Санкт-Петербургский Национальный Исследовательский Университет Информационных Технологий, Механики и Оптики

Кафедра Систем Управления и Информатики

# Лабораторная работа №6

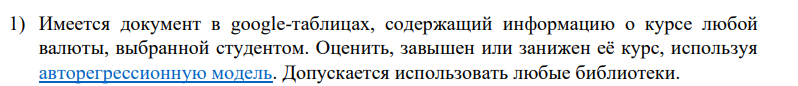
Выполнили: Байков

Тарабанов

Проверил Мусаев А.А.

Санкт-Петербург, 2022

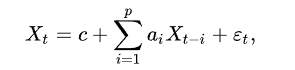
**Задание 1**



**Исходный код**

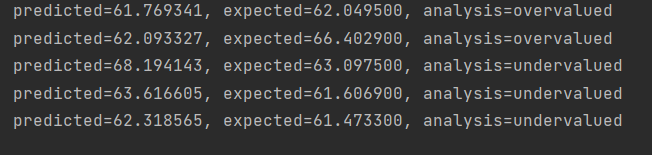
from pandas import read\_csv  
from matplotlib import pyplot  
from statsmodels.tsa.ar\_model import AutoReg  
  
data = read\_csv('3.csv', header=0, index\_col=0, parse\_dates=True, squeeze=True)  
array = data.values  
print(len(array))  
train, test = array, array[-5:]  
print("Test Length", len(test))  
  
window = 50  
print(type(train))  
model = AutoReg(train, lags=50)  
model\_fit = model.fit()  
k = model\_fit.params  
  
cache = train[len(train) - window:]  
cache = [cache[i] for i in range(len(cache))]  
predictions = list()  
for t in range(len(test)):  
 length = len(cache)  
 lag = [cache[i] for i in range(length - window, length)]  
 arr\_predicted = k[0]  
 for d in range(window):  
 arr\_predicted += k[d + 1] \* lag[window - d - 1]  
 arr\_expected = test[t]  
 predictions.append(arr\_predicted)  
 cache.append(arr\_expected)  
 analysis = " "  
 if arr\_predicted > arr\_expected:  
 analysis = "undervalued"  
 if arr\_predicted < arr\_expected:  
 analysis = "overvalued"  
 print(f'predicted=%f, expected=%f, analysis={analysis}' % (arr\_predicted, arr\_expected))  
  
pyplot.plot(test)  
pyplot.plot(predictions, color='red')  
pyplot.show()

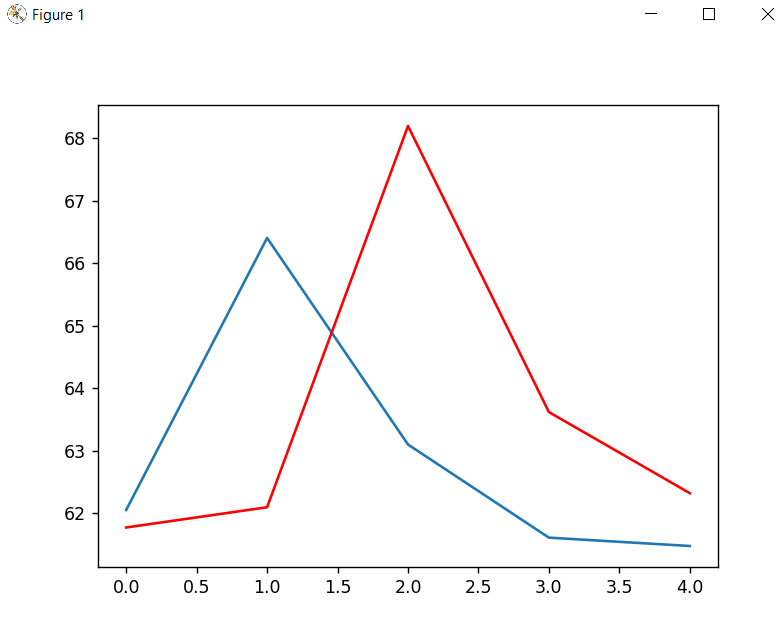
**Решение**



Модель авторегрессии — это модель линейной регрессии, в которой переменные с запаздыванием используются в качестве входных переменных. Для авторегрессии была использована библиотека statsmodels. В ней для реализации AR необходимо указать «lag» переменные вручную и обучить модель линейной регрессии. После прохождении подготовки (test), можно вызывать функцию прогнозирования. Однако в statsmodels не позволяет обновлять модель на мере появления новых наблюдений, поэтому я использую полученные в результате обучения коэффициенты для прогнозирования. Для этого сохраняю 50 последних наблюдений.

**Результат**

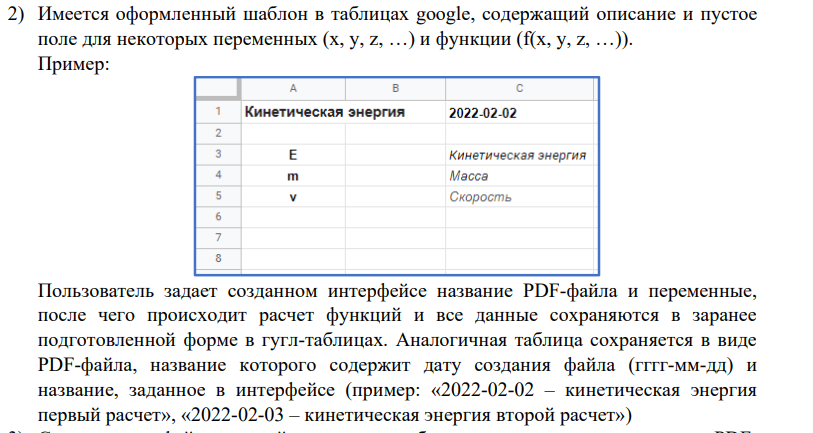




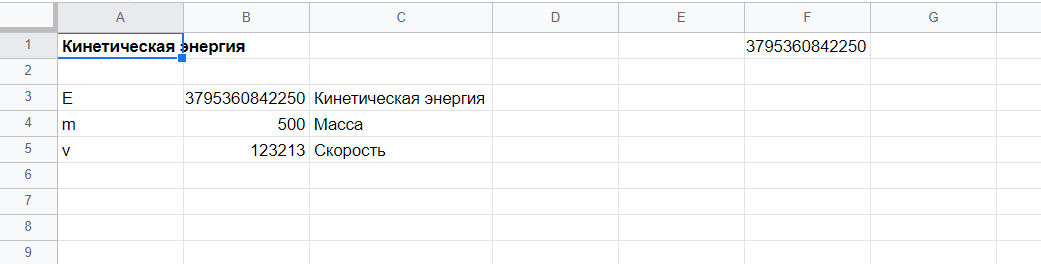
**Вывод**

Был реализован алгоритм прогнозирования методом Авторегрессии. При импользовании большого количества данных для обучения и прогнозирования алгоритм показывает высокий уровень точности, однако на маленьких интервалах наблюдается большая погрешность. Для анализа была взять валютная пара USD/RUB периодом с 01.01.2015 по 01.01.2022 по курсу ЦБ. С помощью алгоритма можно проанализировать, является ли текущий курс завышен или занижен относительно ожидания.

**ЗАДАНИЕ 2**



**ИСХОДНЫЙ КОД**

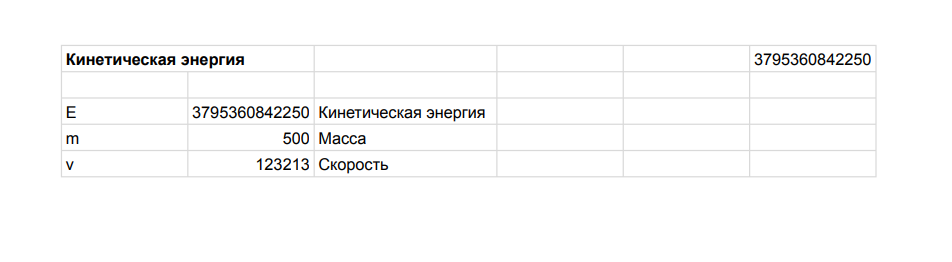


import httplib2  
from googleapiclient.discovery import build  
from datetime import date  
from pathlib import Path  
from borb.pdf import Document  
from borb.pdf import Page  
from borb.pdf import SingleColumnLayout  
from borb.pdf import Paragraph  
from borb.pdf import PDF  
import sys  
  
import os  
from oauth2client.service\_account import ServiceAccountCredentials  
  
  
def get\_service\_sacc():  
 creds\_json = os.path.dirname(\_\_file\_\_) + "/credentials.json"  
 scopes = ['https://www.googleapis.com/auth/spreadsheets']  
 creds\_service = ServiceAccountCredentials.from\_json\_keyfile\_name(creds\_json, scopes).authorize(httplib2.Http())  
 return build('sheets', 'v4', http=creds\_service)  
  
  
print(sys.path)  
  
service = get\_service\_sacc()  
sheet = service.spreadsheets()  
  
sheet\_id = "18q0EJw\_-bNDGfeRsB5Q7EKqre3TsQcyVW1WzhCTjb5w"  
  
resp = sheet.values().batchGet(spreadsheetId=sheet\_id, ranges="name1!B4:B5").execute()  
values = resp['valueRanges'][0]['values']  
m, v = int(values[0][0]), int(values[1][0])  
result = (m \* v \*\* 2) / 2  
  
cell\_update = sheet.values().update(  
 spreadsheetId=sheet\_id,  
 range="name1!B3",  
 valueInputOption="RAW",  
 body={'values': [[result]]}).execute()  
  
print(m, v)  
print(result)  
  
pdf = Document()  
  
page = Page()  
pdf.append\_page(page)  
  
layout = SingleColumnLayout(page)  
  
layout.add(Paragraph("Hello World!"))  
  
pdf\_name = "1к123иewнетичrtyеская энytrергия"  
today = date.today()  
path = (f"{today} - {pdf\_name}.pdf")  
  
with open(Path(path), "wb") as pdf\_file\_handle:  
 PDF.dumps(pdf\_file\_handle, pdf)  
#########################  
url = 'https://docs.google.com/spreadsheets/export?format=pdf&id=' + sheet\_id  
print(url)

**РЕШЕНИЕ**

На скриншоте показана таблица с исходными данными. Кинетическая энергия E высчитывается программой, после чего заносится в таблицу. Для работы с гугл таблицами использована библиотека googleapiclient. После создания проекта, виртуальной электронной почты и подключения бота приложение имеет доступ к таблице с данными. По определенной в программе формуле высчитывается E и заносится в B3, а в F1 для проверки проверяется то же выражение с использованием вычислений самих таблицу. Интерфейс создан простейшим образом через Tkinter.

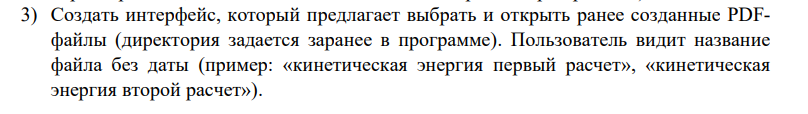




**ВЫВОД**

Для реализации потребовалось изучить работу с Google API и Google Sheets. Интерфейс самый простой, в рамках курса уже была реализована более сложная версия. Также были использованы различные методы с библиотек по типу blob, которая используется для работы с пдф файлами.

**ЗАДАЧА 3**



**ИСХОДНЫЙ КОД**

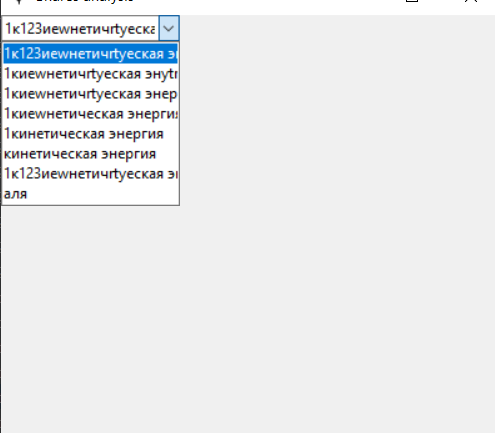
import os  
import re  
from tkinter import \*  
from tkinter.ttk import Combobox  
  
path = "./"  
main = []  
sub = []  
for root, dirs, files in os.walk(path):  
 for file in files:  
 if (file.endswith(".pdf")):  
 temp = file  
 temp = re.sub(r"20\d\d-\d\d-\d\d - ", "", temp)  
 temp = re.sub(r".pdf", "", temp)  
 main.append(file)  
 sub.append(temp)  
# print(main)  
# print(sub)  
  
output\_file = ""  
  
  
def clicked():  
 file\_name = combo\_ticker.get()  
 print("SUB: " , file\_name)  
 i = None  
 i = sub.index(file\_name)  
 a = "./" + main[i]  
 print("MAIN: " , a)  
 os.system(a)  
  
  
Iwindow = Tk()  
Iwindow.title("Shares analysis")  
lbl = Label(Iwindow, text="Choose token")  
Iwindow.geometry('400x400')  
combo\_ticker = Combobox(Iwindow)  
combo\_ticker['values'] = tuple(sub)  
combo\_ticker.current(0)  
combo\_ticker.grid(column=2, row=2)  
btn = Button(Iwindow, text="Don't touch!", command=clicked)  
btn.grid(column=2, row=5)  
  
Iwindow.mainloop()

**РЕШЕНИЕ**

Поиск файлов пдф был реализован через встроенный модуль os и конструкцию из 2х for.Сразу же внутри была создана проверка на наличие даты в названии файла. Дата убирается при помощь регулярного выражения:

temp = re.sub(r"20\d\d-\d\d-\d\d - ", "", temp)

Интерфейс так же реализован через модуль Tkinter. Из себя представляет выпадающий список и одну кнопку. Файл открывается с помощью все того же модуля os => os.system(e).



**Вывод**

Достаточно простая задача с повторением модуля для создания интерфейсов изучением самых основ модуля os. Из сложностей могут быть только регулярные выражения, и тот факт, что их можно писать более рационально.