机器学习 第七章 作业

7.1

极大似然法要先假定一种概率分布形式。
色泽：
对于好瓜，假设

$$
\begin{aligned}
&\mathrm{P}(\text{色泽}=\text{青绿}|\text{好瓜}) = \sigma_1 \\
&\mathrm{P}(\text{色泽}=\text{乌黑}|\text{好瓜}) = \sigma_2 \\
&\mathrm{P}(\text{色泽}=\text{浅白}|\text{好瓜}) = \sigma_3 = 1-\sigma_1-\sigma_2 \\
&L(\sigma) = \bigcap_i P(\text{色泽} = x_i|\text{好瓜}) = \sigma_1^3\sigma_2^4(1-\sigma_1-\sigma_2) \\
&L'(\sigma_1) = \sigma_2^4\sigma_1^2(3-4\sigma_1-3\sigma_2) \\
&L'(\sigma_2) = \sigma_1^3\sigma_2^3(4-4\sigma_1-5\sigma_2) \\
&\text{令}\mathrm{L}'(\sigma_1)=0,\ L'(\sigma_2)=0\text{得}\sigma_1=\frac{3}{8},\ \sigma_1=\frac{1}{2},\ \sigma_3=\frac{1}{8}
\end{aligned}
$$

可以看出$\sigma_1, \sigma_2, \sigma_3$分别对应他们在样本中出现的频率。

对于坏瓜以及两种属性计算方式相同，得出类似的结果

7.3

```
import numpy as np
import pandas as pd
from sklearn.utils.multiclass import type_of_target
from collections import namedtuple


def train_nb(X, y):
    m, n = X.shape
    p1 = (len(y[y == '是']) + 1) / (m + 2)  # 拉普拉斯平滑

    p1_list = []  # 用于保存正例下各属性的条件概率
    p0_list = []

    X1 = X[y == '是']
    X0 = X[y == '否']

    m1, _ = X1.shape
    m0, _ = X0.shape

    for i in range(n):
        xi = X.iloc[:, i]
        p_xi = namedtuple(X.columns[i], ['is_continuous', 'conditional_pro'])  #
用于储存每个变量的情况

        is_continuous = type_of_target(xi) == 'continuous'
        xi1 = X1.iloc[:, i]
        xi0 = X0.iloc[:, i]
        if is_continuous:  # 连续值时，conditional_pro 储存的就是 [mean, var] 即均值和
方差
            xi1_mean = np.mean(xi1)
```

```
            xi1_var = np.var(xi1)
            xi0_mean = np.mean(xi0)
            xi0_var = np.var(xi0)

            p1_list.append(p_xi(is_continuous, [xi1_mean, xi1_var]))
            p0_list.append(p_xi(is_continuous, [xi0_mean, xi0_var]))
        else:  # 离散值时直接计算各类别的条件概率
            unique_value = xi.unique()  # 取值情况
            nvalue = len(unique_value)  # 取值个数

            xi1_value_count = pd.value_counts(xi1)[unique_value].fillna(0) + 1  #
计算正样本中，该属性每个取值的数量，并且加1，即拉普拉斯平滑
            xi0_value_count = pd.value_counts(xi0)[unique_value].fillna(0) + 1

            p1_list.append(p_xi(is_continuous, np.log(xi1_value_count / (m1 +
nvalue))))
            p0_list.append(p_xi(is_continuous, np.log(xi0_value_count / (m0 +
nvalue))))

    return p1, p1_list, p0_list


def predict_nb(x, p1, p1_list, p0_list):
    n = len(x)

    x_p1 = np.log(p1)
    x_p0 = np.log(1 - p1)
    for i in range(n):
        p1_xi = p1_list[i]
        p0_xi = p0_list[i]

        if p1_xi.is_continuous:
            mean1, var1 = p1_xi.conditional_pro
            mean0, var0 = p0_xi.conditional_pro
            x_p1 += np.log(1 / (np.sqrt(2 * np.pi) * var1) * np.exp(- (x[i] -
mean1) ** 2 / (2 * var1 ** 2)))
            x_p0 += np.log(1 / (np.sqrt(2 * np.pi) * var0) * np.exp(- (x[i] -
mean0) ** 2 / (2 * var0 ** 2)))
        else:
            x_p1 += p1_xi.conditional_pro[x[i]]
            x_p0 += p0_xi.conditional_pro[x[i]]

    if x_p1 > x_p0:
        return '是'
    else:
        return '否'


if __name__ == '__main__':
    data_path = r'C:\Users\hanmi\Documents\xiguabook\watermelon3_0_Ch.csv'
    data = pd.read_csv(data_path, index_col=0)

    X = data.iloc[:, :-1]
    y = data.iloc[:, -1]
    p1, p1_list, p0_list = train_nb(X, y)
```

```
    x_test = X.iloc[0, :]   # 书中测1 其实就是第一个数据

    print(predict_nb(x_test, p1, p1_list, p0_list))
```

7.5

首先看一下贝叶斯最优分类器：在书中p148中解释了对于最小化分类错误率的贝叶斯最优分类器可表示为：$h^*(x) = \arg\max_{c\in y} P(c|x)$，由贝叶斯定理即转换为$h^*(x) = \arg\max_{c\in y} P(x|c)P(c)$

那么在数据满足高斯分布时有:

$$h^*(x) = \arg\max_{c\in y} P(x|c)P(c) = \arg\max_{c\in y} log(f(x|c)P(c))$$

$$= \arg\max_{c\in y} log(\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu_c)^T\Sigma^{-1}(x-\mu_c))) + log(P(c))$$

$$= \arg\max_{c\in y} -\frac{1}{2}(x-\mu_c)^T\Sigma^{-1}(x-\mu_c) + log(P(c))$$

$$= \arg\max_{c\in y} x^T\Sigma^{-1}\mu_c - \frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c + log(P(c))$$

在二分类任务中,贝叶斯决策边界可表示为:

$$g(x) = x^T\Sigma^{-1}\mu_1 - x^T\Sigma^{-1}\mu_0 - (\frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1 - \frac{1}{2}\mu_0^T\Sigma^{-1}\mu_0) + log(\frac{P(1)}{P(0)})$$

$$= x^T\Sigma^{-1}(\mu_1-\mu_0) - \frac{1}{2}(\mu_1+\mu_0)^T\Sigma^{-1}(\mu_1-\mu_0) + \log(\frac{P(1)}{P(0)})$$

再看看线性判别分析：
书中p62给出式339，其投影原面可等效于$w = (\Sigma_0+\Sigma_1)^{-1}(\mu_1-\mu_0)$,注意为了和上面的推导一致，这里和书中给出的差了一个负号，但，但w位置没有改变，只是改变了方向而已。在两类别方差相同时有：$w = \frac{1}{2}\Sigma^{-1}(\mu_1-\mu_0)$ 两类别在投影面连线的中点可为
$\frac{1}{2}(\mu_1+\mu_0)^Tw = \frac{1}{4}(\mu_1+\mu_0)^T\Sigma^{-1}(\mu_1-\mu_0)$，那么线性判别分析的决策边界可表示为
$g(x) = x^T\Sigma^{-1}(\mu_1-\mu_0) - \frac{1}{2}(\mu_1+\mu_0)^T\Sigma^{-1}(\mu_1-\mu_0)$。

推导到这里发现贝叶斯最优分类器和线性判别分析的决策边界只相差$log(\frac{P(1)}{P(0)})$。在题目左边小字中有提及，“假设同先验”,所以$log(\frac{P(1)}{P(0)}) = 0$,于是得证。

7.7

这里"假设对于任何先验概率项的估算至少需 30 个样例"，意味着在所有样本中，任意c, x_i 的组合至少出现30次。
当$d = 1$时，即只有一个特征x_1,因为是二值属性，假设取值为 1,0 ,那为了估计$p(y=1, x_1=1)$ 至少需要30个样本，同样 $p(y=1, x_1=0)$ 需要额外30个样本，另外两种情况同理，所以在$d=1$ 时，最好和最坏情况都需要120个样本。
再考虑 $d=2$,多加个特征 x_2 同样取值 1,0 ,为了满足求 $P(c, x_1)$ 已经有了120个样本，且60个正样本和60个负样本，在最好的情况下，在60个正样本中，正好有30个样本中，正好有30个样本$x_2=1$, 30个 $x_2=0$,负样本同理，此时这120个样本也同样满足计算 $P(c, x_2)$ 的条件，所有$d=2$ 时，最好的情况也只需要120个样本，$d=n$ 时同理；在最坏的情况下，120个样子中，x_2 都取相同的值 1 ,那么为了估算 $P(c, x_2=0)$ 需要额外60个样本，总计180个样本，同理计算出$d=2,3,4...$ 时的样本数，即每多一个特征，最坏情况需要多加额外60个样本，$d=n$ 时，需要 $60(n+1)$ 个样本。
那么 d 个二值属性下，最好情况需要120个样本，最好情况需要 $60(d+1)$ 个样本。