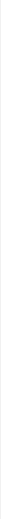


In [49]:

```
1 from numpy import array
2 from keras.models import Sequential
3 from keras.layers import LSTM
4 from keras.layers import Dense
5 import pandas as pd
6 pd.set_option('display.max_columns', 50)
7 pd.set_option('display.max_rows', 550)
8 #pd.set_option('precision', 0)
9 from matplotlib import pyplot
10 from datetime import datetime
11
12 #zaciągnięcie danych
13 dane = pd.read_csv('C:/Users/Laptop/Desktop/Grypa/Dane/total.csv', index_col='data')
14 ucz_s12_s18 = pd.DataFrame
15 test_19 = pd.DataFrame
16 ucz_s12_s18 = dane[(dane.index > '2012-08-23' ) & (dane.index < '2018-03-16' )]
17 test_19 = dane[(dane.index > '2018-04-23') & (dane.index < '2020-05-01')]
18 lista_test = test_19['Total'].tolist()
19 lista_ucz = ucz_s12_s18['Total'].tolist()
20
21 #print(len(lista_ucz), len(lista_test))
22
23 sekwencja = lista_ucz
24 liczba_in = liczba_wejść
25 liczba_out = liczba_wyjść
26
27 #formatowanie danych
28 def podzial_sekwencji(sekwencja, liczba_in, liczba_out):
29     X, y = list(), list()
30     for i in range(len(sekwencja)):
31         end_ix = i + liczba_in
32         out_end_ix = end_ix + liczba_out
33         if out_end_ix > len(sekwencja):
34             break
35         seq_x, seq_y = sekwencja[i:end_ix], sekwencja[end_ix:out_end_ix]
36         X.append(seq_x)
37         y.append(seq_y)
38     return array(X), array(y)
39
40 sekwencja_valid = lista_test
41 liczba_in_valid = liczba_wejść
42 liczba_out_valid = liczba_wyjść
43
44 def podzial_sekwencji_valid(sekwencja_valid, liczba_in_valid, liczba_out_valid):
45     X_valid, y_valid = list(), list()
46     for i in range(len(sekwencja_valid)):
47         end_ix_valid = i + liczba_in_valid
48         out_end_ix_valid = end_ix_valid + liczba_out_valid
49         if out_end_ix_valid > len(sekwencja_valid):
50             break
51         seq_x_valid, seq_y_valid = sekwencja_valid[i:end_ix_valid], sekwencja[end_ix_v
52         X_valid.append(seq_x_valid)
53         y_valid.append(seq_y_valid)
54     return array(X_valid), array(y_valid)
55
56 #wydruk formatowania
57 X, y = podzial_sekwencji(sekwencja, liczba_in, liczba_out)
58 #for i in range(len(X)):
59 #     print(X[i], y[i])
```

```
1 X_valid, y_valid = podzial_sekwencji_valid(sekwencja_valid, liczba_in_valid, liczba_out_valid)
2 for i in range(len(X_valid)):
3     print(X_valid[i], y_valid[i])
4
```



In [29]:

```
1 lista_test =
```

Out[29]:

```
[206028,  
187665,  
126320,  
103570,  
89880,  
81043,  
50979,  
59938,  
48789,  
51251,  
55843,  
53833,  
46153,  
45217,  
42697,  
36178,  
32440,  
34364,  
24946,  
21560,  
22428,  
32946,  
33942,  
54848,  
73468,  
81860,  
91078,  
106831,  
95948,  
106851,  
96062,  
95589,  
99355,  
125794,  
117370,  
125159,  
124773,  
104905,  
115901,  
142768,  
158756,  
227951,  
236454,  
250131,  
199472,  
158698,  
155213,  
157867,  
133758,  
119224,  
102173,  
113081,  
68094,  
80861,  
45834,
```

```
59157,  
58583,  
77873,  
60809,  
49088,  
40785,  
39564,  
37794,  
36789,  
31368,  
35254,  
27541,  
23819,  
25164,  
28457,  
29553,  
54216,  
87589,  
115689,  
111154,  
117374,  
112264,  
129806,  
93602,  
110629,  
106546,  
112575,  
115623,  
124181,  
130460,  
138171,  
89211,  
122430,  
127696,  
204726,  
193482,  
200194,  
211983,  
215994,  
211572,  
158677]
```

In [46]:

```
1 lista_test = [  
2 126320,  
3 103570,  
4 89880,  
5 81043,  
6 50979,  
7 59938,  
8 48789,  
9 51251,  
10 55843,  
11 53833,  
12 46153,  
13 45217,  
14 42697,  
15 36178,  
16 32440,  
17 34364,  
18 24946,  
19 21560,  
20 22428,  
21 32946,  
22 33942,  
23 54848,  
24 73468,  
25 81860,  
26 91078,  
27 106831,  
28 95948,  
29 106851,  
30 96062,  
31 95589,  
32 99355,  
33 125794,  
34 117370,  
35 125159,  
36 124773,  
37 104905,  
38 115901,  
39 142768,  
40 158756,  
41 227951,  
42 236454,  
43 250131,  
44 199472,  
45 158698,  
46 155213,  
47 157867,  
48 133758,  
49 119224,  
50 102173,  
51 113081,  
52 68094,  
53 80861,  
54 45834,  
55 59157,  
56 58583,  
57 77873,  
58 60809,  
59 49088,
```

```
60 40785,  
61 39564,  
62 37794,  
63 36789,  
64 31368,  
65 35254,  
66 27541,  
67 23819,  
68 25164,  
69 28457,  
70 29553,  
71 54216,  
72 87589,  
73 115689,  
74 111154,  
75 117374,  
76 112264,  
77 129806,  
78 93602,  
79 110629,  
80 106546,  
81 112575,  
82 115623,  
83 124181,  
84 130460,  
85 138171,  
86 89211,  
87 122430,  
88 127696,  
89 204726,  
90 193482,  
91 200194,  
92 211983,  
93 215994,  
94 211572,  
95 158677,  
96 109281,  
97 102272, ]
```

In [35]:

```
1 len(lista_test_2)
```

Out[35]:

96

In [33]:

```
1 len(lista_test)
```

Out[33]:

96

In [11]:

```

1 X, y = podzial_sekwencji(sekwencja, liczba_in, liczba_out)
2 for i in range(len(X)):
3     print(X[i], y[i])

```

88 60656]

27864	34232	45357	43668	56433	46712	50854	47865	58867	58130
61968	62147	78779	134358	213906	209903	219511	150439	135150	109313
77151	87372	100518	101205	91358	75947	81863	60122	44866	30305
33372	28421	33776	31103	28628	28363	26616	22480	24834	18950
18614	12487	10904	11055	15220	16193	29241	39036	60162	62488 606

56 59200]

34232	45357	43668	56433	46712	50854	47865	58867	58130	61968
62147	78779	134358	213906	209903	219511	150439	135150	109313	77151
87372	100518	101205	91358	75947	81863	60122	44866	30305	33372
28421	33776	31103	28628	28363	26616	22480	24834	18950	18614
12487	10904	11055	15220	16193	29241	39036	60162	62488	60656 592

00 63941]

45357	43668	56433	46712	50854	47865	58867	58130	61968	62147
78779	134358	213906	209903	219511	150439	135150	109313	77151	87372
100518	101205	91358	75947	81863	60122	44866	30305	33372	28421
33776	31103	28628	28363	26616	22480	24834	18950	18614	12487
10904	11055	15220	16193	29241	39036	60162	62488	60656	59200 639

41 52107]

43668	56433	46712	50854	47865	58867	58130	61968	62147	78779
-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

In [4]:

```
1 len(lista_ucz), len(lista_test)
```

Out[4]:

(266, 96)

In [25]:

```
1 len(zbior)
```

Out[25]:

24

In [6]:

```

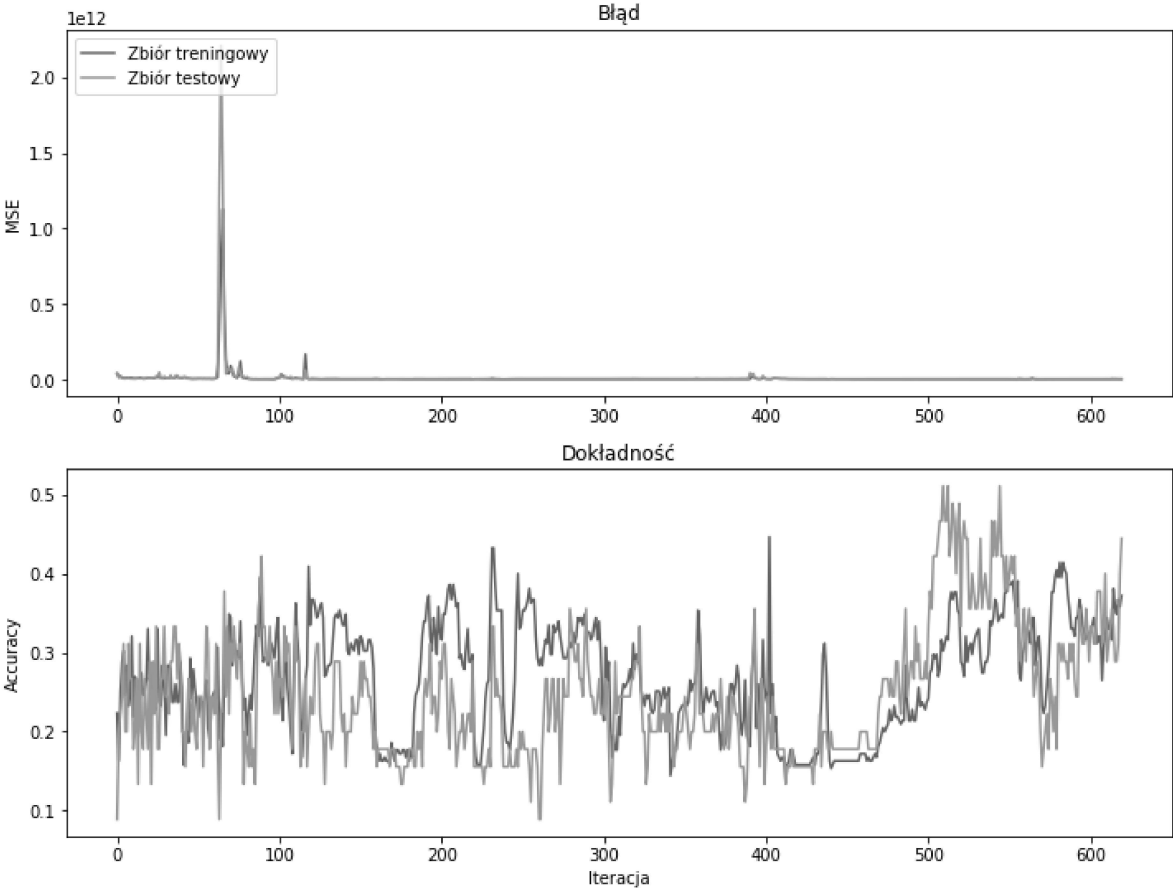
1 #robocze
2 iteracje = 430
3 liczba_wejść = 48
4 liczba_wyjść = 4
5

```


In [19]:

```
1  iteracje =620
2  liczba_wejść = 48
3  liczba_wyjść = 4
4
5  zb_test = len(lista_test)
6  zb_test = str(zb_test)
7  iteracje_str = str(iteracje)
8  liczba_wejść_str = str(liczba_wejść)
9  liczba_wejść_str
10 liczba_wyjść_str = str(liczba_wyjść)
11 now = datetime.now()
12 teraz = now.strftime("%H_%M_%S")
13
14 #model
15 n_features = 1
16 X = X.reshape((X.shape[0], X.shape[1], n_features))
17 X_valid = X_valid.reshape((X_valid.shape[0], X_valid.shape[1], n_features))
18 model = Sequential()
19 model.add(LSTM(100, activation='relu', return_sequences=True, input_shape = (liczba_in
20 model.add(LSTM(100, activation='relu'))
21 model.add(Dense(liczba_out))
22 model.compile(optimizer='adam', loss='mse', metrics=['accuracy'])
23 history = model.fit(X, y, epochs=iteracje, verbose=0, validation_data=(X_valid, y_vali
24 model.summary
25 print(teraz+' GOTOWE: ' + zb_test + '_' + iteracje_str + '_' + liczba_wejść_str + '_' +
26
27
28 #wykres
29 f = pyplot.figure(figsize=(12, 9))
30 pyplot.subplot(211)
31 pyplot.plot(history.history['loss'])
32 pyplot.plot(history.history['val_loss'])
33 #pyplot.xlabel('Iteracja')
34 pyplot.ylabel('MSE')
35 pyplot.legend(['Zbiór treningowy', 'Zbiór testowy'], loc='upper left')
36 pyplot.title('Błąd')
37 pyplot.subplot(212)
38 pyplot.plot(history.history['accuracy'])
39 pyplot.plot(history.history['val_accuracy'])
40 pyplot.xlabel('Iteracja')
41 pyplot.ylabel('Accuracy')
42 #pyplot.legend(['Zbiór treningowy', 'Zbiór testowy'], loc='upper right')
43 pyplot.title('Dokładność')
44 #zapisanie rysunku i modelu
45 f.savefig('C:/Users/Laptop/Desktop/Grypa/Obrázky/LSTM/' + teraz + '_LSTM_' + zb_test + '_'
46 model.save('C:/Users/Laptop/Desktop/Grypa/Modele/' + teraz + '_LSTM_' + zb_test + '_' +
47
```

16_37_21 GOTOWE: 96_620_48_4



In [51]:

```

1  #testowanie
2  wynik_zbioru_testowego = []
3  Tabela_MSE = pd.DataFrame()
4  Tabela_MSE['Model'] = ""
5  Tabela_MSE['MSE'] = ""
6  for x in zbior:
7      x_input = array(x)
8      x_input = x_input.reshape((1, liczba_in, n_features))
9      wynik = model.predict(x_input, verbose=0)
10     #print(wynik)
11     wynik_flat = wynik.flatten()
12     wynik_list = wynik_flat.tolist()
13     wynik_zbioru_testowego.extend(wynik_list)
14     print('Wszystkie dodatnie:', all(n>0 for n in wynik_list) )
15     wynik_zbioru_testowego = [round(x) for x in wynik_zbioru_testowego]
16
17     #wynik_zbioru_testowego
18     tab_wyn = pd.DataFrame()
19     tab_wyn['Oczekiwane'] = test_19['Total']
20     nazwa_kolumny = zb_test + '_' + iteracje_str + '_' + liczba_wejść_str + '_' + liczba_wyjść_str
21     tab_wyn[nazwa_kolumny] = wynik_zbioru_testowego
22     tab_wyn['error'] = tab_wyn['Oczekiwane'] - tab_wyn[nazwa_kolumny]
23     tab_wyn['error^2'] = tab_wyn['error']**2
24     MSE = sum(tab_wyn['error^2'])/96
25     MSE = round(MSE)
26     nazwa_modelu = teraz + '___' + zb_test + '_' + iteracje_str + '_' + liczba_wejść_str + '_' + liczba_wyjść_str
27     wyniki += [nazwa_modelu, MSE]
28     print('MSE = ', f"{MSE:,d}")
29     tab_wyn.to_csv('C:/Users/Laptop/Desktop/Grypa/tabele_wynikowe/' + teraz + nazwa_kolumny + '.csv')

```

Wszystkie dodatnie: True

MSE = 1,920,197,936

In [24]:

1 wyniki

Out[24]:

```

['05_53_06___96_500_48_4',
1088753474,
'05_53_06___96_800_48_4',
6404018607,
'05_53_06___96_630_48_4',
5743437486,
'05_53_06___96_430_48_4',
967682559,
'05_53_06___96_330_48_4',
2498546646,
'05_53_06___96_380_48_4',
2186606684,
'06_20_54___96_440_48_4',
5863894692,
'06_24_19___96_200_48_4',
8607777207,
'06_27_05___96_420_48_4',
1941519090]

```

