STEP 1 — Import Libraries

```
!pip install gensim

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

# Handling arrays and matrices
import numpy as np

# Data handling (optional but useful)
import pandas as pd

# Visualization
import matplotlib.pyplot as plt

# t-SNE for 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 78.6 MB/s eta 0:00:00
Installing collected packages: gensim
Successfully installed gensim-4.4.0
```

STEP 2 — Load Pre-trained Embedding Model

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

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

# Display one example vector
example_word = "king"
print(f"\nVector for '{example_word}':\n", model[example_word])
print("Vector shape:", model[example_word].shape)
```

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

Vector for '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 ]
Vector shape: (100,)
```

STEP 3 — Select Word List

```
word_list = [
    # Royalty
    "king", "queen", "prince", "princess", "monarch",
    "throne", "royal", "crown", "duke", "palace",

    # Vehicles
    "car", "truck", "bus", "train", "bicycle",
    "motorcycle", "airplane", "ship", "boat", "taxi",

    # Fruits
    "apple", "banana", "orange", "mango", "grape",
    "pineapple", "pear", "peach", "cherry", "lemon",

    # Animals
    "dog", "cat", "lion", "tiger", "elephant",
    "horse", "wolf", "fox", "monkey", "zebra"
]
```

STEP 4 — Extract Word Vectors

```
# Extract vectors for selected words
vectors = []

for word in word_list:
    if word in model:
        vectors.append(model[word])
    else:
        print(f"{word} not found in vocabulary.")

vectors = np.array(vectors)

print("Shape of vector matrix:", vectors.shape)
```

```
Shape of vector matrix: (40, 100)
```

STEP 5 — Apply t-SNE (Dimensionality Reduction)

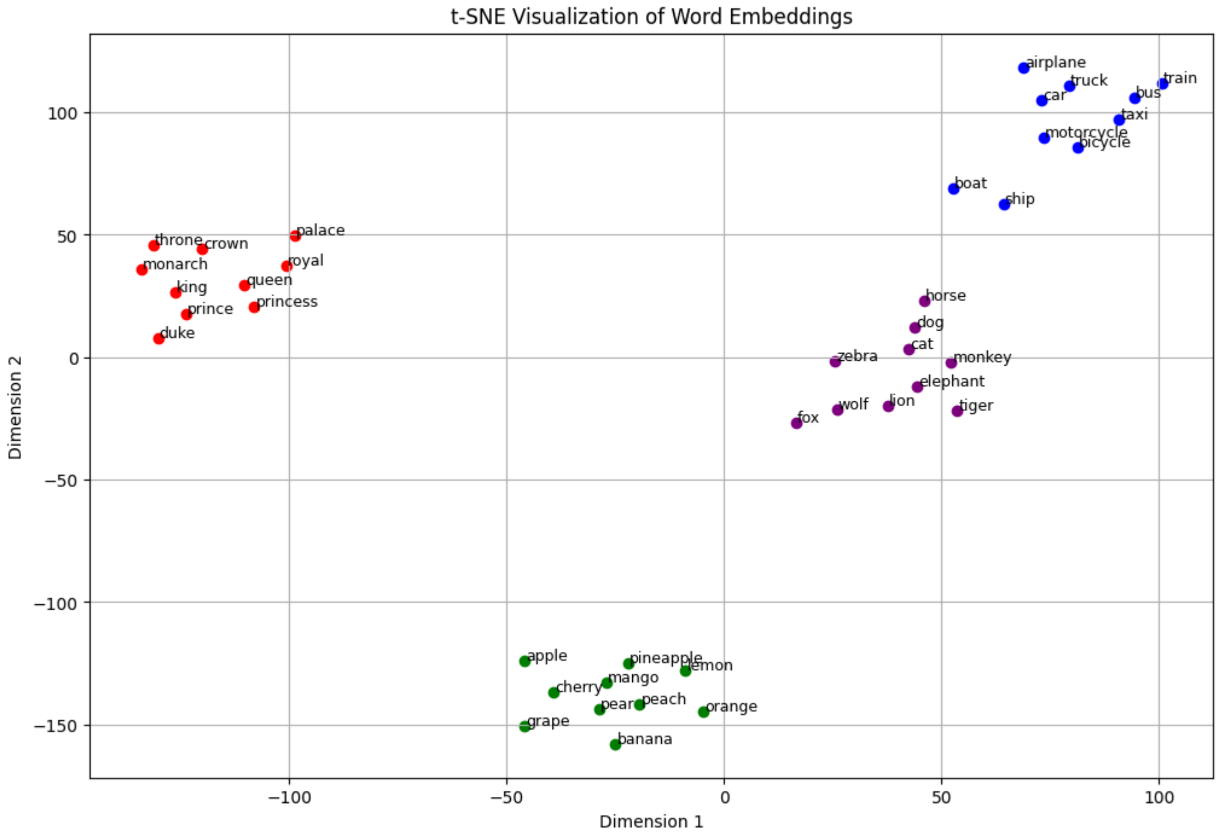
```
# Initialize t-SNE
tsne = TSNE(n_components=2, random_state=42, perplexity=5)

# Transform vectors
reduced_vectors = tsne.fit_transform(vectors)

print("Shape after t-SNE:", reduced_vectors.shape)
```

```
Shape after t-SNE: (40, 2)
```

STEP 6 — Plot Visualization

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

# Assign colors per category
colors = (
    ["red"] * 10 +      # Royalty
    ["blue"] * 10 +     # Vehicles
    ["green"] * 10 +    # Fruits
```

```
        ["purple"] * 10     # Animals
    )

    # Scatter plot
    for i, word in enumerate(word_list):
        x = reduced_vectors[i, 0]
        y = reduced_vectors[i, 1]

        plt.scatter(x, y, color=colors[i])
        plt.text(x + 0.3, y + 0.3, word, fontsize=9)

    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
Dimension 1
Dimension 2
airplane
truck
train
car
bus
taxi
motorcycle
bicycle
boat
ship
palace
throne
crown
monarch
royal
queen
king
prince
princess
duke
horse
dog
cat
zebra
monkey
elephant
wolf
lion
fox
tiger
apple
pineapple
lemon
mango
cherry
peach
pear
orange
grape
banana