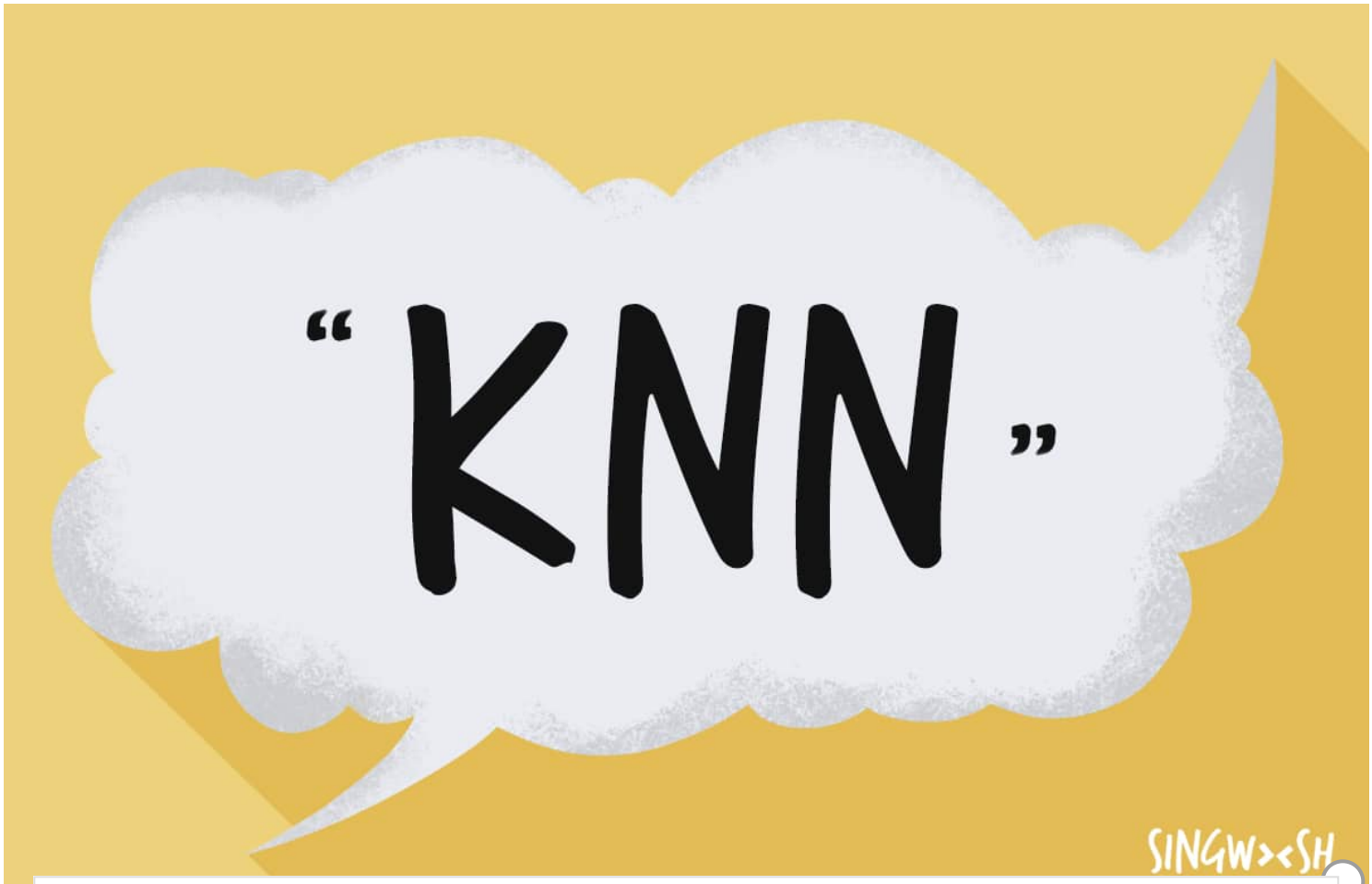
论文代码网址
https://obiirehman.wordpress.com/2017/05/12/knn-classifier-cross-validation/

DATA SCENE

MENU



April 2017

KNN Classifier & Cross Validation in Python

may 12, 2017 by obaid ur rehman, posted in python

In this post, I'll be using PIMA dataset to predict if a person is diabetic or not using **KNN Classifier** based on other features like age, blood pressure, tricep thikness e.t.c

糖尿病

KNN基于不同的特征

三头肌厚度

KNN

K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure. A case is classified by a majority vote of its neighbors, with the case being assigned to the class most common amongst its K nearest neighbors measured by a distance function. If K = 1, then the case is simply assigned to the class of its nearest neighbor.

相似性测度

一个案例是由它的邻居的多数投票来分类的，而这个案例被分配到它的K个近邻中最常见的类中，通过距离函数来测量。

You can study KNN in detail here:

KNN Classifier – WIKI

KNN Classifier-II

Dataset:

https://archive.ics.uci.edu/ml/datasets/pima+indians+diabetes

I am using **Pima Indians Diabetes Data Set.** Can be found here: **PIMA Dataset**

Let's Start

IMPORT REQUIRED LIBS

```
1 import pandas as pd
2 import numpy as np
3 import sklearn as skit
```

Data will look like this:

	nPregnant	glucoseConc	diastolicBP	tricepThikness	serumInsulin	BMI	diabeticPedigree	age	classlabel
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

独立的特征和因变量

SEPARATE FEATURES AND DEPENDENT VARIABLE(THE ONE WE WANT TO PREDICT)

```
features = ['diastolicBP','serumInsulin','BMI','glucoseConc','age']
to_predict='classlabel'
```

标准化数据

NORMALIZE DATA

看原网址的代码

To know what “Normalization” is and when it is needed to be done, visit: **Normalization**

```
min_max_scaler = skit.preprocessing.MinMaxScaler()
np_scaled = min_max_scaler.fit_transform(df)
df_normalized = pd.DataFrame(np_scaled)
df_normalized.columns = ['nPregnant','glucoseConc','diastolicBP','tric
df_normalized.head()
```

这里的作用是给数据重新定义列变量

	nPregnant	glucoseConc	diastolicBP	tricepThikness	serumInsulin	BMI	diabeticPedigree	age	classlabel
0	0.352941	0.743719	0.590164	0.353535	0.000000	0.500745	0.234415	0.483333	1.0
1	0.058824	0.427136	0.540984	0.292929	0.000000	0.396423	0.116567	0.166667	0.0
2	0.470588	0.919598	0.524590	0.000000	0.000000	0.347243	0.253629	0.183333	1.0
3	0.058824	0.447236	0.540984	0.232323	0.111111	0.418778	0.038002	0.000000	0.0
4	0.000000	0.688442	0.327869	0.353535	0.198582	0.642325	0.943638	0.200000	1.0

从SKLEARN导入KNN算法

IMPORT KNN ALGORITHM FROM SKLEARN

Sklearn是一个开放源代码的Python库，它实现了一系列机器学习，预处理，交叉验证和可视化算法。

Sklearn is an open source Python library that implements a range of machine learning, preprocessing, cross-validation and visualization algorithms. **SciKit**

```
1 from sklearn.neighbors import KNeighborsClassifier
```

CODE!

We will create a function that will train our model as well as cross-validate it and will give us the average score

Visit: **Cross Validation** to know what cross-validation means and for what it is used.

交叉验证的代码

```
1  def knnCrossValidate(data,label,model,folds):
2
3   test =[]
4   train =[]
5
6   indexes = data.index.values
7   np.random.shuffle(indexes)
```

```
    trainOne = indexes[ :testStart ]
    trainTwo = indexes[ testEnd:]
    trainFull = np.concatenate([trainOne,trainTwo])
    train.append(trainFull)

    knnscore = 0

    for trainsl,testsl in zip(train,test):
    model.fit(data.loc[trainsl],label[trainsl])
    modelScore = model.score(data.loc[testsl],label.loc[testsl])

    knnscore += modelScore
    return knnscore/folds
```


In above code, we are basically dividing the dataset in two parts (one for training, one for testing) for “folds” number of times. In each iteration, test and train data are altered but contains same population so that we can calculate accuracy for different test data to check if our model is behaving normally or over fitting/under-fitting.
可以计算不同测试数据的准确性，以检查我们的模型是正常运行还是过拟合/欠拟合

model.fit(data.loc[trainsl],label[trainsl])

上图中的model.fit()行覆盖了train数据集
model.fit() line above trains over the train dataset

modelScore = model.score(data.loc[testsl],label.loc[testsl])

score()通过比较预测的标签和实际的标签来确定模型的准确性。
model.score() score the accuracy of model by comparing the predicted and actual labels.

```
k=5

model = KNeighborsClassifier(k)
folds = 10

data = df[features]
classlabel=df[to_predict]
```

In above code, we are creating a model using KNNClassifier with K value of 5.

fold =10 are the number of time we are altering the test and train data set and calculating accuracy each time.

Next, we are separating the data from label (in this case class label)so that we can feed it to our function.

CALL THE FUNCTION

```
knnCrossValidate(data,classlabel,model,folds)
```

The out put of above code is:

```
KNN Accuracy at fold 10 is:  0.734313055366
```

So, the accuracy (for folds =10) is 0.73 (approx). Which means our KNN model is working fine, nor over-fitting neither under-fitting.

We can also use Sklearn for cross-validation.

Search ...

RECENT POSTS

KNN Classifier & Cross Validation in Python

Python-Twitter API & Basic Sentiment Analysis

Visualizing data – Python

Power BI – An amazing business analytics service

ALL POSTS

KNN Classifier & Cross Validation in Python

FOLLOW BLOG VIA EMAIL

Enter your email address to follow this blog and receive notifications of new posts by email.

Join 42 other followers

Enter your email address

FOLLOW

COMMENTS

Obaid Ur Rehman on **Python-Twitter API & Basic...**

archer920gmailcom on **Python-Twitter API & Basic...**

Obaid Ur Rehman on **Visualizing data – ...**

Rana Usman on **Visualizing data – ...**

SOCIAL

Navigation

Home

About

Contact

SHARE THIS:

Twitter Facebook

Like

One blogger likes this.

PUBLISHED BY OBAID UR REHMAN

View all posts by Obaid Ur Rehman

PREVIOUS POST

Python-Twitter API & Basic Sentiment Analysis

LEAVE A REPLY