Name: Abhishek Guleri
Roll No. : 185509
Lab: Data Mining
Assignment No. : 7
Branch: CSE DD

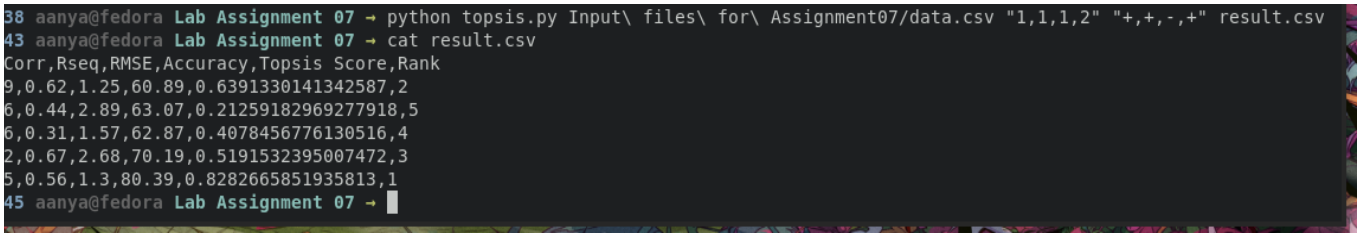
---

1. **Develop a command line program to implement the TOPSIS method**

```
import pandas as pd
import os
import sys

def main():
	# Argument check: args = 5
	if len(sys.argv) != 5:
		args_check = "greater" if (len(sys.argv) > 5) else "less"
		print("Error: Number of arguments is ", args_check, " than 5")
		print("Usage: python topsis.py <InputDataFile> <Weights> <Impachts>
<ResultFileName>")
		exit(1)

	# Input Data File check
	elif not os.path.isfile(sys.argv[1]):
	print(f"ERROR: Input data file doesn't exists! : {sys.argv[1]}")
	exit(1)

	# File extension check
	elif ".csv" != (os.path.splitext(sys.argv[1]))[1]:
	print(f"ERROR: Input data file is not in csv format! : {sys.argv[1]}")
	exit(1)

	else:
	dataset, temp_dataset = pd.read_csv(
		sys.argv[1]), pd.read_csv(sys.argv[1])
	nCol = len(temp_dataset.columns.values)

	# less than 3 columns in input dataset
	if nCol < 3:
		print("ERROR: Input data file have less than 3 columns")
		exit(1)

	# Handling non-numeric value
	for i in range(1, nCol):
		pd.to_numeric(dataset.iloc[:, i], errors='coerce')
		dataset.iloc[:, i].fillna(
			(dataset.iloc[:, i].mean()), inplace=True)
```

```
	# Handling input weights and impacts args
	try:
		weights = [int(i) for i in sys.argv[2].split(',')]
	except:
		print("ERROR: Either the weights contains non-integers or not comma
sperated")
		exit(1)

	impact = sys.argv[3].split(',')
	for i in impact:
		if not (i == '+' or i == '-'):
		print("ERROR: Either the impacts contains symbol other than '+' and
'-' or are not correctly comma seperated")
		exit(1)

	# Checking number of column,weights and impacts is same or not
	if nCol != len(weights)+1 or nCol != len(impact)+1:
		print(
		"ERROR : Number of weights, number of impacts and number of columns
not same")
		exit(1)

	if (".csv" != (os.path.splitext(sys.argv[4]))[1]):
		print("ERROR : Output file extension is wrong")
		exit(1)
	if os.path.isfile(sys.argv[4]):
		os.remove(sys.argv[4])
	# print(" No error found\n\n Applying Topsis Algorithm...\n")
	topsis_pipy(temp_dataset, dataset, nCol, weights, impact)

def normalize(temp_dataset, nCol, weights):
	for i in range(1, nCol):
	temp = 0
	for j in range(len(temp_dataset)):
		temp = temp + temp_dataset.iloc[j, i]**2
	temp = temp**0.5

	for j in range(len(temp_dataset)):
		temp_dataset.iat[j, i] = (
			temp_dataset.iloc[j, i] / temp) * weights[i-1]
	return temp_dataset

def calc_values(temp_dataset, nCol, impact):
	p_sln = (temp_dataset.max().values)[1:]
	n_sln = (temp_dataset.min().values)[1:]
	for i in range(1, nCol):
	if impact[i-1] == '-':
```

```
                p_sln[i-1], n_sln[i-1] = n_sln[i-1], p_sln[i-1]
        return p_sln, n_sln

def topsis_pipy(temp_dataset, dataset, nCol, weights, impact):
        temp_dataset = normalize(temp_dataset, nCol, weights)
        p_sln, n_sln = calc_values(temp_dataset, nCol, impact)

        score = []
        for i in range(len(temp_dataset)):
        temp_p, temp_n = 0, 0
        for j in range(1, nCol):
                temp_p += (p_sln[j-1] - temp_dataset.iloc[i, j])**2
                temp_n += (n_sln[j-1] - temp_dataset.iloc[i, j])**2
        temp_p , temp_n = temp_p**0.5, temp_n**0.5
        score.append(temp_n/(temp_p + temp_n))
        dataset['Topsis Score'] = score

        dataset['Rank'] = (dataset['Topsis Score'].rank(
        method='max', ascending=False))
        dataset = dataset.astype({"Rank": int})

        dataset.to_csv(sys.argv[4], index=False)

if __name__ == "__main__":
        main()
```



2. **Develop a web service for TOPSIS**
   Link: topsis-guleri24.herokuapp.com/
   Source Code: [Github/TOPSIS]