Support Vector Machine Example-1

Gender Recognition by Voice Acoustic Parameters

ယောက်ကျားအသံ နှင့် မိန်းမအသံ များကို အသံသွင်းထားသည့် အချက်အလက်များ ပါသည့် voice.csv ဆိုသည့် ဒေတာဖိုင်ကို အသုံးပြု၍ SVM ဖြင့် ယောက်ကျားအသံဖြစ်သည် သို့မဟုတ် မိန်းမအသံ ဖြစ်သည်ကို ခွဲခြားသည့် ကုဒ်များကို ရှင်းပြထားသည်။ အသံများ(voice and speech)တို့၏ acoustic properties ကို အခြေခံ၍ ယောက်ကျားအသံ သို့မဟုတ် မိန်းမအသံ ဖြစ်သည်ကို ခွဲခြားသည်။ recorded voice samples 3,168 ခု ပါရှိသည်။
Voice Data file, voice.csv(1.02 MB)
http://www.acmv.org/MachineLearning/svm/example1/voice.csv (http://www.acmv.org/MachineLearning/svm/example1/voice.csv)

Voice Data file, voice.csv(1.02 MB) https://www.kaggle.com/primaryobjects/voicegender (https://www.kaggle.com/primaryobjects/voicegender)

following acoustic properties များမှာ

meanfreq: mean frequency (in kHz)
sd: standard deviation of frequency
median: median frequency (in kHz)
Q25: first quantile (in kHz)
Q75: third quantile (in kHz)
IQR: interquantile range (in kHz)
skew: skewness (see note in specprop description)
kurt: kurtosis (see note in specprop description)
sp.ent: spectral entropy
sfm: spectral flatness
mode: mode frequency
centroid: frequency centroid (see specprop)
peakf: peak frequency (frequency with highest energy)
meanfun: average of fundamental frequency measured across acoustic signal
minfun: minimum fundamental frequency measured across acoustic signal
maxfun: maximum fundamental frequency measured across acoustic signal
meandom: average of dominant frequency measured across acoustic signal
mindom: minimum of dominant frequency measured across acoustic signal
maxdom: maximum of dominant frequency measured across acoustic signal
dfrange: range of dominant frequency measured across acoustic signal
modindx: modulation index. Calculated as the accumulated absolute difference between adjacent measurements of fundamental
frequencies divided by the frequency range
label: male or female

Support Vector machine ကို sklearn ထဲမှ linear, gaussian, polynomial စသည့် kernel (၃)မျိုးဖြင့် သရုပ်ဖော် run ပြထားသည်။ best performing model ရရှိစေရန်အတွက် C , gamma နှင့် degree စသည့် parameter (၃)မျိုးကို တန်ဖိုးမည်မျှ ဖြစ်သင့်သည်ကို ရှာဖွေပြထားသည်။

In [1]:
```
# This Python 3 environment comes with many helpful analytics libra
ries installed
# It is defined by the kaggle/python docker image: https://github.c
om/kaggle/docker-python
# For example, here's several helpful packages to load in


# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Ente
r) will list the files in the input directory

#from subprocess import check_output
#print(check_output(["ls", "../input"]).decode("utf8"))

# Any results you write to the current directory are saved as outpu
t.
```

(၁) SVM algorithm အတွက် လိုအပ်ည့် library များကို import လုပ်သည်။

Importing all the necessary libraries

pandas, numpy , seaborn နှင့် matplotlib.pyplot တို့ကို import လုပ်သည်။

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns

import matplotlib.pyplot as plt


%matplotlib inline
```

(၂) Comma Separated Values(.csv) ဖိုင်ကို ဖတ်၍ dataframe ပြုလုပ်သည်။

In [3]:
```
df = pd.read_csv('voice.csv')
df.head()
```

Out[3]:

	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	
0	0.059781	0.064241	0.032027	0.015071	0.090193	0.075122	12.863462	274.402906	0
1	0.066009	0.067310	0.040229	0.019414	0.092666	0.073252	22.423285	634.613855	0
2	0.077316	0.083829	0.036718	0.008701	0.131908	0.123207	30.757155	1024.927705	0
3	0.151228	0.072111	0.158011	0.096582	0.207955	0.111374	1.232831	4.177296	0
4	0.135120	0.079146	0.124656	0.078720	0.206045	0.127325	1.101174	4.333713	0

5 rows × 21 columns

(၃) Checking the correlation between each feature

feature တစ်ခုချင်းစီတို့၏ ဆက်စပ်မှု(correlation)ကို စစ်ဆေးသည်။

In [4]:
```
df.corr()
```

Out[4]:

	meanfreq	sd	median	Q25	Q75	IQR	skew	
meanfreq	1.000000	-0.739039	0.925445	0.911416	0.740997	-0.627605	-0.322327	-0.3
sd	-0.739039	1.000000	-0.562603	-0.846931	-0.161076	0.874660	0.314597	0.3
median	0.925445	-0.562603	1.000000	0.774922	0.731849	-0.477352	-0.257407	-0.2
Q25	0.911416	-0.846931	0.774922	1.000000	0.477140	-0.874189	-0.319475	-0.3
Q75	0.740997	-0.161076	0.731849	0.477140	1.000000	0.009636	-0.206339	-0.1
IQR	-0.627605	0.874660	-0.477352	-0.874189	0.009636	1.000000	0.249497	0.3
skew	-0.322327	0.314597	-0.257407	-0.319475	-0.206339	0.249497	1.000000	0.9
kurt	-0.316036	0.346241	-0.243382	-0.350182	-0.148881	0.316185	0.977020	1.0
sp.ent	-0.601203	0.716620	-0.502005	-0.648126	-0.174905	0.640813	-0.195459	-0.1
sfm	-0.784332	0.838086	-0.661690	-0.766875	-0.378198	0.663601	0.079694	0.1
mode	0.687715	-0.529150	0.677433	0.591277	0.486857	-0.403764	-0.434859	-0.4
centroid	1.000000	-0.739039	0.925445	0.911416	0.740997	-0.627605	-0.322327	-0.3
meanfun	0.460844	-0.466281	0.414909	0.545035	0.155091	-0.534462	-0.167668	-0.1
minfun	0.383937	-0.345609	0.337602	0.320994	0.258002	-0.222680	-0.216954	-0.2
maxfun	0.274004	-0.129662	0.251328	0.199841	0.285584	-0.069588	-0.080861	-0.0
meandom	0.536666	-0.482726	0.455943	0.467403	0.359181	-0.333362	-0.336848	-0.3
mindom	0.229261	-0.357667	0.191169	0.302255	-0.023750	-0.357037	-0.061608	-0.1
maxdom	0.519528	-0.482278	0.438919	0.459683	0.335114	-0.337877	-0.305651	-0.2
dfrange	0.515570	-0.475999	0.435621	0.454394	0.335648	-0.331563	-0.304640	-0.2
modindx	-0.216979	0.122660	-0.213298	-0.141377	-0.216475	0.041252	-0.169325	-0.2

(၄) Checking whether there is any null values

Missing value များ သို့မဟုတ် null value ရှိ မရှိ စစ်ဆေးသည်။

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
meanfreq    0
sd          0
median      0
Q25         0
Q75         0
IQR         0
skew        0
kurt        0
sp.ent      0
sfm         0
mode        0
centroid    0
meanfun     0
minfun      0
maxfun      0
meandom     0
mindom      0
maxdom      0
dfrange     0
modindx     0
label       0
dtype: int64
```

In [6]:
```
# dataframe ၏ row နှင့် ကော်လံ အရေအတွက်ကို ဖတ်သည်။
print('Shape of DataFrame(rows,colums):',df.shape)
```

```
Shape of DataFrame(rows,colums): (3168, 21)
```

Dataset ထဲတွင် sample 3168 ခု ပါဝင်ပြီး 21 feature ၂၁ ခု ပါဝင်သည်။ (There are 21 features and 3168 instances.)

Dataset ထဲမှ label ထဲတွင် ယောက်ကျားအသံ မိန်းမ အသံ မည်မျှ ရှိသည်ကို စစ်ကြည့်သည်။

In [7]:
```
# Counting Number of Male and Number of Female in Data
print("Total number of labels: {}".format(df.shape[0]))
print("Number of male: {}".format(df[df.label == 'male'].shape[0]))
print("Number of female: {}".format(df[df.label == 'female'].shape
[0]))

# print('Number of Males:',df[df['label']=='male'].shape[0])
# print('Number of Femles:',df[df['label']=='female'].shape[0])
```

```
Total number of labels: 3168
Number of male: 1584
Number of female: 1584
```

Dataset ထဲမှ label ထဲတွင် ယောက်ကျားအသံ, မိန်းမအသံ အရေအတွက် တူညီကြသည်။ (equal number of male and female labels)

(၅) Feature နှင့် Label များ ကို ခွဲထုတ်သည်။ (Separating features and labels)

Feature များသည် X ဖြစ်သည်။ Label များသည် y ဖြစ်သည်။

In [8]:
```
# printing keys or features of Voice DataFrame
print('Features of DataFrame: \n', df.keys())
```

```
Features of DataFrame:
 Index(['meanfreq', 'sd', 'median', 'Q25', 'Q75', 'IQR', 'skew',
'kurt',
       'sp.ent', 'sfm', 'mode', 'centroid', 'meanfun', 'minfun',
'maxfun',
       'meandom', 'mindom', 'maxdom', 'dfrange', 'modindx', 'labe
l'],
      dtype='object')
```

In [9]:
```
X=df.iloc[:, :-1]
X.head()      # ပထမဆုံး (၅) ခုကို ကြည့်သည်။
```

Out[9]:

	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	
0	0.059781	0.064241	0.032027	0.015071	0.090193	0.075122	12.863462	274.402906	0
1	0.066009	0.067310	0.040229	0.019414	0.092666	0.073252	22.423285	634.613855	0
2	0.077316	0.083829	0.036718	0.008701	0.131908	0.123207	30.757155	1024.927705	0
3	0.151228	0.072111	0.158011	0.096582	0.207955	0.111374	1.232831	4.177296	0
4	0.135120	0.079146	0.124656	0.078720	0.206045	0.127325	1.101174	4.333713	0

(၆) Converting string value to int type for labels

sklearn.preprocessing မှ LabelEncoder ကို အသုံးပြုရှိ label များတွင် male နှင့် female (string) ဟုဖော်ပြထားသည်ကို 1 နှင့် 0 (Integer)အဖြစ် ပြောင်းသည်။

In [10]:
```
from sklearn.preprocessing import LabelEncoder
y=df.iloc[:,-1]

# Encode label category
# male -> 1
# female -> 0

gender_encoder = LabelEncoder()
y = gender_encoder.fit_transform(y)
y
```

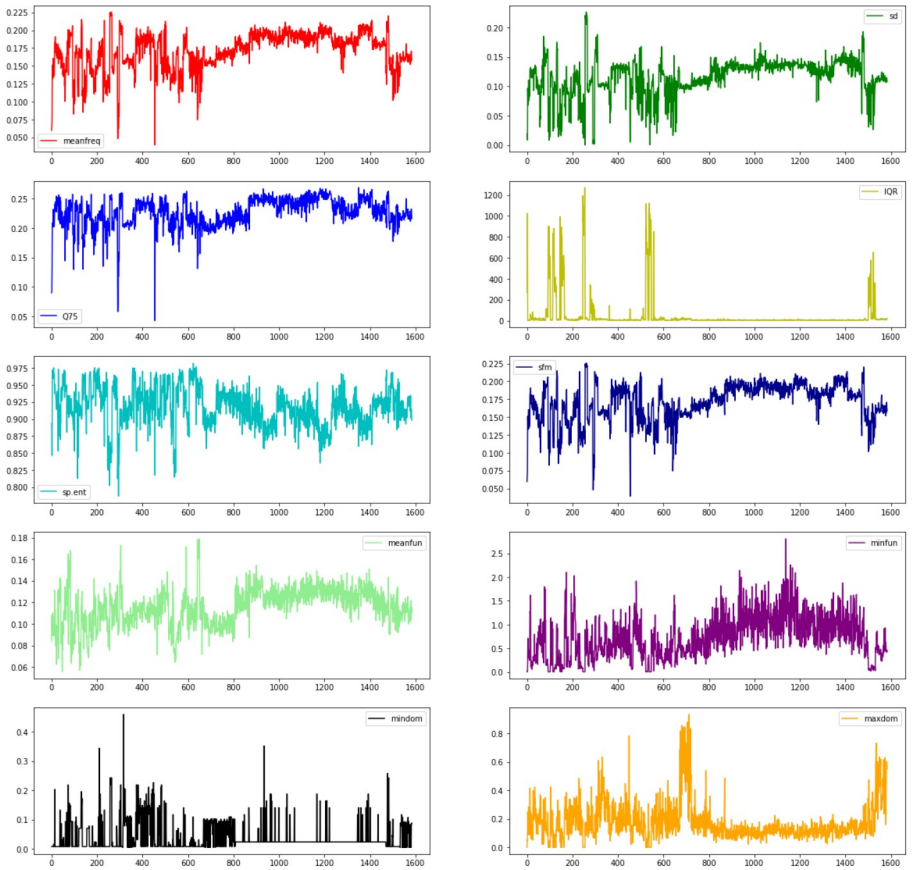
Out[10]:
```
array([1, 1, 1, ..., 0, 0, 0])
```

(၇)ဂရပ်ပုံများဆွဲ၍ Data Visualization လုပ်သည်။

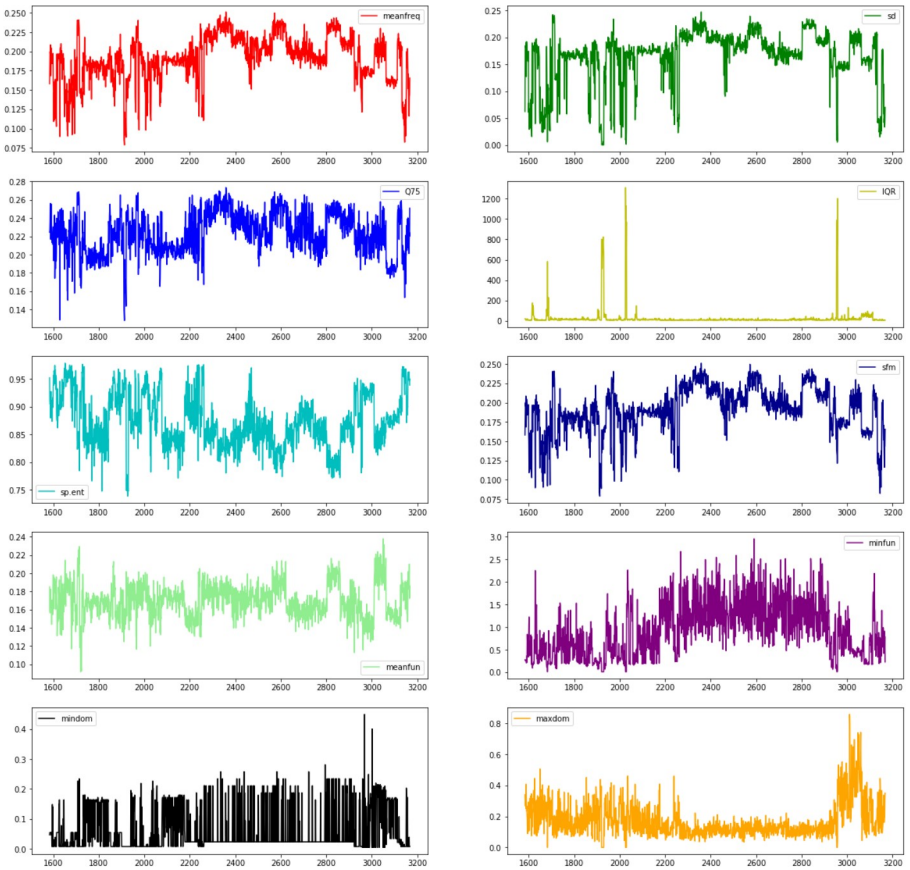
feature 20 ကို ၁၀ခုစီခွဲ၍ ဂရပ်ပုံများဆွဲကြည့်သည်။ 10 features out of 20 of voice (Music) DataFrame

In [11]:
```
# Function to Plotting 10 features out of 20 of voice (Music)

def Plotting_Features(Fun,f):

    i=0 # initial index of features
    j=0 # initial index of color

    color = ['r','g','b','y','c','darkblue','lightgreen',
             'purple','k','orange','olive'] # colors for plots

    # Number of rows
    nrows =5

    # Creating Figure and Axis to plot
    fig, axes = plt.subplots(nrows,2)

    # Setting Figure size
    fig.set_figheight(20)
    fig.set_figwidth(20)

    for row in axes:

        plot1 = Fun[f[i]]
        plot2 = Fun[f[i+3]]

        col = [color[j],color[j+1]]
        label = [f[i],f[i+1]]

        plot(row, plot1,plot2,col,label)

        i=i+4

        j=j+2

    plt.show()

def plot(axrow, plot1, plot2, col, label):

    axrow[0].plot(plot1,label=label[0],color=col[0])
    axrow[0].legend()

    axrow[1].plot(plot2,label=label[1],color=col[1])
    axrow[1].legend()
```

In [12]:
```
# Setting Male Acoustic Parameters
Male = df[df['label']=='male']
Male = Male.drop(['label'],axis=1)
features = Male.keys()
Plotting_Features(Male,features)
```



In [13]:
```
# Setting female Acoustic Parameters
Female = df[df['label']=='female']
Female = Female.drop(['label'],axis=1)
features = Female.keys()
Plotting_Features(Female,features)
```



(၈) Data Standardisation လုပ်သည်။

Standardization refers to shifting the distribution of each attribute to have a mean of zero and a standard deviation of one (unit variance). It is useful to standardize attributes for a model. Standardization of datasets is a common requirement for many machine learning estimators implemented in scikit-learn; they might behave badly if the individual features do not more or less look like standard normally distributed data.

In [14]:
```
# Scale the data to be between -1 and 1
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X)
X = scaler.transform(X)
```

(၉) Splitting dataset into training set and testing set for better generalisation

generalisation ကောင်းစေရန် အတွက် dataset ကို raining set(X_train, X_test) နှင့် testing set(y_train, y_test) အဖြစ် ခွဲသည်။ test_size=0.2 ဆိုသည်မှာ testing set အရေအတွက် dataset တစ်ခုလုံး၏ ၂၀% ဖြစ်သည်။

In [15]:
```
# splitting Data into training and testing Data using cross_validat
ion.train_test_split
# Train - Test data ratio of 80%-20%
# Random State to Randomize data = 1

# from sklearn.cross_validation import train_test_split
from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
=0.2, random_state=1)

# အောက်တွင်  Train - Test data ratio of 75%-25%
#X_train,X_test,Y_train,Y_test = train_test_split(X, y,test_size =
0.25, random_state=123)
```

(၁၀) SVM တွင် default hyperparameter များကိုသာ သုံး၍ Accuracy Score ကို တွက်သည်။

default hyperparameter များဖြင့်သာ ဘာမှ မပြောင်းဘဲ SVM ကို run မည်။ sklearn.svm မှ SVC ကို import လုပ်သည်။ metrics ကို import လုပ်သည်။

In [16]:
```
# importing Support Vector Machine Algorithm for Prediction
from sklearn.svm import SVC
from sklearn import metrics

svc=SVC() # Default hyperparameters ကိုသာ သုံးမည် ဖြစ်သောကြောင့် () ထဲတွင်
ဘာမျှ မထည့်ပါ။

# အောက်က ကုဒ်သည် C = 200 နှင့် gamma = 0.1 တို့ထည့်၍ Classifier creat လုပ်
သည့် ကုဒ်ဖြစ်သည်။
# svm = SVC(C = 200, gamma = 0.1)

svc.fit(X_train,y_train)     # svc model တည်ဆောက်သည်။
y_pred=svc.predict(X_test)   # တည်ဆောက်ထားသည့် svc model ထဲတွင် X_tes
t ထည့်၍ ခန့်မှန်းကြည်သည်။

print('Accuracy Score:')
print(metrics.accuracy_score(y_test,y_pred))  # Accuracy Score ကို
ကြည့်သည်။
```

```
Accuracy Score:
0.9763406940063092
```

In [17]:
```
#Accuracy Score ကို အောက်တွင် ဖော်ပြထားသည့်အတိုင်းလည်း တွက်နိုင်သည်။
"""
from sklearn.metrics import accuracy_score
#Accuracy = accuracy_score(y_pred,y_test)
Accuracy = accuracy_score(y_test,y_pred)
print('Accuracy Score:\n',Accuracy)

"""
```

Out[17]:
```
" \nfrom sklearn.metrics import accuracy_score\n#Accuracy = accur
acy_score(y_pred,y_test)\nAccuracy = accuracy_score(y_test,y_pre
d)\nprint('Accuracy Score:\n',Accuracy)\n\n"
```

(၁၁)အမျိုးမျိုးသော Kernel တို့တွင် Cross-Validation လုပ်သည်။

'linear','rbf','poly' စသည့် Kernel (၃)ခု၏ Cross-Validation score ကို for loop ဖြင့် တွက်ပြီး နှိုင်းယှဉ်သည်။

In [18]:
```
# importing cross_val_score to calculate score
# from sklearn.cross_validation import cross_val_score
from sklearn.model_selection import cross_val_score
```

In [19]:
```
# Defining three different kernels
kernels = ['linear','rbf','poly']

score = []

for i in kernels:
    svc=SVC(kernel = i)
    Accuracy = cross_val_score(svc,X_train,y_train,cv = 15, scoring
='accuracy')
    score.append(Accuracy.mean())
for i in range(len(kernels)):
    print(kernels[i],':',score[i])
```

```
linear : 0.973555931248239
rbf : 0.9798699164083778
poly : 0.9577791866253406
```

(၁၂) အမျိုးမျိုးသော C တန်ဖိုးများ အတွက် Cross-Validation လုပ်သည်။

In [20]:
```
score = []
for i in range(10):
    # clf = svm.SVC(C = i+1)
    svc=SVC(C = i+1)
    Accuracy = cross_val_score(svc, X_train, y_train, cv = 15, scor
ing='accuracy')
    score.append(Accuracy.mean())

for i in range(10):
    print('C =',i+1,': Score =',score[i])
```

```
C = 1 : Score = 0.9798699164083778
C = 2 : Score = 0.9822367803137033
C = 3 : Score = 0.9814478256785949
C = 4 : Score = 0.9826359537897998
C = 5 : Score = 0.9822414764722456
C = 6 : Score = 0.9822391283929744
C = 7 : Score = 0.983028083028083
C = 8 : Score = 0.9810556964403119
C = 9 : Score = 0.981450173757866
C = 10 : Score = 0.981450173757866
```

(၁၃) အမျိုးမျိုးသော gamma တန်ဖိုးများ အတွက် Cross-Validation လုပ်သည်။

In [21]:
```
score = []
gamma_values = [0.0001,0.001,0.01,0.1,1.0,100.0,1000.0]
for i in gamma_values:
    svc=SVC(gamma = i)
    Accuracy = cross_val_score(svc,X_train, y_train, cv = 15, scori
ng='accuracy')
    score.append(Accuracy.mean())
for i in range(len(gamma_values)):
    print('gamma:',gamma_values[i],': Score:',score[i])
```

```
gamma: 0.0001 : Score: 0.8906804733727811
gamma: 0.001 : Score: 0.9629073917535458
gamma: 0.01 : Score: 0.9759274913121065
gamma: 0.1 : Score: 0.9814478256785948
gamma: 1.0 : Score: 0.972372499295576
gamma: 100.0 : Score: 0.5031558185404339
gamma: 1000.0 : Score: 0.5031558185404339
```

(၁၄) Linear kernel တွင် ရမည့် Accuracy Score ကို တွက်သည်။

In [22]:
```
svc=SVC(kernel='linear')    #  Linear kernel ကို အသုံးပြုမည်။

# Classifier ကို training လုပ်သည်။ သို့မဟုတ် တည်ဆောက်ပြီးသည့် model ကို tra
in လုပ်သည်။
svc.fit(X_train,y_train)
```

Out[22]:
```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, c
oef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kerne
l='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=
True,
    tol=0.001, verbose=False)
```

အပေါ်မှ တန်းဖိုးများသည် SVM model တည်ဆောက်ထားသည့် value များ ဖြစ်သည်။

In [23]:
```
# predicting our test data
y_pred=svc.predict(X_test)
y_pred  # တည်ဆောက်ထားသည့် SVM model ခန့်မှန်းပေးသည့် ရလဒ်ကို ကြည့်သည်။
```

Out[23]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1,
0, 0, 1,
       0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0,
1, 1, 1,
       0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1,
1, 0, 1,
       0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0,
0, 1, 1,
       1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0,
0, 0, 0,
       1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1,
1, 0, 1,
       0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1,
0, 0, 0,
       0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
0, 0, 1,
       1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1,
1, 0, 0,
       1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1,
0, 0, 1,
       1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 0,
       0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0,
1, 1, 1,
       1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0,
1, 1, 1,
       1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1,
0, 1, 0,
       0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0,
0, 0, 0,
       0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1,
0, 1, 1,
       0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1,
1, 1, 1,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0,
1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0,
1, 0, 1,
       0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
0, 1, 0,
       1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1,
0, 1, 0,
       0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1,
1, 0, 1,
       1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1,
1, 1, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0,
1, 1, 0,
       1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0,
       1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0,
1, 1, 0,
       1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
0, 0, 1,
       1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1,
0, 1, 1,
       1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1])
```

In [24]:
```
print('Accuracy Score:')
print(metrics.accuracy_score(y_test,y_pred))
```

```
Accuracy Score:
0.9779179810725552
```

(၁၅) Radial basis function (RBF)kernel တွင် ရမည့် Accuracy Score ကို တွက်သည်။

In [25]:
```
svc=SVC(kernel='rbf')

svc.fit(X_train,y_train)
y_pred=svc.predict(X_test)

print('Accuracy Score:')
print(metrics.accuracy_score(y_test,y_pred))
```

```
Accuracy Score:
0.9763406940063092
```

We can conclude from above that svm by default uses **rbf** kernel as a parameter for kernel

(၁၆) Polynomial kernel တွင် ရမည့် Accuracy Score ကို တွက်သည်။

In [26]:
```
svc=SVC(kernel='poly')
svc.fit(X_train,y_train)
y_pred=svc.predict(X_test)
print('Accuracy Score:')
print(metrics.accuracy_score(y_test,y_pred))
```

```
Accuracy Score:
0.9589905362776026
```

Polynomial kernel is performing poorly.The reason behind this maybe it is overfitting the training dataset

(၁၇)Performing K-fold cross validation with different kernels

(၁၇.၁) Cross Validation on Linear kernel

Linear kernel တွင် Cross Validation လုပ်သည်။

In [27]:
```
#from sklearn.cross_validation import cross_val_score

from sklearn.model_selection import cross_val_score
svc=SVC(kernel='linear')

scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy') #cv
is cross validation
print(scores)
```

```
[0.91167192 0.97160883 0.97160883 0.97791798 0.95899054 0.9873817
 0.99369085 0.97791798 0.95253165 0.99367089]
```

We can see above how the accuracy score is different everytime.This shows that accuracy score depends upon how the datasets got split.

In [28]:
```
print(scores.mean())
```

```
0.9696991175178692
```

In K-fold cross validation we generally take the mean of all the scores.

(၁၇.၂) Cross Validation on rbf kernel

rbf kernel တွင် Cross Validation လုပ်သည်။

In [29]:
```
#from sklearn.cross_validation import cross_val_score
from sklearn.model_selection import cross_val_score
svc=SVC(kernel='rbf')
scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy') #cv
is cross validation
print(scores)
```

```
[0.93375394 0.95583596 0.96845426 0.96214511 0.96529968 0.9968454
3
 0.99053628 0.98422713 0.91455696 0.99367089]
```

In [30]:
```
print(scores.mean())
```

```
0.9665325639899376
```

(၁၇.၃) Cross Validation on Polynomial kernel

Polynomial kernel တွင် Cross Validation လုပ်သည်။

In [31]:
```
#from sklearn.cross_validation import cross_val_score
from sklearn.model_selection import cross_val_score
svc=SVC(kernel='poly')
scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy') #cv
is cross validation
print(scores)
```

```
[0.89274448 0.94952681 0.93059937 0.92744479 0.94952681 0.9936908
5
 0.98422713 0.96529968 0.87974684 0.9778481 ]
```

In [32]:
```
print(scores.mean())
```

```
0.9450654873617378
```

K-fold cross validation လုပ်ခဲ့ပြီး ဖြစ်သည်။ iteration တစ်ခုချင်းစီမှ ရရှိသည့် score များကို တွေ့နိုင်သည်။ train_test_split method ကို အသုံးပြုသည်။ train_test_split method သည် dataset ကို testing နှင့် training dataset အဖြစ်ခွဲသည့်အခါ random နည်းဖြင့်(random manner) ခွဲပေးသည်။

K-fold cross validation နည်းသည် dataset ကို အညီအမျှ (၁၀)ပိုင်း ပိုင်း(split into 10 equal parts) သောကြောင့် dataset တစ်ခုလုံးကို ခြုံငုံမိသည်။ ထို့ကြောင့် မတူညီသည့် accuracy score (၁၀)ခု ရရှိသည်။ (got 10 different accuracy score)

(၁၈) linear kernel တွင် C တန်ဖိုး အမျိုးမျိုးတို့၏ accuracy score ကို စစ်ဆေးကြည့်ရအောင်

C parameter သည် SVM optimization အား misclassifying လုပ်မိသည့် training example များကို မည်ကဲ့သို့ လုပ်ရမည်ကို ညွှန်ကြားပေးသည့် parameter ဖြစ်သည်။

C value တန်ဖိုး များလျှင် SVM optimization algorithm သည် margin ကျဉ်းသည့် hyperplane ကို ရွေးချယ်ပေးသည်။ training point များအားလုံးကို မှန်ကန်စွာ ခွဲပေးသည်။(training points classified correctly.)။ overfitting ဖြစ်နိုင်သည်။

C value တန်ဖိုး နည်းလျှင် SVM optimization algorithm သည် margin ကျယ်သည့် hyperplane ကို ရွေးချယ်ပေးသည်။ margin ကျယ်သောကြောင့် mis classified ဖြစ်သည့် point တစ်ခု တစ်လေ ရှိနိုင်သည်။ underfitting ဖြစ်နိုင်သည်။

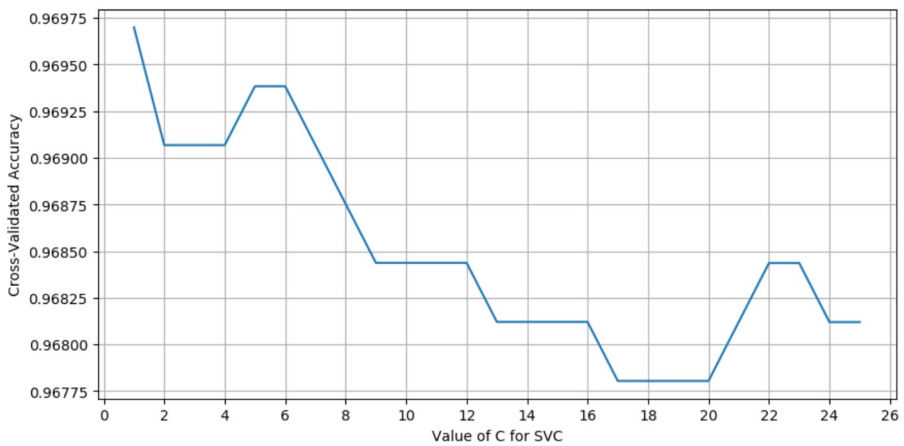
generalised အဖြစ်နိုင်ဆုံးသော C value ကို ရွေးချယ်သင့်သည်။

In [33]:
```
C_range=list(range(1,26))
acc_score=[]
for c in C_range:
    svc = SVC(kernel='linear', C=c)
    scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
    acc_score.append(scores.mean())
print(acc_score)
```

```
[0.9696991175178692, 0.969068202691371, 0.969068202691371, 0.9690
68202691371, 0.9693836601046201, 0.9693836601046201, 0.9690682026
91371, 0.9687527452781215, 0.9684372878648724, 0.968437287864872
4, 0.9684372878648724, 0.9684372878648724, 0.9681208321686698, 0.
9681208321686698, 0.9681208321686698, 0.9681208321686698, 0.96780
43764724673, 0.9678043764724673, 0.9678043764724673, 0.9678043764
724673, 0.9681208321686698, 0.968436289581919, 0.968436289581919,
0.9681198338857164, 0.9681198338857164]
```

(၁၈.၁) C values အမျိုးမျိုး(1 မှ 25 အထိ)တို့မှ ရသည့် accuracy score ကို ဂရပ်ပုံဆွဲသည်။

In [34]:
```
import matplotlib.pyplot as plt
%matplotlib inline

C_values=list(range(1,26))
# plot the value of C for SVM (x-axis) versus the cross-validated a
ccuracy (y-axis)

plt.rcParams['figure.figsize'] = [10, 5]
plt.rcParams['figure.dpi'] = 100

plt.plot(C_values,acc_score)
plt.xticks(np.arange(0,27,2))
plt.xlabel('Value of C for SVC')
plt.ylabel('Cross-Validated Accuracy')
plt.grid()
```



From the above plot we can see that accuracy has been close to 97% for C=1 and C=6 and then it drops around 96.8% and remains constant.

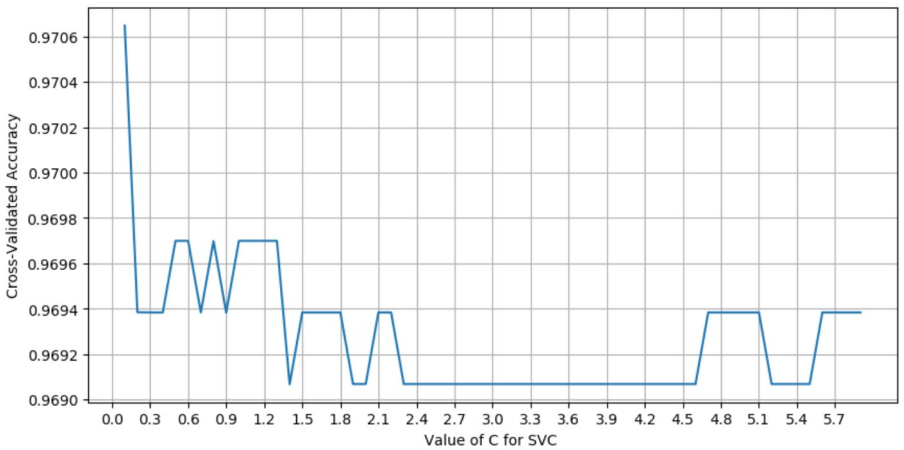
(၁၉) C တန်ဖိုး မည်မျှတွင် accuracy score အကောင်းဆုံး ရနိုင်သည်ကို လေ့လာကြရအောင်

Let us look into more detail of what is the exact value of C which is giving us a good accuracy score

In [35]:
```
C_range=list(np.arange(0.1,6,0.1))
acc_score=[]
for c in C_range:
    svc = SVC(kernel='linear', C=c)
    scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
    acc_score.append(scores.mean())
print(acc_score)
```

```
[0.9706474863235236, 0.9693846583875733, 0.9693836601046201, 0.96
93836601046201, 0.9696991175178692, 0.9696991175178692, 0.9693836
601046201, 0.9696981192349158, 0.9693826618216667, 0.969699117517
8692, 0.9696991175178692, 0.9696991175178692, 0.9696991175178692,
0.9690672044084174, 0.9693836601046201, 0.9693836601046201, 0.969
3836601046201, 0.9693836601046201, 0.969068202691371, 0.969068202
691371, 0.9693836601046201, 0.9693836601046201, 0.96906820269137
1, 0.969068202691371, 0.969068202691371, 0.969068202691371, 0.969
068202691371, 0.969068202691371, 0.969068202691371, 0.96906820269
1371, 0.969068202691371, 0.969068202691371, 0.969068202691371, 0.
969068202691371, 0.969068202691371, 0.969068202691371, 0.96906820
2691371, 0.969068202691371, 0.969068202691371, 0.969068202691371,
0.969068202691371, 0.969068202691371, 0.969068202691371, 0.969068
202691371, 0.969068202691371, 0.969068202691371, 0.96938366010462
01, 0.9693836601046201, 0.9693836601046201, 0.9693836601046201,
0.9693836601046201, 0.969068202691371, 0.969068202691371, 0.96906
8202691371, 0.969068202691371, 0.9693836601046201, 0.969383660104
6201, 0.9693836601046201, 0.9693836601046201]
```

In [36]:
```
C_values=list(np.arange(0.1,6,0.1))
# plot the value of C for SVM (x-axis) versus the cross-validated a
ccuracy (y-axis)
plt.plot(C_values,acc_score)
plt.xticks(np.arange(0.0,6,0.3))
plt.xlabel('Value of C for SVC ')
plt.ylabel('Cross-Validated Accuracy')
plt.grid()
```



Accuracy score is highest for C=0.1.

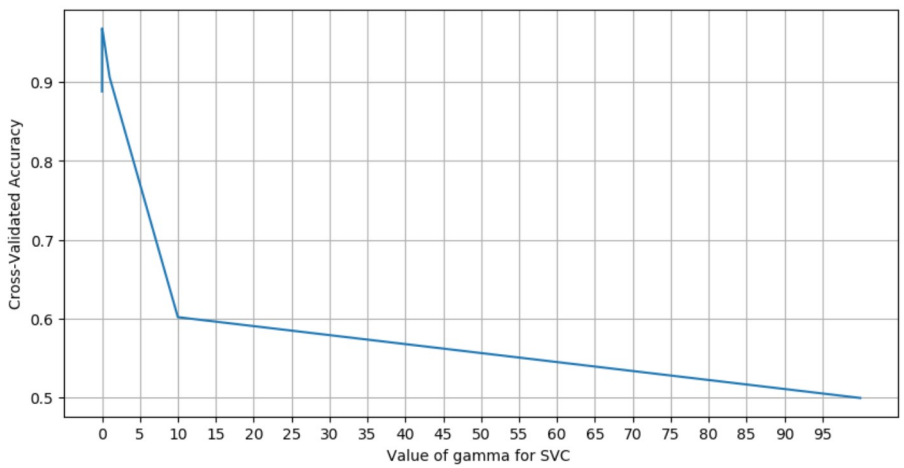
(၂၀) rbf kernel တွင် gamma တန်ဖိုး အမျိုးမျိုးတို့၏ accuracy score ကို လေ့လာကြစို့

Technically, the gamma parameter is the inverse of the standard deviation of the RBF kernel (Gaussian function), which is used as similarity measure between two points. Intuitively, a small gamma value define a Gaussian function with a large variance. In this case, two points can be considered similar even if are far from each other. In the other hand, a large gamma value means define a Gaussian function with a small variance and in this case, two points are considered similar just if they are close to each other

In [37]:
```
gamma_range=[0.0001,0.001,0.01,0.1,1,10,100]
acc_score=[]
for g in gamma_range:
    svc = SVC(kernel='rbf', gamma=g)
    scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
    acc_score.append(scores.mean())
print(acc_score)
```

```
[0.888240226809887, 0.9551820868106857, 0.9681168390368565, 0.963
6874575729744, 0.9061883560276325, 0.6016421754582119, 0.49905362
776025236]
```

In [38]:
```
gamma_range=[0.0001,0.001,0.01,0.1,1,10,100]

# plot the value of C for SVM (x-axis) versus the cross-validated a
ccuracy (y-axis)
plt.plot(gamma_range,acc_score)
plt.xlabel('Value of gamma for SVC ')
plt.xticks(np.arange(0.0001,100,5))
plt.ylabel('Cross-Validated Accuracy')
plt.grid()
```

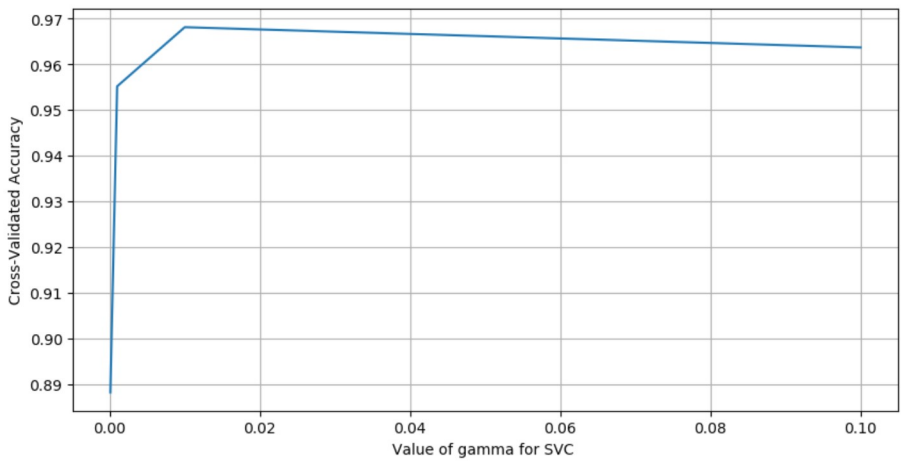


gamma=10 နှင့် 100 kernel ၏ performance သိပ်မကောင်းပါ။ gamma = 1 တွင် accuracy score အနည်းငယ် ကျသွားသည်။ gamma range 0.0001 မှ 0.1 အတွင်း လေ့လာကြရအောင်

In [39]:
```
gamma_range=[0.0001,0.001,0.01,0.1]
acc_score=[]
for g in gamma_range:
    svc = SVC(kernel='rbf', gamma=g)
    scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
    acc_score.append(scores.mean())
print(acc_score)
```

```
[0.888240226809887, 0.9551820868106857, 0.9681168390368565, 0.963
6874575729744]
```

In [40]:
```
gamma_range=[0.0001,0.001,0.01,0.1]

# plot the value of C for SVM (x-axis) versus the cross-validated a
ccuracy (y-axis)
plt.plot(gamma_range,acc_score)
plt.xlabel('Value of gamma for SVC ')
plt.ylabel('Cross-Validated Accuracy')
plt.grid()
```



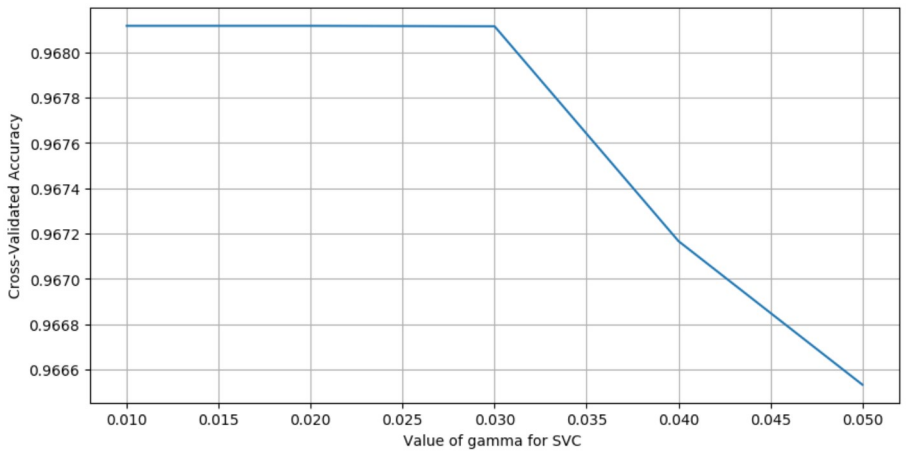
The score increases steadily and raches its peak at 0.01 and then decreases till gamma=1.Thus Gamma should be around 0.01.

(၂၁) gamma value အမျိုးမျိုးတို့ဖြင့် အသေးစိတ် လေ့လာကြရအောင်

In [41]:
```
gamma_range=[0.01,0.02,0.03,0.04,0.05]
acc_score=[]
for g in gamma_range:
    svc = SVC(kernel='rbf', gamma=g)
    scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
    acc_score.append(scores.mean())
print(acc_score)
```

```
[0.9681168390368565, 0.9681168390368565, 0.9681148424709501, 0.96
71664736652957, 0.9665325639899376]
```

In [42]:
```
gamma_range=[0.01,0.02,0.03,0.04,0.05]

# plot the value of C for SVM (x-axis) versus the cross-validated a
ccuracy (y-axis)
plt.plot(gamma_range,acc_score)
plt.xlabel('Value of gamma for SVC ')
plt.ylabel('Cross-Validated Accuracy')
plt.grid()
```



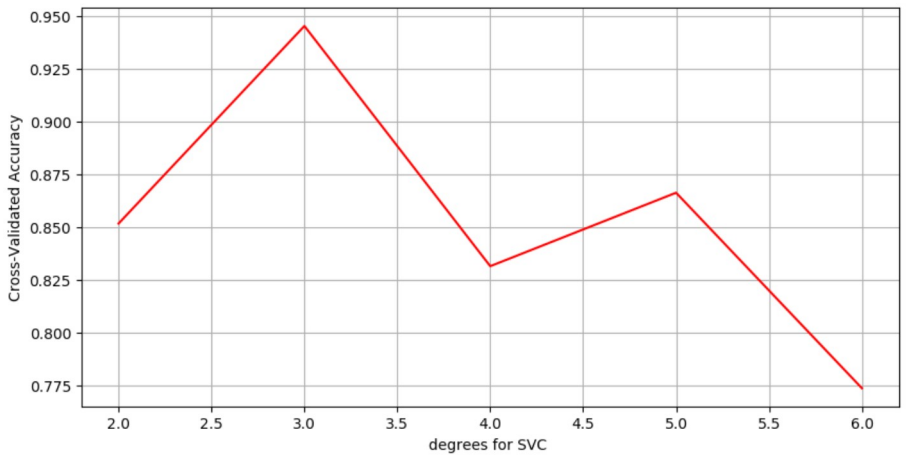
We can see there is constant decrease in the accuracy score as gamma value increase.Thus gamma=0.01 is the best parameter.

(၂၂) Polynomial kernel တွင် degree တန်ဖိုး အမျိုးမျိုးထည့်၍ accuracy score ကို ရှာသည်။

In [43]:
```
degree=[2,3,4,5,6]
acc_score=[]
for d in degree:
    svc = SVC(kernel='poly', degree=d)
    scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
    acc_score.append(scores.mean())
print(acc_score)
```

```
[0.8515842750469194, 0.9450654873617378, 0.8313989937307829, 0.86
61622010142555, 0.7736463283152977]
```

In [44]:
```
degree=[2,3,4,5,6]

# plot the value of C for SVM (x-axis) versus the cross-validated a
ccuracy (y-axis)
plt.plot(degree,acc_score,color='r')
plt.xlabel('degrees for SVC ')
plt.ylabel('Cross-Validated Accuracy')
plt.grid()
```



Score is high for third degree polynomial and then there is drop in the accuracy score as degree of polynomial increases.Thus increase in polynomial degree results in high complexity of the model and thus causes overfitting.

(၂၃) performing SVM by taking hyperparameter C=0.1 and kernel as linear

hyperparameter C=0.1 နှင့် linearkernel တို့ဖြင့် accuracy score ကို ရှာသည်။

In [45]:
```
from sklearn.svm import SVC
svc= SVC(kernel='linear',C=0.1)
svc.fit(X_train,y_train)
y_predict=svc.predict(X_test)
accuracy_score= metrics.accuracy_score(y_test,y_predict)
print(accuracy_score)
```

```
0.9747634069400631
```

hyperparameter C=0.1 နှင့် linearkernel တို့ဖြင့် accuracy score ကို ရှာသည်။ K-fold cross validation(where K=10)

In [46]:
```
#from sklearn.cross_validation import cross_val_score
from sklearn.model_selection import cross_val_score
svc=SVC(kernel='linear',C=0.1)
scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
print(scores)
```

```
[0.90851735 0.97160883 0.97476341 0.97791798 0.95899054 0.9905362
8
 0.99369085 0.97791798 0.95886076 0.99367089]
```

Taking the mean of all the scores

In [47]:
```
print(scores.mean())
```

```
0.9706474863235236
```

The accuracy is slightly good without K-fold cross validation but it may fail to generalise the unseen data.Hence it is advisable to perform K-fold cross validation where all the data is covered so it may predict unseen data well.

(၂၄) hyperparameter gamma=0.01 နှင့် rbf kernel တို့ဖြင့် accuracy score ကို ရှာသည်။

performing SVM by taking hyperparameter gamma=0.01 and kernel as rbf

In [48]:
```
from sklearn.svm import SVC
svc= SVC(kernel='rbf',gamma=0.01)
svc.fit(X_train,y_train)
y_predict=svc.predict(X_test)
metrics.accuracy_score(y_test,y_predict)
```

Out[48]:
```
0.9668769716088328
```

gamma=0.01 နှင့် linear kernel တို့ဖြင့် accuracy score ကို ရှာသည်။ K-fold cross validation(where K=10)

In [49]:
```
svc=SVC(kernel='linear',gamma=0.01)
scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
print(scores)
print(scores.mean())
```

```
[0.91167192 0.97160883 0.97160883 0.97791798 0.95899054 0.9873817
 0.99369085 0.97791798 0.95253165 0.99367089]
0.9696991175178692
```

(၂၅) performing SVM by taking hyperparameter degree=3 and kernel as poly

degree=3 နှင့် poly kernel တို့ဖြင့် accuracy score ကို ရှာသည်။

In [50]:
```
from sklearn.svm import SVC
svc= SVC(kernel='poly',degree=3)
svc.fit(X_train,y_train)
y_predict=svc.predict(X_test)
accuracy_score= metrics.accuracy_score(y_test,y_predict)
print(accuracy_score)
```

```
0.9589905362776026
```

degree=3 နှင့် poly kernel တို့ဖြင့် accuracy score ကို ရှာသည်။ K-fold cross validation(where K=10)

In [51]:
```
svc=SVC(kernel='poly',degree=3)
scores = cross_val_score(svc, X, y, cv=10, scoring='accuracy')
print(scores)
print(scores.mean())
```

```
[0.89274448 0.94952681 0.93059937 0.92744479 0.94952681 0.9936908
5
 0.98422713 0.96529968 0.87974684 0.9778481 ]
0.9450654873617378
```

အကောင်းဆုံး(best) parameter များကို ရှာဖွေရန်အတွက် Grid search technique အသုံးပြုသည်။

In [52]:
```
from sklearn.svm import SVC
svm_model= SVC()
```

In [53]:
```
tuned_parameters = {
 'C': (np.arange(0.1,1,0.1)) , 'kernel': ['linear'],
 'C': (np.arange(0.1,1,0.1)) , 'gamma': [0.01,0.02,0.03,0.04,0.05],
'kernel': ['rbf'],
 'degree': [2,3,4] ,'gamma':[0.01,0.02,0.03,0.04,0.05], 'C':(np.ara
nge(0.1,1,0.1)) , 'kernel':['poly']
                    }
```

(၂၆) GridSearchCV

In [54]:
```
#from sklearn.grid_search import GridSearchCV
from sklearn.model_selection import GridSearchCV
model_svm = GridSearchCV(svm_model, tuned_parameters,cv=10,scoring=
'accuracy')
```

In [55]:
```
model_svm.fit(X_train, y_train)
print(model_svm.best_score_)
```

```
0.9569745728424264
```

In [56]:
```
print(model_svm.cv)
```

```
10
```

In [57]:
```
print(model_svm.best_params_)
```

```
{'C': 0.9, 'degree': 3, 'gamma': 0.05, 'kernel': 'poly'}
```

In [58]:
```
y_pred= model_svm.predict(X_test)
print(metrics.accuracy_score(y_pred,y_test))
```

```
0.9589905362776026
```

Ref

https://www.kaggle.com/akshaymewada7/gender-recognition-by-voice-acoustic-parameters (https://www.kaggle.com/akshaymewada7/gender-recognition-by-voice-acoustic-parameters)

https://github.com/nirajvermafcb/Data-Science-with-python (https://github.com/nirajvermafcb/Data-Science-with-python)