1. Rules of probability theory

Question 1

$$\begin{aligned} P(A, B) &= P(A|B)P(B) \\ &= \frac{P(A, B)}{P(B)}P(B) \\ &= P(A, B) \end{aligned}$$

Correct.

Question 2

$$\begin{aligned} P(A) &= P(A|B)P(B) + P(A|!B)P(!B) \\ &= \frac{P(A, B)}{P(B)}P(B) + \frac{P(A, !B)}{P(!B)}P(!B) \\ &= P(A, B) + P(A, !B) \\ &= P(A) \end{aligned}$$

Correct.

Question 3

$$\begin{aligned} P(A) &= P(A|B)P(B) + P(A|C)P(C) + P(A|D) + P(D) \\ &= P(A, B) + P(A, C) + P(A|D) + P(D) \end{aligned}$$

Incorrect.

Question 4

$$\begin{aligned} P(A|B) &= \frac{P(B|A)P(A)}{P(B)} \\ P(A|B)P(B) &= P(B|A)P(A) \\ P(A, B) &= P(A, B) \end{aligned}$$

Correct.

Question 5

$$\begin{aligned} P(A|B)P(B) &= P(B|A)P(A) \\ P(A, B) &= P(A, B) \end{aligned}$$

Correct.

2. Logarithms and probability distribution

Normal distribution

$$
\begin{aligned}
f &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \\
log(f) &= log(\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}) \\
&= log(e^{-\frac{(x-\mu)^2}{2\sigma^2}}) - log(\sqrt{2\pi\sigma^2}) \\
&= -\frac{(x-\mu)^2}{2\sigma^2} - \frac{\pi\sigma^2}{2}
\end{aligned}
$$

Exponential distribution

$$
\begin{aligned}
f &= \lambda e^{-\lambda x} \\
log(f) &= log(\lambda e^{-\lambda x}) \\
&= log(\lambda) + log(e^{-\lambda x}) \\
&= log(\lambda) - \lambda x
\end{aligned}
$$

Gamma distribution

$$
\begin{aligned}
f &= \frac{1}{\Gamma(\kappa)\theta^\kappa} x^{\kappa-1} e^{-\frac{x}{\theta}} \\
log(f) &= log(\frac{1}{\Gamma(\kappa)\theta^\kappa} x^{\kappa-1} e^{-\frac{x}{\theta}}) \\
&= log(x^{\kappa-1}) + log(e^{-\frac{x}{\theta}}) - log(\Gamma(\kappa)\theta^\kappa) \\
&= (\kappa-1)log(x) - \frac{x}{\theta} - log(\Gamma(\kappa)) - log(\theta^\kappa) \\
&= (\kappa-1)log(x) - \frac{x}{\theta} - log(\Gamma(\kappa)) - \kappa log(\theta)
\end{aligned}
$$

3. Normal distribution

Question 1

As x is distributed according to the normal distribution, the expected value of x is the mean μ, and as $f(x) = x^3 + 3$, the expected value of $f(x)$ is when $x = \mu$, which is $\mu^3 + 3$.

Question 2

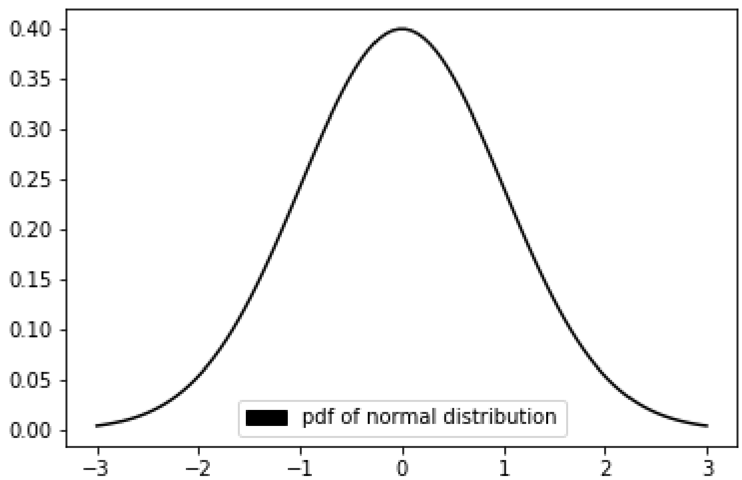
Calculate $P(f(x) \geq 0.5)$

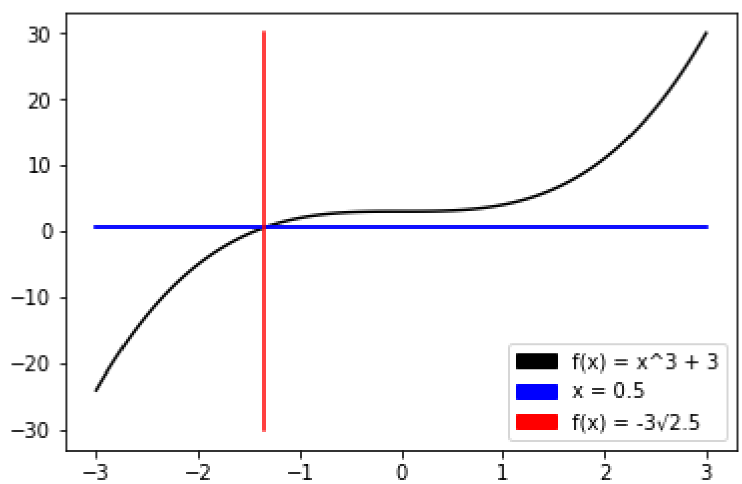
In [6]:
```
from scipy.stats import norm
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches

x = np.linspace(-3, 3, 100)

plt.plot(x, norm.pdf(x), 'k-')
function_legend = mpatches.Patch(color='k', label='pdf of normal distrib
ution')
plt.legend(handles=[function_legend])
plt.show()

plt.plot(x, x**3+3, 'k-')
plt.plot(x, [0.5 for _ in range(100)], 'b-')
plt.plot(x, [0.5 for _ in range(100)], 'b-')
plt.plot([-(2.5)**(1/3) for _ in range(100)], np.linspace(-30, 30, 100),
 'r-')
function_legend = mpatches.Patch(color='k', label='f(x) = x^3 + 3')
x_legend = mpatches.Patch(color='b', label='x = 0.5')
y_legend = mpatches.Patch(color='r', label='f(x) = -3√2.5')
plt.legend(handles=[function_legend, x_legend, y_legend])
plt.show()
```





As shown in the plot, the $P(f(x) \geq 0.5)$ can be maped by the function $f(x) = x^3 + 3$ as

$$P(x \geq -\sqrt[3]{2.5})$$

Use the intergral of the pdf to calculate

$$P(x \geq -\sqrt[3]{2.5}) = \int_{-\sqrt[3]{2.5}}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx$$

Calculate result for $\mu = 0$ $\sigma = 1$

$$P(f(x) \geq 0.5) = \int_{-\sqrt[3]{2.5}}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$$
$$= 0.912643$$

Calculate result for $\mu = 10$ $\sigma = 12$

$$P(f(x) \geq 0.5) = \int_{-\sqrt[3]{2.5}}^{\infty} \frac{1}{\sqrt{2\pi 12^2}} e^{-\frac{(x-10)^2}{2\times 12^2}} dx$$
$$= 0.828036$$

Calculate result for $\mu = -18$ $\sigma = 5$

$$P(f(x) \geq 0.5) = \int_{-\sqrt[3]{2.5}}^{\infty} \frac{1}{\sqrt{2\pi 5^2}} e^{-\frac{(x+18)^2}{2\times 5^2}} dx$$
$$= 0.000436484$$

In [2]:
```
from scipy.stats import norm
import numpy as np

mu_set = [0, 10, -18]
sigma_set = [1, 12, 5]
sample_size = 1000000

for mu, sigma in zip (mu_set, sigma_set):
    distribution = norm(loc=mu, scale=sigma)
    sample = distribution.rvs(size=sample_size)
    success = 0
    for i in range(0, sample_size):
        sample[i] = sample[i]**3 + 3
        if sample[i] > 0.5:
            success += 1
    print ("For μ=%d, σ=%d, and s=%d, the simulated P(f(x)>0.5)=%.6f" % (
mu, sigma, sample_size, success/sample_size))
```

```
For μ=0, σ=1, and s=1000000, the simulated P(f(x)>0.5)=0.912232
For μ=10, σ=12, and s=1000000, the simulated P(f(x)>0.5)=0.828198
For μ=-18, σ=5, and s=1000000, the simulated P(f(x)>0.5)=0.000442
```

The simulated result match the calculated result.

4. Marginal and conditional probabilities

Information

$$\begin{aligned} P(young) &= 0.273 \\ P(unemployed|young) &= 0.382 \\ P(unemployed|notyoung) &= 0.224 \end{aligned}$$

Questions

$$\begin{aligned} P(young) &= 0.273 \\ P(not\ young) &= 1 - 0.273 = 0.727 \\ P(unemployed) &= 0.382 + 0.224 = 0.606 \\ P(not\ unemployed) &= 1 - 0.606 = 0.394 \\ P(unemployed|young) &= 0.382 \\ P(young|unemployed) &= P(young, unemployed)/P(unemployed) \\ &= P(young) * P(unemployed|young)/P(unemployed) \\ &= 0.273 * 0.382/0.606 = 0.172 \\ P(unemployed|not\ young) &= 0.224 \\ P(not\ young|unemployed) &= P(not\ young, unemployed)/P(unemployed) \\ &= P(not\ young) * P(unemployed|not\ young)/P(unemployed) \\ &= 0.727 * 0.224/0.606 = 0.269 \end{aligned}$$

5. Inference

Notation

R represents able to read at Grade 1 level by 6 years old

T represents training in pronouncing simple words and writing some letters

Information

From the reasonable guess

$$P(R) = 0.3$$

From the educational expert

$$P(T|R) = 0.6$$
$$P(T|!R) = 0.1$$

Through simple calculation

$$P(!R) = 0.7$$
$$P(!T|R) = 0.4$$
$$P(!T|!R) = 0.9$$

Question 1

Probability that Olivia will be able to read knowing she has training for reading

$$\begin{aligned} P(R|T) &= \frac{P(T|R)P(R)}{P(T|R)P(R) + P(T|!R)P(!R)} \\ &= \frac{0.6 \times 0.3}{0.6 \times 0.3 + 0.1 \times 0.7} \\ &= \frac{0.18}{0.18 + 0.07} \\ &= 0.72 \end{aligned}$$

Question 2

The variables here are R represents able to read at Grade 1 level by 6 years old, and T represents training in pronouncing simple words and writing some letters. This is a Bayesian statistical modele, and we use the additional information (in this case $P(T|R) = 0.6$ and $P(T|!R) = 0.1$) to update the prior belief (in this case $P(R) = 0.3$) to the posterior belief after observing the new evidence (in this case $P(R|T) = 0.72$)

6. Calculating probabilities

Questiom 1

There are 14 students in this CS146 classroom.

Question 2

As there are 14 students, the total number of permutation of birthday sequence is

$$\# \; of \; permutation = 14!$$

The total number of permutation of birthday sequence is given I am the first or last is

$$\begin{aligned} \# \; of \; permutation &= 1 \times 13! \\ &= 13! \end{aligned}$$

So the probility of me being the first of last in the birth sequence (equivalent to everyone else is older or everyone else is younger than me) is

$$\begin{aligned} P &= \frac{13!}{14!} \\ &= \frac{1}{14} \end{aligned}$$

7. Python script

In [3]:
```
import numpy as np

# generate 14 random number from a normal distribution
# the number will represent how old each student is, which a larger numb
er represent an older student
# assume I am the first person in the generated population
class_size = 14

population = np.random.uniform(size = class_size)
oldest = True
youngest = True
half_oldest = 0
for i in range(1, class_size):
    # there is no change of someone is exactly as old as me, but we will
 include = for implementation purpose
    if population[0] <= population[i]: # if someone is older than me
        oldest = False
    else:
        youngest = False
        half_oldest += 1
print ("The oldest in class:", oldest)
print ("The youngest in class:", youngest)
print ("Older than at least half in class in class:", (half_oldest/14)>
0.5)
```

```
The oldest in class: False
The youngest in class: False
Older than at least half in class in class: True
```

In [4]:
```
import numpy as np

class_size = 14
repetition = 100000
oldest_count = 0
youngest_count = 0
half_oldest_count = 0

for _ in range(repetition):
    population = np.random.uniform(size = class_size)
    oldest = True
    youngest = True
    half_oldest = 0
    for i in range(1, class_size):
        # there is no change of someone is exactly as old as me, but we
 will include = for implementation purpose
        if population[0] <= population[i]: # someone is older than me
            oldest = False
        else: # someone is younger than me
            youngest = False
            half_oldest += 1
    if oldest:
        oldest_count += 1
    if youngest:
        youngest_count += 1
    if (half_oldest/14)>0.5:
        half_oldest_count += 1

print ("Simulated probility of the oldest in class:", oldest_count/repet
ition)
print ("Simulated probility of the youngest in class:", youngest_count/r
epetition)
print ("Simulated probility of older than half in class:", half_oldest_c
ount/repetition)
```

```
Simulated probility of the oldest in class: 0.07161
Simulated probility of the youngest in class: 0.07182
Simulated probility of older than half in class: 0.42859
```