HELP MANUAL

DDCN-2019@CSED, MNNIT Allahabad

Supervised Learning in Python

Classification using KNN Algorithm

- **Required packages to implement KNN Classification algorithm**
  - import pandas as pd
  - import numpy as np
  - import matplotlib.pyplot as plt
  - from sklearn.cross_validation import train_test_split
  - from sklearn.neighbors import KNeighborsClassifier
  - from sklearn.metrics import accuracy_score
  - from sklearn.cross_validation import cross_val_score

Classification using KNN Algorithm Contd….

- **Loading dataset**
  - Define Column Names in a List
  - Use read_csv() method
  - Eg:
    names=['sepal_length','sepal_width','petal_length','petal_width','class']
  - df=pd.read_csv('C:\\Users\\username\\Desktop\\MLA-2017\\IRIS.csv',header=None,names=names)

Classification using KNN Algorithm Contd….

- **Create design matrix X (Feature Set) and target vector Y from the dataset columns**
  - Use np.array(data) method
  - Eg: X=np.array(df.ix[:,0:4])
  - Y=np.array(df['class'])
- **Split the data retrieved from dataset into training sets (X_train, Y_train) and test sets (X_test, Y_test).**
- Use train_test_split(col1,col2,test_size)
- **eg:**
  X_train,X_test,Y_train,Y_test=train_test_split(X,y,test_size=0.33)

Classification using KNN Algorithm Contd….

- **Instantiating KNeighborsClassifier method with specified number of neighbors**
  - Use KNeighborsClassifier(number_neighbors)
  - Eg: knn=KNeighborsClassifier(n_neighbors=3)
- **Fiting Classifer model on training data set**
- Use knn.fit(X_train,Y_train) method
- Eg: knn.fit(X_train,Y_train)

Classification using KNN Algorithm Contd….

- **Predict the target vector using test data set**
  - Use predict(test_data) method
  - Eg: pred=knn.predict(X_test)
- **Evaluating accuracy of the model**
  - **Use accuracy_score(test_data, predicted_value) method**
  - print (' Accuracy score is',accuracy_score(Y_test, pred))

KNN Algorithm: Selection of Optimum K

- **Create list of knn neighbors having multiple elements**
  - **Use list() method to generate different k values**
  - **Eg:**
  - myList=list(range(1,50))
  - # subsetting just the odd ones
  - neighbors=filter(lambda x:x%2!=0, myList)

KNN Algorithm: Selection of Optimum K Contd…

- **Performing cross fold validation for each value of neighbors and store their performance in a list**
  - Use cross_val_score() method to measure performance of cross validation
  - Eg: # Defining empty list for holding cvscores
  - cv_scores=[]
  - # perform 10-fold cross validation
  - for k in neighbors:
  - knn=KNeighborsClassifier(n_neighbors=k) scores=cross_val_score(knn,X_train,Y_train,cv=10,scoring='accuracy')
  - cv_scores.append(scores.mean())

KNN Algorithm: Selection of Optimum K Contd…

- **Compute Mean Standard Error (MSE) for each value of the list cv_score.**
  - Eg: MSE=[1-x for x in cv_scores]
- **Determine best value of k**
  - Find the value of k having minimum MSE
  - Eg: optimal_k=neighbors[MSE.index(min(MSE))]
- Plotting Misclassification Error for all values of k
  - Eg: plt.plot(neighbors,MSE)

KNN Algorithm: Selection of Optimum K Contd…

- Changing labels of plot
  - Xaxis: use plt.xlabel('label_name')
  - Yaxis: use plt.ylabel('label_name')
  - Title: Use plt.title('label_name')
  - eg: plt.xlabel('Number of Neighbors k')
  - plt.ylabel('Misclassification Error')
- Displaying chart
- Use plt.show() method
- Eg: plt.show()