EDA-Seasonality Detection

Seasonality Detection in Exploratory Data Analysis (EDA)

Seasonality detection is an essential step in exploratory data analysis to identify recurring patterns or cycles in time series data. It helps you understand whether your data exhibits regular fluctuations over a specific period, such as daily, weekly, monthly, quarterly, or yearly.

Why is Seasonality Detection important?

1. **Identify trends**: Seasonal patterns can indicate underlying trends that need to be considered when building models or making predictions.
2. **Improve forecasting**: By accounting for seasonality, you can create more accurate forecasts and better understand the relationships between variables.
3. **Data preparation**: Seasonality detection informs data preprocessing steps, such as detrending or deseasonalization.

Methods for Seasonality Detection:

1. **Visual inspection**: Plotting time series data to identify visual patterns.
2. **Autocorrelation function (ACF)**: Measures the correlation between a time series and lagged versions of itself.
3. **Partial autocorrelation function (PACF)**: Identifies the relationship between a time series and its lagged values, controlling for intermediate lags.
4. **Seasonal decomposition**: Separates time series data into trend, seasonal, and residual components.

Example using Python with Statsmodels

```
 import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

# Generate sample time series data with seasonality
np.random.seed(0)
n = 100
t = np.arange(n)
y = np.sin(2 * np.pi * t / 12) + np.sin(2 * np.pi * t / 6) + 10 + np.random.randn(n)

# Create a Pandas DataFrame with the time series data
df = pd.DataFrame(y, index=t, columns=['values'])

# Plot the original time series
plt.figure(figsize=(10, 6))
plt.plot(df.index, df['values'])
plt.title('Original Time Series')
plt.show()

# Perform seasonal decomposition using Statsmodels
decomposition = seasonal_decompose(df['values'], model='additive')

trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid

# Plot the decomposed components
fig, axes = plt.subplots(4, 1, figsize=(10, 12))
axes[0].plot(trend)
axes[0].set_title('Trend')
axes[1].plot(seasonal)
axes[1].set_title('Seasonality')
axes[2].plot(residual)
axes[2].set_title('Residuals')
plt.show()
```

In this example, we generate a sample time series with seasonality (daily and weekly cycles) and apply seasonal decomposition using Statsmodels. The resulting plots reveal the trend, seasonality, and residuals of the original data.

Key Takeaways:

1. **Visual inspection** is crucial for identifying seasonality.
2. **Seasonal decomposition** helps separate the time series into its underlying components (trend, seasonality, and residuals).
3. **Statsmodels** provides a convenient way to perform seasonal decomposition in Python.