Homework 2

STA 307

Malcolm Kahora, Vejay, Rohan, Tyler

February 22, 2024

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from ISLP import load_data
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

AUTO = load_data("Auto")
```

Question 1

```
selected_coloumns = ['mpg','cylinders', 'displacement',  'weight', 'acceleration', ]
X = AUTO[selected_coloumns]
y = AUTO['origin'] # Origin of car (1. American, 2. European, 3. Japanese)
print(X.head)

print("American cars: {0}".format((y == 1).sum())) # American
print("European cars: {0}".format((y == 2).sum())) # European
print("Japanese cars: {0}".format((y == 3).sum())) # Japanese
```

```
<bound method NDFrame.head of       mpg  cylinders  displacement  weight  acceleration
0    18.0          8         307.0    3504          12.0
1    15.0          8         350.0    3693          11.5
2    18.0          8         318.0    3436          11.0
3    16.0          8         304.0    3433          12.0
4    17.0          8         302.0    3449          10.5
..    ...        ...           ...     ...           ...
387  27.0          4         140.0    2790          15.6
388  44.0          4          97.0    2130          24.6
389  32.0          4         135.0    2295          11.6
390  28.0          4         120.0    2625          18.6
391  31.0          4         119.0    2720          19.4

[392 rows x 5 columns]>
American cars: 245
European cars: 68
```

```
Japanese cars: 79
```

Question 2

```
print("Standard deviation")
print(X.std())
print("\n")
print("Mean")
print(X.mean())
```

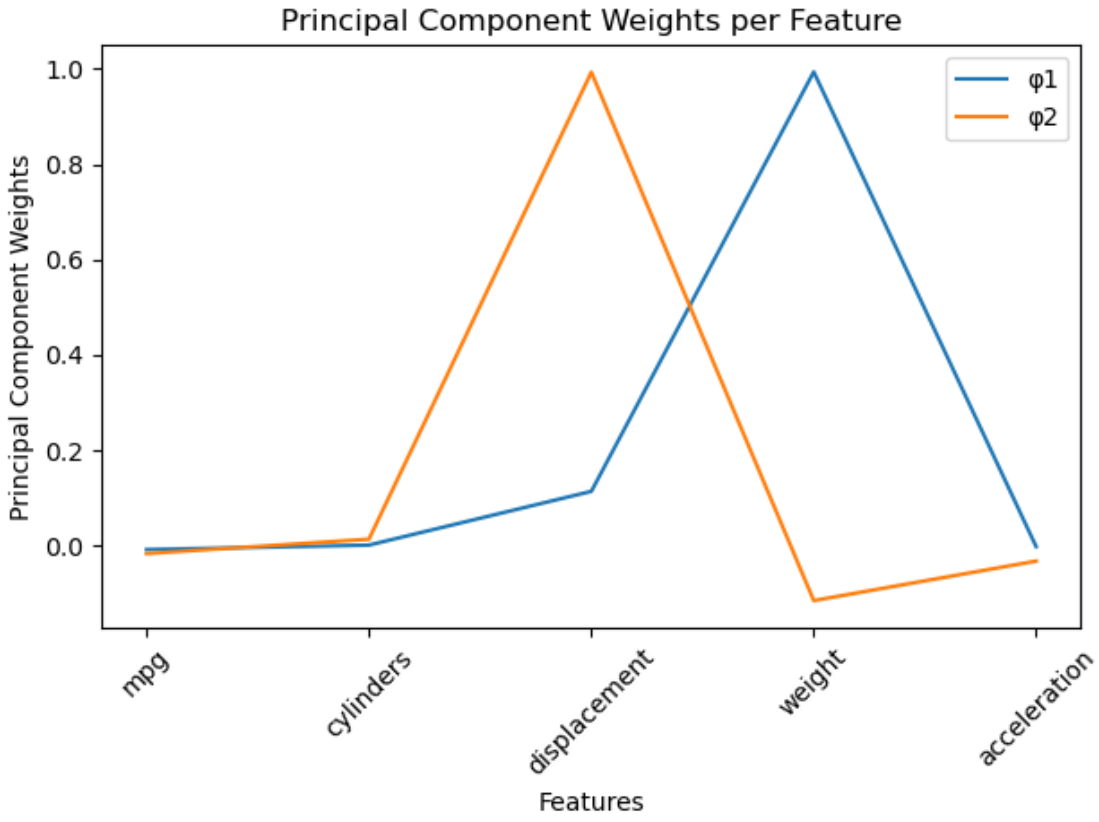
```
Standard deviation
mpg                7.805007
cylinders          1.705783
displacement     104.644004
weight           849.402560
acceleration       2.758864
dtype: float64


Mean
mpg                23.445918
cylinders           5.471939
displacement      194.411990
weight           2977.584184
acceleration       15.541327
dtype: float64
```

Qustion 3

a

```
print("Original data size:",X.shape)
pca = PCA(n_components=2)

X_reduced = pca.fit_transform(X)
print("Reduced data size:", X_reduced.shape)

components = pca.components_
x = np.arange(components.shape[1]) # 6

plt.plot(x, components[0], label='1')
plt.plot(x, components[1], label='2')

names = X.coloumns
plt.xticks(ticks=x, labels=names, rotation=45)

plt.xlabel('Features')
plt.ylabel('Principal Component Weights')
plt.title('Principal Component Weights per Feature')
plt.legend()
```

```
# Show plot
plt.tight_layout()
plt.savefig("plot.png", bbox_inches='tight')
plt.show()
plt.close()
```



b

It appears that PCA is capturing features with larger scales, which can lead to misleading conclusions regarding the importance of these features.

Question 4

```
# Manually standardize the data to have mean 0 and standard deviation 1
means = np.mean(X, axis=0)
stds = np.std(X, axis=0)
Z = (X - means) / stds

# Verify that all features in Z have a mean of approximately zero and a standard deviation of one
means_Z = np.mean(Z, axis=0)
stds_Z = np.std(Z, axis=0)

print("means")
print(means_Z)
print("std")
```

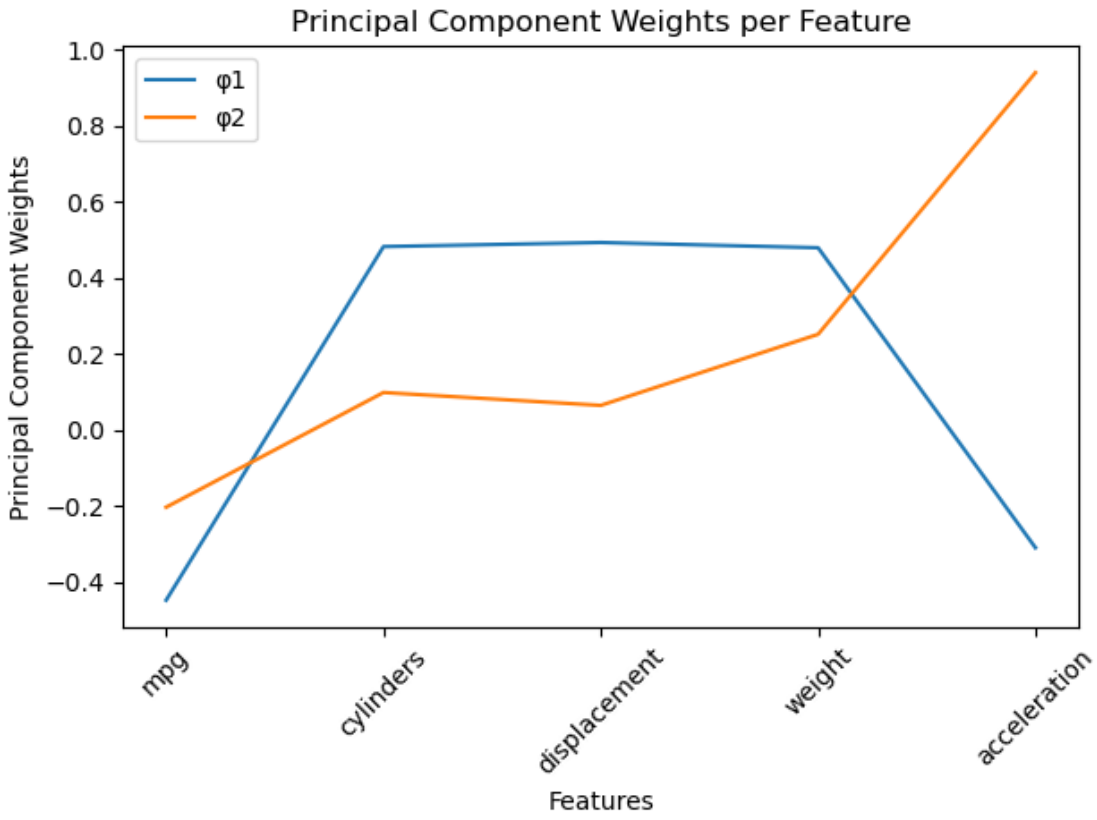
```
print(stds_Z)
```

```
means
mpg             1.450087e-16
cylinders      -1.087565e-16
displacement   -7.250436e-17
weight         -1.812609e-17
acceleration    4.350262e-16
dtype: float64
std
mpg             1.0
cylinders       1.0
displacement    1.0
weight          1.0
acceleration    1.0
dtype: float64
```

Question 5

a

```
pca = PCA(n_components=2)
Z_reduced = pca.fit_transform(Z)
print("Reduced data size:", Z_reduced.shape)
components = pca.components_
plt.plot(x, components[0], label='1')
plt.plot(x, components[1], label='2')

plt.xticks(ticks=x, labels=selected_coloumns, rotation=45)

plt.xlabel('Features')
plt.ylabel('Principal Component Weights')
plt.title('Principal Component Weights per Feature')
plt.legend()

# Show plot
plt.tight_layout()
plt.savefig("plot_standarized.png", bbox_inches='tight')
plt.close()
```



- This plot is less sharp and considers more features than the plot that was not normalized

b

It seems that the three most important features are 'cylinders', 'displacement', and 'weight', as the first principal component ($\phi 1$) weighs these more heavily, with 'displacement' appearing to have the highest contribution to variance by a small margin.

c

```
dot_product = np.dot(components[0], components[1])
print("Dot product: ",dot_product)
magnitude_phi1 = np.linalg.norm(components[0])
print("PC 1 magnitude: ",magnitude_phi1)
magnitude_phi2 = np.linalg.norm(components[1])
print("PC 2 magnitude: ",magnitude_phi2)
```

```
Dot product:  0.0
PC 1 magnitude:  1.0000000000000002
PC 2 magnitude:  1.0000000000000002
```

Question 6