KNN_credit_risk

August 4, 2023

1 Bài toán Dự đoán tín dụng cá nhân:

- Đưa vào một số thông tin của người dùng, dự đoán xem người đó có tín dụng tốt hay là xấu
- Thực hành sử dụng mô hình K-Nearest Neighbors (KNN) để giải quyết bài toán này # Nội dung thực hành:
- Trực quan hóa dữ liệu (Data visualization)
- Tiền xử lý dữ liệu
- Huấn luyện và đánh giá mô hình KNN
- Lựa chọn siêu tham số và các thuộc tính đầu vào cho mô hình

2 *Import libraries*

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
seed = 42
```

3 *Load Bank Personal Loan Modelling dataset & explore*

- Dữ liệu được lấy từ Statlog (German Credit Data) Data Set https://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29
- Thông tin của mỗi người bao gồm 24 trường thuộc tính
- Nhãn 1: là người đó có tín dụng tốt
- Nhãn 0: là người đó có tín dụng xấu
- Ma trận chi phí (cost matrix): Chi phí khi đánh giá sai tín dụng xấu thành tốt (5) cao hơn khi đánh giá sai tín dụng tốt thành xấu (1)

[ ]:
```
cost_matrix = np.array([[0, 5], [1, 0]])
print('Nếu tín dụng xấu được phân là xấu, chi phí:', cost_matrix[0][0])
print('Nếu tín dụng xấu được phân là tốt, chi phí:', cost_matrix[0][1])
print('Nếu tín dụng tốt được phân là xấu, chi phí:', cost_matrix[1][0])
print('Nếu tín dụng tốt được phân là tốt, chi phí:', cost_matrix[1][1])
```

```
Nếu tín dụng xấu được phân là xấu, chi phí: 0
Nếu tín dụng xấu được phân là tốt, chi phí: 5
```

Nếu tín dụng tốt được phân là xấu, chi phí: 1
Nếu tín dụng tốt được phân là tốt, chi phí: 0

[ ]:
```
data_array = np.genfromtxt('german.data-numeric')
data_frame = df = pd.DataFrame(data_array, columns=['A'+str(i) for i in␣
 ↪range(1, 25)]+['label'])
# ban đầu nhãn là 1: good, 2: bad
# để đơn giản ta chuyển về 1: good, 0: bad
data_frame['label'] = data_frame['label'].replace(2, 0)
data_frame
```

[ ]:
```
data_frame.info()
```

[ ]:
```
data_frame.describe()
```

4 *Data visualization*

[ ]:
```
sns.pairplot(data_frame,
             hue='label',
             vars=['A'+str(i) for i in range(1, 10)])
```

[ ]:
```
sns.countplot(data_frame['label'])
```

[ ]:
```
plt.figure(figsize=(16,9))
sns.heatmap(data_frame)
```

[ ]:
```
plt.figure(figsize=(20,20))
sns.heatmap(data_frame.corr(), annot=True, cmap='coolwarm', linewidths=2)
```

5 *Data preprocessing*

Normalization

[ ]:
```
X = data_frame.drop(['label'], axis=1)
# X = (X-X.mean())/X.var()
X = (X-X.min())/(X.max()-X.min())
X
```

[ ]:
```
X.describe()
```

[ ]:
```
plt.figure(figsize=(16,9))
sns.heatmap(X)
```

[ ]:
```
y = data_frame['label']
y
```

5.1 Chia dữ liệu làm 2 phần training và testing

- Training chiếm 80 % dữ liệu
- Testing chiếm 20 % dữ liệu

[ ]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=seed)
print("Dữ liệu training = ", X_train.shape, y_train.shape)
print("Dữ liệu testing = ", X_test.shape, y_test.shape)
```

[ ]:
```
from sklearn.metrics import precision_score, recall_score, accuracy_score
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt

%matplotlib inline
```

6 *K - Nearest Neightbor Classifier*

6.1 Bài toán phân loại sử dụng KNN

Mục tiêu:

- Xây dựng được mô hình KNN sử dụng thư viện sklearn.
- Ứng dụng, hiểu cách áp dụng mô hình KNN vào giải quyết bài toán thực tế (vd: phân loại)
- Sử dụng độ đo Accuracy, Cost để làm độ đo đánh giá chất lượng mô hình.

Vấn đề: - Có một tập các dữ liệu không có nhãn, làm sao để biết dữ liệu này là thuộc về nhãn nào. - => Xây dựng mô hình học máy có thể phân loại.

Dữ liệu: - Dữ liệu Bank Personal Loan Modelling - Xem thêm: https://www.kaggle.com/teertha/personal-loan-modeling

Bài toán: - Input: 1 mẫu dữ liệu $X = [x_1, x_2, ...x_n]$ - Output: nhãn y là 0 hoặc 1

[ ]:
```
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
%matplotlib inline
```

6.2 Mô hình KNN

Sử dụng thư viện sklearn để xây dựng mô hình - `KNeighborsClassifier(n_neighbors = 10, metric = 'minkowski', p = 2)` - Số láng giềng: n_neighbors = 5 - Độ đo khoảng cách: Euclide p = 2

[ ]:
```
knn_classifier = KNeighborsClassifier(n_neighbors = 10, metric = 'minkowski', p␣
↪= 2, weights = 'distance')
knn_classifier.fit(X_train, y_train)
```

[ ]:
```
KNeighborsClassifier(n_neighbors=10, weights='distance')
```

6.3 Testing KNN model

6.4 Đánh giá theo các độ đo

[ ]:
```
def cost(y_true, y_pred):
    true_pos = ((y_true==y_pred)&(y_true==1.0))*0.0
    true_ne = ((y_true==y_pred)&(y_true==0.0))*0.0
    false_ne = ((y_true!=y_pred)&(y_true==1.0))*1.0
    false_pos = ((y_true!=y_pred)&(y_true==0.0))*5.0
    return sum(true_pos + true_ne + false_pos + false_ne)/len(y_true)
```

[ ]:
```
from sklearn.metrics import precision_score, recall_score, accuracy_score

print("Testing...\n")
y_pred_knn = knn_classifier.predict(X_test)
print('Accuracy: ', accuracy_score(y_test, y_pred_knn))
print('Cost: ', cost(y_test, y_pred_knn))
# print('Precision: ', precision_score(y_test, y_pred_knn))
# print('Recall: ', recall_score(y_test, y_pred_knn))
```

6.5 Lựa chọn mô hình

6.5.1 Lựa chọn số lượng láng giềng

- Thay đổi số lượng láng riềng tìm giá trị cho kết quả phân loại tốt nhất

6.5.2 Lựa chọn thuộc tính

- Các thuộc tính: A1-24
- Thử loại bỏ từng thuộc tính ra khỏi dữ liệu xem chúng ảnh hưởng như thế nào tới kết quả phân loại.
- Các thuộc tính nào nên được sử dụng để cho kết quả phân loại tốt nhất ?

6.5.3 Lựa chọn hàm tính khoảng cách

- Hàm tính khoảng cách: minkowski, manhattan, euclidean, chebyshev
- Hàm tính khoảng cách nào là tốt nhất cho bài toán này ?