Cavenati Laura Matricola 864000

l.cavenati1@campus.unimib.it

Insegnamento di Streaming Data Management and Time Series Analysis

Laurea magistrale in Data Science

PREVISIONI AL BUOIO

ABSTRACT

Il progetto ha come obiettivo la definizione, lo sviluppo e la validazione di un sistema predittivo per dati di tipo time series. Vengono testati cinque modelli, sia lineari che non lineari (Machine Learning):

- SARIMA (Seasonal AutoRegressive Integrated Moving Average)
- UCM (modelli a componenti non osservati)
- TBATS (Exponential smoothing state space model with Box-Cox transformation, ARMA errors, Trend and Seasonal components)
- PROPHET
- LSTM (Long Short-Term Memory)

1 DATA PREPROCESS

Il dataset contiene dati orari dal 01/09/2018 al 31/08/2020. Sono presenti dei valori mancanti. Mancano i valori associati alle datetime 2019-03-31 03:00:00 e 2020-03-29 03:00:00 a causa del passaggio dall'ora solare a quella legale. Come consigliato da chi ci ha fornito il dataset sostituisco i due valori mancanti con il valore all'ora precedente. Mancano anche tutti i valori associati alla data 2020-05-31. Facendo una breve ricerca in internet, non sembra essere accaduto niente di eclatante in quella data. Poiché, come si vedrà più avanti, c'è una stagionalità settimanale oltre che giornaliera, ho deciso di sostituire questi valori mancanti con una media dei valori una settimana prima e una dopo, sempre alla stessa ora.

2 VISUALIZZAZIONI INIZIALI

Inizio costruendo delle visualizzazioni per vedere l'andamento dei dati.

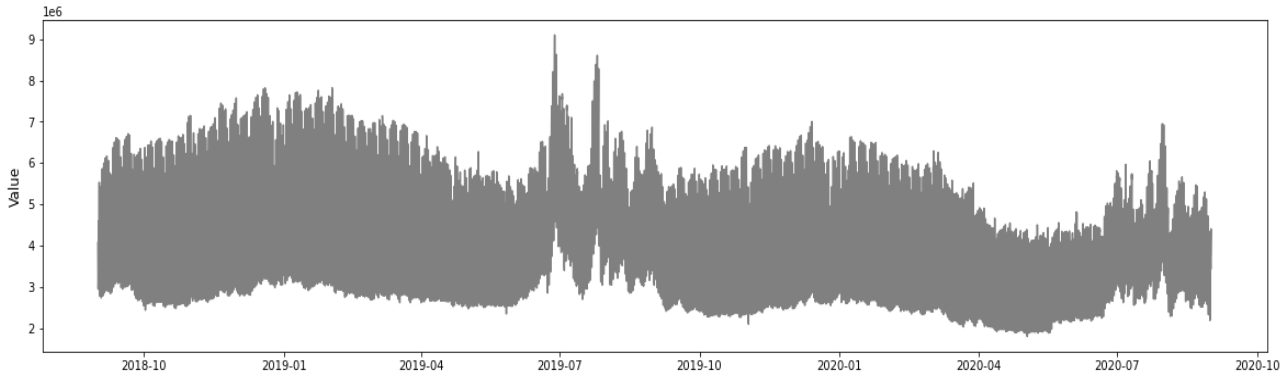

Figura 1-Serie storica da analizzare

Sembrano esserci dei valori anomali verso fine giugno e fine luglio 2019. Se conoscessimo cosa rappresentano i dati forse potremmo analizzare meglio queste anomalie ed eventualmente introdurre delle variabili che le giustifichino.
Nella Figura 2 visualizzo solo una certa ora così è graficamente più chiara la stagionalità settimanale e annuale.

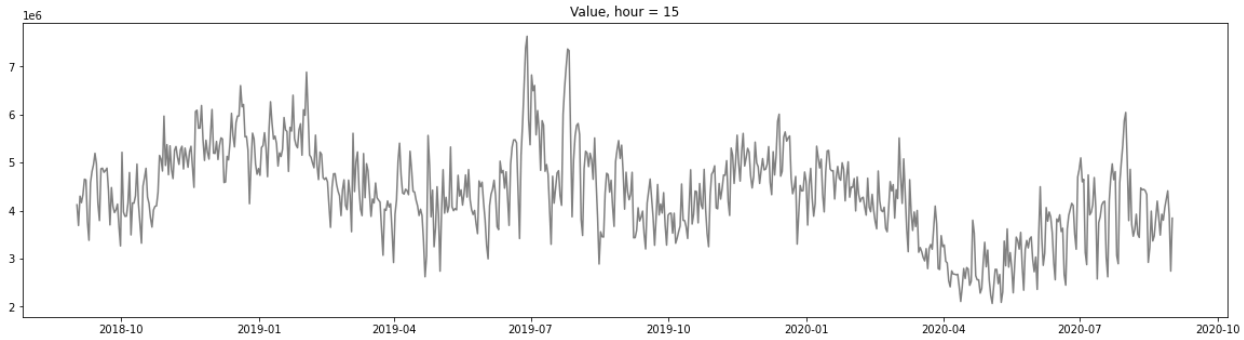

Figura 2 – Serie storica filtrata all'ora 15

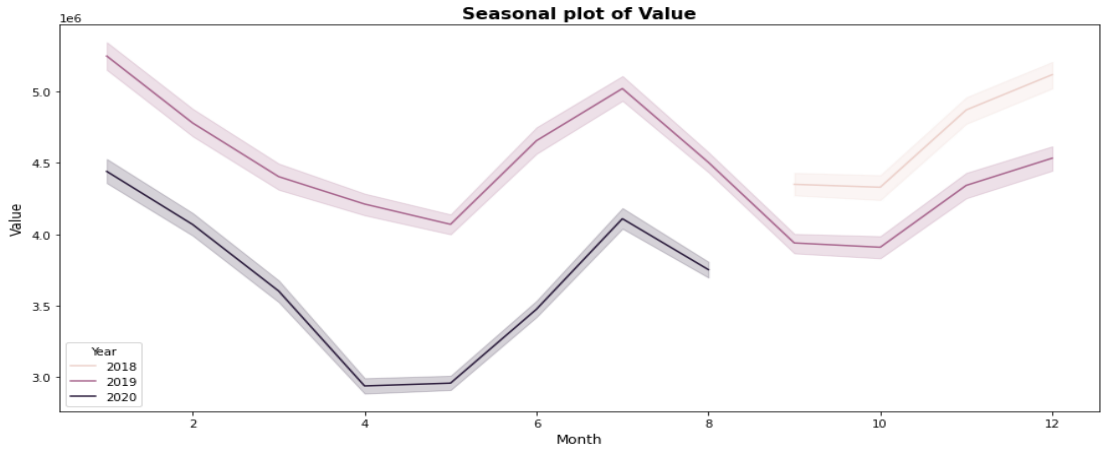

Figura 3 – Stagionalità annuale della serie

Vi è chiaramente un andamento stagionale annuale: si notano dei picchi nei mesi di dicembre-gennaio e luglio. Sembra esserci anche un trend decrescente, ipotesi confermata dalla Figura 4 di sinistra. Per essere sicuri di questa ipotesi dovremmo avere a disposizione più dati; tale trend decrescente potrebbe essere dovuto solo al covid. Dalla Figura 3, nei mesi di marzo-aprile-maggio 2020 si nota una decrescita anomala dovuta al covid-19.

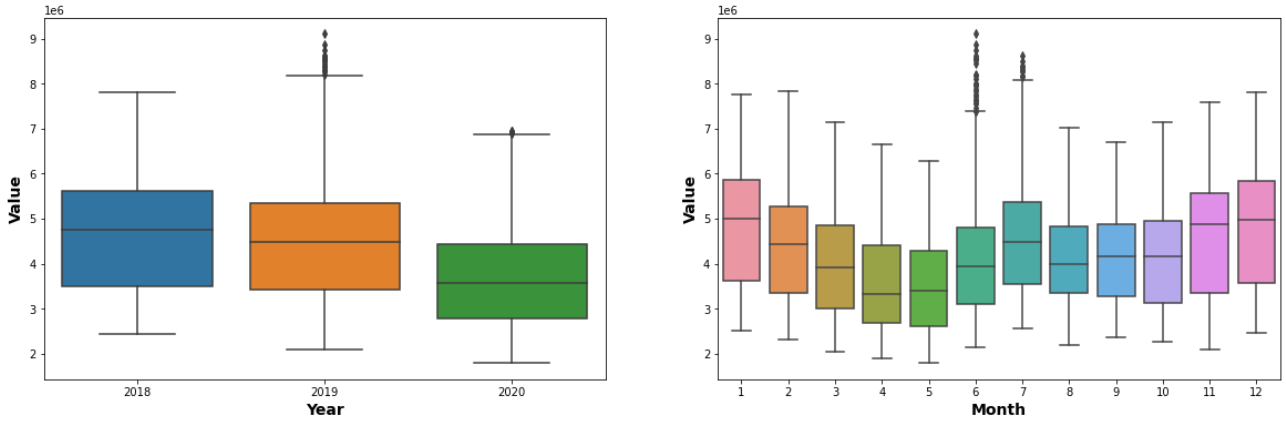

Figura 4 – Box plot dei dati raggruppati annualmente (sinistra) e mensilmente (destra)

I box plot confermano la presenza di outliers nei mesi di giugno e luglio del 2019. L'andamento intra annuo nella Figura 4 di destra non è più così chiaramente visibile come lo era nel grafico precedente.

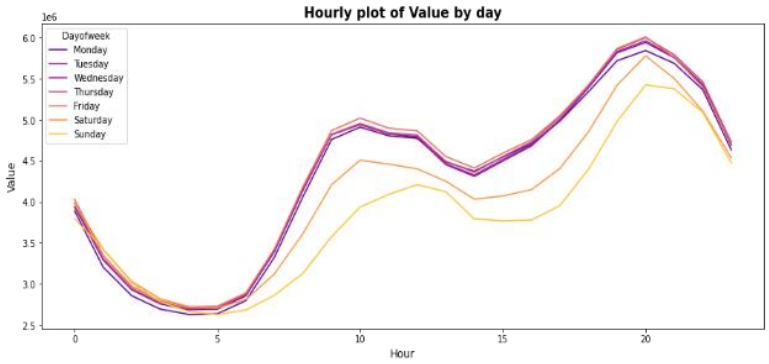

Figura 5 – Andamento giornaliero in relazione al giorno della settimana

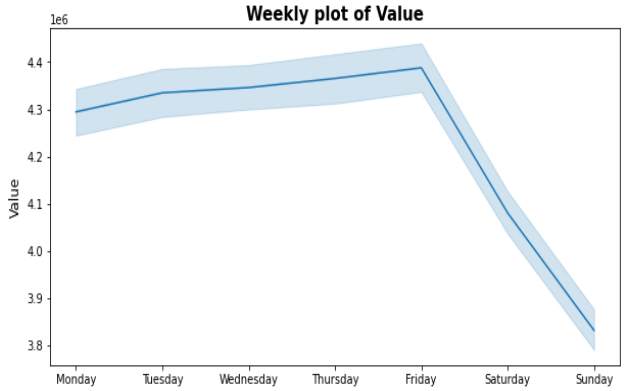

Figura 6 – Andamento settimanale

Come mostrano le Figura 5 e Figura 6 vi è anche un andamento giornaliero e infrasettimanale: ci sono dei picchi verso le 10 e le 20. La domenica il picco mattutino è verso mezzogiorno, nel weekend i valori diminuiscono vistosamente.

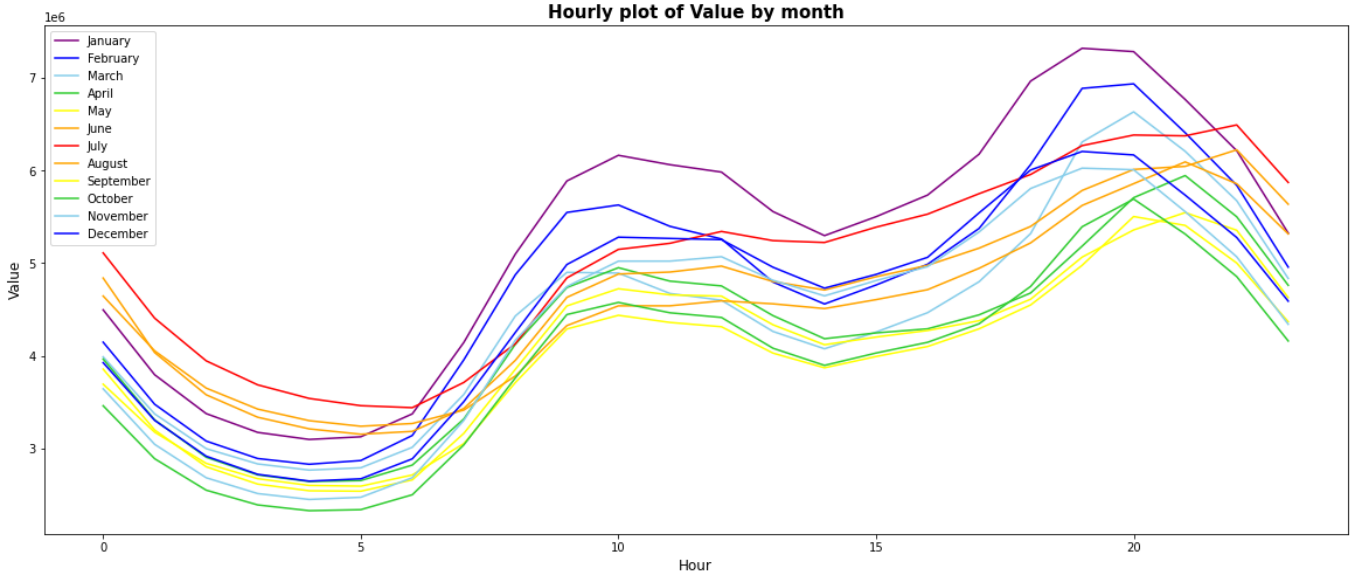

Figura 7 – Andamento giornaliero al variare dei mesi

Come si vede in Figura 7, la stagionalità giornaliera varia al variare dei mesi. Nei mesi estivi (colori giallo-arancio-rosso) i picchi sono una-due ore in ritardo rispetto ai picchi dei mesi invernali (colori azzurro-blu-viola). Questo fa pensare a dei dati influenzati anche da indici naturali come la temperatura.

Il covid sembra aver fortemente influenzato i dati: in questo periodo la serie oltre a diminuire in modo anomalo, presenta una variabilità settimanale molto inferiore rispetto al resto dell'anno. Ho creato quindi una variabile, covid, che assume valore 1 dopo il 2020-03-09, 0 prima.

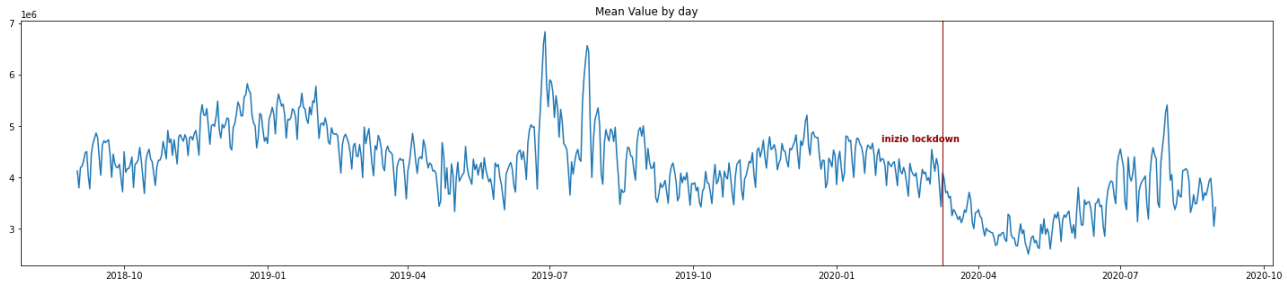

Figura 8 – Andamento giornaliero medio della serie. Focus su fattore covid

Da queste prime analisi visuali la serie sembra essere influenzata, oltre che da fenomeni naturali, anche da fenomeni artificiali. Potrebbe aver senso quindi introdurre delle variabili dummy contenenti le festività. Poiché abbiamo a disposizione solo due anni di dati, e poiché nei grafici non si notano evidenti anomalie nei giorni legati alle festività, decido di non introdurle.

3 MODELLI

Utilizzerò come validation set gli ultimi quattro mesi. In questo modo nel train sono presenti sia marzo che aprile 2020, primi mesi in cui i valori subiscono una variazione a causa del covid.

Observations Train: 14592 83.17%
Observations Validation: 2952 16.83%

3.1 ARIMA

3.1.1 STAZIONARIETÀ

Uno dei requisiti per i modelli ARIMA è che le serie temporali siano stazionarie. Trasformazioni come i logaritmi possono aiutare a stabilizzare la varianza di una serie temporale. Le differenze invece possono aiutare a stabilizzare la media di una serie temporale.
Osservando i dati, non si nota a colpo d'occhio una non stazionarietà in varianza. Provo quindi ad analizzare la relazione tra le medie e le deviazioni standard giornaliere:

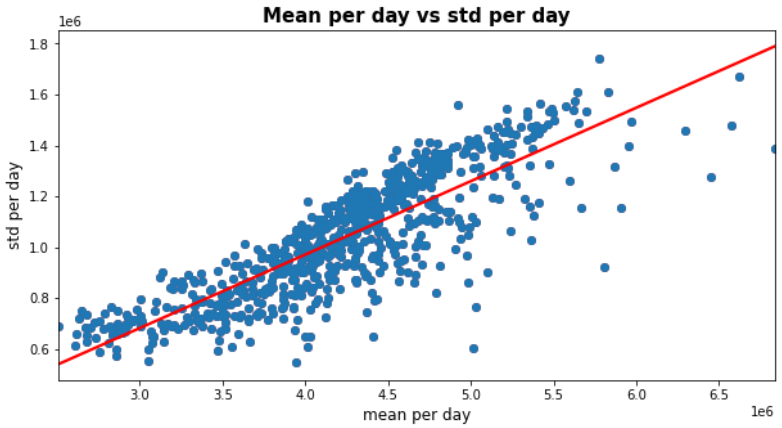

Figura 9 - relazione tra le medie e le deviazioni standard giornaliere

Poiché la relazione tra media e deviazione standard è approssimativamente lineare la trasformazione logaritmica potrebbe essere in grado di rendere la serie storica stazionaria in varianza. Il miglior modo per capire se la trasformazione è necessaria è verificare quale modello, quello con o senza trasformazione risulta migliore. Costruirò quindi dei modelli con e senza trasformazione.

Esistono anche numerosi metodi per verificare la stazionarietà di una serie, tra cui il test di Ljung-Box, il Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test e il test Augmented Dickey-Fuller.
Come mostrato in Figura 10, a un livello di significatività dello 0.95, la serie risulta stazionaria secondo il test ADF e non stazionaria secondo il test KPSS. Secondo la teoria è dunque stazionaria per differenza. Applicando una differenza stagionale la serie risulta stazionaria per entrambi i test.

```
Results of Dickey-Fuller Test:
Test Statistic                -5.810651e+00
p-value                        4.409576e-07
Lags Used                      4.200000e+01
Number of Observations Used    1.454900e+04
Critical Value (1%)           -3.430800e+00
Critical Value (5%)           -2.861739e+00
Critical Value (10%)          -2.566876e+00
```

```
Results of KPSS Test:
Test Statistic            9.070323
p-value                   0.010000
Lags Used                42.000000
Critical Value (10%)      0.347000
Critical Value (5%)       0.463000
Critical Value (2.5%)     0.574000
Critical Value (1%)       0.739000
```

Figura 10 – risultati test ADF e KPSS

Mi sono domandata se avesse senso applicare un'ulteriore differenza semplice. Nonostante la serie non sembri ancora del tutto stazionaria, i test provati (KPSS, ADF) sono concordi nell'affermare di non applicare una differenza semplice oltre a quella stagionale. Decido quindi di applicare solo una differenza stagionale.

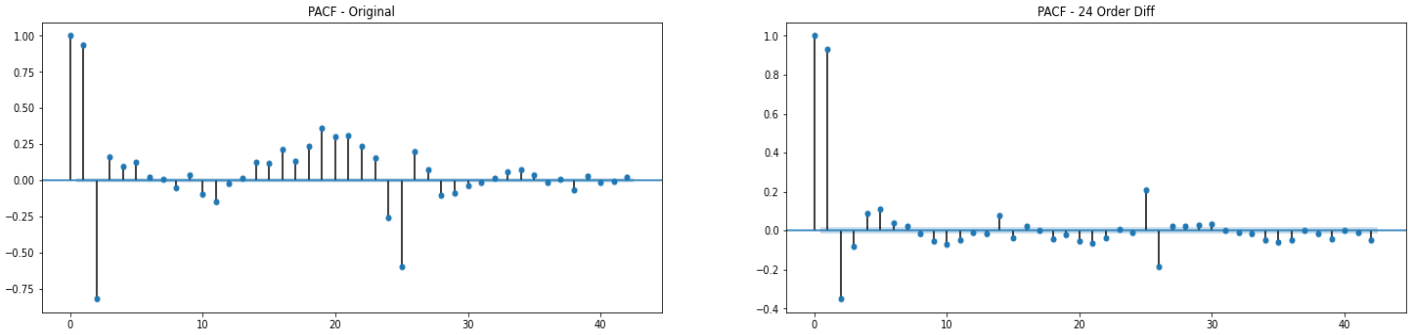

Figura 11 – PACF della serie originale e differenziata

Il grafico di destra della Figura 11 mostra che potrebbe essere buono un ordine p=1 o p=2, P=1, Q=1.

3.1.2 MODELLO SARIMA

Per determinare i miglior valore per i parametri non stagionali, p e q, uso un approccio grid search. Costruisco i modelli sia con i dati originali che con quelli log-trasformati.

	Senza log-trasformazione			Con log-trasformazione		
Modello	**MAE Train**	**MAE Validation**	**AIC**	**MAE Train**	**MAE Validation**	**AIC**
SARIMA(0,0,0)(1,1,1)$_{24}$	276458.0	716203.0	417218	274785.7	726388.8	-29681
SARIMA(0,0,1)(1,1,1)$_{24}$	157702.5	710287.1	406377	152181.4	726320.8	-46611
SARIMA(0,0,2)(1,1,1)$_{24}$	110341.7	707535.0	400669	104576.2	726938.0	-57133
SARIMA(1,0,0)(1,1,1)$_{24}$	84912.7	718818.9	383424	82676.4	726202.6	-63294
SARIMA(1,0,1)(1,1,1)$_{24}$	69836.8	711543.9	378389	66682.4	727167.8	-68421
SARIMA(1,0,2)(1,1,1)$_{24}$	66466.6	710414.0	377501	63291.2	727247.9	-69288
SARIMA(2,0,0)(1,1,1)$_{24}$	67167.6	713421.9	377707	64429.8	726196.4	-68910
SARIMA(2,0,1)(1,1,1)$_{24}$	65770.2	711881.0	377324	62807.1	726737.0	-69326
SARIMA(2,0,2)(1,1,1)$_{24}$	65584.2	711547.3	377291	62564.1	726840.6	-69379

Tabella 1 – performance al variare dei parametri p e q per i dati originali e log trasformati

La trasformazione logaritmica non sembra migliorare, in termini di MAE del validation, i modelli. Decido quindi di non applicare una trasformazione logaritmica ai dati. Il miglior modello in termini di AIC è quello SARIMA(2,0,2)(1,1,1)$_{24}$

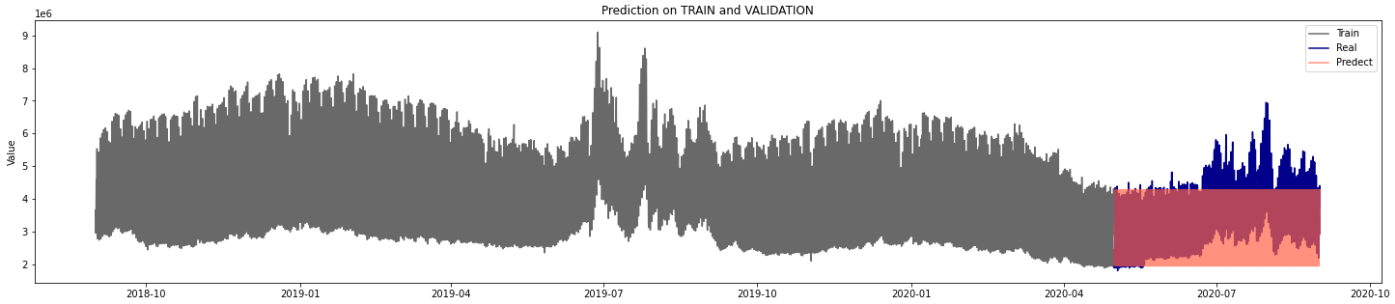

Figura 12 - focus delle previsioni sulla prima parte del validetion set

Come si vede in Figura 12 e Figura 13, il modello ha colto la stagionalità giornaliera. Non considera però la stagionalità infrasettimanale, annua e il fattore covid.

3.1.3 MODELLI SARIMAX

Per far fronte alla stagionalità multipla, è necessario aggiungere regressori esterni al modello SARIMA. Inserisco quindi delle variabili stagionali costruite tramite serie di Fourier con periodo 168 (settimanale) e 8766 (annuale). Per scegliere il più performante numero di parametri applico un approccio grid search.

n° armoniche settimanali	n° armoniche annuali	MAE Train	MAE Validation	AIC
10	5	317730.8	333797.9	109903273
10	10	572463.5	380325.1	665183946
10	15	381439.4	385529.9	336896062
10	20	557208.9	411467.0	628275046
10	25	387912.9	426977.7	141650080
15	5	193175.9	335202.1	18440191
15	10	224196.9	383056.5	29027425
15	15	187260.6	387941.1	18869288
15	20	194926.2	414327.8	13025599
15	25	154852.2	430068.4	10954255

Tabella 2 – performance modello SARIMAX al variare del numero di sinusoidi

Il miglior modello in termini di MAE sembra essere quello SARIMAX $(2,0,2)(1,1,1)_{24}$ con 10 armoniche per la stagionalità settimanale e 5 per quella annuale. Aggiungo infine la variabile step 'covid' precedentemente costruita. Non si hanno miglioramenti in termini di MAE (MAE train: 81200.8, MAE validation: 384351.2).
Decido quindi di non includere la variabile 'covid' nel modello. Come si nota in Figura 15 e Figura 16, stavolta il modello coglie anche la stagionalità settimanale e annuale. Ottengo le seguenti performance: MAE train: 308301.8, MAE validation: 334117.9.

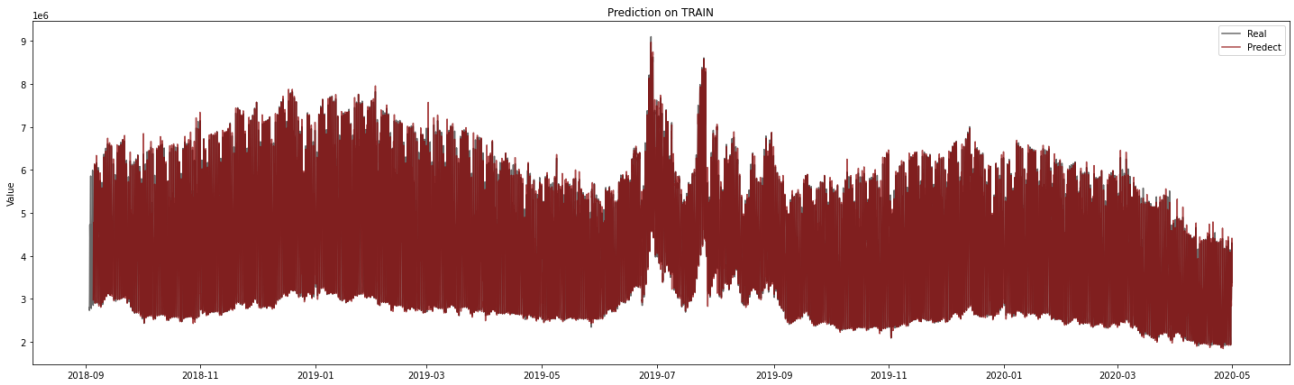

Figura 13 - previsioni sul train set

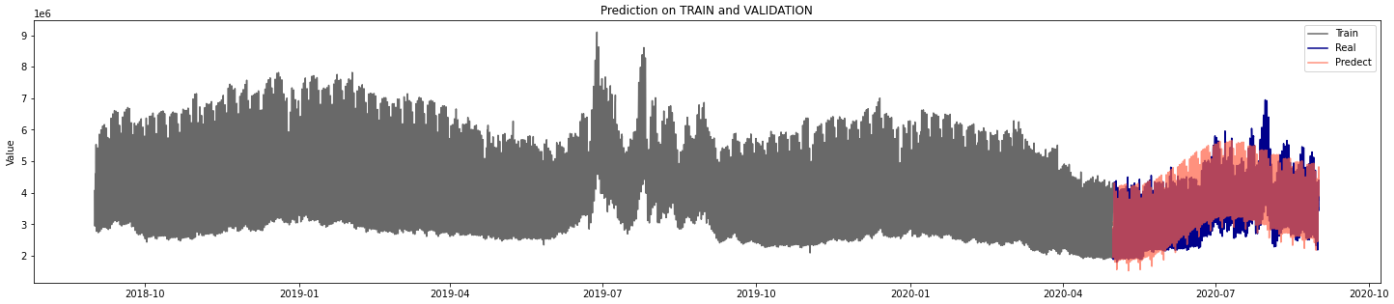

Figura 14 -- previsioni sul validation set

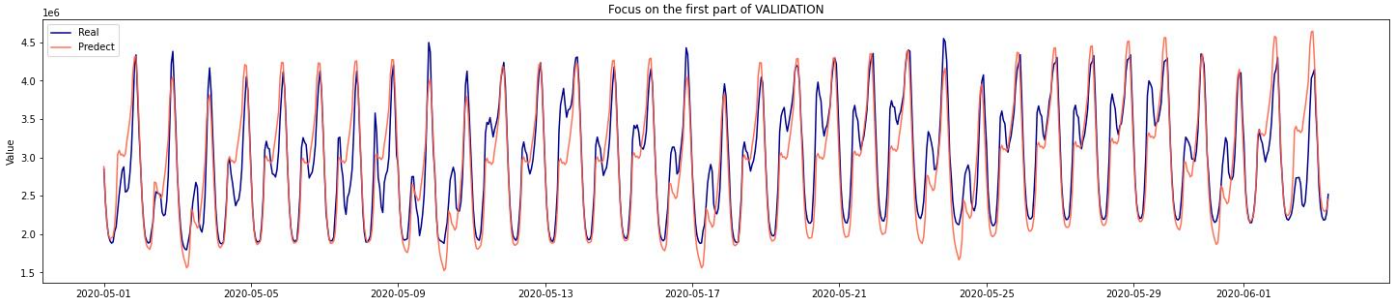

Figura 15 - focus delle previsioni sulla prima parte del validetion set

SARIMAX Results

Dep. Variable:	Value	No. Observations:	14592
Model:	SARIMAX(2, 0, 2)x(1, 1, [1], 24)	Log Likelihood	-428630883.789
Date:	Wed, 13 Jan 2021	AIC	857261841.578
Time:	15:23:01	BIC	857262122.213
Sample:	09-01-2018 - 04-30-2020	HQIC	857261934.831
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
sin(1,168)	-1.94e+05	2.36e-16	-8.23e+20	0.000	-1.94e+05	-1.94e+05
cos(1,168)	-6.468e+04	5.19e-18	-1.25e+22	0.000	-6.47e+04	-6.47e+04
sin(2,168)	-1.063e+05	1.07e-18	-9.98e+22	0.000	-1.06e+05	-1.06e+05
cos(2,168)	1.055e+05	1.98e-19	5.32e+23	0.000	1.06e+05	1.06e+05
sin(3,168)	3.648e+04	3.04e-19	1.2e+23	0.000	3.65e+04	3.65e+04
cos(3,168)	5.221e+04	1.3e-18	4.01e+22	0.000	5.22e+04	5.22e+04
sin(4,168)	9407.1759	1.31e-18	7.2e+21	0.000	9407.176	9407.176
cos(4,168)	-5.327e+04	3.21e-19	-1.66e+23	0.000	-5.33e+04	-5.33e+04
sin(5,168)	-9.2e+04	2.67e-19	-3.45e+23	0.000	-9.2e+04	-9.2e+04
cos(5,168)	-1.736e+04	1.1e-18	-1.58e+22	0.000	-1.74e+04	-1.74e+04
sin(6,168)	-5.415e+04	6.65e-19	-8.14e+22	0.000	-5.42e+04	-5.42e+04
cos(6,168)	8.168e+04	1.67e-19	4.9e+23	0.000	8.17e+04	8.17e+04
sin(7,168)	-1.183e+15	8.39e-27	-1.41e+41	0.000	-1.18e+15	-1.18e+15
cos(7,168)	-1.548e+16	7.33e-27	-2.11e+42	0.000	-1.55e+16	-1.55e+16
sin(8,168)	4.771e+04	7.88e-19	6.05e+22	0.000	4.77e+04	4.77e+04
cos(8,168)	6808.1540	2.75e-19	2.48e+22	0.000	6808.154	6808.154
sin(9,168)	2.24e+04	6.27e-19	3.57e+22	0.000	2.24e+04	2.24e+04
cos(9,168)	-2.252e+04	1.49e-18	-1.51e+22	0.000	-2.25e+04	-2.25e+04
sin(10,168)	-3166.2802	1.9e-18	-1.66e+21	0.000	-3166.280	-3166.280
cos(10,168)	-1.039e+04	9.22e-19	-1.13e+22	0.000	-1.04e+04	-1.04e+04
sin(1,8766)	2.835e+05	1.24e-20	2.28e+25	0.000	2.83e+05	2.83e+05
cos(1,8766)	1.973e+04	3.99e-22	4.95e+25	0.000	1.97e+04	1.97e+04
sin(2,8766)	-4.901e+05	2.48e-20	-1.98e+25	0.000	-4.9e+05	-4.9e+05
cos(2,8766)	-6.373e+04	1.59e-21	-4e+25	0.000	-6.37e+04	-6.37e+04
sin(3,8766)	-1.097e+05	3.71e-20	-2.96e+24	0.000	-1.1e+05	-1.1e+05
cos(3,8766)	-1.397e+05	3.58e-21	-3.9e+25	0.000	-1.4e+05	-1.4e+05
sin(4,8766)	4.719e+04	4.93e-20	9.58e+23	0.000	4.72e+04	4.72e+04
cos(4,8766)	-3.343e+04	6.36e-21	-5.26e+24	0.000	-3.34e+04	-3.34e+04
sin(5,8766)	8.399e+04	6.13e-20	1.37e+24	0.000	8.4e+04	8.4e+04
cos(5,8766)	5.417e+04	9.92e-21	5.46e+24	0.000	5.42e+04	5.42e+04
ar.L1	1.1448	3.86e-10	2.97e+09	0.000	1.145	1.145
ar.L2	-0.3207	5.55e-10	-5.78e+08	0.000	-0.321	-0.321
ma.L1	0.0199	1.5e-08	1.33e+06	0.000	0.020	0.020
ma.L2	0.1319	3.58e-09	3.68e+07	0.000	0.132	0.132
ar.S.L24	-0.0045	1.66e-10	-2.7e+07	0.000	-0.004	-0.004
ma.S.L24	0.0045	1.67e-10	2.68e+07	0.000	0.004	0.004
sigma2	2.041e+10	3.34e-21	6.12e+30	0.000	2.04e+10	2.04e+10

Figura 16 – coefficienti del modello SARIMAX

Tutte le variabili utilizzate sono significative. Forse è rimasta ancora dell'autocorrelazione al lag 1. La distribuzione è normale a parte per la presenza di alcuni outliers che avevo già individuato inizialmente. Sarebbe opportuno aggiungere delle variabili dummy per modellare tali osservazioni.

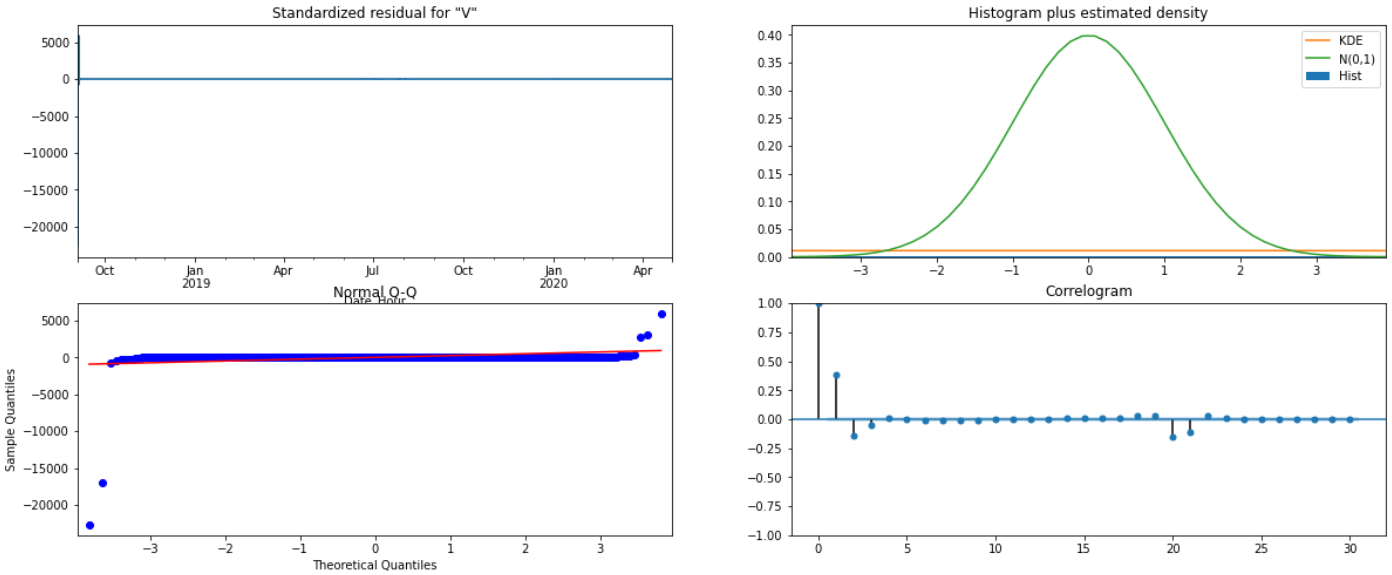

Figura 17 – analisi dei residui del modello

Come modello finale per fare le previsioni, utilizzerò quindi quest'ultimo, un SARIMAX $(2,0,2)(1,1,1)_{24}$ con 10 armoniche per la stagionalità settimanale, 5 per quella annuale e senza la variabile step covid.

3.2 UCM

Come seconda tipologia di modelli riguardanti la sezione dei modelli lineari, ho implementato un modello UCM.
Includo le componenti level-trend e stagionalità mentre non inserisco la variabile ciclo.
Per determinare il miglior tipo di level-trend ho usato un approccio grid search, i cui risultati sono riportati in Tabella 3. I modelli migliori risultano quelli che presentano un level-trend del tipo random walk, local level, local linear deterministic trend e random walk with drift. I modelli con level-trend del tipo dconstant e ntrend sono dieci volte peggiori in termini di MAE sul validation. La stagionalità giornaliera è rappresentata tramite variabili dummy, quella settimanale tramite 15 serie di Fourier.

level-trend	MAE Train	MAE Validation
rwalk	102649	752062
dconstant	1023576	3645172
ntrend	1023536	3645233
llevel	103983	752009
lldtrend	103958	742714
rwdrift	102623	743459
lltrend	105020	13418369
strend	105031	13418508
rtrend	103952	13096502

Tabella 3 – performance del modello UCM al variare della tipologia di level-trend

Ho provato anche ad aggiungere la variabile covid ma non ho ottenuto un miglioramento significativo.
Come modello migliore, che utilizzerò anche per fare le previsioni, ho scelto un modello con level-trend di tipo random walk with drift. Ottengo le seguenti performance: MAE train: 102623.3, MAE validation: 743458.6

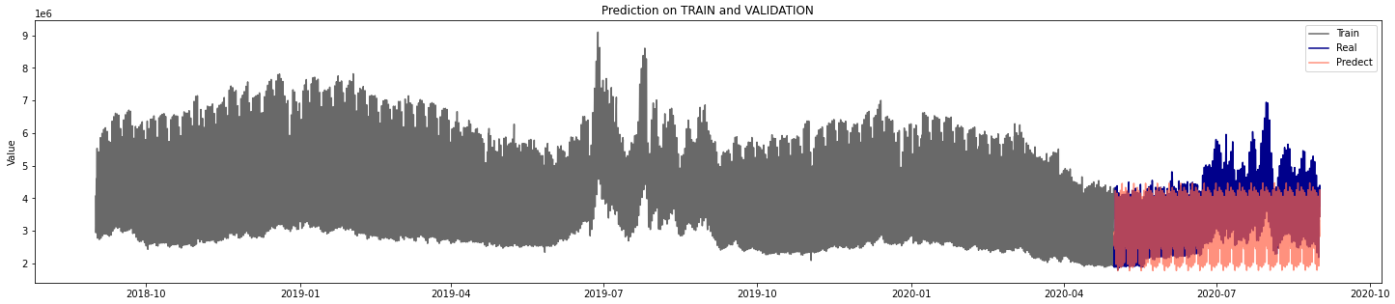

Figura 18 - previsioni sul validation set tramite modello UCM

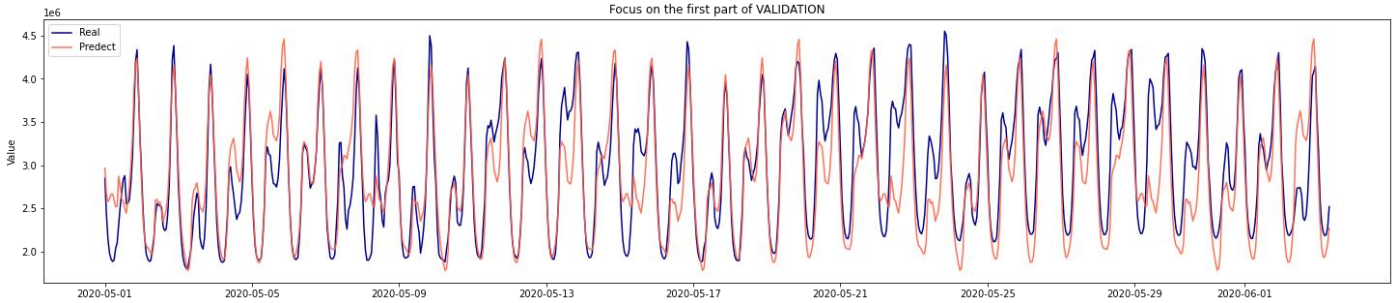

Figura 19 - focus delle previsioni sulla prima parte del validetion set

Le performance sono peggiori rispetto a quelle ottenute col modello SARIMAX e, come si nota in Figura 16 e in Figura 17 il modello coglie la stagionalità giornaliera e settimanale ma non quella annuale. Sembrerebbe opportuno aggiungere delle sinusoidi per rappresentare tale stagionalità. Tale modello però richiede più RAM di quella disponibile in Colab, ambiente dove sto facendo girare il mio codice, mandando in crash il sistema.

Infine, ho provato ad analizzare i disturbance smoother per individuare eventuali cambi repentini nel level-trend, eventualmente collegati al fattore covid. Non si notano particolari anomalie, se non nel periodo di luglio-agosto 2019, fenomeno che avevamo già evidenziato nelle visualizzazioni iniziali.

3.3 TBATS

Tra i modelli lineari che permettono di modellare una stagionalità multipla, c'è anche un metodo chiamato TBATS [1]. Il nome è acronimo per le caratteristiche chiave del modello: stagionalità trigonometrica, trasformazione Box-Cox, errori ARMA, componenti trend e stagionalità. Ogni stagionalità è modellata da una rappresentazione trigonometrica basata sulle serie di Fourier. Il modello migliore che ho trovato, in termini di MAE sul train e validation set, è quello con la trasformazione Box-Cox, modellazione degli errori tramite processo ARMA e una stagionalità multipla con periodo 24, 168 e 8766 ore. Ottengo le seguenti performance: MAE train: 112372.4, MAE validation: 666606.0

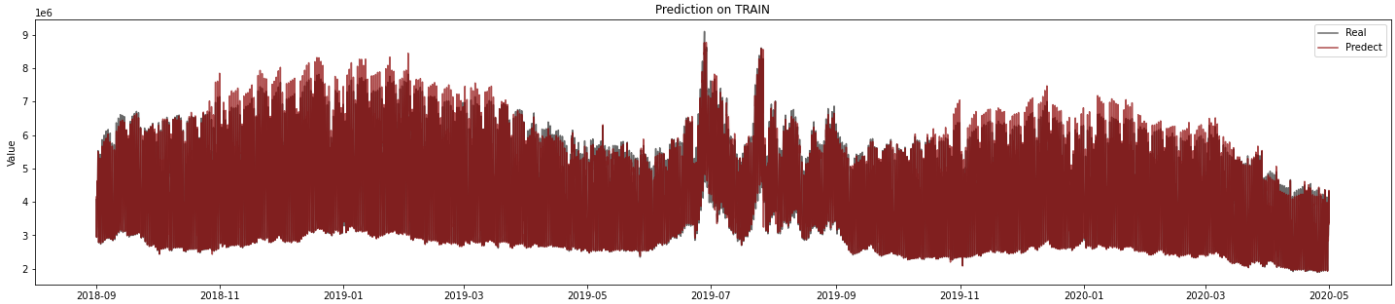

Figura 20 - previsioni sul train set tramite modello TBATS

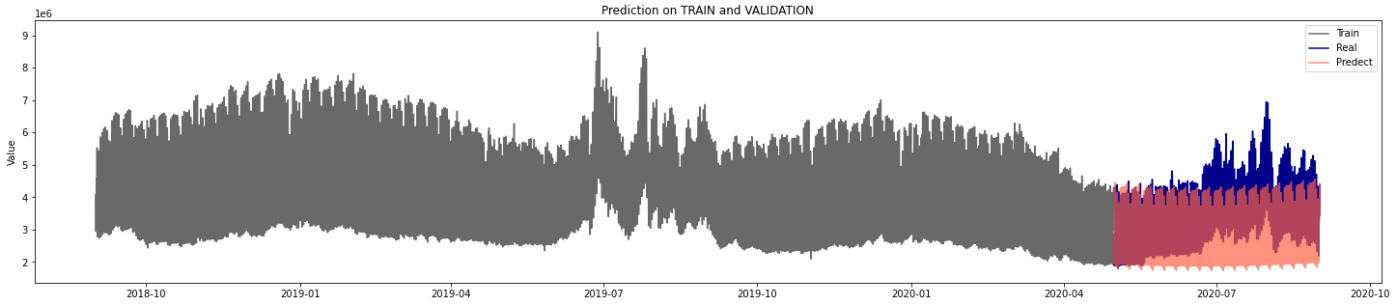

Figura 21 - previsioni sul validation set tramite modello TBATS

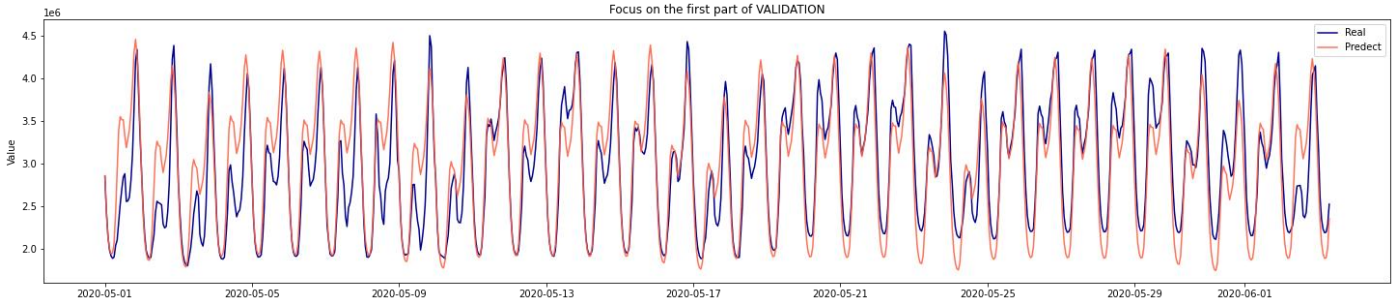

Figura 22 - focus delle previsioni sulla prima parte del validetion set

3.4 PROPHET

Come ultimo modello lineare, costruisco il modello prophet. Prophet è una libreria open source per previsioni di serie temporali univariate sviluppata da Facebook. Prophet implementa un modello additivo che supporta trend, stagionalità e festività ed è progettato per essere facile e completamente automatico. Per ulteriori info [1] [2]
Il modello migliore trovato include una stagionalità giornaliera, settimanale e annuale. Utilizza 10 armoniche per rappresentare la stagionalità annuale, 3 per quella settimanale e 4 per quella giornaliera.
Ottengo le seguenti performance: MAE train: 366467.4, MAE validation: 545685.0

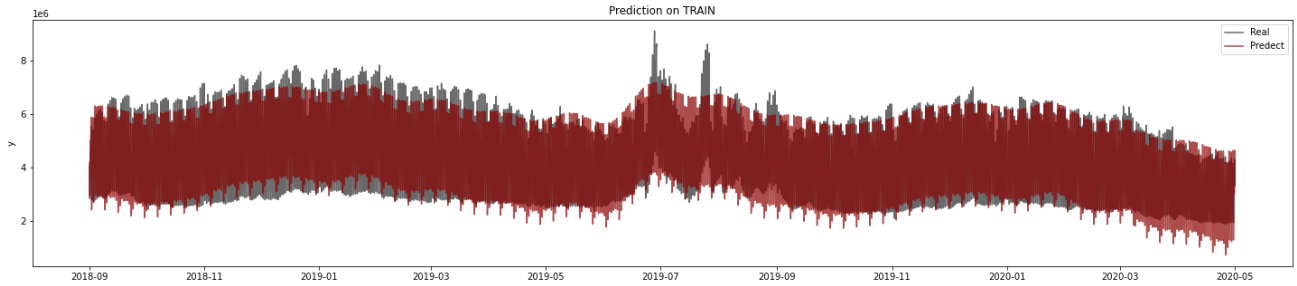

Figura 23 - previsioni sul train set tramite modello prophet

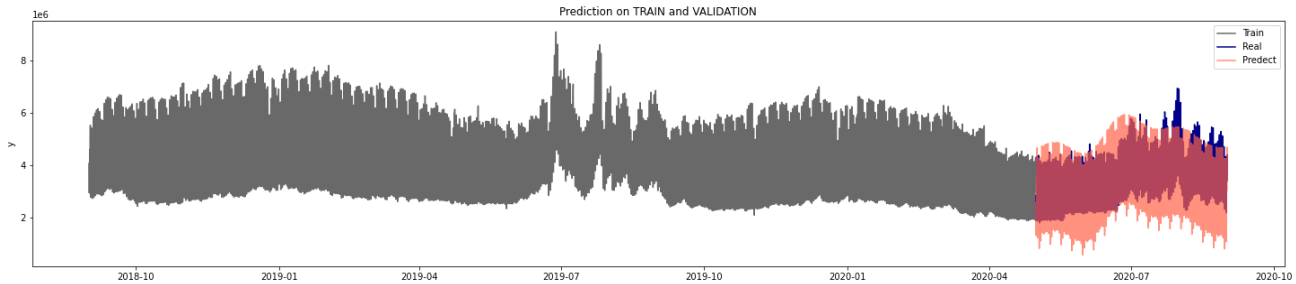

Figura 24 - previsioni sul validation set tramite modello prophet

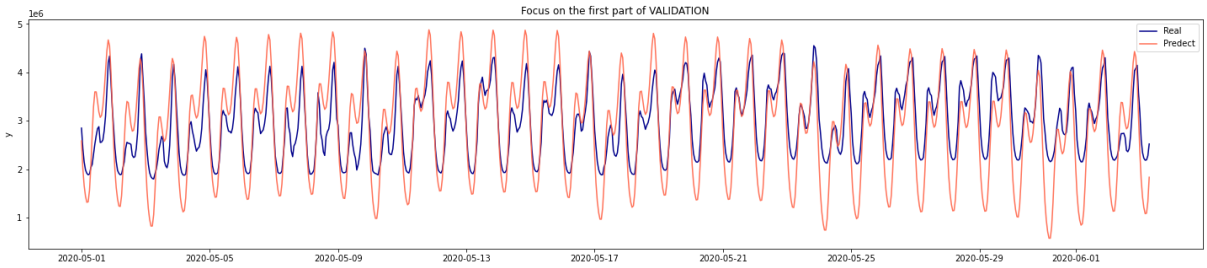

Figura 25 - -focus delle previsioni sulla prima parte del validetion set

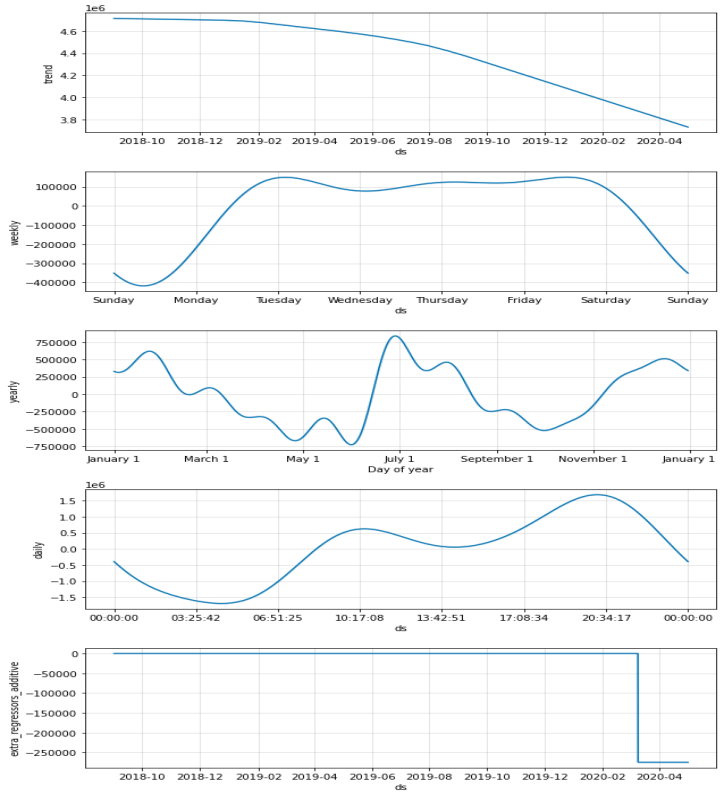

Figura 26 – grafici con le componenti del modello

3.4.1 TUNING DEGLI IPERPARAMETRI

Il modello Prophet ha una serie di iperparametri di input che si possono ottimizzare. Utilizzo la cross-validazione sull'intero dataset per ottimizzarli. I valori che portano a un modello più performante sono: changepoint_prior_scale = 0.01 e seasonality_prior_scale = 0.01.

changepoint_prior_scale	seasonality_prior_scale	MAE
0.001	0.01	616122
0.001	0.1	618625
0.001	1	618714
0.001	10	620077
0.01	0.01	596144
0.01	0.1	610750
0.01	1	612753
0.01	10	606087
0.1	0.01	631724
0.1	0.1	623333
0.1	1	839121
0.1	10	975638
0.5	0.01	988780
0.5	0.1	2952022
0.5	1	3787410
0.5	10	3836528

Tabella 4 – performance del modello prophet al variare degli iperparametri

Utilizzerò tali valori per fare le previsioni.

3.5 LSTM

Per comodità utilizzerò come validation set gli ultimi 61 giorni, lunghezza pari alla finestra di forecast.
Per fare le previsioni multi-step ho utilizzato tre diverse strategie:

- previsioni one-step-ahead: le previsioni per gli step successivi al primo si ottengono in modo ricorsivo utilizzando le previsioni precedenti come input per quelle future.
- previsioni direct output vector: il modello prevede direttamente, in un unico colpo, i dati sull'intera finestra di previsione.
- previsioni ibride: una via di mezzo tra le due strategie precedenti.

Provo inoltre due implementazioni diverse di LSTM: stateful e stateless. La modalità stateful è utile per avere memoria tra i batch in una epoca di addestramento; se invece non abilito il parametro stateful, l'algoritmo non tiene memoria tra un batch e l'altro.
Poiché gli LSTM sono sensibili alla scala dei dati di input; ho scalato i dati nell'intervallo tra 0 a 1. Ho poi costruito una funzione che permette di dividere i dati in sequenze e di rimodellarli in modo che siano del formato adatto per la rete LSTM.

3.5.1 PREVISIONI ONE-STEP-AHEAD

Inizio costruendo un modello con un solo hidden layer con 100 neuroni. Le previsioni sono buone solo per il primo giorno di forecast. Per finestre temporali più lunghe, come si vede in Figura 28 e Figura 29 , a causa dell'accumularsi dell'errore, le previsioni risultano sfalsate. Provando a cambiare gli iperparametri del modello (il numero di neuroni, il numero di layer, il numero di epoche) non si hanno significativi miglioramenti. Un iperparametro che fortemente condiziona il modello è il numero di osservazioni di input Anche questo fattore però non porta a buone previsioni.

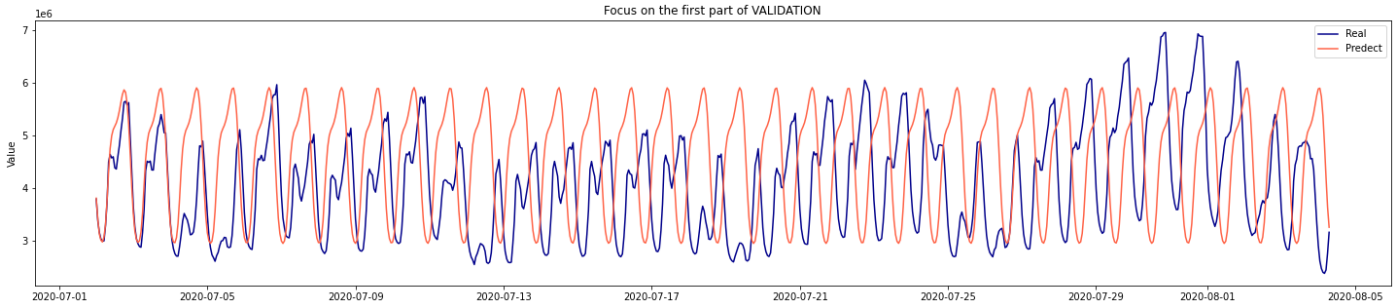

Figura 27 - focus delle previsioni sulla prima parte del validetion set, step di input = 24

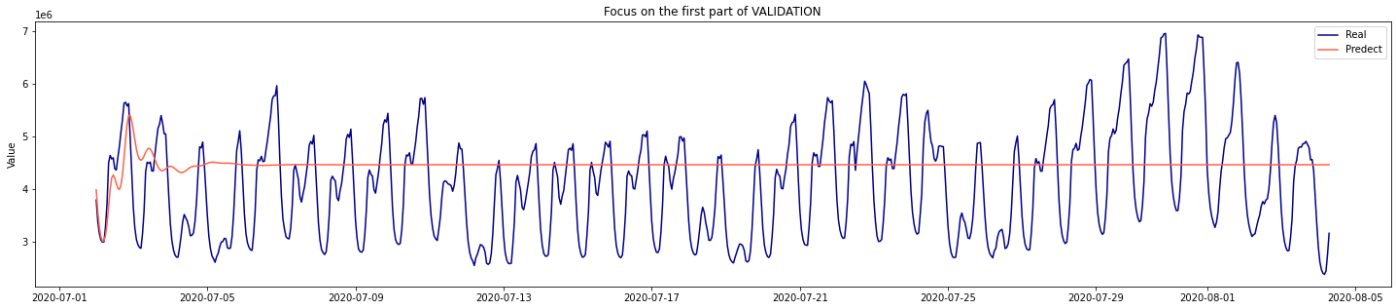

Figura 28 - focus delle previsioni sulla prima parte del validetion set, step di input = 168

3.5.2 PREVISIONI DIRECT OUTPUT VECTOR

Le previsioni ottenute con la strategia direct output sono decisamente migliori rispetto alle precedenti. Il problema di questa metodologia è che i dati relativi agli ultimi 61 giorni del train vengono utilizzati solo come output durante il train.

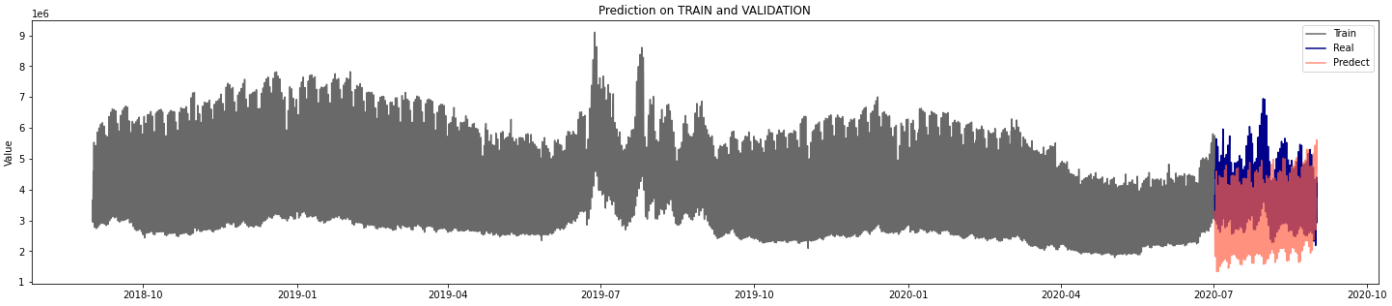

Figura 29 - previsioni sul validation set tramite modello LSTM

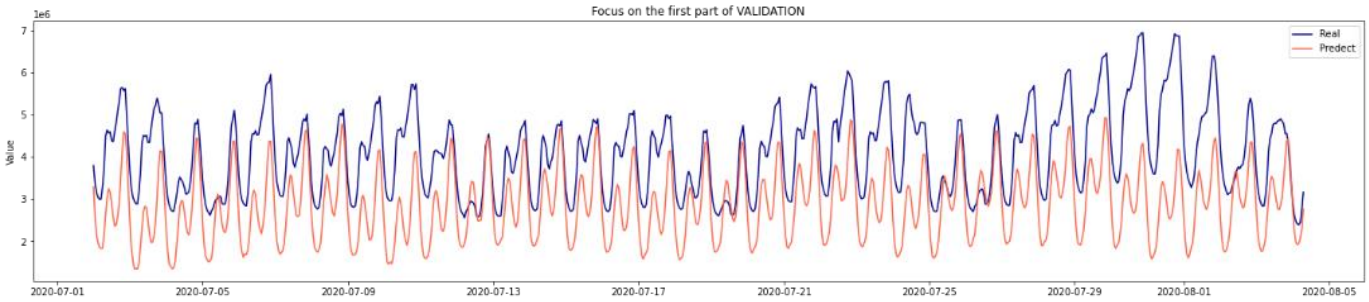

Figura 30 - focus delle previsioni sulla prima parte del validetion set, step di input = 168, implementazione stateful

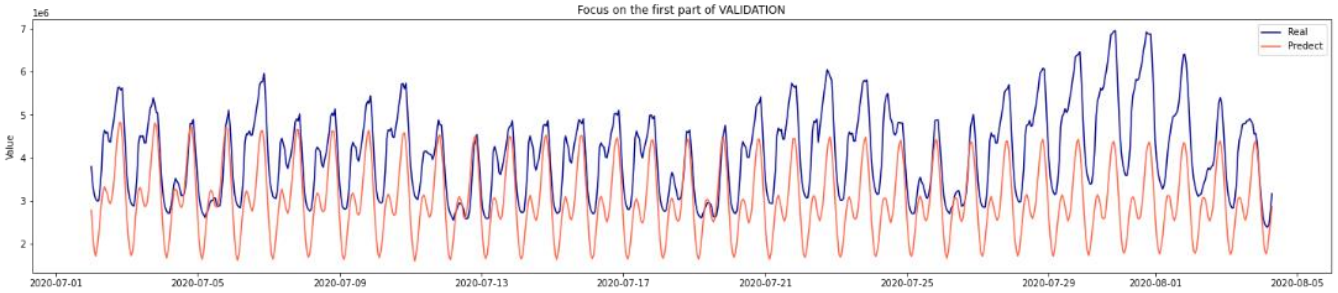

Figura 31 - focus delle previsioni sulla prima parte del validetion set, step di input = 168, implementazione stateless

3.5.3 PREVISIONI IBRIDE

Provo quindi una metodologia ibrida. Questi modelli sono molto instabili: facendo girare il codice più volte si ottengono risultati molto diversi tra loro. L'implementazione stateless è leggermente migliore di quella stateful. Con il modello, la cui architettura è riportata in Figura 32,, ottengo le seguenti performance: MAE train: 1077271.0, MAE validation 634172.1.

```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 100)               40800
_________________________________________________________________
leaky_re_lu_1 (LeakyReLU)    (None, 100)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_1 (Dense)              (None, 744)               75144
=================================================================
Total params: 115,944
Trainable params: 115,944
Non-trainable params: 0
```

Figura 32 – architettura modello LSTM

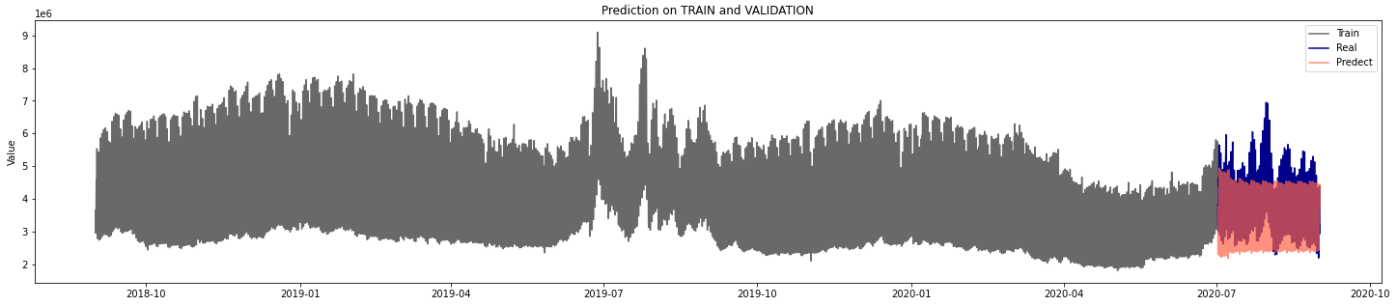

Figura 33 - previsioni sul validation set tramite modello LSTM

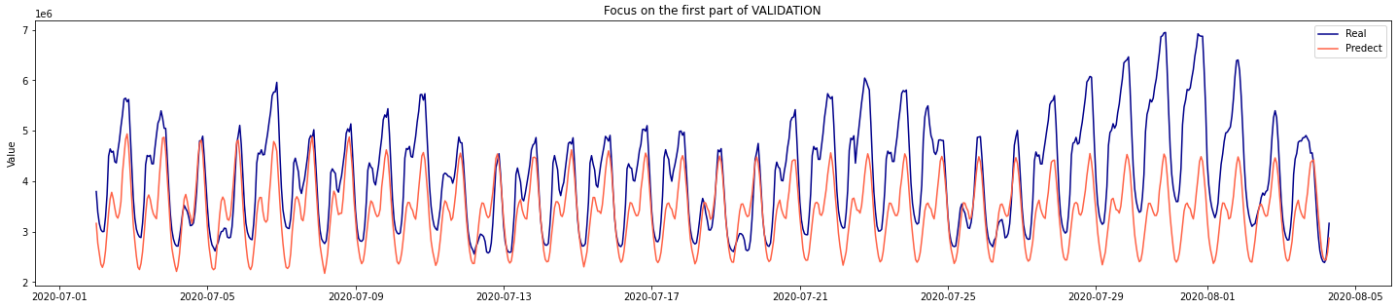

Figura 34 - focus delle previsioni sulla prima parte del validetion set,

I modelli provati non sono eccessivamente performanti. Forse si potrebbero ottenere migliori performance lavorando con la serie resa stazionaria attraverso delle differenze. Oppure si potrebbero applicare degli aggiustamenti stagionali prima della modellazione. Data la multi-stagionalità dei dati potrebbe essere opportuno applicare una CNN, una CNN-LSTM o un modello encoder-decoder. Questo poiché una LSTM lineare spesso si comporta male sui problemi di serie temporali.

4 PREVISIONI

Per ogni tipologia di modello considero quello che ha ottenuto performance migliori in termini di MAE e rieseguo il fit sull'intero dataset. Graficamente ottengo le seguenti previsioni:

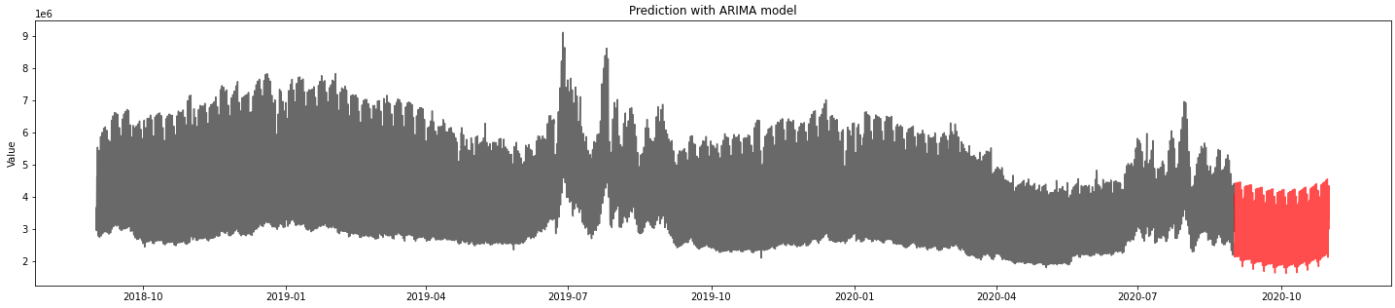

Figura 35 – previsioni con modello SARIMAX

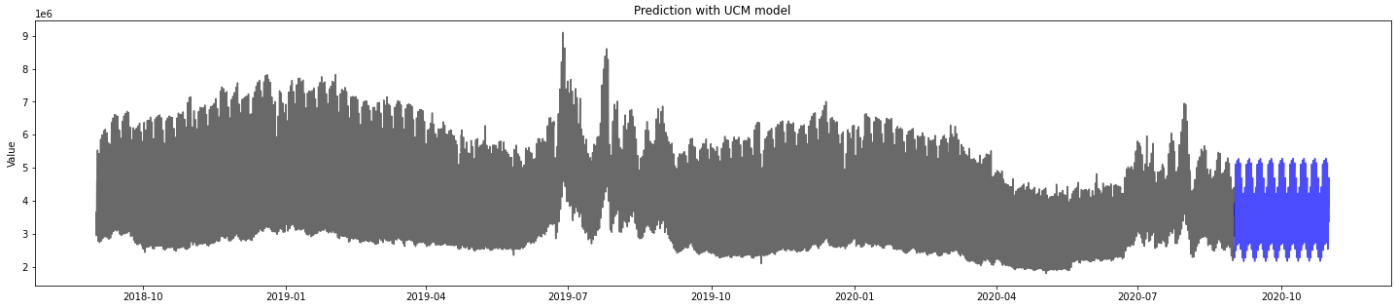

Figura 36 - previsioni con modello UCM

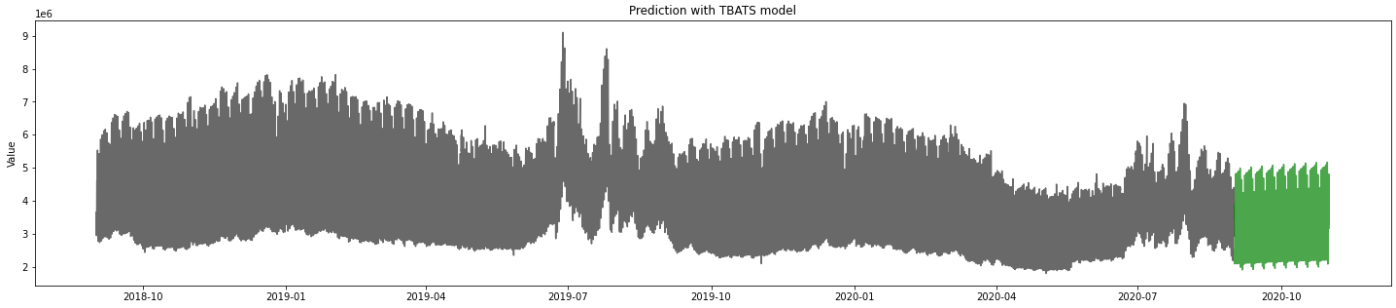

Figura 37 - previsioni con modello TBATS

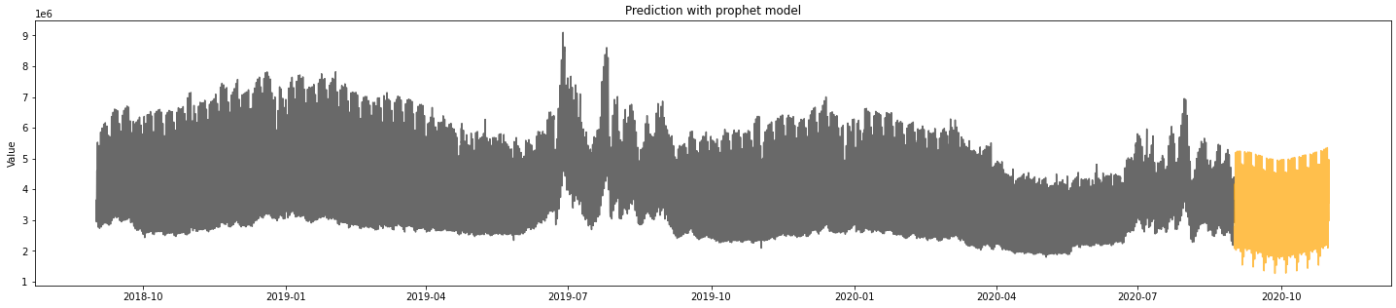

Figura 38 - previsioni con modello PROPHET

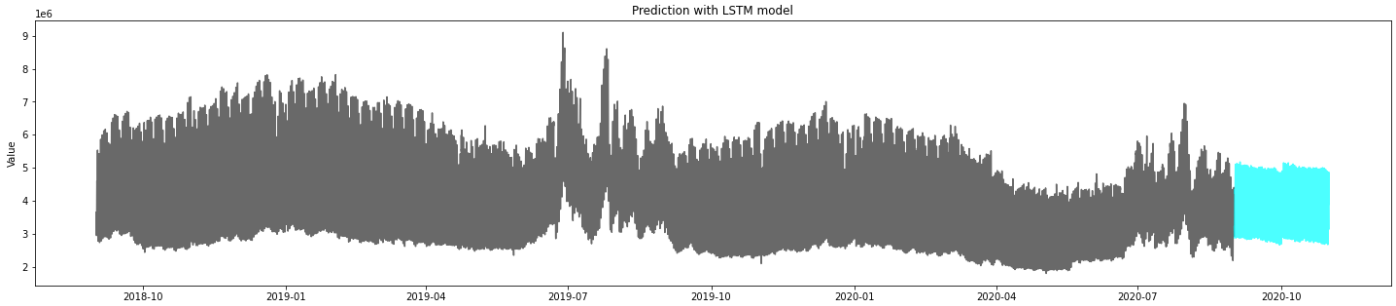

Figura 39 - previsioni con modello PROPHET

5 CONCLUSIONI

Riassumiamo le performance dei modelli in termini di MAE sono:

Modello	MAE Train	MAE Validation
SARIMAX	308301.8	334117.9
UCM	102623.3	743458.6
TBATS	112372.4	666606.0
PROPHET	366467.4	545685.0
LSTM	1077271.0	634172.1

Tabella 5 – perforance dei vari modelli

Il modello SARIMAX risulta il migliore. Tale modello coglie la stagionalità giornaliera, settimanale, annuale e l'autocorrelazione presente nei dati. Anche le previsioni sul lungo periodo sono buone.
Per ottenere risultati ancora più precisi dovremmo sapere cosa rappresentano i dati. Sarebbe utile anche conoscere l'andamento della serie negli anni precedenti al 2018. Con queste informazioni potremmo modellare le anomalie dell'estrae 2019 e il fattore covid in modo più performante. Come lavoro futuro, i modelli potrebbero essere implementati in un ambiente più potente e con più RAM a disposizione. In tal modo si potrebbero implementare dei metodi per ottimizzare gli iperparametri che altrimenti risultano eccessivamente lenti.

6 BIBLIOGRAFIA

http://proceedings.mlr.press/v37/jozefowicz15.pdf

https://www.sciencedirect.com/science/article/pii/S2211467X20300778

https://www.researchgate.net/publication/341609757_Comparative_Analysis_of_Recurrent_Neural_Network_Architectures_for_Reservoir_Inflow_Forecasting

Materiale del corso di Streaming Data Management and Time Series Analysis

Deep Learning for Time Series Forecasting - jasPredict the Future with MLPs, CNNs and LSTMs in Python, Jason Brownlee