Abstract

In this issue, we will learn an algorithm of linear classification - soft output - probability generation model: GDA (Gaussian Discriminant Analysis).

Idea

In the last issue, the logistic regression algorithm we learned belongs to probability discriminant model, so the difference between the discriminant model and the generation model is:

- the discriminant model is used to model the probability $p(y|x)$ directly to obtain its truly probability value.
- the generation model is used to model the joint distribution (x, y) via converting $p(y|x)$ to $p(x|y)p(y)$ according to bayes theorem: $p(y|x) = \frac{p(x|y)p(y)}{p(x)}$. Since $p(x)$ has nothing to do with y, it can be omitted. So, finally we get:

$$p(y|x) \propto p(x|y)p(y) = p(x; y) \tag{15}$$

when we are to predict any samples, we just need to compare $p(y = 0|x)$ and $p(y = 1|x)$.

Algorithm

Firstly, let's make some assumption about the model:

so all the parameters θ of the model are:

$$\theta = (\phi, \mu_1, \mu_2, \Sigma) \tag{16}$$

Then given the loss function of the model:

$$\begin{aligned} J(\theta) = log(p(Y|X)) &= log(\prod_{i=1}^{n} p(y_i|x_i)) \\ &= \sum_{i=1}^{n} log(p(y_i|x_i)) \end{aligned} \tag{17}$$

so:

$$\begin{aligned} \hat{\theta} = argmax(J(\theta)) &= argmax(\sum_{i=1}^{n} log(\frac{p(x_i|y_i)p(y_i)}{p(x_i)})) \\ &= argmax(\sum_{i=1}^{n} log(p(x_i|y_i)p(y_i))) \\ &= argmax(\sum_{i=1}^{n} y_i\ log(N(\mu_1, \Sigma)) + (1 - y_i)\ log(N(\mu_2, \Sigma)) + log(\phi^{y_i}(1-\phi)^{1-y_i})) \end{aligned} \tag{18}$$

solve ϕ

differentiate ϕ:

$$\sum_{i=1}^{N} \frac{y_i}{\phi} + \frac{y_i - 1}{1 - \phi} = 0 \Longrightarrow \phi = \frac{\sum_{i=1}^{N} y_i}{N} = \frac{N_1}{N} \tag{19}$$

In the formula, N, N_1, N_2 denote the number of all samples, positive samples, negative samples.

solve μ

make some derivations based on $J(\theta)$:

$$\begin{aligned}\hat{\mu_1} &= \underset{\mu_1}{argmax}\sum_{i=1}^{N} y_i \log N\left(\mu_1, \Sigma\right) \\ &= \underset{\mu_1}{argmax}\sum_{i=1}^{N} y_i \log\left(\frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}}exp(-\frac{1}{2}(x_i-\mu_1\right)^T(\Sigma)^{-1}(x_i-\mu_1))) \\ &= \underset{\mu_1}{argmin}\sum_{i=1}^{N} y_i(x_i-\mu_1)^T\Sigma^{-1}\left(x_i-\mu_1\right)\end{aligned} \tag{20}$$

In the above derivations, we quote the probability density function of multivariate Gaussian distribution:

$$p(x) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}}exp(-\frac{1}{2}(x_i-\mu_1)^T(\Sigma)^{-1}(x_i-\mu_1)) \tag{21}$$

In the function, p denote the number of random variables. Readers can multiply the probability density function of univariate Gaussian distribution, and derive the multivariate formula with the knowledge of linear algebra.

then differentiate the formula:

$$\begin{aligned}\frac{\partial\Delta}{\partial\mu_1} &= \sum_{i=1}^{N} -2y_i(\Sigma)^{-1}(x_i-\mu_1) = 0 \\ \Longrightarrow \mu_1 &= \frac{\sum_{i=1}^{N} y_i x_i}{\sum_{i=1}^{N} y_i} = \frac{\sum_{i=1}^{N} y_i x_i}{N_1}\end{aligned} \tag{22}$$

Since the positive samples and the negative samples are symmetrical, therefore:

$$\mu_2 = \frac{\sum_{i=1}^{N}(1-y_i)x_i}{N_2} \tag{23}$$

solve Σ

observe the first two terms of the formula:

$$\hat{\theta} = argmax(\sum_{i=1}^{n} y_i\ log(N(\mu_1,\Sigma)) + (1-y_i)\ log(N(\mu_2,\Sigma)) + log(\phi^{y_i}(1-\phi)^{1-y_i})) \tag{24}$$

We note that when $y=0$, the first term equals to 0; when $y=1$, the second term equals to 0.

thus the formula can be updated to:

$$\begin{aligned}\hat{\theta} &= argmax(\sum_{(x_i,y_i)\in C_1} log(N(\mu_1,\Sigma)) + \sum_{(x_i,y_i)\in C_2} log(N(\mu_2,\Sigma))) \\ &= argmax(\sum_{(x_i,y_i)\in C_1} -\frac{1}{2}\log|\Sigma| - \frac{1}{2}(x_i-\mu_1)^T(\Sigma)^{-1}(x_i-\mu_1)\ + \sum_{(x_i,y_i)\in C_2} -\frac{1}{2}|\Sigma| - \frac{1}{2}(x_i-\mu_2)^T(\Sigma)^{-1}(x_i-\mu_2))\end{aligned} \tag{25}$$

note the shape of $(x_i-\mu)^T(\Sigma)^{-1}(x_i-\mu) : (1,p)*(p,p)*(p,1) = (1,1)$, therefore, the trace(tr) operation can be applied to it, and it can be regarded as a matrix. Within the trace, the order of matrices can be exchanged at will.

$$\begin{aligned}\hat{\theta} &= argmax(-\frac{N}{2}\log|\Sigma| - \frac{1}{2}tr(\sum_{(x_i,y_i)\in C_1}(x_i-\mu_1)^T(\Sigma)^{-1}(x_i-\mu_1)) - \frac{1}{2}tr(\sum_{(x_i,y_i)\in C_2}(x_i-\mu_2)^T(\Sigma)^{-1}(x_i-\mu_2))) \\ &= argmax(-\frac{N}{2}\log|\Sigma| - \frac{1}{2}tr(\sum_{(x_i,y_i)\in C_1}(x_i-\mu_1)^T(x_i-\mu_1)(\Sigma)^{-1}) - \frac{1}{2}tr(\sum_{(x_i,y_i)\in C_2}(x_i-\mu_2)^T(x_i-\mu_2)(\Sigma)^{-1})) \\ &= argmax(-\frac{N}{2}\log|\Sigma| - \frac{1}{2}tr(N_1S_1(\Sigma)^{-1}) - \frac{1}{2}tr(N_2S_2(\Sigma)^{-1}))\end{aligned} \tag{26}$$

In the formula, S denote the co-variance matrix.

differentiate the formula:

$$\frac{\partial\Delta}{\partial\Sigma} = -\frac{1}{2}(N\frac{1}{|\Sigma|}|\Sigma|(\Sigma)^{-1} - N_1S_1(\Sigma)^{-2} - N_2S_2(\Sigma)^{-2}) = 0 \tag{27}$$

then we obtain the $\hat{\Sigma}$:

$$\begin{aligned} N\Sigma^{-1} - N_1 S_1^T \Sigma^{-2} - N_2 S_2^T \Sigma^{-2} = 0 \\ \Longrightarrow \hat{\Sigma} = \frac{N_1 S_1 + N_2 S_2}{N} \end{aligned} \tag{28}$$

Finally, when we are to predict any samples, we just need to compare $p(x|y=0)p(y=0)$ and $p(x|y=1)p(y=1)$.

Implement

```
import numpy as np
import os
os.chdir("../")
from models.linear_models import GDA

n1 = 1000
n_test = 100
x = np.linspace(0, 10, n1 + n_test)
w1, w2 = 0.3, 0.5
b1, b2 = 0.1, 0.2
x1 = x[:n1]
x_test = x[n1:]
v1 = x1 * w1 + b1
v2 = x1 * w2 + b2
cla_1 = np.c_[x1, v1]
cla_2 = np.c_[x1, v2]
l1 = np.ones(shape=(cla_1.shape[0], 1))
l2 = np.zeros(shape=(cla_2.shape[0], 1))
train_data = np.r_[cla_1, cla_2]
train_label = np.r_[l1, l2]

v_test = x_test * w2 + b2
data_test = np.c_[x_test, v_test]

model = GDA()
model.fit(train_data, train_label)
print(model.get_params())
print("accuary:", model.evaluate(data_test, 0))
```

```
(0.5, array([[4.54504095],
       [1.46351228]]), array([[4.54504095],
       [2.47252047]]), array([[14.96809381,  8.07251381],
       [ 8.07251381, 14.96809381]]))
accuary: 1
```