CHICAGO CLIMATE FORECASTING

Shwedha Srinivasan
20514543

Priyadarshini Rajendran
A20476470

Kyung Jin Kwak
A20497336

Submitted on 26 April 2024

Project Team 5
Supervisor
Prof. Despina Stasi

Illinois Institute of Technology
Chicago, 60616

Abstract

This report presents a comprehensive approach to developing a predictive analytics framework for modeling Chicago's weather patterns, with a focus on temperature variations exacerbated by global warming. Leveraging a rich dataset spanning three years of historical weather data, augmented by real-time streaming data, our methodology integrates advanced time-series analytical techniques such as ARIMA, SARIMA, and machine learning algorithms. The primary objective is to deliver highly accurate weather predictions to aid urban planners, public health officials, and residents in proactively addressing the impacts of increasingly erratic weather. The dataset, sourced from Visual Crossing Weather Services, covers a wide range of meteorological variables and facilitates in-depth analysis and accurate forecasting. Data cleaning, exploratory data analysis, feature selection, and stationary checks are performed to ensure the reliability and robustness of the models. Modeling efforts include ARIMA and SARIMAX, with parameters optimized to forecast weather conditions for the next 30 days. The study concludes with insights into future work, highlighting the potential for integrating additional data sources and machine learning models to further enhance predictive accuracy and robustness.

Acknowledgment

We express our sincere gratitude to all those who have contributed to the completion of this project. First and foremost, we extend our appreciation to the Visual Crossing Weather Services for providing the extensive dataset essential for our research. Their dedication to maintaining a high-quality and reliable source of meteorological data has been instrumental in our analysis and forecasting efforts.

We would also like to thank the entire team involved in the data collection, cleaning, and preparation processes. Your meticulous attention to detail and commitment to ensuring the accuracy and integrity of the dataset have laid the foundation for the success of our project.

Furthermore, we are deeply grateful to our advisor for her invaluable guidance, support, and encouragement throughout this endeavor. Your expertise and insights have been indispensable in shaping our approach and refining our methodologies.
Last but not least, we acknowledge the broader community of researchers, and practitioners, in the field of climate science and predictive analytics. Your ongoing contributions to advancing our understanding of weather patterns and their impacts on society inspire us to continue pushing the boundaries of knowledge and innovation. Together, these collective efforts have enabled us to develop a robust predictive analytics framework for modeling Chicago's weather patterns, with the ultimate aim of enhancing resilience and sustainability in the face of increasingly erratic weather conditions.

Contents

Chapter 1

Introduction

1.1 Background

This project is designed to develop a sophisticated predictive analytics framework for modelling Chicago's weather patterns, particularly focusing on the pronounced temperature variations during Winter and Summer influenced by global warming. The primary objective is to deliver highly accurate weather predictions that will aid urban planners, public health officials, and residents in proactively addressing the impacts of increasingly erratic weather, thereby enhancing community resilience against climatic adversities. This initiative will employ real-time weather streaming data from Chicago and incorporate advanced time-series forecasting methodologies, such as ARIMA, SARIMA, and machine learning algorithms, to predict weather conditions for the upcoming 30 days, ensuring a comprehensive application of time-series concepts in our analytical processes.

1.2 Problem Description

In recent years, Chicago has experienced significant shifts in its weather patterns, largely attributed to the effects of global warming. These changes have manifested in more extreme temperature fluctuations, particularly during the Winter and Summer months, posing increased challenges to public health, urban planning, and environmental stability.

1.3 Objective

The primary goal of this project is to develop a predictive model that can accurately forecast short-term weather conditions in Chicago, focusing on temperature anomalies during the critical winter and summer seasons. By leveraging advanced time-series analysis, this model aims to provide actionable insights that can assist in planning and mitigating the adverse effects of unexpected weather patterns.

Chapter 2

About the Dataset

The model will be built using three years of historical weather data for Chicago. This dataset includes daily measurements of various meteorological variables, but the focus will be on temperature due to its direct impact on urban and public health planning. Additionally, real-time weather streaming data will be utilized to continuously update and refine the forecasts.

To ensure the robustness and relevance of our predictive model for Chicago's weather, we have integrated a comprehensive dataset sourced from Visual Crossing Weather Services. This dataset encompasses a detailed collection of meteorological data spanning the past four years, providing a rich basis for in-depth analysis and accurate weather forecasting.

2.1 Dataset Overview

Source:

- Visual Crossing Weather Services

Scope:

- Weather data specifically for Chicago, United States

Duration:

- Last four years

Metrics:

- This dataset includes a wide range of meteorological measurements such as temperature, humidity, precipitation, wind speed, and barometric pressure.

Format:

- The data is systematically organized and formatted for easy integration into various analytical tools and models, facilitating efficient time-series analysis.

Link:

- Link to dataset website

2.2 Utilization of Dataset

This dataset is crucial for our project as it offers extensive historical weather data. The four-year coverage allows us to analyze seasonal trends, yearly fluctuations, and other long-term patterns that are essential for understanding and predicting the impacts of global warming on local weather dynamics.

Benefits of the Dataset:

- **Longitudinal Analysis:** The four-year span provides enough data to perform robust longitudinal studies, crucial for identifying and understanding long-term trends and variations in weather patterns.
- **Depth and Variety:** With its comprehensive set of meteorological variables, the dataset supports a multifaceted approach to weather analysis, enabling predictions across different weather phenomena.
- **High Quality and Reliability:** Sourced from a respected provider in weather data services, the dataset guarantees consistency and high fidelity, which are critical for the accuracy of predictive modeling.

2.3 Application in the Project

Incorporating this extensive dataset into our time-series analytical framework allows us to employ advanced modeling techniques, such as ARIMA and SARIMA, tailored to detect and forecast significant weather shifts over time. This capability is particularly valuable in our goal to provide Chicago's city planners, health officials, and residents with reliable forecasts that can inform preparedness and response strategies to weather-induced challenges.

Chapter 3

Exploratory Data Analysis

3.1 Data Cleaning

Data cleaning is a critical step in the preprocessing phase of any data analysis or machine learning project. Its primary purpose is to ensure that the data is accurate, consistent, and usable by resolving issues like missing values, incorrect data types, and outliers. Clean data improves the quality and accuracy of insights derived from data analysis or model predictions.

3.1.1 Specific Steps in Data Cleaning for the Provided Weather Dataset:

- **Checking Data Types:**
  - **Data types:** An overview of the dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 999 entries, 0 to 998
Data columns (total 33 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   name             999 non-null    object
 1   datetime         999 non-null    object
 2   tempmax          999 non-null    float64
 3   tempmin          999 non-null    float64
 4   temp             999 non-null    float64
 5   feelslikemax     999 non-null    float64
 6   feelslikemin     999 non-null    float64
 7   feelslike        999 non-null    float64
 8   dew              999 non-null    float64
 9   humidity         999 non-null    float64
 10  precip           999 non-null    float64
 11  precipprob       999 non-null    int64
 12  precipcover       999 non-null    float64
 13  preciptype        558 non-null    object
 14  snow              999 non-null    float64
 15  snowdepth         999 non-null    float64
 16  windgust          999 non-null    float64
 17  windspeed         999 non-null    float64
 18  winddir           999 non-null    float64
 19  sealevelpressure  999 non-null    float64
...
 31  icon              999 non-null    object
 32  stations          999 non-null    object
dtypes: float64(22), int64(2), object(9)
memory usage: 257.7+ KB
```

  - **Purpose:** Ensure that each column in the dataset is of the appropriate type for subsequent data analysis or model training. For example, date and time columns should be in `datetime` format to facilitate time-series analysis.
  - **Implementation:** The `datetime`, sunrise, and sunset columns were converted from object type to `datetime64[ns]` using `pd.to_datetime()` to enable easy extraction of date and time components and perform time-related calculations.

- **Handling Missing Values:**
  - **Purpose:** Address columns with missing values to avoid errors in analysis and ensure robust performance of the predictive model. Missing data can lead to biased or incorrect analysis results.
  - **Findings:** Columns like preciptype and severerisk have missing values, with preciptype missing in about 44% of the rows and severerisk in approximately 17%.
  - **Approach:** For preciptype, which denotes the type of precipitation, missing values might imply days with no precipitation, so these could either be filled with a category such as 'None' or imputed based on nearby data points if appropriate. For severerisk, which likely indicates the risk of severe weather conditions, missing values might be handled through imputation or by setting a default risk level based on the distribution of available data.

- **Identifying Outliers:**
  - **Purpose:** Detect and handle outliers in data to prevent skewed analysis. Outliers can significantly affect the results of statistical analysis and model predictions, particularly in regression-based models.
  - **Approach:** Use statistical methods or visualizations (like box plots) to identify outliers in temperature, humidity, wind speed, etc. Depending on the context, outliers may be removed, capped, or adjusted if they are errors or kept if they represent true extreme weather events.

- **Data Integrity Checks:**
  - **Purpose:** Verify the accuracy and consistency of data across the dataset. This includes checking for duplicate entries, ensuring consistency in measurement units, and confirming logical integrity (e.g., max temperatures should always be equal to or greater than min temperatures).
  - **Approach:** Perform checks using conditional statements and functions to identify any anomalies or inconsistencies, such as verifying that tempmax is always greater than or equal to tempmin.

3.1.2 Results Derived from Data Cleaning:

- **Improved Data Quality:** By converting data into correct formats, filling or removing missing values, and handling outliers, the data becomes cleaner and more structured, enhancing the reliability of the dataset for analysis.
- **Readiness for Analysis:** Clean data ensures that subsequent steps in data analysis, such as exploratory data analysis (EDA), feature engineering, and model training, can proceed without interruptions caused by data issues.
- **Accurate Predictions:** For predictive modeling, clean data is crucial for training models that are both accurate and robust, capable of generalizing well to new, unseen data.

By diligently cleaning the weather dataset, we ensure that our time-series analysis and predictive models for Chicago's weather conditions are based on solid, reliable data, leading to more accurate forecasts and better decision-making support.

3.2 Visualization

- **Daily maximum, minimum, and average temperatures over time**

  From this, we can observe the seasonal fluctuations, with temperatures peaking in the middle of the year and dropping during the winter months. From Fig. 3.1, we can observe the seasonal fluctuations, with temperatures peaking in the middle of the year and dropping during the winter months.

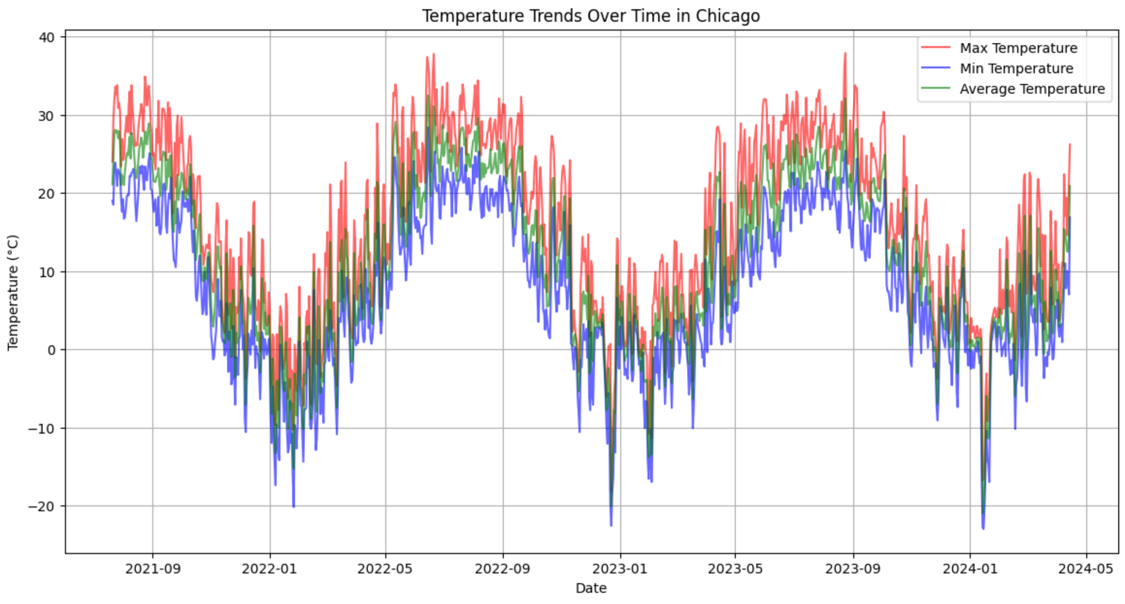


Figure 3.1: Daily temperature trends over time

- **Distribution of average temperatures, humidity, and wind speed**

  Average Temperature: The distribution is somewhat bimodal, reflecting colder temperatures during winter and warmer temperatures in summer.

  Humidity: The distribution is skewed towards higher values, indicating that high humidity levels are common in this dataset.

  Wind Speed: This distribution is more spread out, indicating variability in daily wind conditions but mostly centered around moderate wind speeds.

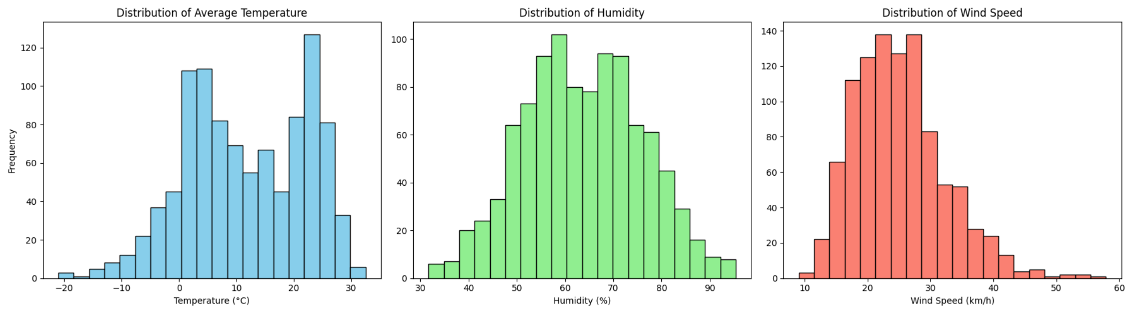


Figure 3.2: Distribution of weather variables

- **Correlation matrix of numerical variables**

  Temperature and Solar Radiation are positively correlated, which is expected as warmer days are typically sunnier.

  Precipitation and Cloud Cover also show a positive correlation, suggesting that higher cloud cover often leads to more precipitation.

  Temperature and Sea Level Pressure exhibit a slight negative correlation, indicating that lower pressures might correspond to higher temperatures, a common scenario during warmer months or storm conditions.

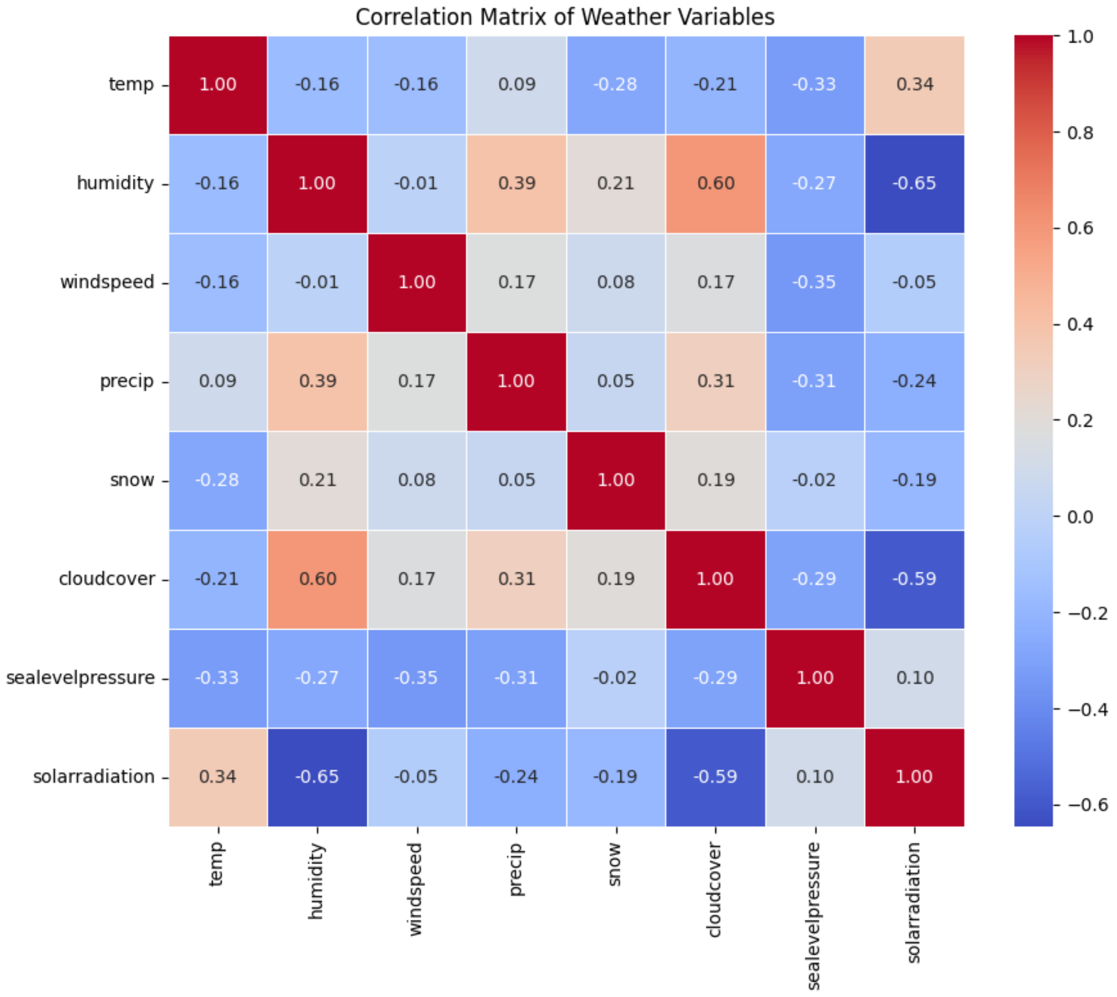


Figure 3.3: Correlation matrix of numerical variables

- **Boxplot of Temperature by Month**

  November to March have a wider range of temperatures, as indicated by larger boxes, which implies greater variability in temperatures during these months.

  Several outliers are shown in winter on the lower end, indicating extremely cold days.

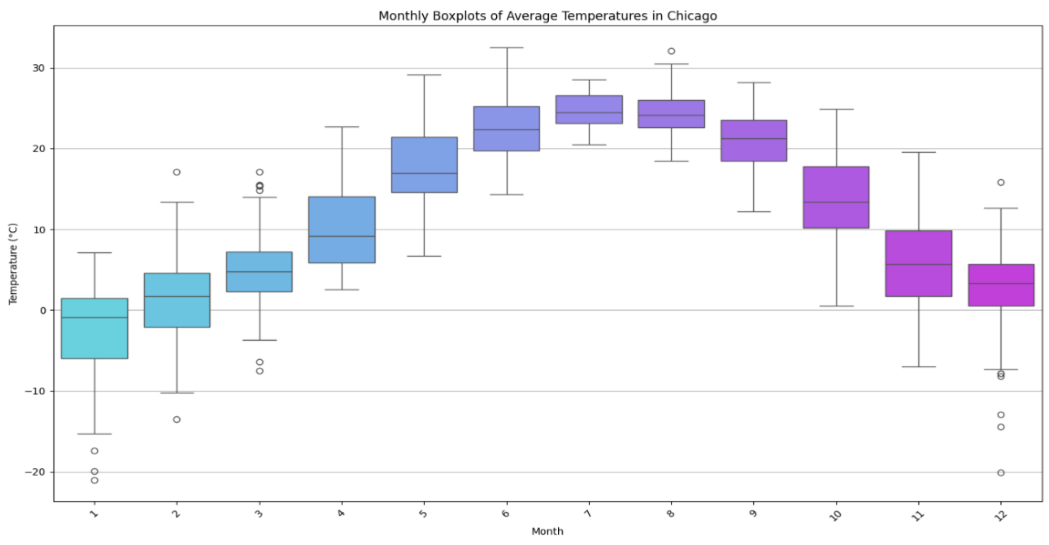


Figure 3.4: Boxplot of Temperature by Month

- **Frequency of weather conditions (Top 5)**

  It appears that cloudy and partially cloudy conditions are the most common, with clear days also occurring frequently. Less common conditions include rain and snow, which are expected given seasonal variations.

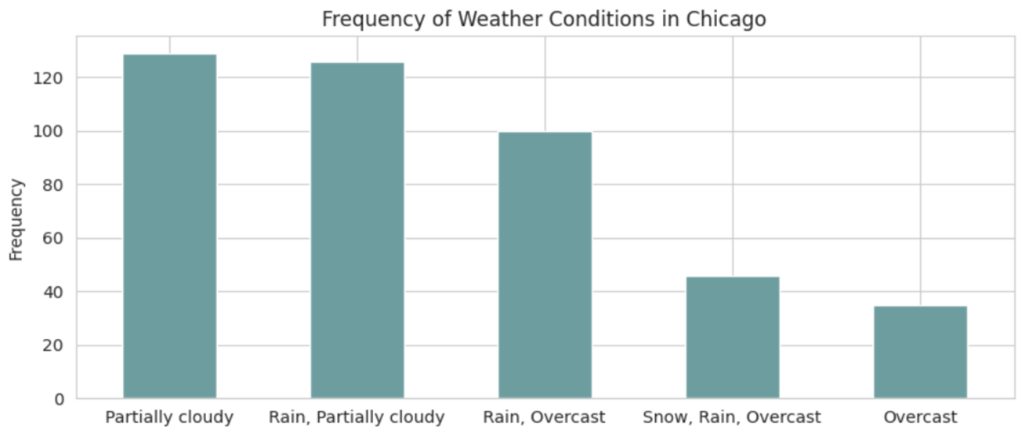


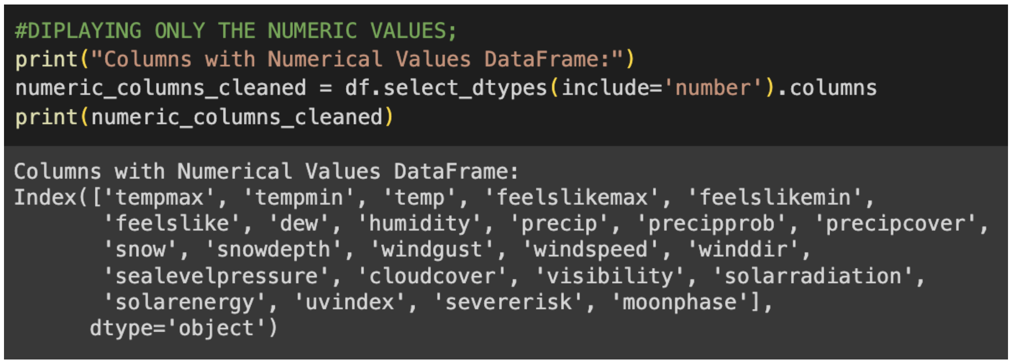
Figure 3.5: Frequency of weather conditions

Chapter 4

Feature selection and Feature engineering using Statistical techniques

4.1 Between numerical values

- There are several numerical values in the dataset it is as follows:



```
#DIPLAYING ONLY THE NUMERIC VALUES;
print("Columns with Numerical Values DataFrame:")
numeric_columns_cleaned = df.select_dtypes(include='number').columns
print(numeric_columns_cleaned)
```

```
Columns with Numerical Values DataFrame:
Index(['tempmax', 'tempmin', 'temp', 'feelslikemax', 'feelslikemin',
       'feelslike', 'dew', 'humidity', 'precip', 'precipprob', 'precipcover',
       'snow', 'snowdepth', 'windgust', 'windspeed', 'winddir',
       'sealevelpressure', 'cloudcover', 'visibility', 'solarradiation',
       'solarenergy', 'uvindex', 'severerisk', 'moonphase'],
      dtype='object')
```

Figure 4.1: Numerical Features

- There are several ways one could plot the relationship between numerical categories. On the basis of Time Series and practicality from the course. We use LAG features.

4.1.1 LAG features

- Lag features are extremely useful for capturing cycles. In a time series, as cycles, we define growths and decays in the target value that are not related to time, but mainly to previous target values. These fluctuations are not seasonal and their frequencies vary.
- To take into account such cycles we need to use the lag features. To visualize the serial dependence we can use lag plots. One of the most popular lag plots is the autocorrelation

(ACF) plot and partial autocorrelation (PACF) plot, which showcases the correlation between the target and one specific lag. In this project, for each day in the dataset, lag features are created by shifting the temperature variable backwards in time by 1 to 7 days. These lag features represent the temperature values observed in the past 1 to 7 days. You can adjust the lag window size based on your specific forecasting requirements. We obtain the following ACF and PACF plots:

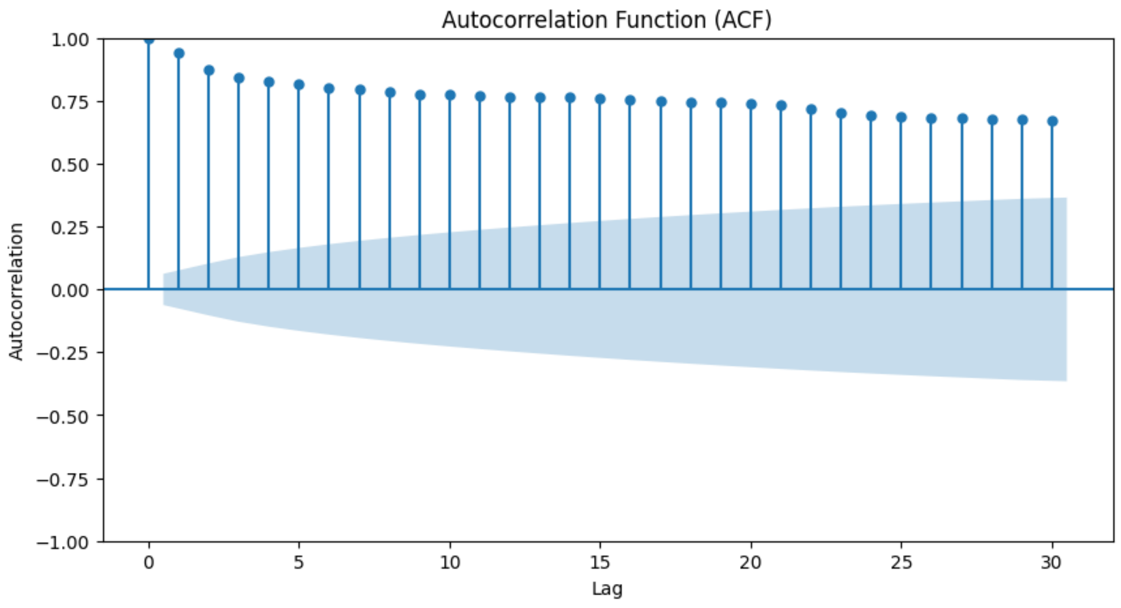


Figure 4.2: ACF plot

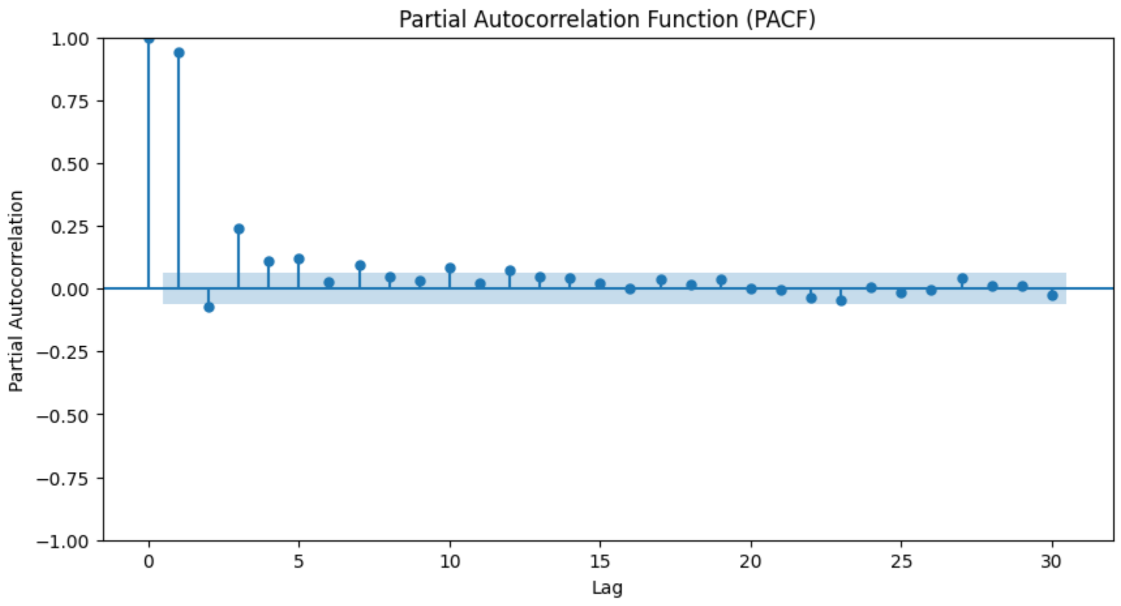


Figure 4.3: PACF plot

4.1.2 Interpretation

- Autocorrelation Function (ACF):
  - The ACF plot shows the autocorrelation of a time series with its lagged values. It helps identify significant lags by plotting the correlation coefficient against different lag values.
  - Significant lags are those where the autocorrelation coefficients are above the confidence intervals (shaded region) in the ACF plot.
- Partial Autocorrelation Function (PACF):
  - The PACF plot shows the partial autocorrelation of a time series with its lagged values, controlling for the effect of intermediate lags. Significant lags in the PACF plot are those that are above the confidence intervals and not explained by earlier lags.
  - Before performing correlation analysis to the LAG features, let's reduce the dataset with feature selection and feature engineering. For this, we used three major tests:
  - Feature selection with `SelectKbest` and `f-regression`:

```
               Feature Scores:
           Feature          Score
4        feelslike  158632.208062
0          tempmax   33599.530251
1          tempmin   28813.476473
3     feelslikemin   27614.237898
2     feelslikemax   24328.992364
5              dew    9718.404992
11       snowdepth     293.350548
20         uvindex     156.928745
18solarradiation     131.384195
19     solarenergy     131.152020
15sealevelpressure     121.091258
21      severerisk      98.038893
10            snow      82.895250
14         winddir      66.178285
17      visibility      46.620882
16      cloudcover      44.714684
6         humidity      27.706885
13       windspeed      25.405898
12        windgust       8.899362
7           precip       8.274553
9      precipcover       1.020599
8       precipprob       0.495636
22       moonphase       0.072387
```

- Collinearity tests
    - We checked for multicollinearity among the variables to avoid redundant information in the dataset. Variables that are highly correlated with each other can be removed to simplify the analysis and reduce redundancy.

```
Highly Correlated Variable Pairs:
[('feels like min', 'feels like max'), ('feelslike', 'feelslikemax'), ('feelslike', 'feelsliken
```

- Outlier detection
    - This is not for columns in fact it is for the rows, this has found 122 outliers and we have removed them.
- After we performed the analysis, we could see that the top 10 numerical features were: ['feelslike', 'tempmax', 'tempmin', 'feelslikemin', 'feelslikemax', 'dew', 'snowdepth', 'severerisk', 'uvindex', 'sealevelpressure']
- Performing the Pearson correlation for these features leads us to the following Correlation plot:

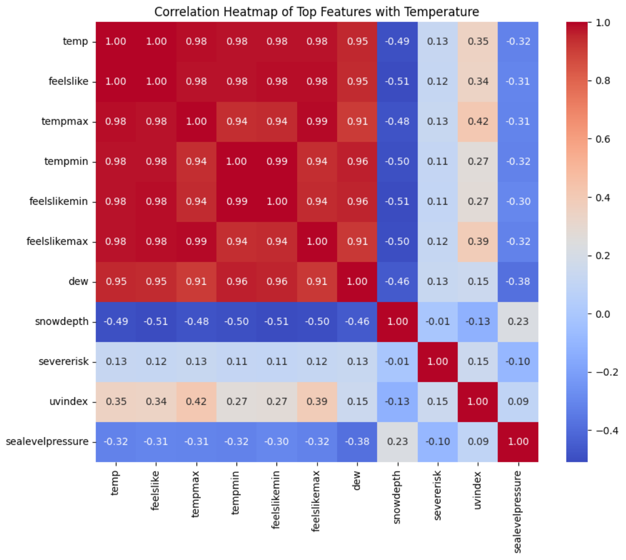


Figure 4.4: Correlation plot

4.2 Between categorical values

There are several categorical values in the dataset it is as follows:

```
            #Identifying Categorical Columns through the given data set
def cat_columns(df, threshold=10):
    categorical_columns = []
    for column in df.columns:
        if df[column].dtype == 'object' or len(df[column].unique()) <= threshold:
            categorical_columns.append(column)
    return categorical_columns

    Categorical columns:
['name',
 'precipprob',
 'preciptype',
 'severerisk',
 'conditions',
 'description',
 'icon',
 'stations']

categorical_columns = cat_columns(df)
display("Categorical columns:", categorical_columns)
```

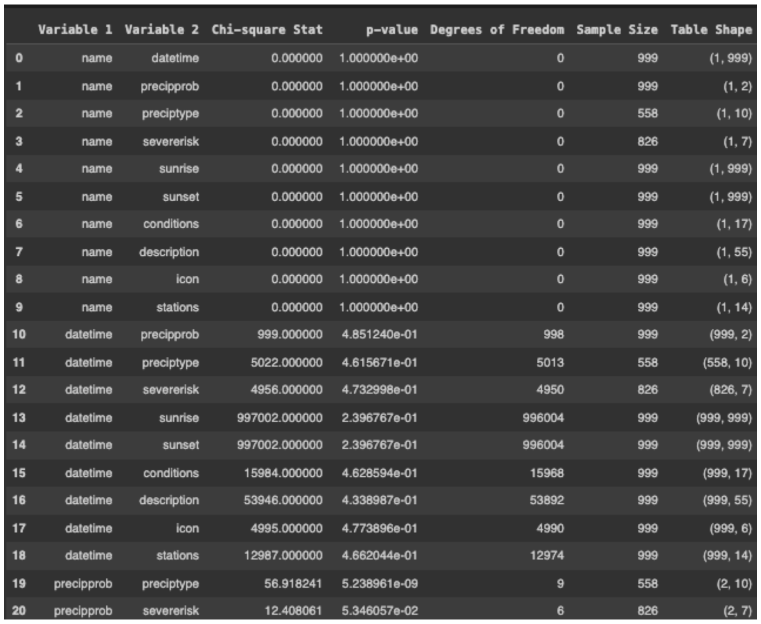
When it comes to examining the relationship between categorical variables, statistical tests like the Chi-square test are valuable. They reveal whether there's a notable connection between two categorical series. If the Chi-square indicates significance, Cramer's V comes into play, quantifying the strength of this relationship. This approach is especially practical when we want to grasp how categories interact within a dataset.

4.2.1 Chi-square test

We perform Chi-square tests to examine the relationships between all pairs of categorical columns in a data frame. We assess the statistical significance of their associations and compile the results into a data frame for straightforward analysis. The results include the Chi-square statistic, which indicates the strength of association, the p-value, which tells us if the result is statistically significant, and the degrees of freedom, which relate to the number of categories in the variables. Additionally, the sample size and the shape of the contingency table are reported, providing a complete picture of the data context. This comprehensive overview helps identify dependencies between variables, guiding further analyses and potential improvements in predictive modeling.

Result

Based on the result, we had 54 different relationships between the variables and in order to find out which two variables have a strong association we set the p-value threshold is set at 0.05 to identify the significant results. That means any association with a p-value below this threshold is considered statistically significant. The results include the Chi-square statistic, which indicates the strength of association, the p-value itself, degrees of freedom, sample size, and the shape of the contingency table. This complete picture of the data context helps us identify dependencies between variables.



	Variable 1	Variable 2	Chi-square Stat	p-value	Degrees of Freedom	Sample Size	Table Shape
0	name	datetime	0.000000	1.000000e+00	0	999	(1, 999)
1	name	precipprob	0.000000	1.000000e+00	0	999	(1, 2)
2	name	preciptype	0.000000	1.000000e+00	0	558	(1, 10)
3	name	severerisk	0.000000	1.000000e+00	0	826	(1, 7)
4	name	sunrise	0.000000	1.000000e+00	0	999	(1, 999)
5	name	sunset	0.000000	1.000000e+00	0	999	(1, 999)
6	name	conditions	0.000000	1.000000e+00	0	999	(1, 17)
7	name	description	0.000000	1.000000e+00	0	999	(1, 55)
8	name	icon	0.000000	1.000000e+00	0	999	(1, 6)
9	name	stations	0.000000	1.000000e+00	0	999	(1, 14)
10	datetime	precipprob	999.000000	4.851240e-01	998	999	(999, 2)
11	datetime	preciptype	5022.000000	4.615671e-01	5013	558	(558, 10)
12	datetime	severerisk	4956.000000	4.732998e-01	4950	826	(826, 7)
13	datetime	sunrise	997002.000000	2.396767e-01	996004	999	(999, 999)
14	datetime	sunset	997002.000000	2.396767e-01	996004	999	(999, 999)
15	datetime	conditions	15984.000000	4.628594e-01	15968	999	(999, 17)
16	datetime	description	53946.000000	4.338987e-01	53892	999	(999, 55)
17	datetime	icon	4995.000000	4.773896e-01	4990	999	(999, 6)
18	datetime	stations	12987.000000	4.662044e-01	12974	999	(999, 14)
19	precipprob	preciptype	56.918241	5.238961e-09	9	558	(2, 10)
20	precipprob	severerisk	12.408061	5.346057e-02	6	826	(2, 7)

Figure 4.5: Dependencies using Chi-squared test

Key findings from these results highlight several important relationships

- Variables like 'name' show no significant associations with others, suggesting unique identifiers with no interaction effects.
- Significant associations are observed between variables such as `precipprob` and `preciptype`, indicating strong dependencies likely driven by their inherent connection in meteorological conditions.
- High Chi-square values in time-related comparisons (e.g., `datetime` with `sunrise` and `sunset`) show variability but not necessarily meaningful associations.
- Weather-related variables such as `preciptype` and `conditions` demonstrate strong statistical links, reinforcing the logical correlation between the nature of precipitation and specific weather conditions.
- Descriptive variables like `conditions` and `description` also exhibit strong associations, pointing to a close alignment between general and detailed descriptions of weather scenarios.

4.2.2 Cramer's V

Building on our previous findings, we've applied Cramer's V to gauge the strength of the significant associations identified by the Chi-square tests. This metric has provided us with a nuanced understanding of the dataset, highlighting strong relationships, particularly between variables like 'precipprob' with 'preciptype' and 'conditions'. The calculated Cramer's V values are now an essential part of our analysis, helping us prioritize the most influential relationships for further investigation and modeling.

Classification of the variables and determining their strong, weak and negotiable associations

Taking the analysis a step further, we've classified the strengths of the associations using a predefined scale. Associations with a Cramer's V value greater than 0.7 are considered **Strong**, between 0.3 to 0.7 are deemed **Moderate**, and below 0.3 are labeled **Weak**. Notably, pairs such as `precipprob` with `conditions` and `description`, as well as `conditions` with `icon`, fall into the 'Strong' category.

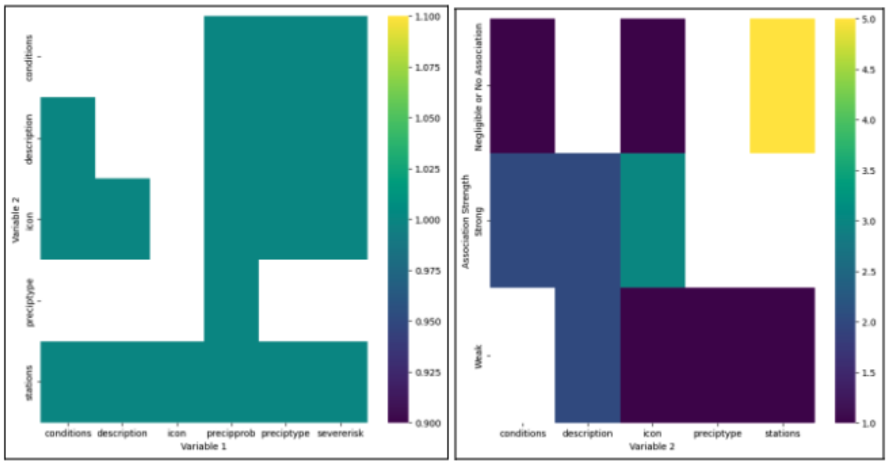


Figure 4.6: Associations

This classification allows us to not only identify significant relationships but also to understand the extent of their impact, providing clear direction on which interactions may be most meaningful in predictive analytics and model building. The relationships deemed **Strong** will be particularly scrutinized for their potential influence on the behavior of the dataset.

The following are the Highly associated variables:

```
    Highly Related Variable Pairs:
precipprob - conditions, precipprob - description, precipprob - icon,
preciptype - conditions, conditions - description, conditions - icon,
description - icon
```

Chapter 5

Checks

Before diving into the modeling, it is weather data, highly seasonal and non-stationary, we need to perform certain tests before proceeding to forecasting.

5.1 Stationary tests

5.1.1 ACF and PACF Plots

As shown in Figure 4.2, the ACF plot illustrates a slow decay in the ACF, where the bars gradually decrease in size and remain outside the confidence interval (the blue-shaded area), suggests the time series has a trend or seasonal structure. The presence of a trend or seasonality violates the stationarity assumption since the mean level of the series changes over time.

Similarly, as shown in Figure 4.3, the PACF shows a significant spike at the first lag and possibly at subsequent lags before cutting off. This indicates that there's some autoregressive component to the series. In a stationary time series, the PACF typically drops to zero relatively quickly (usually after a few lags), indicating that past values have limited or no effect on future values beyond a certain point.

5.1.2 Augmented Dickey-Fuller Method

This test is a very popular test for predicting stationary:

```
                              ADF Test for Temperature:
ADF Statistic: -2.244293156080437
P-Value: 0.19056394340349986
Number of Lags: 13
Number of Observations Used for ADF Regression and Critical Values Calculation: 863
Critical Values:
1%: -3.4379500665211276
```

```
5%: -2.864894878219008
10%: -2.5685563904109867
```

- The ADF Statistic is higher (less negative) than the critical values.
- The p-value is above a common threshold (e.g., 0.05), which means we cannot reject the null hypothesis of the presence of a unit root.

Hence both theoretically and practically its not stationary, lets us proceed with Differentiation to remove the seasonality.

5.2 Differencing

5.2.1 ACF and PACF

To address non-stationarity before applying ARIMA (AutoRegressive Integrated Moving Average) modeling, differencing is commonly used. Differencing involves computing the differences between consecutive observations, which can help stabilize the mean of the time series data. After perform differencing, we obtain the following ACF and PACF plots. The correlations at all lags are now

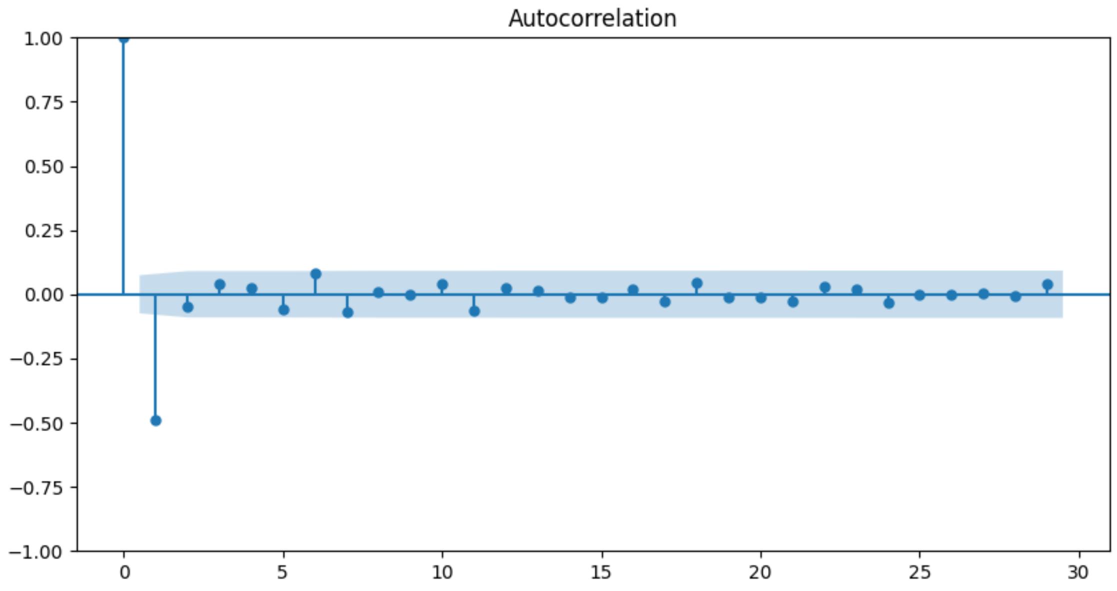


Figure 5.1: ACF plot

within the blue shaded area, which is the confidence interval. This implies there are no significant correlations in the data at different lags, which is what we expect from a stationary time series.

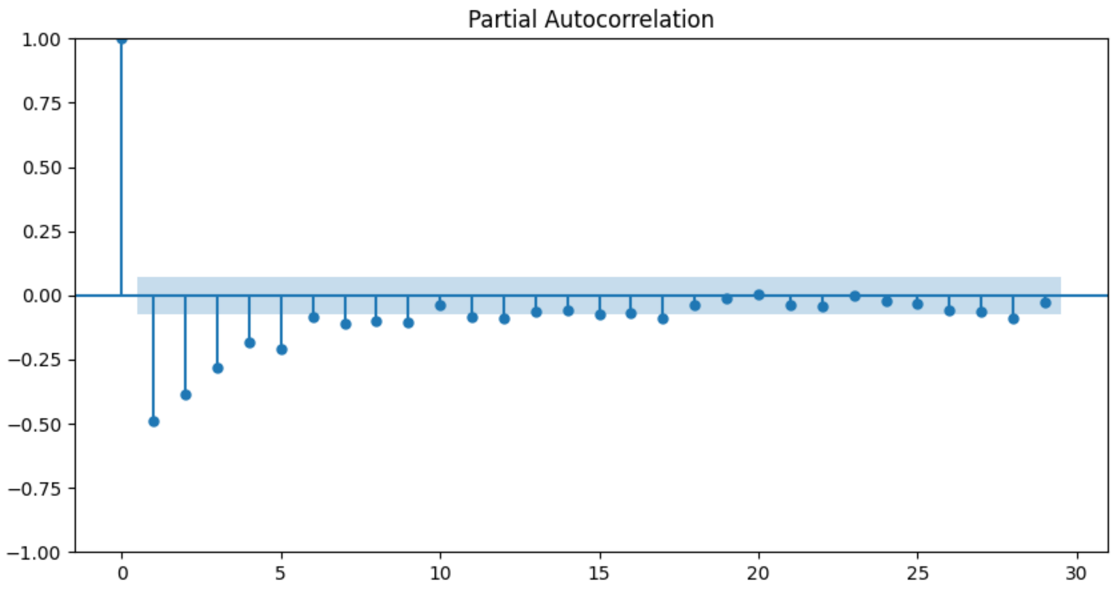


Figure 5.2: PACF plot

Similar to the ACF, the PACF also shows lags within the confidence interval, although there's a slight spike at lag1. The ACF and PACF plots suggest that differencing has helped achieve stationarity in the time series.

5.2.2 ADF Test

The p-value is extremely small (practically zero) from the below code, which strongly suggests the differenced series is stationary. We can reject the null hypothesis of a unit root.

```
adf_test = adfuller(df_train_diff)
print(f'p-value: {adf_test[1]}')
p-value: 2.480128867451355e-21
```

Chapter 6

Models and Results

We are using two models, that are ARIMA and SARIMA.

6.1 ARIMA

ARIMA is a general class of statistical models for time series analysis forecasting. It stands for Auto-Regressive Integrated Moving Average. When applying ARIMA models, we use a time series' past values and/or forecast errors to predict its future values.

AR (Auto-Regressive): This component involves regressing the time series on its past values. In other words, the current value of the series depends linearly on its previous values.

I (Integrated): If the time series exhibits non-stationarity, meaning its statistical properties vary over time, the integrated component can be applied. Integration involves differencing the series to make it stationary, effectively computing the differences between consecutive observations.

MA (Moving Average): Here, the time series is modeled based on its past forecast errors. It's termed "moving average" because each observation is considered a weighted average of its past forecast errors.

The relationship between these components in ARIMA is such that AR and MA models typically operate on stationary time series. However, if the series is non-stationary, the I component can render it stationary (via differencing).

ARIMA parameters p, d, q:

So corresponding to the three components of ARIMA mentioned above, there are three parameters p, d, and q. They take non-negative integer values, indicating which specific ARIMA model is used.

p: The number of past values included in the AR model

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$

where c is a constant, $\phi_1, \ldots, \phi_p$ are parameters, and ε_t is white noise.

d: The number of times the time series is differenced. For example, the first-order differencing is calculated as below:

$$\nabla y_t = y_t - y_{t-1}$$

q: The number of past forecast errors included in the MA model, or the size of the moving average window. It's named the MA model since each y_t can be considered as a weighted moving average of past forecast errors.

$$y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

So the ARIMA models are indeed a general class of models, including AR, MA, and ARMA. For example, ARIMA$(p, 0, 0)$ is equivalent to AR(p), ARIMA$(0, 0, q)$ is equivalent to MA(q). The full model equation of ARIMA(p, d, q) is:

$$\nabla y_t = c + \phi_1 \nabla y_{t-1} + \cdots + \phi_p \nabla y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$$

Here, ∇y_t denotes the differenced time series, which could involve multiple differencing operations to ensure stationarity.

After deciding the parameters of p, d, and q (this happens in the model fitting as it is, gets the optimal value), we can fit the ARIMA model in Python! We'll use the classic Python package `statsmodels`.

```
from statsmodels.tsa.arima.model import ARIMA
model = ARIMA(train, order=(1,1,1))
model_fit = model.fit()
print(model_fit.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                   temp   No. Observations:                  702
Model:                 ARIMA(1, 1, 1)   Log Likelihood               -2618.994
Date:                Fri, 26 Apr 2024   AIC                           5243.987
Time:                        05:13:57   BIC                           5257.645
Sample:                             0   HQIC                          5249.266
                                - 702
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -0.0353      0.039     -0.912      0.362      -0.111       0.041
ma.L1         -0.9999      0.201     -4.971      0.000      -1.394      -0.606
sigma2       101.9949     22.196      4.595      0.000      58.491     145.499
======================================================================================
Ljung-Box (L1) (Q):                   0.01   Jarque-Bera (JB):                22.53
Prob(Q):                              0.92   Prob(JB):                         0.00
Heteroskedasticity (H):               1.08   Skew:                            -0.21
Prob(H) (two-sided):                  0.57   Kurtosis:                         2.22
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

By default, the auto process uses the KPSS unit root test to select the value of parameter d, then uses the AIC information criteria to determine the values of p and q. On further implementation of testing_set, forecasting the weather for the next 30 days leads us to:

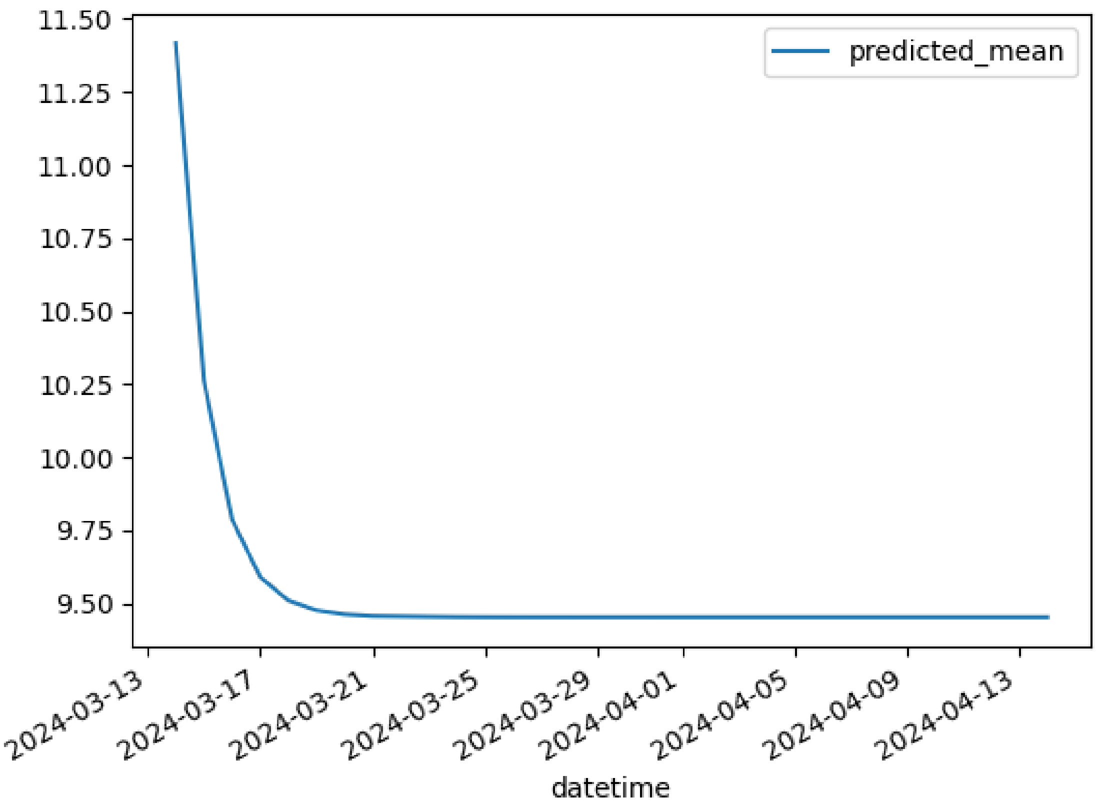


Figure 6.1: Prediction for the next 30 days - Graph

```
2024-04-14    18.955524
2024-04-15    15.731766
2024-04-16    14.413743
2024-04-17    13.874874
2024-04-18    13.654559
2024-04-19    13.564484
2024-04-20    13.527657
```

```
2024-04-21    13.512600
2024-04-22    13.506444
2024-04-23    13.503928
2024-04-24    13.502899
2024-04-25    13.502478
2024-04-26    13.502306
2024-04-27    13.502236
2024-04-28    13.502207
2024-04-29    13.502195
2024-04-30    13.502190
2024-05-01    13.502188
2024-05-02    13.502188
2024-05-03    13.502187
2024-05-04    13.502187
2024-05-05    13.502187
2024-05-06    13.502187
2024-05-07    13.502187
2024-05-08    13.502187
2024-05-09    13.502187
2024-05-10    13.502187
2024-05-11    13.502187
2024-05-12    13.502187
2024-05-13    13.502187
2024-05-14    13.502187
Freq: D, Name: ARIMA Predictions, dtype: float64
```

The temperature is constant over time, so including the parameters from Numerical relationships, might change the results. Further to interpret, ARIMA uses only one feature which is the average temperature over the entire day (`temp`).

6.2 SARIMA

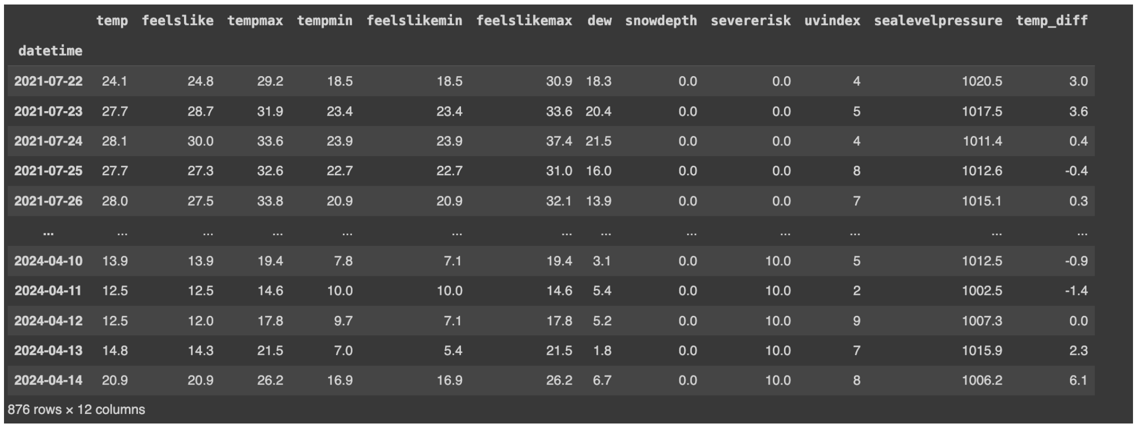
In order to address the issues in ARIMA, lets use SARIMA which uses all features and considers the seasonality as well. SARIMAX, or Seasonal Autoregressive Integrated Moving Average with Exogenous Variables, extends the capabilities of ARIMA models by incorporating additional external variables, also known as exogenous variables. SARIMAX is particularly useful for time series analysis and forecasting when the target variable may be influenced by external factors beyond its own past values.

Exogenous Variables: In addition to the AR, I, and MA components, SARIMAX allows for the inclusion of exogenous variables that may influence the behavior of the target variable. These external factors could include socioeconomic indicators, environmental variables, or other time series data that are believed to impact the series of interest.

SARIMAX parameter (p, d, q, s): Similar to ARIMA, SARIMAX is specified using three sets of parameters: p, d, q for the ARIMA components and (p, d, q, s) for the seasonal components. Additionally, SARIMAX includes parameters to specify the exogenous variables.

- (p, d, q): Define the orders of the autoregressive, differencing, and moving average components, respectively.
- (p, d, q, s): Specify the seasonal orders for the seasonal autoregressive, differencing, and moving average components, along with the seasonal period (s).

Exogenous Variables: Include the relevant exogenous variables in the model, along with their coefficients. Here we involve all the numerical features that exhibit important correlation with the temperature like the following table except for index and predictor (a.k.a temperature):



datetime	temp	feelslike	tempmax	tempmin	feelslikemin	feelslikemax	dew	snowdepth	severerisk	uvindex	sealevelpressure	temp_diff
2021-07-22	24.1	24.8	29.2	18.5	18.5	30.9	18.3	0.0	0.0	4	1020.5	3.0
2021-07-23	27.7	28.7	31.9	23.4	23.4	33.6	20.4	0.0	0.0	5	1017.5	3.6
2021-07-24	28.1	30.0	33.6	23.9	23.9	37.4	21.5	0.0	0.0	4	1011.4	0.4
2021-07-25	27.7	27.3	32.6	22.7	22.7	31.0	16.0	0.0	0.0	8	1012.6	-0.4
2021-07-26	28.0	27.5	33.8	20.9	20.9	32.1	13.9	0.0	0.0	7	1015.1	0.3
...	...	...	...	...	...	...	...	...	...	...	...	...
2024-04-10	13.9	13.9	19.4	7.8	7.1	19.4	3.1	0.0	10.0	5	1012.5	-0.9
2024-04-11	12.5	12.5	14.6	10.0	10.0	14.6	5.4	0.0	10.0	2	1002.5	-1.4
2024-04-12	12.5	12.0	17.8	9.7	7.1	17.8	5.2	0.0	10.0	9	1007.3	0.0
2024-04-13	14.8	14.3	21.5	7.0	5.4	21.5	1.8	0.0	10.0	7	1015.9	2.3
2024-04-14	20.9	20.9	26.2	16.9	16.9	26.2	6.7	0.0	10.0	8	1006.2	6.1

876 rows × 12 columns

Figure 6.2: Exogenous features taken

Now let's begin coding this python with already existing (p,d,q) value and seasonality of 4:

```
                                     SARIMAX Results
==================================================================================
Dep. Variable:                                     temp   No. Observations:     701
Model:                SARIMAX(1, 0, 0)x(1, 1, [1, 2], 4)   Log Likelihood    -206.640
Date:                                  Fri, 26 Apr 2024   AIC                445.280
Time:                                          05:59:48   BIC                518.028
Sample:                                               0   HQIC               473.407
                                                  - 701
Covariance Type:                                    opg
==================================================================================
                       coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------
feelslike            0.6474      0.013     51.144      0.000       0.623       0.672
tempmax              0.4502      0.009     47.528      0.000       0.432       0.469
```

```
tempmin              0.4323      0.014     30.683      0.000       0.405       0.460
feelslikemin        -0.2759      0.011    -24.746      0.000      -0.298      -0.254
feelslikemax        -0.2888      0.009    -32.932      0.000      -0.306      -0.272
dew                  0.0145      0.007      2.170      0.030       0.001       0.028
snowdepth            0.0369      0.020      1.848      0.065      -0.002       0.076
severerisk           0.0004      0.002      0.146      0.884      -0.004       0.005
uvindex             -0.0032      0.007     -0.451      0.652      -0.017       0.011
sealevelpressure     0.0018      0.002      0.815      0.415      -0.003       0.006
temp_diff            0.0002      0.003      0.055      0.956      -0.006       0.006
ar.L1                0.0083      0.041      0.203      0.839      -0.072       0.088
ar.S.L4             -0.8718      0.134     -6.496      0.000      -1.135      -0.609
ma.S.L4             -0.0580      0.123     -0.473      0.636      -0.298       0.182
ma.S.L8             -0.8633      0.114     -7.590      0.000      -1.086      -0.640
sigma2               0.1058      0.005     21.443      0.000       0.096       0.115
===================================================================================
Ljung-Box (L1) (Q):                   0.27   Jarque-Bera (JB):                49.06
Prob(Q):                              0.60   Prob(JB):                         0.00
Heteroskedasticity (H):               0.83   Skew:                            -0.10
Prob(H) (two-sided):                  0.15   Kurtosis:                         4.28
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

After modeling this lets perform the Root mean squared error on the test-set, now in this the model has overfit, due to accuracy and a lesser number of data.

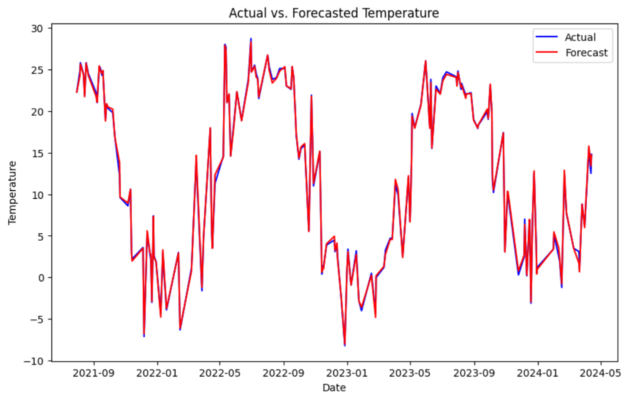


Figure 6.3: Forecast on Test-data

But the forecast for the future 30 days, seem to have captured the patterns and shows a more realistic graph.

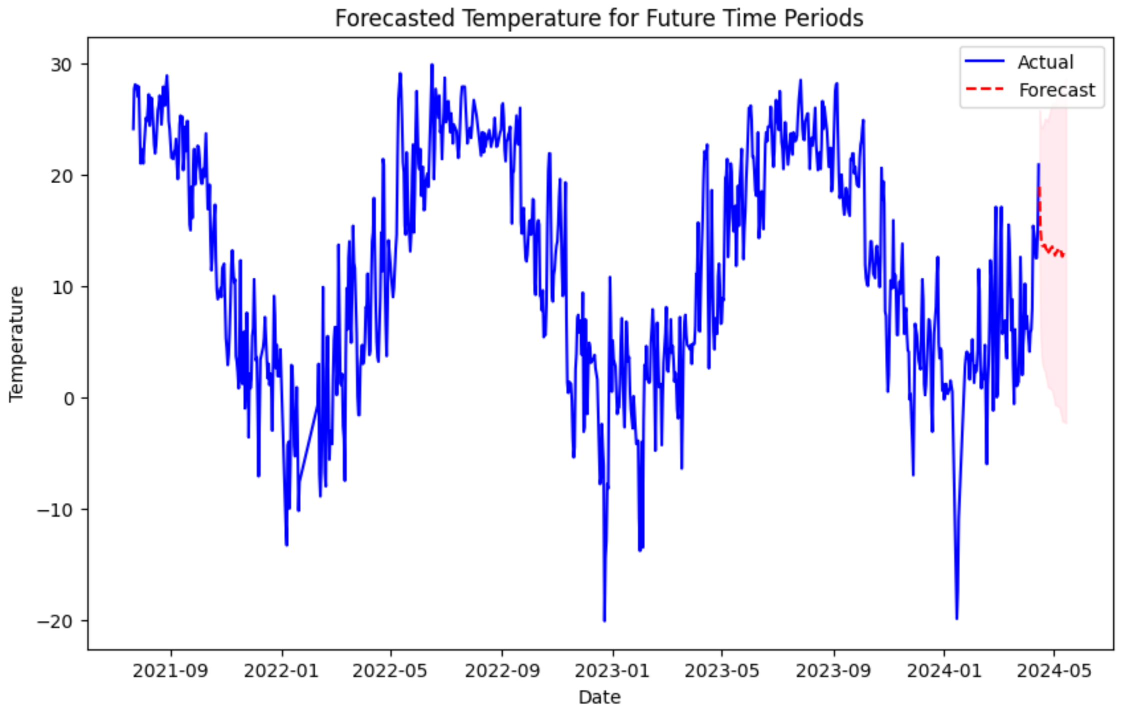


Figure 6.4: Forecast for next 30 days

Chapter 7

Conclusion

- **A wider array of data**: Including a broader range of data can enhance the model's ability to capture complex patterns and relationships in the time series. This could involve incorporating additional variables such as economic indicators, demographic data, or environmental factors that may influence the target variable. By expanding the scope of the dataset, the model can better adapt to diverse and evolving conditions, leading to more accurate predictions.

- **Adding categorical features**: In addition to numerical features, incorporating categorical variables can provide valuable insights into the underlying dynamics of the time series. Categorical features such as weather conditions, seasonality indicators, or event flags can offer contextual information that complements the numerical data. By including these factors in the model, we can capture nuanced patterns and dependencies that may not be apparent from numerical variables alone, thereby improving the predictive accuracy and interpretability of the model.

- **Integrating machine learning models**: While ARIMA and SARIMAX models are powerful tools for time series forecasting, integrating machine learning algorithms such as random forests or neural networks can further enhance predictive performance. These models offer flexibility and scalability, allowing them to capture complex nonlinear relationships and interactions in the data. By combining traditional time series methods with machine learning techniques, we can leverage the strengths of both approaches to develop more accurate and robust forecasting models. Additionally, machine learning models can handle larger and more diverse datasets, making them well-suited for applications requiring high-dimensional or heterogeneous data.

Bibiliography

1. Geophrey Odero. *Time Series Analysis of Weather Data In South Carolina.* Master's thesis, University of South Carolina, 2019. Accepted by: David Hitchcock, Director of Thesis; Karl Gregory, Reader; John Grego, Reader; Cheryl L. Addy, Vice Provost and Dean of the Graduate School.

2. Yuchuan Lai and David A. Dzombak. Use of the Autoregressive Integrated Moving Average (ARIMA) Model to Forecast Near-Term Regional Temperature and Precipitation. Online Publication: 01 Apr 2020. Print Publication: 01 Jun 2020. DOI: `https://doi.org/10.1175/WAF-D-19-0158.1`. Page(s): 959–976.

3. Peng Chen, Aichen Niu, Duanyang Liu, Wei Jiang, and Bin Ma. Time Series Forecasting of Temperatures using SARIMA: An Example from Nanjing. Jiangsu Meteorological Information Center, Nanjing, China. Jiangsu Meteorological Observatory, Nanjing, China. Jiangsu Meteorological Climate Center, China.

4. Ozan Döğer. Time Series Forecasting using SARIMA: Python. `https://www.visualcrossing.com/weather/weather-data-services/Chicago,United%20State`.

5. Title of the video: *Forecasting Future Sales Using ARIMA and SARIMAX, by Krish Naik.* Available online: `https://www.youtube.com/watch?v=2XGSIlgUBDI`.