In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from io import BytesIO
import pandas as pd
import seaborn as sns
from sklearn.preprocessing import normalize
import numpy as np
import cv2
from scipy import signal
from scipy.signal import fftconvolve
```

In [2]:
```
def normxcorr2(Img1, Img2):

    Img1 = Img1 - np.mean(Img1)
    Img2 = Img2 - np.mean(Img2)

    a1 = np.ones(Img1.shape)

    ar = np.flipud(np.fliplr(Img1))
    out = fftconvolve(Img2, ar.conj(), mode="full")
    Img2 = fftconvolve(np.square(Img2), a1, mode="full") - \
            np.square(fftconvolve(Img2, a1, mode="full")) / (np.prod(Img1.shape)


    Img2[np.where(Img2 < 0)] = 0
    Img1 = np.sum(np.square(Img1))
    out = out / np.sqrt(Img2 * Img1)

    out[np.where(np.logical_not(np.isfinite(out)))] = 0
    return out
```

In [3]:
```
Dist_p1=np.zeros((100,100))
Dist_p2=np.zeros((100,100))
for i in range(100):
    Img=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/GallerySet/subject'+
    for j in range(100):
        Img1=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec
        Img2=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec

        #c = np.correlate(a, b, 'full')
        cor1 = np.max(normxcorr2(Img,Img1))
        cor2 = np.max(normxcorr2(Img,Img2))

        #cor2 = signal.correlate2d (Img,Img2)
        Dist_p1[j,i]=cor1
        Dist_p2[j,i]=cor2
```

Genuine and imposter scores

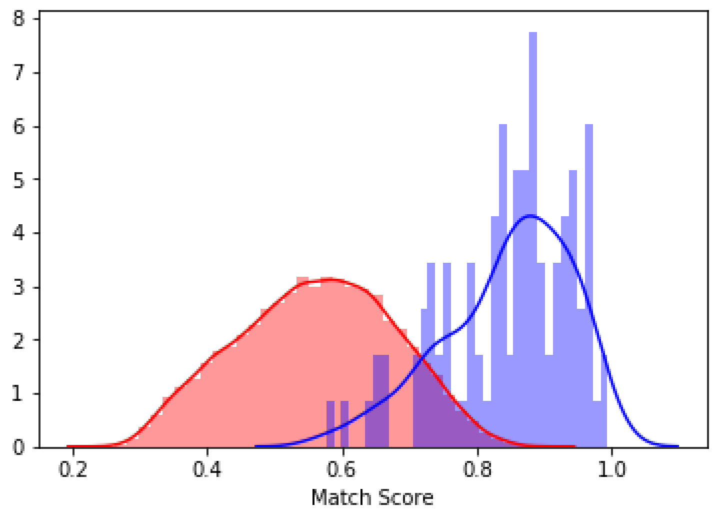
In [4]:
```
Gen=np.append(np.diag(Dist_p1),np.diag(Dist_p1))

non_diag = np.ones(shape=Dist_p1.shape, dtype=bool) ^ np.identity(len(Dist_p1)).
Imp_a=Dist_p1[non_diag==True]
non_diag = np.ones(shape=Dist_p2.shape, dtype=bool) ^ np.identity(len(Dist_p2)).
Imp_b=Dist_p2[non_diag==True]
Imp=np.append(Imp_a,Imp_b)
```

PART I Entire Face Performance

(a) Plot the genuine and impostor score distributions.

In [5]:
```
s1 = sns.distplot(Imp, kde=True,
             bins=int(180/5), color = 'red')

s2 = sns.distplot(Gen, kde=True,
             bins=int(180/5), color = 'blue')

plt.xlabel('Match Score')
```

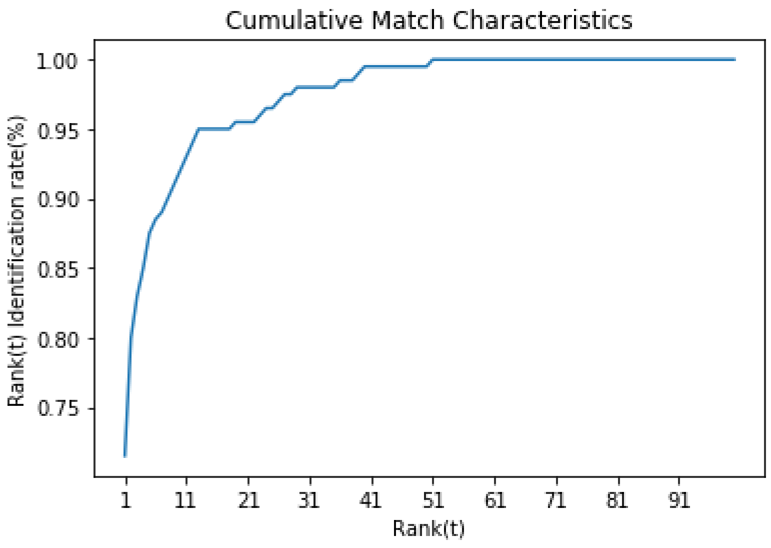
Out[5]:
```
Text(0.5, 0, 'Match Score')
```



b)Cumulative match charecteristics(CMC)

In [6]:
```
T=[]
P_a=[]
for t in range(0,100):
    T.append(t+1)
    temp=0
    for i in range(0,100):
        if i in Dist_p1[i].argsort()[-(t+1):][::-1]:
            temp+=1
        if i in Dist_p2[i].argsort()[-(t+1):][::-1]:
            temp+=1

    p=temp/200
    P_a.append(p)
```

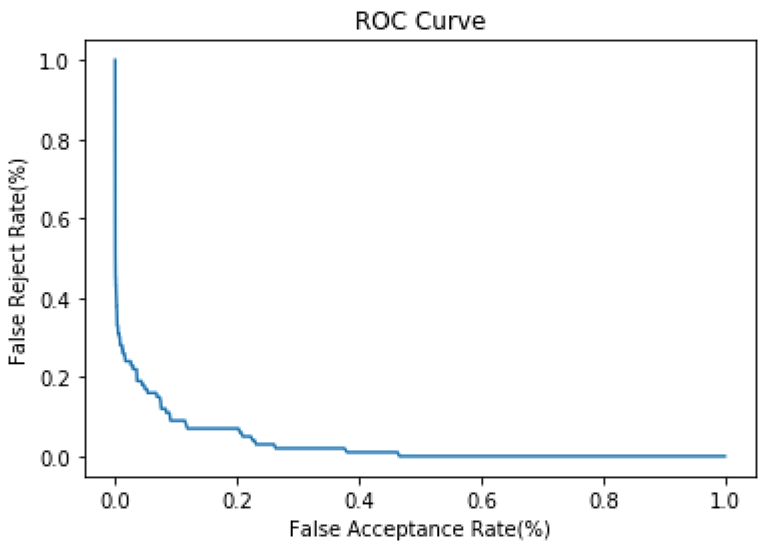
In [7]:
```
plt.plot(T,P_a)
plt.xlabel('Rank(t)')
plt.ylabel('Rank(t) Identification rate(%)')
plt.xticks(np.arange(1, 100, step=10))
plt.title('Cumulative Match Characteristics')
plt.show()
```



C)plot the Receiver Operating Curve (FAR vs. FRR)

In [8]:
```
T_a=np.arange(0, 1, 0.001)
FAR=[]
FRR=[]
for t in T_a:
    far=len(Imp[Imp>t])/len(Imp)
    FAR.append(far)
    frr=len(Gen[Gen<t])/len(Gen)
    FRR.append(frr)

FAR_a=np.array(FAR)
FRR_a=np.array(FRR)

plt.plot(FAR_a,FRR_a)
plt.xlabel('False Acceptance Rate(%)')
plt.ylabel('False Reject Rate(%)')
plt.title('ROC Curve')
```

Out[8]:
```
Text(0.5, 1.0, 'ROC Curve')
```



In [9]:
```
Gen_mean=np.mean(Gen)
Imp_mean=np.mean(Imp)
Gen_var=np.var(Gen)
Imp_var=np.var(Imp)

d_a = np.sqrt(2)*np.absolute(Gen_mean-Imp_mean)/np.sqrt(Gen_var+Imp_var)
print("d_a=",d_a)

IDX=np.argmin(np.absolute(FAR_a-FRR_a))
EER=FRR_a[IDX]
O_p=T_a[IDX]
print("Error_rate=",EER)
print("Operating_pt=",O_p)
```

```
d_a= 2.7013913002089223
Error_rate= 0.09
Operating_pt= 0.719
```

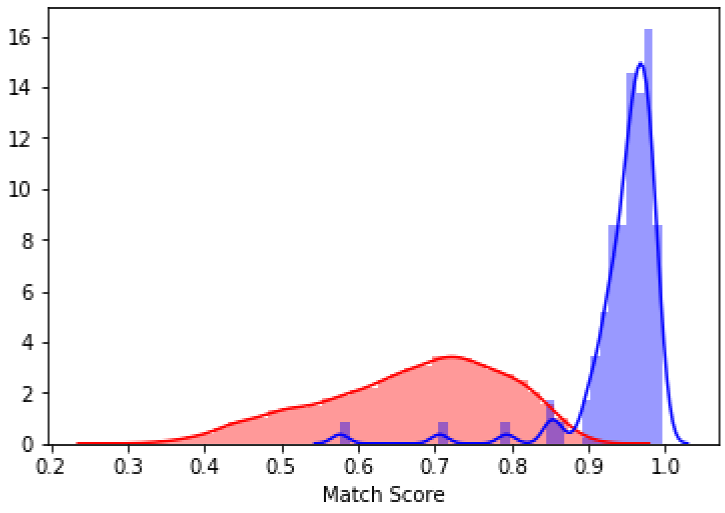
Part 2 partial face performance

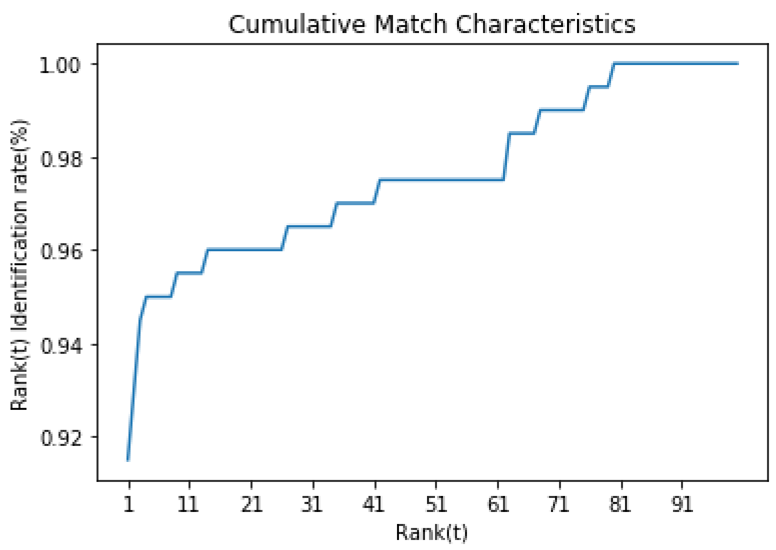
Top half

In [10]:
```
Gal=[]
Dist_p1=np.zeros((100,100))
Dist_p2=np.zeros((100,100))
for i in range(100):
    Img=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/GallerySet/subject'+
    for j in range(100):
        Img1=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec
        Img2=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec

        #c = np.correlate(a, b, 'full')
        cor1 = np.max(normxcorr2(Img,Img1))
        cor2 = np.max(normxcorr2(Img,Img2))

        #cor2 = signal.correlate2d (Img,Img2)
        Dist_p1[j,i]=cor1
        Dist_p2[j,i]=cor2
```

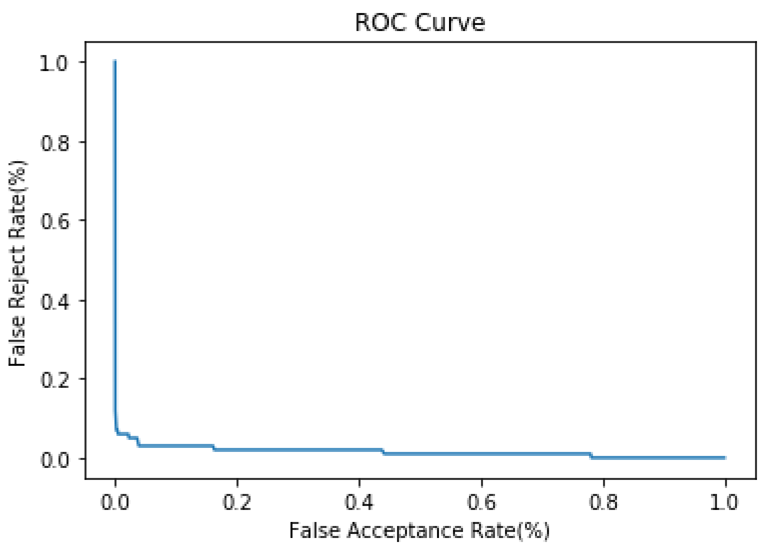
In [11]:
```
Gen=np.append(np.diag(Dist_p1),np.diag(Dist_p1))

non_diag = np.ones(shape=Dist_p1.shape, dtype=bool) ^ np.identity(len(Dist_p1)).
Imp_a=Dist_p1[non_diag==True]
non_diag = np.ones(shape=Dist_p2.shape, dtype=bool) ^ np.identity(len(Dist_p2)).
Imp_b=Dist_p2[non_diag==True]
Imp=np.append(Imp_a,Imp_b)
s1 = sns.distplot(Imp, kde=True,
             bins=int(180/5), color = 'red')

s2 = sns.distplot(Gen, kde=True,
             bins=int(180/5), color = 'blue')

plt.xlabel('Match Score')



plt.figure()
T=[]
P_a=[]
for t in range(0,100):
    T.append(t+1)
    temp=0
    for i in range(0,100):
        if i in Dist_p1[i].argsort()[-(t+1):][::-1]:
            temp+=1
        if i in Dist_p2[i].argsort()[-(t+1):][::-1]:
            temp+=1

    p=temp/200
    P_a.append(p)

plt.plot(T,P_a)
plt.xlabel('Rank(t)')
plt.ylabel('Rank(t) Identification rate(%)')
plt.xticks(np.arange(1, 100, step=10))
plt.title('Cumulative Match Characteristics')
plt.show()

plt.figure()
T_a=np.arange(0, 1, 0.001)
FAR=[]
FRR=[]
for t in T_a:
    far=len(Imp[Imp>t])/len(Imp)
    FAR.append(far)
    frr=len(Gen[Gen<t])/len(Gen)
    FRR.append(frr)

FAR_a=np.array(FAR)
FRR_a=np.array(FRR)

plt.plot(FAR_a,FRR_a)
plt.xlabel('False Acceptance Rate(%)')
plt.ylabel('False Reject Rate(%)')
plt.title('ROC Curve')
```

```
Gen_mean=np.mean(Gen)
Imp_mean=np.mean(Imp)
Gen_var=np.var(Gen)
Imp_var=np.var(Imp)

d_a = np.sqrt(2)*np.absolute(Gen_mean-Imp_mean)/np.sqrt(Gen_var+Imp_var)
print("d_a=",d_a)

IDX=np.argmin(np.absolute(FAR_a-FRR_a))
EER=FRR_a[IDX]
O_p=T_a[IDX]
print("Error_rate=",EER)
print("Operating_pt=",O_p)
```





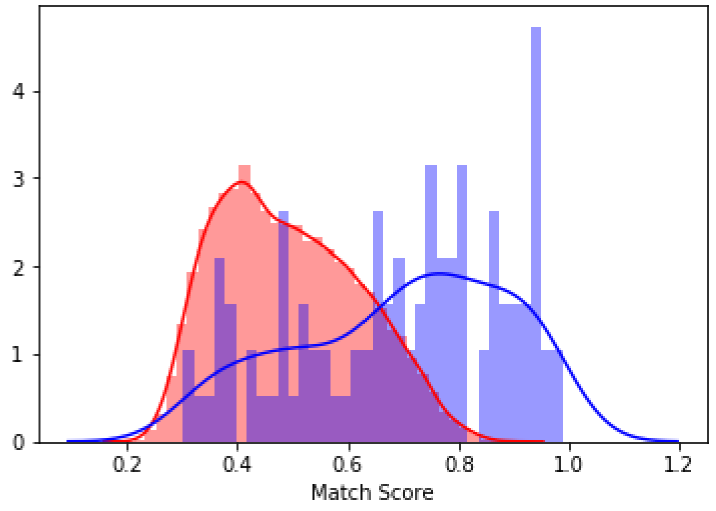
```
d_a= 2.957011723598444
Error_rate= 0.04
Operating_pt= 0.85
```

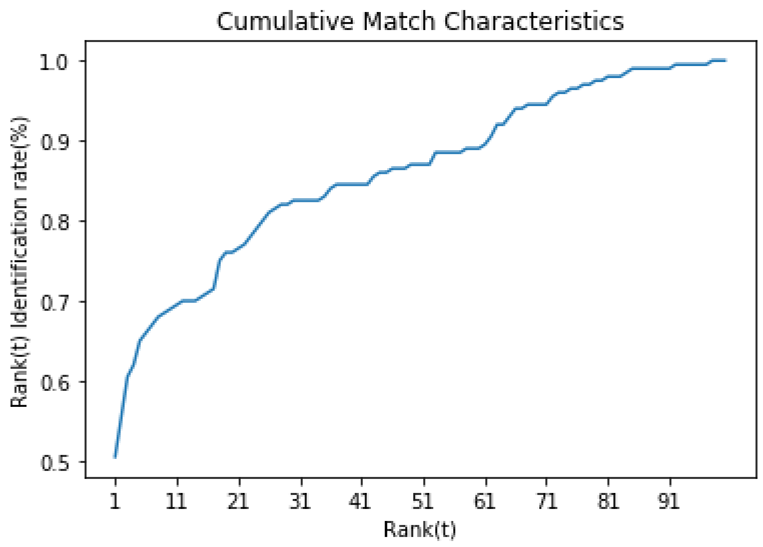


Bottom half

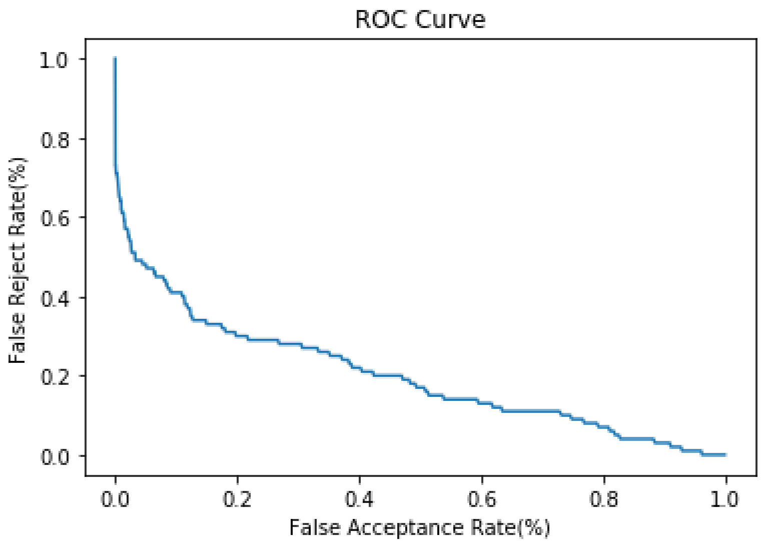
In [12]:
```
Gal=[]
Dist_p1=np.zeros((100,100))
Dist_p2=np.zeros((100,100))
for i in range(100):
    Img=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/GallerySet/subject'+
    for j in range(100):
        Img1=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec
        Img2=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec

        #c = np.correlate(a, b, 'full')
        cor1 = np.max(normxcorr2(Img,Img1))
        cor2 = np.max(normxcorr2(Img,Img2))

        #cor2 = signal.correlate2d (Img,Img2)
        Dist_p1[j,i]=cor1
        Dist_p2[j,i]=cor2
Gen=np.append(np.diag(Dist_p1),np.diag(Dist_p1))

non_diag = np.ones(shape=Dist_p1.shape, dtype=bool) ^ np.identity(len(Dist_p1)).
Imp_a=Dist_p1[non_diag==True]
non_diag = np.ones(shape=Dist_p2.shape, dtype=bool) ^ np.identity(len(Dist_p2)).
Imp_b=Dist_p2[non_diag==True]
Imp=np.append(Imp_a,Imp_b)
s1 = sns.distplot(Imp, kde=True,
             bins=int(180/5), color = 'red')

s2 = sns.distplot(Gen, kde=True,
             bins=int(180/5), color = 'blue')

plt.xlabel('Match Score')



plt.figure()
T=[]
P_a=[]
for t in range(0,100):
    T.append(t+1)
    temp=0
    for i in range(0,100):
        if i in Dist_p1[i].argsort()[-(t+1):][::-1]:
            temp+=1
        if i in Dist_p2[i].argsort()[-(t+1):][::-1]:
            temp+=1

    p=temp/200
    P_a.append(p)

plt.plot(T,P_a)
plt.xlabel('Rank(t)')
plt.ylabel('Rank(t) Identification rate(%)')
plt.xticks(np.arange(1, 100, step=10))
plt.title('Cumulative Match Characteristics')
plt.show()

plt.figure()
```

```
T_a=np.arange(0, 1, 0.001)
FAR=[]
FRR=[]
for t in T_a:
    far=len(Imp[Imp>t])/len(Imp)
    FAR.append(far)
    frr=len(Gen[Gen<t])/len(Gen)
    FRR.append(frr)

FAR_a=np.array(FAR)
FRR_a=np.array(FRR)

plt.plot(FAR_a,FRR_a)
plt.xlabel('False Acceptance Rate(%)')
plt.ylabel('False Reject Rate(%)')
plt.title('ROC Curve')

Gen_mean=np.mean(Gen)
Imp_mean=np.mean(Imp)
Gen_var=np.var(Gen)
Imp_var=np.var(Imp)

d_a = np.sqrt(2)*np.absolute(Gen_mean-Imp_mean)/np.sqrt(Gen_var+Imp_var)
print("d_a=",d_a)

IDX=np.argmin(np.absolute(FAR_a-FRR_a))
EER=FRR_a[IDX]
O_p=T_a[IDX]
print("Error_rate=",EER)
print("Operating_pt=",O_p)
```





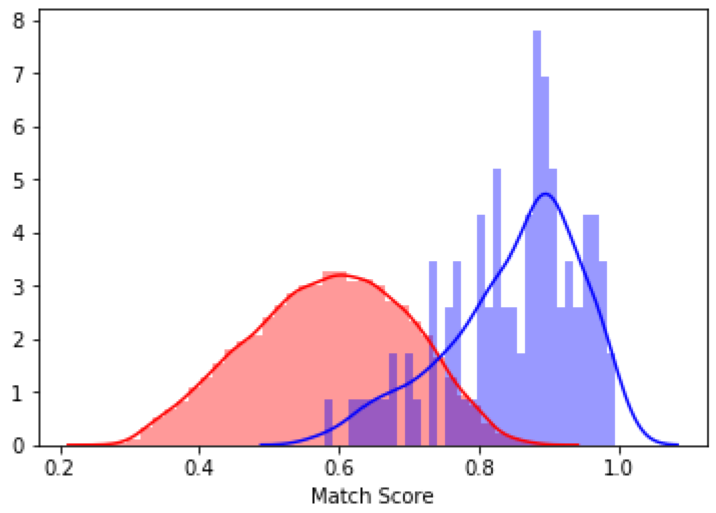
```
d_a= 1.2694841790565468
Error_rate= 0.28
Operating_pt= 0.5740000000000001
```

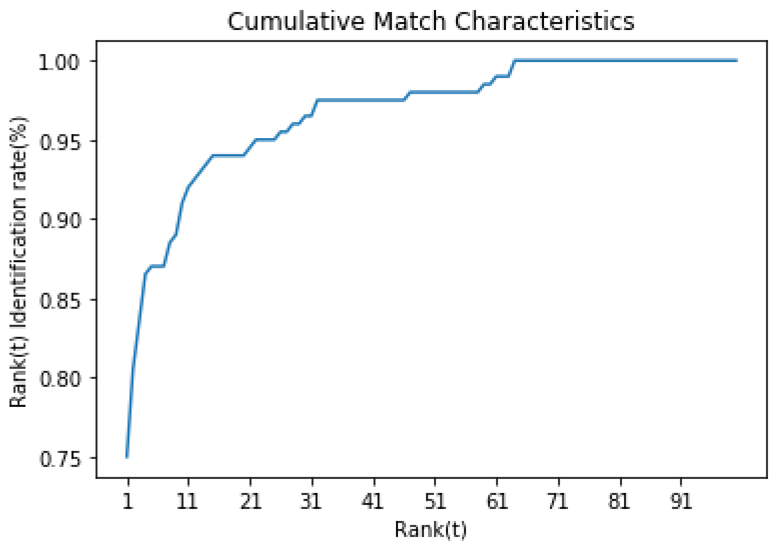


Left half

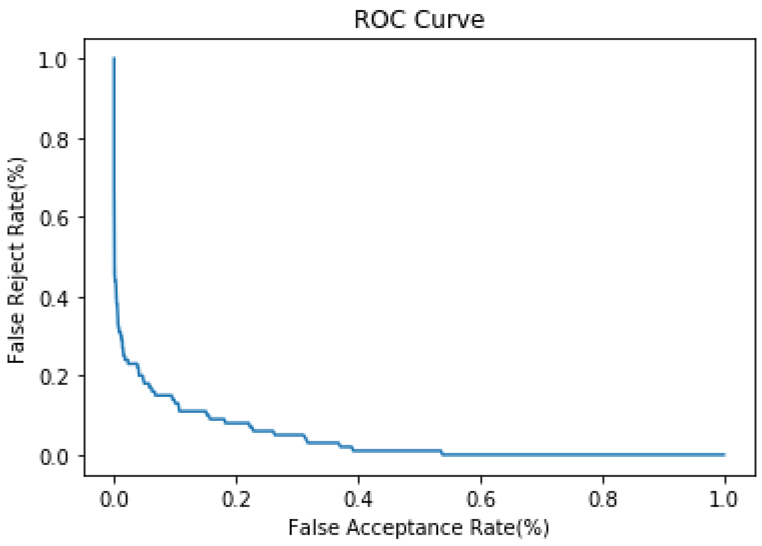
In [13]:
```
Gal=[]
Dist_p1=np.zeros((100,100))
Dist_p2=np.zeros((100,100))
for i in range(100):
    Img=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/GallerySet/subject'+
    for j in range(100):
        Img1=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec
        Img2=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec

        #c = np.correlate(a, b, 'full')
        cor1 = np.max(normxcorr2(Img,Img1))
        cor2 = np.max(normxcorr2(Img,Img2))

        #cor2 = signal.correlate2d (Img,Img2)
        Dist_p1[j,i]=cor1
        Dist_p2[j,i]=cor2
Gen=np.append(np.diag(Dist_p1),np.diag(Dist_p1))

non_diag = np.ones(shape=Dist_p1.shape, dtype=bool) ^ np.identity(len(Dist_p1)).
Imp_a=Dist_p1[non_diag==True]
non_diag = np.ones(shape=Dist_p2.shape, dtype=bool) ^ np.identity(len(Dist_p2)).
Imp_b=Dist_p2[non_diag==True]
Imp=np.append(Imp_a,Imp_b)
s1 = sns.distplot(Imp, kde=True,
             bins=int(180/5), color = 'red')

s2 = sns.distplot(Gen, kde=True,
             bins=int(180/5), color = 'blue')

plt.xlabel('Match Score')



plt.figure()
T=[]
P_a=[]
for t in range(0,100):
    T.append(t+1)
    temp=0
    for i in range(0,100):
        if i in Dist_p1[i].argsort()[-(t+1):][::-1]:
            temp+=1
        if i in Dist_p2[i].argsort()[-(t+1):][::-1]:
            temp+=1

    p=temp/200
    P_a.append(p)

plt.plot(T,P_a)
plt.xlabel('Rank(t)')
plt.ylabel('Rank(t) Identification rate(%)')
plt.xticks(np.arange(1, 100, step=10))
plt.title('Cumulative Match Characteristics')
plt.show()

plt.figure()
```

```
T_a=np.arange(0, 1, 0.001)
FAR=[]
FRR=[]
for t in T_a:
    far=len(Imp[Imp>t])/len(Imp)
    FAR.append(far)
    frr=len(Gen[Gen<t])/len(Gen)
    FRR.append(frr)

FAR_a=np.array(FAR)
FRR_a=np.array(FRR)

plt.plot(FAR_a,FRR_a)
plt.xlabel('False Acceptance Rate(%)')
plt.ylabel('False Reject Rate(%)')
plt.title('ROC Curve')

Gen_mean=np.mean(Gen)
Imp_mean=np.mean(Imp)
Gen_var=np.var(Gen)
Imp_var=np.var(Imp)

d_a = np.sqrt(2)*np.absolute(Gen_mean-Imp_mean)/np.sqrt(Gen_var+Imp_var)
print("d_a=",d_a)

IDX=np.argmin(np.absolute(FAR_a-FRR_a))
EER=FRR_a[IDX]
O_p=T_a[IDX]
print("Error_rate=",EER)
print("Operating_pt=",O_p)
```





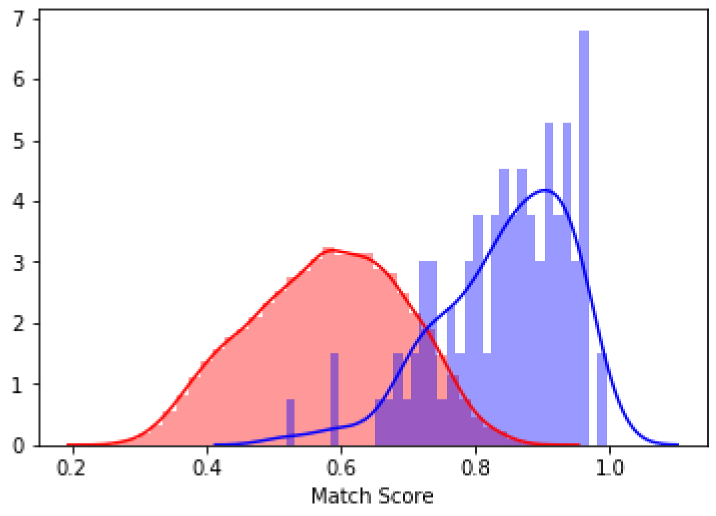
```
d_a= 2.5380856066204722
Error_rate= 0.11
Operating_pt= 0.727
```

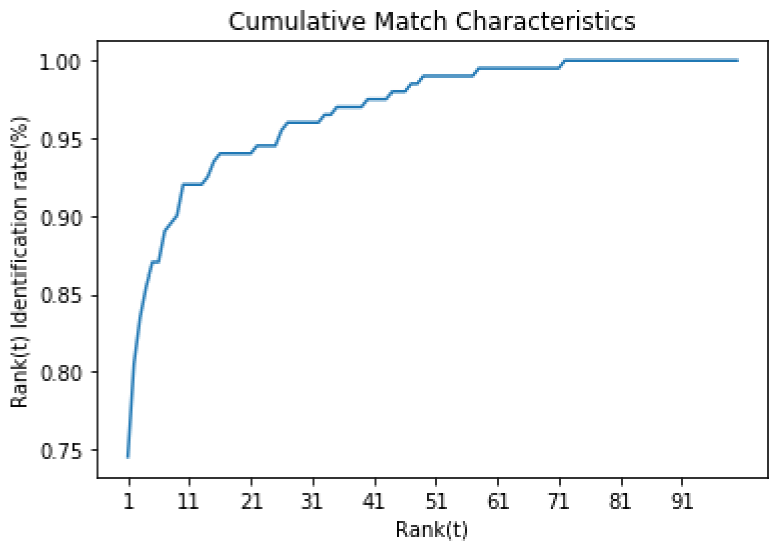


Right half

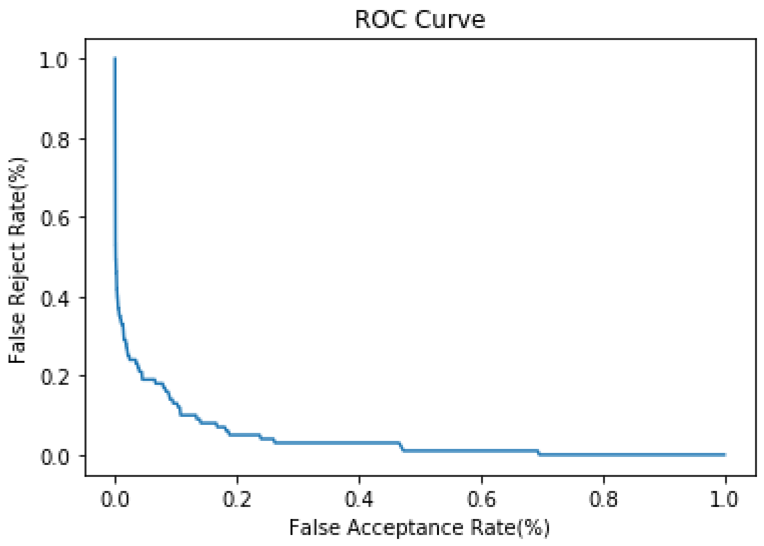
In [14]:
```
Gal=[]
Dist_p1=np.zeros((100,100))
Dist_p2=np.zeros((100,100))
for i in range(100):
    Img=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/GallerySet/subject'+
    for j in range(100):
        Img1=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec
        Img2=cv2.imread('C:/Users/varan/Downloads/Biometrics-HW2/ProbeSet/subjec

        #c = np.correlate(a, b, 'full')
        cor1 = np.max(normxcorr2(Img,Img1))
        cor2 = np.max(normxcorr2(Img,Img2))

        #cor2 = signal.correlate2d (Img,Img2)
        Dist_p1[j,i]=cor1
        Dist_p2[j,i]=cor2
Gen=np.append(np.diag(Dist_p1),np.diag(Dist_p1))

non_diag = np.ones(shape=Dist_p1.shape, dtype=bool) ^ np.identity(len(Dist_p1)).
Imp_a=Dist_p1[non_diag==True]
non_diag = np.ones(shape=Dist_p2.shape, dtype=bool) ^ np.identity(len(Dist_p2)).
Imp_b=Dist_p2[non_diag==True]
Imp=np.append(Imp_a,Imp_b)
s1 = sns.distplot(Imp, kde=True,
             bins=int(180/5), color = 'red')

s2 = sns.distplot(Gen, kde=True,
             bins=int(180/5), color = 'blue')

plt.xlabel('Match Score')



plt.figure()
T=[]
P_a=[]
for t in range(0,100):
    T.append(t+1)
    temp=0
    for i in range(0,100):
        if i in Dist_p1[i].argsort()[-(t+1):][::-1]:
            temp+=1
        if i in Dist_p2[i].argsort()[-(t+1):][::-1]:
            temp+=1

    p=temp/200
    P_a.append(p)

plt.plot(T,P_a)
plt.xlabel('Rank(t)')
plt.ylabel('Rank(t) Identification rate(%)')
plt.xticks(np.arange(1, 100, step=10))
plt.title('Cumulative Match Characteristics')
plt.show()

plt.figure()
```

```
T_a=np.arange(0, 1, 0.001)
FAR=[]
FRR=[]
for t in T_a:
    far=len(Imp[Imp>t])/len(Imp)
    FAR.append(far)
    frr=len(Gen[Gen<t])/len(Gen)
    FRR.append(frr)

FAR_a=np.array(FAR)
FRR_a=np.array(FRR)

plt.plot(FAR_a,FRR_a)
plt.xlabel('False Acceptance Rate(%)')
plt.ylabel('False Reject Rate(%)')
plt.title('ROC Curve')

Gen_mean=np.mean(Gen)
Imp_mean=np.mean(Imp)
Gen_var=np.var(Gen)
Imp_var=np.var(Imp)

d_a = np.sqrt(2)*np.absolute(Gen_mean-Imp_mean)/np.sqrt(Gen_var+Imp_var)
print("d_a=",d_a)

IDX=np.argmin(np.absolute(FAR_a-FRR_a))
EER=FRR_a[IDX]
O_p=T_a[IDX]
print("Error_rate=",EER)
print("Operating_pt=",O_p)
```





```
d_a= 2.508346162093557
Error_rate= 0.1
Operating_pt= 0.724
```



In [15]:
```
Tab_b=[['d-prime','EER'],[2.7,0.09],[2.95,0.04],[1.26,0.28],[2.5,0.11],[2.5,0.1]
pd.DataFrame(np.array(Tab_b).T, columns=["", "Full face","Top half","Bottom half'
```

Out[15]:

		Full face	Top half	Bottom half	Left half	Right half
0	d-prime	2.7	2.95	1.26	2.5	2.5
1	EER	0.09	0.04	0.28	0.11	0.1

Through observing performance of EER and d' values. -> The top part, left part and right part of images perform closer to full image.

->Top part of the face performs better

-> Yes, the performnace of the faces are as expected as the Inter pupular area has maximum uniqueness for every individual and the bottom part of the face decreases its performance. The d' value and EER shows that top part have higher interclass variance and lower intra class similarity.

->A weighted based ensemble approach can be adapted along with the normal face detection. Detection is performed on normal face and partial faces and overall score can be obtained by giving more weight to top part, then left and right parts and less weight to bottom part of face. The reason we dont consider only top part of face even though its performing better is due to the bad performance in the CMC curve. The genuine scores of few faces are very less in top part face, and also we loose lot of inofrmation which may decide during these times. That is why we have to use ensemble approach to solve the face recognition problem.