Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный технический университет имени
Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Компьютерные системы и сети (ИУ-6)»

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

по дисциплине «Поддержка принятия решений в системах мониторинга»

Студент:	Козлов Владимир Михайлович
Группа:	ИУ6-13М
Тип задания:	лабораторная работа
Тема:	Реконструкция модели цифрового двойника человека-оператора в киберфизической системе

Студент ________________ Козлов В.М.
подпись, дата Фамилия, И.О.

Преподаватель ________________ ________________
подпись, дата Фамилия, И.О.

Москва, 2024

Содержание

Цель

Изучение особенностей построения алгоритма реконструкции математической модели человека-оператора по временному ряду.

Задание (альтернативное)

При выполнении лабораторной работы студентам необходимо разработать цифровую модель эксперта, решающего задачу прогнозирования процесса (ситуации) по временному ряду. Нужно взять любой реальный временной ряд и создать систему прогнозирования исходного временного ряда на будущий период времени.

1 Выполнение работы

1.1 Исходные данные

Были выбраны данные о погоде в городе Базель с января по июнь 2020 года.

Рис. 1: Исходные данные

Для прогнозирования временных рядов была выбрата нейросеть SARIMA (Seasonal ARIMA). ARIMA является одной из самый популярных нейросетей для анализа временных рядов на Python, а сезонная природа температуры подталкивает к использованию сезонной версии.
Данные были взяты с сайта meteoblue.com. Так как данные были взяты с избытком, а также потому что сайт предоставляет данные по часам данные следует подготовить.

Листинг 1: Подготовка данных

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import statsmodels.api as sm
from statsmodels.tsa.statespace.sarimax import SARIMAX
from sklearn.metrics import mean_squared_error, mean_absolute_error

dataset = pd.read_csv('weather_data.csv', header=0, parse_dates=['datetime'])

dataset = dataset.set_index('datetime').resample("d").mean().asfreq('d')

df = dataset.iloc[:200,:]
```

Параметрами для модели являются параметры AR (autoregressive) он же p, I (integration) он же d, MA (moving average) он же q, а также аналогичные сезонные параметры P, D, Q и m - количество временных шагов в одном сезоне. Параметр d - сколько нужно продифференцировать ряд, чтобы он стал стационарен,а все остальные параметры примерно можно узнать коррелограммы. Получить параметры можно с помощью следующего кода.

Листинг 2: Получение параметров модели

```
test = sm.tsa.adfuller(df['temperature'])
if test[0]> test[4]['5%']:
    print('есть единичные корни, ряд не стационарен')
else:
    print('единичных корней нет, ряд стационарен')

df_diff = df['temperature'].diff(periods=1).dropna()
test = sm.tsa.adfuller(df_diff)
if test[0]> test[4]['5%']:
    print('есть единичные корни, ряд не стационарен')
else:
    print('единичных корней нет, ряд стационарен')

fig = plt.figure(figsize=(12,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_diff.dropna(), lags=99, ax=ax1)
plt.grid()
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_diff.dropna(), lags=99, ax=ax2)
plt.grid()
plt.savefig('latex/extra/acf_pacf.png')
plt.show()
```

Листинг 3: Вывод кода из листинга 2

```
есть единичные корни, ряд не стационарен
единичных корней нет, ряд стационарен
```

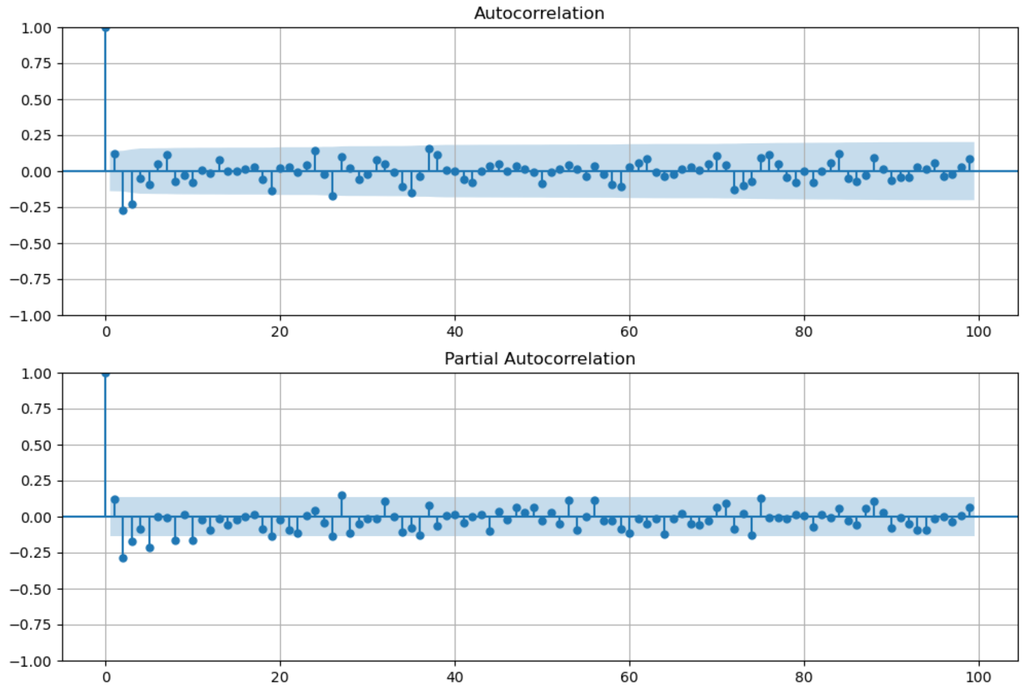

Рис. 2: коррелограммы

Как видно из текстового вывода d=1.
Из коррелограммы p=3, q=2, m=8. Т.к. первый существенный лаг на коррелограмме отрицательный P=0, Q=2. Параметры получены, следующий шаг - обучение модели.

Листинг 4: Обучение модели и прогноз на продолжительный срок

```
df.reset_index()

train = df.iloc[:-50,:]
test = df.iloc[-50:,:]

model = SARIMAX(train, order=(3, 1, 2), seasonal_order=(0, 0, 2, 8))
model_fit = model.fit()

fcast_len = len(test)
fcast = model_fit.forecast(fcast_len)

mse = mean_squared_error(test, fcast)
rmse = np.sqrt(mse)
mae = mean_absolute_error(test, fcast)

print(f'Mean Squared Error: {mse}')
```

```python
print(f'Root Mean Squared Error: {rmse}')
print(f'Mean Absolute Error: {mae}')

plt.title('Температура в городе Базель, Швейцария')
plt.plot(train, label='Train')
plt.plot(fcast, label='Forecast')
plt.plot(test, label='Test')
plt.xlabel('Дата')
plt.ylabel('Температура')
plt.grid()
plt.legend()
plt.savefig('latex/extra/prediction.png')
plt.show()
```

Листинг 5: Процесс обучения модели

```
RUNNING THE L-BFGS-B CODE

  * * *

Machine precision = 2.220D-16
N =              8     M =           10
This problem is unconstrained.

At X0         0 variables are exactly at the bounds

At iterate    0    f=  2.38436D+00    proj g=  3.14481D-01

At iterate    5    f=  2.26655D+00    proj g=  7.15151D-02

At iterate   10    f=  2.25628D+00    proj g=  2.08637D-03

At iterate   15    f=  2.25627D+00    proj g=  4.10184D-04

At iterate   20    f=  2.25626D+00    proj g=  5.49808D-04

  * * *

Tit   = total number of iterations
Tnf   = total number of function evaluations
Tnint = total number of segments explored during Cauchy searches
...
CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH
```

Рис. 3: Результат предсказания модели

Листинг 6: Отклонения результатов модели

```
Mean Squared Error: 12.876863306757866
Root Mean Squared Error: 3.5884346596751437
Mean Absolute Error: 2.9688323338931313
```

Как видно по графику чем дальше он начальных условий, тем хуже предсказание. Причиной служит очень большие колебания температур в близкие даты. Для корректировки модели воспользуемся скользящим прогнозом и, чтобы не усложнять задачу, пронозировать будем на один шаг вперёд, после чего корректировать.

Листинг 7: Использование скользящего прогноза

```
def rolling_forecast(train, test, order, season):
history = train.iloc[:,:].asfreq('d')
model = SARIMAX(history, order=order, seasonal_order=season)
model_fit = model.fit(disp=False)
predictions = []
results = {}
yhat = model_fit.forecast()

predictions.append(yhat)
history = pd.concat([history, pd.DataFrame([test.iloc[0,:]])]).asfreq('d')
for i in range(1, len(test)):
    model = SARIMAX(history, order=order, seasonal_order=season)
    model_fit = model.fit(disp=False)
    yhat = model_fit.forecast()
```

```
        predictions.append(yhat)
        history = pd.concat([history, pd.DataFrame([test.iloc[i,:]])]).asfreq('d')
    mse = mean_squared_error(test, predictions)
    mae = mean_absolute_error(test, predictions)
    rmse = np.sqrt(mse)
    predictions = pd.Series(predictions, index=test.index)
    results['predictions'] = predictions
    results['mse'] = mse
    results['rmse'] = rmse
    results['mae'] = mae
    return results

rolling_fcast = rolling_forecast(train, test, (2, 1, 3), (0, 0, 2, 8))

print(f'Mean Squared Error: {rolling_fcast["mse"]}')
print(f'Root Mean Squared Error: {rolling_fcast["rmse"]}')
print(f'Mean Absolute Error: {rolling_fcast["mae"]}')

plt.title('Температура в городе Базель, Швейцария')
plt.plot(train, label='Train')
plt.plot(rolling_fcast['predictions'], label='Forecast')
plt.plot(test, label='Test')
plt.xlabel('Дата')
plt.ylabel('Температура')
plt.grid()
plt.legend()
plt.savefig('latex/extra/rolling_prediction.png')
plt.show()
```

Рис. 4: Результат предсказания модели с использованием скользящего прогноза

Листинг 8: Отклонения результатов модели с использованием скользящего прогноза

```
Mean Squared Error: 2.885728192709077
Root Mean Squared Error: 1.6987431214604158
Mean Absolute Error: 1.468788413247244
```

Как видно по графику и значениям ошибок модель стала достаточно точно предсказывать температуру на следующий день.

2 Ответы на вопросы

Вопросы

1. Приведите примеры временных рядов при создании цифровых двойников;
2. Что понимают под цифровым двойником эксперта?
3. В чем состоит задача прогнозирования временных рядов?
4. Что понимается под реконструкцией математической модели системы? Какова цель реконструкции ММС?
5. Что такое «переменная состояния» системы? Приведите примеры.
6. Перечислите основные этапы реконструкции математической модели системы
7. Как оценить адекватность разработанной модели?

Ответы

1. Примеры временных рядов при создании цифровых двойников включают данные о температуре, давлении, вибрации и скорости вращения в промышленном оборудовании, показания датчиков в реальном времени для мониторинга состояния объектов, а также данные о потреблении энергии, нагрузке или состоянии здоровья в медицинских устройствах.
2. Цифровой двойник эксперта — это виртуальная модель, которая симулирует знания, опыт и навыки реального специалиста в конкретной области. Такой цифровой двойник может анализировать ситуации, давать рекомендации и принимать решения, имитируя поведение настоящего эксперта, используя данные, алгоритмы и искусственный интеллект.
3. Задача прогнозирования временных рядов состоит в предсказании будущих значений на основе анализа исторических данных, представленных в виде последовательности измерений, сделанных через равные интервалы времени.
4. Реконструкция математической модели системы — это процесс создания или восстановления модели, описывающей поведение реальной системы на основе наблюдаемых данных. Это может включать выявление зависимостей, определение параметров модели или корректировку существующих моделей для более точного отображения процессов. Цель реконструкции ММС заключается в том, чтобы получить адекватное математическое представление системы, которое позволяет точно прогнозировать ее поведение, анализировать различные сценарии и принимать обоснованные решения. Реконструкция важна, когда прямое моделирование невозможно или данные системы неполны.

5. Переменная состояния системы — это величина, которая полностью описывает состояние системы в определенный момент времени. Она отражает всю необходимую информацию для предсказания будущего поведения системы, если известны все входы и законы её динамики.
   Примеры переменных состояния:
   (a) Температура двигателя
   (b) Скорость автомобиля.
6. Основные этапы реконструкции математической модели системы включают:
   (a) Сбор данных
   (b) Анализ системы
   (c) Выбор типа модели
   (d) Идентификация параметров модели
   (e) Калибровка модели
   (f) Проверка и валидация модели
   (g) Использование модели
7. Оценить адекватность модели можно сравнив её с реальными данными.

3 Вывод

При выполнении лабораторной работы была разработана цифровую модель эксперта на основе нейросети SARIMAX, решающая задачу прогнозирования температуры.