente a cui corrisponde la riga.

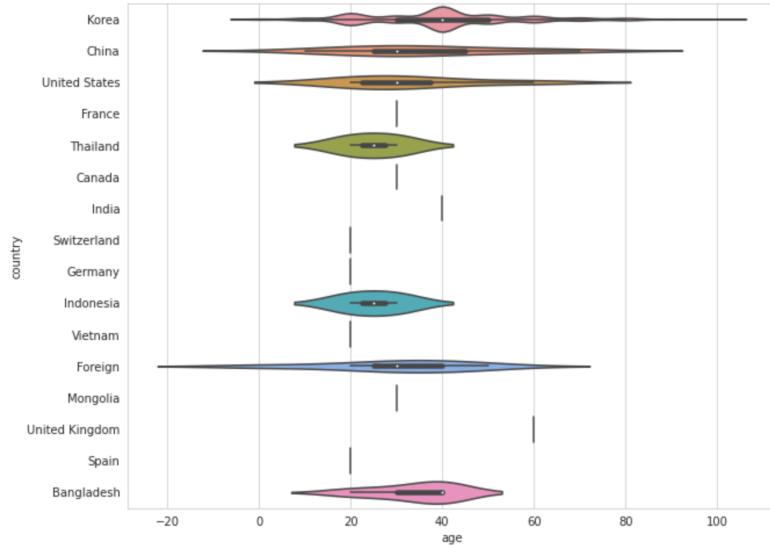


In questo scatterplot si mostra per ogni provincia e ogni fascia d'età se i maggior numero di pazienti è deceduto, è stato isolato o rilasciato (in base al colore dei vari punti). Si osserva, come era facile aspettarsi che nelle fasce d'età fino ai cinquant'anni (più giovani e quindi più resistenti al virus) i pazienti di tutte le province sono stati soprattutto rilasciati o isolati, mentre in due casi, ovvero nella fascia dei sessant'anni a Ulsan e dei novant'anni a Seoul, la maggior parte dei pazienti sono deceduti purtroppo.

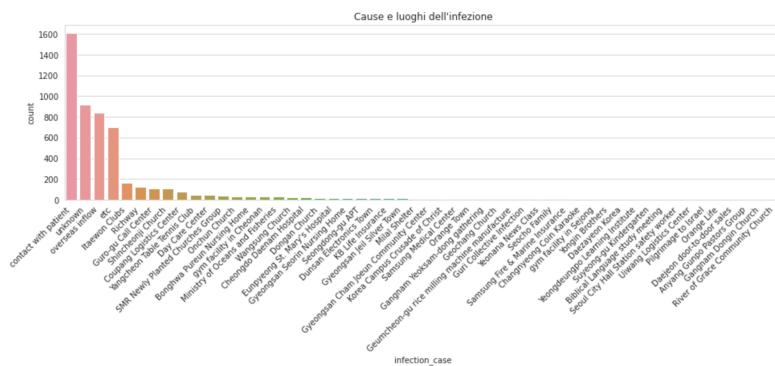


CAPITOLO 2: ANALISI DESCRITTIVA

In questo violinplot vengono analizzati i casi in base al Paese di provenienza, in particolare si mostrano le distribuzioni dei casi nelle varie fasce d'età per Paese di provenienza. I positivi originari della Corea coprono tutte le fasce d'età ovviamente, ma in maniera simile quasi tutte le fasce d'età sono coperte anche dai pazienti provenienti dalla vicina Cina, paese molto colpito dal virus. Molti sono stati anche i positivi tra gli Americani in visita in Corea. Infine compaiono anche Thailandia, Bangladesh e Indonesia, anche se coprono fasce d'età più ristrette.

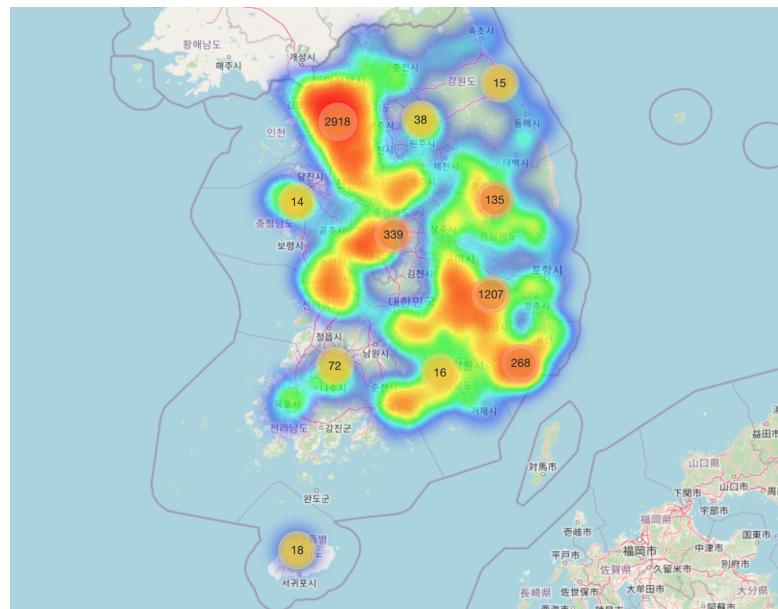


Questo istogramma mostra quali sono state le principali cause e i principali luoghi in cui è avvenuta l'infezione nel Paese. Si nota che nella maggior parte dei casi, circa 1600, il contagio è avvenuto a causa del contatto diretto con il paziente, invece in circa 900 casi le cause sono sconosciute (ciò conferma la difficoltà che si ha nel tracciare il virus e la sua diffusione tra pazienti) e più di 800 sono dovuti a persone provenienti da altri Paesi, in particolare la vicina Cina in cui il virus è nato, che hanno portato il virus in Corea del Sud.



CAPITOLO 2: ANALISI DESCRITTIVA

Questa mappa di calore della Corea del Sud mostra le zone del Paese che hanno registrato un maggior numero di contagi nel periodo considerato dal nostro dataset. Si osserva che la maggior parte dei contagi si sono avuti nella zona nord-occidentale, dove si trova Seoul, la capitale, e nella zona sud orientale in cui ci sono altre province molto abitate. Il numero di casi che si è avuto in ogni area della mappa è scritto all'interno di bolle con un colore che dipende dall'entità numero stesso, inoltre le zone a maggior contagi sono coperte da aree in colore rosso mentre quelle meno colpite sono coperte da aree di colore tendente al blu. Per realizzare questa mappa di calore è stata utilizzata la libreria folium e si è fatto il merge delle colonne “confirmed-date”, “sex”, “age”, “province” e “city” del dataset “p-info” con le colonne “latitude” e “longitude” del dataset “region” ottenendo il dataset “regional-patient”. In questo modo è stato possibile utilizzare longitudine e latitudine per individuare sulla mappa i vari contagi.



Questa ultima mappa di calore mostra sempre il numero di contagi che si hanno avuti in ogni zona della Corea del Sud, ma è presente un filtro in cui è possibile scegliere di visualizzare i casi in base al sesso. Anche per rappresentare questa mappa è stato utilizzato il dataset “regional-patient” precedentemente ottenuto.

A sinistra è visualizzata la mappa con riportati i contagi tra le donne, a destra quella per gli uomini.

CAPITOLO 2: ANALISI DESCRITTIVA

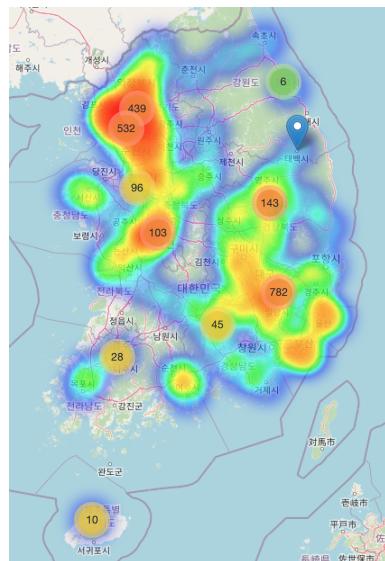


Figura 2.1: donne

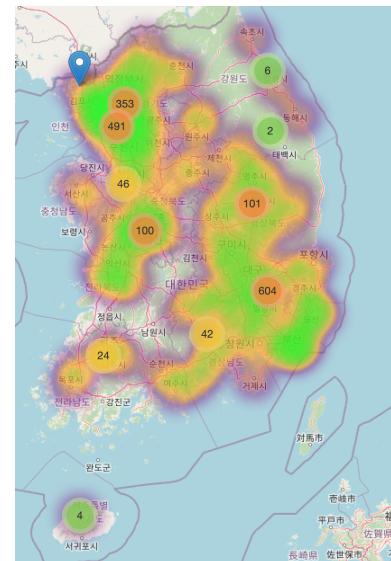
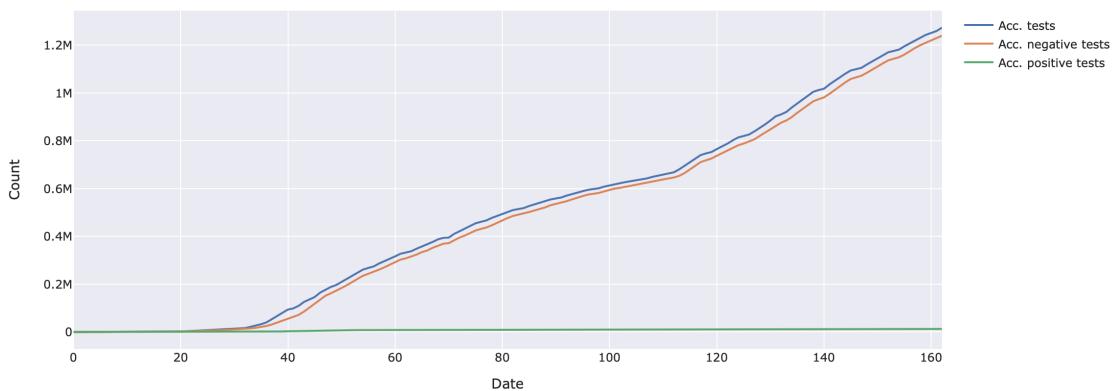


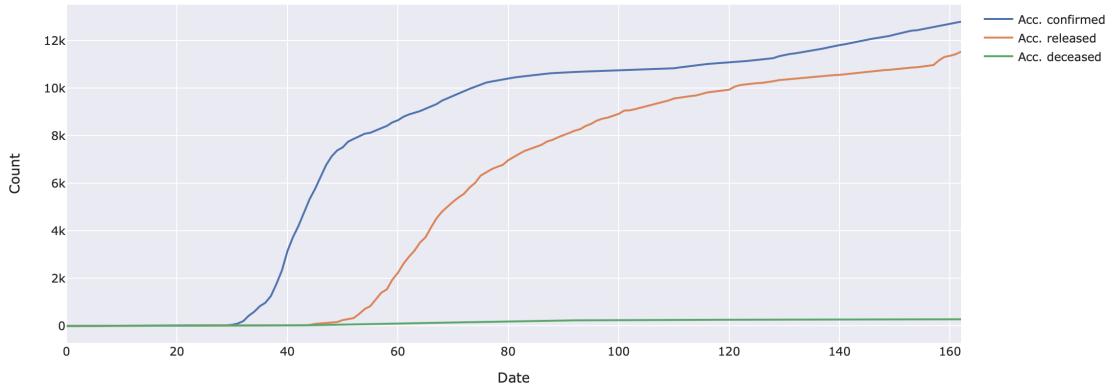
Figura 2.2: uomini

In questo grafico sono mostrati gli andamenti temporali dei valori cumulativi dei test, dei positivi e dei negativi. Ovviamente, essendo bassa la percentuale di test positivi rispetto a quelli negativi, l'andamento dei positivi a prima vista sembra orizzontale, tuttavia guardando attentamente si nota che vi è una crescita e ciò è confermato dal fatto che tra l'andamento dei test e dei negativi vi è uno scostamento corrispondente appunto ai casi positivi. Per realizzare questo grafico è stato preso il dataset con i dati temporali, cioè “time”, e in particolare le sue colonne “test”, “negative” e “confirmed”.



In modo analogo in questo grafico sono mostrati gli andamenti temporali dei valori cumulativi dei deceduti, dei rilasciati e degli isolati. Anche qui essendo bassa la percentuale di mortalità, l'andamento dei positivi ha una crescita lenta. Per realizzare questo grafico è stato preso il dataset con i dati dei pazienti, cioè “p-info”, e in particolare le sue colonne “deceased”, “released” e “isolated”.

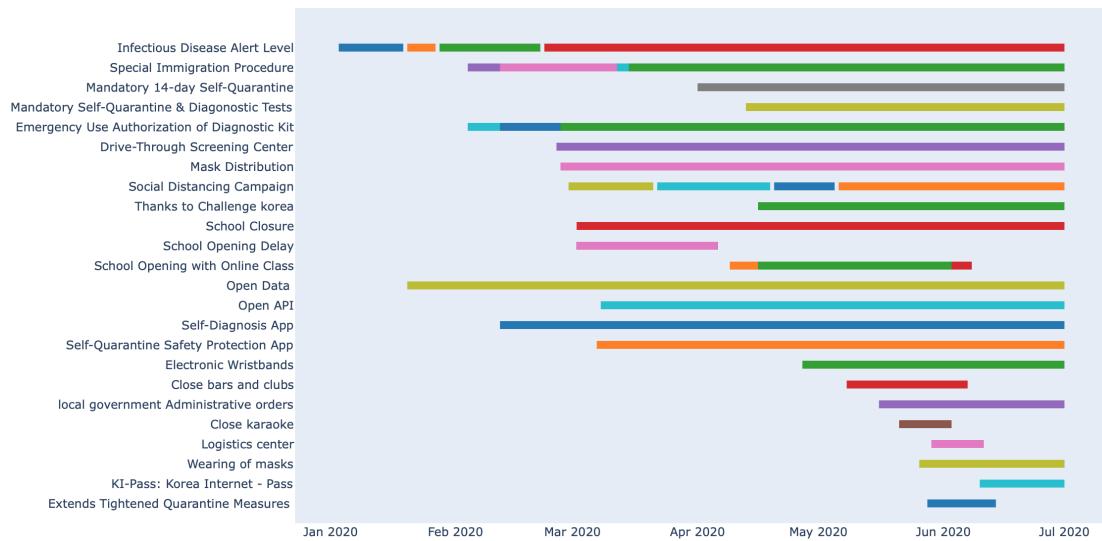
CAPITOLO 2: ANALISI DESCRITTIVA



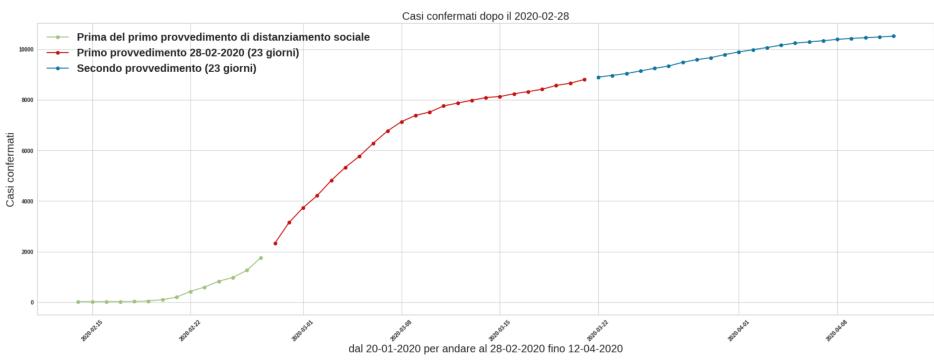
In questo diagramma di Gantt sono state riportate le varie politiche volute dal governo coreano per il contenimento dei contagi. Ogni politica è rappresentata come un singolo processo che ha un inizio e una fine in determinati giorni in base a quanto quel provvedimento è restato in vigore. Essendo il dataset limitato al periodo che va da gennaio a luglio, tutti i provvedimenti che sono rimasti in vigore dopo luglio, nel Gantt terminano a luglio, visto che in questa analisi i mesi successivi non sono di interesse. Inoltre visto che molti provvedimenti hanno lo stesso “record” si nota come questi si troveranno sulla stessa riga del diagramma di Gantt anche se con colori differenti per poterli distinguere. Il primo provvedimento è avvenuto a gennaio ed è stato un primo livello di allerta seguito da altri nei mesi successivi per spingere gli abitanti a restare a casa e a limitare i contatti sociali. Verso febbraio è stata introdotta una procedura speciale per coloro che provenivano da altri paesi, visto il diffondersi dell’epidemia nei paesi vicini dell’Asia. Inoltre tra febbraio e marzo sono state prodotte due app una per l’autodiagnosi e una per l’autoquarantena. Infine a inizio marzo vi è stato un primo provvedimento per il distanziamento sociale seguito dopo circa 20 giorni da un altro più restrittivo e successivamente da altri ancora. Oltre a questi vi sono poi altre varie restrizioni che il governo coreano ha voluto imporre per ridurre i contagi come l’obbligo di indossare mascherine, la chiusura di bar e club, la chiusura delle scuole... ecc. Per realizzare questo diagramma di Gantt è stato usato il dataset con i dati dei provvedimenti governativi cioè “policy”, e in particolare le sue colonne “gov-policy”, “detail”, “start-date” e “end-date”.

In questo grafico si vuole mostrare, dopo aver visto le varie politiche di contenimento del virus del governo coreano, il risultato che si è avuto in termini di contagi. Infatti, la Corea del Sud è stato uno dei primi Paesi al mondo ad essere stato colpito dal virus, ma è stato anche uno dei primi ad abbassare significativamente il numero dei contagi in poco tempo, grazie ai provvedimenti adottati dal governo. Si può vedere

CAPITOLO 2: ANALISI DESCRITTIVA

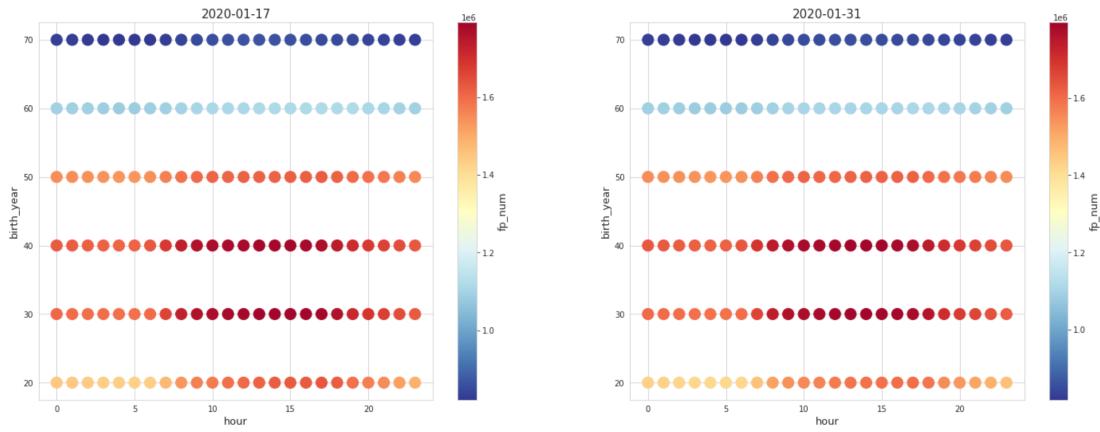


qui un grafico che mostra giornalmente i contagi confermati diviso in tre porzioni di colori diversi corrispondenti ai giorni prima dei provvedimenti di distanziamento sociale (in verde), quelli successivi al primo e meno restrittivo distanziamento iniziato il 29 febbraio (in rosso) e quelli successivi al secondo e più restrittivo provvedimento entrato in vigore il 22 marzo (in blu). Come si nota il primo provvedimento è stato voluto a causa dell'andamento dei contagi che stava tendendo ad un andamento esponenziale (quello in verde), il risultato è stato quello di rallentare significativamente i contagi portandolo ad un andamento non più esponenziale seppur ancora crescente (curva in rosso), quindi per appiattire definitivamente la curva si è deciso di attuare un ulteriore provvedimento più restrittivo ottenendo così una curva quasi piatta (quella blu). Per realizzare questo grafico è stato usato il dataset con i dati temporali dei contagi, cioè "time", e in particolare le sue colonne "date" e "confirmed".



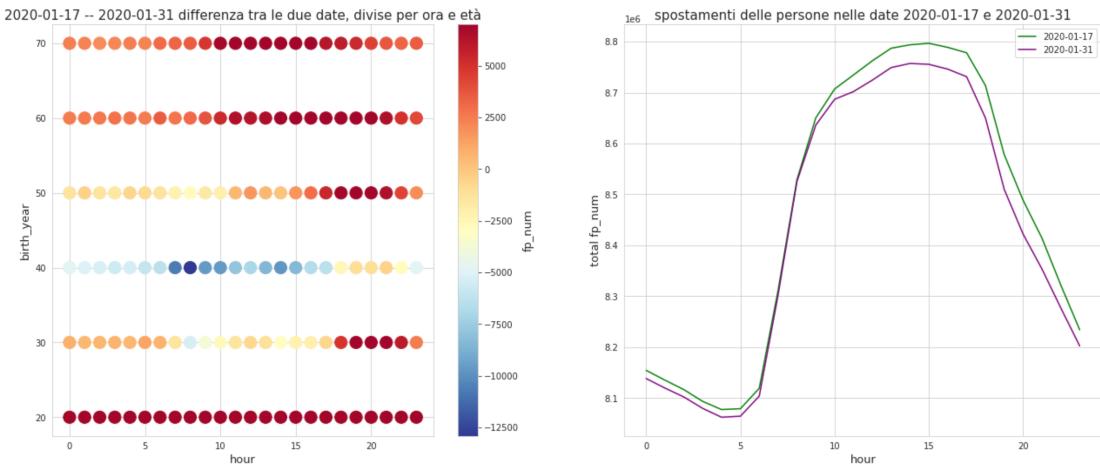
A questo punto si vuole fare un'analisi dello spostamento di persone nella provincia di Seoul (usando il dataset "floating") per osservare se i provvedimenti attuati dal governo e la paura del contagio hanno spinto le persone a uscire e circolare meno. In particolare

in questo grafico di distribuzione si mostra in base alle fasce d'età e all'orario della giornata l'entità degli spostamenti (secondo una scala di colori). Il confronto avviene tra il 17 gennaio (venerdì), prima del primo caso, e il 31 gennaio (venerdì).

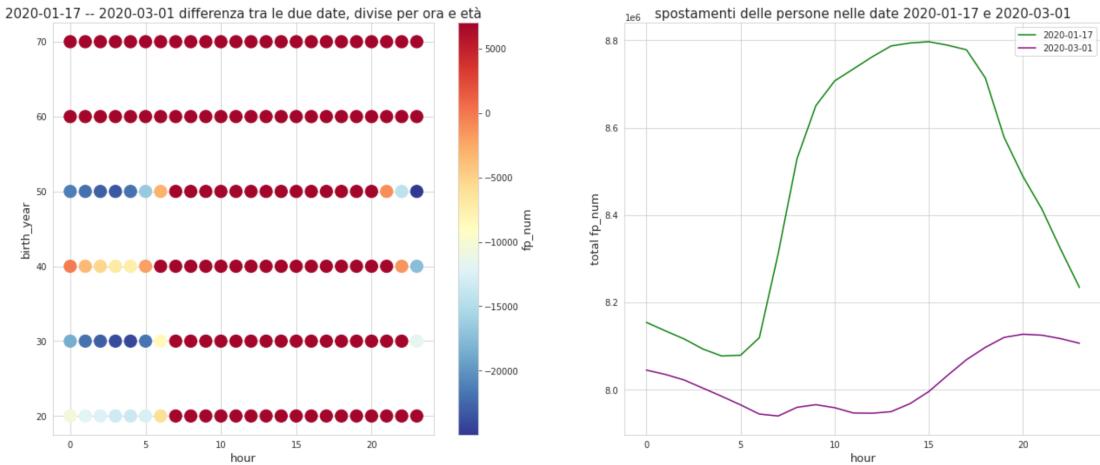


Osservando le precedenti distribuzioni non si percepisce una grande differenza, perciò in questo grafico di distribuzione si mostra in base alle fasce d'età e all'orario della giornata la differenza dell'entità degli spostamenti (secondo una scala di colori) tra 17 e 31 gennaio. Come si può osservare la maggiore differenza si è avuta nelle fasce più anziane (probabilmente impaurite dalla minaccia per la loro salute) e nelle più giovani (probabilmente a causa delle chiusure di scuole o luoghi di divertimento). Invece per le fasce d'età intermedie si nota che le differenze sono addirittura negative in alcuni casi, o comunque il virus non ha cambiato di molto i loro spostamenti, probabilmente perché molte di queste persone hanno continuato a spostarsi per motivi di lavoro principalmente. A destra sono mostrati in un grafico a linee proprio gli spostamenti ora per ora nelle due date considerate e si nota che c'è una differenza soprattutto durante il giorno negli spostamenti, probabilmente dovuti alla paura del contagio.

CAPITOLO 2: ANALISI DESCRITTIVA



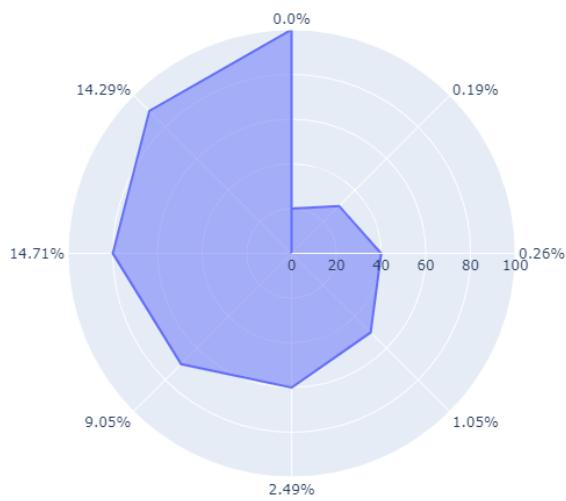
Un analogo confronto è qui fatto tra il 17 gennaio e il 1 marzo (data del primo provvedimento di distanziamento sociale). Si nota che in tal caso la differenza è molto più marcata ovviamente, soprattutto durante il giorno in cui c'è stata una circolazione minore anche rispetto alle ore notturne e serali.



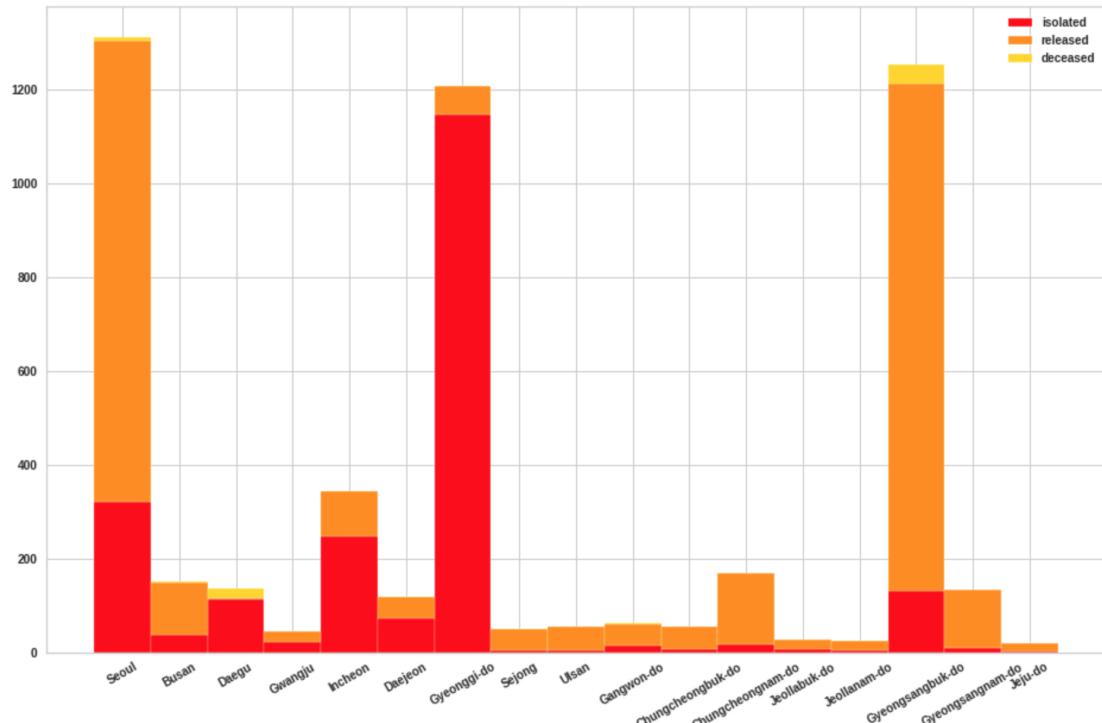
Questa radar chart mostra il tasso di mortalità che si è avuto per fascia d'età in Corea del Sud. Questa rappresentazione grafica consiste di una sequenza di raggi che hanno origine da un centro e formano angoli uguali tra loro e ci fornisce un indice di mortalità della popolazione Coreana tra i contagiati di Covid-19; ogni raggio rappresenta una delle variabili. La distanza dal centro del punto marcato sul raggio è proporzionale al valore della variabile rispetto al valore massimo raggiungibile. I punti sui raggi vengono congiunti con segmenti, così che il grafico ha la forma di una stella o di una ragnatela. Come si può osservare la mortalità è dello 0% fino ai 20 anni e inizia a salire in modo progressivo con l'aumentare dell'età, fino a divenire del 14.29% nella fascia di età di

CAPITOLO 2: ANALISI DESCRITTIVA

ottant'anni . Caso particolare è la fascia d'età dei centenni che comprende un solo campione, cioè una sola persona contagiata che però fortunatamente è guarita, quindi il tasso di mortalità per i centenni è 0%. Il tasso di mortalità è stato calcolato attraverso la percentuale di persone morte rispetto al totale dei contagiati di ogni specifica età. Il grafico è stato ottenuto aggiungendo al dataset “p-info” una colonna contenente per ogni riga il tasso di mortalità delle persone aventi stessa età del paziente a cui corrisponde la riga.



Infine questo istogramma mostra per ogni provincia il numero di pazienti rilasciati, isolati e deceduti nell'arco temporale coperto dal dataset considerato. Ogni barra è divisa in 3 barre sovrapposte di colori diversi (secondo la legenda) che indicano il numero di pazienti rilasciati, isolati e deceduti in quella provincia. Si nota che la provincia che ha avuto il maggior numero di morti è stata nettamente Gyeongsangbuk-do, mentre quella che ha avuto il maggior numero di pazienti isolati è stata sicuramente Gyeonggi-do. Infine si nota che mentre in Seoul e Gyeongsangbuk-do vi sono molti più pazienti rilasciati rispetto a quelli isolati e deceduti, in Gyeonggi-do si è avuto invece che la maggior parte dei pazienti è stata messa in isolamento. Per realizzare questo istogramma è stato preso il dataset con i dati dei pazienti, cioè “p-info”, e in particolare le sue colonne “released”, “deceased” e “isolated”.

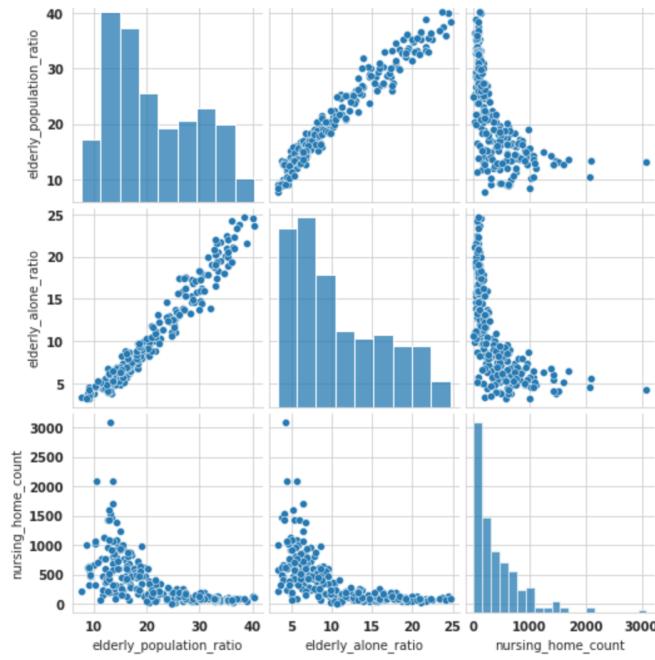


2.2 Analisi sulle relazioni tra le variabili

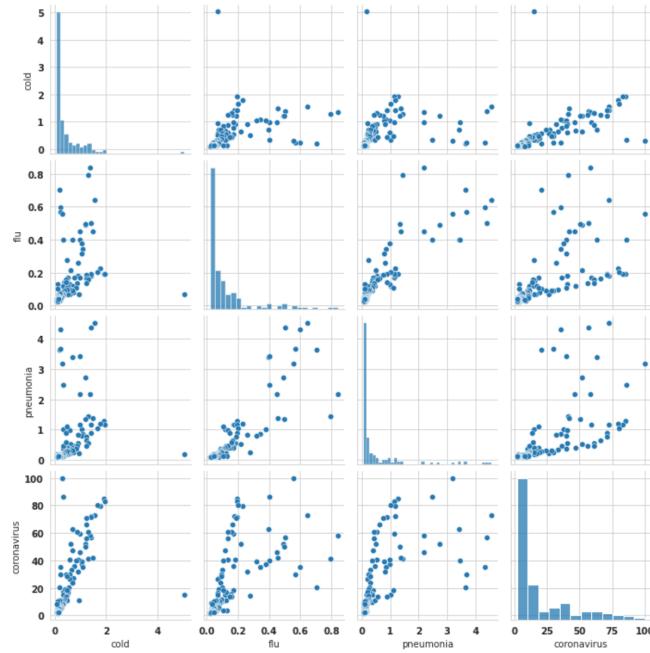
Con questo pairplot si vogliono osservare le relazioni reciproche tra la percentuale di anziani, di anziani soli e il numero di case di riposo di ogni città. Ciò è utile al fine di avere un'idea di come si distribuisce nel Paese la popolazione anziana, cioè la parte più a rischio e con mortalità più alta da Covid-19. Ogni punto corrisponde a una città. Si osserva che tra le percentuali di anziani e di anziani soli c'è una relazione che sembra essere lineare, cioè al crescere dell'una cresce secondo un coefficiente di proporzionalità anche l'altra. Invece nelle relazioni tra percentuali di anziani soli e di anziani e numero di case di riposo si nota che poche sono le città aventi basse percentuali di anziani e gran numero di case di riposo. Queste sono le grandi città, come Seoul che pur avendo età media bassa presentano numerose strutture per anziani. Invece le città con basso numero di strutture per anziani sono probabilmente città di dimensione minore. I dati usati per realizzare questo pairplot sono stati presi dal dataset “region”.

In questo secondo pairplot si sono volute visualizzare le relazioni tra il numero di ricerche su internet delle parole “coronavirus”, “flu”, “pneumonia” e “cold”. Si osserva come in tutte le relazioni tra le ricerche di “flu”, “pneumonia” e “cold” con quelle di “coronavirus” si possano individuare uno o più andamenti quasi lineari crescenti. Ciò significa che al

CAPITOLO 2: ANALISI DESCRITTIVA



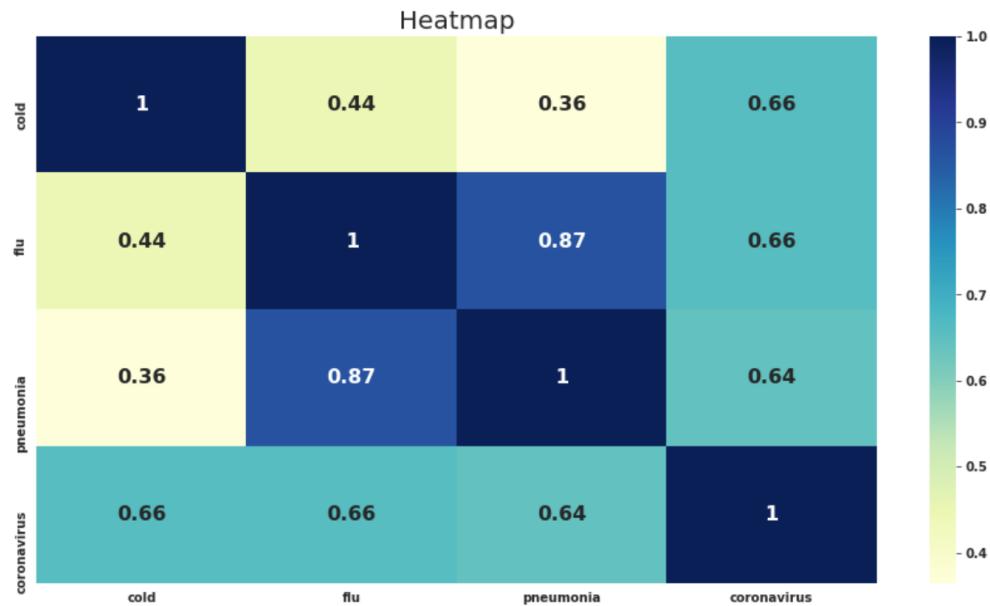
crescere di quelle ricerche anche le ricerche riguardo il coronavirus sono cresciute, ciò probabilmente per la somiglianza dei sintomi del Covid con quelle varie patologie. I dati usati per realizzare questo pairplot sono stati presi dal dataset sulle ricerche su internet “search”.



In questa matrice di correlazione sono riportate le varie correlazioni tra le ricerche di “flu”, “pneumonia”, “cold” e “coronavirus”, con scala di colori in base al valore della correlazione tra le variabili. Si osserva un’altissima correlazione tra le ricerche di flu

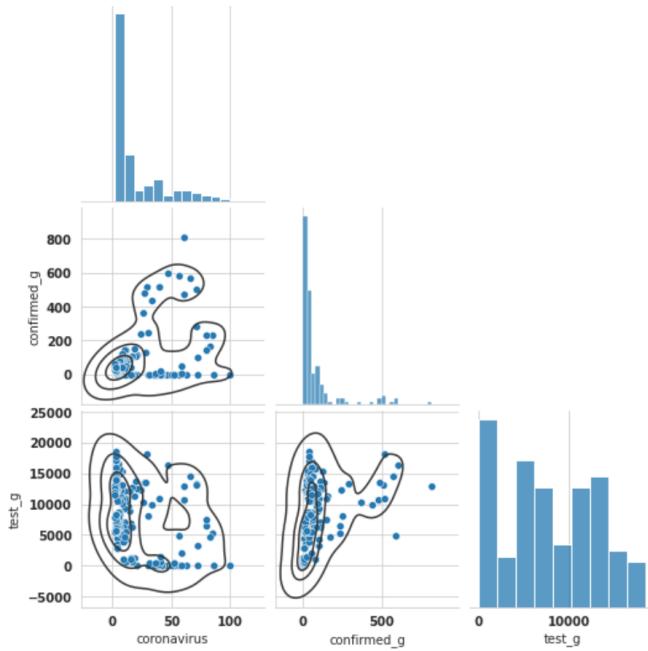
CAPITOLO 2: ANALISI DESCRITTIVA

e pneumonia, pari a 0,87 e correlazioni superiori a 0,6 tra ricerche sul coronavirus e ricerche sulle altre tre patologie. Ciò significa che spesso chi ha cercato informazioni riguardo il Covid-19 ha cercato anche informazioni su queste altre patologie correlate. I dati usati per realizzare questo pairplot sono stati presi dal dataset sulle ricerche su internet “search”.

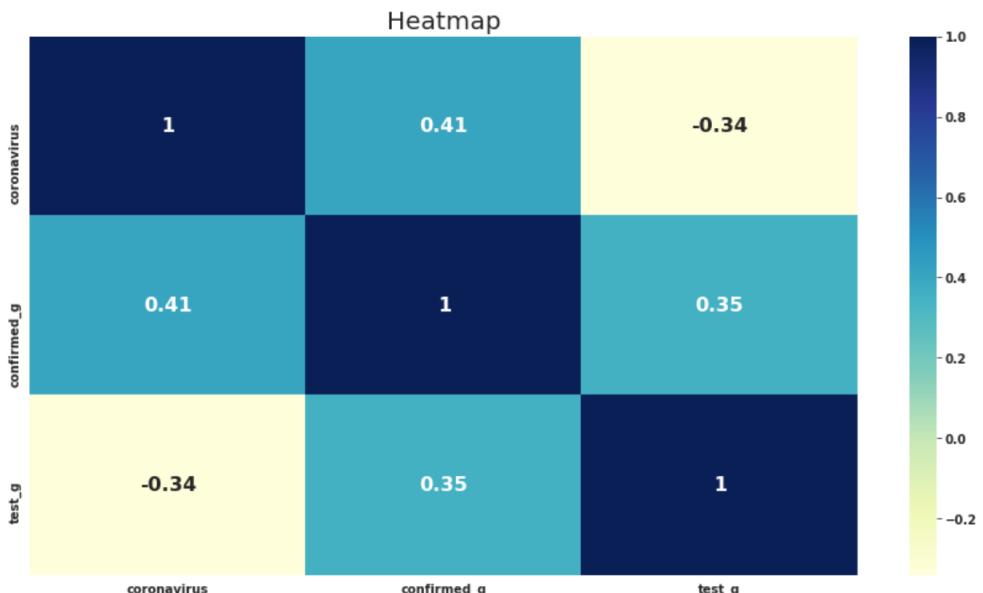


In questo pairplot si sono volute visualizzare le relazioni tra le ricerche della parola “coronavirus” e i positivi e i test giornalieri. Per visualizzare meglio le zone con maggior densità di punti si sono usate delle linee di livello. Si osserva che nella maggior parte dei giorni non c'erano ancora contagiati ma vi erano ugualmente ricerche sul coronavirus, ciò è dovuto al fatto che il dataset “search” contiene tutte le ricerche dal 2016. Poi vi sono stati dei giorni con tante ricerche e pochi contagi, corrispondenti probabilmente all'inizio della pandemia, in cui molti si informavano sul nuovo virus che aveva colpito per prima la vicina Cina. Infine, ci sono stati dei giorni con medie ricerche e tanti contagi, probabilmente corrispondenti al picco della pandemia in Corea del Sud. Per quanto riguarda i test ci sono stati tanti giorni con poche ricerche sul coronavirus ma con tanti test, probabilmente corrispondenti alla fine della pandemia. I dati usati per realizzare questo pairplot sono stati ottenuti dal dataset “trend-time” ottenuto facendo il merge delle colonne “cold”, “flu”, “pneumonia”, “coronavirus” e “date” del dataset “search” e le colonne “date”, “test”, “confirmed” e “deceased” del dataset “time” a cui sono state aggiunte le colonne contenenti i contagi, i test e i morti giornalieri.

In questa matrice di correlazione si mostrano le varie correlazioni tra le ricerche della



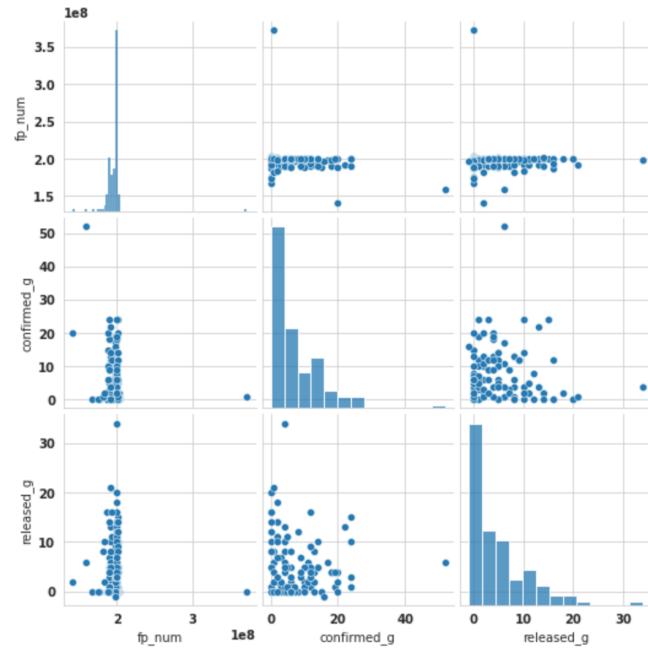
parola “coronavirus”, i positivi giornalieri e i test giornalieri. Si osserva che non ci sono correlazioni alte e le maggiori sono tra positivi giornalieri e ricerche della parola “coronavirus” (0,41) e tra test giornalieri e positivi giornalieri (0,35).



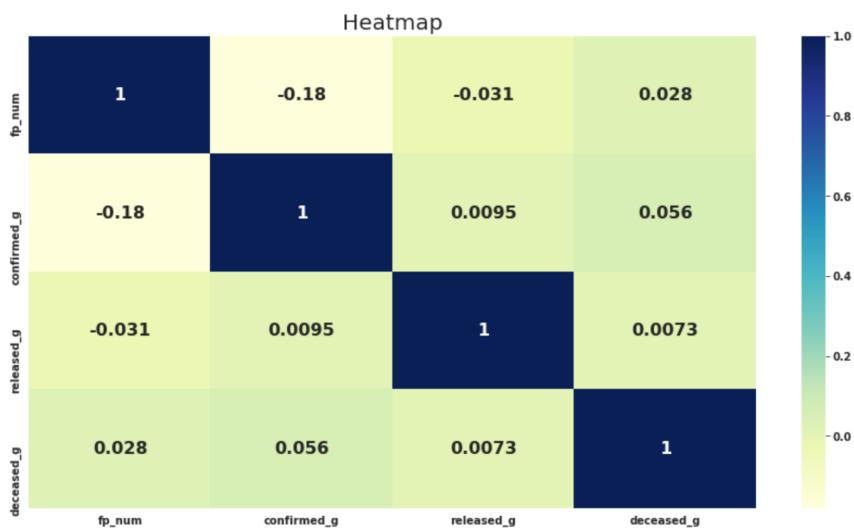
In questo pairplot si sono volute visualizzare le relazioni tra spostamenti delle persone, positivi e pazienti rilasciati. Si è utilizzato il dataset “seoul-info” ottenuto con operazioni di merge, drop e groupby dai dataset “floating” e “t-province”. Osservando in particolare la relazione tra positivi e pazienti dimessi giornalieri si nota che pochi sono stati i giorni con tanti dimessi e pochi positivi, probabilmente verso fine pandemia mentre si sono

CAPITOLO 2: ANALISI DESCRITTIVA

avuti pochi giorni con pochi dimessi e tanti positivi, corrispondenti a inizio pandemia probabilmente.



Visualizzando tramite una matrice di correlazione le varie correlazioni tra spostamenti delle persone, positivi, pazienti rilasciati giornalieri e morti giornalieri, si nota che ci sono correlazioni bassissime in tal caso.

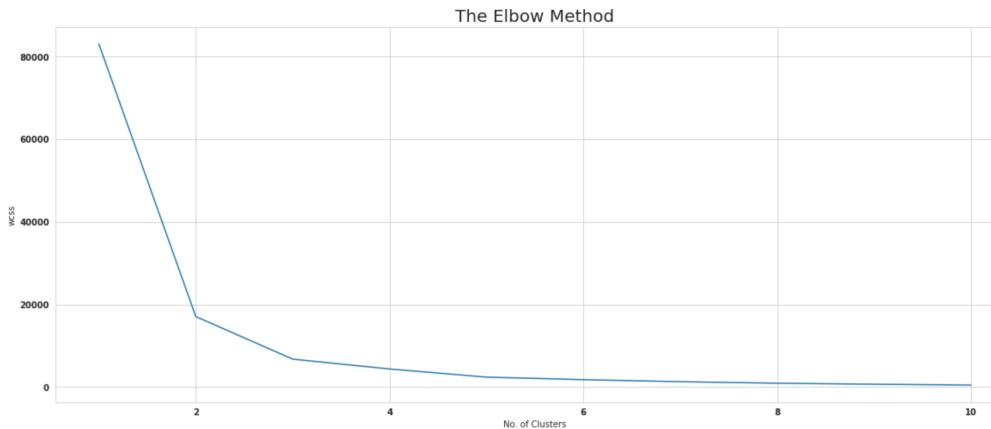


3. Sklearn

3.1 Introduzione al tool

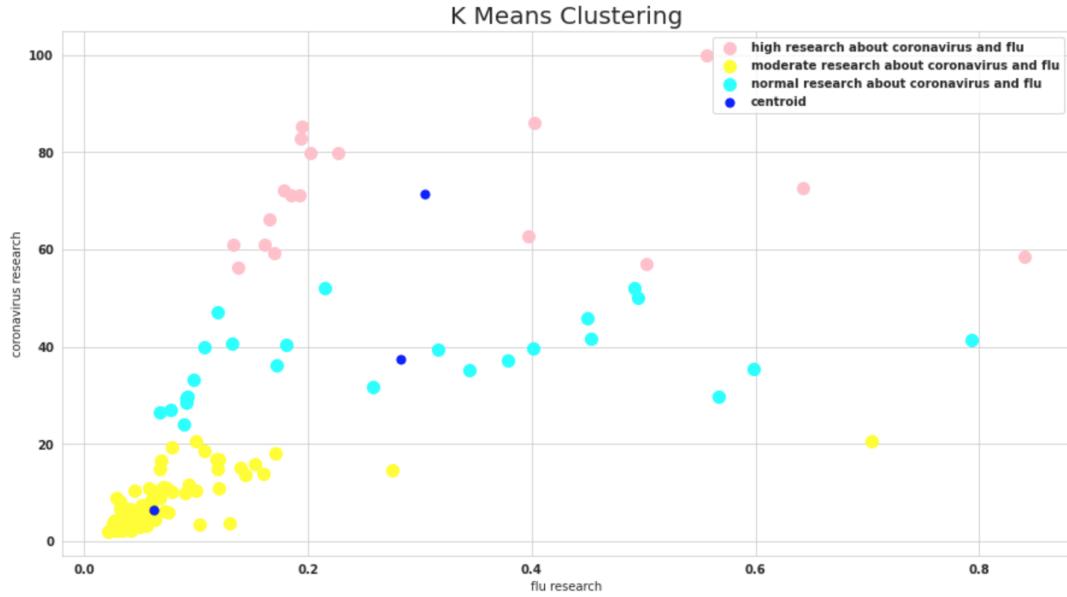
3.2 Divisioni in cluster

Dopo aver fatto un'analisi descrittiva utilizzando i dataset a disposizione si vuole a questo punto trovare cluster interessanti in cui dividere i dati. Come primo metodo di clusterizzazione si considera il k-means, in cui k rappresenta il numero di cluster da individuare. Per trovare tale k si è utilizzato l'Elbow method, ovvero un metodo grafico che consente di scegliere il valore di k ottimale. Le variabili considerate sono le ricerche sul coronavirus e sull'influenza. Ottenuto il grafico dell'inertia sottostante si considera il k per cui si ha l'ultima variazione di pendenza rilevante. In tal caso si è scelto k=3.

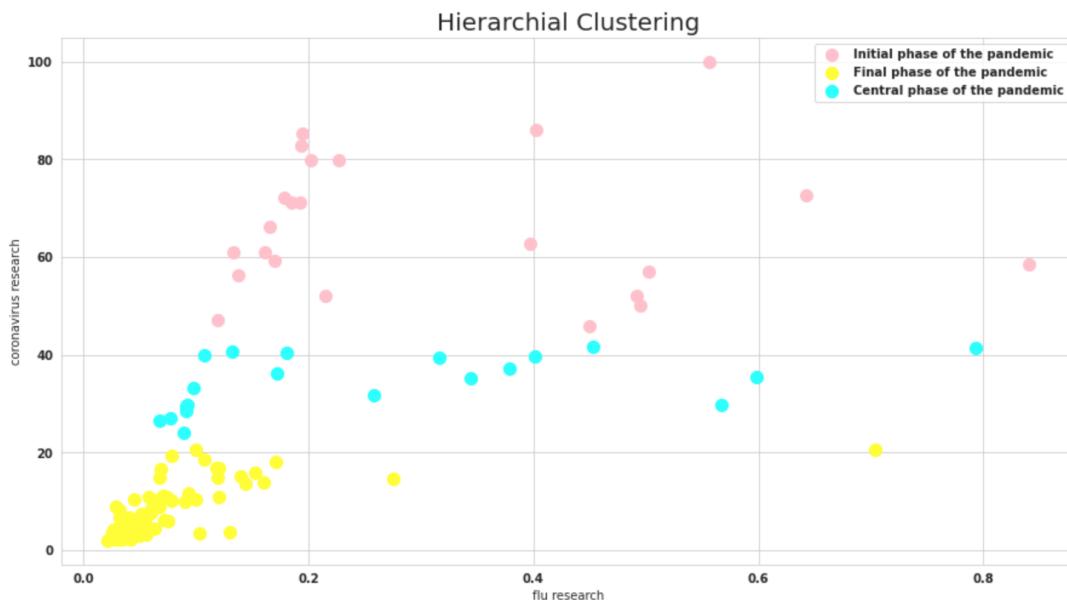


Usando il valore di k sopra calcolato (3) si possono quindi trovare i k cluster. A ogni cluster è stato associato un colore differente. Il cluster giallo corrisponde ai giorni con poche ricerche riguardo il coronavirus e l'influenza, quello verde acqua contiene i giorni con livelli di ricerche intermedi e infine quello rose indica livelli di ricerca elevati. Probabilmente i giorni con maggiori ricerche sono quelli dell'inizio della pandemia in cui

ancora non si sapeva quasi nulla sul virus e in molti si volevano informare al riguardo, invece quelli con poche ricerche probabilmente corrispondono alla fine del periodo di massima diffusione del virus in Corea, in cui ormai gran parte della popolazione si era già informata in precedenza. In blu sono rappresentati i centri dei vari cluster ottenuti.

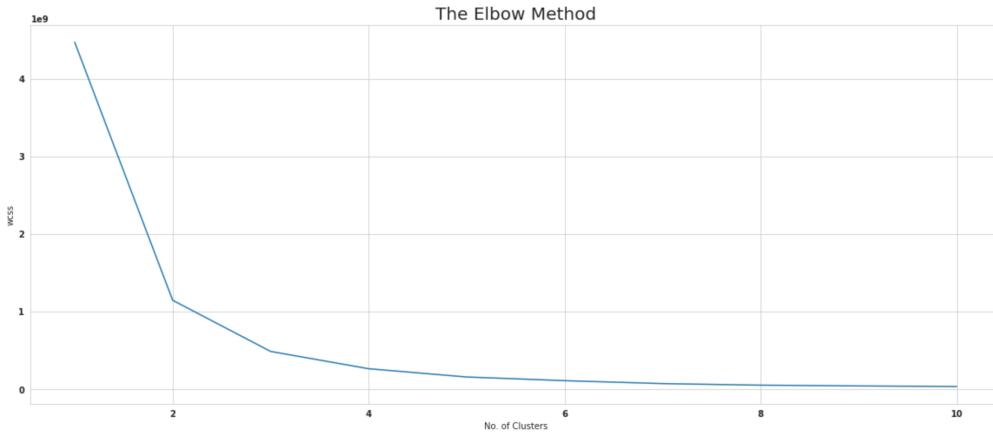


Dopo l'utilizzo del primo metodo per fare clustering si è proseguito con un ulteriore metodo, cioè quello del clustering gerarchico. Utilizzando sempre le stesse variabili e lo stesso valore di k i risultati ottenuti applicando questa seconda tecnica sono equivalenti a quelli ottenuti con il metodo precedente.

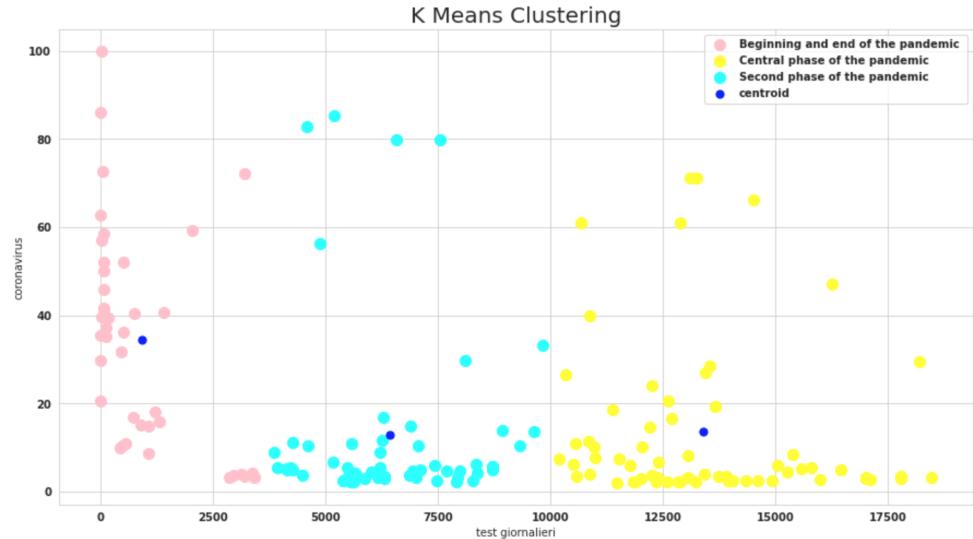


Una seconda divisione in cluster si può poi effettuare tra ricerche riguardo il coronavirus

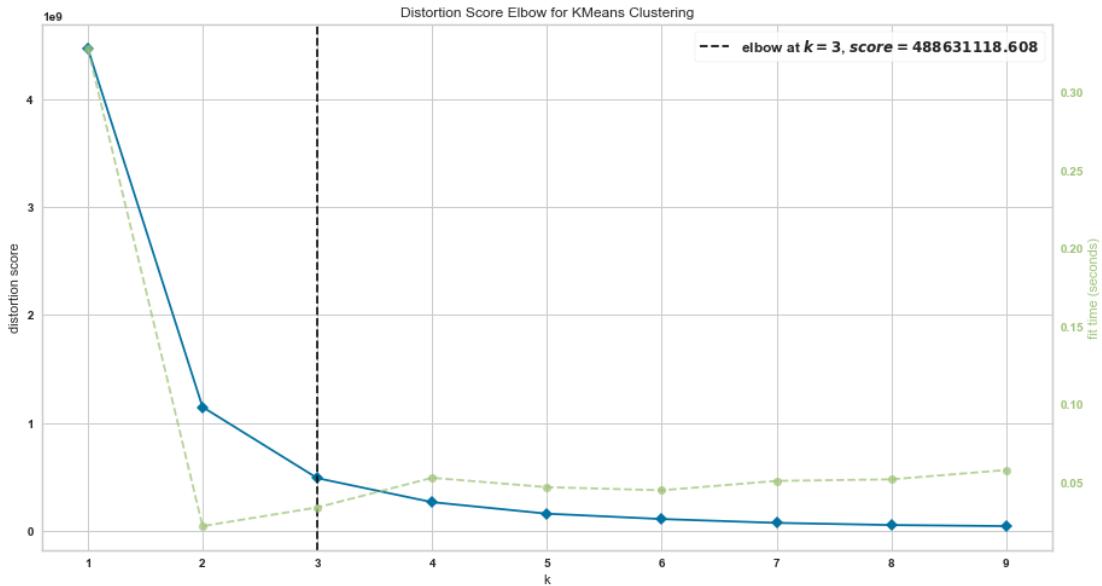
e test giornalieri. Per prima cosa, come fatto nel caso precedente si considera il metodo k-means. Per trovare k si utilizza come prima l'Elbow method. Ottenuto il grafico dell'inertia sottostante si considera il k per cui si ha l'ultima variazione di pendenza rilevante. Si è scelto k=3 anche in questo caso.



Usando il valore di k sopra calcolato (3) si possono quindi trovare i k cluster. A ogni cluster è stato associato un colore differente. Il cluster rosa a sinistra comprende i giorni con pochi test giornalieri, quindi probabilmente sono giorni all'inizio della pandemia (in cui si facevano pochi test e c'erano presumibilmente tante ricerche sul virus) e alla fine della pandemia (in cui c'erano presumibilmente poche ricerche sul virus e pochi test). Il cluster verde acqua invece comprende i giorni con un numero medio di test, probabilmente nella seconda fase della pandemia, e infine il cluster giallo, con tanti test giornalieri corrisponde probabilmente alla fase centrale della pandemia, vicina al picco, in cui si facevano tantissimi test e in cui si facevano mediamente meno ricerche sul virus. In blu sono rappresentati i centri dei vari cluster ottenuti.

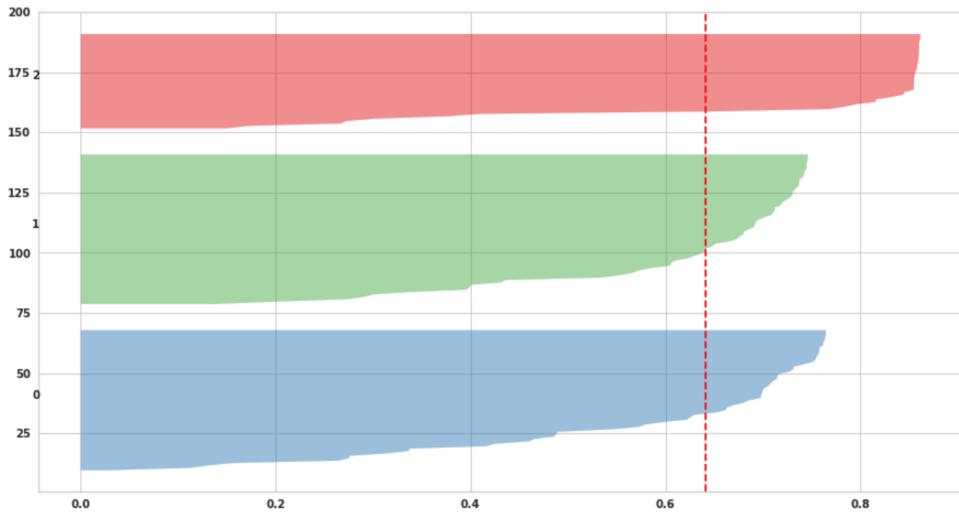


Per confermare i risultati ottenuti con le tecniche precedenti si è utilizzata la libreria yellowbrick. Riapplicando tramite questa libreria la tecnica kmeans si ottiene nuovamente che il valore di k ottimale è 3.

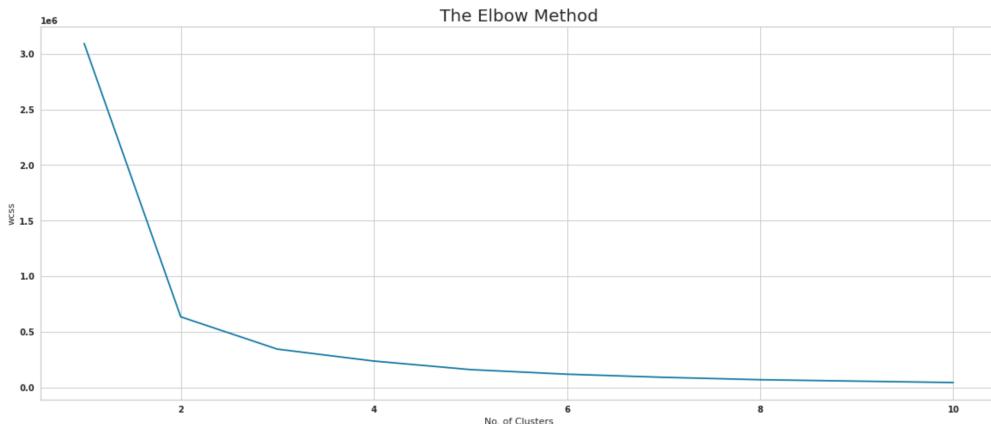


Per visualizzare la bontà dei cluster così ottenuti si è utilizzata la visualizzazione delle silhouette. Si osserva che tutte e tre le silhouette, rappresentanti i tre cluster ottenuti, superano la retta tratteggiata verticale e non vi sono campioni con silhouette negativa, ciò è indice di una buona divisione in cluster.

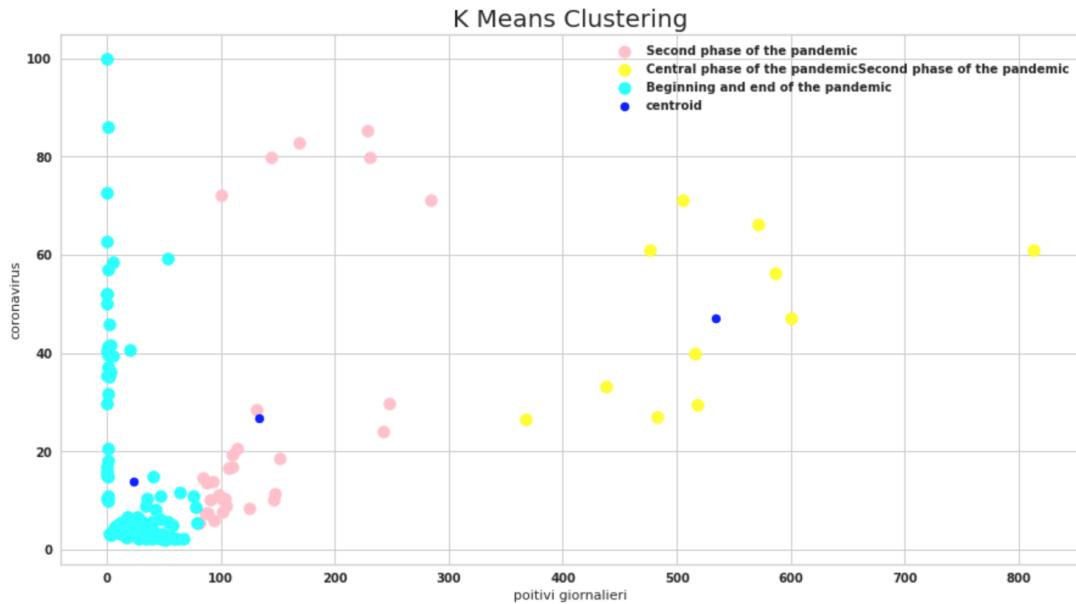
Una terza divisione in cluster si può poi effettuare tra ricerche riguardo il coronavirus e positivi confermati giornalieri. Per prima cosa, come fatto nei casi precedenti si considera il metodo k-means. Per trovare k si utilizza come prima l'Elbow method. Ottenuto



il grafico dell'inertia sottostante, si considera il k per cui si ha l'ultima variazione di pendenza rilevante. Si è scelto ancora k=3.

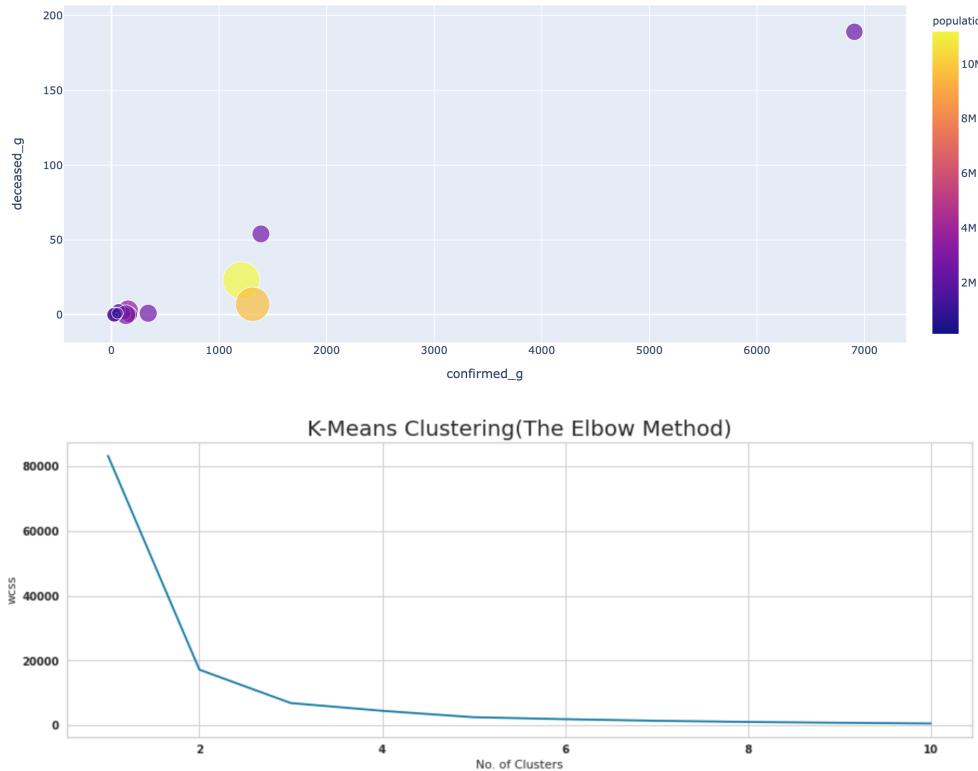


Usando il valore di k sopra calcolato (3) si possono quindi trovare i k cluster. A ogni cluster è stato associato un colore differente. Il cluster a sinistra di colore verde acqua è costituito sia da giorni con alte ricerche sul coronavirus e con pochi positivi giornalieri, quindi probabilmente sono giorni all'inizio della pandemia, che da giorni con basse ricerche sul coronavirus e con pochi positivi giornalieri, quindi probabilmente sono giorni verso la fine della pandemia. Il cluster centrale in rosa invece probabilmente rappresenta i giorni della seconda fase della pandemia in cui si avevano più positivi giornalieri rispetto alle fasi iniziale e finale ma meno ricerche sul Covid-19. Infine, il cluster a destra di colore giallo probabilmente rappresenta i giorni vicini alla fase centrale della pandemia e al picco di essa, con i massimi valori di positivi giornalieri e numero di ricerche minori. In blu sono rappresentati i centri dei vari cluster ottenuti.



In questo scatterplot sono rappresentate le varie province della Corea del Sud in base alla somma di persone decedute e positive. Ogni bolla rappresenta una città e la dimensione e il colore sono scelti in base al numero di abitanti della provincia. Si può osservare che le province si dividono in tre gruppi, uno costituito dalle province meno colpite dal virus in termine di morti e contagi (il gruppo più numeroso), un altro costituito dalle province un po' più colpite tra cui vi sono Seoul, Gyeonggi-do e Gyeongsangbuk-do e infine la provincia che ha registrato maggiori contagi e morti è stata Daegu, pur avendo numero di abitanti molto minore rispetto a province come Gyeongsangbuk-do e Seoul. Scorrendo con il puntatore sopra i vari punti, si visualizza il nome della provincia e la sua popolazione e si può osservare che Daegu è stata la provincia più colpita, anche se dal dataset “p-info” non risultava ciò, probabilmente significa che non sono stati raccolti tutti i dati sui pazienti in questa provincia. Questo scatterplot è stato ottenuto utilizzando il dataset “t-province-ragg”, ricavato a seguito di un’operazione di groupby sulle colonne del dataset “t-province” e aggiungendo i dati sul numero di abitanti delle varie province.

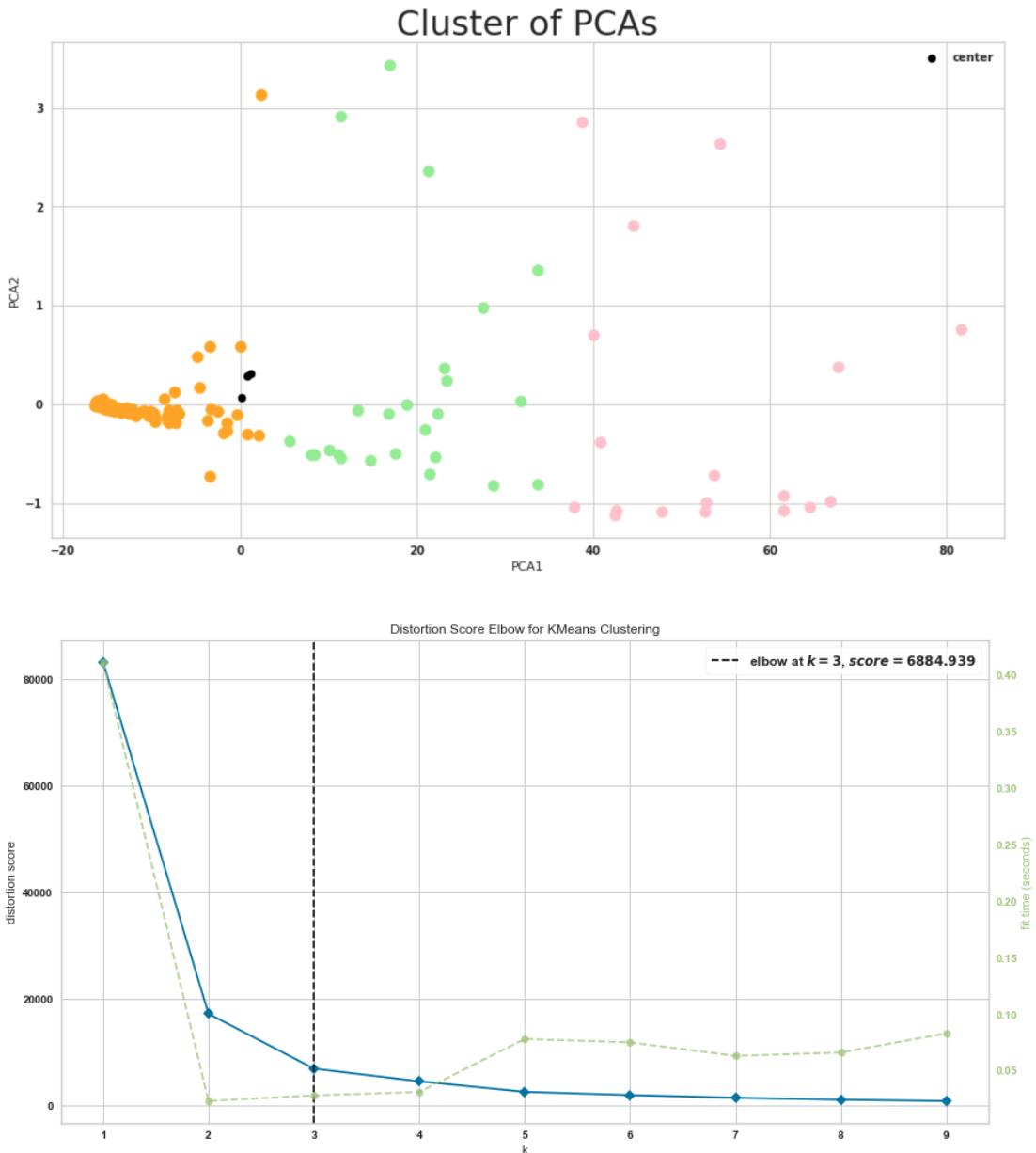
Un’ulteriore divisione in cluster si può poi effettuare tra ricerche riguardo il coronavirus, l’influenza, il raffreddore e la polmonite. Per prima cosa, come fatto nei casi precedenti si considera il metodo k-means. Per trovare k si utilizza come prima l’Elbow method. Ottenuto il grafico dell’inerzia sottostante, si considera il k per cui si ha l’ultima variazione di pendenza rilevante. Si è scelto ancora k=3.



In questo caso essendo più di due le variabili in gioco, i cluster non potranno essere visualizzati in uno spazio bidimensionale, quindi è stata utilizzata la PCA per proiettare l'informazione da uno spazio a quattro dimensioni a uno bidimensionale. I tre cluster che si ottengono sono questi. I loro centri risultano quasi sovrapposti ma ciò è dovuto al fatto che stiamo visualizzando in due dimensioni ciò che andrebbe visualizzato in quattro dimensioni, quindi anche la comprensione dei cluster è più complessa rispetto ai casi precedenti, ma molto probabilmente essi sono stati creati identificando giorni con basse, medie e alte ricerche online delle varie patologie.

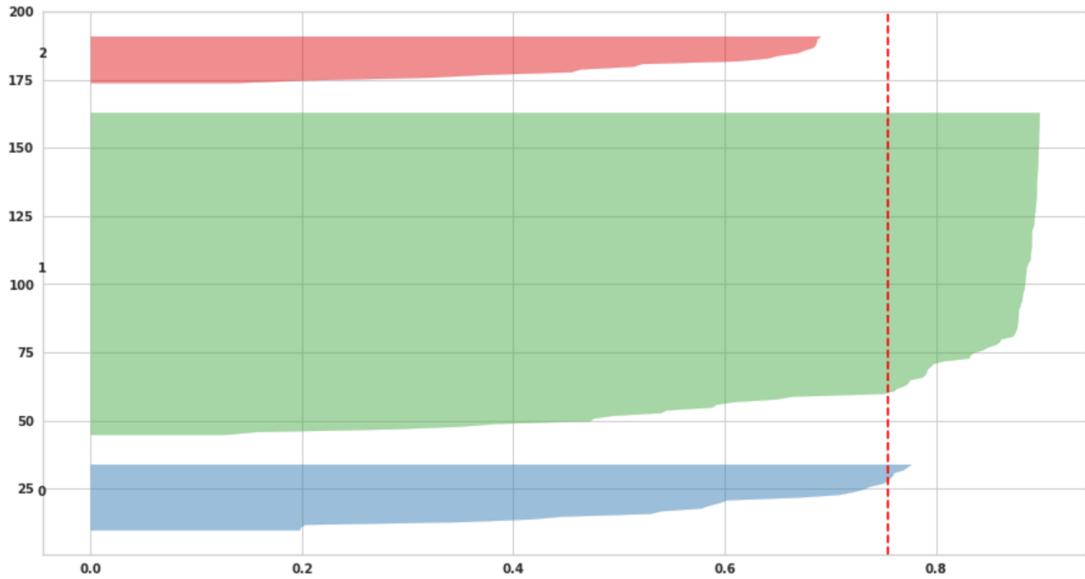
Per confermare i risultati ottenuti è stata qui utilizzata la libreria yellowbrick. Riapplicando tramite questa libreria la tecnica kmeans si ottiene nuovamente che il valore di k ottimale è 3.

Infine, per visualizzare la bontà dei cluster così ottenuti si è utilizzata la visualizzazione delle silhouette. Si osserva che due delle tre silhouette superano la retta tratteggiata verticale e non presentano campioni con silhouette negativa, mentre la silhouette del primo cluster non supera la retta, ciò indica che il primo cluster non è ottimale.

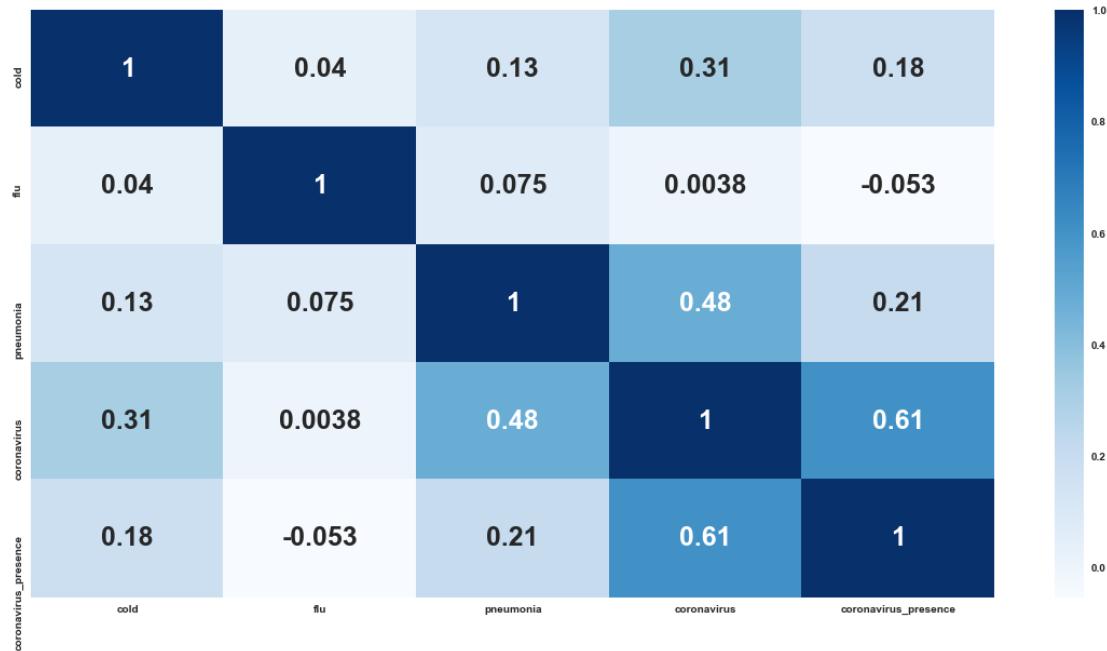


3.3 Classificazione

Dopo un'analisi dei cluster in cui si possono dividere i dati a disposizione, si considerano ora delle classificazioni tra i giorni in base alla presenza o meno del Covid-19. Al dataset “search” è stata aggiunta una colonna che indica la presenza del coronavirus, visto che questo dataset contiene le ricerche della parola “coronavirus”, che tuttavia era cercata anche prima della comparsa del virus. A questo punto avendo il dataset molte più osservazioni con valore di “coronavirus-presence” uguale a false rispetto al valore true, si è deciso di bilanciarlo. Fatto questo si è visualizzata la matrice di correlazione tra le

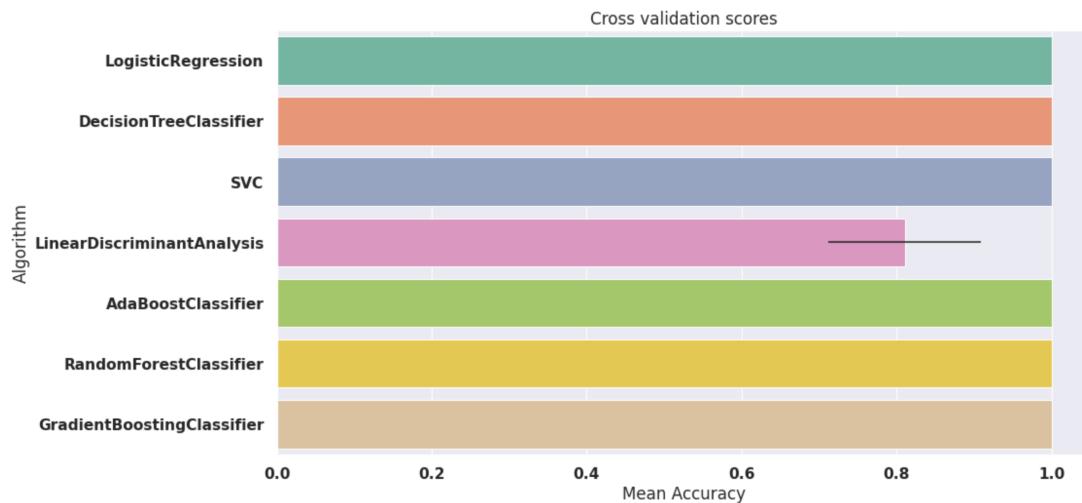


ricerche sul coronavirus, sulla polmonite, sul raffreddore e sull'influenza e la presenza del virus. Si rileva che vi è una correlazione pari a 0,61 tra le ricerche sul coronavirus e la presenza del virus, di 0,21 tra le ricerche sulla polmonite e la presenza del virus e di 0,18 tra le ricerche sul raffreddore e la presenza del virus.

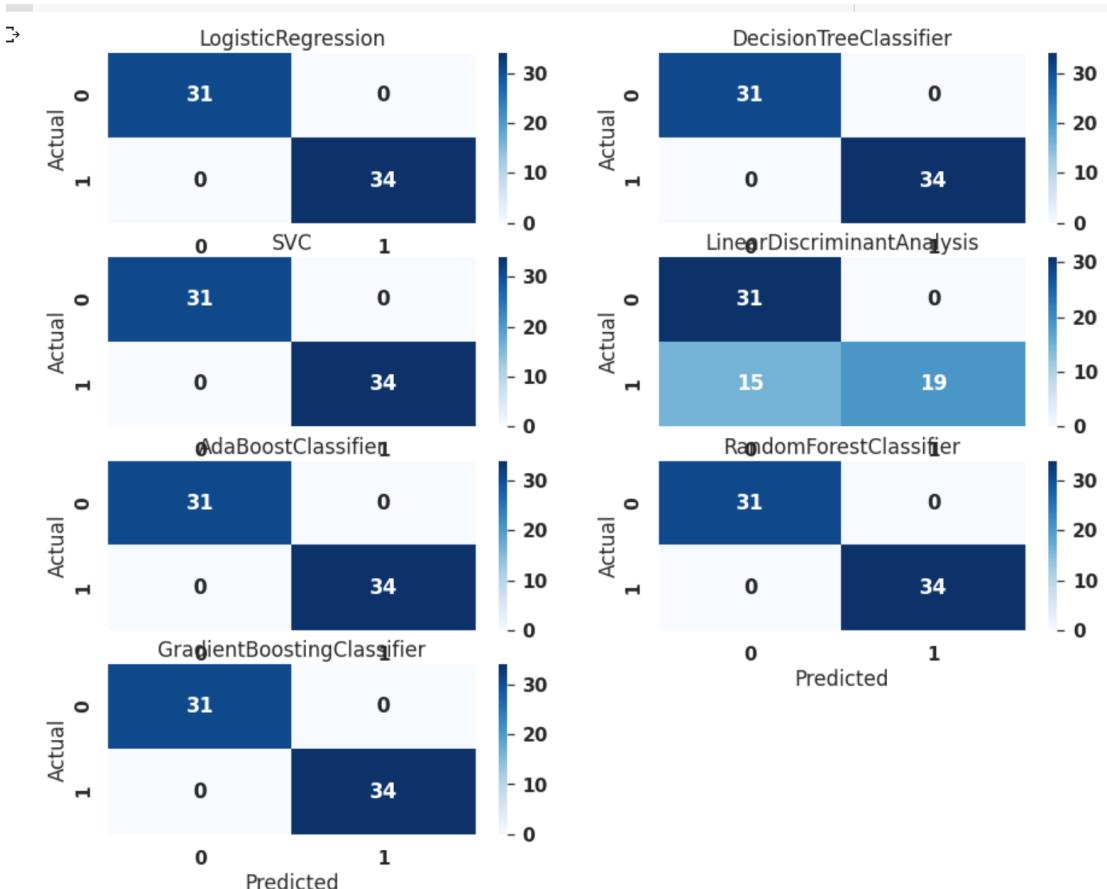


L'obiettivo che ci si pone è fare classificazione dei giorni in base alla presenza del virus o no utilizzando le informazioni sulle ricerche su internet delle varie patologie. In particolare, si vuole utilizzare diversi classificatori, confrontandone i risultati. L'accuratezza delle singole tecniche si classificazione è visualizzata tramite questo istogramma. Si può

osservare che l'accuratezza è molto alta, pari ad 1, tranne in un caso in cui vale 0,77.



Le seguenti matrici di confusione mostrano come tutti tranne uno dei classificatori classificano correttamente in tutti i casi, mentre solo il classificatore “linear discriminant analysis” non classifica correttamente 15 giorni in cui vi è stato il virus, classificandoli invece come giorni senza virus.

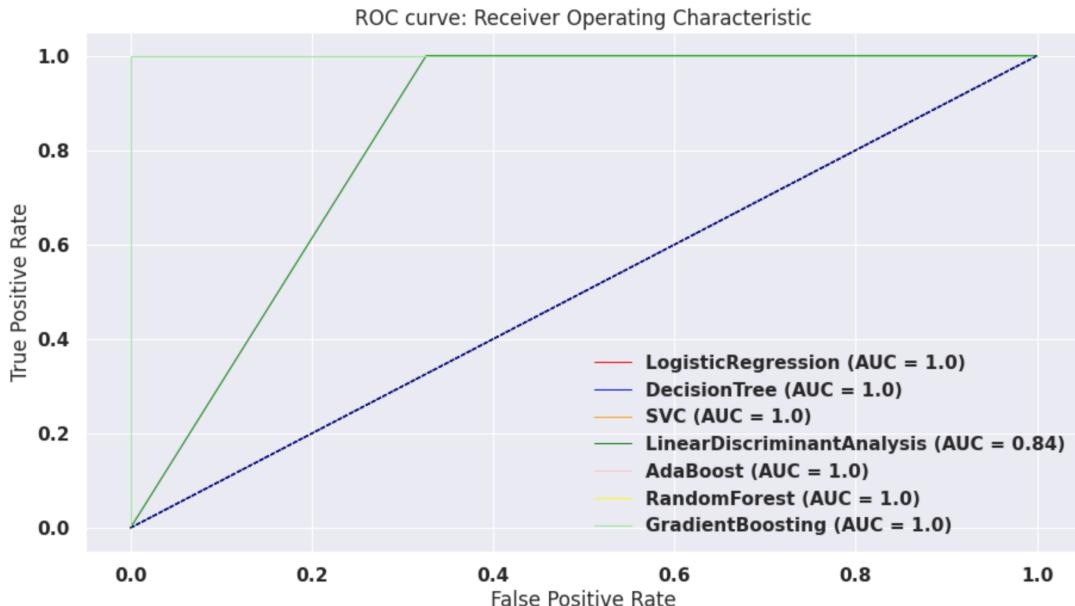


CAPITOLO 3: SKLEARN

È riportato diseguito un report sul funzionamento dei vari classificatori utilizzati.

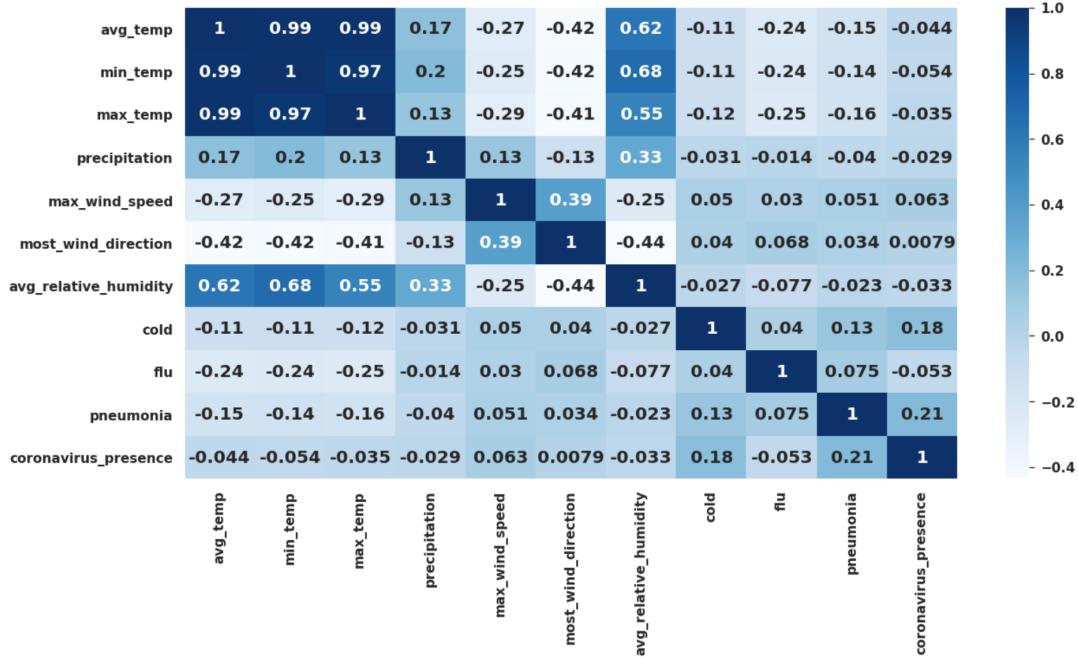
LogisticRegression Classification Report:						
	precision	recall	f1-score	support		
accuracy	0.0	1.00	1.00	1.00	31	
macro avg	1.0	1.00	1.00	1.00	34	
weighted avg	1.00	1.00	1.00	1.00	65	
DecisionTreeClassifier Classification Report:						
	precision	recall	f1-score	support		
accuracy	0.0	1.00	1.00	1.00	31	
macro avg	1.0	1.00	1.00	1.00	34	
weighted avg	1.00	1.00	1.00	1.00	65	
SVC Classification Report:						
	precision	recall	f1-score	support		
accuracy	0.0	1.00	1.00	1.00	31	
macro avg	1.0	1.00	1.00	1.00	34	
weighted avg	1.00	1.00	1.00	1.00	65	
LinearDiscriminantAnalysis Classification Report:						
	precision	recall	f1-score	support		
accuracy	0.0	0.67	1.00	0.81	31	
macro avg	1.0	0.66	0.56	0.72	34	
weighted avg	0.84	0.78	0.76	0.77	65	
AdaBoostClassifier Classification Report:						
	precision	recall	f1-score	support		
accuracy	0.0	1.00	1.00	1.00	31	
macro avg	1.0	1.00	1.00	1.00	34	
weighted avg	1.00	1.00	1.00	1.00	65	
RandomForestClassifier Classification Report:						
	precision	recall	f1-score	support		
accuracy	0.0	1.00	1.00	1.00	31	
macro avg	1.0	1.00	1.00	1.00	34	
weighted avg	1.00	1.00	1.00	1.00	65	
GradientBoostingClassifier Classification Report:						
	precision	recall	f1-score	support		
accuracy	0.0	1.00	1.00	1.00	31	
macro avg	1.0	1.00	1.00	1.00	34	
weighted avg	1.00	1.00	1.00	1.00	65	

Le seguenti ROC curve anche indicano il funzionamento dei vari classificatori. Come si vede tutti tranne uno dei classificatori funzionano benissimo, dato che le curve vanno a 1 quasi istantaneamente.

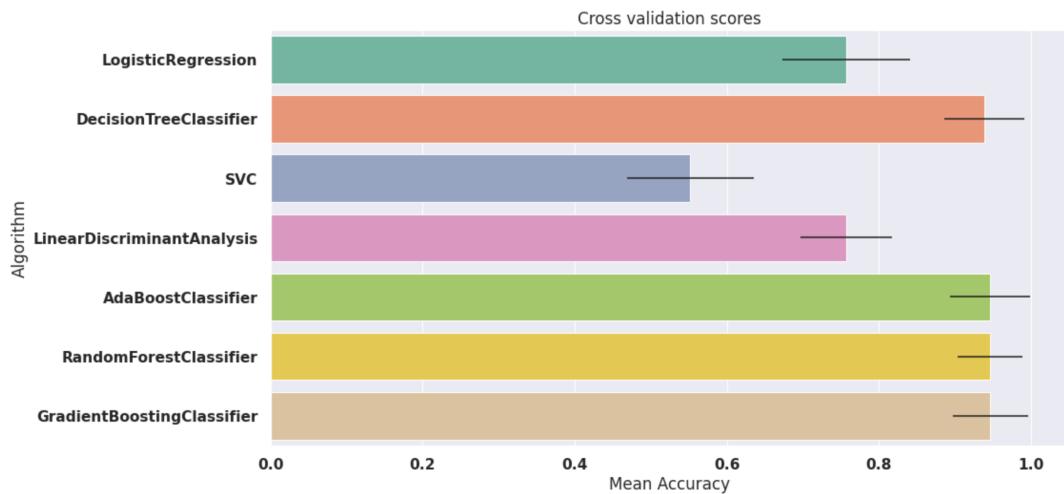


Una seconda classificazione che si vuole fare ora, sempre in base alla presenza del virus o no, è utilizzare le informazioni sulle ricerche su internet delle varie patologie tranne quelle sul coronavirus, sostituendo questa informazione con informazioni sulle condizioni meteorologiche nei giorni, che potrebbero aver influenzato gli spostamenti delle persone nei vari giorni e aver avuto un ruolo importante sulla capacità di diffusione dei vari

virus. Come si osserva dalla seguente matrice di correlazione queste informazioni meteorologiche hanno correlazione bassissima con la presenza del coronavirus, come previsto, tuttavia si sono utilizzate ugualmente queste informazioni per fare classificazione per vedere come avrebbero lavorato i vari classificatori.

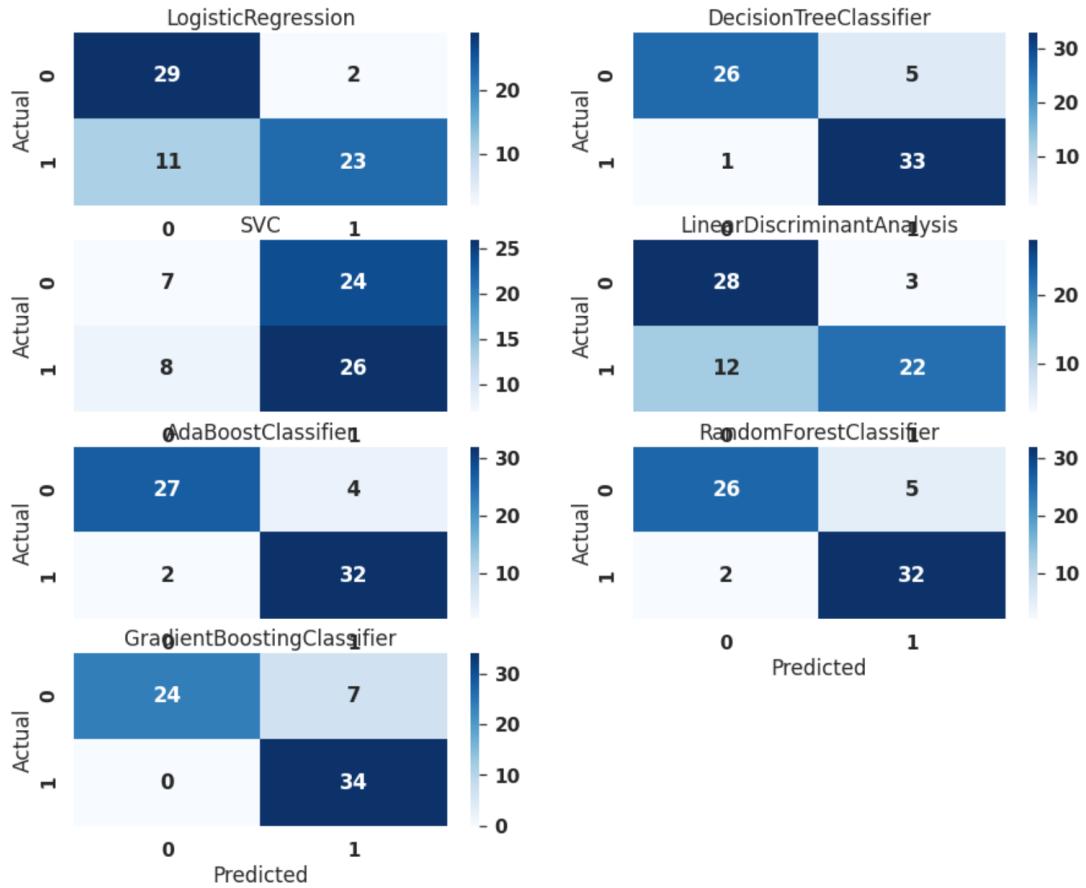


Utilizzando sempre gli stessi classificatori si vuole fare un confronto del loro funzionamento come nel caso precedente. L'accuratezza delle singole tecniche di classificazione è visualizzata tramite questo istogramma. Si osserva che quattro di esse hanno accuratezza molto buona, superiore a 0,90, mentre quella con accuratezza peggiore è l'SVC con 0,51.



Le seguenti matrici di confusione mostrano che i quattro classificatori con accura-

tezza maggiore classificano correttamente i giorni in quasi tutti i casi, mentre, come previsto, quelli con accuratezza minore, come l'SVC, commettono molti più errori di classificazione.

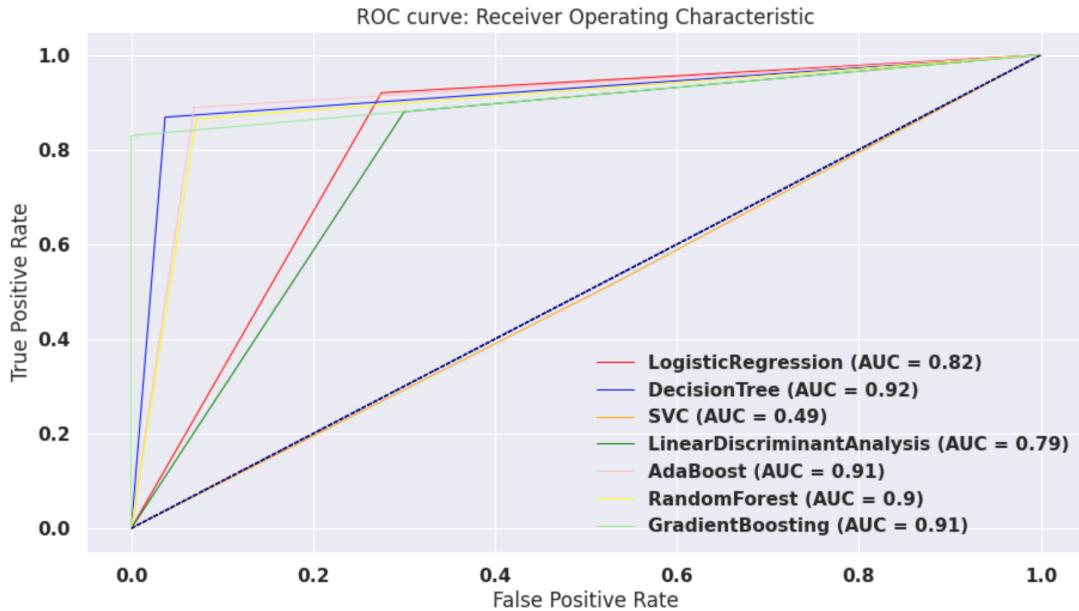


È riportato di seguito un report sul funzionamento dei vari classificatori utilizzati.

LogisticRegression Classification Report:				
	precision	recall	f1-score	support
0.0	0.72	0.94	0.82	31
1.0	0.92	0.68	0.78	34
accuracy			0.80	65
macro avg	0.82	0.81	0.80	65
weighted avg	0.83	0.80	0.80	65
DecisionTreeClassifier Classification Report:				
	precision	recall	f1-score	support
0.0	0.96	0.84	0.88	31
1.0	0.87	0.97	0.92	34
accuracy			0.91	65
macro avg	0.92	0.90	0.91	65
weighted avg	0.91	0.91	0.91	65
SVC Classification Report:				
	precision	recall	f1-score	support
0.0	0.47	0.23	0.38	31
1.0	0.52	0.76	0.62	34
accuracy			0.51	65
macro avg	0.49	0.50	0.46	65
weighted avg	0.49	0.51	0.47	65
LinearDiscriminantAnalysis Classification Report:				
	precision	recall	f1-score	support
0.0	0.70	0.90	0.79	31
1.0	0.88	0.65	0.75	34
accuracy			0.77	65
macro avg	0.79	0.78	0.77	65
weighted avg	0.79	0.77	0.77	65
AdaBoostClassifier Classification Report:				
	precision	recall	f1-score	support
0.0	0.93	0.87	0.90	31
1.0	0.89	0.94	0.91	34
accuracy			0.91	65
macro avg	0.91	0.91	0.91	65
weighted avg	0.91	0.91	0.91	65
RandomForestClassifier Classification Report:				
	precision	recall	f1-score	support
0.0	0.93	0.84	0.88	31
1.0	0.86	0.94	0.90	34
accuracy			0.89	65
macro avg	0.90	0.89	0.89	65
weighted avg	0.90	0.89	0.89	65
GradientBoostingClassifier Classification Report:				
	precision	recall	f1-score	support
0.0	1.00	0.77	0.87	31
1.0	0.83	1.00	0.91	34
accuracy			0.89	65
macro avg	0.91	0.89	0.89	65
weighted avg	0.91	0.89	0.89	65

Le seguenti ROC curve anche indicano il funzionamento dei vari classificatori. Rispetto

al caso precedente le curve hanno in questo caso pendenza iniziale minore, essendo le accuratezze ottenute minori. In particolare l'SVC funziona molto male.

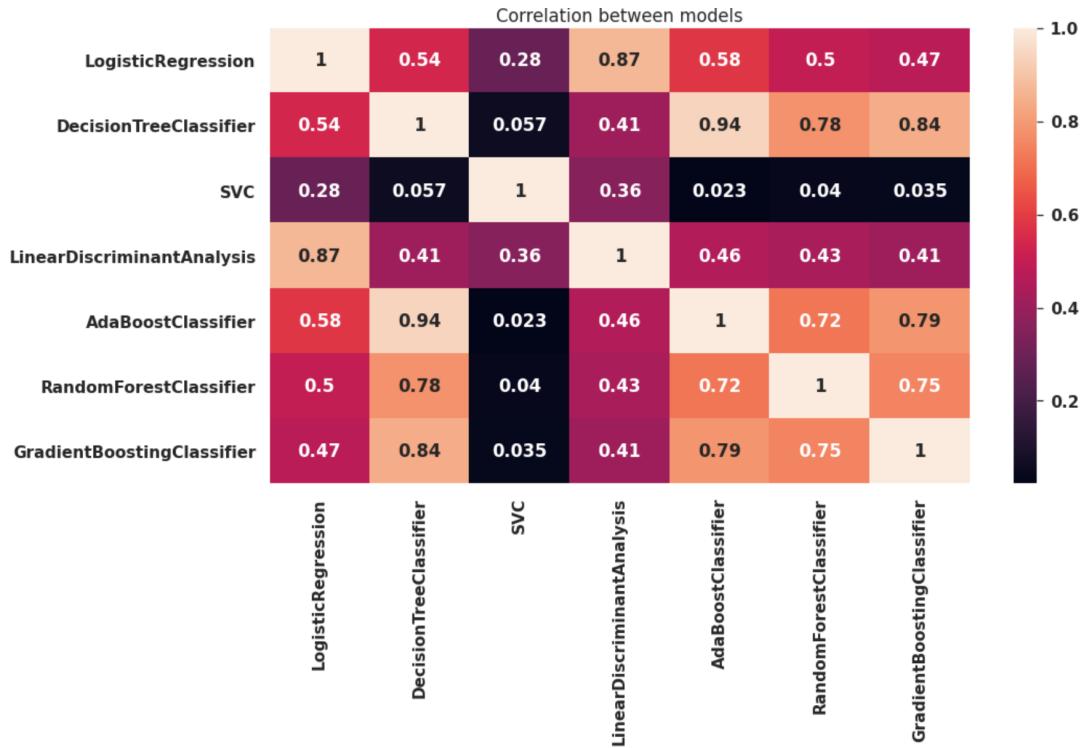


Di seguito si è voluta fare una ricerca dei parametri ottimali per i tre classificatori *decisiontree*, *randomforset* e *exgradientboosting* rispetto a delle liste di parametri attraverso una gridsearch ottenendo i seguenti score rispetto al caso senza gridsearch:

```
score without GridSearchCV: [0.7567692307692307, 0.9381538461538461]
score with GridSearchCV: [0.9346153846153846, 0.9653846153846153, 0.7450769230769231]
```

Nella seguente matrice di correlazione sono riportate le correlazioni tra i vari metodi di classificazione per vedere la correlazione dei modelli dei vari metodi, e capire quindi se vi sono classificatori che restituiscono gli stessi risultati.

CAPITOLO 3: SKLEARN



Infine, si è realizzato un classificatore composto dall'unione dei tre metodi decisiontree, randomforset e gradientboosting, secondo un sistema di voting. Il classificatore finale così ottenuto ha un'accuratezza finale superiore a 0,9, quindi molto buona.

4. Statsmodels

4.1 Introduzione alla libreria

Statsmodels è un modulo Python che fornisce classi e funzioni per la stima di molti modelli statistici diversi, così come per il calcolo di test statistici e l'esplorazione di dati. Un ampio elenco di statistiche dei risultati è disponibile per ogni estimatore. Statsmodels è costruita sopra NumPy, SciPy e matplotlib, ma contiene funzioni più avanzate per i test statistici e una modellazione che non è presente nelle librerie numeriche come NumPy o SciPy. In particolare statsmodels.tsa contiene classi di modelli e funzioni utili per l'analisi delle serie temporali. I modelli di base includono modelli autoregressivi univariati (AR), modelli autoregressivi vettoriali (VAR) e modelli di media mobile autoregressiva univariata (ARMA). I modelli non lineari includono la Markov switching dynamic regression e l'autoregressione. Comprendono anche statistiche descrittive per serie temporali, ad esempio l'autocorrelazione, e la funzione di autocorrelazione parziale, nonché le corrispondenti proprietà teoriche di ARMA.

Utilizzando il modello ARIMA, è possibile prevedere una serie temporale attraverso i valori passati. Verrà costruito un modello ARIMA ottimale partendo da zero ed estendendolo ai modelli ARIMA stagionali (SARIMA) e SARIMAX.

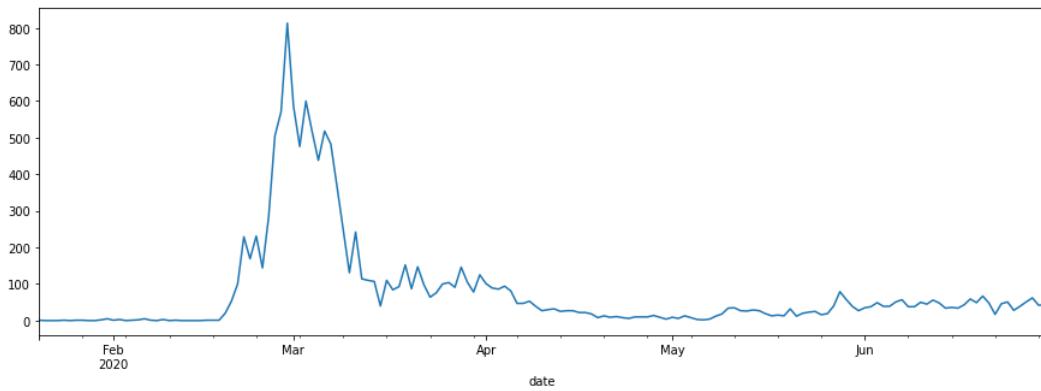
4.2 Analisi temporali e predizioni

Utilizzando la libreria statsmodels, sono stati applicati i modelli ARIMA e SARIMAX a due diverse serie temporali, che sono state estrapolate dal dataset *Time.csv* e *SeoulFloating.csv*

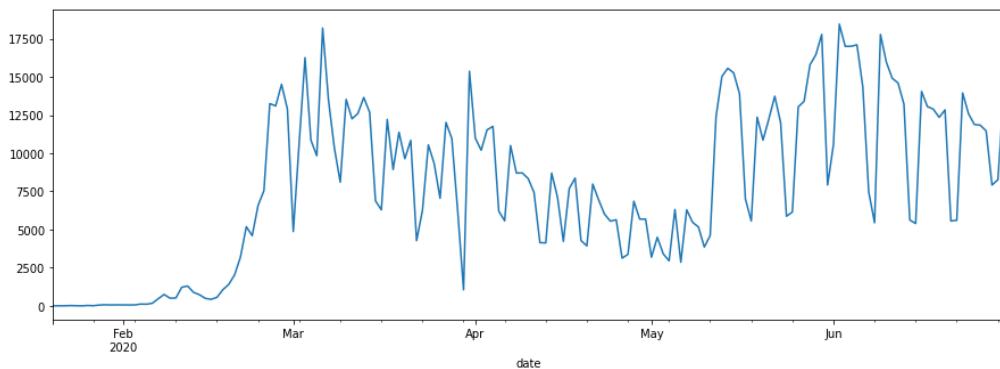
4.2.1 Andamento dei test

Si inizia l'analisi delle serie temporali dal dataset *Time.csv*. In questo file vi sono i dati cumulativi, giorno per giorno, dei test, dei positivi, dei negativi, dei guariti e dei deceduti, nell'arco di tempo che va dal 20 Gennaio 2020 al 30 Giugno 2020. In realtà per le analisi temporali non è richiesto il valore cumulativo delle diverse variabili, ma si necessita del valore giornaliero. Per fare ciò basta semplicemente sottrarre ad ogni riga il valore cumulato presente alla riga precedente, ovvero il giorno precedente, andando così a stimare gli incrementi giornalieri.

In particolare la variabile più significativa sulla quale andare a fare una previsione è il numero dei positivi, ma graficando tale andamento si vede benissimo che da aprile in poi il numero dei contagiati è rimasto sempre costante ad un valore molto basso, dovuto all'ottima politica anticovid messa in campo dal governo sud-coreano.



Invece rappresentando graficamente l'andamento dei test, si vede come essi siano comunque rimasti su numeri alti, e come essi abbiano anche una forte stagionalità settimanale, dovuta al fatto che il sabato e la domenica il numero di tamponi effettuati è molto minore rispetto agli altri giorni.



Per effettuare la previsione, verrà applicato prima un modello ARIMA e successivamente SARIMAX.

Un modello ARIMA è caratterizzato da 3 termini:

- p è l'ordine del termine AR (Auto Regressive),
- q è l'ordine del termine MA (Moving Average),
- d è il numero di differenziazioni necessarie per rendere stazionarie le serie temporali (Integrated),

e quindi per ottenere un buon modello ARIMA bisogna stimare i valori p, d, q.

Si inizia valutando il parametro d, ovvero il parametro di differenziazione (o integrazione). Bisogna fare attenzione a non differenziare eccessivamente le serie, perché una serie troppo differenziata può essere ancora stazionaria, il che a sua volta influenzera i parametri del modello. Il giusto ordine di differenza è la differenza minima necessaria per ottenere una serie quasi stazionaria che si aggira intorno ad una media definita e il grafico ACF arriva a zero abbastanza velocemente. Un modo per valutare se la serie è stazionaria è attraverso l'Augmented Dickey Fuller test. L'ipotesi nulla del test ADF è che le serie temporali non siano stazionarie. Quindi, se il valore p del test è inferiore al livello di significatività ($p < 0.05$) allora si rifiuta l'ipotesi nulla e si deduce che la serie temporale è effettivamente stazionaria. Applicando quindi l'ADF alla serie temporale in esame, ovvero quella dei test giornalieri, si osserva che il p-value è pari a 0.000170 e si deduce che la serie è già stazionaria.

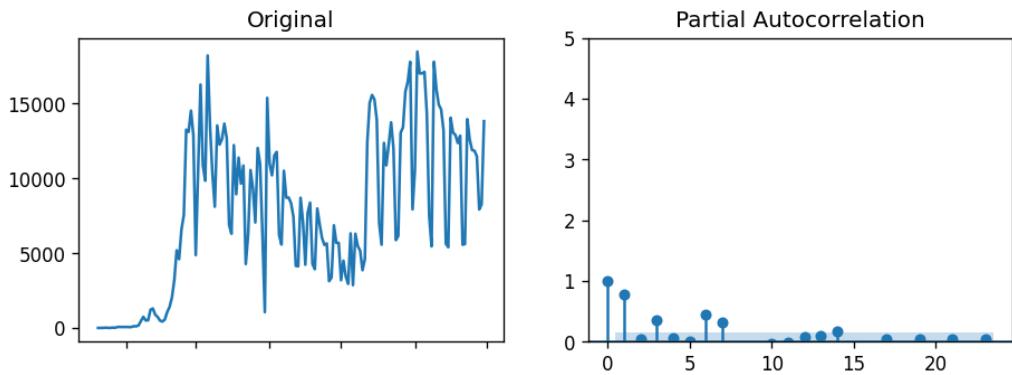
```
In [14]: result = adfuller(test)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print(result)

ADF Statistic: -4.534839
p-value: 0.000170
(-4.53483930673814, 0.00017001925947810972, 0, 162, {'1%': -3.471374345647024, '5%': -2.8795521079291966, '10%': -2.5763733302850174}, nan)
```

Perciò l'ordine d di differenziazione viene impostato uguale a 0.

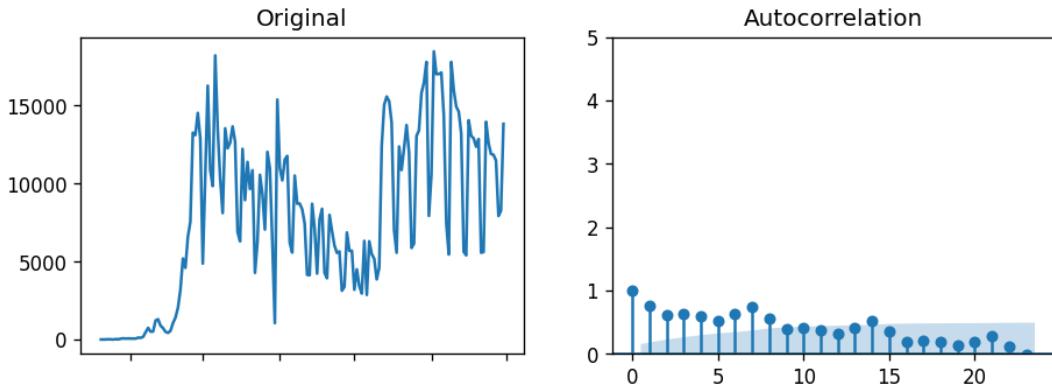
Il passo successivo è quello di identificare se il modello ha bisogno di un contributo da parte della componente AR. È possibile scoprire l'ordine di AR necessario ispezionando il grafico dell'Autocorrelazione Parziale (PACF).

Qualsiasi autocorrelazione in una serie stazionaria può essere spiegata aggiungendo un numero sufficiente di termini AR. Quindi, inizialmente si prende l'ordine dei termini AR pari al numero dei ritardi che superano il limite nel grafico PACF.



Si può osservare che il PACF lag 1 è abbastanza significativo in quanto si trova ben al di sopra dell'area limite, cosa che invece non succede per il lag 2. Quindi per ora si fissa il termine p pari a 1.

Proprio come è stato guardato il grafico PACF per il numero di termini AR, si può guardare il grafico ACF per il numero di termini MA. L'ACF è il grafico di auto correlazione e indica quanti termini MA sono necessari per rimuovere qualsiasi autocorrelazione nella serie stazionaria.



Si può facilmente osservare che circa 8 ritardi sono ben al di sopra della linea di significato, ma solo il primo è nettamente distante da essa. Nel caso di dubbio si sceglie inizialmente sempre il modello più semplice, quindi fissiamo provvisoriamente q a 1.

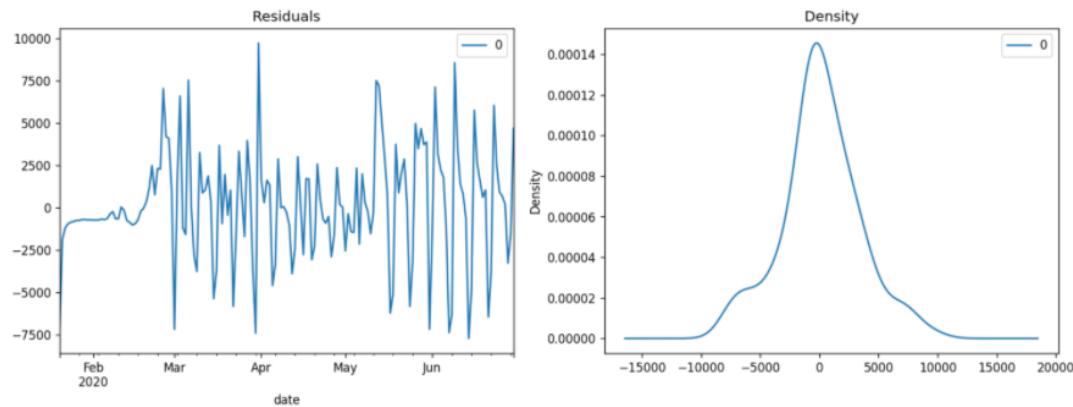
Ora, avendo determinato tutti e tre i coefficienti p, d e q si può allenare il modello ARIMA.

```
model = ARIMA(test, order=(1,0,1))
model_fit = model.fit()
print(model_fit.summary())
#ax = plt.gca()
#plt.plot(test)
#plt.plot(model_fit.fittedvalues, color='red')
#ax.legend(['test giornalieri', 'Forecast'])
```

```
ARMA Model Results
=====
Dep. Variable: test_g No. Observations: 163
Model: ARMA(1, 1) Log Likelihood -1550.423
Method: css-mle S.D. of innovations 3258.003
Date: Wed, 23 Dec 2020 AIC 3108.846
Time: 17:48:49 BIC 3121.221
Sample: 01-20-2020 HQIC 3113.870
- 06-30-2020
=====
            coef    std err      z   P>|z|      [0.025      0.975]
-----
const    7351.5069  2331.525     3.153    0.002    2781.802  1.19e+04
ar.L1.test_g  0.9592   0.033    29.446    0.000      0.895   1.023
ma.L1.test_g -0.5840   0.123    -4.730    0.000     -0.826  -0.342
Roots
=====
          Real      Imaginary      Modulus      Frequency
-----
AR.1      1.0425 +0.0000j  1.0425      0.0000
MA.1      1.7123 +0.0000j  1.7123      0.0000
-----
```

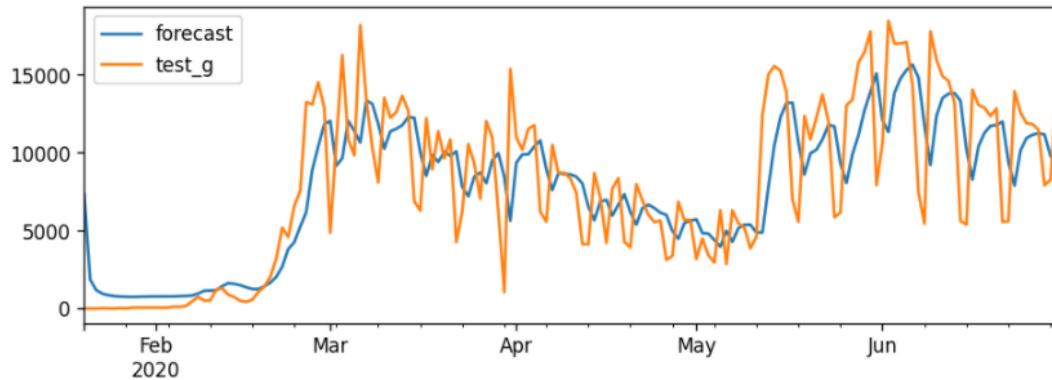
Si può subito osservare che i termini AR1 e MA1 sono molto buoni, poiché hanno un livello di significatività molto alto ($P>|z| \ll 0.05$).

Si possono ora graficare i residui per esser sicuri che non ci siano pattern particolari (cioè si cerca media e varianza costante).



I risultati ottenuti sembrano buoni, poiché gli errori residui sembrano essere contenuti con una media prossima allo zero e una varianza uniforme.

Infine vengono graficati insieme l'intera serie temporale e i dati di forecast del modello.



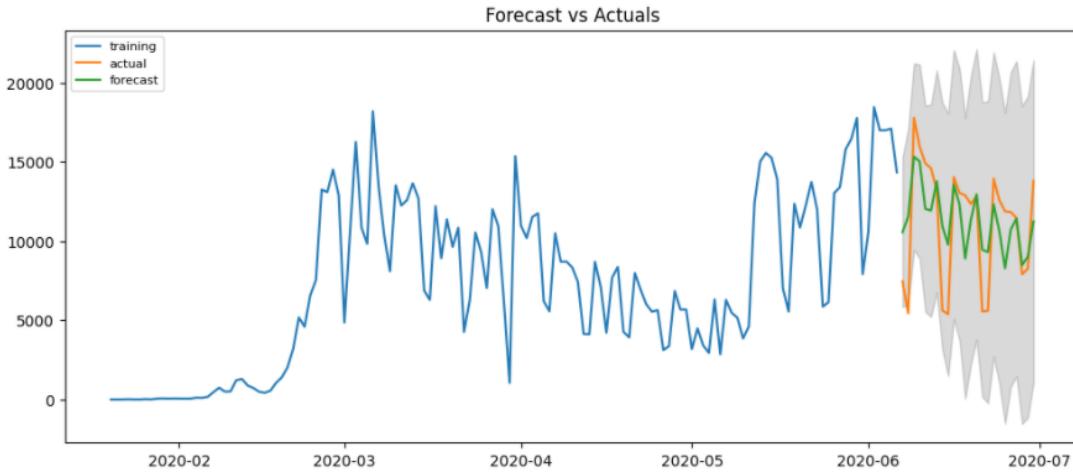
Si procede ora con la Out-of-Time Cross-Validation, nella quale si fanno alcuni passi indietro a livello temporale e si prevede il valore futuro confrontandolo con quello reale.

Per effettuare la Out-of-Time Cross-Validation, è necessario creare il set di training e di testing dividendo la serie temporale in 2 parti contigue in un rapporto di circa 85:15 (o anche 80:20) o in una proporzione ragionevole basata sulla frequenza temporale delle serie. In questo caso nell'insieme di training si prendono tutti i campioni prima del 6 Giugno 2020, mentre per quelli di testing si prendono i restanti campioni fino al 30 Giugno 2020.

Si costruisce allora il modello ARIMA con gli ordini 1, 0, 1, ma si nota subito che la previsione non è affatto buona. Perciò si decide di aumentare gli ordini di p e di q in maniera iterativa fino a raggiungere la previsione migliore, che è con gli ordini 3, 0, 10. In effetti sul grafo dell'autocorrelazione parziale anche il Lag3 è sopra la soglia, e nel grafo dell'autocorrelazione ci sono circa una decina di Lag sopra il limite.

Nella previsione si vede come il trend sia a ribasso, essendo infatti nella fase finale della pandemia e quindi con un numero di tamponi in decrescita. Inoltre, anche se il modello non è adatto per serie temporali stagionali si nota una forte stagionalità settimanale dovuta al calo dei tamponi il sabato e la domenica.

Come già detto il modello utilizzato non è il più adatto, perciò si decide di utilizzare



il modello SARIMA. Il metodo della media mobile autoregressiva integrata stagionale (Seasonal AutoRegressive Integrated Moving Average - SARIMA) modella il passo successivo nella sequenza come funzione lineare delle diverse osservazioni, degli errori residui ottenuti dal calcolo della media nelle fasi precedenti, della stagionalità e degli errori residui sulla stagionalità ottenuti nelle fasi precedenti. Esso combina il modello ARIMA con la capacità di eseguire la stessa autoregressione, differenziazione e modellazione della media mobile a livello stagionale. Il metodo è adatto per serie temporali univariate con trend e/o componenti stagionali. La notazione per il modello comporta la specificazione dell'ordine per i modelli AR(p), I(d) e MA(q) come parametri di una funzione ARIMA e dei parametri AR(P), ID(D) e MA(Q) ed m per la componente stagionale; m rappresenta il numero di passi in ciascuna stagione (e quindi il periodo della stagione).

Siccome la stagionalità della serie è settimanale, si definisce $m = 7$, poiché si ha un campione al giorno. Il modello creato sarà allora il seguente (si usano i coefficienti p , d e q iniziali, ovvero 1, 0, 1).

CAPITOLO 4: STATSMODELS

```
# Construct the model
mod = sm.tsa.SARIMAX(test_g, order=(1,0,1), seasonal_order=(1, 0, 1, 7))
# Estimate the parameters
res = mod.fit()

print(res.summary())

```

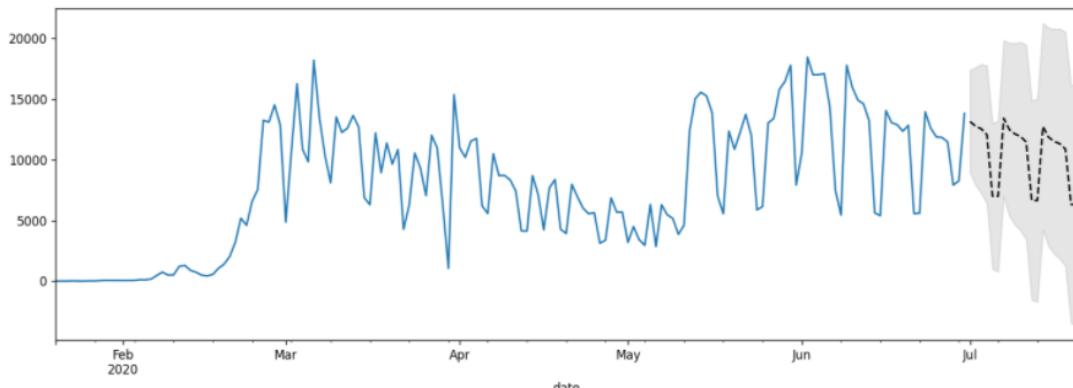
SARIMAX Results

Dep. Variable:	test_g	No. Observations:	163			
Model:	SARIMAX(1, 0, 1)x(1, 0, 1, 7)	Log Likelihood	-1486.962			
Date:	Wed, 23 Dec 2020	AIC	2983.923			
Time:	17:47:58	BIC	2999.392			
Sample:	01-20-2020 - 06-30-2020	HQIC	2990.203			
Covariance Type:	opg					
coef	std err	z	P> z	[0.025	0.975]	
ar.L1	0.9251	0.027	34.394	0.000	0.872	0.978
ma.L1	-0.3741	0.068	-5.541	0.000	-0.506	-0.242
ar.S.L7	0.9487	0.040	23.939	0.000	0.871	1.026
ma.S.L7	-0.5664	0.106	-5.360	0.000	-0.774	-0.359
sigma2	4.718e+06	3.81e+05	12.374	0.000	3.97e+06	5.47e+06

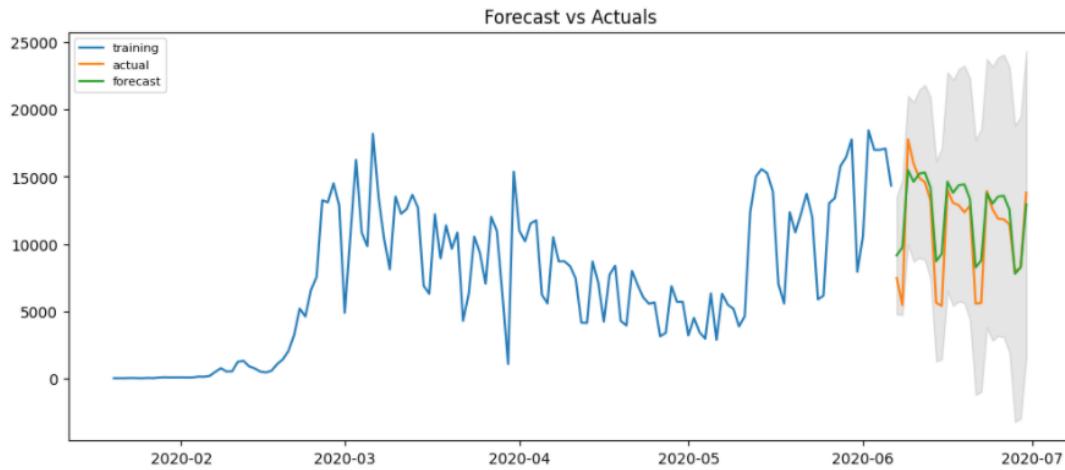
Ljung-Box (L1) (Q): 0.02 Jarque-Bera (JB): 55.11
Prob(Q): 0.90 Prob(JB): 0.00
Heteroskedasticity (H): 0.85 Skew: 0.45
Prob(H) (two-sided): 0.56 Kurtosis: 5.70

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).

I valori dei termini P sono tutti < 0.05, il che è ottimo. Infatti se si grafica la previsione fatta si vede immediatamente che la previsione è ottima. In questo caso come insieme di training viene usata l'intera serie temporale, e perciò non si hanno valori di testing con i quali confrontare la previsione.

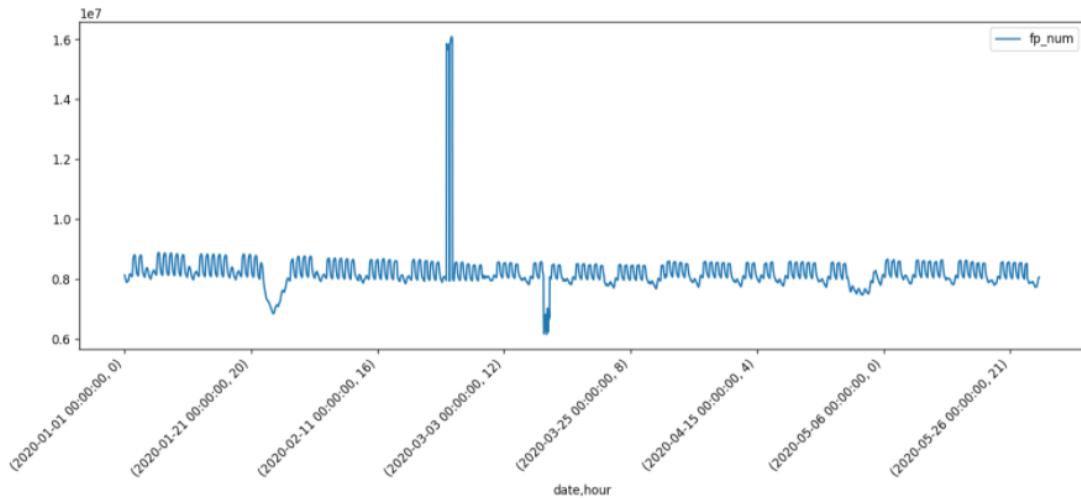


Perciò effettuiamo la stessa divisione fatta in precedenza, per gli insiemi di training e di test. In questo modo possiamo eseguire una Out-of-Time Cross-Validation e confrontare i nostri risultati con i valori reali. Anche in questo caso costruiamo il modello SARIMA con gli ordini di p, d e q = 1, 0, 1 , gli ordini P , D, Q = 1, 0, 1 e m = 7.



4.2.2 Andamento degli spostamenti

Nel dataset *SeoulFloating.csv* vi sono informazioni sul numero di persone in movimento e che si stanno spostando all'interno della provincia di Seoul. In particolare tale informazione viene raccolta ora per ora, dividendo il numero dei *floating people* per fascia di età, per sesso e per città all'interno della provincia. Per rappresentare un'unica serie temporale si vanno allora a raggruppare i valori di ogni categoria, ora per ora, creando così un unico valore che rappresenta l'intera popolazione in movimento all'interno della provincia di Seoul.



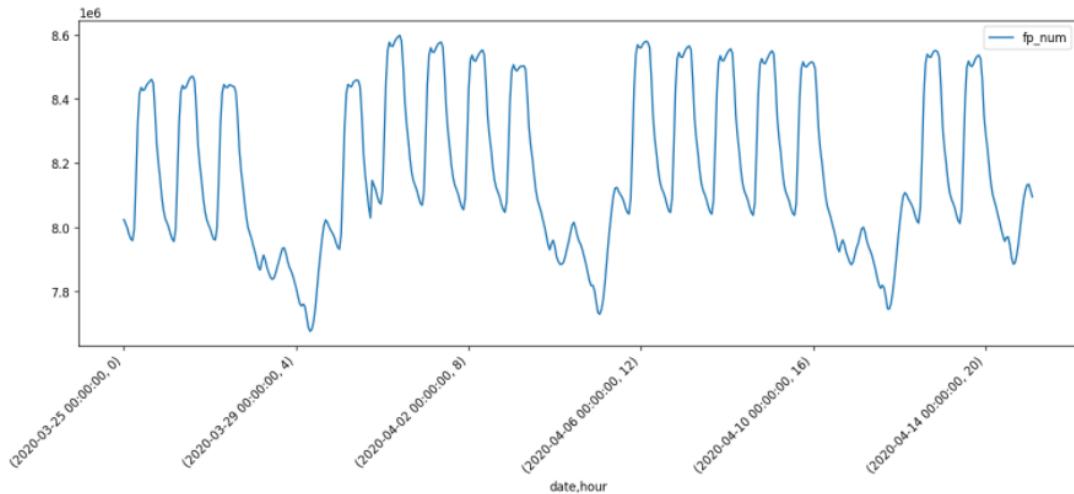
Si può subito osservare che c'è stato un picco di persone in movimento nel giorno 23 Febbraio 2020.

Proprio in quei giorni in Corea del Sud c'è stato un raduno a Daegu presso la Chiesa di Gesù Shincheonji, il Tempio del Tabernacolo della Testimonianza, che ha mosso

fp_num		
date	hour	
2020-02-23	20	16096140

grandissime masse di persone da tutta la Corea.

Se si prova a focalizzare ed ingrandire l'andamento in un periodo successivo a tale picco, si osserva la presenza di una doppia stagionalità, ovvero una giornaliera con un calo degli spostamenti nelle ore notturne e una settimanale con un calo durante il fine settimana. Questo avviene proprio perché la Corea del Sud è un paese molto avanzato e le persone sono poco dediti ai divertimenti e allo svago, e spesso esse escono di casa solo per recarsi nel posto di lavoro.



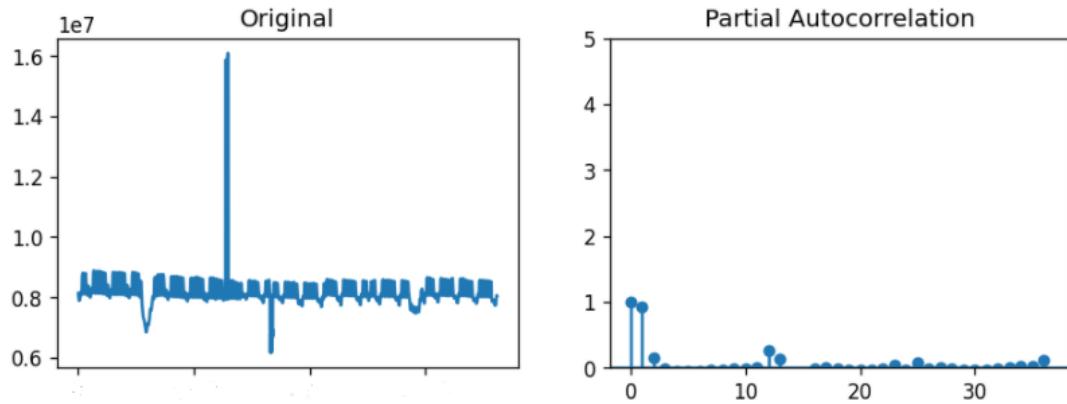
Si inizia ora con la costruzione del modello ARIMA. Come fatto in precedenza si va per prima cosa a determinare se la serie è stazionaria o meno, attraverso il test ADF.

```
result = adfuller(spostamenti)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print(result)

ADF Statistic: -12.014647
p-value: 0.000000
(-12.014647405193678, 3.122787220810876e-22, 0, 3615, {'1%': -3.4321602211641724, '5%': -2.8623398545767365, '10%': -2.56719577511483}, nan)
```

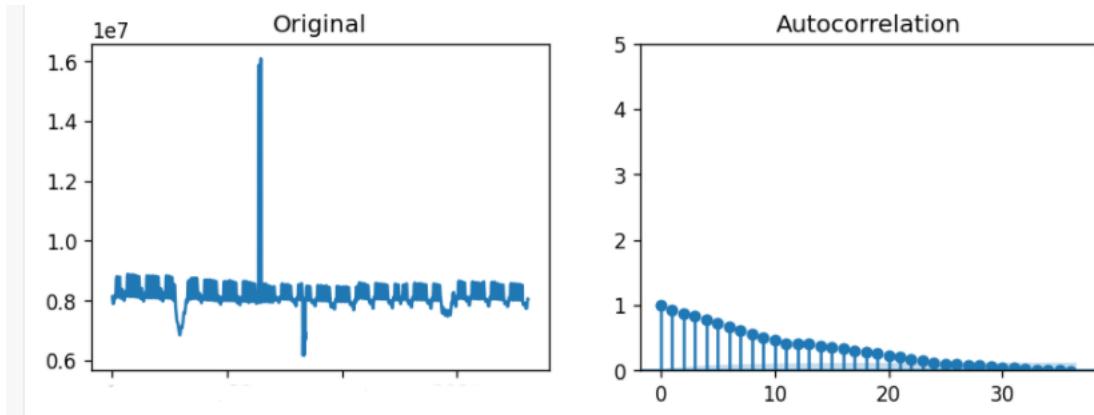
Dal risultato si apprende che il p-value è quasi nullo e < 0.05 , perciò la serie è stazionaria e non ha bisogno di essere differenziata: l'ordine d è allora uguale a 0.

Si passa poi al calcolo dell'ordine dei termini AR. Per fare ciò si costruisce il grafico di autocorrelazione parziale.



Dal grafico di autocorrelazione parziale si può osservare che i primi due lag sono al di sopra della soglia, perciò si sceglie il termine p uguale a 2.

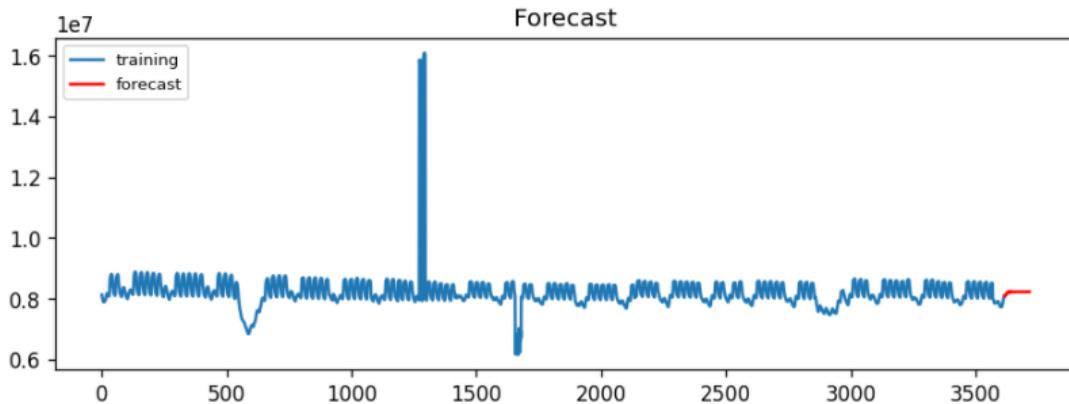
Infine per il calcolo del numero dei termini MA si utilizza il grafico dell'autocorrelazione.



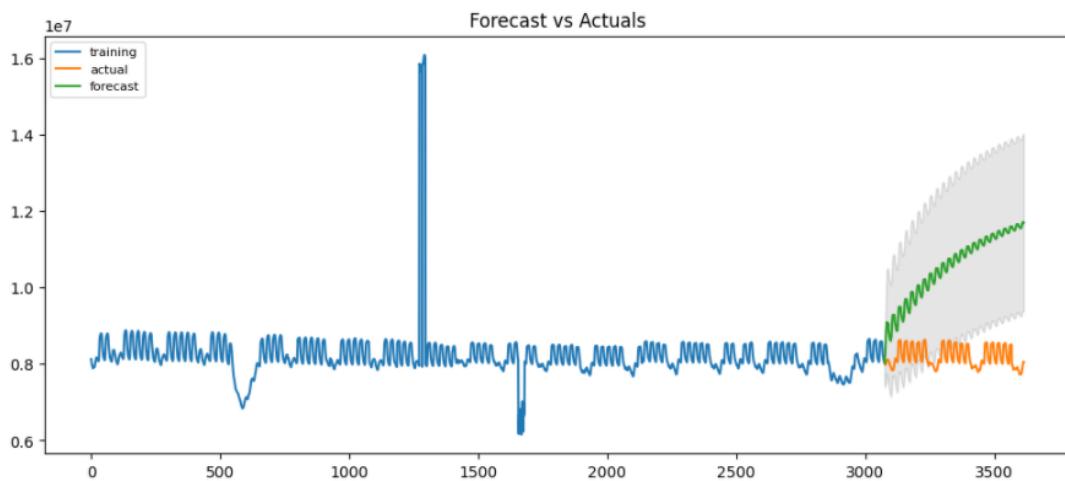
In questo caso si hanno ben 20 lag sopra la soglia limite, perciò nel modello si dovrebbe avere un ordine q pari a 20.

Come primo modello si va a sviluppare ARIMA, con ordini 2, 0, 20. Poiché, come già detto precedentemente, la serie presenta una forte e doppia stagionalità, il modello ARIMA non sarà molto adatto, ed infatti i risultati ottenuti non sono per niente buoni.

Si va allora a creare un modello SARIMA nella quale si potrà considerare la stagionalità della serie, ma purtroppo neanche in questo modello c'è la possibilità di inserire una doppia stagionalità. Come prima cosa si divide il dataset in due insiemi, ovvero quello di training e quello di testing, con percentuali 85% - 15%. Anche con il modello SARIMA non si hanno buoni risultati poiché si è costretti a diminuire la complessità ($p, d, q = 2, 0, 10$) del modello e utilizzare $m = 24$, anche se si sarebbero ottenuti risultati migliori con una stagionalità pari a una settimana, quindi $m = 168$, ma se si



provasse ad implementare questa soluzione i tempi di esecuzione del codice non sarebbero ragionevoli.



Perciò per provare ad ottenere previsioni migliori utilizzando modelli non troppo complessi si è deciso di utilizzare un dataset ridotto che comprenda soltanto i valori dall’indice 1776 in poi, ovvero i valori che si trovano sia dopo il picco positivo che negativo di spostamenti.

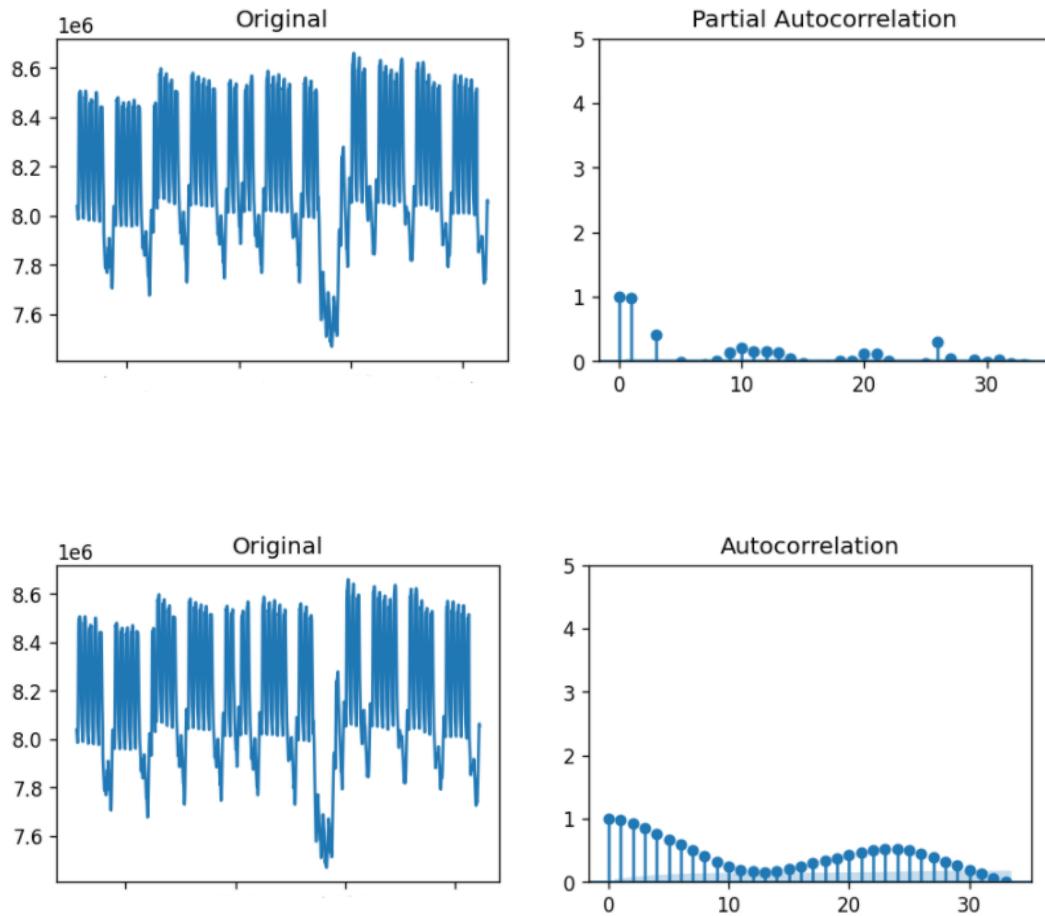
Come già fatto, bisogna anche in questo caso calcolare gli ordini p, d e q del modello SARIMA, e lo si fa attraverso il p-value, il grafico di autocorrelazione parziale e il grafico di autocorrelazione.

```

result = adfuller(spostamenti[1776:])
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print(result)

ADF Statistic: -4.521882
p-value: 0.000179
(-4.521882171288332, 0.000179491909860238, 0, 1839, {'1%': -3.4339108761687758, '5%': -2.8631129277746887, '10%': -2.5676073721
917216}, nan)

```



Siccome il p-value è « 0.05 la serie è stazionaria e quindi l'ordine di differenziazione d è uguale a 0. Nel grafo di autocorrelazione parziale si ha il primo lag ben al di sopra della soglia, e quindi si sceglie l'ordine p pari a 1. Infine nel grafo di autocorrelazione si hanno ben 9 lag sopra la soglia limite, e quindi si sceglie l'ordine q pari a 9 (inizialmente si sceglierà questo ordine molto minore per provare a semplificare il modello).

Anche in questo caso si divide il dataset in due insiemi: uno di training e uno di testing, con percentuali 85% - 15%. Si costruisce quindi il modello SARIMA con gli ordini p, d, e q uguali a 1, 0, 3 , e ordini P, D , Q pari a 1, 0, 1, con una stagionalità settimanale, quindi con m = 168 poiché i dati sono per ogni ora. Si sceglie q = 3 perché i risultati sono comunque molto buoni e al tempo stesso si è ridotta anche la complessità del modello. Inoltre in questo caso si riesce ad impostare m = 168 perché la serie temporale ha ridotto notevolmente la sua complessità, e quindi il calcolo avviene in tempi ragionevoli.

```
# Construct the model
mod = sm.tsa.SARIMAX(train, order=(1,0,3), seasonal_order=(1, 0, 1, 168), trend='c')
# Estimate the parameters
res = mod.fit()

print(res.summary())
```

```
SARIMAX Results
=====
Dep. Variable:          fp_num    No. Observations:             1564
Model:                 SARIMAX(1, 0, 3)x(1, 0, [1], 168)   Log Likelihood:        -17796.167
Date:                  Mon, 28 Dec 2020   AIC:                   35608.334
Time:                  14:30:00       BIC:                   35651.174
Sample:                  0 - 1564   HQIC:                  35624.260
Covariance Type:            opg
=====
              coef    std err      z   P>|z|      [0.025    0.975]
-----
intercept  1.136e+05  2.4e+04    4.735   0.000   6.66e+04  1.61e+05
ar.L1      0.9118     0.016    57.085   0.000     0.881    0.943
ma.L1      1.1540     0.028    41.695   0.000     1.100    1.208
ma.L2      0.8055     0.060    13.321   0.000     0.687    0.924
ma.L3      0.3267     0.051    6.386   0.000     0.226    0.427
ar.S.L168  0.8426     0.016    53.426   0.000     0.812    0.873
ma.S.L168 -0.4000     0.032   -12.438   0.000    -0.463   -0.337
sigma2     7.581e+08  4.603   1.65e+08   0.000   7.58e+08  7.58e+08
-----
Ljung-Box (L1) (Q):      9.61   Jarque-Bera (JB):        14780.72
Prob(Q):                  0.00   Prob(JB):                  0.00
Heteroskedasticity (H):  1.05   Skew:                      0.12
Prob(H) (two-sided):     0.62   Kurtosis:                  18.06
=====
```

