دوره دیتا ساینس کاربردی

Machine Learning

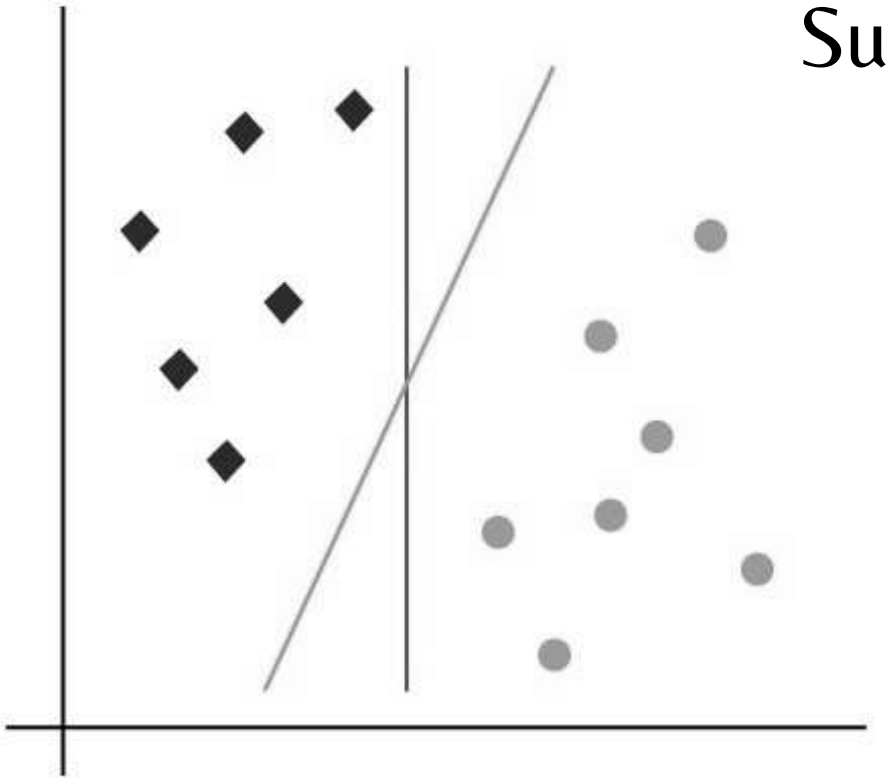
Support Vector Machine

dataroadmap

مدرس: مونا حاتمی

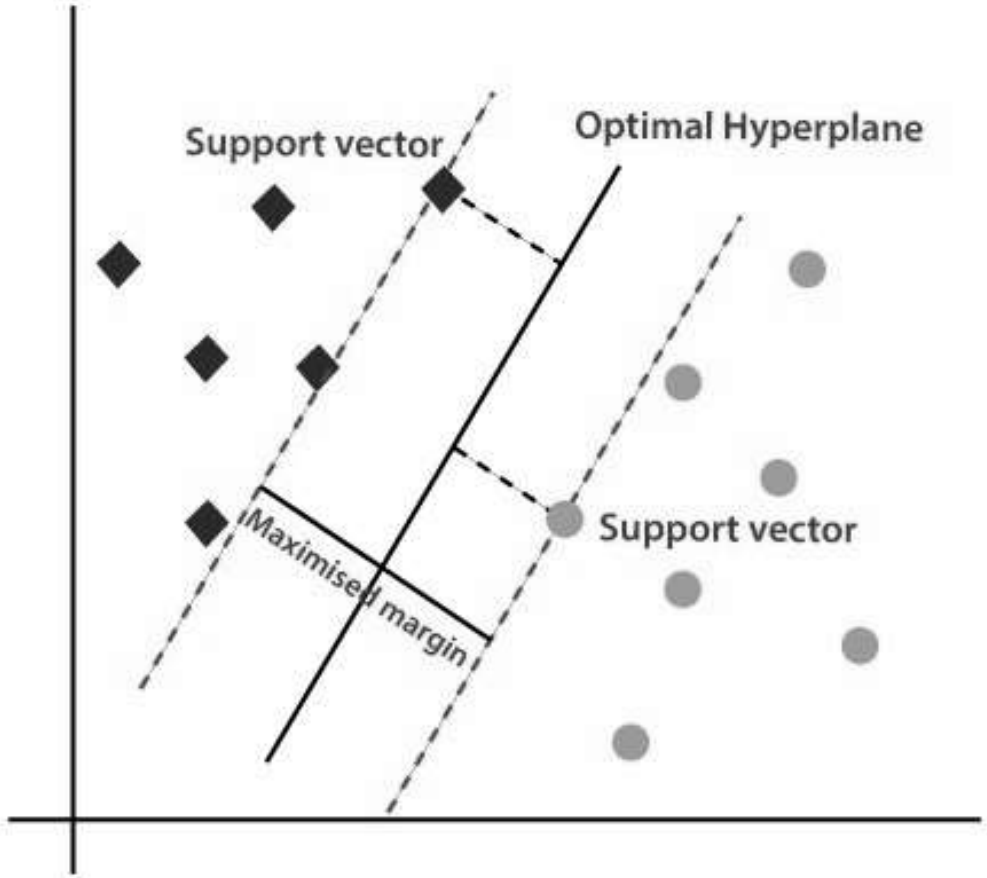
جلسه نهم

Support Vector Machine (SVM)

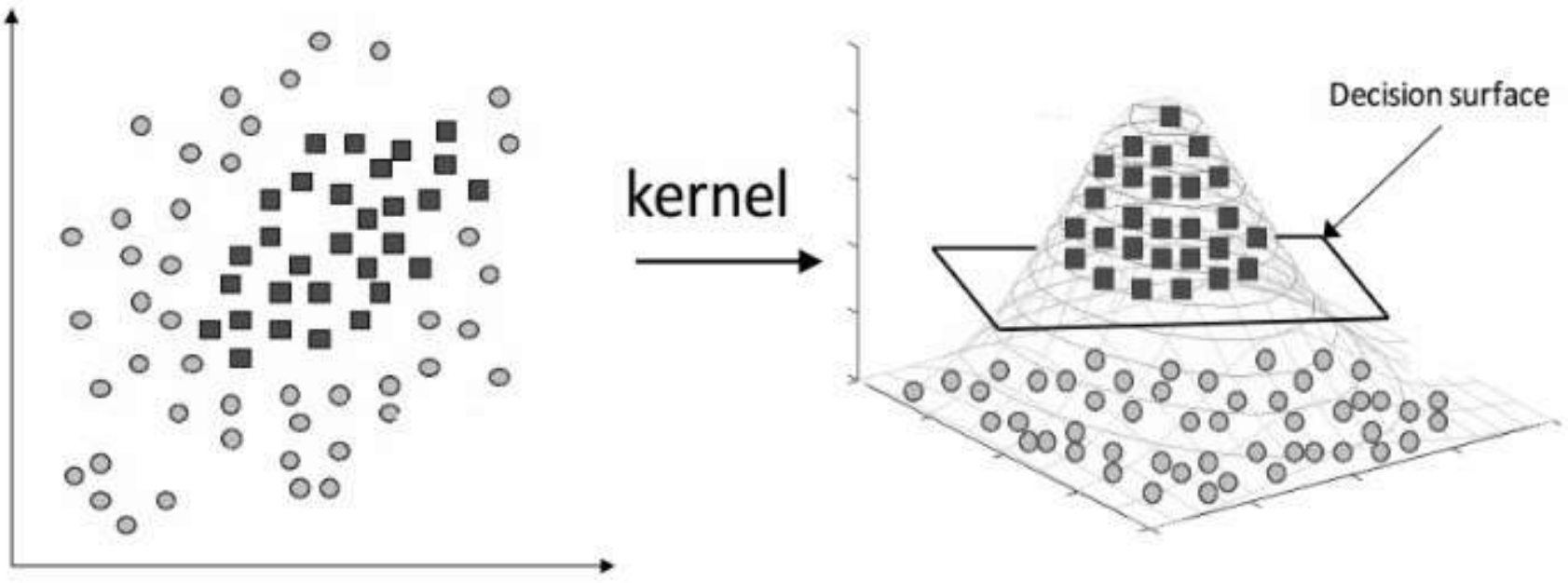


Supervised Learning

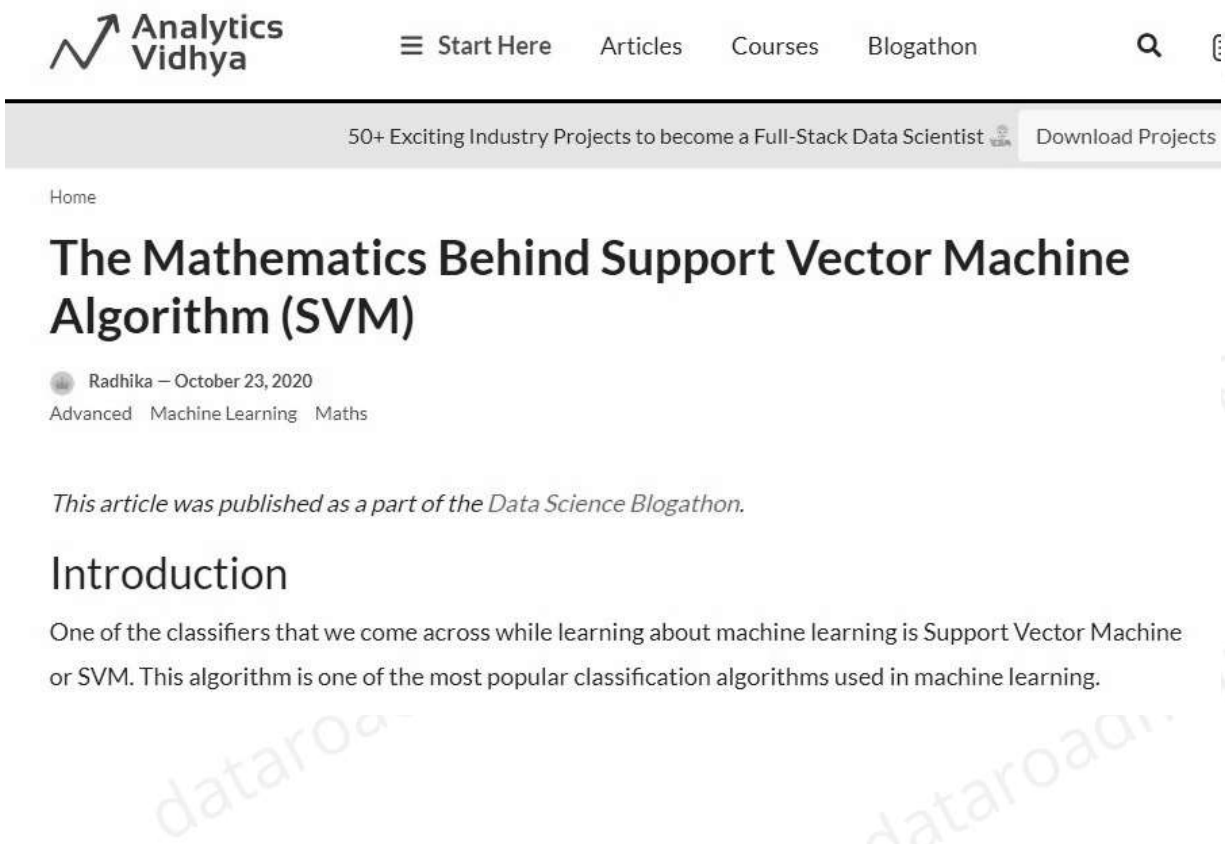
Support Vector Machine (SVM)



Support Vector Machine (SVM)



Reference



Analytics Vidhya

Start Here Articles Courses Blogathon

50+ Exciting Industry Projects to become a Full-Stack Data Scientist Download Projects

Home

The Mathematics Behind Support Vector Machine Algorithm (SVM)

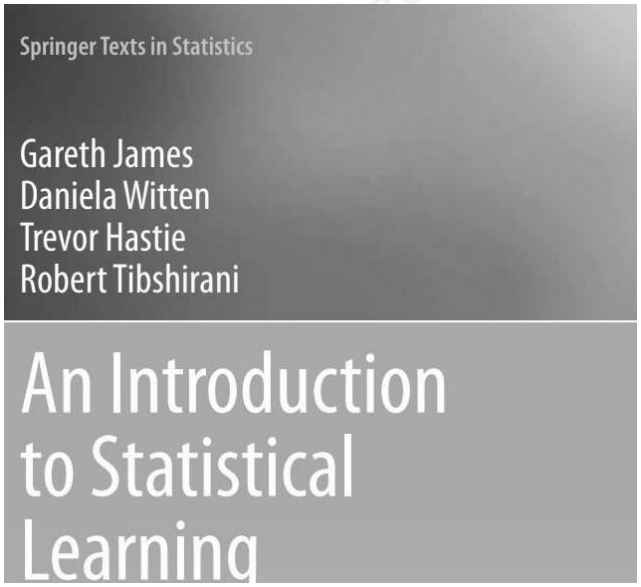
Radhika — October 23, 2020

Advanced Machine Learning Maths

This article was published as a part of the Data Science Blogathon.

Introduction

One of the classifiers that we come across while learning about machine learning is Support Vector Machine or SVM. This algorithm is one of the most popular classification algorithms used in machine learning.



Springer Texts in Statistics

Gareth James
Daniela Witten
Trevor Hastie
Robert Tibshirani

An Introduction to Statistical Learning

Read the Data

```
import pandas as pd
```

The Data ¶

```
df = pd.read_csv('preprocessed_dataset.csv')
```

```
df.head()
```

[3]:

	Unnamed: 0	PayloadMass	Flights	GridFins	Reused	Legs	Block	ReusedCou
0	0	6104.959412	1	0	0	0	1.0	
1	1	525.000000	1	0	0	0	1.0	
2	2	677.000000	1	0	0	0	1.0	

EDA

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90 entries, 0 to 89
Data columns (total 89 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Unnamed: 0                              90 non-null     int64
 1   PayloadMass                             90 non-null     float64
 2   Flights                                 90 non-null     int64
```

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

Train set- Test set in sklearn library

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y, test_size=0.2,
                                                    random_state=101)
```

Instantiate the model and Train

In [8]:
```
# Support Vector Machine classification algorithm
from sklearn.svm import SVC
```

In [9]:
```
svm = SVC()
```

In [10]:
```
svm.fit(X_train,y_train)
```

Out[10]:
```
SVC()
```

Prediction vs y_test

```
In [11]: predictions = svm.predict(X_test)

In [12]: predictions

Out[12]: array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], dtype=int64)

In [13]: y_test

Out[13]: 50    0
         6     1
         51    0
         54    1
         53    1
         69    1
         32    1
         31    1
         21    1
```

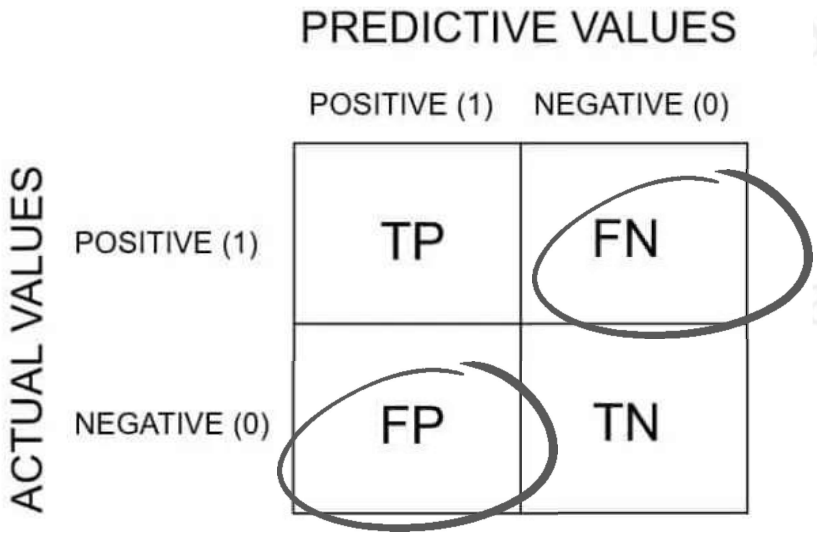
Evaluation

In [14]:
```
from sklearn.metrics import confusion_matrix
```

In [15]:
```
confusion_matrix(y_test,predictions)
```

Out[15]:
```
array([[ 0,  7],
       [ 0, 11]], dtype=int64)
```

Confusion Matrix Error I & Error II




```
([[ 0,  7],
  [ 0, 11]],
```

Evaluation

```
In [16]: from sklearn.metrics import accuracy_score

In [17]: accuracy_score(y_test,predictions, normalize=False)

Out[17]: 11

In [18]: accuracy_score(y_test,predictions, normalize=True)

Out[18]: 0.6111111111111112
```

Classification Report

In [19]:
```
from sklearn.metrics import classification_report
```

In [20]:
```
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         7
           1       0.61      1.00      0.76        11

    accuracy                           0.61        18
   macro avg       0.31      0.50      0.38        18
weighted avg       0.37      0.61      0.46        18
```

Hyperparameters

```
In [22]: svm_1 = SVC()

In [23]: Init signature:
         SVC(
             *,
             C=1.0,
             kernel='rbf',
             degree=3,
             gamma='scale',
             coef0=0.0,
             shrinking=True,
             probability=False,

In [24]:

Out[24]:
```

Grid Search-Hyperparameters

In [21]:
```
# Allows us to test parameters of classification algorithms and find the best
from sklearn.model_selection import GridSearchCV
```

In [22]:
```
svm_1 = SVC()
```

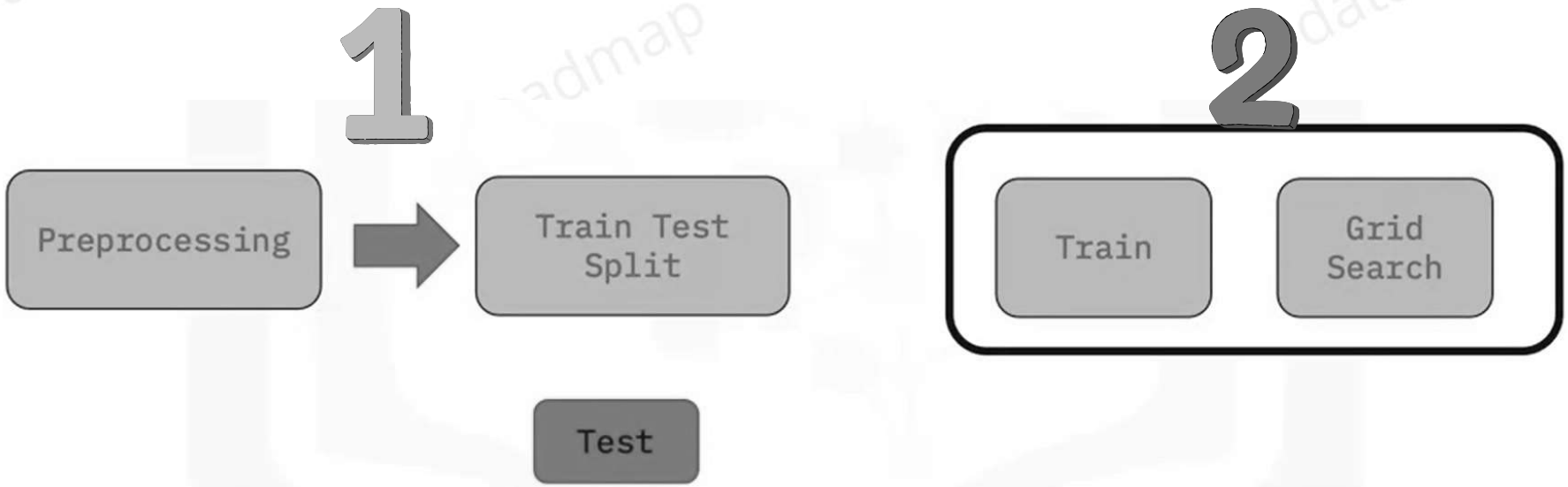
In [23]:
```
parameters = {'C': [0.5, 1, 1.5],
              'kernel':['linear', 'rbf','sigmoid']}
```

In [24]:
```
svm_cv = GridSearchCV(svm_1, parameters)
svm_cv.fit(X_train, y_train)
```

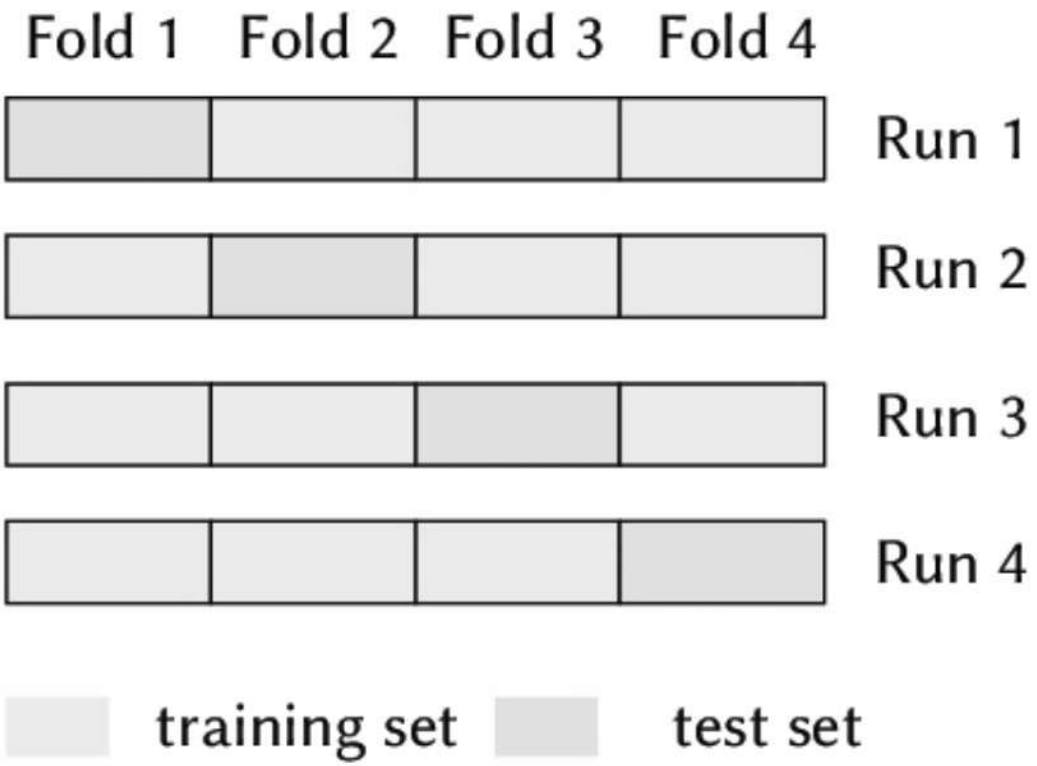
Out[24]:
```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [0.5, 1, 1.5],
                         'kernel': ('linear', 'rbf', 'sigmoid')})
```

How Grid Search find the best hyperparameters??

```
svm_cv = GridSearchCV(svm_1, parameters)
svm_cv.fit(X_train, y_train)
```



k-fold cross validation



cv for k-fold cross validation

```
svm_cv = GridSearchCV(svm_1, parameters, cv=4)
svm_cv.fit(X_train, y_train)
```

```
: GridSearchCV(cv=4, estimator=SVC(C=0.5, kernel='linear'),
               param_grid={'C': [0.5, 1, 1.5],
                           'kernel': ('linear', 'rbf', 'sigmoid')})
```

verbose

```
svm_cv = GridSearchCV(svm_1, parameters, cv=4, verbose=3 )
svm_cv.fit(X_train, y_train)
```

```
Fitting 4 folds for each of 9 candidates, totalling 36 fits
[CV 1/4] END ..............C=0.5, kernel=linear;, score=0.889 total time= 1.4min
[CV 2/4] END ..............C=0.5, kernel=linear;, score=0.833 total time=   8.1s
[CV 3/4] END ..............C=0.5, kernel=linear;, score=0.833 total time=   6.9s
[CV 4/4] END ..............C=0.5, kernel=linear;, score=1.000 total time=   5.2s
[CV 1/4] END ................C=0.5, kernel=rbf;, score=0.667 total time=   0.0s
[CV 2/4] END ................C=0.5, kernel=rbf;, score=0.667 total time=   0.0s
[CV 3/4] END ................C=0.5, kernel=rbf;, score=0.667 total time=   0.0s
[CV 4/4] END ................C=0.5, kernel=rbf;, score=0.722 total time=   0.0s
[CV 1/4] END ............C=0.5, kernel=sigmoid;, score=0.556 total time=   0.0s
[CV 2/4] END ............C=0.5, kernel=sigmoid;, score=0.444 total time=   0.0s
[CV 3/4] END ............C=0.5, kernel=sigmoid;, score=0.611 total time=   0.0s
[CV 4/4] END ............C=0.5, kernel=sigmoid;, score=0.722 total time=   0.0s
[CV 1/4] END ................C=1, kernel=linear;, score=0.944 total time=  31.3s
[CV 2/4] END ................C=1, kernel=linear;, score=0.833 total time=   1.1s
[CV 3/4] END ................C=1, kernel=linear;, score=0.833 total time=   7.6s
```

Tune the model with best hyperparameters

In [25]:
```
print("tuned hpyerparameters :(best parameters) ",svm_cv.best_params_)
```

```
tuned hpyerparameters :(best parameters)  {'C': 0.5, 'kernel': 'linear'}
```

In [44]:
```
svm_1 = SVC(C= 0.5, kernel= 'linear')
```

In [45]:
```
svm_1.fit(X_train,y_train)
```

Out[45]:
```
SVC(C=0.5, kernel='linear')
```

Evaluate

```
In [46]: predictions_1 = svm_1.predict(X_test)
```

```
In [47]: confusion_matrix(y_test,predictions_1)
```

```
Out[47]: array([[ 5,  2],
                [ 1, 10]], dtype=int64)
```

```
In [48]: accuracy_score(y_test,predictions_1, normalize=False)
```

```
Out[48]: 15
```

Evaluate

```
In [49]: accuracy_score(y_test,predictions_1, normalize=True)

Out[49]: 0.8333333333333334
```

```
In [50]: print(classification_report(y_test,predictions))
```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	7
1	0.61	1.00	0.76	11
accuracy			0.61	18
macro avg	0.31	0.50	0.38	18
weighted avg	0.37	0.61	0.46	18

Assignment:

تمرین:

کدهای ارائه شده در درس را در نوتبوک جدیدی انجام داده و در صورت نیاز از نوتبوک هفته نهم استفاده کنید.

جلسه دوازدهم دوره منتورینگ دیتاساینس در کانال یوتیوب را انجام دهید.

رزومه خود را آپدیت کرده و مهارتهایی که تا به امروز فراگرفته اید را اضافه کنید.