# Group ID - MSc in Data Analytics

Author: Afshan Mehmood

e-mail: 2025102@student.cct.ie

Student ID: 2025102

## Abstract

*My dataset comprises the Expenditure of Foreign Resident Overnight Visitors, sourced from the Inbound Tourism Series available on the CSO (Central Statistics Office) website. The data spans from January 2023 to February 2025. Using this dataset, I have analyzed the expenditure patterns of foreign resident visitors to identify which regions contribute the most to overall tourism expenditure. Additionally, I examined monthly expenditure trends over nearly two years and studied how different types of expenditures (e.g., accommodation, fares) contribute to the total. The analysis includes Exploratory Data Analysis (EDA), data preparation and visualization, as well as statistical evaluation. I have also applied machine learning models to support my findings. These insights can assist in enhancing Ireland’s tourism revenue by informing data-driven marketing strategies and policy making.*

## Introduction

My report will majorly comprise four sections. These sections will correspond to the coding sections in my ipnyb code file.

1. Data Preparation and Visualization

2. Machine learning for Data Analytics

3. Statistics for Data Analytics

4. Programming for Data Analytics

First I will start with the Question Formulation part i.e. through my dataset analysis, what type of questions can arise:

## Question Formulation:

The dataset examines the monthly expenditure of overnight foreign visitors in Ireland, spanning from January 2003 to February 2025.

Based on the dataset, following questions can be formulated:

**Question 1:** How has the total expenditure of overnight foreign visitors in Ireland changed over time?  
**Relevant Techniques:** trend analysis (line plots), seasonal decomposition.

**Question 2:** What is the distribution of expenditure by residency group (e.g., USA & Canada, Great Britain, Other Europe)?  
**Relevant Techniques:** Grouping by residency, box-plots.

**Question 3:** Which expenditure categories (e.g., accommodation, travel, prepayments) dominate the spending of foreign visitors?  
**Relevant Techniques:** Grouping by expenditure type, bar plots.

**Question 4:** Are there any seasonal variations in the expenditure of overnight foreign visitors in Ireland?  
**Relevant Techniques:** Month-over-month comparison, seasonality analysis.

**Question 5:** How does the expenditure of foreign visitors relate to the number of visitors from each residency group?

**Relevant Techniques:** Correlation analysis, scatter plots, regression.

**Question 6:** Based on historical data, can we predict the future expenditure of foreign visitors?  
**Relevant Techniques:** Regression models (e.g., linear regression, Random Forest regression).

### **Question 8:** Are there any extreme spending behaviors (outliers) that significantly impact the overall expenditure data? **Relevant Techniques:** Boxplots, IQR-based outlier detection.

**Question 9:** Can we predict the expenditure of foreign visitors based on their residency and expenditure type?  
**Relevant Techniques:** Classification models (e.g., Random Forest, Decision Trees), regression models.

## Section1: Data Preparation and Visualisation

I will discuss the Exploratory Data Analysis and Data Preparation part. The visualizations according to Tufts principle will follow on throughout the report.

### Exploratory Data Analysis:

*Understanding my data structure:*

1.Initializing libraries

2.Reading the csv file: df=pd.read\_csv("exp.csv")

3.For displaying top 5 rows of my data set:

df.head()



I get the following insights:

* **Categorical columns**: Residency, Statistic Label, Expenditure Type, UNIT
* **Numeric column**: VALUE
* **Time column**: Month

Each row gives a **Value** for a specific **Statistic Label** along with the **Expenditure Typ**e and region of **Residency**, for a particular **Month.**.

Right now the data is in long format.

*Initial Checks:*

1. Checking the data types:

df.dtypes

**Statistic Label object**

**Month object**

**Residency object**

**Expenditure Type object**

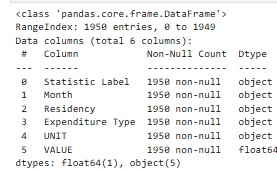
**UNIT object**

**VALUE float64**

This shows that except the VALUE column, all other columns have non-numeric entries since the data type is object.

2. To get a summary of a DataFrame, including **Class type**, **Index range**, **Column names**, **Non-null count** and **Data types**:

df.info()



This shows there are 1950 entries in total i.e. 1950 rows and 6 columns.

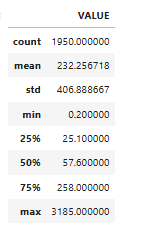
3. To find dimensions of the dataset:

df.shape

I get (1950,6) confirming no of rows and columns.

4.To provide summary statistics for the **numeric columns** in a DataFrame by default.

df.describe()



It computes a set of descriptive statistics that are useful for understanding the distribution and spread of numerical data.

However, in my case, the VALUE column has different values for different Statistic Labels accordingly.

5. There are unique values in the **'Statistic Label', ‘Residency’ and ‘Expenditure Type’** columns of the DataFrame. In order to see all the distinct values present in those specific columns:

unique\_Stat\_Label = df['Statistic Label'].unique()

Within the Statistic Label, I have three unique values:

Expenditure of Overnight Foreign Visitors', 'Percentage of Expenditure of Overnight Foreign Visitors', 'Mean Expenditure of Overnight Foreign Visitors

Then,

unique\_residency = df['Residency'].unique()

Within the Residency, I have five unique values:

'Great Britain (England, Scotland & Wales)', 'Other Europe (3)', 'USA & Canada', 'Other Residencies', 'All Residencies'

Next,

unique\_exp\_type=df['Expenditure Type'].unique()

Within the Expenditure Type, I have five unique values:

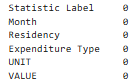
Fare', 'Prepayments', 'Accommodation', 'Day-to-Day Expenditure', 'All Travel Expenditure’

This tells that my data is in long format and I need to arrange it in a wide format to do analysis in terms of Expenditure Type, Residency, etc.

*Check for missing values:*

To check for missing values (null or NaN values) in my DataFrame:

df.isnull().sum()



This shows there are no missing values in my DataFrame.

In my report, I will discuss following methods of EDA:

*1.Univariate Non-Graphical EDA (Numerical Data):*

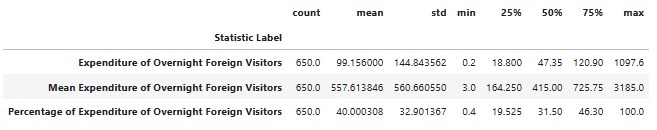
Descriptive Statistics:

The reason for applying descriptive statistics (count, mean, std, min, quartiles, max), is that it will give me understanding about my data in terms of central tendencies and dispersion. It will also help me to identify possible outliers or unusual ranges and help me understand each metric's distribution before modeling.

In order to check the general distribution of each Statistic Label:

df.groupby("Statistic Label")["VALUE"].describe()

Here, df.groupby("Statistic Label")["VALUE"].describe() will group the dataset df by the Statistic Label column and then apply the describe() function to the VALUE column within each group. The describe() function will return key statistics for each group.



### Key Insights:

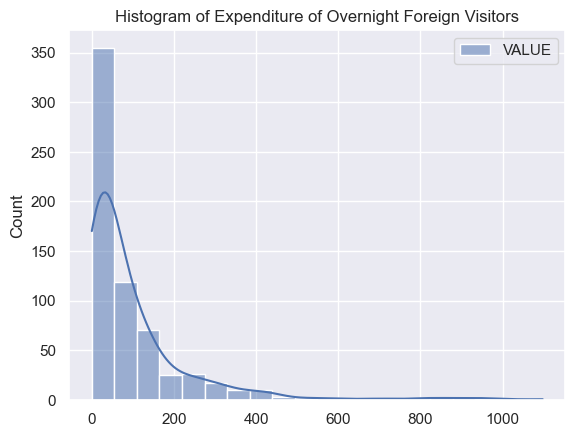
* High variability: The high standard deviations in all three expenditure-related columns suggest that the data has a broad range of values. This variability is important to keep in mind when performing analyses or building models.
* Possible outliers: The minimum and maximum values in each column suggest the presence of outliers. For example, expenditures as low as 0.2 or as high as 3185.0 indicate extreme cases that may need to be handled separately (e.g., outlier detection, capping, or transformation).
* Skewness: Given the large range between the 25th percentile (Q1) and the 75th percentile (Q3), the data may be right-skewed, especially for the "Expenditure of Overnight Foreign Visitors" column, where a small number of observations (the upper quartile) have disproportionately high values.
* Inbound tourism spending shows strong growth and significant contribution despite excluding fares. This information can be used by the tourism department in a practical scenario to develop programs for improving the tourism industry and increasing tourism revenue in the future. (There is an opportunity to increase in-country spending through premium experiences, longer stays, and diversified travel offerings.)
* This information can be useful for trend forecasting.

*2. Univariate Graphical EDA (Numerical Data):*

In order to visualize the distribution of individual numerical variables to identify patterns, outliers, or skewness.

#### Histogram:

Plotting a histogram to visualize the distribution of the Expenditure of Overnight Foreign Visitors. This helps assess whether the data is normally distributed or if there are any skewed patterns.



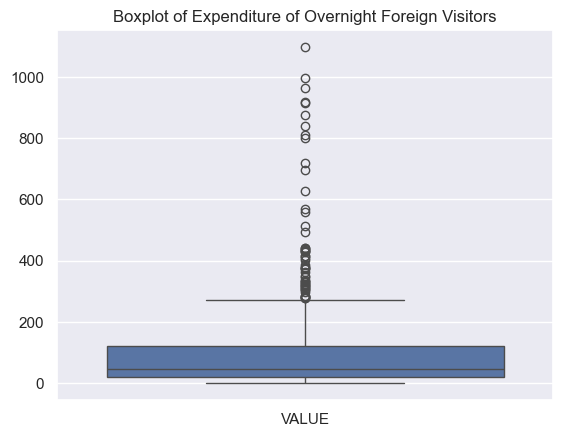
The histogram allows us to see the frequency of different expenditure values and whether they follow a specific distribution. In the above plot, it shows positively skewed distribution. In the right-skewed distribution, most of the data is clustered towards the lower end, but there is a long tail extending towards the higher values. This means there are a small number of observations with very high expenditures.  
Moreover, the bulk of the visitors have low or moderate expenditure.

Impact on analysis: The mean (average) expenditure will be higher than the median due to the influence of the higher values in the right tail. Further, the long right tail indicates the presence of outliers or a few very high-spending visitors, which significantly increase the mean expenditure. These outliers could be wealthy tourists, special events, or high-budget packages.

Right-skewed data can often be normalized or made more symmetric through a logarithmic transformation. When applying machine learning models or statistical techniques, skewness can influence predictions. The high spenders (in the right tail) could represent a specific segment of the population (e.g., luxury tourists, corporate travelers, or special events). Understanding their behavior might provide insights into targeted marketing or policy recommendations.

Boxplot:

The boxplot can be used to visualize the spread and identify any outliers in the expenditure data.



In the above plot for ‘**Expenditure of Overnight Foreign Visitors’**, there are significant outliers, showing values that are significantly different from the majority of the data, Outliers above the upper whisker typically indicate extremely high-spending visitors. These could be wealthy tourists, special groups (e.g., business tourists, luxury travelers), or large organized travel events with higher budgets.  
The high outliers may represent premium tourists who spend significantly more than the average visitor.

### Impact on Analysis: Outliers can increase the mean expenditure significantly, making it higher than what most of the visitors actually spend. The median is usually less influenced by outliers and may provide a more reliable measure of central tendency in this case. Also, the presence of outliers often leads to a right-skewed distribution (as already observed), where the majority of the data lies on the lower side, but a small number of very high values distort the overall shape of the distribution.

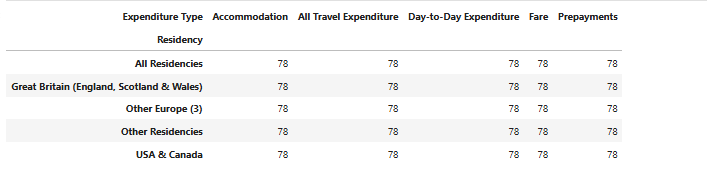
Applying a log transformation to the data can reduce the effect of extreme values and bring the distribution closer to normal.Understanding the nature of the outliers can lead to more accurate interpretations of the data and effective decision-making in tourism management.

*3. Bivariate Non-Graphical EDA (Categorical & Quantitative Variables):*

In order to analyze the relationship between one categorical explanatory variable (e.g., Residency) and one quantitative outcome variable (e.g., Expenditure of Overnight Foreign Visitors), we can use different methods like cross-tabulation or correlation.

Cross-tabulation:

It is a method for examining the relationship between two categorical variables. In my case, Residency and Expenditure Type are categorical, while Expenditure is numerical.

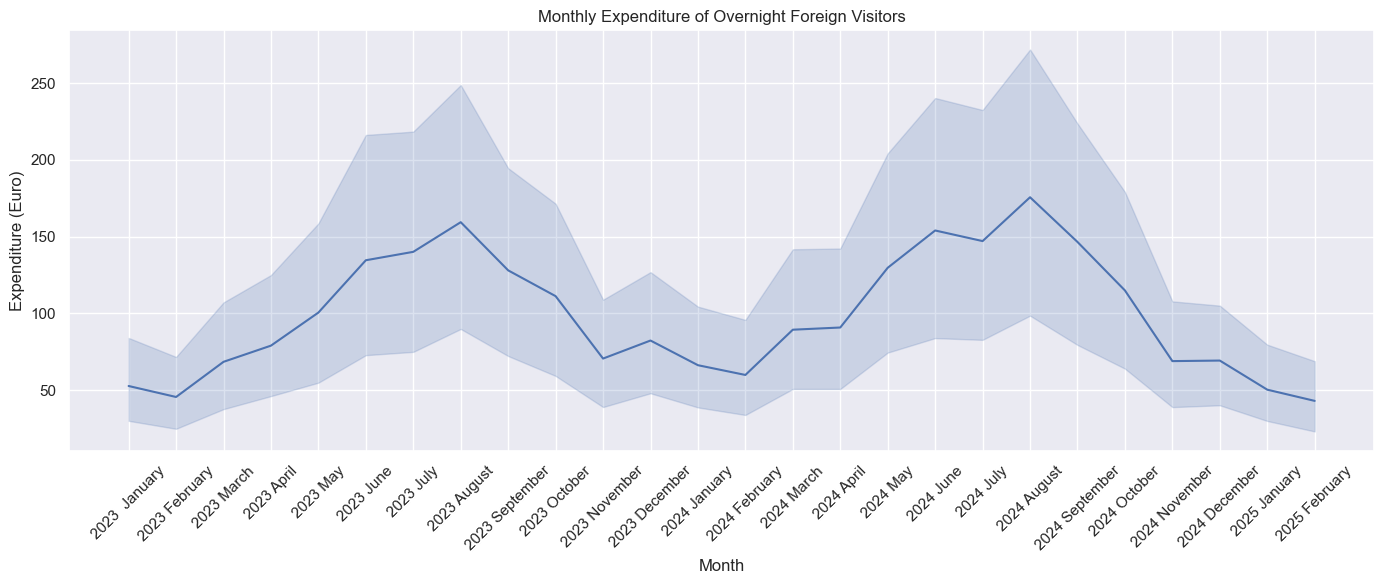


As it can be noted, this helps to observe the frequency of each category combination and analyze the distribution of Expenditure Type across Residency groups.

*4. Bivariate Graphical EDA:*

Line plot:

We can see the monthly expenditure trend using a line plot.



This shows the trend for monthly Expenditure of Overnight Foreign Visitors. We can see that the highest expenditure occurs in August 2003 and August 2004.

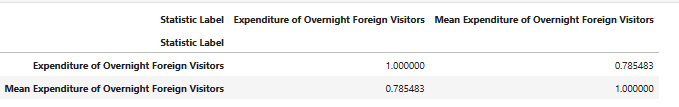
This statistic can be used to enhance tourism activities in Summer months based on the high expenditure during summer months.

*5.Multivariate Non-Graphical EDA:*

Used in order to analyze the relationships between multiple variables, including both categorical and numerical.

#### Correlation Matrix (for Numerical Variables):

We can create a correlation matrix to examine the linear relationships between all numerical variables in our dataset (e.g., Expenditure of Overnight Foreign Visitors, Mean Expenditure, etc.).



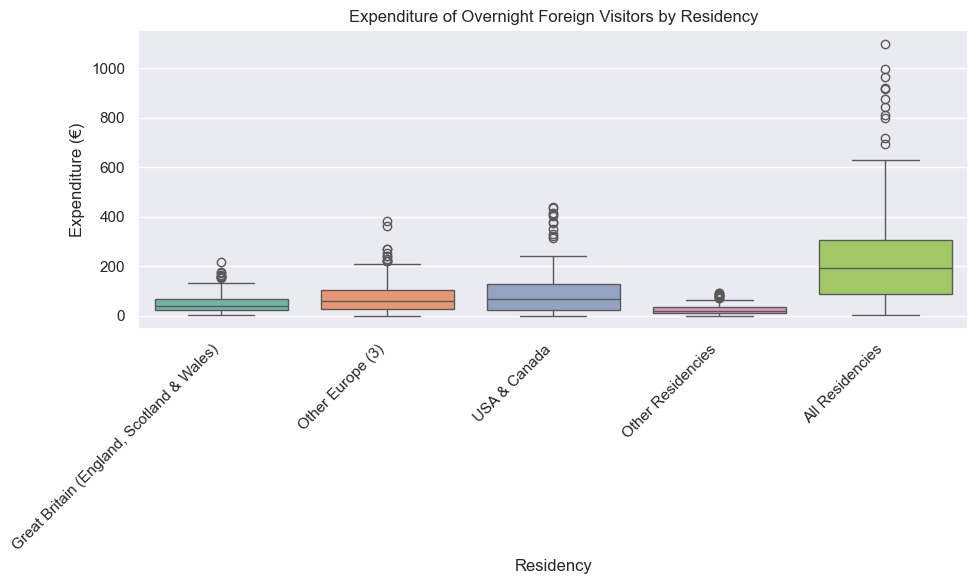
A correlation matrix helps identify strong or weak relationships between numerical features, which can guide feature selection for modeling. It can be noted from the statistics above that mean expenditure and expenditure of overnight foreign visitors is highly correlated. As the total expenditure by overnight foreign visitors increases, the mean (average) expenditure per visitor also tends to increase.

Policy-wise, if tourism initiatives increase visitor spending overall, it’s likely to impact per-person spending as well, this in turn is useful for targeting premium tourism.

### *6. Multivariate Graphical EDA:*

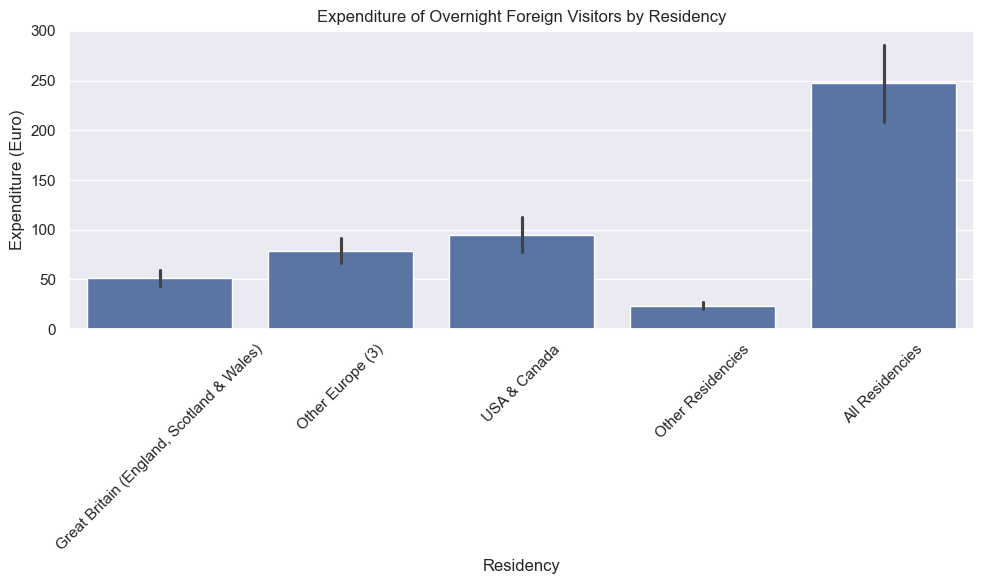
Used in order to visualize relationships between a categorical variable and quantitative variables to understand how expenditure differs across categories.

Side-by-side boxplots allow comparison of the Expenditure of Overnight Foreign Visitors across different Residency categories. This can help identify which residency group has the highest or lowest expenditure.



The above figure shows the highest expenditure with All residencies and highest number of outliers. Since this is an aggregate we can remove this category for further analysis. Besides this, we see the highest expenditure from residents of the USA and Canada.

Besides this, barplot also provides meaningful insights into how expenditure by overnight foreign visitors in Ireland varies by residency group.

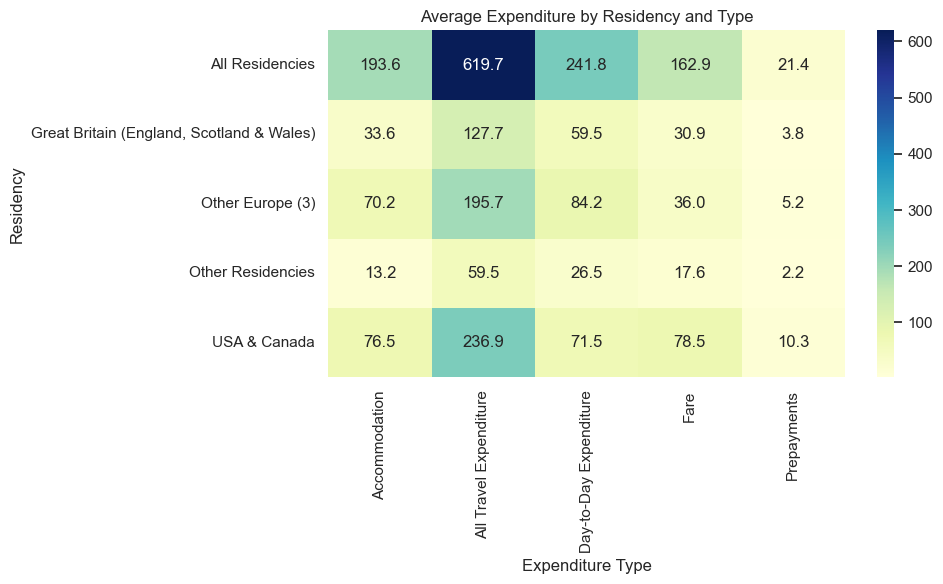


In the above plot, bars represent the mean expenditure for each group while error bars (95% ci) represent the confidence interval, giving a sense of variation and reliability in the mean expenditure.  
The height of each bar tells us which residency group spends more on average during overnight visits. For example, in the above plot, *USA & Canada* category has the tallest bar which suggests that visitors from those areas tend to spend more. Although the ‘All Residencies’ category has the highest bar, however, since it is just showing an aggregate therefore, we can drop it in further analysis later.

Further, narrow confidence intervals (short error bars) represent more consistent spending behavior across the months. While wide confidence intervals show higher variation in expenditure by that group.

Policy Implications: If some groups spend more consistently, marketing or tourism campaigns can be tailored to attract them.

Heatmap: It is also a way to see how different features are correlated. The below plot shows apart from All Residencies, USA and Canada and All Travel Expenditure categories are highly correlated. This means that people from these area spend more on All Travel Expenditures.



Now we will discuss the next part of data preparation.

Data Preparation:

The dataset was initially in a long format, with each row representing a combination of:

Month, Expenditure Type, a Statistic Label, Residency, a corresponding VALUE and a UNIT. While this structure is ideal for flexible analysis and visualization, it is not directly suitable for machine learning models, which typically require one row per observation and one column per feature.

Therefore, the first step is to arrange the data in a wide format.

1. Combine Statistic Label with Unit for clearer column headers

df['Stat\_Unit'] = df['Statistic Label'] + ' (' + df['UNIT'] + ')'

This creates a new column Stat\_Unit by combining the existing Statistic Label and UNIT.

2. Pivot the table

df\_wide = df.pivot\_table(

index=['Month', 'Residency', 'Expenditure Type'],

columns='Stat\_Unit',

values='VALUE',

aggfunc='first' # if duplicates exist, take the first

).reset\_index()

This converts the data from long format to wide format. Now each unique value in Stat\_Unit becomes a new column (instead of rows). The new table has rows indexed by: Month, Residency, Expenditure Type and columns as: Different statistics like Expenditure, Mean, etc., now spread across columns.

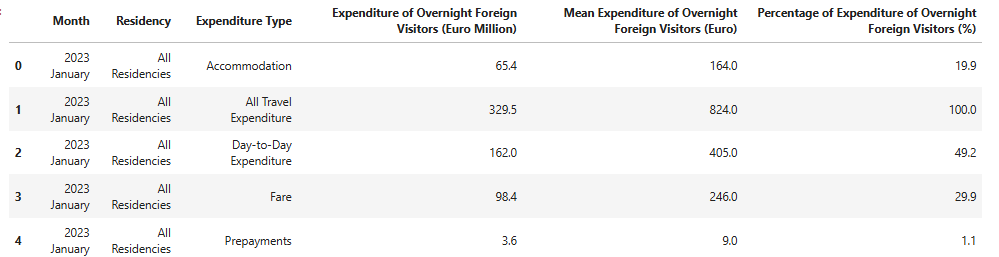
3. Flatten the column names (optional if you get a multi-index)

df\_wide.columns.name = None # remove the name of the columns axis

This removes the column name label (which can appear in pivoted DataFrames) and prevents multi-level column headers. It also makes the DataFrame cleaner and easier to work with.

Overall, this transformation reshapes the dataset into a wide format, where each row corresponds to a unique combination of Month, Residency, Expenditure Type and each Statistic Label also becomes a separate column. This is essential for supervised ML models, which expect a consistent feature layout.

The data now looks like this:



Next we see the dimensions of the dataset after converting into wide format:

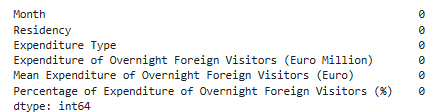
df\_wide.shape

This gives (650, 6) showing 650 rows and 6 columns.

4. Check for missing values:

df\_wide.isnull().sum()

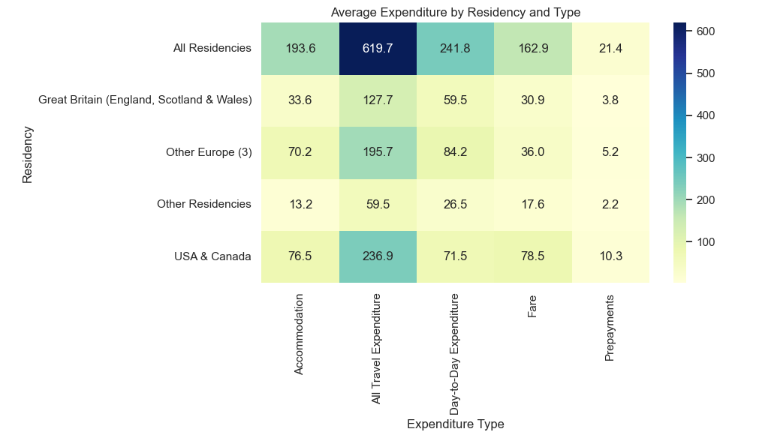
The output is this:



This indicates that the data does not have any missing values.

5. For understanding the relationship between my categorical and numerical values, a heatmap is plotted.

sns.heatmap(pivot, annot=True, cmap='YlGnBu', fmt='.1f')



Here, we observe that 'All Residencies' represents an aggregate summary rather than distinct analytical data, so it can be excluded from the analysis.

6. Now, we remove the unwanted Residency categories after pivoting

df\_wide\_filtered = df\_wide[~df\_wide['Residency'].isin(['All Residencies'])]

Df\_wide\_filtered is the filtered dataset after removing All Residencies.

When I see the shape now:

df\_wide\_filtered.shape

It shows (520, 6) indicating reduction in rows after removal.

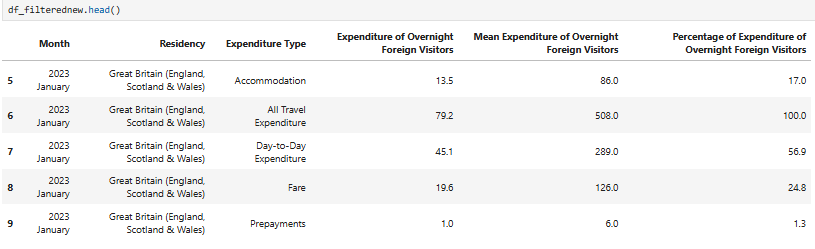
7. Since my dataset contains monthly values from January 2003 to February 2025, I will exclude January and February 2025 to retain only complete years. This allows for consistent year-by-year analysis later on.

df\_filterednew = df\_wide\_filtered[~df\_wide\_filtered['Month'].isin(['2025 January', '2025 February'])]

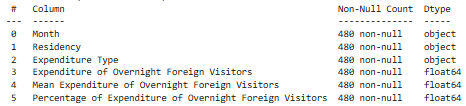
This will remove the rows where the month is January or February 2025. Now looking at the shape of my dataset shows that rows have further reduced to 480.

8. Arranging the Month category into a proper format:

The current form of dataset looks like as follows:



If I look at the data types using, df\_filterednew.info(), I get below information:



It can be seen from above that Month is object data type and in the Month column, the values are like this: ‘2023 January’, ‘2023 March’, etc. Also they are not sorted.

So to convert Month to datetime format:

First, clean and convert the Month column safely:

df\_filterednew = df\_filterednew.copy() # Make an explicit copy to avoid chained assignment warning

df\_filterednew.loc[:, 'Month'] = df\_filterednew['Month'].astype(str).str.strip()

df\_filterednew.loc[:, 'Month'] = pd.to\_datetime(df\_filterednew['Month'], format='%Y %B')

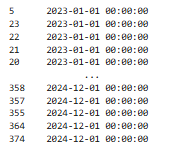
This creates an explicit copy of the DataFrame to avoid modifying the original by accident. Further, it ensures the Month values are converted to string type. Then. str.strip() removes any leading/trailing whitespace, which might cause issues during datetime conversion.

Finally, in order to convert the Month column from string format like "2023 January" to proper datetime objects (e.g., 2023-01-01), we use the format '%Y %B that' means: year (4-digit) followed by full month name (e.g., "January").

Next we sort the data and view it:

df\_filterednew = df\_filterednew.sort\_values('Month')

df\_filterednew['Month']



9. Encoding the data:

We need to encode our categorical columns to be used in the Machine Learning models.

# One-hot encode categorical columns

df\_encoded = pd.get\_dummies(df\_filterednew, columns=['Residency', 'Expenditure Type'], drop\_first=True)

Here pd.get\_dummies() is a function in pandas that converts categorical columns into a set of binary columns (i.e., one-hot encoded columns). For each unique category in "Residency", a new column is created with a 1 or 0 to indicate whether that row belongs to that category.

Here I am encoding the 'Residency' and 'Expenditure Type' columns because most machine learning models require numerical inputs and they can not directly process categorical (text) data. Further, I have chosen One-Hot Encoding here because these columns are nominal (no natural order), e.g., 'USA & Canada', 'Great Britain', etc. One-Hot Encoding creates a binary column for each category, preserving neutrality and it does not assume any ordinal relationship.

The reason for not choosing Label Encoding is that it assigns integers (e.g., 'USA & Canada': 0, 'Great Britain':1), which implies order where none exists. This can mislead the algorithms into thinking one category is greater or less than another.

In another step, I extract the year and month number values from my Month column.

# Extract year and month from the 'Month' column

df\_encoded['Year'] = df\_encoded['Month'].dt.year

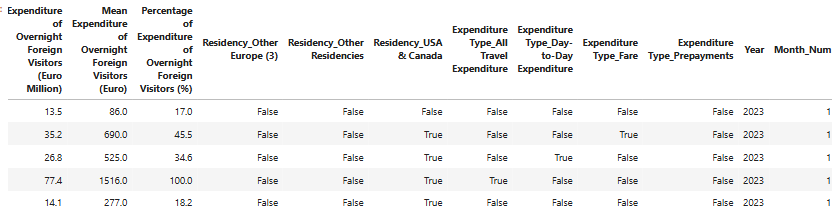
df\_encoded['Month\_Num'] = df\_encoded['Month'].dt.month

# Drop the original 'Month' column as it's no longer needed

df\_encoded.drop(columns='Month', inplace=True)

Finally, the original Month column is removed, as it is no longer necessary after extracting the year and month number. Breaking down the date into Year and Month\_Num helps in seasonal or trend analysis, provides useful features for machine learning models, and offers greater flexibility for grouping and visualization, such as by year or month.

The dataset looks like this now:



10. Scaling the data:

From earlier analysis, it was seen using histogram and box plot that ‘Expenditure of Overnight Foreign Visitors’ is positively skewed and has outliers, making it necessary to standardize it to a common scale.

Scaling expenditure values is crucial because they have different magnitudes, which can affect the performance of machine learning algorithms.

# Scale expenditure values

scaler = StandardScaler()

df\_encoded1[Expenditure of Overnight Foreign Visitors (Euro Million)

'] = scaler.fit\_transform(df\_encoded[['Expenditure of Overnight Foreign Visitors (Euro Million)

']])

Here the StandardScaler is a tool from scikit-learn used to standardize (or z-score normalize) your data. It transforms each feature (i.e., column) so that the mean becomes 0 and the standard deviation becomes 1.The result is a NumPy array where all the numeric features are now on the same scale. Each column is now centered around 0 and spread equally. This is required for applying Machine Learning models that are sensitive to scale as they work better when features are on a similar scale.

For my dataset, scaling the "Expenditure of Overnight Foreign Visitors" allows for better comparison and improved model performance, especially in algorithms sensitive to the scale of data, such as regression and clustering.

**Part3. Visualization:**

**Part4. Tufts principle:**

**The above are incorporated throughout the report and code file.**

Now we will move to the Machine learning section:

## Section2: Machine Learning for Data Analytics

First I will discuss the project framework, then technique selection and then about machine learning models.

Project Management Framework:

When working on any data science project, it is crucial to follow a structured framework to guide the analysis, model development, and deployment phases. The CRISP-DM (Cross-Industry Standard Process for Data Mining) framework is widely regarded as one of the most flexible and comprehensive frameworks for data science projects. Below, I will discuss CRISP-DM, KDD (Knowledge Discovery in Databases), and SEMMA (Sample, Explore, Modify, Model, and Assess), and explain why CRISP-DM is the most suitable for my project.

#### CRISP-DM (Cross-Industry Standard Process for Data Mining):

**CRISP-DM** is a robust and well-established framework for data mining and data science projects. It is structured around six phases:

1. Business/Research Understanding: In this phase, we define project requirements and objectives. Then translate objectives into data exploration problem definition and finally prepare a preliminary strategy to meet objectives.
2. Data Understanding: This phase involves collecting data and performing Exploratory Data Analysis (EDA). We also assess data quality and optionally, select interesting subsets.
3. Data Preparation: In this phase, data is prepared for modeling in subsequent phases. It involves selecting cases and variables appropriate for analysis. We clean and prepare data so it is ready for modeling tools. Also, we perform transformation of certain variables, if needed.
4. Modeling: This phase involves selecting and applying one or more modeling techniques. We can also calibrate model settings to optimize results.
5. Evaluation: In this phase, we evaluate one or more models for effectiveness and determine whether defined objectives were achieved. Decisions are made regarding data exploration results before deploying to field
6. Deployment: This phase involves making use of models created i.e. the final model is deployed in a real-world scenario

* Simple deployment example: generate report
* Complex deployment example: implement parallel data exploration effort in another department
* In businesses, customer often carries out deployment based on the developed model

Justification for using CRISP-DM:

Comprehensive & Iterative: CRISP-DM's iterative approach allows flexibility in revisiting earlier stages based on model performance, which is especially important in real-world applications where initial results may not meet expectations.

Domain Agnostic: CRISP-DM can be applied to any industry, making it ideal for my tourism data analysis project. Whether it is analyzing expenditure trends, forecasting tourist behavior, or segmenting customers, CRISP-DM helps tailor the project to specific business goals.

Relevance to real-life Scenario: For instance, in my project for tourism analysis, it involves all the six phases of SEMMA.

Business Understanding: This phase focuses on understanding my project objectives and requirements from a business perspective, ensuring that the data science effort aligns with the business goals. E.g. Understanding customer behavior (like expenditure of foreign visitors) involves first defining business goals (like increase in tourism revenue, targeted marketing) by analysing foreign visitors data.

Data Understanding: Gathering data and understanding the initial data quality and potential issues. In my project context, it involves gathering data from CSO website about ‘Expenditure of Foreign Resident Overnight Visitors’ in Ireland and understanding it to make predictions for tourist behaviour in future and making policies accordingly.

Data Preparation: In this phase, the tourism data is cleaned, transformed, and structured in preparation for analysis. This includes identifying and visualising the trends like correlation of expenditure with expenditure type and residency region of the tourists.

Modeling: In this phase, Machine learning models are built, trained, and tuned to make predictions or classifications (e.g., predicting future expenditure by foreign residents based on historical trends).

Evaluation: The model's performance is evaluated based on criteria set in the Business Understanding phase.e.g. How increase in tourism revenue can be ensured by evaluating expenditure in particular months e.g. high expenditure during summers and then developing policies to attract more tourists in those months.

Deployment: The final model is deployed in a real-world scenario, where it can be monitored and refined e.g. the model can be used by tourism department to increase tourism revenue.

#### KDD (Knowledge Discovery in Databases):

KDD is an older framework that focuses on the process of discovering useful knowledge from large datasets. It consists of the following steps:

1. Developing an understanding of the application domain
2. Selecting and creating a data set on which discovery will be performed
3. Pre-processing and cleansing. In this stage, data reliability is enhanced
4. Data transformation
5. Choosing the appropriate Data Mining task
6. Choosing the Data Mining algorithm
7. Employing the Data Mining algorithm
8. Evaluation
9. Using the discovered knowledge

Justification for not using KDD: While KDD focuses on knowledge discovery, its process is very similar to CRISP-DM. However, KDD places a heavier emphasis on discovering knowledge in unstructured data (such as text, images, or sensor data), which might be less relevant in structured data projects like mine.

#### SEMMA (Sample, Explore, Modify, Model, Assess):

SEMMA is a data mining framework specifically designed by SAS for statistical modeling and analysis. A graphical user interface (GUI) provides a user-friendly front end to the SEMMA data mining process. Its steps are:

1. Sample: Sample the data by creating one or more data tables. The samples should be large enough to contain the significant information, yet small enough to process.
2. Explore: Explore the data by searching for anticipated relationships, unanticipated trends, and anomalies in order to gain understanding and ideas.
3. Modify:Modify the data by creating, selecting, and transforming the variables to focus the model selection process.
4. Model: Model the data by using the analytical tools to search for a combination of the data that reliably predicts a desired outcome.
5. Assess: Assess the data by evaluating the usefulness and reliability of the findings from the data mining process.

Justification for not using SEMMA:While SEMMA focuses on statistical modeling, which is great for supervised learning, it lacks the emphasis on understanding the business or broader context of the project (like CRISP-DM). SEMMA also does not emphasize iteration and feedback loops as much as CRISP-DM does, which is crucial in real-world projects where data and business requirements can evolve over time.

Technique Selection:

For my dataset on Expenditure of Foreign Resident Overnight Visitors, the appropriate machine learning technique depends on my Research Questions defined earlier. I have used Supervised learning approaches like Decision Trees and Random Forest im my project.

E.g. Question: Based on historical data, can we predict the future expenditure of foreign visitors?  
Relevant Techniques: Regression models (e.g., linear regression, Random Forest regression).

Below is an explanation of which type of learning technique (supervised, unsupervised, or semi-supervised) fits best for my dataset and why:

#### 1.Supervised Learning:

Supervised learning involves training a model on labeled data, where the target variable is known. This is appropriate for regression or classification tasks where you have specific predictions to make

E.g. Question: Based on historical data, can we predict the future expenditure of foreign visitors?  
Relevant Techniques: Regression models (e.g., linear regression, Random Forest regression).

Why Supervised Learning is Appropriate for my dataset:

Prediction Task: Since my dataset includes target variables like Expenditure, Mean Expenditure, and Percentage of Expenditure, it is appropriate to use supervised learning to predict future expenditures, identify trends, or forecast other related variables based on historical data.

Regression Model: For continuous values like Expenditure, regression models such as Linear Regression, Random Forest Regressor, can be used to predict future expenditures.

Classification Task: For grouping or classifying data into predefined categories (e.g., types of tourists), classification algorithms like Logistic Regression, K-Nearest Neighbors (KNN), etc. can be applied.

Example Use Case: Predicting the Expenditure of foreign tourists based on their Residency and Expenditure Type can be framed as a supervised learning problem where I can train a model to learn the relationships between input features (e.g., Residency, Expenditure Type) and the target variable (Expenditure).

#### 2.Unsupervised Learning:

Unsupervised learning is used when there is no labeled data and the goal is to uncover hidden patterns or groupings within the data. It is ideal for tasks like clustering and dimensionality reduction.

Why Unsupervised Learning Could Be Used:

Segmentation/Clustering: Unsupervised learning techniques like K-Means Clustering can be applied to segment the tourists into groups based on their expenditure patterns, which would be useful for market segmentation or targeted marketing.

Example Use Case: Using K-Means Clustering to segment tourists based on their Expenditure Type and Residency, helping to identify which regions tend to spend the most in different categories (Accommodation, Fare, etc.).

#### 3.Semi-Supervised Learning:

Semi-supervised learning is a hybrid approach where a small amount of labeled data is combined with a large amount of unlabeled data. It’s useful when labeling the entire dataset is costly or time-consuming.

Why Semi-Supervised Learning is Less Suitable Here:  
In my case, since I already have a dataset with labeled data (Expenditure, Expenditure Type, Residency, etc.), supervised learning is more suitable for most tasks. Semi-supervised learning is typically more beneficial when labeling data is difficult, which is not the case here.

**Machine Learning Models:**

Supervised Learning Approaches: Decision Trees and Random Forest

In my project, I have used Decision Trees and Random Forest for modelling. This is because I want to **predict** expenditure (a continuous variable), making it a **regression problem**. Based on my dataset and goals, I used:

#### 1. Decision Tree Regressor

Reason: Easy to interpret, handles categorical variables well, doesn’t require feature scaling, captures non-linear relationships.  
Use Case for my dataset: Understanding how Residency, Expenditure Type, and Month affect Expenditure.

#### 2. Random Forest Regressor

Reason: An ensemble of Decision Trees, it improves accuracy by reducing overfitting and variance.  
Use Case for my dataset: More robust predictions; can also give the feature importance.

Justification: These are both relevant because my data is mixed (categorical + numerical). Since I am not working with a very huge dataset, so, both are efficient. Moreover, my objective is to gain insight as well as accuracy, and Decision Trees give interpretability while Random Forest gives performance.

*Steps for Modelling:*

1. Define X and y after cleaning and preparing phase

2. Train/Test split the data

3. Model Building & Hyperparameter Tuning:

Define hyperparameter grids:

param\_grids = {

"Decision Tree": {

'max\_depth': [4, 8, 10, None],

'min\_samples\_split': [2, 5, 10]

},

"Random Forest": {

'n\_estimators': [100, 200],

'max\_depth': [4, 8, 10, None],

'max\_features': ['sqrt', 'log2']

}

}

Chosen parameters:

* max\_depth controls the depth of each tree. Using None means nodes are expanded until all leaves are pure.
* min\_samples\_split controls the minimum number of samples required to split an internal node. Higher values prevent overfitting.
* n\_estimators is the number of trees in the forest (for Random Forest only).
* max\_features controls the number of features to consider when looking for the best split:  
  + 'sqrt' is recommended for classification
  + 'log2' is another common choice

Next we initialize the models. After this, hyperparameter tuning is done using GridSearchCV. It is a model tuning tool from sklearn.model\_selection that automates the process of trying different combinations of hyperparameters to find the best-performing one based on a chosen scoring metric.

Explanation:

### 1. estimator=model

* This is the machine learning model we are tuning e.g., a Decision Tree or Random Forest. The model here is selected from the models dictionary.

### 2. param\_grid=param\_grids[name]

* This is the dictionary of hyperparameters we are testing for that model. param\_grids contains possible values for model parameters like max\_depth, min\_samples\_split, etc. It will test all combinations of these values.

### 3. cv=5

* This means we are using 5-fold cross-validation. This implies that the training data is split into 5 parts. The model trains on 4 parts and validates on the 5th..This rotates 5 times so that every part is used for validation once.

The reason for doing cross-validation is that it reduces overfitting and gives a better estimate of how the model performs on unseen data.

### 4. n\_jobs=-1

* Tells the computer to use all available CPU cores to speed up the grid search. -1 means “use everything you’ve got”. This helps to save time, especially when testing many hyperparameter combinations.

5. verbose=1

* Controls how much output is shown during the search. 1 means we will see progress updates printed to the console (like “Fitting 5 folds for each of 12 candidates…”).

### 6. scoring='r2'

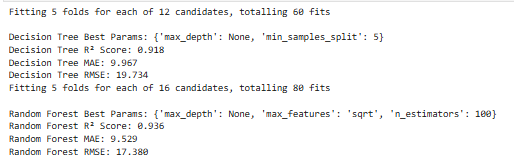
* This tells GridSearchCV to evaluate each model based on the R² Score (coefficient of determination). It will choose the combination of parameters that gives the highest R² on average across the 5 folds.  
  The reason behind using this score is that we are doing regression (predicting a numerical value like expenditure), and R² tells you how well the model explains the variance in the data.

After using Grid Search with Cross-Validation, we train the Best Model. Then use this model to predict tourism expenditure (or another target value) based on the testing features.

We evaluates the predictions using:

* R² Score: How well the model explains the variance.
* MAE (Mean Absolute Error): Average magnitude of errors.
* RMSE (Root Mean Squared Error): Penalizes large errors more.

The results are as follows:



Next step will be the comparison of models.

**Comparison:**

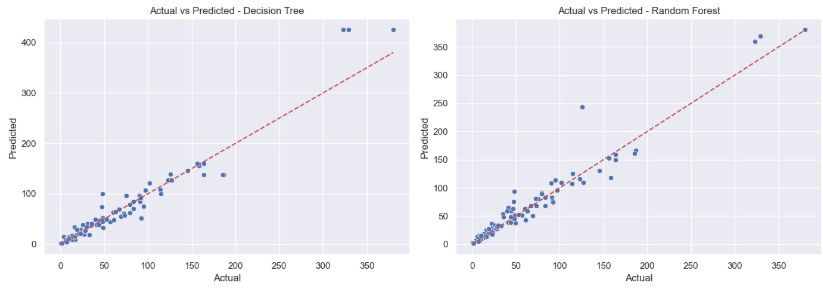
The following table summarizes the results of both models after hyperparameter tuning:

****

#### Key Insights:

* Random Forest outperforms Decision Tree in terms of R² Score (0.936 vs. 0.918), which means the model explains a higher proportion of the variance in the target variable.
* The Mean Absolute Error (MAE) and RMSE for Random Forest are lower than for Decision Tree, suggesting more accurate predictions.
* Random Forest also benefits from ensemble learning, which reduces overfitting and leads to more stable performance.

The following graphs also show the comparison for both the models based on actual and predicted plot.:



Graphical analysis: Since dots are close to the red line so both models are performing well (predictions are close to actual values) initially.

Lastly, we will see the similarities and differences.

**Similarities and differences:**

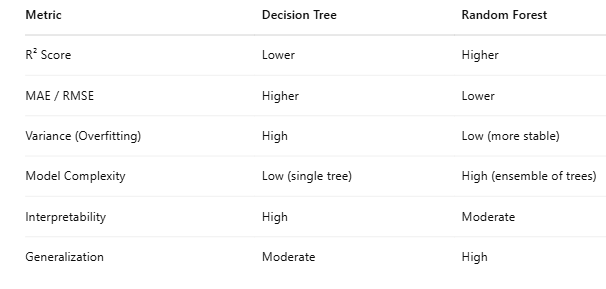
#### Similarity Between the Models:

Both Decision Tree and Random Forest are tree-based models:

* They can handle categorical and numerical data effectively.
* Both are capable of capturing nonlinear patterns in the data.
* Neither requires feature scaling or normalization.
* Both models produce interpretable results such as feature importance.

During evaluation, both models achieved relatively high R² scores and low error metrics, indicating they are well-suited to the regression task.

#### Difference Between the Models:



Random Forest combines predictions from multiple decision trees, reducing the risk of overfitting and resulting in better generalization.

Decision Tree, while simpler and more interpretable, tends to overfit the training data, leading to less accurate predictions on unseen data.

Overall, Random Forest achieved higher accuracy (e.g., R² score) and lower prediction error, indicating better reliability.

*Relevance:*

The Random Forest model demonstrates superior predictive accuracy compared to the Decision Tree, making it more suitable for forecasting tasks where generalization and stability are essential. This is particularly relevant in real-world applications like in my project for predicting tourism expenditure across various time periods or regions, where the ability to make accurate predictions on new, unseen data is crucial.

On the other hand, the Decision Tree model excels in interpretability, which is beneficial when the goal is to understand how specific variables such as residency or expenditure type influence the tourist spending. Its visual structure allows policymakers and stakeholders to easily grasp decision-making patterns. Additionally, Decision Trees are faster to train, making them practical when computational efficiency is prioritized.

Another reason is that my expenditure data is positively skewed and I did not apply log transformation over it. While log transformation is commonly used to reduce skewness for linear models (like linear regression), Decision Trees and Random Forests are not sensitive to skewed distributions in the same way.

*Effectiveness:*

Random Forest Regressor proves more effective due to its ensemble nature, which leads to better predictive accuracy and robustness.  
For a task like predicting tourism expenditure, where complex interactions and seasonal patterns may exist, Random Forest's ability to generalize well makes it the most suitable model for practical applications like Budget forecasting, Tourism planning and Policy decision support.  
Decision Tree Regressor, while less accurate, offers transparency and can help explain decisions e.g., identifying which months or types of expenditure are most influential and contribute more to Ireland’s tourism income. This makes it useful for exploratory analysis or stakeholder communication.

*Practical implications:*

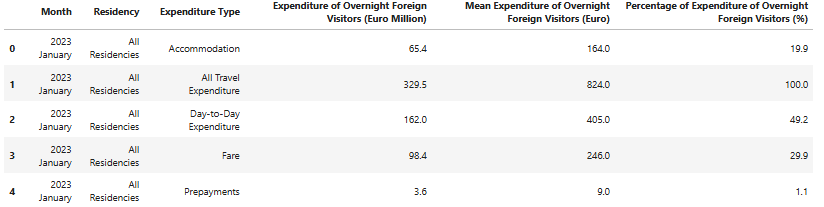
The improved performance of Random Forest suggests that ensemble methods should be preferred for forecasting financial metrics like expenditure, where accuracy and generalization are crucial. However, Decision Trees offer interpretability, which might be useful when explaining individual decisions or identifying key expenditure drivers.  
The findings can support data-driven tourism policy, such as targeting high-spending tourist segments or adjusting seasonal marketing strategies.

This section will focus on the Statistical part.

## Section3: Statistics for Data Analytics

Descriptive statistics and Plots:

The dataset looks like this in wide format:

****

For descriptive statistic we needs:

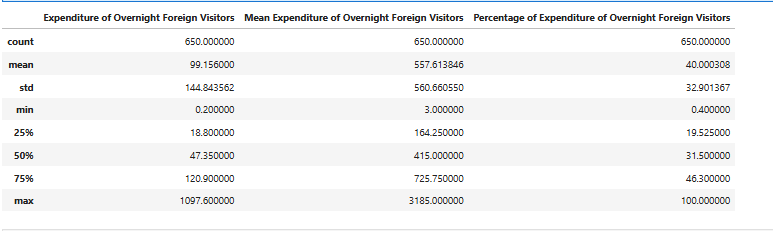
Central tendency: mean, median

Spread: standard deviation, range, IQR

Shape: skewness

In our code, in order to get the summary of statistics for numerical columns, we use:

df\_wide.describe()



From this we can deduce:

### Expenditure of Overnight Foreign Visitors

* Mean: €99.16
* Median (50%): €47.35
* Max: €1,097.60
* Min: €0.20
* Std Dev: €144.84

Insights: There’s a large gap between the mean and median, suggesting right skewness, a small number of entries have very high expenditures, pulling the average up. The wide range (from €0.20 to over €1,000) and high standard deviation indicate significant variability in how much visitors spend overnight. A few big spenders may heavily influence total expenditure calculations. This is important to flag for potential outlier treatment or log transformation in modeling.

### 2. Mean Expenditure of Overnight Foreign Visitors

* Mean: €557.61
* Median: €415.00
* Max: €3,185.00
* Min: €3.00
* Std Dev: €560.66

Insights:This column has extremely high values, with some very large means that may reflect longer stays or premium travel segments (e.g., US or Canada-based travelers). Again, mean > median, showing skewness. Also the max is nearly six times the 75th percentile, confirming the presence of extreme values/outliers.  
Here we can consider normalizing or log-transforming this column for feeding it into ML models.

### 3. Percentage of Expenditure of Overnight Foreign Visitors

* Mean: 40.00%
* Median: 31.50%
* Max: 100%
* Min: 0.4%
* Std Dev: 32.90

Insights: This is clearly bounded between 0 and 100, but again shows strong variability and skewness. A median of 31.5% compared to a mean of 40% tells us that a few high-percentage entries dominate. It is possible that some Expenditure Types (e.g., 'All Travel') account for most of the total.

Interquartile Range, Skewness and Kurtosis:

From code IQR:

Q1 = df\_wide.quantile(0.25, numeric\_only=True)

Q3 = df\_wide.quantile(0.75, numeric\_only=True)

IQR = Q3 - Q1

print(IQR)

Expenditure of Overnight Foreign Visitors (Euro Million) 102.100

Mean Expenditure of Overnight Foreign Visitors (Euro) 561.500

Percentage of Expenditure of Overnight Foreign Visitors (%) 26.775

Insights: Interquartile Range (IQR) values indicate substantial variability across expenditure metrics. The IQR for *Expenditure of Overnight Foreign Visitors* is €102.1 million, suggesting significant differences in total spending across different months, regions, or categories. Similarly, the *Mean Expenditure per Visitor* shows an IQR of €561.5, highlighting a wide range in individual spending behaviors. The *Percentage of Expenditure* also varies considerably, with an IQR of 26.78%, reflecting differences in the relative contribution of each expenditure type. These spreads point to diverse spending patterns among tourists, which are important to consider when segmenting tourists or forecasting future expenditure.

Skewness:

print(df\_wide.skew(numeric\_only=True))

Expenditure of Overnight Foreign Visitors (Euro Million) 3.368838

Mean Expenditure of Overnight Foreign Visitors (Euro) 1.752423

Percentage of Expenditure of Overnight Foreign Visitors (%) 0.893165

Insights: The skewness values reveal the asymmetry in the distribution of expenditure data. *Expenditure of Overnight Foreign Visitors* shows a high positive skew of 3.37, indicating that while most values are relatively low, there are a few very high expenditure records pulling the distribution to the right. *Mean Expenditure per Visitor* also has a positive skew of 1.75, suggesting that individual spending is generally concentrated at the lower end with some outliers spending significantly more. The *Percentage of Expenditure* shows a mild skew of 0.89, implying a more balanced distribution, though still slightly right-skewed. These findings highlight the presence of outliers and the need for robust methods or transformations when modeling.

Kurtosis:

print(df\_wide.skew(numeric\_only=True))

Expenditure of Overnight Foreign Visitors (Euro Million) 3.368838

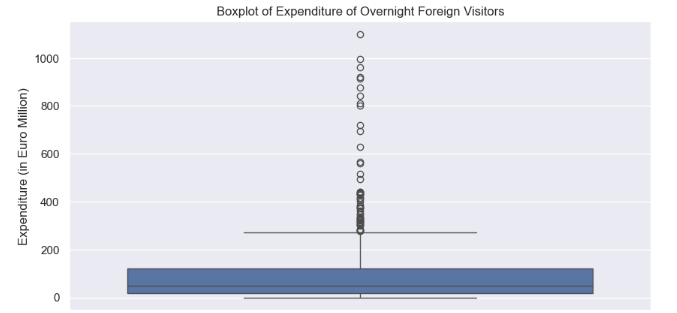
Mean Expenditure of Overnight Foreign Visitors (Euro) 1.752423

Percentage of Expenditure of Overnight Foreign Visitors (%) 0.893165

Insights:The Expenditure of Overnight Foreign Visitors (Euro Million) exhibits a very high kurtosis of 14.63, indicating a distribution with heavy tails and a sharp peak. This suggests the presence of extreme outliers and highly concentrated expenditure values around the mean. The Mean Expenditure (Euro) also has leptokurtic behavior with a kurtosis of 3.09, implying a distribution more peaked than normal. Conversely, the Percentage of Expenditure shows a slightly platykurtic distribution with a kurtosis of -0.43, meaning it is flatter than a normal distribution and has fewer outliers.

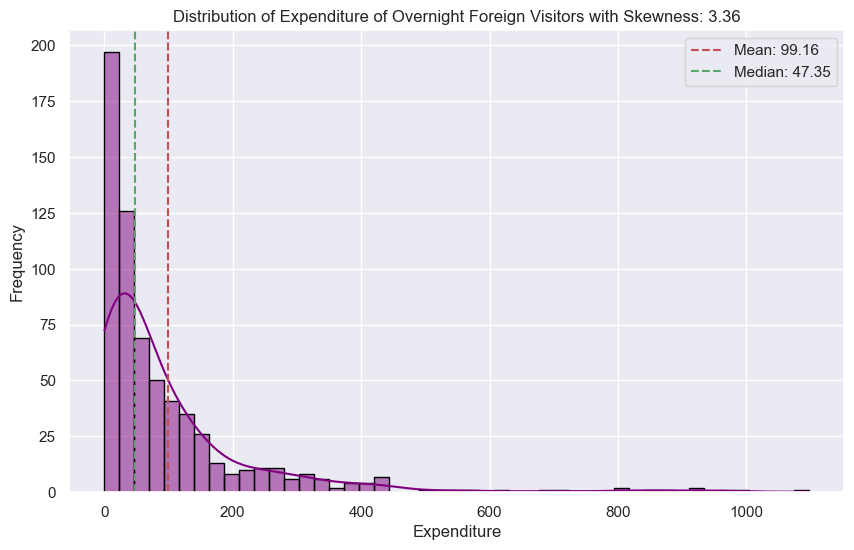
*Plots:*

1. Boxplot: To examine outliers in Expenditure of Overnight Foreign Visitors.



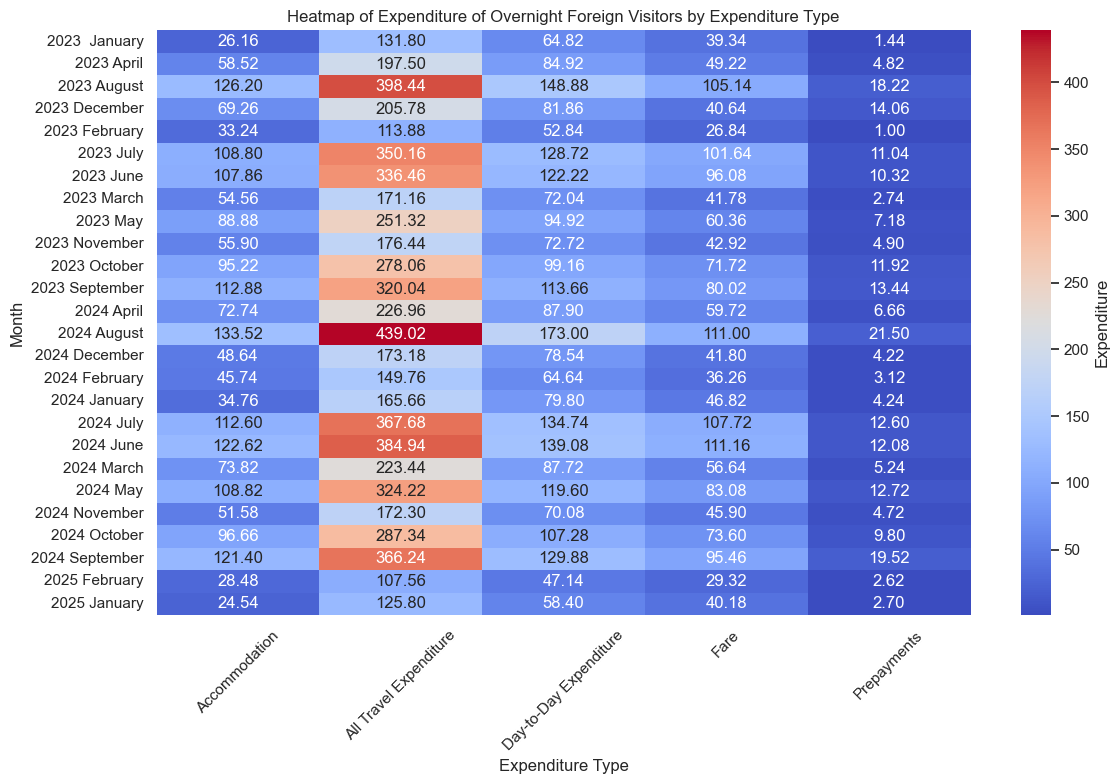
The boxplot reveals a few high-value outliers in the "Expenditure of Overnight Foreign Visitors" column. These could reflect seasonal surges or exceptional years (e.g., events, policy changes). However, since tree-based models are robust to outliers, we chose not to remove them.

1. Distribution plot to observe skewness:To see skewness in Expenditure of Overnight Foreign Visitors.



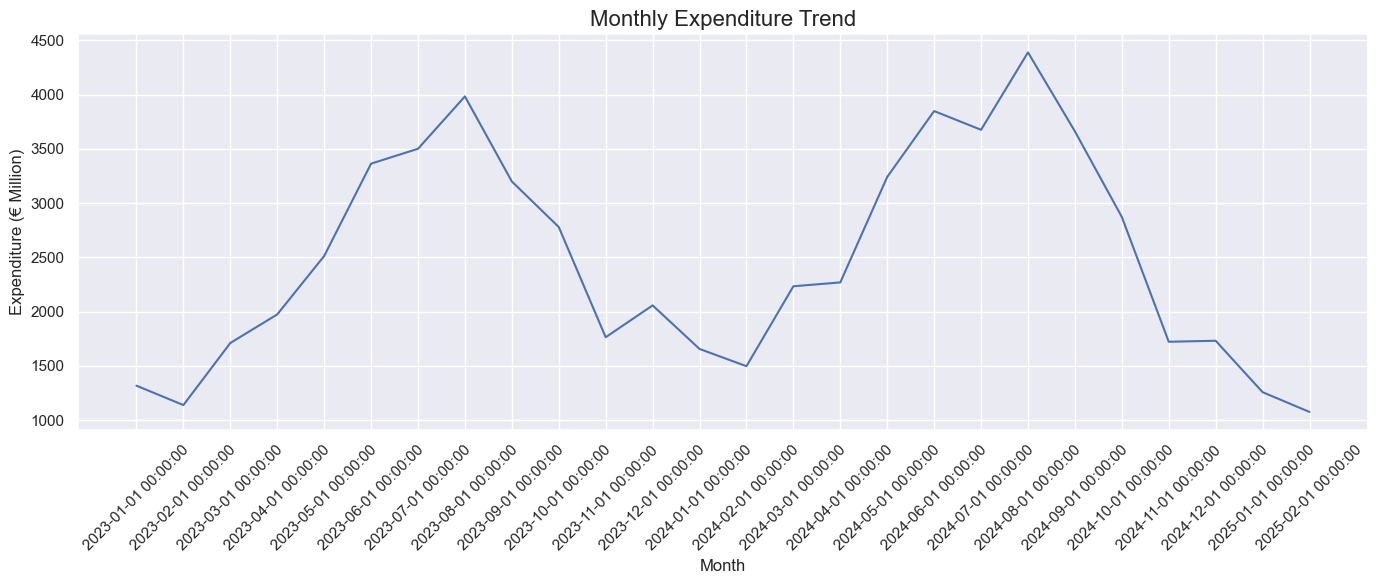
Skewness = 3.36: This suggests a right-skewed distribution, where the data is concentrated on the lower end, and the tail extends toward the higher expenditure values.

1. Heatmap: To observe correlation between Expenditure of Overnight Foreign Visitors by Expenditure Type.



The heatmap shows that there is a high correlation between expenditure and expenditure type of ‘all travel expenses’. The lowest correlation is between expenditure and prepayments. From this we can conclude that all travel expenses contribute to highest expenditure while prepayments contribute to lowest expenditure over different months.

1. Line plot:To observe monthly expenditure trend of Expenditure of Overnight Foreign Visitors.



The graph shows that the highest expenditure was in July 2023 and then in July 2024 with fluctuations throughout the two years. This can suggest that more visitors were coming in summers leading to high expenditures.

Discrete Distribution:

*Using Poisson Distribution:*

The Poisson distribution is used when:

* You are modeling the number of events in a fixed interval (time, space)
* The events occur independently, and the average rate (λ) is known.

Use case for my dataset:

Case 1: I can apply Poisson distribution to the Residency column as well since it has categorical values, and I can count how many times each Residency appears per month, thus applying discrete distribution.

Case 2: Count the number of months with very high expenditure spikes (e.g., > threshold), assuming these are rare events. I will have to use a threshold since expenditures is a numerical column with continuous values and Poisson is a discrete distribution which cannot be applied on continuous values directly.

Case 1: For a particular Residency:

Step 1: Filter the Dataset for Relevant Residency

Step 2: Count GB Expenditure Records per Month

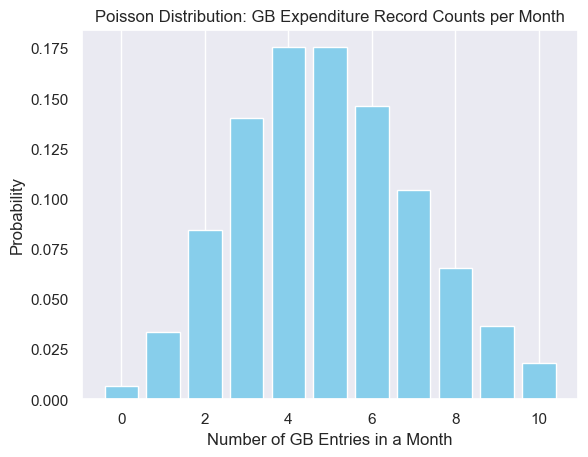
Step 3: Calculate λ (average number of events per month)

Step 4: Calculate Poisson Probabilities

Example: What’s the probability of getting exactly k events (e.g., 5 entries in a month) for Great Britain?

Step 5: Visualize the Poisson Distribution

Poisson Distribution: GB Expenditure Record Counts per Month:

****

Comparison with small and large samples:

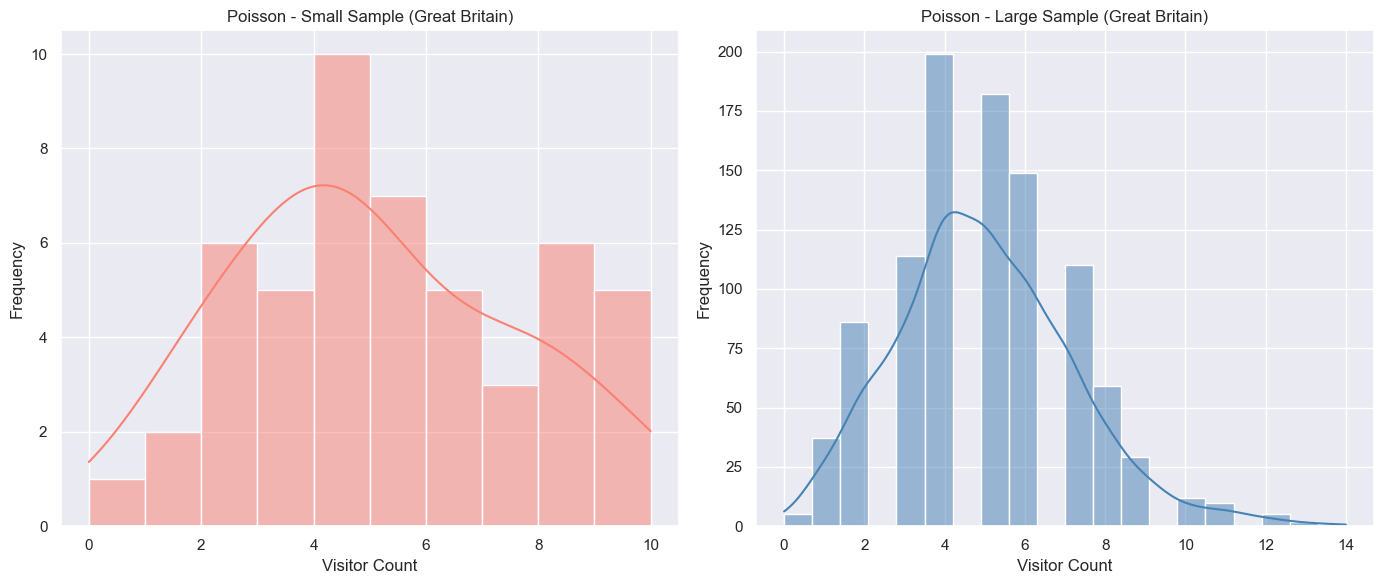
The code in the comparison section shows plots for simulated Poisson distributions to model and analyze the monthly frequency of expenditure records specifically for the residency "Great Britain (England, Scotland & Wales)" in the dataset.

It first calculates the average number of expenditure entries (or counts) for Great Britain across all months, which serves as the Poisson distribution's rate parameter (λ). Using this λ, it generates two sets of simulated data—one with a small sample size (50) and one with a large sample size (1000)—to illustrate how the distribution behaves at different sample sizes.

small\_sample = np.random.poisson(lam=lambda\_gb, size=50)

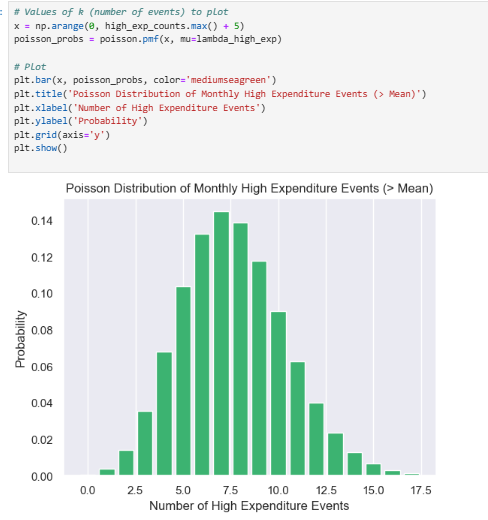
large\_sample = np.random.poisson(lam=lambda\_gb, size=1000)

As we increase the number of months or data points, the Poisson distribution will begin to approximate a normal distribution (Central Limit Theorem). This happens because as λ (the rate of occurrence) grows, the distribution becomes smoother and less skewed. This can be visualized from the plots below:



Case 2: For expenditure:

From code, after applying Poisson Distribution on Expenditure i.e. Expenditure of Overnight Foreign Visitors (Euro Million) and applying mean as the threshold value, we get below plot:



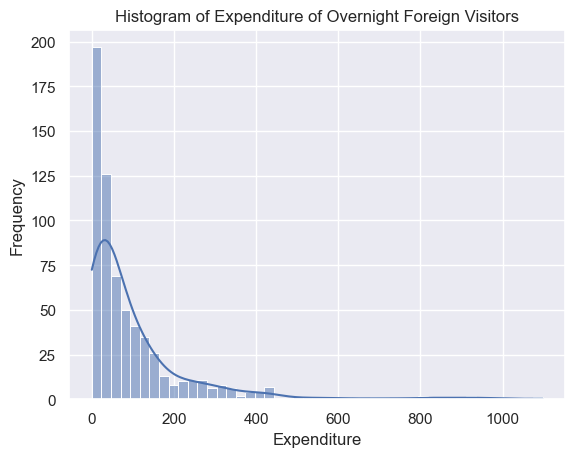
The plot reveals which counts of high expenditure events are most probable in a typical month. For example, the peak is at 7, which means that observing 7 high expenditure events per month is most common.

Rationale for using Poisson: The Poisson distribution is appropriate for modeling count data, especially when we're interested in the number of times an event occurs within a fixed interval (e.g., monthly tourism entries from a specific residency).

Normal distribution:

Plotting histogram for a column (e.g., Expenditure of Overnight Foreign Visitors) to see if it is following normal distribution;

sns.histplot(df\_wide['Expenditure of Overnight Foreign Visitors (Euro Million)'], kde=True)

****

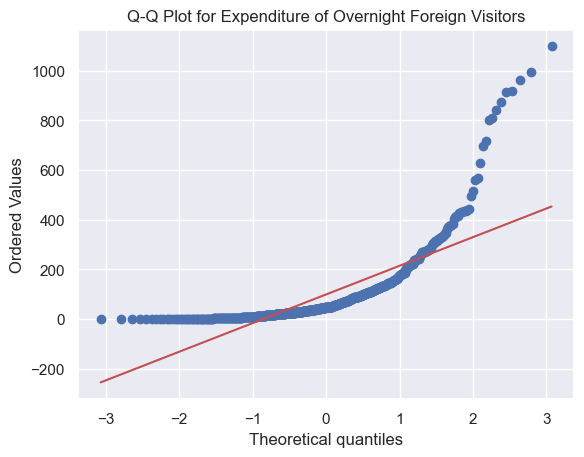
The histogram of Expenditure of Overnight Foreign Visitors (Euro Million) is showing a right-skewed distribution which indicates that most expenditure values are clustered at the lower end, while a few observations have exceptionally high expenditures.

This suggests that the data is not normally distributed, as a normal distribution would appear symmetric. Instead, the long tail on the right reflects occasional spikes in tourism expenditure, which could be due to seasonal peaks, major events, or sudden increases in high-spending tourists. This skewness has important implications for analysis—such as the need for transformations or using non-parametric models that are less sensitive to non-normality.

QQ-Plot:

A QQ (Quantile-Quantile) plot is a graphical tool used to assess if a dataset follows a particular theoretical distribution, such as the normal distribution. The plot compares the quantiles of the observed data to the quantiles of the theoretical distribution. If the points on the QQ plot fall approximately along a straight line, this suggests that the data follows the assumed distribution. If the points deviate from the straight line, it indicates that the data does not follow the distribution.

The QQ-plot shown below also shows that data is not normally distributed.



Importance:

In this analysis, I used Poisson and Normal distribution to better understand the characteristics of the Expenditure of Foreign Resident Overnight Visitors dataset. The choice of these distributions and the variables used were made based on the nature of the data, and their relevance to answering the research questions regarding expenditure patterns, tourist behavior, and statistical modeling.

### **Importance of the Distributions Used:**

Poisson Distribution: The Poisson distribution is used for modeling count data, particularly the number of events that happen in a fixed interval of time or space. It is ideal for modeling the count of visitors in a particular category (e.g., visitors from Great Britain or USA & Canada) or the number of times a particular Expenditure Type occurs in the dataset (e.g., occurrences of Accommodation expenditure types). In real-life scenarios, this can be used to model the number of visitors from different regions or the frequency of specific expenditure types.

Why Important: This distribution is helpful in modeling specific characteristics of the dataset if we want to understand how often certain events occur. It also provides insight into the probabilities of different categories or the frequency of occurrences, which can be critical for planning, policy-making, and forecasting in the tourism and expenditure domain.

Normal Distribution: The Normal distribution is the most widely used continuous probability distribution in statistics. It is important because many statistical models and machine learning algorithms assume that the data follows a normal distribution, particularly for the residuals (errors) in regression models, which are crucial for making inferences about the data.

Why Important:  
Normality Check: The Normal distribution helps to check whether the Expenditure values (e.g., Expenditure of Overnight Foreign Visitors) are symmetrically distributed around the mean, which is a key assumption for many statistical techniques (such as Linear Regression).

Modeling Expenditure: By checking if the expenditure data follows a normal distribution, we can better determine which modeling techniques to apply. For example, if the expenditure data is normally distributed, techniques like Linear Regression are suitable. On the other hand, if the data is skewed, transformations or non-parametric models may be necessary.

*Variable choice Justification:*

The variables I selected for the analysis are: Expenditure of Overnight Foreign Visitors, Residency, and Expenditure Type. They were chosen based on their importance to the core objectives of the analysis, which is to understand expenditure patterns and model visitor behavior in the context of tourism.

1. Expenditure of Overnight Foreign Visitors: This is the primary target variable in the dataset. It is a continuous variable that represents the amount of money spent by foreign visitors during their stay in a given month. The analysis focuses on modeling and predicting this expenditure, and it is important to determine whether it follows a normal distribution to apply suitable models like regression.
2. Residency: The Residency variable represents different regions (e.g., Great Britain, USA & Canada, etc.) and is an important categorical feature. The distribution of visitors by residency influences the expenditure patterns, as different regions may have different spending behaviors. For modeling purposes, Residency can be encoded for machine learning models (e.g., One-Hot Encoding).
3. Expenditure Type: The Expenditure Type variable indicates different categories of spending (e.g., Accommodation, All Travel Expenditure). This is crucial for understanding how different types of spending affect overall expenditure patterns and how various visitors allocate their budgets. Expenditure Type is a categorical feature, and in the analysis, we can explore whether it correlates with the Expenditure of Overnight Foreign Visitors or if any patterns emerge that could help us classify or predict spending. For modeling purposes, it can also be encoded for machine learning models (e.g., One-Hot Encoding).

### *Could the Variables in Discrete Distribution be Used as Normal Distribution?*

It is important to assess whether the discrete variables used for Poisson distribution can be modeled using a Normal distribution.

1. Residency (Categorical Variable): Normal Distribution is not applicable to Residency because it is a categorical variable with different regions as distinct groups. The distribution of categorical variables like Residency does not conform to the assumptions of a normal distribution (which requires continuous data).
2. Expenditure Type (Categorical Variable): Expenditure Type is also a categorical variable. While we can examine the distribution of categories, it is not appropriate to model it using a Normal distribution.
3. Expenditure (Continuous Variable): The Expenditure variable has continuous values so it could potentially follow a Normal distribution, but this depends on the skewness and distribution of the data as well.

Moreover, If the Expenditure variable is not normally distributed, we can apply logarithmic transformation to make it more Gaussian.

This section will focus on the programming part.

**Section 4: Programming for Data Analytics**

I will discuss the programming paradigms here.

Programming Paradigms:

A paradigm is a way of viewing the world and a programming paradigm is a way of viewing how to program.

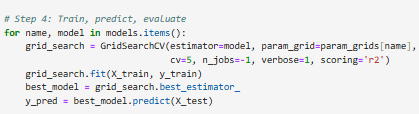
Programming paradigms are fundamental approaches to programming that dictate how developers conceptualize and structure their code. We can think of them as the guiding philosophies or blueprints for writing code because they offer a unique lens through which to view and solve problems. Following are some of the programming paradigms:

1. Imperative programming:

Imperative programming sees a program as a series of steps/instructions to be executed in order. It is all about giving explicit instructions on how to perform a task. It directly controls the flow of the program through statements, loops, and conditionals.

Relevance to my project:

* Used for control flow: loops, conditions, and stepwise logic construction e.g. in the application of ML models:



* Created custom logic for Poisson thresholding and filtering by categories.
* Useful in setting up experimental procedures or looping through months/residencies.

1. Procedural Programming:

This paradigm is based on sequences of instructions that are executed step-by-step to achieve a desired result. It is like Imperative, but makes use of “procedures” or “subroutines”. In Python, we use functions for this.

Relevance to my project:

* EDA and data cleaning steps were written in a linear, step-by-step fashion e.g.



* Used pandas functions in sequence to filter, clean, and reshape data (long to wide format)

1. Object Oriented Programming:

Object Oriented programming conceptualises a system as a collection of objects (built from classes) that communicate with each other

These can be physical, like people; non-physical, like a restaurant booking and can be intangible software things, like a DataFrame.

Overall, OOP organizes code into objects that encapsulate both data and the functions that operate on that data.

Relevance to my project: OOP principles were applied through the use of machine learning models and visual libraries that are structured as objects and methods.

* Libraries like sklearn, seaborn, and matplotlib are based on OOP e.g.



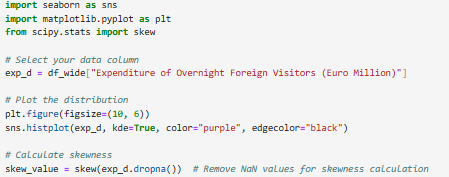
* Models such as DecisionTreeRegressor() and RandomForestRegressor() are instantiated as **objects**

1. Declarative programming:

Declarative programming specifies the goal of what you want the program to do, rather than the instructions to do. Thus, in this paradigm, you specify what you want to achieve (the result), without detailing how to achieve it.

Relevance to my project:

* Mostly used in statistical modeling e.g. for statistical distributions (like Poisson) and analytical techniques without manual iteration.
* Libraries like scipy.stats, and seaborn abstract complex statistical logic e.g.



* Declarative code was useful in statistical analysis for model assumptions, distributions, and probability-based analysis.

1. Functional Programming:

This paradigm focuses on writing **pure functions** that take inputs and produce outputs without side effects Functional programming considers functions to be “first-class citizens”. As much as possible is done via functions:

* Functions can be assigned to variables
* Functions can be function inputs
* Function can be function outputs

Relevance to my project: Not used much

## *Impact of paradigms on Project Design:*

* Procedural programming ensured clean data flows for reproducibility.
* Object Oriented Programming from libraries enabled encapsulated model development and tuning.
* Functional techniques could be used to support concise transformations and feature engineering.
* Declarative programming made complex mathematical modeling simple and expressive.
* Imperative programming helped organize looping experiments and simulation setups.

*Conclusion:*

In my project, various programming paradigms significantly influenced both my design decisions and problem-solving approaches. Object-Oriented Programming (OOP) played a crucial role in structuring the machine learning models I used, such as Random Forest and Decision Trees, from the sklearn library. The object-oriented nature of these models allowed me to easily train, predict, and evaluate the models by treating them as reusable objects with defined methods.

For data manipulation and exploration, Procedural Programming proved essential, particularly with pandas, where I performed step-by-step data cleaning, transformation, and aggregation for Exploratory Data Analysis (EDA). This allowed me to directly control the flow of operations and handle large datasets efficiently.

Finally, Declarative Programming influenced my use of stats-models for statistical analysis, where I could define the statistical relationships between variables without specifying every detail of the underlying computation.

The combination of these paradigms enabled a flexible, modular, and efficient approach to solving the complex challenges of analyzing and modeling my dataset.

## References

[1] Central Statistics Office (2025) Central Statistics Office. Available at: https://data.cso.ie/ (Accessed 6 April 2025).

Central Statistics Office (2025). ITM06 - Expenditure of Foreign Resident Overnight Visitors [Dataset].

https://datasalsa.com/dataset/?catalogue=data.gov.ie&name=itm06-expenditure-of-foreign-resident-overnight-visitors

[2]Iqbal, M. (2025) “Project Management Methodologies” [Lecture], M4: Machine Learning for Data Analytics. CCT College Dublin. 12 March.

[3]Iqbal, M. (2025) “Classification using Decision Trees and RF” [Lecture], M4: Machine Learning for Data Analytics. CCT College Dublin. 26 February.

[4]Ali, M.M., et al. (2021). ‘Heart disease prediction using supervised machine learning algorithms: Performance analysis and comparison’, Computers in Biology and Medicine, 136, p.104672. <https://doi.org/10.1016/j.compbiomed.2021.104672>.

[5] Ahmed, M. (2025) “Uniform, Normal, Binomial, Poison, and Geometric distributions” [Lecture], Statistics for Data Analytics. CCT College Dublin. 13 March.

[6]McQuaid, D. (2025) “What is Exploratory Data Analysis?” [Lecture], M3: Data Preparation and Visualisation. CCT College Dublin. 26 February.

[7] McQuaid, D. (2025) “Principles of Data Visualization” [Lecture], M3: Data Preparation and Visualisation. CCT College Dublin. 12 March.

[8]McQuaid, D. (2025) “Encoding Categorical Data ” [Lecture], M3: Data Preparation and Visualisation. CCT College Dublin. 26 March.

[9] DataCamp, Inc. (2024) Datacamp. Available at: https://www.datacamp.com/blog/introduction-to-programming-paradigms (Accessed 6 April 2025).

[10] Weiss, S. (2025) “Programming Paradigms for Data Analytics”[Lecture], M1: Programming for Data Analytics. CCT College Dublin. 13 March.