WEATHER PREDICTION USING MACHINE LEARNING

Problem Description:

The goal is to predict weather conditions based on historical data and various meteorological features.

This involves forecasting weather parameters such as temperature, humidity, precipitation, wind speed, and other relevant factors like atmospheric pressure, cloud cover, etc. The aim is to build a model that can accurately predict these weather conditions for a given location and time.

The prediction can help in providing accurate weather forecasts, which is crucial for various applications such as agriculture, transportation, water management and disaster management.

Type of Problem:

Weather prediction can be framed as a **Regression** problem as we aim to predict numerical values such as min. temperature, max. temperature, humidity, pressure....

For a regression problem like weather prediction, labelled data with actual weather measurements is crucial for training the model.

Supervised learning is a technique where a model learns from labelled data to make predictions or classifications. Supervised learning can be used for regression tasks where the output is a continuous variable, or classification tasks, where the output is a discrete variable.

In the context of weather and climate forecasting, supervised learning can be used for tasks such as predicting the temperature or precipitation for a given location and time.

Machine Learning Algorithm can be broadly classified into three types:

Supervised Learning Algorithms

Unsupervised Learning Algorithms

Reinforcement Learning algorithm.

There are various machine learning algorithms that can be used for the prediction such as like Linear regression, Decision Tree, Random Forest, Support Vector Machines, K-Nearest Neighbour, K-Means Clustering, Logistic regression.

Linear regression is used for predicting continuous variables like temperature or humidity.

Linear regression is divided into two types.

Simple Linear Regression, Multiple Linear Regression

Simple Linear Regression - a single independent variable is used to predict the value of the dependent variable.

Multiple Linear Regression - more than one independent variables are used to predict the value of the dependent variable.

Data Sources:

For this, various sources of data can be considered:

- Historical weather data from meteorological agencies or weather stations.
- Satellite imagery data for cloud cover, temperature distribution, etc.
- IoT devices providing real-time weather updates.
- Weather APIs that offer historical and current weather data.

The Data used here is obtained from an open source Kaggle – Historical Weather Data for Indian Cities.

The Dataset can be found here.

Data covers a significant period from 2009 – 2020 which contains hourly weather data to ensure completeness. This data can be used to visualize the change in data due to global warming or can be used to predict the weather for upcoming days, weeks, months, seasons, etc.

Displaying top 5 rows of the dataset

```
import pandas as pd
weather_data = pd.read_csv('bengaluru.csv')
weather_data.head()
```

✓ 0.2s

	date_time	maxtempC	mintempC	totalSnow_cm	sunHour	uvIndex	uvIndex.1	moon_illumination	moonrise	moonset	...	WindChillC	WindGustKmph
0	2009-01-01 00:00:00	27	12	0.0	11.6	5	1	31	09:58 AM	10:03 PM	...	18	11
1	2009-01-01 01:00:00	27	12	0.0	11.6	5	1	31	09:58 AM	10:03 PM	...	17	9
2	2009-01-01 02:00:00	27	12	0.0	11.6	5	1	31	09:58 AM	10:03 PM	...	16	7
3	2009-01-01 03:00:00	27	12	0.0	11.6	5	1	31	09:58 AM	10:03 PM	...	15	5
4	2009-01-01 04:00:00	27	12	0.0	11.6	5	1	31	09:58 AM	10:03 PM	...	18	5

The describe() method returns description of the data in the DataFrame.

```
weather_data.describe()
```

✓ 0.0s

	maxtempC	mintempC	totalSnow_cm	sunHour	uvIndex	uvIndex.1	moon_illumination	DewPointC	FeelsLikeC	HeatIndexC	WindChillC
count	96432.000000	96432.000000	96432.0	96432.000000	96432.000000	96432.000000	96432.000000	96432.000000	96432.000000	96432.000000	96432.000000
mean	29.646093	19.336735	0.0	10.653484	5.900448	4.013751	46.094077	16.085418	25.002261	25.269662	24.422152
std	3.446427	2.773771	0.0	1.986738	0.851346	2.869923	31.249725	4.161604	4.586926	4.430811	4.516766
min	18.000000	11.000000	0.0	4.200000	4.000000	1.000000	0.000000	-9.000000	12.000000	13.000000	12.000000
25%	27.000000	18.000000	0.0	8.800000	5.000000	1.000000	18.000000	14.000000	21.000000	22.000000	21.000000
50%	29.000000	20.000000	0.0	11.600000	6.000000	5.000000	46.000000	18.000000	25.000000	25.000000	24.000000
75%	32.000000	21.000000	0.0	11.600000	6.000000	7.000000	73.000000	19.000000	28.000000	28.000000	27.000000
max	40.000000	28.000000	0.0	12.900000	8.000000	10.000000	100.000000	25.000000	43.000000	43.000000	42.000000

Checking if there are any null values in the dataset.

```
weather_data.isnull()
```

✓ 0.0s

	date_time	maxtempC	mintempC	totalSnow_cm	sunHour	uvIndex	uvIndex.1	moon_illumination	moonrise	moonset	...	WindChillC	WindGustKmph	cloudcover	humidity
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
96427	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
96428	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
96429	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
96430	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False
96431	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False

96432 rows × 25 columns

```
weather_data.isnull().any()
```

✓ 0.0s

```
date_time           False
maxtempC            False
mintempC            False
totalSnow_cm        False
sunHour             False
uvIndex             False
uvIndex.1           False
moon_illumination   False
moonrise            False
moonset             False
sunrise             False
sunset              False
DewPointC           False
FeelsLikeC          False
HeatIndexC          False
WindChillC          False
WindGustKmph        False
cloudcover          False
humidity            False
precipMM            False
pressure            False
tempC               False
visibility          False
winddirDegree       False
windspeedKmph       False
dtype: bool
```

Checking the datatypes of the data

```
weather_data.dtypes
✓ 0.0s
```

```
date_time              object
maxtempC                int64
mintempC                int64
totalSnow_cm          float64
sunHour               float64
uvIndex                 int64
uvIndex.1               int64
moon_illumination       int64
moonrise               object
moonset                object
sunrise                object
sunset                 object
DewPointC               int64
FeelsLikeC              int64
HeatIndexC              int64
WindChillC              int64
WindGustKmph            int64
cloudcover              int64
humidity                int64
precipMM              float64
pressure                int64
tempC                   int64
visibility              int64
winddirDegree           int64
windspeedKmph           int64
dtype: object
```

Number of rows and columns in the data.

```
weather_data.shape
```

✓ 0.0s

```
(96432, 25)
```

```
row_count = len(weather_data)

print(f'The DataFrame has {row_count} rows.')
```

✓ 0.0s

```
The DataFrame has 96432 rows.
```

Obtaining the required fields

```
weather_data_num=weather_data.loc[:,['maxtempC','mintempC','cloudcover','humidity','tempC', 'sunHour','HeatIndexC', 'precipMM', 'pressure','windspe
weather_data_num.head()
```

Python

	maxtempC	mintempC	cloudcover	humidity	tempC	sunHour	HeatIndexC	precipMM	pressure	windspeedKmph
0	27	12	2	91	14	11.6	18	0.0	1014	8
1	27	12	2	93	14	11.6	17	0.0	1014	6
2	27	12	2	94	13	11.6	16	0.0	1014	4
3	27	12	2	96	12	11.6	15	0.0	1014	3
4	27	12	1	88	14	11.6	18	0.0	1015	3

New Columns

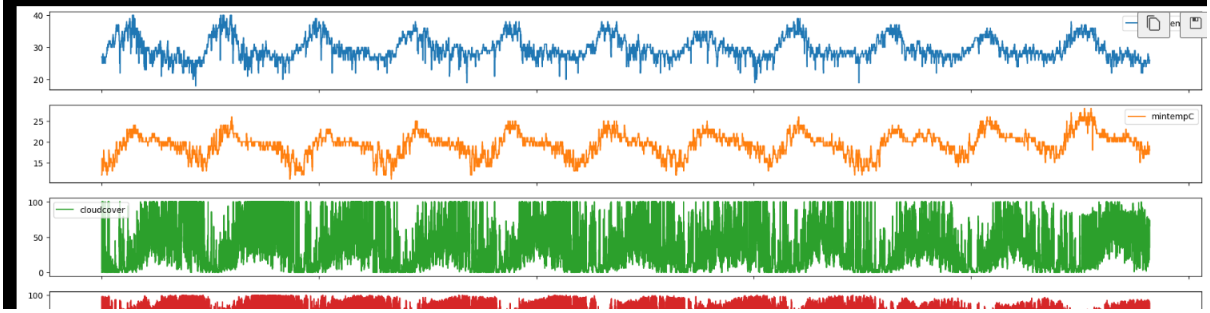
```
weather_data_num.columns
```

```
Index(['maxtempC', 'mintempC', 'cloudcover', 'humidity', 'tempC', 'sunHour',
       'HeatIndexC', 'precipMM', 'pressure', 'windspeedKmph'],
      dtype='object')
```

Plotting the columns

```
weather_data_num.plot(subplots=True, figsize=(25,20))
```

```
array([<Axes: >, <Axes: >, <Axes: >, <Axes: >, <Axes: >, <Axes: >,
       <Axes: >, <Axes: >, <Axes: >, <Axes: >], dtype=object)
```



Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

From scikit-learn(sklearn) library, library linear regression, decision tree regression, random forest regression models can be used for the prediction. L Linear Regression method can be used to create linear regression object. Using fit() method can be used to train the model and predict() method can be used to make new predictions on the new data. train_test_split() module of scikit-learn can be used for splitting the data into a train set and a test set.