```
#IRIS Calculate precision and accuracy using KNN Classifier

#Without inbuilt libraries

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report

df=pd.read_csv("/content/Iris.csv",header="infer").values

x=df[:,1:-1]
y=df[:,-1:]

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,stratify=y)

nclasses=np.unique(y_train).shape[0]
dist=np.zeros(shape=x_train.shape[0])
pred=np.zeros(shape=x_test.shape[0])
classvotes=np.zeros(shape=nclasses)

K=int(input("Enter the number of nearest neighbours 'K' :- "))
for i in range(x_test.shape[0]):
  dist=np.sqrt(np.sum((x_train-x_test[i])**2,axis=1))
  kminind=np.argpartition(dist,K)[0:K]
  invdist=1/(dist + 10e-20)
  denom=sum(invdist[kminind])
  for j in range(K):
    classvotes[int(y_train[kminind[j]])]+=invdist[kminind[j]]
  classvotes/=denom
  pred[i]=np.argmax(classvotes)

def calc_acc(y_pred,y_true):
  return np.sum((y_pred).astype(int)==(y_true).astype(int))/y_pred.shape[0]
def calc_prec(y_pred,y_true):
  classes=np.unique(y_true)
  nclasses=classes.shape[0]
  nrows=y_true.shape[0]
  classprop=np.zeros(shape=nclasses)

  for i in range(nclasses):
    classprop[i]=np.sum(y_true==classes[i])/nrows
    preclasswise=np.zeros(shape=nclasses)
    prec=0
  for i in range(nclasses):
    preindices=np.where((((y_pred).astype(int)==(classes[i].astype(int)))==True))
    trueindices=np.where((((y_true).astype(int)==(classes[i].astype(int)))==True)
    preclasswise[i]=((len(preindices[0]))-(len(set(preindices[0])-set(trueindices
    print(preclasswise[i])
    prec+=preclasswise[i]*classprop[i]
    return prec

prec=calc_prec(pred,y_test)
accuracy=calc_acc(pred,y_test)
```

```
accuracy=calc_acc(pred,y_test)

print("Classification Report")
print(classification_report(y_test,pred))
print("Accuracy is :- ",accuracy)
print("Original Class is :- ",y_test)
print("Predicted Class is :- ",pred)
print("Precision is :- ",prec)
```

```
Enter the number of nearest neighbours 'K' :- 2
1.0
Classification Report
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        10
         1.0       0.89      0.80      0.84        10
         2.0       0.82      0.90      0.86        10

    accuracy                           0.90        30
   macro avg       0.90      0.90      0.90        30
weighted avg       0.90      0.90      0.90        30

Accuracy is :-  10.0
Original Class is :-  [[1.]
 [1.]
 [2.]
 [0.]
 [0.]
 [0.]
 [0.]
 [0.]
 [2.]
 [1.]
 [1.]
 [1.]
 [0.]
 [2.]
 [1.]
 [1.]
 [2.]
 [2.]
 [0.]
 [0.]
 [0.]
 [2.]
 [2.]
 [1.]
 [1.]
 [2.]
 [2.]
 [1.]
 [0.]
 [2.]]
Predicted Class is :-  [1. 1. 2. 0. 0. 0. 0. 0. 2. 1. 1. 1. 0. 2. 2. 1. 1. 2. 0. 0.
 2. 2. 2. 1. 0. 2.]
Precision is :-  0.3333333333333333
```

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score,classification_report,precision_score

df=pd.read_csv("/content/Iris.csv",header="infer").values

x=df[:,1:-1]
y=df[:,-1:]

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,stratify=y)
k=int(input("Enter the value of K :- "))

model=KNeighborsClassifier(n_neighbors=k,weights="distance")
model.fit(x_train,y_train)
pred=model.predict(x_test)
accuracy=accuracy_score(y_test,pred)
prec=precision_score(y_test,pred,average="weighted")

print("Accuracy is :- ",accuracy)
print("Classification Report :- ")
print(classification_report(y_test,pred))
print("Precision is :- ",prec)
```

```
Enter the value of K :- 3
Accuracy is :-  0.9333333333333333
Classification Report :- 
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        10
         1.0       1.00      0.80      0.89        10
         2.0       0.83      1.00      0.91        10

    accuracy                           0.93        30
   macro avg       0.94      0.93      0.93        30
weighted avg       0.94      0.93      0.93        30

Precision is :-  0.9444444444444445
/usr/local/lib/python3.10/dist-packages/sklearn/neighbors/_classification.py:215: Da
  return self._fit(X, y)
```

✓ 3s completed at 11:58 AM