**Univariate Time Series Analysis for Weather Prediction using IBM Cloud**

**1. INTRODUCTION:-**

**1.1. Overview:-**

Any data associated with the time and is dependent on time-related matters can be termed as time-series data. Time series analysis comprises methods for analyzing time series data in order to extract meaningful statistics like trends, non-stationarity, and seasonality based on a daily, weekly, yearly basis and other characteristics of the data. Time series forecasting is the use of a model to predict future values based on previously observed values.

The main objective of our project is to build a time series model for weather forecasting using Prophet library

**1.2.Purpose:-**

The purpose of our project is to predict the accurate weather temperature in Celsius using prophet library. Using this project we will know temperature on particular day we select. Time series analysis helps organizations understand the underlying causes of trends or systemic patterns over time.

**2. LITERATURE SURVEY:-**

**2.1. Existing problem:-**

* Design and successful implementation of weather predicting model is very difficult task because weather changes time to time.
* Weather forecasting is the application of science and technology to predict the conditions of the atmosphere for a given time. People have attempted to predict the weather informally for millennia and formally since the 19th century.
* There are many models present for the predictive analysis of time series like Machine learning ARIMA (Auto-Regressive Integrated Moving Average model), Auto-Regressive model, Exponential Smoothing, LSTM (Long Short Term Memory), etc. These models require the data to be fed and with certain tweaking and fine-tuning they help us to make predictions

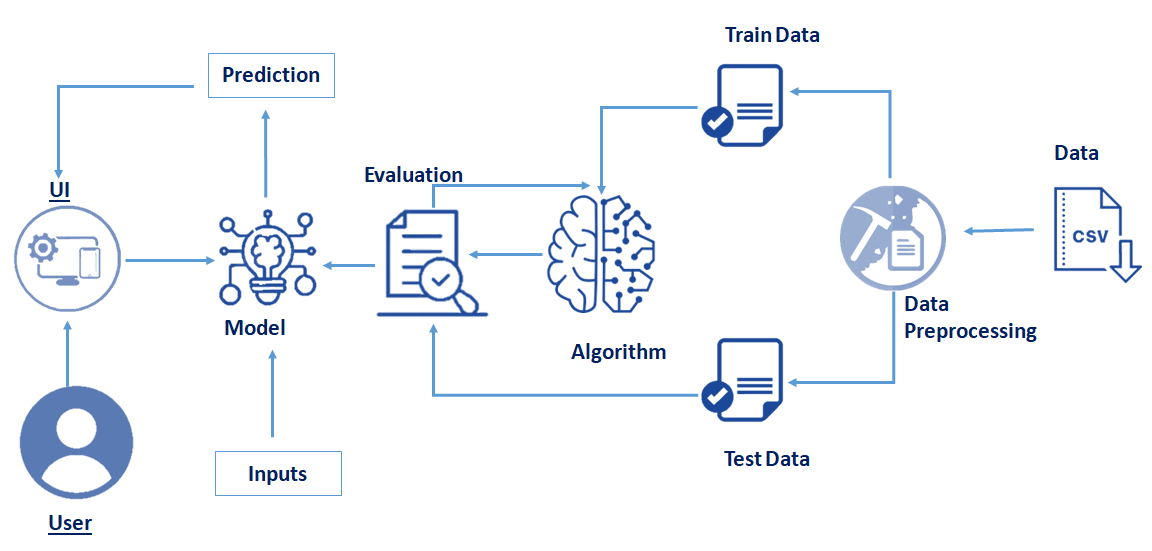
**2.2. Proposed solution:-**

 Facebook Prophet library is a third party library that could perform all the fine-tuning part within and we just need to feed the model. The main idea of our project is to create an application for forecasting the weather by using this library.

**3. THEORITICAL ANALYSIS:-**

**3.1. Block diagram:-**





**3.2. Hardware / software designing**

**Software Requirements:**

* OS – Windows XP,7,8,10
* Google Colab
* Spyder Software
* Anaconda Command Prompt

**Hardware Components:**

* Processor – i3
* Hard Disk Storage – 10 GB
* RAM – 1GB

**4. EXPERIMENTAL INVESTIGATIONS:-**

We are doing this research in order to bring best and accurate results. Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data.

Prophet is robust to missing data and shifts in the trend, and typically handles outliers well. Prophet is open source software released by Facebook’s Core Data Science team.

**5. FLOW CHART:-**

Collect the dataset

Data preprocessing

Model building

Application building

Predict the weather

**6. Results:-**

· Date, month and year are the inputs to be considered.

· The output is generated related to the analysis based on the input given to the predict the weather.

User interacts with the UI (User Interface) i.e the website built using flask to give the date as input. The date is analyzed by the model which is then integrated. Once model analyses the uploaded date, the prediction i.e the temperature in degree celcius is showcased on the UI.

**7. ADVANTAGES AND DISADVANTAGES:-**

**Advantages:-**

* The weather forecast can help to guide and encourage tourists to visit certain areas.
* People can schedule there plans based on the weather conditions.
* Very useful in agriculture
* Useful for everyone to know temperature on a particular date.
* It is a simple model i.e easy to build, test and understand.
* It is useful for making analysis of the future based on the past available data.

**Disadvantages:-**

* Forecasts are never 100% and it is almost impossible to predict the future with certainty.
* In some cases it may mislead people with wrong predictions.
* If we want to predict with 100% percent accuracy it becomes more cost.

**8. APPLICATIONS:-**

* **Severe weather alerts and advisories:**

A major part of modern weather forecasting is the severe weather alerts and advisories, which the national weather services issue in the case that severe or hazardous weather is expected. This is done to protect life and property.

* **Air Traffic:**

Because the aviation industry is especially sensitive to the weather, accurate weather forecasting is essential considering the fact that a greater number of plane crashes recorded the world over have weather related causes.

* **Agriculture:**

Farmers rely on weather forecasts to decide what work to do on any particular day. For example, drying hay is only feasible in dry weather. Prolonged periods of dryness can ruin cotton, wheat, and corn crops.

* **Military applications:**

Military weather forecasters present weather conditions to the war fighters, community. Military weather forecasters provide preflight weather briefs to pilots and provide real time resource protection services for military installations.

**9. CONCLUSION :-**

We have used prophet library it is accurate, fast and fully automatic. By taking date as a input we are predicting weather with great accuracy so it is easy for the people to known about weather on a particular date.

**10. FUTURE SCOPE :-**

Time series analysis for weather prediction using Prophet Library gives accurate weather prediction. Since weather is an important factor in our life. Every person day to day activities are decided based on the weather conditions. We are predicting weather with less cost and more accuracy it has more future scope and it is useful for the people.

**11. BIBILOGRAPHY:-**

Mohan, J. M. and Morgan, M. D. (1991). Meteorology: The Atmosphere and Science of Weather, Fourth edition. Macmillan Ontario, pp. 356 – 381.

Lutgens, F. K. and TarBuck, E. J. (1989). The Atmosphere: An Introduction to Meteorology, Fourth edition. Prentice Hall, New Jersey, pp. 299 – 331.

Ackerman, S. A. and Knox, J. A. (2003). Meteorology: Understanding the Atmosphere. Brooks/Cole USA, pp 362 – 379

https://machinelearningmastery.com/time-series-forecasting-with-prophet-in-python/

**APPENDIX**

**Source Code(Model Building):**

from fb prophet import Prophet

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

data=pd.read\_csv("/content/drive/MyDrive/testset.csv")

data.head()

data.tail()

data['datetime\_utc'] = pd.to\_datetime(data['datetime\_utc'])

data.set\_index('datetime\_utc',inplace=True)

data

data = data.resample('D').mean()

data

data = data[[' \_tempm']]

data.info()

data.isnull().any()

data.reset\_index(inplace=True)

data.reset\_index(inplace=True)

data.rename(columns = {'datetime\_utc':'ds',' \_tempm':'y'},inplace=True)

data.head()

data["ds"]

plt.plot(data["ds"],data["y"])

plt.xlabel("date")

plt.ylabel("Temperature")

plt.title("weather prediction")

plt.grid()

plt.figure(figsize=(10,8))

plt.plot(data.set\_index(["ds"]))

model = Prophet()

model.fit(data)

future = model.make\_future\_dataframe(periods=365,freq="D")

future.tail()

forecast = model.predict(future)

forecast.head()

forecast[['ds','yhat','yhat\_lower','yhat\_upper']].tail()

forecast[['ds','yhat','yhat\_lower','yhat\_upper']].head()

model.plot(forecast)

fig = model.plot\_components(forecast)

from fbprophet.diagnostics import cross\_validation

df\_cv = cross\_validation(model,initial='730 days',period='180 days',horizon='365 days')

df\_cv.head()

from fbprophet.diagnostics import performance\_metrics

df\_p = performance\_metrics(df\_cv)

df\_p.head()

from fbprophet.plot import plot\_cross\_validation\_metric

fig = plot\_cross\_validation\_metric(df\_cv,metric='rmse')

import pickle

pickle.dump(model,open("weather\_prediction.pickle","wb"))

**Source Code (app.py):**

import numpy as np

import pandas as pd

from flask import Flask,request,jsonify,render\_template

import pickle

app = Flask(\_\_name\_\_)

model = pickle.load(open('weather\_prediction.pickle','rb'))

@app.route('/')

def home():

return render\_template('home.html')

@app.route('/predict',methods=['POST'])

def predict():

if request.method == "POST":

ds = request.form["date"]

a = {"ds":[ds]}

ds = pd.DataFrame(a)

prediction = model.predict(ds)

output = round(prediction.iloc[0,18],2)

print(output)

return render\_template('home.html',output="Temperature on selected date is. {} degree celsius".format(output))

#return "<h1>Temperature on selected date is. {} degree celsius</h1>".format(output)

return render\_template("home.html")

if \_\_name\_\_ == "\_\_main\_\_":

app.run(port=5000,debug=False)

**HTML CODE:**

|  |
| --- |
| <!DOCTYPE html> |
|  | <html> |
|  | <head> |
|  | <title>Weather prediction</title> |
|  | <link rel="stylesheet" href="home.css"> |
|  | <link rel= "stylesheet" type= "text/css" href= "{{ url\_for('static',filename='css/home.css') }}"> |
|  | </head> |
|  | <body> |
|  | <div class="navbar"> |
|  | <p>Weather Prediction</p> |
|  | </div> |
|  | <div class="heading"> |
|  | <h1>Weather Forecast using Prophet Library</h1> |
|  | </div> |
|  | <div class="para"> |
|  | <p>Weather forecasting is the application of science and technology to predict the conditions of the atmosphere for a given time. People have attempted to predict the weather informally for millennia and formally since the 19th century.Here, we predict the accurate weather temperature in Celsius using prophet library. |
|  |  |
|  |  |
|  |  |
|  | </p> |
|  | </div> |
|  | <br> |
|  | <br> |
|  |  |
|  | <form method="POST" action="{{ url\_for('predict') }}" class="form" style="margin-top:30px"> |
|  | <div class="date" style="margin-left:100px;"> |
|  | <h3>Select date to predict weather</h3> |
|  | <input type="date" name="date" id="date" required="required"> |
|  | </div> |
|  | <div class="button"> |
|  | <button type="submit">submit</button> |
|  | </div> |
|  | </form> |
|  |  |
|  | </div> |
|  | <h2 style="margin-left:90px">{{ output }}</h2> |
|  | </div> |
|  | <body> |
|  | </html> |

