 **TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**BÁO CÁO ĐỒ ÁN 3**

**Đề tài: Tìm hiểu giải thuật K-neraest neighbor**

**GVHD:** *Ths***.Từ Tuyết Hồng**

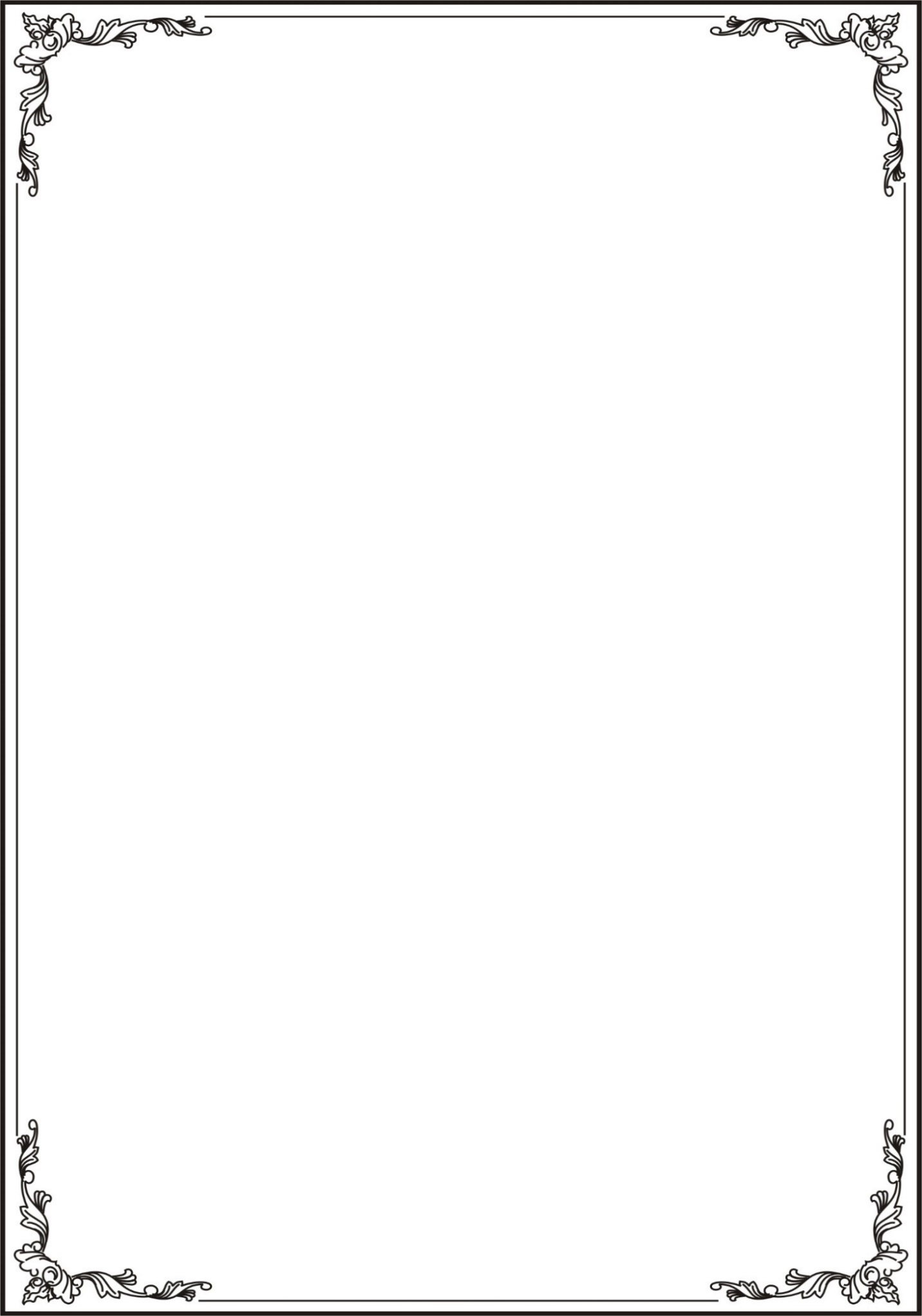
**SVTH: Phan Thanh Hoài**

**MSSV: 16110075**

**SVTH: Nguyễn Thành Đạt**

**MSSV: 17110120**

**TP. Hồ Chí Minh, tháng 12 năm 2020**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**BÁO CÁO ĐỒ ÁN 3**

**Đề tài: Tìm hiểu giải thuật K-neraest neighbor**

**GVHD:** *Ths***.Từ Tuyết Hồng**

**SVTH: Phan Thanh Hoài**

**MSSV: 16110075**

**SVTH: Nguyễn Thành Đạt**

**MSSV: 17110120**

**TP. Hồ Chí Minh, tháng 12 năm 2020**

**NHẬN XÉT CỦA GIẢNG VIÊN**

**………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………**

**………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………**

**Giảng viên**

**Lời cam đoan**

Nhóm của em xin cam đoan tất cả đều là quá trình học tập, nghiên cứu các giải thuật đã yêu cầu thực hiện cho các giải thuật. Nhóm không đạo văn và đạo code, nếu có nhóm xin chịu trách nhiệm hoàn toàn về các vấn đề xảy ra về sau.

Mục Lục

[**Chương 1: Lý thuyết** 1](#_Toc57889534)

[1.1. Giới thiệu thuật toán K-Nearest Neighbors. 1](#_Toc57889535)

[1.2. Đầu vào/ đầu ra của thuật toán. 1](#_Toc57889536)

[1.4. Các ứng dụng của thuật K-Nearest Neighbors 3](#_Toc57889537)

[**Chương 2: Code demo thuật toán K-Nearest Neighbors.** 4](#_Toc57889538)

[**Chương 3: Đánh giá model** 16](#_Toc57889539)

[**Chương 4: Kết luận** 17](#_Toc57889540)

[**Tài liệu tham khảo** 17](#_Toc57889541)

# **Chương 1: Lý thuyết**

## Giới thiệu thuật toán K-Nearest Neighbors.

K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất (mà hiệu quả trong một vài trường hợp) trong Machine Learning. Khi training, thuật toán này không học một điều gì từ dữ liệu training (đây cũng là lý do thuật toán này được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán Supervised learning là Classification và Regression. KNN còn được gọi là một thuật toán Instance based hay memory learned-based learning.

Mô hình được xây dựng chỉ bao gồm việc lưu trữ dữ liệu tập huấn *(training dataset)*. Để dự đoán được một điểm dữ liệu mới, thuật toán sẽ tìm ra những *láng giềng*trong dữ liệu tập huấn *(training dataset),* đó là láng giềng *(nearest neightbors)*.

## Đầu vào/ đầu ra của thuật toán.

Thuật toán có 2 đầu vào, một là tập các dữ liệu đã biết trước kiểu(loại) của từng dữ liệu(hay còn gọi là tập huấn luyện - training set), đầu vào thứ 2 là dữ liệu, chúng ta chưa biết kiểu(loại) dữ liệu đó. Đầu ra của thuật toán kNN là kiểu dữ liệu của đầu vào thứ 2

* 1. Sử dụng

Là phương pháp để phân lớp các đối tượng dựa vào khoảng cách gần nhất giữa đối tượng cần phân lớp (Query point) và tất cả các đối tượng trong Training Data

* 1. Mô tả thuật thoán K-Nearest Neighbors(KNN)

Bước 1:   Xác định giá trị tham số K (số láng giềng gần nhất)

Bước 2: Tính khoảng cách giữa đối tượng cần phân lớp (Query Point) với tất cả các đối tượng trong training data (thường sử dụng khoảng các Euclidean)

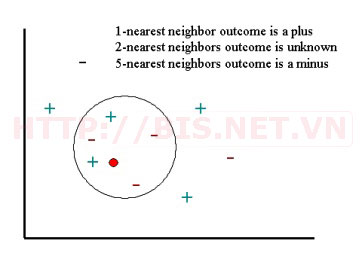
Bước 3: Sắp xếp khoảng cách theo thứ tự tăng dần và xác định K láng giềng gần nhất với Query Point

Bước 4: Lấy tất cả các lớp của K láng giềng gần nhất đã xác định

Bước 5:  Dựa vào phần lớn lớp của láng giềng gần nhất để xác định lớp cho Query Point

**Để hiểu K-NN được dùng để phân lớp thế nào ta xem minh họa dưới đây.**

Trong hình dưới đây, training Data được mô tả bởi dấu (+) và dấu (-), đối tượng cần được xác định lớp cho nó (Query point) là hình tròn đỏ. Nhiệm vụ của chúng ta là ước lượng (hay dự đoán) lớp của Query point dựa vào việc lựa chọn số láng giềng gần nhất với nó. Nói cách khác chúng ta muốn biết liệu Query Point sẽ được phân vào lớp (+) hay lớp (-)



**Ta thấy rằng:**

1-Nearest neighbor : Kết quả là + (Query Point được xếp vào lớp dấu +)

2-Nearest neighbors : không xác định lớp cho Query Point vì số láng giềng gần nhất với nó là 2 trong đó 1 là lớp + và 1 là lớp – (không có lớp nào có số đối tượng nhiều hơn lớp kia)

5-Nearest neighbors : Kết quả là - (Query Point được xếp vào lớp dấu – vì trong 5 láng giềng gần nhất với nó thì có 3 đối tượng thuộc lớp - nhiều hơn lớp + chỉ có 2 đối tượng).

## 1.4. Các ứng dụng của thuật K-Nearest Neighbors

Ví dụ 1:  giả sử bạn là một quản trị viên của một trang web, công việc hằng ngày của bạn là phân loại phim theo các thể loại(hài, hành động, tình cảm,...), vấn đề sẽ dễ dàng nếu số lượng phim của bạn là nhỏ, bạn có thể tự coi và dựa vào cảm nhận, quyết định xem phim đó thuộc thể loại nào :). Tuy nhiên vấn đề sẽ trở nên phức tạp, khi số lượng phim rất lớn, thì việc coi từng phim và cảm nhận là điều không tưởng, quá thiếu hiệu quả, lúc đó bạn sẽ bắt đầu nghĩ ngay tới việc sử dụng phần mềm giúp bạn tự động phân loại phim. Nói cách khác, việc phân loại phim chính là bài toán phân loại dữ liệu, và chúng ta có thể sử dụng thuật toán KNN trong trường hợp này.  
  
Ví dụ 2: Bạn được yêu cầu viết một phần mềm, nhận vào một tệp ảnh và đưa ra kết quả đó là tệp ảnh đó miêu tả con số nào từ 0 đến 9. Việc chúng ta làm ở đây phát biểu theo một cách khác đó là xác định xem tệp ảnh đầu vào nằm ở trong kiểu nào? kiểu số 0 hay kiểu số 1 hay kiểu số 2,... hay kiểu số 9.

# **Chương 2: Code demo thuật toán K-Nearest Neighbors.**

Bước 1: Thêm thư viện pandas, numpy, matplotlib, seaborn

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

* Thư viện pandas trong python là một thư viện mã nguồn mở, hỗ trợ đắc lực trong thao tác dữ liệu. Đây cũng là bộ công cụ phân tích và xử lý dữ liệu mạnh mẽ của ngôn ngữ lập trình python. Thư viện này được sử dụng rộng rãi trong cả nghiên cứu lẫn phát triển các ứng dụng về khoa học dữ liệu. Thư viện này sử dụng một cấu trúc dữ liệu riêng là Dataframe. Pandas cung cấp rất nhiều chức năng xử lý và làm việc trên cấu trúc dữ liệu này.
* Numpy là một thư viện lõi phục vụ cho khoa học máy tính của Python, hỗ trợ cho việc tính toán các mảng nhiều chiều, có kích thước lớn với các hàm đã được tối ưu áp dụng lên các mảng nhiều chiều đó. Numpy đặc biệt hữu ích khi thực hiện các hàm liên quan tới Đại Số Tuyến Tính.
* Để thực hiện các suy luận thống kê cần thiết, cần phải trực quan hóa dữ liệu của bạn và Matplotlib là một trong những giải pháp như vậy cho người dùng Python. Nó là một thư viện vẽ đồ thị rất mạnh mẽ hữu ích cho những người làm việc với Python và NumPy.
* Seaborn là một trong những thư viện Python được đánh giá cao nhất thế giới được xây dựng nhằm mục đích tạo ra các hình ảnh trực quan đẹp mắt. Nó có thể được coi là một phần mở rộng của một thư viện khác có tên là Matplotlib vì nó được xây dựng trên đó.

Bước 2: Đọc file Car.CSV bằng thư viện Pandas.ReadCSV và đặt tên từng cột dữ liệu tương ứng.

df = pd.read\_csv("car.csv",names = ['Price','Maint','Doors','Persons','Luggage','Safety','Acceptibility'])

#đọc dữ liệu từ file CSV bằng thư viện pandas và đặt tên tương ứng với các cột dữ liệu

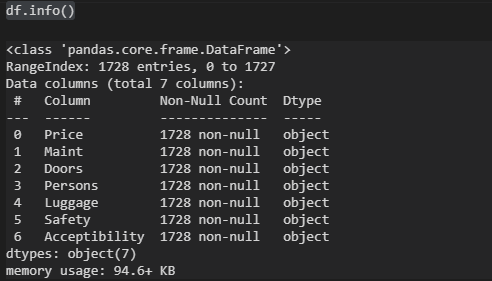
df.head()# hiện thỉ các giá trị đầu tiên của dataframe

df.info() #hiển thị thông tin của bảng dữ liệu

Sau đó hiển thị những dòng dữ liệu đầu để xem kết quả, ta được bảng dữ liệu:



Sau đó dùng df.info() để hiện được thông tin của dataframe:



Dữ liệu gồm có 1728 dòng và 7 cột được hiển thị như hình.

Bước 3: Gán X với các cột dữ liệu và Y với cột giá trị đánh giá.

df = pd.read\_csv("car.csv",names = ['Price','Maint','Doors','Persons','Luggage','Safety','Acceptibility'])

#đọc dữ liệu từ file CSV bằng thư viện pandas và đặt tên tương ứng với các cột dữ liệu

df.head()# hiện thỉ các giá trị đầu tiên của dataframe

df.info() #hiển thị thông tin của bảng dữ liệu

Sau đó dùng thư viện SNS để thực hiện vẽ các cột dữ liệu so sánh với khả năng chấp nhận tương ứng với hàm countplot của thư viện Seaborn.

sns.countplot(x='Price',hue = 'Acceptibility',data=df)

#cột Price với khả năng chấp nhận tương ứng

sns.countplot(x='Maint',hue='Acceptibility',data=df)

#cột Maint với khả năng chấp nhận tương ứng

sns.countplot(x='Doors',hue='Acceptibility',data=df)

#cột Doors với khả năng chấp nhận tương ứng

sns.countplot(x='Luggage',hue='Acceptibility',data=df)

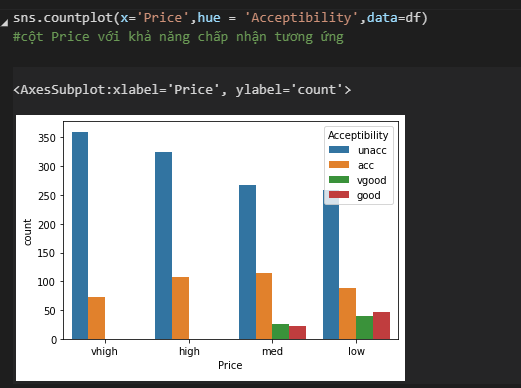
#cột Persons với khả năng chấp nhận tương ứng

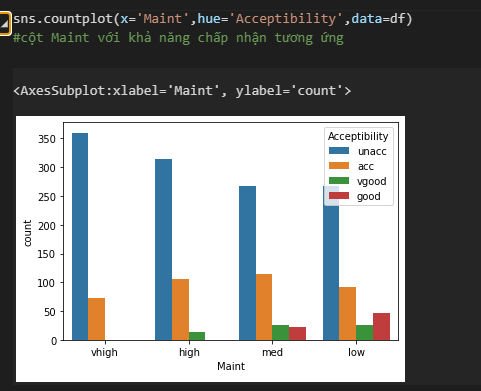
sns.countplot(x='Persons',hue='Acceptibility',data=df)

#cột Luggage với khả năng chấp nhận tương ứng

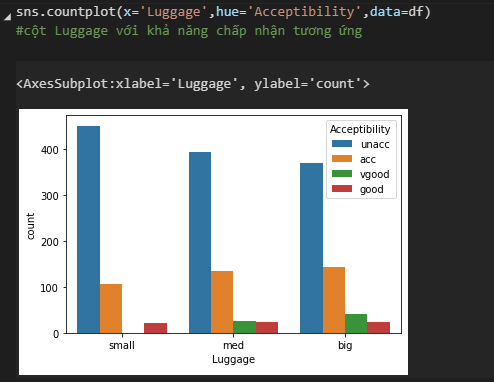
sns.countplot(x='Safety',hue='Acceptibility',data=df)

#cột Safety với khả năng chấp nhận tương ứng













Bước 4: Dùng hàm replace dataframe của Pandas để thực hiện chuyển đổi các dữ liệu kiểu string về dạng số tương ứng của các cột dữ liệu:

#chuyển dữ liệu cột Accceptibility về dạng số tương ứng Unacc = 0, Acc =1 , Good =2, Vgood = 3.

df.head(5)

df.Luggage.replace(('small','med','big'),(0,1,2),inplace=True)

#chuyển dữ liệu cột Luggage về dạng số tương ứng small = 0, med =1 , big =2.

df.head(5)

df.Safety.replace(('low','med','high'),(0,1,2),inplace=True)

#chuyển dữ liệu cột safety về dạng số tương ứng low = 0, med =1 , high =2.

df.head(5)

df.Maint.replace(('low','med','high','vhigh'),(0,1,2,3),inplace=True)

#chuyển dữ liệu cột  Maint về dạng số tương ứng low = 0, med =1 , high =2, vhigh = 3.

df.head(5)

df.Price.replace(('low','med','high','vhigh'),(0,1,2,3),inplace=True)

#chuyển dữ liệu cột Price về dạng số tương ứng low = 0, med =1 , high =2, vhigh = 3.

df.head(5)

df.Persons.replace(('more'),5,inplace=True)

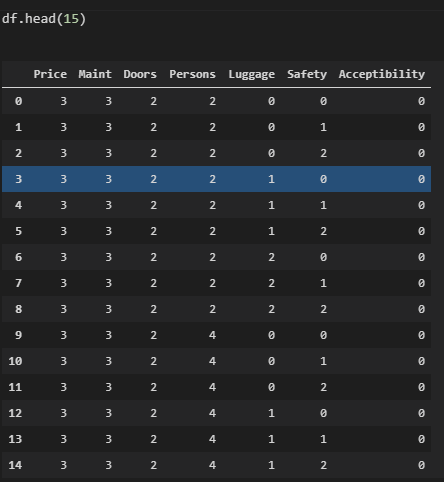
#chuyển dữ liệu cột Persons về dạng số tương ứng More = 5.

df.Doors.replace(('5more'),5,inplace=True)

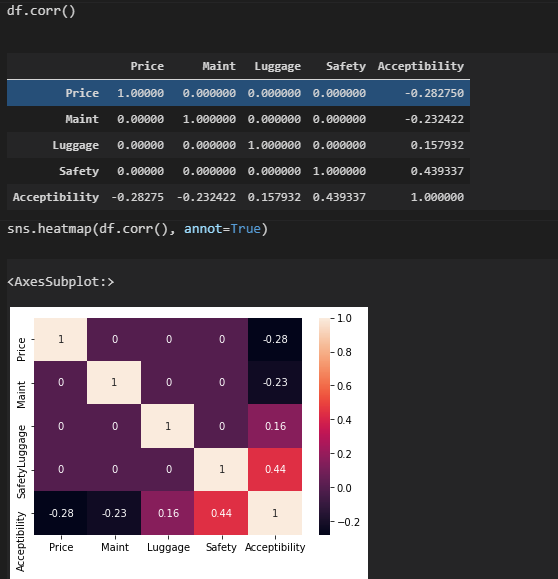
#chuyển dữ liệu cột Doors về dạng số tương ứng 5More = 5.

df.head(15)# hiển thị 15 dòng dữ liệu đầu của df để xem

Dữ liệu của df đã được chuyển đổi sang dạng số tương ứng như 15 dòng đầu tiên:



Tiếp theo ta sử dụng hàm corr của pandas.dataframe df.corr() để thể hiện mối tương quan giữa các cột dữ liệu với nhau, sau đó dùng hàm heatmap của thư viện seaborn để vẽ lại một cách dễ nhìn hơn.



Bước 5: Chọn dữ liệu và thực hiện train

x = df.iloc[:,:6]

y = df.iloc[:,6]

print("Shape of x:-",x.shape)

print("Shape of y:-",y.shape)

#chọn dữ liệu

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

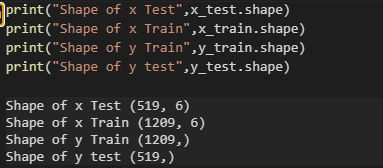
#import thư viện train\_test\_split để thực hiện chia dữ liệu thành tập hợp con train và test ngẫu nhiên

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.30, random\_state = 1)

#khai báo dữ liệu train và dữ liệu test và kích cỡ test\_size = 0.30 (30% của df) và xét giá trị random =1

* Sklearn là thư viện về machine learning phổ biến nhất của Python. Nó viết sẵn các thuật toán lằng nhằng phức tạp, bạn chỉ việc nhét dữ liệu vào, chờ nó tính toán rồi lấy kết quả.
* Style : train\_test\_split Chia mảng hoặc ma trận thành tập hợp con thử nghiệm và huấn luyện ngẫu nhiên

Dữ liệu sẽ được chia thành train và test tương ứng với 0.3 như sau:



Thực hiện import style :’ StandardScaler’ từ thư viện Sklearn.preprocessing và thực hiện biến đổi dữ liệu sao cho phân phối của dữ liệu sẽ có giá trị trung bình là 0 và độ lệch chuẩn là 1. Với phân phối dữ liệu, mỗi giá trị trong tập dữ liệu sẽ bị trừ giá trị trung bình mẫu, và sau đó chia cho độ lệch chuẩn của toàn bộ dữ liệu.

KNeighbours Classification

Tạo mô hình:

**from sklearn.preprocessing import StandardScaler**

**sc = StandardScaler()**

**x\_train = sc.fit\_transform(x\_train)**

**x\_test = sc.transform(x\_test)**

from sklearn.neighbors import KNeighborsClassifier# Bộ phân loại thực hiện  bỏ phiếu KNN gần nhất

from sklearn.metrics import confusion\_matrix#thư viện ma trận nhầm lẫn

knn = KNeighborsClassifier(n\_neighbors=3)# số lượng giá trị xung quanh để sử dụng cho việc xác định khả năng xác nhận

Cung cấp dữ liệu đào tạo vào mô hình:

#cung cấp dữ liệu đào tạo vào mô hình

knn.fit(x\_train, y\_train)# điều chỉnh mô hình bằng cách  sử dụng X làm dữ liệu  đào tạo và y làm giá trị mục tiêu

Dự đoán giá trị cho X\_test

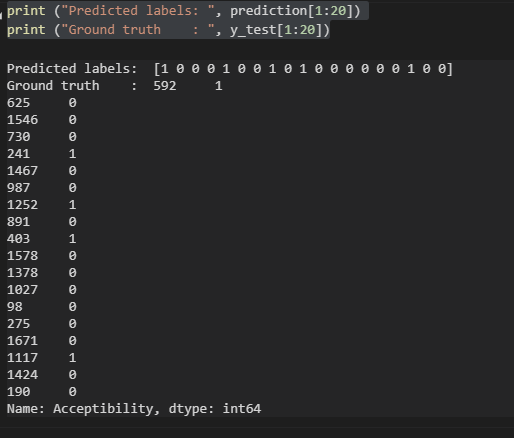
prediction = knn.predict(x\_test)# dự đoán giá trị mục tiêu cho dữ liệu test

In kết quả 20 dòng đầu của label dự đoán và label thật của test data

print ("Predicted labels: ", prediction[1:20])

print ("Ground truth    : ", y\_test[1:20])

Ta được kết quả dự đoán gần giống với kết quả thật của test data



Bước 6: Phương pháp đánh giá

Để đánh giá độ chính xác của thuật toán KNN classifier này, chúng ta xem xem có bao nhiêu điểm trong test data được dự đoán đúng. Lấy số lượng này chia cho tổng số lượng trong tập test data sẽ ra độ chính xác. training và testing bằng hàm KNN.Score của thư viện KNN:

print("Training Accuracy:",knn.score(x\_train,y\_train)) # Trả lại độ chính xác trung bình trên các nhãn và dữ liệu train.

OutPut:

Training Accuracy: 0.9917287014061208

print("Testing Accuracy:",knn.score(x\_test,y\_test)) # Trả lại độ chính xác trung bình trên các nhãn và dữ liệu train

Output:

Testing Accuracy: 0.953757225433526

print ("Predicted labels: ", prediction[1:20])

print ("Ground truth    : ", y\_test[1:20])

Kết quả đưa ra cho thấy độ chính xác của khi cho K = 3 ở tập dữ liệu Trainning là 0.99 và độ chính xác của dữ liệu khi K = 3 ở tập testing là 0.95. Cho thấy tập dữ liệu này ổn định và chính xác cao.

# **Chương 3: Đánh giá model**

Ưu điểm của thuật toán KNN

* Độ phức tạp tính toán của quá trình training là bằng 0.
* Việc dự đoán kết quả của dữ liệu rất đơn giản.
* Dễ sử dụng và cài đặt.

Nhược điểm của thuật toán KNN.

* KNN nhiễu dễ đưa ra kết quả không chính xác khi K nhỏ.
* KNN là một thuật toán mà mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới từng điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian.

# **Chương 4: Kết luận**

Trong quá trình tìm hiểu, gặp rất nhiều khó khăn với những kiến thức mới. Nhờ sự giúp đỡ từ thầy cô và các bạn nên nhóm cũng đã hoàn thành được đề tài. Về cơ bản, nhóm đã tìm hiểu được thuật toán Meanshift Clustering, ưu điểm và nhược điểm của thuật toán, lý thuyết nền tảng của thuật toán. Bên cạnh đó, đã viết được cơ bản code demo thuật toán trong quá trình tìm hiểu. Tuy nhiên vẫn còn một số khó khăn trong việc tìm hiểu sâu hơn về code demo trong thời gian ngắn. Mong thầy cô có nhận xét và góp ý hơn về đề tài giúp nhóm rút kinh nghiệm và tìm hiểu thêm về những kiến thức còn hạn chế mà nhóm còn chưa khắc phục.

# **Tài liệu tham khảo**

<https://archive.ics.uci.edu/ml/datasets/Car+Evaluation>

<https://machinelearningcoban.com/2017/01/08/knn/>

<https://codelearn.io/sharing/thuat-toan-k-nearest-neighbors-knn>

<https://blog.vietnamlab.vn/thuc-thi-thuat-toan-k-lang-gieng-gan-nhat-k-nearest-neighbour/>