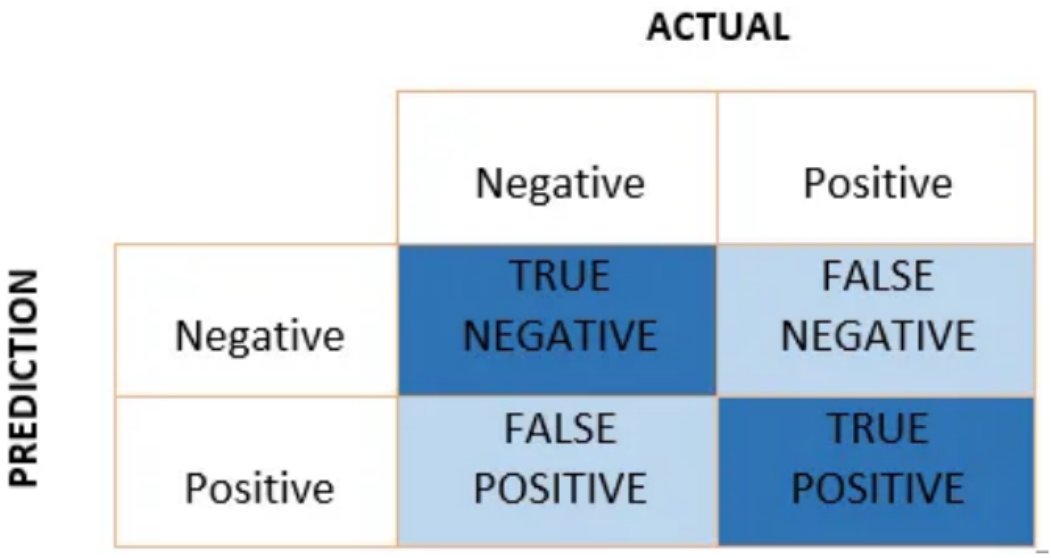
machine learning - confusion matrix and cross validation

CONFUSION MATRIX:

It is a table that is used in classification problems to assess where errors in the model were made.

The confusion matrix gives better intuition of prediction results as compared to accuracy .



Confusion Matrix

True Positive (TP) — model correctly predicts the positive class

True Negative (TN) — model correctly predicts the negative class

False Positive (FP) — model gives the wrong prediction of the negative class

False Negative (FN) — model wrongly predicts the positive class

note : Even if data is imbalanced, we can figure out that our model is working well or not. For that,
the values of TPR and TNR should be high, and FPR and FNR should be as low as possible.

example of a confusion matrix in python :

```
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
import numpy as np
import matplotlib.pyplot as plt

blobs_random_seed = 42
centers = [(0,0), (5,5), (0,5), (2,3)]
cluster_std = 1.3
frac_test_split = 0.33
num_features_for_samples = 4
num_samples_total = 500
```

Generate data

```
inputs, targets = make_blobs(n_samples = num_samples_total, centers = centers,
n_features = num_features_for_samples, cluster_std = cluster_std)
X_train, X_test, y_train, y_test = train_test_split(inputs, targets,
test_size=frac_test_split, random_state=blobs_random_seed)
```

Generate scatter plot for training data

```
plt.scatter(X_train[:,0], X_train[:,1])
plt.title('Linearly separable data')
plt.xlabel('X1')
plt.ylabel('X2')
plt.show()
```

```
from sklearn import svm
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

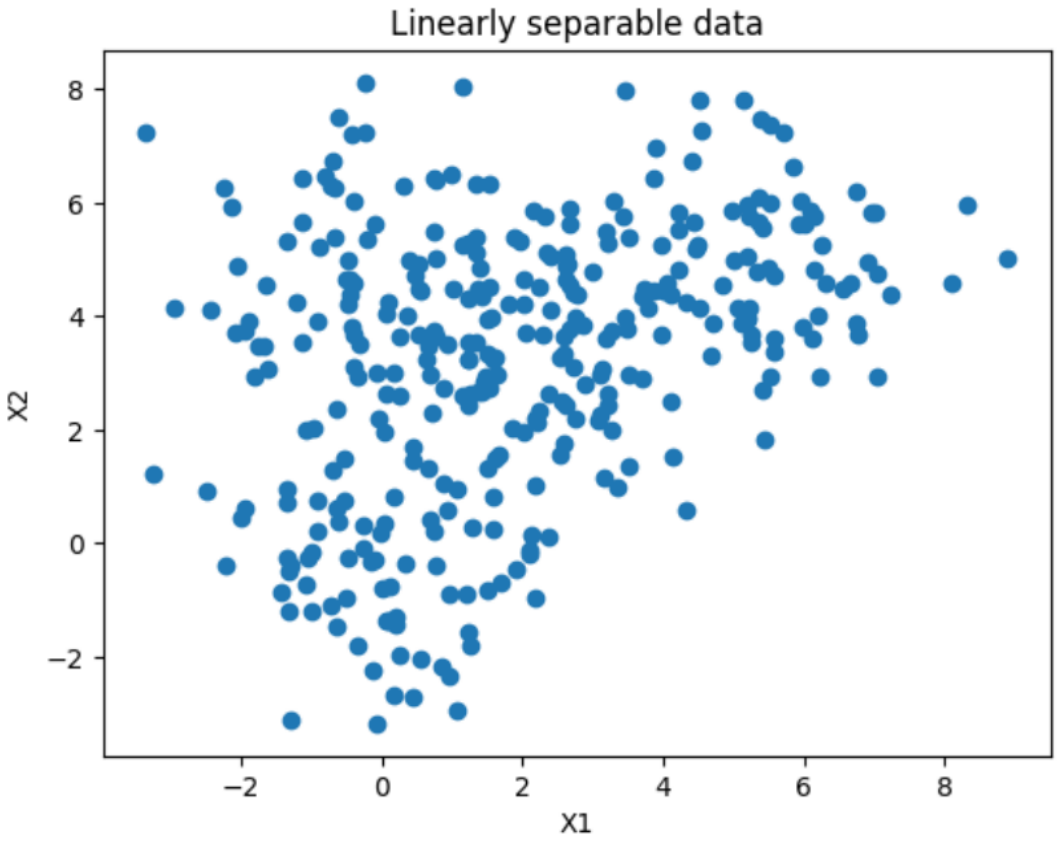
Initialize SVM classifier

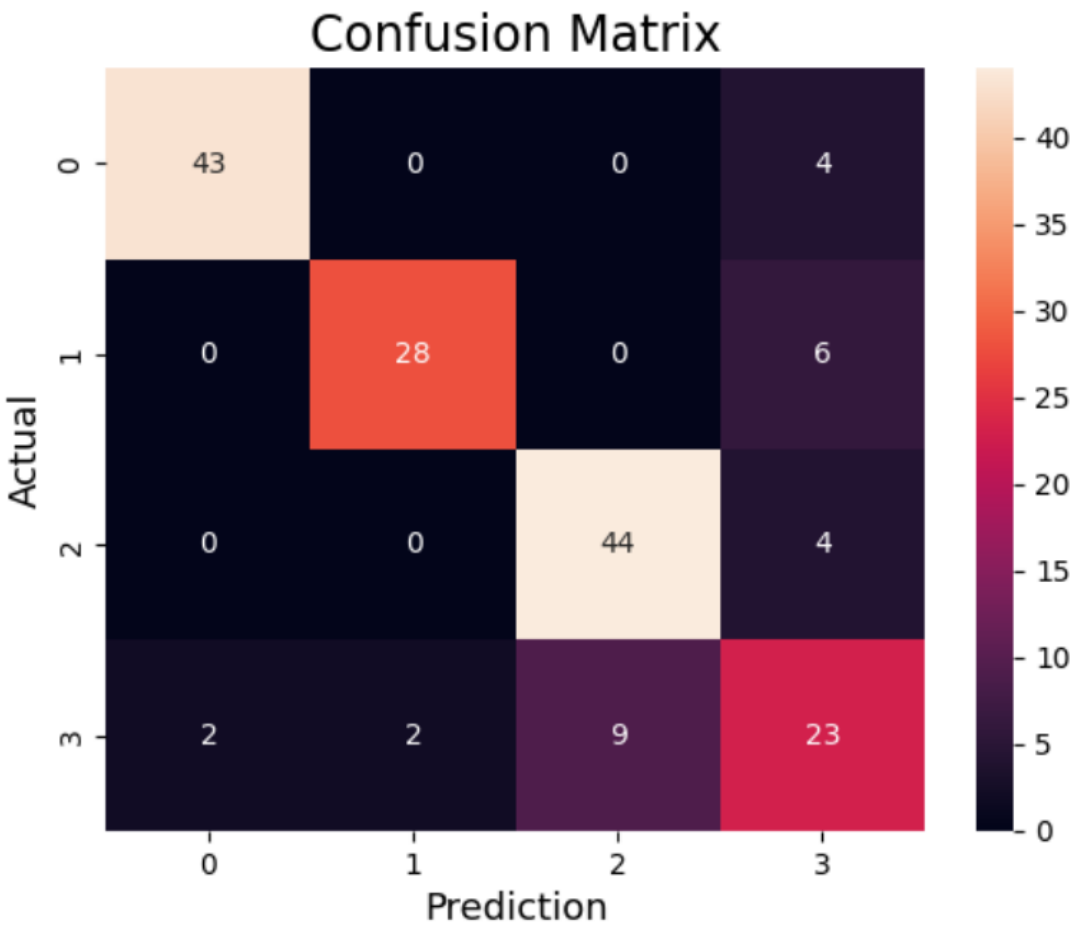
```
clf = svm.SVC(kernel='linear')
```

Fit data

```
clf = clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

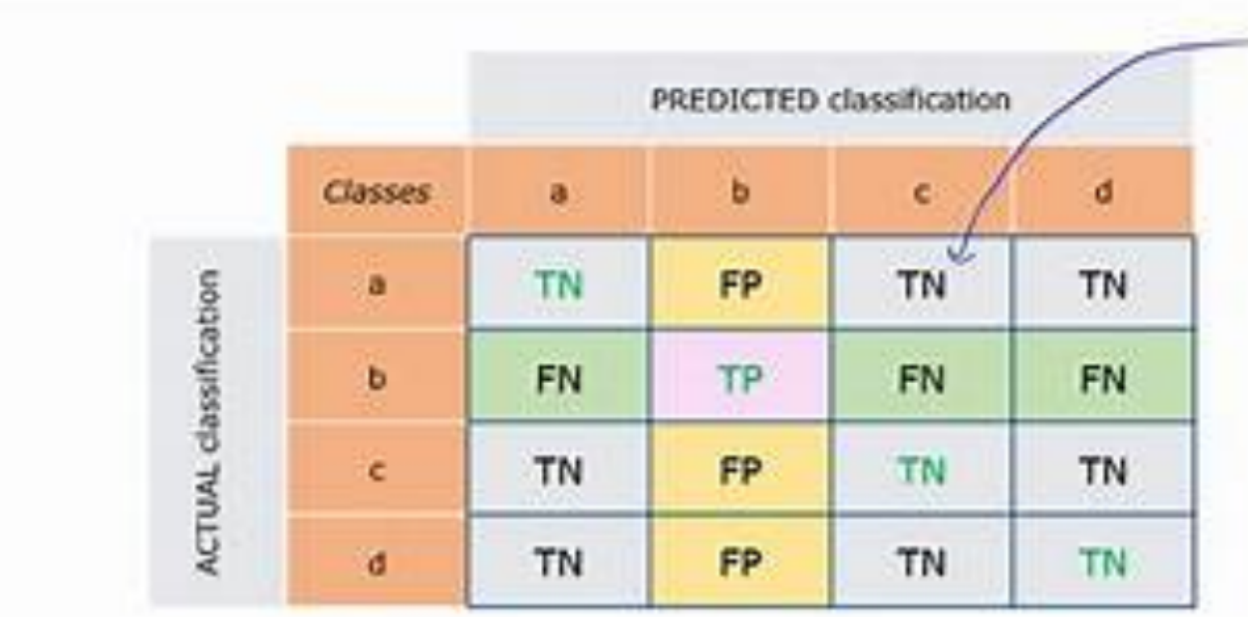
```
cm = confusion_matrix(y_test , y_pred)
sns.heatmap(cm,
annot=True,)
plt.xlabel('Prediction',fontsize=13)
plt.ylabel('Actual',fontsize=13)
plt.title('Confusion Matrix',fontsize=17)
plt.show()
```


Linearly separable data
X2
X1
8
6
4
2
0
−2
−2
0
2
4
6
8



this is how a confusion matrics predicts between actual and predicted values

reference for 4 × 4 confusion matrix is given below:



CROSS VALIDATION :

Cross validation is an important step in the **machine learning** process and helps to ensure that the model selected for deployment is robust and generalizes well to new data.

The main purpose of cross validation is to prevent **overfitting**, which occurs when a model is trained too well on the training data and performs poorly on new, unseen data. By evaluating the model on multiple validation sets, cross validation provides a more realistic estimate of the model's generalization performance, i.e., its ability to perform well on new, unseen data.

types of cross validation are :

k-fold cross validation

leave-one-out cross validation

Holdout validation

Stratified Cross-Validation

1. Holdout Validation

In **Holdout Validation**, we perform training on the 50% of the given dataset and rest 50% is used for the testing purpose. It's a simple and quick way to evaluate a model. The major drawback of this method is that we perform training on the 50% of the dataset, it may possible that the remaining 50% of the data contains some important information which we are leaving while training our model i.e. higher bias

2. LOOCV (Leave One Out Cross Validation)

In this method, we perform training on the whole dataset but leaves only one data-point of the available dataset and then iterates for each data-point. In LOOCV, the model is trained on

$$n - 1$$

samples and tested on the one omitted sample, repeating this process for each data point in the dataset. It has some advantages as well as disadvantages also.

An advantage of using this method is that we make use of all data points and hence it is low bias.

The major **drawback** of this method is that it leads to **higher variation** in the testing model as we are testing against one data point. If the data point is an outlier it can lead to higher variation. Another drawback is it **takes a lot of execution time** as it iterates over 'the number of data points' times.

3. Stratified Cross-Validation

It is a technique used in machine learning to ensure that each fold of the cross-validation process maintains the same class distribution as the entire dataset. This is particularly important when dealing with imbalanced datasets, where certain classes may be underrepresented. In this method,

1. The dataset is divided into k folds while maintaining the proportion of classes in each fold.
2. During each iteration, one-fold is used for testing, and the remaining folds are used for training.

3. The process is repeated k times, with each fold serving as the test set exactly once.

Stratified Cross-Validation is essential when dealing with classification problems where maintaining the balance of class distribution is crucial for the model to generalize well to unseen data.

4. K-Fold Cross Validation

In **K-Fold Cross Validation**, we split the dataset into k number of subsets (known as folds) then we perform training on the all the subsets but leave one(k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purpose each time.

example with a code:

```
from sklearn.model_selection import cross_val_score, KFold
from sklearn.svm import SVC
from sklearn.datasets import load_iris

iris = load_iris()
X , y = iris.data , iris.target

svm_classifier = SVC(kernel = 'linear')

num_folds = 5
kf = KFold(n_splits = num_folds , shuffle =True , random_state = 42)

cross_val = cross_val_score(svm_classifier , X , y , cv = kf)


print(f'Cross-Validation Results (Accuracy): {cross_val}')
print(f'Mean Accuracy: {cross_val.mean()}')
```

```
Cross-Validation Results (Accuracy): [1.         1.         0.96666667 0.93333333 0.96666667]
Mean Accuracy: 0.9733333333333334
```

note:

- After partitioning data set into two sets, training set is used to build a model/classifier.
- After construction of classifier, we use data items in test set, to test accuracy, error rate and error estimate of model/classifier.