School of Data and Computer Science, Sun Yat-sen University

E11 Naive Bayes (C++/Python)

18340149 孙新梦

2020 年 11 月 26 日

目录

1 Datasets 2

2 Naive Bayes 3

3 Task 5

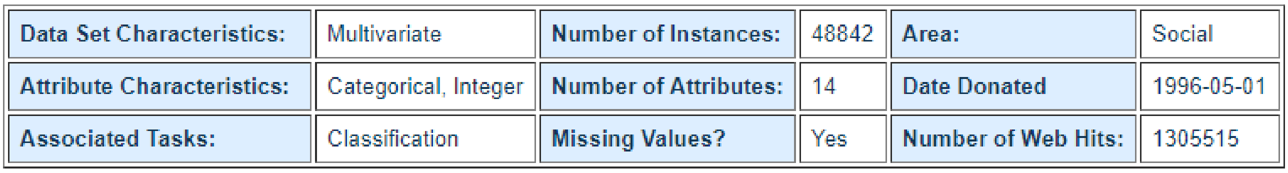
4 Codes and Results 5

5 感想和分析 9

1 Datasets

The UCI dataset (`http://archive.ics.uci.edu/ml/index.php`) is the most widely used dataset for machine learning. If you are interested in other datasets in other areas, you can refer to `https://www.zhihu.com/question/63383992/answer/222718972`.

Today's experiment is conducted with the **Adult Data Set** which can be found in `http://archive.ics.uci.edu/ml/datasets/Adult`.



Data Set Characteristics:	Multivariate	**Number of Instances:**	48842	**Area:**	Social
Attribute Characteristics:	Categorical, Integer	**Number of Attributes:**	14	**Date Donated**	1996-05-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	1305515

You can also find 3 related files in the current folder, `adult.name` is the description of **Adult Data Set**, `adult.data` is the training set, and `adult.test` is the testing set. There are 14 attributes in this dataset:

```
>50K, <=50K.

1. age: continuous.
2. workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov,
State-gov, Without-pay, Never-worked.
3. fnlwgt: continuous.
4. education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm,
Assoc-voc, 9th, 7th-8th, 12th, Masters, 5. 1st-4th, 10th, Doctorate, 5th-6th,
Preschool.
5. education-num: continuous.
6. marital-status: Married-civ-spouse, Divorced, Never-married, Separated,
Widowed, Married-spouse-absent, Married-AF-spouse.
7. occupation: Tech-support, Craft-repair, Other-service, Sales,
Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct,
Adm-clerical ,Farming-fishing ,Transport-moving ,Priv-house-serv ,
    Protective-serv ,
Armed-Forces.
8. relationship: Wife ,Own-child ,Husband ,Not-in-family ,Other-
    relative ,Unmarried.
9. race: White, Asian-Pac-Islander , Amer-Indian-Eskimo, Other,
    Black.
10. sex: Female, Male.
11. capital-gain: continuous.
12. capital-loss: continuous.
13. hours-per-week: continuous.
14. native-country: United-States , Cambodia ,England ,Puerto-Rico ,
    Canada ,Germany ,
Outlying-US(Guam-USVI-etc) ,India ,Japan ,Greece , South ,China ,Cuba ,
    Iran ,Honduras ,
Philippines , Italy , Poland , Jamaica , Vietnam , Mexico , Portugal ,
    Ireland , France ,
Dominican-Republic ,Laos ,Ecuador ,Taiwan , Haiti , Columbia ,Hungary ,
    Guatemala ,
Nicaragua ,Scotland ,Thailand ,Yugoslavia ,El-Salvador , Trinadad&
    Tobago ,Peru ,Hong ,
Holand-Netherlands.
```

Prediction task is to determine whether a person makes over 50K a year.

2 Naive Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that **the value of a particular feature is independent of the value of any other feature**, given the class variable.

For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features.

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. Bayes' theorem states the following relationship, given class variable y and dependent feature vector x_1 through x_n:

$$P(y \mid x_1, ..., x_n) = \frac{P(y)P(x_1, ..., x_n \mid y)}{P(x_1, ..., x_n)}$$

Using the naive conditional independence assumption that

$$P(x_i \mid y, x_1, ..., x_{i-1}, x_{x+1}, ..., x_n) = P(x_i \mid y)$$

, for all i, this relationship is simplified to

$$P(y \mid x_1, ..., x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, ..., x_n)}$$

Since $P(x_1, ..., x_n)$ is constant given the input, we can use the following classification rule:

$$P(y \mid x_1, ..., x_n) \propto P(y)\prod_{i=1}^{n} P(x_i \mid y)$$

$$\hat{y} = \arg\max_{y} P(y)\prod_{i=1}^{n} P(x_i \mid y),$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i \mid y)$, the former is then the relative frequency of class y in the training set.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i \mid y)$.

- When attribute values are discrete, $P(x_i \mid y)$ can be easily computed according to the training set.

- When attribute values are continuous, an assumption is made that the values associated with each class are distributed according to Gaussian i.e., Normal Distribution. For example, suppose the training data contains a continuous attribute x. We first segment the data by the class, and then compute the mean and variance of x in each class. Let μ_k be the mean of the values in x associated with class y_k, and let σ_k^2 be the variance of the values in x associated with class y_k. Suppose we have collected some observation value x_i. Then, the probability distribution of

x_i given a class y_k, $P(x_i \mid y_k)$ can be computed by plugging x_i into the equation for a Normal distribution parameterized by μ_k and σ_k^2. That is,

$$P(x = x_i \mid y = y_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(x_i-\mu_k)^2}{2\sigma_k^2}}$$

3 Task

- Given the training dataset `adult.data` and the testing dataset `adult.test`, please accomplish the prediction task to determine whether a person makes over 50K a year in `adult.test` by using Naive Bayes algorithm (C++ or Python), and compute the accuracy.
- Note: keep an eye on the discrete and continuous attributes.
- Please finish the experimental report named `E11_YourNumber.pdf`, and send it to `ai_2020@foxmail.com`

4 Codes and Results

Code

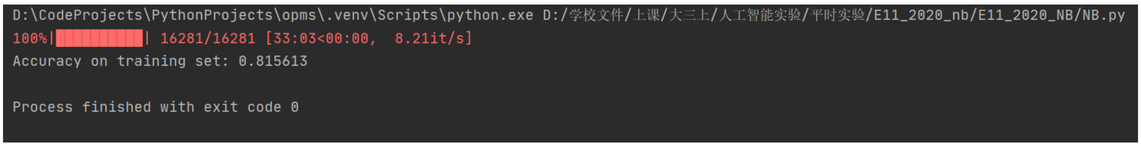
```
”””
E11 NB.py姓名：孙新梦学号：

18340149
TASK使用:算法实现决策树，判断一个人是否每年能拿到超过万ID35输入：两个文件：

    adult.：训练集data
    adult.：测试集test输出：个测试样例的精确度，最后的平均精确度
    10运行：

    python NB.py 普通版 #
”””
import numpy as np
import pandas as pd
from tqdm import tqdm

# 连续变量使用高斯分布
```

```
def continuous_cond_prob(y, x, xi, Data):
    # Use pandas to locate the target attribute:
    # .loc[boolean]: filter by boolean to get the slices.
    x = Data.loc[Data['Salaries'] == y][x]
    # Ensure every element are of int64 type,
    # for calculation below.
    x = np.array(x, dtype=np.int64)
    # Calculate mu, sigma
    mu = np.average(x)
    sigma = np.var(x)
    # Calculate the Gassusian distribution.
    return 1.0 / np.sqrt(2 * np.pi * sigma) * np.exp(-(int(xi) - mu) ** 2 / (2 * sigma))

# 离散变量计算概率
def discrete_cond_prob(y, x, Data):
    # Use pandas to locate the target attribute:
    # .loc[boolean]: filter by boolean to get the slices.
    x = list(Data.loc[Data['Salaries'] == y][x])
    # Use set() to found how many distinguished values
    # the target features have.
    # And also count out.
    x_set = list((set(x)))
    x_count = [x.count(val) / len(Data) for val in x_set]
    # A dictionary storing all discrete conditinal pr.
    return dict(zip(x_set, x_count))

# 朴素贝叶斯模型
def NaiveBayes(Data, sample, labels):
    # Initialize Probabilities of each label
    Prob = [len(Data.loc[Data['Salaries'] == label]) / len(Data) for label in labels]
    # Calculating Conditional probabilities and multiply to Prob..
    for idx, col in enumerate(Data.columns[:-1]):
        if col in continuous_cols:
            for idx_p, P in enumerate(Prob):
                Prob[idx_p] *= continuous_cond_prob(labels[idx_p], col, sample[idx], Data)
        else:
            for idx_p, P in enumerate(Prob):
                Prob[idx_p] *= discrete_cond_prob(labels[idx_p], col, Data).get(sample[idx], 0)
    max_prob = max(Prob)
    return labels[Prob.index(max_prob)]


if __name__ == '__main__':
    # 读文件
    train_data_path = 'dataset/adult.data'
    test_data_path = 'dataset/adult.test'

    header = ['age', 'workclass', 'fnlwgt', 'education', 'education-num',
              'marital-status', 'occupation', 'relationship', 'race', 'sex',
              'capital-gain', 'capital-loss', 'hours-per-week', 'native-country', 'Salaries']

    train_data = pd.read_csv(train_data_path, names=header)
    test_data = pd.read_csv(test_data_path, names=header)
    test_data.drop(0, inplace=True)
    test_data.reset_index(drop=True, inplace=True)

    # 处理空数据
    train_data.replace(' ?', np.nan, inplace=True)
    train_data.fillna(train_data.mode().iloc[0], inplace=True)
```

```python

    test_data.replace(' ?', np.nan, inplace=True)
    test_data.fillna(test_data.mode().iloc[0], inplace=True)

    continuous_cols = ['age', 'fnlwgt', 'education-num', 'capital-gain', 'capital-loss', 'hours-per-week']


    miss_rate = 0
    for i in tqdm(test_data.index):
        label = test_data.iloc[i][-1]
        predition = NaiveBayes(test_data, test_data.iloc[i][:-1], [' >50K.', ' <=50K.'])
        if label != predition:
            miss_rate += 1

    print("Accuracy on training set: {:4f}".format(1 - miss_rate / len(test_data)))
```

Result

下面是运行后的截图，可以看到在16281个测试样例上，准确率达到了81.6



```
D:\CodeProjects\PythonProjects\opms\.venv\Scripts\python.exe D:/学校文件/上课/大三上/人工智能实验/平时实验/E11_2020_nb/E11_2020_NB/NB.py
100%|██████████| 16281/16281 [33:03<00:00,  8.21it/s]
Accuracy on training set: 0.815613

Process finished with exit code 0
```

5 感想和分析

本次实验由于和上回较为相近，所以对模型变得熟悉了很多，再加上学着使用了Jupyter Notebook，学会了很多东西。

这回实验有了一些心得和体会：

1.首先是朴素贝叶斯模型的确不如上回的决策树精确，原因在我们理论课的时候有过了解：因为我们实际的问题模型是更加符合决策树的形式的，也就是一个人工资的的确确是由决策树那样一步步判断来得出的。而朴素贝叶斯的概率模型的一个大前提假设是，属性之间相互独立，这是不合常理的，因此在精确度方面略逊一筹。

2.其次是运行时间真的很长，原因大概在于计算条件概率用了很多的数学计算，相当耗时。