AML Lab Assignment-1:

Name: Gurvinder Kaur
PRN: 1032230432
Roll No.: PA14

AIM:

Demonstrate Feature Engineering concepts using dimensionality reduction methods (PCA/t-SNE), Feature Selection

THEORY:

Feature engineering is the process of selecting, manipulating, and transforming raw data into features that can be used in supervised learning. It can produce new features for both supervised and unsupervised learning, with the goal of simplifying and speeding up data transformations while also enhancing model accuracy.

Dimensionality reduction is the process of reducing the number of features or variables in a dataset while preserving its important underlying structure or patterns.
PCA (Principal Component Analysis) is a statistical method used for reducing the dimensionality of data by transforming it into a new set of variables, called principal components, that capture the most important information while minimizing information loss.
t-SNE (t-distributed Stochastic Neighbor Embedding) is a nonlinear dimensionality reduction technique that maps high-dimensional data to a lower-dimensional space, emphasizing the local structure and clustering of data points in the new representation

Feature selection involves choosing the most relevant and informative features from a dataset while excluding irrelevant or redundant ones, aiming to improve model performance and reduce computational complexity.

CODE EXECUTION & OUTPUT:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from sklearn.impute import SimpleImputer
import numpy as np
from sklearn.manifold import TSNE
```

```
data = pd.read_csv('/content/heart.csv')
```

```
data.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

```
data.tail()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1025 non-null   int64
 1   sex       1025 non-null   int64
 2   cp        1025 non-null   int64
 3   trestbps  1025 non-null   int64
 4   chol      1025 non-null   int64
 5   fbs       1025 non-null   int64
 6   restecg   1025 non-null   int64
 7   thalach   1025 non-null   int64
 8   exang     1025 non-null   int64
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64
 11  ca        1025 non-null   int64
 12  thal      1025 non-null   int64
 13  target    1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

```
data.shape
```

```
(1025, 14)
```

```
data.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	tha
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	1025.0
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	149.1
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.0
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.0
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.0
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.0
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.0
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	202.0

```
data.isnull().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

```
data.nunique()
```

```
age          41
sex           2
cp            4
trestbps     49
chol        152
fbs           2
restecg       3
thalach      91
exang         2
oldpeak      40
slope         3
ca            5
thal          4
target        2
dtype: int64
```

```
data['ca'].unique()
```

```
array([2, 0, 1, 3, 4])
```

```
(data.isnull().sum()/(len(data)))*100
```

```
age         0.0
sex         0.0
cp          0.0
trestbps    0.0
chol        0.0
fbs         0.0
restecg     0.0
thalach     0.0
exang       0.0
oldpeak     0.0
slope       0.0
ca          0.0
thal        0.0
target      0.0
dtype: float64
```

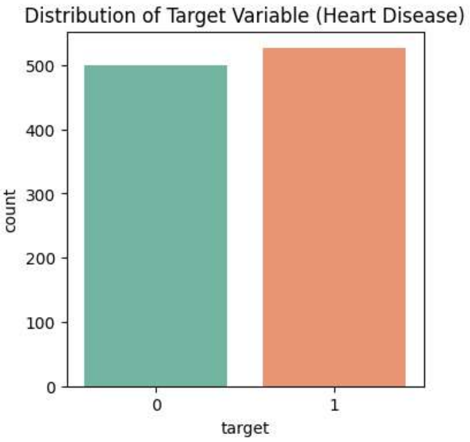
```
data.columns
```

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

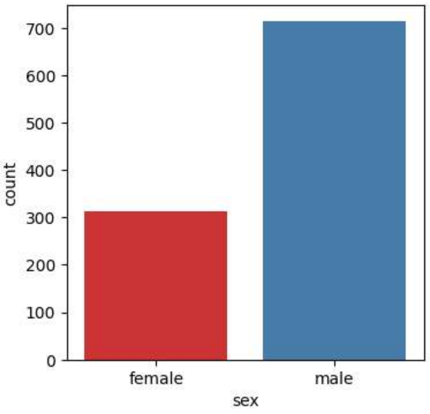
```
data.target.value_counts()
```

```
1    526
0    499
Name: target, dtype: int64
```

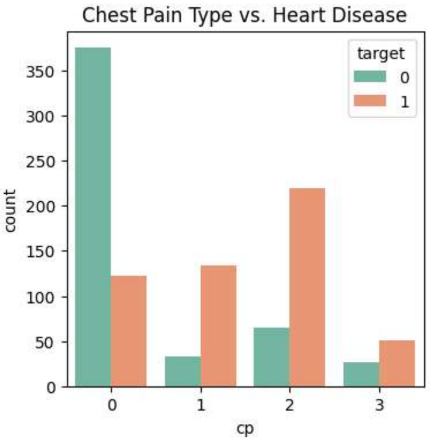
```
plt.figure(figsize=(4, 4))
sns.countplot(data=data, x='target', palette='Set2')
plt.title('Distribution of Target Variable (Heart Disease)')
plt.show()
```



```
plt.figure(figsize=(4, 4))
sns.countplot(x='sex', data=data, palette='Set1')
plt.xticks(ticks=[0, 1], labels = ["female", "male"])
plt.show()
```

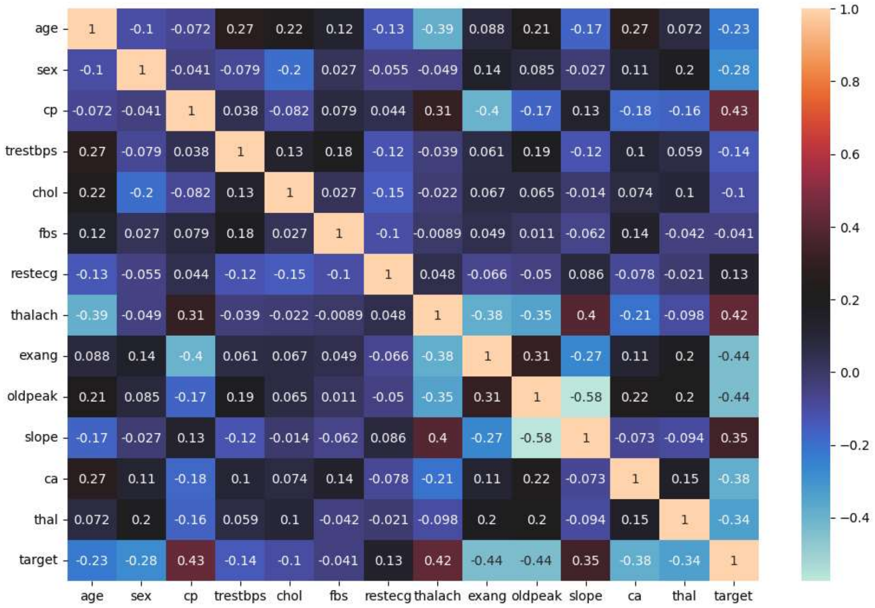


```
plt.figure(figsize=(4, 4))
sns.countplot(data=data, x='cp', hue='target', palette='Set2')
plt.title('Chest Pain Type vs. Heart Disease')
plt.show()
```



```
plt.figure(figsize=(12,8))
sns.heatmap(data.corr(), annot=True, cbar=True, cmap='icefire')
```

```
<Axes: >
```

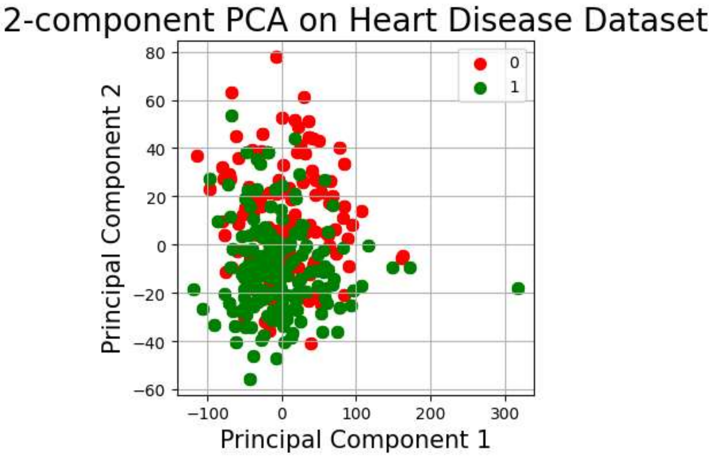


	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1	-0.1	-0.072	0.27	0.22	0.12	-0.13	-0.39	0.088	0.21	-0.17	0.27	0.072	-0.23
sex	-0.1	1	-0.041	-0.079	-0.2	0.027	-0.055	-0.049	0.14	0.085	-0.027	0.11	0.2	-0.28
cp	-0.072	-0.041	1	0.038	-0.082	0.079	0.044	0.31	-0.4	-0.17	0.13	-0.18	-0.16	0.43
trestbps	0.27	-0.079	0.038	1	0.13	0.18	-0.12	-0.039	0.061	0.19	-0.12	0.1	0.059	-0.14
chol	0.22	-0.2	-0.082	0.13	1	0.027	-0.15	-0.022	0.067	0.065	-0.014	0.074	0.1	-0.1
fbs	0.12	0.027	0.079	0.18	0.027	1	-0.1	-0.0089	0.049	0.011	-0.062	0.14	-0.042	-0.041
restecg	-0.13	-0.055	0.044	-0.12	-0.15	-0.1	1	0.048	-0.066	-0.05	0.086	-0.078	-0.021	0.13
thalach	-0.39	-0.049	0.31	-0.039	-0.022	-0.0089	0.048	1	-0.38	-0.35	0.4	-0.21	-0.098	0.42
exang	0.088	0.14	-0.4	0.061	0.067	0.049	-0.066	-0.38	1	0.31	-0.27	0.11	0.2	-0.44
oldpeak	0.21	0.085	-0.17	0.19	0.065	0.011	-0.05	-0.35	0.31	1	-0.58	0.22	0.2	-0.44
slope	-0.17	-0.027	0.13	-0.12	-0.014	-0.062	0.086	0.4	-0.27	-0.58	1	-0.073	-0.094	0.35
ca	0.27	0.11	-0.18	0.1	0.074	0.14	-0.078	-0.21	0.11	0.22	-0.073	1	0.15	-0.38
thal	0.072	0.2	-0.16	0.059	0.1	-0.042	-0.021	-0.098	0.2	0.2	-0.094	0.15	1	-0.34
target	-0.23	-0.28	0.43	-0.14	-0.1	-0.041	0.13	0.42	-0.44	-0.44	0.35	-0.38	-0.34	1

```
X = data.iloc[:,0:13].values
y = data[['target']]
```

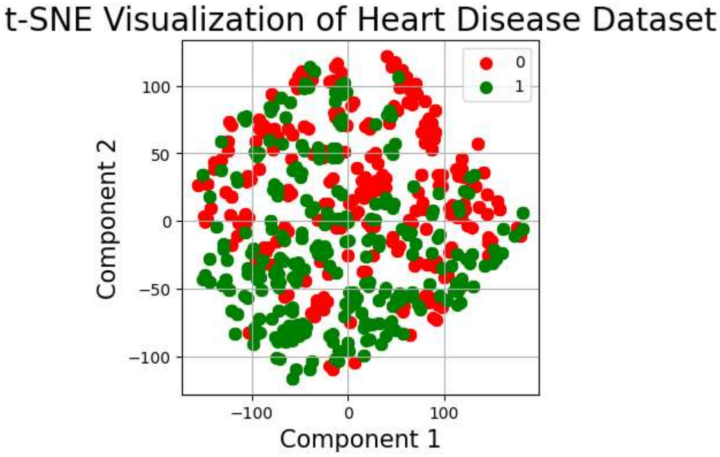
PCA

```
pca = PCA(n_components=2)
principal_components = pca.fit_transform(X)

principal_df = pd.DataFrame(data=principal_components, columns=['PC1', 'PC2'])
final_df = pd.concat([principal_df, y], axis=1)

# Plotting
fig = plt.figure(figsize=(4, 4))
ax = fig.add_subplot(1, 1, 1)
ax.set_xlabel('Principal Component 1', fontsize=15)
ax.set_ylabel('Principal Component 2', fontsize=15)
ax.set_title('2-component PCA on Heart Disease Dataset', fontsize=20)

targets = [0, 1]
colors = ['red', 'green']

for target, color in zip(targets, colors):
    indices_to_keep = final_df['target'] == target
    ax.scatter(final_df.loc[indices_to_keep, 'PC1'],
               final_df.loc[indices_to_keep, 'PC2'],
               c=color,
               s=50)

ax.legend(targets)
ax.grid()

plt.show()
```



t-SNE

```
tsne = TSNE(n_components=2, random_state=42)
tsne_components = tsne.fit_transform(X)

tsne_df = pd.DataFrame(data=tsne_components, columns=['Component 1', 'Component 2'])
final_df = pd.concat([tsne_df, y], axis=1)

# Plotting
fig = plt.figure(figsize=(4, 4))
ax = fig.add_subplot(1, 1, 1)
ax.set_xlabel('Component 1', fontsize=15)
ax.set_ylabel('Component 2', fontsize=15)
ax.set_title('t-SNE Visualization of Heart Disease Dataset', fontsize=20)

targets = [0, 1]
colors = ['red', 'green']

for target, color in zip(targets, colors):
    indices_to_keep = final_df['target'] == target
    ax.scatter(final_df.loc[indices_to_keep, 'Component 1'],
               final_df.loc[indices_to_keep, 'Component 2'],
               c=color,
               s=50)

ax.legend(targets)
ax.grid()

plt.show()
```


t-SNE Visualization of Heart Disease Dataset
Component 2
Component 1