**#Python Script for Principle Component Analysis**

from gensim.models import Word2Vec

from sklearn.decomposition import PCA

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

import spacy

nlp = spacy.load("en\_core\_web\_sm")

data = df["clean\_tokens2"]

model = Word2Vec(data,

vector\_size = 100,

window = 5,

min\_count = 3,

workers = 3,

sg = 0,

hs = 1,

negative = 0,

epochs = 50)

word\_vectors = model.wv

words = word\_vectors.index\_to\_key

vectors = word\_vectors.vectors

# Define the range of clusters to try

max\_clusters = 7

# Reduce dimensionality using PCA

pca = PCA(n\_components=3, random\_state=50)

reduced\_vectors = pca.fit\_transform(vectors)

# Initialize lists to store inertia and distortion values

inertia = []

distortions = []

# Determine the optimal number of clusters using K-means

for k in range(1, max\_clusters + 1):

kmeans = KMeans(n\_clusters=k, random\_state=50)

kmeans.fit(reduced\_vectors)

inertia.append(kmeans.inertia\_)

distortion = kmeans.inertia\_ / reduced\_vectors.shape[0]

distortions.append(distortion)

# Set the font family and size

font = {'family': 'serif', 'weight': 'normal', 'size': 14}

plt.rc('font', \*\*font)

# Plot the elbow curve to determine the optimal number of clusters

plt.plot(range(1, max\_clusters + 1), inertia, marker='o')

plt.xlabel('Number of Clusters')

plt.ylabel('Inertia')

plt.gca().spines['top'].set\_visible(False)

plt.gca().spines['right'].set\_visible(False)

plt.savefig('twitter\_elbow\_curve\_2.png')

plt.show()

# Use the optimal number of clusters for dimension reduction with PCA

optimal\_k = 3

pca = PCA(n\_components=optimal\_k, random\_state=50)

reduced\_vectors = pca.fit\_transform(vectors)

cluster\_labels = kmeans.fit\_predict(reduced\_vectors)

#PCA Scatter plot

plt.figure(figsize=(10, 10))

plt.scatter(reduced\_vectors[:, 0], reduced\_vectors[:, 1], c=cluster\_labels, cmap='tab10', marker='.')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.gca().spines['top'].set\_visible(False)

plt.gca().spines['right'].set\_visible(False)

plt.show()