IMPORTING LIBRARIES

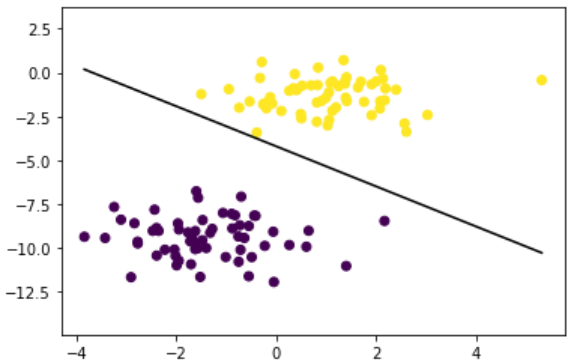
In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import datasets
```

Perceptron Model Class

In [2]:
```
class Perceptron:
    def __init__(self, l_r=0.01, n_iters=1000):
        self.lr = l_r
        self.n_iters = n_iters
        self.act_func = self.step_func
        self.w = None
        self.b = None

    def fit(self, X, y):
        n_samples, n_features = X.shape

        # init parameters
        self.w = np.zeros(n_features)
        self.b = 0

        y_ = np.array([1 if i > 0 else 0 for i in y])

        for _ in range(self.n_iters):

            for idx, x_i in enumerate(X):

                linear_output = np.dot(x_i, self.w) + self.b
                y_predicted = self.act_func(linear_output)

                # Perceptron update rule
                update = self.lr * (y_[idx] - y_predicted)

                self.w += update * x_i
                self.b += update

    def predict(self, X):
        linear_output = np.dot(X, self.w) + self.b
        y_predicted = self.act_func(linear_output)
        return y_predicted

    def step_func(self, x):
        return np.where(x >= 0, 1, 0)

```

In [11]:
```
def accuracy(y_true, y_pred):
    accuracy = np.sum(y_true == y_pred) / len(y_true)
    return accuracy

X, y = datasets.make_blobs(
    n_samples=150, n_features=2, centers=2, cluster_std=1.05, random_state=2
)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=123
)

p = Perceptron(l_r=0.01, n_iters=100)
p.fit(X_train, y_train)
predictions = p.predict(X_test)

print("Perceptron classification accuracy", accuracy(y_test, predictions))

fig = plt.figure()
ax = fig.add_subplot(1, 1, 1)
plt.scatter(X_train[:, 0], X_train[:, 1], marker="o", c=y_train)

x0_1 = np.amin(X_train[:, 0])
x0_2 = np.amax(X_train[:, 0])

x1_1 = (-p.w[0] * x0_1 - p.b) / p.w[1]
x1_2 = (-p.w[0] * x0_2 - p.b) / p.w[1]

ax.plot([x0_1, x0_2], [x1_1, x1_2], "k")

ymin = np.amin(X_train[:, 1])
ymax = np.amax(X_train[:, 1])
ax.set_ylim([ymin - 3, ymax + 3])

plt.show()

```

```
Perceptron classification accuracy 1.0
```



In [ ]:
```

```