بسمه تعالی

گزارش تمرین نهایی ماشین بردار پشتیبان درس هوش مصنوعی

سینا مظاهری

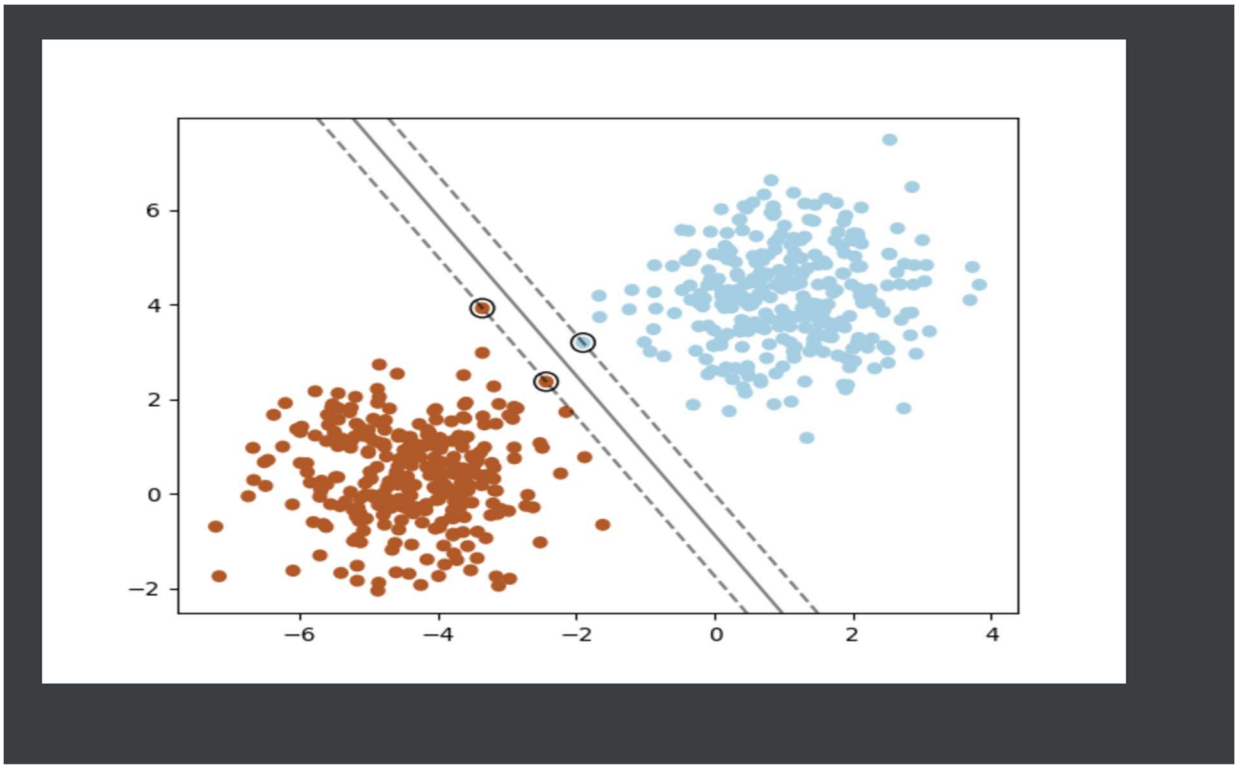
بخش اول)

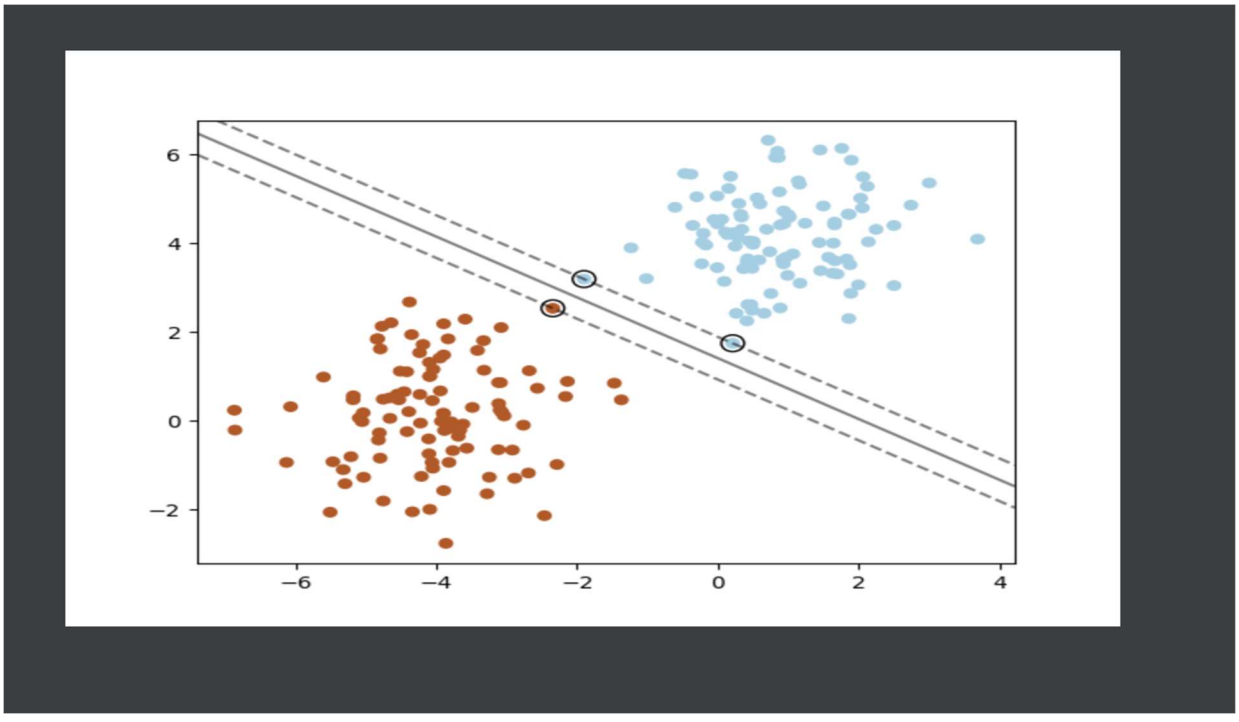
خب در این بخش ابتدا چند نقطه در فضای دو بعدی تولید می کنیم که به شکل خطی جدا پذیر هستند. Margin و خط جدا کننده SVM را هم رسم می کنیم. این فرآیند ها طی قطعه کد های زیر آمده است.

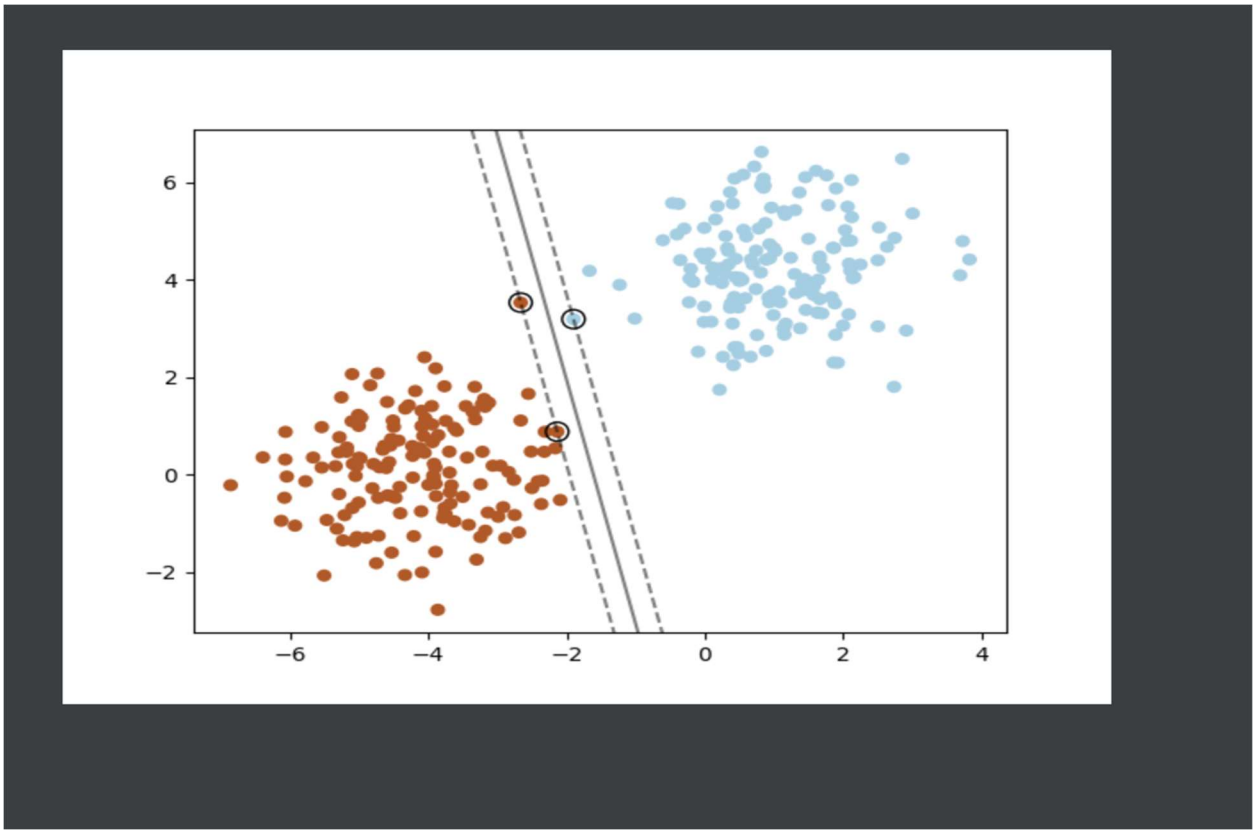
```
#
for i, C in [(1, 300), (2, 400), (3, 1000), (4, 320), (5, 700)]:
    X, y = make_blobs(n_samples=100 * (i + 1), centers=2, random_state=3)
    clf = svm.SVC(kernel="linear", C=1000)
    clf.fit(X, y)
    plt.scatter(X[:, 0], X[:, 1], c=y, s=30, cmap=plt.cm.Paired)
    ax = plt.gca()
    xlim = ax.get_xlim()
    ylim = ax.get_ylim()
    xx = np.linspace(xlim[0], xlim[1], 30)
    yy = np.linspace(ylim[0], ylim[1], 30)
    YY, XX = np.meshgrid(yy, xx)
    xy = np.vstack([XX.ravel(), YY.ravel()]).T
    Z = clf.decision_function(xy).reshape(XX.shape)
    ax.contour(
        XX, YY, Z, colors="k", levels=[-1, 0, 1], alpha=0.5, linestyles=["--", "-", "--"]
    )
    ax.scatter(
        clf.support_vectors_[:, 0],
        clf.support_vectors_[:, 1],
        s=100,
        linewidth=1,
        facecolors="none",
        edgecolors="k",
    )
    plt.show()
```

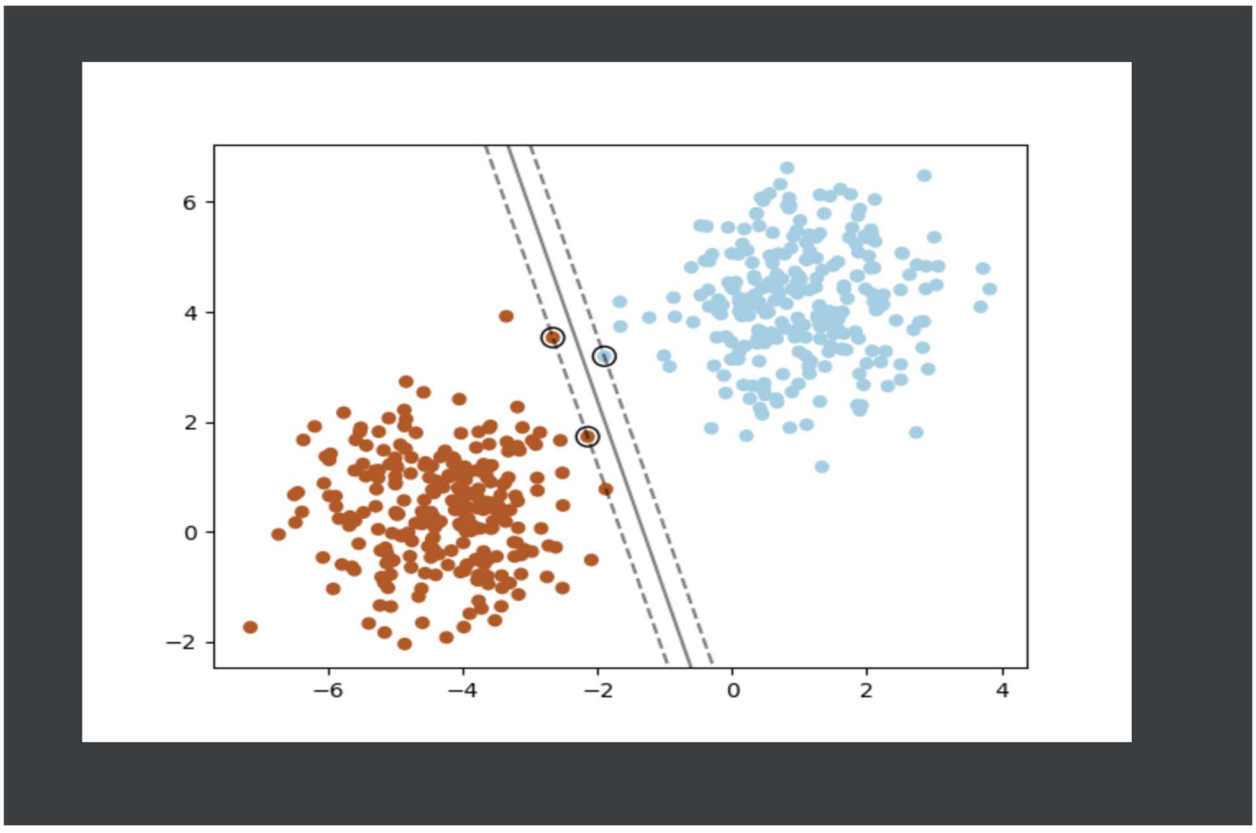
نقاط تولید شده توسط قطعه کد بالا به شکل زیر نمایش داده شده اند و با استفاده از kernel خطی (که به شکل خطی هم جداپذیر می باشند) و با محدودیت های مختلف C طبقه بندی شده اند. می دانیم که هر چقدر

C بیشتر باشد؛ سخت گیرانه تر طبقه بندی انجام می شود و مقادیر مرزی دو کلاس نیز به دقت بیشتری طبقه بندی می شوند.






6
4
2
0
−2
−6
−4
−2
0
2
4

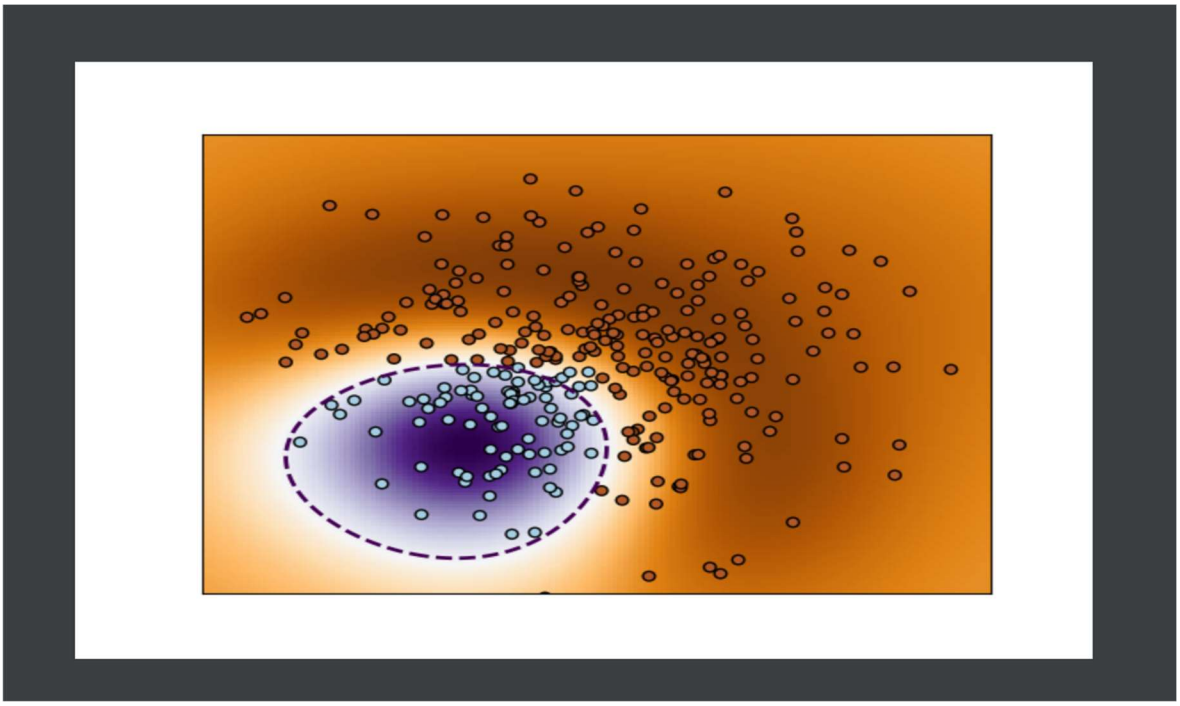

6
4
2
0
−2
−6
−4
−2
0
2
4

حالا تابع y را به شکل زیر پیاده سازی می کنیم:

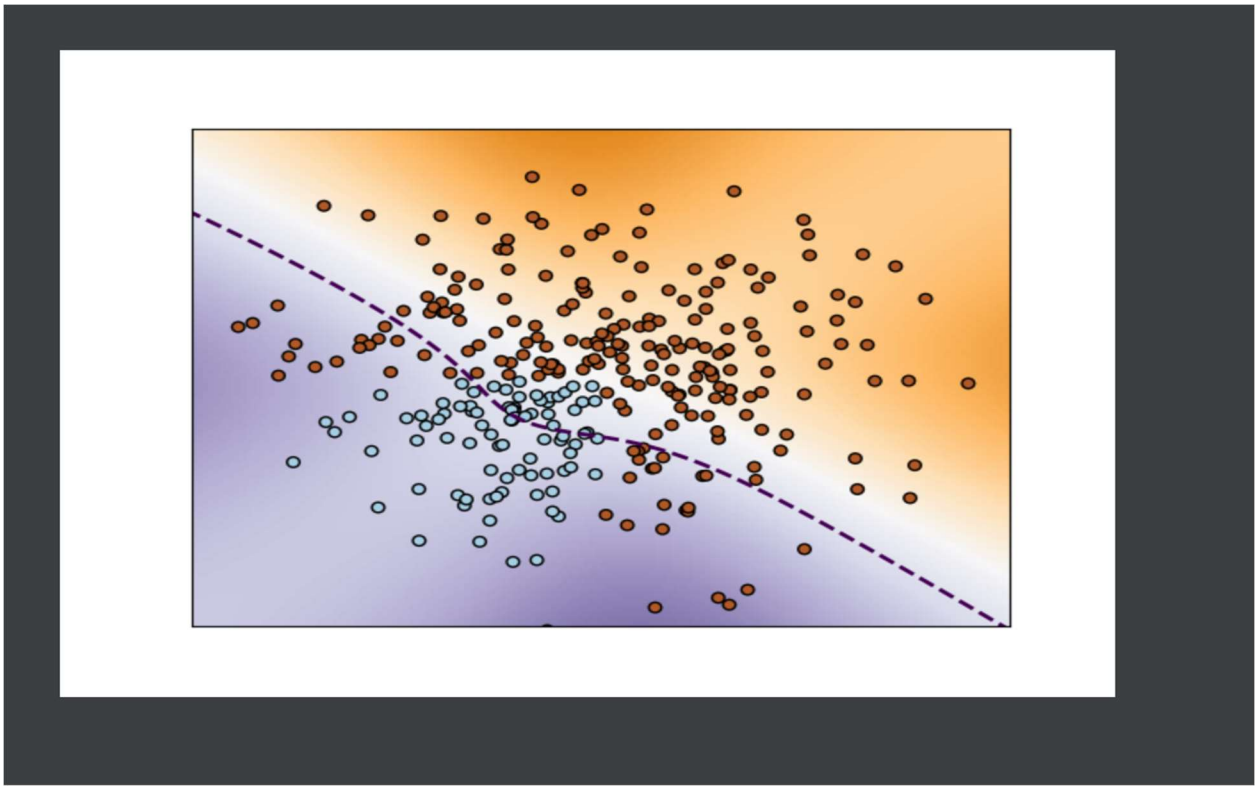
$$y = \begin{cases} 0 & x_1 < 0 \text{ and } x_2 < 0 \\ 1 & \text{otherwise} \end{cases}$$

```
##
for tup in ['rbf','poly','sigmoid']:
    xx, yy = np.meshgrid(np.linspace(-10, 10, 500), np.linspace(-10, 10, 500))
    np.random.seed(0)
    X = np.random.randn(300, 2)
    Y = np.logical_or(X[:, 0] > 0, X[:, 1] > 0)
    clf = svm.NuSVC(kernel=tup, gamma='auto', coef0=0.04, degree=2)
    clf.fit(X, Y)
    Z = clf.decision_function(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    plt.imshow(
        Z,
        interpolation="nearest",
        extent=(xx.min(), xx.max(), yy.min(), yy.max()),
        aspect="auto",
        origin="lower",
        cmap=plt.cm.PuOr_r,
    )
    contours = plt.contour(xx, yy, Z, levels=[0], linewidths=2, linestyles="dashed")
    plt.scatter(X[:, 0], X[:, 1], s=30, c=Y, cmap=plt.cm.Paired, edgecolors="k")
    plt.xticks(())
    plt.yticks(())
    plt.axis([-3, 3, -3, 3])
    plt.show()
#
```

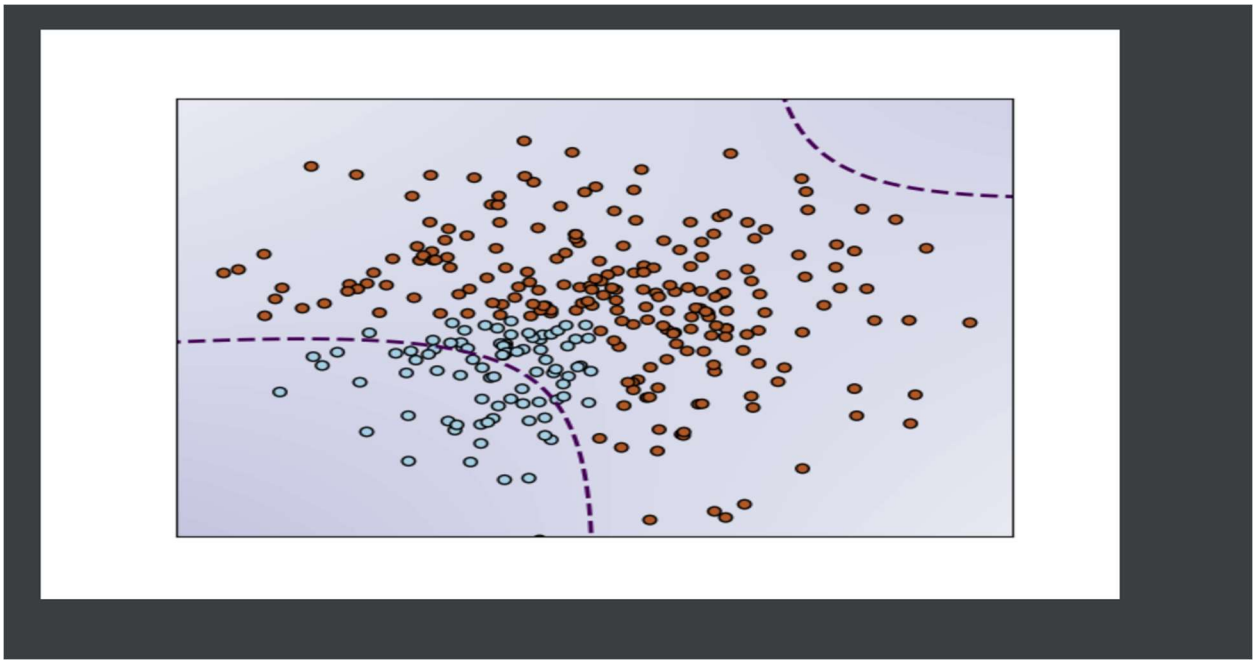
از هسته های گاوسی، چند جمله ای و سیگموید به ترتیب استفاده می کنیم و می بینیم که بهترین آن ها برای گاوسی است: (پارامتر های آن ها نیز یا به طور خودکار است و یا پیش فرض) برای گاوسی در شکل زیر حاصل کار آمده است:



برای چند جمله ای نیز حاصل کار به شکل زیر می شود.



برای سیگموید نیز به شکل زیر می شود:

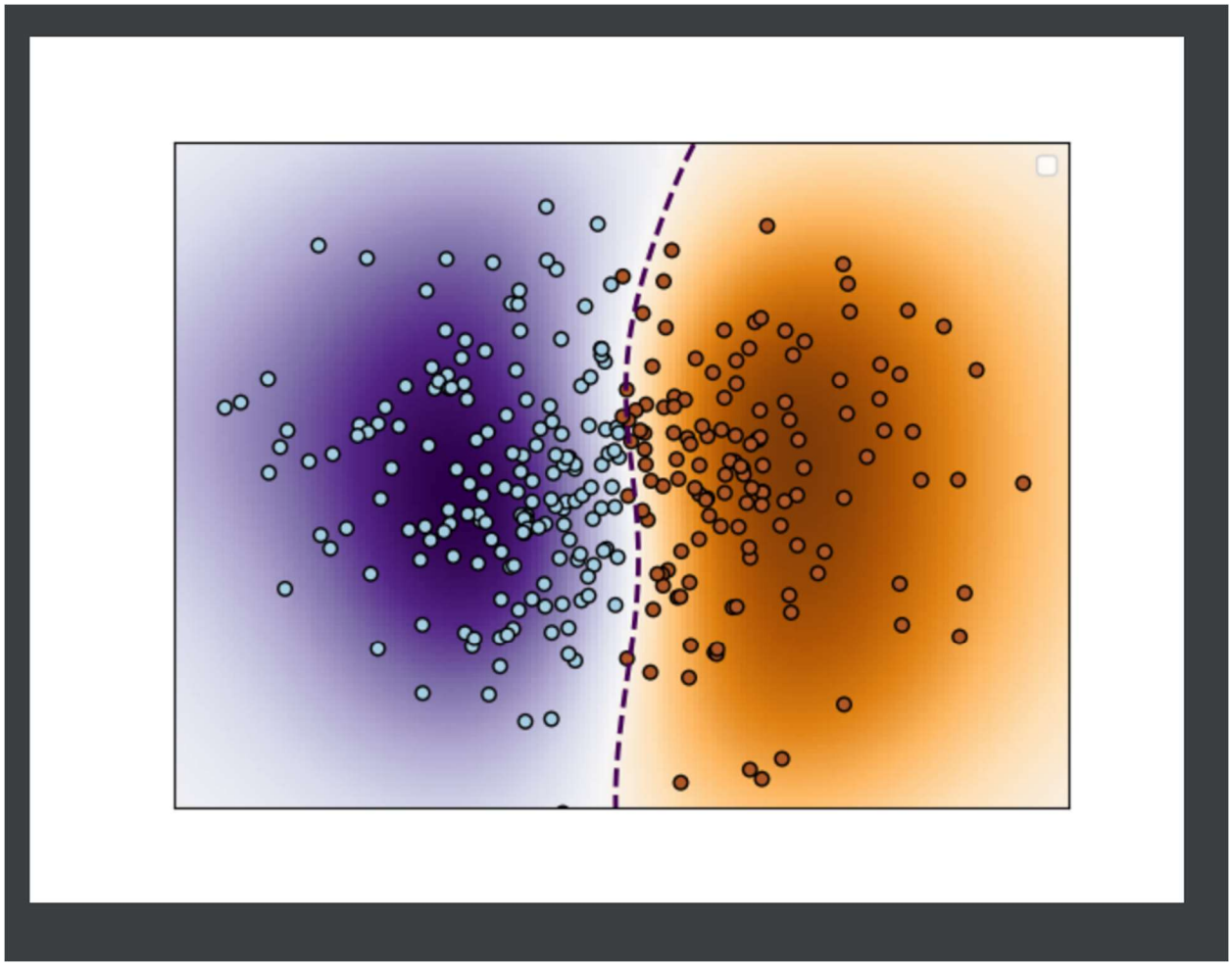


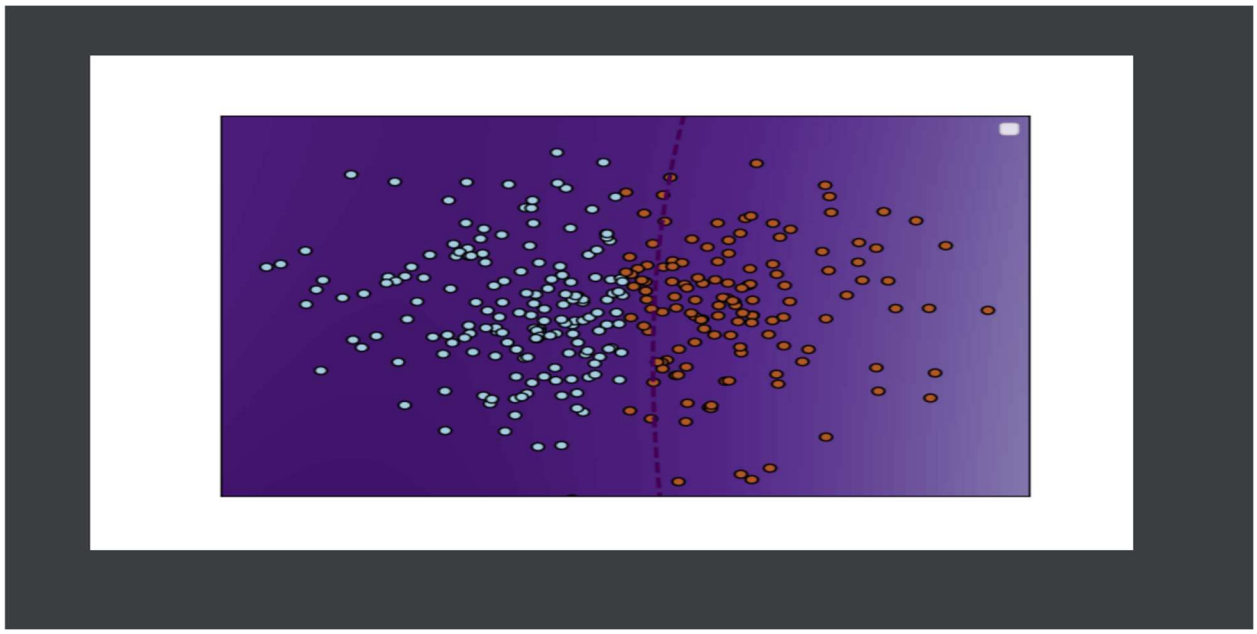
حالا تابع y را به شکل زیر پیاده سازی می کنیم:

$$y = \begin{cases} 0 & x_1 < 0 \\ 1 & \text{otherwise} \end{cases}$$

```
for tup in ['rbf', 'poly', 'sigmoid']:
    xx, yy = np.meshgrid(np.linspace(-10, 10, 500), np.linspace(-10, 10, 500))
    np.random.seed(0)
    X = np.random.randn(300, 2)
    Y = np.logical_not(X[:, 0] < 0)
    clf = svm.NuSVC(kernel=tup, gamma='auto', coef0=0.04, degree=2)
    clf.fit(X, Y)
    Z = clf.decision_function(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    plt.imshow(
        Z,
        interpolation="nearest",
        extent=(xx.min(), xx.max(), yy.min(), yy.max()),
        aspect="auto",
        origin="lower",
        cmap=plt.cm.PuOr_r,
    )
    plt.legend([tup])
    contours = plt.contour(xx, yy, Z, levels=[0], linewidths=2, linestyles="dashed")
    plt.scatter(X[:, 0], X[:, 1], s=30, c=Y, cmap=plt.cm.Paired, edgecolors="k")
    plt.xticks(())
    plt.yticks(())
    plt.axis([-3, 3, -3, 3])
    plt.show()
```

نتایج به ترتیب برای هسته های گاوسی، چند جمله ای و سیگموید به صورت زیر می شوند:





بخش دوم)

حال به بررسی عملکرد ماشین بردار پشتیبان بر روی پایگاه داده مورد نظر می پردازیم. پایگاه داده ای که ما انتخاب کردیم همان مشابه MNIST می باشد. این پایگاه داده Digit می باشد و به شکل زیر است:

```
digits = datasets.load_digits()
n_samples = len(digits.images)
data = digits.images.reshape((n_samples, -1))
clf = svm.NuSVC(kernel='rbf', gamma=0.001)
X_train, X_test, y_train, y_test = train_test_split(
    data, digits.target, test_size=0.5, shuffle=True
)
clf.fit(X_train, y_train)
predicted = clf.predict(X_test)
print(
    f"Classification report for classifier {clf}:\n"
    f"{metrics.classification_report(y_test, predicted)}\n"
)
##
```

دقت این پایگاه داده در طبقه بندی **زمانی به بیشینه خود می رسد که از ماشین بردار پشتیبان با هسته گاوسی و ضریب 0.001 استفاده شود. پس از تقسیم بندی داده ها به دو دسته تمرینی و تست به نسبت 50 ، 50 حاصل کار به صورت زیر در می آید:**

```
Classification report for classifier NuSVC(gamma=0.001):
              precision    recall  f1-score   support

           0       1.00      0.99      0.99        91
           1       0.91      0.99      0.95        87
           2       1.00      0.99      0.99        92
           3       0.99      0.95      0.97        84
           4       0.98      0.98      0.98        94
           5       0.97      0.98      0.97        95
           6       0.99      0.99      0.99        94
           7       0.86      1.00      0.92        74
           8       0.97      0.89      0.92        97
           9       0.96      0.89      0.93        91

    accuracy                           0.96       899
   macro avg       0.96      0.96      0.96       899
weighted avg       0.97      0.96      0.96       899
```

دقت آن در مقایسه با شبکه عصبی یکی است و تفاوت آن ایجاب میکند که در زمان بسیار کمتری ماشین بردار پشتیبان به نتیجه می رسد. در حالی که شبکه عصبی زمان بسیاری برای تمرین کردن نیاز دارد.

بخش سوم)

پایگاه داده انتخاب شده را ابتدا در قالب ماتریس در می آوریم و آن را متناظرا با اعداد 0 تا 4 طبقه بندی می کنیم. پس از آن و آزمون های فراوان ، هسته چند جمله ای را بر می گزینیم تا مقدار دقت بیشینه شود. دقت شود که داده ها به دو دسته آموزشی و آزمایشی تقسیم بندی می شوند و شافل می خورند و به شکل رندوم مورد آزمایش و تست قرار می گیرند.

```
X = []
y = []

mypath = './persian_LPR/'
class_list = ['2', '3', '7', 'S', 'W']
class_dict = {'2': 0, '3': 1, '7': 2, 'S': 3, 'W': 4}
for classes in ['2', '3', '7', 'S', 'W']:
    final = mypath + classes
    for file in listdir(final):
        X.append(skimage.io.imread(final + '/' + file).tolist())
        y.append(class_dict[classes])


X = np.reshape(np.array(X), (1500, -1))
y = np.array(y)
clf = svm.NuSVC(kernel='poly')
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, shuffle=True
)
clf.fit(X_train, y_train)
predicted = clf.predict(X_test)
print(
     f"Classification report for classifier {clf}:\n"
    f"{metrics.classification_report(y_test, predicted)}\n")
```

نتیجه کار نیز به شکل زیر می شود.

```
Classification report for classifier NuSVC(kernel='poly'):
              precision    recall  f1-score   support

           0       0.96      0.91      0.93       100
           1       0.91      0.99      0.95        80
           2       0.98      0.96      0.97        99
           3       0.94      0.93      0.94        86
           4       0.93      0.94      0.94        85

    accuracy                           0.94       450
   macro avg       0.94      0.95      0.94       450
weighted avg       0.95      0.94      0.94       450
```