Hurricane Milton

Weather Analysis

DS 4/544 - Final Project

12/02/2024

Sang Xing

Table of Contents

PART 1
Introduction

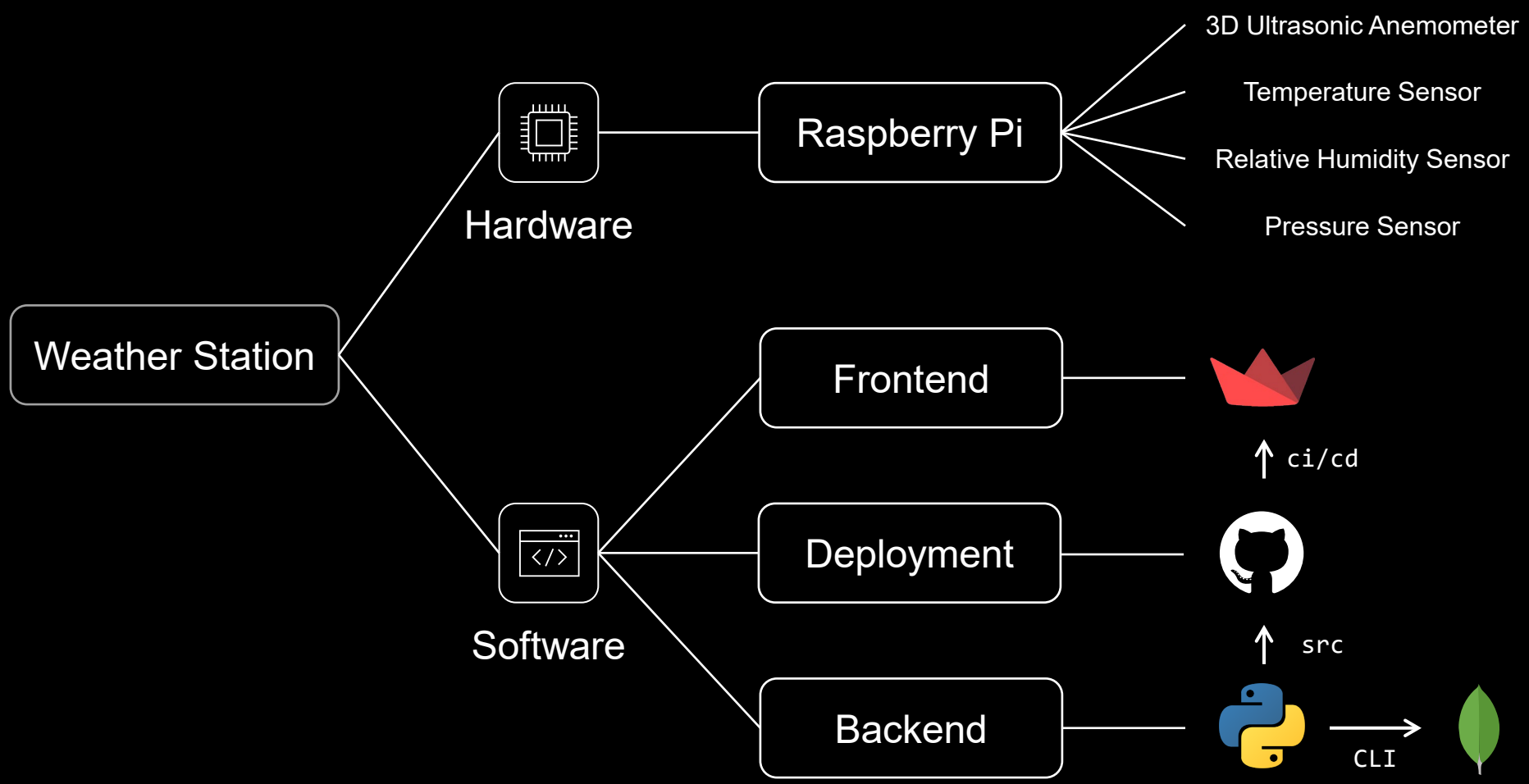
- Problem Statement
- Technology Stack
- Project Goal

Problem Statement

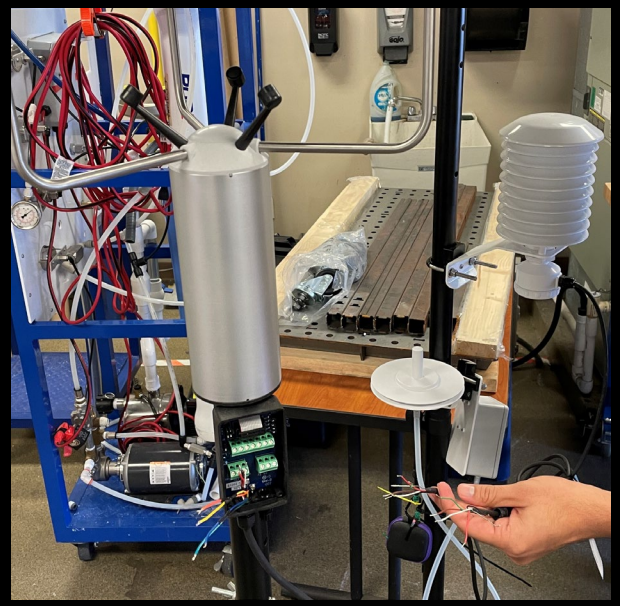
1. What environmental conditions are affected during a hurricane?

2. What environmental factors can be used to classify hurricane periods?

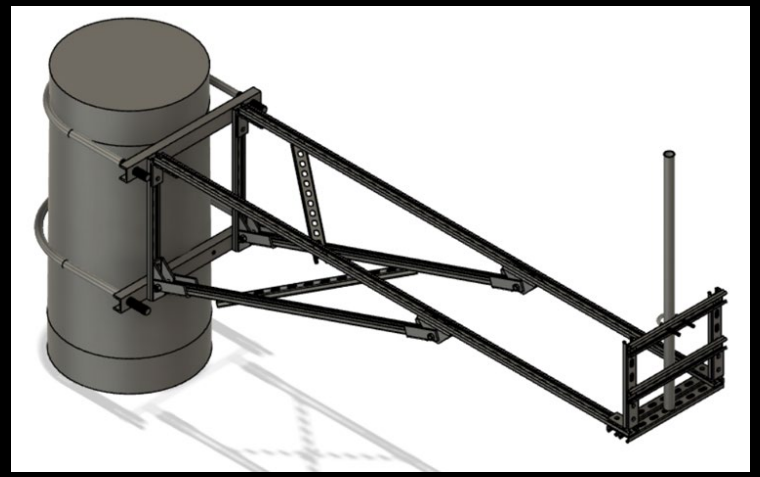
Technology Stack

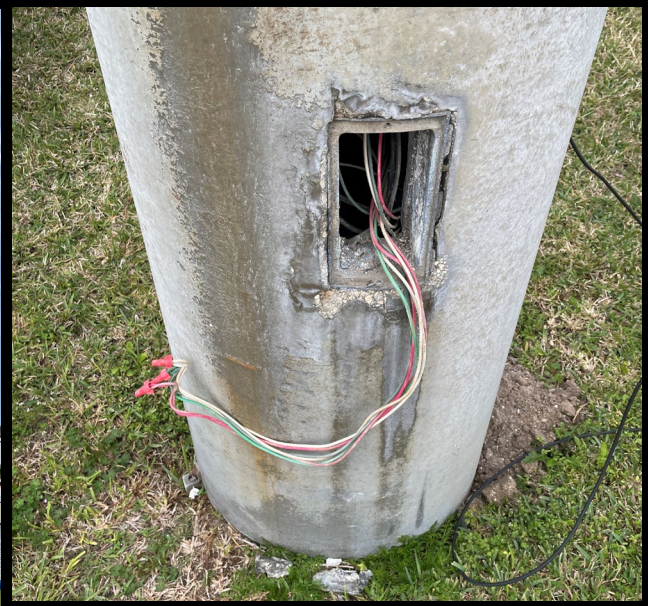


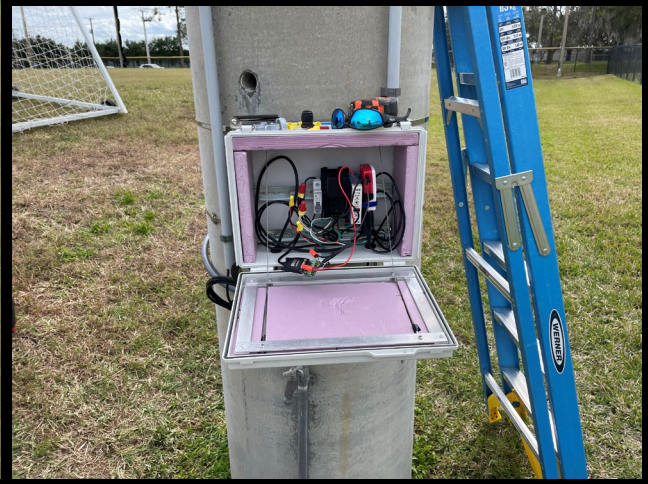
Hardware



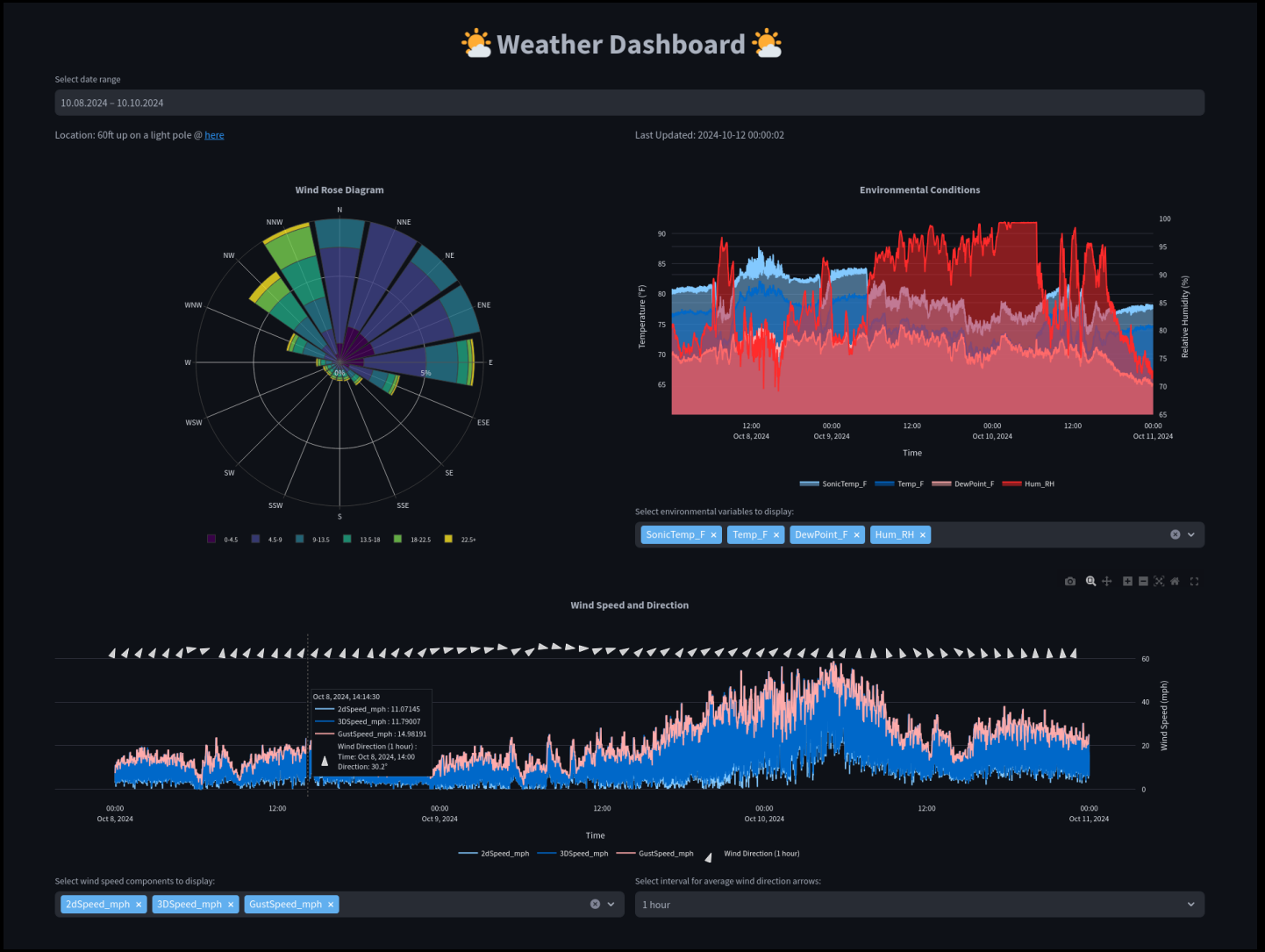
Hardware







Project Goal



Interactive Dashboard @

https://hurricane-milton.streamlit.app/

Open-sourced Repo @

https://github.com/Sang-Buster/weather-dashboard

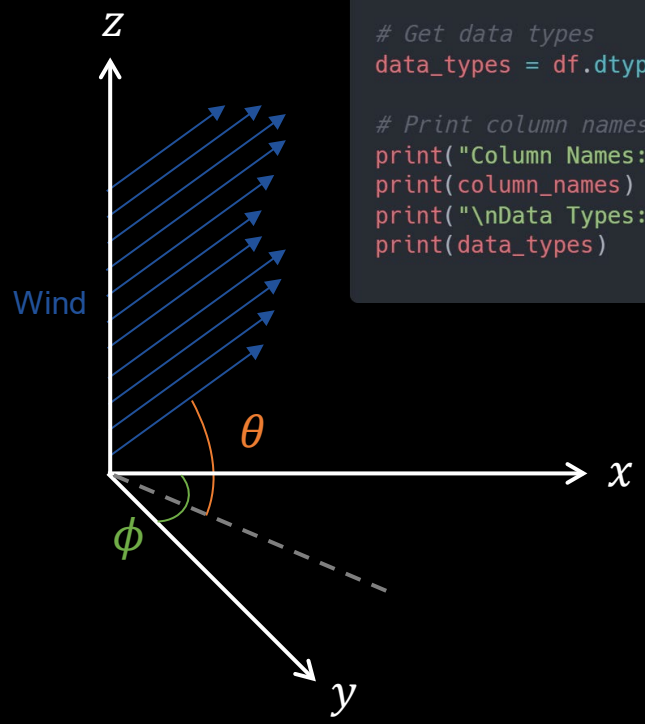
PART 2

Exploration

- Exploratory Data Analysis (EDA)
- Principal Component Analysis (PCA)

EDA – Data Attributes

```python
import pandas as pd

# Read the CSV file
df = pd.read_csv('weather_data.csv')

# Get column names
column_names = df.columns.tolist()

# Get data types
data_types = df.dtypes

# Print column names and data types
print("Column Names:")
print(column_names)
print("\nData Types:")
print(data_types)
```



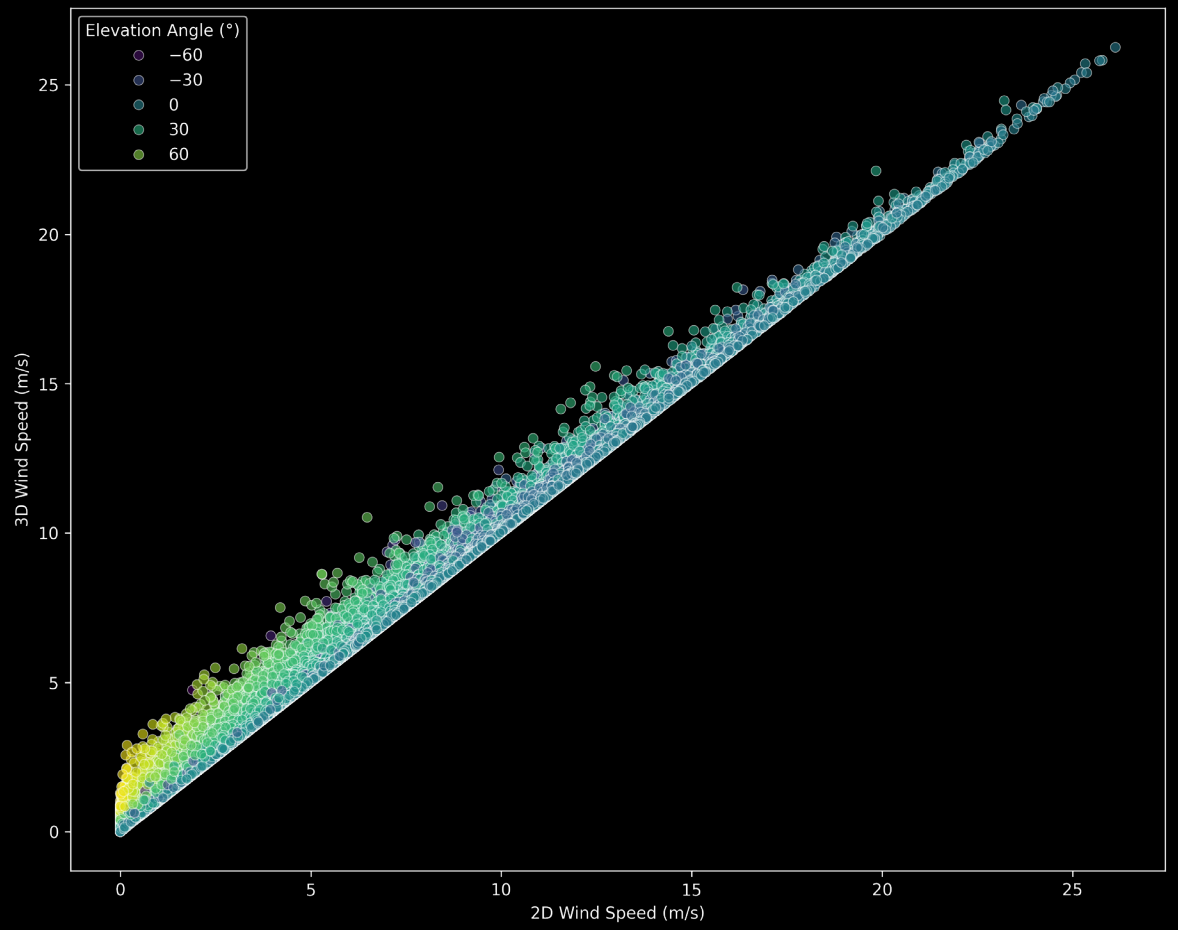
Date Range (records)

- 10/08/2024 (84.601)
- 10/09/2024 (79,042)
- 10/10/2024 (62,301)

Total: 225,944

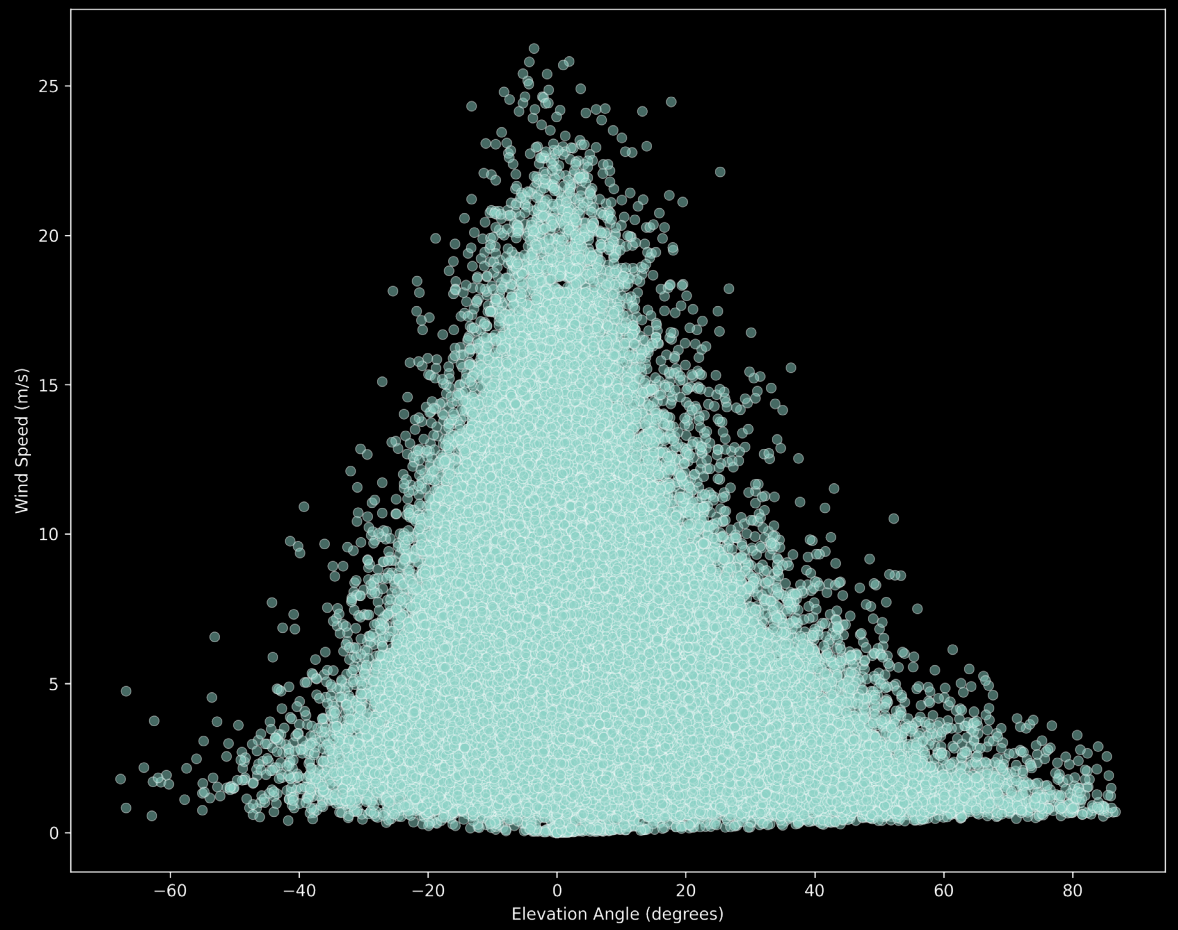
tNow: measurement timestamp | YYYY-MM-DD HH:MM:SS | datetime64
Press_Pa: atmospheric pressure | pascal | float
Temp_C: air temperature in | °C | float
Hum_RH: relative humidity in | 0-100% | float
SonicTemp_C: sonic temperature | °C | float
Error: u_m_s error value from the Anemometer | float
u_m_s: x-direction wind speed | m/s | float
v_m_s: y-direction wind speed | m/s | float
w_m_s: vertical wind speed | m/s | float
2dSpeed_m_s: horizontal wind speed | m/s | float
3DSpeed_m_s: total wind speed | m/s | float
Azimuth_deg: Wind angle in the horizontal plane | 0-360° | float
Elev_deg: elevation angle of the wind vector | -90-90° | float

EDA – Findings



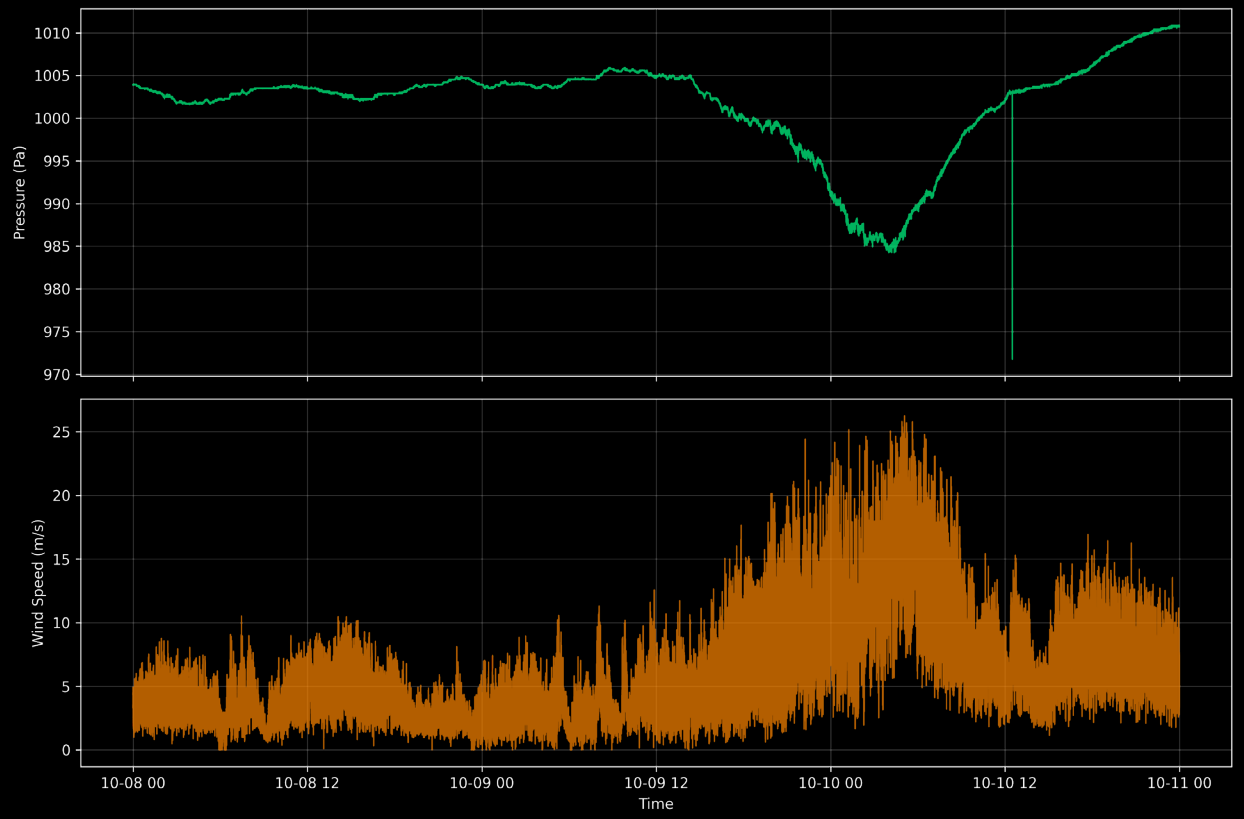
- 2D Wind Speed Vector is almost identical to 3D Wind Speed Vector

EDA – Findings

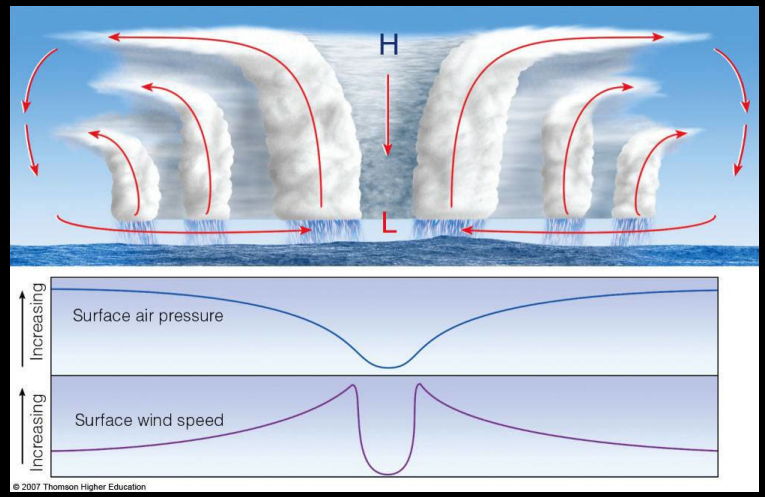


- 2D Wind Speed Vector is almost identical to 3D Wind Speed Vector
- As Elevation angle ↓, Wind speed ↑

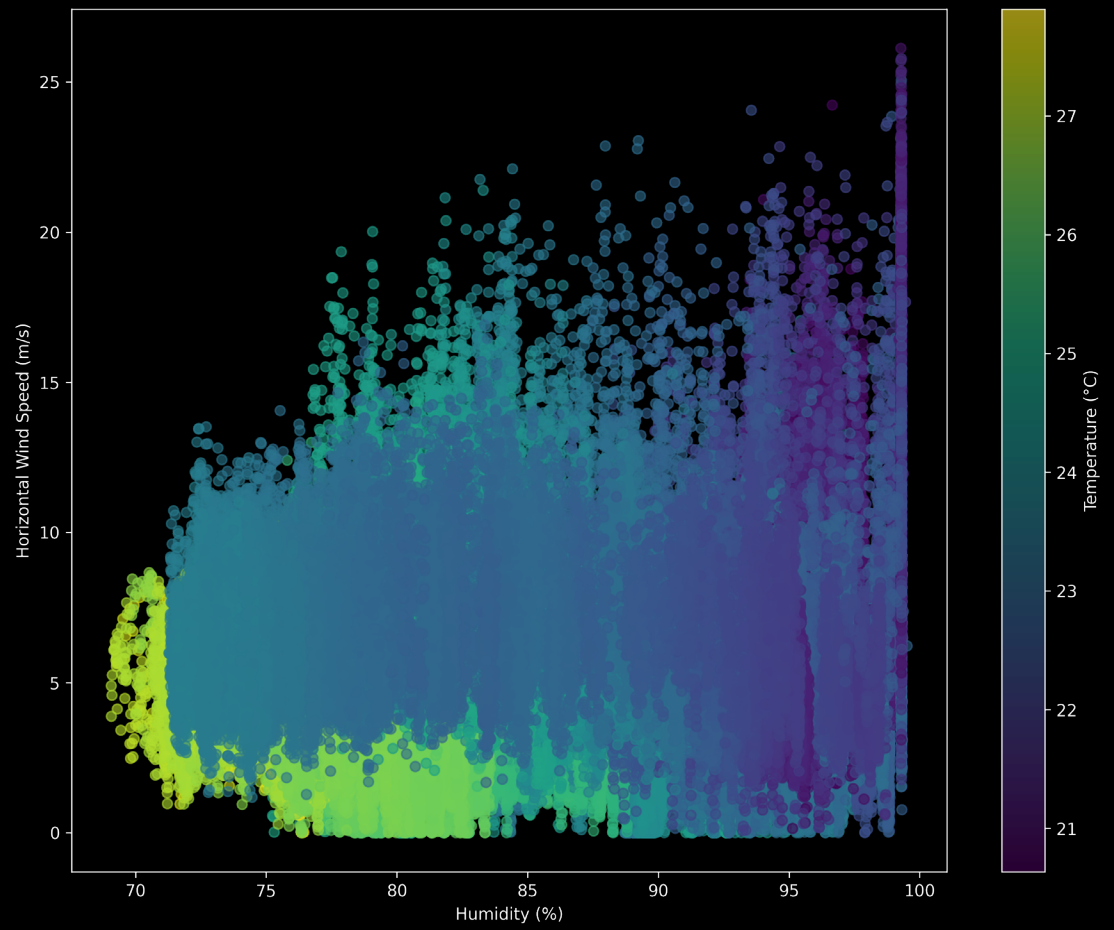
EDA – Findings



- 2D Wind Speed Vector is almost identical to 3D Wind Speed Vector
- As Elevation angle ↓, Wind speed ↑
- As Wind speed ↑, Pressure ↓

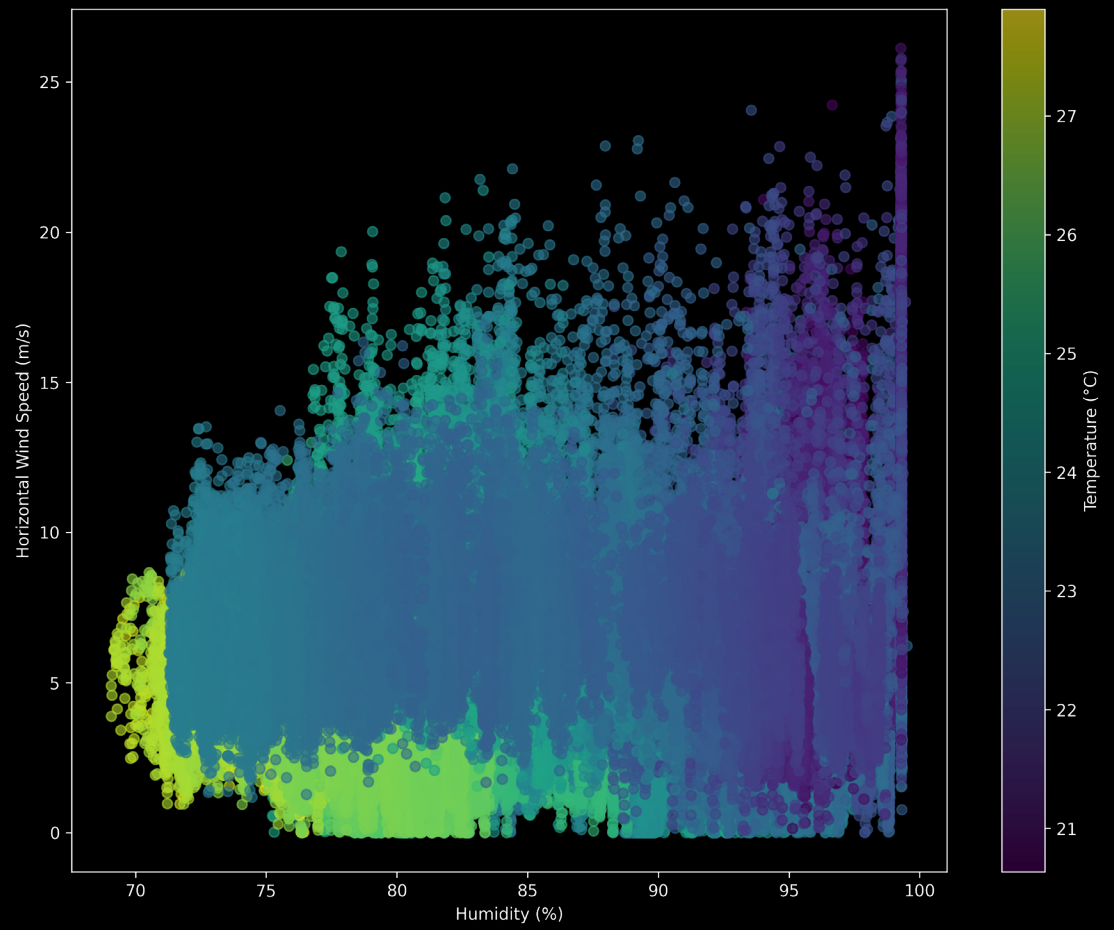


EDA – Findings



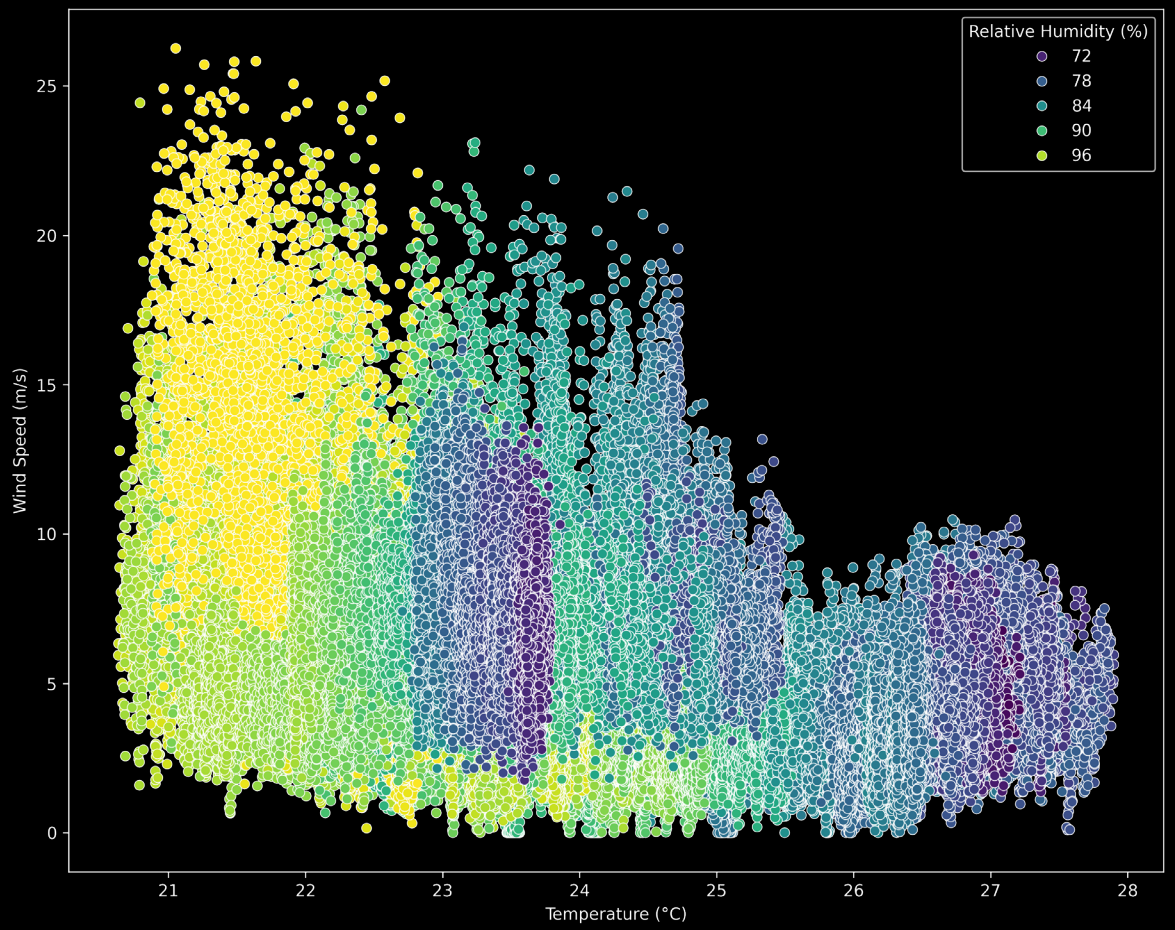
- 2D Wind Speed Vector is almost identical to 3D Wind Speed Vector
- As Elevation angle ↓, Wind speed ↑
- As Wind speed ↑, Pressure ↓
- As Wind speed ↑, RH ↓

EDA – Findings



- 2D Wind Speed Vector is almost identical to 3D Wind Speed Vector
- As Elevation angle ↓, Wind speed ↑
- As Wind speed ↑, Pressure ↓
- As Wind speed ↑, RH ↓
- As RH ↓, Temp ↑

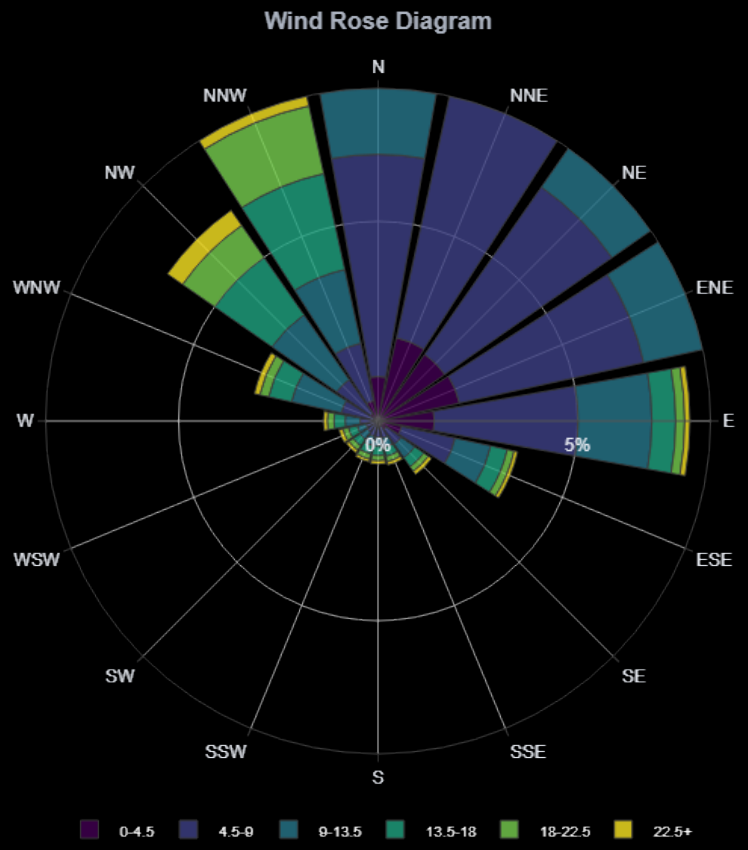
EDA – Findings



- 2D Wind Speed Vector is almost identical to 3D Wind Speed Vector
- As Elevation angle ↓, Wind speed ↑
- As Wind speed ↑, Pressure ↓
- As Wind speed ↑, RH ↓
- As RH ↓, Temp ↑

As Elevation angle ↓, Wind speed ↑, Pressure ↓, RH ↓, Temp ↑

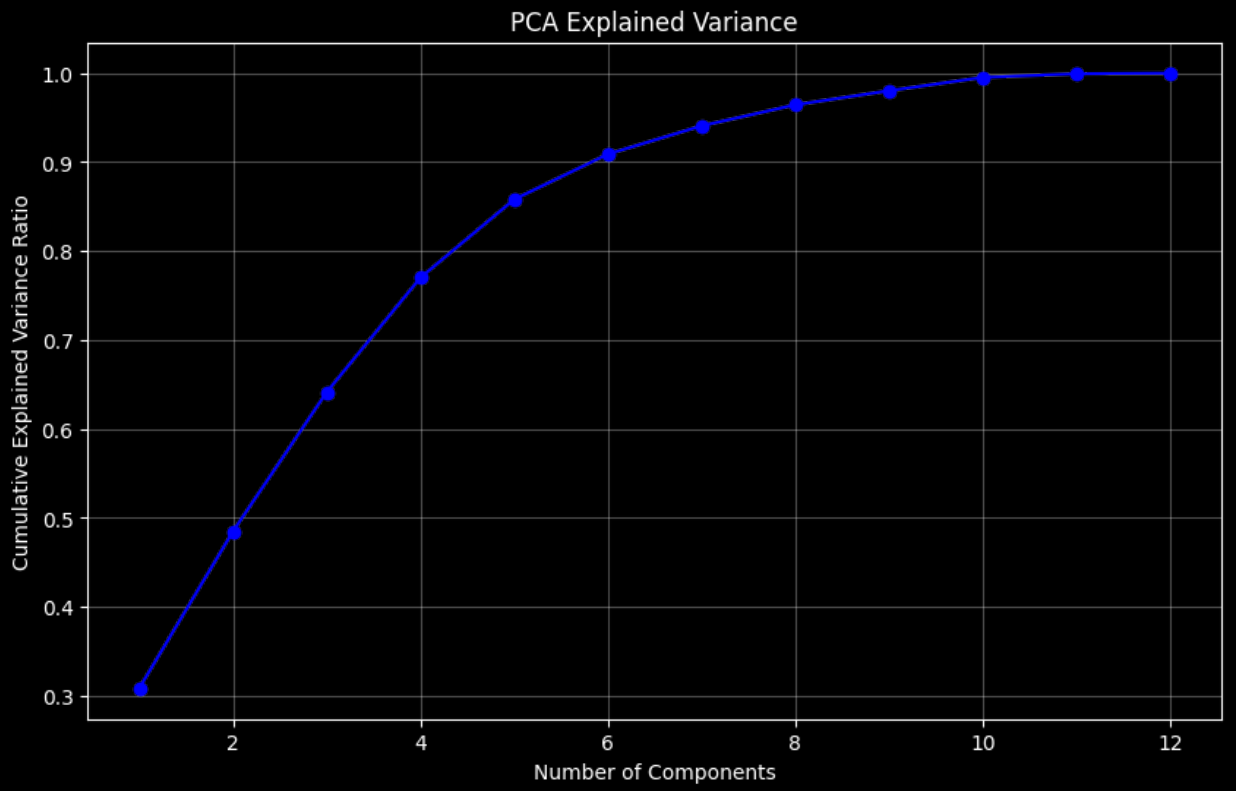
EDA – Findings

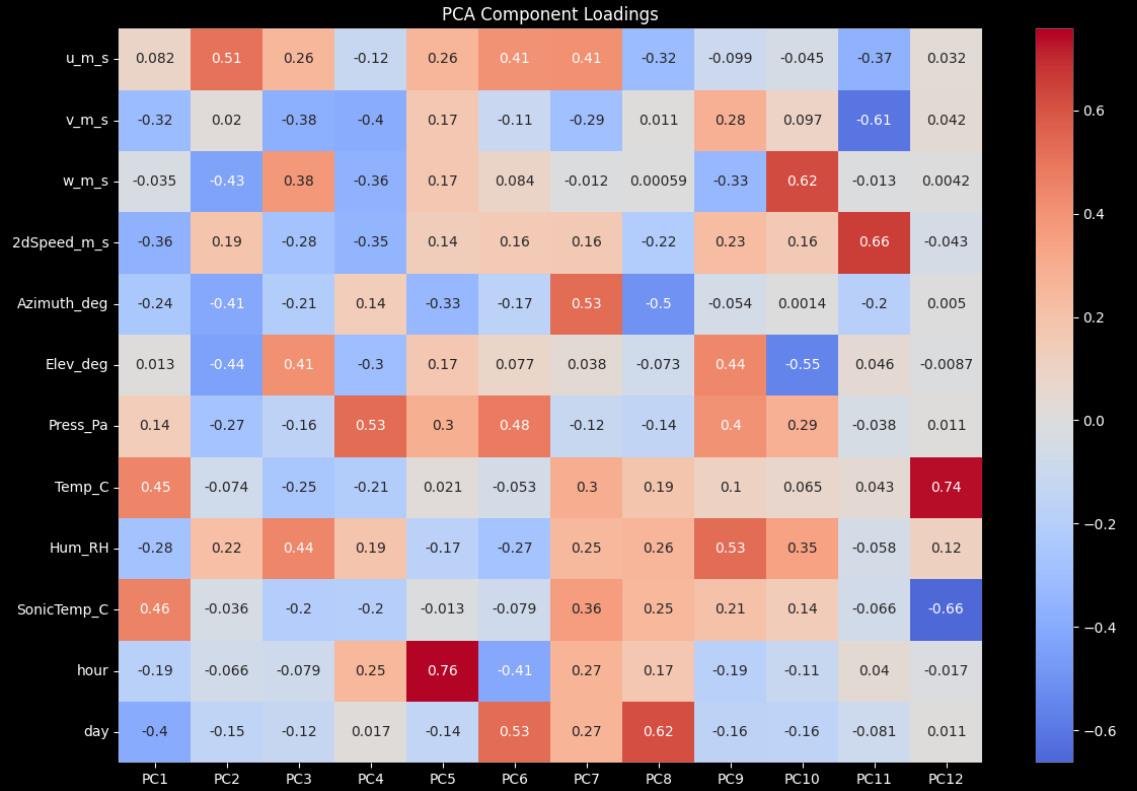


- 2D Wind Speed Vector is almost identical to 3D Wind Speed Vector
- As Elevation angle ↓, Wind speed ↑
- As Wind speed ↑, Pressure ↓
- As Wind speed ↑, RH ↓
- As RH ↓, Temp ↑
- Mainly N/NE hurricanes → The Coriolis Effect!

As Elevation angle ↓, Wind speed ↑, Pressure ↓, RH ↓, Temp ↑

Principal Component Analysis (PCA)





PART 3
Methodology

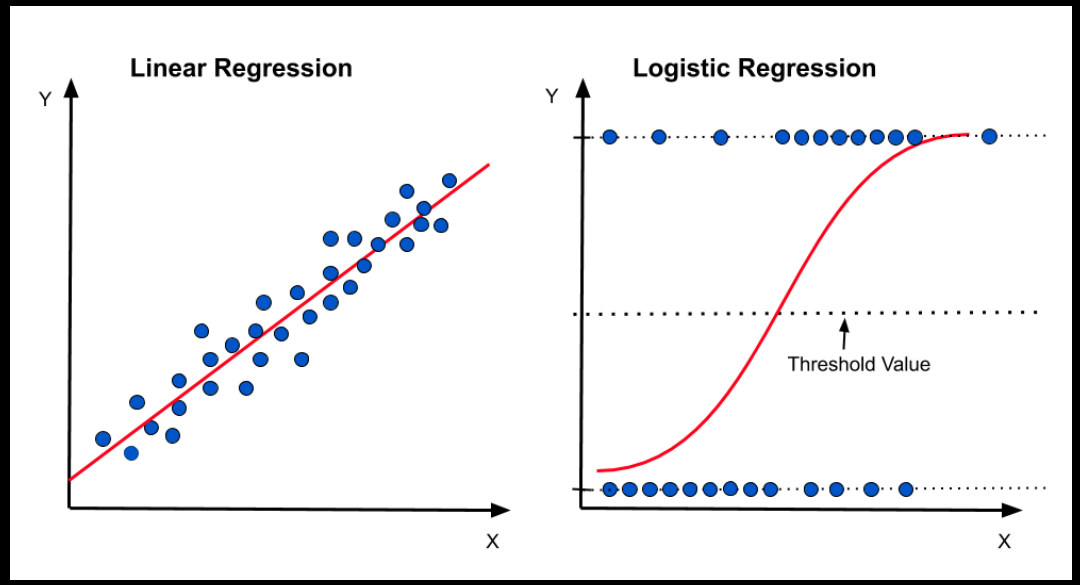
- Logistic Regression
- Decision Tree
- Random Forest
- Model Performance

Logistic Regression

Logistic Regression is a fundamental classification algorithm that models the probability of a binary outcome.

Despite its name, it's used for classification rather than regression. It works by applying a logistic function to a linear combination of input features, producing probability values between 0 and 1.

It's particularly effective when the relationship between input features and output is linear, and it provides easily interpretable results with probability scores for predictions.

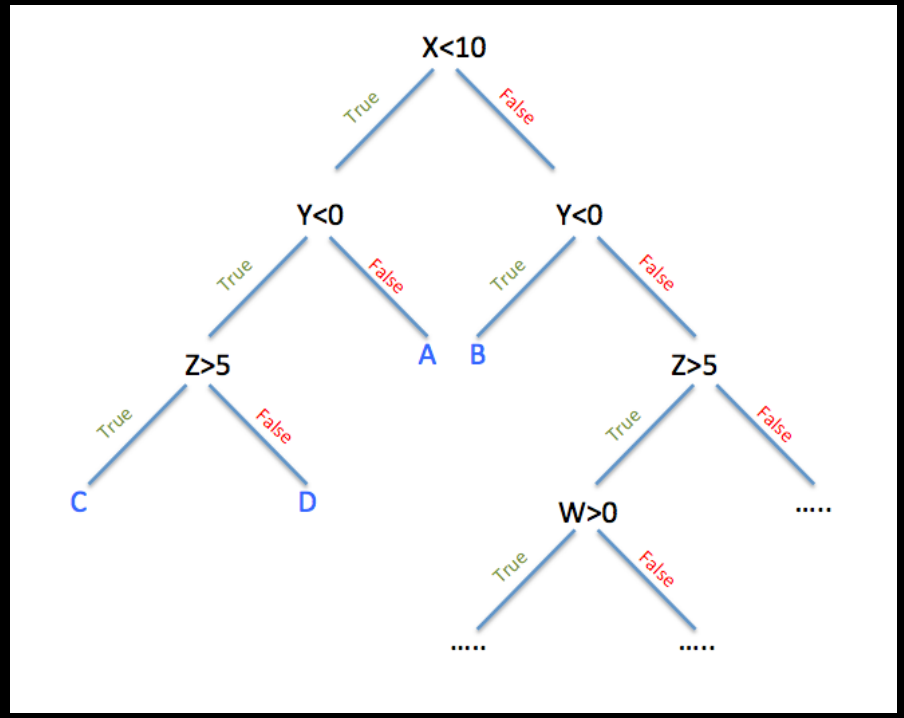


Decision Tree

Decision Tree is a flowchart-like algorithm that makes decisions by splitting data into subsets based on different conditions.

It creates a tree structure where each internal node represents a decision based on a feature, each branch represents an outcome of that decision, and each leaf node represents a final classification.

The algorithm is intuitive to understand and visualize, can handle both numerical and categorical data, and requires minimal data preparation, though it can be prone to overfitting.

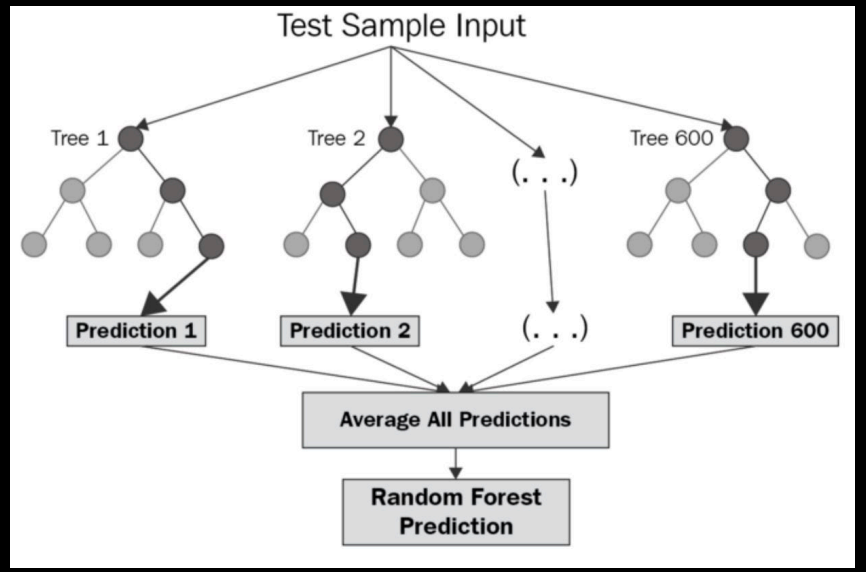


Random Forest

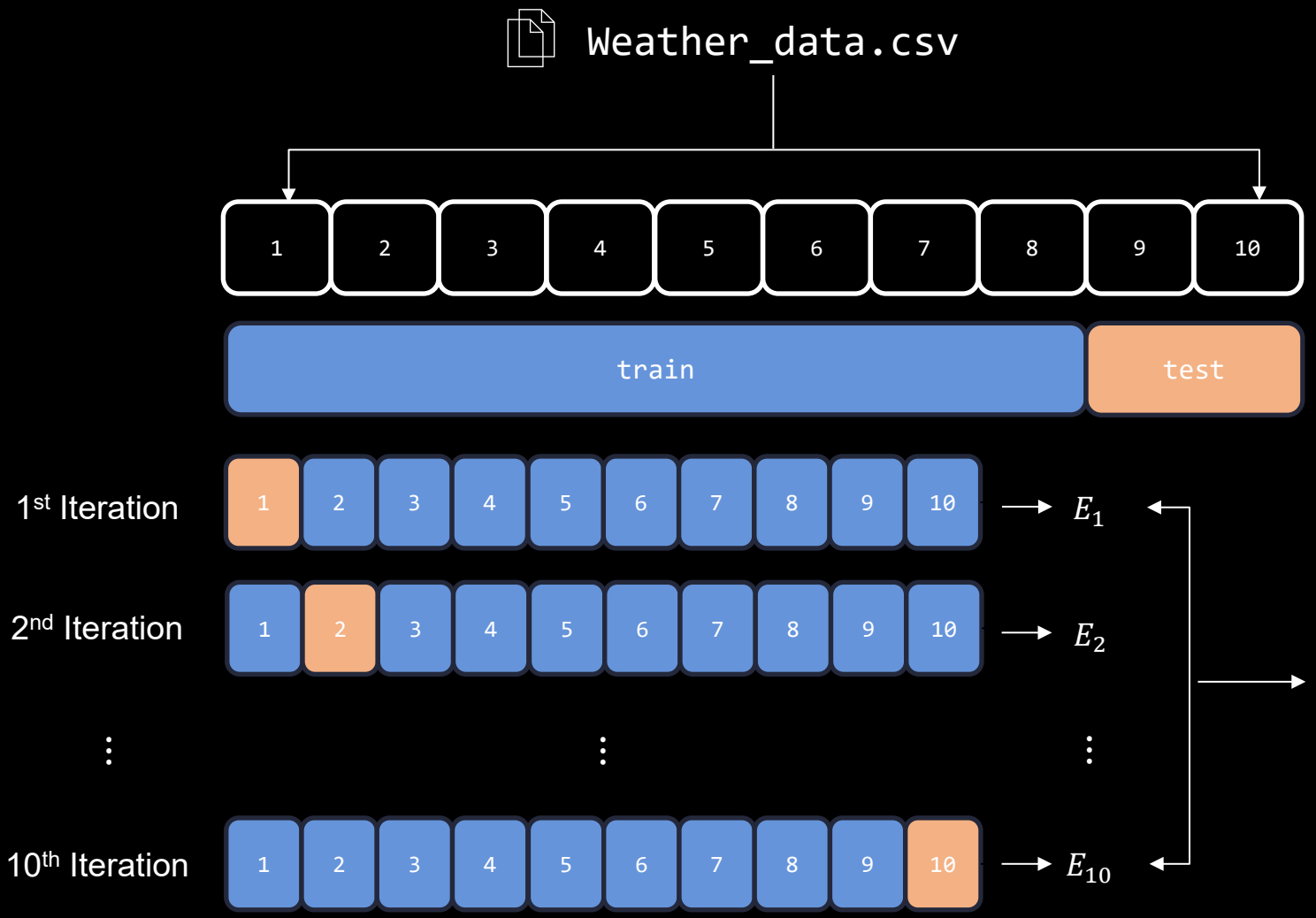
Random forest is a machine learning algorithm that combines multiple decision trees to make predictions.

It creates an ensemble of decision trees and aggregates their predictions to generate the final prediction. Random forest are robust against overfitting and can handle a large numbers of input variables without feature selection.

It can be used to analyze the relationships between predictor variables and response variables and make predictions based on the collective knowledge of the decision trees.



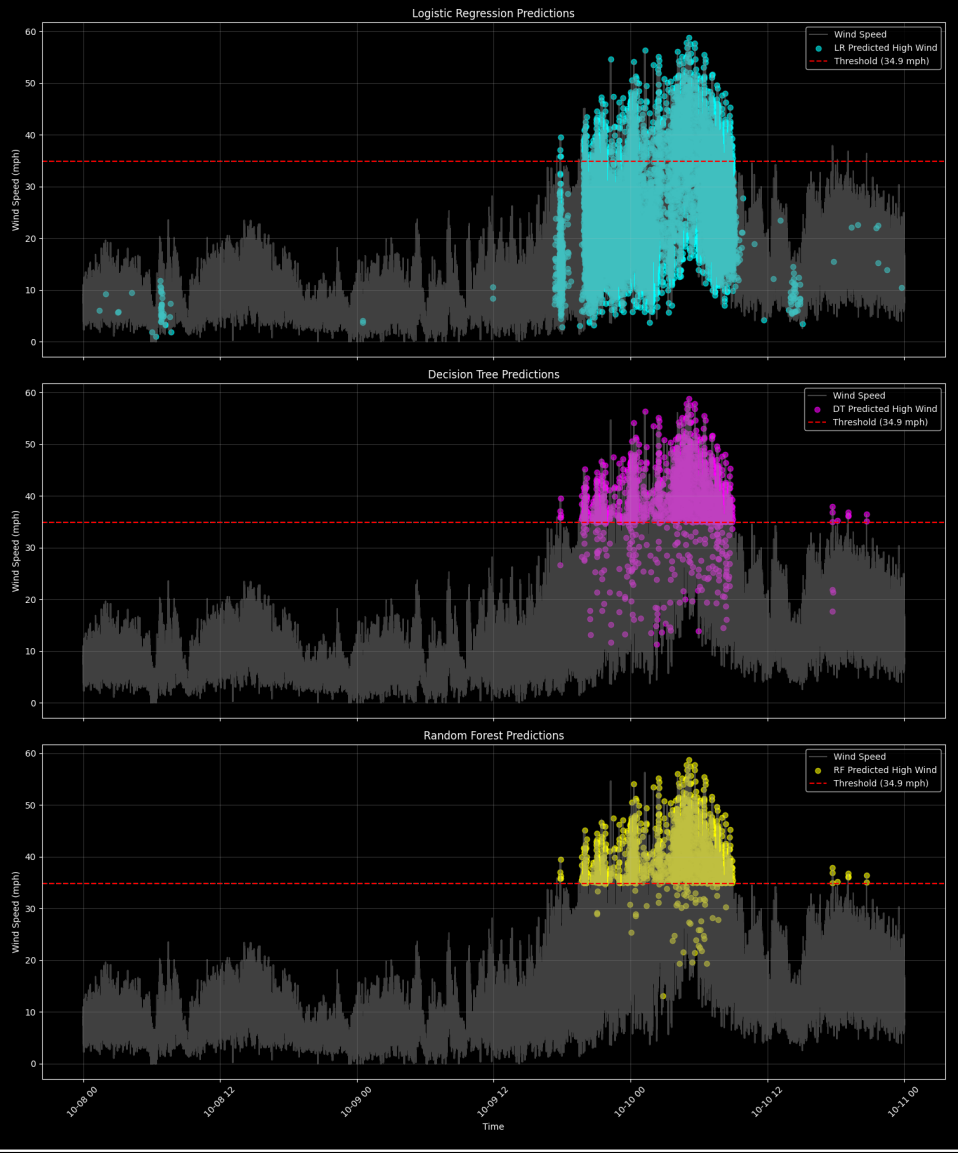
K-fold Cross Validation

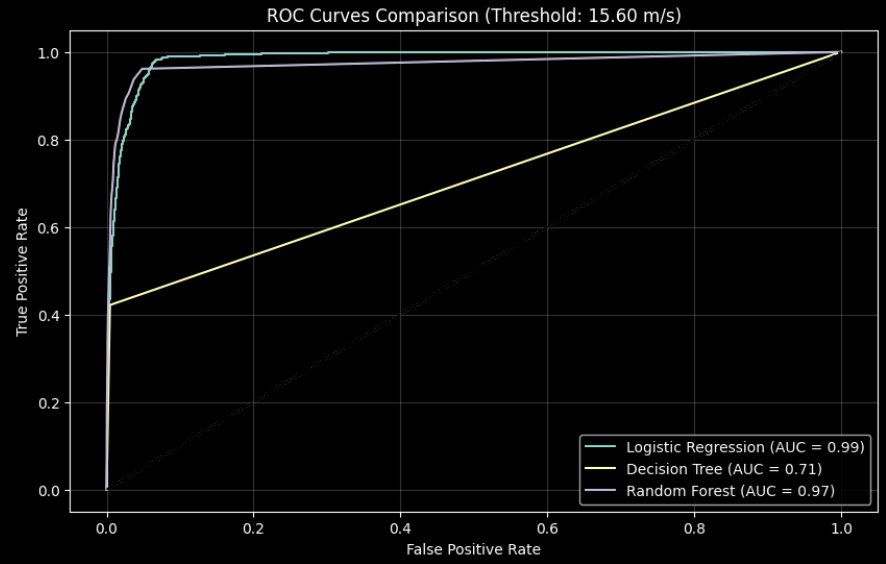


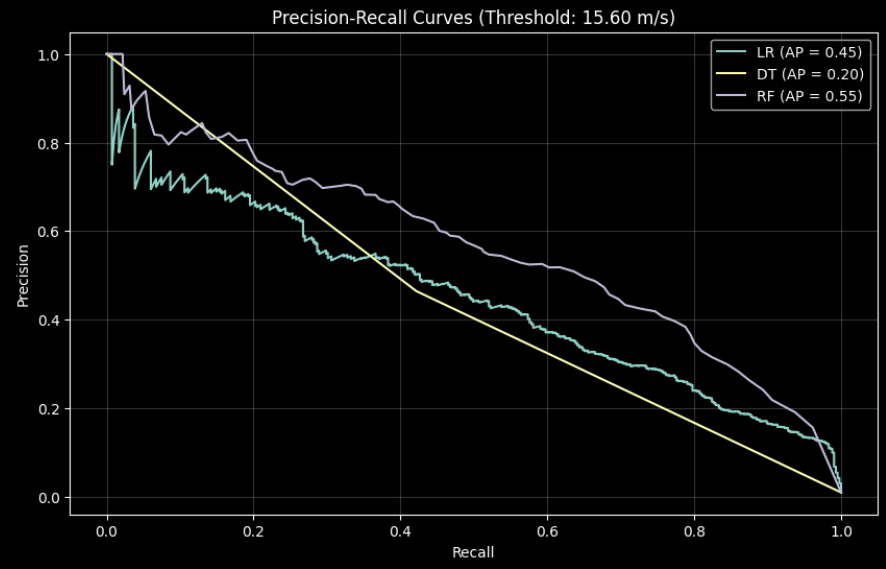

```python
def evaluate_with_cv(X, y, model_name, model, n_splits=5):
    if len(np.unique(y)) < 2:
        return {"accuracy": 1.0, "roc_auc": 0.5}

    cv = StratifiedKFold(n_splits=n_splits, shuffle=True, random_state=42)
    scores = {"accuracy": [], "roc_auc": []}

    for fold, (train_idx, test_idx) in enumerate(cv.split(X, y), 1):
        X_train, X_test = X[train_idx], X[test_idx]
        y_train, y_test = y[train_idx], y[test_idx]

        print(f"\nFold {fold}: Train Class 0: {sum(y_train==0)}, Class 1: {sum(y_train==1)};
"
              f"Test Class 0: {sum(y_test==0)}, Class 1: {sum(y_test==1)}")

        # Scaling and training
        model.fit(StandardScaler().fit_transform(X_train), y_train)
        y_prob = model.predict_proba(StandardScaler().transform(X_test))[:, 1]

        fold_roc_auc = roc_auc_score(y_test, y_prob)
        scores["accuracy"].append(accuracy_score(y_test,
model.predict(StandardScaler().transform(X_test))))
        scores["roc_auc"].append(fold_roc_auc)

        print(f"Fold ROC-AUC: {fold_roc_auc:.3f}")

    avg_roc_auc, avg_acc = np.mean(scores["roc_auc"]), np.mean(scores["accuracy"])
    print(f"\nAvg ROC-AUC: {avg_roc_auc:.3f}, Avg Accuracy: {avg_acc:.3f}")

    return {"accuracy": avg_acc, "roc_auc": avg_roc_auc}
```

$$OER_{(k=1)} = \frac{1}{10}\sum_{i=1}^{10} E_i$$

Model Performance







PART 4
Conclusion

Conclusion

Our analysis revealed several key findings:

1. Wind patterns showed strong directional preferences, with highest speeds in the W-N quadrant, which explains the Coriolis effect.
2. Temperature and humidity demonstrated strong inverse relationships
3. PCA identified distinct patterns in temperature and wind components
4. Random Forest achieved the best performance in classifying high-wind events

Future Work

Additional Environmental Sensors

Real-time Prediction Capabilities

Weather Events Analysis Extension

Advanced Machine Learning Techniques

Acknowledgements

Contributors	Tasks
Erik Liebergall	Sensor bracket design and fabrication
Brendon	lower box and mount (lower on pole to hold RPi))
Avinash Muthu Krishnan, Dominick Strollo	Sensor wiring, testing, coding
Avinash Muthu Krishnan, Marc Compere	RS-485 interface to RPI
Kaleb Nails	Discord bot, web based interface for real time and logged data access
Kaleb Nails, Marc Compere	WiFi access to nearest outdoor access point (small high gain antenna)
John Ingram	Facilities for installation with the high-lift

Faculty Advisor: Avinash Muthu Krishnan, Marc Compere, Kevin A. Adkins

THANKS

Questions?