FLIGHTS

November 29, 2022

Name: Elena Lisa Monika Gaggia

Student ID: 12023845

Course: 1360 Applications of Data Science: Robotic Process Automation with Machine Learning

1 Flight Booking Application

Attention: this programme was created on a Macbook running macOS Ventura 13.1. Because of this, the commands in the terminal may differ from those of other operating systems.

1.1 Project Idea

The motivation behind my project is that I will do my semester abroad in Singapore in 2023. The prices for the flights are very high and fluctuate greatly. Because of this, the bot should help me find the cheapest possible flight with the highest possible guarantee that it will take off on time.

The idea is to program a bot that is able to identify flights between two specified airports (i.e., Singapore Changi Airport and Paris Charles de Gaulle Airport) within a specified timeframe.

The bot should use Machine Learning to determine if the flight will take off with a high enough probability, based on past experiences with the different flights.

If the probability is high enough the bot should book the flight if it also fulfills a specific price criteria.

As the prices for flights can be quite volatile, the goal is to book a reliable and cheap flight without having to check the criteria myself.

Since I have neither enough data nor do I want the bot to actually book a real flight right now, this project serves as a prototype.

The bot is able to perform Machine Learning from a dataset and predict the probability that a flight will be delayed. It then books a flight on the website https://blazedemo.com, which is a flight-booking-simulation.

1.2 1. Get familiar with the data

[23]:
```
import sqlite3
import random
import pandas as pd
import numpy as np
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction import text
import pickle
```

[24]:
```
# Display the dataset
import pandas as pd
df = pd.read_csv("Airlines.csv")
df.head()
```

[24]:

	id	Airline	Flight	AirportFrom	AirportTo	DayOfWeek	Time	Length	Delay
0	1	CO	269	SFO	IAH	3	15	205	1
1	2	US	1558	PHX	CLT	3	15	222	1
2	3	AA	2400	LAX	DFW	3	20	165	1
3	4	AA	2466	SFO	DFW	3	20	195	1
4	5	AS	108	ANC	SEA	3	30	202	0

[25]:
```
# Check the different datatypes of the columns
df = pd.DataFrame(df)
print (df.dtypes)
```

```
id              int64
Airline        object
Flight          int64
AirportFrom    object
AirportTo      object
DayOfWeek       int64
Time            int64
Length          int64
Delay           int64
dtype: object
```

[26]:
```
# Check length of dataset
len(df)
```

[26]: 539383

[27]:
```
# In order to receive a trustworthy result check for duplicates and drop them
# As the flight number ('Flight') is the same for identical flights evaluate␣
 ↪this column
```

```
df = df.drop_duplicates(subset = "Flight")
```

[28]:
```
# Check new length of the dataset
df = df.drop_duplicates(subset = "Flight")
len(df)
```

[28]: 6585

[29]:
```
# Save the pre-processed dataset as a seperate CSV file
df.to_csv("Airlines_processed.csv", index = False)
```

1.2.1 Description of the dataset

The new dataset Airlines_processed.csv has 6585 entries. All columns are integers, except for the Airline, AirportFrom and AirportTo column, which are objects.

Columns: - **id** (serial number) - **Airline** (airline of the flight) - **Flight** (type of aircraft) - **AirportFrom** (source airport) - **AirportTo** (destination airport) - **DayOfWeek** (weekday) - **Time** (time of flight) - **Length** (length of flight) - **Delay** (whether the flight was delayed or not)

Airlines: - Alaska Airlines AS / ASA - American Airlines AA/AAL - Air Canada AC/ACA - Aeromexico AM / AMX - Continental Airlines CO / COA - Delta Airlines DL / DAL - FedEx FX / FDX - Hawaiian Airlines HA / HAL - Northwest Airlines NW / NWA - Polar Air Cargo PO / PAC - Southwest Airlines SW / SWA - United Airlines UA / UAL - United Parcel (UPS) 5X / UPS - Virgin Atlantic VS / VIR - VivaAerobús VB / VIV - WestJet WS / WJ

Airports: - ATL - Hartsfield-Jackson Atlanta International Airport - Georgia - AUS - Austin-Bergstrom International Airport - Texas - BNA - Nashville International Airport - Tennessee - BOS - Boston Logan International Airport - Massachusetts - BWI - Baltimore-Washington International Thurgood Marshall Airport - Washington - CLT - Charlotte Douglas International Airport - North Carolina - DAL - Dallas Love Field - Texas - DCA - Ronald Reagan Washington National Airport - Arlington, Virginia - DEN - Denver International Airport - Colorado - DFW - Dallas/Fort Worth International Airport - Texas - DTW - Detroit Metropolitan Airport - Michigan - EWR - Newark Liberty International Airport - New Jersey - FLL - Fort Lauderdale–Hollywood International Airport - Florida - HNL - Daniel K. Inouye International Airport - Honolulu, Hawaii - HOU - William P. Hobby Airport - Houston, Texas - IAD - Dulles International Airport - Virginia - IAH - George Bush Intercontinental Airport - Houston, Texas - JFK - John F. Kennedy International Airport - Queens, New York - LAS - McCarran International Airport - Las Vegas, Nevada - LAX - Los Angeles International Airport - California - LGA - LaGuardia Airport - Queens, New York - MCO - Orlando International Airport - Florida - MDW - Chicago Midway International Airport - Illinois - MIA - Miami International Airport - Florida - MSP - Minneapolis–Saint Paul International Airport - Minnesota - MSY - Louis Armstrong New Orleans International Airport - Louisiana - OAK - Oakland International Airport - California - ORD - O'Hare International Airport - Chicago, Illinois - PDX - Portland International Airport - Oregon - PHL - Philadelphia International Airport - Pennsylvania - PHX - Phoenix Sky Harbor International Airport - Arizona - RDU - Raleigh-Durham International Airport - North Carolina - SAN - San Diego International Airport - California - SEA - Seattle–Tacoma International Airport - Washington - SFO - San Francisco International Airport - California - SJC - Norman Y. Mineta San Jose International Airport - California - SLC - Salt Lake City International Airport - Utah - SMF - Sacramento International

Airport - California - STL - St. Louis Lambert International Airport - Missouri - TPA - Tampa International Airport - Florida

1.3 2. Start the Webdriver manager

First we need to start the Webdriver manager using the command 'sudo webdrivermanager chrome' in the Terminal. This application makes it easy to automate tasks.

1.4 3. Build Flask application

In the next step we create a python file named 'flights.py' in which we build the Flask app. Simply create a new .py-file in your directory and copy the code below into it. As soon as the programme has been executed, the application runs on a local host, which is now accessible via the URL http://127.0.0.1:4040/. This Flask app accesses the dataset, converts it to a database and displays it in a subpage. It also creates the two files "flights.db" and "tmpdat.csv".

```
from flask import Flask, render_template, request
import sqlite3

app = Flask(__name__)

def getconn():
    return sqlite3.connect("flights.db")

@app.route("/")
def flights():
    return """<h1>Flights App</h1>
    <ul>
    <li><a href=listflight>List</a></li>
    </ul>"""

@app.route('/listflight')
def list_flights():
    conn = getconn()
    cur = conn.cursor()
    rows = cur.execute("select id, airline, flight, airportfrom, airportto, dayofweek, time, le
    html = "<h3>Flights</h3><table>\n"
    for row in rows:
        html += ("<tr>"
        + "<td align=right>%s"
        + "<td align=right>%s"
        + "<td align=right>%s"
        + "<td align=right>%s"
        + "<td align=right>%s"
        + "<td align=right>%s"
        + "<td align=right>%s"
        + "<td align=right>%s\n") % row
    conn.close()
```

```
    return html + "</table>\n"

import pandas as pd
import numpy as np

df = pd.read_csv("Airlines_processed.csv")
df.head()

data = np.asarray(df)

conn = sqlite3.connect("flights.db")
cur = conn.cursor()
cur.execute("drop table if exists flights")
conn.commit()

fout = open("tmpdat.csv", "w")
fout.write("id,airline,flight,airportfrom,airportto,dayofweek,time,length,delay\n")
for x in data:
    fout.write("%s,%s,%s,%s,%s,%s,%s,%s,%s\n" % tuple(x[:9]))
fout.close()
pd.read_csv("tmpdat.csv").to_sql("flights", conn, index = False)
conn.commit()

app.run(port=4040)
```

1.5 4. Creation of the first bot

The most important characteristic of the bot is that it uses the Chrome browser. Of course this depends on which web driver manager was started before. In addition, it accesses functions of the library 'mytools.py', which we will create in the next step. Save the text below as 'flightbot_1.txt'.

*** Settings *** Library SeleniumLibrary Library mytools.py

*** Variables *** ${url} http://127.0.0.1:4040 ${browser} Chrome

*** Test Cases *** Start Flights Open Browser ${url} ${browser} Create My File list.txt Create My File result.txt

Enter Flight Click Link //a[@href="listflight"] Iterate Through ${elements}= Get WebElements //td FOR ${element} IN @elements Append My File list.txt ${element.text} END

In addition, the following code must be saved as 'mytools.py' so that the bot can access it. This library allows the bot to iterate trough and process the website entries in the text file 'list.txt'. If you want to work with all 6585 entries, it takes a long time to iterate and save them in the text file. However, as much data as possible should be used for the evaluation. If the result is not that important, you can also set the limit to of selected data in 'flighty.py' to 100 or 500, for example, to speed up the process.

```
import sqlite3
import random
import pandas as pd
```

```
import numpy as np
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction import text
import pickle

def create_my_file(fn):
    fp = open(fn, 'w')
    fp.close()

def append_my_file(fn, txt):
    fp = open(fn, 'a')
    fp.write(txt + '\n')
    fp.close()

def append_my_file2(fn, txt):
    fp = open(fn, 'a')
    fa = open(fn)
    if len(fa.readlines())%3 == 2:
        fp.write(str(len(fp.readlines())) + ';' + "\n")
    else:
        fp.write(txt + ';')
    fp.close()
```

In the next step you can execute the bot using the command 'robot flightbot_1.txt'.

1.6 5. Perform Machine Learning in order to build mytools.py library

In the next step, the machine learning function is created, which is then included in 'mytools.py' so that the bot itself can perform machine learning.

1.6.1 Pre-Processing

[33]:
```
# Open and read the textfile created by the bot
f = open("list.txt")
flights = f.readlines()
# Split it by the number of columns
flights_2 = np.array_split(flights, len(flights)/9)

# Choose labels and create data frame
labels = ["id", "Airline", "Flight", "AirportFrom", "AirportTo", "DayOfWeek",␣
 ↪"Time", "Length", "Delay"]
df = pd.DataFrame.from_records(flights_2, columns = labels)
df = df.replace('\n', '', regex=True)
df.head()
```

[33]:

	id	Airline	Flight	AirportFrom	AirportTo	DayOfWeek	Time	Length	Delay
0	1	CO	269	SFO	IAH	3	15	205	1
1	2	US	1558	PHX	CLT	3	15	222	1
2	3	AA	2400	LAX	DFW	3	20	165	1
3	4	AA	2466	SFO	DFW	3	20	195	1
4	5	AS	108	ANC	SEA	3	30	202	0

[34]:
```
# The value count reveals an imbalance in the number of observations per delay␣
 ↪classes
df.value_counts('Delay')
```

[34]:
```
Delay
0    4108
1    2477
dtype: int64
```

[35]:
```
# Get equal numbers of observations
minobs = min(df.value_counts('Delay').values)
df = df.groupby('Delay').sample(n=minobs, random_state=1).sample(frac=1,␣
 ↪random_state=1)
df.iloc[:5,:]
print(df['Delay'].value_counts())
```

```
0    2477
1    2477
Name: Delay, dtype: int64
```

1.6.2 Deep Learning

Assumption: From experience it seems that regardless of severe weather or other uncontrollable variables, the time of departure has an impact on whether a flight is delayed. The earlier in the day a flight departs, the less likely it is to be delayed.

[41]:
```
# Convert 'Time' column into a numpy array X
X = np.asarray(df["Time"])
X = X.reshape(-1, 1)

# Convert 'Delay' column into a numpy array Y
Y = np.asarray(df["Delay"], dtype='int8')
Y = Y.reshape(-1, 1)
Y = Y.ravel()
```

[44]:
```
# Use a simple feed-forward net as provided by the science kit learn package,␣
 ↪the Multi-Layer-Perceptron Classifier
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.1)
print('training size:', X_train.shape[0],
      'testing size:', X_test.shape[0],
      'label counts:', np.unique(y_train, return_counts=True)[1])
```

```
clf = MLPClassifier(hidden_layer_sizes=(100,20,), max_iter=50).fit(X_train,␣
 ↪y_train)
print('score train:', clf.score(X_train, y_train))
print('score test: ', clf.score(X_test, y_test))
```

```
training size: 4458 testing size: 496 label counts: [2216 2242]
score train: 0.4970838941229251
score test:  0.5262096774193549
```

1.7 6. Creation of the second bot

This optimised bot is now also able to pre-process the data from 'list.txt' and perform Machine Learning on it. save the text below as 'flightbot_2.txt'.

*** Settings *** Library SeleniumLibrary Library mytools.py

*** Variables *** ${url} http://127.0.0.1:4040 ${browser} Chrome

*** Test Cases *** Start Flights Open Browser ${url} ${browser} Create My File list.txt Create My File result.txt

Enter Flight Click Link //a[@href="listflight"]

Iterate Through ${elements}= Get WebElements //td FOR ${element} IN @elements Append My File list.txt ${element.text} END

Predict Delay Predict Delay list.txt result.txt

However, the 'mytools.py' file must be updated the following so that the bot can be executed. Additionally, compared to the library before, the bot will now also perform Machine Learning and save the result in a pickle file in order to be available for later use.

```
import sqlite3
import random
import pandas as pd
import numpy as np
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction import text
import pickle

def create_my_file(fn):
    fp = open(fn, 'w')
    fp.close()
def append_my_file(fn, txt):
    fp = open(fn, 'a')
    fp.write(txt + '\n')
    fp.close()
```

```
def predict_delay(fn, fp):
    f = open(fn)
    flights = f.readlines()
    flights_2 = np.array_split(flights, len(flights)/9)

    labels = ["id", "Airline", "Flight", "AirportFrom", "AirportTo", "DayOfWeek", "Time", "Leng
    df = pd.DataFrame.from_records(flights_2, columns = labels)
    df = df.replace('\n', '', regex=True)

    minobs = min(df.value_counts('Delay').values)
    df = df.groupby('Delay').sample(n=minobs, random_state=1).sample(frac=1, random_state=1)
    df.iloc[:5,:]

    X = np.asarray(df["Time"])
    X = X.reshape(-1, 1)

    Y = np.asarray(df["Delay"], dtype='int8')
    Y = Y.reshape(-1, 1)
    Y = Y.ravel()

    clff, Xmax2 = pickle.load(open('clf.pkl', 'rb'))
    fpp = open(fp, 'a')

    df = df.reset_index()
    results = clff.predict(X)
    for i in range(len(df["Time"])):
            fpp.write(df["Flight"][i] + " - " + df["Delay"][i] + " - " + str(results[i]) + '\n

    fpp.close()


def append_my_file2(fn, txt):
    fp = open(fn, 'a')
    fa = open(fn)
    if len(fa.readlines())%3 == 2:
        fp.write(str(len(fp.readlines())) + ';' + "\n")
    else:
        fp.write(txt + ';')
    fp.close()
```

1.8 7. Creation of the third bot

This optimized and final bot is now finally additionally able to book the flight. Save the text below as 'flightbot_3.txt' in your register and execute it.

*** Settings *** Library SeleniumLibrary Library mytools.py

*** Variables *** ${url1} http://127.0.0.1:4040 ${url2} http://blazedemo.com/ ${browser} Chrome

*** Test Cases *** Start Flights Open Browser ${url1} ${browser} Create My File list.txt Create My File result.txt

Enter Flight Click Link //a[@href=”listflight”] Iterate Through ${elements}= Get WebElements //td FOR ${element} IN @elements Append My File list.txt ${element.text} END

Predict Delay Predict Delay list.txt result.txt

Search Flight [Tags] search_flights Open browser ${url2} ${browser} Select From List By Value xpath://select[@name='fromPort'] Mexico City Select From List by Value xpath://select[@name='toPort'] Buenos Aires Click Button css:input[type=‘submit’] Click Button css:input[type=‘submit’]

Book Flight (Personal Data) [Tags] book_flight_personal Input Text //input[@name=”inputName”] Elena Gaggia Input Text //input[@name=”address”] 5th Avenue Central Park Input Text //input[@name=”city”] New York Input Text //input[@name=”state”] New York Input Text //input[@name=”zipCode”] 10019

Book Flight (Credit Card Data) [Tags] book_flight_payment
Input Text //input[@name=”creditCardNumber”] 123412312341234 Input Text //input[@name=”creditCardYear”] 2021 Input Text //input[@name=”nameOnCard”] Elena Gaggia Click Button css:input[type=‘submit’]

1.9 Appendix and future ideas

In order to implement this project in the long term, 3 points still need to be considered:

1. Data must be found which, if possible, cover all air routes worldwide.
2. A site must be found that offers flights of all kinds (i.e., Google Flights).
3. The bot could be complemented with two conditional statements when configuring the chosen page. The first would check the result from Machine Learning and determine the lowest possible value that would indicate a delay. A second conditional could also set a minimum price level.

This project can serve as a prototype for such a bot.

1.10 Links

- Lecture Resources: http://mitloehner.com/lehre/rpa/
- Dataset: https://www.kaggle.com/code/jeffhollis/airline-delay-predictions/notebook
- Blazedemo: https://blazedemo.com