```
import findspark

findspark.init()


from pyspark.sql import SparkSession

from pyspark.ml.feature import VectorAssembler, StandardScaler, PCA

from pyspark.ml.clustering import KMeans

import matplotlib.pyplot as plt


# Initialize Spark Session

spark = SparkSession.builder.appName("PCA_with_KMeans").getOrCreate()


# Load Dataset

df = spark.read.csv("/content/segmentation data.csv", header=True, inferSchema=True)


# Rename columns to remove special characters

for col_name in df.columns:

   cleaned_name = col_name.replace(".", "_").replace(" ", "_")

   df = df.withColumnRenamed(col_name, cleaned_name)


# Assemble features into a single column

feature_columns = df.columns[:-1]  # Exclude the label column if present

vector_assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")

df = vector_assembler.transform(df).select("features")


# Standardize data

scaler = StandardScaler(inputCol="features", outputCol="scaled_features", withStd=True,
withMean=True)

scaler_model = scaler.fit(df)

df = scaler_model.transform(df).select("scaled_features")


# Apply PCA
```

```
pca = PCA(k=2, inputCol="scaled_features", outputCol="pca_features")

pca_model = pca.fit(df)

df = pca_model.transform(df).select("pca_features")


# Function to run KMeans and visualize

def kmeans_with_pca():

    kmeans = KMeans(featuresCol="pca_features", k=3, seed=42)

    model = kmeans.fit(df)

    predictions = model.transform(df)


    # Convert Spark DataFrame to Pandas for visualization

    pandas_df = predictions.select("pca_features", "prediction").toPandas()


    # Extract PCA components

    pandas_df["PCA1"] = pandas_df["pca_features"].apply(lambda x: x[0])

    pandas_df["PCA2"] = pandas_df["pca_features"].apply(lambda x: x[1])


    # Scatter plot with clusters

    plt.figure(figsize=(8, 6))

    scatter = plt.scatter(pandas_df["PCA1"], pandas_df["PCA2"], c=pandas_df["prediction"],
cmap="viridis", alpha=0.7)

    plt.colorbar(scatter, label="Cluster")

    plt.xlabel("PCA Component 1")

    plt.ylabel("PCA Component 2")

    plt.title("PCA with KMeans Clustering (PySpark)")

    plt.show()


# Menu-Driven Interface

while True:

    print("\nMenu:")

    print("1. PCA with KMeans")
    print("2. Exit")
    choice = input("Enter your choice (1/2): ")


    if choice == '1':
       kmeans_with_pca()
    elif choice == '2':
       print("Exiting the program. Goodbye!")
       spark.stop()
       break
    else:
       print("Invalid choice. Please try again.")
```