MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
V.N. Karazin Kharkiv National University
School of Mathematics and Computer Science
Department of Theoretical and Applied Informatics

Master's Thesis

"Prediction of the dynamics COVID19 epidemic process of using the 5th degree Polynomial regression model"

Done by: 2-th year student, group MCS-64
specialty: Computer Sciences and Information Technologies
educational program: "Computer Sciences"
Fan Quanlong

Supervisor: Kostiantyn Nosov
Reviewer: Kseniia Bazilevych
Adviser: Ruslan Borodai

Kharkiv, 2024

TABLE OF CONTENT

INTRODUCTION

The COVID-19 pandemic has had a profound and unprecedented impact on societies and economies worldwide, disrupting nearly every aspect of human life. From its onset, the pandemic exposed vulnerabilities in healthcare systems, economies, and social structures, prompting a reevaluation of global preparedness and resilience. Healthcare systems faced extraordinary challenges, with hospitals overwhelmed by the influx of patients, shortages of critical medical supplies, and unprecedented demands on healthcare workers. These strains revealed significant disparities in healthcare access and infrastructure between developed and developing nations, exacerbating preexisting inequalities [1].

Economically, the pandemic triggered one of the most severe global recessions in recent history. Lockdowns and social distancing measures implemented to curb the virus's spread led to widespread business closures, layoffs, and disruptions in supply chains. Key industries, such as tourism, hospitality, and retail, suffered substantial losses, while unemployment rates surged, leaving millions of people financially vulnerable. Governments worldwide responded with fiscal stimulus packages, but the long-term economic scars, including heightened national debts and uneven recoveries, remain evident [2].

On a societal level, the pandemic reshaped daily life and social interactions. Remote work and online education became the norm for many, accelerating the adoption of digital technologies but also highlighting the digital divide. Social isolation, uncertainty, and anxiety about the virus's effects significantly impacted mental health, increasing the prevalence of stress, depression, and other psychological issues [3]. Vulnerable groups, including the elderly, individuals with preexisting health conditions, and low-income populations, faced heightened risks, further emphasizing the pandemic's unequal burden [4].

The global nature of the crisis underscored the interdependence of nations and the critical need for international cooperation. Collaborative efforts in vaccine development and distribution demonstrated the potential of global partnerships but

also highlighted inequities in access, with wealthier countries securing the majority of supplies. The pandemic has served as a stark reminder of the importance of resilience, innovation, and equity in addressing complex global challenges, offering lessons that extend far beyond the immediate crisis [5].

Predicting the dynamics of a pandemic like COVID-19 presents significant challenges due to the inherent instability and variability of statistical data and the complex interplay of social, economic, and medical factors. The accuracy of any predictive model is contingent on the quality and consistency of the data used. However, pandemic-related data often exhibit high levels of inconsistency, stemming from variations in testing rates, reporting practices, and data collection methodologies across regions and countries. These discrepancies create gaps and noise in the datasets, complicating efforts to derive reliable trends and patterns. Additionally, changes in testing strategies, such as prioritizing symptomatic individuals or scaling back mass testing efforts, can skew reported case numbers, further limiting the reliability of the data [6].

Beyond data inconsistencies, the spread of the virus is influenced by a myriad of interconnected social, economic, and medical factors that add layers of complexity to predictive modeling. Social behaviors, such as adherence to preventive measures like mask-wearing and social distancing, vary widely across populations and are influenced by cultural norms, misinformation, and public trust in authorities. Economic considerations, such as workforce mobility, economic activity levels, and urbanization, also play a crucial role, as densely populated areas and economically active regions tend to experience different transmission dynamics compared to rural or economically stagnant areas. Furthermore, medical factors, including vaccination rates, healthcare capacity, and the emergence of new variants, significantly impact the trajectory of the pandemic. Variants with increased transmissibility or resistance to existing treatments can quickly alter the course of an outbreak, rendering previous predictions obsolete.

The dynamic nature of these influencing factors, combined with the inherent uncertainty in human behavior and policy responses, makes long-term predictions

particularly challenging. Models must constantly adapt to incorporate new data and evolving conditions, highlighting the critical need for robust and flexible approaches in pandemic forecasting. Understanding and addressing these challenges is essential for improving the reliability of predictive models and supporting effective public health decision-making [7].

The absence of universal mathematical models capable of predicting complex dynamic processes such as pandemics represents a significant challenge in epidemiological research and public health planning. Pandemics are inherently multifaceted phenomena, influenced by a wide range of biological, social, economic, and environmental factors that interact in unpredictable ways. No single mathematical framework can fully encapsulate the intricacies of these processes, as the dynamics of disease spread are shaped by ever-changing variables, including viral mutations, population behavior, healthcare system responses, and government policies.

Traditional epidemiological models, such as the Susceptible-Infectious-Recovered (SIR) framework, provide valuable insights into basic disease dynamics but are often inadequate for capturing the complexity of real-world scenarios. These models typically rely on simplifying assumptions, such as homogeneous mixing of populations or static transmission rates, which fail to reflect the heterogeneous and dynamic nature of human interactions and the influence of external interventions like vaccination campaigns or lockdown measures. As a result, their predictive accuracy diminishes in the face of rapidly evolving circumstances [7].

Moreover, pandemics like COVID-19 have highlighted the limitations of applying general models across diverse geographic and socioeconomic contexts. Factors such as healthcare infrastructure, cultural practices, and economic resilience vary significantly between regions, necessitating localized modeling approaches that account for these differences. The lack of a standardized methodology to integrate such diverse variables into a cohesive predictive framework underscores the need for more adaptive and context-sensitive models.

The development of universal models is further constrained by the unpredictable nature of pandemics, including the emergence of new variants, changing population immunity levels, and global interconnectivity that accelerates disease spread. These factors demand a degree of flexibility and responsiveness that traditional models are often ill-equipped to provide. Addressing this gap requires an interdisciplinary approach that combines advances in mathematical modeling, machine learning, and data science with domain-specific knowledge from epidemiology, sociology, and economics. Such efforts are essential to create robust tools capable of adapting to the complexities and uncertainties inherent in pandemic dynamics.

The development of a regression model that accounts for the temporal variability of parameters is essential for accurately predicting the dynamics of complex phenomena like pandemics. Epidemic processes are inherently dynamic, characterized by shifts in transmission rates, changes in public health policies, and evolving population behavior. A static approach to modeling these processes fails to capture the nuanced fluctuations over time, leading to inaccuracies in forecasts and limiting the utility of such models for real-world decision-making [8].

Temporal variability arises from a multitude of factors that influence disease spread, including the implementation or relaxation of mitigation measures, such as lockdowns or vaccination campaigns, and the emergence of new virus variants with differing transmissibility or resistance to immunity. Additionally, population-level changes, such as increasing immunity due to infection or vaccination, further alter the trajectory of an epidemic. These dynamic elements require models that can incorporate time-sensitive data and adjust predictions as new information becomes available.

Regression models, particularly those designed to handle time-series data, offer a robust framework for integrating temporal variability into predictive analytics. By capturing the relationships between independent variables, such as time, intervention measures, or mobility trends, and dependent variables, such as case counts or hospitalizations, these models can dynamically adapt to changing

conditions. Advanced techniques, such as piecewise regression or models with time-varying coefficients, allow for more precise representations of periods of rapid change, such as during the onset of a new wave of infections.

Moreover, integrating temporal variability into regression models enhances their relevance for policy-making and resource allocation. Accurate, time-sensitive forecasts enable public health officials to anticipate surges in cases, allocate medical supplies effectively, and implement targeted interventions. Without accounting for temporal changes, models risk providing static and outdated insights, undermining their reliability in a dynamic and rapidly evolving context.

The COVID-19 pandemic has presented an unprecedented challenge to the global healthcare system, exposing weaknesses in preparedness and resilience across countries. The scale and speed of the virus's spread emphasized the urgent need for accurate forecasting to ensure the rational allocation of critical resources, such as medical supplies, hospital beds, and personnel. Effective predictions play a pivotal role in mitigating the pandemic's impact by informing public health strategies and enabling timely interventions. This highlights the importance of developing robust predictive models to address current and future health crises [9].

From a scientific perspective, the pandemic offers a unique opportunity to explore the potential of regression models in epidemiological forecasting. Regression techniques are essential tools for analyzing complex, dynamic datasets, and their application to COVID-19 allows researchers to assess their strengths and limitations in modeling disease progression. Additionally, the decision to focus on France as the object of study provides a context-rich scenario for analysis. France's advanced healthcare system, comprehensive COVID-19 data reporting, and diverse regional characteristics make it an ideal case for examining the interplay of epidemiological, social, and economic factors in shaping the pandemic's trajectory [10].

On a practical level, this work emphasizes the use of Python and its powerful libraries, such as Scikit-learn and Pandas, for data modeling and analysis. These tools enable efficient handling of large datasets, the implementation of sophisticated

regression algorithms, and the automation of forecasting processes. By leveraging these technologies, this study aims to demonstrate how programming can streamline the development of predictive models, making them accessible and actionable for addressing real-world challenges. Ultimately, the integration of scientific inquiry with practical applications underscores the relevance and impact of this research in both academic and applied domains.

The primary objective of this research is to analyze and compare various approaches to forecasting epidemiological processes, with a focus on their applicability and effectiveness in the context of the COVID-19 pandemic. By exploring different modeling techniques, the study aims to identify key methodologies that can reliably predict the progression of complex health crises.

A core goal of this work is the development, calibration, and testing of a regression model specifically designed to forecast the dynamics of COVID-19 in France. This involves leveraging comprehensive datasets and advanced analytical tools to build a robust model capable of reflecting the unique characteristics of the pandemic in the selected country. The accuracy of the model will be rigorously evaluated using established statistical metrics, and recommendations for its further improvement will be provided, ensuring its adaptability to changing conditions and new data.

Additionally, the project aims to demonstrate the critical role of programming in addressing global epidemiological crises. By utilizing Python and its ecosystem of powerful libraries, the research highlights how programming can facilitate efficient data processing, model development, and predictive analysis. This underscores the value of computational tools in supporting evidence-based decision-making during pandemics and in preparing for future public health challenges.

1 REVIEW OF PUBLICATIONS ON THE RESEARCH TOPIC

The study of regression modeling, particularly in the context of predicting epidemiological processes such as the dynamics of COVID-19, has been the focus of numerous scientific publications. This chapter provides a comprehensive review of key research works, highlighting advancements in regression techniques, their application in public health forecasting, and the challenges associated with modeling complex and dynamic datasets.

The research "A hybrid approach to forecast the COVID-19 epidemic trend" explores a combined modeling methodology that integrates statistical and computational tools to improve the accuracy of COVID-19 trend predictions.

Studying the progress and trend of the novel coronavirus pneumonia (COVID-19) transmission mode will help effectively curb its spread. Some commonly used infectious disease prediction models are introduced. The hybrid model is proposed, which overcomes the disadvantages of the logistic model’s inability to predict the number of confirmed diagnoses and the drawbacks of too many tuning parameters of the SEIR (Susceptible, Exposed, Infectious, Recovered) model.

The realization and superiority of the prediction of the proposed model are proven through experiments. At the same time, the influence of different initial values of the parameters that need to be debugged on the hybrid model is further studied, and the mean error is used to quantify the prediction effect. By forecasting epidemic size and peak time and simulating the effects of public health interventions, this paper aims to clarify the transmission dynamics of COVID-19 and recommend operation suggestions to slow down the epidemic. It is suggested that the quick detection of cases, sufficient implementation of quarantine and public self-protection behaviours are critical to slow down the epidemic [11].

The research "Predictive Mathematical Models of the COVID-19 Pandemic: Underlying Principles and Value of Projections" examines the foundational principles of mathematical modeling and evaluates their significance in projecting the trajectory of the COVID-19 pandemic.

The COVID-19 pandemic has posed unprecedented challenges to medical education, necessitating rapid adaptation to ensure the continuity of learning while maintaining safety. This article examines the multifaceted impact of COVID-19 on medical training, focusing on the transition from traditional in-person instruction to virtual platforms. It highlights the disruption of clinical rotations and the implications for skill acquisition and patient interaction, critical components of medical training.

The authors discuss strategies employed by educational institutions to mitigate these challenges, including the incorporation of telemedicine, virtual simulations, and hybrid models of learning. Furthermore, the paper explores the broader implications for the future of medical education, emphasizing the potential for sustained integration of technology, novel teaching methodologies, and redefined competencies to prepare future physicians for emergent health crises. By examining the immediate responses and long-term adaptations, this work underscores the resilience of medical education systems in addressing the challenges imposed by a global pandemic [12].

The research "COVID-19 epidemic prediction and the impact of public health interventions: A review of COVID-19 epidemic models" provides a comprehensive overview of epidemic models, emphasizing their predictive capabilities and the influence of public health measures on COVID-19 dynamics.

The coronavirus disease outbreak of 2019 (COVID-19) has been spreading rapidly to all corners of the word, in a very complex manner. A key research focus is in predicting the development trend of COVID-19 scientifically through mathematical modelling. We conducted a systematic review of epidemic prediction models of COVID-19 and the public health intervention strategies by searching the Web of Science database. 55 studies of the COVID-19 epidemic model were reviewed systematically. It was found that the COVID-19 epidemic models were different in the model type, acquisition method, hypothesis and distribution of key input parameters. Most studies used the gamma distribution to describe the key time

period of COVID-19 infection, and some studies used the lognormal distribution, the Erlang distribution, and the Weibull distribution.

The setting ranges of the incubation period, serial interval, infectious period and generation time were 4.9–7 days, 4.41–8.4 days, 2.3–10 days and 4.4–7.5 days, respectively, and more than half of the incubation periods were set to 5.1 or 5.2 days. Most models assumed that the latent period was consistent with the incubation period. Some models assumed that asymptomatic infections were infectious or pre-symptomatic transmission was possible, which overestimated the value of R0. For the prediction differences under different public health strategies, the most significant effect was in travel restrictions.

There were different studies on the impact of contact tracking and social isolation, but it was considered that improving the quarantine rate and reporting rate, and the use of protective face mask were essential for epidemic prevention and control. The input epidemiological parameters of the prediction models had significant differences in the prediction of the severity of the epidemic spread. Therefore, prevention and control institutions should be cautious when formulating public health strategies by based on the prediction results of mathematical models [13].

The research " Multi-source dynamic ensemble prediction of infectious disease and application in COVID-19 case" investigates the influence of meteorological factors, specifically air temperature and humidity.

The development of an epidemic always exhibits multiwave oscillation owing to various anthropogenic sources of transmission. Particularly in populated areas, the large-scaled human mobility led to the transmission of the virus faster and more complex. The accurate prediction of the spread of infectious diseases remains a problem. To solve this problem, we propose a new method called the multi-source dynamic ensemble prediction (MDEP) method that incorporates a modified susceptible-exposed-infected-removed (SEIR) model to improve the accuracy of the prediction result.

The modified SEIR model is based on the compartment model, which is suitable for local-scale and confined spaces, where human mobility on a large scale is not considered. Moreover, compartmental models cannot be used to predict multiwave epidemics. The proposed MDEP method can remedy defects in the compartment model. In this study, multi-source prediction was made on the development of coronavirus disease 2019 (COVID-19) and dynamically assembled to obtain the final integrated result. We used the real epidemic data of COVID-19 in three cities in China: Beijing, Lanzhou, and Beihai. Epidemiological data were collected from 17 April, 2022 to 12 August, 2022.

Compared to the one-wave modified SEIR model, the MDEP method can depict the multiwave development of COVID-19. The MDEP method was applied to predict the number of cumulative cases of recent COVID-19 outbreaks in the aforementioned cities in China. The average accuracy rates in Beijing, Lanzhou, and Beihai were 89.15%, 91.74%, and 94.97%, respectively.

The MDEP method improved the prediction accuracy of COVID-19. With further application to other infectious diseases, the MDEP method will provide accurate predictions of infectious diseases and aid governments make appropriate directives [14].

The research "On the reliability of predictions on COVID-19 dynamics: A systematic and critical review of modelling techniques" critically examines the reliability of various modeling approaches used to predict COVID-19 dynamics, highlighting their strengths, limitations, and applicability.

Since the emergence of the novel 2019 coronavirus pandemic in December 2019 (COVID-19), numerous modellers have used diverse techniques to assess the dynamics of transmission of the disease, predict its future course and determine the impact of different control measures. In this study, we conducted a global systematic literature review to summarize trends in the modelling techniques used for Covid-19 from January 1st, 2020 to November 30th, 2020. We further examined the accuracy and precision of predictions by comparing predicted and observed values for cumulative cases and deaths as well as uncertainties of these predictions. From

an initial 4311 peer-reviewed articles and preprints found with our defined keywords, 242 were fully analysed.

Most studies were done on Asian (78.93%) and European (59.09%) countries. Most of them used compartmental models (namely SIR and SEIR) (46.1%) and statistical models (growth models and time series) (31.8%) while few used artificial intelligence (6.7%), Bayesian approach (4.7%), Network models (2.3%) and Agent-based models (1.3%). For the number of cumulative cases, the ratio of the predicted over the observed values and the ratio of the amplitude of confidence interval (CI) or credibility interval (CrI) of predictions and the central value were on average larger than 1 indicating cases of inaccurate and imprecise predictions, and large variation across predictions.

There was no clear difference among models used for these two ratios. In 75% of predictions that provided CI or CrI, observed values fall within the 95% CI or CrI of the cumulative cases predicted. Only 3.7% of the studies predicted the cumulative number of deaths. For 70% of the predictions, the ratio of predicted over observed cumulative deaths was less or close to 1. Also, the Bayesian model made predictions closer to reality than classical statistical models, although these differences are only suggestive due to the small number of predictions within our dataset (9 in total). In addition, we found a significant negative correlation (rho = - 0.56, $p = 0.021$) between this ratio and the length (in days) of the period covered by the modelling, suggesting that the longer the period covered by the model the likely more accurate the estimates tend to be. Our findings suggest that while predictions made by the different models are useful to understand the pandemic course and guide policy-making, some were relatively accurate and precise while other not [15].

The research "Simulations and Predictions of COVID-19 Pandemic With the Use of SIR Model" explores the application of the SIR model for simulating and forecasting the progression of the COVID-19 pandemic, demonstrating its utility and limitations in capturing disease dynamics.

The COVID-19 pandemic is of great interest to researchers due to high mortality and a very negative impact to the world economy. A detailed scientific

analysis of the phenomenon is yet to come, but the public is already interested in the problems of duration of the epidemic, the expected number of patients, where and when the pandemic started. Correct simulation of the pandemic dynamics needs complicated mathematical models and many efforts for unknown parameters identification. In this article, preliminary estimates for many countries and world will be presented, summarized and discussed.

The optimal values of the SIR model parameters were identified with the use of statistical approach for epidemic dynamics in USA, Germany, UK, the Republic of Korea, and in the world. The actual number of cases and the number of patients spreading the infection versus time were calculated. The hidden periods, durations and final sizes of the epidemic were evaluated. In particular, the pandemic began in China no later than October, 2019. If current trends continue, the end of the pandemic should be expected no earlier than March 2021, the global number of cases will exceed 5 million. A simple method for assessing the risk of premature weakening of quarantines is proposed.

The SIR model and statistical approach to the parameter identification are helpful to make some reliable estimations for the epidemic dynamics, e.g., the real time of the outbreak, final size and duration of the epidemic and the number of persons spreading the infection versus time. This information will be useful to regulate the quarantine activities and to predict the medical and economic consequences of the pandemic [16].

2 THEORETICAL CHAPTER

2.1 Fundamentals of mathematical modeling in epidemiology

2.1.1 Definition of mathematical modeling and Its role in forecasting dynamic processes

Mathematical modeling is a systematic and structured approach to understanding, analyzing, and predicting complex real-world phenomena by translating them into mathematical frameworks [17].

In the context of epidemiology, mathematical modeling plays a pivotal role in describing the dynamics of disease transmission, forecasting outbreaks, and evaluating the impact of interventions. By abstracting real-world complexities into equations, variables, and parameters, mathematical models provide a simplified yet powerful representation of how diseases spread within populations and how various factors influence their progression.

At its core, mathematical modeling involves the identification of key components of a system and the relationships between them. For epidemiological processes, this typically includes factors such as the rate of infection, recovery, and mortality, as well as external influences like public health measures, vaccination campaigns, and societal behaviors. These components are represented as variables and constants, linked together by equations that govern the system's dynamics over time. For example, classic compartmental models, such as the Susceptible-Infectious-Recovered (SIR) framework, divide the population into distinct groups and use differential equations to describe transitions between these states.

The role of mathematical modeling in forecasting dynamic processes extends beyond theoretical analysis. Models are invaluable tools for predicting the future course of an epidemic, enabling policymakers to make informed decisions about resource allocation and intervention strategies. For instance, during the COVID-19

pandemic, mathematical models were widely used to estimate the number of cases, hospitalizations, and deaths under different scenarios, such as varying levels of social distancing or vaccination coverage. These forecasts helped governments and health organizations plan for surges in healthcare demand and allocate resources such as ventilators, personal protective equipment, and hospital beds more effectively [18].

One of the primary advantages of mathematical modeling is its ability to incorporate diverse datasets and adapt to changing conditions. Epidemiological models can integrate data from multiple sources, including case counts, mobility patterns, demographic information, and genomic sequencing of pathogens. This adaptability allows models to provide real-time insights as new data becomes available, making them highly relevant in fast-evolving situations like pandemics. Furthermore, mathematical models are instrumental in scenario analysis, enabling researchers to simulate the effects of different interventions and assess their potential outcomes before implementation.

However, the effectiveness of mathematical modeling depends on the accuracy of the underlying assumptions and the quality of the data used. Models are simplifications of reality and cannot account for all factors influencing disease spread. For example, human behavior, which plays a significant role in the transmission of infectious diseases, is often difficult to quantify and predict. Similarly, variability in data collection and reporting practices can introduce uncertainty into model outputs. Despite these limitations, mathematical modeling remains a cornerstone of epidemiological research, providing a foundation for understanding complex dynamic processes and supporting evidence-based decision-making.

Mathematical modeling bridges the gap between theoretical understanding and practical application in epidemiology. By representing disease dynamics in mathematical terms, models enable researchers to analyze trends, predict outcomes, and optimize interventions, thereby playing a critical role in managing and mitigating the impact of epidemics on public health.

2.1.2 Use of mathematical models in the study of the COVID-19 pandemic

The COVID-19 pandemic has underscored the critical importance of mathematical models in understanding and managing global health crises. From the early days of the outbreak, mathematical modeling played a central role in guiding public health responses by providing insights into the transmission dynamics of the virus, predicting future trends, and evaluating the potential impact of various interventions. By simulating complex epidemiological processes, these models have served as essential tools for decision-making at both local and global levels [19].

One of the most notable applications of mathematical models during the COVID-19 pandemic was in predicting the spread of the virus. Models such as the Susceptible-Infectious-Recovered (SIR) framework and its variations were widely used to estimate key metrics, including the basic reproduction number (R0), which indicates how many secondary infections arise from a single case in a fully susceptible population. These models helped epidemiologists and policymakers understand the contagious nature of the virus and the conditions required to suppress its transmission. By incorporating real-time data, such as case numbers, testing rates, and mobility patterns, these models provided forecasts that were crucial for planning healthcare resources and implementing containment measures.

Beyond predicting the spread of the virus, mathematical models were instrumental in assessing the effectiveness of public health interventions, such as lockdowns, social distancing, and mask mandates. By simulating scenarios with and without these measures, models demonstrated their impact on reducing the rate of infection and preventing healthcare systems from becoming overwhelmed. For instance, during periods of exponential case growth, models highlighted the urgency of implementing non-pharmaceutical interventions (NPIs) and provided evidence to justify their continuation or relaxation based on epidemiological trends.

Vaccination strategies were another area where mathematical models proved invaluable. Models were used to simulate the rollout of vaccination campaigns, determine optimal allocation strategies, and estimate the impact of achieving

specific coverage levels on herd immunity. They accounted for factors such as vaccine efficacy, distribution logistics, and public willingness to be vaccinated. These analyses helped shape policies aimed at maximizing the benefits of vaccination while minimizing disparities in access across different regions and populations.

Additionally, mathematical models contributed to understanding the emergence and spread of new variants of the virus. As variants with increased transmissibility or resistance to immunity appeared, models adapted to include these factors, allowing for more accurate predictions and timely adjustments to public health responses. For example, models incorporating genomic data helped assess the potential impact of variants on infection rates and vaccine effectiveness, guiding strategies to mitigate their spread.

Despite their widespread use and utility, the application of mathematical models in the COVID-19 pandemic was not without challenges. The accuracy of model predictions depended heavily on the quality and completeness of the input data, which varied significantly across regions due to differences in testing, reporting, and surveillance. Moreover, models often had to make simplifying assumptions about human behavior, healthcare capacity, and other complex factors, which could lead to discrepancies between predictions and observed outcomes. These limitations highlighted the need for continuous refinement and validation of models as new data and insights became available.

Mathematical models have been indispensable in the study and management of the COVID-19 pandemic. They provided a framework for analyzing the spread of the virus, evaluating interventions, planning vaccination strategies, and responding to the challenges posed by new variants. While models are not perfect representations of reality, their ability to synthesize diverse data sources and provide actionable insights has made them a cornerstone of pandemic response efforts. Their use during COVID-19 has not only advanced the field of epidemiological modeling but also demonstrated the vital role of mathematical tools in addressing complex global health challenges.

2.1.3 Overview of model types

2.1.3.1 Stochastic models

Stochastic models are a class of mathematical models that incorporate random variables and probabilistic processes to represent systems where uncertainty and inherent randomness play a significant role. In epidemiology, stochastic models are particularly valuable for capturing the variability and unpredictability of disease spread within populations. Unlike deterministic models, which yield the same outcomes for a given set of initial conditions, stochastic models acknowledge that real-world phenomena are influenced by random events, such as individual variations in behavior, environmental factors, and chance interactions between infectious and susceptible individuals [20].

At the core of stochastic models is the recognition that disease transmission is not a uniform process but is influenced by numerous random factors. For instance, the probability of a susceptible individual contracting an infection depends on their interaction with infectious individuals, which occurs randomly in time and space. Similarly, recovery rates, the effectiveness of interventions, and the emergence of new cases are all subject to inherent variability that deterministic models cannot adequately capture. Stochastic models address these complexities by assigning probabilities to different events and simulating multiple possible outcomes, often using computational methods such as Monte Carlo simulations.

A fundamental feature of stochastic models is their ability to represent the random fluctuations in small populations, where chance events can have a significant impact on disease dynamics. For example, in a small community, a single infectious individual might lead to a rapid outbreak if they interact with many susceptible individuals, or the disease might die out entirely if they recover before spreading the infection. Stochastic models capture this variability by simulating multiple trajectories of disease spread, providing a range of possible outcomes rather than a single deterministic prediction.

One widely used approach in stochastic epidemiological modeling is the stochastic extension of the Susceptible-Infectious-Recovered (SIR) model. In this framework, transitions between compartments (e.g., from susceptible to infectious) occur probabilistically, with rates determined by parameters such as the infection rate and recovery rate. By incorporating randomness, the model generates distributions of possible outcomes rather than fixed values, offering a more realistic depiction of disease dynamics under varying conditions.

Stochastic models are particularly useful for understanding rare events or phenomena that are highly sensitive to initial conditions. For instance, they can estimate the likelihood of an epidemic's extinction in its early stages or predict the probability of a large outbreak given specific conditions. They are also instrumental in evaluating the impact of interventions, such as vaccination campaigns or quarantine measures, by accounting for uncertainties in implementation and population responses.

Despite their advantages, stochastic models come with challenges. They are computationally intensive, requiring repeated simulations to generate reliable predictions. Additionally, their reliance on random variables and distributions demands high-quality data to accurately parameterize the model. The interpretation of stochastic outcomes, often presented as probability distributions or confidence intervals, can also be more complex compared to the straightforward results of deterministic models.

In conclusion, stochastic models are a powerful tool for capturing the inherent randomness and uncertainty in epidemiological processes. Their ability to represent variability and simulate multiple scenarios makes them indispensable for understanding disease dynamics, particularly in small populations or situations where chance events significantly influence outcomes. By providing probabilistic insights, stochastic models enhance our ability to predict, plan, and respond to the unpredictable nature of infectious disease spread.

2.1.3.2 Deterministic models

Deterministic models are a foundational approach in mathematical modeling, characterized by their use of fixed equations and parameters to describe systems where the outcomes are entirely determined by the initial conditions. In epidemiology, deterministic models are extensively used to study the spread of infectious diseases by defining the relationships between key variables, such as the number of susceptible, infectious, and recovered individuals in a population. These models operate on the principle that the same set of initial conditions and parameters will always produce identical results, offering a structured and predictable framework for analyzing disease dynamics [21].

The deterministic approach assumes that populations are homogeneously mixed, meaning every individual has an equal probability of interacting with others. This simplification allows the formulation of equations that describe the rates of change in different compartments of a population over time. For instance, in the classic Susceptible-Infectious-Recovered (SIR) model, the transitions between the compartments are represented by a system of ordinary differential equations (ODEs). These equations capture how the number of individuals in each compartment changes based on parameters such as the transmission rate (β) and recovery rate (γ).

A key strength of deterministic models lies in their analytical clarity and computational efficiency. By providing exact solutions or numerical approximations, these models can quickly generate insights into the overall behavior of an epidemic. For example, deterministic models are commonly used to estimate the basic reproduction number (R_0), a critical threshold parameter indicating the average number of secondary infections caused by a single infectious individual in a fully susceptible population. When $R_0>1$, the disease spreads, whereas $R_0<1$ indicates eventual extinction. This parameter provides a concise summary of the epidemic potential of a disease and informs public health strategies.

Deterministic models are particularly well-suited for large populations, where individual-level variations average out, and aggregate trends dominate. They excel

in scenarios where the focus is on understanding the general trajectory of an epidemic rather than capturing the nuances of stochastic fluctuations. For instance, deterministic models are widely used to simulate the effects of interventions, such as vaccination or social distancing, by incorporating changes in parameters like contact rates or immunity levels.

However, deterministic models have limitations that stem from their simplifying assumptions. The homogeneity assumption, while mathematically convenient, overlooks the complexities of real-world populations, such as heterogeneous contact patterns, demographic differences, and geographic clustering. These models also assume continuous changes in variables, which may not accurately represent discrete events, such as the occurrence of individual infections or recoveries. Additionally, deterministic models are less effective in small populations or early epidemic stages, where random events and individual variability significantly influence outcomes.

Despite these challenges, deterministic models remain a cornerstone of epidemiological research and public health planning. They provide a robust framework for exploring the underlying mechanisms of disease transmission and evaluating the potential impact of control measures. Moreover, deterministic models often serve as a foundation for more complex modeling approaches, including hybrid models that combine deterministic and stochastic elements to address their respective limitations.

In summary, deterministic models offer a systematic and reliable means of studying the dynamics of infectious diseases. Their simplicity, computational efficiency, and ability to produce reproducible results make them an indispensable tool for understanding epidemic behavior, assessing intervention strategies, and informing policy decisions. While they may not capture the full complexity of real-world systems, deterministic models provide essential insights that serve as a foundation for more advanced analyses.

2.1.3.3 Regression models

Regression models are a versatile and widely used class of mathematical models that establish relationships between a dependent variable (response) and one or more independent variables (predictors). In the context of epidemiology, regression models are employed to analyze trends, predict outcomes, and identify factors influencing the spread and impact of infectious diseases. Unlike compartmental models, which simulate disease dynamics using predefined compartments, regression models rely on statistical techniques to derive patterns and predictions directly from observed data. This data-driven approach makes regression models particularly effective for forecasting and understanding complex interactions between variables [22].

At their core, regression models aim to quantify how changes in independent variables influence the dependent variable. For instance, in modeling the spread of COVID-19, the dependent variable could be the number of new daily cases, while independent variables might include mobility patterns, vaccination rates, or weather conditions. By estimating coefficients for each predictor, regression models provide insights into the magnitude and direction of these relationships. For example, a positive coefficient for mobility could indicate that increased movement is associated with higher transmission rates.

Regression models come in various forms, each suited to specific types of data and relationships. The most basic is linear regression, which assumes a straight-line relationship between the dependent and independent variables. Its simplicity and interpretability make it a popular choice for analyzing linear trends and predicting outcomes over short time horizons. However, many epidemiological processes are inherently nonlinear, requiring more advanced models such as polynomial regression, logistic regression, or generalized linear models. Polynomial regression, for instance, extends linear regression by incorporating higher-order terms, allowing it to capture curves and more complex patterns in the data.

One of the strengths of regression models is their adaptability to diverse datasets. They can handle continuous, categorical, or mixed types of data and

incorporate multiple predictors simultaneously. This flexibility enables regression models to account for a wide range of factors influencing disease dynamics, from demographic and socioeconomic variables to public health interventions. Moreover, regression techniques can be extended to include time series analysis, making them suitable for forecasting temporal trends in case numbers, hospitalizations, or deaths.

Regression models also offer robust tools for evaluating the significance and contribution of individual predictors. Statistical measures, such as p-values and confidence intervals, help determine whether a variable has a meaningful impact on the dependent variable. This feature is particularly useful in identifying key drivers of disease spread and assessing the effectiveness of interventions. For example, regression analysis can reveal whether mask mandates significantly reduce transmission rates or if vaccination coverage correlates with a decline in severe cases.

Despite their many advantages, regression models are not without limitations. Their accuracy depends heavily on the quality and completeness of the input data. Missing values, outliers, or biases in data collection can skew results and undermine the reliability of predictions. Additionally, regression models assume that the relationships between variables remain stable over time, which may not hold in dynamic scenarios like a pandemic. Sudden changes, such as the emergence of new virus variants or shifts in public behavior, can disrupt established patterns and reduce model accuracy.

Another challenge lies in the potential for overfitting, particularly when using complex models with many predictors or high-order polynomial terms. Overfitting occurs when a model captures noise in the data rather than the underlying trend, leading to poor generalization to new or unseen data. To mitigate this, techniques such as regularization (e.g., Ridge or Lasso regression) or cross-validation are often employed.

Regression models are powerful tools for analyzing and predicting epidemiological processes. Their data-driven nature, flexibility, and interpretability make them indispensable for understanding the relationships between variables and

forecasting disease trends. While they have limitations, careful data preparation, model selection, and validation can enhance their reliability and applicability, enabling researchers and policymakers to make informed decisions during health crises. Regression models bridge the gap between statistical analysis and practical application, providing actionable insights that are critical for effective public health responses.

2.2 Regression analysis as a modeling tool

2.2.1 Concept of regression analysis

Regression analysis is a statistical technique used to model and analyze the relationships between a dependent variable and one or more independent variables. Its primary objective is to quantify how changes in the independent variables influence the dependent variable, enabling predictions and insights into underlying patterns. At its core, regression analysis seeks to find the best-fit equation that describes the observed data, often minimizing the error between predicted and actual values through methods like the least squares approach [23].

The foundation of regression analysis lies in its ability to capture the direction and magnitude of relationships. For example, in epidemiological contexts, it can estimate how factors such as population density, vaccination rates, or mobility trends affect the spread of infectious diseases. The versatility of regression extends across simple linear models, which assume a straight-line relationship between variables, to more complex forms such as polynomial regression, logistic regression, and multiple regression models. This adaptability allows regression analysis to address a wide array of problems, from straightforward trend estimation to modeling nonlinear and multifactorial phenomena.

2.2.2 Applications of regression in rorecasting

Regression analysis is widely used in forecasting, where the goal is to predict future outcomes based on historical data. Its applications span multiple domains, including epidemiology, economics, and environmental science. In the context of epidemiology, regression models are pivotal for understanding the dynamics of disease outbreaks and projecting future trends [24].

One key approach is time-series regression, where historical data is analyzed to predict future values over a specific time horizon. For instance, a time-series regression model can forecast daily COVID-19 case counts by analyzing past trends, adjusting for seasonal variations, and incorporating external variables such as government interventions or vaccination rollouts.

Another critical application is multivariate regression, where multiple predictors are included to account for the multifactorial nature of a phenomenon. This approach is especially valuable in public health, where variables such as demographics, healthcare capacity, and socioeconomic conditions interact to influence disease outcomes. By capturing these interactions, multivariate regression provides a comprehensive understanding of the factors driving observed trends.

Regression analysis also facilitates scenario modeling, where hypothetical changes in independent variables are simulated to predict their impact on the dependent variable. For example, a regression model could estimate the effect of increasing vaccination rates on the expected number of hospitalizations, helping policymakers evaluate the potential benefits of targeted interventions.

2.2.3 Advantages of regression analysis compared to other methods

Regression analysis offers several distinct advantages that make it a preferred tool for modeling and forecasting in a wide range of fields:

1) Simplicity of implementation

Regression models, particularly linear regression, are relatively easy to implement and understand. They rely on straightforward mathematical principles and require minimal computational resources, making them accessible even to researchers with limited expertise in advanced modeling techniques. Additionally, the wide availability of regression tools in programming libraries such as Scikit-learn and R further simplifies their implementation.

2) Interpretability of results

One of the standout features of regression analysis is its interpretability. Regression coefficients directly indicate the direction (positive or negative) and magnitude of the relationship between predictors and the target variable. For instance, in a linear regression model, the coefficient for a predictor variable shows the expected change in the dependent variable for a one-unit change in the predictor, holding all other variables constant. This clarity enables researchers and decision-makers to derive actionable insights from the model's results.

3) Flexibility

Regression analysis is highly adaptable, accommodating a wide variety of data types and relationships. It can handle continuous, categorical, or binary data, making it suitable for diverse applications. Furthermore, advanced forms of regression, such as polynomial or ridge regression, allow for the modeling of nonlinear relationships and the inclusion of regularization techniques to prevent overfitting.

4) Scalability

Regression models are scalable and can be extended to include additional predictors as needed. This makes them particularly useful in dynamic environments, where new variables or datasets may become relevant over time. For instance, in the evolving

context of a pandemic, regression models can be updated to incorporate new data on virus variants or public health measures.

5) Robustness for exploratory and predictive analysis

Regression analysis is effective both for explanatory purposes, where the goal is to understand relationships between variables, and for predictive tasks, where accurate forecasts are essential. This dual capability enhances its utility across various stages of research and decision-making.

While regression analysis has limitations, such as sensitivity to multicollinearity and reliance on high-quality data, its simplicity, interpretability, and flexibility make it an indispensable tool in modeling and forecasting. These strengths position regression as a critical approach for tackling complex problems, particularly in fields like epidemiology, where understanding and predicting dynamic processes are paramount.

2.3 Linear regression

2.3.1 Mathematical basis of linear regression

Linear regression is a foundational statistical method used to model the relationship between a dependent variable (y) and an independent variable (x) by fitting a linear equation to the observed data. This approach is widely used due to its simplicity, interpretability, and effectiveness in capturing linear relationships. The fundamental idea of linear regression is to determine the best-fit line that minimizes the error between the predicted values ($\hat{y}$)and the actual values (y) in the dataset [25].

The general equation for a simple linear regression model is as follows:

$$y = \beta_0 + \beta_1 \cdot x$$

Where:

– y is the dependent variable (the outcome being predicted),

– x is the independent variable (the predictor),

– b_0 is the intercept (the value of y when x=0),

– b_1 is the slope (the change in y for a one-unit change in x).

The parameters b_0 and b_1 are estimated from the data using the method of least squares, which minimizes the sum of squared errors (residuals) between the observed values and the values predicted by the regression model.

Least Squares Method for Parameter Estimation

The least squares method is the most commonly used approach to estimate the parameters (b_0 and b_1) in a linear regression model. The objective of this method is to minimize the residual sum of squares (RSS), defined as:

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Where:

– y_i represents the actual value of the dependent variable for the i-th observation,

– $\hat{y} = b_0 + b_1 x_i$ is the predicted value of y for the i-th observation,

– n is the total number of observations.

To minimize RSS, partial derivatives of RSS with respect to b_0 and b_1 are set to zero, leading to the following normal equations:

$$b_1 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})\,(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

$$b_0 = \bar{y} - b_1 \bar{x}$$

Where:

– $\bar{x}$ is the mean of the independent variable,

– $\bar{y}$ is the mean of the dependent variable.

These formulas ensure that the regression line minimizes the total squared deviations of the observed values from the predicted values, providing the optimal estimates for b_0 and b_1.

Interpretation of parameters

1. Intercept (b_0): Represents the baseline value of y when x=0. In real-world scenarios, b_0 may or may not have practical significance, depending on whether x=0 is meaningful in the context of the data.

2. Slope (b_1): Indicates the rate of change in y for a one-unit increase in x. The sign of b_1 (positive or negative) reflects the direction of the relationship between x and y.

Assumptions of linear regression

The validity of linear regression relies on several assumptions:

– Linearity: The relationship between x and y is linear.
– Independence: Observations are independent of each other.
– Homoscedasticity: The variance of residuals is constant across all levels of x.
– Normality: The residuals are normally distributed.

When these assumptions are met, linear regression provides reliable estimates and interpretable results. It remains a cornerstone of predictive modeling due to its simplicity and effectiveness in capturing linear relationships in data.

2.3.2 Application of Linear Regression for Forecasting

Interpretation of coefficients

Linear regression coefficients provide a straightforward interpretation of the relationship between the dependent and independent variables. The intercept (b_0) represents the predicted value of the dependent variable (y) when the independent variable (x) equals zero. While the intercept may not always have a practical interpretation, it anchors the regression line on the y-axis.

The slope coefficient (b_1) quantifies the rate of change in y for a one-unit increase in x. A positive slope indicates a direct relationship, where increases in x lead to increases in y, while a negative slope signifies an inverse relationship. For example, in a model predicting COVID-19 cases based on the number of days since the first recorded case, the slope might represent the average daily increase in cases. Understanding these coefficients is critical for deriving actionable insights and communicating results to stakeholders.

Forecasting using time-series data

Linear regression is often applied to time-series data for forecasting purposes. In such cases, the independent variable (x) typically represents time, such as days or weeks, and the dependent variable (y) represents the outcome of interest, such as the number of new cases, hospitalizations, or deaths.

To build a time-series forecast, the regression model is trained on historical data to estimate the relationship between time and the dependent variable. The model can then extrapolate future values by extending the time variable beyond the observed range. For instance, if a linear regression model predicts the progression of COVID-19 cases over time, it can estimate the expected number of cases for future dates based on the established trend.

While linear regression provides a straightforward approach to time-series forecasting, its effectiveness depends on the data exhibiting a consistent linear trend. Deviations from linearity, such as seasonality or sudden shifts, may reduce the

model's accuracy and require additional techniques for preprocessing or incorporating external variables.

2.3.3 Advantages of linear regression

Simplicity of Implementation and Interpretation

Linear regression is one of the most accessible and widely understood statistical methods. Its mathematical simplicity and availability in popular programming libraries, such as Scikit-learn in Python, make it straightforward to implement. Additionally, the clear interpretability of regression coefficients allows practitioners to communicate findings effectively to non-technical audiences. This simplicity makes linear regression an ideal starting point for exploratory analysis and predictive modeling.

Effectiveness in Modeling Linear Relationships

When the relationship between variables is approximately linear, linear regression is highly effective at capturing and quantifying the association. It is computationally efficient, even for large datasets, and can serve as a robust baseline model for more complex scenarios. In many practical applications, such as short-term forecasting or analyzing relationships with limited variability, linear regression provides reliable and meaningful results.

2.3.4 Limitations of linear regression

Inability to model nonlinear relationships

A significant limitation of linear regression is its assumption of linearity between the independent and dependent variables. When the true relationship is nonlinear, the model fails to capture the underlying patterns, leading to inaccurate predictions. For instance, in epidemiological modeling, disease spread often follows

exponential or logistic growth patterns, which cannot be adequately represented by a linear model. To address this limitation, more advanced techniques, such as polynomial regression or nonlinear regression, may be required.

Sensitivity to outliers

Linear regression is highly sensitive to outliers, as the method of least squares assigns equal weight to all residuals but disproportionately penalizes larger deviations. Outliers can significantly distort the regression line and reduce the model's accuracy. For example, a single day with an unusually high spike in reported COVID-19 cases might skew the regression results, affecting forecasts and interpretations. Techniques such as robust regression or data preprocessing are often necessary to mitigate the impact of outliers.

2.4. Polynomial regression

2.4.1 Mathematical basis of polynomial regression

Polynomial regression is an extension of linear regression that allows for modeling nonlinear relationships by incorporating polynomial terms of the independent variable. While linear regression is limited to straight-line relationships, polynomial regression introduces additional flexibility by fitting curves to the data, making it suitable for scenarios where the relationship between variables is not adequately captured by a straight line [26].

The general form of a polynomial regression model is expressed as:

$$y = b_0 + b_1 \cdot x + b_2 \cdot x^2 + \cdots + b_n \cdot x^n$$

Where:

- y is the dependent variable (target or response),
- x is the independent variable (predictor),

- b_0, b_1, $b_2 \ldots b_n$ are the coefficients of the polynomial,
- n is the degree of the polynomial.

In this formulation, the model retains the structure of a linear regression model with respect to the coefficients (b_0, b_1, $b_2 \ldots b_n$) but extends it to include powers of x as additional predictors. This allows the regression equation to capture curvature and more complex patterns in the data.

Generalization for nonlinear dependencies

Polynomial regression generalizes linear regression by expanding the feature set to include higher-order terms of the independent variable. For example:

– A second-degree polynomial regression (quadratic) introduces x^2 alongside x.

– A third-degree polynomial regression (cubic) introduces x^3, and so on.

Each additional term increases the model's capacity to fit more complex patterns, but it also increases the risk of overfitting, especially with higher-degree polynomials.

To apply polynomial regression, the independent variable x is transformed into additional features (x, x^2, x^3 … x^n) before performing a standard linear regression on the expanded feature set. The coefficients are estimated using the least squares method, minimizing the residual sum of squares (RSS) between the observed values (y) and the predicted values ($\hat{y}$).

Benefits of polynomial regression

1. Capturing Nonlinear Relationships: By incorporating higher-order terms, polynomial regression can model curves and other nonlinear trends in the data.

2. Preservation of Linear Framework: Despite modeling nonlinear dependencies, polynomial regression remains a linear model in terms of its parameters, making it computationally efficient and interpretable.

Example: quadratic and higher-degree models

– Second-degree polynomial (quadratic):

$$y=b_0 + b_1x + b_2x^2$$

Captures a single curve, such as a U-shape or inverted U-shape trend.

– Fifth-degree polynomial:

$$y=b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5$$

Captures complex relationships with multiple inflection points, allowing the model to adapt to intricate patterns in the data.

Considerations in polynomial regression

While polynomial regression is powerful for modeling nonlinear relationships, it requires careful attention to model selection. Higher-degree polynomials provide greater flexibility but increase the risk of overfitting, where the model fits the training data perfectly but fails to generalize to new observations. Regularization techniques or cross-validation are often used to mitigate this risk and find the optimal polynomial degree.

2.4.2 Polynomial Regression of the 5th Degree

Features of Construction

A fifth-degree polynomial regression model represents a specific case of polynomial regression where the degree of the polynomial (n) is set to 5.

The process of constructing a fifth-degree polynomial regression model involves the following steps:

1. Feature Transformation: The original independent variable (x) is expanded to include higher-order terms (x_2, $x_3 \ldots x_5$). This transformation creates additional predictors that represent the polynomial features of the data.

2. Model Fitting: A standard linear regression algorithm, such as the least squares method, is applied to the transformed dataset to estimate the coefficients. Despite its nonlinear appearance, the model remains linear in terms of its parameters.

3. Model Selection and Validation: Regularization techniques or cross-validation are used to ensure the model does not overfit the data. Overfitting is a common issue in high-degree polynomial models due to their flexibility, which can result in poor generalization to new data.

Examples of Use in Modeling Complex Dependencies

1. Epidemiological Modeling: In the context of infectious diseases like COVID-19, a fifth-degree polynomial regression can capture complex, nonlinear trends in case numbers or hospitalization rates. For instance, it might model periods of rapid increase, plateaus, and declines within a single framework, reflecting the effects of interventions or behavioral changes.

2. Environmental Studies: In climate data analysis, a fifth-degree polynomial regression might be used to describe fluctuations in temperature or pollutant levels over time, accounting for seasonal and long-term trends.

3. Economic Data: Complex economic indicators, such as stock market trends or GDP growth rates, often exhibit nonlinear patterns that are well-suited to higher-degree polynomial models.

By capturing these intricate dependencies, fifth-degree polynomial regression provides a powerful tool for analyzing and forecasting datasets with significant variability [27].

2.4.3 Advantages of polynomial regression

Ability to model nonlinear and complex relationships

One of the primary advantages of polynomial regression is its capacity to model nonlinear dependencies between variables. Unlike linear regression, which assumes a straight-line relationship, polynomial regression can fit curves, peaks, and troughs, making it suitable for datasets with complex patterns. For example, in epidemiological studies, where disease spread may follow nonlinear trajectories, polynomial regression can more accurately capture the progression of an outbreak.

The ability to represent such complexities is particularly valuable when the underlying relationship between variables cannot be adequately described by simpler models. This flexibility allows researchers to explore relationships that may otherwise remain hidden in the data.

Flexibility in adapting to trends

Polynomial regression is highly adaptable, enabling the inclusion of additional polynomial terms as needed to fit evolving trends in the data. This makes it particularly effective in dynamic environments where the relationships between variables change over time. For instance:

- In time-series data, polynomial regression can model long-term trends while accounting for short-term fluctuations.
- In multidimensional datasets, it can represent interactions between variables by combining polynomial features with additional predictors.

This adaptability also extends to domain-specific applications, where custom polynomial degrees can be selected to balance model complexity and predictive

accuracy. The ability to tailor the model to specific trends and patterns provides researchers and practitioners with a versatile tool for analysis and forecasting.

2.4.4 Limitations of Polynomial Regression

Tendency to overfit (overfitting), especially for high degrees

A significant drawback of polynomial regression is its susceptibility to overfitting, particularly when using high-degree polynomials. Overfitting occurs when the model becomes too complex, fitting the noise in the data rather than the underlying trend. While this may result in a near-perfect fit for the training data, the model's generalization performance on unseen data often deteriorates. For example, a fifth-degree polynomial might create exaggerated curves to accommodate outliers, leading to unrealistic predictions for new inputs.

To mitigate overfitting, techniques such as cross-validation, regularization (e.g., Ridge or Lasso regression), or restricting the degree of the polynomial are commonly employed.

Complicated interpretation of coefficients

As the degree of the polynomial increases, the number of coefficients grows, complicating their interpretation. Unlike linear regression, where each coefficient has a straightforward and intuitive meaning, the higher-order terms in polynomial regression are less directly tied to specific features of the data. For instance, understanding the role of a fifth-degree term (b_5x^5) in shaping the model's predictions may not be immediately apparent, especially to non-technical stakeholders.

This complexity can make polynomial regression less suitable for applications where interpretability is a primary concern, such as policy-making or exploratory analysis.

Increased computational complexity

Higher-degree polynomial regression involves a larger number of terms and coefficients, leading to increased computational demands. As the degree of the polynomial rises, the process of fitting the model becomes more resource-intensive, particularly for large datasets. This complexity not only affects computational efficiency but also introduces challenges in optimizing and validating the model.

Moreover, the inclusion of high-degree terms may exacerbate multicollinearity, a condition where predictor variables are highly correlated. This can destabilize the model and make coefficient estimates unreliable, further complicating analysis and prediction.

2.5 Comparison of linear and 5th degree polynomial regression

2.5.1 Criteria for comparison

Model Simplicity

Linear regression is simpler in both formulation and computation. It requires estimating only two parameters—the intercept (b_0) and slope (b_1) making it computationally efficient and easy to implement. In contrast, a 5th degree polynomial regression involves estimating six coefficients (b_0, b_1…b_5), resulting in increased complexity and higher computational demands.

Interpretation of results

Linear regression offers straightforward interpretability, where the slope (b_1) represents the rate of change in the dependent variable for a one-unit change in the independent variable. Polynomial regression, especially at higher degrees, complicates interpretation as the coefficients of higher-order terms (b_2x^2, b_3x^3…b5x5) are less intuitive and harder to relate to real-world phenomena.

Prediction accuracy for short- and long-term periods

Linear regression performs well for short-term predictions when the relationship between variables remains linear. However, its performance declines for data with nonlinear trends or longer-term projections. Polynomial regression, particularly of the 5th degree, excels at capturing nonlinear patterns and can provide more accurate short-term predictions for complex datasets. However, its accuracy for long-term forecasting may suffer due to overfitting or excessive sensitivity to minor fluctuations in the training data.

Robustness to outliers

Linear regression is moderately robust to outliers, as it minimizes the overall squared error without being overly influenced by individual data points. In contrast, 5th degree polynomial regression is highly sensitive to outliers, as the inclusion of higher-order terms can amplify the effect of extreme values, leading to distorted predictions.

2.5.2 Advantages of linear regression over polynomial regression

Lower risk of overfitting

Linear regression has a minimal tendency to overfit, as it captures only the general linear trend in the data. This makes it more reliable for datasets with limited variability or when generalization to new data is a priority.

Ease of integration into automated systems

Linear regression's simplicity and computational efficiency make it easier to implement in automated workflows or real-time applications. Its straightforward mathematical structure requires fewer resources and less tuning, enhancing its usability in practical scenarios.

Consistent predictive accuracy for linear trends

When the data follows a clear linear pattern, linear regression provides reliable and accurate predictions. Adding complexity through polynomial terms in such cases offers no significant improvement and may introduce unnecessary noise.

2.5.3 Advantages of 5th degree polynomial regression model over linear regression

Capability to model complex trends

The 5th degree polynomial regression model can accommodate intricate relationships, including curves, peaks, and multiple inflection points. This makes it ideal for datasets with pronounced nonlinear patterns, where linear regression would fail to capture the underlying dynamics.

Adaptation to nonlinear changes in data

Higher-degree polynomial models are better suited to scenarios where the relationship between variables changes over time or exhibits nonlinearity. For example, in epidemiological modeling, they can reflect the fluctuating nature of disease spread influenced by interventions, seasonal effects, or changing population behaviors.

2.5.4 Balancing complexity and predictive accuracy

The choice between linear and polynomial regression involves a trade-off between model complexity and predictive accuracy. Linear regression is preferred for its simplicity, interpretability, and robustness, particularly when the relationship between variables is straightforward. However, its limitations become apparent when dealing with nonlinear data or trends.

The 5th degree polynomial regression model, while powerful, introduces significant complexity. It is prone to overfitting, especially when the degree of the polynomial exceeds the inherent complexity of the data. This can result in excellent performance on training data but poor generalization to new datasets. Regularization techniques, such as Ridge or Lasso regression, and cross-validation are essential to mitigate these risks and ensure the model remains both accurate and practical.

Linear regression is a reliable choice for simple, linear relationships or when interpretability and ease of use are paramount. In contrast, 5th degree polynomial regression excels in capturing complex patterns and nonlinear dependencies but requires careful handling to avoid overfitting and maintain predictive accuracy. The decision between the two approaches depends on the nature of the data, the specific goals of the analysis, and the acceptable level of model complexity.

2.6 Adaptability of models to dynamic conditions

2.6.1 Use of models in scenarios with variable parameters

Dynamic systems, such as the progression of pandemics or evolving economic trends, often involve variables that change over time. Adapting regression models to such scenarios requires flexibility in incorporating time-sensitive data and adjusting parameters dynamically. In epidemiology, for example, factors such as the rate of

transmission, vaccination coverage, and public compliance with health measures can shift rapidly, influencing the course of disease spread.

To address these variations, models must be designed to integrate new information and account for changing parameters. Polynomial regression, for instance, can capture nonlinear trends over time by introducing time as a higher-order term. Similarly, time-varying regression models allow coefficients to evolve, reflecting the dynamic influence of predictors on outcomes. Techniques such as rolling regression, where the model is recalibrated over a moving window of recent data, are particularly useful in capturing short-term changes while maintaining focus on overarching trends.

2.6.2 Importance of regular model updates with new data

The accuracy and relevance of regression models depend heavily on the quality and timeliness of the input data. In dynamic contexts, static models that rely on outdated information are prone to becoming obsolete, leading to inaccurate predictions and suboptimal decision-making. Regular updates are essential to ensure that models reflect current realities.

For instance, during the COVID-19 pandemic, incorporating data on emerging variants, vaccination progress, and shifting public behaviors allowed models to provide more accurate forecasts. Automated pipelines for data ingestion and model retraining are critical in scenarios where data availability is continuous and real-time decisions are necessary. Techniques such as online learning, where models are incrementally updated as new data arrives, help maintain adaptability without requiring complete retraining from scratch.

Regular updates also help mitigate issues such as model drift, where the relationship between predictors and the target variable changes over time. Continuous monitoring of model performance, coupled with periodic recalibration, ensures that predictions remain reliable and actionable.

2.6.3 Extending models for multifactor analysis

Real-world systems are often influenced by a multitude of interconnected factors, requiring models that can handle multiple predictors simultaneously. Extending regression models to include additional variables enhances their explanatory power and predictive accuracy, particularly in complex scenarios. For example, in epidemiology, factors such as population density, healthcare capacity, environmental conditions, and socioeconomic status can all interact to shape disease dynamics.

Multivariate regression models are particularly suited for such analyses, allowing the inclusion of numerous predictors and their interactions. Polynomial regression can also be extended to multifactor analysis by incorporating interaction terms and higher-order effects of multiple variables. These extensions enable a more nuanced understanding of how variables jointly influence outcomes, supporting more comprehensive decision-making.

Moreover, techniques such as stepwise regression or regularization (e.g., Ridge or Lasso regression) can help identify the most significant predictors and reduce overfitting when dealing with a large number of variables. Incorporating domain knowledge and expert input during model development further ensures that the chosen factors are relevant and interpretable.

2.7 Conclusions on theoretical aspects of regression modeling

2.7.1 Advantages and limitations of regression models for predicting epidemiological processes

Regression models play a pivotal role in predicting and analyzing epidemiological processes, offering significant advantages that make them indispensable tools in public health planning and response. Their primary strength

lies in their ability to quantify relationships between variables and provide interpretable results. Linear regression, for example, is straightforward to implement and understand, making it ideal for scenarios where relationships between variables are simple and linear. Polynomial regression extends this capability, allowing for the modeling of more complex and nonlinear dependencies, which are often present in epidemiological datasets.

Regression models are highly flexible, enabling the inclusion of multiple predictors and their interactions. This adaptability is crucial for capturing the multifactorial nature of disease dynamics, where variables such as population density, healthcare capacity, and public behavior interact in complex ways. Additionally, regression techniques are computationally efficient and can be applied to large datasets, making them suitable for real-time analysis during health crises.

However, regression models also have limitations. Linear regression is constrained by its assumption of linearity, which may oversimplify the relationships in complex systems. Polynomial regression, while addressing nonlinearity, introduces risks such as overfitting, especially for high-degree models. Overfitted models may perform well on training data but fail to generalize to new or unseen data, reducing their predictive reliability. Furthermore, the interpretation of coefficients becomes increasingly challenging as the model complexity grows, limiting the practical utility of more intricate regression formulations.

Another limitation is the sensitivity of regression models to the quality of input data. Missing values, outliers, and biases in data collection can significantly affect model performance. In dynamic conditions, where relationships between variables may change over time, regression models require frequent updates to remain relevant, adding to the computational and logistical demands.

2.7.2 Importance of a balanced approach to model selection based on objectives and data characteristics

The choice of an appropriate regression model depends on the specific objectives of the analysis and the characteristics of the data. A balanced approach that considers both the complexity of the problem and the practical constraints of modeling is essential to maximize the utility of regression techniques.

For tasks where relationships are linear or where simplicity and interpretability are paramount, linear regression is often the preferred choice. Its robustness, computational efficiency, and ease of implementation make it suitable for exploratory analyses and short-term predictions. In contrast, polynomial regression is better suited for datasets with clear nonlinear trends or where capturing intricate patterns is critical. However, careful consideration must be given to the degree of the polynomial to avoid overfitting and ensure the model remains generalizable.

In dynamic scenarios, regular updates and validation are crucial to maintaining the accuracy and reliability of regression models. Incorporating mechanisms for handling variable parameters, such as time-varying coefficients or adaptive modeling techniques, enhances the model's responsiveness to changing conditions. Additionally, the integration of domain knowledge during model development can guide the selection of relevant predictors, reducing the risk of overfitting and improving interpretability.

The use of advanced techniques such as regularization or ensemble methods can help strike a balance between model complexity and predictive accuracy. These approaches mitigate the limitations of both linear and polynomial regression, ensuring that the chosen model aligns with the specific needs of the analysis while remaining practical and effective.

Regression models offer a powerful framework for understanding and predicting epidemiological processes, with their strengths lying in flexibility,

computational efficiency, and interpretability. However, their limitations, including susceptibility to overfitting and sensitivity to data quality, highlight the need for a thoughtful and balanced approach to model selection. By aligning the model choice with the objectives and characteristics of the data, and by incorporating strategies to enhance adaptability, regression models can provide reliable and actionable insights, supporting effective decision-making in public health and beyond.

3 PRACTICAL CHAPTER

The COVID-19 pandemic has been one of the most significant global challenges of the 21st century, affecting millions of lives and disrupting economies worldwide. The unpredictable nature of the virus, combined with its rapid spread, has highlighted the critical need for accurate forecasting to guide public health decisions and allocate resources effectively. Understanding the trends and dynamics of COVID-19 cases is essential for mitigating its impact, especially in planning medical infrastructure, implementing preventive measures, and preparing for potential future outbreaks.

Forecasting the progression of a pandemic requires robust analytical tools capable of processing large datasets and identifying patterns. Traditional methods alone are insufficient to manage the complexity of such vast and dynamic data. This is where the power of programming and machine learning comes into play. By leveraging programming languages such as Python and libraries like Scikit-learn and Pandas, researchers can build predictive models that analyze historical data and provide reliable forecasts.

The importance of integrating programming into solving pandemic-related challenges cannot be overstated. Programming enables automation, scalability, and precision, making it possible to handle the intricate details of epidemiological data. In this project, we aim to utilize programming techniques to construct and evaluate predictive models for COVID-19 case trends. This work not only demonstrates the practical application of programming but also emphasizes its role as a critical tool in addressing global health crises.

3.1 Selection of a country for analysis

I chose France as the country for my research. France was chosen for this project for several important reasons. First, the country has one of the most advanced healthcare systems in the world, ensuring the high accuracy and reliability of statistical data. This is crucial for building a regression model, as the quality of the data directly impacts the accuracy of the forecasts. Moreover, France actively publishes detailed statistics on the COVID-19 pandemic, including data on cases, testing, and vaccination.

The COVID-19 pandemic has significantly affected France, making it a compelling case study. The country's diverse regions—ranging from densely populated urban areas to sparsely populated rural regions—provide an opportunity to analyze various scenarios of virus spread. This also enables the inclusion of social, economic, and demographic factors that influence the dynamics of the pandemic.

France combines accessible high-quality data, a significant impact from the pandemic, and broad opportunities to explore various aspects of COVID-19 spread and response.

3.2 Collection of covid-19 statistics

According to the assignment for the course work, we collect statistical information on the incidence of COVID-19 from the first case to September 30, 2024. We use [28] and [29] as sources of information.

The COVID-19 Dashboard by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (JHU) is a widely recognized tool that was developed to track the global outbreak of the COVID-19 pandemic. The dashboard provides real-time updates and visualizations of COVID-19 data, offering a centralized resource for researchers, policymakers, and the public to understand the spread and impact of the virus.

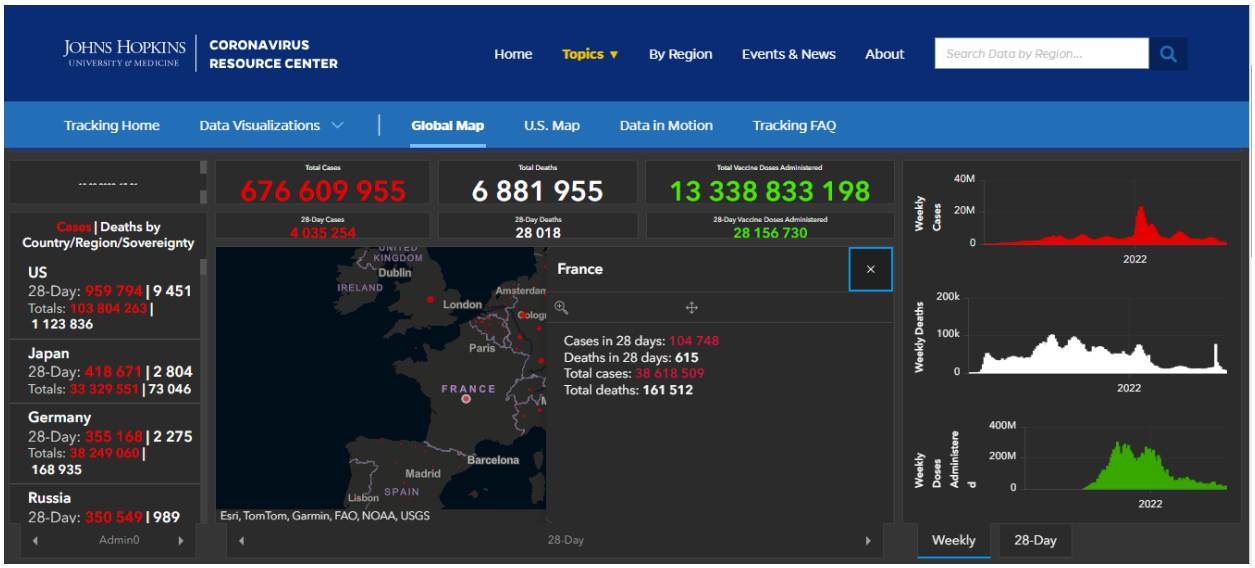

Figure 1 – The COVID-19 Dashboard by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (JHU) [1]

As of March 10, 2023, the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University has stopped updating data on its COVID-19 Dashboard. This decision was made due to a decrease in the frequency of pandemic reporting by many US states and improved data tracking by the federal government.

We use the following table as a starting point for morbidity statistics [30].

Next, we write a Python script that selects the statistics of Covid-19 cases in the selected country for the entire period for which the statistics are collected from the owid-covid-data.csv file.

The name of the file is France_total_statistics.csv. This is an intermediate statistics file that contains data up to 2024-08-04, which is contained in the table owid-covid-data.csv. The script is saved in Appendix 1.

After that, you need to add statistics until 2024-09-30. We use the website [29] as a source of information. On this resource, we find a graph of the incidence of Covid-19 in France for 2024 and use the data on the increase in the incidence. The graph is shown in Figure 2. We take this data into account by supplementing the file France_total_statistics.csv and save it under the name France_total_amount.csv.

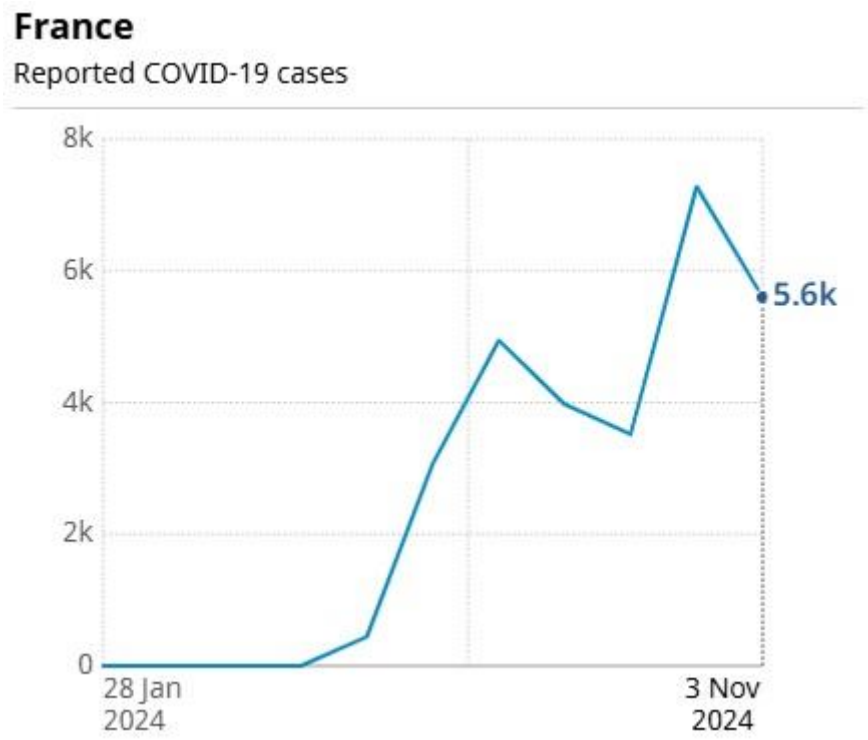

Figure 2 – Reported COVID-19 cases from WHO COVID-19 dashboard [21]

In a separate file, called cases_per_year.cvs, we collect statistics on the incidence of COVID-19 from January 1, 2024, to September 30, 2024.

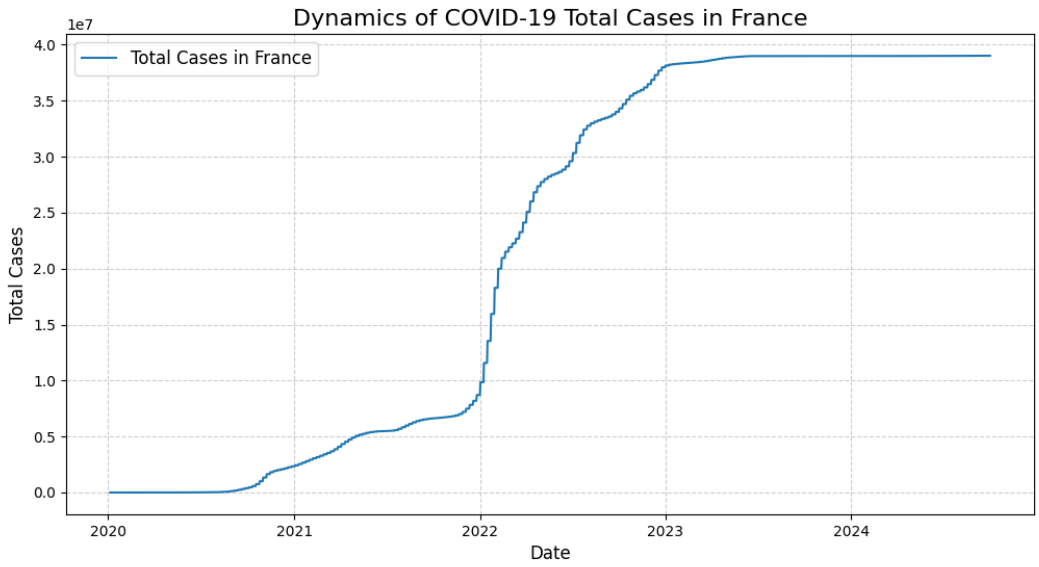

Figure 3 – Dynamics of COVID-19 Total Cases in France

3.3 Development and implementation of a regression model

3.3.1 The importance of mathematical modeling in data analysis

Mathematical modeling is a fundamental approach in data analysis that enables researchers and analysts to describe, understand, and predict complex real-world phenomena using mathematical structures and algorithms. The process of creating a mathematical model involves defining relationships between variables, identifying patterns, and constructing equations or algorithms that approximate the behavior of the system under study.

One of the primary benefits of mathematical modeling is its ability to simplify complex systems. Real-world problems often involve multiple interacting variables, making them challenging to analyze directly. Mathematical models abstract these complexities into manageable forms, providing a structured framework for interpretation and decision-making.

In data analysis, mathematical modeling plays a crucial role in uncovering trends and patterns hidden within datasets. Models such as regression, classification, and clustering are widely used to derive insights, predict future outcomes, and optimize processes. For example, regression models are employed to forecast economic growth, assess the spread of diseases, and predict consumer behavior.

Another advantage of mathematical modeling is its capacity to generalize findings. A well-designed model can be applied to similar datasets or scenarios, making it a powerful tool for extrapolation and hypothesis testing. Furthermore, it allows analysts to simulate various scenarios, enabling decision-makers to evaluate potential outcomes before implementing policies or interventions.

Mathematical modeling is a cornerstone of modern data analysis. It bridges the gap between theoretical understanding and practical application, enabling the transformation of raw data into actionable insights. As data continues to grow in complexity and volume, the importance of mathematical modeling in driving innovation and informed decision-making remains indispensable.

3.3.2 The role of regression models in forecasting

Regression models are among the most widely used tools in forecasting, offering a systematic way to predict outcomes based on historical data. By identifying and quantifying the relationships between dependent and independent variables, regression models provide insights into how changes in certain factors influence the target variable. This makes them invaluable in a wide range of fields, from economics and healthcare to marketing and engineering.

At the core of regression models is the ability to capture trends and patterns in data. For instance, linear regression models analyze the linear relationship between variables, making them suitable for predicting outcomes with a relatively stable trend. On the other hand, more complex regression models, such as polynomial or logistic regression, are better suited for capturing nonlinear relationships or categorical outcomes.

In forecasting, regression models excel in their ability to generate precise numerical predictions. They are particularly effective when the relationship between variables is well-defined and the dataset is clean and representative. For example, in the context of epidemiology, regression models are commonly used to predict the spread of diseases based on factors such as population density, mobility patterns, and vaccination rates.

Another critical role of regression models in forecasting is their interpretability. Unlike many machine learning models, regression models provide clear, interpretable coefficients that indicate the magnitude and direction of the impact of each predictor variable on the target variable. This transparency helps decision-makers understand the drivers behind the forecast and supports evidence-based policy-making.

Regression models are also highly versatile and scalable. They can be adapted to incorporate additional variables or constraints, making them applicable to a variety of forecasting tasks. Furthermore, advancements in computational tools,

such as the scikit-learn library, have made it easier to implement regression models with high efficiency and accuracy.

Regression models are a cornerstone of modern forecasting techniques. Their ability to capture relationships, provide interpretable results, and adapt to diverse datasets makes them an essential tool for predicting future trends and enabling informed decision-making.

3.3.3 Basics of regression analysis: concept and applications

Regression analysis is a statistical technique used to model and analyze the relationships between a dependent variable (target) and one or more independent variables (predictors). The primary objective of regression analysis is to identify how changes in the independent variables influence the dependent variable, enabling predictions and insights about future outcomes.

Regression analysis has broad applicability across various domains:

1. Economics: Regression is extensively used to forecast economic indicators such as GDP growth, inflation, or unemployment rates based on factors like consumer spending, interest rates, and trade balances.
2. Healthcare: In epidemiology, regression models help analyze the spread of diseases by correlating infection rates with factors like population density, vaccination coverage, and mobility patterns.
3. Marketing: Businesses use regression to predict customer behavior, such as purchasing decisions, based on demographic and psychographic data.
4. Environmental Science: Regression is applied to study climate change impacts by analyzing relationships between carbon emissions, temperature changes, and deforestation.

Regression is not limited to linear relationships. Advanced regression techniques, such as polynomial regression, logistic regression, and multivariate regression, address more complex scenarios where relationships between variables are nonlinear or involve categorical outcomes.

One of the key strengths of regression analysis is its interpretability. Unlike more complex machine learning models, regression provides clear coefficients that indicate the strength and direction of relationships between variables. This makes it a valuable tool for both predictive modeling and explanatory analysis.

3.3.4 Types of regression models

Regression models come in various forms, each designed to address specific types of relationships between variables. While the fundamental goal remains the same—to model the relationship between a dependent variable and one or more independent variables—different models are used depending on the nature of the data and the underlying patterns.

Linear Regression

Linear regression is the simplest and most widely used regression model. It assumes a linear relationship between the dependent variable y and one or more independent variables x. The equation for a simple linear regression is:

$$y = \beta_0 + \beta_1 \cdot x + \varepsilon$$

Here, y is the dependent variable, x is the independent variable, β_0 is the intercept, β_1 is the slope of the line, and ε is the error term accounting for deviations not explained by the model.

Linear regression is highly interpretable and effective when the relationship between variables is approximately linear. It is commonly used in economics, healthcare, and social sciences for tasks like trend analysis and forecasting.

Polynomial Regression

Polynomial regression extends linear regression by modeling the relationship between the dependent and independent variables as a polynomial. The equation takes the form:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_n x^n + \epsilon$$

This model is particularly useful for capturing nonlinear relationships, where a straight line is insufficient to describe the data. For example, polynomial regression is often used in physics, biology, and environmental sciences to model phenomena with curves or peaks.

Other types of regression models

Logistic regression:

Logistic regression is used when the dependent variable is categorical (e.g., binary outcomes like success/failure or yes/no). Instead of modeling a direct relationship, it predicts the probability of an event occurring. Logistic regression is widely applied in fields such as medicine, marketing, and fraud detection.

Multiple linear regression:

Multiple linear regression models the relationship between one dependent variable and multiple independent variables. This approach captures the combined influence of multiple factors on the target variable, making it valuable for analyzing complex systems.

Ridge and lasso regression:

These are regularized forms of regression designed to prevent overfitting in datasets with many predictors. Ridge regression adds a penalty term for large coefficients, while Lasso regression can shrink some coefficients to zero, effectively performing variable selection.

Support vector regression (SVR):

A more advanced technique that uses support vector machines to predict continuous outcomes. SVR is effective in handling nonlinear relationships and outliers.

Decision tree and ensemble regression:

Decision tree regression models the data by splitting it into branches based on decision rules. Ensemble methods like Random Forest and Gradient Boosting combine multiple trees to improve accuracy and robustness.

I chose a polynomial regression model because it allows modeling nonlinear relationships that are characteristic of the dynamics of COVID-19 spread. Unlike linear regression, which assumes a straight-line relationship between variables, polynomial regression provides the ability to account for complex trends and patterns, such as sharp increases, plateaus, or declines that occur at different stages of an epidemic. In the case of COVID-19 dynamics, which depend on various factors (government measures, seasonal changes, population behavior), this approach enables a more accurate representation of the real situation and the prediction of its development. Polynomial regression is the optimal choice as it ensures flexibility in capturing complex relationships without significantly complicating the model itself.

I chose a 5th-degree polynomial because it strikes a balance between model accuracy and avoiding overfitting. A 5th-degree polynomial allows the model to account for more complex nonlinear relationships compared to lower-degree models, such as the 2nd or 3rd degree, while avoiding excessive complexity that might arise from using higher degrees like the 6th ... 10th. In COVID-19 dynamics, there are often both gradual and sharp changes that need to be captured for precise forecasting. The higher-degree terms, such as x^4 and x^5, make it possible to model these complex trends while keeping the model straightforward to use.

3.3.5 Theoretical foundations of 5th Degree Polynomial Regression

Polynomial regression is an extension of linear regression that models the relationship between the independent variable x and the dependent variable y as a polynomial equation. This approach allows capturing nonlinear patterns in data, making it particularly useful for situations where the relationship between variables is not well-represented by a straight line. The general form of a polynomial regression equation of degree n is:

$$y = b_0 + b_1 \cdot x + b_2 \cdot x^2 + \cdots + b_n \cdot x^n + \varepsilon$$

Here, y is the dependent variable, x is the independent variable,

b_0, b_1, $b_2 \ldots b_n$ are the coefficients of the polynomial terms, and ε represents the error term.

For a 5th-degree polynomial regression model, the equation becomes:

$$y = b_0 + b_1 \cdot x + b_2 \cdot x^2 + b_3 \cdot x^3 + b_4 \cdot x^4 + \varepsilon$$

The coefficients $b_1 \ldots b_5$ capture the contribution of each polynomial term to the dependent variable, while b_0 represents the intercept, or the value of y when x=0.

The key idea behind polynomial regression is to transform the input feature x into a set of polynomial features: x, x^2, x^3, x^4, x^5. These features are then used in a linear regression model to learn the optimal coefficients that minimize the error between the predicted and actual values. This approach allows polynomial regression to retain the simplicity of linear regression while being able to model complex nonlinear relationships.

The 5th-degree polynomial regression model is particularly effective for datasets where the relationship between the variables exhibits significant nonlinearity. It can capture intricate patterns, such as rapid growth, oscillations, or

flattening trends, that simpler models like linear regression or lower-degree polynomials may fail to represent. Additionally, it provides a balance between model complexity and the risk of overfitting, as it incorporates sufficient flexibility without becoming overly sensitive to noise in the data.

3.3.6 Choosing a library for implementation

For the implementation of the 5th-degree Polynomial Regression Model in this project, the Python programming language was selected due to its versatility, extensive ecosystem of libraries, and ease of use for data analysis and machine learning tasks. Within Python, the scikit-learn library was chosen as the primary tool for building and evaluating the regression model. This decision was based on several key factors that make scikit-learn an ideal choice for this task.

Scikit-learn is a widely used machine learning library that provides efficient and user-friendly tools for implementing regression models, including polynomial regression. Its modular structure and built-in functions for data preprocessing, feature transformation, and model evaluation significantly simplify the workflow of building machine learning models. Specifically, the PolynomialFeatures class from scikit-learn allows for the generation of polynomial terms from input data, which are essential for polynomial regression. Additionally, the LinearRegression class offers a straightforward way to fit a regression model using these polynomial features [35].

Another advantage of scikit-learn is its integration with other Python libraries commonly used for data analysis, such as numpy and pandas. These libraries enable efficient handling and manipulation of large datasets, making them well-suited for tasks involving COVID-19 case data, which may include thousands of observations. Furthermore, scikit-learn supports model validation and evaluation through functions like mean_squared_error and cross_val_score, which are critical for assessing the accuracy and reliability of the regression model.

The choice of scikit-learn was also influenced by its comprehensive documentation and active community support. The library provides clear and detailed documentation for all its functions and classes, ensuring that even complex implementations can be performed with relative ease. The active community around scikit-learn ensures that common issues and questions are well-documented and resolved quickly, further reducing development time [35].

Alternatives to scikit-learn were considered, such as TensorFlow and PyTorch, which are powerful frameworks for machine learning and deep learning tasks. However, these libraries are more suited for complex models like neural networks and may add unnecessary overhead for a relatively straightforward regression task. Similarly, libraries such as statsmodels were reviewed, but they lack the streamlined functionality and seamless integration with modern data pipelines that scikit-learn offers.

Scikit-learn was chosen for its robust feature set, ease of use, and compatibility with other essential data analysis tools in Python. Its ability to handle polynomial feature generation, linear regression, and model evaluation in an integrated and efficient manner makes it the optimal choice for implementing the 5th-degree Polynomial Regression Model in this project. This selection ensures that the project can achieve its objectives effectively while maintaining clarity and reproducibility in the implementation process.

3.3.7 Steps for building a regression models

In this project, I plan to build regression models using Python and the scikit-learn library. The process involves transforming features into polynomial terms, training the models on these transformed features, and saving the trained models for future use. The scikit-learn library will serve as the primary tool for this implementation due to its robust functionality and ease of use.

To begin, I will use the PolynomialFeatures class from scikit-learn to transform the input data into polynomial features. This transformation will expand the feature set by including nonlinear terms such as x^2, x^3, x^4 and x^5. The degree of the polynomial will be set to 5, which provides a balance between model flexibility and the risk of overfitting. This step ensures that the models can effectively capture complex trends in the dynamics of COVID-19 cases.

Once the polynomial features are generated, I will proceed to train the models using the LinearRegression class. The training process involves optimizing the coefficients of the polynomial terms to minimize the mean squared error (MSE) between the predicted and actual values. During training, I will divide the process by datasets, training one model on the complete dataset and another on the data specific to 2024. This separation will allow each model to specialize in capturing trends relevant to its respective dataset.summa

After training, I plan to evaluate the performance of the models using metrics such as MSE to ensure they accurately represent the underlying data patterns. Any adjustments needed to improve performance will be incorporated at this stage.

The final step will involve saving the trained models in .pkl format using the joblib library. This format facilitates efficient storage and quick retrieval of the models for later use in prediction or analysis. The saved models will be ready for deployment in subsequent stages of the project.

3.3.8 Creating a script for models preparation

Then I wrote a script to create the model. The script (name it “create_polynomial_models.py”) is designed to construct a regression model for predicting COVID-19 cases based on historical data.

The script create_polynomial_models.py is designed to generate two distinct 5th-degree polynomial regression models based on COVID-19 data for France. The first model is trained on the complete dataset, which contains statistics from the

beginning of the pandemic up to September 30, 2024. The second model is trained exclusively on the data for the year 2024, specifically from January 1 to September 30, 2024. Both models are saved in separate files for later use.

The script starts by importing essential libraries such as pandas for data handling, scikit-learn for creating and training the regression models, and joblib for saving the trained models to files. File paths for the input datasets and the output models are defined, ensuring clarity and modularity in handling data. The full dataset is loaded from the file France_total_amount.csv, which contains columns for dates and total COVID-19 cases. Similarly, the 2024 dataset is loaded from the file case_per_year.csv. Both datasets are checked for missing values to ensure data quality before modeling.

The data preparation involves using the row index as the input feature (representing the sequence of days) and the column total_cases as the target variable (representing the total number of cases). For the complete dataset, the model uses all available data to capture trends across the entire pandemic. For the 2024 dataset, only data from the specified year is used, allowing the model to focus exclusively on more recent trends.

Two separate polynomial regression models are constructed using a pipeline that combines polynomial feature generation with linear regression. Both models are trained on their respective datasets. After training, the models are saved in .pkl files. The model trained on the full dataset is saved as polynomial_model.pkl, while the model trained on the 2024 dataset is saved as polynomial_model_2024.pkl. The script provides feedback in the console, indicating the successful creation and saving of each model.

This script is structured to be efficient and reusable. By saving the trained models, it eliminates the need for retraining during subsequent analyses, making it ideal for deployment or further studies. The saved models can be directly loaded and used for prediction tasks, ensuring flexibility in analyzing different datasets. The script encapsulates all steps from data preprocessing to model training and saving,

making it a comprehensive tool for regression modeling in this project. The script is shown in Appendix 2.

The model polynomial_model.pkl consists of two main steps: polynomialfeatures and linearregression. The first step, polynomialfeatures, generates polynomial features from the input data, allowing the model to account for nonlinear relationships. In this case, the polynomial degree is 5. The second step, linearregression, performs linear regression on the generated polynomial features to determine the optimal coefficients that minimize the difference between the predicted and actual values. The model uses a 5th-degree polynomial, which means it considers dependencies up to the fifth power of the input data. For example, for a variable x, the model generates features such as x^0, x^1, x^2, x^3, x^4, x^5. This approach allows the model to capture complex nonlinear relationships in the data.

The regression coefficients are: [0.00, 44438.12, −265.97, 0.59, −0.00044, 0.0000001]. These coefficients correspond to the influence of each polynomial feature (from x^1 to x^5) on the prediction. The coefficient for x^1 is 44438.12, representing the linear dependency and showing the largest contribution, which indicates a strong linear relationship with time. The coefficient for x^2 is −265.97, representing the quadratic contribution, and the negative value indicates that this term counteracts linear growth. The coefficient for x^3 is 0.59, representing a smaller cubic contribution. The coefficients for x^4 and x^5 are −0.00044 and 0.0000001, respectively, showing minimal contributions from these higher-degree terms. The coefficient for x^0 (the constant term) is 0, as the model focuses on relative changes.

The intercept of the linear regression is −1561974.77. This value represents the model's prediction when all input features are zero. Although the negative value does not have a meaningful physical interpretation in the context of COVID-19 total cases, it is typical in polynomial regression models with a high degree. This reflects the mathematical fitting process rather than the actual real-world trend.

The polynomial_model_2024.pkl model consists of two main steps: polynomialfeatures and linearregression. The first step, polynomialfeatures, generates polynomial features from the input data to account for nonlinear relationships. In this case, the polynomial degree is 5, meaning the model considers dependencies up to the fifth power of the input data. For a variable x, the model generates features such as x^0, x^1, x^2, x^3, x^4, x^5. This setup allows the model to capture complex nonlinear trends specific to the 2024 dataset.

The regression coefficients are:

[0.0,118.625756, −3.39171326,0.0330769287, −0.000119108986, 0.000000154765885]. These coefficients represent the contribution of each polynomial feature to the model's prediction. The coefficient for x^1 is 118.625756, indicating a strong linear dependency and serving as the most significant contributor to the predictions. The coefficient for x^2 is −3.39171326, showing a negative quadratic influence that slightly offsets the linear trend. The coefficient for x^3 is 0.0330769287, representing a minimal cubic effect. Higher-degree terms (x^4 and x^5) have coefficients of −0.000119108986 and 0.000000154765885 respectively, indicating negligible contributions from these terms.

The intercept of the linear regression is 38996615. This value represents the model's prediction when all input features are zero. In the context of the COVID-19 dataset for 2024, this high intercept likely reflects the overall scale of the data, where the total number of cases is already significant even at the start of the year. This model is tailored to recent trends in the 2024 dataset and reflects the dynamics of the data with a focus on capturing short-term nonlinear patterns.

Next, I created a script to test the models. The script test_polynomial_models.py is designed to evaluate the performance of two pre-trained 5th-degree polynomial regression models. One model is trained on a complete dataset of COVID-19 cases in France (polynomial_model.pkl), while the other is trained exclusively on 2024 data (polynomial_model_2024.pkl). The script

calculates prediction errors, provides a statistical analysis of the data, and visualizes the actual versus predicted values for both datasets.

The script begins by importing necessary libraries, including numpy for numerical operations, pandas for data handling, joblib for loading pre-trained models, and matplotlib for plotting. Paths to the models and datasets are defined to ensure the script can locate the required files. The datasets are loaded, and any missing values are removed using dropna() to ensure clean data for analysis.

For the full dataset (France_total_amount.csv), the index of rows is used as the input feature (representing day indices), while the column total_cases is used as the target variable (representing the total number of cases). Similar preparation is done for the 2024 dataset (case_per_year.csv). Before predictions, the script performs a statistical analysis of the full dataset, calculating the minimum, maximum, and mean number of cases, along with the count of days with zero cases. This step provides insight into the data's characteristics, especially potential issues like zero-case entries.

The models are then used to predict values for their respective datasets. The absolute and relative errors are calculated for both datasets. To address the issue of division by zero during relative error computation, the script uses a conditional calculation (np.where) that avoids dividing by zero by assigning NaN for such cases. For the full dataset, errors are computed twice: once including all data (with potential zero-case issues) and once excluding zero-case entries. Mean absolute and relative errors are calculated for both approaches, with np.nanmean used to handle NaN values gracefully.

The script visualizes the results using matplotlib. It generates separate plots for the full dataset and the 2024 dataset, showing the actual versus predicted values (Figures 4-5). These plots help assess the models' accuracy visually and highlight areas where predictions deviate from actual data.

The script outputs a detailed summary of the errors. For the full dataset, it reports both unfiltered and filtered errors. For the 2024 dataset, it reports the mean absolute and relative errors, which are notably low, indicating good model

performance on this subset. The script provides comprehensive feedback, helping identify potential problems with the data (e.g., zero-case entries) and the models' performance.

The script test_polynomial_models.py is shown in Appendix 3.

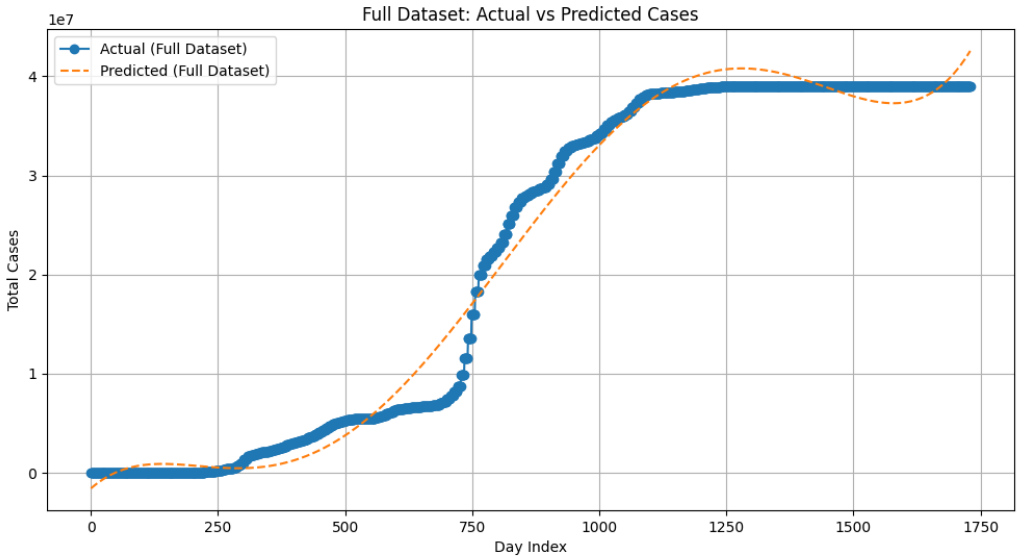

Figure 4 – Visualization of model testing for the whole period

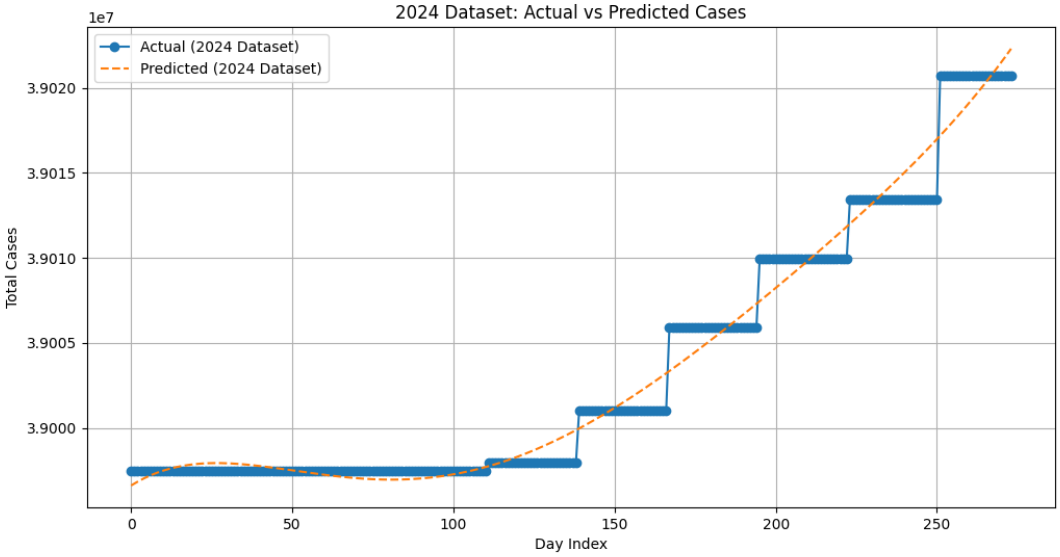

Figure 5 – Visualization of model testing for 2024

3.4 Visualizations of results using the model for the whole period

Conduct a statistical study (to train the model, use data from the first day of the pandemic in the selected country) according to the scheme in Table 1.

To complete this task, I created a script predict_and_visualize_covid.py.

The script predict_and_visualize_covid.py is designed to generate and save visualizations of actual and predicted COVID-19 total cases for specified forecasting periods. Using a pre-trained 5th-degree polynomial regression model, the script predicts total cases over various periods, such as 3, 5, 7, 10, 14, 21, and 30 days. The results are plotted as graphs where actual data transitions smoothly into the predicted data, creating a continuous visualization that illustrates the accuracy and trends of the model.

The script begins by importing necessary libraries, including numpy for numerical operations, pandas for data handling, joblib for loading the trained model, and matplotlib.pyplot for creating and saving visualizations. Paths to the model file, input data, and output directory are defined to ensure the script can locate the necessary resources and save the resulting graphs. The script then checks whether the output directory exists and creates it if necessary to prevent errors during file saving.

The pre-trained polynomial regression model is loaded using joblib, and the historical COVID-19 data is loaded into a pandas DataFrame. After removing any missing values from the data, the script identifies the day indices as the feature (X_all) and the corresponding total cases as the target variable (y_all). It then defines forecasting periods, each characterized by the number of days for which predictions are required. These periods include indices indicating the range of days for prediction.

Table 1 – Scheme of statistical study

	Actual data end date	Dates included in prediction	Calculate the prediction error
Prediction for 3 days	27.09.24	September 28-30	Absolute error: real data as of September 30 minus data as of September 30 produced by the model, modulo $\Delta a \geq \lvert A - a \rvert,$ so $a - \Delta a \leq A \leq a + \Delta a$ or $A = a \pm \Delta a.$ Relative error: $\delta a = \frac{\lvert A - a \rvert}{\lvert a \rvert}$ or $\delta a = \frac{\Delta a}{\lvert a \rvert}.$
Prediction for 5 days	25.09.24	September 26-30	
Prediction for 7 days	23.09.24	September 24-30	
Prediction for 10 days	20.09.24	September 21-30	
Prediction for 14 days	16.09.24	September 17-30	
Prediction for 21 days	09.09.24	September 10-30	
Prediction for 30 days	31.08.24	September 1-30	

For each forecasting period, the script divides the data into actual data up to the start of the prediction period and predicted data for the specified days. Predictions are generated by passing the relevant indices into the loaded model. The actual and predicted data are then combined to create a smooth transition from the historical data to the forecasted values.

The visualization for each period consists of two parts: a blue line representing the actual historical data and a red dashed line representing the predicted data. These two segments are plotted seamlessly, with the actual data transitioning into the

predicted data at the point where forecasting begins. Each graph includes a title indicating the forecast period, labeled axes for day indices and total cases, and a legend distinguishing between actual and predicted data. A grid is added to improve readability.

Finally, the script saves each graph to the specified output directory using filenames that correspond to the forecast period, such as 3_days_prediction.png or 30_days_prediction.png. The saved graphs provide a clear visual representation of the model's performance over different forecasting horizons. The script is fully automated, allowing users to generate predictions and corresponding visualizations effortlessly. Its structure ensures modularity and clarity, making it easily adaptable for other datasets or forecasting scenarios.

The script predict_and_visualize_covid.py is shown in Appendix 4.

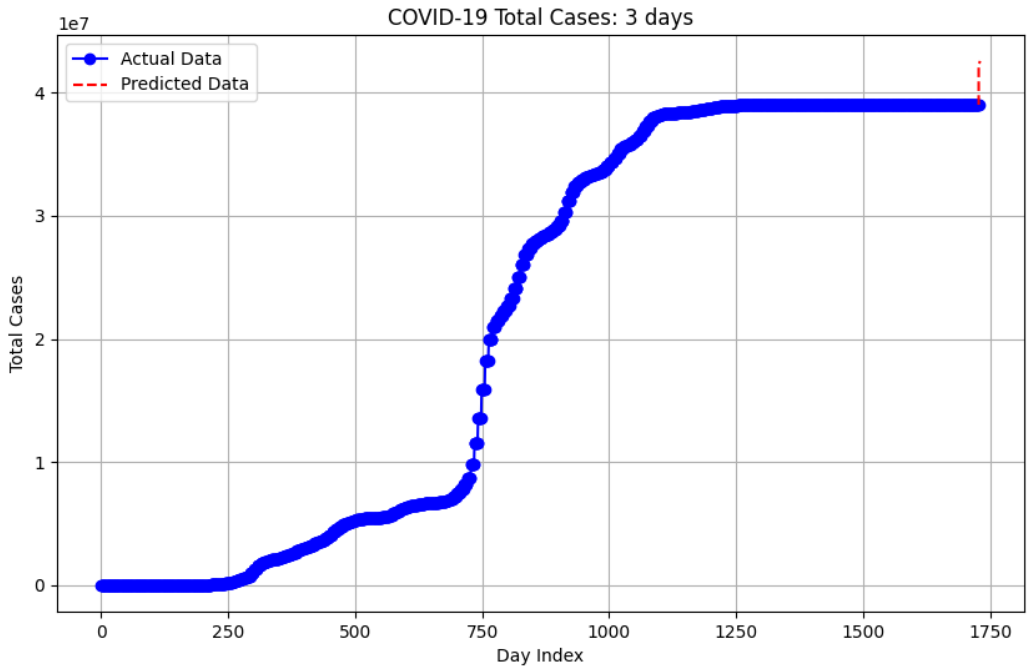

Figure 6 – Actual and predicted data visualization (Whole period, 3 days)

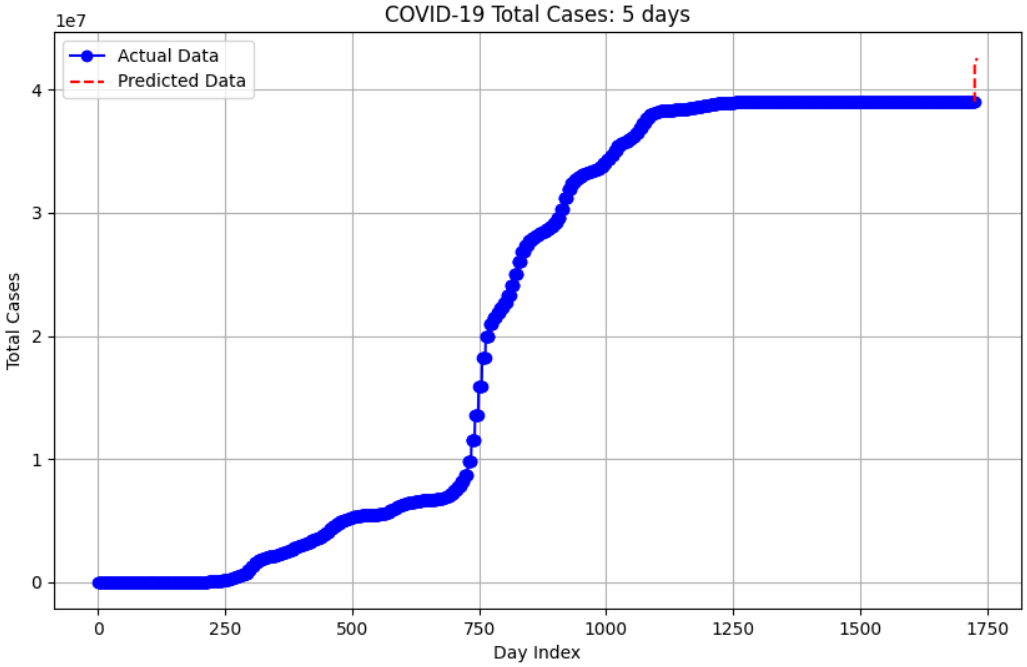

Figure 7 – Actual and predicted data visualization (Whole period, 5 days)

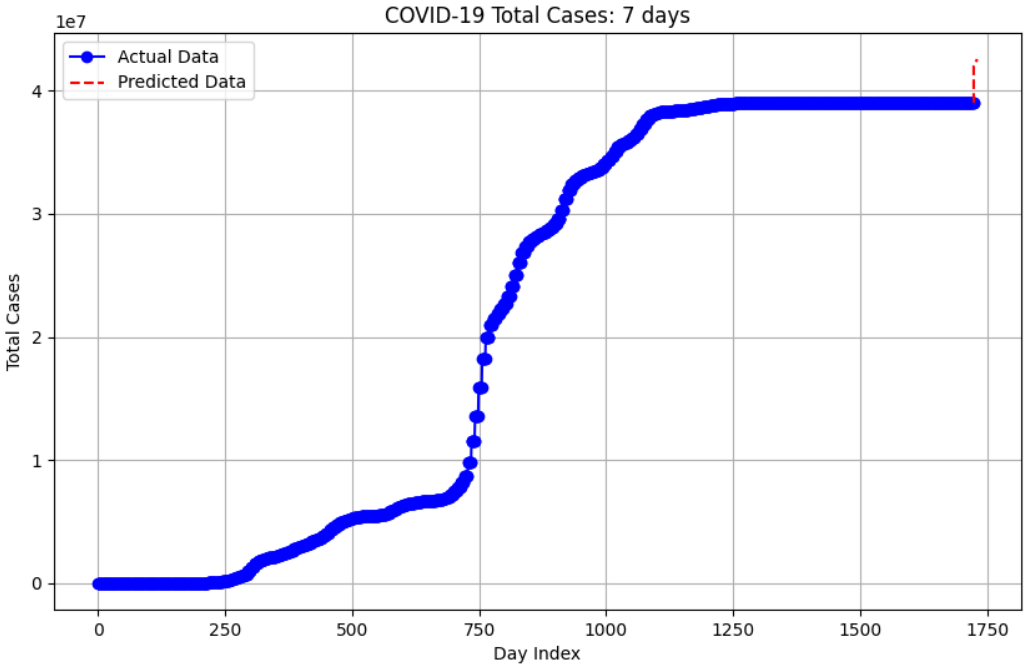

Figure 8 – Actual and predicted data visualization (Whole period, 7 days)

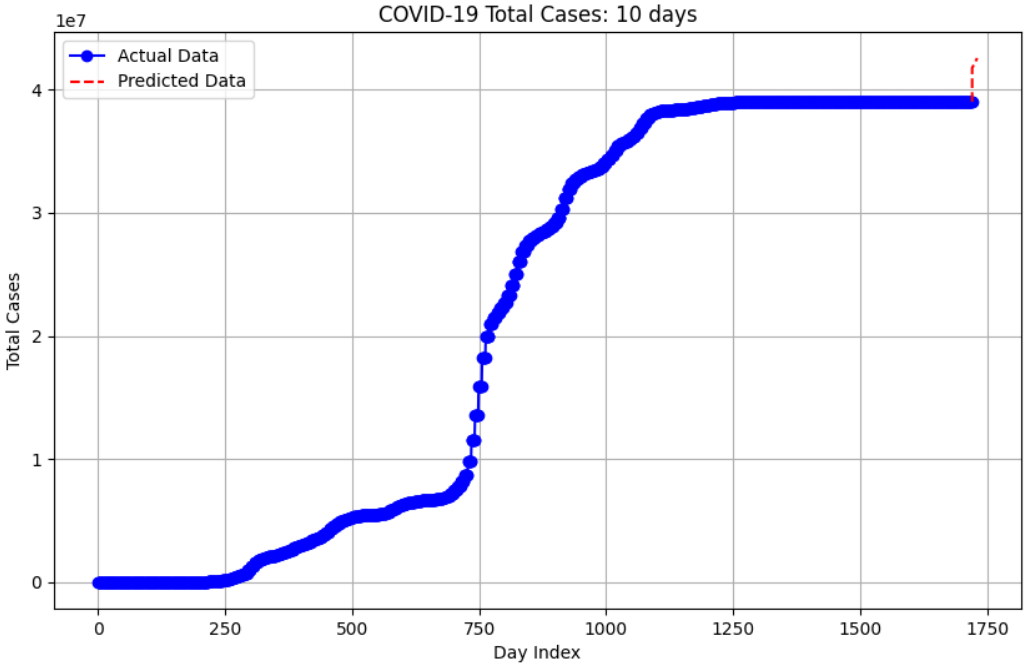

Figure 9 – Actual and predicted data visualization (Whole period, 10 days)

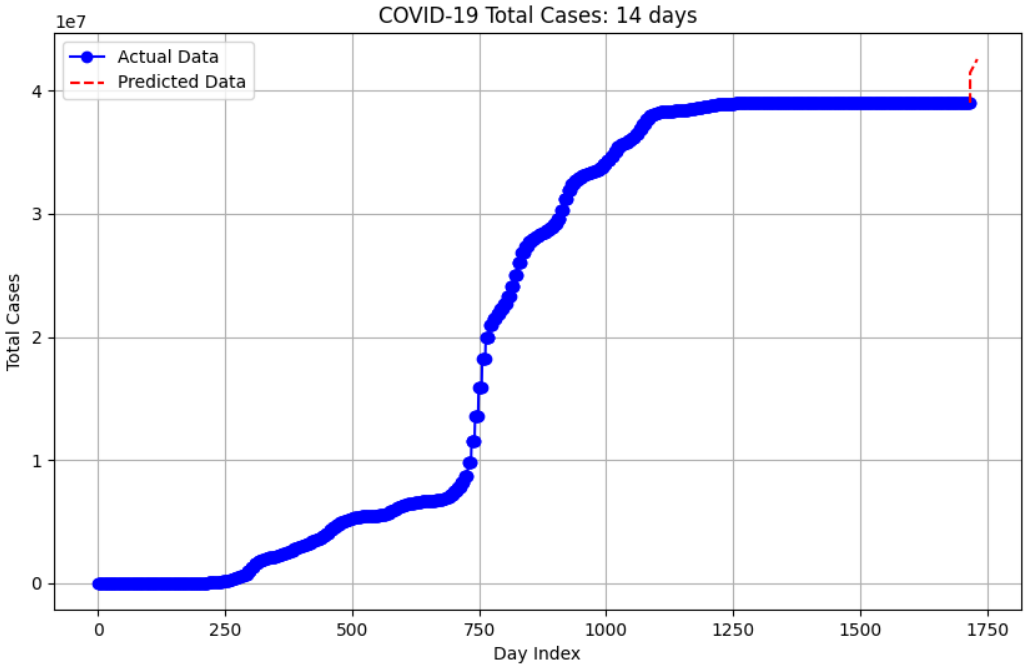

Figure 10 – Actual and predicted data visualization (Whole period, 14 days)

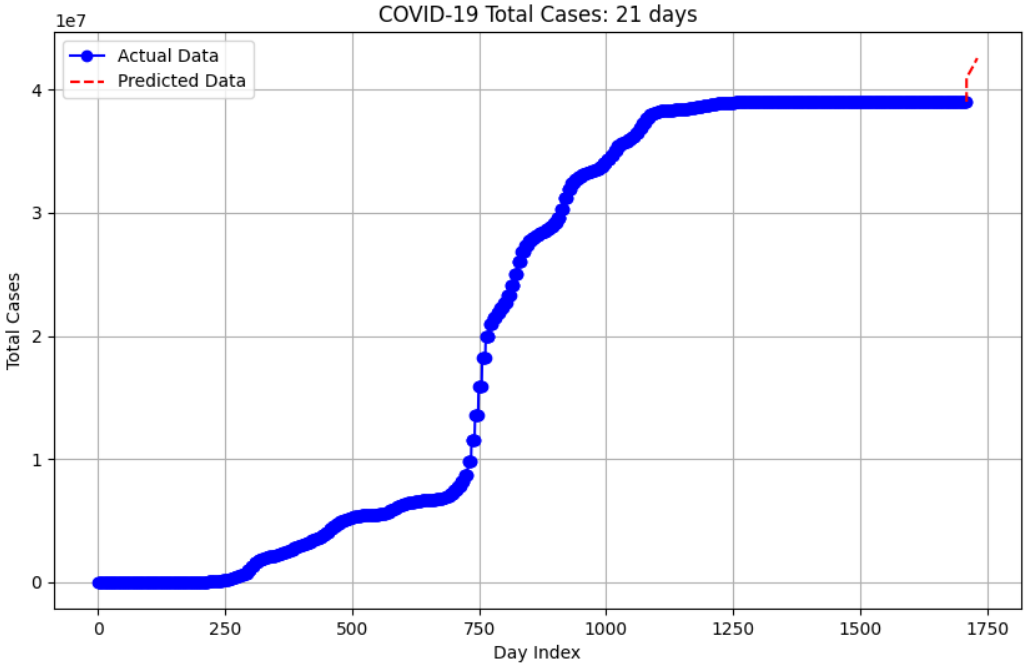

Figure 11 – Actual and predicted data visualization (Whole period, 21 days)

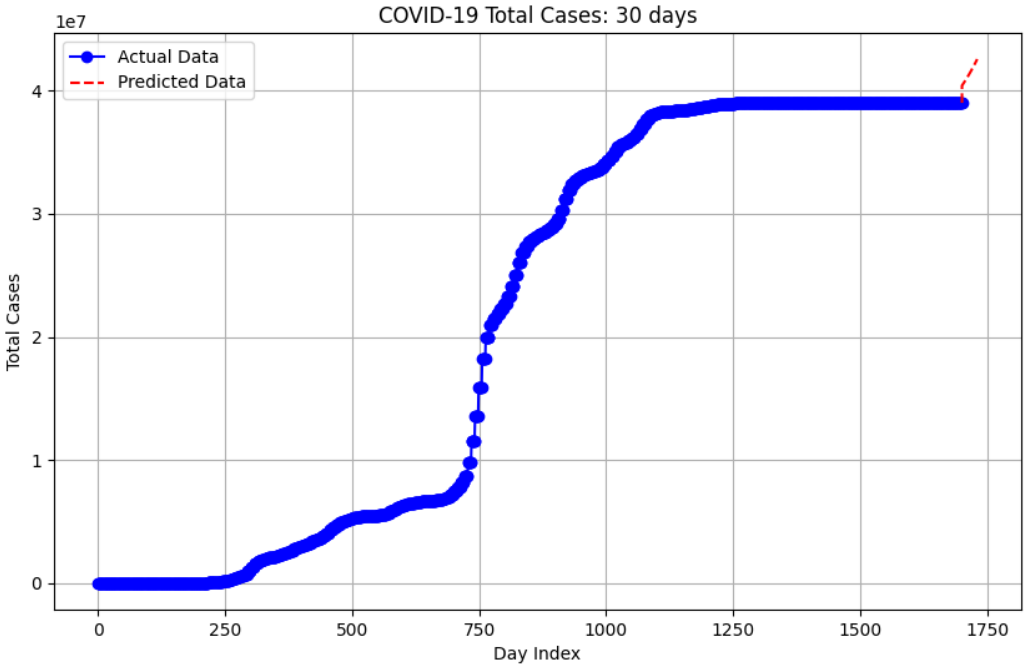

Figure 12 – Actual and predicted data visualization (Whole period, 30 days)

The presented in figures 6 – 12 graphs depict the dynamics of COVID-19 total cases over time, combining actual data with predictions made for future periods. The blue solid lines represent the actual cumulative case counts, while the red dashed lines illustrate the predictions generated by the 5th-degree polynomial regression model. Each graph focuses on a specific forecasting period: 3, 5, 7, 10, 14, 21, and 30 days ahead.

The graphs demonstrate a consistent trend where the actual data plateaus after a significant increase during the earlier phases of the pandemic. The predictive portion, marked by the red dashed line, extends beyond the plateau, providing estimates for the number of cases in the specified forecasting periods. As the prediction horizon increases, the deviation between the predicted trajectory and the actual trend becomes more apparent due to the inherent challenges of forecasting over longer durations.

In these graphs, the transition between the observed data and the forecasted data is seamless, ensuring the prediction aligns with the historical trend. This approach highlights the utility of the 5th-degree polynomial regression model in capturing non-linear patterns in the data and generating forecasts. These visualizations effectively illustrate the model's capacity to predict future values based on historical trends, making them valuable for analyzing and anticipating the progression of COVID-19 cases.

3.5 Visualizations of results using the model for the 2024 period

Conduct a statistical study (for training models using data from the first day of 2024 in the selected country) according to the scheme (Table 2):

Table 2 – Scheme of statistical study

	Actual data end date	Dates included in prediction	Calculate the prediction error
Prediction for 3 days	27.09.24	September 28-30	Absolute error: real data as of September 30 minus data as of September 30 produced by the model, modulo $\Delta a \geq \lvert A - a \rvert$, so $a - \Delta a \leq A \leq a + \Delta a$ or $A = a \pm \Delta a$. Relative error: $\delta a = \frac{\lvert A - a \rvert}{\lvert a \rvert}$ or $\delta a = \frac{\Delta a}{\lvert a \rvert}$.
Prediction for 5 days	25.09.24	September 26-30	
Prediction for 7 days	23.09.24	September 24-30	
Prediction for 10 days	20.09.24	September 21-30	
Prediction for 14 days	16.09.24	September 17-30	
Prediction for 21 days	09.09.24	September 10-30	
Prediction for 30 days	31.08.24	September 1-30	

To complete this task, I created a script predict_and_visualize_covid_2024.py.

The script predict_and_visualize_covid_2024.py is designed to generate and save visualizations of actual and predicted COVID-19 total cases for the year 2024. It utilizes a pre-trained 5th-degree polynomial regression model, polynomial_model_2024.pkl, to forecast the number of cases over various periods: 3, 5, 7, 10, 14, 21, and 30 days. The script generates graphs for each period, combining actual and predicted data into a single seamless visualization and saves these graphs in the specified directory with filenames that include the forecasting period and the year.

The script begins by importing necessary libraries, including numpy for numerical computations, pandas for handling tabular data, joblib for loading the trained model, and matplotlib.pyplot for creating and saving the visualizations. The paths to the model file, input data file (case_per_year.csv), and output directory are defined to ensure accessibility and proper organization of files. Before proceeding, the script checks whether the output directory exists and creates it if necessary, ensuring the graphs can be saved without errors.

The model is loaded using joblib, and the data for 2024 is read into a pandas DataFrame. Any missing values in the data are removed to maintain data integrity. The script prepares the data by extracting the indices of days as features (X_all) and the total cases as the target variable (y_all). Forecasting periods are defined with start and end indices, specifying the range of days to be predicted for each period.

For each forecasting period, the script separates the actual data up to the start of the prediction period and generates predictions for the specified number of days using the trained model. The actual and predicted data are combined to create a continuous dataset, where the actual data transitions smoothly into the predicted data at the point where forecasting begins.

The script generates a graph for each forecasting period. Each graph features a blue line representing the actual data and a red dashed line representing the predicted data. The graphs include a title that specifies the forecasting period and

year, labeled axes for day indices and total cases, and a legend to distinguish between actual and predicted data. A grid is added to enhance readability. The script then saves each graph in the specified directory using filenames that correspond to the forecasting period, such as 3_days_2024_prediction.png or 30_days_2024_prediction.png.

The script provides an automated and efficient way to visualize model predictions alongside historical data for the year 2024. Its structure ensures that the generated graphs are clear and informative, making it easy to assess the model's performance over different forecasting horizons. The naming convention of the output files ensures proper organization and easy identification of the graphs. This script is modular, allowing for straightforward modifications if needed for other datasets or forecasting scenarios.

The code for creating the script is given in the Appendix 5.

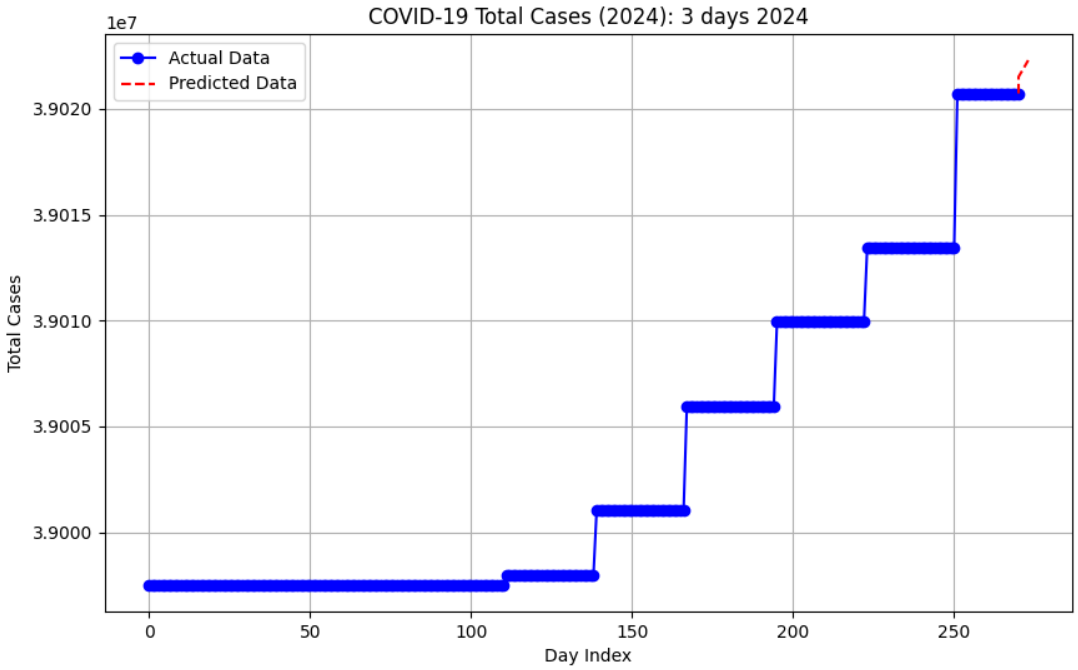

Figure 13 – Actual and predicted data visualization (2024 period, 3 days)

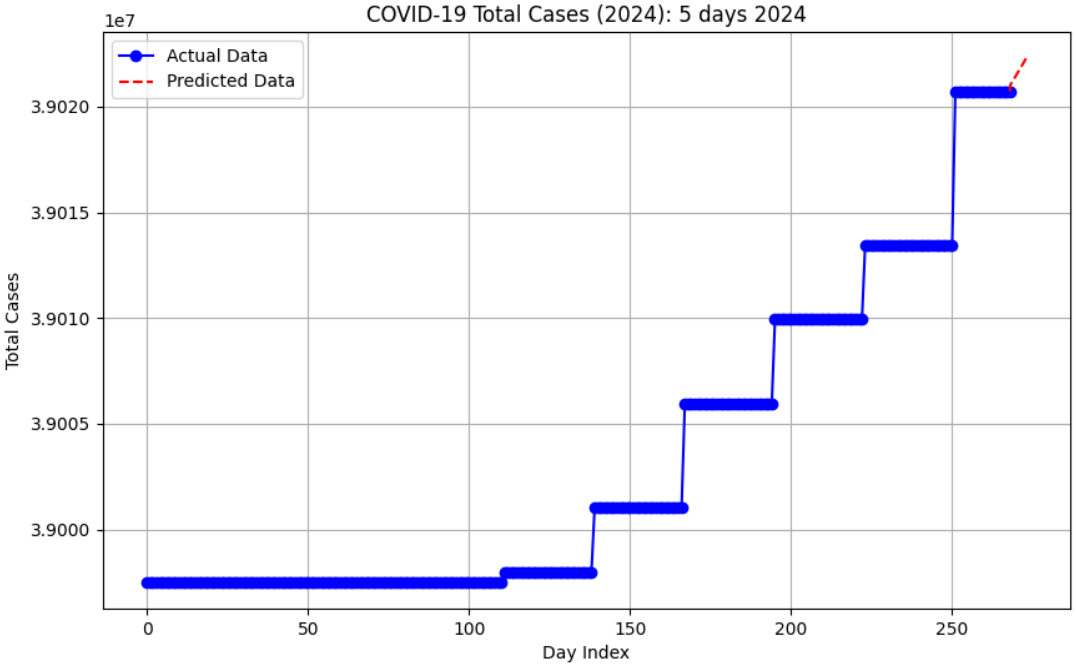

Figure 14 – Actual and predicted data visualization (2024 period, 5 days)

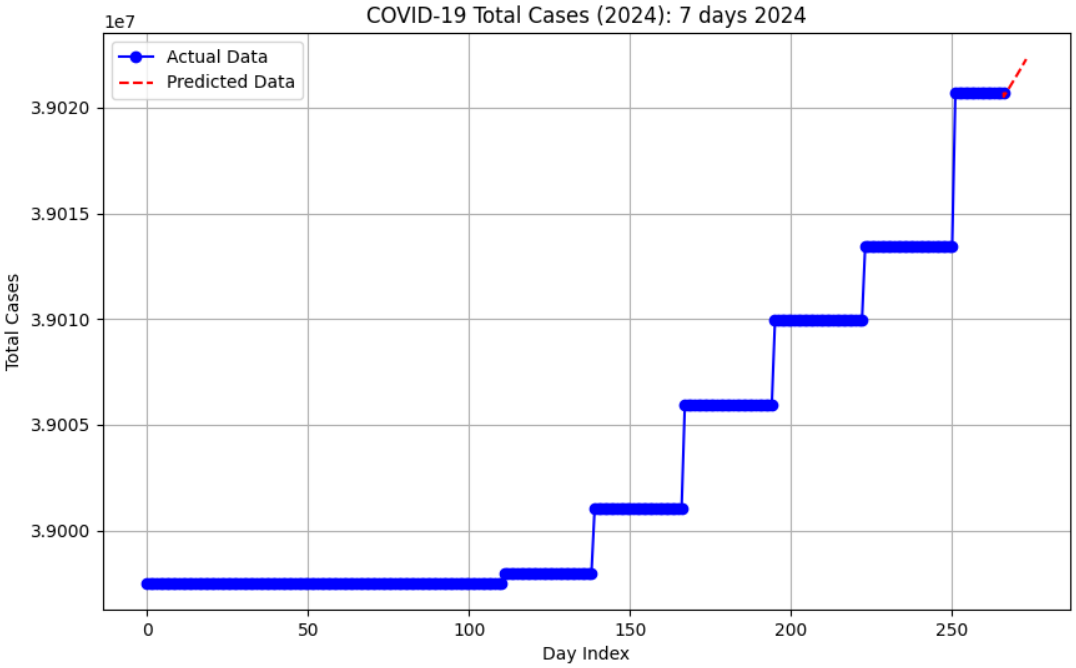

Figure 15 – Actual and predicted data visualization (2024 period, 7 days)

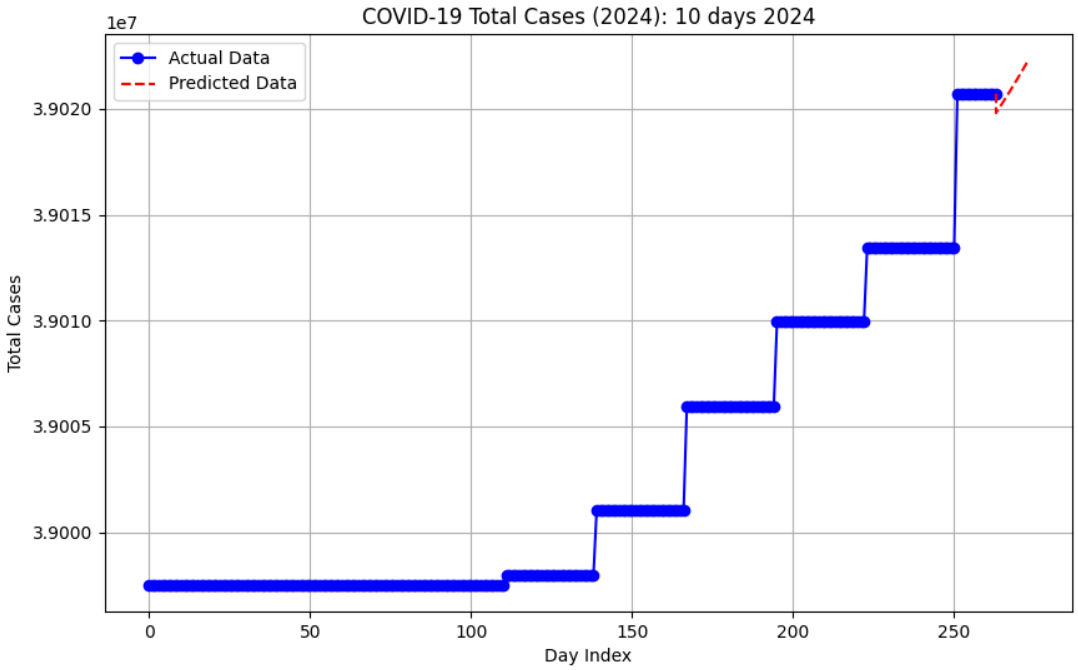

Figure 16 – Actual and predicted data visualization (2024 period, 10 days)

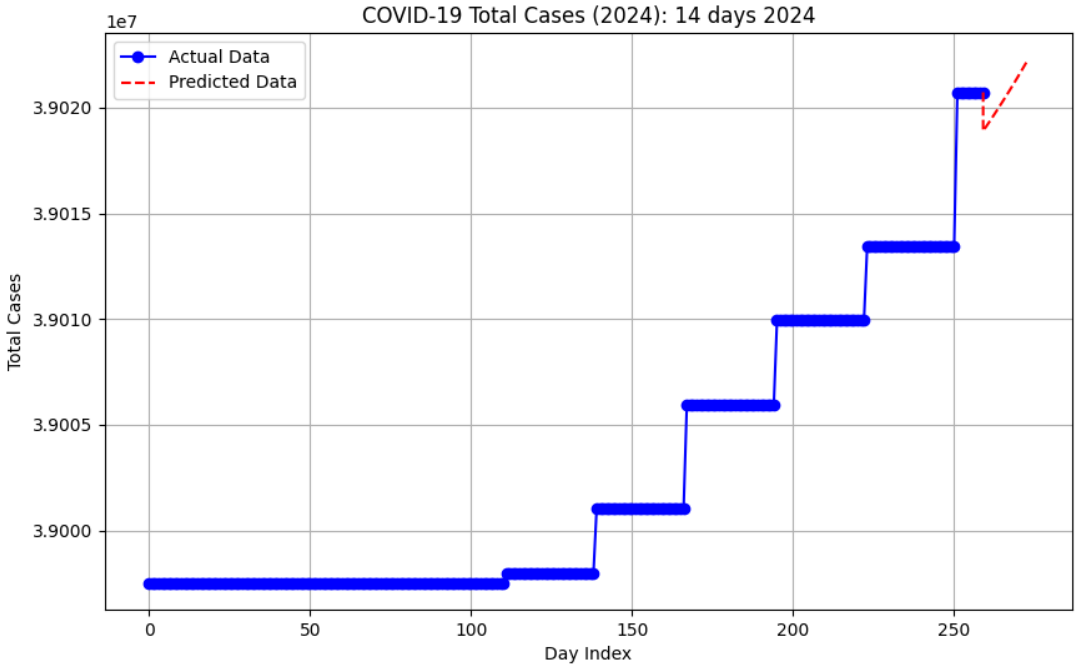

Figure 17 – Actual and predicted data visualization (2024 period, 14 days)

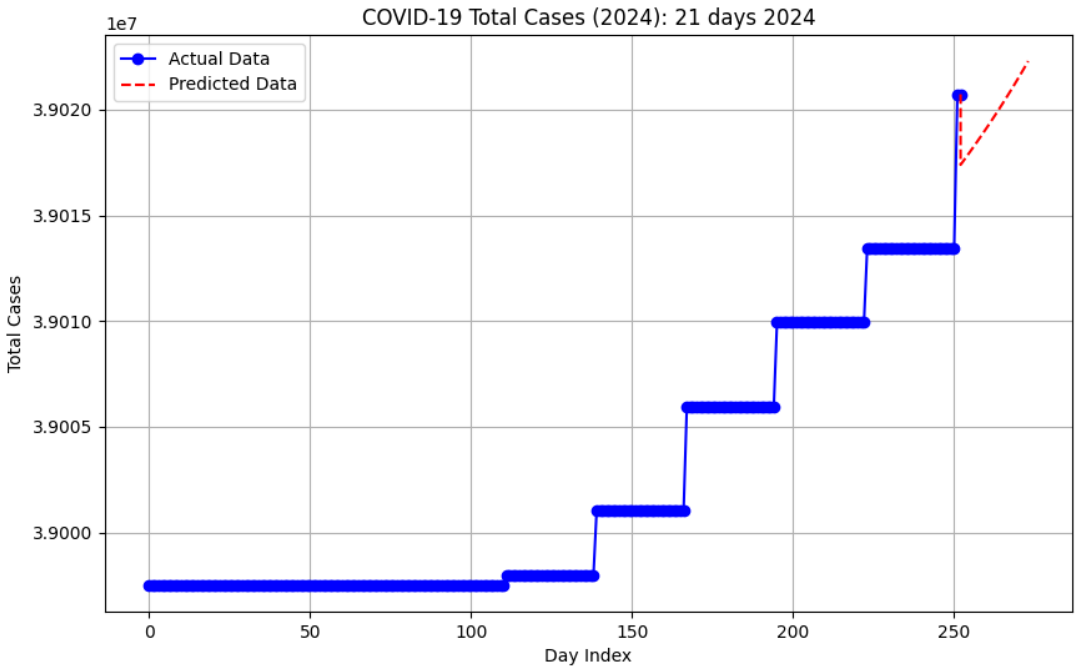

Figure 18 – Actual and predicted data visualization (2024 period, 21 days)

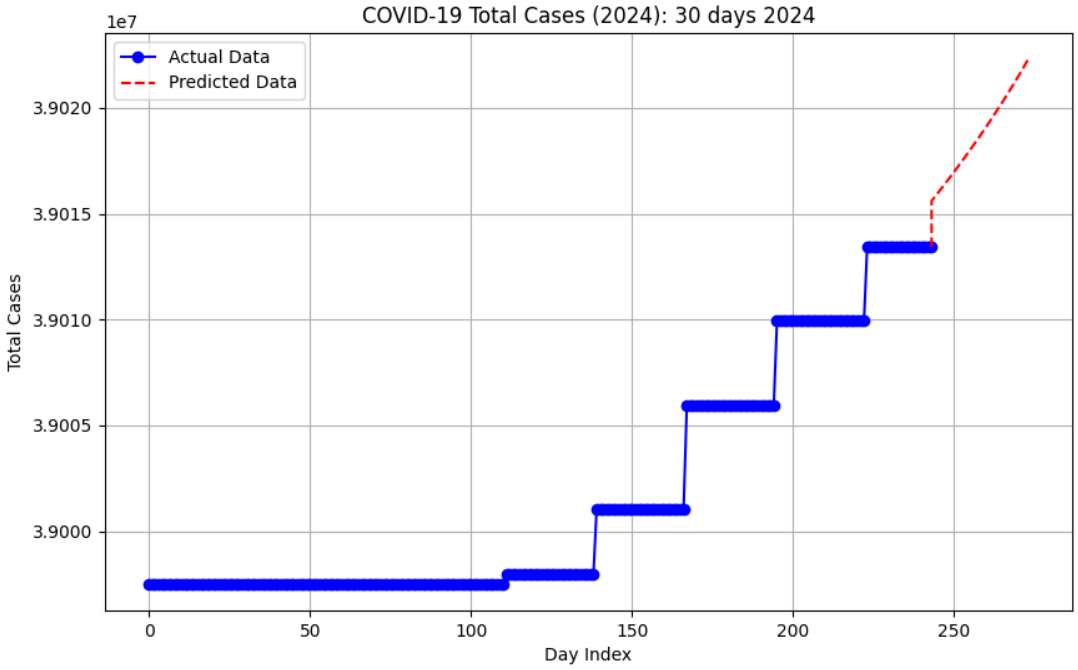

Figure 19 – Actual and predicted data visualization (2024 period, 30 days)

The displayed in figures 13 – 19 graphs illustrate the total cumulative cases of COVID-19 for the year 2024, combining actual observations with predictions for future periods. The blue solid line represents the actual recorded data, while the red dashed line projects the predicted values generated using the 5th-degree polynomial regression model. Each graph corresponds to a specific forecast horizon, ranging from 3 to 30 days ahead.

The graphs highlight the nature of the dataset for 2024, where the actual data displays a step-like progression due to periodic updates in recorded cases. The predicted values seamlessly extend from the actual data, preserving the continuity and reflecting the underlying patterns observed in the historical data. For shorter forecasting periods, such as 3 or 5 days, the predicted trend closely aligns with the recent increments in the actual data. However, as the prediction horizon increases to 10, 14, 21, or 30 days, the forecasts become progressively influenced by the model's assumptions about future trends, leading to more noticeable divergence.

3.6 Analysis and comparison of model accuracy across two data sets

Next, I created a script to compare the models.

The script save_model_comparison.py is designed to evaluate and compare the performance of two pre-trained polynomial regression models for forecasting COVID-19 total cases. The first model is trained on the entire dataset of cases (polynomial_model.pkl), while the second is trained only on data from 2024 (polynomial_model_2024.pkl). The script performs predictions for specific forecast periods, calculates error metrics for both models, and saves the results in a comparison table as an Excel file.

The script begins by importing the necessary libraries, including numpy for numerical computations, pandas for handling data tables, joblib for loading the models, and os for file path management. The paths to the model files, input datasets (France_total_amount.csv for the full dataset and case_per_year.csv for 2024), and

the output file are defined to ensure all resources are properly located and results are saved in the specified directory.

Both models are loaded using joblib, and the datasets are read into pandas DataFrames. The script processes the datasets by removing missing values to maintain data integrity. It then defines prediction periods of 3, 5, 7, 10, 14, 21, and 30 days. For each period, the script calculates the performance of the two models by comparing the predicted values against the actual data.

For the model trained on the full dataset, the script identifies the relevant range of indices for each prediction period and generates forecasts using the model. The same approach is applied to the model trained on 2024 data. The script calculates two error metrics for each model and period: Mean Absolute Error (MAE), which measures the average magnitude of the errors, and Mean Absolute Percentage Error (MAPE), which quantifies the errors as percentages of the actual values. These metrics are used to assess the accuracy and reliability of the models over different forecast periods.

The results of the evaluations are stored in a list of dictionaries, where each dictionary represents the error metrics for a specific period and model. This list is then converted into a pandas DataFrame for easier manipulation and visualization. Before saving the results, the script ensures that the output directory exists, creating it if necessary. The results are saved as an Excel file named model_comparison.xlsx in the specified directory. This file contains columns for the forecast period, MAE and MAPE for the model trained on the full dataset, and MAE and MAPE for the model trained on 2024 data.

The script automates the process of evaluating and comparing the models, providing a clear summary of their performance across multiple forecast periods. It is particularly useful for determining which model performs better under specific conditions.

Table 3 – Comparison of Forecast Errors

Forecast Period, days	Whole Period Absolute Error	Whole Period Relative Error (%)	2024 Period Absolute Error	2024 Period Relative Error (%)
3	3463137	8,87	1305,1660	0,003345
5	3381166	8,66	1046,3540	0,002682
7	3300310	8,45	792,2154	0,002030
10	3181099	8,15	689,7695	0,001768
14	3025976	7,75	855,5283	0,002192
21	2764827	7,08	1400,1280	0,003588
30	2449544	6,27	1898,6210	0,004866

The Excel output (Table 3) ensures that the results are organized and easily accessible for further analysis or reporting. The script is modular and can be adapted for different datasets, models, or error metrics as needed.

The script save_model_comparison.py is shown in Appendix 4.

3.7 List and description of used libraries

In my project, I used a number of libraries for the Python programming language. Below is an introduction to each of the libraries and a summary table of how the libraries are used in the scripts I created for the project.

- pandas
- numpy
- matplotlib.pyplot
- joblib
- os
- scikit-learn: PolynomialFeatures
- scikit-learn: LinearRegression
- scikit-learn: make_pipeline

Pandas is a powerful open-source data analysis and manipulation library in Python. It provides high-performance, easy-to-use data structures and functions for handling structured data, such as tabular or time-series data. The two primary data structures in pandas are the Series, which is a one-dimensional labeled array, and the DataFrame, a two-dimensional labeled data structure similar to a spreadsheet or SQL table. Pandas excels in tasks like cleaning, transforming, and analyzing data efficiently. It supports various file formats, such as CSV, Excel, and JSON, allowing seamless data import and export. The library also integrates well with other Python libraries, such as NumPy, Matplotlib, and Scikit-learn, making it an essential tool in data science workflows. Its intuitive syntax and rich feature set, including filtering, grouping, and reshaping data, make pandas highly popular among data scientists and analysts [36].

NumPy is a fundamental library for numerical computing in Python, providing support for creating and manipulating large, multi-dimensional arrays and matrices. It is designed to perform high-performance mathematical operations, such as element-wise computations, linear algebra, Fourier transforms, and random

number generation, making it an essential tool for scientific and engineering applications. The core feature of NumPy is its ndarray object, which offers a fast and memory-efficient way to store and operate on data. NumPy integrates seamlessly with other Python libraries like Pandas, Matplotlib, and Scikit-learn, forming the backbone of the Python data science ecosystem. Its optimized C-based operations allow it to handle computations much faster than native Python lists. NumPy is widely used in fields such as machine learning, data analysis, and scientific research for its ability to efficiently process large-scale numerical data [38].

Matplotlib's pyplot module is a widely used library for creating static, interactive, and animated visualizations in Python. It provides an interface similar to MATLAB, making it easy for users to generate high-quality plots and charts with minimal code. Using pyplot, users can create a variety of visualizations, including line plots, bar charts, scatter plots, histograms, and more. It allows fine-grained control over visual elements such as titles, labels, axes, and gridlines, enabling customization to suit specific presentation needs. The library integrates well with other scientific libraries like NumPy and Pandas, allowing seamless visualization of data stored in arrays or DataFrames. Additionally, it supports multiple output formats, such as PNG, PDF, and interactive plots in Jupyter Notebooks. Matplotlib's flexibility and extensive functionality make it an indispensable tool for data visualization in data science, engineering, and research [39].

The joblib library in Python is a powerful tool designed for efficient serialization and deserialization of Python objects, particularly in the context of machine learning workflows. It provides robust support for saving and loading complex objects, such as trained models, datasets, and intermediate processing pipelines. Unlike Python's built-in pickle module, joblib is optimized for handling large numerical data arrays efficiently by compressing them into binary formats, which reduces both storage space and input/output time. This makes it particularly useful when working with machine learning models, where the data and model parameters can become large and cumbersome to manage.

In addition to its serialization capabilities, joblib also offers tools for parallel computing, enabling users to execute operations in parallel to speed up processing tasks. This is achieved through its Parallel and delayed functionalities, which simplify parallelization in Python while maintaining compatibility with the existing codebase. The library's API is user-friendly, allowing developers to integrate it seamlessly into their workflows.

Joblib is widely used in the machine learning ecosystem and is often a preferred choice for saving models in libraries such as scikit-learn. It ensures the reproducibility of experiments by allowing models and data transformations to be saved and shared easily. Overall, joblib is a versatile and reliable library that enhances efficiency and scalability in Python applications [37].

The os module in Python provides a way to interact with the operating system and its underlying functionalities. It offers a wide range of tools for performing system-level tasks, such as navigating the file system, managing environment variables, and handling processes. With the os module, users can perform file and directory operations, such as creating, deleting, or renaming files and directories, as well as checking their existence or properties. It also allows access to environment variables, which can be useful for managing configurations in a platform-independent manner. The module supports path manipulation, enabling seamless handling of file paths across different operating systems. Additionally, os provides functions to work with system processes, such as executing shell commands or terminating processes. This module is essential for creating scripts and applications that need to interface with the operating system in a portable and efficient way [40].

Scikit-learn is one of the most widely used libraries in Python for machine learning, offering simple and efficient tools for data analysis and modeling. It is built on top of other powerful libraries such as numpy, scipy, and matplotlib, making it highly compatible with the broader Python ecosystem. Scikit-learn provides a wide range of supervised and unsupervised machine learning algorithms, including regression, classification, clustering, and dimensionality reduction techniques. Its modular design allows seamless integration of preprocessing, model training,

evaluation, and deployment into a single workflow. Scikit-learn is known for its user-friendly interface, extensive documentation, and active community support, which makes it an excellent choice for beginners and professionals alike.

Scikit-learn: PolynomialFeatures is a preprocessing tool from the sklearn.preprocessing module. It transforms input data by generating polynomial and interaction features from the original features, which is crucial for modeling non-linear relationships in regression tasks. The class allows users to specify the degree of the polynomial, include or exclude interaction terms, and decide whether to include a bias (intercept) term in the output. By automating the creation of higher-order terms such as squares, cubes, and interactions, PolynomialFeatures simplifies the process of expanding a dataset to better fit complex models. This transformation is typically used in conjunction with linear regression to enable it to model nonlinear relationships effectively.

Scikit-learn: LinearRegression is a core component of the sklearn.linear_model module and implements the ordinary least squares regression technique. It is used to fit a linear model to the data by finding the optimal coefficients that minimize the residual sum of squares between the observed targets and the predicted values. LinearRegression can handle single or multiple predictors and includes options for regularization when dealing with ill-posed problems. It is often paired with polynomial features or other preprocessing steps to extend its applicability to nonlinear problems. The simplicity and efficiency of the implementation make it a popular choice for regression analysis.

Scikit-learn: make_pipeline is a utility function from the sklearn.pipeline module that simplifies the creation of machine learning pipelines. It allows for the sequential chaining of multiple preprocessing steps and estimators into a single object. By doing so, it automates and streamlines the workflow, ensuring that the data transformation and model fitting processes are performed in a consistent and reproducible manner. The make_pipeline function ensures that all components of the pipeline are executed in the correct order, passing intermediate results automatically from one step to the next. This utility is particularly useful when

combining preprocessing tools like PolynomialFeatures with estimators like LinearRegression, as it keeps the workflow modular and maintainable [35].

Table 4 – Using the libraries

Appendix	1	2	3	4	5	6
Library	create_france_statistics.py	create_polynomial_models.py	test_polynomial_models.py	predict_and_visualize_covid.py	predict_and_visualize_covid_2024.py	save_model_comparison.py
pandas	+	+	+	+	+	+
numpy			+	+	+	+
matplotlib.pyplot			+	+	+	
joblib		+	+	+	+	+
os		+	+	+	+	+
scikit-learn: PolynomialFeatures		+				
scikit-learn: LinearRegression		+				
scikit-learn: make_pipeline		+				

In Table 4, I have summarized the libraries used in the scripts written. Such information allows to show the use of libraries in a visual form.

CONCLUSIONS

The comparison of the models indicates that the model trained on the full dataset (polynomial_model.pkl) has higher errors (MAE and MAPE) compared to the model trained exclusively on the 2024 data (polynomial_model_2024.pkl). This outcome is understandable, as the full dataset model attempts to account for trends over the entire pandemic period, including earlier data that may significantly differ from recent trends. In contrast, the 2024 model more accurately captures the current dynamics because it focuses solely on recent data.

The polynomial_model_2024.pkl demonstrates the smallest MAE and MAPE values for shorter forecast periods (3–10 days). This highlights its high accuracy for short-term forecasts, making it more effective for analyzing contemporary COVID-19 trends in 2024.

Therefore, for short-term predictions and the analysis of current trends, the 2024 model is preferable. For long-term forecasts or studying the overall dynamics of the pandemic, the model trained on the full dataset is more suitable.

REFERENCES

1. World Bank. The Global Economic Outlook During the COVID-19 Pandemic: A Changed World. [Online resource]. Available at: https://www.worldbank.org/en/news/feature/2020/06/08/the-global-economic-outlook-during-the-covid-19-pandemic-a-changed-world

2. World Bank. Chapter 1. The Economic Impacts of the COVID-19 Crisis. [Online resource]. Available at: https://www.worldbank.org/en/publication/wdr2022/brief/chapter-1-introduction-the-economic-impacts-of-the-covid-19-crisis

3. SpringerLink. Research on Vulnerable People and Digital Inclusion: Toward a Consolidated Taxonomical Framework. [Online resource]. Available at: https://link.springer.com/article/10.1007/s10209-022-00867-x

4. Cambridge University Press. The Digital Divide: Amplifying Health Inequalities for People With Severe Mental Illness in the Time of COVID-19. [Online resource]. Available at: https://www.cambridge.org/core/journals/the-british-journal-of-psychiatry/article/digital-divide-amplifying-health-inequalities-for-people-with-severe-mental-illness-in-the-time-of-covid19/BC84F788C557E37B4D18F920EF9373C4

5. World Health Organization. Vaccine Inequity Undermining Global Economic Recovery. [Online resource]. Available at: https://www.who.int/news/item/22-07-2021-vaccine-inequity-undermining-global-economic-recovery

6. SpringerLink. Dynamic Regression Models for Epidemic Forecasting. [Online resource]. Available at: https://link.springer.com/article/10.1007/s43621-024-00213-6

7. ArXiv. Modeling Adaptive Forward-Looking Behavior in Epidemics on Networks. [Online resource]. Available at: https://arxiv.org/abs/2301.04947

8. Karazin University. Mathematical Modeling of Pandemic Dynamics. [Online resource]. Available at: https://periodicals.karazin.ua/mia/article/download/17031/15709

9. Gavi Alliance. How the COVID-19 Pandemic Has Affected Healthcare Around the World. [Online resource]. Available at: https://www.gavi.org/vaccineswork/how-covid-19-pandemic-has-affected-healthcare-around-world

10. SpringerLink. Geographical and Temporal Weighted Regression: Examining Spatial Variability in COVID-19 Mortality. [Online resource]. Available at: https://link.springer.com/article/10.1007/s43762-024-00117-1

11. Nawaz SA, Li J, Bhatti UA, Bazai SU, Zafar A, Bhatti MA, et al. A hybrid approach to forecast the COVID-19 epidemic trend. *PLoS ONE.* 2021;16(10):e0256971. doi:10.1371/journal.pone.0256971.

12. Jewell NP, Lewnard JA, Jewell BL. Predictive Mathematical Models of the COVID-19 Pandemic: Underlying Principles and Value of Projections. *JAMA.* 2020;323(19):1893–1894. doi:10.1001/jama.2020.6585.

13. Xiang Y, Jia Y, Chen L, Guo L, Shu B, Long E. COVID-19 epidemic prediction and the impact of public health interventions: A review of COVID-19 epidemic models. *Infect Dis Model.* 2021;6:324-342. doi:10.1016/j.idm.2021.01.001. PMID: 33437897; PMCID: PMC7790451.

14. Huang J, Zhao Y, Yan W, Lian X, Wang R, Chen B, Chen S. COVID-19 Pandemic Spreading in Europe: Analyzing the Role of Air Temperature and Humidity. *J Thorac Dis.* [Online resource]. Available at: https://jtd.amegroups.org/article/view/76631/html.

15. Gnanvi JE, Salako KV, Kotanmi GB, Glèlè Kakaï R. On the reliability of predictions on Covid-19 dynamics: A systematic and critical review of modelling techniques. *Infect Dis Model.* 2021;6:258-272. doi:10.1016/j.idm.2020.12.008. PMID: 33458453; PMCID: PMC7802527.

16. Nesteruk I. Simulations and Predictions of COVID-19 Pandemic With the Use of SIR Model. *Innov Biosyst Bioeng.* [Online resource]. 2020;4(2):110-121. Available at: https://ibb.kpi.ua/article/view/204274.

17. Complexica. Mathematical Modeling. Available at: https://www.complexica.com/narrow-ai-glossary/mathematical-modeling.

18. Carson ER. The Role of Dynamic Mathematical Models. In: Worden AN, Parke DV, Marks J, editors. *The Future of Predictive Safety Evaluation.* Dordrecht: Springer; 1986. Available at: https://doi.org/10.1007/978-94-009-4139-7_16.

19. Marín-Machuca O, Chacón RD, Alvarez-Lovera N, Pesantes-Grados P, Pérez-Timaná L, Marín-Sánchez O. Mathematical Modeling of COVID-19 Cases and Deaths and the Impact of Vaccinations during Three Years of the Pandemic in Peru. *Vaccines (Basel).* 2023;11(11):1648. doi:10.3390/vaccines11111648. PMID: 38005980; PMCID: PMC10674587.

20. Stochastic Modelling Group. Available at: https://www.maths.lu.se/english/research/research-groups/stochastic-modelling/.

21. Deterministic Models in Mathematical Modeling. Socratica. Available at: https://learn.socratica.com/en/topic/applied-mathematics/mathematical-modeling/deterministic-models.

22. Alooba. Regression Models in Data Science. Available at: https://www.alooba.com/skills/concepts/data-science/regression-models/.

23. Julius AI. Regression Analysis in Statistics. Available at: https://julius.ai/articles/regression-analysis-in-statistics.

24. Bookdown. Overview of Regression Models in Public Health. Available at: https://www.bookdown.org/rwnahhas/RMPH/overview-why.html.

25. Forecasting and Regression Models. *Forecasting: Principles and Practice.* Available at: https://otexts.com/fpp3/forecasting-regression.html.

26. GeeksforGeeks. Python Implementation of Polynomial Regression. Available at: https://www.geeksforgeeks.org/python-implementation-of-polynomial-regression/.

27. ResearchGate. An Automated Height Transformation Using Precise Geoid Models. Available at: https://www.researchgate.net/publication/228574971_An_automated_height_transformation_using_precise_geoid_models.

28. ArcGIS. COVID-19 Dashboard. Available at: https://gisanddata.maps.arcgis.com/apps/dashboards/bda7594740fd40299423467b48e9ecf6.

29.World Health Organization. Public Health Emergency Dashboard. Available at: https://extranet.who.int/publicemergency/.

30. Our World in Data. COVID-19 Data. Available at: https://github.com/owid/covid-19-data/blob/c6b482425695ed67d3fff85ce614fc4189cf2c17/public/data/owid-covid-data.csv.

31. World Health Organization. COVID-19 Cases Dashboard. Available at: https://data.who.int/dashboards/covid19/cases?n=c.

32. Plackett RL. The Discovery of the Method of Least Squares. Available at: https://hedibert.org/wp-content/uploads/2016/08/plackett1972-thediscoveryofthemethodofleastsquares.pdf.

33. Statistics by Jim. Mean Squared Error (MSE). Available at: https://statisticsbyjim.com/regression/mean-squared-error-mse/.

34. Scribbr. Coefficient of Determination Explained. Available at: https://www.scribbr.com/statistics/coefficient-of-determination/.

35. Scikit-learn. Machine Learning in Python. Available at: https://scikit-learn.org/.

36. Pandas. Python Data Analysis Library. Available at: https://pandas.pydata.org/.

37. Joblib: running Python functions as pipeline jobs. Available at: https://joblib.readthedocs.io/en/latest/

38. NumPy. The Fundamental Package for Scientific Computing in Python. Available at: https://numpy.org/.

39. Matplotlib. Visualization with Python. Available at: https://matplotlib.org/.

40. GeeksforGeeks. OS Module in Python with Examples. Available at: https://www.geeksforgeeks.org/os-module-python-examples.

Appendix 1

create_france_statistics.py

```
import pandas as pd
# Path to the original data file
input_file_path = r'C:\Python\Prediction_COVID19_Project_1.04\owid-covid-data.csv'

# Path to save the resulting file
output_file_path = r'C:\Python\Prediction_COVID19_Project_1.04\France_total_statistics.csv'

# Load the dataset
data = pd.read_csv(input_file_path)

# Filter for France
france_data = data[data['location'] == 'France']

# Select relevant columns and sort by date
france_statistics = france_data[['date', 'total_cases']].copy()
france_statistics['date'] = pd.to_datetime(france_statistics['date'])
france_statistics = france_statistics.sort_values(by='date')

# Fill missing values in total_cases with 0
france_statistics['total_cases'] = france_statistics['total_cases'].fillna(0)

# Save to CSV
france_statistics.to_csv(output_file_path, index=False)
print(f"File saved successfully at: {output_file_path}")
```

Appendix 2

create_polynomial_models.py

```
# Import necessary libraries
import pandas as pd
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import make_pipeline
import joblib
import os

# Define file paths
data_dir = r"C:\Python\Prediction_COVID19_Project_1.05"  # Update the path if necessary
france_data_path = os.path.join(data_dir, "France_total_amount.csv")
case_data_path = os.path.join(data_dir, "case_per_year.csv")

# Define output paths for the models
output_model_all_path = os.path.join(data_dir, "polynomial_model.pkl")
output_model_2024_path = os.path.join(data_dir, "polynomial_model_2024.pkl")

# Load datasets
# Load full dataset
france_data = pd.read_csv(france_data_path)

# Load 2024 dataset
case_data = pd.read_csv(case_data_path)

# Data preprocessing for full dataset
france_data.dropna(inplace=True)  # Ensure no missing values
```

```
X_all = france_data.index.values.reshape(-1, 1)  # Using index as a feature
y_all = france_data['total_cases'].values  # Target: total cases

# Data preprocessing for 2024 dataset
case_data.dropna(inplace=True)  # Ensure no missing values
X_2024 = case_data.index.values.reshape(-1, 1)  # Using index as a feature
y_2024 = case_data['total_cases'].values  # Target: total cases

# Build and train the model for the full dataset
polynomial_model_all = make_pipeline(PolynomialFeatures(5),
LinearRegression())
polynomial_model_all.fit(X_all, y_all)

# Save the full dataset model
joblib.dump(polynomial_model_all, output_model_all_path)
print(f"Full dataset model saved to: {output_model_all_path}")

# Build and train the model for 2024 dataset
polynomial_model_2024 = make_pipeline(PolynomialFeatures(5),
LinearRegression())
polynomial_model_2024.fit(X_2024, y_2024)

# Save the 2024 dataset model
joblib.dump(polynomial_model_2024, output_model_2024_path)
print(f"2024 dataset model saved to: {output_model_2024_path}")
```

Appendix 3

test_polynomial_models.py

```
# Import necessary libraries
import numpy as np
import pandas as pd
import joblib
import matplotlib.pyplot as plt

# Define paths to models and data
data_dir = r"C:\Python\Prediction_COVID19_Project_1.05"  # Update the path if necessary
model_all_path = r"polynomial_model.pkl"
model_2024_path = r"polynomial_model_2024.pkl"
france_data_path = r"France_total_amount.csv"
case_data_path = r"case_per_year.csv"

# Load the models
model_all = joblib.load(model_all_path)
model_2024 = joblib.load(model_2024_path)

# Load the datasets
france_data = pd.read_csv(france_data_path)
case_data = pd.read_csv(case_data_path)

# Prepare data for predictions
# For full dataset
france_data.dropna(inplace=True)  # Ensure no missing values
X_all = france_data.index.values.reshape(-1, 1)  # Feature: day indices
y_all = france_data['total_cases'].values  # Target: total cases
```

```
# For 2024 dataset
case_data.dropna(inplace=True)  # Ensure no missing values
X_2024 = case_data.index.values.reshape(-1, 1)  # Feature: day indices
y_2024 = case_data['total_cases'].values  # Target: total cases

# Analyze the full dataset
print("Full Dataset Analysis:")
print(f"Min cases: {y_all.min()}, Max cases: {y_all.max()}, Mean cases:
{y_all.mean()}")
print(f"Number of zero cases: {np.sum(y_all == 0)}")

# Predict using both models
y_all_pred = model_all.predict(X_all)
y_2024_pred = model_2024.predict(X_2024)

# Calculate errors
# Full dataset (with zero cases)
absolute_error_all = np.abs(y_all - y_all_pred)
relative_error_all = np.where(y_all != 0, (absolute_error_all / y_all) * 100, np.nan)
# Avoid division by zero
mean_relative_error_all = np.nanmean(relative_error_all)  # Ignore NaN values for
the mean

# Full dataset (filtered for non-zero cases)
non_zero_indices_all = y_all > 0
y_all_filtered = y_all[non_zero_indices_all]
y_all_pred_filtered = y_all_pred[non_zero_indices_all]

absolute_error_all_filtered = np.abs(y_all_filtered - y_all_pred_filtered)
```

```
relative_error_all_filtered = (absolute_error_all_filtered / y_all_filtered) * 100
mean_relative_error_all_filtered = relative_error_all_filtered.mean()

# 2024 dataset
absolute_error_2024 = np.abs(y_2024 - y_2024_pred)
relative_error_2024 = np.where(y_2024 != 0, (absolute_error_2024 / y_2024) *
100, np.nan)  # Avoid division by zero
mean_relative_error_2024 = np.nanmean(relative_error_2024)  # Ignore NaN
values for the mean

# Plot the results for the full dataset
plt.figure(figsize=(12, 6))
plt.plot(france_data.index, y_all, label="Actual (Full Dataset)", marker="o")
plt.plot(france_data.index, y_all_pred, label="Predicted (Full Dataset)",
linestyle="--")
plt.title("Full Dataset: Actual vs Predicted Cases")
plt.xlabel("Day Index")
plt.ylabel("Total Cases")
plt.legend()
plt.grid(True)
plt.show()

# Plot the results for the 2024 dataset
plt.figure(figsize=(12, 6))
plt.plot(case_data.index, y_2024, label="Actual (2024 Dataset)", marker="o")
plt.plot(case_data.index, y_2024_pred, label="Predicted (2024 Dataset)",
linestyle="--")
plt.title("2024 Dataset: Actual vs Predicted Cases")
plt.xlabel("Day Index")
plt.ylabel("Total Cases")
```

```
plt.legend()
plt.grid(True)
plt.show()

# Print summary of errors
print("Full Dataset (with zero cases):")
print(f"Mean Absolute Error: {absolute_error_all.mean():.2f}")
print(f"Mean Relative Error: {mean_relative_error_all:.2f}%")

print("\nFull Dataset (filtered non-zero cases):")
print(f"Mean Absolute Error (filtered): {absolute_error_all_filtered.mean():.2f}")
print(f"Mean Relative Error (filtered): {mean_relative_error_all_filtered:.2f}%")

print("\n2024 Dataset:")
print(f"Mean Absolute Error: {absolute_error_2024.mean():.2f}")
print(f"Mean Relative Error: {mean_relative_error_2024:.2f}%")
```

Appendix 4

predict_and_visualize_covid.py

```
import numpy as np
import pandas as pd
import joblib
import matplotlib.pyplot as plt
import os

# Define paths
data_dir = r"C:\Python\Prediction_COVID19_Project_1.05"
model_path = os.path.join(data_dir, "polynomial_model.pkl")  # Path to the model
france_data_path = os.path.join(data_dir, "France_total_amount.csv")  # Path to
the data
output_dir = data_dir  # Directory for saving graphs

# Ensure output directory exists
if not os.path.exists(output_dir):
    os.makedirs(output_dir)

# Load the model and data
model = joblib.load(model_path)
france_data = pd.read_csv(france_data_path)

# Prepare the dataset
france_data.dropna(inplace=True)
X_all = france_data.index.values.reshape(-1, 1)  # Feature: day indices
y_all = france_data['total_cases'].values  # Target: total cases

# Define prediction periods and their respective indices
```

```
prediction_periods = {
    "3_days": {"start": len(france_data) - 4, "end": len(france_data) - 1},
    "5_days": {"start": len(france_data) - 6, "end": len(france_data) - 1},
    "7_days": {"start": len(france_data) - 8, "end": len(france_data) - 1},
    "10_days": {"start": len(france_data) - 11, "end": len(france_data) - 1},
    "14_days": {"start": len(france_data) - 15, "end": len(france_data) - 1},
    "21_days": {"start": len(france_data) - 22, "end": len(france_data) - 1},
    "30_days": {"start": len(france_data) - 31, "end": len(france_data) - 1},
}

# Create predictions and save graphs for each period
for period, indices in prediction_periods.items():
    start_idx, end_idx = indices["start"], indices["end"]
    X_pred = np.arange(start_idx, end_idx + 1).reshape(-1, 1)
    y_actual = y_all[:start_idx + 1]  # Include only actual data up to the start of prediction
    y_pred = model.predict(X_pred)

    # Combine actual and predicted data for plotting
    X_combined = np.arange(start_idx + 1).tolist() + X_pred.flatten().tolist()
    y_combined = y_actual.tolist() + y_pred.tolist()

    # Plot the results
    plt.figure(figsize=(10, 6))
    plt.plot(X_combined[:len(y_actual)], y_actual, label="Actual Data", color="blue", marker="o")
    plt.plot(X_combined[len(y_actual) - 1:], y_combined[len(y_actual) - 1:], label="Predicted Data", color="red", linestyle="--")
    plt.title(f"COVID-19 Total Cases: {period.replace('_', ' ')}")
    plt.xlabel("Day Index")
```

```
    plt.ylabel("Total Cases")
    plt.legend()
    plt.grid(True)

    # Save the graph
    graph_path = os.path.join(output_dir, f"{period}_prediction.png")
    plt.savefig(graph_path)
    plt.close()

print("Graphs for all prediction periods have been saved successfully!")
```

Appendix 5

predict_and_visualize_covid_2024.py

```
import numpy as np
import pandas as pd
import joblib
import matplotlib.pyplot as plt
import os

# Define paths
data_dir = r"C:\Python\Prediction_COVID19_Project_1.05"
model_path = os.path.join(data_dir, "polynomial_model_2024.pkl")  # Path to the model for 2024
case_data_path = os.path.join(data_dir, "case_per_year.csv")  # Path to the 2024 data
output_dir = data_dir  # Directory for saving graphs

# Ensure output directory exists
if not os.path.exists(output_dir):
    os.makedirs(output_dir)

# Load the model and data
model = joblib.load(model_path)
case_data = pd.read_csv(case_data_path)

# Prepare the dataset
case_data.dropna(inplace=True)
X_all = case_data.index.values.reshape(-1, 1)  # Feature: day indices
y_all = case_data['total_cases'].values  # Target: total cases
```

```
# Define prediction periods and their respective indices
prediction_periods = {
    "3_days_2024": {"start": len(case_data) - 4, "end": len(case_data) - 1},
    "5_days_2024": {"start": len(case_data) - 6, "end": len(case_data) - 1},
    "7_days_2024": {"start": len(case_data) - 8, "end": len(case_data) - 1},
    "10_days_2024": {"start": len(case_data) - 11, "end": len(case_data) - 1},
    "14_days_2024": {"start": len(case_data) - 15, "end": len(case_data) - 1},
    "21_days_2024": {"start": len(case_data) - 22, "end": len(case_data) - 1},
    "30_days_2024": {"start": len(case_data) - 31, "end": len(case_data) - 1},
}

# Create predictions and save graphs for each period
for period, indices in prediction_periods.items():
    start_idx, end_idx = indices["start"], indices["end"]
    X_pred = np.arange(start_idx, end_idx + 1).reshape(-1, 1)
    y_actual = y_all[:start_idx + 1]  # Include only actual data up to the start of prediction
    y_pred = model.predict(X_pred)

    # Combine actual and predicted data for plotting
    X_combined = np.arange(start_idx + 1).tolist() + X_pred.flatten().tolist()
    y_combined = y_actual.tolist() + y_pred.tolist()

    # Plot the results
    plt.figure(figsize=(10, 6))
    plt.plot(X_combined[:len(y_actual)], y_actual, label="Actual Data", color="blue", marker="o")
    plt.plot(X_combined[len(y_actual) - 1:], y_combined[len(y_actual) - 1:], label="Predicted Data", color="red", linestyle="--")
    plt.title(f"COVID-19 Total Cases (2024): {period.replace('_', ' ')}")
```

```
    plt.xlabel("Day Index")
    plt.ylabel("Total Cases")
    plt.legend()
    plt.grid(True)

    # Save the graph
    graph_path = os.path.join(output_dir, f"{period}_prediction.png")
    plt.savefig(graph_path)
    plt.close()

print("Graphs for all prediction periods (2024) have been saved successfully!")
```

Appendix 6

save_model_comparison.py

```
import numpy as np
import pandas as pd
import os
import joblib

# Define paths
data_dir = r"C:\Python\Prediction_COVID19_Project_1.05"
model_2024_path = os.path.join(data_dir, "polynomial_model_2024.pkl")  # Model for 2024
model_all_path = os.path.join(data_dir, "polynomial_model.pkl")  # Model for full dataset
data_all_path = os.path.join(data_dir, "France_total_amount.csv")  # Full dataset
data_2024_path = os.path.join(data_dir, "case_per_year.csv")  # 2024 dataset
output_file = os.path.join(data_dir, "model_comparison.xlsx")  # Output file

# Load models
model_all = joblib.load(model_all_path)
model_2024 = joblib.load(model_2024_path)

# Load datasets
data_all = pd.read_csv(data_all_path).dropna()
data_2024 = pd.read_csv(data_2024_path).dropna()

# Define prediction periods
prediction_periods = [3, 5, 7, 10, 14, 21, 30]

# Initialize results storage
```

```
results = []

# Evaluate models for each period
for period in prediction_periods:
    # For full dataset model
    start_idx_all = len(data_all) - period
    X_all_pred = np.arange(start_idx_all, len(data_all)).reshape(-1, 1)
    y_all_actual = data_all['total_cases'].values[start_idx_all:]
    y_all_pred = model_all.predict(X_all_pred)

    mae_all = np.mean(np.abs(y_all_actual - y_all_pred))  # Mean Absolute Error
    mape_all = np.mean(np.abs((y_all_actual - y_all_pred) / y_all_actual)) * 100  # Mean Absolute Percentage Error

    # For 2024 dataset model
    start_idx_2024 = len(data_2024) - period
    X_2024_pred = np.arange(start_idx_2024, len(data_2024)).reshape(-1, 1)
    y_2024_actual = data_2024['total_cases'].values[start_idx_2024:]
    y_2024_pred = model_2024.predict(X_2024_pred)

    mae_2024 = np.mean(np.abs(y_2024_actual - y_2024_pred))  # Mean Absolute Error
    mape_2024 = np.mean(np.abs((y_2024_actual - y_2024_pred) / y_2024_actual)) * 100  # Mean Absolute Percentage Error

    # Store results
    results.append({
        "Period": period,
        "MAE_All": mae_all,
        "MAPE_All (%)": mape_all,
```

```
        "MAE_2024": mae_2024,
        "MAPE_2024 (%)": mape_2024
    })

# Convert results to DataFrame
results_df = pd.DataFrame(results)

# Ensure the output directory exists
if not os.path.exists(data_dir):
    os.makedirs(data_dir)

# Save the results to an Excel file
results_df.to_excel(output_file, index=False)

print(f"Comparison table has been saved to: {output_file}")
```