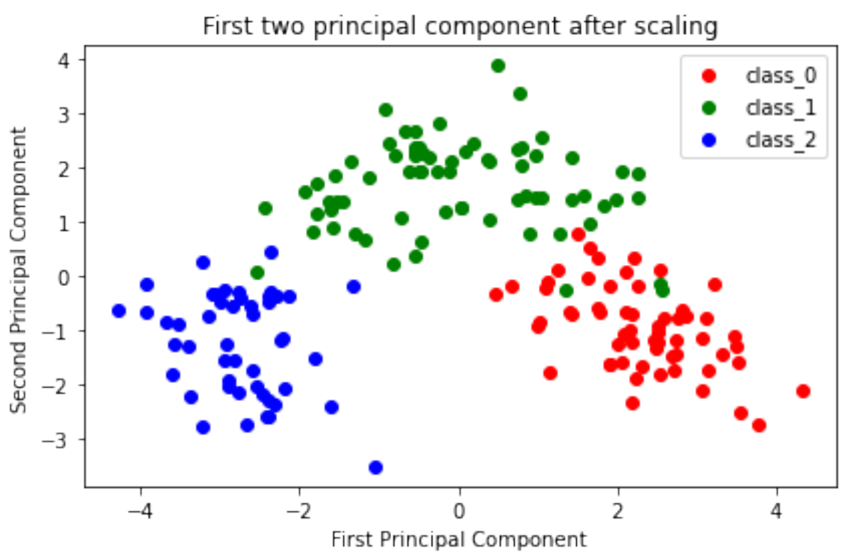
```
from sklearn.datasets import load_wine
wine = load_wine()
```

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler


# Load the wine data
wine = load_wine()
df = wine


# Standardize the data
scaler = StandardScaler()
wine_scaled = scaler.fit_transform(wine.data)


# Apply PCA
pca = PCA(n_components=2)
wine_pca = pca.fit_transform(wine_scaled)


# Plot the first two principle components using matplotlib's scatter function,
# with different colors for each target/class of wine.
colors = ['red', 'green', 'blue']
target_names = wine.target_names
```

```
for target, color in zip(np.unique(wine.target), colors):
    plt.scatter(wine_pca[wine.target == target, 0],
                wine_pca[wine.target == target, 1],
                c=color,
                label=target_names[target])

# Label the x-axis as 'First Principal Component'
# Label the y-axis as 'Second Principal Component'
# Add a legend to show the class of each wine
plt.xlabel('First Principal Component')
plt.ylabel('Second Principal Component')
plt.title("First two principal component after scaling")
plt.legend(loc='best')

# Display the plot
plt.tight_layout()
plt.show()
```



Weihong Yang

Question 2

$$P(x|\lambda) = \frac{e^{-\lambda}\lambda^{x_i}}{x_i!}$$

$$L(\lambda|x) = \prod_{i=1}^{n} \cdot P(x_i|\lambda)$$

$$= \prod_{i=1}^{n} \cdot \frac{e^{-\lambda}\lambda^{x_i}}{x_i!}$$

$$\ln[L(\lambda|x)] = \sum_{i=1}^{n}[\ln(e^{-\lambda}) + \ln(\lambda^{x_i}) - \ln(x_i!)]$$

$$= -n\lambda + \ln(\lambda)\cdot\sum_{i=1}^{n} x_i - \sum_{i=1}^{n}\ln(x_i!)$$

$$\frac{d}{d\lambda}\ln[L(\lambda|x)] = -n + \frac{\Sigma x_i}{\lambda} = 0$$

$$\lambda = \frac{1}{n}\sum_{i=1}^{n} x_i$$

So, $\lambda = \frac{1}{n}\sum_{i=1}^{n} x_i$