Московский государственный технический университет им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления» – ИУ5

Факультет «Радиотехнический» – РТ5

**Отчёт по лабораторной работе №6 по курсу**

**Технологии машинного обучения**

5

(количество листов)

|  |  |  |
| --- | --- | --- |
| Исполнитель |  |  |
| студент группы РТ5-61б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Нижаметдинов М. Ш. |
|  |  | “\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2023 г. |
|  |  |  |
| Проверил |  |  |
| Преподаватель кафедры ИУ5 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гапанюк Ю. Е. |
|  |  | “\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2023 г. |

Москва, 2023 г.

**Задание**

1. Выберите набор данных (датасет) для решения задачи прогнозирования временного ряда.

2. Визуализируйте временной ряд и его основные характеристики.

3. Разделите временной ряд на обучающую и тестовую выборку.

4. Произведите прогнозирование временного ряда с использованием как минимум двух методов.

5. Визуализируйте тестовую выборку и каждый из прогнозов.

Оцените качество прогноза в каждом случае с помощью метрик.

**Набор данных**

<https://raw.githubusercontent.com/mansurik1/MLT/master/LW%206/Project/data/DailyDelhiClimateTrain.csv>

<https://raw.githubusercontent.com/mansurik1/MLT/master/LW%206/Project/data/DailyDelhiClimateTest.csv>

**Исходный текст проекта**

import numpy as np

import pandas as pd

from matplotlib import pyplot

import matplotlib.pyplot as plt

def clear(data):

data['humidity'] = None

data['meantemp'] = None

data['meanpressure'] = None

data\_start = pd.read\_csv('https://raw.githubusercontent.com/mansurik1/MLT/master/LW%206/Project/data/DailyDelhiClimateTrain.csv', header=0, index\_col=0, parse\_dates=True)

clear(data\_start)

data\_test = pd.read\_csv('https://raw.githubusercontent.com/mansurik1/MLT/master/LW%206/Project/data/DailyDelhiClimateTest.csv', header=0, index\_col=0, parse\_dates=True)

clear(data\_test)

data\_train = data\_start['wind\_speed']

data\_train.head()

fig, ax = pyplot.subplots(1, 1, sharex='col', sharey='row', figsize=(10,5))

fig.suptitle('Временной ряд в виде графика')

data\_train.plot(ax=ax, legend=False)

pyplot.show()

for i in range(1, 5):

fig, ax = pyplot.subplots(1, 1, sharex='col', sharey='row', figsize=(5,4))

fig.suptitle(f'Лаг порядка {i}')

pd.plotting.lag\_plot(data\_train, lag=i, ax=ax)

pyplot.show()

fig, ax = pyplot.subplots(1, 1, sharex='col', sharey='row', figsize=(10,5))

fig.suptitle('Плотность вероятности распределения данных')

data\_train.plot(ax=ax, kind='kde', legend=False)

pyplot.show()

from statsmodels.graphics.tsaplots import plot\_acf

plot\_acf(data\_train, lags=30)

plt.tight\_layout()

# https://www.statsmodels.org/dev/generated/statsmodels.tsa.seasonal.seasonal\_decompose.html

from statsmodels.tsa.seasonal import seasonal\_decompose

# Аддитивная модель

def plot\_decompose(data=data\_train, model='add'):

result\_add = seasonal\_decompose(data, model = 'add')

fig = result\_add.plot()

fig.set\_size\_inches((12, 8))

# Перерисовка

fig.tight\_layout()

plt.show()

plot\_decompose()

data\_train2 = data\_start.copy()

# Простое скользящее среднее (SMA)

data\_train2['SMA\_10'] = data\_train2['wind\_speed'].rolling(10, min\_periods=1).mean()

data\_train2['SMA\_20'] = data\_train2['wind\_speed'].rolling(20, min\_periods=1).mean()

print(data\_train2)

fig, ax = pyplot.subplots(1, 1, sharex='col', sharey='row', figsize=(10,5))

fig.suptitle('Временной ряд со скользящими средними')

data\_train2[:100].plot(ax=ax, legend=True)

pyplot.show()

!pip install gplearn

from gplearn.genetic import SymbolicRegressor

xnum = list(range(data\_train.shape[0]))

print(data\_train)

function\_set = ['add', 'sub', 'mul', 'div', 'sin']

est\_gp = SymbolicRegressor(population\_size=2000, metric='mse',

generations=40, stopping\_criteria=0.01,

init\_depth=(5, 10), verbose=1, function\_set=function\_set,

const\_range=(-100, 100), random\_state=0)

est\_gp.fit(np.array(xnum).reshape(-1, 1), data\_train.values.reshape(-1, 1))

import graphviz

dot\_data = est\_gp.\_program.export\_graphviz()

graph = graphviz.Source(dot\_data)

graph

# Записываем предсказания в DataFrame

data\_test['predictions\_GPLEARN'] = list(y\_gp)

fig, ax = pyplot.subplots(1, 1, sharex='col', sharey='row', figsize=(10,5))

fig.suptitle('Предсказания временного ряда (тестовая выборка)')

data\_test.plot(ax=ax, legend=True)

pyplot.show()

# Целочисленная метка шкалы времени

xnum = list(range(data\_start.shape[0] + data\_test.shape[0]))

# Разделение выборки на обучающую и тестовую

train\_size = len(data\_start)

train, test = data\_start['wind\_speed'].values, data\_test['wind\_speed'].values

history\_arima = [x for x in train]

history\_es = [x for x in train]

print(train)

print(test)

from sklearn.metrics import mean\_squared\_error

from statsmodels.tsa.arima.model import ARIMA

from statsmodels.tsa.holtwinters import ExponentialSmoothing

# Параметры модели (p,d,q)

arima\_order = (6,1,0)

# Формирование предсказаний

predictions\_arima = list()

for t in range(len(test)):

model\_arima = ARIMA(history\_arima, order=arima\_order)

model\_arima\_fit = model\_arima.fit()

yhat\_arima = model\_arima\_fit.forecast()[0]

predictions\_arima.append(yhat\_arima)

history\_arima.append(test[t])

# Вычисление метрики RMSE

mean\_squared\_error(test, predictions\_arima, squared=False)

data\_test['predictions\_ARIMA'] = list(predictions\_arima)

fig, ax = pyplot.subplots(1, 1, sharex='col', sharey='row', figsize=(10,5))

fig.suptitle('Предсказания временного ряда (тестовая выборка)')

data\_test.plot(ax=ax, legend=True)

pyplot.show()