VIVA Questions and Answer

1. What libraries are commonly used for time series analysis in Python?

The commonly used libraries are:

1. **Pandas**: For data manipulation and handling time-indexed data.
2. **NumPy**: For numerical operations.
3. **Matplotlib** and **Seaborn**: For visualization.
4. **Statsmodels**: For statistical time series models (e.g., ARIMA, SARIMA).
5. **Scikit-learn**: For machine learning models on time series data.
6. **Prophet**: For time series forecasting.

2. How do you create a time series in Python using pandas?

You can create a time series by setting the DatetimeIndex in a DataFrame.

```
import pandas as pd

data = {'Date': ['2023-01-01', '2023-01-02', '2023-01-03'], 'Value': [100, 200, 300]}
df = pd.DataFrame(data)
df['Date'] = pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
print(df)
```

Output:

```
            Value
Date
2023-01-01    100
2023-01-02    200
2023-01-03    300
```

3. How do you resample a time series to a different frequency?

Use the .resample() method in pandas.

```
df_resampled = df.resample('W').sum()  # Weekly frequency
print(df_resampled)
```

Explanation:

- 'W' indicates weekly frequency.
- You can use other frequencies like 'D' (daily), 'M' (monthly), or 'Y' (yearly).

4. How can you detect missing values in a time series?

Use the .isnull() and .sum() methods to detect missing values.

```
print(df.isnull().sum())
```

Output Example:

```
Value    1
dtype: int64
```

5. How do you handle missing values in a time series?

You can handle missing values using:

1. **Forward fill**:
2. df_filled = df.fillna(method='ffill')
3. **Backward fill**:
4. df_filled = df.fillna(method='bfill')

5. **Interpolation**:
6. df_interpolated = df.interpolate()

6. How do you visualize a time series in Python?

You can use Matplotlib or Seaborn.

```
import matplotlib.pyplot as plt

df['Value'].plot(title='Time Series')
plt.xlabel('Date')
plt.ylabel('Value')
plt.show()
```

7. How do you decompose a time series into its components?

Use the seasonal_decompose function from statsmodels.

```
from statsmodels.tsa.seasonal import seasonal_decompose

result = seasonal_decompose(df['Value'], model='additive', period=365)
result.plot()
plt.show()
```

8. How do you check if a time series is stationary in Python?

Use the Augmented Dickey-Fuller (ADF) test from statsmodels.

```
from statsmodels.tsa.stattools import adfuller

result = adfuller(df['Value'])
print('ADF Statistic:', result[0])
print('p-value:', result[1])
```

Output:

If the p-value is less than 0.05, the time series is stationary.

9. How do you forecast a time series using ARIMA in Python?

Use the ARIMA model from statsmodels.

```
from statsmodels.tsa.arima.model import ARIMA

model = ARIMA(df['Value'], order=(1, 1, 1))
model_fit = model.fit()
forecast = model_fit.forecast(steps=5)
print(forecast)
```

10. How do you evaluate the performance of a time series model?

Use error metrics like MAE, MSE, or RMSE. Example:

```
from sklearn.metrics import mean_squared_error
import numpy as np

actual = [100, 200, 300]
predicted = [110, 190, 310]
mse = mean_squared_error(actual, predicted)
rmse = np.sqrt(mse)
```

print('RMSE:', rmse)
Output Example:
RMSE: 10.0