SVM_C

April 3, 2022

1 Support Vector Machine (SVM) Classifier

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix,␣
↪ConfusionMatrixDisplay
```

1.0.1 Reading the dataset

[ ]:
```
col_names = ['timeStamp', 'Gender', 'Grade', 'Age', 'Length', 'Weight',␣
↪'ShoesSize']
dataframe = pd.read_csv("../human_features.csv", names = col_names,␣
↪skiprows=(0, ))
```

[ ]:
```
training_data = dataframe.loc[:, ['Age', 'Length', 'Gender']]
```

1.0.2 Label encoding

[ ]:
```
label_encoder = LabelEncoder()
training_data['Gender'] = label_encoder.fit_transform(training_data['Gender'])
```

1.0.3 Feature scaling

[ ]:
```
scaler = MinMaxScaler()
scaler.fit(training_data)
scaled = scaler.fit_transform(training_data)
scaled_df = pd.DataFrame(scaled, columns=training_data.columns)
```

[ ]:
```
X = scaled_df.loc[:, scaled_df.columns != 'Gender']
Y = scaled_df.loc[:, 'Gender']
```

[ ]:
```
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.33,␣
↪random_state=0)
```

1.0.4 Creating the classifier

[ ]:
```
clf = SVC(kernel='linear', C=1000)
```

1.0.5 Training the classifier

[ ]:
```
clf.fit(x_train.values, y_train.values)
```

[ ]:
```
SVC(C=1000, kernel='linear')
```

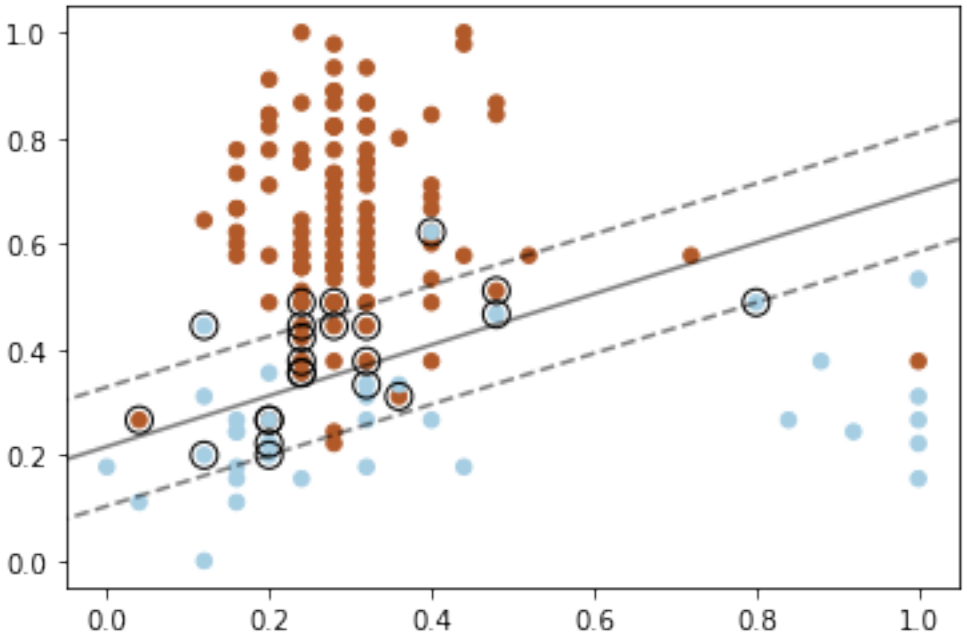
2 Drawing the support vectors

[ ]:
```
plt.scatter(X.iloc[:, 0], X.iloc[:, 1], c=Y, s=30, cmap=plt.cm.Paired)

# plot the decision function
ax = plt.gca()
xlim = ax.get_xlim()
ylim = ax.get_ylim()

# create grid to evaluate model
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)

# plot decision boundary and margins
ax.contour(
    XX, YY, Z, colors="k", levels=[-1, 0, 1], alpha=0.5, linestyles=["--", "-",␣
↪"--"]
)
# plot support vectors
ax.scatter(
    clf.support_vectors_[:, 0],
    clf.support_vectors_[:, 1],
    s=100,
    linewidth=1,
    facecolors="none",
    edgecolors="k",
)
plt.show()
```



2.1 Prediction

[ ]:
```
y_predict = clf.predict(x_test.values)
```

2.1.1 Accuracy

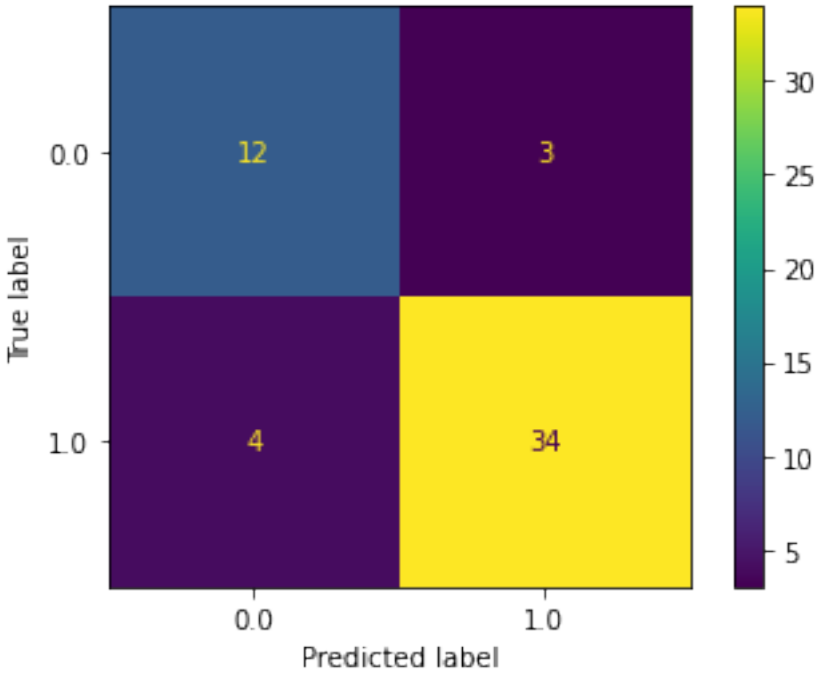
[ ]:
```
round((accuracy_score(y_test, y_predict) * 100), ndigits=2)
```

[ ]:
```
86.79
```

2.2 Confusion Matrix

[ ]:
```
cm = confusion_matrix(y_test, y_predict, labels=clf.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_)
disp.plot()
```

[ ]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x7f7c8535ba30>
```


12
3
4
34
0.0
1.0
True label
0.0
1.0
Predicted label
30
25
20
15
10
5