National College of Ireland

The Influence of Weather on Fashion Retail: Developing a Sustainable and Interpretable Forecasting Model

MSc Research Project
Artificial Intelligence for Business

César Antonio Ledesma González
Student ID: 23425016

School of Computing
National College of Ireland

Supervisor: Victor del Rosal

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	César Antonio Ledesma González
Student ID:	23425016
Programme:	MsC AI for Business **Year:** 2024
Module:	Practicum 2: MSc Research Project
Lecturer:	Victor del Rosal
Submission Due Date:	11/08/2025
Project Title:	"The Influence of Weather on Fashion Retail: Developing a Sustainable and Interpretable Forecasting Model" // Configuration Manual
Word Count:	750 **Page Count:** 10 pages

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	César Antonio Ledesma González
Date:	11/08/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

César Antonio Ledesma González
Student ID: 23425016

1 Introduction

This Configuration Manual details the environment setup as well as the execution steps for the forecasting model developed during the MSc project "The Influence of Weather on Fashion Retail: Developing a Sustainable and Interpretable Forecasting Model." It contains all details regarding the environment setup, and steps needed to reproduce experiments and results.

2 Hardware Requirements

The system setup for this research is described in Figure 1 along with the software specs used.

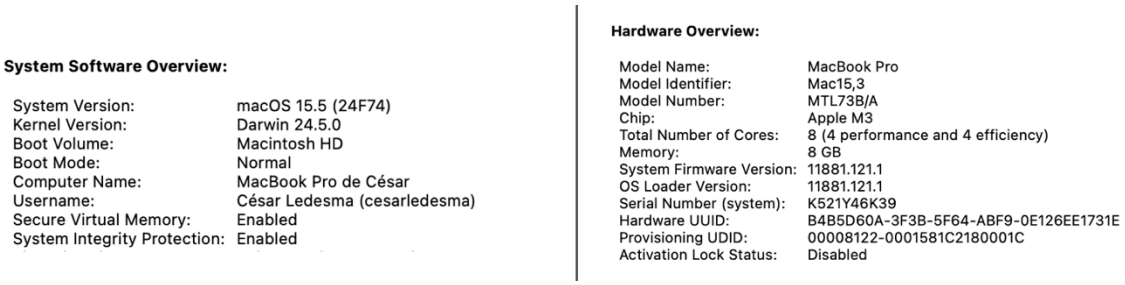


Fig 1. System Configuration

To effectively implement the forecasting model, the following specifications are recommended:

Minimum:

- CPU: Intel® Core™ i5/ Apple M1
- RAM: 8 GB
- Disk: 500 MB Available

Recommended:

- RAM: 16 GB (for faster SHAP/Prophet processing)

3 Environment Setup

Operation System: macOS Sequoia 15.5

Development tools:

- Visual Studio Code, Version: 1.102.2 (Universal)
- Python Virtual Environment (venv)
- Terminal (macOS default for iTerm2)

Programming Language:

- Python 3.12.11

4 Required Libraries and Dependencies

Prior to conduction any data analysis, must ensure that required libraries and packages that facilitate data cleaning, processing, modelling, and evaluation are installed. The critical libraries that have been references in thesis are:

- **Package Manager:** pip
- **Installation File:** requirement.txt

Requirements.txt

```txt
#Core libraries
pandas==2.2.2
numpy==1.26.4
matplotlib==3.8.4
seaborn==0.13.2
scikit-learn==1.5.0

#Machine learning models
xgboost==2.0.3
lightgbm==4.3.0
statsmodels==0.14.2

#Time series and forecasting
prophet==1.1.5
pmdarima==2.0.4

#Model interpretation
shap==0.45.0

#Clustering and dimensionality reduction
scipy==1.13.0
umap-learn==0.5.6

#CmdStan backend for Prophet (required if using
Prophet with Stan)
cmdstanpy==1.2.1
```

Install with:

```bash
pip install -r requirements.txt
```

5 System Configuration

The development of this research project was carried using Visual Studio Code
Virtual Environment Setup:

```bash
Python3 -m nenv .venv
source .venv/bin/activate #macOS/Linux
```

Directory Structure:

```
/Modelling
├── prepared_sales_weather_dataset.csv
├── shap_feature_analysis.py
├── Prophet_forecast_comparison.py
├──clustering_and_forecasting_sub_category.py
├── outputs/ (plots & logs)
```

Environment Variables: None required explicitly.

This setup can also be adapted to Windows with Anaconda or Google Colab if desired.

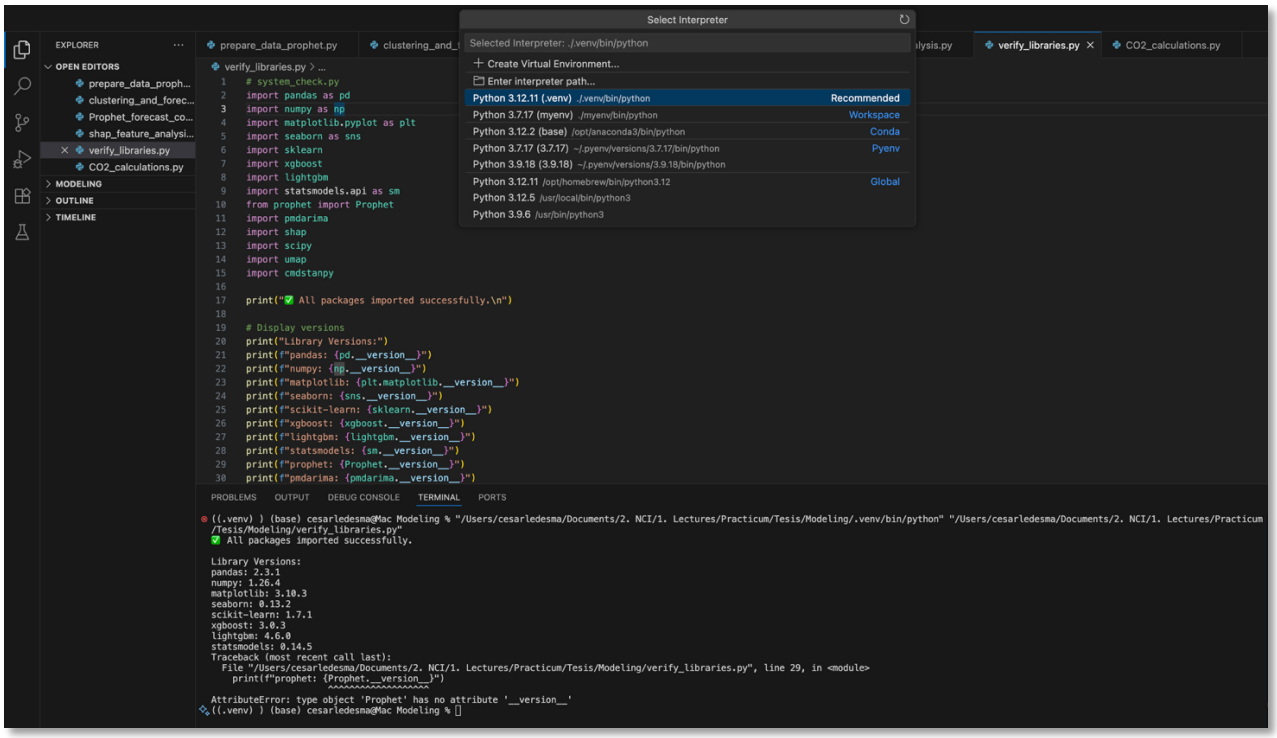


Fig 2. Environment Setup Visual Studio Code®

6 Data Collection

For this project, a public dataset titled “H&M Sales Data” was used, which contains historical sales records from the retail chain. This dataset is suitable for time series forecasting and demand analysis in the fashion retail domain. It can be accessed from Kaggle: https://www.kaggle.com/datasets/tulasiram574/hm-sales-data

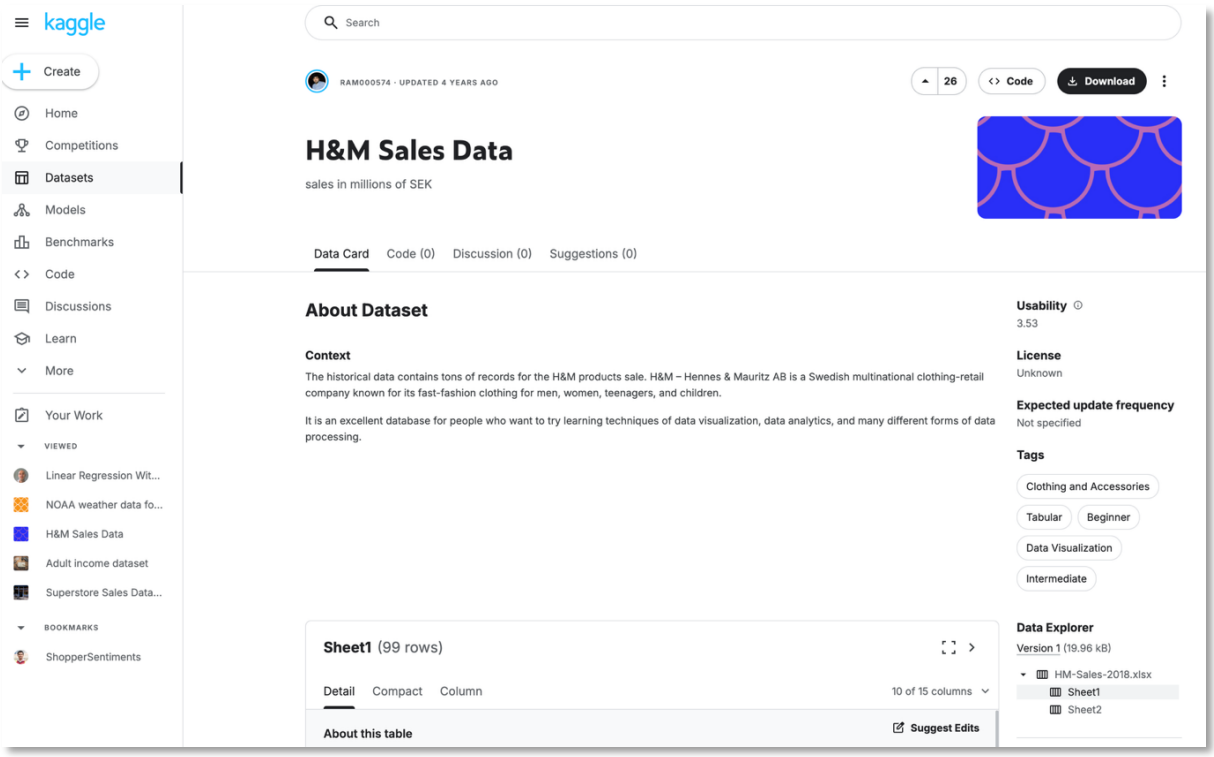
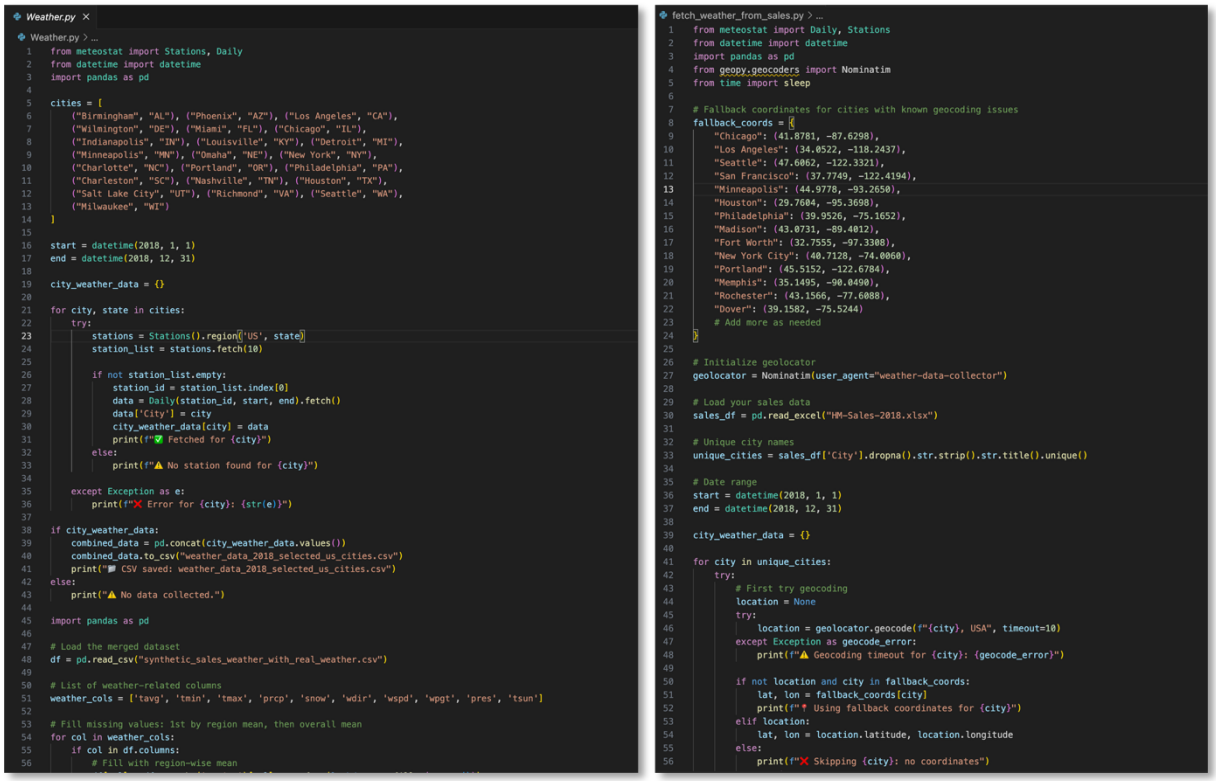


Fig 3. Dataset

7 Data preprocessing

The initial data preparation involved addressing incomplete records by imputing missing values and excluding features such as store identifiers. Temporal attributes were included into a standard data time format before joining the dataset with external weather data. To further refine the dataset and emphasize underlying patterns. The resultant dataset was then exported and saved as *prepared_sales_weather_dataset.csv.*



```python
# Weather.py
from meteostat import Stations, Daily
from datetime import datetime
import pandas as pd

cities = [
    ("Birmingham", "AL"), ("Phoenix", "AZ"), ("Los Angeles", "CA"),
    ("Wilmington", "DE"), ("Miami", "FL"), ("Chicago", "IL"),
    ("Indianapolis", "IN"), ("Louisville", "KY"), ("Detroit", "MI"),
    ("Minneapolis", "MN"), ("Omaha", "NE"), ("New York", "NY"),
    ("Charlotte", "NC"), ("Portland", "OR"), ("Philadelphia", "PA"),
    ("Charleston", "SC"), ("Nashville", "TN"), ("Houston", "TX"),
    ("Salt Lake City", "UT"), ("Richmond", "VA"), ("Seattle", "WA"),
    ("Milwaukee", "WI")
]

start = datetime(2018, 1, 1)
end = datetime(2018, 12, 31)

city_weather_data = {}

for city, state in cities:
    try:
        stations = Stations().region('US', state)
        station_list = stations.fetch(10)

        if not station_list.empty:
            station_id = station_list.index[0]
            data = Daily(station_id, start, end).fetch()
            data['City'] = city
            city_weather_data[city] = data
            print(f"✅ Fetched for {city}")
        else:
            print(f"⚠️ No station found for {city}")

    except Exception as e:
        print(f"❌ Error for {city}: {str(e)}")

if city_weather_data:
    combined_data = pd.concat(city_weather_data.values())
    combined_data.to_csv("weather_data_2018_selected_us_cities.csv")
    print("📁 CSV saved: weather_data_2018_selected_us_cities.csv")
else:
    print("⚠️ No data collected.")

import pandas as pd

# Load the merged dataset
df = pd.read_csv("synthetic_sales_weather_with_real_weather.csv")

# List of weather-related columns
weather_cols = ['tavg', 'tmin', 'tmax', 'prcp', 'snow', 'wdir', 'wspd', 'wpgt', 'pres', 'tsun']

# Fill missing values: 1st by region mean, then overall mean
for col in weather_cols:
    if col in df.columns:
        # Fill with region-wise mean
```

```python
# fetch_weather_from_sales.py
from meteostat import Daily, Stations
from datetime import datetime
import pandas as pd
from geopy.geocoders import Nominatim
from time import sleep

# Fallback coordinates for cities with known geocoding issues
fallback_coords = {
    "Chicago": (41.8781, -87.6298),
    "Los Angeles": (34.0522, -118.2437),
    "Seattle": (47.6062, -122.3321),
    "San Francisco": (37.7749, -122.4194),
    "Minneapolis": (44.9778, -93.2650),
    "Houston": (29.7604, -95.3698),
    "Philadelphia": (39.9526, -75.1652),
    "Madison": (43.0731, -89.4012),
    "Fort Worth": (32.7555, -97.3308),
    "New York City": (40.7128, -74.0060),
    "Portland": (45.5152, -122.6784),
    "Memphis": (35.1495, -90.0490),
    "Rochester": (43.1566, -77.6088),
    "Dover": (39.1582, -75.5244)
    # Add more as needed
}

# Initialize geolocator
geolocator = Nominatim(user_agent="weather-data-collector")

# Load your sales data
sales_df = pd.read_excel("HM-Sales-2018.xlsx")

# Unique city names
unique_cities = sales_df['City'].dropna().str.strip().str.title().unique()

# Date range
start = datetime(2018, 1, 1)
end = datetime(2018, 12, 31)

city_weather_data = {}

for city in unique_cities:
    try:
        # First try geocoding
        location = None
        try:
            location = geolocator.geocode(f"{city}, USA", timeout=10)
        except Exception as geocode_error:
            print(f"⚠️ Geocoding timeout for {city}: {geocode_error}")

        if not location and city in fallback_coords:
            lat, lon = fallback_coords[city]
            print(f"📍 Using fallback coordinates for {city}")
        elif location:
            lat, lon = location.latitude, location.longitude
        else:
            print(f"❌ Skipping {city}: no coordinates")
```

Fig. 4 Data preprocessing pipeline overview

8 Instructions to run the Model

4.1 SHAP Feature Importance

The SHAP analysis was performed to interpret the feature influence on the sales prediction model. The model was fitted to the pre-processed dataset and SHAP values were generated to assess the effect of variables including "*Temperature", "Humidity"* and "*Day of the Week"* upon the sales forecast. The output includes the summary plot, which orders the features according to their marginal contribution, this clarifying and enhancing interpretability of the model and facilitating informed choices regarding feature retention and actionable insights for business strategies.

- SHAP Feature Importance Analysis:

```bash
python shap_feature_analysis.py
```

shap_feature_analysis.py > ...

```python
import pandas as pd
import shap
import xgboost as xgb
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

# Load dataset
df = pd.read_csv("prepared_sales_weather_dataset.csv")

# Prepare data
features = ['tavg_normalized', 'prcp_normalized', 'wspd_normalized', 'Month']
df = pd.get_dummies(df, columns=["Sub-Category"], drop_first=True)
X = df[features + [col for col in df.columns if "Sub-Category_" in col]]
y = df["Sales"]

# Train model
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = xgb.XGBRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# SHAP analysis
explainer = shap.Explainer(model)
shap_values = explainer(X_train)

# Plot
shap.summary_plot(shap_values, X_train, show=False)
plt.tight_layout()
plt.savefig("shap_summary_plot.png")
plt.close()
```

Fig 5. SHAP feature analysis

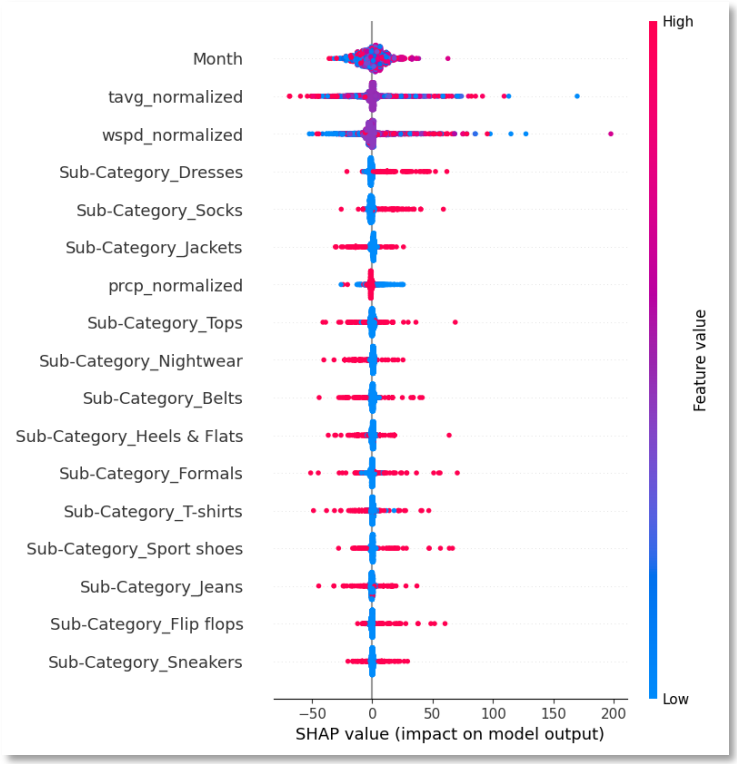


Fig 6. SHAP expected output plot

4.2 Forecasting and Clustering Prophet

This step focuses on builds a robust forecasting model using Prophet both at an aggregate and subcategory level. It includes applying clustering techniques to group similar subcategories, which allows more specific forecasts. Seasonal patterns are also examined to enhance temporal understanding and model granularity.

```bash
python Prophet_forecast_comparison.py
```

Expected Output:

- **Forecast plots:** store in outputs/
- **SHAP summary plot:** shap_summary_plot.png
- Forecast RMSE logs printed to console

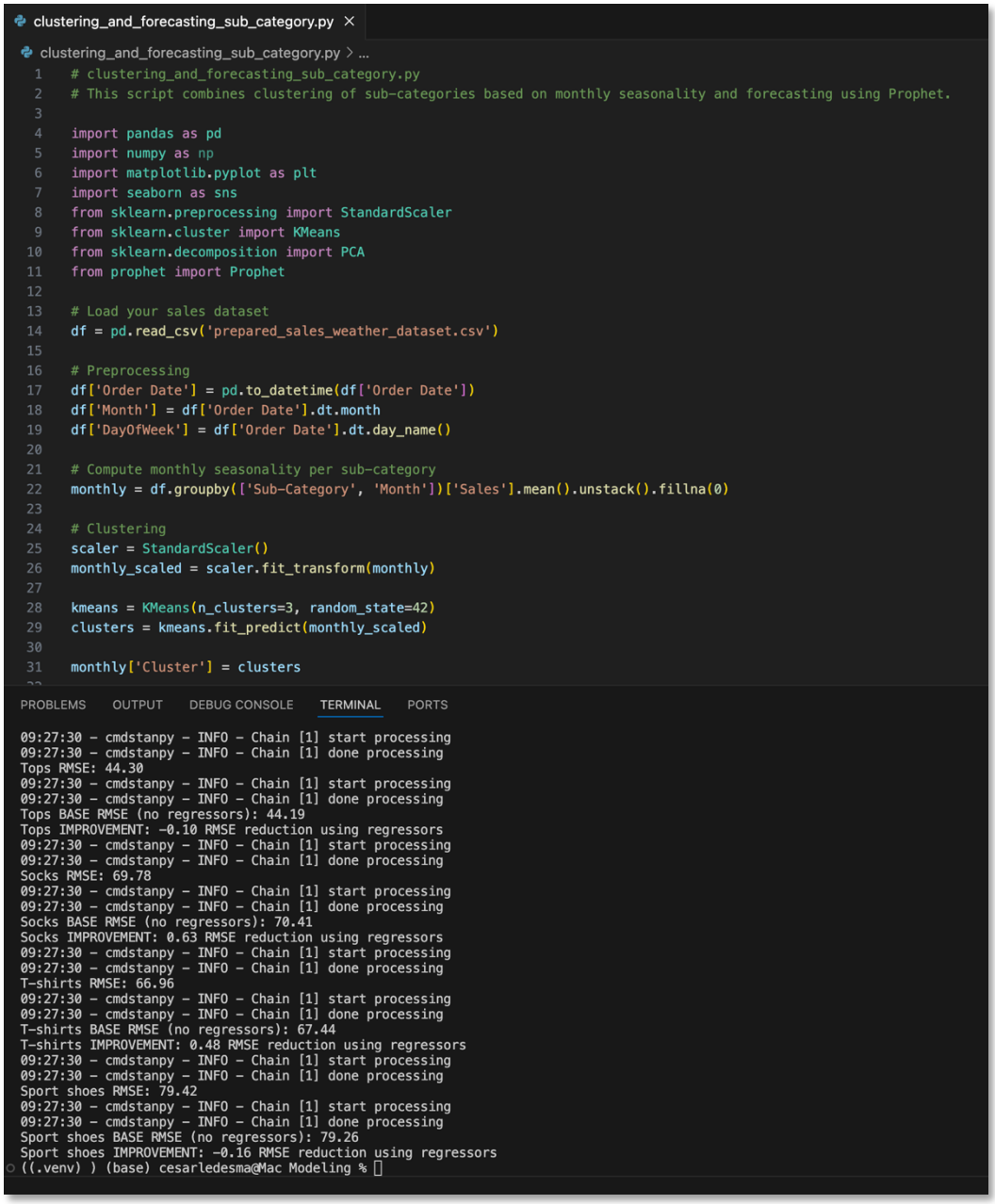


Fig 7. Forecasting and Clustering with Prophet

4.3 Model Evaluation

To assess the performance of the forecasting models generated in this work modelevaluationplot.py was facilitated. Performance indicators such as RMSE and MAE, the script enable a comparison of forecasting accuracy among the Prophet model configurations and the clusters. The visualisations clarified the performance of predictive subcategories, affirming the contributions of feature importance, the design of clustering scheme, and incorporation of seasonality.

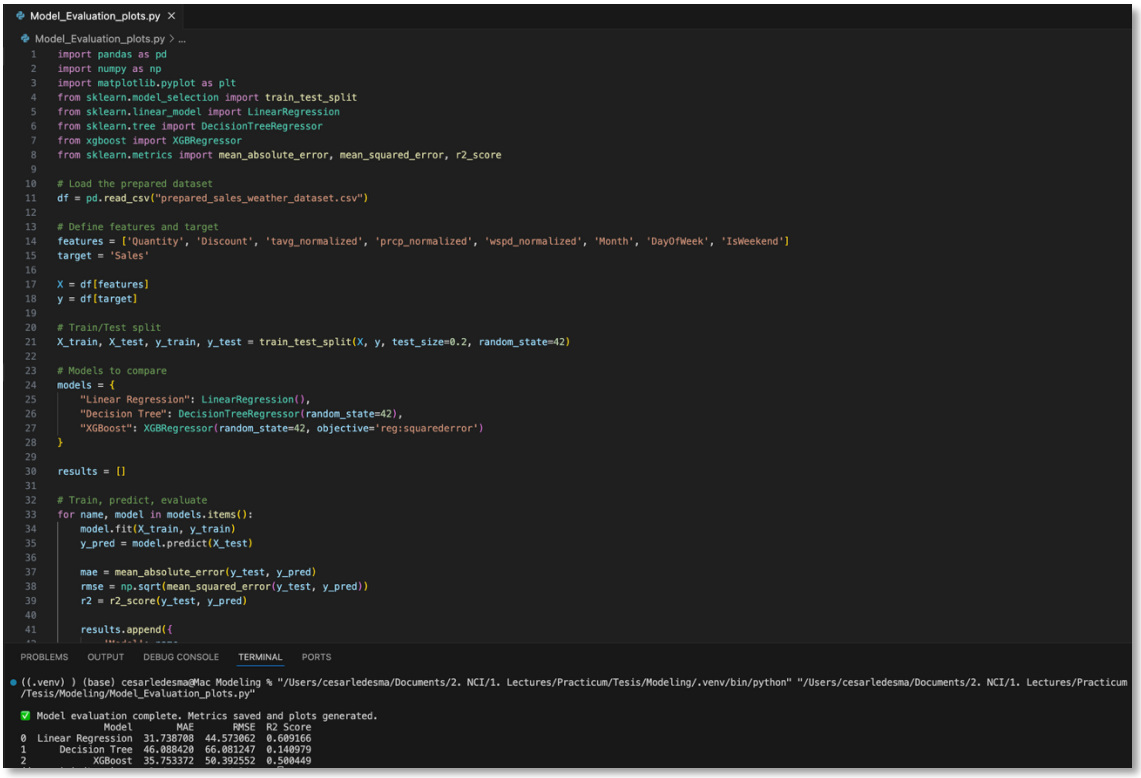


Fig 8. Model Evaluation script

4.4 Sustainability Impact Calculations

The final stage measures the environmental impact of improved forecasting. By considering estimations of potential CO_2 emissions, offering a quantification for sustainable computational operations.

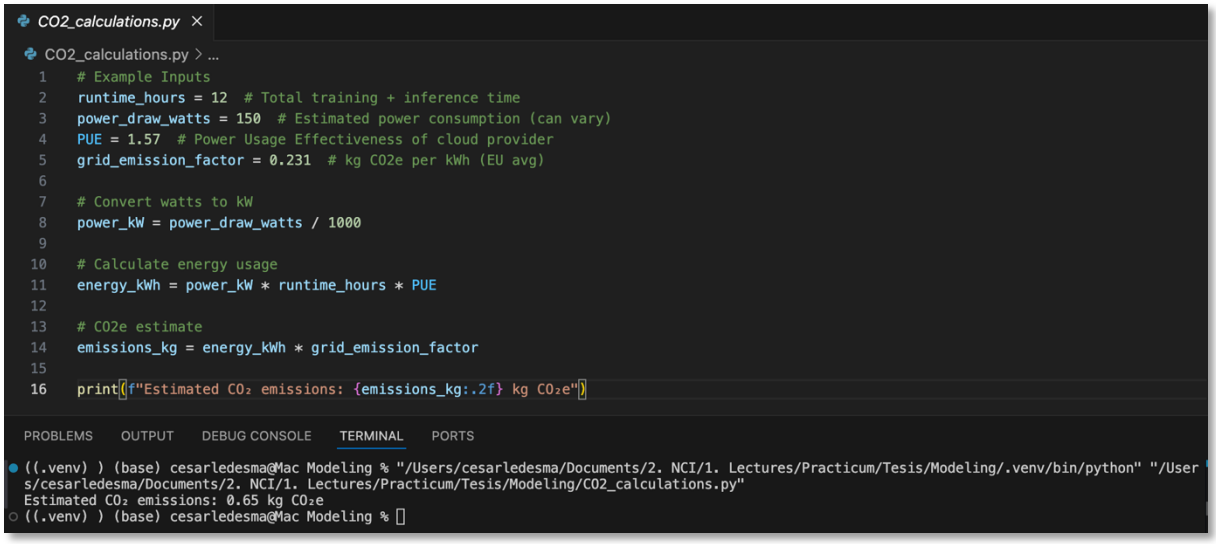


Fig 9. CO2 Calculations script

9 Troubleshooting Tips

Issue	Solution
`KeyError: 'Date'`	Ensure the CSV has a properly formatted `Date` column. Check with `df.columns`
`TypeError: unexpected keyword argument 'square'`	Use `mean_squared_error(...,squared=False)` with compatible sklearn version
`ModuleNotFoundError: No module name 'Prophet'`	Run `pip install prophet`. For M1/M2 chips, install dependencies like `cmdstanpy` first.
`SHAP slow to compute`	Use `X_train.sample(1000)` to reduce memory load
`Plot doesn't display`	Use `plt.savefig()` instead of `plt.show()` when running from scripts

References

Breiman, L. (2001). Random Forests. Machine Learning, 45(1), pp.5–32. https://doi.org/10.1023/A:1010933404324

Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp.785–794. https://doi.org/10.1145/2939672.2939785

Lundberg, S.M. and Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. In: Advances in Neural Information Processing Systems 30 (NIPS 2017). [online] Available at: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

Taylor, S.J. and Letham, B. (2018). Forecasting at Scale. The American Statistician, 72(1), pp.37–45. https://doi.org/10.1080/00031305.2017.1380080

Pedregosa, F. et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, pp.2825–2830. https://jmlr.csail.mit.edu/papers/volume12/pedregosa11a/pedregosa11a.pdf

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In: van der Walt, S. and Millman, J. eds., Proceedings of the 9th Python in Science Conference. Austin, TX: SciPy, pp.51–56.

Prophet (n.d.). Facebook Prophet Documentation. [online] Available at: https://facebook.github.io/prophet/ https://peerj.com/preprints/3190/

SHAP (n.d.). SHAP Documentation. [online] Available at: https://shap.readthedocs.io/

NCI Library (n.d.). NCI Referencing Guide. [online] Available at: https://libguides.ncirl.ie/referencingandavoidingplagiarism

E. Hernández, An Ensemble Model to Predict the Classification of Goods Using Text Description, M.Sc. thesis, School of Computing, National College of Ireland, Dublin, Ireland, 2023. [Online]. Available: https://norma.ncirl.ie/id/eprint/7526