NATIONAL TECHNICAL UNIVERSITY OF UKRAINE
"IGOR SIKORSKY KYIV POLYTECHNIC INSTITUTE"

Educational and Research Institute for Applied System Analysis

Department of Artificial Intelligence

The defense allowed:

Acting head of the department

_____ Iryna DZHYGYREY

«___»_____20__ p.

Diploma work

for a bachelor's degree
in Educational and Professional Program
«Systems and Methods of Artificial Intelligence»
specialty 122 «Computer sciences»
on the topic: « Time series analysis and forecasting of demographics of developed and developing countries»

Completed:
IV year student, group KI-01
Maznichenko Lev Vladyslavovych _____

Supervisor:
Ph. D, Associate professor
Huskova V.H. _____

Adviser for the economic section:
Ph. D, Professor
Shevchuk O.A.

Adviser for the control of norms:
the first category specialist of the Department of AI,
Kravets P.V. _____

Reviewer:
Doctor of Technical Sciences, Professor
Bidiuk P.I. _____

Hereby I certify that this diploma work does not contain any borrowed material from the works of other authors without appropriate references.
Student _____

Kyiv – 2024 year

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут прикладного системного аналізу

Кафедра штучного інтелекту

До захисту допущено:

В. о. завідувачки кафедри

_____ Ірина ДЖИГИРЕЙ

«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системи і методи штучного інтелекту»

спеціальності 122 «Комп'ютерні науки»

на тему: « Прогнозування демографічних показників розвинених та розвиваючихся країн з використанням часових рядів»

Виконав (-ла):
студент (-ка) IV курсу, групи КІ-01
Мазніченко Лев Владиславович _____

Керівник:
Кандидат тех. наук, доцент
Гуськова В.Г. _____

Консультант з економічного розділу:
доктор економ. наук, професор
Шевчук Олена Анатоліївна _____

Консультант з нормоконтролю:
фахівець першої категорії кафедри ШІ,
Кравець П.В. _____

Рецензент:
доктор техн. наук, професор
Бідюк П.І. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2024 року

National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

Educational and Research Institute for Applied System Analysis

Department of Artificial Intelligence

Level of higher education – first (bachelor)

Specialty (program subject area) – 122 «Computer sciences»

Educational and Professional Program –
«Systems and Methods of Artificial Intelligence»

APPROVED
Acting head of the department
_____ Iryna DZHYGYREY
31st January 2024

ASSIGNMENT

for the student's diploma work

Maznichenko Lev Vladyslavovych

1. Topic of the project « **Time series analysis and forecasting of demographics of developed and developing countries**», work's supervisor Huskova. V. H., Ph.D., Associate professor approved by the University Order of «___»_____ 2024 #_____

2. Deadline for submission of the project: 10th June 2024.

3. Initial data for the project: Demographics of developed and developing countries

4. Content of the text part 1. Analysis of the subject area; 2. Mathematical and theoretical foundations of work; 3. Development of a forecasting model; 4. Functional and cost analysis of the software product.

5. List of the graphic material: presentation, resulting plots.

6. Advisers for sections of the work

Section	Surname, initials and position of the adviser	Signature, date	
		task issued	task accepted
Economic	Shevchuk Olena, Assosiate Professor		

7. Date of the task issue: 5th February 2024.

Calendar plan

№ by number	The deadline for completing the stages of the diploma work	Deadline	Notes
1	Study of literature on the topic of work	13.04.2024	Completed
2	Preparation of the first chapter	20.04.2024	Completed
3	Preparation of the second chapter	26.04.2024	Completed
4	Development of a software product	08.05.2024	Completed
5	Preparation of the third section	11.05.2024	Completed
6	Preparation of the economic part	22.05.2024	Completed
7	Design of sections	25.05.2024	Completed
8	Preparation of the presentation of the repor	04.06.2024	Completed
9	Preparation of thesis	06.06.2024	Completed

Student Lev MAZNICHENKO

Supervisor Vira HUSKOVA

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Системи і методи штучного інтелекту»

ЗАТВЕРДЖУЮ
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«31» січня 2024 р.

ЗАВДАННЯ
на дипломну роботу студенту
Мазніченку Леву Владиславовичу

1. Тема роботи « Прогнозування та аналіз демографічних показників розвинених та розвиваючихся країн з використанням часових рядів», керівник роботи Гуськова Віра Геннадіївна, затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом роботи «10» червня 2024 року.

3. Вихідні дані до роботи: Демографічні показники розвинених країн та країн, що розвиваються

4. Зміст роботи: 1. Аналіз предметної області; 2. Математичні та теоретичні основи роботи; 3. Розробка моделі прогнозування; 4. Функціональний та вартісний аналіз програмного продукту.

5. Перелік ілюстративного матеріалу: презентація, результуючі графіки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Шевчук Олена Анатоліївна, професор, д. е. н.		

7. Дата видачі завдання «05» лютого 2024 року.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Вивчення літератури за темою роботи	13.04.2024	Виконано
2	Підготовка першого розділу	20.04.2024	Виконано
3	Підготовка другого розділу	26.04.2024	Виконано
4	Розробка програмного продукту	08.05.2024	Виконано
5	Підготовка третього розділу	11.05.2024	Виконано
6	Підготовка економічної частини	22.05.2024	Виконано
7	Оформлення розділів	25.05.2024	Виконано
8	Підготовка презентації доповіді	04.06.2024	Виконано
9	Оформлення дипломної роботи	06.06.2024	Виконано

Студент Лев МАЗНІЧЕНКО

Керівник Віра ГУСЬКОВА

ABSTRACT

Bachelor thesis: 107 p., 6 figures, 5 tables, 32 references, 3 appendixes.

TIME SERIES, FORECASTING, TIME SERIES ANALYSIS, DEMOGRAPHICS, TOTAL FERTILITY RATE, DEMOGRAPHIC TRANSITION, ARIMA, FERTILTY, MOVING AVERAGE.

The object of the study is the demographics of developed and developing countries.

The subject of research is the time series analysis and forecasting methods applied to demographic data.

The purpose of the work is to is to develop and evaluate a model for predicting demographic trends in various countries using time series analysis techniques.

The relevance of this thesis lies in the development of the field of demographics (TFR) forecasting, which is essential for understanding and planning for the needs of society and Humankind itself. Accurate population projections help in policy making, resource allocation and economic planning. Traditional models often fail to capture the nuances of demographic changes, especially under different economic, political and cultural conditions.

During research, a sophisticated forecasting model was developed to predict demographic trends in both developed and developing countries. This model uses advanced time series analysis and forecasting techniques to provide increased accuracy and insight into future demographic shifts, thereby assisting in effective decision-making and planning.

РЕФЕРАТ

Дипломна робота: 107 с., 6 рисунків, 5 таблиць, 32 найменування, 3 додатки.

ЧАСОВІ РЯДИ, ПРОГНОЗУВАННЯ, АНАЛІЗ ЧАСОВИХ РЯДІВ, ДЕМОГРАФІЯ, СУМАРНИЙ КОЕФІЦІЄНТ НАРОДЖУВАНОСТІ, ДЕМОГРАФІЧНИЙ ПЕРЕХІД, АРІМА, ФЕРТИЛЬНІСТЬ, КОВЗНА СЕРЕДНЯ.

Об'єктом дослідження є демографія розвинених країн та країн, що розвиваються.

Предметом дослідження є методи аналізу часових рядів та прогнозування, що застосовуються до демографічних даних.

Метою роботи є розробка та оцінка моделі для прогнозування демографічних тенденцій в різних країнах з використанням методів аналізу часових рядів.

Актуальність даної дипломної роботи полягає у розвитку галузі демографічного прогнозування (СКР), яка є важливою для розуміння та планування потреб суспільства та самого людства. Точні демографічні прогнози допомагають у формуванні політики, розподілі ресурсів та економічному плануванні. Традиційні моделі часто не можуть врахувати нюанси демографічних змін, особливо за різних економічних, політичних і культурних умов.

Під час дослідження було розроблено складну модель прогнозування для передбачення демографічних показників як у розвинених країнах, так і в країнах, що розвиваються. Ця модель використовує передові методи аналізу часових рядів і прогнозування, щоб забезпечити підвищену точність і розуміння майбутніх демографічних зрушень, тим самим допомагаючи в ефективному прийнятті рішень і плануванні.

CONTENTS

INTRODUCTION

Analysis and forecasting of time series is extremely important for many fields, where certain indicators tend to change over time. One of such areas is demography, where literally everything is changing over time.

This thesis will mainly focus on demographics of developed and developing countries, as they are much more predictable and have statistics that can be trusted, which is not the case with least developed countries.

Recently we can see how the whole world is changing rapidly - Africa and Asia have long ago surpassed Europe and both Americas in terms of population, there is a rapid growth of population in African countries at the moment, which, according to the UN and leading experts, will soon surpass China and other countries that in the past were leading in terms of population.

Migration rates and flows are also changing - migration from some countries where the population is either growing very slowly or declining is decreasing, and migration from countries with rapid population growth is increasing. In many countries, the ethno-religious composition is changing - as members of different groups – both ethnic and religious – within the same country can have very different demographics.

However, what is most important – and what is the cornerstone of this paper– is that these indicators are all changing over time, and by knowing the current trends in demographics changes, their causes and consequences, we can try to predict future indicators with some confidence.

The purpose of this paper is to develop a model that will allow us to correctly predict future changes in the demographics of a particular developed or developing country based on data about that country's demographics.

The objectives of this thesis are:

The first section describes what is time series analysis and forecasting, methods of working with time series, data processing methods and other possibilities of working with time series, their analysis and forecasting.

The second section reviews existing software solutions for analyzing and forecasting time series and forecasting demographic indicators on the basis of available data.

SECTION 1 ANALYSIS OF THE SUBJECT AREA

1.1 Task relevance

Forecasting and analyzing demographic indicators is obviously a sub-task of forecasting and analyzing time series as such. This task is incredibly complex and multifaceted and many researchers are trying to work on it. The nuance is that qualitative research requires large resources - both human and in terms of time. For developed countries this is less of a problem, as quality statistics on demographic data are often provided by the state, but for developing and least developed countries it is a frequent problem, also because the state is interested in changing the indicators for one reason or another to make the country look more attractive. However, this is not limited to countries, international organizations are often affected - the UN is often criticized for politicizing its forecasts or conclusions, the clearest example of which is China - where population decline has already begun [4] and fertility is falling rapidly [5], but the UN in its older forecasts always put China on a fertility plateau of 1.5 children per woman [6], which many researchers rightly doubted. Unfortunately, or fortunately, only states and large international organizations are interested in this kind of analysis and forecasting, since for business such analysis is irrelevant or too inaccurate due to specificity and variability.

Also, in many countries there are regional peculiarities that are often not taken into account in these or those forecasts, which are based only on quantitative indicators, such as religion, support of the state, the level of atomization of society, its development, and so on.

An extremely important term in demography is Demographic Transition [7]. Researchers still argue whether there is only one demographic transition that stretches over a couple of centuries or two [8], but they all agree on approximately the same changes in society, and those researchers who are inclined to the existence of a second demographic transition as a phenomenon often associate it with the second epidemiological transition [9].

The *first or "classic" demographic* transition refers to the historical declines in mortality and fertility, as witnessed from the 18th century onward in several European populations and continuing at present in most developing countries. The end point of the first demographic transition (FDT) was supposed to be an older stationary population corresponding with replacement fertility (i.e., just over two children on average), zero population growth, and life expectancies higher than 70 y. Because there would be an ultimate balance between deaths and births, there would be no "demographic" need for sustained immigration. Moreover, households in all parts of the world would converge toward the nuclear and conjugal type, composed of married couples and their offspring. Such were the expectations in the early 1970s. Thereafter, as the baby boom of the 1960s was followed by the baby bust of the 1970s, these expectations were altered to accommodate the possibility of oscillating fertility as a function of labor-market conditions [3].

The second demographic transition (SDT) viewpoint, jointly formulated by Lesthaeghe and van de Kaa in 1986 [10], in contrast, sees no such equilibrium as the end point. Rather, they argue that new developments from the 1970s onward can be expected to bring about sustained subreplacement fertility, a multitude of living arrangements other than marriage, a disconnection between marriage and procreation, and no stationary population [11, 12]. Furthermore, populations will face declining sizes if not complemented by new migrants (i.e., "replacement migration"), and they will also be much older than envisaged by the FDT as a result of lower fertility and considerable additional gains in longevity. Migration streams will not be capable of stemming aging altogether, however, because migrants also age and lower their own fertility with time spent in receiving nations.

The second epidemiologic transition is characterized as the "Age of Receding Pandemics", and is marked by declining mortality rates that become steeper as epidemics occur less frequently, an increase in average life expectancy from about 30 years to about 50 years of age, and more sustained population growth that eventually [13].

To summarize, we can say that the following institutions are interested in forecasting demographic indicators:

- international organizations;
- research institutes;
- governments.

And for all of them, the existence tool, which could take into account nuances and free from politicized editing would be extremely useful and necessary.

1.2 Definition, goals, and methods of time series analysis and forecasting

A *time series* is a sequence of observations taken sequentially in time.A time series is a sequence of observations taken sequentially in time. Many sets of data appear as time series: a monthly sequence of the quantity of goods shipped from a factory, a weekly series of the number of road accidents, daily rainfall amounts, hourly observations made on the yield of a chemical process, and so on. Examples of time series abound in such fields as economics, business, engineering, the natural sciences (especially geophysics and meteorology), and the social sciences. An intrinsic feature of a time series is that, typically, adjacent observations are dependent. The nature of this dependence among observations of a time series is of considerable practical interest [1, 2].

Time series analysis is concerned with techniques for the analysis of this dependence. This requires the development of stochastic and dynamic models for time series data and the use of such models in important areas of application [1].

Time series forecasting is forecasting of future values of a time series from current and past values [1,2].

Amongst goals in time series analysis and forecasting we can derive.

Exploratory analysis for time series mainly involves visualization with time series plots, decomposition of the series into deterministic and stochastic parts, and studying the dependency structure in the data [2[1]].

The formulation of a stochastic model, as it is for example also done in regression, can and does often lead to a deeper understanding of the series. The formulation of a suitable model usually arises from a mixture between background knowledge in the applied field, and insight from exploratory analysis. Once a suitable model is found, a central issue remains, i.e. the estimation of the parameters, and subsequent model diagnostics and evaluation.

An often-heard motivation for time series analysis is the forecasting of future observations in the series. This is an ambitious goal, because time series forecasting relies on extrapolation, and is generally based on the assumption that past and present characteristics of the series continue. It seems obvious that good forecasting results require a very good comprehension of a series' properties, be it in a more descriptive sense, or in the sense of a fitted model.

Time series regression – rather than just forecasting by extrapolation, we can try to understand the relation between a so-identified response time series, and one or more explanatory series. If all of these are observed at the same time, we can in principle employ the ordinary least squares (OLS) regression framework. However, the all-to-common assumption of (serially) uncorrelated errors in OLS is usually violated in a time series setup. We will illustrate how to properly deal with this situation, in order to generate correct confidence and prediction intervals.

Process control – many production or other processes are measured quantitatively for the purpose of optimal management and quality control. This usually results in time series data, to which a stochastic model is fit. This allows understanding the signal in the data, but also the noise: it becomes feasible to

[1] here and further

monitor which fluctuations in the production are normal, and which ones require intervention.

Amongst important definitions we will work with:

Trend is a systematic change in the mean level of a time series that does not appear to be periodic.

Seasonality is a repeating pattern over a fixed period.

Stationarity implies that the probabilistic character of the series must not change over time, i.e. that any section of the time series is “typical” for every other section with the same length.

Autocorrelation of two random variables X_{t+k} and X_t can be defined as a function of a lag k: $\rho(k) = \mathrm{Cor}(X_{t+k}, X_t)$.

A time series $(W_1, W_2, \dots, W_n)$is called White Noise if the random variables $W_1, W_2, \dots, W_n$ are independent and identically distributed with mean zero .

A moving average process of order q, or abbreviated, an MA(q) model for a series X_t is a linear combination of the current innovation term E_t, plus the q most recent ones $E_{t-1}, \dots, E_{t-q}$. The model equation is $X_t = E_t + \beta_1 E_{t-1} + \cdots + \beta_q E_{t-q}$.

An autoregressive model of order p , abbreviated as AR(p) , is based on a linear combination of past observations according to the following equation: $X_t = \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \dots + \alpha_p X_{t-p} + E_t$ [1, 2].

A series X_t follows an ARIMA(p, d , q) model if the d-th order lag 1 difference of X_t is an ARMA(p, q) process.

1.3 Definitions and important notes of demographics

Total fertility rate (TFR) compares figures for the average number of children that would be born per woman if all women lived to the end of their childbearing years and bore children according to a given fertility rate at each age. TFR is a more

direct measure of the level of fertility than the crude birth rate, since it refers to births per woman [29].

Crude Birth Rate (CBR): The number of live births per 1,000 people in a given year. This rate helps gauge the birth trends within a population [30].

Crude Death Rate (CDR): The number of deaths per 1,000 people in a given year. This rate is vital for understanding mortality within a population [30].

Population Density: The number of people living per unit of area, often per square kilometer or mile. This metric helps assess how crowded an area is [30].

Life Expectancy: The average number of years a person can expect to live, based on current mortality rates. This is an indicator of the overall health and quality of life in a population [30].

Dependency Ratio: The ratio of non-working (young and old) to working-age population. This ratio indicates the economic burden on the productive segment of the population [30].

Migration Rate: The difference between the number of immigrants and emigrants per 1,000 people. Migration significantly affects population size and composition [30].

It is important to understand that after second demographic transition, when TFR falls below 2.05 (2.10 for some countries) this variable is highly dependent on economic situation in the country and the government programs. In results from section 3 we will see that for many countries a significant drop can be observed during recessions and crisis' – more of economic sense, than political – Sweden in 2015 (refugee crisis which led to drop in living standard), Ukraine, Russia and Bulgaria after 1990(economic crisis after fall of USSR), USA in 2008, Iran after drop of oil prices and significant economic sanctions in 2016 [31].

However, significant financial help can increase TFR rapidly, especially if this financial help includes all possible groups of women, such programs exist (and successfully improve situation) in Republic of Korea, Russia, Hungary, France and Ukraine [32, 33].

Effect of these measures can also be seen on graphs – “New family policy” program started in 2004 in Ukraine – and even the 2008 financial crisis could not stop the TFR growth in this country. Similar situation is seen in Hungary – second Orban government raised family expenses from 0,1% of GPD to 4.9% in a span of couple years, which can be seen in data starting from 2011.

1.4 Stages of building the model

Steps that we need to perform in order to successfully develop and build our analytical and predictive model are quite trivial and similar to *cross-industry* standard process for data mining [14] which can be described as following.

1. Business understanding: the first step is trying to get a better idea of what business needs should be extracted from data. The analyst has to understand what the customer really wants from a business perspective. The customer often has several competing goals and restrictions that need to be properly coordinated. Moreover the business understanding phase is about defining the specific goals and requirements for data mining. The result of this phase is the formulation of the task and the description of the planned rough procedure to achieve both business and data mining goals [14, 15 [2]].
2. Data Understanding: as part of the data understanding, an attempt is made to get a first overview of the available data and their quality. This involves checking whether all the required data (to meet the Data Mining goals) is actually available as well as developing a plan to determine which data is required.

---

[2] here and further

3. Data Preparation: in this phase, the data is prepared for the further data mining process. Data preparation is one of the most important and often time-consuming aspects of data mining. In fact, it is estimated that data preparation usually takes 50-70% of a project's time and effort. Business decisions rely on analytics. But, if the data is inaccurate or incomplete, your analytics inform wrong business decisions. Bad analytics means poor business decisions.
4. Modeling: modeling is the analytical core of the data mining process. This is where the selection and use of modeling techniques take place. Before actually building a model, you typically separate the dataset into train, test and validation sets. Then you build the models on the train set. Many modeling techniques make specific assumptions about the data, for example that all attributes have uniform distributions or no missing values are allowed. Besides, with any modeling tool there are often a large number of parameters that can be adjusted. You have to record any assumptions made and list the parameters and their chosen values.
5. Evaluation: the evaluation ensures an exact comparison of the created data models with the task and selects the most suitable model. The results of the previous steps are evaluated using the business criteria established at the beginning of the project. So this phase is about checking whether the data mining solution satisfies the business problem and seeking to determine if there is some business reason why a model is deficient.
6. Deployment: after data preparation, model building and model verification, the selected model is used in the deployment phase. Generating a model is generally not the end of the project. Even if the goal was to deepen the knowledge of the data, the knowledge gained must now be processed and presented to the customer so that the customer can use it without any problems.

In our case everything follows steps, described above. Important note is that all these steps are interchangeable in terms of order.

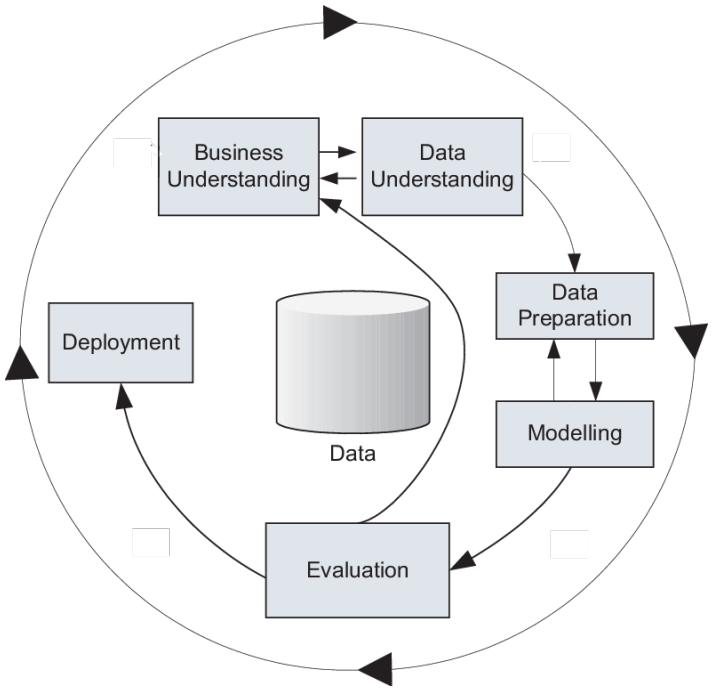


Figure 1.1 – CRISP-DM structure [14]

1.5 Problem statement

Let's define and specify the tasks for the effective implementation of demographics forecasting and analysis. It is necessary to prepare and form a set of data sufficient to use time series methods and models and further research and obtain the expected positive or negative results. With the help of the developed model, it will be possible to obtain a TFR for a country based on the demographics data of this country and it's specifications. In addition, it is necessary to compare the algorithms and methods of time series forecasting and analyse their impact on the accuracy of the forecasting model, the impact of the percentage of the data set divided into training and test data, the quality of existing data for selected country and the impact of gaps in the dataset.

The forecasting problem can be formulated and defined as follows.

1. Form a dataset with demographic data of a countries using existing databases.
2. Research methods, approaches and models of time series analysis and forecasting for processing the dataset and performing modelling and forecasting of TFR.
3. Review modern tools and software for time series analysis and forecasting.
4. Justify the choice of environment and programming language for solving the problem of developing a software package.
5. Carry out exploratory analysis of raw data (initial data analysis, data visualization).
6. Carry out pre-processing of the dataset (converting categorical columns into numerical ones, filling in gaps, normalizing data).
7. Develop mathematical models for predicting TFR.
8. Apply existing models in order to forecast TFR (ARMA, ARIMA, ARMAX).
9. Analyze the results of training and research.

1.6 Overview of existing tools for time series analysis and forecast

The main programming languages used for time series analysis and forecasting are R and Python. Both of these languages have an extremely convenient set of libraries, which is suitable for preprocessing and data visualisation, as well as for certain types of data analysis with subsequent use of models for forecasting.

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application

Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues. R can be considered as a different implementation of S. There are some important differences, but much code written for S runs unaltered under R [16].

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, …) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity [17].

For this thesis we will use Python, as the author is more familiar with this language, as Python has more libraries and tools, is more flexible and more convenient for visualization and interactive programming using Jupyter Notebook.

On the other hand, R is a more popular language among researchers because of its more convenient code format and the ability to create documents using R Markdown, however, since the bulk of this thesis is written in MS Word and Python will only be used for the actual development and testing of the model, it is easy to conclude that Python is the more logical and justified choice of programming language.

Jupyter Notebook (formerly known as IPython Notebook) - is an interactive web application for creating and sharing computational documents. The project was first named IPython and later renamed Jupyter in 2014. It is a fully open-source

product, and users can use every functionality available for free. It supports more than 40 languages including Python, R, and Scala [20].

PyCharm is an integrated development surroundings (IDE) used for programming in Python. It is advanced by means of the Czech organization JetBrains and is to be had for Windows, macOS, and Linux. PyCharm is to be had in two variations: Community Edition and Professional Edition. The Community Edition is loose and open supply, while the Professional Edition is a paid subscription carrier [21].

Since we have chosen Python, following libraries were used for this thesis.

NumPy is a popular Python package for numerical and scientific computations. It is the de facto paradigm in Python for numerical data. It is the foundation for several other numerical and time-series libraries, directly or indirectly, via pandas. NumPy offers an advanced interface for rapid numerical processing. This is an indispensable tool for data scientists and analysts dealing with massive amounts of numerical information. It also includes several scientific and statistical functions, such as linear algebra procedures like linear regression [18[3]].

Pandas is a practical data analysis and manipulation toolkit. It includes fundamental data structures, most notably the Data Frame, for expressing and interacting with statistical and numerical information. Pandas handles various types of data, including time-series data.

Matplotlib, a popular Python plotting library, offers numerous data visualization options. It enables users to build both basic exploration plots and complex scientific visualizations. With Matplotlib, you can demonstrate your imagination while effectively displaying your data. The graphs include bagplots, line charts, histograms, and line graphs. It also excels at dealing with time-series data, allowing you to build visually appealing visualizations.

---

[3] here and further

Matplotlib recognizes the significance of visualizing time-dependent data, an essential part of time-series analysis [18].

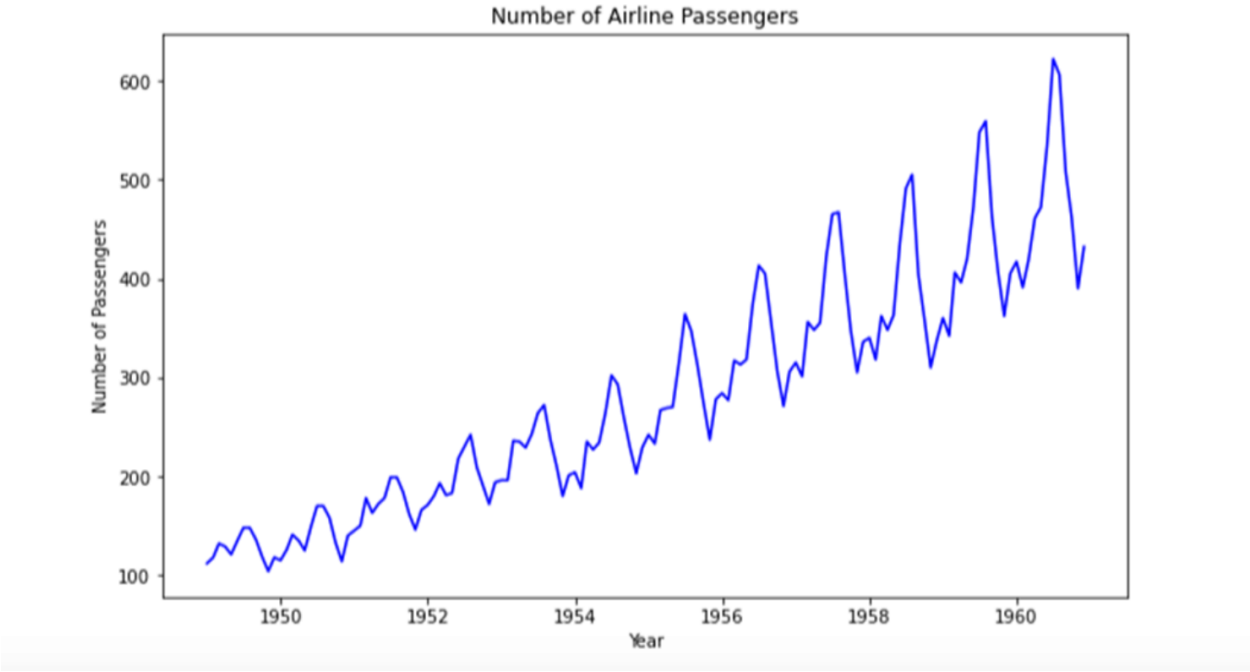


Figure 1.2 – Example of plotting using matplotlib library

Sktime is an open-source Python machine-learning package specializing in time-series data. It is meant to be interoperable with scikit-learn, which means that while dealing with time-series data, users may take advantage of the capabilities of scikit-learn's algorithms and evaluation metrics.

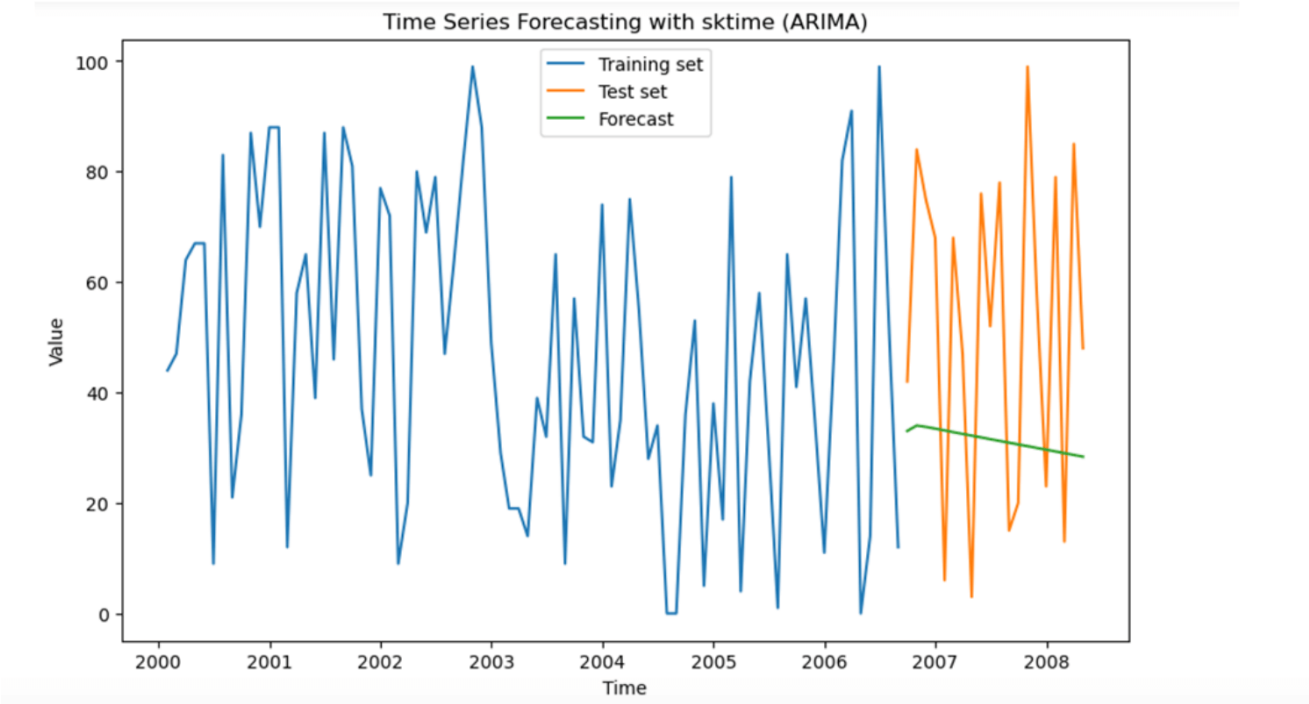


Figure 1.3 – Example of visualization of time series data using Sktime and matplotlib

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics [1].

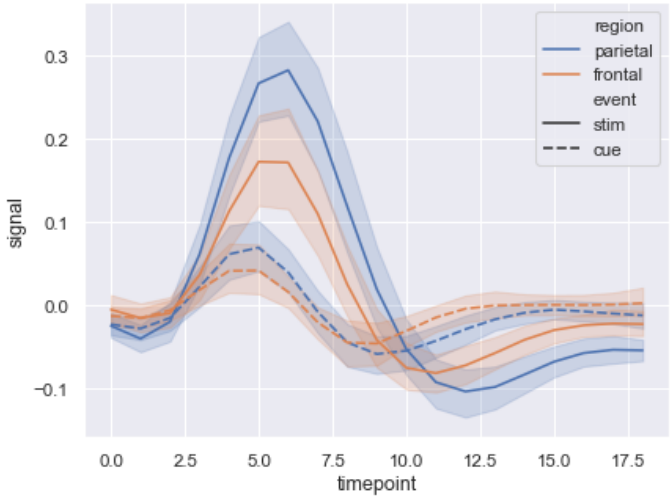


Figure 1.4 – Example of visualization of time series data using Sktime and matplotlib

Conclusions to the section 1

In the first section, the concept of time series and demography was considered, the purpose of time series analysis and forecast was defined, the structure of future datasets was described, methods, models, algorithms were defined, the types of tasks solved by the time series analysis tools was carried out, the main stages of the research were identified and studied, and the tasks of the study of demographics forecasting were formulated, modern tools for the implementation of Time Series Forecasting and Analysis are considered, the expediency of using the capabilities of the Python programming language, the PyCharm code editor and the Jupyter Notebook IDE is substantiated, libraries and modules in Python that should be used to perform the task of price forecasting are identified.

SECTION 2 THEORETICAL AND MATHEMATICAL FOUNDATIONS OF WORK

2.1 Mathematical concepts

To perform anything beyond basic exploratory time series analysis, even from a more applied point of view, it is necessary to introduce the mathematical concept of what a time series is and to learn some basic probabilistic properties, namely moments and the concept of stationarity. We already described main concepts in Section 1, however, it is obviously necessary to derive them from more formal point of view.

2.1.1 Definition of a Time Series

As we have explained in section 1.2, observations that have been collected over fixed sampling intervals form a time series. Following a statistical approach, we consider such series as realizations of random variables. A sequence of random variables, defined at such fixed sampling intervals, is sometimes referred to as a discrete-time stochastic process, though the shorter names time series model or time series process are more popular and will mostly be used in this scriptum. It is very important to make the distinction between a time series, i.e. observed values, and a process, i.e. a probabilistic construct [1, 2[4]].

Definition: a time series process is a set of random variables $\{X_t, t \in T\}$, where T is the set of times at which the process was, will or can be observed. We assume that each random variable X_t is distributed according some univariate distribution

[4] here and further

function E_t . Please note that for our entire course and hence scriptum, we exclusively consider time series processes with equidistant time intervals, as well as real-valued random variables X_t . This allows us to enumerate the set of times, so that we can write $T = \{1,2,3,\dots\}$.

An observed time series, on the other hand, is seen as a realization of the random vector $X = \{X_1, X_2, \dots, X_n\}$, and is denoted with small letters $x = (x_1, x_2, \dots, x_n)$. It is important to note that in a multivariate sense, a time series is only one single realization of the n-dimensional random variable X , with its multivariate, n-dimensional distribution function $F_{1:n}$. As we all know, we cannot do statistics with just a single observation. As a way out of this situation, we need to impose some conditions on the joint distribution function $F_{1:n}$.

2.1.2 Trend, seasonality, cycles and residuals

One simple method of describing a series is that of *classical decomposition*. The notion is that the series can be decomposed into four elements:

- *trend* (T_t) — long term movements in the mean;
- *seasonal effects* (I_t) — cyclical fluctuations related to the calendar;
- *cycles* (C_t) — other cyclical fluctuations (such as a business cycles);
- *residuals* (E_t) — other random or systematic fluctuations.

The idea is to create separate models for these four elements and then combine them, either additively:

$$X_t = T_t + I + C_t + E_t.$$

or multiplicatively:

$$X_t = T_t \cdot I \cdot C_t \cdot E_t \text{ [22].}$$

2.1.3 Stationarity

The aforementioned condition on the joint distribution $F_{1:n}$ will be formulated as the concept of *stationarity*. In colloquial language, stationarity means that the probabilistic character of the series must not change over time, i.e. that any section of the time series is "typical" for every other section with the same length. More mathematically, we require that for any indices *s, t* and *k*, the observations $x_t, \ldots, x_{t+k}$ could have just as easily occurred at times $s, \ldots, s+k$. If that is not the case practically, then the series is hardly stationary.

Imposing even more mathematical rigor, we introduce the concept of strict stationarity. A time series is said to be strictly stationary if and only if the

$(k+1)$ -dimensional joint distribution of $X_t, \ldots, X_{t+k}$ coincides with the joint distribution of $X_s, \ldots, X_{s+k}$ for any combination of indices *t, s* and *k*. For the special case of $k = 0$ and $t = s$, this means that the univariate distributions F_t of all X_t are equal. For strictly stationary time series, we can thus leave off the index t on the distribution. As the next step, we will define the unconditional moments:

– expectation $\mu = E[X_t]$;
– variance $\mu = Var(X_t)$;
– covariance $\mu = Cov(X_t, X_{t+h})$.

In other words, strictly stationary series have *constant (unconditional) expectation, constant (unconditional) variance*, and the covariance, i.e. the dependency structure, depends only on the *lag h*, which is the time difference between the two observations. However, the covariance terms are generally different from 0, and thus, the X_t are usually dependent. Moreover, the conditional expectation given the past of the series, $E[X_t|X_{t-1}, X_{t-2}, \ldots]$, is typically non-constant, denoted as μ_t . In some (rarer, e.g. for financial time series) cases, even the conditional variance $Var(X_t|X_{t-1}, X_{t-2}, \ldots)$ can be non-constant.

In practice however, except for simulation studies, we usually have no explicit knowledge of the latent time series process. Since strict stationarity is defined as a property of the process' joint distributions (all of them), it is impossible to verify from an observed time series, i.e. a single data realization. We can, however, try to verify whether a time series process shows constant unconditional mean and variance, and whether the dependency only depends on the lag h. This much less rigorous property is known as weak stationarity.

In order to do well-founded statistical analyses with time series, weak stationarity is a necessary condition. It is obvious that if a series' observations do not have common properties such as constant mean/variance and a stable dependency structure, it will be impossible to statistically learn from it. On the other hand, it can be shown that weak stationarity, along with the additional property of ergodicity (i.e. the mean of a time series realization converges to the expected value, independent of the starting point), is sufficient for most practical purposes such as model fitting, forecasting, etc.. We will, however, not further embark in this subject.

2.1.4 Autoregressive Processes

The *autoregressive process* of order p is denoted AR(p), and defined by:

$$X_t = \sum_{r=1}^{p} \phi_r X_{t-r} + \epsilon_t,$$

where $\phi_1, \ldots, \phi_r$ are fixed constants and $\{\epsilon_r\}$ is a sequence of independent (or uncorrelated) random variables with mean 0 and varianceσ^2.

The AR(1) process is defined by:

$$X_t = \phi_1 X_{t-1} + \epsilon_t.$$

To find its autocovariance function we make successive substitutions, to get:

$$X_t = \epsilon_t + \phi_1\big(\epsilon_{t-1} + \phi_1(\epsilon_{t-2} + \cdots)\big) = \epsilon_t + \phi_1\epsilon_{t-1} + \phi_1^2\epsilon_{t-2} + \ldots$$

The fact that $\{X_t\}$ is second order stationary follows from the observation that $\mathbb{E}(X_t) = 0$ and that the autocovariance function can be calculated as follows:

$$\gamma_0 = \mathbb{E}(\epsilon_t + \phi_1\epsilon_{t-1} + \phi_1^2\epsilon_{t-2} + \cdots)^2 = (1 + \phi_1^2 + \phi_1^4 + \cdots)\sigma^2 = \frac{\sigma^2}{1-\phi_1^2}.$$

$$\gamma_k = \mathbb{E}\left(\sum_{r=0}^{\infty}\phi_1^r\epsilon_{t-r}\sum_{s=0}^{\infty}\phi_1^s\epsilon_{t+k-s}\right) = \frac{\sigma^2\phi_1^k}{1-\phi_1^2}.$$

There is an easier way to obtain these results. Multiply previous equation by X_{t-k} and take the expected value, to give:

$$E(X_tX_{t-k}) = E(\phi_1X_{t-1}X_{t-k}) + E(\epsilon_tX_{t-k}).$$

Thus:

$$\gamma_k = \phi_1\gamma_{k-1}, k = 1, 2, \ldots.$$

Similarly, squaring autocovariance function and taking the expected value gives:

$$E(X_t^2) = \phi_1E(X_{t-1}^2) + 2\phi_1E(X_{t-1}\varepsilon_t) + E(\varepsilon_t^2) = \phi_1^2E(X_{t-1}^2) + 0 + \sigma^2,$$

$$\text{and so} = \sigma^2/(1-\phi_1^2).$$

More generally, the AR(p) process is defined as:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t.$$

Again, the autocorrelation function can be found by multiplying X_t by $X_t - k$, taking the expected value and dividing by γ_0, thus producing the Yule-Walker equations:

$$\rho_k = \phi_1 \rho_{k-1} + \phi_2 \rho_{k-2} + \cdots + \phi_p \rho_{k-p}, \qquad k = 1, 2, \ldots$$

These are linear recurrence relations, with general solution of the form:

$$\rho_k = C_1 \omega_1^k + \cdots + C_p \omega_p^k,$$

where $\omega_1, \ldots, \omega_p$ are the roots of

$$\omega^p - \phi_1 \omega^{p-1} - \phi_2 \omega^{p-2} - \cdots - \phi_p = 0,$$

and $C_1, \ldots, C_p$ are determined by $\rho_0 = 1$ and the equations for $k = 1, \ldots, p - 1$. It is natural to require $\gamma_k \to 0$ as $k \to \infty$, in which case the roots must lie inside the unit circle, that is, $|\omega_1| < 1$. Thus there is a restriction on the values of $\omega_1, \ldots, \omega_p$ that can be chosen [1, 2, 22[5]].

2.1.5 Moving Average Processes

[5] here and further

The *moving average process* of order q is denoted *MA(q)* and defined by:

$$X_t = \sum_{s=0}^{q} \theta_s \varepsilon_{t-s},$$

where $\theta_1, \ldots, \theta_q$ are fixed constants, $\theta_0 = 1$, and $\{\epsilon_t\}$ is a sequence of independent (or uncorrelated) random variables with mean 0 and variance σ^2.

It is clear from the definition that this is second order stationary and that:

$$\gamma_k = \begin{cases} 0, & |k| > q \\ \sigma^2 \sum_{s=0}^{q-|k|} \theta_s \theta_{s+k}, & |k| \leq q \end{cases},$$

We remark that two moving average processes can have the same autocorrelation function. For example:

$$X_t = \varepsilon_t + \theta \varepsilon_{t-1}, \text{ and } \quad X\_t = \varepsilon\_t + (1\backslash\theta)\varepsilon_{t-1},$$

both have:

$$\rho_1 = \frac{\theta}{1+\theta^2}, \quad \rho_k = 0, \quad \text{for } |k| > 1.$$

However, the first gives:

$$\varepsilon_t = X_t - \theta \varepsilon_{t-1} = X_t - \theta(X_{t-1} - \theta \varepsilon_{t-2}) = X_t - \theta X_{t-1} + \theta^2 X_{t-2} - \ldots$$

This is only valid for $|\theta| < 1$, a so-called invertible process. No two invertible processes have the same autocorrelation function.

2.1.5 White noise

The sequence $\{\epsilon_t\}$, consisting of independent (or uncorrelated) random variables with mean 0 and variance σ^2 is called *white noise* (for reasons that will become clear later.) It is a second order stationary series with $\gamma_0 = \sigma^2$ and $\gamma_k = 0$, $k \neq 0$.

2.2 Models of stationary processes

2.2.1 Purely indeterministic processes

Suppose X_t is a second order stationary process, with mean 0. Its *autocovariance function* is:

$$\gamma_k = E(X_t X_{t+k}) Cov(X_t, X_{t+k}), \quad k \in Z.$$

1. As X_t is stationary, γ_k does not depend on t.
2. A process is said to be *purely-indeterministic* if the regression of X_t on $X_{t-q}, X_{t-q-1}, \ldots$ has explanatory power tending to 0 as $q \to \infty$. That is, the residual variance tends to $var(X_t)$.

An important theorem due to Wold (1938) states that every purely indeterministic second order stationary process $\{X_t\}$ can be written in the form:

$$X_t = \mu + \theta_0 Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} + \cdots,$$

where $\{Z_t\}$ is a sequence of uncorrelated random variables.

3. A Gaussian process is one for which $X_t, \ldots, X_{t_n}$ has a joint normal distribution for all $t_1, \ldots, t_n$. No two distinct Gaussian processes have the same autocovariance function.

2.2.2 ARMA Processes

The autoregressive moving average process, ARMA(p,q), is defined by:

$$X_t = \sum_{r=1}^{p} \phi_r X_{t-r} - \sum_{s=0}^{q} \theta_s \epsilon_{t-s},$$

where again $\{\epsilon_t\}$ is white noise. This process is stationary for appropriate ϕ, θ.

2.2.3 ARIMA Processes

If the original process $\{Y_t\}$ is not stationary, we can look at the first order difference process:

$$X_t = \nabla Y_t = Y_t - Y_{t-1},$$

or the second order differences:

$$X_t = \nabla^2 Y_t = \nabla(\nabla Y_t) = Y_t - 2Y_{t-1} + Y_{t-2},$$

and so on. If we ever find that the differenced process is a stationary process we can look for a ARMA model of that.

The process $\{Y_t\}$ is said to be an autoregressive integrated moving average process, ARIMA(p,d,q), if $X_t = \nabla^d Y_t$ is an ARMA(p,q) process.

AR, MA, ARMA and ARIMA processes can be used to model many time series.

A key tool in identifying a model is an estimate of the autocovariance function.

2.2.4 Estimation of the autocovariance function

Suppose we have data $(X_1, \ldots, X_T)$ from a stationary time series. We can estimate

- the mean by mean:

$$\overline{X} = \frac{1}{T}\sum_{t=1}^{T} X_t\,;$$

- the autocovariance by:

$$c_k = \widehat{\gamma_k} = \frac{1}{T}\sum_{t=k+1}^{T}\left(X_t - \overline{X}\right)\left(X_{t-k} - \overline{X}\right);$$

- the autocorrelation by:

$$r_k = \widehat{\rho_k} = \frac{\widehat{\gamma_k}}{\widehat{\gamma_0}}.$$

The plot of r_k against k is known as the *correlogram*. If it is known that μ is 0 there is no need to correct for the mean and γ_k can be estimated by;

$$\widehat{\gamma_k} = \frac{1}{T}\sum_{t=k+1}^{T} X_t X_{t-k}.$$

Notice that in defining $\gamma_k$we divide by T rather than by $(T - k)$. When T is large relative to k it does not much matter which divisor we use. However, for mathematical simplicity and other reasons there are advantages in dividing by T.

Suppose the stationary process $\{X_t\}$ has autocovariance function $\{\gamma_k\}$.Then:

$$var\left(\sum_{t=1}^{T} a_t X_t\right) = \sum_{t=1}^{T}\sum_{s=1}^{T} a_t a_s Cov(X_t, X_s) = \sum_{t=1}^{T}\sum_{s=1}^{T} a_t a_s \gamma_{|t-s|} \geq 0.$$

A sequence $\{\gamma_k\}$ for which this holds for every $T \geq 1$ and set of constants $(a_1, \ldots, a_t)$is called a *nonnegative definite sequence*. The following theorem states that $\{\gamma_k\}$ is a valid autocovariance function if and only if it is nonnegative definite.

Theorem of Blochner The following are equivalent.

1. There exists a stationary sequence with autocovariance function $\{\gamma_k\}$.
2. $\{\gamma_k\}$ is nonnegative definite.
3. The spectral density function,
4.

$$f(\omega) = \frac{1}{\pi}\sum_{k=-\infty}^{\infty} \gamma_k e^{ik\omega} = \frac{1}{\pi}\gamma_0 + \left(\frac{2}{\pi}\sum_{k=1}^{\infty} \gamma_k \cos(\omega k)\right)$$

is positive if it exists.

Dividing by T rather than by $(T - k)$ in the definition of $\widehat{\gamma_k}$ensures that $\{\widehat{\gamma_k}\}$ is nonnegative definite (and thus that it could be the autocovariance function of a stationary process), and can reduce the L^2-error of r_k.

2.2.5 Identifying a MA(q) process

The MA(q) process X_t has $\rho_t = 0$ for all $k, |k| > q$. So a diagnostic for MA(q) is that $|r_k|$ drops to near zero beyond some threshold.

2.2.6 Identifying an AR(p) process

The AR(p) process has ρ_k decaying exponentially. This can be difficult to recognize in the correlogram. Suppose we have a process X_t which we believe is AR(k) with:

$$X_t = \sum_{j=1}^{k} \phi_{j,k} X_{t-j} + \varepsilon_t,$$

with ϵ_t independent of $X_1, \ldots, X_{t-1}$

Given the data $X_1, \ldots, X_T$, the least squares estimates of ($\varphi_{1,k}, \ldots, \varphi_{k,k}$) are obtained by minimizing:

$$\frac{1}{T} \sum_{t=k+1}^{T} \left(X_t - \sum_{j=1}^{k} \phi_{j,k} X_{t-j} \right)^2 .$$

This is approximately equivalent to solving equations similar to the Yule-Walker equations:

$$\widehat{\gamma_j} = \sum_{l=1}^{k} \phi_{l,j}\, \widehat{\gamma_{|j-l|}}, \quad j = 1, \ldots, k.$$

These can be solved by the *Levinson-Durbin recursion*:

Step 0.

$$\sigma_0^2 = \widehat{\gamma_0}, \quad \widehat{\phi_{1,1}} = \frac{\widehat{\gamma_1}}{\widehat{\gamma_0}}, \quad k := 0.$$

Repeat until $\widehat{\phi_{k,k}}$ near 0:

$$k := k + 1,$$

$$\widehat{\phi_{k,k}} := \frac{\widehat{\gamma_k} - \sum_{j=1}^{k-1} \widehat{\phi_{j,k-1}}\, \widehat{\gamma_{k-j}}}{\sigma_{k-1}^2},$$

$$\widehat{\phi_{j,k}} := \widehat{\phi_{j,k-1}} - \widehat{\phi_{k,k}}\widehat{\phi_{k-j,k-1}}, \quad \text{for } j = 1, \ldots, k-1,$$

$$\sigma_k^2 := \sigma_{k-1}^2\left(1 - \widehat{\phi_{k,k}^2}\right).$$

We test whether the order k fit is an improvement over the order $k-1$ fit by looking to see if $\widehat{\phi_{k,k}}$ is far from zero.

The statistic $\widehat{\phi_{k,k}}$ is called the kth *sample partial autocorrelation coefficient* (PACF). If the process X_t is genuinely AR(p) then the population PACF, $\phi_{k,k}$, is exactly zero for all $k > p$. Thus a diagnostic for AR(p) is that the sample PACFs are close to zero for $k > p$.

2.2.7 Distribution of ACF and PACF

Both the sample ACF and PACF are approximately normally distributed about their population values, and have standard deviation of about $1/\sqrt{T}$, where T is the length of the series. A rule of thumb it that ρ_k is negligible (and similarly $\phi_{k,k}$,) if r_k (similarly $\widehat{\phi_{k,k}}$) lies between $\pm 2/\sqrt{T}$,. (2 is an approximation to 1.96. Recall that if $Z_1, \ldots, Z_n \sim N(\mu, 1)$, a test of size 0.05 of the hypothesis $H0 : \mu = 0$ against $H1 : \mu \neq 0$ rejects H_0 if and only if $\bar{z}$ lies outside $\pm 1.96/\sqrt{n}$

Care is needed in applying this rule of thumb. It is important to realize that the sample autocorrelations, r_1 , r_2,.. (and sample partial autocorrelations, $(\widehat{\phi_{1,1}}, \widehat{\phi_{2,2}}, \ldots)$are not independently distributed. The probability that any one r_k should lie outside $\pm 2/\sqrt{T}$ depends on the values of the other r_k.

A 'portmanteau' test of white noise (due to Box & Pierce and Ljung & Box) can be based on the fact that approximately:

$$Q'_m = T(T+2)\sum_{k=1}^{m}(T-k)^{-1}r_k^2 \sim \chi^2_m.$$

.

The sensitivity of the test to departure from white noise depends on the choice of m. If the true model is *ARMA(p,q)* then greatest power is obtained (rejection of the white noise hypothesis is most probable) when m is about $p + q$.

2.2.8 SARIMA process

The seasonal model applied in this research is the most general form of a univariate class of models originally presented by Box and Jenkins [24]. It has been

extensively studied and used in different fields, such as economy, industry and, more recently, in public health. This model building process is designed to take advantage of the association in the sequentially lagged relationships that usually exist in data collected periodically [23[6]].

An important concept for the model building process is that of stationarity, which implies that the probabilistic structure of the series does not change with time. However, very often epidemiological time series in a public health surveillance system have a trend component, that is, they have no fixed mean. It has been found that removal of trends in the mean can usually be achieved by differencing the time series. In this investigation, we modelled the trend in the data with difference equations [24]. A first-order difference of the series (z_t) is given by (w_t), the difference between points in the series one unit apart, calculated as $(w_t = z_t - z_{t-1})$. We may also write (w_t) in terms of the backward shift operator $(B) as (w_t = (1 - B)z_t)$, and so the *d*-th order differencing is obtained as $((1 - B)^d z_t)$.

Besides trend components, epidemiological time series may exhibit seasonality. Box and Jenkins have extended the above idea of ordinary differencing by forming seasonal differences $(w_t = z_t - z_{t-s} = (1 - B^s)z_t)$, where *s* is the seasonal period of the data, say 12 months. Therefore, the most general Box-Jenkins model, the seasonal autoregressive integrated moving average (SARIMA), has the following form:

$$\phi(B)\Phi(B^s)(1 - B^s)^D(1 - B)^d z_t = \theta(B)\Theta(B^s)a_t.$$

With

$$\phi(B) = 1 - \varphi_1 B - \cdots - \varphi_p B^p,$$
$$\theta(B) = 1 - \theta_1 B - \cdots - \theta_q B^q,$$

[6] here and further

$$\Phi(B^s) = 1 - \Phi_1 B^s - \cdots - \Phi_P B^{sP},$$
$$\Theta(B^s) = 1 - \Theta_1 B^s - \cdots - \Theta_Q B^{sQ},$$

where p is the autoregressive order, q the moving average order, d is the number of differencing operations, and PP, DD and QQ are the corresponding seasonal orders.

After removal of trend and seasonal components, the process of model fitting involves identification, parameter estimation and diagnostic validation. At the identification stage, a tentative autoregressive moving average (ARMA) process is developed based on the estimated autocorrelation function (ACF) and the estimated partial autocorrelation function (PACF). The shape of the ACF and PACF of the real epidemiological time series is compared with the shape of the theoretical model. This process of comparison allows the definition of p and q, the orders of the ARMA model.

Having specified an initial model, the parameters of the candidate model are estimated. The last stage, diagnostic checking, examines the residuals to determine the adequacy of the model, or how well the residuals are randomly distributed about zero. If the model is judged adequate, it is used in the forecasting stage. If the fitted model is not adequate, the process of identification, estimate and diagnostic checking is repeated until a satisfactory model is found.

2.2.9 Long short-term memory model

LSTM is a kind of recurrent neural network (RNN) [25]. It has the capability of the earlier stages value remembering. So, the values can be used as the future. LSTM contains four neural network layers and different memory blocks called cells. The first neural network layer is sigmoid neural net layer (0 and 1). The second layer is tanh layer. It shows the cell state. The third layer is the language layer to drop the information and finally the output layer. These layers have been discussed in detail

with Fig. 2.1 explanation. The cells are helpful in the information retention. Gates is helpful in the manipulations of the memory. Fig. 2.1 shows the complete LSTM procedure. Weight matrix shows that it is not updated according to the time at each step while it is fixed based on the inference. It covers the present input along with the past cell status. It has been processed through two inputs. First is the current input and the second is the previous output in the cell. It has been inputted to the gate for the weight matrix multiplication. The sigmoid activation was performed which squishes values between 0 and 1. These values are useful for multiplicative factor as 0 is useful for data disappear and 1 is useful to keep the value same. The information has been removed, which may be not used in the future process. It has been followed by the bias addition. Then it has been converted to 0 (forgotten information) and 1 (retained information) through the activation process. It can be learned by the network. Then cell state and processing came into the picture. It considers the input gate output and perform the addition to updates the cell state (new cell state). The forget gate is used for the information removal which are not relevant. Then the role of input gate is started. In this, the information is regulated through the sigmoid function and the retained information has been fetched. Then, with the help of tanh function a vector is created. The range of this vector is from − 1 to 1. Then it is multiplied to produce the useful information. Then the role of output gate is started. First, by the help of the tanh function a vector is created. Then, with the help of sigmoid function the information is regulated and the retained information has been fetched (Output Squashing). Then these values are multiplied to produce the output for the next cell. Adam optimizer has been used as it is found to be prominent as compared to other stochastic optimization methods. It is found to be useful in handling sparse and noisy problems [25]. It combines the properties of AdaGrad and RMSProp [25].

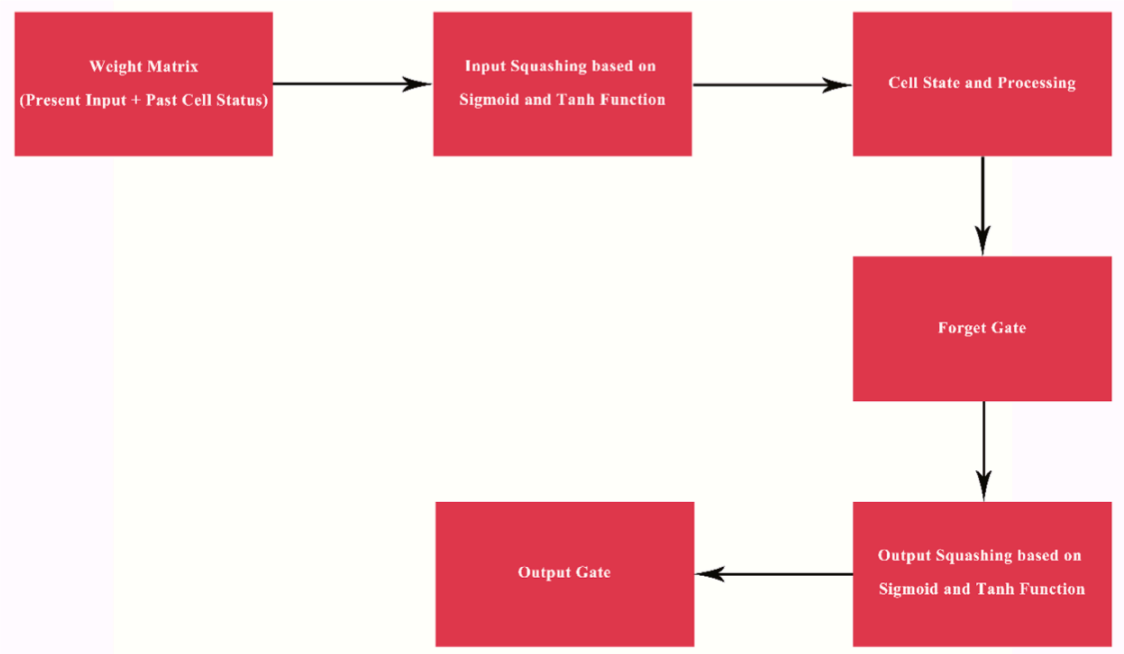


Figure 2.1 – LSTM procedure

2.2.10 Holt-Winters Exponential Smoothing

The Holt-Winters method uses exponential smoothing to encode lots of values from the past and use them to predict "typical" values for the present and future. Exponential smoothing refers to the use of an exponentially weighted moving average (EWMA) to "smooth" a time series. If you have some time series x_t, you can define a new time series s_t that's a smoothed version of x_t:

$$s_t = \alpha x_t + (1 - \alpha) s_{t-1}.$$

In the late 1950s, Charles Holt recognized the issue with the simple EWMA model with time series with trend. He modified the simple exponential smoothing model to account for a linear trend. This is known as Holt's exponential smoothing. This model is a little more complicated. It consists of two EWMAs: one for the

smoothed values of x_t, and another for its slope. The terms level and trend are also used:

$$s_t = \alpha x_t + (1-\alpha)(s_{t-1} + b_{t-1}),$$
$$b_t = \beta(s_t - s_{t-1}) + (1-\beta)b_{t-1}.$$

To forecast with this model, you have to make a slight adjustment. Because there's another term for the slope, you'll have to consider that in the forecast. Suppose you're trying to forecast the value in mm time steps in the future. The formula for the mm-step-ahead forecast, $F_{t+m}$is:

$$F_{t+m} = s_t + mb_t.$$

Holt's student, Peter Winters, extended his teacher's model by introducing an additional term to factor in seasonality. Notice how there's another variable LL, which depends on the period of the seasonality and has to be known in advance.

The three aspects of the time series behavior—value, trend, and seasonality—are expressed as three types of exponential smoothing, so Holt-Winters is called triple exponential smoothing. The model predicts a current or future value by computing the combined effects of these three influences. The model requires several parameters: one for each smoothing (α, β, γ), the length of a season, and the number of periods in a season.

Conclusions to the section 2

In the second section, we derived main mathematical and theoretical foundations of our work, the definition of ARMA and ARIMA models was defined, the structure of future work was described, methods, models, algorithms were

defined, the types of tasks that will be solved by the time series analysis tools was carried out, the main point of foundations were identified and studied, and the tasks of the study of demographics forecasting were formulated.

SECTION 3 DEVELOPMENT OF A FORECASTING MODEL

In this section we develop a forecasting model to successfully forecast future TFR for each country starting from selected year for custom number of steps (in our case – years). We use and compare all models described above, even in cases, when, for obvious reason (absence of seasonality). We then output the result for each selected country to compare it. We use RMSE to derive which model does predict better than others and plot predicted values vs historical data.

3.1 Overview of the data

All data were taken from https://www.humanfertility.org. For each country there are few files each with records for available years. Files are (unfortunately) in txt format with few descriptive lines which needed to be removed. For each country file name starts with country code – for example, for Austria it's AUT, for Iceland – ISL, for Sweden – SWE and so on. For each country we use exactly 8 files, here is the example for Austria, country code can be changed to any.

1. AUTpprVHbo.txt

   Description: this file contains cohort-specific parity progression ratios (PPRs). The PPR indicates the probability of having another child for women who have already had a certain number of children.

   Columns:

   Cohort: the birth cohort of women.

   PPR0_1: the probability of progressing from 0 to 1 child.

   PPR1_2: the probability of progressing from 1 to 2 children.

   PPR2_3: the probability of progressing from 2 to 3 children.

   PPR3_4: the probability of progressing from 3 to 4 children.

2. AUTasfrRR.txt

   Description: this file contains the age-specific fertility rates (ASFR) for different years.

   Columns:

   Year: the year of observation.

   Age: the age of women.

   ASFR: the age-specific fertility rate.

3. AUTasfrTR.txt

   Description: this file provides the age-specific fertility rates (ASFR) with cohort information.

   Columns:

   Year: the year of observation.

   Age: the age of women.

   Cohort: the birth cohort of women.

   ASFR: the age-specific fertility rate.

4. AUTasfrVH.txt

   Description: this file includes cohort-specific age-specific fertility rates (ASFR).

   Columns:

   Cohort: the birth cohort of women.

   Age: the age of women.

   ASFR: the age-specific fertility rate.

5. AUTasfrVV.txt

   Description: this file contains age-specific fertility rates (ASFR) with additional demographic details.

   Columns:

   Year: the year of observation.

   ARDY: the age reached during the year.

   Cohort: the birth cohort of women.

   ASFR: the age-specific fertility rate.

6. AUTasfrRRbo.txt

   Description: this file provides detailed age-specific fertility rates (ASFR) for different orders of births.

   Columns:

   Year: the year of observation.

   Age: the age of women.

   ASFR: the age-specific fertility rate for all births.

   ASFR1: the age-specific fertility rate for first births.

   ASFR2: the age-specific fertility rate for second births.

   ASFR3: the age-specific fertility rate for third births.

   ASFR4: the age-specific fertility rate for fourth births.

   ASFR5p: the age-specific fertility rate for fifth and higher-order births.

7. AUTasfrVHbo.txt

   Description: this file contains cohort-specific age-specific fertility rates (ASFR) for different birth orders.

   Columns:

   Cohort: the birth cohort of women.

   Age: the age of women.

   ASFR: the age-specific fertility rate for all births.

   ASFR1: the age-specific fertility rate for first births.

   ASFR2: the age-specific fertility rate for second births.

   ASFR3: the age-specific fertility rate for third births.

   ASFR4: the age-specific fertility rate for fourth births.

   ASFR5p: the age-specific fertility rate for fifth and higher-order births.

8. AUTasfrVVbo.txt

   Description: this file provides age-specific fertility rates (ASFR) with birth order and demographic details.

   Columns:

Year: the year of observation.

ARDY: the age reached during the year.

Cohort: the birth cohort of women.

ASFR: the age-specific fertility rate for all births.

ASFR1: the age-specific fertility rate for first births.

ASFR2: the age-specific fertility rate for second births.

ASFR3: the age-specific fertility rate for third births.

ASFR4: the age-specific fertility rate for fourth births.

ASFR5p: the age-specific fertility rate for fifth and higher-order births.

3.2 Algorithm description

TFRForecast class was designed to forecast TFR for selected country. It processes all stages from initial data transformation and cleaning to analyzing the data, forecasting it and evaluating accuracy as well as plotting the results using matplotlib library and printing out the comparative table of forecasted results vs historical data.

When initialized, the class sets up several key attributes:

- start_year: the year from which the forecasting process begins;
- data_files_path: the path to the directory containing the necessary data files;
- file_configs: a dictionary which provides configuration details for each used data file - number of rows to skip and the column names;
- country_name: the name of the country being analyzed (needed for output);
- forecast_steps: the number of future years (steps) for which the TFR will be forecasted.

Then we proceed to data processing: using function clean_and_load_data.

This function is responsible for reading data from a passed, cleaning it, and converting it into a needed format (in our case – DataFrame from Pandas library). It takes the following parameters:

file_path: the path to the data file.

skip_rows: the number of initial rows to skip in the file, typically used to bypass headers or metadata.

col_names: a list of column names to assign to the DataFrame.

Then the function reads the file line by line, skips the specified rows (which are description of the data in each file), and then processes the remaining lines into a df. It ensures all data is numeric, converting any non-numeric entries to NaN.

Then using function load_data*:*

- it orchestrates the loading of all necessary data files. It iterates over each entry in the *file_configs* dictionary,
  calling *clean_and_load_data* for each file and storing the resulting DataFrame in *self.cleaned_data*;
- after loading the data, it processes the age-specific fertility rate (ASFR) data;
- converts age data to numeric format;
- aggregates the ASFR data by year and age;
- filters the ASFR data to include only relevant age groups (15-49 years);
- pivots the ASFR data so that years are the index and ages are columns;
- fills any missing values in the DataFrame;
- finally, it calculates the TFR by summing the ASFR values across all age groups for each year.

Then we use function *forecast_tfr*, it includes several models described above:

- ARIMA;
- ARMA;
- SARIMA;
- ARIMAX;
- ETS;
- LSTM .

For each model, the function:

- attempts to fit the model to the historical TFR data;
- forecasts TFR for the specified number of future years (*steps*);
- stores the forecasted values in a dictionary (*results*);
- if a model fails to fit the data, the function catches the exception and prints an error message (which is important step but does not really happen since our data is high-quality and preprocessed properly).

Then we have Static Method create_lstm_data*:*

This helper method prepares the data for the LSTM model by creating lagged input sequences. It generates pairs of input-output sequences for training the LSTM model.

Then we use Static Method calculate_rmse*:*

This method calculates the Root Mean Squared Error between actual and forecasted values. RMSE is a standard metric for measuring the accuracy of forecasts, where lower values indicate better model performance.

As final step we run function *run_analysis,* This is the main function that runs the entire analysis pipeline*:*

- load Data: calls load_data to prepare the historical TFR data;
- split Data: separates the historical TFR data into two parts: the portion before the start_year (used for training the models) and the portion from start_year onwards (used for comparison with forecasts);
- forecasting: calls forecast_tfr to generate forecasts using the historical TFR data;

- merge Data: creates a DataFrame (comparison_df) that combines historical TFR values with the forecasted values from each model;
- plotting: Plots both the historical TFR and the forecasts on a single graph for visual comparison;
- calculate RMSE: Calculates the RMSE for each model by comparing the forecasted values to the actual historical TFR values from the start_year onwards;
- identify Best Model: Determines which model has the lowest RMSE, indicating the best performance;
- outputs:
  a plot showing historical TFR and forecasted values.
  A DataFrame (*comparison_df*) with historical and forecasted TFR values.
  RMSE values for each model.
  The name of the best-performing model based on RMSE.

3.3 Output and evaluation of results

Example of the output for Bulgaria:

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS
65	2012.0	1.48949	1.561238	1.550433	1.515057	1.543130	1.496449
66	2013.0	1.47074	1.572411	1.553204	1.429795	1.596037	1.455270
67	2014.0	1.52378	1.607323	1.572130	1.370164	1.631891	1.420503
68	2015.0	1.52050	1.589144	1.574665	1.315011	1.684580	1.425229
69	2016.0	1.53402	1.589020	1.593072	1.292692	1.688013	1.462643
70	2017.0	1.55042	1.589701	1.595381	1.253515	1.672076	1.448352

71	2018.0	1.54879	1.589534	1.613285	1.227254	1.697802	1.446960
72	2019.0	1.57511	1.600735	1.615379	1.178650	1.723785	1.439471
73	2020.0	1.55013	1.597118	1.632795	1.198343	1.737009	1.480653

	LSTM
65	1.471910
66	1.420256
67	1.381722
68	1.356951
69	1.322277
70	1.294379
71	1.270266
72	1.245684
73	1.224566

RMSE for ARIMA: 0.06390280749208717

RMSE for ARMA: 0.059873290120834775

RMSE for SARIMA: 0.2447230819145974

RMSE for ARIMAX: 0.13226538524130255

RMSE for ETS: 0.0834122477953325

RMSE for LSTM: 0.21269433594542442

The best model based on RMSE is: ARMA

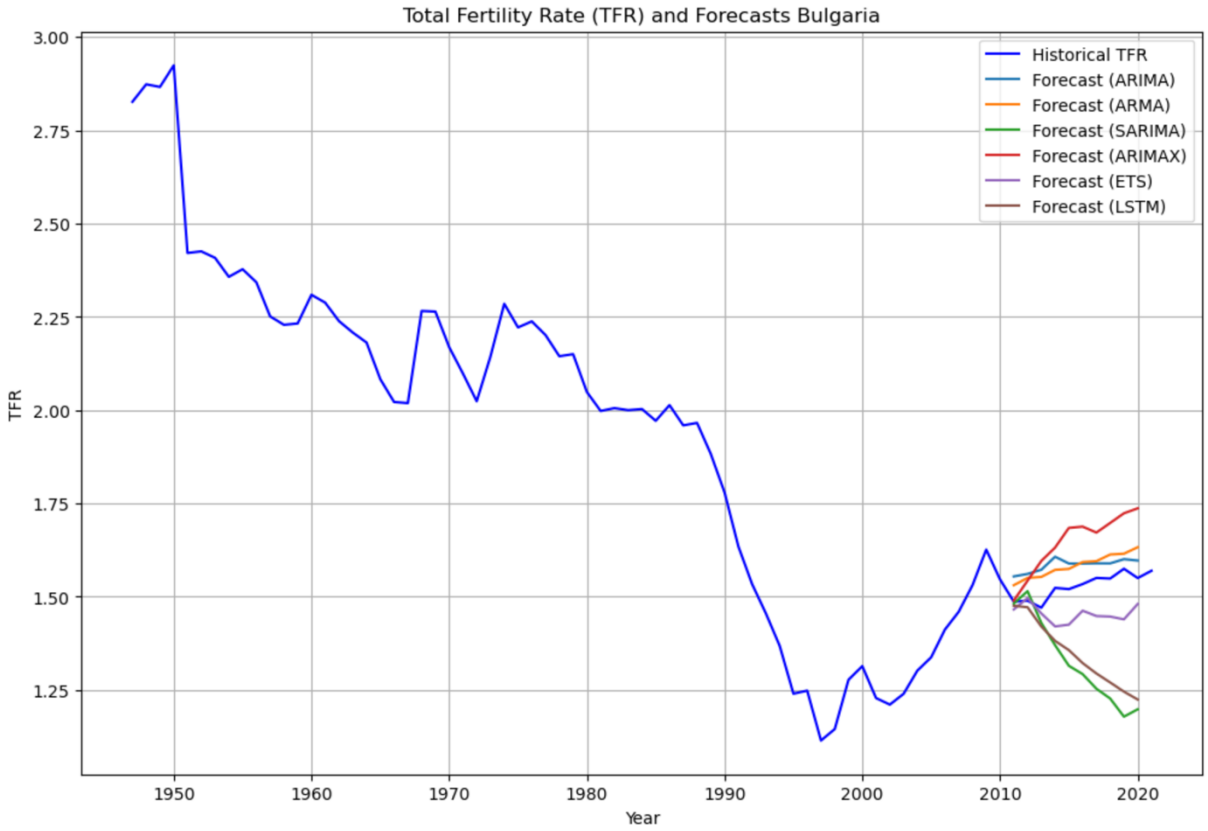


Figure 3.4 – Example of TFR forecast plot for Bulgaria.

In this example we can see the forecasted values of TFR for Bulgaria, starting from year 2011. We can see that some models do forecast worse values (SARIMA) and some predict much better values (ARIMAX). The general observation is:

- ARIMA Model: Generally performed well in countries with more stable and predictable TFR pattern which got affected by some outer influence (crisis, some stimulation, recession, etc);
- ARMA Model: Often provided accurate forecasts for countries where simpler models could capture the underlying patterns without the need for differencing;
- SARIMA and ETS Models: These models were less consistent in their performance, sometimes providing accurate forecasts but often being outperformed by ARIMA and ARMA models;

- LSTM Model: These neural network models were generally less accurate in this context, possibly due to the limited amount of data available for training or the complexity of the model relative to the data.

Conclusions to the section 3

In the development of the forecasting model for the Total Fertility Rate across different countries, various statistical models and forecasting techniques were employed. The application allows users to utilize multiple forecasting approaches and visualize the results through clear graphs, providing insights into different prediction methods.

The core of the application involves running several forecasting models such as ARIMA, ARMA, SARIMA, ARIMAX, ETS, and LSTM, and selecting the best one based on their performance. This selection process is crucial as it determines the most accurate model for each country's TFR data. Given the extensive nature of this task, the application iterates through all the forecasting methods to identify the optimal model, ensuring high-quality results.

Applying this software to real-world scenarios offers significant benefits. Policymakers and demographers can use the forecasts generated by the software to make informed decisions regarding population growth, resource allocation, and social services planning. For instance, understanding future fertility trends helps in planning for educational facilities, healthcare services, and pension systems.

As mentioned above, ARIMA and ARMA performed much better than other models, and it is clear that ARIMA worked better for non-stationary situations and ARMA performed better for stationary cases (see Appendix). Poor performance of SARIMA is essential, since our data is not seasonal by any means. ETS did not

perform well, and we see the same situation for LSTM – it is possible that the reason for that is lack of the data to train and complexity of the process.

SECTION 4 FUNCTIONAL AND COST ANALYSIS OF A SOFTWARE PRODUCT

Functional cost analysis (FCA) is a technology that allows you to estimate the real value of a product or service regardless of the organisational structure of the company. FCA is carried out to identify cost savings through more efficient production options and a better balance between the consumer value of a product and the cost of its manufacture. The analysis is based on economic, technical and design information.

4.1 Formulation of the design problem

The paper applies the FBA method to conduct a feasibility analysis of the development of a system for forecasting the sustainability of financial indicators. Since the decisions on the design and implementation of the components of the system under development affect the entire system, each individual subsystem must satisfy it. Therefore, the actual analysis is an analysis of the functions of a software product designed to collect, process and analyse company data.

The technical requirements for the software product are as follows:

- operation on personal computers with a standard set of components;
- convenience and clarity for the user;
- speed of data processing and access to information in real time;
- easy scaling and maintenance;
- minimal costs for the implementation of the software product.

4.2 Justification of the functions of the software product

Main function F_0 - Development of a possible software product that allows analysing various characteristics that directly affect the sustainability of the enterprise. Based on this function, we can distinguish the following:

F_1 - the choice of the programme itself.

F_2 - qualitative data analysis.

F_3 - graphical indicators.

Each of these functions has several implementation options:

Function F_1:

a) Python.

b) R.

Function F_2:

a) Usage of built-in functions.

b) Creation of custom value calculations.

Function F_3:

a) Template charts usage.

b) Custom charts creation.

Options for implementing the main functions are shown in the morphological map of the system (Fig. 4.1).

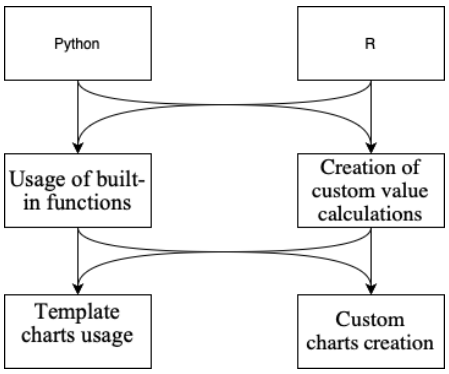


Figure 4.1 - Morphological map

The morphological map shows the set of all possible variants of the main functions. The positive-negative matrix is shown in Table 4.1.

Table 4.1 - Positive-negative matrix

Features.	*Implemen-tation options*	*Advantages*	*Disadvantages*
F_1	A	Publicly available programme, accessibility of many libraries	The need for full implementation of the algorithm
	B	Easy to implement program for various calculations	Basic skills and abilities required
F_2	A	Accessibility and ease of writing	May not always match the required tasks
	B	Comprehensive description of all necessary characteristics	High implementation costs
F_3	A	Commonly accepted implementation	May not meet expected values
	B	Better ability to communicate findings from own research	Time-consuming.

Based on the analysis of the positive-negative matrix, we conclude that when developing a software product, some options for implementing functions should be rejected because they do not meet the tasks set for the software product. These options are marked in the morphological map.

Function F_1:

We prefer public accessibility, to simplify the work of writing code, option B should be rejected.

Function F_2*:*

The program allows you to select both options, both options can be chosen. Function F_3*:*

Implementation of the first option is receptive to the programme, this is option A.

Thus, we will consider the following option for the implementation of the PP:

$$F_1\text{a} - F_2\text{a} - F_3\text{a},$$
$$F_1\text{a} - F_2 b - F_3\text{a}.$$

To evaluate the quality of the functions under consideration, we chose the system of parameters described below.

4.3 Justification of the software product parameter system

Based on the data discussed above, the main selection parameters that will be used to calculate the technical level coefficient are determined.

To describe the software product, we will use the following parameters:

- X_1 *is the* speed of the programming language;
- X_2 *is the* amount of memory for computing and data storage;
- X_3 - data training time;

- X_4 *is the* potential amount of program code.

The worst, average and best values of parameters are selected based on customer requirements and conditions that characterise the operation of the software product, as shown in Table 4.2.

Table 4.2 - Main parameters of the software product

Title. Parameter.	*Symbols and notation*	*Units of measurement*	*The value of the parameter*		
			worse	*medium*	*best*
The speed of the programming language	X_1	operations/ms	60	80	110
Memory capacity	X_2	Mb	60	50	30
Data pre-processing time	X_3	ms	80	70	60
Potential amount of software code	X_4	number of lines of code	35	25	20

4.4 Analysing the expert evaluation of parameters

After a detailed discussion and analysis, each expert assesses the degree of importance of each parameter for a specific goal - the development of a software product that gives the most accurate results when finding the parameters of adaptive forecasting models and calculating forecast values.

The importance of each parameter is determined by a pairwise comparison. The assessment is carried out by an expert committee of 7 people. Determining the significance coefficients involves:

- determining the level of significance of a parameter by assigning different ranks;

− checking the suitability of expert opinions for further use;

− determination of the pairwise priority of parameters;

− processing the results and determining the significance coefficient.

The results of the expert ranking are presented in Table 4.3.

Table 4.3 - Results of the parameters ranking

Parameter designation	Parameter name	Units of measurement	The rank of the parameter according to the expert's assessment						Sum of ranks R_1		Deviation Δ_1	$\Delta_1{}^2$
			1	2	3	4	5	6	7			
X_1	The speed of the programming language	operations/ ms	1	2	2	1	1	1	2	10	-7,5	56,25
X_2	Memory capacity	Mb	3	4	3	3	4	3	4	24	6,5	42,25
X_3	Data pre-processing time	ms	2	1	1	2	2	2	1	11	-6,5	42,25
X_4	Potential amount of software code	Number of lines of code	4	3	4	4	3	4	3	25	7,5	56,25
			10	10	10	10	10	10	10	70	0	197

To check the degree of reliability of expert opinions, we will define the following parameters:

The sum of the ranks of each of the parameters and the total sum of the ranks:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70,$$

where *N is the* number of experts,

n - number of parameters.

The average sum of ranks:

$$T = \frac{1}{n} R_{ij} = 17{,}5.$$

Deviation of the sum of ranks of each parameter from the average sum of ranks:

$$\Delta_i = R_i - T.$$

The sum of deviations for all parameters must be 0.

Total sum of squared deviations:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 197.$$

Let's calculate the coefficient of consistency:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 197}{7^2(4^3 - 4)} = 0{,}754 > W_k = 0{,}67.$$

The ranking can be considered reliable, as the coefficient of consistency found exceeds the standard coefficient of 0.67.

Using the results of the ranking, we will conduct a pairwise comparison of all parameters and record the results in Table 4.4.

Table 4.4 - Pairwise comparison of parameters

Parameters.	Experts							Final assessment	Numerical value
	1	2	3	4	5	6	7		
X1 and X2	<	<	<	<	<	<	<	<	0,5
X1 and X3	<	>	>	<	<	<	>	<	0,5
X1 and X4	<	<	<	<	<	<	<	<	0,5
X2 and X3	>	>	>	>	>	>	>	>	1,5
X2 and X4	<	>	<	<	>	<	<	<	0,5
X3 and X4	<	<	<	<	<	<	<	<	0,5

The numerical value that determines the degree of superiority of the *i-th* parameter over the *j-th* parameter, a_{ij} , is determined by the formula:

$$a_{ij} = \{1.5\ for\ X_i > X_j\ 1.0\ for\ X_i = X_j\ 0.5\ for\ X_i < X_j\ ,$$

From the obtained numerical estimates of preference, we will compile the matrix $A = \| a_{ij} \|$.

For each parameter, we calculate the weight of K_{wi} using the following formulas:

$$K_{w\text{i}} = \frac{b_i}{\sum_{i=1}^{n} b_i}\ ,$$

$$b_i = \sum_{i=1}^{N} a_{ij}\ .$$

The relative scores are calculated several times until the next values differ slightly from the previous ones (less than 2%). In the second and subsequent steps, the relative scores are calculated using the following formulas:

$$K_{\text{Bi}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'},$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j.$$

As can be seen from Table 4.5, the difference in the values of the weighting coefficients does not exceed 2%, so no more iterations are required.

Table 4.5 - Calculation of parameter weights

Parameters. $x_{\$}$	Parameters. $x_{\%}$				The first iteration.		The second iteration.		The third iter	
	X1	X2	X3	X4	$b_{\$}$	K_{Bi}	$b_{\$!}$	$K_{\text{Bi!}}$	$b_{\$''}$	$K_{\text{Bi''}}$
X1	1	0,5	0,5	0,5	2,66	0,14	10,08	0,16	34,72	0,16
X2	1,5	1	1,5	0,5	4,94	0,26	16,38	0,26	56,42	0,26
X3	1,5	0,5	1	0,5	3,99	0,21	12,6	0,2	43,4	0,2
X4	1,5	1,5	1,5	1	7,41	0,39	23,94	0,38	82,46	0,38
Total:					19	1	63	1	217	1

4.5 Analysis of the quality level of options for implementing functions

We determine the quality level of each variant of the main functions separately.

The absolute values of the parameters *X2* (memory capacity), *X3* (data preprocessing time) and *X4* (potential program code size) meet the technical requirements of the functioning conditions of this software package.

The absolute value of the parameter *X1* (the speed of the programming language) was not the worst.

The technical level coefficient for each implementation option is calculated as follows (Table 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{wi,j} B_{i,j} ,$$

where *N is the* number of parameters;

K_{wi} - is the weighting factor of the *i-th* parameter;

B_i - the score of the *i-th* parameter in points.

Table 4.6 - Calculation of quality level indicators of options for implementing the main functions of the PP

Main functions	Option for implementing the function	Parameters.	Absolute value of the parameter	Scoring of the parameter	Parameter weighting factor	Quality level coefficient
F1	A	X1	98	23	0,16	3,68
F3	A	X2	85	27	0,26	7,02
	Б	X3	25	17	0,2	3,4
F4	A	X4	23	21	0,38	7,98

Based on data from Table above using the formula:

$$K_K = K_{QC}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{QC}[F_{zk}],$$

determine the quality level of each option:

$K_{K1} = 3,68 + 7,02 + 7,98 = 18,68,$

$K_{K2} = 3,68 + 3,4 + 7,98 = 15,06 .$

As can be seen from the calculations, the preferred option is option 2, for which the technical level coefficient is the highest.

4.6 Economic analysis of the options for developing the PP

To determine the cost of developing a software application, let's first calculate the labour intensity.

All options include two separate tasks:

1. Development of a software product project;
2. Software development;

Task 1 belongs to Group A in terms of novelty, Task 2 - to Group B. In terms of complexity, the algorithms used in Task 1 belong to Group 1; and in Task 2 - to Group 3.

Task 1 uses background information, and task 2 uses information in the form of data.

We will calculate the time standards for development and programming for each of the tasks.

The total labour intensity is calculated as:

$$T_D = T_P \cdot K_C \cdot K_{CC} \cdot K_M \cdot K_{ST} \cdot K_{ST.M},$$

where T_D is the labour intensity of the PP development;

K_C - correction coefficient;

$K_{CC}$is the coefficient for the complexity of the input information;

$K_M$is the coefficient of the programming language level;

$K_{ST}$is the ratio of the use of standard modules and applications;

$K_{ST.M}$- coefficient of standard mathematical support.

For the first task, based on the time norms for computational tasks of novelty degree A and complexity group 1 of the algorithm, the labour intensity is equal to: T_D = 37 person-days. Correction factor that takes into account the type of regulatory and reference information for the first task: $К_C$ = 1.8. The correction factor that takes into account the complexity of controlling incoming and outgoing information for all seven tasks is 1: $К_{CC}$ = 1. Since standard modules are used in the development of the first task, we will take this into account with the coefficient $К_{ST}$ = 0.9. Then the total complexity of programming the first task is equal:

$$T_1 = 37 \cdot 1.8 \cdot 0.9 = 59{,}94 \text{ man-days.}$$

Let's make similar calculations for further tasks.

For the second task (using the algorithm of the third complexity group, degree of novelty B), i.e. T_D = 29 man-days, $К_C$ = 0.9, $К_{CC}$ = 1, $К_{ST}$ = 0.8:

$$T_2 = 29 \cdot 0.9 \cdot 0.8 = 20.88 \text{ man-days.}$$

We add up the labour intensity of the relevant tasks for each of the selected programme implementation options to get their labour intensity:

$$T_I = (59{,}94 + 20.88 + 4.8 + 20.88) \cdot 8 = 852 \text{ man-hours.}$$
$$T_{II} = (59{,}94 + 20.88 + 6.91 + 20.88) \cdot 8 = 868{,}88 \text{ man-hours.}$$

Option II has the highest labour intensity.

Two programmers with a salary of UAH 27,000 and one data analyst with a salary of UAH 29,000 are involved in the development. Determine the average salary per hour using the formula:

$$AS = \frac{\text{M}}{T_m \cdot t} UAH,$$

where M is the monthly salary of employees;

T_m- number of working days per week;

t - the number of working hours per day.

$$AS = \frac{27000 + 27000 + 29000}{3 \cdot 21 \cdot 8} = 164{,}68 \;\; UAH.$$

Then, let's calculate the average paid salary using the formula:

$$APS = W_h \cdot L_i \cdot W_a,$$

where W_h is the hourly wage of a programmer;

L_i - the labour intensity of the relevant task;

W_a is a standard that takes into account additional wages.

The salary of developers by option is:

I. W_d = 164.68 · 852 · 1.2 = 168368,83 UAH.

II. W_d = 164.68 · 868.88 · 1.2 = 171704,59 UAH.

The unified social tax contribution is 22%:

I. W_{def} = 168368.33 ∗ 0.22 = 37041.14 UAH.

II. W_{def} = 171704.59 ∗ 0.22 = 37775.01 UAH.

Now let's determine the cost of paying for one machine hour. W_m

Since one computer serves one programmer with a salary of 27000 UAH, with an employment rate of 0.2, we get the following for one machine:

$$W_d = 12 * M * K_s * 0.2 = 64800 \; UAH.$$

Including additional salary:

$$C = W_d(1 + 0.2) = 77760\,\text{UAH}.$$

Social security contributions:

$$C_{\text{SSC}} = C \times 0.22 = 77760 \times 0.22 = 17107.2\,\text{UAH}.$$

Depreciation charges are calculated at a depreciation rate of 25% and a computer cost of UAH 10,000.

$$C_{\text{Dep}} = K_{\text{TM}} \times K_{\text{A}} \times C_{\text{Contract}} = 1.4 \times 0.25 \times 10000 = 3500\,\text{UAH},$$

where K_{TM} is a coefficient that takes into account the cost of transport and installation of the device at the user's premises;

K_{A} - annual depreciation rate;

C_{Contract} - the contract price of the device.

We calculate repair and maintenance costs as follows:

$$C_{\text{Repair}} = K_{\text{TM}} \times C_{\text{Contract}} \times K_{\text{Repair}} = 1.4 \times 10000 \times 0.08 = 1120\,\text{UAH},$$

where K_{Repair} is the percentage of expenditures on current repairs.

We calculate the effective hourly time fund of a PC for a year using the formula:

$$T_{\text{Eff}} = \left(D_{\text{Cal}} - D_{\text{Off}} - D_{\text{Holidays}} - D_{\text{Repair}}\right) \times t_{\text{day}} \times K_{\text{Util}} = (365 - 104 - 12 - 16) \times 8 \times 0.38 = 708.32\,\text{hours},$$

Where D_{Cal} is the calendar number of days in a year;

D_{Off}, D_{Holidays} - the number of days off and holidays, respectively;

D_{Repair} - number of days of scheduled equipment repairs;

t_{day} is the number of working hours per day;

K_{Util} is the device utilisation rate over time during a shift.

We calculate electricity costs using the formula:

$$C_{\text{Electricity}} = T_{\text{Eff}} \times N_{\text{Power}} \times K_s \times C_{\text{Tariff}} = 708.32 \times 0.2 \times 0.3 \times 5.23 = 222{,}27 \text{ UAH.}$$

where N_{Power} is the average power consumption of the device;

K_s is the device occupancy rate;

C_{Tariff} - tariff per 1 kWh of electricity.

Overheads are calculated using the formula:

$$C_{\text{Overhead}} = C_{\text{Contract}} \times 0.67 = 10000 \times 0.67 = 6700 \text{ UAH.}$$

Then, the annual operating costs will be lower:

$$C_{\text{Operating}} = C + C_{\text{SSC}} + C_{\text{Dep}} + C_{\text{Repair}} + C_{\text{Electricity}} + C_{\text{Overhead}} = 77760 + 17107.2 + 3500 + 1120 + 222{,}27 + 6700 = 106409{,}47 \text{ UAH.}$$

The cost of one machine hour of a computer will be equal:

$$C_{\text{MH}} = \frac{C_{\text{Operating}}}{T_{\text{Eff}}} = \frac{106409{,}47}{708.32} = 150.23 \text{ UAH/hour.}$$

Since in this case all the work related to the development of a software product is carried out on a computer, the cost of paying for machine time, depending on the chosen implementation option, is:

$$C_{\text{MH}} = \frac{C_{\text{Operating}}}{T_{\text{Eff}}} = \frac{106409{,}47}{708.32} = 150.23\,\text{UAH/hour.}$$

I. $C_{\text{M}} = C_{\text{MH}} \times T = 150.23 \times 852 = 127994{,}22\,\text{UAH.}$

II. $C_{\text{M}} = 150.23 \times 868.88 = 130531{,}84\,\text{UAH.}$

Overheads account for 67% of salaries:

I. $C_{\text{Overhead}} = C_{\text{ЗП}} \times 0.67 = 168368.83 \times 0.67 = 112807.12\,\text{UAH,}$

II. $C_{\text{Overhead}} = 171704.59 \times 0.67 = 115042.08\,\text{UAH.}$

Thus, the cost of developing the PP by option is as follows:

$$C_{\text{Dev}} = C_{\text{ЗП}} + C_{\text{SSC}} + C_{\text{M}} + C_{\text{Overhead}}.$$

I. $C_{\text{Dev}} = 168368{,}83 + 37041{,}14 + 127994{,}22 + 112807{,}12 = 446211{,}31\,\text{UAH,}$

II. $C_{\text{Dev}} = 171704{,}59 + 37775{,}01 + 130531{,}84 + 115042{,}08 = 454983{,}52\,\text{UAH.}$

4.7 Selection of the best option for the PP at the technical and economic level

Let's calculate the technical and economic level coefficient using the formula:

$$K_{\text{TechEco}} = \frac{K_{\text{Coeff}}}{C_{\text{Dev}}},$$

$$K_{\text{TechEco1}} = \frac{18{,}68}{446211{,}31} = 4.19 \times 10^{-5},$$

$$K_{\text{TechEco2}} = \frac{15{,}06}{454983{,}52} = 3.31 \times 10^{-5}.$$

As we can see, the most effective is the first option of the programme implementation with the technical and economic level coefficient

$$K_{TEP1} = 4.208 \cdot 10^{-}$$

After performing a functional and cost analysis of the software complex under development, it can be concluded that of the alternatives remaining after the first selection of two variants of the software complex, the first variant of the software product implementation is optimal. It has the best indicator of the technical and economic level of quality

$$K_{\text{TechEco1}} = 4.19 \times 10^{-5}.$$

This option of the software product has the following parameters:

- the choice of a software product is Python;
- implement an important production with the help of built-in functions;
- use the standard interface to build values.

This option of the software package provides the user with a user-friendly interface, fast implementation of the programme and accessible functionality for work.

Conclusions to the section 4

In this part, a full functional and cost analysis of the software product was carried out. The main functions of the software product were also assessed.

This study also shows various implementation options to ensure the most correct and optimal selection strategy, which has an impact on economic factors and compatibility with the future software product.

As a result of the functional and cost analysis of the software system under development, the main functions of the software product were identified and evaluated, and the parameters that characterise it were found.

Based on the analysis, a software product implementation option was selected.

CONCLUSIONS

This thesis explored the use of time series to analyze and forecast demographics of developed and developing countries. It also developed a self-sufficient and user-friendly algorithm for forecasting TFR using input data from a selected country for a custom number of years into the future. Such models as ARMA, ARIMA, SARIMA, ARIMAX, LSTM and ETS were considered. With the help of RMSE, the most appropriate model for different cases and situations was determined. Result is not surprising – ARMA and ARIMA performed better than other models – SARIMA can't perform well since our data is not seasonal, ARIMAX can't as well, since we don't have exogenous variable, for LSTM we don't have enough meaningful historical data and for ETS TFR data is not stationary enough.

Our model can be used by non-profit organizations, research institutes and international influential organizations in an attempt to predict the demographic future of a country or region. However, it is important to recognize (and this is evident in the results) that fragile indicators such as TFR are incredibly difficult to predict, as they depend on many factors, such as housing costs, childbearing conditions, level of community development, government assistance, culture and social structure, and so on. It is also important to note that when there are crises or, on the contrary, sudden economic upswings, as well as large government involvement (financially) in childbearing, TFR can rise or fall significantly in a short period of time - examples of this are the USA, China, Russia, Hungary, Ukraine and virtually all developed and developing countries where there are serious economic fluctuations - only the most stable and/or smallest of them - be it Austria, Iceland, Ireland or the Czech Republic - can usually avoid problems.

The data and code of this paper can be used for further research and attempts to establish more specific effects of external factors on fertility against other circumstances - examples are Georgia, Russia, Poland and France (and some other developed countries).

The results have been presented in graphs and tables (see Appendix B and C) and even people far from science can easily understand what we are talking about and in which direction we should expect changes.

REFERENCES

1. G. T. Wilson, “Time Series Analysis: Forecasting and Control, 5th Edition, by George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel and Greta M. Ljung, 2015. Published by John Wiley and Sons Inc., Hoboken, New Jersey, pp. 712. ISBN: 978-1-118-67502-1,” Journal of Time Series Analysis, vol. 37, no. 5, pp. 709–711, Mar. 2016, doi: https://doi.org/10.1111/jtsa.12194.
2. Marcel Dettling “Applied Time Series Analysis” Accessed: May 22, 2024 https://ethz.ch/content/dam/ethz/special-interest/math/statistics/sfs/Education/Advanced%20Studies%20in%20Applied%20Statistics/course-material-1921/Zeitreihen/ATSA_Script_v200504.pdf
3. Ron Lesthaeghe “The second demographic transition: A concise overview of its development” Accessed: May 22, 2024. https://www.pnas.org/doi/10.1073/pnas.1420441111#:~:text=The%20first%20or%20“classic”%20demographic,present%20in%20most%20developing%20countries.
4. BBC News "China records population decline for second straight year" Accessed: May 22, 2024. https://www.bbc.com/news/world-asia-china-68002803
5. PIIE "China's Population Decline: Getting Close to Irreversible" Accessed: May 22, 2024. https://www.piie.com/research/piie-charts/2024/chinas-population-decline-getting-close-irreversible
6. United Nations "World Population Prospects" Accessed: May 22, 2024. http://data.un.org/Data.aspx?d=PopDiv&f=variableID%3A54
7. Kirk, D. (1996). Demographic Transition Theory. Population Studies, 50(3), 361–387. https://doi.org/10.1080/0032472031000149536

8. PNAS "The second demographic transition: A concise overview of its development" Accessed: May 22, 2024. https://www.pnas.org/doi/abs/10.1073/pnas.1420441111
9. Nancy L. Fleischer, Robert E. McKeown "The Second Epidemiologic Transition from an Epidemiologist's Perspective" Accessed: May 22, 2024. https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118504338.ch19
10. R. Lesthaeghe, D. van de Kaa, “Twee demografische transities? [Two demographic transitions?]” Bevolking–Groei en Krimp, Mens en Maatschappij, eds R Lesthaeghe, D van de Kaa (Van Loghum Slaterus, Deventer, The Netherlands), pp. 9–24, Dutch. (1986).
11. R. Lesthaeghe, J. Surkyn, “Cultural dynamics and economic theories of fertility change”. Popul Dev Rev 14, 1–45 (1988).
12. R. Lesthaeghe, “The second demographic transition in Western countries. Gender and Family Change in Industrialized Countries”, eds K Oppenheim Mason, A-M Jensen (Clarendon, Oxford), pp. 17–62 (1995).
13. McKeown RE. “The Epidemiologic Transition: Changing Patterns of Mortality and Population Dynamics”. Am J Lifestyle Med. 2009 Jul 1;3(1 Suppl):19S-26S. doi: 10.1177/1559827609335350.
14. Pete Chapman (1999) ”The CRISP-DM User Guide” Accessed: May 22, 2024. https://s2.smu.edu/~mhd/8331f03/crisp.pdf
15. Patricia Sowa “Cross Industry Standard Process for Data Mining” Accessed: May 22, 2024. https://hpi.de/fileadmin/user_upload/fachgebiete/rabl/Lectures/PDE_Poster/PDE_Patricia_Sowa.pdf
16. Python Software Foundation, "What is Python?" Accessed: May 22, 2024. https://www.python.org/doc/essays/blurb/
17. R Foundation, "About R" Accessed: May 22, 2024. https://www.r-project.org/about.html

18. Timescale, "Tools for Working with Time Series Analysis in Python" Accessed: May 22, 2024. https://www.timescale.com/blog/tools-for-working-with-time-series-analysis-in-python/
19. Seaborn, "Seaborn: statistical data visualization" Accessed: May 22, 2024. https://seaborn.pydata.org
20. Domino Data Lab, "Jupyter Notebook - Data Science Dictionary" Accessed: May 22, 2024. https://domino.ai/data-science-dictionary/jupyter-notebook
21. GeeksforGeeks, "What is PyCharm?" Accessed: May 22, 2024. https://www.geeksforgeeks.org/what-is-pycharm/
22. Richard Weber, "Time Series Analysis" Accessed: May 22, 2024. http://www.statslab.cam.ac.uk/~rrw1/timeseries/t.pdf
23. Nobre, F.F., Monteiro, A.B.S., Telles, P.R. and Williamson, G.D. (2001), "Dynamic linear model and SARIMA: a comparison of their forecasting performance in epidemiology". Statist. Med., 20: 3051-3069. https://doi.org/10.1002/sim.963, Accessed: June 4, 2024
24. Box EP, Jenkins GM. "Time Series Analysis Forecasting and Control". 1976.
25. Ashutosh Kumar Dubey, Abhishek Kumar, Vicente García-Díaz, Arpit Kumar Sharma, Kishan Kanhaiya "Study and analysis of SARIMA and LSTM in forecasting time series data" ,Sustainable Energy Technologies and Assessments,Volume 47,2021,101474,ISSN 2213-1388,https://doi.org/10.1016/j.seta.2021.101474
26. SolarWinds. "Holt-Winters Forecasting and Exponential Smoothing Simplified." *Orange Matter*, 15 Dec. 2019, https://orangematter.solarwinds.com/2019/12/15/holt-winters-forecasting-simplified/. Accessed June 4, 2024.
27. "Statsmodels: Statistical Modeling with Python." Statsmodels, https://www.statsmodels.org/stable/index.html. Accessed 4 June 2024.

28. NVIDIA. "TensorFlow – What Is It and Why Does It Matter?" *NVIDIA Glossary,* https://www.nvidia.com/en-us/glossary/tensorflow/. Accessed 4 June 2024.
29. CIA, "Total Fertility Rate - Country Comparison," Accessed: June 7, 2024. https://www.cia.gov/the-world-factbook/field/total-fertility-rate/country-comparison/
30. World Health Organization, "Indicator Metadata Registry," Accessed: June 7, 2024. https://www.who.int/data/gho/indicator-metadata-registry
31. Pew Research Center, "Economic Recession and Fertility in the Developed World," Accessed: June 7, 2024. https://www.pewresearch.org/wp-content/uploads/sites/3/2010/10/economic-recession-and-fertillity-2009.pdf
32. Bloomberg Opinion, "South Korea's Fertility Crisis: Could Public Subsidies Solve It?" Accessed: June 7, 2024. https://www.bloomberg.com/opinion/articles/2024-05-15/south-korea-fertility-crisis-could-public-subsidies-solve-it?utm_content=view&cmpid%3D=socialflow-twitter-view&utm_medium=social&utm_source=twitter&utm_campaign=socialflow-organic&sref=htOHjx5Y
33. The Wall Street Journal, "Global Decline in Birthrates and Its Causes," Accessed: June 7, 2024. https://www.wsj.com/world/birthrates-global-decline-cause-ddaf8be2?st=2mhv5xs0aubm2sp&reflink=desktopwebshare_permalink

APPENDIX A – PROGRAM CODE

```
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.holtwinters import ExponentialSmoothing
from statsmodels.tsa.api import VAR
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
import numpy as np
import os
import warnings
warnings.filterwarnings('ignore')
class TFRForecast:
    def __init__(self, start_year, data_files_path, file_configs, country_name, forecast_steps):
        self.start_year = start_year
        self.data_files_path = data_files_path
        self.file_configs = file_configs
        self.country_name = country_name
        self.cleaned_data = {}
        self.tfr = None
        self.forecast_results = {}
        self.comparison_df = None
        self.rmses = {}
        self.forecast_steps = forecast_steps
```

```
        def clean_and_load_data(self, file_path, skip_rows, col_names):
            data = []
            with open(file_path, 'r', encoding='utf-8') as file:
                lines = file.readlines()[skip_rows:]
                for line in lines:
                    if line.strip():
                        data.append(line.split())
            df = pd.DataFrame(data)
            df.columns = col_names
            df = df.apply(pd.to_numeric, errors='coerce')
            return df

        def load_data(self):
            for file_name, config in self.file_configs.items():
                file_path = os.path.join(self.data_files_path, file_name)
                self.cleaned_data[file_name] = self.clean_and_load_data(file_path,
skip_rows=config['skip_rows'], col_names=config['col_names'])

            asfr_rr = self.cleaned_data[list(self.file_configs.keys())[1]]
            asfr_rr['Age'] = asfr_rr['Age'].apply(self.clean_age)
            asfr_rr['Age'] = pd.to_numeric(asfr_rr['Age'], errors='coerce')
            asfr_rr_agg = asfr_rr.groupby(['Year', 'Age']).agg({'ASFR':
'sum'}).reset_index()
            asfr_rr_filtered = asfr_rr_agg[(asfr_rr_agg['Age'] >= 15) &
(asfr_rr_agg['Age'] <= 49)]
            asfr_rr_pivot = asfr_rr_filtered.pivot(index='Year', columns='Age',
values='ASFR')
            asfr_rr_pivot = asfr_rr_pivot.fillna(method='ffill').fillna(method='bfill')
            self.tfr = asfr_rr_pivot.sum(axis=1)
```

```
@staticmethod
def clean_age(age):
    if isinstance(age, str):
        return age.strip('-')
    return age

@staticmethod
def create_lstm_data(data, n_lags=1):
    X, y = [], []
    for i in range(len(data) - n_lags):
        X.append(data[i:(i + n_lags)])
        y.append(data[i + n_lags])
    return np.array(X), np.array(y)

def forecast_tfr(self, tfr, steps=10):
    results = {}

    try:
        model_arima = ARIMA(tfr, order=(5, 1, 0))
        model_fit_arima = model_arima.fit()
        forecast_arima = model_fit_arima.forecast(steps=steps)
        results['ARIMA'] = forecast_arima
    except Exception as e:
        print(f"Failed to fit ARIMA model for TFR: {e}")

    try:
        model_arma = ARIMA(tfr, order=(2, 0, 2))
        model_fit_arma = model_arma.fit()
        forecast_arma = model_fit_arma.forecast(steps=steps)
```

```
            results['ARMA'] = forecast_arma
        except Exception as e:
            print(f"Failed to fit ARMA model for TFR: {e}")

        try:
            model_sarima = SARIMAX(tfr, order=(1, 1, 1), seasonal_order=(1, 1,
1, 12))
            model_fit_sarima = model_sarima.fit(disp=False)
            forecast_sarima = model_fit_sarima.forecast(steps=steps)
            results['SARIMA'] = forecast_sarima
        except Exception as e:
            print(f"Failed to fit SARIMA model for TFR: {e}")

        try:
            exog = tfr.shift(1).dropna()
            tfr_aligned = tfr.loc[exog.index]
            exog = exog.values.reshape(-1, 1)
            model_arimax = SARIMAX(tfr_aligned, exog=exog, order=(5, 1, 0))
            model_fit_arimax = model_arimax.fit(disp=False)
            forecast_exog = np.tile(exog[-1], (steps, 1))
            forecast_arimax = model_fit_arimax.forecast(steps=steps,
exog=forecast_exog)
            results['ARIMAX'] = forecast_arimax
        except Exception as e:
            print(f"Failed to fit ARIMAX model for TFR: {e}")

        try:
            model_ets = ExponentialSmoothing(tfr, seasonal='add',
seasonal_periods=12)
            model_fit_ets = model_ets.fit()
```

```
            forecast_ets = model_fit_ets.forecast(steps=steps)
            results['ETS'] = forecast_ets
        except Exception as e:
            print(f"Failed to fit ETS model for TFR: {e}")

        try:
            n_lags = 3
            X, y = self.create_lstm_data(tfr.values, n_lags=n_lags)
            X = X.reshape((X.shape[0], X.shape[1], 1))
            lstm_model = Sequential()
            lstm_model.add(LSTM(50, activation='relu', input_shape=(n_lags, 1)))
            lstm_model.add(Dense(1))
            lstm_model.compile(optimizer='adam', loss='mse')
            lstm_model.fit(X, y, epochs=200, verbose=0)
            lstm_input = tfr.values[-n_lags:].reshape((1, n_lags, 1))
            lstm_forecast = []
            for _ in range(steps):
                lstm_pred = lstm_model.predict(lstm_input, verbose=0)
                lstm_forecast.append(lstm_pred[0, 0])
                lstm_input = np.append(lstm_input[:, 1:, :], lstm_pred).reshape((1, n_lags, 1))
            results['LSTM'] = pd.Series(lstm_forecast, index=range(int(tfr.index[-1]) + 1, int(tfr.index[-1]) + 1 + steps))
        except Exception as e:
            print(f"Failed to fit LSTM model for TFR: {e}")

        return results

    @staticmethod
```

```
        def calculate_rmse(actual, predicted):
            return np.sqrt(((predicted - actual) ** 2).mean())

        def run_analysis(self):
            self.load_data()

            historical_tfr = self.tfr[self.tfr.index < self.start_year]
            self.forecast_results = self.forecast_tfr(historical_tfr,
steps=self.forecast_steps)

            forecast_index = range(self.start_year, self.start_year +
self.forecast_steps)
            forecast_df = pd.DataFrame({'Year': forecast_index})

            historical_tfr_df = pd.DataFrame(self.tfr).reset_index()
            historical_tfr_df.columns = ['Year', 'Historical TFR']

            for model, forecast in self.forecast_results.items():
                forecast_df[model] = forecast.values

            self.comparison_df = pd.merge(historical_tfr_df, forecast_df, on='Year',
how='outer')

            plt.figure(figsize=(12, 8))
            plt.plot(historical_tfr_df['Year'], historical_tfr_df['Historical TFR'],
label='Historical TFR', color='blue')
            for model, forecast in self.forecast_results.items():
                plt.plot(forecast_index, forecast, label=f'Forecast ({model})')

            plt.xlabel('Year')
```

```
            plt.ylabel('TFR')
            plt.title(f'Total Fertility Rate (TFR) and Forecasts {self.country_name}')
            plt.legend()
            plt.grid(True)
            plt.show()

            print("Comparison of Forecasted Values:")
            print(self.comparison_df[-self.forecast_steps:-1])

            actual = self.comparison_df.loc[self.comparison_df['Year'] >= self.start_year, 'Historical TFR']
            for model in self.forecast_results.keys():
                predicted = self.comparison_df.loc[self.comparison_df['Year'] >= self.start_year, model]
                self.rmses[model] = self.calculate_rmse(actual, predicted)

            for model, rmse in self.rmses.items():
                print(f"RMSE for {model}: {rmse}")

            best_model = min(self.rmses, key=self.rmses.get)
            print(f"The best model based on RMSE is: {best_model}")

            return best_model, self.rmses
        data_files_path_usa = 'data/USA/20230328/'

        file_configs_usa = {
            'USApprVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'PPR0_1', 'PPR1_2', 'PPR2_3', 'PPR3_4']},
            'USAasfrRR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR']},
```

```
        'USAasfrTR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'Cohort',
'ASFR']},
        'USAasfrVH.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR']},
        'USAasfrVV.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR']},
        'USAasfrRRbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'USAasfrVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'USAasfrVVbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']}
    }
    tfr_forecast_usa = TFRForecast(start_year=2012,
data_files_path=data_files_path_usa, file_configs=file_configs_usa,
country_name="USA", forecast_steps=9)
    best_model, rmses = tfr_forecast_usa.run_analysis()
    data_files_path_swe = 'data/SWE/20230414/'

    file_configs_swe = {
        'SWEpprVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'PPR0_1',
'PPR1_2', 'PPR2_3', 'PPR3_4']},
        'SWEasfrRR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR']},
        'SWEasfrTR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'Cohort',
'ASFR']},
        'SWEasfrVH.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR']},
        'SWEasfrVV.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR']},
        'SWEasfrRRbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
```

```
        'SWEasfrVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'SWEasfrVVbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']}
    }
    tfr_forecast_swe = TFRForecast(start_year=2012,
data_files_path=data_files_path_swe, file_configs=file_configs_swe,
country_name="Sweden", forecast_steps=10)
    best_model, rmses = tfr_forecast_swe.run_analysis()
    data_files_path_ukr = 'data/UKR/20160118/'

    file_configs_ukr = {
        'UKRpprVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'PPR0_1',
'PPR1_2', 'PPR2_3', 'PPR3_4']},
        'UKRasfrRR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR']},
        'UKRasfrTR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'Cohort',
'ASFR']},
        'UKRasfrVH.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR']},
        'UKRasfrVV.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR']},
        'UKRasfrRRbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'UKRasfrVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'UKRasfrVVbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']}
    }
    tfr_forecast_ukr = TFRForecast(start_year=2009,
data_files_path=data_files_path_ukr, file_configs=file_configs_ukr,
country_name="Ukraine", forecast_steps=7)
```

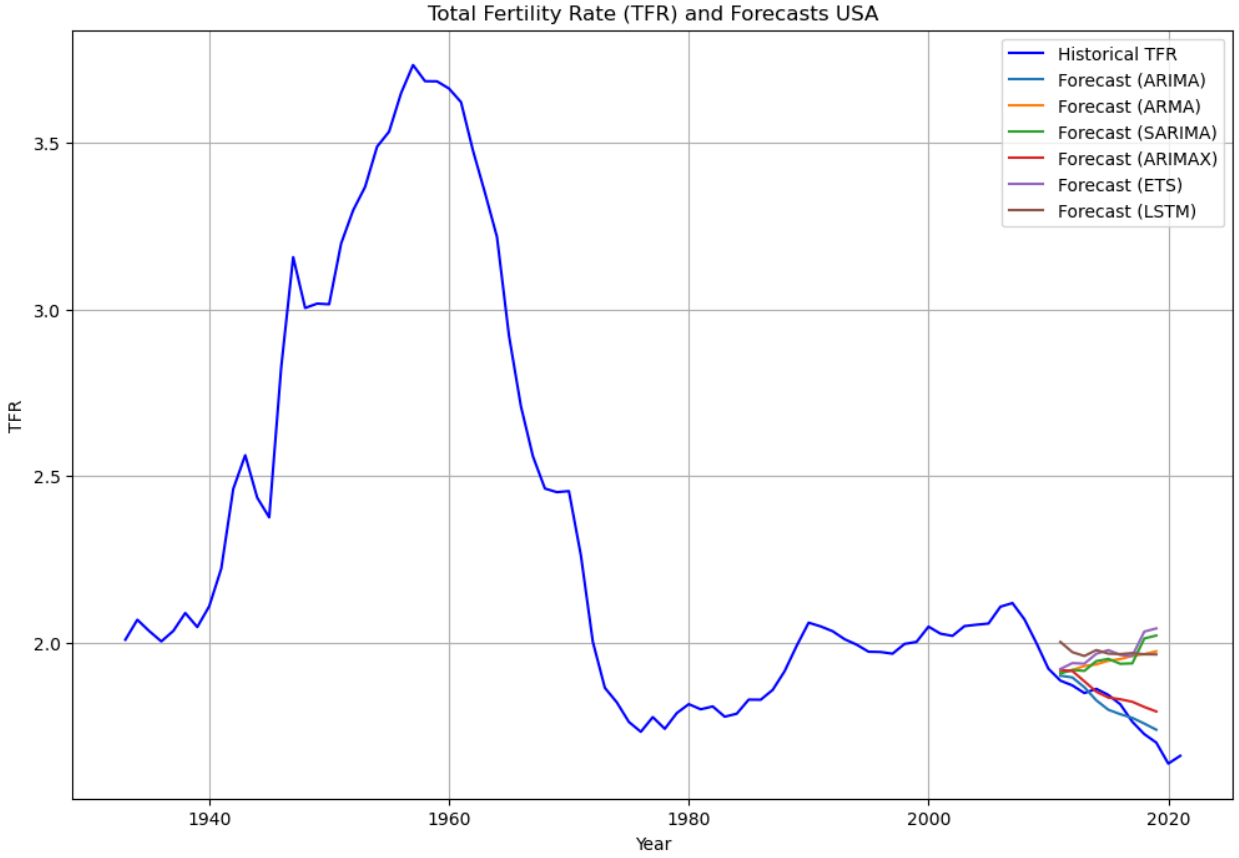
```
best_model, rmses = tfr_forecast_ukr.run_analysis()
data_files_path_bgr = 'data/BGR/20230307/'

file_configs_bgr = {
  'BGRpprVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'PPR0_1', 'PPR1_2', 'PPR2_3', 'PPR3_4']},
  'BGRasfrRR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR']},
  'BGRasfrTR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'Cohort', 'ASFR']},
  'BGRasfrVH.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR']},
  'BGRasfrVV.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort', 'ASFR']},
  'BGRasfrRRbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
  'BGRasfrVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
  'BGRasfrVVbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort', 'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']}
}
tfr_forecast_isl = TFRForecast(start_year=2011, data_files_path=data_files_path_bgr, file_configs=file_configs_bgr, country_name="Bulgaria", forecast_steps=10)
best_model, rmses = tfr_forecast_isl.run_analysis()
data_files_path_hun = 'data/HUN/20220314/'

file_configs_hun = {
  'HUNpprVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'PPR0_1', 'PPR1_2', 'PPR2_3', 'PPR3_4']},
  'HUNasfrRR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR']},
```

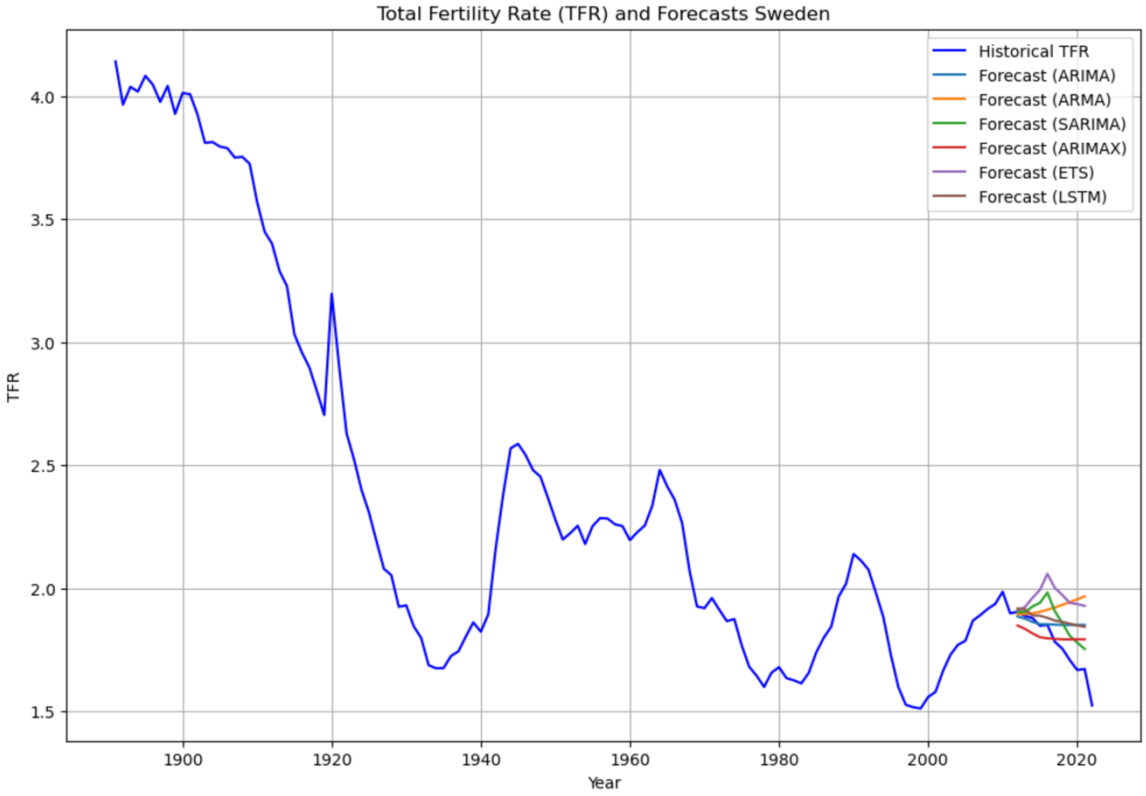
```
        'HUNasfrTR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'Cohort',
'ASFR']},
        'HUNasfrVH.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR']},
        'HUNasfrVV.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR']},
        'HUNasfrRRbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'HUNasfrVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'HUNasfrVVbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']}
    }
    tfr_forecast_rus = TFRForecast(start_year=2011,
data_files_path=data_files_path_hun, file_configs=file_configs_hun,
country_name="Hungary", forecast_steps=10)
    best_model, rmses = tfr_forecast_rus.run_analysis()
    data_files_path_rus = 'data/RUS/20201118/'

    data_files_path_cze = 'data/CZE/20230307/'

    file_configs_cze = {
        'CZEpprVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'PPR0_1',
'PPR1_2', 'PPR2_3', 'PPR3_4']},
        'CZEasfrRR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR']},
        'CZEasfrTR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'Cohort',
'ASFR']},
        'CZEasfrVH.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR']},
        'CZEasfrVV.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR']},
```

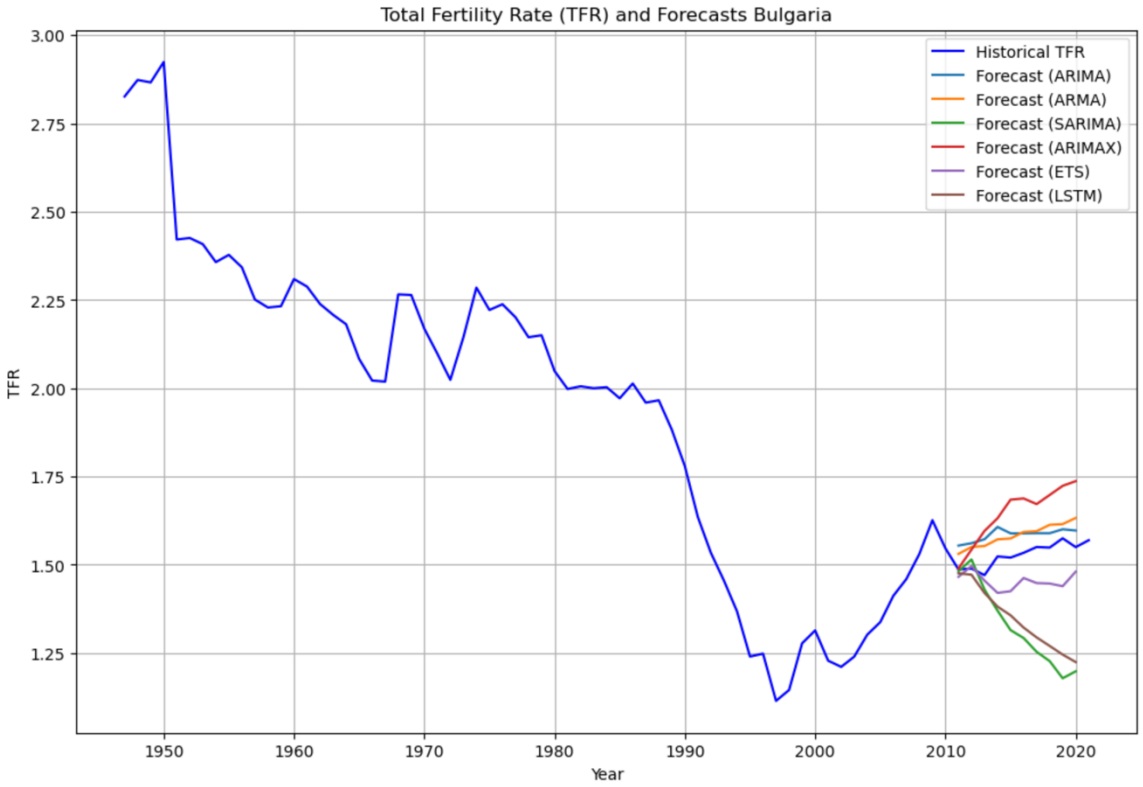
```
        'CZEasfrRRbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'CZEasfrVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'CZEasfrVVbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']}
    }
    tfr_forecast_cze = TFRForecast(start_year=2009,
data_files_path=data_files_path_cze, file_configs=file_configs_cze,
country_name="Czech Republic", forecast_steps=10)
    best_model, rmses = tfr_forecast_cze.run_analysis()

    file_configs_aut = {
        'AUTpprVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'PPR0_1',
'PPR1_2', 'PPR2_3', 'PPR3_4']},
        'AUTasfrRR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR']},
        'AUTasfrTR.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'Cohort',
'ASFR']},
        'AUTasfrVH.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR']},
        'AUTasfrVV.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR']},
        'AUTasfrRRbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'AUTasfrVHbo.txt': {'skip_rows': 2, 'col_names': ['Cohort', 'Age', 'ASFR',
'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']},
        'AUTasfrVVbo.txt': {'skip_rows': 2, 'col_names': ['Year', 'ARDY', 'Cohort',
'ASFR', 'ASFR1', 'ASFR2', 'ASFR3', 'ASFR4', 'ASFR5p']}
    }
```

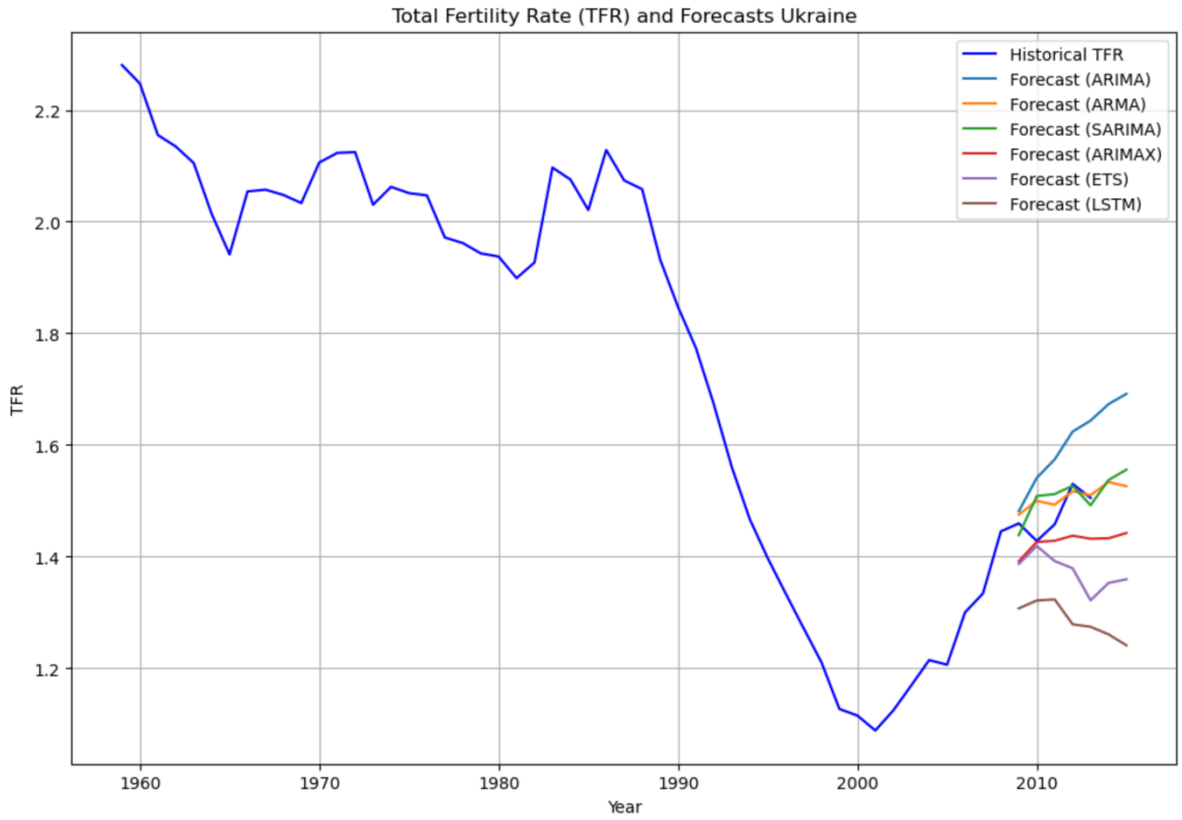
```
        tfr_forecast_aut = TFRForecast(start_year=2010,
data_files_path=data_files_path_aut, file_configs=file_configs_aut,
country_name="Austria", forecast_steps=10)
        best_model, rmses = tfr_forecast_aut.run_analysis()
```

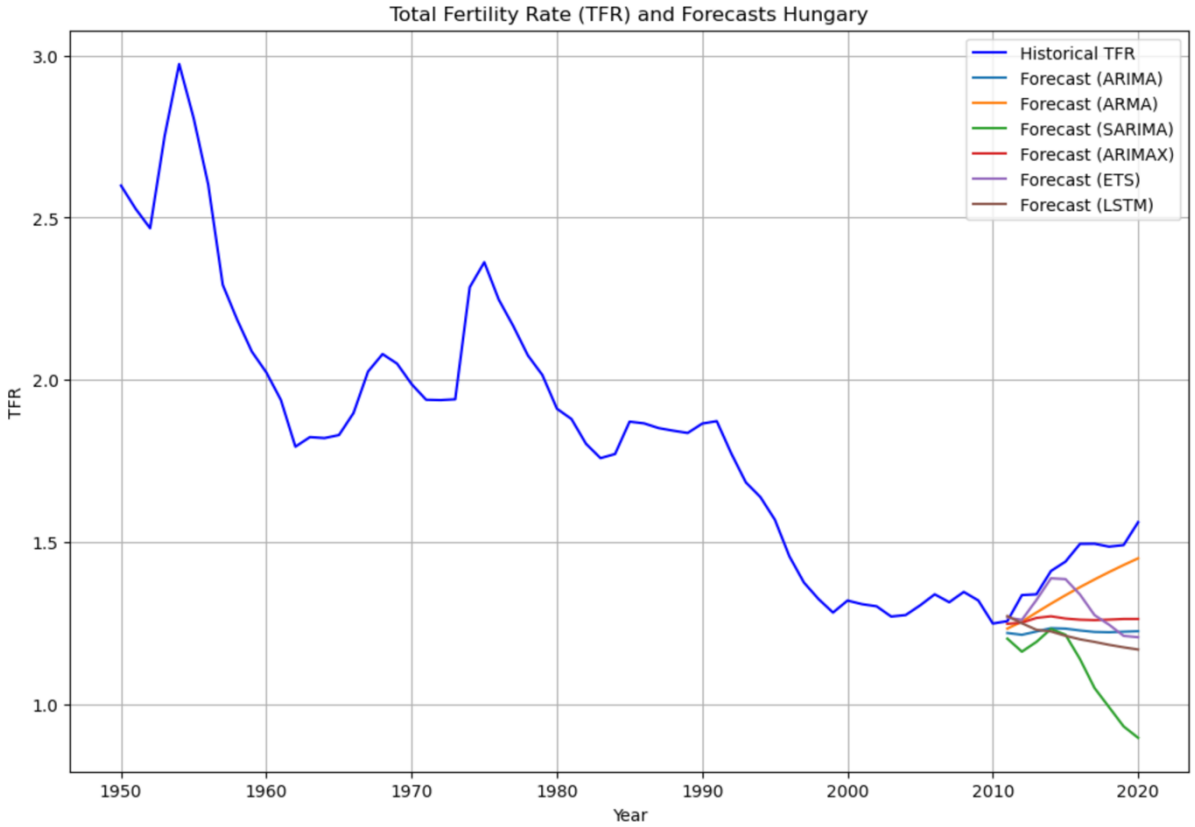
APPENDIX B – PLOTTED RESULTS

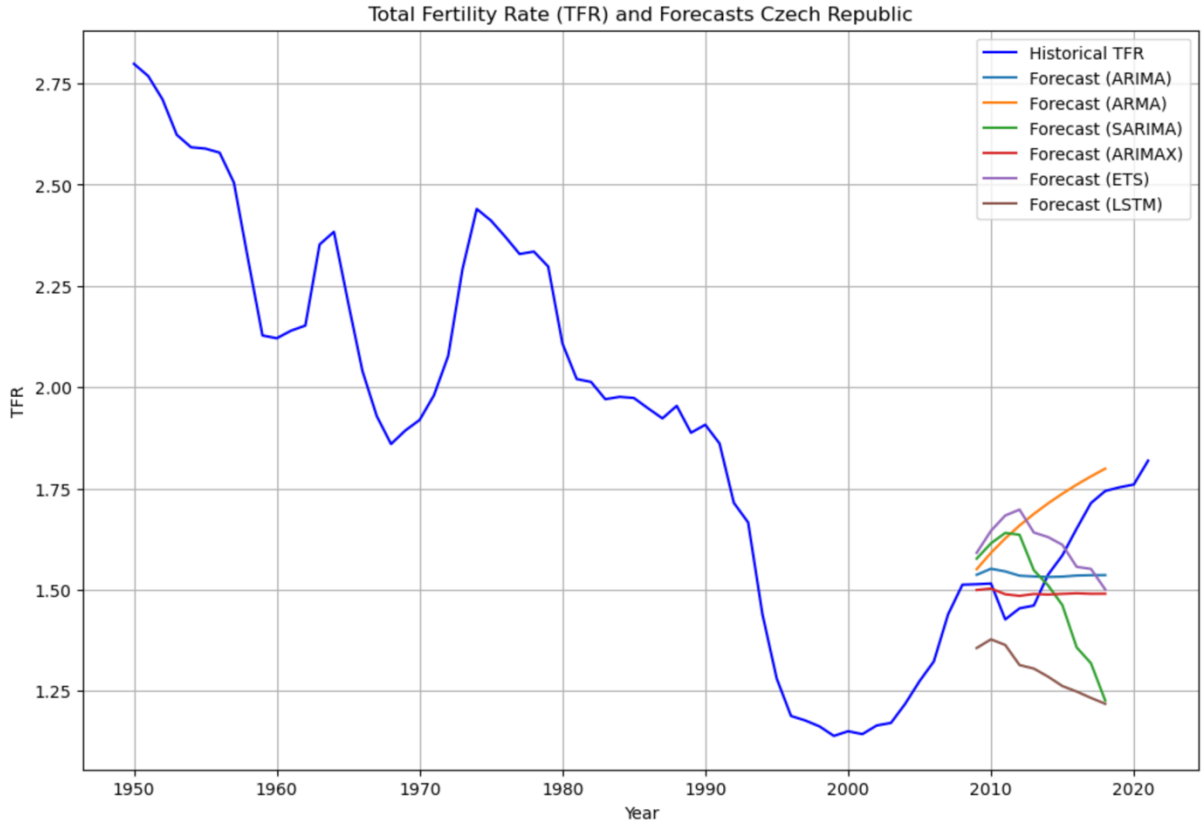



Total Fertility Rate (TFR) and Forecasts Sweden
Historical TFR
Forecast (ARIMA)
Forecast (ARMA)
Forecast (SARIMA)
Forecast (ARIMAX)
Forecast (ETS)
Forecast (LSTM)
TFR
Year
4.0
3.5
3.0
2.5
2.0
1.5
1900
1920
1940
1960
1980
2000
2020


Total Fertility Rate (TFR) and Forecasts Bulgaria
Historical TFR
Forecast (ARIMA)
Forecast (ARMA)
Forecast (SARIMA)
Forecast (ARIMAX)
Forecast (ETS)
Forecast (LSTM)
TFR
Year
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1950
1960
1970
1980
1990
2000
2010
2020


Total Fertility Rate (TFR) and Forecasts Ukraine
Historical TFR
Forecast (ARIMA)
Forecast (ARMA)
Forecast (SARIMA)
Forecast (ARIMAX)
Forecast (ETS)
Forecast (LSTM)
TFR
Year
2.2
2.0
1.8
1.6
1.4
1.2
1960
1970
1980
1990
2000
2010


Total Fertility Rate (TFR) and Forecasts Hungary
Historical TFR
Forecast (ARIMA)
Forecast (ARMA)
Forecast (SARIMA)
Forecast (ARIMAX)
Forecast (ETS)
Forecast (LSTM)
TFR
Year
3.0
2.5
2.0
1.5
1.0
1950
1960
1970
1980
1990
2000
2010
2020


Total Fertility Rate (TFR) and Forecasts Czech Republic
Historical TFR
Forecast (ARIMA)
Forecast (ARMA)
Forecast (SARIMA)
Forecast (ARIMAX)
Forecast (ETS)
Forecast (LSTM)
TFR
Year
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1950
1960
1970
1980
1990
2000
2010
2020

APPENDIX C – RESULTS IN TABLE FORMAT

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS	LSTM
80	2013.0	1.84920	1.866599	1.930343	1.915769	1.884453	1.937378	1.960399
81	2014.0	1.86182	1.827294	1.935278	1.945134	1.853215	1.967164	1.978147
82	2015.0	1.84357	1.799158	1.946007	1.951592	1.835673	1.978118	1.967552
83	2016.0	1.81542	1.785806	1.951184	1.936963	1.830668	1.963472	1.966144
84	2017.0	1.76318	1.774780	1.960732	1.938277	1.823024	1.960589	1.969181
85	2018.0	1.72602	1.757617	1.966002	2.012866	1.807494	2.033640	1.965568
86	2019.0	1.70055	1.739211	1.974560	2.021794	1.793931	2.043161	1.965110
87	2020.0	1.63762	NaN	NaN	NaN	NaN	NaN	NaN

RMSE for ARIMA: 0.02935950381332521

RMSE for ARMA: 0.15472478777650886

RMSE for SARIMA: 0.16883445619916446

RMSE for ARIMAX: 0.051036917889441265

RMSE for ETS: 0.1872481975632821

RMSE for LSTM: 0.1690491427062067

The best model based on RMSE is: ARIMA

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS	LSTM
122	2013.0	1.88752	1.877359	1.895218	1.904030	1.835597	1.924660	1.914644
123	2014.0	1.87984	1.864050	1.898743	1.926440	1.817243	1.961718	1.889697
124	2015.0	1.84792	1.854768	1.904951	1.941360	1.801828	1.994045	1.889090
125	2016.0	1.85238	1.854592	1.913089	1.983441	1.797204	2.059239	1.880281
126	2017.0	1.78314	1.852662	1.922601	1.907862	1.795174	2.002220	1.869502
127	2018.0	1.75560	1.851437	1.933072	1.858879	1.793273	1.973206	1.864120
128	2019.0	1.70891	1.851488	1.944198	1.807606	1.792973	1.941112	1.856691
129	2020.0	1.66851	1.851638	1.955750	1.780193	1.793338	1.936521	1.849681
130	2021.0	1.67278	1.851527	1.967562	1.754071	1.793500	1.929447	1.844006

RMSE for ARIMA: 0.10029773452478224

RMSE for ARMA: 0.16827097859794155

RMSE for SARIMA: 0.09132192640998514

RMSE for ARIMAX: 0.07324900183267327

RMSE for ETS: 0.18954065149866797

RMSE for LSTM: 0.10330326968013477

The best model based on RMSE is: ARIMAX

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS	LSTM
50	2009.0	1.45958	1.481471	1.475303	1.438592	1.391754	1.387040	1.307367
51	2010.0	1.42796	1.541109	1.499900	1.508511	1.426301	1.419038	1.321504
52	2011.0	1.45798	1.574161	1.493034	1.512125	1.428492	1.392089	1.323430
53	2012.0	1.53036	1.623940	1.517210	1.527030	1.437566	1.379169	1.278805
54	2013.0	1.50516	1.643658	1.509948	1.491985	1.432067	1.321852	1.274356
55	2014.0	NaN	1.673150	1.533721	1.537142	1.432933	1.352918	1.260931

RMSE for ARIMA: 0.10461252871645284

RMSE for ARMA: 0.03700593266401938

RMSE for SARIMA: 0.04482247902368429

RMSE for ARIMAX: 0.062331332217799955

RMSE for ETS: 0.11501637150906216

RMSE for LSTM: 0.1839324481650408

The best model based on RMSE is: ARMA

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS	LSTM
65	2012.0	1.48949	1.561238	1.550433	1.515057	1.543130	1.496449	1.471910
66	2013.0	1.47074	1.572411	1.553204	1.429795	1.596037	1.455270	1.420256
67	2014.0	1.52378	1.607323	1.572130	1.370164	1.631891	1.420503	1.381722
68	2015.0	1.52050	1.589144	1.574665	1.315011	1.684580	1.425229	1.356951
69	2016.0	1.53402	1.589020	1.593072	1.292692	1.688013	1.462643	1.322277
70	2017.0	1.55042	1.589701	1.595381	1.253515	1.672076	1.448352	1.294379
71	2018.0	1.54879	1.589534	1.613285	1.227254	1.697802	1.446960	1.270266
72	2019.0	1.57511	1.600735	1.615379	1.178650	1.723785	1.439471	1.245684
73	2020.0	1.55013	1.597118	1.632795	1.198343	1.737009	1.480653	1.224566

RMSE for ARIMA: 0.06390280749208717

RMSE for ARMA: 0.059873290120834775

RMSE for SARIMA: 0.2447230819145974

RMSE for ARIMAX: 0.13226538524130255

RMSE for ETS: 0.0834122477953325

RMSE for LSTM: 0.21269433594542442

The best model based on RMSE is: ARMA

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS	LSTM
61	2011.0	1.25612	1.219613	1.233100	1.202128	1.247916	1.267305	1.271610
62	2012.0	1.33641	1.213950	1.254492	1.162027	1.251518	1.260010	1.249640
63	2013.0	1.33881	1.224116	1.282639	1.192072	1.265910	1.321243	1.229336
64	2014.0	1.41067	1.234391	1.309935	1.232042	1.271091	1.388168	1.224385
65	2015.0	1.43961	1.233048	1.336020	1.214071	1.264121	1.385652	1.210955
66	2016.0	1.49434	1.227349	1.360927	1.139002	1.260543	1.338098	1.200119
67	2017.0	1.49474	1.222846	1.384709	1.049895	1.259188	1.274593	1.192078
68	2018.0	1.48579	1.221840	1.407415	0.990870	1.260819	1.245108	1.183077
69	2019.0	1.49060	1.223529	1.429095	0.931632	1.262859	1.210674	1.175370

RMSE for ARIMA: 0.22414688407336406

RMSE for ARMA: 0.09138966717641893

RMSE for SARIMA: 0.382264116739097

RMSE for ARIMAX: 0.19092460589043037

RMSE for ETS: 0.18566102020243447

RMSE for LSTM: 0.251017501955049

The best model based on RMSE is: ARMA

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS	LSTM
62	2012.0	1.45419	1.534763	1.658562	1.635784	1.484899	1.697882	1.314213
63	2013.0	1.46109	1.532859	1.687212	1.548451	1.489651	1.641403	1.305551
64	2014.0	1.53742	1.531574	1.713337	1.510729	1.488333	1.630056	1.285486
65	2015.0	1.58499	1.532458	1.737326	1.462068	1.489945	1.610988	1.262371
66	2016.0	1.65106	1.535132	1.759468	1.357735	1.491366	1.556578	1.248787
67	2017.0	1.71409	1.535995	1.779986	1.318673	1.490135	1.551568	1.233077
68	2018.0	1.74388	1.536264	1.799054	1.226334	1.490312	1.500783	1.218523
69	2019.0	1.75294	NaN	NaN	NaN	NaN	NaN	NaN
70	2020.0	1.75986	NaN	NaN	NaN	NaN	NaN	NaN

RMSE for ARIMA: 0.10891794158257916

RMSE for ARMA: 0.14606715077246887

RMSE for SARIMA: 0.2502331133201577

RMSE for ARIMAX: 0.12545045160632232

RMSE for ETS: 0.16863331566055453

RMSE for LSTM: 0.3046800735053643

The best model based on RMSE is: ARIMA

Comparison of Forecasted Values:

	Year	Historical TFR	ARIMA	ARMA	SARIMA	ARIMAX	ETS	LSTM
59	2010.0	1.44276	1.393295	1.401940	1.408570	1.419661	1.393013	1.381252
60	2011.0	1.43043	1.387721	1.413738	1.412655	1.411345	1.379501	1.382352
61	2012.0	1.44002	1.382955	1.429247	1.438152	1.412536	1.373091	1.371546
62	2013.0	1.43559	1.380785	1.447937	1.440451	1.410769	1.353765	1.364632
63	2014.0	1.46365	1.377664	1.469286	1.491264	1.416021	1.378244	1.360299
64	2015.0	1.49008	1.375577	1.492784	1.528484	1.411416	1.412493	1.353784
65	2016.0	1.52912	1.373621	1.517938	1.576850	1.414086	1.474200	1.348772
66	2017.0	1.51753	1.371976	1.544277	1.592577	1.412932	1.484049	1.344344
67	2018.0	1.47425	1.370637	1.571357	1.587702	1.412999	1.451639	1.339774

RMSE for ARIMA: 0.09768213663296785

RMSE for ARMA: 0.05625661429411081

RMSE for SARIMA: 0.059318297459533674

RMSE for ARIMAX: 0.06397207094593733

RMSE for ETS: 0.05938426266023958

RMSE for LSTM: 0.11883380756341365

The best model based on RMSE is: ARMA