НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Курсовий проект

із дисципліни «Алгоритми і системи комп’ютерної математики»

На тему

«Прогнозування кількості хворих на COVID-19»

Етап №6

Виконав:

студент групи КМ-93

Костенко О. А.

Керівник:

доцент

Олефір О. С.

Київ — 2022

Зміст

Опис обраного математичного методу

Підхід авторегресійного інтегрованого ковзного середнього (англ. autoregressive integrated moving average, ARIMA) до часових рядів полягає в тому, що в першу чергу оцінюється стаціонарность ряду. Різними тестами виявляються наявність поодиноких коренів і порядок інтегрованості часового ряду (зазвичай обмежуються першим або другим порядком). Далі, при необхідності, (якщо порядок інтегрованості більше нуля) ряд перетворюється взяттям різниці відповідного порядку і вже для перетвореної моделі будується деяка ARMA-модель, оскільки передбачається, що отриманий процес є стаціонарним, на відміну від вихідного нестаціонарного процесу (разностно-стаціонарного або інтегрованого процесу порядку d).

ARIMA - модель і методологія аналізу часових рядів. Є розширенням моделей ARMA для нестаціонарних часових рядів, які можна зробити стаціонарними взяттям різниць деякого порядку від вихідного часового ряду (так звані інтегровані або різносно-стаціонарні часові ряди). Модель $ARIMA(p,d,q)$ означає, що різниці часового ряду порядку d належать моделі $ARMA(p,q)$.

Модель $ARIMA(p,d,q)$ для нестаціонарного часового ряда Xt має вигляд:,

$$\Delta^d = c + \varepsilon_t + \sum_{i=1}^{p} a_i \Delta^d X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \,,$$

де X_t – це зпрогнозоване значення на період t;

c – константа, зазвичай для спрощення рівняється нулю;

a_i – параметри моделі, коефіцієнти авторегресії;

θ_i – параметри моделі, ковзного середнього;

Δ^d – оператор різниці часового ряду порядку d (послідовне взяття d раз різниць першого порядку - спочатку від тимчасового ряду, потім від отриманих різниць першого порядку, потім від другого порядку і т.д.)

ε_t – білий шум.

Також, дана модель інтерпретується як $ARMA(p+d,q)$ модель з d одиничними розв'язками. При d = 0 маємо звичайну ARMA модель.

ARIMA-моделі дозволяють моделювати інтегровані або разностностаціонарні часові ряди (DS-ряди, diference stationary). Часовий ряд називається інтегрованим порядку k, якщо різниці ряду порядку Δkxt, тобто є стаціонарними, в той час як різниці меншого порядку (включаючи нульового порядку, тобто сам тимчасової ряд) не є стаціонарними щодо деякого тренда рядами (TS-рядами, trend stationary). В зокрема $I(0)$ – це стаціонарний процес. Порядок інтегрованості часового ряду і є порядок d моделі.

Методологія Бокса-Дженкінса. Методологія Бокса-Дженкінса підбору ARIMA моделі для даного ряду спостережень складається з 5 кроків.

Крок 1. Отримання стаціонарного ряду. Ми тестуємо ряд на стаціонарність, використовуючи описані вище методи: візуальний аналіз графіка, візуальний аналіз ACF і PACF, тести на одиничні коріння. Якщо виходить стаціонарний ряд, то переходимо до наступного пункту, якщо немає стаціонарності ряду, то застосовуємо оператор взяття послідовної різниці і повторюємо тестування. Напрактиці послідовна різниця береться, як правило, не більше двох разів.

Крок 2. Після того, як отримано стаціонарний часовий ряд, будуються його вибіркові ACF і PACF, які, як було показано вище, є своєрідними «відбитками пальців» ARMA(p, q) процесу і дозволяють сформулювати кілька гіпотез про можливі порядки авторегресії p і змінного середнього q. Зазвичай рекомендується використовувати моделі можливо більш низького порядку, як правило, з p q 3 (якщо немає сезонної компоненти).

Крок 3. Для кожної з обраних на першому етапі моделей оцінюються їх параметри і обчислюються залишки.

Крок 4. Кожна з моделей перевіряється, наскільки вона відповідає даним. З моделей, адекватних даними, вибирається найпростіша модель, тобто з найменшою кількістю параметрів.

Крок 5. Прогнозування. Після того, як обрана модель, можна будувати прогноз на один або кілька кроків за часом і оцінювати довірчі інтвервали прогнозованих значень.

Модель $SARIMA(p{,}d{,}q{,}$ P, D, Q) - seasonal autoregressive integrated moving average, дуже схожа на модель $ARIMA(p{,}d{,}q)$, за винятком того, що вона бере до уваги сезонність даних, з чого випливає додатковий набор компонентів авторегресії та ковзного середнього, що компенсуються частотою сезонності:

$$\Delta^d = c + \varepsilon_t + \sum_{i=1}^{p} a_i \Delta^d X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \sum_{i=1}^{P} \boldsymbol{\varphi_i \Delta^d X_{t-si}} + \sum_{i=1}^{Q} \boldsymbol{\eta_i \varepsilon_{t-si}}$$

Модель $SARIMA$X($p{,}d{,}q{,}$ r, P, D, Q), окрім сезонності, також враховує й екзогенні змінні, іншими словами, використовує зовнішні данні в прогнозі:

$$\Delta^d = c + \varepsilon_t + \sum_{i=1}^{p} a_i \Delta^d X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \sum_{i=1}^{P} \varphi_i \Delta^d X_{t-si} + \sum_{i=1}^{Q} \eta_i \varepsilon_{t-si} + \sum_{i=1}^{r} \boldsymbol{\beta_i \epsilon_{i_t}}$$

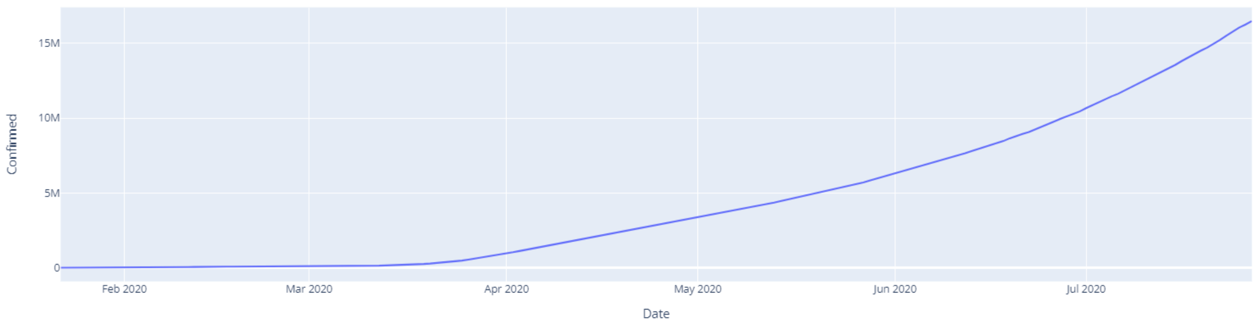
Контрольний приклад

Завантажимо дані про COVID-19 з файлу .csv. Залишивши в ньому тільки інформацію про дату та кількість хворих, отримуємо часовий ряд кількості хворих на COVID-19.

Розглянемо вхідні дані. Перші 10 записів файлу:

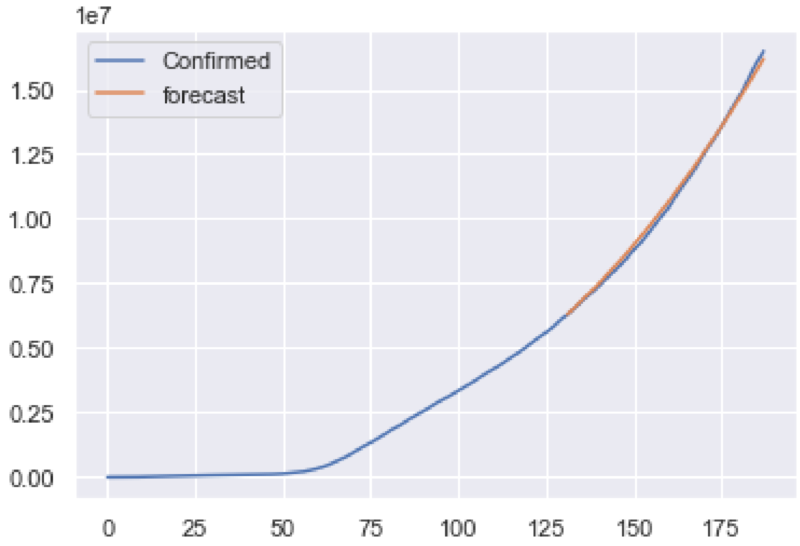
	Date	Confirmed
0	2020-01-22	555
1	2020-01-23	654
2	2020-01-24	941
3	2020-01-25	1434
4	2020-01-26	2118
5	2020-01-27	2927
6	2020-01-28	5578
7	2020-01-29	6166
8	2020-01-30	8234
9	2020-01-31	9927

Побудова завантажених даних:



Для прогнозування кількості хворих на COVID-19 використаємо бібліотеку *statsmodels*, а саме модель *SARIMAX*. Розділимо дані на тренувальну та тестову вибірки з відношенням 70:30 та навчимо модель на тренувальних даних.

Результат прогноз моделі з урахуванням тестової вибірки:



Архітектура програмних засобів

Для забезпечення можливості прогнозування кількості хворих на COVID-19, була написана програма мовою Python, з використанням бібліотек:

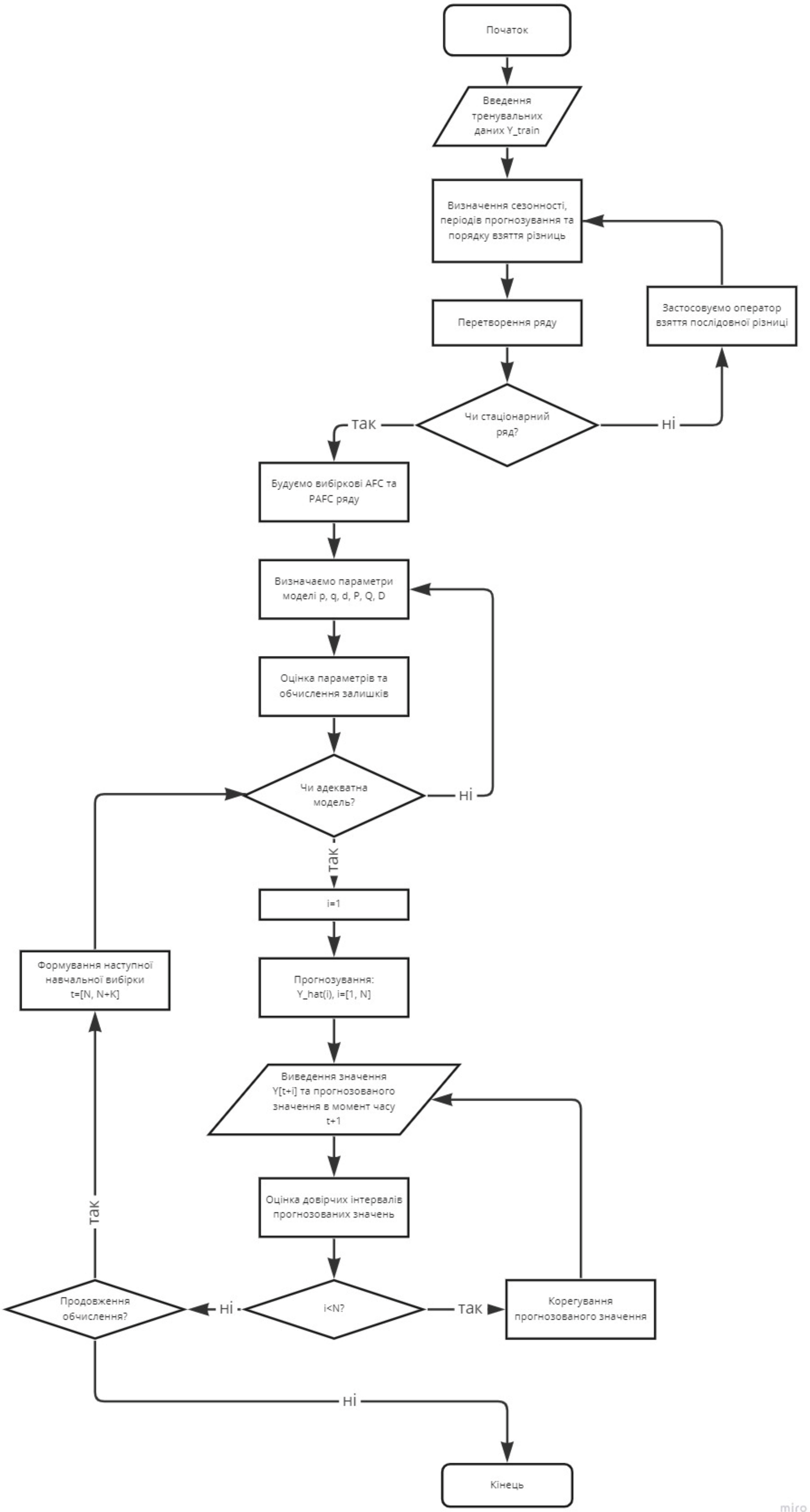
- pandas – зчитування даних формату .csv, та робота з дата фреймом;
- statsmodel – побудова моделі SARIMAX;
- plotly – побудова інтерактивних графіків.

Формати вихідних даних

Вихідні дані: .csv файл, що містить інформацію про дату та кількість підтверджених випадків COVID-19, що відповідають цій даті.

Результат роботи програми: прогноз у вигляді .csv файлу, а також графік з фактичними та прогнозованими значеннями у форматі .png.

Блок-схема алгоритму



Код програми

```
import pandas as pd
import plotly.express as px
from statsmodels.tsa.statespace.sarimax import SARIMAX
from sklearn.metrics import mean_squared_log_error
import plotly.graph_objects as go

TRAINING_SPLIT=0.7

def split(df):
    row_number=df.shape[0]

    global TRAIN_NUM
    TRAIN_NUM=int(row_number*TRAINING_SPLIT)

    df_train=df.iloc[:TRAIN_NUM, :]
    df_test=df.iloc[TRAIN_NUM:, :]

    return df_train, df_test

def losses(confirmed, df):
    actual=list(confirmed['Confirmed'])
    predcited=list(df['Forecast'])
    return mean_squared_log_error(actual, predcited)

def plot_results(df_test, df):
    fig=go.Figure()
    fig.add_trace(go.Scatter(
        x=df_test['Date'],
        y=df_test['Confirmed'],
        mode='lines',
        name='Actual values'
    ))
    fig.add_trace(go.Scatter(
        x=df.iloc[TRAIN_NUM:,0],
        y=df.iloc[TRAIN_NUM:,2],
        mode='lines',
        name='Predicted values'
    ))
    fig.show()

def build_model(df_train):
    sarimax_model=SARIMAX(df_train['Confirmed'], order=(4, 2, 0),
seasonal_order=(0, 1, 1, 7))
```

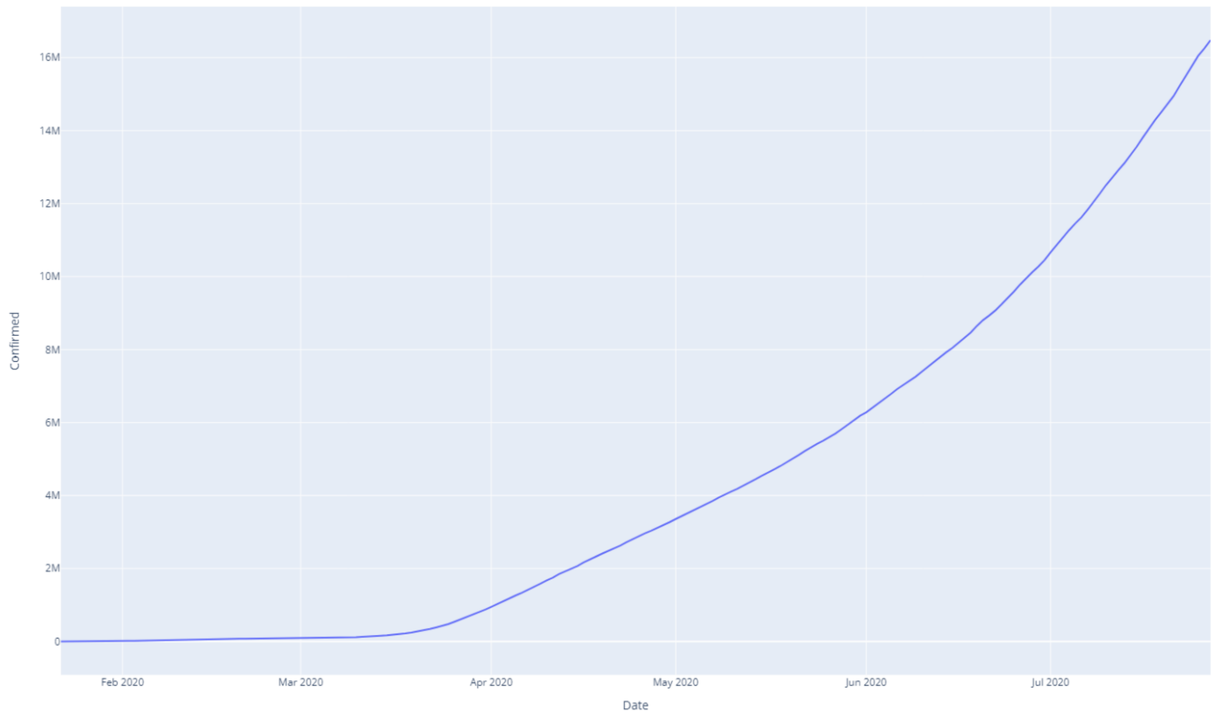
```python
    sarimax_model_fit=sarimax_model.fit()

    return sarimax_model_fit

def sarimax_predict(model, df_test):
    predicted=pd.DataFrame()
    forecast_test=model.forecast(len(df_test))

    predicted['Date']=df_test['Date']
    predicted['Forecast']=list(forecast_test)

    return predicted

def sarimax(df):
    print(f'Raw Data:\n{df.head(10)}')
    fig=px.line(df, x='Date', y='Confirmed')
    fig.show()

    df_train, df_test=split(df)

    model=build_model(df_train)
    predicted=sarimax_predict(model, df_test)

    df['Forecast']=[None]*TRAIN_NUM+list(predicted['Forecast'])
    df.plot()

    print(f'Predicted Values: \n{predicted.head(10)}')

    plot_results(df_test, df)
    plot_results(df, df)

    loss=losses(df_test, predicted)
    print(f'Losses: Mean Squared Log Error: {losses}')



if __name__=='__main__':
    df=pd.read_csv('day_wise.csv')
    covid_df=df[['Date', 'Confirmed']].dropna()
    sarimax(covid_df)
```

Опис результатів

Вхідні дані є .csv файл, що містить інформацію про дату та кількість підтверджених випадків COVID-19, що відповідають цій даті.

```
Raw Data:
         Date  Confirmed
0  2020-01-22        555
1  2020-01-23        654
2  2020-01-24        941
3  2020-01-25       1434
4  2020-01-26       2118
5  2020-01-27       2927
6  2020-01-28       5578
7  2020-01-29       6166
8  2020-01-30       8234
9  2020-01-31       9927
```

Візуалізація вхідних даних:



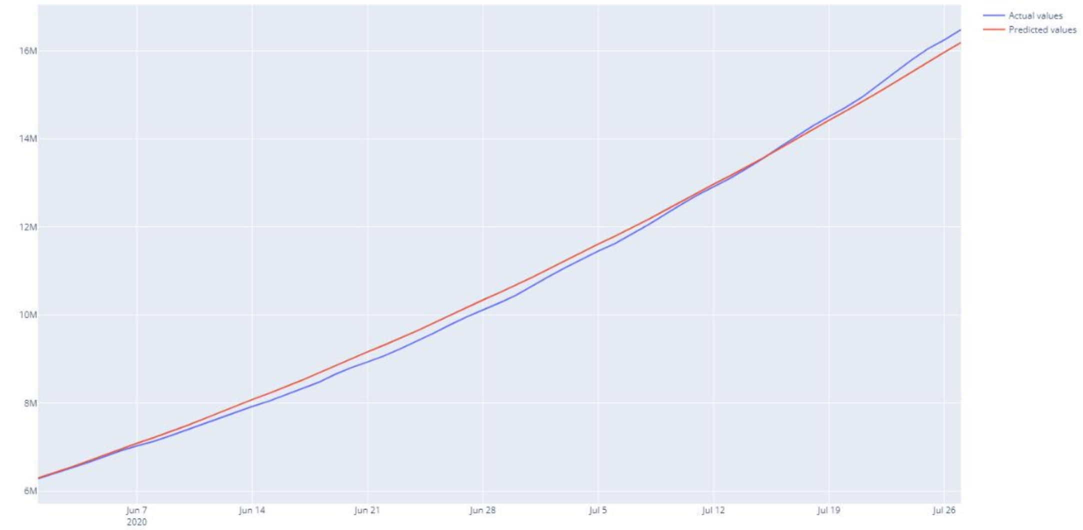
Розділимо вхідні дані на навчальну та тестову вибірки, де навчальна вибірка складає 70% від вхідного набору даних та використовується для навчання моделі SARIMAX, а тестова для перевірки результатів прогнозування.

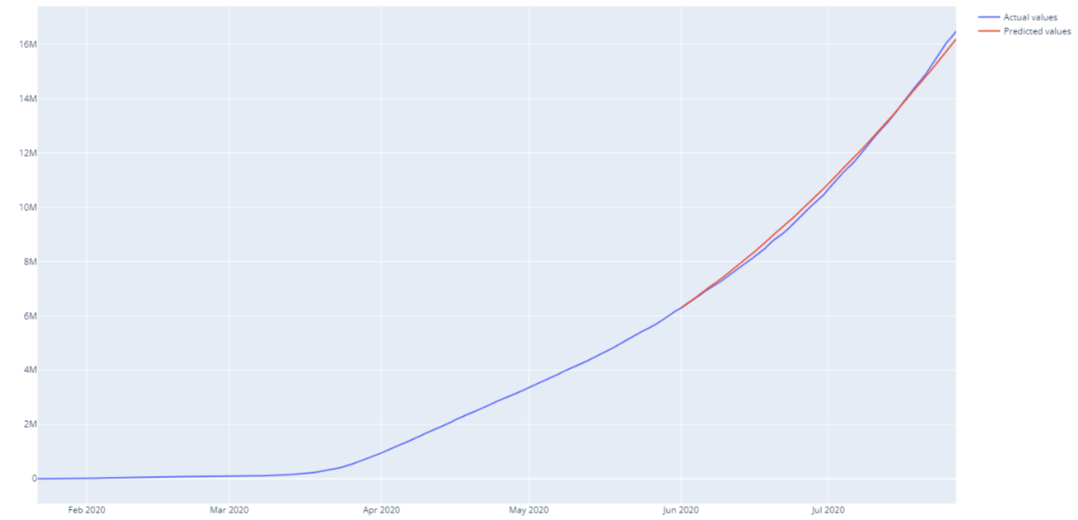
В результаті прогнозування було отримано такі результати:

```
Predicted Values:
           Date      Forecast
131  2020-06-01  6.297768e+06
132  2020-06-02  6.415621e+06
133  2020-06-03  6.545412e+06
134  2020-06-04  6.679584e+06
135  2020-06-05  6.818049e+06
136  2020-06-06  6.961794e+06
137  2020-06-07  7.087377e+06
138  2020-06-08  7.211995e+06
139  2020-06-09  7.343850e+06
140  2020-06-10  7.484690e+06
Losses: Mean Squared Log Error: 0.0002473046672078623
```

Похибка прогнозування складає 0,0002473 на тестових даних, що означає, що модель дає досить точні результати.

Візуалізація результатів прогнозування:





Висновки

В ході курсової роботи було розглянуто підхід авторегресійного інтегрованого ковзного середнього та його варіацій до часових рядів, в результаті чого було створено, навчено модель сезонного авторегресійного інтегрованого ковзного середнього SARIMAX. Дана модель була застосована для прогнозування кількості хворих на COVID-19.

Тестування моделі дало похибку, що становить 0,0002473, яка означає, що модель дає достатньо точні результати для подальшого використання у майбутньому. Натомість, вхідні дані потрібно оновлювати, а модель навчати заново, підібравши нові параметри.

Результати роботи програми представлено графічно.