Python Programming Without Libraries

Introduction-Python Programming Without Libraries

This report is based on extensive research and implementation of different machine learning algorithms that will be implemented in the software part of this project. The system provides book recommendations based on analysis of historical data of book details and user pReferences. The system implements different machine learning algorithms and uses python language to implement those algorithms into the datasets. Based on these algorithms, a correlation between users and books are found. Based on these correlation results, recommendations to users are made. For the implementation, euclidean distance, Minkowski distance, spearman correlation, Chebyshev distance, Hamming distance, cosine similarity, and Pearson correlation have been done. Based on these results and input from users, a recommendation system has been built.

Implementation and Justification

All the implementations in this project have been done to meet specific requirements. Decisions related to choosing and implementing different algorithms required in-depth knowledge of different machine learning algorithms and their application areas. The Euclidean distance has been used for this implementation as this helps identify the dissimilarity and similarity between users. This algorithm has a heavy implementation in clustering in machine learning. It is the most commonly used distance measure algorithm that is used in machine learning projects for distance calculation when the variables are continuous. In addition, it is the most suitable algorithm when the implementation is done between two columns or data frames with integer or float values (Machinelearningmastery, 2021).

The Minkowski distance has been used in this report for measuring the distance between the nearest variables present in the dataset. It is helpful while finding the similarity of distances between two vectors. In addition, in machine learning, Minkowski distance is used for determining distance in sizes. In addition, hamming distance will be implemented for measuring the distance between two length lines with equal length. It is used for different areas of error correction and comparison of opposing datasets (Datascience, 2020). As the project work includes finding correlations and similarities between users and books, these algorithms are used for the specific reasons that are already mentioned in their descriptions.

The project work includes works with three datasets such as details of books, users, and user ratings; these algorithms are used for finding differences between users and books for help in building recommendation systems.

In addition, the spearman correlation has been chosen for working with the monotonous relationships. As the datasets that are selected for this work includes high dependency for the recommendation system, the spearman correlation will help in finding the correlation on monotonous values in datasets. Hence, this algorithm is important for this implementation (Towardsdatascience, 2020).

In addition, Chebyshev distance has been used for the implementation of the projects for finding distance similarities and cluster entities between entries in datasets that are common in nature. In the case of working with various groups of data, these measures are implemented (Sciencedirect, 2021). Cosine similarity is used for calculating the similarity between users and the similarity between different books. The relation is mapped to similar datasets and finds the similarity. More the value, the more the similarity will be. Higher similarity helps in providing recommendations which is the aim of this project. For this reason, this algorithm has been chosen and used in this implementation (Sciencedirect, 2020).

Structure of program

The implementation of the program starts with loading datasets in a jupyter notebook. Here, three datasets are added, such as book reviews, books, users. All these datasets are in csv format. In the first module of the program, all the datasets are loaded, and a dictionary for user preference has been built. This dictionary contains user ids, ISBN, Book titles, authors, and all other relevant details from all three datasets.

After completing this task, the second module will be built. This module includes different functions for implementing distance measuring algorithms. These algorithms will provide similarity outcomes between users and books. The functions used are Euclidean distance, Minkowski distance, spearman correlation, Chebyshev distance, Hamming distance, cosine similarity, and Pearson correlation. These functions will calculate distance and similarity among datasets created data frames targeted to different columns. All these functions will take three parameters from user preferences.

After this implementation, the programme will be developed for computing similarity between books as well. This function will come under the similarity module.

The last part of the program will include a book recommendation system which will include taking input from users and providing book recommendations based on cosine similarity and distances that were implemented earlier. This part includes finding similar users from the existing database and based on their similarity count and this program will provide book recommendations to the user. A pictorial representation of structure of the program is provided below.

Figure 1: Structure of Program

Design of functions

Design of the algorithms is based on the user ids that are available after the calculation of user preferences. With the screenshots of codes, functionalities will be defined.

Loading Data

Figure 2: Loading Data Into Data frames

The above figure shows the codes for loading data into data frames. For doing this task of loading data, pandas library has been imported as pd and data read by using pd. Top 5000 data has been selected for ease of calculations.

Figure 3: Output after merge operation of user and rating data

The above output shows the result after merge of two datasets, user and rating. Inner join has been done based on User-ID.

Figure 4: Output of merge operation on output 1 and book data

The above output shows the output after the merge operation on output one and output 2. Basically, after this operation, all the datasets are merged based on the same id.

Figure 5: User Preference dictionary

Figure 4 shows the user preference that has been created from the output 3. It includes selected columns based on user id. For doing this, columns have been selected from output 3 showing output for individual users.

The above figure represents the function of calculating the Euclidean distance between user id. For doing this task, from the math library of python, the sqrt function has been imported that will be used in the function for mathematical calculations. In addition, from scipy.spatial, distance, euclidean and cityblock has been imported. In the data frame of eu1 from output1, the top 715 data has been extracted as there is the same number of data in the user-preference data frame. In the function Euclidean_Dist, user-preference, eu1 has been passed, and their User-ID has been used for calculating the distance between user Ids.

The figure 6 shows the call of the function Euclidean_Dist and passing data frames named eu1 and user-preference for calculation of distance between user ids. In addition, it shows the result.

The Minkowski function has been developed for calculating the Minkowski distance of user ids. For this function, the math library of python is needed. Hence, the library has been

The program implements hamming distance. A function has been made for calculating Hamming distance. The value of x and y has been passed in the function. The value of i and count is set to 0. Then the while loop executes till the length of i reaches the length of x, and calculation is made based on these areas. Then in the print function, the hammingDist function is called by sending x and y values and printing the output. In this case, the output is 28.

Justification for Similarity Metric

For calculating the similarities, different similarity matrices present in the machine learning algorithm libraries concepts have been used. In this project, similarity matrices such as euclidean distance, manhattan distance, cosine similarity, and many other functions have been formed and used. According to the euclidean matrices, different distances have been shown in the form of matrices. The result of Minkowski distance has come out as 8685.000, which is a justified value for calculating similarities. The Spearman correlation matrix provides the output as 0.767, which is very close to the desired value. Hence, the accuracy of this matrix is very high. Hamming distance has provided the value as 28. The Chebyshev Distance output came as 1616. Cosine similarity has come as 0.999, which is a very high similarity matrix with very high accuracy. In addition, the Pearson correlation has given an output of 0.735, which is a very high correlation result. In addition, the similarity between users and books depending on inputs are 0.44 and 0.66, respectively. 0.66 is a high relation, and 0.44 is a very low relation. These results will depend on the user inputs. The recommendation of these areas will be based on user similarity. The similarity will be calculated on the basis of their activity, such as ratings of their books and similar books from their choices. For all these operations, cosine similarity has been used. And depending on the result, the recommendation has been made. Hence, from all these outputs, it can be shown that the outputs are very justified, and the most accurate value has come from cosine similarity and Spearman correlation matrices.

Self Reflection

The project has been done based on research on different areas of machine learning. Machine learning programs include different algorithms for ease of execution of complex programs. I have done research on different algorithms for measuring distances between users and books such as euclidean distance, Minkowski distance, spearman correlation, Chebyshev distance, Hamming distance, cosine similarity, and Pearson correlation. All these functions serve different purposes. Through these algorithms, differences between similar entities are calculated. In this case of the project, similarities of books and different users are calculated. Mathematical explanation and deep understanding regarding equations and implementations was the toughest part of the job. Despite that, the implementation of these algorithms is heavily dependent on the proceeding of data. This took some additional time for implementation. In addition, for calculating the Spearman correlation, the alpha value has been set to 0.05. Changes in this value will generate different results. Hence, this is a challenge that will be faced in case of further usages of spearman correlation. In the case of calculating Minkowski distance, the p-value has been to 3 by default. In case of changes in the value, again, the result will be changed. These are the challenges that I have faced while developing this project. In addition, all the functions have been generated from scratch in spite of using the inbuilt functions that are available in the python library. Hence, this has taken additional time to develop. In task 4, a recommendation system has been developed where there is a requirement of taking user input. Users are required to provide user ids and books ISBN, and based on their inputs; the recommendation will be made by calculating the similarity matrix. Hence changes in functions have been difficult to implement. All over, the project has helped me gain in-depth knowledge in different areas of python language as well as its usages in machine learning implementations. This project required solo work and self-knowledge. Hence, this has been an amazing experience for me as it helped in upskilling my knowledge.

Conclusion

The overall project work and the report include a deep understanding of different areas of machine learning model and algorithms. For building this recommendation system, different machine learning algorithms such as euclidean distance, Minkowski distance, spearman correlation, Chebyshev distance, Hamming distance, cosine similarity, and Pearson correlation have been used for calculating similarity and distances between users and books. These findings have been used in the final part that is building the book recommendation system. The system takes input from users in the form of their id and provides book recommendations by using different machine learning algorithms. In addition, this project developed a similarity matrix for users and books so that it becomes helpful for finding similar books. The recommendation system that has been developed provides recommendations with maintaining higher accuracy. In the future, the system can be developed further by implementing more advanced machine learning algorithms. Hence, this will provide recommendations with better accuracy. In addition, the system also provides similarity scores for books which help in identifying similar books for recommendations. The last part includes a recommendation system, where the user provides input and the system works by using those inputs. Machine learning algorithms have been used for this operation.

Reference

Data science. 2020. *Distance Metrics in Machine Learning*. Viewed on 2nd December 2021. From https://datascience.foundation/datatalk/distance-metrics-in-machine-learning

Machinelearningmastery. 2021. *4 Distance Measures For Machine Learning.* Viewed on 2nd December 2021. From>

Sciencedirect. 2020. *Getting to know your data.*Viewed on 2nd December 2021.>

Sciencedirect. 2021. *Chebyshev Distance*. Viewed on 2nd December 2021. From https://www.sciencedirect.com/topics/computer-science/chebyshev-distance

Towardsdatascience. 2020. *Spearman Correlation Coefficient*. Viewed on 2nd December 2021. Fromhttps://towardsdatascience.com/clearly-explained-pearson-v-s-spearman-correlation-coefficient-ada2f473b8

Appendix

Task1

```
import pandas as pd
import numpy as np
,)
,>,>
user.columns
rating.columns
book.columns
rating,
,)
output1
book,,)
output2.columns
output2
"ISBN", "Book-Title", "Book-Author", "Year-Of-Publication", "Book-Rating"]]
user_pereference.info()
```

Task 2

euclidean distance

```
from math import sqrt
from scipy.spatial.distance import cityblock, euclidean
>
eu1.head()
def Euclidean_Dist(user_pereference, eu1,>
return np.linalg.norm(user_pereference[cols].values - eu1[cols].values,
```

>

Euclidean_Dist(user_pereference, eu1)

minkowski distance

```
>>
from math import *
from decimal import Decimal
def p_root(value, root):
/ float(root)
return round (Decimal(value) **
Decimal(root_value), 3)
def minkowski_distance(x, y, p_value):
# pass the p_root function to calculate
# all the value of vector parallelly
return (p_root(sum(pow(abs(a-b), p_value)
for a, b in zip(x, y)), p_value))
# Driver Code
>
print(minkowski_distance(x, y, p))
# generate related variables
from numpy.random import rand
* 20 + (rand(1000) * 10)
data1.shape
```

Spearman correlation

```
import matplotlib.pyplot as plt
>>
from scipy.stats import spearmanr
coef, y)
print('Spearmans correlation coefficient: %.3f' % coef)
>
if p > alpha:
print('Samples are uncorrelated (fail to reject H0) % p)
else:
print('Samples are correlated (reject H0) % p)
```

chebyshev distance

import scipy

from scipy.spatial import distance

scipy.spatial.distance.chebyshev(x,y)

Hamming distance

def hammingDist(x, y):

>>

while(i < len(x)):

if(x[i]>

count>

i>

return count

print(hammingDist(x, y))

pearson correlation

>>

from scipy.stats import pearsonr

corr,>

print('Pearsons correlation: %.3f' % corr)

cosine similarity

pip install similarity

import math

>>

def cosine_similarity(co1, co2):

return sum([i*j for i,j in zip(co1, co2)])/(math.sqrt(sum([i*i for i in co1]))* math.sqrt(sum([i*i for i in co2])))

cosine_similarity(co1,co2)

print(co1, co2, cosine_similarity(co1,co2))

Task 3

>>>>

import nltk

nltk.download('stopwords')

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
> =[]
# remove stop words from the string
for w in co1 if not w in sw}
for w in co2 if not w in sw}
# form a set containing keywords of both strings
>
for w in rvector:
if w in X_set: l1.append(1) # create a vector
else: l1.append(0)
if w in Y_set: l2.append(1)
else: l2.append(0)
>
# cosine formula
# cosine formula
for i in range(len(rvector)):
> / float((sum(l1)*sum(l2))**0.5)
print("similarity: ", cosine)
```

Task 4

Similarity of Two Books

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
> =[] First Book ISBN") Second Book ISBN")
# remove stop words from the string
for w in co1 if not w in sw} for w in co2 if not w in sw}
# form a set containing keywords of both strings
>
for w in rvector:
if w in X_set: l1.append(1) # create a vector
else: l1.append(0)
```

```
if w in Y_set: l2.append(1)
else: l2.append(0)
>
# cosine formula
for i in range(len(rvector)):
> / float((sum(l1)*sum(l2))**0.5)
print("similarity: ", cosine)
```

Similarity of Two User

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
> =[] First User") Second User")
# remove stop words from the string
for w in co1 if not w in sw} for w in co2 if not w in sw}
# form a set containing keywords of both strings
>
for w in rvector:
if w in X_set: l1.append(1) # create a vector
else: l1.append(0)
if w in Y_set: l2.append(1)
else: l2.append(0)
>
# cosine formula
for i in range(len(rvector)):
> / float((sum(l1)*sum(l2))**0.5)
print("similarity: ", cosine)
```