# Cosine Similarity,

This metric measures the cosine of the angle between two non-zero vectors in a multi-dimensional space. It is particularly useful for interval and ratio-scaled attributes, where the magnitude of the vectors is not as important as their direction.

### Importing required Libraries

import pandas as pd  
import numpy as np  
from sklearn.preprocessing import LabelEncoder  
from scipy.spatial.distance import cosine

### Load Datasets

# Load datasets  
adult\_df = pd.read\_csv("../adult/adult\_trim.data", header=None) # No header  
titanic\_df = pd.read\_csv('../titanic/titanic\_trim.csv') # Has header  
  
# Rename columns for clarity  
adult\_df.columns = ["age", "workclass", "fnlwgt", "education", "education\_num",   
 "marital\_status", "occupation", "relationship", "race", "sex",   
 "capital\_gain", "capital\_loss", "hours\_per\_week", "native\_country", "income"]  
adult\_df.dropna(inplace=True)

adult\_df

age workclass fnlwgt education education\_num \  
0 39 State-gov 77516 Bachelors 13   
1 50 Self-emp-not-inc 83311 Bachelors 13   
2 38 Private 215646 HS-grad 9   
3 53 Private 234721 11th 7   
4 28 Private 338409 Bachelors 13   
.. ... ... ... ... ...   
95 29 Local-gov 115585 Some-college 10   
96 48 Self-emp-not-inc 191277 Doctorate 16   
97 37 Private 202683 Some-college 10   
98 48 Private 171095 Assoc-acdm 12   
99 32 Federal-gov 249409 HS-grad 9   
  
 marital\_status occupation relationship race sex \  
0 Never-married Adm-clerical Not-in-family White Male   
1 Married-civ-spouse Exec-managerial Husband White Male   
2 Divorced Handlers-cleaners Not-in-family White Male   
3 Married-civ-spouse Handlers-cleaners Husband Black Male   
4 Married-civ-spouse Prof-specialty Wife Black Female   
.. ... ... ... ... ...   
95 Never-married Handlers-cleaners Not-in-family White Male   
96 Married-civ-spouse Prof-specialty Husband White Male   
97 Married-civ-spouse Sales Husband White Male   
98 Divorced Exec-managerial Unmarried White Female   
99 Never-married Other-service Own-child Black Male   
  
 capital\_gain capital\_loss hours\_per\_week native\_country income   
0 2174 0 40 United-States <=50K   
1 0 0 13 United-States <=50K   
2 0 0 40 United-States <=50K   
3 0 0 40 United-States <=50K   
4 0 0 40 Cuba <=50K   
.. ... ... ... ... ...   
95 0 0 50 United-States <=50K   
96 0 1902 60 United-States >50K   
97 0 0 48 United-States >50K   
98 0 0 40 England <=50K   
99 0 0 40 United-States <=50K   
  
[100 rows x 15 columns]

titanic\_df

PassengerId Survived Pclass \  
0 1 0 3   
1 2 1 1   
2 3 1 3   
3 4 1 1   
4 5 0 3   
.. ... ... ...   
150 151 0 2   
151 152 1 1   
152 153 0 3   
153 154 0 3   
154 155 0 3   
  
 Name Sex Age SibSp \  
0 Braund, Mr. Owen Harris male 22.0 1   
1 Cumings, Mrs. John Bradley (Florence Briggs Th... female 38.0 1   
2 Heikkinen, Miss. Laina female 26.0 0   
3 Futrelle, Mrs. Jacques Heath (Lily May Peel) female 35.0 1   
4 Allen, Mr. William Henry male 35.0 0   
.. ... ... ... ...   
150 Bateman, Rev. Robert James male 51.0 0   
151 Pears, Mrs. Thomas (Edith Wearne) female 22.0 1   
152 Meo, Mr. Alfonzo male 55.5 0   
153 van Billiard, Mr. Austin Blyler male 40.5 0   
154 Olsen, Mr. Ole Martin male NaN 0   
  
 Parch Ticket Fare Cabin Embarked   
0 0 A/5 21171 7.2500 NaN S   
1 0 PC 17599 71.2833 C85 C   
2 0 STON/O2. 3101282 7.9250 NaN S   
3 0 113803 53.1000 C123 S   
4 0 373450 8.0500 NaN S   
.. ... ... ... ... ...   
150 0 S.O.P. 1166 12.5250 NaN S   
151 0 113776 66.6000 C2 S   
152 0 A.5. 11206 8.0500 NaN S   
153 2 A/5. 851 14.5000 NaN S   
154 0 Fa 265302 7.3125 NaN S   
  
[155 rows x 12 columns]

### Select relevant columns from Adult dataset (mix of nominal and ratio-scaled)

adult\_df = adult\_df[["age", "workclass", "education", "education\_num", "sex"]]  
  
adult\_df

age workclass education education\_num sex  
0 39 State-gov Bachelors 13 Male  
1 50 Self-emp-not-inc Bachelors 13 Male  
2 38 Private HS-grad 9 Male  
3 53 Private 11th 7 Male  
4 28 Private Bachelors 13 Female  
.. ... ... ... ... ...  
95 29 Local-gov Some-college 10 Male  
96 48 Self-emp-not-inc Doctorate 16 Male  
97 37 Private Some-college 10 Male  
98 48 Private Assoc-acdm 12 Female  
99 32 Federal-gov HS-grad 9 Male  
  
[100 rows x 5 columns]

### Encode nominal attributes as integers for processing

label\_encoders = {}  
for column in adult\_df.columns:  
 if adult\_df[column].dtype == object:  
 le = LabelEncoder()  
 adult\_df[column] = le.fit\_transform(adult\_df[column])  
 label\_encoders[column] = le  
  
adult\_df

C:\Users\debat\AppData\Local\Temp\ipykernel\_15644\183426126.py:5: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy  
 adult\_df[column] = le.fit\_transform(adult\_df[column])  
C:\Users\debat\AppData\Local\Temp\ipykernel\_15644\183426126.py:5: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy  
 adult\_df[column] = le.fit\_transform(adult\_df[column])  
C:\Users\debat\AppData\Local\Temp\ipykernel\_15644\183426126.py:5: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy  
 adult\_df[column] = le.fit\_transform(adult\_df[column])

age workclass education education\_num sex  
0 39 6 7 13 1  
1 50 5 7 13 1  
2 38 3 9 9 1  
3 53 3 1 7 1  
4 28 3 7 13 0  
.. ... ... ... ... ...  
95 29 2 12 10 1  
96 48 5 8 16 1  
97 37 3 12 10 1  
98 48 3 5 12 0  
99 32 1 9 9 1  
  
[100 rows x 5 columns]

### Clean and preprocess Titanic dataset

titanic\_df.dropna(inplace=True)  
titanic\_df

PassengerId Survived Pclass \  
1 2 1 1   
3 4 1 1   
6 7 0 1   
10 11 1 3   
11 12 1 1   
21 22 1 2   
23 24 1 1   
27 28 0 1   
52 53 1 1   
54 55 0 1   
62 63 0 1   
66 67 1 2   
75 76 0 3   
88 89 1 1   
92 93 0 1   
96 97 0 1   
97 98 1 1   
102 103 0 1   
110 111 0 1   
118 119 0 1   
123 124 1 2   
124 125 0 1   
136 137 1 1   
137 138 0 1   
139 140 0 1   
148 149 0 2   
151 152 1 1   
  
 Name Sex Age SibSp \  
1 Cumings, Mrs. John Bradley (Florence Briggs Th... female 38.0 1   
3 Futrelle, Mrs. Jacques Heath (Lily May Peel) female 35.0 1   
6 McCarthy, Mr. Timothy J male 54.0 0   
10 Sandstrom, Miss. Marguerite Rut female 4.0 1   
11 Bonnell, Miss. Elizabeth female 58.0 0   
21 Beesley, Mr. Lawrence male 34.0 0   
23 Sloper, Mr. William Thompson male 28.0 0   
27 Fortune, Mr. Charles Alexander male 19.0 3   
52 Harper, Mrs. Henry Sleeper (Myna Haxtun) female 49.0 1   
54 Ostby, Mr. Engelhart Cornelius male 65.0 0   
62 Harris, Mr. Henry Birkhardt male 45.0 1   
66 Nye, Mrs. (Elizabeth Ramell) female 29.0 0   
75 Moen, Mr. Sigurd Hansen male 25.0 0   
88 Fortune, Miss. Mabel Helen female 23.0 3   
92 Chaffee, Mr. Herbert Fuller male 46.0 1   
96 Goldschmidt, Mr. George B male 71.0 0   
97 Greenfield, Mr. William Bertram male 23.0 0   
102 White, Mr. Richard Frasar male 21.0 0   
110 Porter, Mr. Walter Chamberlain male 47.0 0   
118 Baxter, Mr. Quigg Edmond male 24.0 0   
123 Webber, Miss. Susan female 32.5 0   
124 White, Mr. Percival Wayland male 54.0 0   
136 Newsom, Miss. Helen Monypeny female 19.0 0   
137 Futrelle, Mr. Jacques Heath male 37.0 1   
139 Giglio, Mr. Victor male 24.0 0   
148 Navratil, Mr. Michel ("Louis M Hoffman") male 36.5 0   
151 Pears, Mrs. Thomas (Edith Wearne) female 22.0 1   
  
 Parch Ticket Fare Cabin Embarked   
1 0 PC 17599 71.2833 C85 C   
3 0 113803 53.1000 C123 S   
6 0 17463 51.8625 E46 S   
10 1 PP 9549 16.7000 G6 S   
11 0 113783 26.5500 C103 S   
21 0 248698 13.0000 D56 S   
23 0 113788 35.5000 A6 S   
27 2 19950 263.0000 C23 C25 C27 S   
52 0 PC 17572 76.7292 D33 C   
54 1 113509 61.9792 B30 C   
62 0 36973 83.4750 C83 S   
66 0 C.A. 29395 10.5000 F33 S   
75 0 348123 7.6500 F G73 S   
88 2 19950 263.0000 C23 C25 C27 S   
92 0 W.E.P. 5734 61.1750 E31 S   
96 0 PC 17754 34.6542 A5 C   
97 1 PC 17759 63.3583 D10 D12 C   
102 1 35281 77.2875 D26 S   
110 0 110465 52.0000 C110 S   
118 1 PC 17558 247.5208 B58 B60 C   
123 0 27267 13.0000 E101 S   
124 1 35281 77.2875 D26 S   
136 2 11752 26.2833 D47 S   
137 0 113803 53.1000 C123 S   
139 0 PC 17593 79.2000 B86 C   
148 2 230080 26.0000 F2 S   
151 0 113776 66.6000 C2 S

### Select relevant columns from Titanic dataset (mix of nominal and ratio-scaled)

titanic\_df = titanic\_df[["Age", "Sex", "Pclass", "Fare", "Embarked"]]  
titanic\_df

Age Sex Pclass Fare Embarked  
1 38.0 female 1 71.2833 C  
3 35.0 female 1 53.1000 S  
6 54.0 male 1 51.8625 S  
10 4.0 female 3 16.7000 S  
11 58.0 female 1 26.5500 S  
21 34.0 male 2 13.0000 S  
23 28.0 male 1 35.5000 S  
27 19.0 male 1 263.0000 S  
52 49.0 female 1 76.7292 C  
54 65.0 male 1 61.9792 C  
62 45.0 male 1 83.4750 S  
66 29.0 female 2 10.5000 S  
75 25.0 male 3 7.6500 S  
88 23.0 female 1 263.0000 S  
92 46.0 male 1 61.1750 S  
96 71.0 male 1 34.6542 C  
97 23.0 male 1 63.3583 C  
102 21.0 male 1 77.2875 S  
110 47.0 male 1 52.0000 S  
118 24.0 male 1 247.5208 C  
123 32.5 female 2 13.0000 S  
124 54.0 male 1 77.2875 S  
136 19.0 female 1 26.2833 S  
137 37.0 male 1 53.1000 S  
139 24.0 male 1 79.2000 C  
148 36.5 male 2 26.0000 S  
151 22.0 female 1 66.6000 S

### Encode Nominal as Integers for processing

label\_encoders\_titanic = {}  
for column in titanic\_df.columns:  
 if titanic\_df[column].dtype == object:  
 le = LabelEncoder()  
 titanic\_df[column] = le.fit\_transform(titanic\_df[column])  
 label\_encoders[column] = le  
  
titanic\_df

C:\Users\debat\AppData\Local\Temp\ipykernel\_15644\3305425594.py:5: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy  
 titanic\_df[column] = le.fit\_transform(titanic\_df[column])  
C:\Users\debat\AppData\Local\Temp\ipykernel\_15644\3305425594.py:5: SettingWithCopyWarning:   
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead  
  
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy  
 titanic\_df[column] = le.fit\_transform(titanic\_df[column])

Age Sex Pclass Fare Embarked  
1 38.0 0 1 71.2833 0  
3 35.0 0 1 53.1000 1  
6 54.0 1 1 51.8625 1  
10 4.0 0 3 16.7000 1  
11 58.0 0 1 26.5500 1  
21 34.0 1 2 13.0000 1  
23 28.0 1 1 35.5000 1  
27 19.0 1 1 263.0000 1  
52 49.0 0 1 76.7292 0  
54 65.0 1 1 61.9792 0  
62 45.0 1 1 83.4750 1  
66 29.0 0 2 10.5000 1  
75 25.0 1 3 7.6500 1  
88 23.0 0 1 263.0000 1  
92 46.0 1 1 61.1750 1  
96 71.0 1 1 34.6542 0  
97 23.0 1 1 63.3583 0  
102 21.0 1 1 77.2875 1  
110 47.0 1 1 52.0000 1  
118 24.0 1 1 247.5208 0  
123 32.5 0 2 13.0000 1  
124 54.0 1 1 77.2875 1  
136 19.0 0 1 26.2833 1  
137 37.0 1 1 53.1000 1  
139 24.0 1 1 79.2000 0  
148 36.5 1 2 26.0000 1  
151 22.0 0 1 66.6000 1

### Combine the datasets into a list for further processing

# Combine the datasets into a list for further processing  
datasets = {  
 "Adult Dataset": adult\_df,  
 "Titanic Dataset": titanic\_df  
}

### Compute Cosine Similarity

def cosine\_similarity(a, b):  
 """Calculate the Cosine Similarity between two vectors."""  
 try:  
 return 1 - cosine(a, b) # scipy returns distance, so we subtract from 1 to get similarity  
 except Exception as e:  
 return np.nan  
  
# Function to create the Cosine Similarity matrix  
def calculate\_cosine\_similarity\_matrix(dataset):  
 n = len(dataset)  
 cosine\_matrix = np.zeros((n, n))  
   
 for i in range(n):  
 for j in range(n):  
 cosine\_matrix[i, j] = cosine\_similarity(dataset.iloc[i].values, dataset.iloc[j].values)  
   
 return pd.DataFrame(cosine\_matrix)

### Calculate Cosine SImilarity

#### For Adult Dataset

cosine\_matrix\_adult = calculate\_cosine\_similarity\_matrix(adult\_df)  
cosine\_matrix\_adult

0 1 2 3 4 5 6 \  
0 1.000000 0.996120 0.992260 0.967533 0.991062 0.993478 0.969143   
1 0.996120 1.000000 0.995356 0.985101 0.980434 0.987759 0.986363   
2 0.992260 0.995356 1.000000 0.973431 0.980291 0.991643 0.980816   
3 0.967533 0.985101 0.973431 1.000000 0.934177 0.948206 0.997445   
4 0.991062 0.980434 0.980291 0.934177 1.000000 0.996981 0.936376   
.. ... ... ... ... ... ... ...   
95 0.976034 0.968522 0.984786 0.919479 0.983085 0.991265 0.933007   
96 0.998883 0.997556 0.993527 0.971956 0.991562 0.994680 0.972707   
97 0.987173 0.985626 0.996606 0.951403 0.982858 0.993749 0.962654   
98 0.991174 0.998577 0.991648 0.990072 0.974700 0.982658 0.989681   
99 0.987921 0.989193 0.997422 0.959821 0.984550 0.994882 0.968122   
  
 7 8 9 ... 90 91 92 93 \  
0 0.987947 0.986347 0.996862 ... 0.985347 0.986888 0.975612 0.991905   
1 0.996085 0.974868 0.998307 ... 0.996532 0.985455 0.965694 0.988742   
2 0.996382 0.980890 0.995442 ... 0.991265 0.996296 0.981828 0.996523   
3 0.987336 0.923486 0.976317 ... 0.994705 0.951232 0.913116 0.953271   
4 0.965545 0.997933 0.988638 ... 0.963205 0.983161 0.983478 0.988883   
.. ... ... ... ... ... ... ... ...   
95 0.967070 0.991735 0.976569 ... 0.954591 0.994996 0.998900 0.994023   
96 0.989034 0.986548 0.999319 ... 0.989125 0.987346 0.974227 0.991879   
97 0.986846 0.987841 0.988844 ... 0.977234 0.999692 0.993583 0.998922   
98 0.994338 0.967490 0.996608 ... 0.998355 0.979346 0.955547 0.982578   
99 0.988367 0.987459 0.993277 ... 0.983371 0.997737 0.988346 0.997355   
  
 94 95 96 97 98 99   
0 0.994913 0.976034 0.998883 0.987173 0.991174 0.987921   
1 0.991649 0.968522 0.997556 0.985626 0.998577 0.989193   
2 0.990753 0.984786 0.993527 0.996606 0.991648 0.997422   
3 0.958745 0.919479 0.971956 0.951403 0.990072 0.959821   
4 0.995978 0.983085 0.991562 0.982858 0.974700 0.984550   
.. ... ... ... ... ... ...   
95 0.982860 1.000000 0.976136 0.995441 0.959385 0.992139   
96 0.997575 0.976136 1.000000 0.987524 0.994687 0.990804   
97 0.988614 0.995441 0.987524 1.000000 0.979044 0.998150   
98 0.988623 0.959385 0.994687 0.979044 1.000000 0.985352   
99 0.993014 0.992139 0.990804 0.998150 0.985352 1.000000   
  
[100 rows x 100 columns]

#### For TItanic Dataset

cosine\_matrix\_titanic = calculate\_cosine\_similarity\_matrix(titanic\_df)  
cosine\_matrix\_titanic

0 1 2 3 4 5 6 \  
0 1.000000 0.995550 0.950379 0.953797 0.794964 0.753466 0.983664   
1 0.995550 1.000000 0.975197 0.927993 0.847999 0.811493 0.996103   
2 0.950379 0.975197 1.000000 0.830554 0.944041 0.920198 0.990421   
3 0.953797 0.927993 0.830554 1.000000 0.609816 0.565279 0.897217   
4 0.794964 0.847999 0.944041 0.609816 1.000000 0.996740 0.889719   
5 0.753466 0.811493 0.920198 0.565279 0.996740 1.000000 0.858714   
6 0.983664 0.996103 0.990421 0.897217 0.889719 0.858714 1.000000   
7 0.914004 0.872311 0.742794 0.971280 0.480649 0.422964 0.827375   
8 0.996915 0.999761 0.971814 0.931194 0.840094 0.802464 0.994507   
9 0.949365 0.974291 0.999899 0.827611 0.945111 0.921172 0.989670   
10 0.999876 0.996019 0.951987 0.952835 0.797846 0.756918 0.984815   
11 0.741537 0.800941 0.912633 0.552546 0.995315 0.999406 0.848895   
12 0.703764 0.766632 0.887558 0.516672 0.985513 0.995827 0.818987   
13 0.920035 0.879602 0.752778 0.973587 0.493846 0.436473 0.835675   
14 0.987848 0.997989 0.987099 0.904854 0.879097 0.846107 0.999611   
15 0.809779 0.860649 0.951917 0.627451 0.999486 0.994687 0.900684   
16 0.989890 0.972398 0.897178 0.979511 0.701412 0.654025 0.949109   
17 0.974767 0.949966 0.857635 0.985645 0.640159 0.589457 0.920013   
18 0.969902 0.988348 0.997505 0.866148 0.918423 0.890551 0.997659   
19 0.923689 0.883987 0.758953 0.974731 0.501963 0.444933 0.840815   
20 0.763595 0.820538 0.925919 0.578173 0.997874 0.999505 0.866089   
21 0.992686 0.999551 0.980901 0.917484 0.861941 0.826884 0.998122   
22 0.990109 0.998842 0.983530 0.915496 0.869931 0.836496 0.998803   
23 0.992710 0.999548 0.980661 0.918877 0.861284 0.826546 0.998180   
24 0.980876 0.958475 0.872046 0.983677 0.661958 0.612193 0.930736   
25 0.894276 0.932092 0.988775 0.749304 0.981410 0.968018 0.959776   
26 0.985364 0.965439 0.883920 0.983066 0.680559 0.631707 0.939538   
  
 7 8 9 ... 17 18 19 20 \  
0 0.914004 0.996915 0.949365 ... 0.974767 0.969902 0.923689 0.763595   
1 0.872311 0.999761 0.974291 ... 0.949966 0.988348 0.883987 0.820538   
2 0.742794 0.971814 0.999899 ... 0.857635 0.997505 0.758953 0.925919   
3 0.971280 0.931194 0.827611 ... 0.985645 0.866148 0.974731 0.578173   
4 0.480649 0.840094 0.945111 ... 0.640159 0.918423 0.501963 0.997874   
5 0.422964 0.802464 0.921172 ... 0.589457 0.890551 0.444933 0.999505   
6 0.827375 0.994507 0.989670 ... 0.920013 0.997659 0.840815 0.866089   
7 1.000000 0.879364 0.740421 ... 0.981288 0.788166 0.999692 0.436753   
8 0.879364 1.000000 0.971077 ... 0.954297 0.985949 0.890788 0.811660   
9 0.740421 0.971077 1.000000 ... 0.855681 0.997135 0.756698 0.926885   
10 0.912095 0.997160 0.950829 ... 0.973872 0.971209 0.921846 0.766711   
11 0.406600 0.791583 0.913638 ... 0.574645 0.881853 0.428725 0.999417   
12 0.358592 0.756428 0.888446 ... 0.531106 0.853892 0.381081 0.994150   
13 0.999879 0.886464 0.750455 ... 0.984037 0.797324 0.999940 0.450268   
14 0.840397 0.996910 0.986426 ... 0.928908 0.995935 0.853398 0.854003   
15 0.502247 0.853312 0.953092 ... 0.658975 0.927929 0.523320 0.995829   
16 0.962005 0.975764 0.895647 ... 0.996483 0.926090 0.968415 0.665266   
17 0.981288 0.954297 0.855681 ... 1.000000 0.891790 0.985667 0.601552   
18 0.788166 0.985949 0.997135 ... 0.891790 1.000000 0.802980 0.897225   
19 0.999692 0.890788 0.756698 ... 0.985667 0.802980 1.000000 0.458551   
20 0.436753 0.811660 0.926885 ... 0.601552 0.897225 0.458551 1.000000   
21 0.858832 0.999026 0.980147 ... 0.941250 0.992170 0.871109 0.835285   
22 0.849936 0.997672 0.982543 ... 0.935341 0.993696 0.862468 0.845044   
23 0.859375 0.998942 0.979817 ... 0.941682 0.992030 0.871606 0.834844   
24 0.975362 0.962606 0.870352 ... 0.999504 0.904389 0.980484 0.624022   
25 0.636737 0.926405 0.988988 ... 0.773225 0.976086 0.655360 0.971460   
26 0.969558 0.968991 0.882138 ... 0.998476 0.914699 0.975217 0.643654   
  
 21 22 23 24 25 26   
0 0.992686 0.990109 0.992710 0.980876 0.894276 0.985364   
1 0.999551 0.998842 0.999548 0.958475 0.932092 0.965439   
2 0.980901 0.983530 0.980661 0.872046 0.988775 0.883920   
3 0.917484 0.915496 0.918877 0.983677 0.749304 0.983066   
4 0.861941 0.869931 0.861284 0.661958 0.981410 0.680559   
5 0.826884 0.836496 0.826546 0.612193 0.968018 0.631707   
6 0.998122 0.998803 0.998180 0.930736 0.959776 0.939538   
7 0.858832 0.849936 0.859375 0.975362 0.636737 0.969558   
8 0.999026 0.997672 0.998942 0.962606 0.926405 0.968991   
9 0.980147 0.982543 0.979817 0.870352 0.988988 0.882138   
10 0.993348 0.990909 0.993459 0.979977 0.896648 0.984597   
11 0.816352 0.826794 0.815936 0.597622 0.963001 0.617824   
12 0.783153 0.794927 0.783146 0.554669 0.946423 0.575226   
13 0.866438 0.857785 0.866945 0.978540 0.648211 0.973147   
14 0.999382 0.999423 0.999338 0.939137 0.952714 0.947365   
15 0.874245 0.881368 0.873584 0.680496 0.985656 0.698297   
16 0.965910 0.961020 0.966199 0.998536 0.822869 0.999363   
17 0.941250 0.935341 0.941682 0.999504 0.773225 0.998476   
18 0.992170 0.993696 0.992030 0.904389 0.976086 0.914699   
19 0.871109 0.862468 0.871606 0.980484 0.655360 0.975217   
20 0.835285 0.845044 0.834844 0.624022 0.971460 0.643654   
21 1.000000 0.999406 0.999964 0.950564 0.941469 0.957961   
22 0.999406 1.000000 0.999494 0.944830 0.947106 0.953060   
23 0.999964 0.999494 1.000000 0.950895 0.941296 0.958281   
24 0.950564 0.944830 0.950895 1.000000 0.790968 0.999515   
25 0.941469 0.947106 0.941296 0.790968 1.000000 0.805963   
26 0.957961 0.953060 0.958281 0.999515 0.805963 1.000000   
  
[27 rows x 27 columns]

### Explanation

Cosine Similarity Calculation: This metric focuses on the orientation of the data points in a multi-dimensional space. A value of 1 means that the vectors are identical, while 0 indicates orthogonal vectors (no similarity), and -1 indicates opposite vectors.

Handling Different Data Types: Cosine similarity works best with interval and ratio-scaled data, as it considers the direction (or relative distribution) rather than the magnitude of the data points.

### Observation and Analysis

The resulting matrices will show the similarity between data points based on the angle between their corresponding vectors. This is especially useful in text mining and other applications where the magnitude of features is less important than their relative importance.

Cosine Similarity is not affected by the scale of the data, making it useful when the attributes have been normalized or when dealing with sparse data.