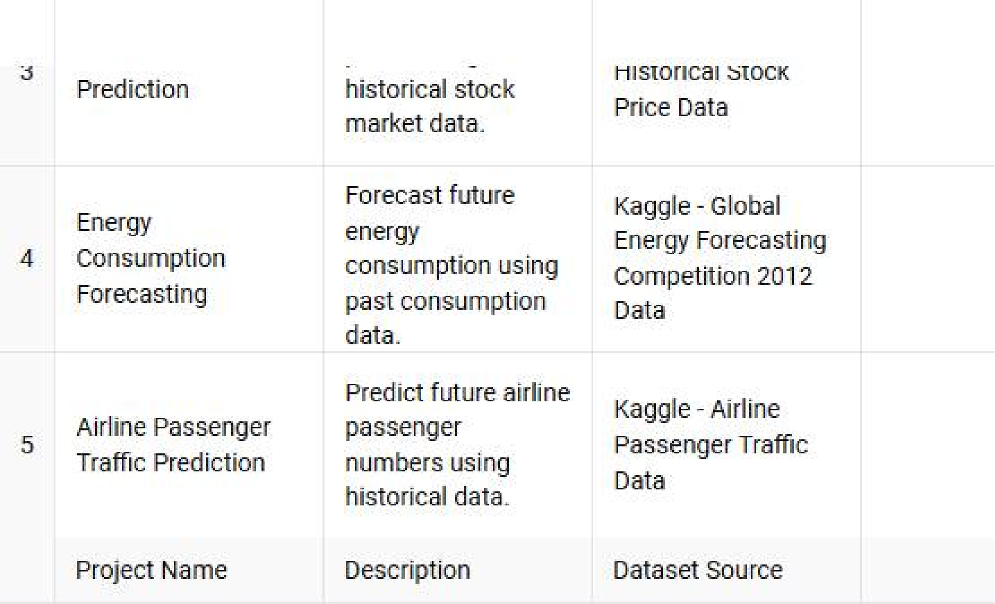
Time Series Forecasting Guide

Practice ML Projects for Time Series Forecasting



3	Prediction	historical stock market data.	Historical Stock Price Data
4	Energy Consumption Forecasting	Forecast future energy consumption using past consumption data.	Kaggle - Global Energy Forecasting Competition 2012 Data
5	Airline Passenger Traffic Prediction	Predict future airline passenger numbers using historical data.	Kaggle - Airline Passenger Traffic Data
	Project Name	Description	Dataset Source

Here are some practice machine learning projects for time series forecasting, each with a description and a free dataset source. Let's proceed with creating a detailed workflow for one of these projects, including instructions, a pipeline, and an answer key.

Project: Retail Sales Forecasting

Description

Predict future sales for a retail store using historical sales data. This project will involve exploring, preprocessing, and modeling the data to make accurate forecasts.

Dataset

Source: UCI Machine Learning Repository - Online Retail Dataset

Instructions and Steps

1. **Download and Load the Dataset**
    - Visit the UCI Machine Learning Repository.
    - Download the dataset and load it into a pandas DataFrame.
2. **Data Exploration and Preprocessing**
    - Explore the dataset to understand its structure and contents.
    - Handle missing values and outliers.
    - Convert the 'InvoiceDate' column to a datetime format.
    - Aggregate sales data on a daily or weekly basis.
3. **Feature Engineering**
    - Create new features such as 'day of the week', 'month', and 'year'.

   - Consider adding lag features for previous sales data points.
4. **Train-Test Split**
   - Split the data into training and testing sets.
5. **Model Selection and Training**
   - Choose a suitable model for time series forecasting, such as ARIMA, Prophet, or LSTM.
   - Train the model using the training data.
6. **Model Evaluation**
   - Evaluate the model's performance on the test data using appropriate metrics (e.g., RMSE, MAE).
7. **Forecast Future Sales**
   - Use the trained model to forecast future sales.
8. **Visualization**
   - Visualize the actual vs. predicted sales data.

Fully Functional Pipeline in Python

Here is a simplified version of a time series forecasting pipeline using the ARIMA model:

```python
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
import numpy as np

# Load the dataset
url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/00352/Online%20Retail.xlsx'
data = pd.read_excel(url)

# Data preprocessing
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'])
data['Sales'] = data['Quantity'] * data['UnitPrice']
daily_sales = data.resample('D', on='InvoiceDate').sum()['Sales']

# Split the data into training and testing sets
train_data, test_data = daily_sales[:int(len(daily_sales)*0.8)], daily_sales[int(len(daily_sales)*0.8):]

# Train the ARIMA model
model = ARIMA(train_data, order=(5, 1, 0))
model_fit = model.fit()

# Forecast
forecast = model_fit.forecast(steps=len(test_data))

# Evaluation
rmse = np.sqrt(mean_squared_error(test_data, forecast))
print(f'RMSE: {rmse:.2f}')

# Plotting
plt.figure(figsize=(12, 6))
plt.plot(train_data.index, train_data, label='Training Data')
plt.plot(test_data.index, test_data, label='Actual Sales')
plt.plot(test_data.index, forecast, label='Predicted Sales', color='red')
plt.legend()
plt.show()
```

Answer Key

1. **RMSE Calculation**
    - Root Mean Squared Error (RMSE) is calculated to evaluate the model's performance.
    - RMSE Formula: $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$
2. **Model Parameters**
    - ARIMA order `(p, d, q)`: For this example, we used `(5, 1, 0)`.
    - These parameters can be tuned based on the data characteristics.

This pipeline provides a basic framework for a time series forecasting project. You can expand it by trying different models and parameter tuning for improved performance. If you have any questions or need further assistance, feel free to ask!

Let's create a time series forecasting pipeline using the Prophet library for the same Retail Sales Forecasting project. Prophet is a powerful library developed by Facebook for forecasting time series data.

Fully Functional Pipeline in Python Using Prophet

Here's how you can implement a time series forecasting pipeline using Prophet:

```python
import pandas as pd
import matplotlib.pyplot as plt
from prophet import Prophet
from sklearn.metrics import mean_squared_error
import numpy as np

# Load the dataset
url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/00352/Online%20Retail.xlsx'
data = pd.read_excel(url)

# Data preprocessing
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'])
data['Sales'] = data['Quantity'] * data['UnitPrice']
daily_sales = data.resample('D', on='InvoiceDate').sum()['Sales'].reset_index()

# Prepare the data for Prophet
daily_sales.rename(columns={'InvoiceDate': 'ds', 'Sales': 'y'}, inplace=True)

# Split the data into training and testing sets
train_data = daily_sales[:int(len(daily_sales) * 0.8)]
test_data = daily_sales[int(len(daily_sales) * 0.8):]

# Initialize and train the Prophet model
model = Prophet()
model.fit(train_data)

# Forecast future sales
future = model.make_future_dataframe(periods=len(test_data))
forecast = model.predict(future)

# Evaluation
rmse = np.sqrt(mean_squared_error(test_data['y'], forecast['yhat'].tail(len(test_data))))
print(f'RMSE: {rmse:.2f}')

# Plotting
fig1 = model.plot(forecast)
plt.show()
```

```
    fig2 = model.plot_components(forecast)
    plt.show()
```

Instructions

1. **Data Loading and Preprocessing**
    - Load the dataset using `pandas` and convert the `InvoiceDate` column to datetime.
    - Calculate daily sales by resampling the data.
2. **Prepare Data for Prophet**
    - Rename columns to `ds` (for dates) and `y` (for sales) as required by Prophet.
3. **Model Training**
    - Initialize the Prophet model and fit it to the training data.
4. **Forecasting**
    - Create a future dataframe and use the model to predict future sales.
5. **Evaluation**
    - Calculate RMSE to evaluate model performance.
6. **Visualization**
    - Use Prophet's built-in plotting functions to visualize the forecast and its components.

Key Points

- **Prophet's Advantages**: It's designed to handle missing data, outliers, and holiday effects, making it suitable for business data.
- **Component Analysis**: Prophet provides a breakdown of trends, seasonality, and holidays, which helps in understanding the model's predictions.

Feel free to try this code and explore further by tuning the model's parameters or adding holiday effects if applicable! Let me know if you have any questions or need more details.

ChatGPT can make mistakes. Check important info.