دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی بر ق

درس:ماشین لرنینگ

پویا شریفی

9823117

تمرین عملی svm

هدف از انجام این تمرین:

بخش اول

دسته بندی انواع قورباغه ها بر حسب families آن ها,که این دسته بندی را به چند روش مختلف میتوان انجام داد ,اما در این جا از svm و روش های مختلف آن استفاده میکنیم.

در ابتدا لایبرری های مورد نیاز را import میکنیم:

```
import numpy
import matplotlib.pyplot as plt
from sklearn import svm
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, KFold
from sklearn.preprocessing import MinMaxScaler
```

برای راحت تر شدن کار ,دیتا ست را بر روی google drive خودم قرار دادم و بعد colab را بر روی آن mount کردم تا بتوانم دسرسی به آن داشته باشم:

```
from google.colab import drive
drive.mount('/content/gdrive')
```

سپس دیتا را میخوانیم:

```
df= pd.read_csv("/content/gdrive/MyDrive/ml_svm_dataset/Frogs_MFCCs.csv")
```

در دیتا ست ما ,یک سری از ستون ها عدد نیستند و به صورت categorical هستند و نیاز است که آن ها را به عدد تبدیل کنیم ,که اینجا از دستور cat.code استفاده میکنیم:

```
index = []
for i in df.select_dtypes(include='object').columns:
    df[i] = df[i].astype('category').cat.codes
    index.append(i)
```

```
X = df.drop(['Family', 'RecordID'],axis=1)
y = df['Family']
```

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,
 shuffle=True)
```

fit کردن

برای دستبندی ,از لایبری خود sklearn استفاده میکنیم که به ما svm میدهد.

به کمک svc که کرنل آن linear است دیتا را دسته بندی میکنیم:

```
clf=svm.SVC(kernel='linear' , random_state=0)
```

سپس آن را فیت میکنیم:

```
clf.fit(x_train, y_train)
```

```
[58] y_pred = clf.predict(x_test)

[59] y_pred

     array([3, 3, 3, ..., 3, 3, 3], dtype=int8)

[60] from sklearn import metrics
     print("accuracy of our data")

     print(metrics.accuracy_score(y_pred,y_test))

     accuracy of our data
     0.9763779527559056
```

همانطور که میبینید ,وقتی داده ی تست را روی آن تست کردیم ,به accuracy 97% رسیدیم ,که این بسیار خوب است.

تعداد svm ها را نیز از تابع خود sklearn استفاده میکنیم و میبابیم چون خود sklearn تابع support_vectors_ دارد:

```
print (np.array(clf.support_vectors_).shape)
```

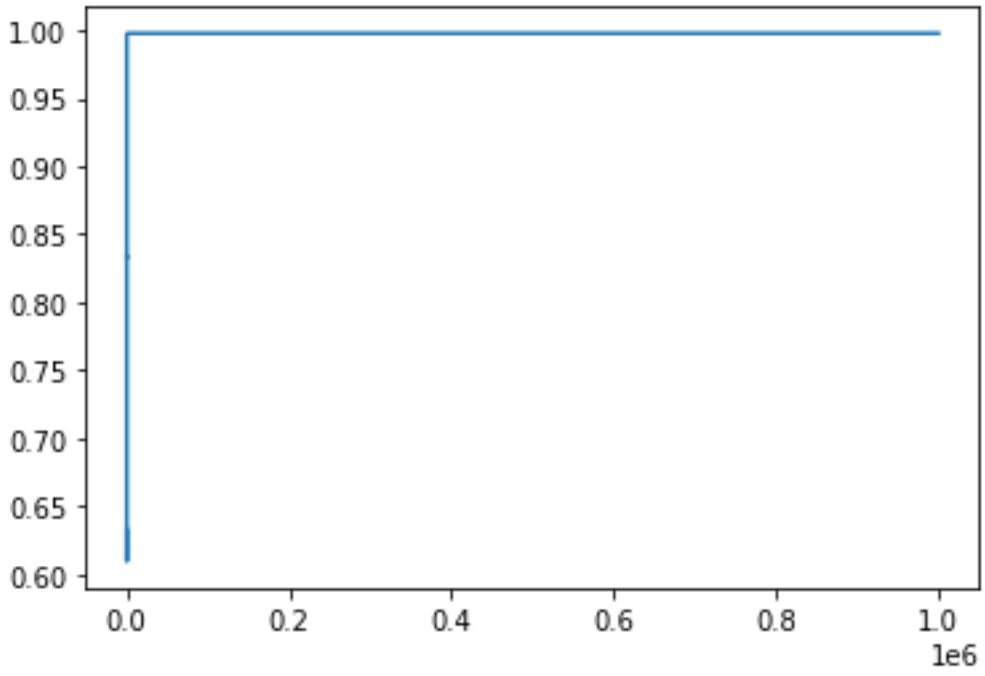
قسمت ب)

Soft non linear svm

برای این قسمت از همان svm قبل استفاده میکنیم و اما اینبار kernel را بر poly قرار می دهیم تا چند جمله ای باشد و linear نباشد.

و سپس بر حسب اعداد مختلف ضریب c همه ی آن ها را رسم میکنیم تا بهترین آن را ببینیم.

```
c_range = [0.0001,0.001, 0.01, 0.1, 1, 10, 100, 1000,10000]
acc = []
accuracy_lst=[]
for i in c_range:

    Nl_cl = svm.SVC(kernel='poly',C=i)
    Nl_cl.fit(x_train, y_train)
    y_pred_train = Nl_cl.predict(x_train)
    y_pred = Nl_cl.predict(x_test)
    print("accuracy of our data")

    print(metrics.accuracy_score(y_pred,y_test))
    accuracy_lst.append(metrics.accuracy_score(y_pred,y_test))
plt.plot(c_range,accuracy_lst)
```



همانطور که می بینید accuracy با بیشتز c ها 99% است.

در این بخش 95 تا support vector داشتیم.

بخش پ)

Rbf and nonlinear

در این بخش از کرنل rbf استفاده میکنیم که در آن از بهترین C ای که در بخش قبلی پیدا شده بود استفاده کردیم.

```
Classifier=svm.SVC(kernel='rbf',C=1000)
Classifier.fit(x_train, y_train)
y_pred = Classifier.predict(x_test)

print("accuracy of our data")

print(metrics.accuracy_score(y_pred,y_test))
```

```
accuracy of our data
1.0
```

که به accuracy فوقعلاده ای دست یافتیم.

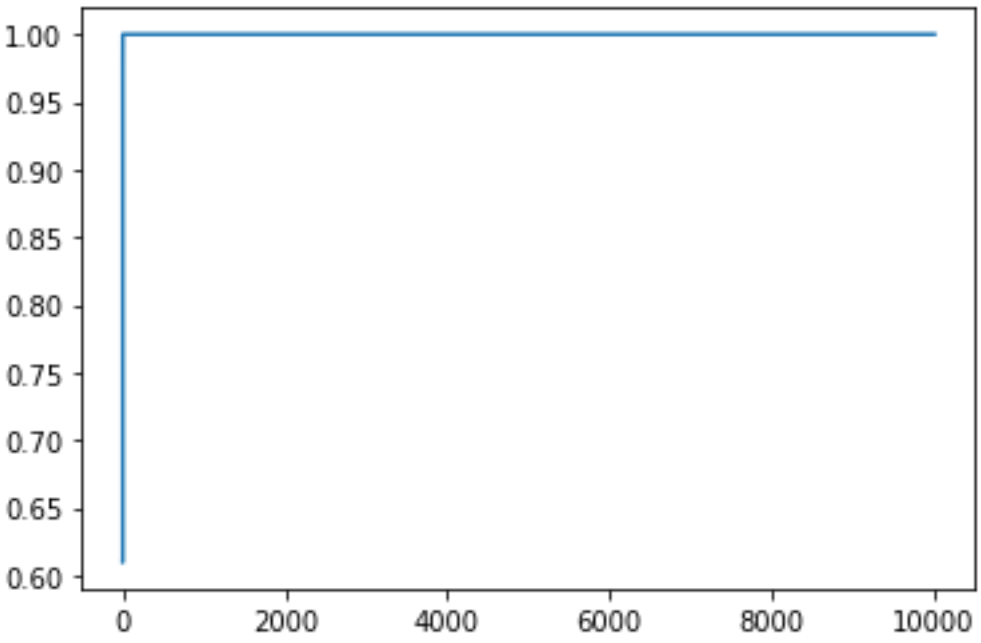
حالا همین کار را فقط با غیر خطی تکرار میکنیم:

```
[81] Classifier=svm.SVC(kernel='poly',C=1000)
     Classifier.fit(x_train, y_train)
     y_pred = Classifier.predict(x_test)

     print("accuracy of our data")

     print(metrics.accuracy_score(y_pred,y_test))
```

```
accuracy of our data
0.997684130152848
```

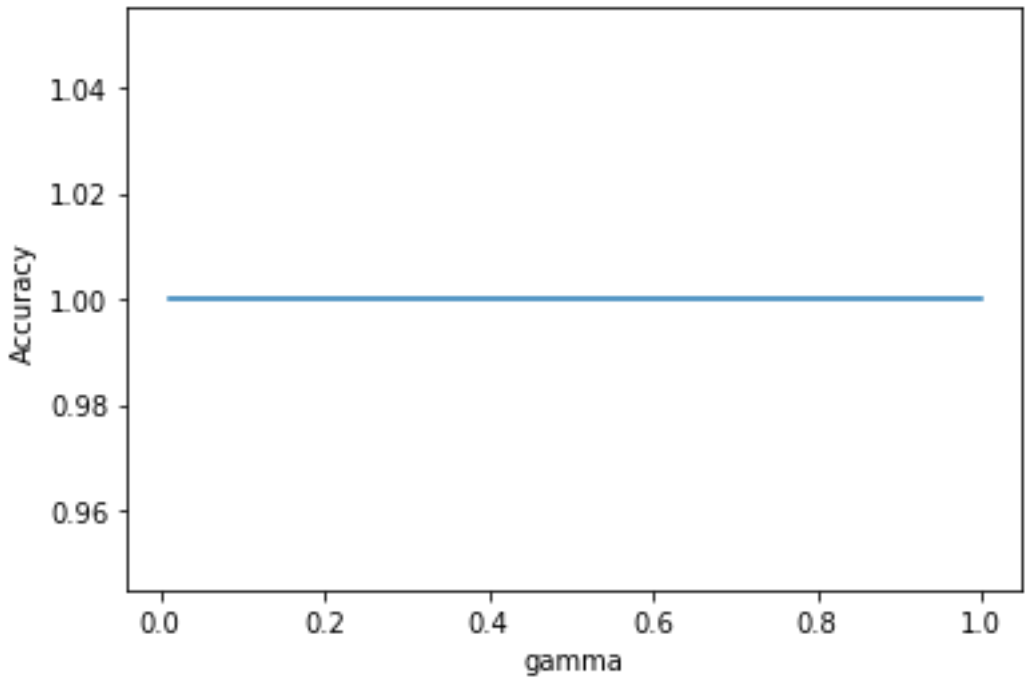
```
c_range = [0.00001,0.0001,0.001, 0.01, 0.1, 1, 10, 100, 1000,10000]
acc = []
accuracy_lst=[]
for i in c_range:
    Classifier=svm.SVC(kernel='rbf',C=i)
```

```
    Classifier.fit(x_train, y_train)
    y_pred = Classifier.predict(x_test)

    print("accuracy of our data")

    print(metrics.accuracy_score(y_pred,y_test))

    accuracy_lst.append(metrics.accuracy_score(y_pred,y_test))
plt.plot(c_range,accuracy_lst)
```



بخش ت)

با for زدن روی گاما بهترین رو انتخاب میکنیم ,اما می بینیم که همهی آن ها یک شد



پس بهترین کلسیفایر را یافتیم که:

```
best_classifier = svm.SVC(kernel='rbf' , C=1000 , gamma=0.01)
```

بخش ث)

حالا همین classifier را در kfolds قرار میدهیم :

```
kf = KFold(n_splits=4, shuffle=True)
score = cross_val_score(best_classifier, x_train, y_train, cv=kf)
print(np.mean(score))
```

و آن به ما accuracy `0.9992057188244639 میدهید`