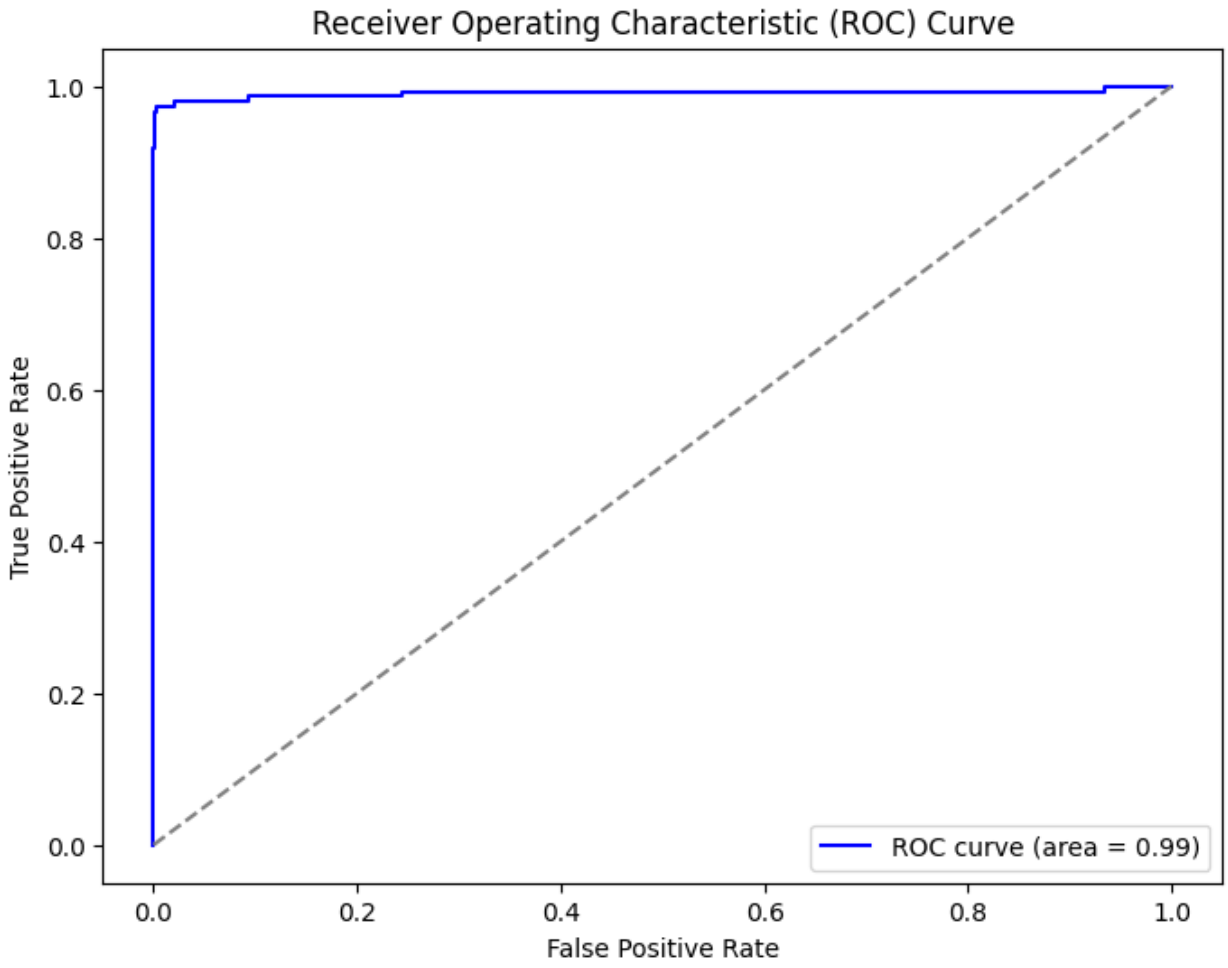
به نام خدا

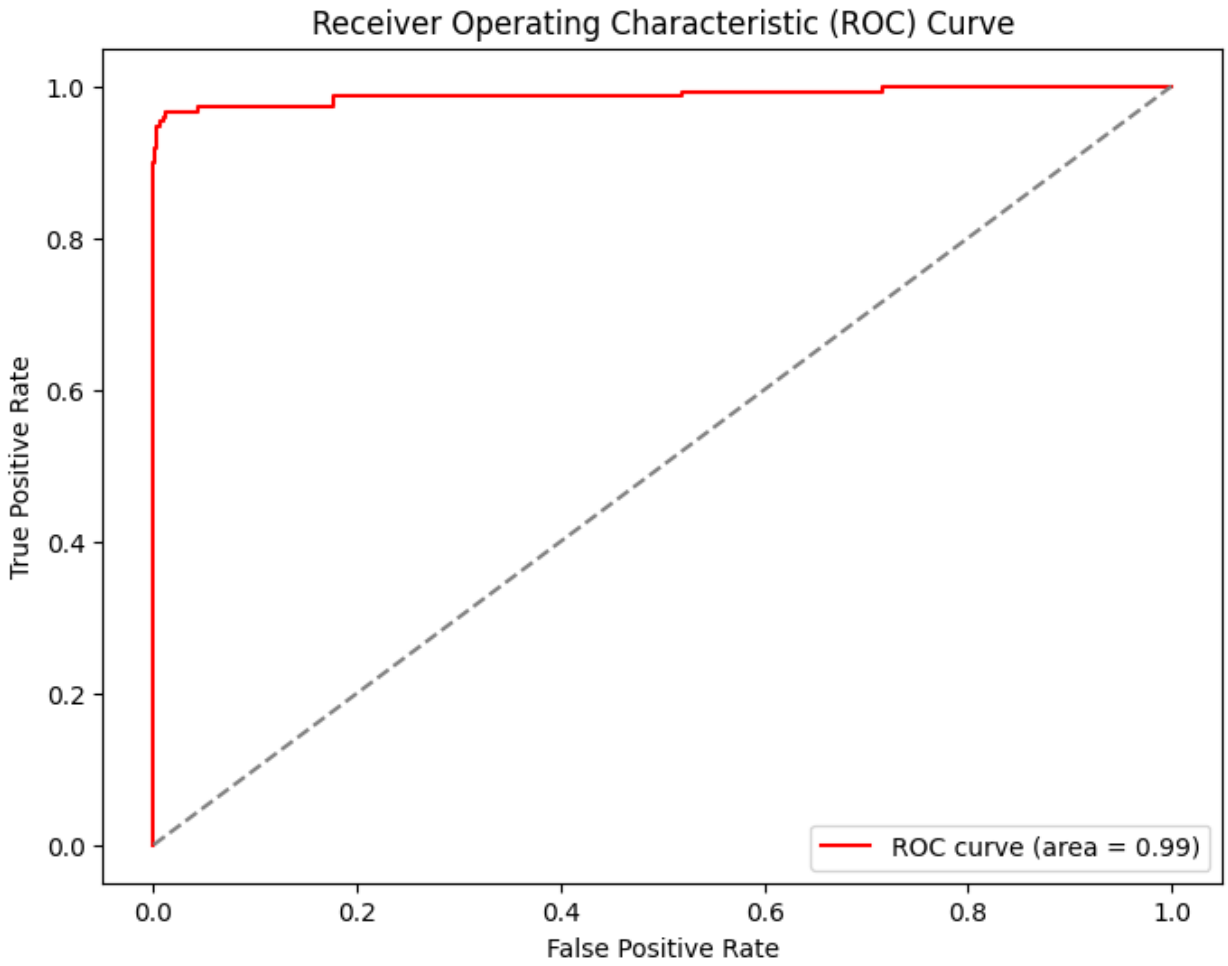
پروژه چهارم - هوش مصنوعی **نیایش خانی ۹۹۵۲۱۲۳۵**

سوال ۱) برای حل این سوال طبق توضیحات داده شده پیش رفتم و از متود TF-IDF برای پیش پردازش استفاده کردم. با استفاده از این متود خروجی ها به صورت زیر خواهد بود :




```
Accuracy: 0.99
ROC AUC: 0.99
```

با تحقیق بیشتر متود های دیگری نیز مانند Count Vectorization ، Word Embeddings و ... برای پیش پردازش پیدا کردم و به عنوان نمونه کد قبل را با استفاده از Count Vectorization پیاده سازی کردم که نتیجه به صورت زیر بود :

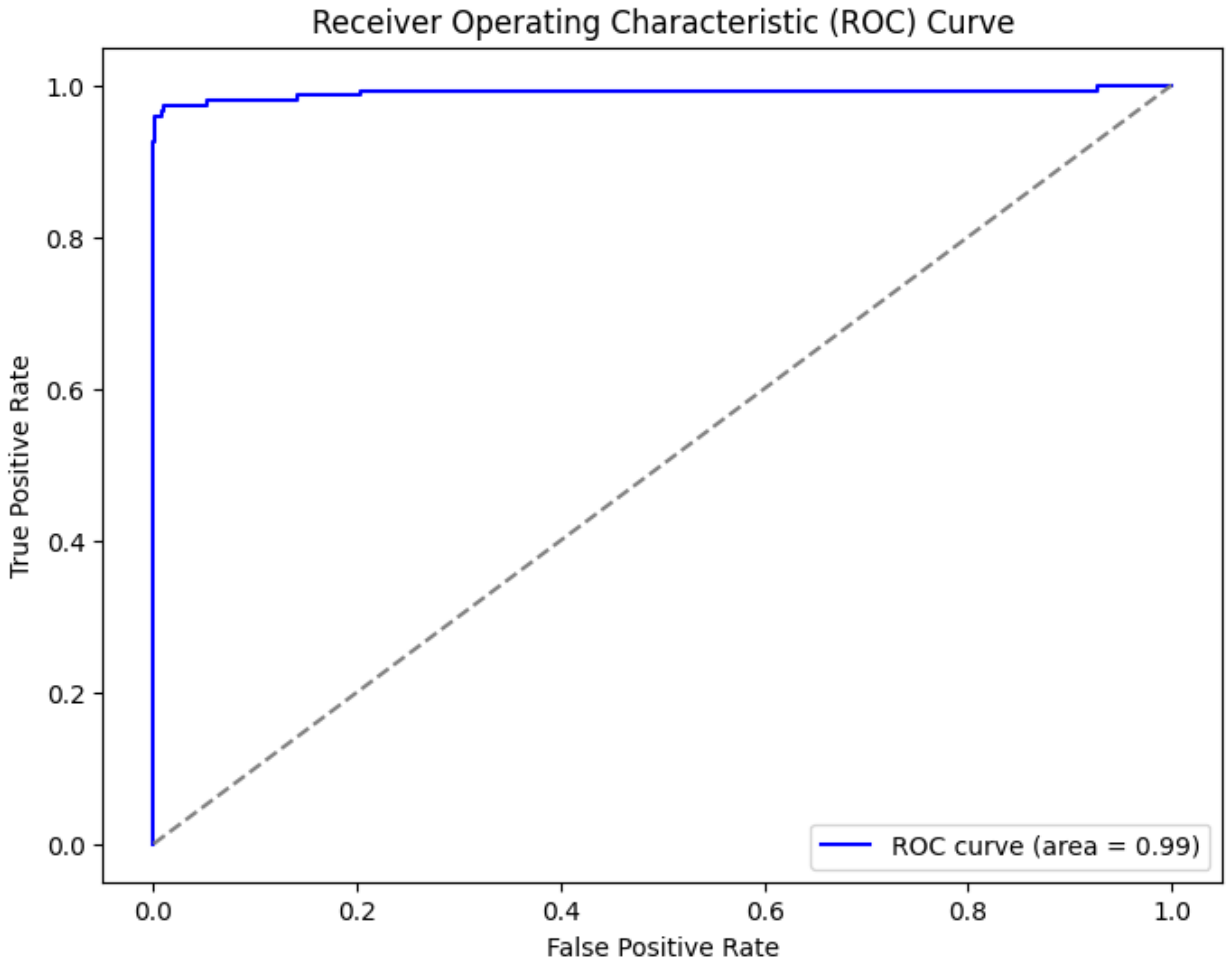



```
Accuracy: 0.99
ROC AUC: 0.99
```

همانطور که مشاهده می شود، خروجی هر دو متد تقریبا یکسان است (ممکن است به دلیل ساده و کوچک بودن دیتاست باشد) اما اگر به زمان ران شدن آنها توجه کنیم می بینیم که متود TF-IDF مدت زمان بیشتری طول کشیده است تا ران شود زیرا پیچیده تر است و محاسبات کندتر می شود.

با این حال TF-IDF، متد انتخابی است زیرا با تاکید بر کلمات منحصر به فرد و متمایز عملکرد بهتری دارد و اغلب برای classification انتخاب می شود.

همچنین kernel را نیز از خطی (پیشنهادی) به RBF تغییر دادم که خروجی بدست آمده به این صورت بود :




```
Accuracy: 0.98
ROC AUC: 0.99
```

همانطور که مشاهده می شود، علاوه بر کاهش دقت، ران تایم هم افزایش یافته است زیرا کرنل پیچیده تری نسبت به linear است. در کل نتیجه مطلوبی بدست نیامد و کرنل انتخابی همان خطی است.

سوال ۲) ابتدا طبق توضیحات پیشنهادی پیش رفتم به وسیله کد زیر ریپورت را دریافت کردم :

```
y_pred = svm_model.predict(X_test_scaled)
report = classification_report(y_test, y_pred)
print(report)
```

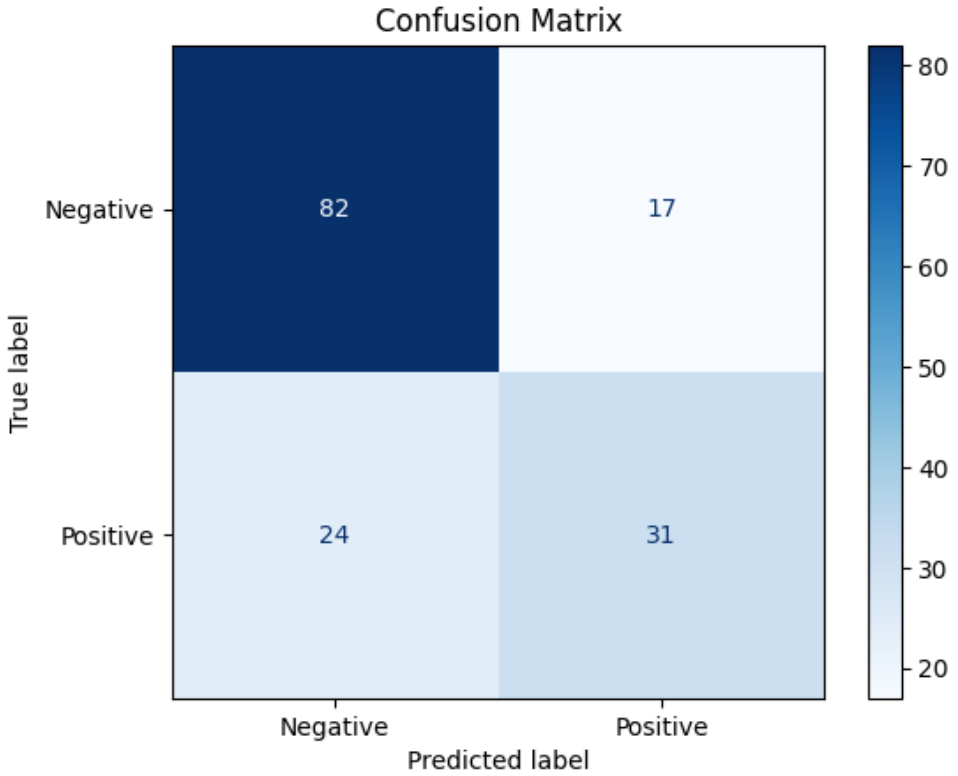
```
              precision    recall  f1-score   support

           0       0.77      0.83      0.80        99
           1       0.65      0.56      0.60        55

    accuracy                           0.73       154
   macro avg       0.71      0.70      0.70       154
weighted avg       0.73      0.73      0.73       154
```

همانطور که مشاهده می شود این ریپورت شامل precision (دقت) ، recall (فراخوانی) و F1-score است که برای کلاس ۰ (negative class) و کلاس ۱ (positive class) مقادیر را نشان می دهد.
سپس مقدار accuracy را داریم که ۷۳٪ است.
سپس با استفاده از فانکشن confusion_matrix از کتابخانه sklearn.metrics ، ماتریس سردرگمی را نشان می دهیم که به این صورت خواهد بود:

```
Confusion Matrix:
[[82 17]
 [24 31]]
```

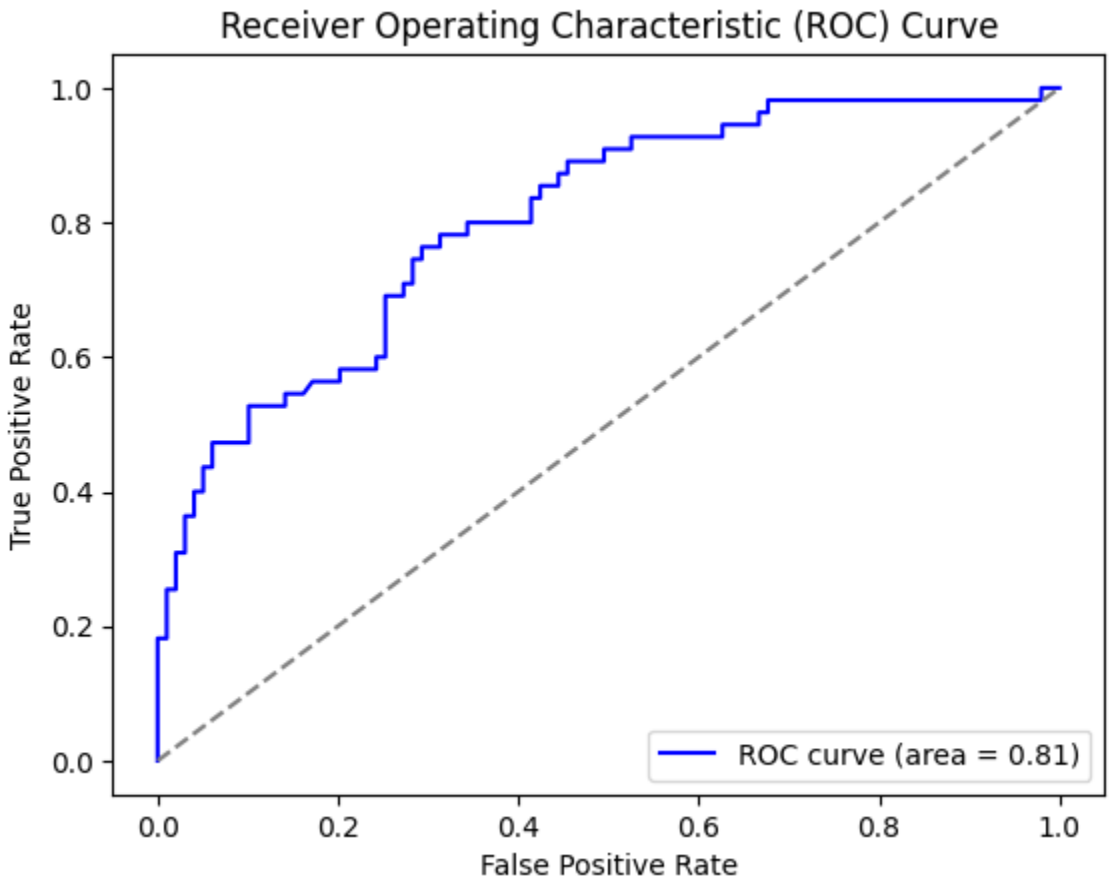


همانطور که مشاهده می شود مقدار True Negative برابر ۸۲، False Negative برابر ۲٤، True Positive برابر ۳۱ و False Positive برابر ۱۷ است.
به طور کلی، این مدل برای کلاس ۰ (کلاس منفی) در مقایسه با کلاس ۱ (کلاس مثبت) به ویژه از نظر فراخوانی بهتر عمل می کند.

برای ROC خروجی ها به این صورت است :

```
AUC Score: 0.805050505050505
```

همانطور که مشاهده می شود دقت دریافت شده از ROC ، ۸۰٪ است.
نمودار ROC :

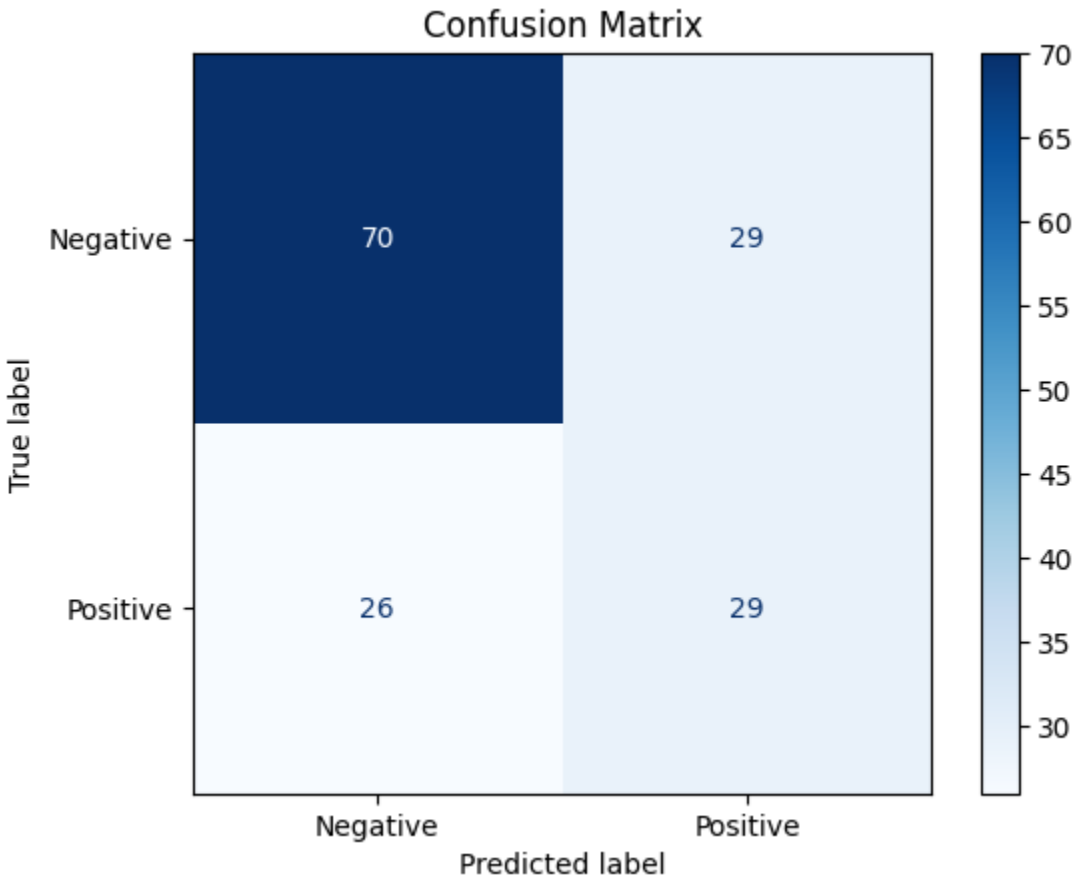


همچنین این کد را با کرنل sigmoid هم امتحان کردم که نتیجه به این صورت بود :

	precision	recall	f1-score	support
0	0.73	0.71	0.72	99
1	0.50	0.53	0.51	55
accuracy			0.64	154
macro avg	0.61	0.62	0.62	154
weighted avg	0.65	0.64	0.64	154

همانطور که مشاهده می شود، تمامی معیار های ارزیابی کاهش یافته اند. همچنین اگر نگاهی به ماتریس سردرگمی بیندازیم، خواهیم دید :

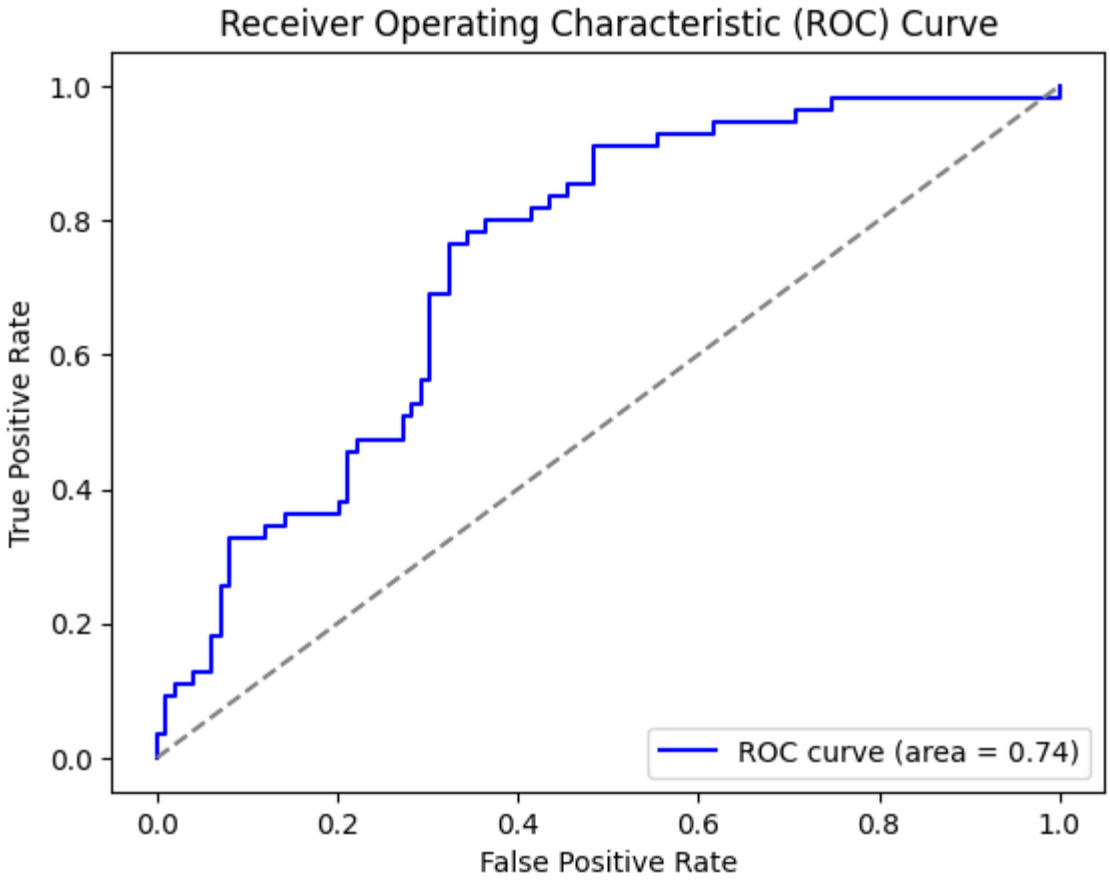
```
Confusion Matrix:
[[70 29]
 [26 29]]
```



واضح است که خطای مدل بیشتر شده است و مقادیری که به درستی predict می کرد، کاهش یافته اند.
انتظار می رود که دقت ROC و منحنی آن نیز بدتر شده باشند.

```
AUC Score: 0.7393939393939395
```

بنابراین توانایی کرنل RBF برای مدل‌سازی روابط غیرخطی، انعطاف‌پذیری و استحکام آن در تنظیمات پیش‌فرض هایپرپارامتر آن را به انتخاب بهتری برای دیتاست دیابت در مقایسه با هسته سیگموئید تبدیل می‌کند.



سوال ۳) ابتدا دیتاست مسکن بوستون را از OpenML لود کردم که شامل ویژگی‌های مختلفی (مانند میزان جرم، تعداد اتاق‌ها و غیره) و متغیر هدف MEDV، ارزش متوسط خانه‌های تحت استفاده است. سپس طبق توضیحات این مراحل را پیش گرفتم :

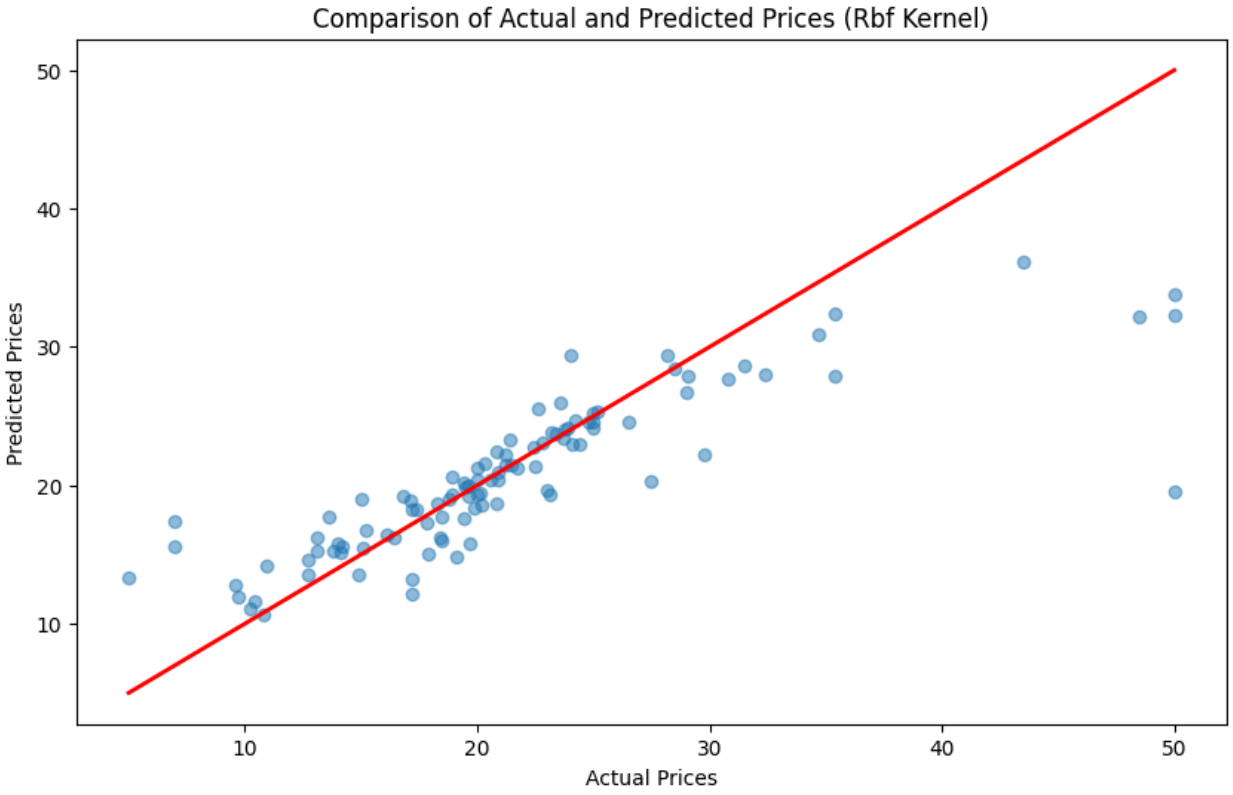
تقسیم بندی دیتا ها را به مجموعه های آموزشی و آزمایشی و استاندارد سازی دیتا ها. سپس به جای استفاده از کرنل پیشنهادی، سعی کردم از چند کرنل استفاده کنم و به ازای هر کدام ، معیار های MSN و $R^2\ Score$ را محاسبه کردم. بعد بهترین کرنل را بر اساس بالاترین امتیاز R^2 انتخاب کرده و نمودار آن را رسم کردم.

خروجی ها به این صورت بود :

```
Linear Kernel - Mean Squared Error: 28.91852267161847, R^2 Score: 0.6056589279132574
Poly Kernel - Mean Squared Error: 25.82919767676698, R^2 Score: 0.6477858285273655
Rbf Kernel - Mean Squared Error: 25.668539678396044, R^2 Score: 0.6499766059760035
Sigmoid Kernel - Mean Squared Error: 38.151095072597, R^2 Score: 0.47976098561299174

Best Kernel: Rbf
Best Kernel - Mean Squared Error: 25.668539678396044, R^2 Score: 0.6499766059760035
```

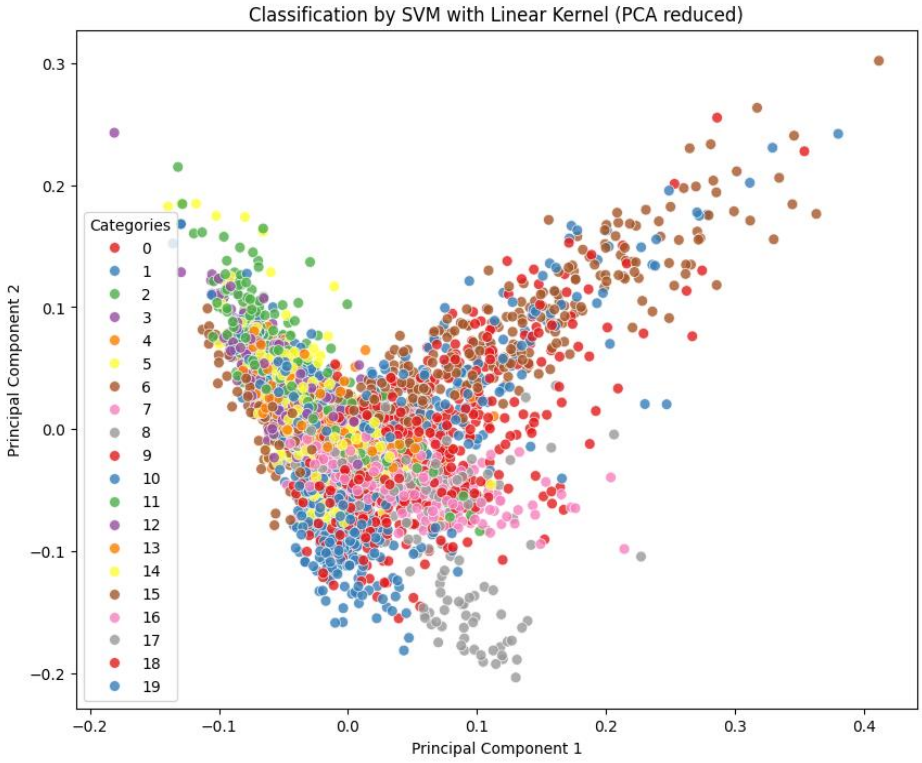
همانطور که مشاهده می شود کمترین مقدار خطا و بیشترین $R^2\ Score$ متعلق به RBF است که کرنل پیشنهادی سوال نیز بود. نمودار بدست آمده از این کرنل :



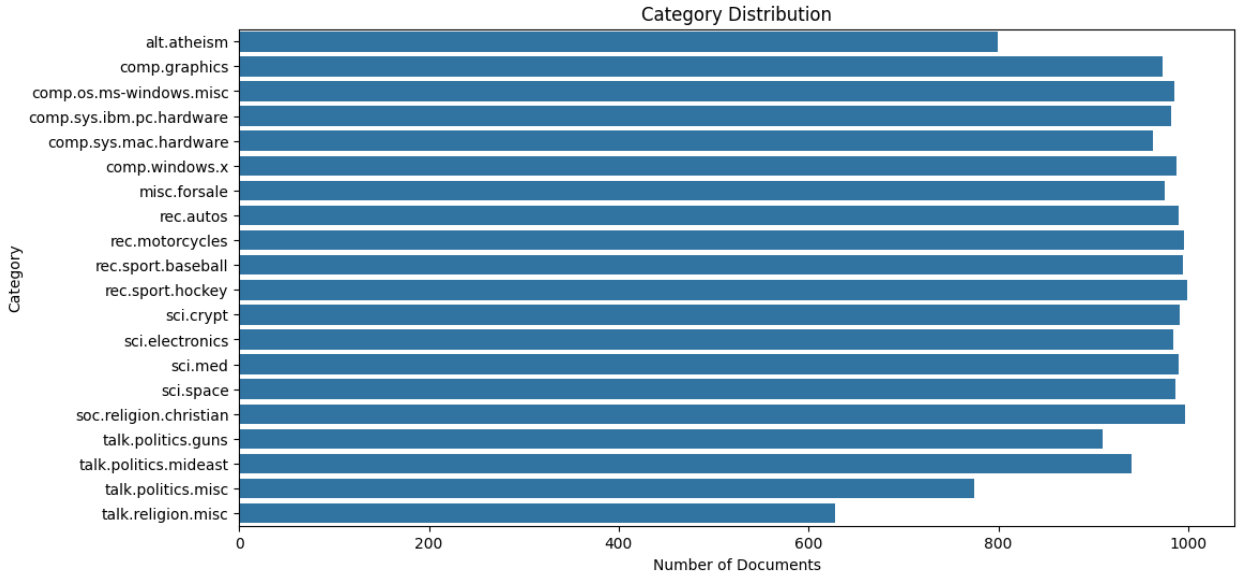
سوال ٤) ابتدا دیتاست را از sklearn.datasets لود کردم و باقی مراحل را طبق توضیحات پیش رفتم. به دلیل سنگین بودن دیتاست ، ران کردن هرباره کد وقت زیادی می برد بنابراین متاسفانه به همان کرنل خطی و پیش پردازش TF-IDF بسنده کردم.
دقت این مدل :

```
Accuracy: 0.9160
```

همچنین نمودار دسته بندی آن را به این شکل بدست آوردم :



همانطور که مشاهده می شود بیست category داریم که تعداد هر کدام روی شکل مشخص است اما به دلیل زیاد بودن دیتا برای هر دسته بندی، راه بهتر نشان دادن نمودار می تواند به صورت زیر باشد :



طبق نمودار می بینیم که همچنان بیست category داریم که تعداد دیتای هرکدام نیز مشخص شده است و به این صورت متن ها دسته بندی شده و نویسنده آنها تشخیص داده شده است.

سوال ۵) برای حل سوال از طریق لینک قرار شده شده در سوال، دیتاستی پیدا نکردم بنابراین مجبور شدم از لینکی که یکی از دوستان در گروه فرستاده بود، استفاده کنم. لینک دیتاست :
https://raw.githubusercontent.com/kshedden/statswpy-nhanes/master/merged/nhanes_2015_2016.csv
برای تشخیص بیماری های مزمن در این دیتاست ستون به خصوصی وجود نداشت بنابراین تصمیم گرفتم فاکتورهایی که منجر به تشخیص بیماری مزمن می شوند را معیار قرار بدهم. پس یک متغیر هدف بر اساس آستانه‌های خاصی برای چند شاخص سلامت ایجاد کردم که اگر هر یک از شرایط زیر وجود داشته باشد، آن بیماری مزمن است:

۱. فشار خون بالا : فشار خون سیستولیک (BPXSY۱ > ۱٤۰) یا فشار خون دیاستولیک (BPXDI۱ > ۹۰).
۲. BMI بالا: شاخص توده بدن (BMXBMI > ۳۰)
۳. دور کمر بالا: برای مردان دور کمر (BMXWAIST > 102 cm) و برای زنان (BMXWAIST > 88cm).

بنابراین متغیر هدف به این صورت ساخته شد :

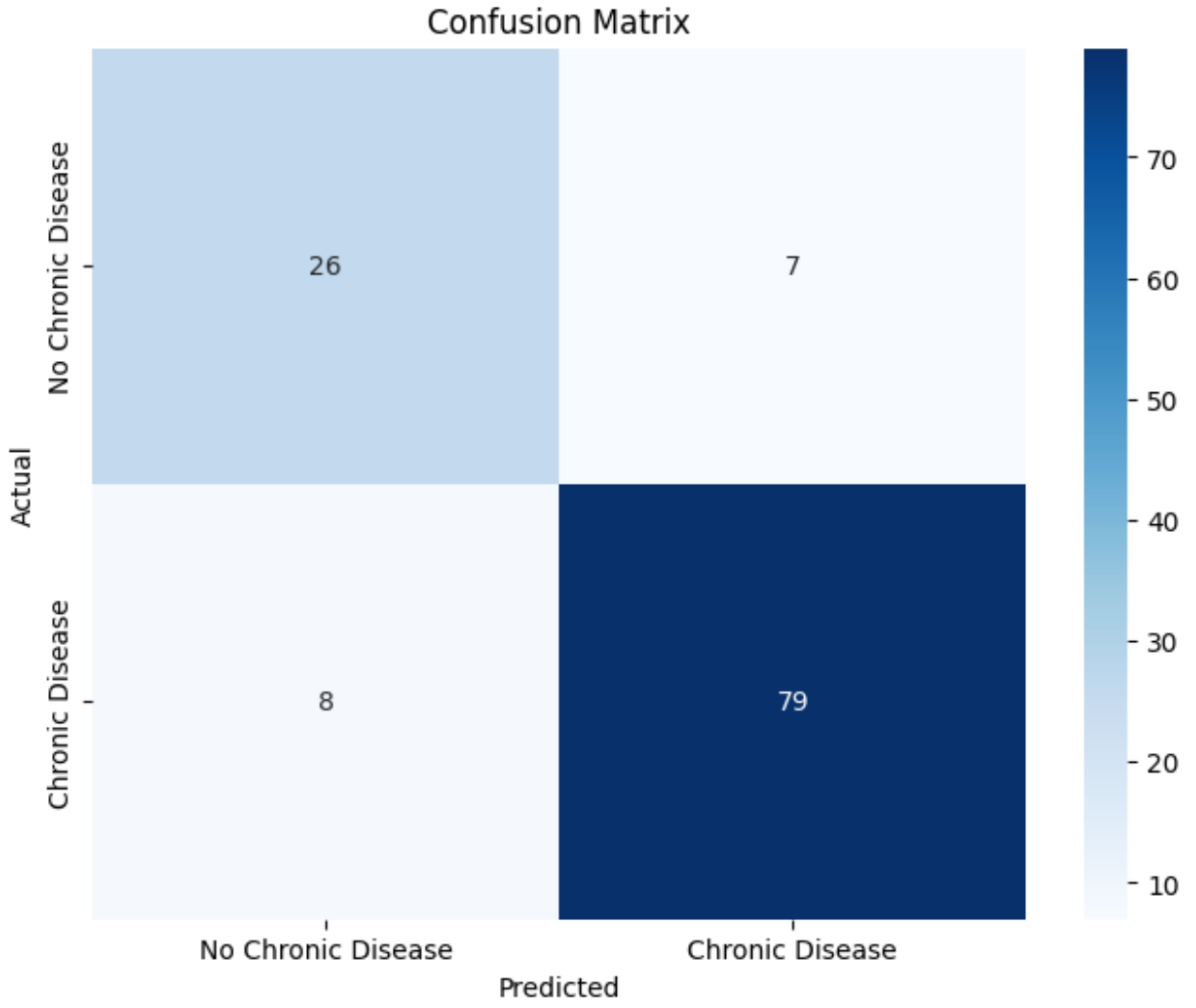
```
# Create the custom target variable
data['CHRONIC'] = (
    (data['BPXSY1'] > 140) |
    (data['BPXDI1'] > 90) |
    (data['BMXBMI'] > 30) |
    ((data['RIAGENDR'] == 1) & (data['BMXWAIST'] > 102)) |  # Men with waist > 102 cm
    ((data['RIAGENDR'] == 2) & (data['BMXWAIST'] > 88))    # Women with waist > 88 cm
).astype(int)
```

سپس طبق توضیحات پیش رفتم و در آخر معیار ها به این صورت شد :

	precision	recall	f1-score	support
0	0.76	0.79	0.78	33
1	0.92	0.91	0.91	87
accuracy			0.88	120
macro avg	0.84	0.85	0.84	120
weighted avg	0.88	0.88	0.88	120

همانطور که مشاهده می شود precision (دقت) ، recall (فراخوانی) و F1-score را برای کلاس ۰ (negative class) و کلاس ۱ (positive class) اندازه گیری کردیم.مقدار accuracy نیز ۸۸٪ است.

ماتریس سردرگمی :



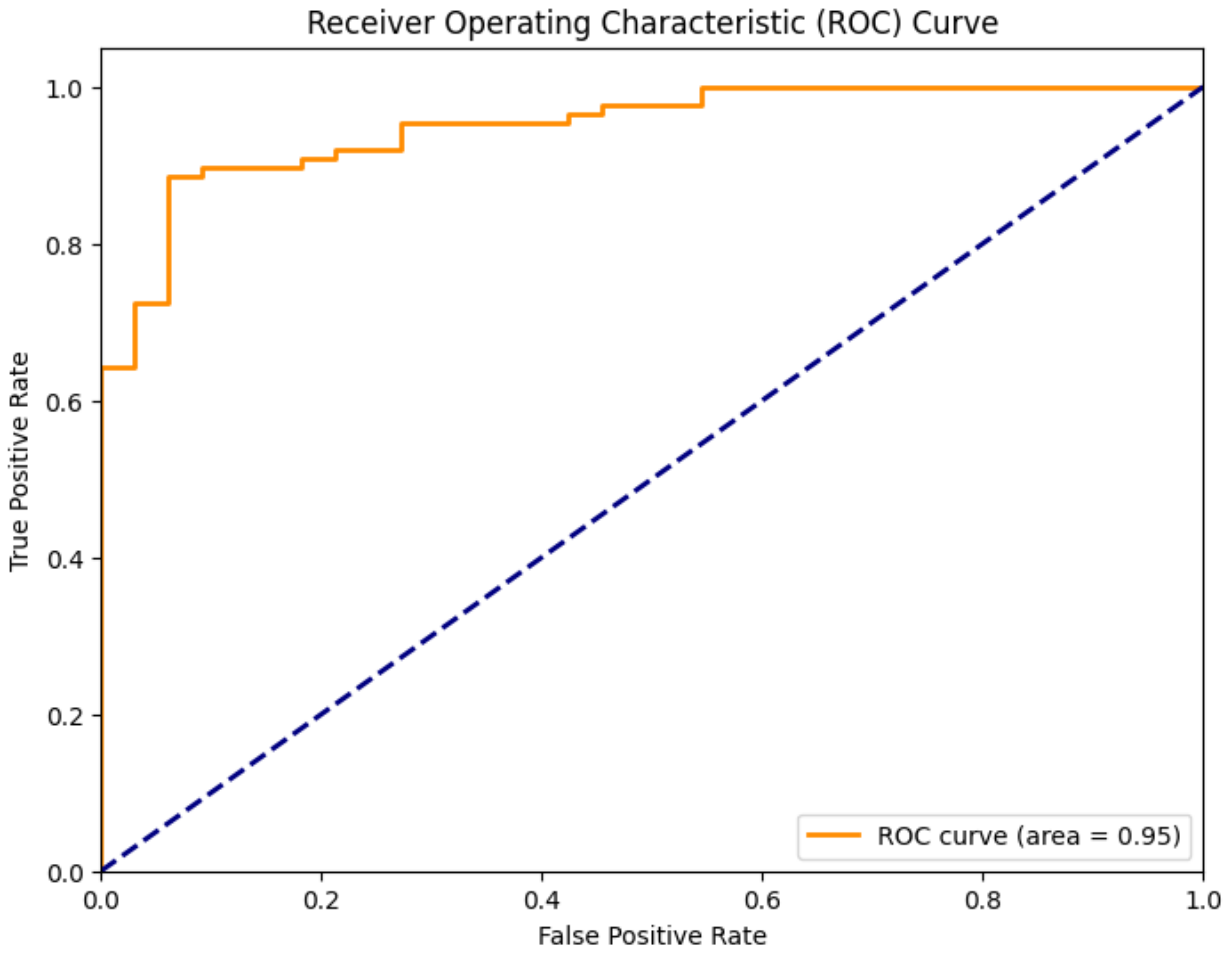
همانطور که مشاهده نتیجه های True - Positive ، ۷۹ تا بوده است. (سایز تست ۰,٤ در نظر گرفته شده است.)

در ROC نیز خروجی ها به این صورت است :

```
AUC Score: 0.9501915708812261
```

همانطور که مشاهده می شود دقت دریافت شده از ROC ، ۹۵٪ است.

نمودار ROC :



برای مقایسه کرنل های متفاوت، مانند سوال سه فانکشنی نوشتم تا مدل SVM با کرنل های مختلف آموزش و ارزیابی شود و سپس بهترین کرنل را بر اساس بالاترین امتیاز F۱ کند. سپس ماتریس سردرگمی و منحنی ROC را برای بهترین مدل ترسیم کردم. نتیجه به این صورت بود :

```
Linear Kernel - F1-Score: 0.9418604651162791
Poly Kernel - F1-Score: 0.9180327868852459
Rbf Kernel - F1-Score: 0.9132947976878613
Sigmoid Kernel - F1-Score: 0.9142857142857143

Best kernel: linear
Best F1-Score: 0.9418604651162791

Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.88      0.85        33
           1       0.95      0.93      0.94        87

    accuracy                           0.92       120
   macro avg       0.89      0.90      0.90       120
weighted avg       0.92      0.92      0.92       120
```

همانطور که مشاهده می شود تمامی کرنل ها عملکرد تقریبا یکسانی دارند. اما کرنل خطی بیشترین امتیاز را دارد بنابراین انتخاب می شود و تمامی معیار های ارزیابی آن را بدست می آوریم.
همچنین ماتریس سردرگمی و منحنی ROC آن نیز به این صورت خواهد بود :