Week 5

Generate Data

We use `iris` dataset from `sklearn`

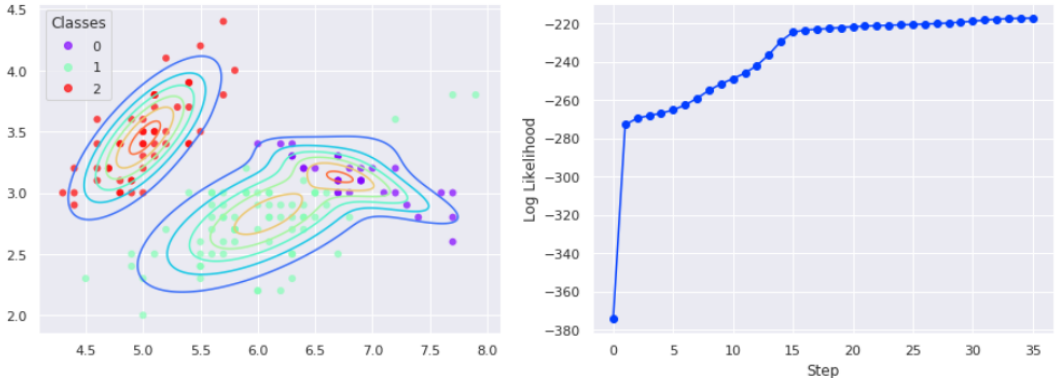
In [15]:
```
iris = datasets.load_iris()
iris_x = np.array(iris.data[:, :2])  # we only take t
iris_t = np.array(iris.target)

def plot_iris(legend=True, classes=iris_t, target=plt
    scatter = target.scatter(iris_x[:, 0], iris_x[:, 
    if legend:
        legend = target.legend(*scatter.legend_elemen
        return (scatter, legend)
    return (scatter, )

plot_iris()
```

Out[15]:
```
(<matplotlib.collections.PathCollection at 0x7ffc3be6e
 <matplotlib.legend.Legend at 0x7ffc3bd842d0>)
```

It will have also an animation here:

In [20]:
```
def create_animation(all_steps_em, data_x):

    fig, (ax, ax2) = plt.subplots(1, 2, figsize=(15,5))

    log_likelihoods = list(map(lambda x: x[3], all_steps_em))

    def animate(i):
        ax.cla()
        ax2.cla()

        predictions = all_steps_em[i][4]

        plot1 = plot_gaussian_mixtures(all_steps_em[i][0], all_steps_em[i][1], all_steps_em[i][2], iris_x, predictions, tar

        ax2.plot(list(range(i)), log_likelihoods[:i], '-o')
        plt.xlabel('Step')
        plt.ylabel('Log Likelihood')
        return plot1

    anim = FuncAnimation(
        fig, animate,
        frames=len(all_steps_em), interval=500, blit=True
    )
    return HTML(anim.to_html5_video())

create_animation(all_steps_em, iris_x)
```

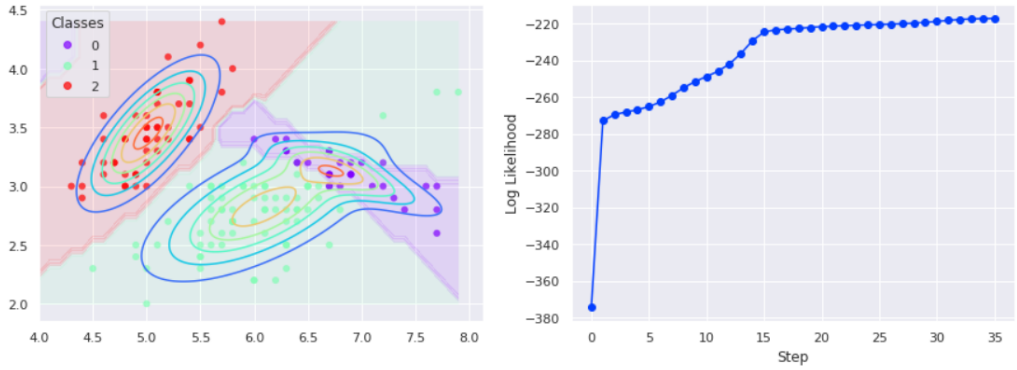
Out[20]:



And here:

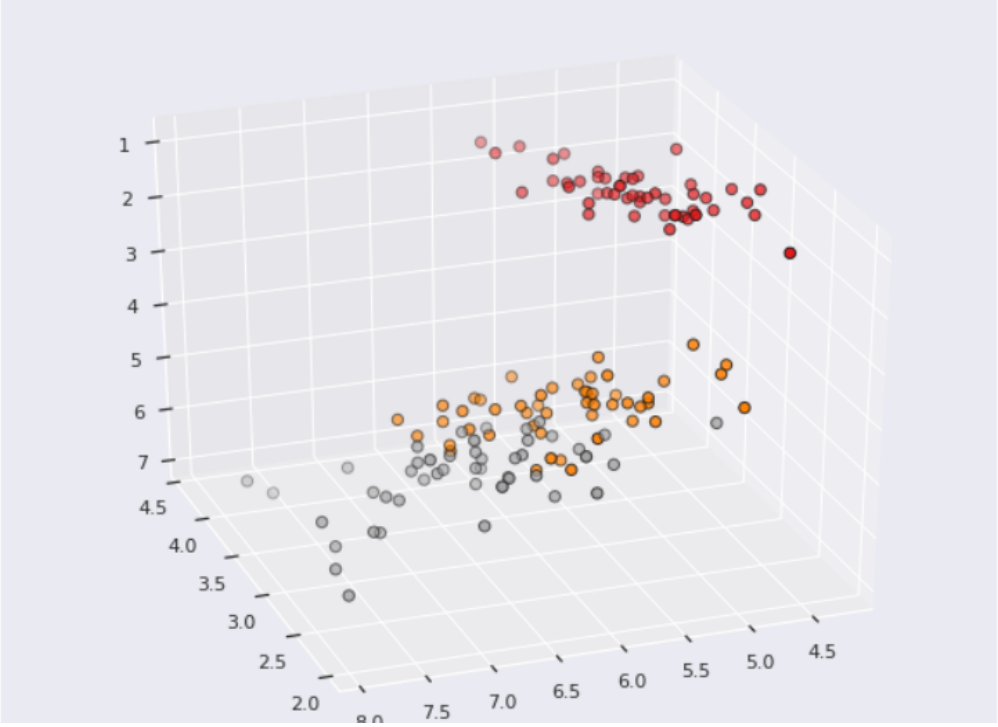
```
    anim = FuncAnimation(
        fig, animate,
        frames=len(all_steps_em), interval=500, blit=True
    )
    return HTML(anim.to_html5_video())

create_animation(all_steps_em, iris_x)
```

Out[22]:



2.1) Generate data

In [23]:
```
iris4_x = np.array(iris.data[:, :4])

def plot_classes_3d(data, classes):

    from mpl_toolkits.mplot3d import Axes3D
    from mpl_toolkits.mplot3d import proj3d

    fig = plt.figure(1, figsize=(8, 6))
    ax = Axes3D(fig, elev=-150, azim=110)
    ax.scatter(data[:, 0], data[:, 1], data[:, 2], c=classes,
               cmap=plt.cm.Set1, edgecolor='k', s=40)
    plt.show()
plot_classes_3d(iris4_x[:, :3], iris_t)
```



2.3) Display projection

[38]:
```
def plot_classes(data, classes, legend=True, target=plt):
    scatter = target.scatter(data[:, 0], data[:, 1], c=cla
    if legend:
        legend = target.legend(*scatter.legend_elements(),
        return (scatter, legend)
    return (scatter, )

projected_data = pca_projection(iris4_x, 2)
plot_classes(projected_data, iris_t)
```

[38]:
```
(<matplotlib.collections.PathCollection at 0x7f320121a990>,
 <matplotlib.legend.Legend at 0x7f31a22a5250>)
```

[39]:
```
projected_data = pca_projection(iris4_x, 3)
plot_classes_3d(projected_data, iris_t)
```