НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

КАФЕДРА АВТОМАТИЗОВАНИХ СИСТЕМ ОБРОБКИ ІНФОРМАЦІЇ ТА УПРАВЛІННЯ

**КУРСОВА РОБОТА**

з дисципліни «Аналіз даних в інформаційно-управляючих системах»

на тему: «Прогнозування погодних умов за часовими рядами»

Студента  2  курсу ФІОТ групи ІП-92

напряму підготовки ІПЗ

спеціальності 121 Інженерія програмного

забезпечення

Залізчука Д.Д.

Керівник: старший викладач Олійник Юрій Олександрович

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка:  ECTS \_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| Члени комісії |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |
|  |  |  |  |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (підпис) |  | (вчене звання, науковий ступінь, прізвище та ініціали) |

Київ - 2021 рік

**ЛИСТ ЗАВДАННЯ**

Національний технічний університет України “КПІ”

Кафедра автоматизованих систем обробки інформації і управління

Дисципліна  Аналіз даних в інформаційно-управляючих системах

Спеціальність 121 "Інженерія програмного забезпечення"

Курс  2          Група          ІП-92                        Семестр   4

**ЗАВДАННЯ**

**на курсову роботу студента**

|  |
| --- |
|  |

|  |  |
| --- | --- |
| 1. Тема роботи Прогнозування погодних умов за часовими рядами |  |
| мамдб | |
|  | |

|  |  |
| --- | --- |
| 2. Строк здачі студентом закінченої роботи | 03.06.2021 |

|  |  |
| --- | --- |
| 3. Вихідні дані до роботи | https://www.kaggle.com/grubenm/austin-weather |
|  | |
|  | |
|  | |

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

Вступ, основна частина, постановка задачі, очищення даних - ETL-процес, обґрунтування методу прогнозування, проведення прогнозування, висновки, перелік посилань, Додаток А Тексти програмного коду

|  |
| --- |
|  |
|  |
|  |

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень )

|  |
| --- |
|  |
|  |
|  |
|  |

|  |  |
| --- | --- |
| 6. Дата видачі завдання |  |

АНОТАЦІЯ

В ході виконання курсової роботи було закріплено знання фундаментальних основ, термінів та понять в області сховищ даних та оперативного аналізу даних, а також засвоєно загальні принципи проведення аналізу даних. Продемонстровано навички з прогнозування даних за часовими рядами. Проаналізовано три алгоритми підходу до прогнозування: Prophet, Simple Moving Average, ARIMA. Також результатом отримали похибки роботи цих алгоритмів на даних, було побудовано графіки для наочного порівняння результатів прогнозування.

ЗМІСТ

ВСТУП……………………………………………………………………………………..5

ОСНОВНА ЧАСТИНА……………………………………………………………………6

1. ПОСТАНОВКА ЗАДАЧІ……………………………………………...………….6
2. ОЧИЩЕННЯ ДАНИХ – ETL-ПРОЦЕСИ……………………………………….7
3. ОБҐРУНТУВАННЯ ВИБОРУ МЕТОДІВ ПРОГНОЗУВАННЯ……………....13
4. ПРОВЕДЕННЯ ПРОГНОЗУВАННЯ…………………………………………..14
5. Метод “Prophet”……………………………………………………...15
6. Метод “Simple Moving Average”…………………………………….18
7. Метод “ARIMA”……………………………………………………..21
8. Порівняння результатів……………………………………………...24

ВИСНОВКИ……………………………………………………………………………...25

ПЕРЕЛІК ПОСИЛАНЬ…………………………………………………………………..27

ДОДАТКИ………………………………………………………………………………...28

Додаток А: тексти програмного коду…………………………………………..…28

ВСТУП

Курсова робота з дисципліни «Аналіз даних в інформаційно-управляючих системах» має за собою ціль закріплення знань та набуття практичних навичок, а саме систематизація знань роботи з даними, очищення даних, виконання алгоритмів над даними, керування даними.

Вирішуваною задачею стало прогнозування погодних умов з набору даних, що має набір дат та значення погодних умов (температура, вологість, точка роси тощо). Необхідно нормалізувати дані, видалити зайві, розробити ETL-процеси та виконати прогнозування погодних умов, використовуючи відомі алгоритми. Застосуємо метод Prophet (адитивна регресійна модель), Simple Moving Average (просте рухаюче середнє) та ARIMA (авторегресивна інтегрована рухаюча середня). Знайдемо результати прогнозів та візуалізуємо результати роботи.

ОСНОВНА ЧАСТИНА

1. ПОСТАНОВКА ЗАДАЧІ

Предметна область роботи – погодні умови.

Джерело даних: <https://www.kaggle.com/grubenm/austin-weather>

Дані наведені у форматі .csv (значення, що розділені комою).

Задача: спрогнозувати значення за допомогою трьох алгоритмів прогнозування, порівняти результати.

1. ОЧИЩЕННЯ ДАНИХ – ETL-ПРОЦЕСИ

Створимо діаграму сховища даних:

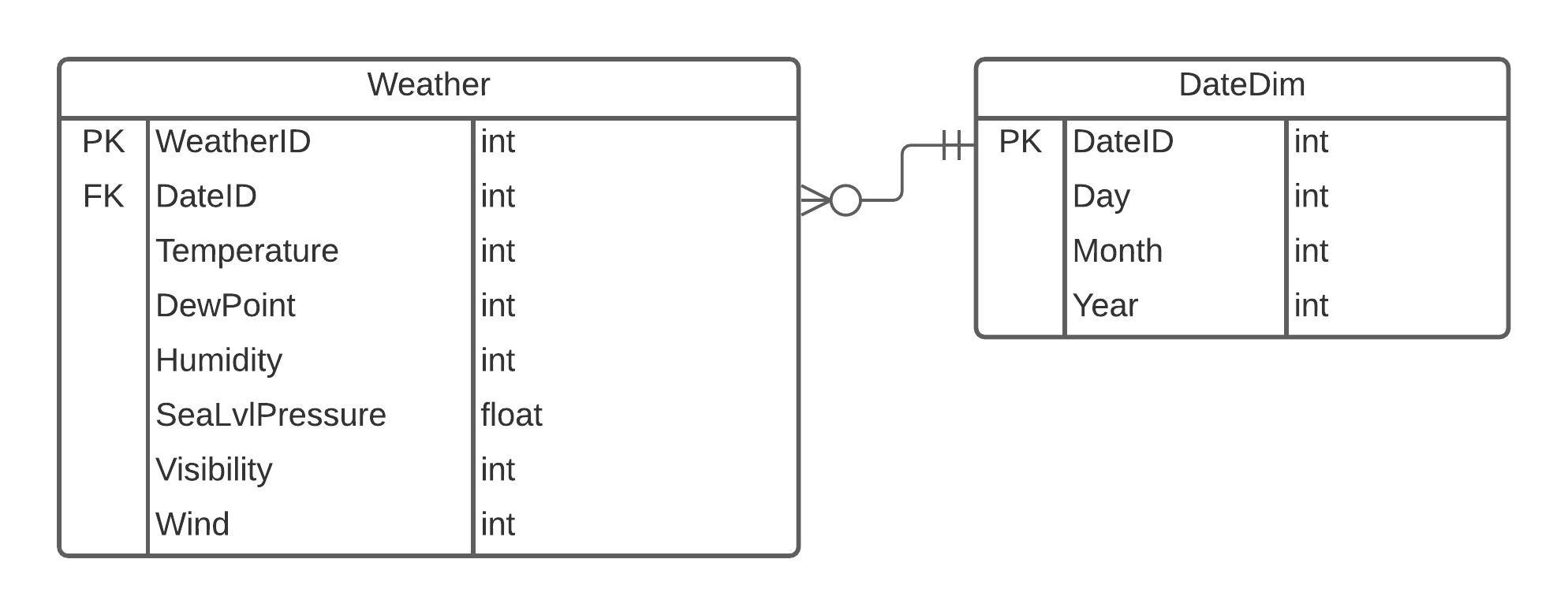


Рисунок 2.1 – Діаграма сховища даних

Таблиця 2.1 – Опис таблиці Weather

|  |  |
| --- | --- |
| Поле таблиці | Опис |
| WeatherID | Унікальний ключ для запису в таблиці |
| DateID | Посилання но зовнішній ключ у таблиці DateDim |
| Temperature | Температура повітря (за Фаренгейтом) |
| DewPoint | Точка роси (за Фаренгейтом) |
| Humidity | Вологість (у відсотках) |
| SeaLvlPressure | Тиск на рівні моря |
| Visibility | Видимість (Милі) |
| Wind | Швидкість вітру (Миль в годину) |

Таблиця 2.2 – Опис таблиці DateDim

|  |  |
| --- | --- |
| Поле таблиці | Опис |
| DateID | Унікальний ключ для запису в таблиці |

Продовження таблиці 2.2

|  |  |
| --- | --- |
| Поле таблиці | Опис |
| Day | День |
| Month | Місяць |
| Year | Рік |

Для організації сховища даних використаємо систему управління базами даних (СУБД) Microsoft SQL Server. Створимо таблицю для початкового завантаження даних до сховища.

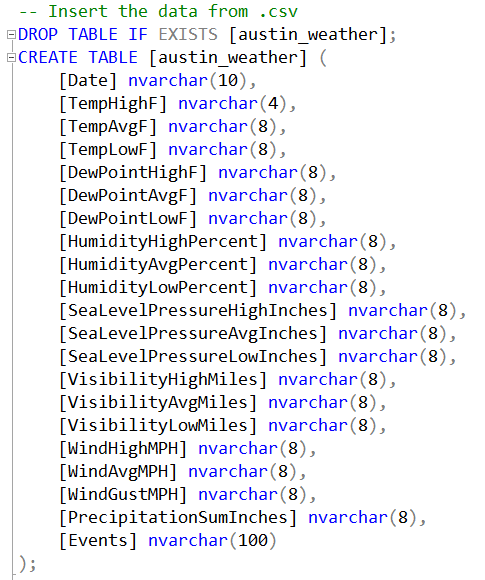


Рисунок 2.2 – Створення початкової таблиці

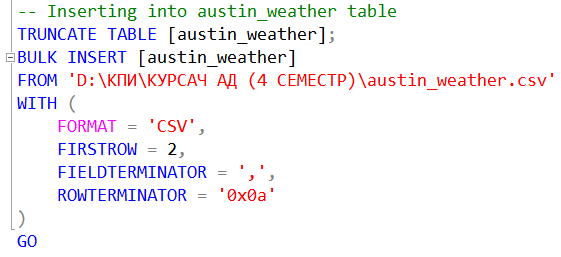


Рисунок 2.3 – Завантаження даних до початкової таблиці

Далі підготуємо модель сховища даних відповідно до діаграми.

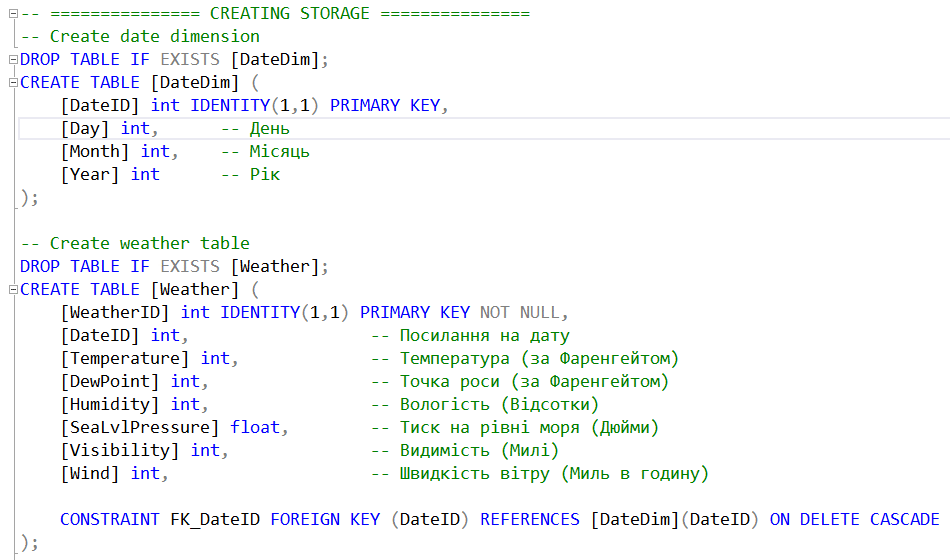


Рисунок 2.4 – Створення моделі сховища даних

Для розробки ETL-процесів застосуємо Integration Services від Microsoft. Проект такого типу дозволяє створювати та налаштовувати ETL-процеси за допомогою зручного графічного інтерфейсу.

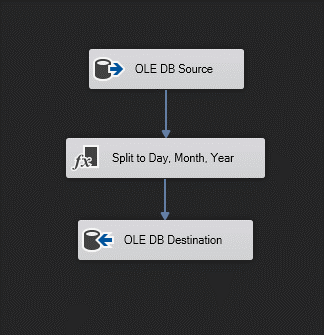


Рисунок 2.5 – Завантаження даних про дату до таблиці DateDim

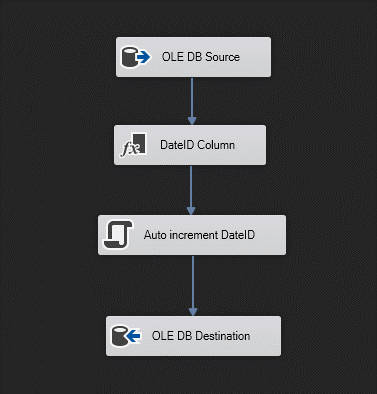


Рисунок 2.6 – Завантаження даних про погодні умови до таблиці Weather

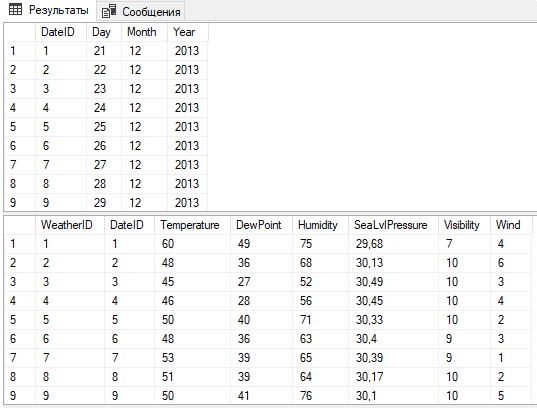


Рисунок 2.7 – Фрагменти даних з таблиць після виконання ETL-процесів

Після проведення очистки даних та створення сховища, завантажимо новий .csv файл з потрібними даними для зручного подальшого прогнозування.

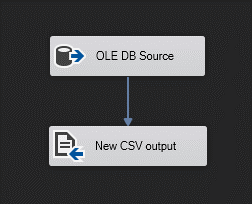


Рисунок 2.8 – Завантаження нового .csv файлу на основі даних сховища

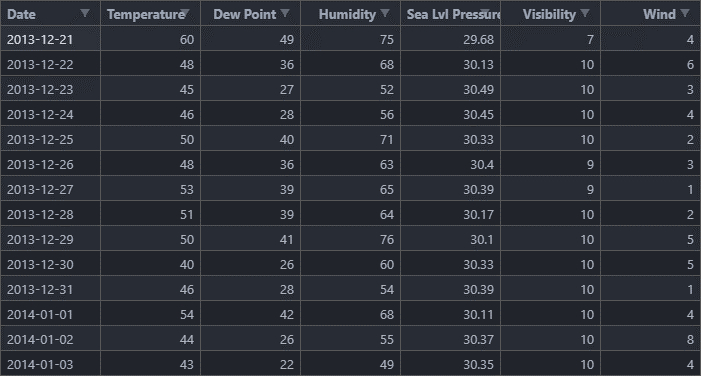


Рисунок 2.9 – Фрагмент даних з нового .csv файлу, створеного на основі даних сховища

1. ОБҐРУНТУВАННЯ ВИБОРУ МЕТОДІВ ПРОГНОЗУВАННЯ

Оскільки перед нами стоїть задача провести прогнозування, то потрібно дізнатися які існують методи та алгоритми до вирішення цієї задачі.

Виконавши аналіз деяких джерел у мережі Інтернет, я дізнався, що для вирішення задачі прогнозування можна використовувати наступні методи:

* Прогнозування на основі адитивної моделі (Prophet);
* Просте рухаюче середнє (Simple Moving Average);
* Авторегресивне інтегроване рухаюче середнє (ARIMA).

Тому, для виконання прогнозування будемо використовувати саме ці методи. Для цього мною була обрана мова програмування Python, що має багато бібліотек для аналізу даних різними способами, а також прогнозування.

Застосуємо такі бібліотеки:

* Pandas – для завантаження даних у програму, їх перетворення у таблиці;
* Matplotplib – для побудови графічних звітів (графіків тощо);
* Sklearn – для розрахунку середньоквадратичної помилки;
* Numpy – для використання ефективних структур даних;
* Statsmodels – для виконання алгоритмів прогнозування;
* Fbprophet – для виконання прогнозування за методом Prophet.

1. ПРОВЕДЕННЯ ПРОГНОЗУВАННЯ

Почнемо процес написання програмного забезпечення із завантаження даних з файлу та їх перетворення на DataFrame.

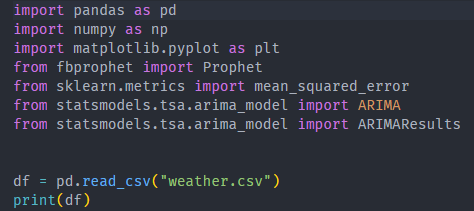


Рисунок 4.1 – Імпортування бібліотек та завантаження файлу

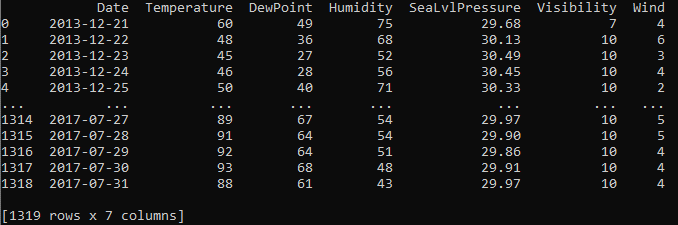


Рисунок 4.2 – Фрагмент даних, завантажених з файлу

* 1. Метод “Prophet”

Розпочнемо роботу з методу Prophet. Давайте трохи обговоримо, як же працює бібліотека Prophet. По суті, це адитивна регресійна модель, що складається з наступних компонент:

* Сезонні компоненти *s(t)* відповідають за моделювання періодичних змін, пов'язаних з тижневою та річною сезонністю. Тижнева сезонність моделюється за допомогою dummy variables. Додаються 6 додаткових ознак, наприклад, [Monday, Tuesday, Wednesday, Thursday, Friday, Saturday], які приймають значення 0 і 1 в залежності від дати. Ознаку sunday, відповідна до сьомого дня тижня, не додають, тому що вона буде лінійно залежати від інших днів тижня і це буде впливати на модель. Річна ж сезонність моделюється рядами Фур'є.
* Тренд *g(t)* - це кусково-лінійна або логістична функція. З лінійною функцією все зрозуміло. Логістична ж функція виду дозволяє моделювати зростання з насиченням, коли при збільшенні показника знижується темп його зростання. Типовий приклад - це зростання аудиторії програми або сайту.

Крім усього іншого, бібліотека вміє за історичними даними вибирати оптимальні точки зміни тренду. Але їх також можна задати і вручну (наприклад, якщо відомі дати релізів нової функціональності, які сильно вплинули на ключові показники).

* Компонента *h(t)* відповідає за задані користувачем аномальні дні, в тому числі і нерегулярні, такі як, наприклад, Black Fridays.
* Помилка *e(t)* містить інформацію, яка не врахована моделлю.

Створимо функцію fbprophet\_model, що буде виконувати прогнозування за набором даних методом Prophet.



Рисунок 4.1.1 – Прогнозування методом Prophet

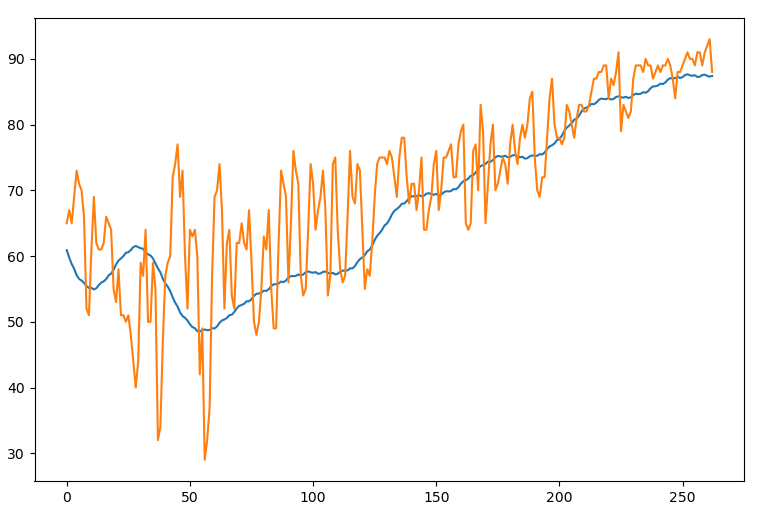


Рисунок 4.1.2 – Результат прогнозування методом Prophet



Рисунок 4.1.3 – Середньоквадратична помилка методу Prophet

* 1. Метод “Simple Moving Average”

Рухаюче середнє (Moving Average) - загальна назва для сімейства функцій, значення яких в кожній точці визначення дорівнюють деякому середньому значенню вихідної функції за попередній період.

Рухаючі середні зазвичай використовуються з даними часових рядів для згладжування короткострокових коливань і виділення основних тенденцій або циклів. Математично рухаюча середня є одним з видів згортки.

Просте рухаюче середнє, або арифметичне рухаюче середнє (Simple Moving Average) чисельно дорівнює середньому арифметичному значень вихідної функції за встановлений період і обчислюється за формулою:

де - значення простого ковзного середнього в точці *t*;

*n* - кількість значень вихідної функції для розрахунку рухаючого середнього (згладжує інтервал), чим ширше згладжує інтервал, тим більш плавним виходить графік функції;

- значення вихідної функції в точці *t – i*;

Отримане значення простого рухаючого середнього відноситься до середини обраного інтервалу, однак, традиційно його відносять до останньої точки інтервалу.

З попереднього свого значення просте рухаюче середнє може бути отримано за такою рекурентною формулою:

де - значення простого рухаючого середнього в точці *t*; - попереднє значення простого рухаючого середнього;

- значення вихідної функції в точці *t - n* (в разі часового ряду, саме «раннє» значення вихідної функції, що використовується для обчислення попереднього рухаючого середнього);

- значення досліджуваної функції в точці (в разі часового ряду, поточне - останнє значення).

Цією формулою зручно користуватися, щоб уникнути регулярного підсумовування всіх значень.

Створимо функцію moving\_average, що буде виконувати прогнозування за набором даних методом SMA (Simple Moving Average).

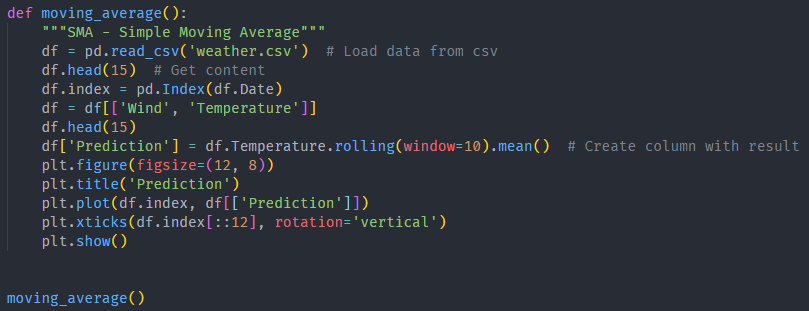


Рисунок 4.2.1 – Прогнозування методом SMA

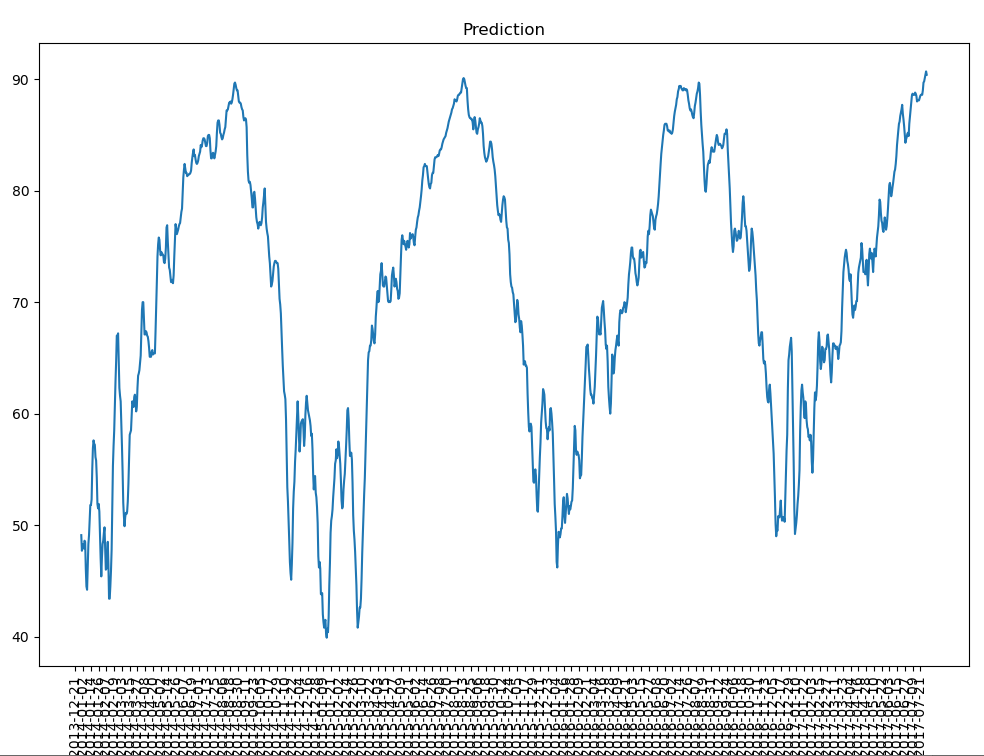


Рисунок 4.2.2 – Результат прогнозування методом SMA



Рисунок 4.2.3 – Середньоквадратична помилка методу SMA

* 1. Метод “ARIMA (AutoRegressive Integrated Moving Average)”

ARIMA - це абревіатура від AutoRegressive Integrated Moving Average. Це узагальнення більш простого авторегресійного рухаючого середнього і додає поняття інтеграції.

Ця абревіатура носить описовий характер і відображає ключові аспекти самої моделі. Використовується стандартне позначення ARIMA (p, d, q), де параметри замінюються цілочисельними значеннями для швидкої вказівки конкретної використовуваної моделі ARIMA.

Параметри моделі ARIMA визначаються наступним чином:

* **P**: Число спостережень відставання, включених в модель, також називається порядком відставання.
* **D**: Кількість разів, коли вихідні спостереження розрізняються, також називається ступенем відмінності.
* **Q**: Розмір вікна рухаючого середнього, також званий порядком ковзної середньої.

Значення 0 може бути використано для параметра, який вказує, що цей елемент моделі не використовується. Таким чином, модель ARIMA може бути налаштована для виконання функції моделі ARMA і навіть простої моделі AR, I або MA.

Ухвалення моделі ARIMA для часового ряду передбачає, що базовий процес, який справив спостереження, є процесом ARIMA. Це може здатися очевидним, але допомагає мотивувати необхідність підтвердження припущень моделі в необроблених спостереженнях і в залишкових помилках прогнозів з моделі.

Створимо функцію ARIMAS, що буде виконувати прогнозування за набором даних методом ARIMA.



Рисунок 4.3.1 – Прогнозування методом ARIMA

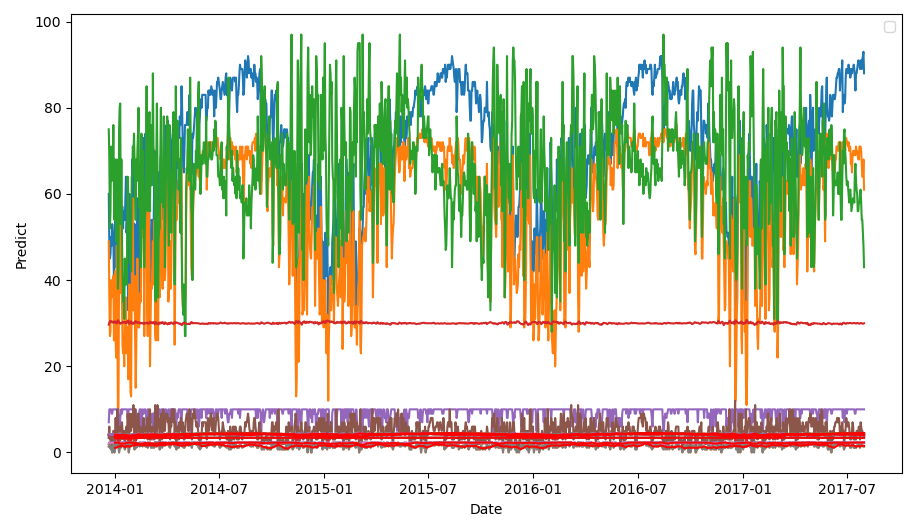


Рисунок 4.3.2 – Результат прогнозування методом ARIMA



Рисунок 4.3.3 – Середньоквадратична помилка методу ARIMA

* 1. Порівняння результатів

Під час дослідження було застосовано три методи прогнозування: Prophet, Simple Moving Average, ARIMA. Порівняємо їх середньоквадратичні помилки:

* Prophet: 6.37;
* Simple Moving Average: 12.75;
* ARIMA: 11.38.

Порівнюючи результати, можна сказати, що найкраще прогнозування дав метод Prophet, але і воно не було ідеальне, значення достатньо відрізнялись, але працював цей метод повільніше, ніж Simple Moving Average та ARIMA. Останні два методи один від одного за результатами не дуже відрізняються та працюють однаково швидко.

ВИСНОВКИ

Під час виконання курсової роботи переді мною стояла задача прогнозування погодних умов кількома методами. Для ефективного виконання роботи, її було розділено на такі підзадачі:

* Вибір датасету – було обрано датасет, що містить інформацію про погодні умови у різні дні;
* ETL-процеси – дані, що представлено у форматі .csv, відправляються до сховища даних за допомогою розроблених ETL-процесів, видаляються зайві дані, та завантажується новий .csv файл з потрібними даними, що буде використовуватися в подальшому;
* Обґрунтування методів прогнозування – на цьому етапі було здійснено пошук методів, якими можливо виконувати прогнозування. Було обрано 3 методи: Prophet, Simple Moving Average, ARIMA;
* Проведення прогнозування – дані з .csv файлу були завантажені до програми та було розроблено 3 функції для прогнозування трьома методами відповідно. Було обчислено середньоквадратичні помилки та побудовані графіки для наглядної оцінки результатів. Все це було виконано мовою програмування Python та бібліотек pandas, matplotlib, numpy, sklearn, statsmodels, fbprophet;
* Порівняння результатів – результати похибок були зведені і було знайдено найкращий та найгірший результати. Найкращим серед обраних методів опинився Prophet, але він програє у швидкості іншим методам. Найгіршим опинився Simple Moving Average, але його отримана похибка не дуже відрізняється від похибки методу ARIMA, але ці обидва методи працюють найшвидше.

Отже, використовуючи знання з дисципліни «Аналіз даних в інформаційно-управляючих системах», було проведено прогнозування погодних умов різними методами та їх порівняння. Мені вдалося закріпити знання та навички, отримані на лекціях та практичних заняттях, тому робота є виконаною.

ПЕРЕЛІК ПОСИЛАНЬ

1. Austin Weather. Historical temperature, precipitation, humidity, and windspeed for Austin, Texas [Електронний ресурс] / GrubenM // https://www.kaggle.com/. – 2017. – Режим доступу до ресурсу: <https://www.kaggle.com/grubenm/austin-weather>.
2. SQL Server Integration Services (SSIS) для починаючих [Електронний ресурс] / Leran2002 // https://habr.ru/. – 2017. – Режим доступу до ресурсу: <https://habr.com/ru/post/330618/>.
3. Prophet, forecasting at scale [Електронний ресурс] / Facebook // https://facebook.github.io/. – 2021. – Режим доступу до ресурсу: <https://facebook.github.io/prophet/>.
4. Time Series Forecasting with Prophet in Python [Електронний ресурс] / Jason Brownlee // https://machinelearningmastery.com/. – 2020. – Режим доступу до ресурсу: <https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/>.
5. Understanding a Simple Moving Average (SMA) [Електронний ресурс] / Adam Hayes // https://www.investopedia.com/. – 2021. – Режим доступу до ресурсу: <https://www.investopedia.com/terms/s/sma.asp>.
6. How to Create an ARIMA Model for Time Series Forecasting in Python [Електронний ресурс] / Jason Brownlee // https://machinelearningmastery.com/. – 2017. – Режим доступу до ресурсу: <https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/>.
7. Plotting with matplotlib [Електронний ресурс] / pandas team // https://pandas.pydata.org/. – 2014. – Режим доступу до ресурсу: <https://pandas.pydata.org/pandas-docs/version/0.13/visualization.html>.

ДОДАТКИ

Додаток А: тексти програмного коду

Сховище даних:

-- Insert the data from .csv

DROP TABLE IF EXISTS [austin\_weather];

CREATE TABLE [austin\_weather] (

    [Date] nvarchar(10),

    [TempHighF] nvarchar(4),

    [TempAvgF] nvarchar(8),

    [TempLowF] nvarchar(8),

    [DewPointHighF] nvarchar(8),

    [DewPointAvgF] nvarchar(8),

    [DewPointLowF] nvarchar(8),

    [HumidityHighPercent] nvarchar(8),

    [HumidityAvgPercent] nvarchar(8),

    [HumidityLowPercent] nvarchar(8),

    [SeaLevelPressureHighInches] nvarchar(8),

    [SeaLevelPressureAvgInches] nvarchar(8),

    [SeaLevelPressureLowInches] nvarchar(8),

    [VisibilityHighMiles] nvarchar(8),

    [VisibilityAvgMiles] nvarchar(8),

    [VisibilityLowMiles] nvarchar(8),

    [WindHighMPH] nvarchar(8),

    [WindAvgMPH] nvarchar(8),

    [WindGustMPH] nvarchar(8),

    [PrecipitationSumInches] nvarchar(8),

    [Events] nvarchar(100)

);

-- Inserting into austin\_weather table

TRUNCATE TABLE [austin\_weather];

BULK INSERT [austin\_weather]

FROM 'D:\КПИ\КУРСАЧ АД (4 СЕМЕСТР)\austin\_weather.csv'

WITH (

    FORMAT = 'CSV',

    FIRSTROW = 2,

    FIELDTERMINATOR = ',',

    ROWTERMINATOR = '0x0a'

)

GO

-- =============== CREATING STORAGE ===============

-- Create date dimension

DROP TABLE IF EXISTS [DateDim];

CREATE TABLE [DateDim] (

    [DateID] int IDENTITY(1,1) PRIMARY KEY,

    [Day] int,      -- День

    [Month] int,    -- Місяць

    [Year] int      -- Рік

);

-- Create weather table

DROP TABLE IF EXISTS [Weather];

CREATE TABLE [Weather] (

    [WeatherID] int IDENTITY(1,1) PRIMARY KEY NOT NULL,

    [DateID] int,                  -- Посилання на дату

    [Temperature] int,             -- Температура (за Фаренгейтом)

    [DewPoint] int,                -- Точка роси (за Фаренгейтом)

    [Humidity] int,                -- Вологість (Відсотки)

    [SeaLvlPressure] float,        -- Тиск на рівні моря (Дюйми)

    [Visibility] int,              -- Видимість (Милі)

    [Wind] int,                    -- Швидкість вітру (Миль в годину)

    CONSTRAINT FK\_DateID FOREIGN KEY (DateID) REFERENCES [DateDim](DateID) ON DELETE CASCADE

);

Програмне забезпечення для прогнозування:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from fbprophet import Prophet

from sklearn.metrics import mean\_squared\_error

from statsmodels.tsa.arima\_model import ARIMA

from statsmodels.tsa.arima\_model import ARIMAResults

df = pd.read\_csv("weather.csv")

print(df)

train\_size, test\_size, validation\_size = 0.6, 0.2, 0.2

train = df.iloc[:int(len(df)\*train\_size), :]

test = df.iloc[int(len(df)\*train\_size+1):int(len(df)\*train\_size+len(df)\*test\_size):]

validation = df.iloc[int(len(df)\*train\_size+len(df)\*test\_size+1):, :]

def fbprophet\_model(train, test, validation):

    """Розбиваємо дані на train, test та validation set, використовуємо метрики rmse та mse для порівняння."""

    work\_set = train.copy().append(test.copy())

    work\_set = work\_set[["Date", "Temperature"]]

    work\_set = work\_set.rename({"Date": "ds", "Temperature": "y"}, axis=1)

    comp\_set = validation.copy()[["Date", "Temperature"]]

    comp\_set = comp\_set.rename({"Date": "ds", "Temperature": "y"}, axis=1)

    model = Prophet(daily\_seasonality=True)

    model.fit(work\_set)

    future = model.make\_future\_dataframe(periods=len(comp\_set))

    res\_set = model.predict(future)[["ds", "yhat"]].yhat.values[-len(comp\_set):]

    rmse = mean\_squared\_error(comp\_set["y"].values, res\_set, squared=False)

    return res\_set, rmse

fbprophet\_model\_res, fbprophet\_model\_rmse = fbprophet\_model(train, test, validation)

plt.plot(fbprophet\_model\_res)

plt.plot(validation["Temperature"].values)

def moving\_average():

    """SMA - Simple Moving Average"""

    df = pd.read\_csv('weather.csv')  # Load data from csv

    df.head(15)  # Get content

    df.index = pd.Index(df.Date)

    df = df[['Wind', 'Temperature']]

    df.head(15)

    df['Prediction'] = df.Temperature.rolling(window=10).mean()  # Create column with result

    plt.figure(figsize=(12, 8))

    plt.title('Prediction')

    plt.plot(df.index, df[['Prediction']])

    plt.xticks(df.index[::12], rotation='vertical')

    plt.show()

moving\_average()

def ARIMAS():

    """Auto Regressive Integrated Moving Average"""

    df=pd.read\_csv('weather.csv')

    df.head()

    # Updating the header

    df['Date'] = pd.to\_datetime(df['Date'],infer\_datetime\_format=True)  #convert from string to datetime

    df.head()

    df.describe()

    indexedDataset = df.set\_index(['Date'])

    plt.xlabel('Date')

    plt.ylabel('Predict')

    plt.plot(indexedDataset)

    rolmean = indexedDataset.rolling(window=12).mean()  # Can be outputed

    rolstd = indexedDataset.rolling(window=12).std()  # Can be outputed

    plt.legend(loc='best')

    indexedDataset\_logScale = np.log(indexedDataset)

    plt.plot(indexedDataset\_logScale)

    movingAverage = indexedDataset\_logScale.rolling(window=12).mean()

    plt.plot(indexedDataset\_logScale)

    plt.plot(movingAverage, color='red')

    datasetLogScaleMinusMovingAverage = indexedDataset\_logScale - movingAverage

    datasetLogScaleMinusMovingAverage.head(12)

    datasetLogScaleMinusMovingAverage.dropna(inplace=True)

    datasetLogScaleMinusMovingAverage.head(10)

    plt.show()

ARIMAS()