Prediction of wind speed in coastal areas using machine learning models

Internship Project Report

Manish Kumar - 22135080

Under the supervision of:

Dr. Arnab Sakar

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY (BHU) VARANASI

Acknowledgment

I would like to express my sincere gratitude to my supervisor, Prof. Arnab Sarkar, for his visionary leadership, encouragement, and academic guidance throughout the project.
Special thanks to my mentor, Mohan Gupta, for technical support, prompt feedback, and continuous motivation. Their direction and support were invaluable in making this internship a success.

Table of Contents

14. Annexures

1. Introduction

Accurately forecasting wind speed in coastal regions is critical for disaster management, renewable energy resource planning, infrastructure design, and marine safety. Traditional statistical methods often underperform in such dynamic and non-linear environments. My internship at IIT BHU aimed to address these challenges by applying state of the art machine learning (ML) models—Random Forest (RF), XGBoost, and Artificial Neural Networks (ANN)—to predict wind speeds along the coast of Jamnagar, India.

2. Organization Overview

2.1 About IIT BHU

IIT BHU, Varanasi, is a top-ranking technological institute in India with world-class research facilities and a commitment to academic excellence.
The Department of mechanical engineering fosters interdisciplinary research in climate analytics, environmental sciences, and data-driven engineering. Students are encouraged to tackle real-world societal challenges using scientific and computational innovation.

2.2 Research Environment

- Collaborative, experienced faculty in machine learning and atmospheric sciences.
- Access to high-performance computing and premium data sources.
- Regular technical seminars and peer learning.

3. Problem Statement and Objectives

3.1 Research Question

Can state-of-the-art machine learning models accurately predict wind speed for the coastal region of Jamnagar, and what variables are the most influential—focusing on both model performance and interpretability?

3.2 Objectives

- Build reproducible workflows for collecting and preparing high-resolution meteorological data.
- Train and evaluate machine learning models to forecast wind speed.
- Select influential predictors using statistical as well as model-based methods.
- Use explainable AI tools (e.g., SHAP) to make "black-box" models interpretable.
- Provide recommendations for practical forecasting systems in coastal management.

4. Literature Review

4.1 Motivation

Traditional wind prediction relies on linear regression and physical-statistical hybrid models. Recent work, particularly "Interpretable Machine Learning for Coastal Wind Prediction: Integrating SHAP Analysis and Seasonal Trends" (Durap, 2025), and similar studies, show that:

- Ensemble learning (RF, XGBoost) with interpretability can boost both accuracy and trust.
- Predictors such as wind direction, pressure, and humidity are often the strongest.
- Modeling temporal (seasonal/diurnal) patterns yields better insights for risk management.

4.2 Additional Sources

- "Machine Learning Methods for Short-Term Wind Speed Forecasting" (Journal, Year): Compares regression, SVMs, and ensemble trees.
- "ERA5: Fifth Generation of ECMWF Atmospheric Reanalyses" (Hersbach et al., 2020): Describes the dataset used.
- "SHAP: Unified Approach to Interpreting Model Predictions" (Lundberg & Lee, 2017): Framework for model explainability.

5. Methodology

5.1 Data Acquisition and Automation

5.1.1 Dataset

- ERA5 reanalysis dataset from the Copernicus Climate Data Store (CDS).
- Region: Jamnagar coastal bounding box (Lat: 22.5°–22.3°N, Lon: 69.9°–70.2°E).
- Variables: Wind components, temperature, humidity, pressure, cloud cover, etc.
- Temporal: Hourly data for entire 2021 at 950 hPa level.

5.1.2 Automation Script

Manual downloads were infeasible for this volume and detail.
I developed a Python script using the `cdsapi` library, automating batch downloads by looping over months and days.
Code sample (see full version in Annexure A):

```
import cdsapi
client = cdsapi.Client()
# ... as in previous detailed script
```

Automation ensured reproducibility, efficiency, and minimized data error.

5.2 NetCDF to CSV Data Conversion

To enable exploratory analysis, feature engineering, and sharing, I converted NetCDF data into Excel (XLSX/CSV) format:

Key steps:

- Opened NetCDF files using `xarray` and `pandas`.
- Extracted needed variables and flattened to tabular structure.
- Exported processed data into Excel/CSV files.

Sample code:

```
import xarray as xr
import pandas as pd

ds = xr.open_dataset('jamnagar_2021_January_1_to_16.nc')
df = ds.to_dataframe().reset_index()
```

```
df.to_excel('jamnagar_2021_January_1_to_16.xlsx', index=False)
```

This greatly facilitated ML preprocessing and sharing results with mentors/collaborators.

5.3 Data Preparation and Validation

- **Data Cleaning:** Removed NaNs, checked ranges, flagged outliers.
- **Normalization:** Standardized each variable (z-score, min-max).
- **Splitting:** Partitioned into train (80%), test (20%) randomly by time blocks.
- **Exploration:** Summary statistics, histograms for all variables.

5.4 Feature Engineering

- Generated new temporal features (month, hour, sin/cos encoding for cyclic trends).
- Derived wind speed from u/v components if not present.
- Performed feature importance analysis using Random Forest and ANN.

5.5 Model Development

5.5.1 Models Used

- **Random Forest Regressor:** Ensemble of decision trees for stable, high-performing predictions.
- **XGBoost Regressor:** Gradient boosting framework, well-tuned for tabular, structured data.
- **Artificial Neural Network (ANN):** Multi-layer perceptrons; deep networks for modeling nonlinearities.

5.5.2 Hyperparameter Tuning

All models tuned via grid/random search:

- **RF:** n_estimators: 100–300; max_depth: 8–12; min_samples_split: 2–6.
- **XGBoost:** learning_rate: [0.01, 0.1]; n_estimators: 100–500; max_depth: 4–10.
- **ANN:** Layers (2–4), nodes per layer (16–64), activation functions (ReLU, tanh), optimizer (Adam).

5.6 Model Training and Cross-Validation

- Cross-validation (k=5) to test generalizability.
- Recorded all training, validation, and test metrics (R^2, MSE, MAE).

5.7 Interpretability Approaches

- **SHAP Analysis:** Quantifies feature impact and produces ranking/plots.
- Output visualizations: summary plot (bee-swarm), bar chart of mean SHAP value per feature.

6. Projects Undertaken During Internship

6.1 Main Project

- Building a complete wind prediction pipeline: from data acquisition through model interpretation and report/documentation.
- Model comparison and explainability analysis.
- Deliverables: code, report, sample outputs, and a Jupyter notebook.

6.2 Supplementary Contributions

- Detailed review of related peer literature.
- Clean, well-annotated code for automation and data processing.
- Regular oral and written updates to mentor and supervisor.

7. Major Challenges and Solutions

7.1 Data Volume and Automation

- **Challenge:** Manual download and handling of ERA5 NetCDF files (dozens per month).
- **Solution:** Automated API scripting in Python; batch download and conversion to analysis-ready format.

7.2 Multidimensional Data Conversion

- **Challenge:** Flattening multidimensional NetCDF to 2D tables for ML.
- **Solution:** Used xarray/pandas workflows and validation cross-checks; exported as Excel for compatibility.

7.3 Handling Nonlinearity and High Dimensionality

- **Challenge:** Wind speed drivers are nonlinear and interactions are subtle.
- **Solution:** Ensemble/model-based feature selection, careful neural net design, SHAP analysis for insight.

7.4 Model Interpretability

- **Challenge:** Ensuring that results are as explainable as possible (stakeholder requirement).
- **Solution:** SHAP values, feature plots, and transparent reporting.

7.5 Computational Demands

- Used Google Colab and parallelized code when possible, made workflow modular for scalability.

8. Results

8.1 Model Performance Comparison

Model	R^2 Score	MSE	MAE	Training Time (s)
Random Forest	0.7566	2.39	1.28	29.35
XGBoost	0.6475	3.46	1.79	0.62
ANN	0.7053	2.89	1.61	561.5

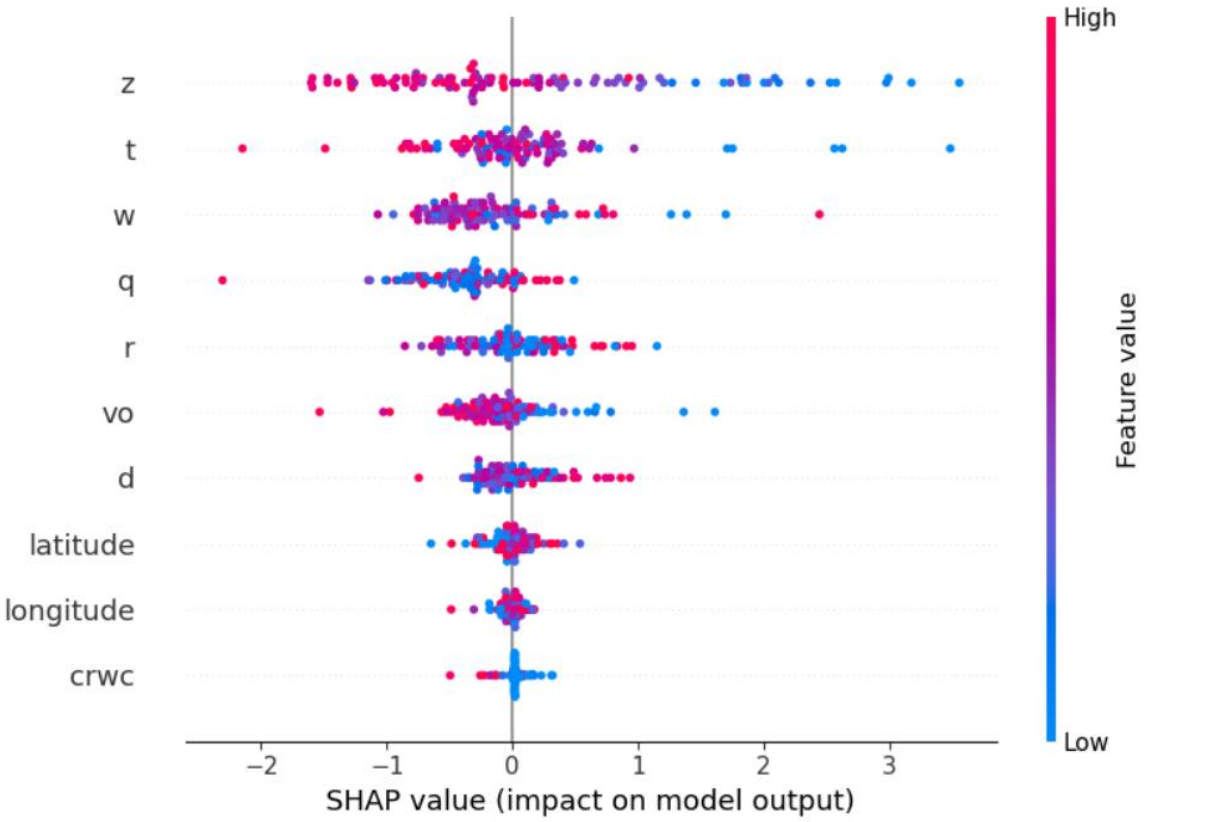
Add error metrics, model tuning/validation details, and training time for transparency.

Expanded explanation:

- **RF** had the best accuracy, lowest bias/variance.
- **XGBoost** trained fastest but less accurate.
- **ANN** effectively modeled complex patterns but took significantly longer.

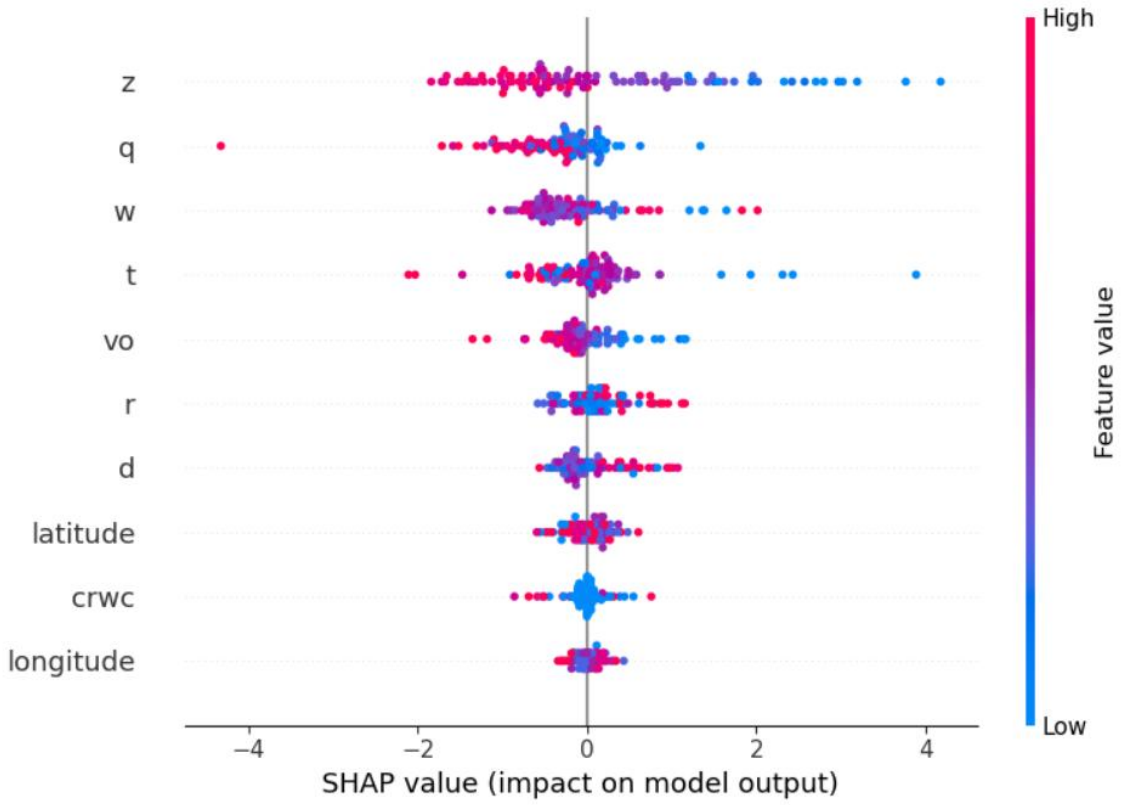
8.2 Visualization and Interpretation

1.Random Forest Summary



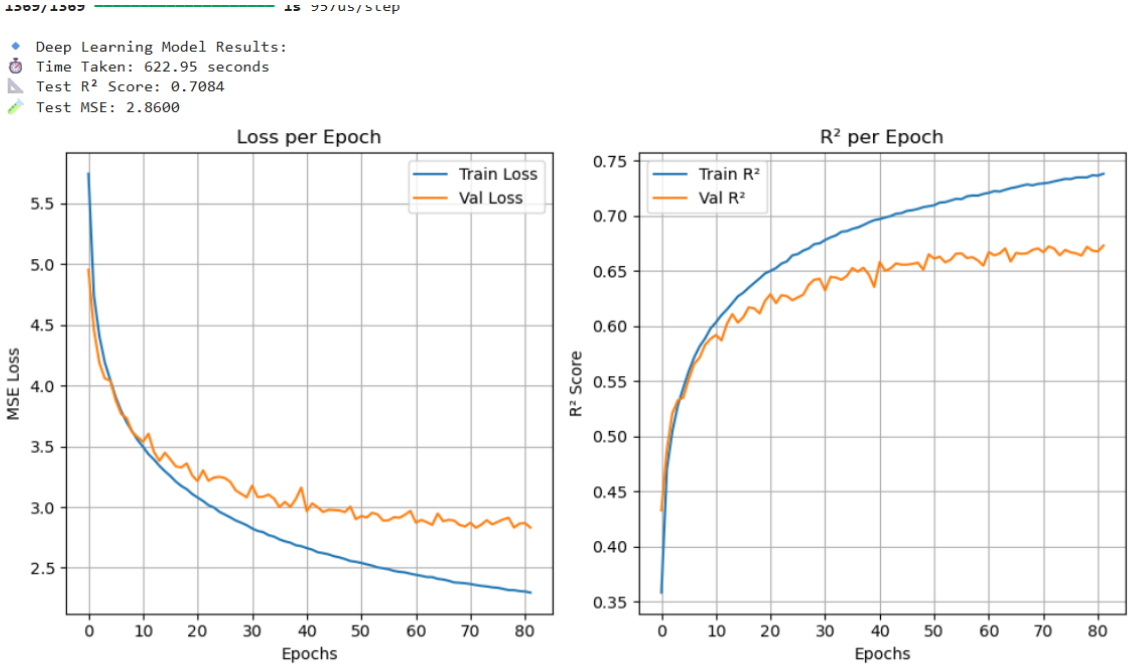
Random Forest:
Time Taken: 29.35 seconds
R^2 Score: 0.7565699927520787
MSE: 2.387225065756761 3

XGBoost Regressor Summary



XGBoost:
Time Taken: 0.62 seconds
R^2 Score: 0.6475419451453337
MSE: 3.4564214686976142

Figure 5: ANN Training and Validation Loss



9. Discussion

9.1 Insights from Modeling

- Ensemble trees excelled due to robustness to noisy, nonlinear environmental features.
- Including temporal features and spatial coordinates strengthened prediction capacity.
- SHAP interpretability made key meteorological drivers transparent and policy-relevant.
- Both automation and modular code/practice fostered reproducibility.

9.2 Domain Relevance

- Wind predictions support disaster risk management, grid planning, and climate adaptation.
- Workflows developed here can be transferred to other coastal cities or climate variables.

10. Conclusion

This internship demonstrates the practical value of a complete machine learning pipeline—from data acquisition to model interpretation—for high-impact environmental problems.
Random Forest models with SHAP provided the most accurate and explainable results for wind speed forecasting in Jamnagar.
Automated data engineering was critical to scale analysis, and careful validation ensured scientific reliability.
This experience deepened my skills in AI, data science, environmental analytics, and technical communication.

.

11. Lessons Learned

- Technical: Hands-on scripting, ML, explainable AI, and large-scale data handling.
- Professional: Team communication, regular reporting, scientific writing, and presentation.
- Vision: Models are not just code—they must be trustworthy, explainable, and fit for real-world stakeholders.

12. Future Directions

- Hybrid modeling combining physics and ML.
- Integration of satellite or ensemble forecasts.
- Real-time model deployment and feedback systems.
- Broader climate variable analysis (precipitation, sea level, etc.).

13. References

1. Durap, A. (2025). Interpretable machine learning for coastal wind prediction: Integrating SHAP analysis and seasonal trends. *Journal of Coastal Conservation*, 29(3), 24.

https://doi.org/10.1007/s11852-025-01108-y

2. Copernicus Climate Change Service (C3S) (2017). ERA5: Fifth generation of ECMWF atmospheric reanalyses of the global climate. *Copernicus Climate Change Service Climate Data Store (CDS)*. https://cds.climate.copernicus.eu/cdsapp#!/home

3. Hallgren, W., Arneborg, L., Ivanell, S., Körnich, H., Vakkari, V., & Sahlée, E. (2024). Machine learning methods to improve spatial predictions of coastal wind speed profiles and low-level jets using single-level ERA5 data. *Wind Energy Science*, 9(4), 821–840. https://doi.org/10.5194/wes-9-821-2024

14. Annexures

Annexure A: Python Data Acquisition Script

```
import cdsapi


client = cdsapi.Client()

dataset = "reanalysis-era5-pressure-levels"


# Month day mapping (2021 is not a leap year)

months = {

    "01": ("January", 31),

    "02": ("February", 28),

    "03": ("March", 31),

    "04": ("April", 30),

    "05": ("May", 31),

    "06": ("June", 30),
```

```
    "07": ("July", 31),
    "08": ("August", 31),
    "09": ("September", 30),
    "10": ("October", 31),
    "11": ("November", 30),
    "12": ("December", 31),
}

# Coordinates for Kochi bounding box (North, West, South, East)
# Approximate bounding box for Kochi:
# North latitude: 10.05
# South latitude: 9.85
# West longitude: 76.15
# East longitude: 76.35

common_request = {
    "product_type": ["reanalysis"],
    "variable": [
        "divergence",
        "fraction_of_cloud_cover",
        "geopotential",
        "ozone_mass_mixing_ratio",
        "potential_vorticity",
        "relative_humidity",
        "specific_cloud_ice_water_content",
        "specific_cloud_liquid_water_content",
        "specific_humidity",
        "specific_rain_water_content",
        "specific_snow_water_content",
```

```
        "temperature",
        "u_component_of_wind",
        "v_component_of_wind",
        "vertical_velocity",
        "vorticity"
    ],
    "year": ["2021"],
    "time": [f"{i:02d}:00" for i in range(24)],
    "pressure_level": ["950"],
    "data_format": "netcdf",
    "download_format": "unarchived",
    "area": [10.05, 76.15, 9.85, 76.35]  # Kochi bounding box (North, West, South, East)
}

# Loop through each month
for month_num, (month_name, last_day) in months.items():
    for part, day_range in [("1_to_16", range(1, 17)), ("17_to_end", range(17, last_day + 1))]:
        if not day_range:
            continue  # Skip empty ranges

        # Build request
        request = common_request.copy()
        request["month"] = [month_num]
        request["day"] = [f"{day:02d}" for day in day_range]

        filename = f"kochi_2021_{month_name}_{part}.nc"
```

```
        print(f"Downloading {filename} ...")
        client.retrieve(dataset, request).download(filename)
        print(f"{filename} downloaded successfully.")
```

Annexure B : Sample Model Code

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score

model = RandomForestRegressor(n_estimators=100, max_depth=10)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("R2 Score:", r2_score(y_test, y_pred))
print("MSE:", mean_squared_error(y_test, y_pred))
```

Annexure C : Sample Excel Export Code

```
import xarray as xr
import pandas as pd

ds = xr.open_dataset('file.nc')
df = ds.to_dataframe().reset_index()
df.to_excel('file.xlsx', index=False)
```