BÀI THỰC HÀNH SỐ 8-9

Phân lớp

Giới thiệu một số thư viện và lệnh trong phân lớp:

Thư viện

Xây dựng mô hình

from sklearn import model_selection

thư viện báo cáo pkeets quả phân lớp

from sklearn.metrics import classification_report

from sklearn.metrics import confusion_matrix

#tính độ chính xác phân lớp

from sklearn.metrics import accuracy_score

#xây dựng cây quyết định

from sklearn.tree import DecisionTreeClassifier

#thư viện tạo ra cây quyết định

from sklearn import tree

#thư viện chia dữ liệu ra thành tập train và test

from sklearn.model_selection import train_test_split

#thư viện phân lớp bằng phương pháp naïve bayes.

from sklearn.naive_bayes import GaussianNB

các lệnh phân lớp:

#CHIA DỮ LIỆU 80% TRAIN, 20% TEST

X_train, X_test, y_train, y_test= train_test_split(X, y, test_size=0.2, random_state=3)

#Phân lớp theo cây quyết định

decision_tree = tree.DecisionTreeClassifier(criterion='gini')

#xây dựng cây quyết định dựa trên tập dữ liệu train

decision_tree.fit(X_train, y_train)

#Phân lớp naiv bayer

clfNB = GaussianNB()

#Train the model using the training sets y_pred=clf.predict(X_test)

clfNB.fit(X_train,y_train)

y_pred=clfNB.predict(X_test)

Bài 1. Thực hành theo 2 bài mẫu sau về phânl ớp theo cây quyết định

Bài mẫu NaiveBayesClassifier052020_house.ipynb

Và bài mẫu sau:

Phân lớp dữ liệu theo cây quyết định: **bai thu hanh mau cay quyet dinh _house.ipynb**

In [61]:
```
import pandas as pd
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
melbourne_data = pd.read_csv('D:\\hoc python\\house.csv')
melbourne_data
```

Out[61]:

	Unnamed: 0	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	...	Bathroom	Car	Landsize	BuildingArea
0	1	Abbotsford	85 Turner St	2	h	1480000	S	Biggin	03/12/16	2.5	...	1	1.0	202	NaN
1	2	Abbotsford	25 Bloomburg St	2	h	1035000	S	Biggin	04/02/16	2.5	...	1	0.0	156	79.0
2	4	Abbotsford	5 Charles St	3	h	1465000	SP	Biggin	04/03/17	2.5	...	2	0.0	134	150.0
3	5	Abbotsford	40 Federation La	3	h	850000	PI	Biggin	04/03/17	2.5	...	2	1.0	94	NaN
4	6	Abbotsford	55a Park St	4	h	1600000	VB	Nelson	04/06/16	2.5	...	1	2.0	120	142.0

In [62]:
```
cut_labels = ['Low','Medium','High']
cut_bins = [0, 900000, 1200000, 10000000] #0: min, 100: max
melbourne_data['Price_lable'] = pd.cut(melbourne_data['Price'], bins=cut_bins, labels=cut_labels)
```

In [63]:
```
melbourne_data.columns
melbourne_data.head()
y = melbourne_data.Price_lable #NHÃN PHÂN LỚP
#CHỌN CÁC THUỘC TÍNH PHÂN LỚP
melbourne_features = ['Rooms', 'Bathroom', 'Landsize', 'Lattitude', 'Longtitude']
X = melbourne_data[melbourne_features]
X.describe()
X.head()
y
#melbourne_data.Price_lable
```

Out[63]:
```
0          High
1        Medium
2          High
3           Low
4          High
5        Medium
6          High
7          High
8        Medium
9          High
10       Medium
```

In [68]:
```
melbourne_data.columns
melbourne_data.head()
y = melbourne_data.Price_lable #NHÃN PHÂN LỚP
#CHỌN CÁC THUỘC TÍNH PHÂN LỚP
melbourne_features = ['Rooms', 'Bathroom', 'Landsize', 'Lattitude', 'Longtitude']
X = melbourne_data[melbourne_features]
X.describe()
X.head()
```

Out[68]:

	Rooms	Bathroom	Landsize	Lattitude	Longtitude
0	2	1	202	-37.7996	144.9984
1	2	1	156	-37.8079	144.9934
2	3	2	134	-37.8093	144.9944
3	3	2	94	-37.7969	144.9969
4	4	1	120	-37.8072	144.9941

In [64]:
```
#CHIA DỮ LIỆU 80% TRAIN, 20% TEST
X_train, X_test, y_train, y_test= train_test_split(X, y, test_size=0.2, random_state=3)
```

In [65]:
```
from sklearn import tree
#Create tree object
decision_tree = tree.DecisionTreeClassifier(criterion='gini')
#Train Decision Tree based on training set
decision_tree.fit(X_train, y_train)
```

Out[65]:
```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

In [66]:
```
predictions = decision_tree.predict(X_test)
print(accuracy_score(y_test, predictions))
print(confusion_matrix(y_test, predictions))
print(classification_report(y_test, predictions))
```

```
0.7226746678096871
[[664  61 120]
 [ 39 824 142]
 [141 144 198]]
              precision    recall  f1-score   support

        High       0.79      0.79      0.79       845
         Low       0.80      0.82      0.81      1005
      Medium       0.43      0.41      0.42       483

    accuracy                           0.72      2333
   macro avg       0.67      0.67      0.67      2333
weighted avg       0.72      0.72      0.72      2333
```

In [67]:
```
target=melbourne_data['Price_lable'].unique()
```

In [56]:
```
# Load libraries
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn import datasets
from IPython.display import Image
import pydotplus
# Create DOT data
dot_data = tree.export_graphviz(decision_tree, out_file=None,
                                feature_names=melbourne_features,
                                class_names=target)
```

In [57]:
```
# Draw graph
graph = pydotplus.graph_from_dot_data(dot_data)
# Show graph
Image(graph.create_png())
```

```
dot: graph is too large for cairo-renderer bitmaps. Scaling by 0.295174 to fit
```

Out[57]:

In [59]:
```
graph.write_pdf("D:/cayquyetdinh.pdf")
# Create PNG
graph.write_png("D:/marketing.png")
```

Bài 2: Phân lớp cho dữ liệu sau:
Trên dữ liệu **train.csv** thực hiện phân lớp như sau:
Thuộc tính phân lớp: y = home_data.SalePrice
Phân lớp trên các thuộc tính:

* LotArea
* YearBuilt
* 1stFlrSF
* 2ndFlrSF
* FullBath
* BedroomAbvGr
* TotRmsAbvGrd

Thực hiện phân lớp và cho hiển thị mô hình.

Bài 3. Làm theo bài mẫu NaiveBayesClassifier phân lớp theo Naïve bayer.

Thực hiện trên tệp dữ liệu như bài tập 2.

Bài 4*. Mở rộng tìm hiểu làm theo thuật toán random forest với các tệp dữ liệu trên