Hands-on Data Science

WEATHER TYPE CLASSIFICATION

YUCHENG MA

School of Computer Science

10/25/2024

https://github.com/markfromcd/Weather-Type-Classification.git

1. INTRO

INTRO: A CLASSIFICATION PROBLEM

The problem: Given a set of weather-related features such as **temperature**, **humidity**, **wind speed**, **precipitation percentage**, **cloud cover**, **atmospheric pressure**, **UV index**, **season**, **visibility**, and **location**, the objective is to predict the **weather type** (e.g., Rainy, Cloudy, Sunny, Snowy). This is a **multiclass classification problem**, where each weather type is a discrete category.

Objective: **To build a machine learning model that can accurately predict the weather type based on the given input features.** I will use the dataset to train the model, evaluate its performance, and use it to classify unseen data into one of the possible weather types.

Why important: Weather predictions **impact a wide range of industries and everyday life decisions**. Like: Real-world applicability, safety and disaster preparedness, environmental protection. In details: agriculture, transportation, retail, early warnings...

Source: https://www.kaggle.com/datasets/nikhil7280/weather-type-classification/data

PLEASE NOTE

- **This dataset is synthetically produced and does not convey real-world weather data.** It includes intentional outliers to provide opportunities for practicing outlier detection and handling. The values, ranges, and distributions may not accurately represent real-world conditions, and the data should primarily be used for educational and experimental purposes.

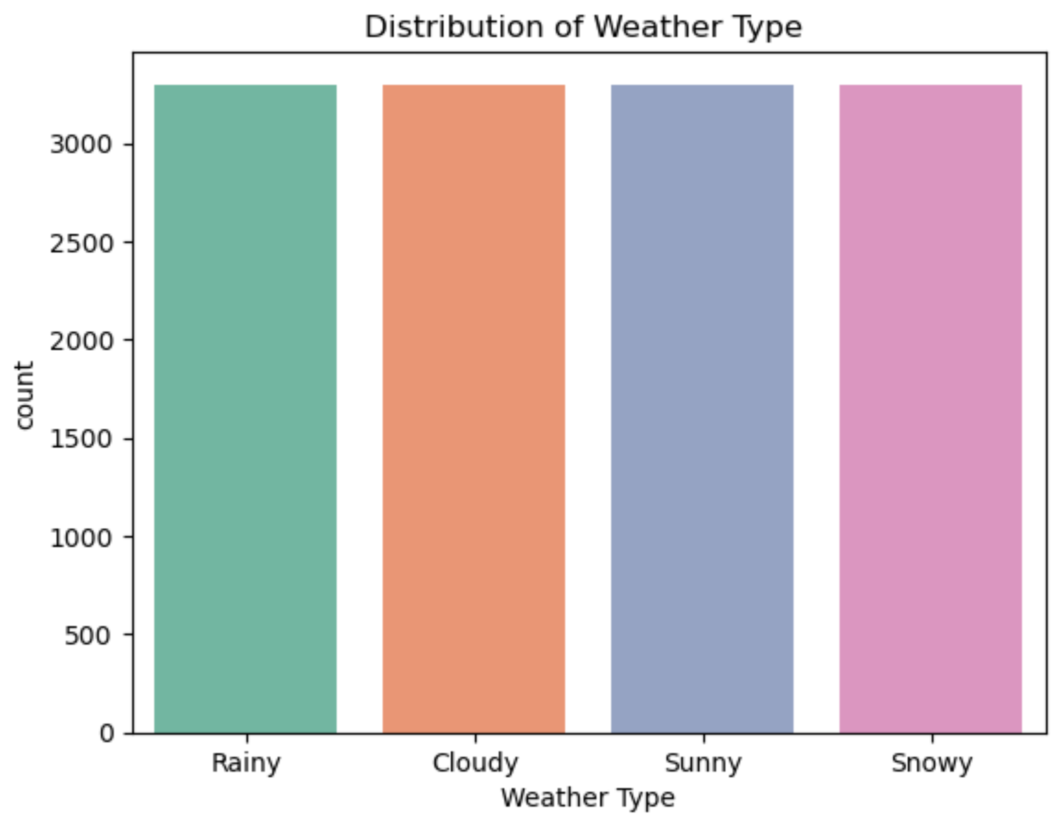
2. EDA

1. DATA

- **Target data distribution:**
- **Nicely balanced.**

- **13,200 data points in total,**
- **11 columns**

- **11 indexes:** **temperature, humidity**, **wind speed**, **precipitation percentage**, **cloud cover**, **atmospheric pressure**, **UV index**, **season**, **visibility**, and **location**



2. DATA TYPE

Contains both continuous and categorical variables

```python
#numerical_columns & categorical_columns
cat = df.select_dtypes(include=object).columns.tolist()
num = df.select_dtypes(exclude=object).columns.tolist()
print(' categorical_columns: \n' ,cat)
print('\n numerical_columns: \n' , num)
```

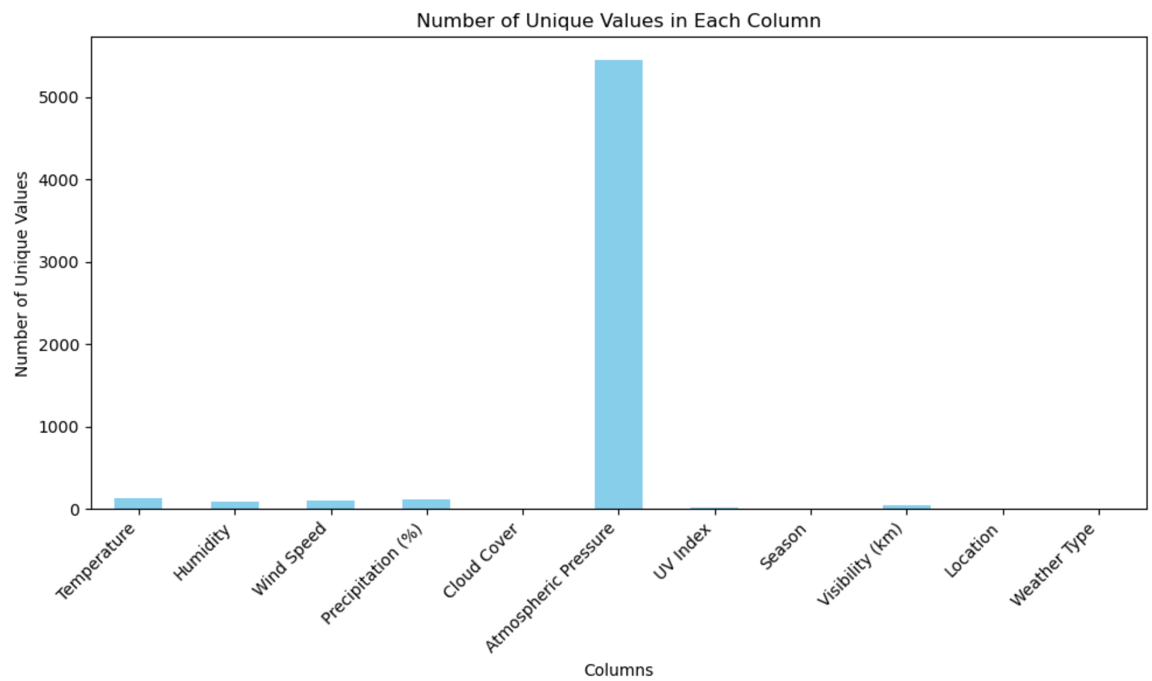
✓ 0.0s Python

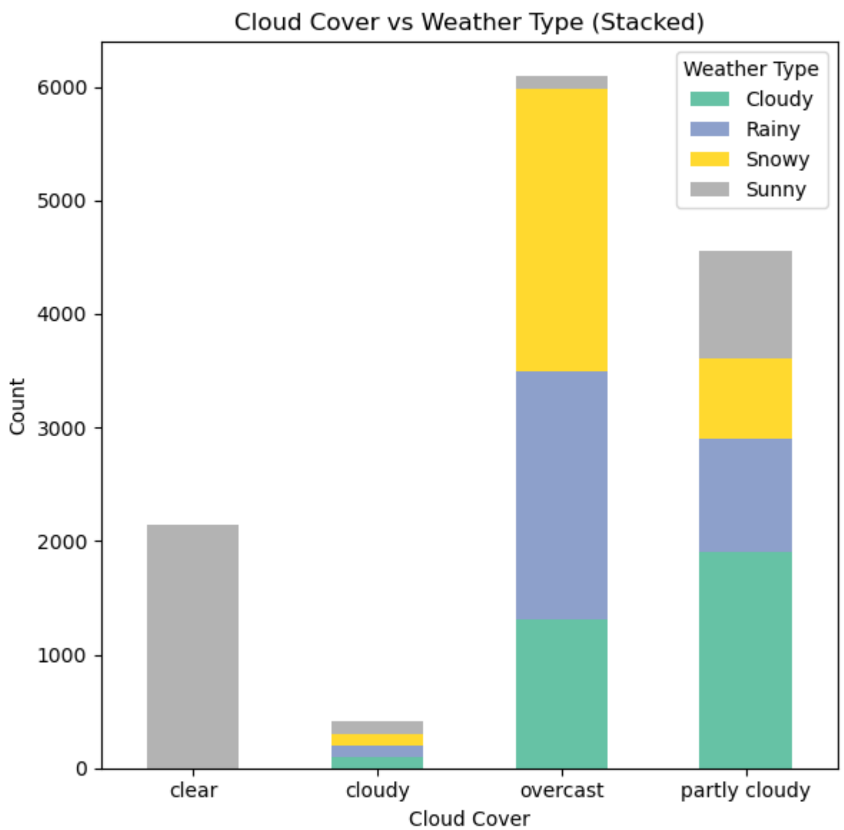
```
categorical_columns:
['Cloud Cover', 'Season', 'Location', 'Weather Type']

numerical_columns:
['Temperature', 'Humidity', 'Wind Speed', 'Precipitation (%)', 'Atmospheric
```

3. CHECK THE NUNIQUE VALUE OF EACH COLUMN

Only atmospheric pressure has a lot values, but it really plays a vital role in weather prediction, so I **will not drop it.**

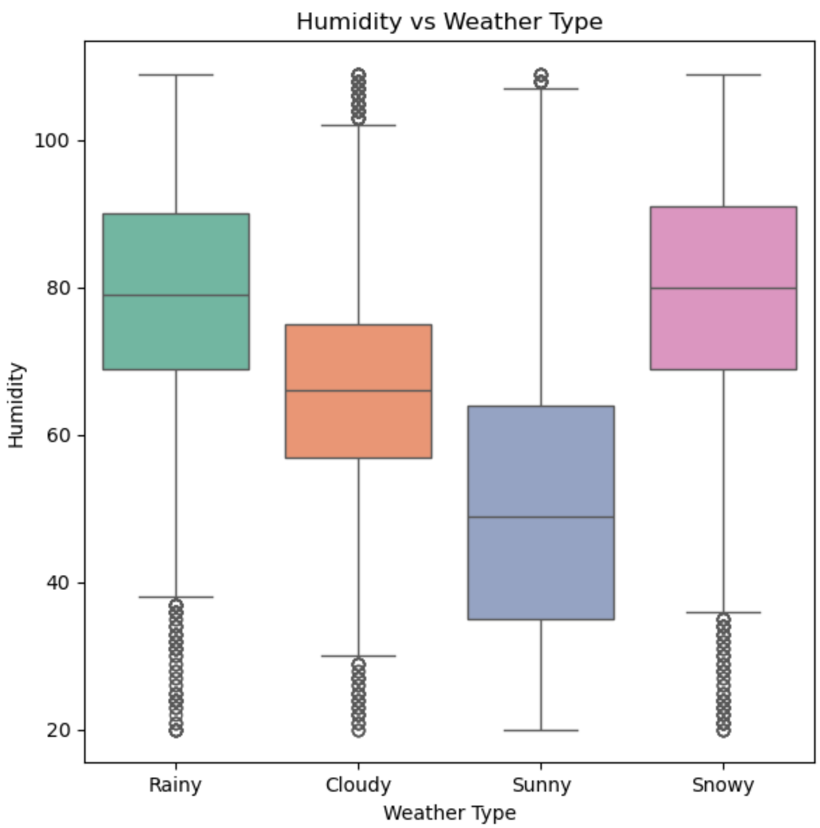




4. CATEGORICAL VALUES PLAY ROLES IN MODEL

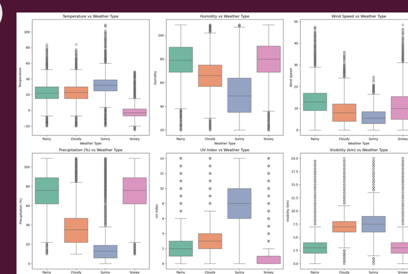
DIFFERENT COLUMN HAS DIFFERENT DISTRIBUTION

(A EXAMPLE HERE)



5. 6 CONTINUOUS VALUES ARE USEFUL

THE DISTRIBUTIONS OF BOX PLOT ARE DIFFERENT. ONLY WIND SPEED IS SLIGHTLY UNIMPORTANT. (A EXAMPLE HERE)



3. SPLITTING

STRATIFIED SHUFFLE SPLIT

```
Training set actual size: 7920
Validation set actual size: 2640
Test set actual size: 2640
X_train head:
       Temperature  Humidity  Wind Speed  Precipitation (%)    Cloud Cover  \
80            27.0        73         9.5               47.0  partly cloudy
3456          21.0        64         8.5               80.0       overcast
1416          26.0        39         9.5                6.0          clear
2256          -8.0        90         7.0               78.0       overcast
7335          14.0        75        12.5               33.0  partly cloudy

      UV Index  Season  Visibility (km) Location
80           1  Summer              7.0  coastal
3456         0  Spring              2.0   inland
1416         7  Spring              9.0  coastal
2256         1  Winter              4.0   inland
7335         4  Winter             10.5   inland
X_val head:
        Temperature  Humidity  Wind Speed  Precipitation (%)    Cloud Cover
10671          14.0        76        13.5               35.0  partly cloudy
10892          -7.0        82         0.5               72.0       overcast
1801           27.0        21         5.5               15.0  partly cloudy
5721           32.0        84        15.0               84.0       overcast
7970          -10.0        64         5.0               87.0       overcast

       UV Index  Season  Visibility (km)  Location
10671         3  Spring              5.5  mountain
10892         1  Winter              3.5    inland
1801         11  Autumn              8.0   coastal
5721          3  Spring              1.0   coastal
```

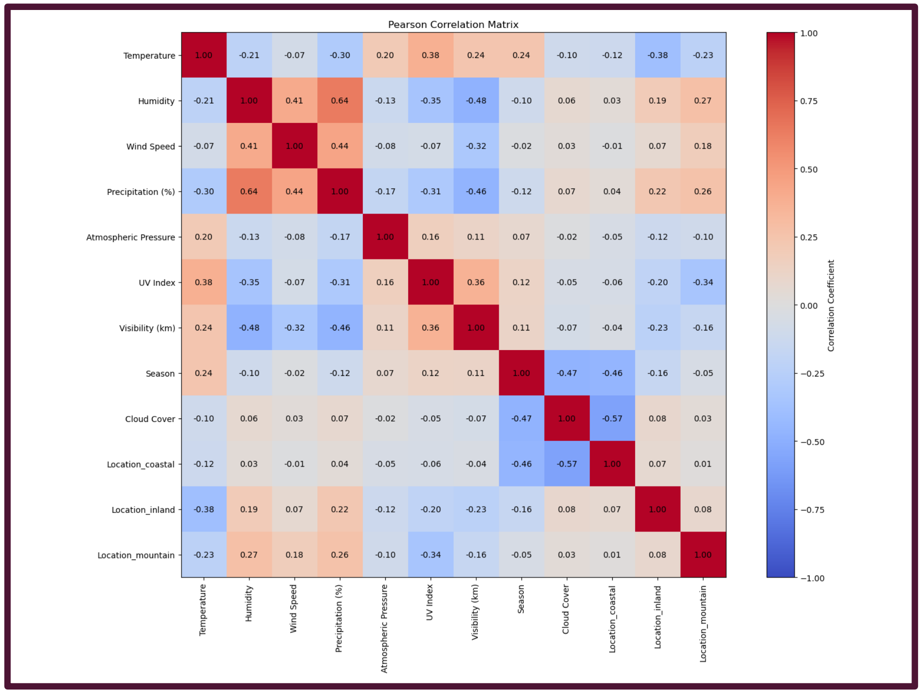
- Since the target data is nicely balanced, using it is to ensure that the class proportions (the balance between categories in the target variable) remain consistent across the **training**, **validation**, and **test sets**.
- Training set: 60%, Validation set: 20%, Test set: 20%

4. PREPROCESSING

PREPROCESSORS THAT I USED

- I used StandardScaler to ensures that each continuous feature has a **mean of 0** and a **standard deviation of 1** so that to **make model better perform.**
- I used OneHotEncoder to transform **Location feature**, **allowing models to interpret categorical features correctly**, treating them as distinct entities rather than imposing a numerical relationship. This ensures the model **doesn't misinterpret categorical data**.
- I used OrdinalEncoder to transform **Season and Cloud Cover** since they may contain inherent orders. OneHotEncoding will make them independent.
- So, help **have better performance, avoids biases due to feature scaling, interpret categorical data correctly and even help convergence.**

FEATURES' DIFFERENCE AFTER PREPROCESSING



- **By checking the correlation matrix**, no more feature needed to be removed.
- Since **no missing value**, no rows needed to be replaced or removed.
- Since used one-hot encoding for categorical columns, **the features from 10 used to 12 used.** (temperature, humidity, wind speed, precipitation percentage, cloud cover, UV index, season, visibility **plus** location's sub-categories.)
- Also, I changed **target value from words to ordinal encoding.**

THANKS FOR LISTENING