```
!pip install gensim

# Loading pre-trained embeddings
import gensim.downloader as api

# Handling numerical data
import numpy as np
import pandas as pd

# Visualization
import matplotlib.pyplot as plt

# Dimensionality reduction
from sklearn.manifold import TSNE
```

```
Collecting gensim
  Downloading gensim-4.4.0-cp312-cp312-manylinux_2_24_x86_64.manylinux_2_28_x86_64.whl.metadata (8
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.12/dist-packages (from gens
Requirement already satisfied: scipy>=1.7.0 in /usr/local/lib/python3.12/dist-packages (from gensi
Requirement already satisfied: smart_open>=1.8.1 in /usr/local/lib/python3.12/dist-packages (from
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-packages (from smart_open>=
Downloading gensim-4.4.0-cp312-cp312-manylinux_2_24_x86_64.manylinux_2_28_x86_64.whl (27.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 27.9/27.9 MB 59.5 MB/s eta 0:00:00
Installing collected packages: gensim
Successfully installed gensim-4.4.0
```

```
# Load pre-trained GloVe model (100D)
model = api.load("glove-wiki-gigaword-100")

# Print vocabulary size
print("Vocabulary size:", len(model))

# Display example vector
print("\nVector for word 'king':")
print(model['king'])
```

```
[==================================================] 100.0% 128.1/128.1MB downloaded
Vocabulary size: 400000

Vector for word 'king':
[-0.32307  -0.87616   0.21977   0.25268   0.22976   0.7388   -0.37954
 -0.35307  -0.84369  -1.1113   -0.30266   0.33178  -0.25113   0.30448
 -0.077491 -0.89815   0.092496 -1.1407   -0.58324   0.66869  -0.23122
 -0.95855   0.28262  -0.078848  0.75315   0.26584   0.3422   -0.33949
  0.95608   0.065641  0.45747   0.39835   0.57965   0.39267  -0.21851
  0.58795  -0.55999   0.63368  -0.043983 -0.68731  -0.37841   0.38026
  0.61641  -0.88269  -0.12346  -0.37928  -0.38318   0.23868   0.6685
 -0.43321  -0.11065   0.081723  1.1569    0.78958  -0.21223  -2.3211
 -0.67806   0.44561   0.65707   0.1045    0.46217   0.19912   0.25802
  0.057194  0.53443  -0.43133  -0.34311   0.59789  -0.58417   0.068995
  0.23944  -0.85181   0.30379  -0.34177  -0.25746  -0.031101 -0.16285
  0.45169  -0.91627   0.64521   0.73281  -0.22752   0.30226   0.044801
 -0.83741   0.55006  -0.52506  -1.7357    0.4751   -0.70487   0.056939
 -0.7132    0.089623  0.41394  -1.3363   -0.61915  -0.33089  -0.52881
  0.16483  -0.98878 ]
```

```
words = [
    # Animals
    "cat", "dog", "lion", "tiger", "elephant", "horse", "cow", "monkey",
```

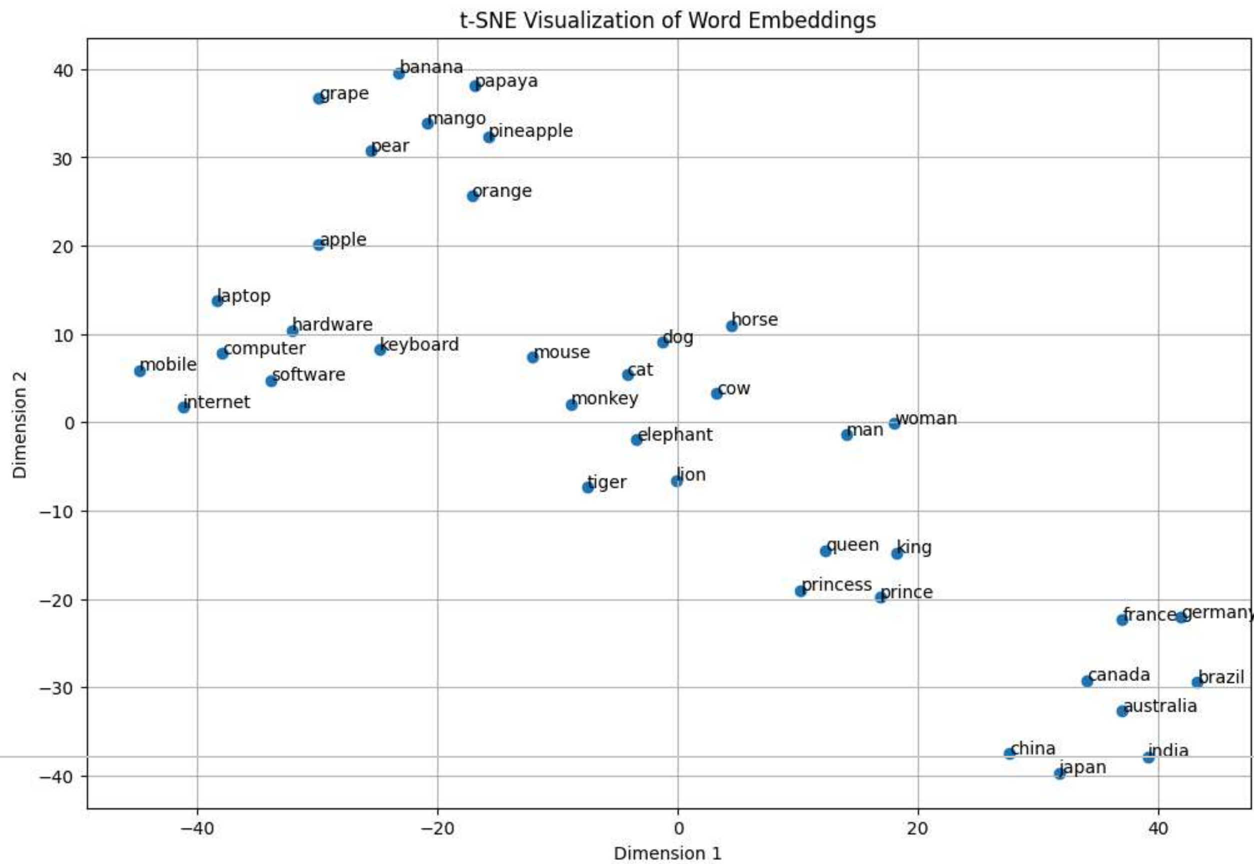
```
    # Countries
    "india", "china", "france", "germany", "japan", "brazil", "canada", "australia",

    # Fruits
    "apple", "banana", "orange", "grape", "mango", "pineapple", "papaya", "pear",

    # Technology
    "computer", "laptop", "keyboard", "mouse", "mobile", "internet", "software", "hardware",

    # Royalty / People
    "king", "queen", "man", "woman", "prince", "princess"
]
```

```
word_vectors = []

valid_words = []

for word in words:
    if word in model:
        word_vectors.append(model[word])
        valid_words.append(word)

word_vectors = np.array(word_vectors)
```

```
tsne = TSNE(n_components=2, random_state=42, perplexity=10)

tsne_results = tsne.fit_transform(word_vectors)

print("Shape after reduction:", tsne_results.shape)
```

```
Shape after reduction: (38, 2)
```

```
plt.figure(figsize=(12, 8))

x = tsne_results[:, 0]
y = tsne_results[:, 1]

plt.scatter(x, y)

for i, word in enumerate(valid_words):
    plt.annotate(word, (x[i], y[i]))

plt.title("t-SNE Visualization of Word Embeddings")
plt.xlabel("Dimension 1")
plt.ylabel("Dimension 2")

plt.grid(True)
plt.show()
```


t-SNE Visualization of Word Embeddings
banana
papaya
grape
mango
pineapple
pear
orange
apple
laptop
hardware
keyboard
computer
mobile
software
internet
mouse
dog
horse
cat
cow
monkey
elephant
man
woman
tiger
lion
queen
king
princess
prince
france
germany
canada
brazil
australia
china
india
japan
Dimension 1
Dimension 2
40
30
20
10
0
−10
−20
−30
−40
−40
−20
0
20
40