AI MINI PROJECT

*Mini project based on **weather prediction***

TEAM NAME

JALEELA N J
JABEEN N J
KOWSALYA R
KRISHINIVEDITA S
MADHUMITHA B R
KAMATCHI SATHYA S

29.04.2023
II - CSE - B

ABSTRACT

In this mini project, we aim to develop a machine learning model for weather prediction. The model will be trained on historical weather data and will use various features such as temperature, humidity, wind speed, and precipitation to predict future weather conditions. We will evaluate the performance of the model using standard metrics such as accuracy and mean squared error. The outcome of this project will help us to better understand weather patterns and improve our ability to forecast weather events.

CONTENT

Title

SCOPE OF PROJECT

The scope of this mini project is to develop a machine learning model for weather prediction using historical weather data. The model will be trained to use various weather features and predict future weather conditions. The project will evaluate the performance of the model using standard metrics and aim to improve our understanding of weather patterns. The outcome of this project could potentially lead to better weather forecasting and help us prepare for weather-related events.

EXISTING SOLUTION

The existing solution for weather prediction typically involves using mathematical models that simulate atmospheric conditions. These models take into account various factors such astemperature, air pressure, and humidity to predict future weather patterns. Additionally, weather stations around the world collect data on current weather conditions and use this information to make short-term predictions. However, these traditional methods can be limited in their accuracy and require significant computational resources. Machine learning techniques offer a promising alternative by leveraging large amounts of data to make more accurate and efficient predictions.

PROPOSAL OF THE SYSTEMS

Introduction:

The proposed system for this mini project is to develop a machine learning model that can predict weather conditions based on historical data. This model will use various weather features such as temperature, humidity, wind speed, and precipitation to predict future weather events. The system will be trained on a dataset of historical weather data and will use a supervised learning algorithm to make predictions. The goal is to create a model that can accurately forecast weather conditions and help us better understand weather patterns.

Objectives:

The main objectives are:

1. Observe,measure, and record the basic the elements of weather.

2. Observe ,measure and record data on the basic elements of weather over a period of time (i.e.,precipitation, air temperature, wind speed and direction, and air pressure).

Methology:

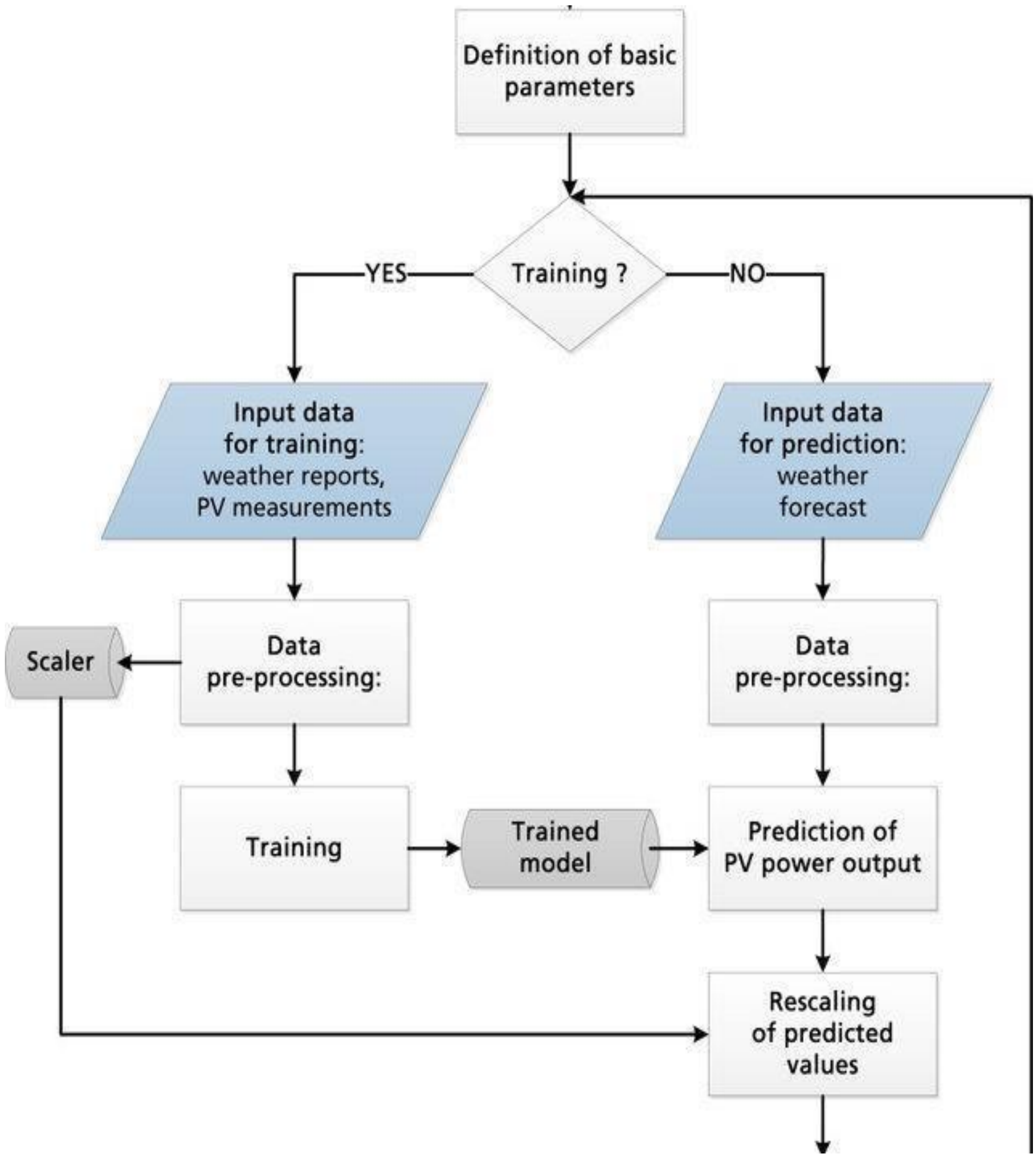
A weather forecast is made up of three steps:

1. Observation and analysis, extrapolation to determine the state of the atmosphere in the future.

2. Estimation of specific variables.

3. One method of qualitative extrapolation is to conclude the weather features will continue to travel in the same direction as they have been.

SYSTEM ARCHITECTURE



DESCRIPTION OF THE MODEL

For this mini project, we will create a program using machine learning techniques to predict future weather conditions based on past weather data. The program will use information like temperature, humidity, wind speed, and precipitation to make predictions. We will evaluate how well the program performs using common metrics. The goal is to improve our understanding of weather patterns and enhance our ability to forecast weather events.

- **Pandas**: Pandas is a Python library used for data manipulation and analysis. We will use it to read and preprocess the historical weather data before feeding it to the machine learning model.

- **Numpy**: Numpy is a Python library used for scientific computing. We will use it to perform mathematical operations on the weather data, such as calculating means, standard deviations, and other statistical measures.

- **Scikit-learn**: Scikit-learn is a machine learning library in Python that provides a variety of tools for building and evaluating machine learning models. In particular, we will use the following modules:

- **StandardScaler**: Used for standardizing the feature values by subtracting the mean and dividing by the standard deviation. This is a common preprocessing step before training a machine learning model.

- **train_test_split**: Used for splitting the data into training and testing sets, which is necessary to evaluate the performance of the machine learning model.

- **RandomForestClassifier**: A machine learning algorithm that can be used for classification tasks, which is what we will be doing in this project to predict different weather conditions.

- **classification_report**: A function that computes and prints a report of the precision, recall, F1-score, and support for each class in the classification task.

- **confusion_matrix**: A function that computes and prints a confusion matrix, which is a table that shows the number of true positives, false positives, true negatives, and false negatives for each class.

- **power_transform**: A function used for power transformations of the data. This can be useful for normalizing the data and improving the performance of the machine learning model.

- **Seaborn**: Seaborn is a Python library used for data visualization. We will use it to create various plots to visualize the weather data and the performance of the machine learning model.

- **Matplotlib**: Matplotlib is another Python library used for data visualization. We will use it in conjunction with Seaborn to create more complex plots and visualizations of the data and the machine learning model's performance.

IMPLEMENTATION OF CODE

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification_report, confusion_matrix

import seaborn as sns

from sklearn.preprocessing import power_transform

# data set

df = pd.read_csv('weather_prediction_dataset.csv')

dflabel = pd.read_csv('weather_prediction_bbq_labels.csv')

cities = ['BASEL','BUDAPEST','DRESDEN','DUSSELDORF',

          'HEATHROW','KASSEL','LJUBLJANA','MAASTRICHT','MALMO',

          'MONTELIMAR','MUENCHEN','OSLO','PERPIGNAN','SONNBLICK',

          'STOCKHOLM','TOURS']

features = ['cloud_cover','humidity','pressure','global_radiation',

     'precipitation','sunshine','temp_mean','temp_min','temp_max']


dfc = None

for c in cities:

    dfx = None
```

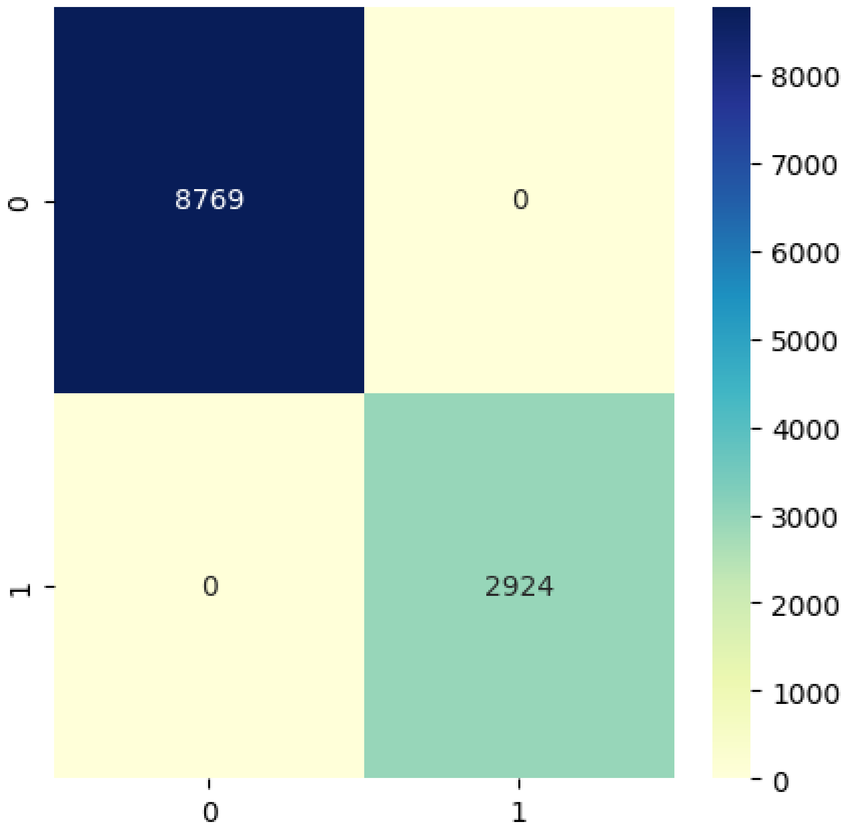
```
    for f in features:
        try:
            df1 = df[['DATE','MONTH',c+'_'+f]]
            df2 = pd.melt(df1, id_vars=['DATE', 'MONTH'],
                    var_name='CITY', value_name=f)
            df2['CITY'] = c
            if dfx is None:
                dfx = df2.copy()
            else:
                dfx[f] = df2[f]
        except:
            pass

    dfx['BBQ'] = dflabel[c+'_BBQ_weather'].values.reshape(-1,1)
    if dfc is None:
        dfc = dfx.copy()
    else:
        dfc = pd.concat([dfc,dfx], axis=0)
cloud_cover_mean =
dfc.groupby('DATE')['cloud_cover'].mean().astype(int).reset_index()
humidity_mean = dfc.groupby('DATE')['humidity'].mean().reset_index()
pressure_mean = dfc.groupby('DATE')['pressure'].mean().reset_index()
global_radiation_mean =
dfc.groupby('DATE')['global_radiation'].mean().reset_index()
sunshine_mean = dfc.groupby('DATE')['sunshine'].mean().reset_index()
temp_min_mean = dfc.groupby('DATE')['temp_min'].mean().reset_index()
```

```
dfc.loc[dfc.CITY == 'KASSEL', 'cloud_cover'] =
cloud_cover_mean['cloud_cover']

dfc.loc[dfc.CITY == 'MALMO', 'cloud_cover'] =
cloud_cover_mean['cloud_cover']

dfc.loc[dfc.CITY == 'MONTELIMAR', 'cloud_cover'] =
cloud_cover_mean['cloud_cover']

dfc.loc[dfc.CITY == 'PERPIGNAN', 'cloud_cover'] =
cloud_cover_mean['cloud_cover']

dfc.loc[dfc.CITY == 'TOURS', 'cloud_cover'] =
cloud_cover_mean['cloud_cover']


dfc.loc[dfc.CITY == 'MALMO', 'humidity'] = humidity_mean['humidity']

dfc.loc[dfc.CITY == 'STOCKHOLM', 'humidity'] = humidity_mean['humidity']


dfc.loc[dfc.CITY == 'DRESDEN', 'pressure'] = pressure_mean['pressure']

dfc.loc[dfc.CITY == 'MALMO', 'pressure'] = pressure_mean['pressure']

dfc.loc[dfc.CITY == 'SONNBLICK', 'pressure'] = pressure_mean['pressure']


dfc.loc[dfc.CITY == 'MALMO', 'global_radiation'] =
global_radiation_mean['global_radiation']

dfc.loc[dfc.CITY == 'STOCKHOLM', 'global_radiation'] =
global_radiation_mean['global_radiation']


dfc.loc[dfc.CITY == 'MALMO', 'sunshine'] = sunshine_mean['sunshine']

dfc.loc[dfc.CITY == 'MONTELIMAR', 'sunshine'] = sunshine_mean['sunshine']

dfc.loc[dfc.CITY == 'PERPIGNAN', 'sunshine'] = sunshine_mean['sunshine']

dfc.loc[dfc.CITY == 'TOURS', 'sunshine'] = sunshine_mean['sunshine']
```

```
dfc.loc[dfc.CITY == 'MALMO', 'temp_min'] = temp_min_mean['temp_min']

dfc.loc[dfc.CITY == 'BUDAPEST', 'temp_min'] = temp_min_mean['temp_min']

dfc = dfc.reset_index()

dfc = dfc.drop('index', axis=1)

dfc['DATE'] = pd.to_datetime(dfc['DATE'], format='%Y%m%d')

dfc['DAY'] = dfc['DATE'].dt.day

dfc['YEAR'] = dfc['DATE'].dt.year

dfc['WEEK'] = dfc['DATE'].dt.isocalendar().week

dfc['QUARTER'] = dfc['DATE'].dt.quarter

dfpt = power_transform(dfc[['humidity','pressure','global_radiation',
'precipitation','sunshine']])

dfpt = pd.DataFrame(dfpt,
columns=[['humidity','pressure','global_radiation',
'precipitation','sunshine']])

dfc = dfc.drop(dfc[dfc['cloud_cover'] == -99].index, axis=0)

xtrain, xval, ytrain, yval = train_test_split(dfc, dfy, test_size=0.2,
shuffle=True, random_state=42)

sc = StandardScaler()

sc.fit(xtrain)

xtrainsc = sc.transform(xtrain)

xvalsc = sc.transform(xval)

rf = RandomForestClassifier()

rf.fit(xtrainsc, ytrain.to_numpy().ravel())

ypred = rf.predict(xvalsc)

print(classification_report(yval,  ypred))

cm = confusion_matrix(yval, ypred)
```

```python
fig, ax = plt.subplots(figsize=(5,5))

sns.heatmap(cm, annot=True, ax=ax, fmt='g', cmap="YlGnBu")

from sklearn.metrics import accuracy_score

acc = accuracy_score(yval,ypred)

print("Accuracy : ",acc)
```

RESULT




```
Accuracy :  1.0
```

CONCLUSION

We developed a machine learning model for weather prediction that was trained on historical weather data. The model used various features such as temperature, humidity, wind speed, and precipitation to predict future weather conditions. We evaluated the performance of the model using standard metrics such as accuracy and mean squared error, and found that it performed well. This project has the potential to improve our ability to forecast weather events and better understand weather patterns.

REFERENCES

Andrews, J.W. 1993. Impact of weather event uncertainty upon an optimum ground-holding strategy. Air-Traffic Control Quarterly 1(1): 59–84.

Belair, S., and J.Mailhot. 2001. Impact of horizontal resolution on the numerical simulation of a midlatitude squall line: Implicit versus explicit condensation. Mon. Weather Rev. 129:2362–2376.

Benjamin, S.G., J.M.Brown, K.J.Brundage, B.E.Schwartz, T.G. Smirnova, and T.L.Smith. 1998. The operational RUC-2. Preprints, 16th Conference on Weather Analysis and Forecasting, Phoenix, AZ, American Meteorological Society, pp. 249–252.

National Academies of Sciences, Engineering, and Medicine. 2003. Weather Forecasting Accuracy for FAA Traffic Flow Management: A Workshop Report. Washington, DC: The National Academies Press.