PCA On Wine data

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [ ]:
```
# laod he data
data = pd.read_csv("wine.data")
data
```

Out[ ]:

	1	14.23	1.71	2.43	15.6	127	2.8	3.06	.28	2.29	5.64	1.04	3.92	1065
0	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
1	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
2	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
3	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735
4	1	14.20	1.76	2.45	15.2	112	3.27	3.39	0.34	1.97	6.75	1.05	2.85	1450
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
172	3	13.71	5.65	2.45	20.5	95	1.68	0.61	0.52	1.06	7.70	0.64	1.74	740
173	3	13.40	3.91	2.48	23.0	102	1.80	0.75	0.43	1.41	7.30	0.70	1.56	750
174	3	13.27	4.28	2.26	20.0	120	1.59	0.69	0.43	1.35	10.20	0.59	1.56	835
175	3	13.17	2.59	2.37	20.0	120	1.65	0.68	0.53	1.46	9.30	0.60	1.62	840
176	3	14.13	4.10	2.74	24.5	96	2.05	0.76	0.56	1.35	9.20	0.61	1.60	560

177 rows × 14 columns

In [ ]:
```
## data frame

df = pd.DataFrame(data)
df.head(3)
```

Out[ ]:

	1	14.23	1.71	2.43	15.6	127	2.8	3.06	.28	2.29	5.64	1.04	3.92	1065
0	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
1	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
2	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 177 entries, 0 to 176
Data columns (total 14 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   1       177 non-null    int64
 1   14.23   177 non-null    float64
 2   1.71    177 non-null    float64
 3   2.43    177 non-null    float64
 4   15.6    177 non-null    float64
 5   127     177 non-null    int64
 6   2.8     177 non-null    float64
 7   3.06    177 non-null    float64
 8   .28     177 non-null    float64
 9   2.29    177 non-null    float64
 10  5.64    177 non-null    float64
 11  1.04    177 non-null    float64
 12  3.92    177 non-null    float64
 13  1065    177 non-null    int64
dtypes: float64(11), int64(3)
memory usage: 19.5 KB
```

In [ ]:
```
df.columns
```

Out[ ]:
```
Index(['1', '14.23', '1.71', '2.43', '15.6', '127', '2.8', '3.06', '.28',
       '2.29', '5.64', '1.04', '3.92', '1065'],
      dtype='object')
```

In [ ]:
```
# check the null value

df.isnull().sum()
```

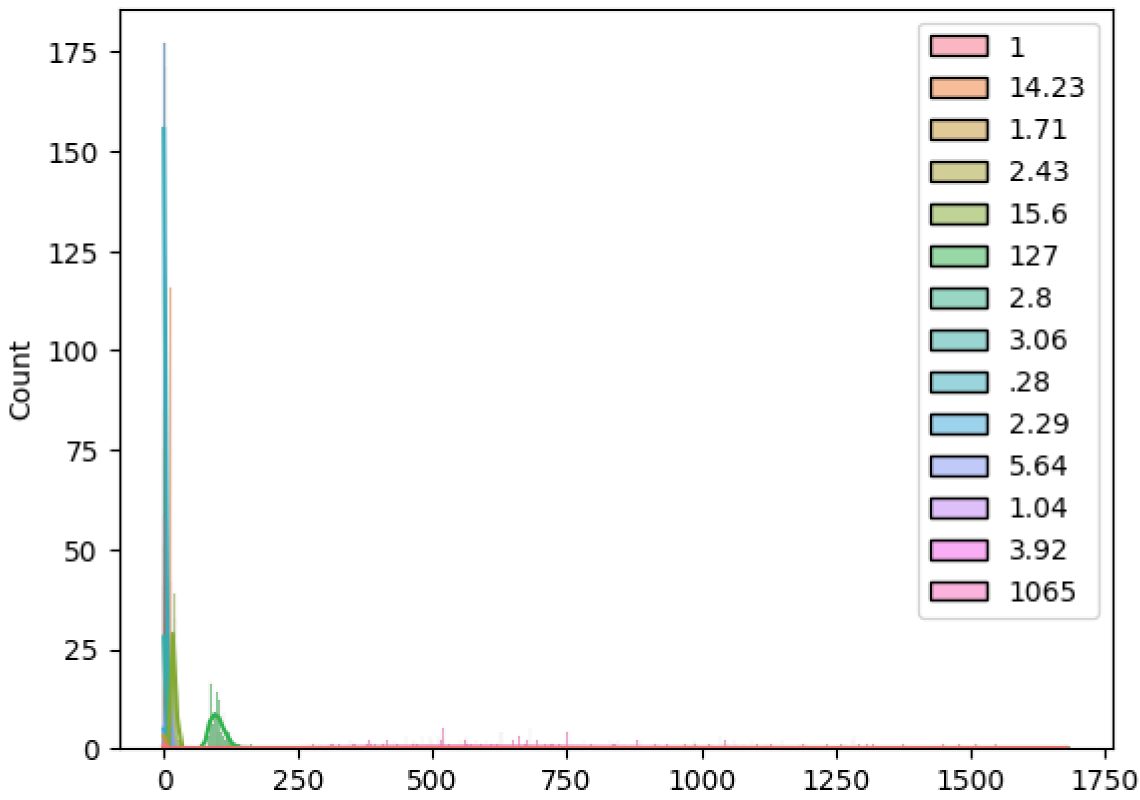
Out[ ]:
```
1        0
14.23    0
1.71     0
2.43     0
15.6     0
127      0
2.8      0
3.06     0
.28      0
2.29     0
5.64     0
1.04     0
3.92     0
1065     0
dtype: int64
```

Insights:

- There is no missing values in the dataset

In [ ]:
```
## check data distribution

sns.histplot(data  = df, kde = True)
```

Out[ ]:
```
<Axes: ylabel='Count'>
```



Checking the correlation between the features

In [ ]:

```
## correlation

df.corr()
```

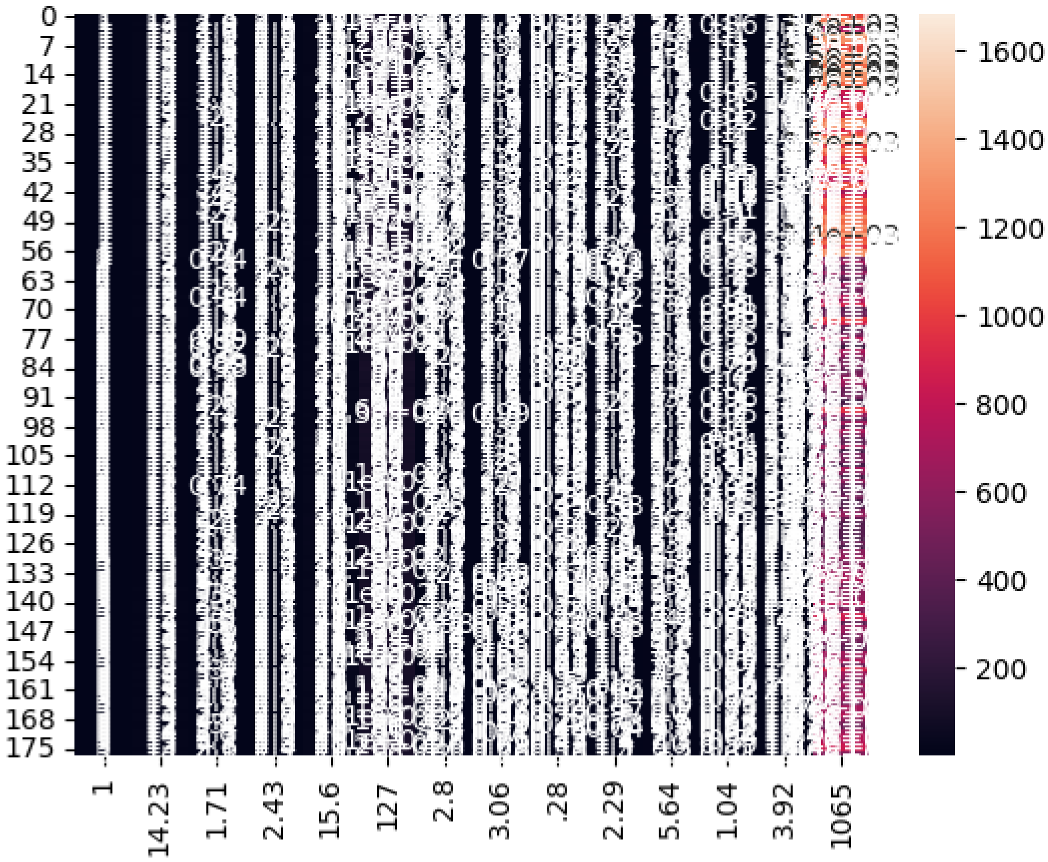
Out[ ]:

	1	14.23	1.71	2.43	15.6	127	2.8	
1	1.000000	-0.321238	0.436127	-0.048260	0.513963	-0.198944	-0.717933	-0.846
14.23	-0.321238	1.000000	0.099963	0.210964	-0.303350	0.258742	0.284543	0.230
1.71	0.436127	0.099963	1.000000	0.164955	0.286148	-0.049049	-0.333512	-0.409
2.43	-0.048260	0.210964	0.164955	1.000000	0.446698	0.287107	0.128176	0.114
15.6	0.513963	-0.303350	0.286148	0.446698	1.000000	-0.071707	-0.317583	-0.346
127	-0.198944	0.258742	-0.049049	0.287107	-0.071707	1.000000	0.208200	0.187
2.8	-0.717933	0.284543	-0.333512	0.128176	-0.317583	0.208200	1.000000	0.864
3.06	-0.846485	0.230133	-0.409324	0.114084	-0.346922	0.187101	0.864046	1.000
.28	0.487215	-0.151445	0.291501	0.187354	0.359395	-0.252091	-0.448301	-0.536
2.29	-0.494887	0.127561	-0.217975	0.008082	-0.190779	0.226504	0.610533	0.650
5.64	0.268562	0.547883	0.250053	0.258643	0.020478	0.199337	-0.056401	-0.174
1.04	-0.617690	-0.075375	-0.560854	-0.075181	-0.272719	0.052042	0.432987	0.543
3.92	-0.786428	0.057417	-0.366720	0.001503	-0.268186	0.046961	0.699566	0.786
1065	-0.631227	0.641068	-0.189512	0.222979	-0.436858	0.387542	0.495839	0.491

In [ ]:

```
sns.heatmap(df, annot=True)
```

Out[ ]: <Axes: >



Splitting data into the dependent and Independent features

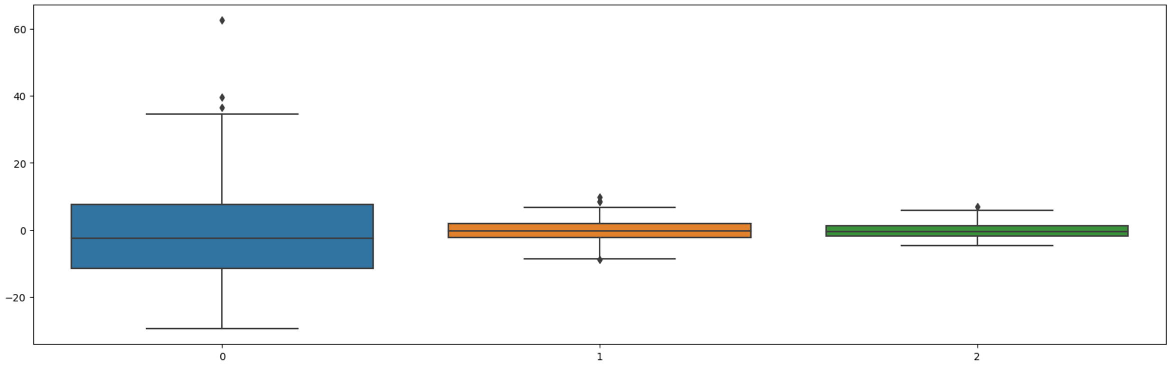
In [ ]:
```
# Select the features columns
X = df.iloc[:, :-1]

# Select the target column
y = df.iloc[:, -1]
```

In [ ]:
```
X.shape ,y.shape
```

Out[ ]: ((177, 13), (177,))

In [ ]:
```
## box plot of the outliers check
ax,fig = plt.subplots(figsize=(20,6))
sns.boxplot(X)
```

Out[ ]: <Axes: >



Insights:

- There are 13 Independent features. which is impossible to visulaize so

We will try to reduce demesnions from 13 to 3 components

Train - test split

In [ ]:
```
from sklearn.model_selection import train_test_split
```

In [ ]:
```
X_train , X_test, y_train,y_test = train_test_split(X,y ,test_size = 0.33,random
```

In [ ]:
```
X_train.head()
```

Out[ ]:

	1	14.23	1.71	2.43	15.6	127	2.8	3.06	.28	2.29	5.64	1.04	3.92
22	1	12.85	1.60	2.52	17.8	95	2.48	2.37	0.26	1.46	3.93	1.090	3.63
145	3	13.88	5.04	2.23	20.0	80	0.98	0.34	0.40	0.68	4.90	0.580	1.33
97	2	12.37	1.07	2.10	18.5	88	3.52	3.75	0.24	1.95	4.50	1.040	2.77
69	2	12.29	1.61	2.21	20.4	103	1.10	1.02	0.37	1.46	3.05	0.906	1.82
166	3	12.82	3.37	2.30	19.5	88	1.48	0.66	0.40	0.97	10.26	0.720	1.75

Implementing PCA on the preprocessed dataset using the scikit-learn library.

In [ ]:

```
# import PCA from the sklaern.decomposition
from sklearn.decomposition import PCA
```

In [ ]:

```
## create an instance for the pca

## we will reduce its dimension from 13 to 3

pca = PCA(n_components=3)
pca
```

Out[ ]:

```
PCA
PCA(n_components=3)
```

Fit and Transform th data

In [ ]:

```
X_train = pca.fit_transform(X_train)
X_train
```

```
Out[ ]: array([[-4.37850260e+00, -2.09176393e+00, -1.24721610e+00],
       [-1.93897793e+01,  1.16768724e+00,  1.75345283e+00],
       [-1.13390624e+01, -1.47279677e+00, -8.16320248e-01],
       [ 3.54093238e+00,  8.72162591e-01, -2.06188432e+00],
       [-1.12107463e+01,  8.22532512e-01,  5.76490989e+00],
       [-1.02968518e+01, -3.78134036e+00,  9.27814950e-01],
       [ 2.16518931e+01, -2.16304798e+00, -6.38524043e-01],
       [ 7.65614828e+00, -5.71001076e-01, -1.11798331e+00],
       [ 1.26931905e+01, -1.86879468e-01,  3.25678101e+00],
       [ 1.27427621e+01, -4.55573249e+00,  1.37390879e+00],
       [ 3.26582744e+01, -3.42389832e-01, -1.05171083e+00],
       [ 3.44960606e+01,  2.00207123e+00, -3.70233510e+00],
       [-3.39218962e+00,  5.28882554e-02,  1.07934161e+00],
       [-1.33631556e+01, -3.66348038e+00,  9.94038337e-02],
       [-4.32743598e+00, -3.01771182e+00,  1.91183559e-01],
       [-1.33005944e+01,  2.72044197e+00,  3.45324308e+00],
       [-1.74111165e+01,  1.27108765e+00, -1.89711519e+00],
       [ 6.42172561e-01, -8.49109322e+00, -5.46601996e-02],
       [-1.43649272e+01, -1.70366751e+00,  1.19302214e+00],
       [-1.33725731e+01,  1.57080810e+00, -1.40166263e+00],
       [-1.36164964e+00, -1.60201940e+00, -6.66023526e-01],
       [ 1.16656139e+01, -7.06608487e-01,  1.31151535e-01],
       [-1.44160689e+01, -3.78474289e+00, -1.59066322e+00],
       [-3.33490356e+00, -3.86362574e+00, -1.22816414e-01],
       [-9.36057471e+00, -8.72033913e-01, -2.39669329e-01],
       [-7.43541476e+00,  8.57804128e+00, -3.11337074e+00],
       [-6.33021673e+00, -3.73595510e+00, -5.58653575e-02],
       [-3.38201674e+00, -3.64597719e-02, -1.83962308e+00],
       [-1.13621714e+01,  5.07085981e-01,  1.76898073e+00],
       [-2.14557828e+01,  4.05012778e+00, -2.98195044e+00],
       [-1.34412724e+01, -9.27325789e-01, -2.55347073e+00],
       [-4.43737065e-01, -1.83717561e+00, -2.50998606e+00],
       [ 1.77095757e+01, -3.62563263e+00,  3.30325144e-01],
       [-1.44169362e+00,  1.24500837e+00, -2.48024852e+00],
       [-1.14583255e+01,  6.09582355e+00, -3.20697222e+00],
       [ 1.96871179e+01,  8.41433816e+00, -1.86389076e+00],
       [-4.29769015e+00,  1.85416341e+00,  3.75834613e+00],
       [ 1.37495709e+01,  2.75351361e+00,  4.11651065e+00],
       [ 3.72164473e+00, -1.01536086e+00,  8.10077380e-01],
       [ 6.25204044e+01, -7.57859985e-01, -4.47964890e+00],
       [-2.41652966e+00,  4.27203198e+00, -3.66558383e-01],
       [ 1.17997474e+01,  4.25989667e+00,  5.16725845e+00],
       [-5.39891904e+00, -1.70400652e+00, -8.85254803e-01],
       [ 1.57778785e+00,  1.07962443e+00, -2.60110337e+00],
       [ 1.06272395e+01, -8.49316306e-01, -1.33246349e+00],
       [-1.44291008e+01,  1.13258876e+00, -2.30719253e+00],
       [ 6.59906801e+00,  2.42594896e+00,  1.63683973e-01],
       [ 1.60010791e+00,  4.74285212e+00, -2.75127762e-01],
       [-1.23485165e+01, -9.23926079e-01, -8.19296406e-01],
       [ 2.07412550e+01,  1.17790023e+00,  3.76010160e+00],
       [-1.14303321e+01,  1.16575860e+00, -2.30782557e+00],
       [-1.34752171e+01,  9.40612792e-01, -3.09965225e+00],
       [-1.94108293e+01, -7.22409849e-01, -1.24272444e+00],
       [ 1.86859556e+01, -3.35794048e+00,  1.72056844e-01],
       [-3.30892341e+00, -5.18063866e+00,  6.59095688e-01],
       [-1.12972773e+01,  2.11925569e+00,  3.15952063e+00],
       [ 1.86271501e+01,  1.17558532e+00, -1.64820640e+00],
       [-1.13312514e+01,  3.57638089e+00,  2.46564749e+00],
       [-7.23603656e+00, -7.23503295e+00,  2.52593282e+00],
       [ 1.06437158e+01, -4.23319343e+00, -6.05570581e-01],
```

```
[-1.33813115e+01, -9.73040119e-01, -9.39933971e-01],
[ 3.65561641e+01, -4.77045347e+00, -2.41825747e+00],
[ 6.74490036e-01,  5.27283578e-01, -5.88474246e-01],
[ 1.72435413e+00, -3.87700196e+00,  8.63592641e-01],
[-2.29906060e+00, -5.84276939e+00,  7.21273541e-01],
[ 7.66563084e+00, -2.86591241e+00, -4.89881027e-01],
[-1.33751018e+01,  4.84243345e+00, -2.16755547e+00],
[-1.04448240e+01,  2.09944788e+00, -1.07183692e-02],
[-1.14587615e+01,  1.17194618e+00, -2.81772141e+00],
[-1.94262197e+01,  1.01158831e+00, -1.47310999e+00],
[-5.42118594e+00, -1.81706973e+00, -2.16031272e+00],
[-1.02342584e+01,  7.34207229e-01,  4.91604584e+00],
[ 7.63491262e+00,  9.81683879e-01,  1.23937839e+00],
[-1.84532036e+01, -1.13075526e+00, -1.91730978e+00],
[-1.23995999e+00, -3.81185767e+00,  1.90501221e+00],
[ 2.79227319e+00, -7.70079557e+00,  2.73184155e+00],
[ 1.55310269e+00, -3.46297915e+00, -1.36461222e+00],
[-9.22449940e+00,  3.09204402e+00,  4.81656260e+00],
[-1.37274540e+00,  4.77559757e+00,  5.08014750e-01],
[ 8.81840251e+00, -3.24343399e+00,  2.99564206e+00],
[ 3.95688784e+01,  9.80106908e+00, -4.62240472e+00],
[-6.29098705e+00,  3.28347902e+00,  3.63057807e+00],
[-5.32370645e+00, -4.46345823e-01,  2.60189933e+00],
[-7.26458969e+00,  2.57350034e+00,  3.75324488e+00],
[ 6.60563156e+00, -4.48530855e-01,  8.46487438e-01],
[ 4.54915654e+00, -6.35237839e-01, -2.27833986e+00],
[ 1.67067271e+01,  6.39022246e-01,  5.02647275e-02],
[-7.43586584e+00, -8.48596046e-01, -2.36855415e+00],
[-5.29352680e+00, -2.68011920e+00,  5.07506829e-01],
[ 2.07671568e+01,  1.43896661e+00,  4.99065059e+00],
[-1.14888228e+01, -8.82151271e+00, -1.47361775e+00],
[ 8.81423667e+00, -2.29465283e+00,  3.07629681e+00],
[-2.94651543e+01,  3.59428748e+00, -3.14591488e+00],
[ 1.63017121e+00, -3.36453509e+00, -1.31836012e+00],
[ 8.74759159e+00, -3.15577980e+00,  1.14319618e+00],
[-8.20948652e+00,  1.09292294e+00,  5.14253224e+00],
[ 2.25639564e+01, -1.26678894e+00, -1.14688218e+00],
[-1.38512596e+00, -4.33266889e+00, -1.12345648e+00],
[-1.07148981e+00,  5.89197052e+00,  6.94167724e+00],
[ 5.73836381e+00,  5.43021358e+00,  3.06541333e+00],
[ 1.70333387e+00, -1.22842770e+00,  2.05346836e-01],
[ 1.57078865e+01, -2.61063810e+00,  6.69595642e-01],
[ 2.37240687e+01,  5.01634364e+00,  2.64625505e+00],
[ 4.61605562e+00,  9.35391976e-01,  6.92800150e-01],
[ 1.66822763e+01,  6.20826546e+00,  9.29551882e-01],
[-1.44280128e+01,  6.49971906e-01, -2.03717899e+00],
[-2.42882503e+00, -2.15168003e+00, -1.63421848e+00],
[ 8.52371180e+00,  2.59836945e+00, -3.74701686e+00],
[-1.54465722e+01,  1.90438548e+00, -2.17522630e+00],
[-2.42792927e+00, -3.40567355e+00, -8.49927635e-01],
[ 2.52589974e+00,  6.79606034e+00, -3.46454808e+00],
[-3.23188385e+00,  5.60890022e+00,  4.29744153e+00],
[ 2.63542438e+00, -1.01328856e+00, -5.27093226e-01],
[-1.23981983e+01,  4.14882290e+00, -1.48033814e+00],
[-1.54184571e+01,  2.75303251e+00, -1.63433710e+00],
[ 1.27418503e+01, -2.45119957e+00,  1.61285233e+00],
[-1.14494359e+01, -1.72537348e+00, -2.35658851e+00],
[-1.34685083e+01, -2.58138910e-01, -2.66622805e+00]])
```

In [ ]:
```
X_test = pca.fit_transform(X_test)
X_test
```

```
Out[ ]: array([[ 2.61762218e+01,  2.35912575e+00,  2.46353452e-02],
               [ 2.21291244e+00,  3.67532090e+00,  2.62609056e-01],
               [-4.20471191e+00, -1.82492612e+00, -5.89642831e-02],
               [ 6.08222390e+00,  3.48242021e-01,  1.62849914e+00],
               [ 9.98945213e+00,  1.84684486e+00, -1.46721494e+00],
               [ 1.50047947e+01, -9.83216676e-01,  9.72207288e-01],
               [-4.13216174e+00,  1.22665403e-02, -1.93281594e+00],
               [ 1.76534069e+00, -4.37993431e+00,  2.16788982e+00],
               [ 6.85311088e+00, -2.36368850e-01, -2.63868665e+00],
               [-1.10891895e+01, -4.86925833e-01,  1.35043826e+00],
               [ 2.35797871e+01, -6.17849459e+00, -3.04038243e+00],
               [ 5.74175814e+00, -3.49432334e+00,  3.57058384e-01],
               [ 1.29376180e+00,  4.81233270e+00,  8.64730389e-01],
               [-1.31462868e+01,  1.31929240e+00, -3.16273459e+00],
               [ 1.99831775e+01, -1.28164763e+00,  4.82424214e-01],
               [ 2.83096140e+00, -8.45636061e-01, -1.42774986e+00],
               [-1.62471588e+01, -1.61810232e+00, -1.89249102e+00],
               [-1.52143215e+01, -2.34655856e+00,  1.99831183e-01],
               [ 5.12559796e+00,  1.63942146e+00,  7.08570739e-01],
               [-1.52488464e+01, -7.25237834e-01, -2.99139580e+00],
               [ 1.61978940e+01,  3.44454005e+00, -4.89607949e-02],
               [-2.20155380e+01,  1.31466444e+00,  4.16593550e-01],
               [ 8.57955815e-02,  2.22486524e+00,  1.30030493e+00],
               [ 1.16693168e+01, -6.99668304e+00,  2.63781728e+00],
               [-1.39334329e+01, -3.77149290e-01,  5.65433354e+00],
               [ 1.19054798e+01, -4.23598958e+00,  5.16636245e+00],
               [-1.99052528e+01,  4.17847114e+00, -1.63726958e-01],
               [-2.16860614e+01,  6.01909003e+00,  1.60038814e+00],
               [-1.42358393e+01, -1.66394541e+00, -1.12933405e+00],
               [ 1.82045960e+01,  2.42906043e+00,  1.09091176e+00],
               [ 8.82166295e-01, -2.40085522e+00,  2.87553268e-01],
               [-1.23866394e+01, -3.66569812e+00, -2.97837225e+00],
               [ 1.75651814e+00, -2.19614298e+00, -7.82828903e-01],
               [ 4.07966367e+00,  2.42467869e+00, -9.48881926e-01],
               [-8.45816891e+00,  8.90251299e+00,  1.84578484e+00],
               [ 3.05609154e+00,  1.87729693e+00, -5.57490247e-01],
               [-3.45110585e+00, -7.89880207e+00,  1.91716295e+00],
               [-4.43813867e+00, -5.97587778e+00, -2.39640153e-01],
               [-1.20109618e+01,  2.50755884e+00, -2.17842751e+00],
               [-1.09813618e+01, -3.12658681e+00,  6.96809885e+00],
               [ 2.82894648e+01,  4.64386355e+00, -2.12904204e-01],
               [-6.12754038e+00,  3.41523420e-01, -2.39951571e+00],
               [-1.03175250e+01, -2.46039966e+00, -2.45439836e+00],
               [ 5.08672145e+01, -3.20244851e-01, -3.75342936e+00],
               [-8.13579898e+00, -6.91925188e-01,  8.29301295e-02],
               [ 1.32584579e+01,  2.26157881e+00,  2.54596024e+00],
               [-1.20734156e+01,  1.75057965e+00, -1.79396150e+00],
               [-1.42058742e+01, -1.04017864e+00, -2.10383941e+00],
               [ 2.92922944e+00, -1.47234421e+00,  2.70704982e+00],
               [ 1.13823339e+01,  5.09298745e+00,  2.15552065e+00],
               [ 1.18629339e+01,  4.98809232e-01, -2.56431043e+00],
               [ 2.18230457e+00,  2.77885909e+00,  1.15714127e+00],
               [-5.90138876e+00,  3.05155027e+00, -4.27408993e-01],
               [-1.86767036e+00, -1.21897467e-01,  5.92248060e+00],
               [-2.02587644e+01, -1.38361102e+00, -1.92278169e+00],
               [-1.02605750e+01, -1.71384410e+00, -2.20239325e+00],
               [ 6.78117731e-01, -3.16443025e+00, -2.91906020e+00],
               [-1.95743244e+00,  1.83812063e+00, -4.42512827e-01],
               [-2.03551630e+00,  1.71452026e+00, -1.63867387e+00]])
```

Determining the optimal number of principal components to retain based on the explained variance ratio.

In [ ]:
```
## components

pca.components_
```

Out[ ]:
```
array([[ 1.,  0.,  0.],
       [-0.,  1., -0.],
       [-0.,  0.,  1.]])
```

In [ ]:
```
## pricipal componets based on explained_variances_ratio

pca.explained_variance_ratio_
```

Out[ ]:
```
array([0.91901   , 0.05444818, 0.02654182])
```

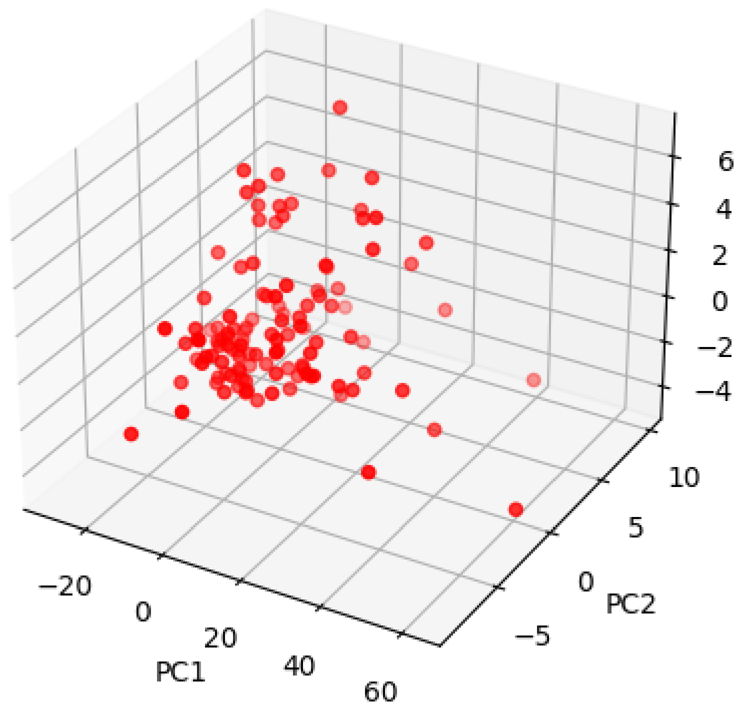
Insights:

- First components capturing the ~ 91 % variance in the data.
- while other two have no significant variance capture.

Visualizing PCA using the 3d Scatter plot

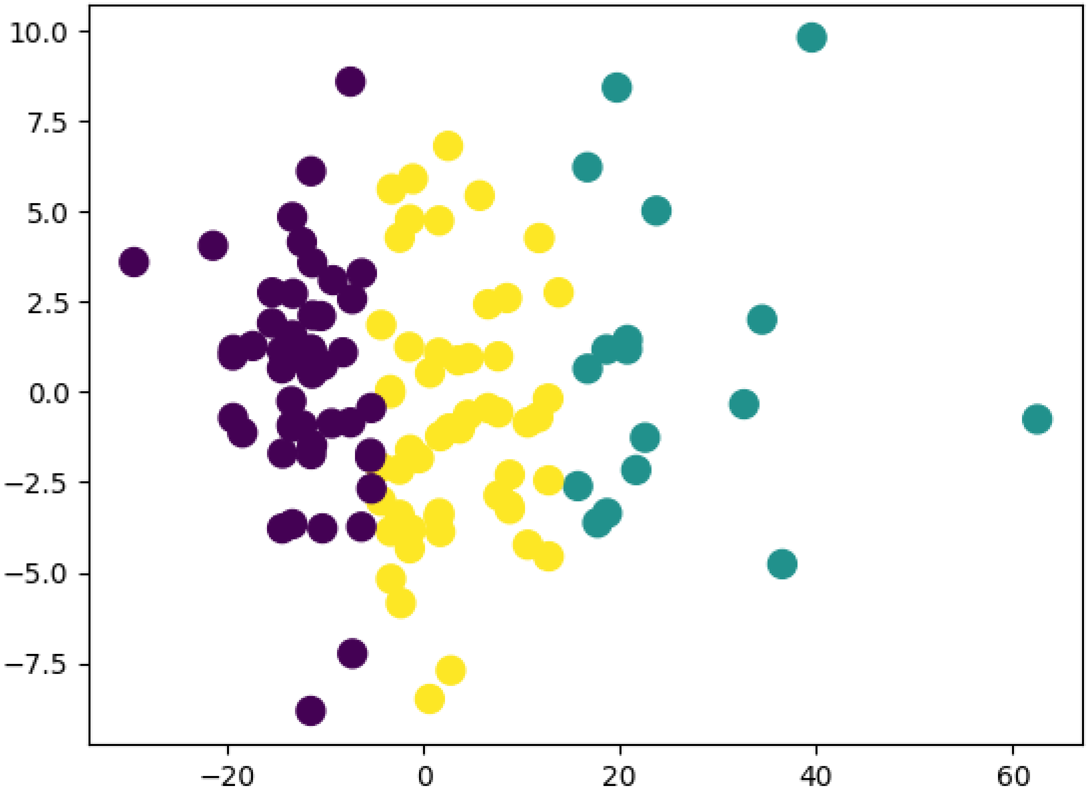
In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D



# Create a PCA object.
pca = PCA(n_components=3)

# Fit the PCA object to the training data.
pca.fit(X_train)

# Transform the training data using the PCA object.
X_train_pca = pca.transform(X_train)

# Create a 3D figure.
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

# Plot the transformed training data.
ax.scatter(X_train_pca[:, 0], X_train_pca[:, 1], X_train_pca[:, 2], c='red')

# Set the axes labels.
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
```

```python
# Show the plot.
plt.show()
```



Performing clustering on the PCA-transformed data using K-Means clustering algorithm.

In [ ]:
```python
import numpy as np
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore",)

# Create a K-Means object.
kmeans = KMeans(n_clusters=3)

# Fit the K-Means object to the PCA-transformed data.
kmeans.fit(X_train_pca)

# Predict the cluster labels for the PCA-transformed data.
y_train_pred = kmeans.predict(X_train_pca)

# Plot the PCA-transformed data with the cluster labels.
plt.scatter(X_train_pca[:, 0], X_train_pca[:, 1], c=y_train_pred, s=100)
plt.show()
```



Report on the above Analysis by applying PCA and K-means Clustering

Insights after applying K-mean clustering algo:-

The figure shows a scatter plot of the PCA-transformed data, with the points colored according to their cluster label. The points are labeled as follows:

Cluster 0: These points are located in the lower left corner of the plot. They are likely to be small, light, and slow.

Cluster 1: These points are located in the upper right corner of the plot. They are likely to be large, dark, and fast.

Cluster 2: These points are located in the middle of the plot. They are likely to be medium-sized, medium-dark, and medium-fast . It is important to note that these are just general interpretations. The actual meaning of the clusters will depend on the specific data that you are clustering.

Here are some additional things to keep in mind when interpreting the results of clustering:

The number of clusters that you choose will affect the results of the clustering. If you choose too few clusters, then some of the data points may be misclassified. If you choose too many clusters, then the clusters may not be meaningful. The clustering algorithm that you choose will also affect the results of the clustering. Different algorithms will group data points together in different ways. It is important to evaluate the results of the clustering to make sure that they make sense. You can do this by looking at the cluster labels, the cluster centroids, and the silhouette plots.

Insights after applying PCA

- Firstly after applying PCA , 13 features reduced in 3 featutres.
- Secondly after applying PCA , The first two components capturing the 92% of the vraince in the data.

Major advantage of PCA:

- Reduces dimensionality:

  PCA can be used to reduce the dimensionality of a dataset without losing too much information. This can be useful for visualization, as it can make it easier to see the relationships between the data points. PCA can also be used to improve the performance of machine learning algorithms, as it can make the data easier to fit.

Hence in this case it reduce 13 to 3 components

- Identifies the most important features:

  PCA can be used to identify the most important features in a dataset. This can be useful for feature selection, as it can help you to focus on the features that are most relevant to your task.

Provides a better understanding of the data: PCA can be used to provide a better understanding of the data. This is because PCA can help you to see the relationships between the data points and to identify the most important features.