MAP classification and 1-d discriminant analysis

Let $X \in R$ represent a feature, and $Y = 0$ or $Y = 1$ the class label.

The distribution of X depends on the label:

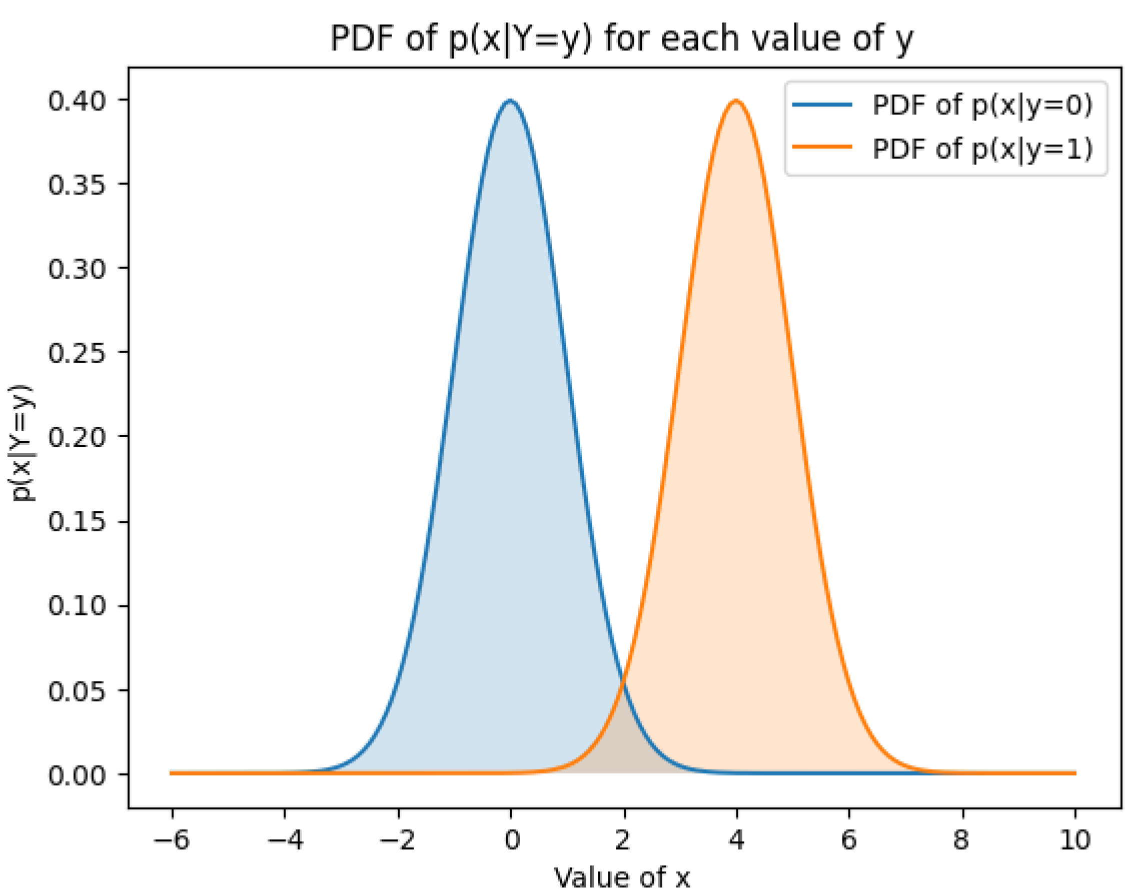
- $(X|Y = 0) \sim N(0, 1)$
- $(X|Y = 1) \sim N(4, 1)$

In [1]:
```
import math
import numpy as np
import matplotlib.pyplot as plt
```

(a) Use a computer to create a plot with both pdfs p(x|y = 0) and p(x|y = 1) on the same axis.

In [2]:
```
def get_gaussian_pdf(mu,var):
    sigma = math.sqrt(var)
    def gaussian_pdf(x):
        return np.exp(-((x-mu)/sigma)**2/2) / math.sqrt(2*math.pi*sigma**2)
    return gaussian_pdf
```

In [18]:
```
y0_gaussian_pdf = get_gaussian_pdf(0, 1)
y1_gaussian_pdf = get_gaussian_pdf(4, 1)
xrange = np.arange(-60,101)/10
plt.plot(xrange, y0_gaussian_pdf(xrange), label="PDF of p(x|y=0)")
plt.fill_between(xrange, y0_gaussian_pdf(xrange), alpha=0.2)
plt.plot(xrange, y1_gaussian_pdf(xrange), label="PDF of p(x|y=1)")
plt.fill_between(xrange, y1_gaussian_pdf(xrange), alpha=0.2)
plt.legend()
plt.title("PDF of p(x|Y=y) for each value of y")
plt.xlabel("Value of x")
plt.ylabel("p(x|Y=y)")
plt.show()
```



(b) Use Bayes and total probability to find an expression for the posterior p(y|x).

p(y|x) = p(x|y)p(y)/p(x) = p(x|y)p(y) / (sum over i of p(x|y_i)p(y_i))

In our case, the possible values y takes are 0 and 1

p(y|x) = p(x|y)p(y)/ (p(x|0)p(0)+p(x|1)p(1))

and P(Y=0)=3/4 and P(y=1)=1/4

(c) Use a computer to evaluate p(y = 0|x = 2) using your expression above. What is p(y = 0|x = 2)?

In [39]:
```
def posterior(y,x):
    if y not in [0,1]:
        return 0
    pdfs = [lambda x: get_gaussian_pdf(0, 1)(x)*3/4, lambda x: get_gaussian_pdf(4, 1)(x)*1/4]
    own_pdf = pdfs[y]
    other_pdf = pdfs[(y+1)%2]
    return own_pdf(x) / (own_pdf(x)+other_pdf(x))

print(f"The value of p(y=0|x=2) is {posterior(0,2)}")
```

```
The value of p(y=0|x=2) is 0.7499999999999999
```

(d) Use maximum a posteriori to design a classification rule that will predict if Y = 0 or Y = 1 given X = x.

In [40]:
```
def classifier(x):
    p0 = posterior(0,x)
    p1 = posterior(1,x)
    if p0<p1:
        return 1
    else:
        return 0
```

(e) What is the true risk of your MAP classifier? Use a computer to find a numerical answer.

In [41]:
```
# numerical answer
samples = 0
loss = 0
for _ in range(69420):
    x0 = np.random.normal(0,1)
    x1 = np.random.normal(4,1)
    pred0 = classifier(x0)
    pred1 = classifier(x1)
    if pred0 != 0:
        loss += 3/4
    if pred1 != 1:
        loss += 1/4
```

```python
    samples += 1
print(loss/samples)
```

```
0.019612503601267647
```

In [ ]: