**Applied Machine Learning**

**Report**

**Weather Prediction Model**

Nida Aiyman

Prerna Kakade

Bhuvana Venu

****

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**PRESIDENCY UNIVERSITY**

**BENGALURU**

**November 18, 2023**

**Abstract**

Prediction of weather has always been a challenging task because of the rapid changes  
in weather. In recent years, machine learning techniques have shown promising results in  
improving the accuracy and efficiency of weather prediction.

This study focuses on the use of Machine Learning using the AI Technology, that is Google  
Colab, and implementing different Machine Learning algorithms. The training dataset is  
carefully preprocessed to handle missing values and outliers. Feature engineering techniques are applied to extract meaningful information from raw data.

Results indicate that the ML-based weather prediction model demonstrates improved accuracy compared to traditional methods, particularly in short-term predictions. The model’s ability to adapt to changing weather conditions and learn from diverse datasets makes it a valuable tool for enhancing the reliability of weather predictions.

**Introduction**

Weather prediction is a critical aspect of modern society, influencing various sectors such as agriculture, transportation, and emergency management. Accurate and timely weather predictions are essential for making informed decisions.

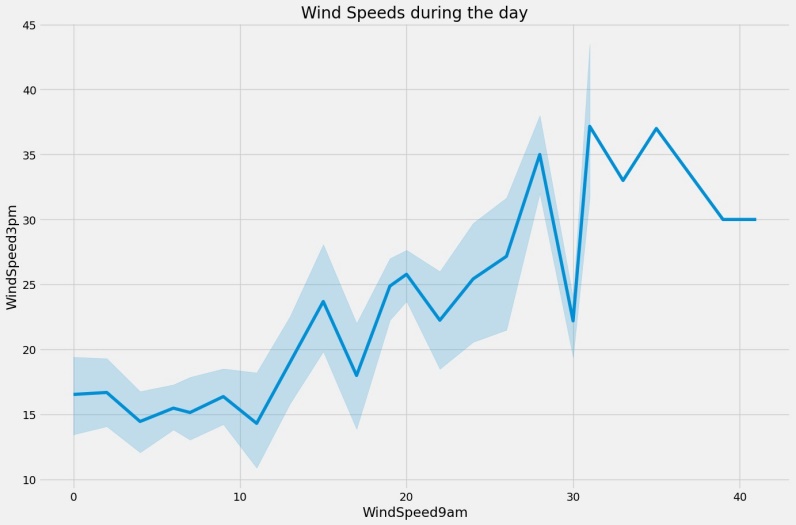
Machine Learning based models are used for independent evaluation. ML models can predict weather events such as storms, temperature changes, and rainfall with remarkable precision by identifying patterns in historical data. By training on a diverse dataset that includes both historical and real-time information, the model aims to provide more accurate and timely predictions. This research contributes to the evolving field of ML-based weather prediction by presenting a comprehensive model that combines historical data, satellite imagery, and advanced ML techniques.

The following sections will deal with the methodology, data preprocessing, and evaluation  
metrics employed to develop and assess the effectiveness of the proposed weather prediction model. Ultimately, the goal is to advance the capabilities of weather prediction, providing decision-makers and the general public with more reliable information for planning and responding to weather related events.

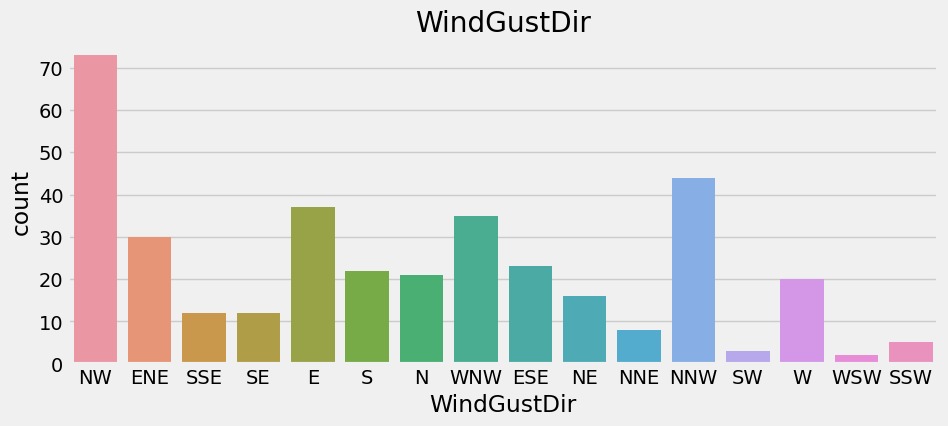
**Techniques Used**

1. Data Visualization: The process of finding trends and correlations in our data by representing it pictorially is called Data Visualization.

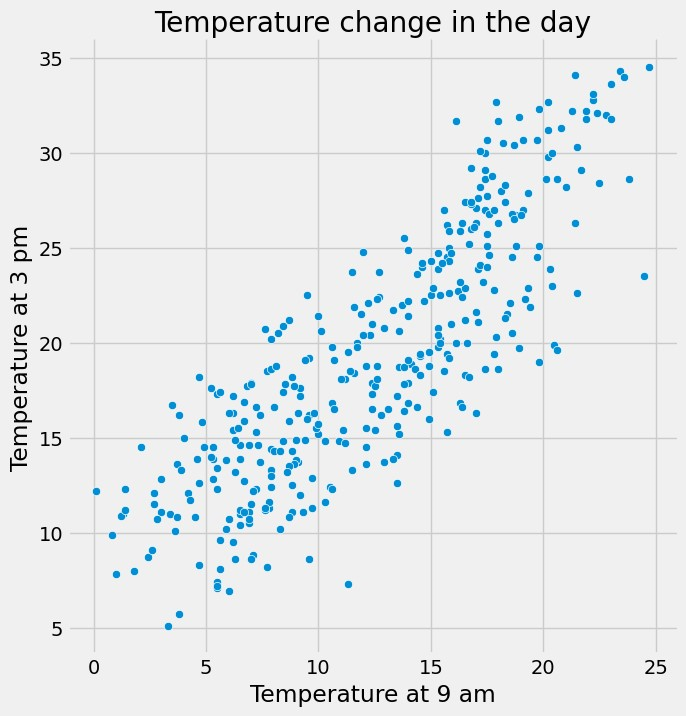
* Line Plot: A line plot is created by connecting the values in the input data with straight lines. It is also known as a dot plot or stem plot and is a graphical representation of data on a number line using dots, crosses, or any other symbol. Each mark represents a specific quantity which is the scale of the graph.



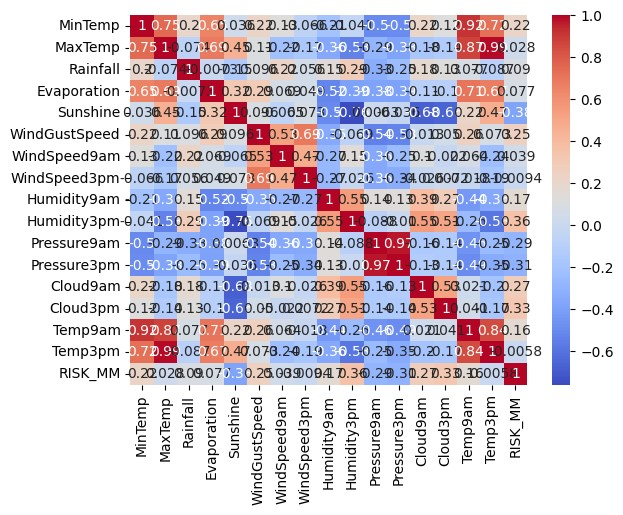
* Bar Plot : The Bar graph is a graphical representation of the categorical data with rectangular bars with heights proportional to the values that they present. The Bar graph can be a horizontal or vertical representation. In Bar plot, one axis represents the categories and another axis represents the measured value in each category.



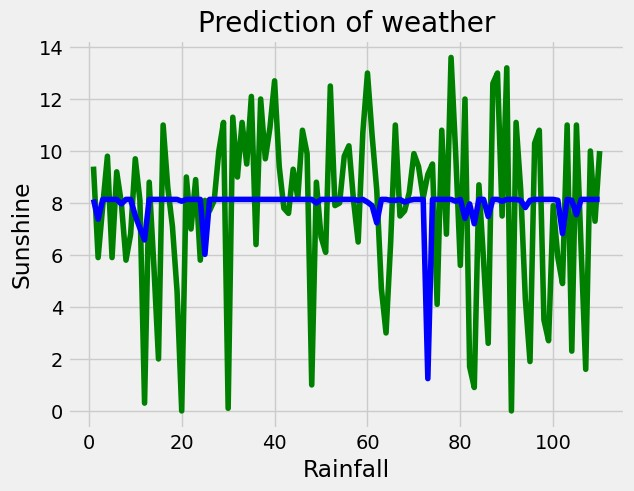
* Pairplot : A seaborn pairplot is a visualization that shows the relationship between all pairs of variables in a dataset. It is a great way to explore data and identify patterns. The pairplot creates a grid of plots, with each variable on the x-axis and y-axis. The plots can be either scatter plots, histograms, or a combination of both.



* Heatmap : A heatmap in Python is a two-dimensional graphical representation of data, each value in a matrix is represented by a different colour. With the help of the Matplotlib tools provided by the Seaborn package, annotated heatmaps can be created and customised to the creator's specifications.



2. Linear Regression using SciKit Learn: Linear regression is a statistical method that predicts the relationship between two variables. It's used in data science and machine learning for predictive analysis.



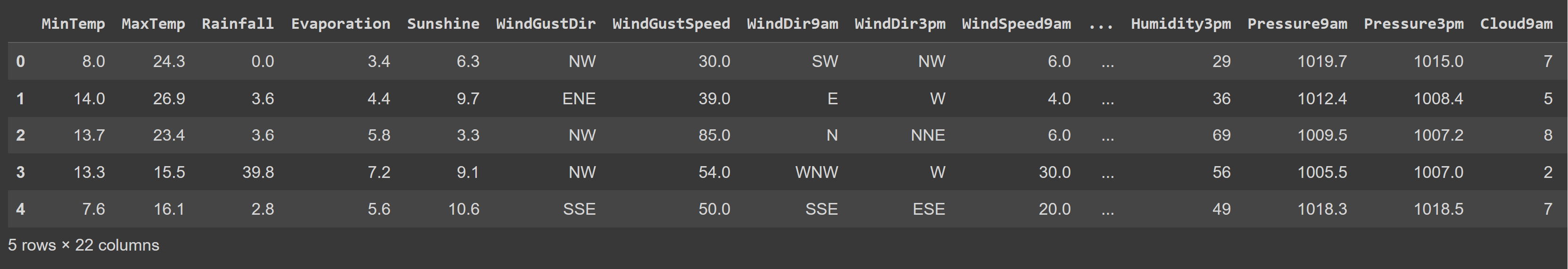
**Implementation**

import pandas as pd

data = pd.read\_csv('/content/weather.csv')

Performing exploratory data analysis:

data.head()



data.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 366 entries, 0 to 365

Data columns (total 22 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 MinTemp 366 non-null float64

1 MaxTemp 366 non-null float64

2 Rainfall 366 non-null float64

3 Evaporation 366 non-null float64

4 Sunshine 363 non-null float64

5 WindGustDir 363 non-null object

6 WindGustSpeed 364 non-null float64

7 WindDir9am 335 non-null object

8 WindDir3pm 365 non-null object

9 WindSpeed9am 359 non-null float64

10 WindSpeed3pm 366 non-null int64

11 Humidity9am 366 non-null int64

12 Humidity3pm 366 non-null int64

13 Pressure9am 366 non-null float64

14 Pressure3pm 366 non-null float64

15 Cloud9am 366 non-null int64

16 Cloud3pm 366 non-null int64

17 Temp9am 366 non-null float64

18 Temp3pm 366 non-null float64

19 RainToday 366 non-null object

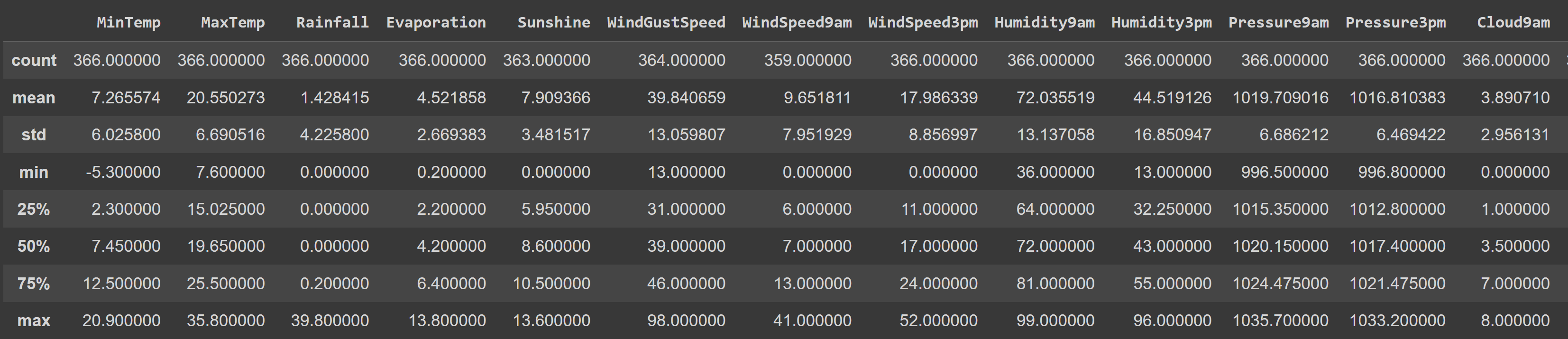
20 RISK\_MM 366 non-null float64

21 RainTomorrow 366 non-null object

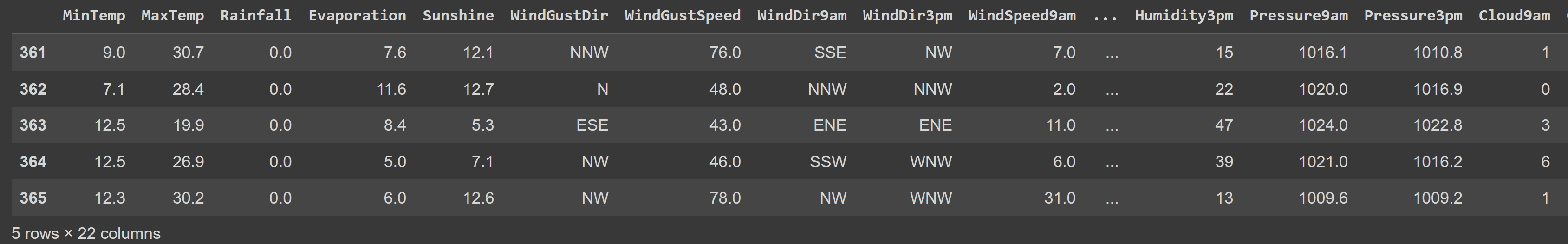
dtypes: float64(12), int64(5), object(5)

memory usage: 63.0+ KB

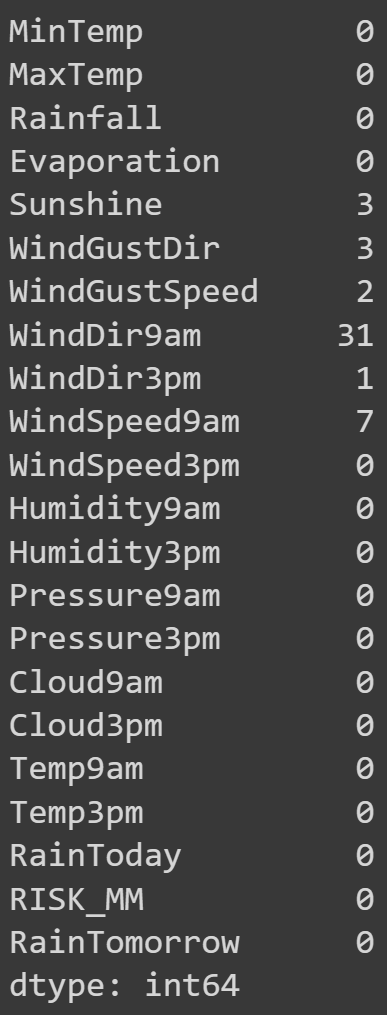
data.describe()



data.tail()

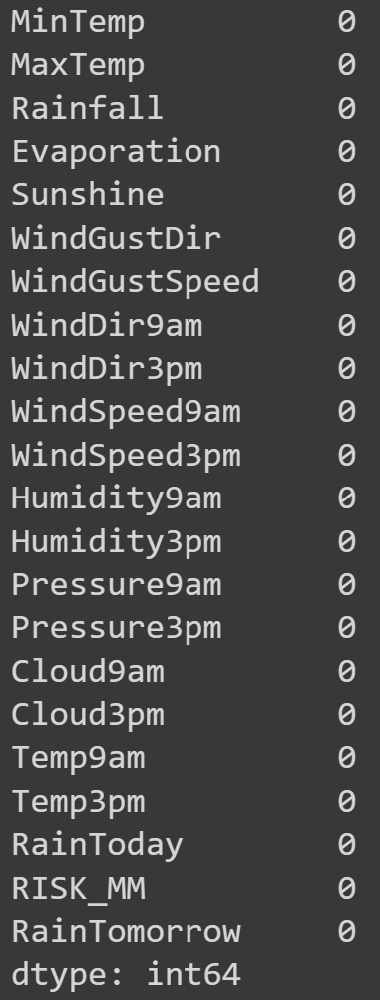


data.isnull().sum() #To check the null values in the data



dropped = data.dropna() #Dropping the null values

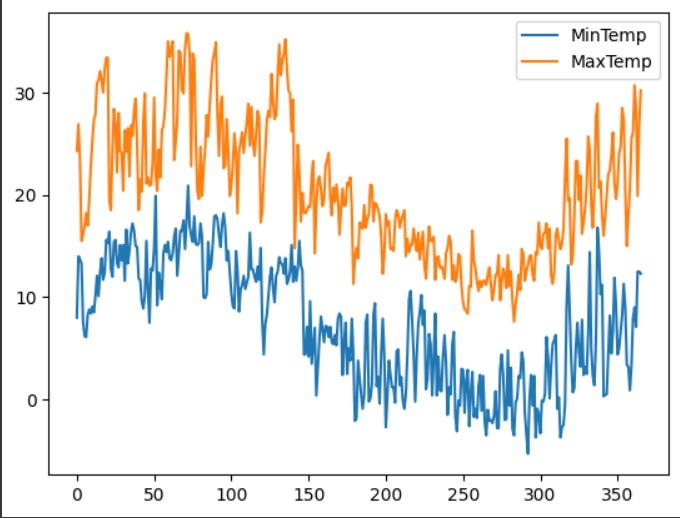
print(dropped.isnull().sum())



import matplotlib.pyplot as plt

import seaborn as sns

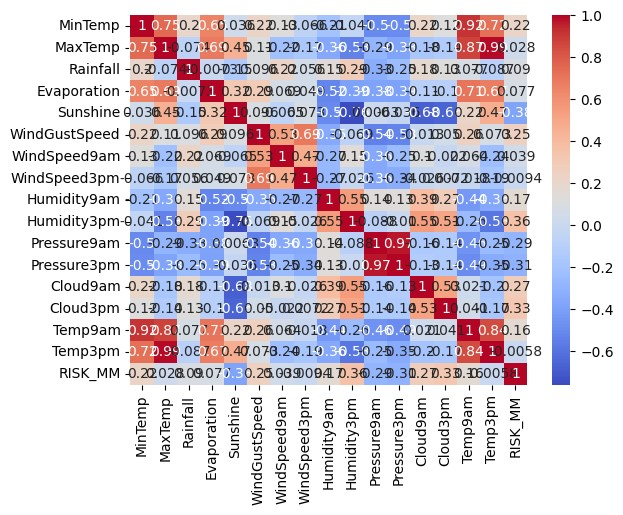
data.plot(kind='line',y=(['MinTemp','MaxTemp']))



Creating a heatmap of the correlation in weather

sns.heatmap(data.corr(),annot=True,cmap='coolwarm')

plt.show()



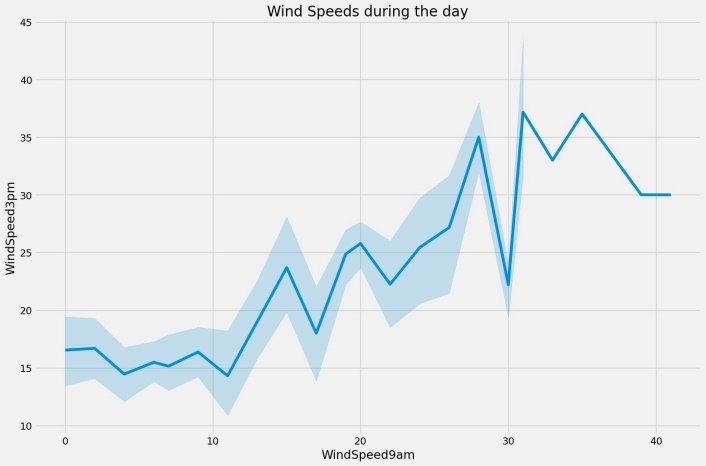
plt.style.use('fivethirtyeight')

plt.figure(figsize=(15, 10))

plt.title("Wind Speeds during the day")

sns.lineplot(data = data, x='WindSpeed9am', y='WindSpeed3pm')

plt.show()



plt.figure(figsize=(12,6))

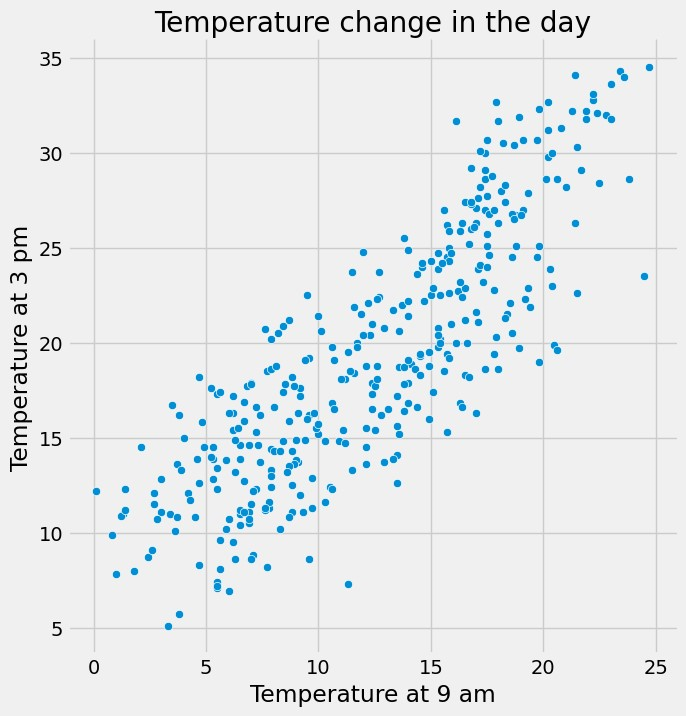
sns.pairplot(data,x\_vars=['Temp9am'],y\_vars=['Temp3pm'],size=7,kind='scatter')

plt.xlabel('Temperature at 9 am')

plt.ylabel('Temperature at 3 pm')

plt.title('Temperature changes in the day')

plt.show()



plt.figure(figsize=(12,6))

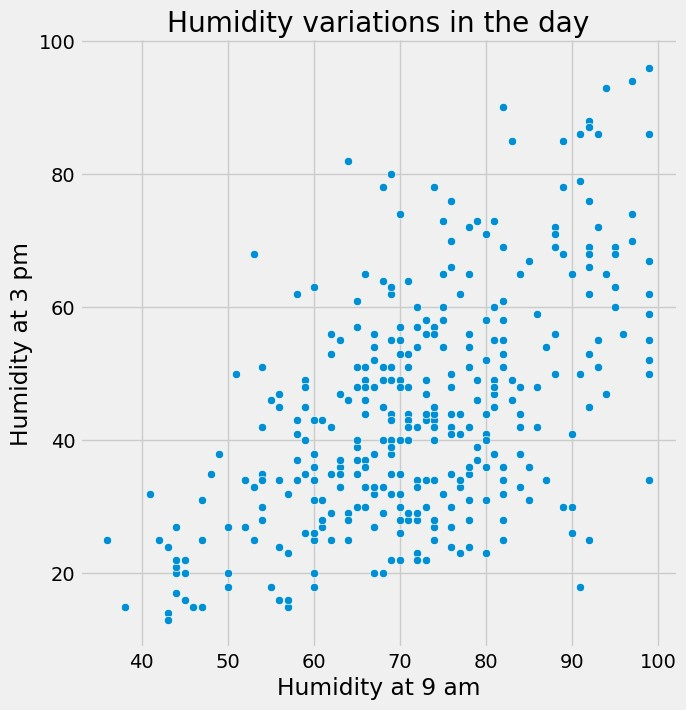
sns.pairplot(data,x\_vars=['Humidity9am'],y\_vars=['Humidity3pm'],size=7,kind='scatter')

plt.xlabel('Humidity at 9 am')

plt.ylabel('Humidity at 3 pm')

plt.title('Humidity variations in the day')

plt.show()



plt.figure(figsize=(12,6))

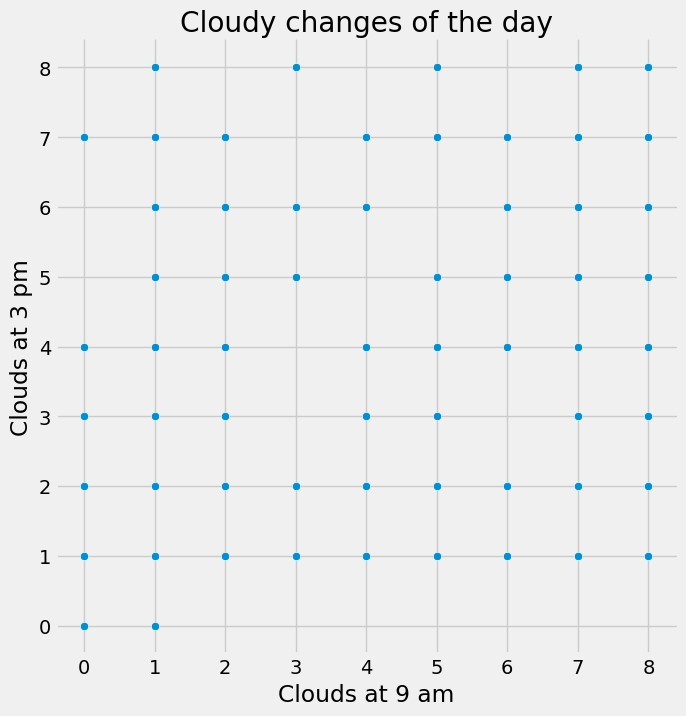
sns.pairplot(data,x\_vars=['Cloud9am'],y\_vars=['Cloud3pm'],size=7,kind='scatter')

plt.xlabel('Clouds at 9 am')

plt.ylabel('Clouds at 3 pm')

plt.title('Cloudy changes of the day')

plt.show()

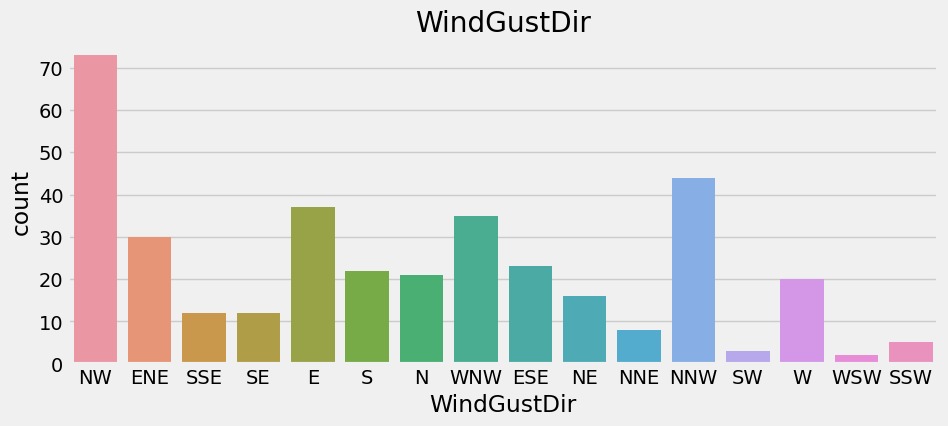


fig, ax = plt.subplots(figsize=(10, 4))

sns.countplot(x="WindGustDir", data=data)

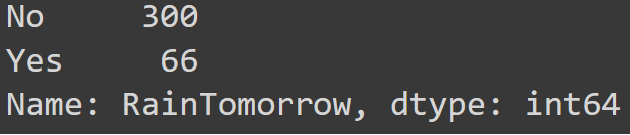
plt.title("WindGustDir")

plt.show()



The probability of rain tomorrow:

data.RainTomorrow.value\_counts()

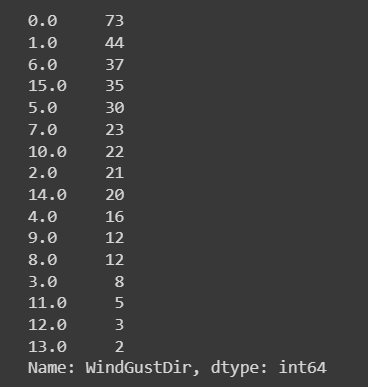


Data Wrangling: Converting raw data into usable form.

(Wind can change it's direction at any point.)

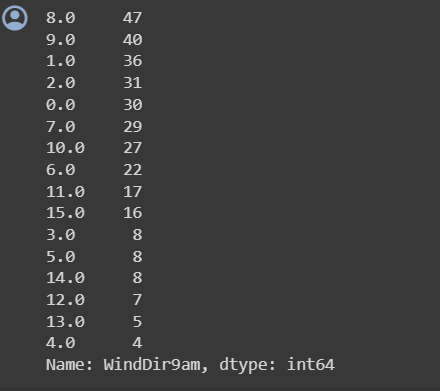
data['WindGustDir'] = data['WindGustDir'].replace(['NW', 'NNW', 'N', 'NNE', 'NE', 'ENE', 'E', 'ESE','SE', 'SSE', 'S', 'SSW', 'SW', 'WSW', 'W', 'WNW'],[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15])

data['WindGustDir'].value\_counts()



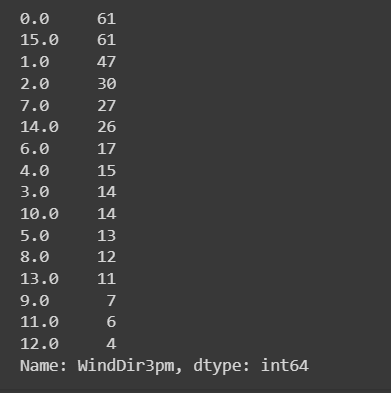
data['WindDir9am'] = data['WindDir9am'].replace(['NW', 'NNW', 'N', 'NNE', 'NE', 'ENE', 'E', 'ESE','SE', 'SSE', 'S', 'SSW', 'SW', 'WSW', 'W', 'WNW'],[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15])

data['WindDir9am'].value\_counts()



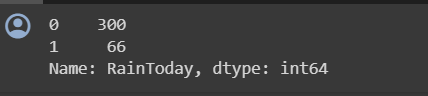
data['WindDir3pm'] = data['WindDir3pm'].replace(['NW', 'NNW', 'N', 'NNE', 'NE', 'ENE', 'E', 'ESE','SE', 'SSE', 'S', 'SSW', 'SW', 'WSW', 'W', 'WNW'],[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15])

data['WindDir3pm'].value\_counts()



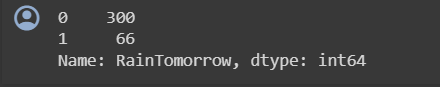
data['RainToday'] = data['RainToday'].replace(["Yes", "No"],[1, 0])

data['RainToday'].value\_counts()

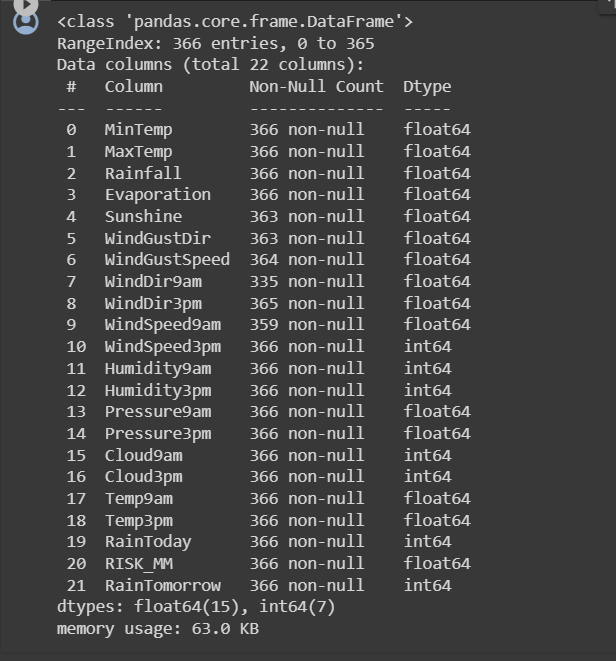


data['RainTomorrow'] = data['RainTomorrow'].replace(["Yes", "No"], [1, 0])

data['RainTomorrow'].value\_counts()



data.info()



data['Sunshine'].isnull().sum()



import numpy as np

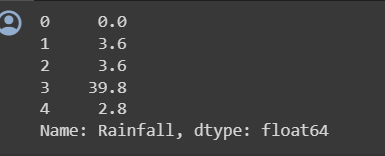
data['Sunshine']=data['Sunshine'].replace(np.NaN,1)

Segregating data from scikit learn:

from sklearn.model\_selection import train\_test\_split

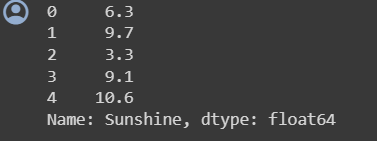
x=data['Rainfall']

x.head()



y = data['Sunshine']

y.head()



x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,train\_size = 0.7,random\_state = 100) #Splitting the data for train and test

Creating new axis for x column:

x\_train = x\_train[:,np.newaxis]

x\_test = x\_test[:,np.newaxis]

Generating a linear regression model from scikit learn

from sklearn.linear\_model import LinearRegression

#Fitting the model

lr = LinearRegression()

lr.fit(x\_train,y\_train)



Predicting the maximum temperature by using the test values:

y\_pred = lr.predict(x\_test)

Plotting the actual and predicted values :

c = [i for i in range(1,len(y\_test)+1,1)]

plt.plot(c,y\_test,color='g',linestyle = '-')

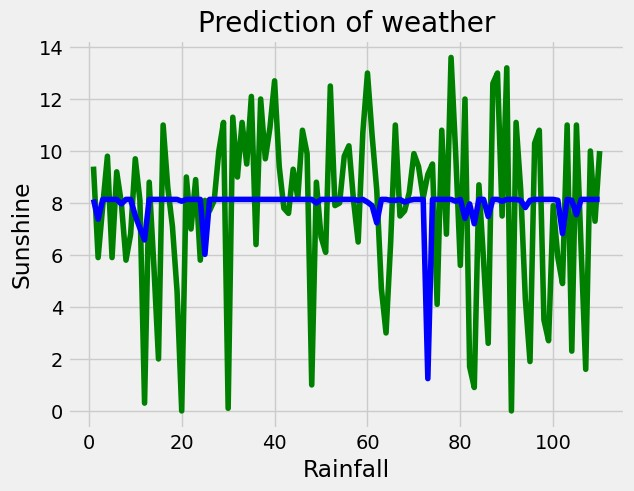
plt.plot(c,y\_pred,color='b',linestyle='-')

plt.xlabel('Rainfall')

plt.ylabel('Sunshine')

plt.title('Prediction of weather')

plt.show()



**References**

* Chatgpt
* <https://www.researchgate.net/publication/360365118_Smart_Weather_Prediction_Using_Machine_Learning>
* [weather-prediction · GitHub Topics · GitHub](https://github.com/topics/weather-prediction)
* sciencedirect.com
* geeksforgeeks.com