from prophet import Prophet  
import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd  
  
# 读取数据  
df = pd.read\_excel(r"C:\Users\阿韩想养二哈\Desktop\a.xlsx")  
# 选择需要的数据  
#df = df[['date','Close']]  
  
  
# 注意：Prophet模型对于数据格式有要求，日期字段必须是datetime格式，这里通过pd.to\_datetime来进行转换。  
df['销售日期'] = pd.to\_datetime(df['销售日期'])  
# 这里需要将目标数据进行log处理，可以理解为归一化，注意最终绘图的时候需要将目标数据复原。  
df['销量(千克)'] = np.log(df['销量(千克)'])  
# 更改列名，更改为Prophet指定的列名ds和y  
df = df.rename(columns={'销售日期': 'ds', '销量(千克)': 'y'})  
# 划分数据，划分为训练集和验证集，将前十年的数据作为训练集，后一年的数据作为测试集。  
df\_train = df[:7307]  
df\_test = df[7307:]  
  
  
import plotly.express as px  
fig = px.line(df\_train, x="ds", y="y")  
fig.show()  
  
model = Prophet()  
model.fit(df\_train)  
# make\_future\_dataframe: 作用是告诉模型我们要预测多长时间，以及时间的周期是什么。这里设置为365，即预测一年时间的数据。  
#  
future = model.make\_future\_dataframe(periods=365, freq='D')  
# 进行预测，返回预测的结果forecast  
forecast = model.predict(future)  
# forecast['additive\_terms'] = forecast['weekly'] + forecast['yearly']；  
# 有：forecast['yhat'] = forecast['trend'] + forecast['additive\_terms'] 。  
# 因此：forecast['yhat'] = forecast['trend'] +forecast['weekly'] + forecast['yearly']。  
# 如果有节假日因素，那么就会有forecast['yhat'] = forecast['trend'] +forecast['weekly'] + forecast['yearly'] + forecast['holidays']。  
print(forecast)  
  
# 对数据机型可视化操作，黑点表示真实数据，蓝线表示预测结果。蓝色区域表示一定置信程度下的预测上限和下限。  
model.plot(forecast)  
plt.show()  
# 通过plot\_componets()可以实现对数据的年、月、周不同时间周期下趋势性的可视化。  
model.plot\_components(forecast)  
  
# 测试，把ds列，即data\_series列设置为索引列  
df\_test = df\_test.set\_index('ds')  
# 把预测到的数据取出ds列，预测值列yhat，同样把ds列设置为索引列。  
forecast = forecast[['ds','yhat']].set\_index('ds')  
# 把数据转换回去，因为之前经过了log函数  
df\_test['y'] = np.exp(df\_test['y'])  
forecast['yhat'] = np.exp(forecast['yhat'])  
  
# join:按照索引进行连接，  
# dropna：能够找到DataFrame类型数据的空值（缺失值），将空值所在的行/列删除后，将新的DataFrame作为返回值返回。  
df\_all = forecast.join(df\_test).dropna()  
df\_all.plot()  
# 设置左上角小标  
plt.legend(['true', 'yhat'])  
plt.show()