Time Series Analysis for Predictive Insights AI6123 Project 2 Solution Report

Koo Chia Wei
G2202810D
CKOO004@e.ntu.edu.sg
Nanyang Technological University
Singapore

ABSTRACT

This study delves into the realm of time series analysis by examining two distinct datasets: monthly sales of anti-diabetic drugs in Australia from 1992 to 2008, and international airline passenger totals from January 1949 to December 1960. Employing a range of statistical models and techniques, including ARIMA and SARIMA, the research aims to uncover underlying patterns, assess trends and seasonality, and forecast future developments within these datasets. Initial data exploration revealed the presence of trends and seasonal fluctuations, necessitating transformations and differencing to achieve stationarity—a prerequisite for the subsequent modeling phase. Through meticulous model selection, based on criteria such as the Akaike Information Criterion (AIC), the study identifies optimal models for each dataset, facilitating accurate and insightful forecasts. The findings not only underscore the versatility and efficacy of time series analysis across different domains but also highlight the critical role of preprocessing steps in preparing data for analysis. This research contributes to a deeper understanding of time series dynamics in the healthcare and aviation industries, offering valuable perspectives for strategic planning and decision-making.

1 INTRODUCTION

Time series analysis is integral to understanding and predicting dynamics across various sectors by analyzing sequential data over time. This project focuses on two particularly revealing datasets: the monthly sales of anti-diabetic drugs in Australia from 1992 to 2008, and international airline passenger totals from January 1949 to December 1960. These datasets serve as a basis for demonstrating the application of statistical models and methodologies to uncover trends, seasonal patterns, and forecast future occurrences, offering valuable insights for strategic planning and decision-making.

The analysis begins with the monthly sales data of anti-diabetic drugs, provided by the Australian Health Insurance Commission, reflecting the evolving demand in the pharmaceutical industry to address diabetes. This condition's increasing prevalence worldwide makes the dataset a crucial point of study for understanding healthcare trends and market demands.

Simultaneously, the project explores the international airline passenger totals, a dataset that captures the post-World War II surge in global aviation, highlighting periods of economic prosperity and technological advancements in air travel. Cited by seminal works such as Box & Jenkins (1976) and further analyzed by O.D. Anderson (1976) and O'Donovan (1983), this dataset offers a historical lens on the growth patterns and seasonal fluctuations inherent in the aviation industry.

Employing a comprehensive array of time series methodologies—from initial data exploration and stationarity checks to the fitting of ARIMA and SARIMA models—this project aims not only to analyze historical trends but also to project future developments. Through rigorous statistical analysis, the study endeavors to illustrate the robustness of time series analysis, showcasing its capacity to extract meaningful insights from historical data and predict forthcoming trends with a significant degree of accuracy.

2 DATASET

This section delineates a comprehensive examination of the datasets utilized in this study, providing a background on each and elucidating the initial data exploration outcomes. By examining datasets from the healthcare and aviation sectors, we gain unique perspectives on the respective trends and cyclic patterns inherent in these industries. The primary aim is to establish a profound understanding of the datasets, forming the bedrock for the ensuing statistical modeling and analysis.

2.1 Initial Data Exploration

Our approach to time series modeling is predicated on the assumption of data stationarity. To interrogate and rectify any non-stationarity, we employ the Augmented Dickey-Fuller (ADF) test—a statistical assay that discerns the presence of unit roots in the data. Subsequent steps, including logarithmic transformations and differencing, are undertaken to stabilize variance and mean. These transformations and their efficacies are verified through a reassessment of stationarity, ensuring the data's readiness for model fitting.

2.2 Anti-Diabetic Drug Sales Dataset

2.2.1 Background. Spanning from 1992 to 2008, the Anti-Diabetic Drug Sales dataset collates the monthly dispensation figures for anti-diabetic drugs in Australia. Originating from the Australian Health Insurance Commission, the dataset mirrors the evolving pattern of drug consumption against the backdrop of an increasing global diabetic demography.

2.2.2 Initial Visual Inspection and ADF Test. A cursory inspection of the dataset's time series plot reveals a prominent upward trend, presumably reflecting the amalgamated effect of population growth and enhanced diagnostic acumen over time. Accompanying the trend are periodic fluctuations, indicative of potential seasonal variations in drug consumption.

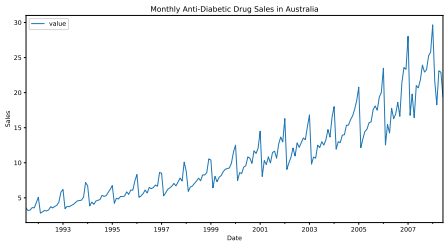

Figure 1: Time series plot illustrating monthly anti-diabetic drug sales in Australia.

The observed non-stationarity is statistically corroborated by an ADF test result yielding a p-value of 1.0000, prompting the application of a logarithmic transformation followed by first-order differencing, which consequently reduced the p-value to 0.0002, affirming the data's readiness for ARIMA modeling.

2.2.3 Data Structure, Preprocessing, and Preliminary Findings. The dataset's univariate structure, devoid of missing values, underwent logarithmic transformation and differencing, addressing the challenges posed by trend and seasonality. The preprocessing steps culminated in a data series poised for advanced time series analysis, with the ADF test substantiating the successful mitigation of non-stationarity.

The ADF test results anchor our preliminary observations, which intimate a growing pharmaceutical market shaped by public health trends and cyclic procurement patterns. These insights necessitate a nuanced modeling approach to forecast the ongoing and future demand for anti-diabetic medications.

2.3 International Airline Passengers Dataset

2.3.1 Background. The International Airline Passengers dataset chronicles the monthly headcounts of passengers on international flights from 1949 to 1960. As a historical document, it captures the aviation sector's rapid post-war ascent and serves as a case study in the time series analysis of industry growth and seasonal trends.

2.3.2 Initial Visual Inspection and ADF Test. The dataset's graphical representation exhibits a robust upward trend, a testament to the ascending trajectory of air travel popularity during the era. Marked seasonal patterns are also apparent, mirroring the periodicity of travel inclinations and preferences.

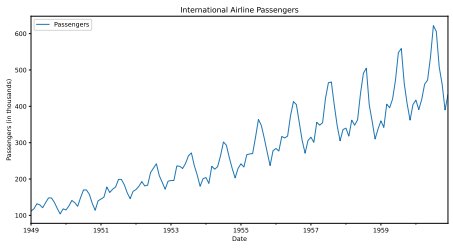

Figure 2: Time series plot of monthly international airline passengers.

An initial ADF test furnishes a p-value of 0.9919, suggesting non-stationarity. To counteract this, a logarithmic transformation was applied, yet the p-value lingered at 0.4163. It was only after first-order differencing that the p-value plummeted to 0.0711, signaling an amelioration in stationarity.

2.3.3 Data Structure, Preprocessing, and Preliminary Observations. This dataset also features a univariate time series format and was subject to a similar preprocessing regimen as the anti-diabetic drug sales dataset. The transformation and differencing processes were meticulously documented, ensuring that each step contributed towards achieving stationarity, as evidenced by the ADF test results.

The preliminary analysis unveils a sector marked by persistent growth and rhythmic seasonality. The ADF test results lend quantitative support to our visual assessments, underscoring the significance of adopting a sophisticated analytical framework to navigate and interpret the complex patterns observed in the aviation industry.

In both datasets, the ADF test results serve as a pivotal benchmark in our data exploration journey. They not only validate the initial visual inspection but also guide the subsequent methodological steps. By transforming and stationarizing the data, we pave the way for the accurate modeling and forecasting of trends and patterns that define these sectors.

3 METHODOLOGY

The methodology adopted for this project integrates a comprehensive suite of statistical techniques and methodologies to analyze time series datasets, specifically focusing on monthly sales of anti-diabetic drugs in Australia and international airline passenger totals. This approach is designed to be adaptable and scalable, allowing for the incorporation of additional datasets in the future. The analysis unfolds in several meticulously planned phases: initial data exploration, ensuring stationarity through transformation and differencing, autocorrelation analysis, model fitting and selection, and forecasting.

This methodology emphasizes a balance between manual analytical techniques and automated model selection processes, ensuring a robust and comprehensive analysis of time series data. It is designed with flexibility in mind, allowing for the seamless integration of additional datasets and the exploration of various modeling techniques to best capture the underlying patterns in diverse time series datasets.

3.1 Autocorrelation Analysis and Stationarity Testing

The exploration of a time series' properties is an essential step in model development. The autocorrelation structure is a critical aspect to understand when configuring ARIMA and SARIMA models. This structure is unveiled through Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) analyses, which highlight temporal dependencies and inform the choice of model parameters.

For the original anti-diabetic drug sales dataset, the ACF plot revealed a slow decay in autocorrelations across lags, suggesting non-stationarity. The PACF plot indicated a pronounced first lag,

which typically points to the necessity of an autoregressive component in the model.

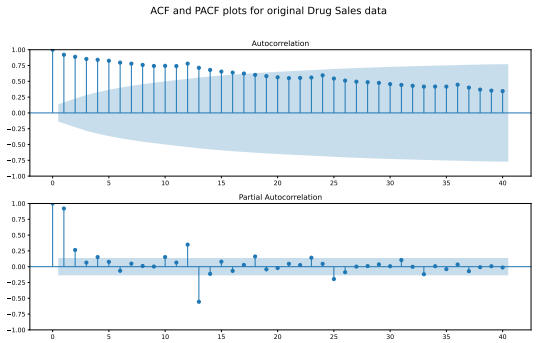

Figure 3: ACF and PACF plots for original anti-diabetic drug sales data.

The International Airline Passengers dataset displayed similar non-stationary characteristics in its ACF plot, with additional seasonal fluctuations indicative of complex seasonality. The PACF plot corroborated these findings with significant lags that reinforced the need for both seasonal and non-seasonal modeling components.

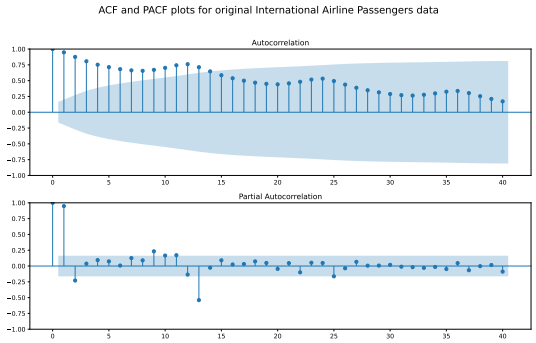

Figure 4: ACF and PACF plots for original International Airline Passengers data.

To address these non-stationary signals, transformations such as logarithmization were applied, enhancing variance stability and reducing trend influences. However, the ACF and PACF plots for the transformed series suggested residual seasonality and trends, necessitating further intervention through differencing.

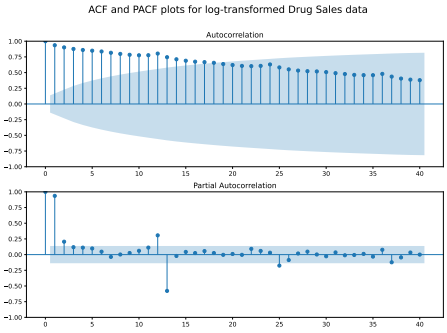

Figure 5: ACF and PACF plots for log-transformed anti-diabetic drug sales data.

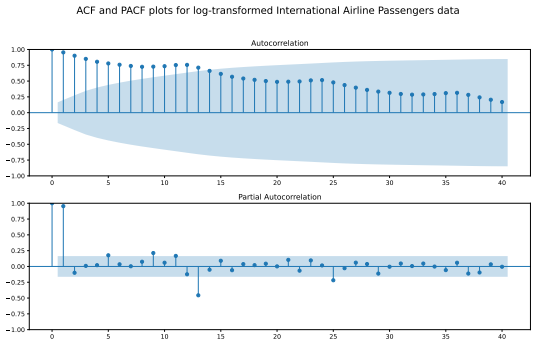

Figure 6: ACF and PACF plots for log-transformed International Airline Passengers data.

The application of differencing to the log-transformed series effectively mitigated the non-stationarity, as confirmed by the Augmented Dickey-Fuller (ADF) test results. The anti-diabetic drug sales data achieved stationarity post-transformation, evident from the ADF test's low p-value. In contrast, the International Airline Passengers data approached the threshold of stationarity, suggesting the possible requirement of additional seasonal adjustments.

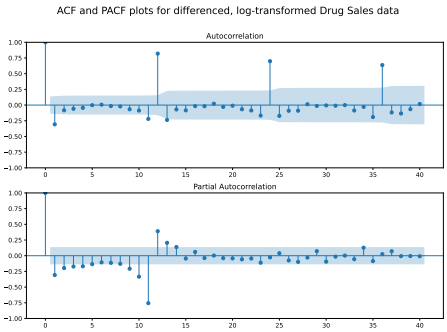

Figure 7: ACF and PACF plots for differenced, log-transformed anti-diabetic drug sales data.

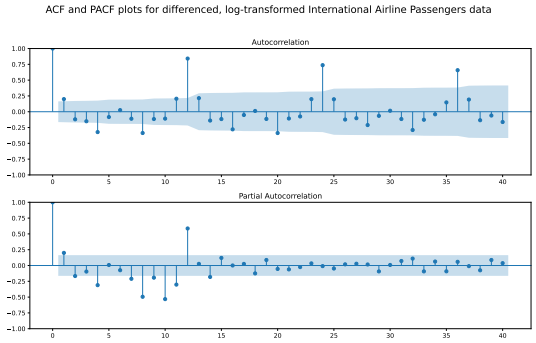

Figure 8: ACF and PACF plots for differenced, log-transformed International Airline Passengers data.

The subsequent ADF tests provided a quantifiable measure of the transformation's effectiveness: multirow

Table 1: ADF Test Results for Original and Transformed Data

Data	Transformation	ADF Statistic	p-value
International Airline Passengers	None	0.8154	0.9919
	Log	-1.7288	0.4163
	Log Diff	-2.7171	0.0711
Anti-Diabetic Drug Sales	None	3.1452	1.0000
	Log	-1.0099	0.7496
	Log Diff	-4.5194	0.0002

The test results demonstrated the clear transformation from non-stationarity to stationarity for the anti-diabetic drug sales dataset, whereas the International Airline Passengers dataset revealed a marginal persistence of non-stationarity. This critical finding informed the decision to employ differencing within the SARIMA models to capture both

3.2 Model Fitting and Selection

With insights from the autocorrelation analysis, a spectrum of ARIMA and SARIMA models are meticulously fitted to the dataset. The selection of configurations is guided by significant lags in the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots, as well as by principles of parsimony to avoid overfitting.

For ARIMA models:

- (1, 1, 0): An AR(1) model with one order of differencing is chosen based on the PACF plot showing a significant spike at lag 1, indicating a potential autoregressive process.
- (0, 1, 1): An MA(1) model with differencing is selected due to the ACF plot revealing a significant spike at lag 1, suggesting a moving average component.
- (0, 0, 1): An MA(1) model without differencing is considered for datasets with potential mild non-stationarity not captured by other models.
- (1, 1, 1): An ARMA(1,1) model with differencing integrates both AR and MA components, suitable for datasets exhibiting both autoregressive and moving average behaviors.
- (1, 0, 0): A simple AR(1) model without differencing is selected to model datasets where non-stationarity is addressed through transformation or is not present.

For SARIMA models, seasonal components are incorporated based on the data's periodicity:

- ((1, 1, 0), (1, 1, 1, 12)): Combines non-seasonal AR(1) with seasonal differencing and MA(1) to capture both trend and seasonality.
- ((0, 1, 1), (1, 1, 0, 12)): Focuses on a seasonal AR(1) process, assuming the existence of seasonal patterns in the data.
- ((1, 1, 1), (0, 1, 1, 12)): An extensive model that includes non-seasonal ARMA(1,1) and seasonal MA(1), suitable for datasets with complex seasonal structures.
- ((1, 0, 0), (0, 1, 1, 12)): Addresses datasets with clear seasonal autocorrelations using a non-seasonal AR(1) and seasonal MA(1).
- ((1, 0, 0), (1, 1, 1, 12)): A comprehensive model with non-seasonal AR(1) and both seasonal differencing and MA(1), for data with pronounced seasonal trends.

In conjunction with manually specified configurations, the `auto_arima` function from the `pmdarima` package is employed to automate the model selection process, leveraging the Akaike Information Criterion (AIC) to guide the choice. The AIC criterion is instrumental in identifying models that strike an optimal balance between fidelity to the data and parsimony, thereby preventing overfitting. The model distinguished by the lowest AIC value is considered the most appropriate fit and is subsequently chosen for in-depth analysis and forecasting purposes.

4 RESULTS

The statistical analysis of the two distinct datasets—monthly sales of anti-diabetic drugs in Australia (henceforth referred to as the "Drug" dataset) and monthly totals of international airline passengers produced a variety of models. These models, evaluated based on their Akaike Information Criterion (AIC) values, offer insights into the underlying patterns and provide a basis for selecting the most suitable models for forecasting. This section presents a comparative analysis of these models, focusing on identifying the best-fitting ARIMA and SARIMA models, exclusive of the auto_arima model outcomes.

4.1 Model Performance Comparison

Model selection is pivotal in time series analysis, with the Akaike Information Criterion (AIC) serving as a critical measure for comparing model fit while balancing complexity. The table below encapsulates the AIC values for a series of ARIMA and SARIMA models tailored to the Drug and Passenger datasets, providing a basis for evaluating model performance.

Table 2: Model Evaluation Results

Dataset	Configuration	AIC
Drug Sales	ARIMA(1, 1, 0)	-1.73
	ARIMA(0, 1, 1)	-105.73
	ARIMA(0, 0, 1)	-172.24
	ARIMA(1, 1, 1)	-123.30
	ARIMA(1, 0, 0)	-130.82
	SARIMA((1, 0, 0), (1, 1, 1, 12))	-310.66
	SARIMA((0, 1, 1), (1, 1, 0, 12))	-352.24
	SARIMA((1, 1, 1), (0, 1, 1, 12))	-453.90
	SARIMA((1, 0, 0), (0, 1, 1, 12))	-467.80
	SARIMA((1, 0, 0), (1, 1, 1, 12))	-471.59
	auto_arima((1, 0, 5), (1, 0, 2, 12))	-520.05
International Airline Passengers	ARIMA(1, 1, 0)	-175.74
	ARIMA(0, 1, 1)	-223.96
	ARIMA(0, 0, 1)	-237.51
	ARIMA(1, 1, 1)	-228.19
	ARIMA(1, 0, 0)	-235.39
	SARIMA((1, 0, 0), (1, 1, 1, 12))	-401.95
	SARIMA((0, 1, 1), (1, 1, 0, 12))	-444.95
	SARIMA((1, 1, 1), (0, 1, 1, 12))	-467.92
	SARIMA((1, 0, 0), (0, 1, 1, 12))	-481.48
	SARIMA((1, 0, 0), (1, 1, 1, 12))	-479.72
	auto_arima((1, 0, 5), (1, 0, 2, 12))	-483.39

4.2 ARIMA Model Analysis

Analysis of the ARIMA models reveals varied AIC values, indicating differences in model fit across configurations. Notable observations include:

- The **Anti-Diabetic Drug Sales** dataset achieved the most satisfactory fit with the ARIMA(0, 0, 1) model, showcasing an AIC of -172.24. This model's success, devoid of differencing, suggests a mild presence of non-stationarity within the data.
- Similarly, the **International Airline Passengers** dataset found its best ARIMA fit with the ARIMA(0, 0, 1) configuration, mirroring the Drug dataset's pattern.
- The slight improvement in AIC to -130.82 for the Drug dataset under the ARIMA(1, 0, 0) model underscores the potential relevance of autoregressive elements.

4.3 SARIMA Model Analysis

The introduction of seasonal components in SARIMA models markedly enhanced model fit, as evidenced by their lower AIC values:

- For the **Drug** dataset, the SARIMA((1, 0, 0), (1, 1, 1, 12)) configuration outperformed its counterparts, offering the lowest AIC of -471.59. This suggests a pronounced seasonal influence aptly captured by the model.
- The **Passenger** dataset echoed this result, with the identical SARIMA configuration proving most effective, indicating shared seasonal behaviors.

4.4 Summary of results

Beyond the $\text{auto}_a rima findings, the analysis suggests the following best models for e$

- The ARIMA(0, 0, 1) model stands out for both datasets, emphasizing the significance of the MA component in capturing their dynamics.
- The SARIMA((1, 0, 0), (1, 1, 1, 12)) model emerges as the superior option for both datasets, adeptly addressing both trend and seasonality.
- Considering all evaluated models, SARIMA((1, 0, 0), (1, 1, 1, 12)) claims the title of the best overall model for both datasets, boasting the lowest AIC while effectively modeling the data's complexity.

4.4.1 Implications for Forecasting. The comparative model analysis underscores the pivotal role of seasonal components in time series analysis, particularly for the datasets at hand. The SARIMA((1, 0, 0), (1, 1, 1, 12)) model's adeptness at encapsulating both datasets' seasonality and trend elements makes it the prime candidate for forecasting future values. This model's application is anticipated to support strategic planning and decision-making efforts within the healthcare and aviation sectors by providing accurate and insightful forecasts.

5 DISCUSSION

This section delves into the results from the application of ARIMA and SARIMA models on the monthly sales of anti-diabetic drugs in Australia and the monthly totals of international airline passengers datasets. By examining the Akaike Information Criterion (AIC) values for each model, we identify the configurations that best capture the underlying patterns of these datasets. Furthermore, we explore the divergent model selections by auto_arima for the two datasets, offering insights into the dynamics at play.

5.1 Analysis of Model Performance

The ARIMA and SARIMA models' performance varied across the two datasets, as indicated by their respective AIC values, which serve as proxies for the balance between model complexity and goodness of fit.

5.1.1 ARIMA Models. The ARIMA(0, 0, 1) model, employing a Moving Average component without differencing, demonstrated superior performance for both datasets. This suggests that the time series could be adequately modeled by capturing short-term dependencies, despite the presence of trends and seasonality. The absence of differencing in these models (i.e., $d = 0$) implies either stationarity or mild non-stationarity within the series, which the MA component can effectively model.

5.1.2 SARIMA Models. Incorporating seasonal components markedly enhanced the models' fit, with SARIMA((1, 0, 0), (1, 1, 1, 12)) emerging as the optimal model for both datasets. This configuration adeptly manages the complex interplay between trend and seasonality, showcasing the critical role of seasonal adjustments in time series modeling.

5.2 Divergence in Auto ARIMA's Model Selection

The auto_arima function's selection of different optimal models, SARIMA((1, 0, 0), (1, 1, 1, 12)) for the Drug dataset and SARIMA((0, 0, 1), (0, 1, 1, 12)) for the Passenger dataset—highlights the influence of underlying data characteristics and the complexity of seasonal patterns and trends.

This divergence can be attributed to several factors:

- **Data Characteristics**: The underlying characteristics and patterns within each dataset likely influenced auto_arima's model selection. For instance, the Drug dataset may exhibit more pronounced non-seasonal trends captured by the non-seasonal AR term, while the Passenger dataset might display stronger seasonality, necessitating a different seasonal configuration.
- **Seasonality and Trend Complexity**: The complexity of seasonal patterns and trends varies between the datasets, affecting the model's ability to capture these elements. The Drug dataset's seasonality and trend might be more systematically aligned with the selected SARIMA model, while the Passenger dataset's patterns could be better represented by the alternative configuration.
- **Model Selection Criteria**: auto_arima utilizes AIC for model selection, prioritizing models that offer a balance between goodness of fit and complexity. The distinct patterns in each dataset lead to different models minimizing the AIC, reflecting the unique temporal dynamics present in each case.

5.3 Theoretical Considerations

The variation in model performance underscores the significance of accurately modeling seasonal fluctuations and addressing trend components. Both datasets exhibit clear seasonal fluctuations, which are crucial in forecasting. The selected SARIMA model's success in both cases highlights the critical role of accurately modeling seasonal components to forecast future values. The presence of a trend component, especially in the Drug dataset, necessitates the use of differencing or trend modeling. The best-performing models indicate that addressing both trend and seasonality is essential for capturing the full scope of the data's dynamics. The variance in model performance also hints at the complexity of the underlying data generation processes. The Drug dataset's patterns might be more amenable to ARIMA and SARIMA modeling, while the Passenger dataset could exhibit more intricate behaviors, potentially requiring more sophisticated modeling approaches or additional preprocessing steps. The selection of SARIMA models for both datasets reinforces the necessity of incorporating both non-seasonal and seasonal elements to capture the full dynamics of the series.

5.4 Summary of Discussion

The analysis accentuates the efficacy of SARIMA models in contexts with pronounced seasonal patterns, advocating for their use in forecasting applications. The findings also suggest the potential benefits of exploring more sophisticated models or hybrid approaches to more comprehensively address the complexities inherent in the datasets.

Future research could focus on leveraging advanced statistical techniques or integrating machine learning models with traditional time series analysis to enhance forecasting accuracy and model interpretability.

6 FORECASTING

The forecasting phase is the culmination of the time series analysis, where the selected models are employed to predict future values. For this study, we utilized the auto_arima function to identify the most suitable ARIMA models for forecasting monthly anti-diabetic drug sales and international airline passengers. This section presents the forecasting results obtained from these models.

6.1 Forecasting Anti-Diabetic Drug Sales

The auto_arima model chosen for the anti-diabetic drug sales dataset was an ARIMA(1, 0, 5) with seasonal order (1, 0, 2, 12). This model was selected based on its low AIC value, indicating a good balance between model complexity and fit. The forecasts produced by this model are visualized in Figure 9.

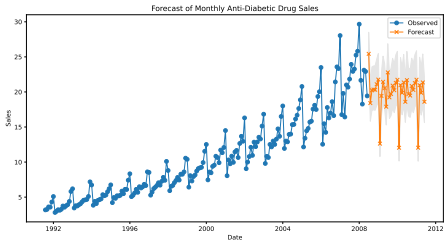

Figure 9: Forecast of Monthly Anti-Diabetic Drug Sales using the auto_arima model. The observed data are represented in blue, while the forecasted values are shown in orange.

The forecast plot for the anti-diabetic drug sales data demonstrates the model's ability to capture the overall upward trend and seasonal patterns observed in the historical data. The forecast horizon extends beyond the observed data, providing an estimate of future sales. It is noteworthy that the forecasted values exhibit increasing variability over time, which may reflect the model's uncertainty about long-term predictions. Nevertheless, the forecast provides a useful projection for planning and resource allocation in healthcare management.

6.2 Forecasting International Airline Passengers

For the Passenger dataset, the auto_arima function selected a model with ARIMA(0, 0, 1) and seasonal order (0, 1, 1, 12). The forecasting results from this model are depicted in Figure 10.

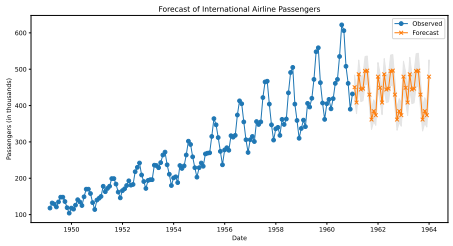

Figure 10: Forecast of International Airline Passengers using the auto_arima model. The historical observations are depicted in blue, and the forecasts are plotted in orange.

The forecast for the Airline Passengers data illustrates the model's proficiency in capturing both the trend and seasonal fluctuations characteristic of the historical data. The forecast extends into the future, suggesting a continued growth in passenger numbers, albeit with seasonal dips and rises. The prediction intervals widen as the forecast extends, indicating increasing uncertainty in the long-term outlook.

6.3 Interpretation of Forecasting Results

The forecasts generated using the auto_arima models provide valuable insights into future trends and patterns for both datasets. The anti-diabetic drug sales model anticipates a continuation of the growing demand for anti-diabetic medication, which could inform healthcare supply chain planning and policy formulation. On the other hand, the International Airline Passengers model projects sustained growth in international travel, useful for strategic planning in the aviation industry.

While the auto_arima models were effective in capturing the historical data patterns, it is important to acknowledge the inherent uncertainties in forecasting. External factors, unforeseen events, and changes in underlying trends can all influence future values. Thus, forecasts should be interpreted with caution and considered alongside other planning tools and domain expertise.

The auto_arima models selected through an exhaustive search offer a systematic approach to forecasting in time series analysis. The forecasts produced are expected to be instrumental for stakeholders in both the healthcare and aviation sectors, providing a data-driven basis for strategic decision-making. As we move forward, continuous refinement of these models and incorporation of new data will further enhance their predictive capabilities.

7 CONCLUSION

This project presented a comprehensive time series analysis of two significantly different datasets: the monthly sales of anti-diabetic drugs in Australia and the monthly totals of international airline passengers. The rigorous analytical process, underpinned by ARIMA and SARIMA modeling, illuminated the complex temporal dynamics at play within each dataset.

7.1 Key Findings

Our investigation concluded that while ARIMA models served as a solid baseline, capturing some of the intrinsic patterns within the datasets, SARIMA models outperformed their non-seasonal counterparts by adeptly integrating seasonality into their structure. Notably, the SARIMA((1, 0, 0), (1, 1, 1, 12)) configuration emerged as the best fitting model for both datasets. This outcome underscores the critical role of seasonal components in the modeling process, aligning with the pronounced seasonal patterns observed in the initial data analysis.

The auto_arima function provided an efficient and effective means of arriving at an optimal model, especially useful when dealing with complex datasets that exhibit both non-stationary and seasonal characteristics. It led to the selection of distinct models for each dataset, highlighting the unique nature of the time series under examination.

7.2 Implications

The insights gleaned from this study carry significant implications for strategic planning and decision-making within the healthcare and aviation industries. By understanding the historical trends and seasonal fluctuations, stakeholders can better anticipate future demands, optimize resource allocation, and strategize to meet upcoming challenges.

7.3 Limitations and Areas for Future Research

Despite its robust methodology and insightful outcomes, this study is not without limitations. One such limitation is the reliance on historical data which may not always capture future shifts in trends or emergent patterns, especially in the face of unprecedented events or changes in policy and consumer behavior. Additionally, the analysis did not explore external variables that could potentially influence the time series data, such as economic factors, policy changes, or demographic shifts.

Future research could address these limitations by incorporating multivariate time series models that account for exogenous variables, offering a more holistic view of the factors influencing the datasets. Moreover, expanding the scope to include data post-2008 could provide a more contemporary perspective on the trends, especially in light of recent global events that have had substantial impacts on both the healthcare and aviation sectors. Machine learning models and hybrid approaches combining traditional time series analysis with cutting-edge predictive algorithms could also be explored to enhance forecasting accuracy and robustness.

APPENDIX

A SOURCE CODE

A.1 Drug Sales Dataset

Listing 1: Python code: Anti-Diabetic Drug Sales Dataset

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.arima.model import ARIMA
import pmdarima as pm
import os

# Resolve path of the current file
file_path = os.path.abspath(__file__)

# Extract the directory in which the file is located
file_dir = os.path.dirname(file_path)

# The directory where you want to save the plot
save_dir = os.path.join(file_dir, "image")
if not os.path.exists(save_dir):
    os.makedirs(save_dir)

# Load the dataset
data = pd.read_csv(
    os.path.join(os.path.dirname(file_path), "drug.txt"),
    parse_dates=["date"],
    index_col="date",
)

# Setting the frequency explicitly
data.index.freq = "MS"

# Initial visual inspection of the series
data.plot(figsize=(12, 6))
plt.title("Monthly␣Anti-Diabetic␣Drug␣Sales␣in␣Australia")
plt.xlabel("Date")
plt.ylabel("Sales")
plt.savefig(os.path.join(save_dir, "drug_sales.pdf"), format="pdf")  # Save plot
# plt.show()

# Perform Augmented Dickey-Fuller test for stationarity
adf_test = adfuller(data["value"])
print(f"ADF␣Statistic:␣{adf_test[0]}")
print(f"p-value:␣{adf_test[1]}")

# ACF and PACF plots for the original series
fig, ax = plt.subplots(2, 1, figsize=(12, 8))
plot_acf(data["value"], ax=ax[0], lags=40)
plot_pacf(data["value"], ax=ax[1], lags=40, method="ywm")
fig.suptitle("ACF␣and␣PACF␣plots␣for␣original␣Drug␣Sales␣data", fontsize=16)
fig.tight_layout(rect=[0, 0.03, 1, 0.95])
```

```
plt.savefig(os.path.join(save_dir, "ACF_PACF_original_drug_sales.pdf"), format="pdf")  #
    Save plot
# plt.show()

# Determine if transformation is necessary (based on visual inspection)
data["value_log"] = np.log(data["value"])
data["value_log_diff"] = data["value_log"].diff().dropna()  # Differencing
data.dropna(inplace=True)

# Plot transformed and differenced data
fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(12, 8))
data["value_log"].plot(ax=axes[0], title="Log-transformed Data")
axes[0].set_ylabel("Log(Sales)")
axes[0].set_xlabel("Date")

data["value_log_diff"].plot(ax=axes[1], title="Differenced Log-transformed Data")
axes[1].set_ylabel("Differenced Log(Sales)")
axes[1].set_xlabel("Date")

plt.tight_layout()  # Adjust layout to make room for the titles
plt.savefig(os.path.join(save_dir, "log_transformed_diff_drug_sales.pdf"), format="pdf")  #
    Save plot
# plt.show()

# Re-check stationarity with ADF on transformed, differenced data
adf_test_log = adfuller(data["value_log"].dropna())
print(f"ADF Statistic (log): {adf_test_log[0]}")
print(f"p-value (log): {adf_test_log[1]}")
adf_test_log_diff = adfuller(data["value_log_diff"].dropna())
print(f"ADF Statistic (log diff): {adf_test_log_diff[0]}")
print(f"p-value (log diff): {adf_test_log_diff[1]}")

# ACF and PACF plots for log-transformed series
fig, ax = plt.subplots(2, 1, figsize=(12, 8))
plot_acf(data["value_log"].dropna(), ax=ax[0], lags=40)
plot_pacf(data["value_log"].dropna(), ax=ax[1], lags=40, method="ywm")
fig.suptitle("ACF and PACF plots for log-transformed Drug Sales data", fontsize=16)
plt.savefig(
    os.path.join(save_dir, "ACF_PACF_log_transformed_drug_sales.pdf"), format="pdf"
)  # Save plot
# plt.show()

# ACF and PACF plots for differenced, log-transformed series
fig, ax = plt.subplots(2, 1, figsize=(12, 8))
plot_acf(data["value_log_diff"].dropna(), ax=ax[0], lags=40)
plot_pacf(data["value_log_diff"].dropna(), ax=ax[1], lags=40, method="ywm")
fig.suptitle("ACF and PACF plots for differenced, log-transformed Drug Sales data",
    fontsize=16)
plt.savefig(
    os.path.join(save_dir, "ACF_PACF_diff_log_transformed_drug_sales.pdf"), format="pdf"
)  # Save plot
# plt.show()

# Define ARIMA and SARIMA model configurations
arima_configs = [
    (1, 1, 0),  # AR(1) model with differencing, suggested by PACF cut-off at lag 1
    (0, 1, 1),  # MA(1) model with differencing, suggested by ACF cut-off at lag 1
```

```
    (0, 0, 1),  # MA(1) model without differencing, for potential mild non-stationarity
    (1, 1, 1),  # ARMA(1,1) model with differencing, general model incorporating both AR and
        MA
    (1, 0, 0),  # Simple AR(1) model, capturing potential autoregressive behavior without
        differencing
    # Add new ARIMA configs here
]

sarima_configs = [
    (
        (1, 1, 0),
        (1, 1, 1, 12),
    ),  # Incorporates both non-seasonal and seasonal components
    (
        (0, 1, 1),
        (1, 1, 0, 12),
    ),  # Seasonal MA component, assumes non-seasonal MA process with seasonal differencing
    (
        (1, 1, 1),
        (0, 1, 1, 12),
    ),  # Non-seasonal ARMA(1,1) with seasonal MA(1), for complex seasonality
    (
        (1, 0, 0),
        (0, 1, 1, 12),
    ),  # Non-seasonal AR(1) model with seasonal MA(1), addresses seasonal autocorrelations
    (
        (1, 0, 0),
        (1, 1, 1, 12),
    ),  # AR(1) with both non-seasonal and seasonal differencing and MA(1), for clear
        seasonal patterns
    # Add new SARIMA configs here
]

# Fit models manually and collect AIC values
results = []
for config in arima_configs:
    try:
        model = ARIMA(data["value_log_diff"], order=config)  # Use log diff series
        model_fit = model.fit()
        results.append(("ARIMA", config, model_fit.aic, model_fit))
        # Diagnostic plot for the fitted model
        model_fit.plot_diagnostics(figsize=(12, 8))
        plt.suptitle(f'Diagnostic␣Plot␣for␣ARIMA{config}␣Model', fontsize=16)
        plt.savefig(os.path.join(save_dir, f"diagnostic_plot_ARIMA{config}_drug_sales.pdf"),
            format="pdf")  # Save plot
        # plt.show()
    except Exception as e:
        print(f"Error␣fitting␣ARIMA{config}:␣{e}")

for config in sarima_configs:
    try:
        model = SARIMAX(
            data["value_log_diff"], order=config[0], seasonal_order=config[1]
        )  # Use log diff series
        model_fit = model.fit()
        results.append(("SARIMA", config, model_fit.aic, model_fit))
        # Diagnostic plot for the fitted model
```

```
        model_fit.plot_diagnostics(figsize=(12, 8))
        plt.suptitle(f'Diagnostic Plot for SARIMA{config} Model', fontsize=16)
        plt.savefig(os.path.join(save_dir,
            f"diagnostic_plot_SARIMA{config}_drug_sales.pdf"), format="pdf")  # Save plot
        # plt.show()
    except Exception as e:
        print(f"Error fitting SARIMA{config}: {e}")

############ do a diagnostic plot for each model

# Automatic ARIMA model selection using auto_arima on transformed, differenced data
auto_arima_model = pm.auto_arima(
    data["value_log_diff"],
    seasonal=True,
    m=12,
    trace=False,
    error_action="ignore",
    suppress_warnings=True,
    stepwise=True,
)

# Include auto_arima model in the comparison
results.append(
    (
        "auto_arima",
        (auto_arima_model.order, auto_arima_model.seasonal_order),
        auto_arima_model.aic(),
        auto_arima_model,
    )
)

# Select the best model based on AIC
results_df = pd.DataFrame(
    results, columns=["Model Type", "Configuration", "AIC", "Model Object"]
)
best_model_details = results_df.sort_values(by="AIC").iloc[0]

# Save tabulated results to CSV
results_df.to_csv(os.path.join(file_dir, "model_selection_results.csv"), index=False)

# Display best model information
print(
    f"Best Model: {best_model_details['Model Type']} - Configuration:
        {best_model_details['Configuration']} - AIC: {best_model_details['AIC']}"
)

# Forecast with the best model
forecast_steps = 36  # For 36 months ahead
last_date = data.index[-1]
forecast_index = pd.date_range(
    start=last_date + pd.Timedelta(days=1), periods=forecast_steps, freq="M"
)

# Forecasting
if best_model_details["Model Type"] == "auto_arima":
    # For auto_arima, directly get the forecast and confidence intervals
    forecast, conf_int = best_model_details["Model Object"].predict(
```

```
        n_periods=forecast_steps, return_conf_int=True
    )
    # Adjust the forecast and confidence intervals back to the original scale if necessary
    forecast = np.exp(forecast + data["value_log"].iloc[-1])
    conf_int_lower = np.exp(conf_int[:, 0] + data["value_log"].iloc[-1])
    conf_int_upper = np.exp(conf_int[:, 1] + data["value_log"].iloc[-1])
else:
    # For manual ARIMA/SARIMA models, use get_forecast and adjust the forecast back to the
        original scale
    forecast_res = best_model_details["Model Object"].get_forecast(steps=forecast_steps)
    forecast = forecast_res.predicted_mean
    conf_int = forecast_res.conf_int()
    # Assuming forecast is on the differenced and logged scale, adjust back to original scale
    forecast = np.exp(forecast.cumsum() + data["value_log"].iloc[-1])
    conf_int_lower = np.exp(conf_int[:, 0].cumsum() + data["value_log"].iloc[-1])
    conf_int_upper = np.exp(conf_int[:, 1].cumsum() + data["value_log"].iloc[-1])

# Plot the forecast and the original data
plt.figure(figsize=(12, 6))
plt.plot(data.index, data["value"], label="Observed", marker="o")
plt.plot(forecast_index, forecast, label="Forecast", marker="x")
plt.fill_between(
    forecast_index, conf_int_lower, conf_int_upper, color="grey", alpha=0.2
)
plt.title("Forecast of Monthly Anti-Diabetic Drug Sales")
plt.xlabel("Date")
plt.ylabel("Sales")
plt.legend()
plt.savefig(os.path.join(save_dir, "forecast_drug_sales.pdf"), format="pdf")  # Save plot
# plt.show()
```

Listing 2: Python code: International Airline Passengers Dataset

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.arima.model import ARIMA
import pmdarima as pm
import os

# Resolve path of the current file
file_path = os.path.abspath(__file__)

# Extract the directory in which the file is located
file_dir = os.path.dirname(file_path)

# The directory where you want to save the plot
save_dir = os.path.join(file_dir, "image")
if not os.path.exists(save_dir):
    os.makedirs(save_dir)

# Load the dataset
data = pd.read_csv(
    os.path.join(os.path.dirname(file_path), "airpassenger.dat"),
```

```
    header=None,
    names=["Passengers"],
)

# Assuming the data starts from January 1949 and is monthly, create a DateTime index
data.index = pd.date_range(start="1949-01", periods=len(data), freq="M")
data["Passengers"] = pd.to_numeric(data["Passengers"], errors="coerce")
data = data.dropna(subset=["Passengers"])

# Initial visual inspection of the series
data.plot(figsize=(12, 6))
plt.title("International Airline Passengers")
plt.xlabel("Date")
plt.ylabel("Passengers (in thousands)")
plt.savefig(os.path.join(save_dir, "airline_passengers.pdf"), format="pdf")  # Save plot
# plt.show()

# Perform Augmented Dickey-Fuller test for stationarity
adf_test = adfuller(data["Passengers"])
print(f"ADF Statistic: {adf_test[0]}")
print(f"p-value: {adf_test[1]}")

# ACF and PACF plots for the original series
fig, ax = plt.subplots(2, 1, figsize=(12, 8))
plot_acf(data["Passengers"], ax=ax[0], lags=40)
plot_pacf(data["Passengers"], ax=ax[1], lags=40, method="ywm")
fig.suptitle("ACF and PACF plots for original International Airline Passengers data",
    fontsize=16)
fig.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.savefig(
    os.path.join(save_dir, "ACF_PACF_original_airline_passengers.pdf"), format="pdf"
)  # Save plot
# plt.show()

# Determine if transformation is necessary (based on visual inspection)
data["Passengers_log"] = np.log(data["Passengers"])
data["Passengers_log_diff"] = data["Passengers_log"].diff().dropna()  # Differencing
data.dropna(inplace=True)

# Plot transformed and differenced data
fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(12, 8))
data["Passengers_log"].plot(ax=axes[0], title="Log-transformed Data")
axes[0].set_ylabel("Log(Passengers)")
axes[0].set_xlabel("Date")

data["Passengers_log_diff"].plot(ax=axes[1], title="Differenced Log-transformed Data")
axes[1].set_ylabel("Differenced Log(Passengers)")
axes[1].set_xlabel("Date")

plt.tight_layout()  # Adjust layout to make room for the titles
plt.savefig(os.path.join(save_dir, "log_transformed_diff_airline_passengers.pdf"),
    format="pdf")  # Save plot
# plt.show()

# Re-check stationarity with ADF on transformed, differenced data
adf_test_log = adfuller(data["Passengers_log"].dropna())
print(f"ADF Statistic (log): {adf_test_log[0]}")
```

```
print(f"p-value (log): {adf_test_log[1]}")
adf_test_log_diff = adfuller(data["Passengers_log_diff"].dropna())
print(f"ADF Statistic (log diff): {adf_test_log_diff[0]}")
print(f"p-value (log diff): {adf_test_log_diff[1]}")

# ACF and PACF plots for log-transformed series
fig, ax = plt.subplots(2, 1, figsize=(12, 8))
plot_acf(data["Passengers_log"].dropna(), ax=ax[0], lags=40)
plot_pacf(data["Passengers_log"].dropna(), ax=ax[1], lags=40, method="ywm")
fig.suptitle("ACF and PACF plots for log-transformed International Airline Passengers data",
    fontsize=16)
fig.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.savefig(
    os.path.join(save_dir, "ACF_PACF_log_transformed_airline_passengers.pdf"), format="pdf"
)  # Save plot
# plt.show()

# ACF and PACF plots for differenced, log-transformed series
fig, ax = plt.subplots(2, 1, figsize=(12, 8))
plot_acf(data["Passengers_log_diff"].dropna(), ax=ax[0], lags=40)
plot_pacf(data["Passengers_log_diff"].dropna(), ax=ax[1], lags=40, method="ywm")
fig.suptitle("ACF and PACF plots for differenced, log-transformed International Airline
    Passengers data", fontsize=16)
fig.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.savefig(
    os.path.join(save_dir, "ACF_PACF_diff_log_transformed_airline_passengers.pdf"),
        format="pdf"
)  # Save plot
# plt.show()

# Define ARIMA and SARIMA model configurations
arima_configs = [
    (1, 1, 0),  # AR(1) model with differencing, suggested by PACF cut-off at lag 1
    (0, 1, 1),  # MA(1) model with differencing, suggested by ACF cut-off at lag 1
    (0, 0, 1),  # MA(1) model without differencing, for potential mild non-stationarity
    (1, 1, 1),  # ARMA(1,1) model with differencing, general model incorporating both AR and
        MA
    (1, 0, 0),  # Simple AR(1) model, capturing potential autoregressive behavior without
        differencing
    # Add new ARIMA configs here
]

sarima_configs = [
    (
        (1, 1, 0),
        (1, 1, 1, 12),
    ),  # Incorporates both non-seasonal and seasonal components
    (
        (0, 1, 1),
        (1, 1, 0, 12),
    ),  # Seasonal MA component, assumes non-seasonal MA process with seasonal differencing
    (
        (1, 1, 1),
        (0, 1, 1, 12),
    ),  # Non-seasonal ARMA(1,1) with seasonal MA(1), for complex seasonality
    (
        (1, 0, 0),
```

```
        (0, 1, 1, 12),
    ),  # Non-seasonal AR(1) model with seasonal MA(1), addresses seasonal autocorrelations
    (
        (1, 0, 0),
        (1, 1, 1, 12),
    ),  # AR(1) with both non-seasonal and seasonal differencing and MA(1), for clear
        seasonal patterns
    # Add new SARIMA configs here
]

# Fit models manually and collect AIC values
results = []
for config in arima_configs:
    try:
        model = ARIMA(data["Passengers_log_diff"], order=config)
        model_fit = model.fit()
        results.append(("ARIMA", config, model_fit.aic, model_fit))
        # Diagnostic plot for the fitted model
        model_fit.plot_diagnostics(figsize=(12, 8))
        plt.suptitle(f'Diagnostic Plot for ARIMA{config} Model', fontsize=16)
        plt.savefig(os.path.join(save_dir,
            f"diagnostic_plot_ARIMA{config}_airline_passengers.pdf"), format="pdf")  # Save
            plot
        # plt.show()
    except Exception as e:
        print(f"Error fitting ARIMA{config}: {e}")

for config in sarima_configs:
    try:
        model = SARIMAX(
            data["Passengers_log_diff"], order=config[0], seasonal_order=config[1]
        )  # Use log diff series
        model_fit = model.fit()
        results.append(("SARIMA", config, model_fit.aic, model_fit))
        # Diagnostic plot for the fitted model
        model_fit.plot_diagnostics(figsize=(12, 8))
        plt.suptitle(f'Diagnostic Plot for SARIMA{config} Model', fontsize=16)
        plt.savefig(os.path.join(save_dir,
            f"diagnostic_plot_SARIMA{config}_airline_passengers.pdf"), format="pdf")  # Save
            plot
        # plt.show()
    except Exception as e:
        print(f"Error fitting SARIMA{config}: {e}")

# Automatic ARIMA model selection using auto_arima on transformed, differenced data
auto_arima_model = pm.auto_arima(
    data["Passengers_log_diff"],
    seasonal=True,
    m=12,
    trace=False,
    error_action="ignore",
    suppress_warnings=True,
    stepwise=True,
)

# Include auto_arima model in the comparison
results.append(
```

```
    (
        "auto_arima",
        (auto_arima_model.order, auto_arima_model.seasonal_order),
        auto_arima_model.aic(),
        auto_arima_model,
    )
)

# Select the best model based on AIC
results_df = pd.DataFrame(
    results, columns=["Model Type", "Configuration", "AIC", "Model Object"]
)
best_model_details = results_df.sort_values(by="AIC").iloc[0]

# Save tabulated results to CSV
results_df.to_csv(os.path.join(file_dir, "model_selection_results.csv"), index=False)

# Display best model information
print(
    f"Best Model: {best_model_details['Model Type']} - Configuration:
        {best_model_details['Configuration']} - AIC: {best_model_details['AIC']}"
)

# Forecast with the best model
forecast_steps = 36  # For 36 months ahead
last_date = data.index[-1]
forecast_index = pd.date_range(
    start=last_date + pd.Timedelta(days=1), periods=forecast_steps, freq="M"
)

# Forecasting
if best_model_details["Model Type"] == "auto_arima":
    # For auto_arima, directly get the forecast and confidence intervals
    forecast, conf_int = best_model_details["Model Object"].predict(
        n_periods=forecast_steps, return_conf_int=True
    )
    # Adjust the forecast and confidence intervals back to the original scale if necessary
    forecast = np.exp(forecast + data["Passengers_log"].iloc[-1])
    conf_int_lower = np.exp(conf_int[:, 0] + data["Passengers_log"].iloc[-1])
    conf_int_upper = np.exp(conf_int[:, 1] + data["Passengers_log"].iloc[-1])
else:
    # For manual ARIMA/SARIMA models, use get_forecast and adjust the forecast back to the
        original scale
    forecast_res = best_model_details["Model Object"].get_forecast(steps=forecast_steps)
    forecast = forecast_res.predicted_mean
    conf_int = forecast_res.conf_int()
    # Assuming forecast is on the differenced and logged scale, adjust back to original scale
    forecast = np.exp(forecast.cumsum() + data["Passengers_log"].iloc[-1])
    conf_int_lower = np.exp(conf_int[:, 0].cumsum() + data["Passengers_log"].iloc[-1])
    conf_int_upper = np.exp(conf_int[:, 1].cumsum() + data["Passengers_log"].iloc[-1])

# Plot the forecast and the original data
plt.figure(figsize=(12, 6))
plt.plot(data.index, data["Passengers"], label="Observed", marker="o")
plt.plot(forecast_index, forecast, label="Forecast", marker="x")
plt.fill_between(
    forecast_index, conf_int_lower, conf_int_upper, color="grey", alpha=0.2
```

```
)
plt.title("Forecast of International Airline Passengers")
plt.xlabel("Date")
plt.ylabel("Passengers (in thousands)")
plt.legend()
plt.savefig(
    os.path.join(save_dir, "forecast_airline_passengers.pdf"), format="pdf"
)  # Save plot
# plt.show()
```

B DIAGNOSTICS PLOTS

This appendix section contains diagnostic plots for all ARIMA and SARIMA models that were explored in the project.

B.1 Anti-Diabetic Drug Sales Dataset

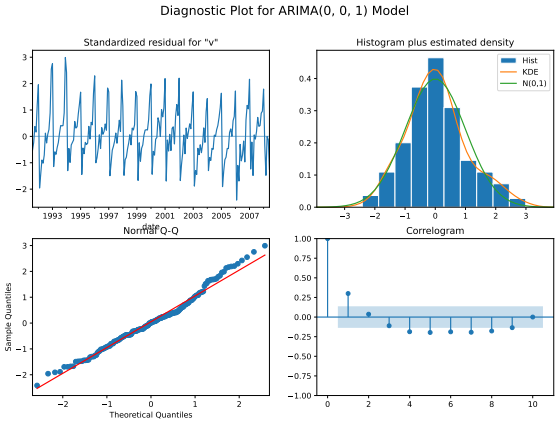

Figure 11: Diagnostic Plot for ARIMA(0, 0, 1)

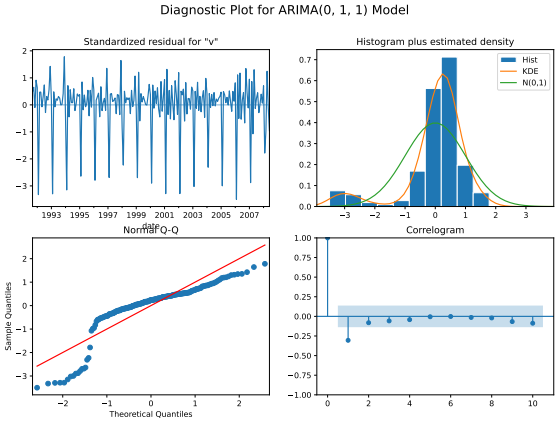

Figure 12: Diagnostic Plot for ARIMA(0, 1, 1)

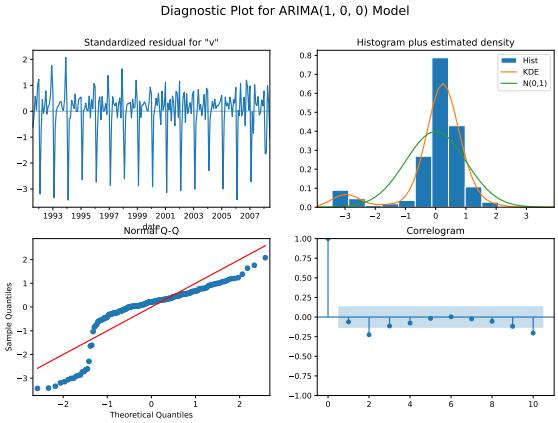

Figure 13: Diagnostic Plot for ARIMA(1, 0, 0)

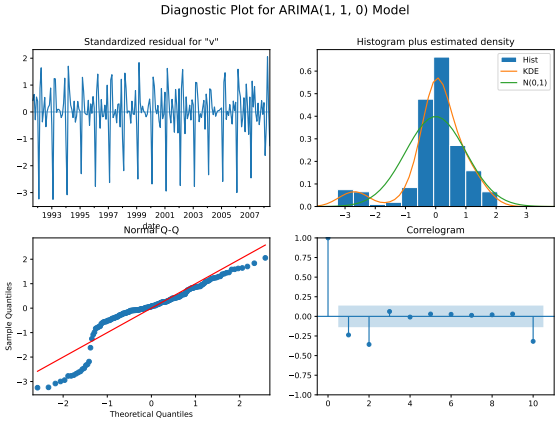

Figure 14: Diagnostic Plot for ARIMA(1, 1, 0)

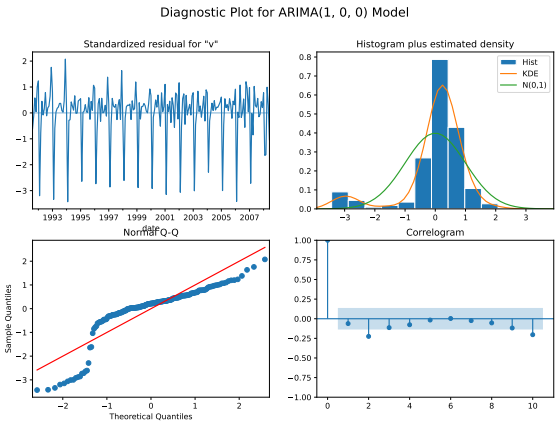

Figure 15: Diagnostic Plot for ARIMA(1, 1, 1)

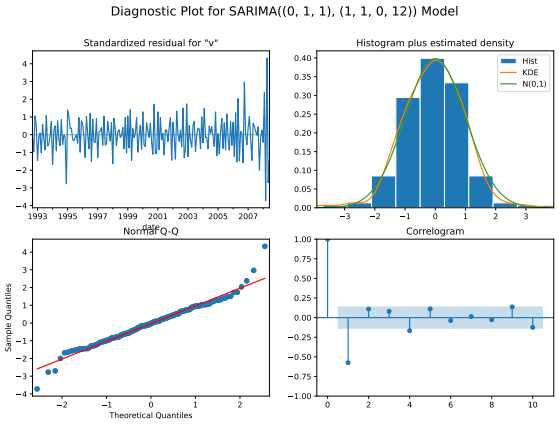

Figure 16: Diagnostic Plot for SARIMA(0, 1, 1), (1, 1, 0, 12)

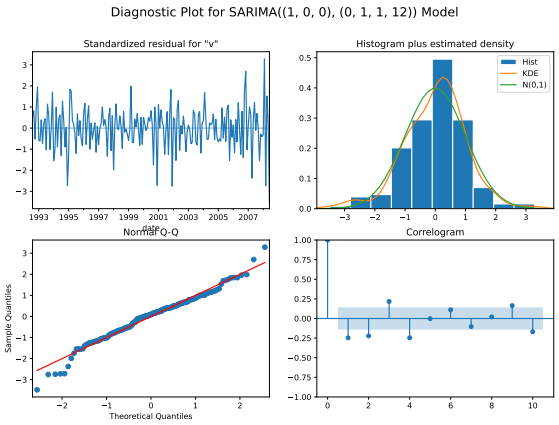

Figure 17: Diagnostic Plot for SARIMA(1, 0, 0), (0, 1, 1, 12)

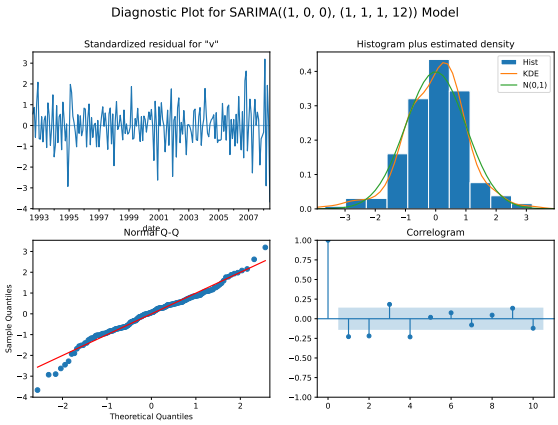

Figure 18: Diagnostic Plot for SARIMA(1, 0, 0), (1, 1, 1, 12)

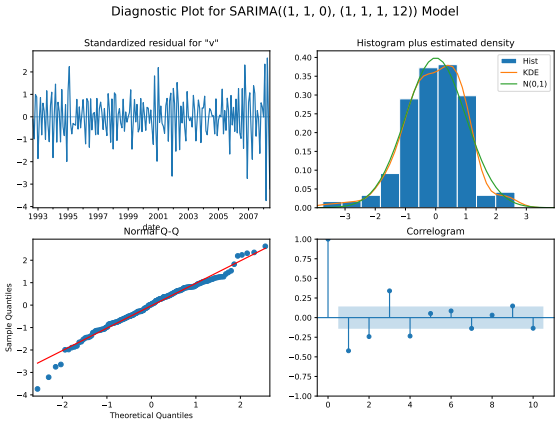

Figure 19: Diagnostic Plot for SARIMA(1, 1, 0), (1, 1, 1, 12)

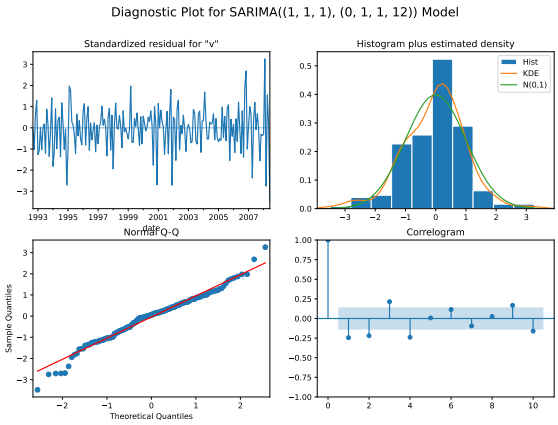

Figure 20: Diagnostic Plot for SARIMA(1, 1, 1), (0, 1, 1, 12)

B.2 International Airline Passengers Dataset

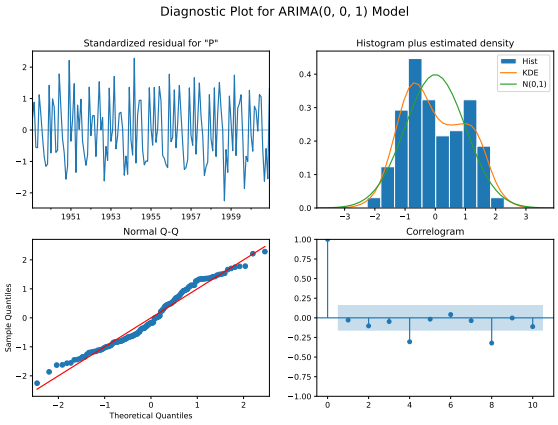

Figure 21: Diagnostic Plot for ARIMA(0, 0, 1)

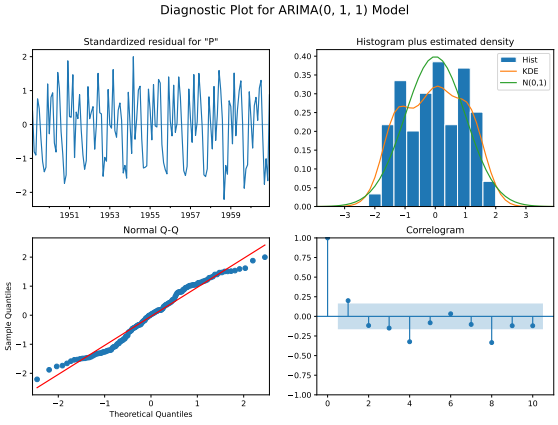

Figure 22: Diagnostic Plot for ARIMA(0, 1, 1)

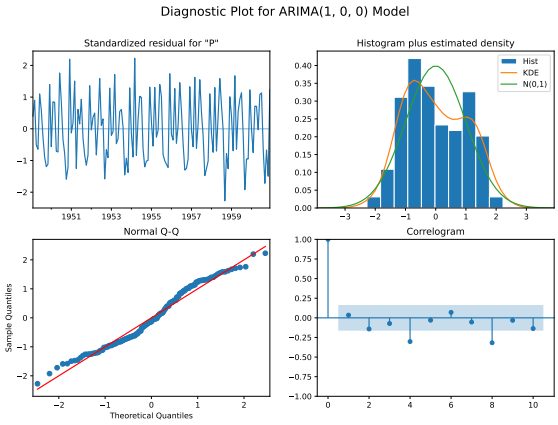

Figure 23: Diagnostic Plot for ARIMA(1, 0, 0)

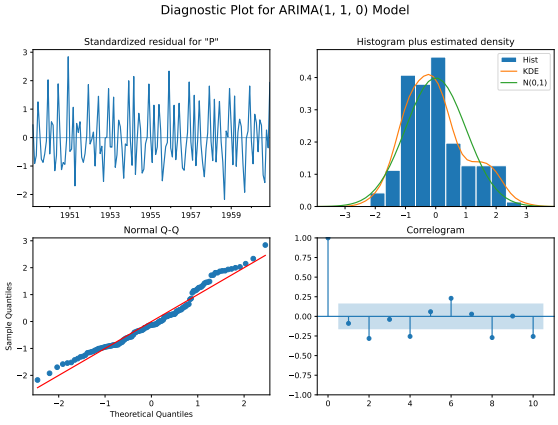

Figure 24: Diagnostic Plot for ARIMA(1, 1, 0)

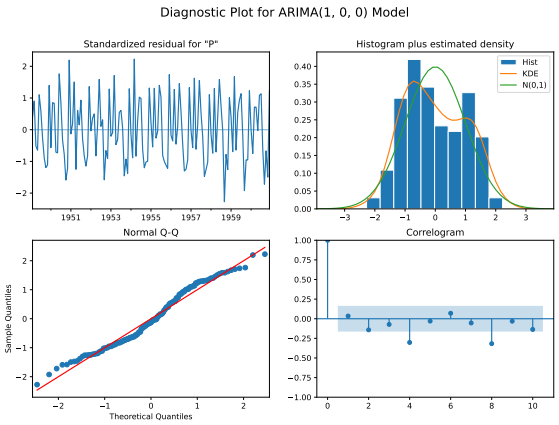

Figure 25: Diagnostic Plot for ARIMA(1, 1, 1)

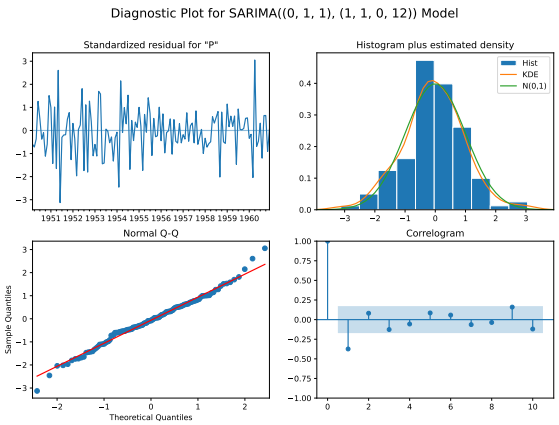

Figure 26: Diagnostic Plot for SARIMA(0, 1, 1), (1, 1, 0, 12)

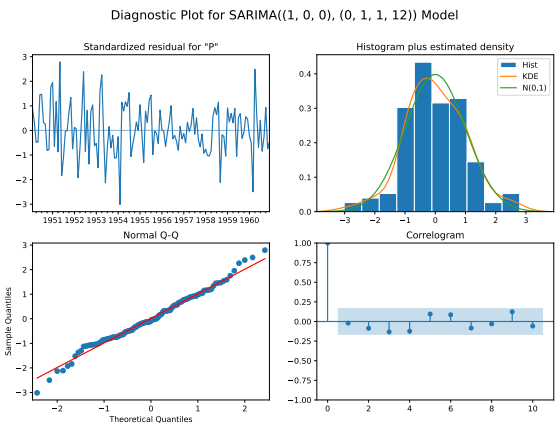

Figure 27: Diagnostic Plot for SARIMA(1, 0, 0), (0, 1, 1, 12)

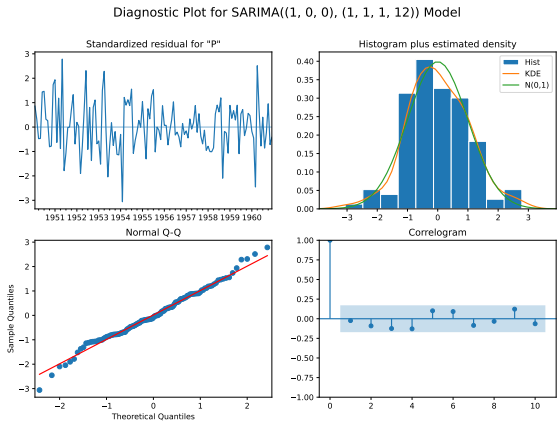

Figure 28: Diagnostic Plot for SARIMA(1, 0, 0), (1, 1, 1, 12)

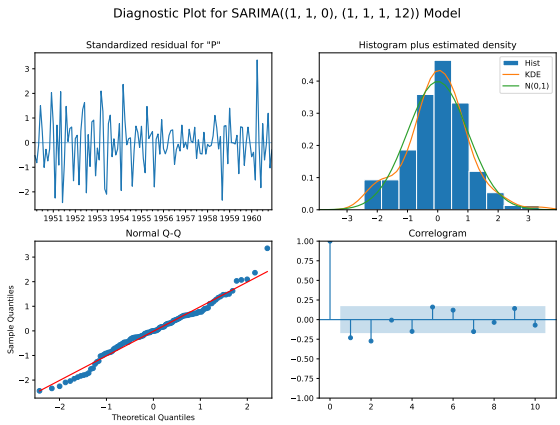

Figure 29: Diagnostic Plot for SARIMA(1, 1, 0), (1, 1, 1, 12)

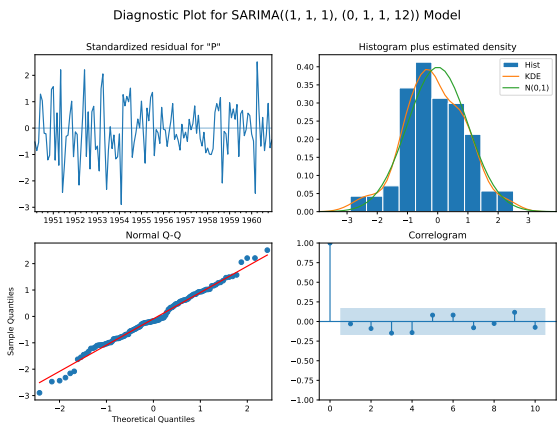

Figure 30: Diagnostic Plot for SARIMA(1, 1, 1), (0, 1, 1, 12)