**Code KNN**

"KNN Model 0 & 1 Classification Heart Data"

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Display for up to 50 columns

pd.set\_option('display.max\_column',50)

# Import data excel file

data = pd.read\_excel('Heart Data1.xlsx')

print(data.shape)

# Prints columns names

print(list(data.columns))

# Gives all columns and irst 5 rows

data.head()

X=data.iloc[:,: 13]

y=data.iloc[:,13]

import seaborn as sns

data[y].value\_counts()

sns.countplot(x=y,data=data,palette='hls')

plt.show()

"Test with random 0 - 4"

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20, random\_state=4)

# Standardized data (normalize it)

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaler.fit(X\_train)

# Train & Test data

X\_train = scaler.transform(X\_train)

X\_test = scaler.transform(X\_test)

#Knn algorithm, use 5 for k

from sklearn.neighbors import KNeighborsClassifier

classifier = KNeighborsClassifier(n\_neighbors=3)

classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test)

#Import scikit-learn metrics module for accuracy calculation

from sklearn import metrics

# Model Accuracy, how often is the classifier correct?

print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))

from sklearn.metrics import classification\_report, confusion\_matrix

print(confusion\_matrix(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

import seaborn as sns

mat = confusion\_matrix(y\_test, y\_pred)

sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False)

plt.xlabel('true label')

plt.ylabel('predicted label');

error = []

for i in range(1, 16):

knn = KNeighborsClassifier(n\_neighbors=i)

knn.fit(X\_train, y\_train)

pred\_i = knn.predict(X\_test)

error.append(np.mean(pred\_i != y\_test))

plt.figure(figsize=(12, 6))

plt.plot(range(1, 16), error, color='red', linestyle='dashed', marker='o',

markerfacecolor='blue', markersize=10)

plt.title('Error Rate K Value')

plt.xlabel('K Value')

plt.ylabel('Mean Error')

# Code to perform Cross Calidation to see if we are overfitting the model

from sklearn import model\_selection

from sklearn.model\_selection import cross\_val\_score

kfold = model\_selection.KFold(n\_splits=10, random\_state=7)

classifier = KNeighborsClassifier(n\_neighbors=8)

scoring = 'accuracy'

results = model\_selection.cross\_val\_score(classifier, X\_train, y\_train, cv=kfold, scoring=scoring)

print("10-fold cross validation average accuracy: %.3f" % (results.mean()))

"KNN Model 0,1 & 2 Classification Heart Data"

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Display for up to 50 columns

pd.set\_option('display.max\_column',50)

# Import data excel file

data = pd.read\_excel('Heart Data2.xlsx')

print(data.shape)

# Prints columns names

print(list(data.columns))

# Gives all columns and irst 5 rows

data.head()

X=data.iloc[:,: 13]

y=data.iloc[:,13]

import seaborn as sns

data[y].value\_counts()

sns.countplot(x=y,data=data,palette='hls')

plt.show()

"Test with random 0 - 4"

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20, random\_state=4)

# Standardized data (normalize it)

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaler.fit(X\_train)

# Train & Test data

X\_train = scaler.transform(X\_train)

X\_test = scaler.transform(X\_test)

#Knn algorithm, use 5 for k

from sklearn.neighbors import KNeighborsClassifier

classifier = KNeighborsClassifier(n\_neighbors=1)

classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test)

#Import scikit-learn metrics module for accuracy calculation

from sklearn import metrics

# Model Accuracy, how often is the classifier correct?

print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))

from sklearn.metrics import classification\_report, confusion\_matrix

print(confusion\_matrix(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

import seaborn as sns

mat = confusion\_matrix(y\_test, y\_pred)

sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False)

plt.xlabel('true label')

plt.ylabel('predicted label');

error = []

for i in range(1, 16):

knn = KNeighborsClassifier(n\_neighbors=i)

knn.fit(X\_train, y\_train)

pred\_i = knn.predict(X\_test)

error.append(np.mean(pred\_i != y\_test))

plt.figure(figsize=(12, 6))

plt.plot(range(1, 16), error, color='red', linestyle='dashed', marker='o',

markerfacecolor='blue', markersize=10)

plt.title('Error Rate K Value')

plt.xlabel('K Value')

plt.ylabel('Mean Error')

# Code to perform Cross Calidation to see if we are overfitting the model

from sklearn import model\_selection

from sklearn.model\_selection import cross\_val\_score

kfold = model\_selection.KFold(n\_splits=10, random\_state=4)

classifier = KNeighborsClassifier(n\_neighbors=1)

scoring = 'accuracy'

results = model\_selection.cross\_val\_score(classifier, X\_train, y\_train, cv=kfold, scoring=scoring)

print("10-fold cross validation average accuracy: %.3f" % (results.mean()))

**Code SVM**

**"SVM Model Classification Heart Data 0 & 1"**

**# Import libraries**

**import pandas as pd**

**import numpy as np**

**from sklearn import preprocessing**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**from scipy import stats**

**# Display for up to 50 columns**

**pd.set\_option('display.max\_column',50)**

**data=pd.read\_excel('Heart Data1.xlsx')**

**print(data.shape)**

**# Prints columns names**

**print(list(data.columns))**

**# Gives all columns and irst 5 rows**

**data.head()**

**#Define X & y**

**X=data.iloc[:,: 13]**

**y=data.iloc[:,13]**

**# Import train\_test\_split function**

**from sklearn.model\_selection import train\_test\_split**

**# Split dataset into training set and test set**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y, test\_size=0.2,random\_state=0)**

**#Import svm model**

**from sklearn import svm**

**#Create a svm Classifier (Linear kernel)**

**model1 = svm.SVC(kernel='linear')**

**#Train the model using the training sets**

**model1.fit(X\_train, y\_train)**

**#Predict the response for test dataset**

**y\_pred = model1.predict(X\_test)**

**#Import scikit-learn metrics module for accuracy calculation**

**from sklearn import metrics**

**# Model Accuracy: how often is the classifier correct?**

**print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))**

**from sklearn.metrics import classification\_report, confusion\_matrix**

**print(confusion\_matrix(y\_test, y\_pred))**

**print(classification\_report(y\_test, y\_pred))**

**import seaborn as sns**

**mat = confusion\_matrix(y\_test, y\_pred)**

**sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False)**

**plt.xlabel('true label')**

**plt.ylabel('predicted label');**

**"SVM Model Classification Heart Data 0,1 & 2"**

**# Import libraries**

**import pandas as pd**

**import numpy as np**

**from sklearn import preprocessing**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**from scipy import stats**

**# Display for up to 50 columns**

**pd.set\_option('display.max\_column',50)**

**data=pd.read\_excel('Heart Data2.xlsx')**

**print(data.shape)**

**# Prints columns names**

**print(list(data.columns))**

**# Gives all columns and irst 5 rows**

**data.head()**

**#Define X & y**

**X=data.iloc[:,: 13]**

**y=data.iloc[:,13]**

**# Import train\_test\_split function**

**from sklearn.model\_selection import train\_test\_split**

**# Split dataset into training set and test set**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y, test\_size=0.2,random\_state=0)**

**#Import svm model**

**from sklearn import svm**

**#Create a svm Classifier (Linear kernel)**

**model1 = svm.SVC(kernel='linear')**

**#Train the model using the training sets**

**model1.fit(X\_train, y\_train)**

**#Predict the response for test dataset**

**y\_pred = model1.predict(X\_test)**

**#Import scikit-learn metrics module for accuracy calculation**

**from sklearn import metrics**

**# Model Accuracy: how often is the classifier correct?**

**print("Accuracy:",metrics.accuracy\_score(y\_test, y\_pred))**

**from sklearn.metrics import classification\_report, confusion\_matrix**

**print(confusion\_matrix(y\_test, y\_pred))**

**print(classification\_report(y\_test, y\_pred))**

**import seaborn as sns**

**mat = confusion\_matrix(y\_test, y\_pred)**

**sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False)**

**plt.xlabel('true label')**

**plt.ylabel('predicted label');**