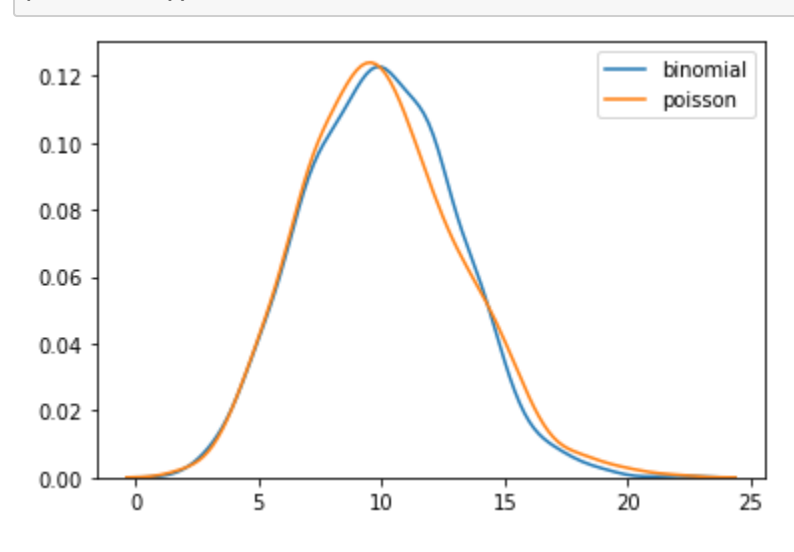
In [3]:

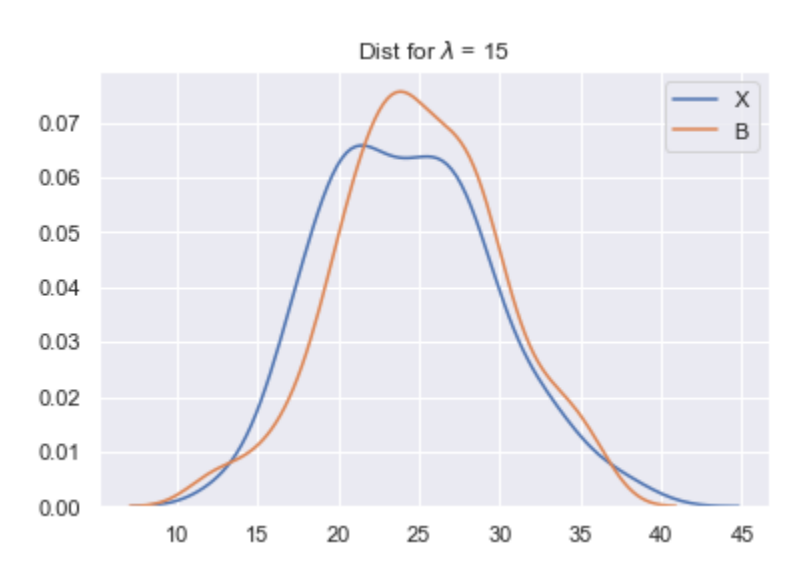
```python
"""
Created on Tues Sep  1 13:45:06 2020



@Group 3: Youssaf & Rashedur



problem: Use the Poisson distribution as a candidate distribution to generate samples from t
he bino-
mial distribution



Prof: Josic & Prof:Stewart



"""
from numpy import random
#here we ill import matplotlib
import matplotlib.pyplot as plt
#now we will import seaborn
import seaborn as sns
#we will plot a displot here


sns.distplot(random.binomial(n=200,p=0.05,size=1000), hist=False, label='binomial')
sns.distplot(random.poisson(lam=10,size=1000), hist=False, label='poisson')


# now we have the plot printed
plt.show()
```



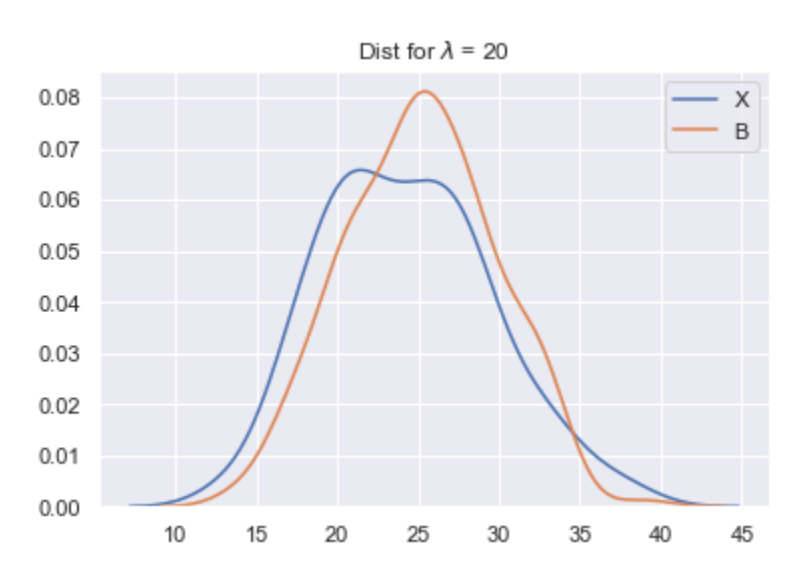
In [4]:

```python
import numpy as np
import scipy.stats as sps
import matplotlib.pyplot as plt
import seaborn as sns


sns.set()

# f is the target function : binomial
# g is the candidate function: poisson


#parameters for binomial
n=500
p=0.05
#n*p = 25
#parameter for poisson
lamda=[15,20,25,35]

# Rejection Sampling
X = []

c = 1.5 # 5,10,15
print("c1= ", c)

iteration=200
for i in range(len(lamda)):

    while(len(X) < iteration):

        X1=np.random.poisson(lamda[i])   #simulating X variable from poisson
        U=np.random.uniform(0,1)         #uniform
        f=sps.binom.pmf(X1, n, p)
        g=sps.poisson.pmf(X1, lamda[i])

        if(U <(f/(c*g))):
            X.append(X1)

        c= max(c, f/g)

    plt.title("Rejection Sampling Dist for $\lambda$ = " + str(lamda[i]))


    B = np.random.binomial(n,p,iteration)

    plt.title("Dist for $\lambda$ = " + str(lamda[i]))


    sns.distplot(X, hist=False, label= "X")
    sns.distplot(B, hist=False, label= "B")

    plt.show()
    print("acceptance rate= ", 100/c)
```

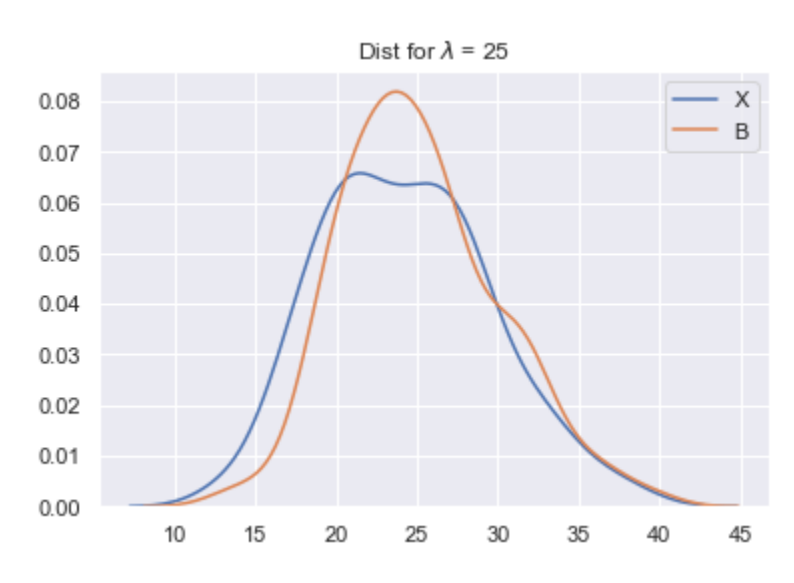
```
c1=  1.5
```



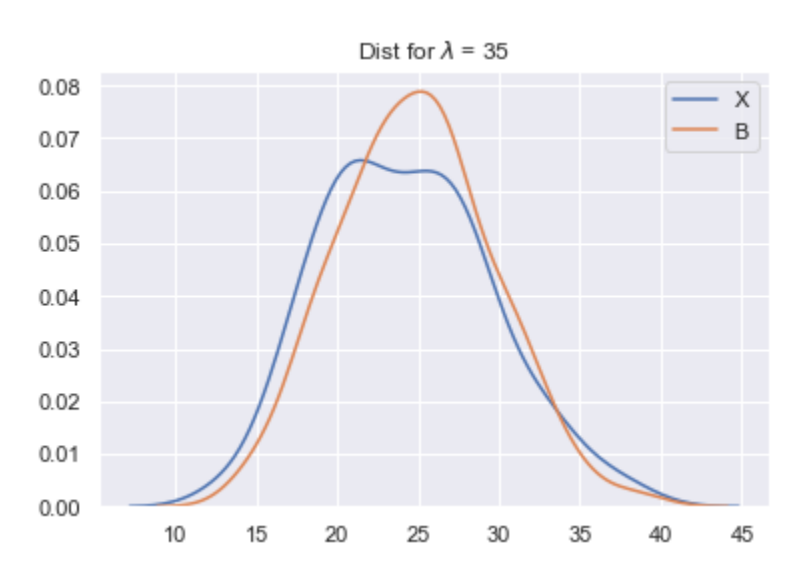

```
acceptance rate=  0.005803773860973675
```




```
acceptance rate=  0.005803773860973675
```




```
acceptance rate=  0.005803773860973675
```

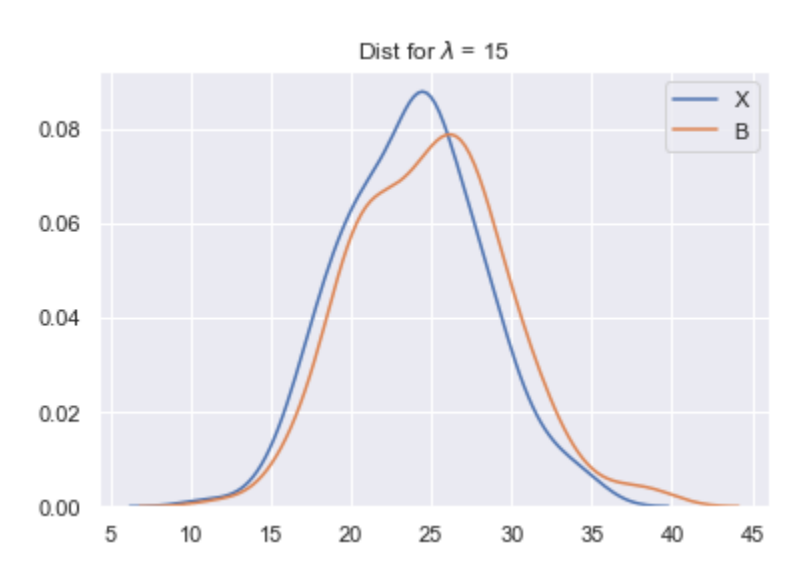



```
acceptance rate=  0.005803773860973675
```

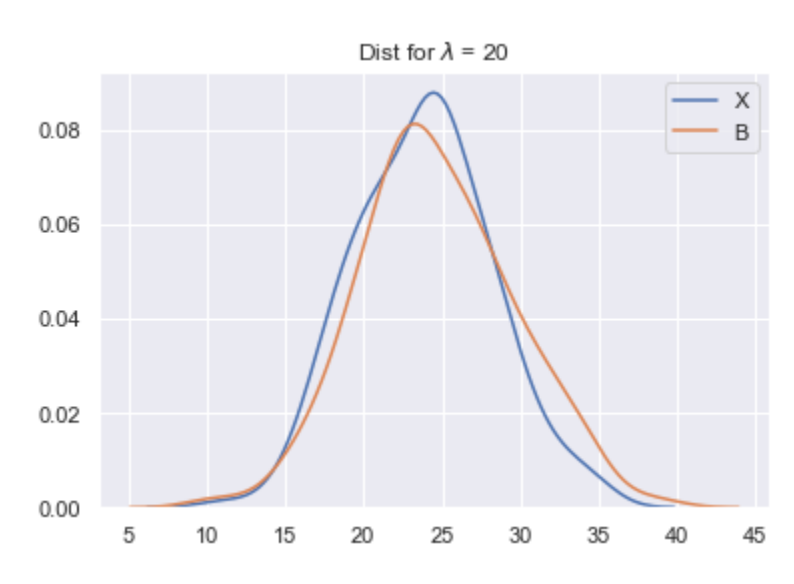
In [5]:

```python
import numpy as np
import scipy.stats as sps
import matplotlib.pyplot as plt
import seaborn as sns


sns.set()

# f is the target function : binomial
# g is the candidate function: poisson


#parameters for binomial
n=500
p=0.05
#n*p = 25
#parameter for poisson
lamda=[15,20,25,35]

# Rejection Sampling
X = []

c = 5 # 1.5, 5,10,15
print("c1= ", c)

iteration=200
for i in range(len(lamda)):

    while(len(X) < iteration):

        X1=np.random.poisson(lamda[i])   #simulating X variable from poisson
        U=np.random.uniform(0,1)         #uniform
        f=sps.binom.pmf(X1, n, p)
        g=sps.poisson.pmf(X1, lamda[i])

        if(U <(f/(c*g))):
            X.append(X1)

        c= max(c, f/g)

    plt.title("Rejection Sampling Dist for $\lambda$ = " + str(lamda[i]))


    B = np.random.binomial(n,p,iteration)

    plt.title("Dist for $\lambda$ = " + str(lamda[i]))


    sns.distplot(X, hist=False, label= "X")
    sns.distplot(B, hist=False, label= "B")

    plt.show()
    print("acceptance rate= ", 100/c)
```

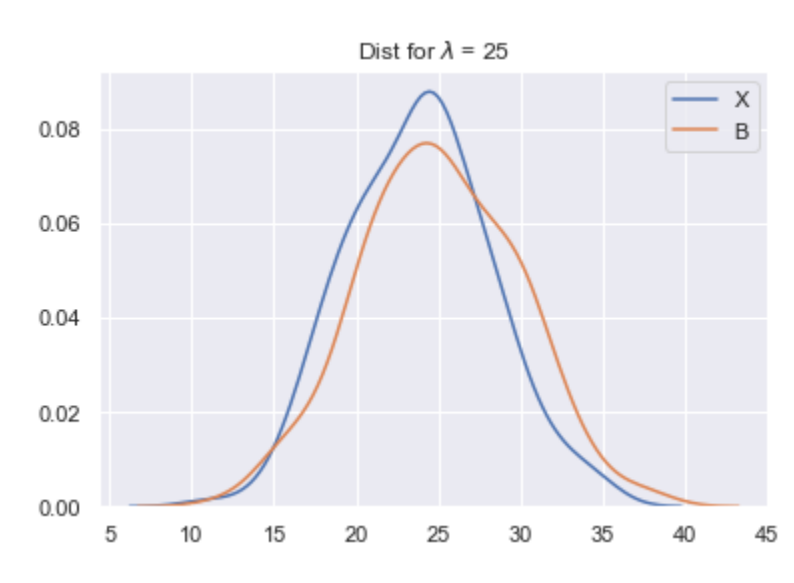
```
c1=  5
```



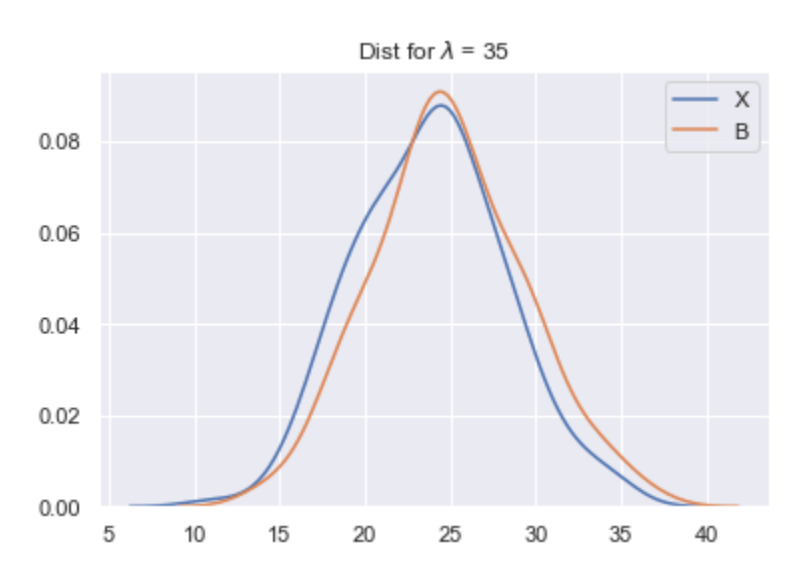

```
acceptance rate=  0.040599897970112514
```




```
acceptance rate=  0.040599897970112514
```




```
acceptance rate=  0.040599897970112514
```

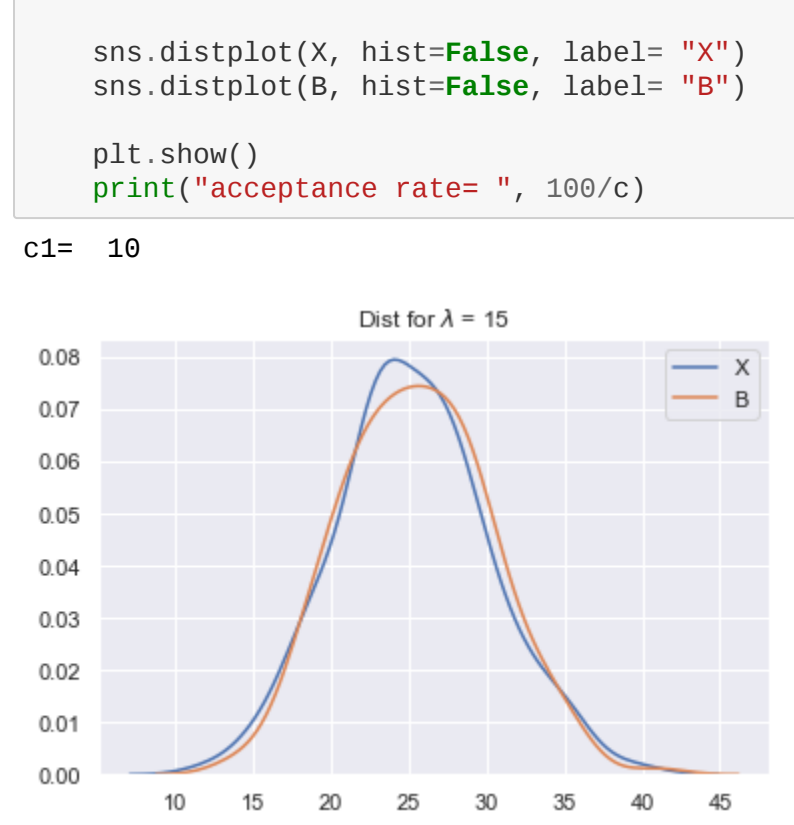



```
acceptance rate=  0.040599897970112514
```

In [6]:

```python
import numpy as np
import scipy.stats as sps
import matplotlib.pyplot as plt
import seaborn as sns


sns.set()

# f is the target function : binomial
# g is the candidate function: poisson


#parameters for binomial
n=500
p=0.05
#n*p = 25
#parameter for poisson
lamda=[15,20,25,35]

# Rejection Sampling
X = []
c=10
print("c1= ", c)

iteration=200
for i in range(len(lamda)):

    while(len(X) < iteration):

        X1=np.random.poisson(lamda[i])   #simulating X variable from poisson
        U=np.random.uniform(0,1)         #uniform
        f=sps.binom.pmf(X1, n, p)
        g=sps.poisson.pmf(X1, lamda[i])

        if(U <(f/(c*g))):
            X.append(X1)

        c= max(c, f/g)

    plt.title("Rejection Sampling Dist for $\lambda$ = " + str(lamda[i]))


    B = np.random.binomial(n,p,iteration)

    plt.title("Dist for $\lambda$ = " + str(lamda[i]))


    sns.distplot(X, hist=False, label= "X")
    sns.distplot(B, hist=False, label= "B")

    plt.show()
    print("acceptance rate= ", 100/c) 10 # 5,10,15
print("c1= ", c)

iteration=200
for i in range(len(lamda)):

    while(len(X) < iteration):

        X1=np.random.poisson(lamda[i])   #simulating X variable from poisson
        U=np.random.uniform(0,1)         #uniform
        f=sps.binom.pmf(X1, n, p)
        g=sps.poisson.pmf(X1, lamda[i])

        if(U <(f/(c*g))):
            X.append(X1)

        c= max(c, f/g)

    plt.title("Rejection Sampling Dist for $\lambda$ = " + str(lamda[i]))


    B = np.random.binomial(n,p,iteration)

    plt.title("Dist for $\lambda$ = " + str(lamda[i]))


    sns.distplot(X, hist=False, label= "X")
    sns.distplot(B, hist=False, label= "B")

    plt.show()
    print("acceptance rate= ", 100/c)
```

```
c1=  10
```




```
acceptance rate=  0.003588016378257917
```


```
acceptance rate=  0.003588016378257917
```

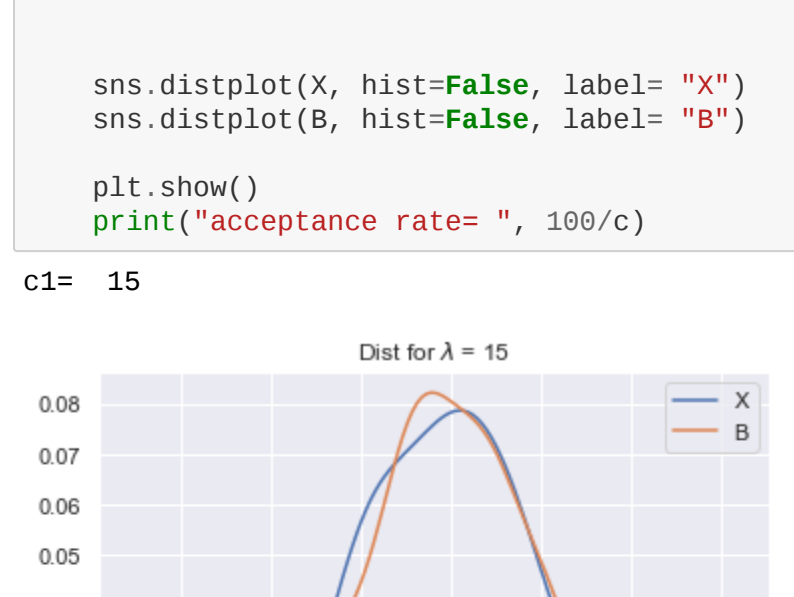

```
acceptance rate=  0.003588016378257917
```


```
acceptance rate=  0.003588016378257917
```

In [9]:

```python
import numpy as np
import scipy.stats as sps
import matplotlib.pyplot as plt
import seaborn as sns


sns.set()

# f is the target function : binomial
# g is the candidate function: poisson


#parameters for binomial
n=500
p=0.05
#n*p = 25
#parameter for poisson
lamda=[15,20,25,35]

# Rejection Sampling
X = []

c = 15 # 5,10,15
print("c1= ", c)

iteration=200
for i in range(len(lamda)):

    while(len(X) < iteration):

        X1=np.random.poisson(lamda[i])   #simulating X variable from poisson
        U=np.random.uniform(0,1)         #uniform
        f=sps.binom.pmf(X1, n, p)
        g=sps.poisson.pmf(X1, lamda[i])

        if(U <(f/(c*g))):
            X.append(X1)

        c= max(c, f/g)

    plt.title("Rejection Sampling Dist for $\lambda$ = " + str(lamda[i]))


    B = np.random.binomial(n,p,iteration)

    plt.title("Dist for $\lambda$ = " + str(lamda[i]))


    sns.distplot(X, hist=False, label= "X")
    sns.distplot(B, hist=False, label= "B")

    plt.show()
    print("acceptance rate= ", 100/c)
```

```
c1=  15
```




```
acceptance rate=  0.024883808433292787
```


```
acceptance rate=  0.024883808433292787
```


```
acceptance rate=  0.024883808433292787
```


```
acceptance rate=  0.024883808433292787
```

In [ ]: