SVM_Kernelss_Implementation

March 11, 2021

0.1 SVM (Support Vector Machine) Kernels Indepth Intuition And Practical Explanation

SVM (Support Vector Machine) Kernal: Implementation Using Sklearn- Machine Learning SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types. For example linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid.

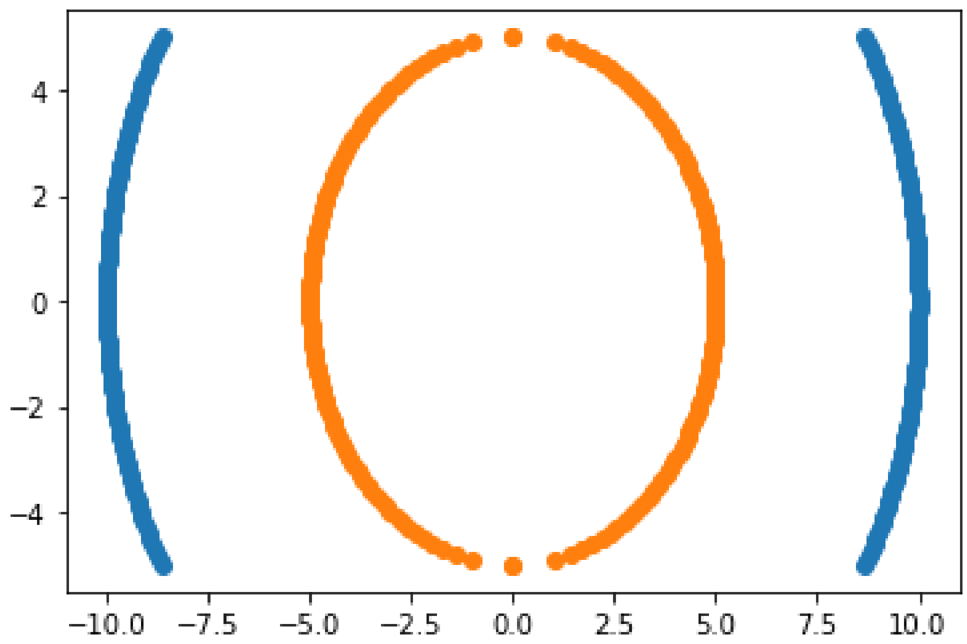
[1]:
```
#importing required libreries

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

[2]:
```
# Creating data
x = np.linspace(-5.0, 5.0, 100)
y = np.sqrt(10**2 - x**2)
y=np.hstack([y,-y])
x=np.hstack([x,-x])
```

[3]:
```
x1 = np.linspace(-5.0, 5.0, 100)
y1 = np.sqrt(5**2 - x1**2)
y1=np.hstack([y1,-y1])
x1=np.hstack([x1,-x1])
```

[4]:
```
# Visualization
plt.scatter(y,x)
plt.scatter(y1,x1)
```

[4]:
```
<matplotlib.collections.PathCollection at 0x26bf80c4bb0>
```



[5]:

```
df1 =pd.DataFrame(np.vstack([y,x]).T,columns=['X1','X2'])
df1['Y']=0
df2 =pd.DataFrame(np.vstack([y1,x1]).T,columns=['X1','X2'])
df2['Y']=1
df = df1.append(df2)
df.head(5)
```

[5]:

```
         X1       X2  Y
0  8.660254 -5.00000  0
1  8.717792 -4.89899  0
2  8.773790 -4.79798  0
3  8.828277 -4.69697  0
4  8.881281 -4.59596  0
```

[6]:

```
### Independent and Dependent features
X = df.iloc[:, :2]
y = df.Y
```

[7]:

```
y
```

[7]:

```
0      0
1      0
2      0
3      0
4      0
      ..
```

```
195    1
196    1
197    1
198    1
199    1
Name: Y, Length: 400, dtype: int64
```

[8]:

```
## Split the dataset into train and test
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
 ↪25,random_state=0)
```

[9]:

```
y_train
```

[9]:

```
50     1
63     0
112    1
159    0
83     1
      ..
123    1
192    0
117    0
47     0
172    0
Name: Y, Length: 300, dtype: int64
```

[10]:

```
from sklearn.svm import SVC
classifier=SVC(kernel="rbf")
classifier.fit(X_train,y_train)
```

[10]:

```
SVC()
```

[11]:

```
from sklearn.metrics import accuracy_score
y_pred = classifier.predict(X_test)
accuracy_score(y_test, y_pred)
```

[11]:

```
1.0
```

[12]:

```
df.head()
```

[12]:

```
         X1       X2  Y
0  8.660254 -5.00000  0
1  8.717792 -4.89899  0
2  8.773790 -4.79798  0
3  8.828277 -4.69697  0
4  8.881281 -4.59596  0
```

0.1.1 Polynomial Kernel

0.2 K(x, y) $= (x^T y + c)^d$

[13]:

```
# We need to find components for the Polynomical Kernel
#X1,X2,X1_square,X2_square,X1*X2
df['X1_Square']= df['X1']**2
df['X2_Square']= df['X2']**2
df['X1*X2'] = (df['X1'] *df['X2'])
df.head()
```

[13]:

```
         X1       X2  Y  X1_Square  X2_Square      X1*X2
0  8.660254 -5.00000  0  75.000000  25.000000 -43.301270
1  8.717792 -4.89899  0  75.999898  24.000102 -42.708375
2  8.773790 -4.79798  0  76.979390  23.020610 -42.096467
3  8.828277 -4.69697  0  77.938476  22.061524 -41.466150
4  8.881281 -4.59596  0  78.877155  21.122845 -40.818009
```

[14]:

```
### Independent and Dependent features
X = df[['X1','X2','X1_Square','X2_Square','X1*X2']]
y = df['Y']
```

[15]:

```
y
```

[15]:

```
0      0
1      0
2      0
3      0
4      0
      ..
195    1
196    1
197    1
198    1
199    1
Name: Y, Length: 400, dtype: int64
```

[16]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25,␣
 ↪random_state = 0)
```

[17]:

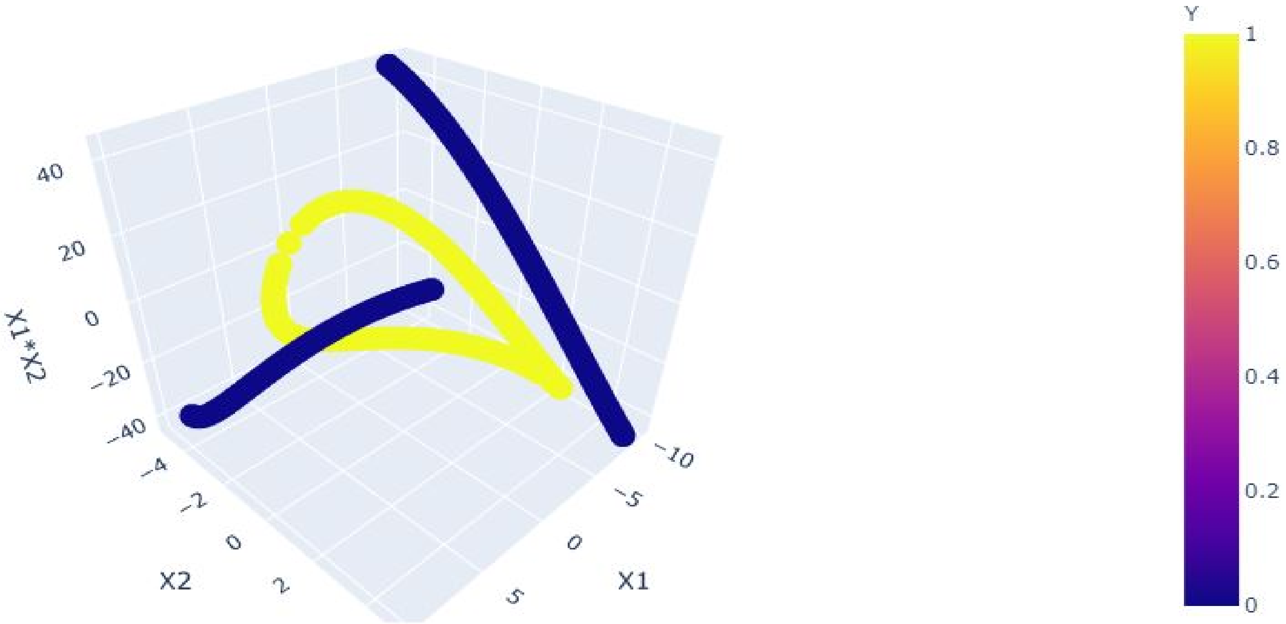
```
X_train
```

[17]:

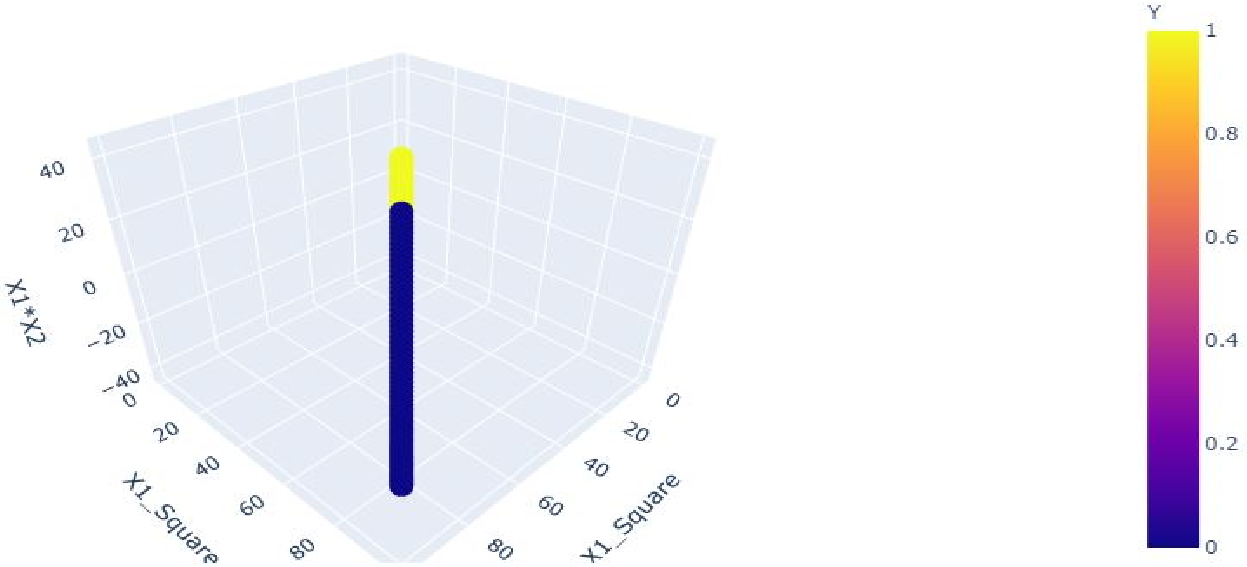
```
           X1        X2  X1_Square  X2_Square      X1*X2
50   4.999745  0.050505  24.997449   0.002551   0.252512
63   9.906589  1.363636  98.140496   1.859504  13.508984
112 -3.263736  3.787879  10.651974  14.348026 -12.362637
159 -9.953852 -0.959596  99.079176   0.920824   9.551676
83   3.680983  3.383838  13.549638  11.450362  12.455852
```

```
..          ...        ...         ...        ...        ...
123 -4.223140  2.676768  17.834915   7.165085 -11.304366
192 -9.031653 -4.292929  81.570758  18.429242  38.772248
117 -9.445795  3.282828  89.223038  10.776962 -31.008922
47   9.996811 -0.252525  99.936231   0.063769  -2.524447
172 -9.738311 -2.272727  94.834711   5.165289  22.132526

[300 rows x 5 columns]
```

[18]:
```
import plotly.express as px

fig = px.scatter_3d(df, x='X1', y='X2', z='X1*X2', color='Y')
fig.show()
```



[19]:
```
fig = px.scatter_3d(df, x='X1_Square', y='X1_Square', z='X1*X2', color='Y')
fig.show()
```



[20]:
```
classifier = SVC(kernel="linear")
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
accuracy_score(y_test, y_pred)
```

[20]: 1.0