Weather Prediction

1 INTRODUCTION

1.1 Overview

Traditionally, weather predictions are performed with the help of large complex models of physics, which utilize different atmospheric conditions over a long period of time. These conditions are often unstable because of perturbations of the weather system, causing the models to provide inaccurate forecasts. The models are generally run-on hundreds of nodes in a large High-Performance Computing (HPC) environment which consumes a large amount of energy. In this project, we present a weather prediction technique that utilizes historical data from multiple weather stations to train simple machine learning models, which can provide usable forecasts about certain weather conditions for the near future within a very short period of time. The models can be run on much less resource intensive environments. The evaluation results show that the accuracy of the models is good enough to be used alongside the current state-of-the-art techniques.

1.2 Purpose

Physicists define climate as a “complex system”. While there are a lot of interpretations about it, in this specific case we can consider “complex” to be “unsolvable in analytical ways”. Since weather prediction is hard and more expensive. If the weather can be predicted at low cost and with more accuracy it will be useful.

Machine learning can be used to predict and it is more convenient and less expensive compared to traditional methods. And we can use the data provided by satellites or other sensors. The huge amount of data can be processed using a machine and predict the weather. The complex operation can be done using machine learning which will increase the accuracy.

Hence machine learning can be used to predict the weather with high accuracy.

2 LITERATURE SURVEY

2.1 Existing Problem

We use the data of weather data collected over years to predict the weather. Since the data are unprocessed it may contain useful and unuseful information. And certain values of data may be null. This causes the prediction to become inaccurate. If there is unuseful data it may cause resources to be wasted. To avoid it ,the data needs to undergo a certain process to clean itself.

Even after all this there are many models that we can use to predict the weather. So first we need to decide which model to use. And also need to find the accuracy.

2.2 Proposed solution

Since data that is used is unprocessed, it needs to Pre-processing and extract the data that are needed.

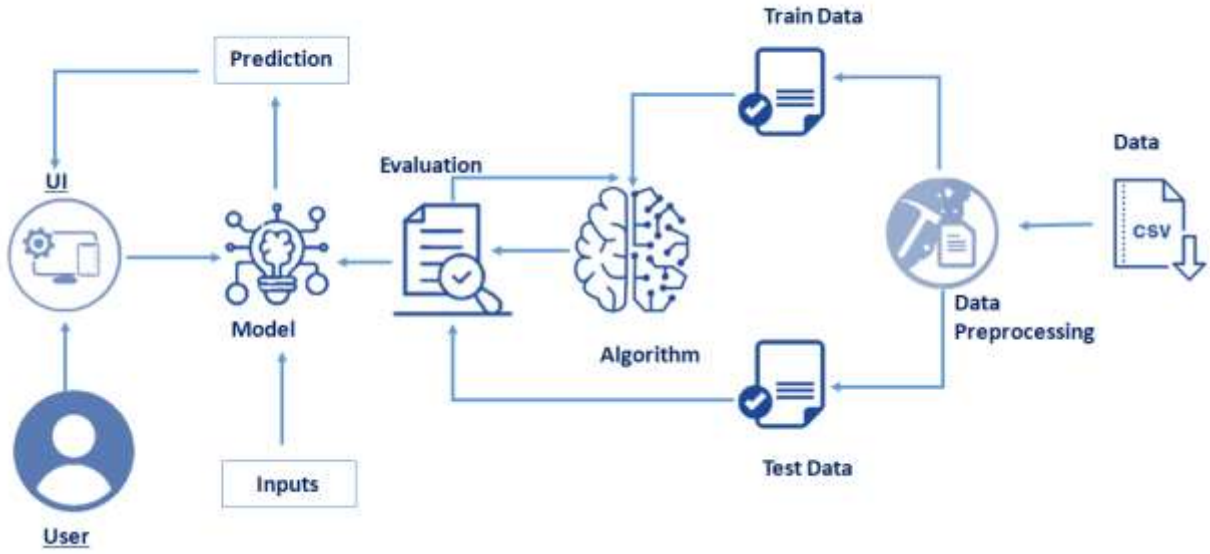
- Analyze the data
- Resampling the dataset
- Pre-processing the data
- Taking care of Missing Data
- Data visualization
- Splitting the Dataset into Dependent and Independent variable
- Splitting the data into Train and Test

After these process it will do following step:

- Training with different algorithms and testing the model
- Evaluation of Model
- Testing on random values
- Save the Model

3 THEORETICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software designing

Hardware Requirements

- RAM: 4 or above
- harddisk/SSD :250 or above

Software Requirement

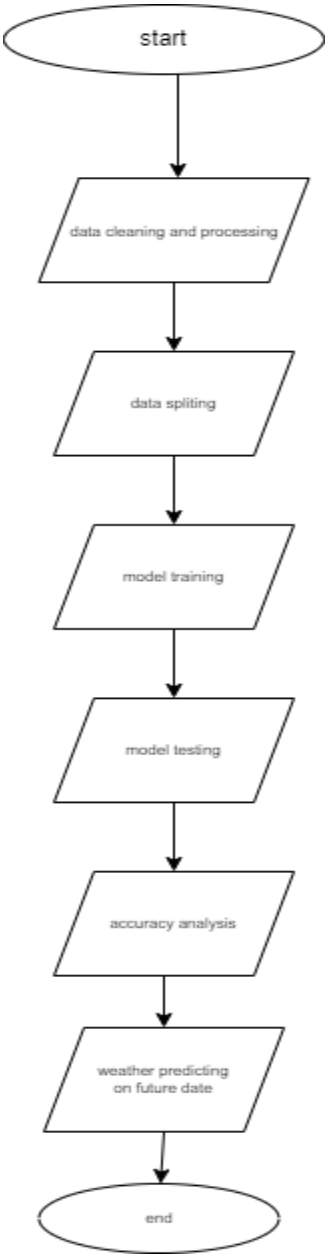
- OS :Window 8 or above
- Anaconda
- IBM cloud (watson studio)

4 .EXPERIMENTAL INVESTIGATIONS

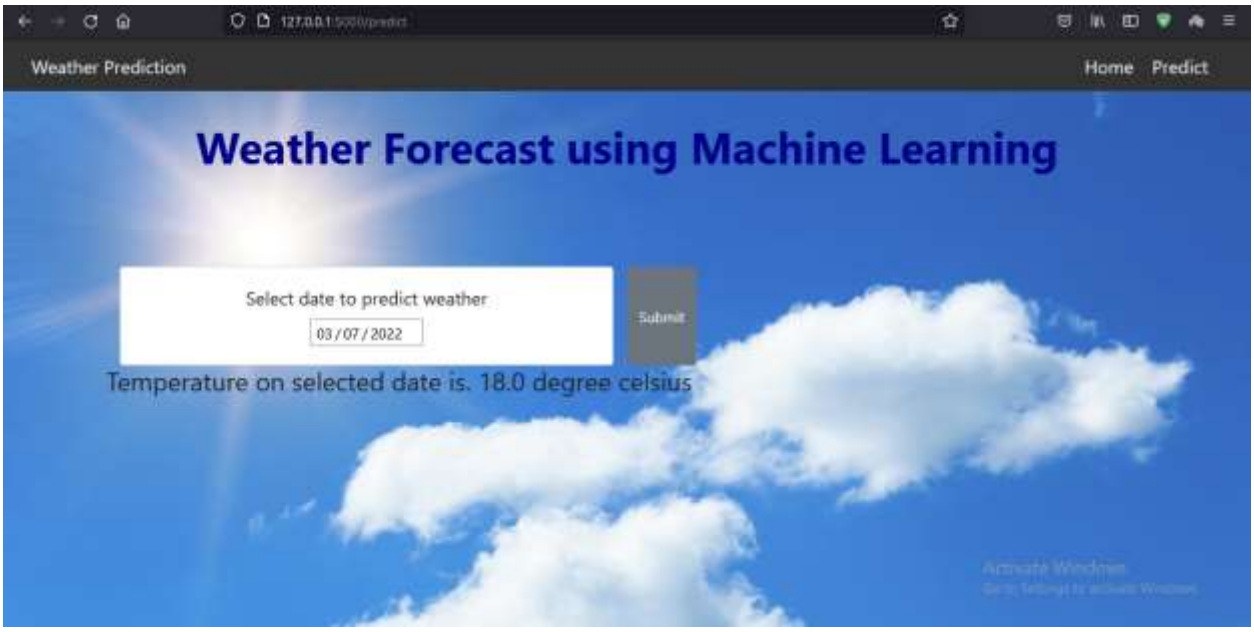
ML depends heavily on data, without data, it is impossible for an “AI” to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

During data clean the most challenging was to remove the null values. To fill the data that we need to use the mean of the column. Also finding the proper model is also a challenging task. After all that, it need to test the accuracy of the model and save the model.

5. FLOWCHART



6. RESULT



7. ADVANTAGES & DISADVANTAGES

- **Advantage**

1. Speed of prediction.
2. It is cheaper.
3. highly localized physics-free predictions using the data.

4. Trained ML model, it allows high-resolution weather forecasting.

- **Disadvantage**

1. Limited data availability for certain weather conditions
2. Differences in technology and hardware standards.

8.APPLICATIONS

- Air traffic
- Marine
- Agriculture
- Forestry
- Military applications

9.CONCLUSION

Weather predictions are performed with the help of large complex models of physics, which utilize different atmospheric conditions over a long period of time. These conditions are often unstable because of perturbations of the weather system, causing the models to provide inaccurate forecasts. In this project, we present a weather prediction technique that utilizes historical data from multiple weather stations to train simple machine learning models, which can provide usable forecasts about certain weather conditions for the near future within a very short period of time. With the help of this proposed system weather forecasting in to the future can be done in an easy and accurate manner

10 FUTURE SCOPE

In older days weather forecasting was done using physics model and other sort of things . These conditions are often unstable because of perturbations of the weather system, causing the models to provide inaccurate forecasts.this proposed system help in faster and accurate prediction of weather on the basis of previously occurred weather changes. Because the system is cheaper it can be highly used among variety of organization,it will be available for those who seeks it.This system can be used in the areas like military,agriculture,marine,forestry etc.

11 .BIBILOGRAPH

APPENDIX

A. Source Code

Jupyter code:

```
#importing necessary libraries
#from fbprophet import Prophet
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_0bf1b4ab9dc64d6190965509651bf46a = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='_KewsWcHCZ4yeDeiGb5RS7SBdV6th4B3ewyw16jHZmeu',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
```

```
        config=Config(signature_version='oauth'),
        endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

body =
client_0bf1b4ab9dc64d6190965509651bf46a.get_object(Bucket='weatherprediction-
donotdelete-pr-yhkrrjm7vpumpf',Key='weather_forecast.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

weather_data = pd.read_csv(body)
weather_data.head()

#•    head() method is used to return top n (5 by default) rows of a DataFrame or series.
weather_data.head()

#tail() method is used to return bottom n (5 by default) rows of a DataFrame or series.
weather_data.tail()

weather_data.dtypes

weather_data['datetime_utc'] = pd.to_datetime(weather_data['datetime_utc'])

weather_data.set_index('datetime_utc', inplace= True)
Weather_data

weather_data =weather_data.resample('D').mean()

weather_data = weather_data[[' _tempm' ]]

type(weather_data[[' _tempm']])

weather_data.info()

weather_data.isnull().any()

weather_data[' _tempm'].fillna(weather_data[' _tempm'].mean(), inplace=True) # we will
fill the null row

weather_data.head()
```

```
weather_data.reset_index(inplace=True)

weather_data.rename(columns = {'datetime_utc':'ds',' _tempm':'y'}, inplace = True)

 weather_data.head()

plt.figure(figsize=(12,8))
plt.plot(weather_data.set_index(["ds"]))

weather_data['year'] = pd.DatetimeIndex(weather_data['ds']).year
weather_data['month'] = pd.DatetimeIndex(weather_data['ds']).month
weather_data['day'] = pd.DatetimeIndex(weather_data['ds']).day

weather_data.drop('ds', axis=1, inplace=True)
model = Prophet()
model.fit(weather_data)

x=weather_data.iloc[:,1:4].values #inputs
y=weather_data.iloc[:,0:1].values #output price only

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,
                    test_size=0.2,random_state=0)

x_train.shape

x_test.shape

from sklearn.tree import DecisionTreeRegressor
dtr=DecisionTreeRegressor()
#fitting the model or training the model
dtr.fit(x_train,y_train)

y_pred=dtr.predict(x_test)
Y_pred

from sklearn.metrics import r2_score
dtraccuracy=r2_score(y_test,y_pred)
Dtraccuracy
```

```
y_p= dtr.predict([[2005,1,23]])
y_p

import pickle
pickle.dump(dtr,open('weather.pkl','wb'))

!pip install -U ibm-watson-machine-learning

from ibm_watson_machine_learning import APIClient
import json

wml_credentials = {
        "apikey":"Y1PZtY26CqnT7buS4DHWpxhGwReOd29HZSvYiJ0y3jGp",
        "url":"https://us-south.ml.cloud.ibm.com"
}

wml_client = APIClient(wml_credentials)
wml_client.spaces.list()

SPACE_ID="6ef4de3a-01aa-42b5-80cf-5770e6b77759"

wml_client.set.default_space(SPACE_ID)

wml_client.software_specifications.list()

MODEL_NAME = 'WeatherforecastModel'
DEPLOYMENT_NAME = 'weather_forecast_deploy'
WF_MODEL = dtr

software_spec_uid =
wml_client.software_specifications.get_id_by_name('default_py3.8')

# Setup model meta
model_props = {
        wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
        wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn_0.23',
        wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:
software_spec_uid
}
```

```
model_details = wml_client.repository.store_model(
        model=WF_MODEL,
        meta_props=model_props,
        training_data=x_train,
        training_target=y_train
)
```

App_ibm.py

```
import numpy as np
import pandas as pd
from flask import Flask, request, jsonify, render_template
#import pickle
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your
IBM Cloud account.
API_KEY = "Y1PZtY26CqnT7buS4DHWpxhGwReOd29HZSvYiJ0y3jGp"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey": API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

app = Flask(__name__)#our flask app
#model = pickle.load(open('weather_prediction.pickle', 'rb')) #loading the model

@app.route('/')
def home():
        return render_template('home.html')#rendering html page

@app.route('/pred')
def index():
        return render_template('index.html')#rendering prediction page

@app.route('/predict',methods=['POST'])
def y_predict():
        if request.method == "POST":
        ds = request.form["Date"]
        #Converting date input to a dataframe
```

```
        a={"ds":[ds]}
        ds=pd.DataFrame(a)
        ds['year'] = pd.DatetimeIndex(ds['ds']).year
        ds['month'] = pd.DatetimeIndex(ds['ds']).month
        ds['day'] = pd.DatetimeIndex(ds['ds']).day
        ds.drop('ds', axis=1, inplace=True)
        ds=ds.values.tolist()
        payload_scoring = {"input_data": [{"fields": [["year", "month","date"]], "values": ds}]}
        #payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values": [array_of_values_to_be_scored, another_array_of_values_to_be_scored]}]}
        response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/54f8f5b6-3db8-4031-a8ef-933867c71746/predictions?version=2022-03-05', json=payload_scoring, headers={'Authorization': 'Bearer ' + mltoken})
        print("Scoring response")
        pred= response_scoring.json()
        print(pred)
        output= pred['predictions'][0]['values'][0][0]
        print(output)

        return render_template('index.html',prediction_text="Temperature on selected date is. {} degree celsius".format(output))
        return render_template("index.html")

if __name__ == "__main__":
        app.run(debug=False)
```