GDA & Naive Bayes

Patryk Krukowski

Politechnika Wrocławska
Wydział Matematyki

29 kwietnia 2021

Spis treści

1. Wstęp
2. Wielowymiarowy rozkład normalny - powtórzenie wiadomości
3. GDA
4. Naive Bayes

Wstęp

Rozważmy problem klasyfikacyjny dla ciągłych obserwacji.

Wstęp

Rozważmy problem klasyfikacyjny dla ciągłych obserwacji. Pomysły, jak do niego podejść:

- Regresja logistyczna

Wstęp

Rozważmy problem klasyfikacyjny dla ciągłych obserwacji. Pomysły, jak do niego podejść:

- Regresja logistyczna
- Perceptron

Wstęp

Rozważmy problem klasyfikacyjny dla ciągłych obserwacji. Pomysły, jak do niego podejść:

- Regresja logistyczna
- Perceptron

Zestaw takich algorytmów, które starają się dopasować pewną hiperpłaszczyznę rozdzielającą skupiska punktów pod warunkiem, że dysponujemy pewnymi obserwacjami ze zbioru uczącego, nazywamy **discriminative learning algorithms**.

Wstęp

Rozważmy problem klasyfikacyjny dla ciągłych obserwacji. Pomysły, jak do niego podejść:

- Regresja logistyczna
- Perceptron

Zestaw takich algorytmów, które starają się dopasować pewną hiperpłaszczyznę rozdzielającą skupiska punktów pod warunkiem, że dysponujemy pewnymi obserwacjami ze zbioru uczącego, nazywamy **discriminative learning algorithms**. Przeciwwagą do tych algorytmów są **generative learning algorithms**.

Wstęp

Główną ideą algorytmów typu generative jest stworzenie k (liczba klas) 'osobnych' modeli do tych skupisk, a następnie, biorąc nowe obserwacje, dopasowujemy je do odpowiednich grup.

Wstęp

Główną ideą algorytmów typu generative jest stworzenie k (liczba klas) 'osobnych' modeli do tych skupisk, a następnie, biorąc nowe obserwacje, dopasowujemy je do odpowiednich grup. Formalnie: algorytmy te uczą $P(x|y)$ oraz $P(y)$.

Wstęp

Główną ideą algorytmów typu generative jest stworzenie k (liczba klas) 'osobnych' modeli do tych skupisk, a następnie, biorąc nowe obserwacje, dopasowujemy je do odpowiednich grup. Formalnie: algorytmy te uczą $P(x|y)$ oraz $P(y)$.
Prawdopodoieństwo warunkowe można łatwo zamienić, używając wzoru Bayesa:

$$P(y|x) = \frac{P(x|y)\,P(y)}{P(x)},$$

gdzie

$$P(x) = \sum_{j=1}^{k} P(x|y=j)\,P(y=j)\,.$$

Wielowymiarowy rozkład normalny - powtórzenie wiadomości

Gęstość wielowymiarowego rozkładu normalnego definujemy jako:

Wielowymiarowy rozkład normalny - powtórzenie wiadomości

Gęstość wielowymiarowego rozkładu normalnego definujemy jako:

$$p\left(x;\mu,\Sigma\right) := \frac{1}{\left(2\pi\right)^{\frac{d}{2}}\left|\Sigma\right|^{\frac{1}{2}}} exp\left(-\frac{1}{2}\left(x-\mu\right)^{T}\Sigma^{-1}\left(x-\mu\right)\right),$$

gdzie μ to średnia, a Σ to macierz kowariancji.

Wielowymiarowy rozkład normalny - powtórzenie wiadomości

Gęstość wielowymiarowego rozkładu normalnego definujemy jako:

$$p\left(x;\mu,\Sigma\right) := \frac{1}{\left(2\pi\right)^{\frac{d}{2}}\left|\Sigma\right|^{\frac{1}{2}}} exp\left(-\frac{1}{2}\left(x-\mu\right)^{T}\Sigma^{-1}\left(x-\mu\right)\right),$$

gdzie μ to średnia, a Σ to macierz kowariancji. Przykłady - Jupyter.

GDA

GDA to skrót od Gaussian Discriminant Analysis. Z GDA wyróżniamy kolejne dwa modele: QDA, LDA. My się skoncentrujemy na LDA. Ponadto o modelu LDA zakładamy, że mamy rozkład warunkowy $x|y \sim \mathcal{N}(\mu_y, \Sigma)$

GDA

GDA to skrót od Gaussian Discriminant Analysis. Z GDA wyróżniamy kolejne dwa modele: QDA, LDA. My się skoncentrujemy na LDA. Ponadto o modelu LDA zakładamy, że mamy rozkład warunkowy $x|y \sim \mathcal{N}(\mu_y, \Sigma)$

$$p(x|y=k) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} exp\left(-\frac{1}{2}(x-\eta_k)^T \Sigma^{-1}(x-\eta_k)\right),$$

$$y \sim Beronulli(\phi)$$

GDA

GDA to skrót od Gaussian Discriminant Analysis. Z GDA wyróżniamy kolejne dwa modele: QDA, LDA. My się skoncentrujemy na LDA. Ponadto o modelu LDA zakładamy, że mamy rozkład warunkowy $x|y \sim \mathcal{N}(\mu_y, \Sigma)$

$$p(x|y=k) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} exp\left(-\frac{1}{2}(x-\eta_k)^T \Sigma^{-1}(x-\eta_k)\right),$$

$$y \sim Beronulli(\phi)$$

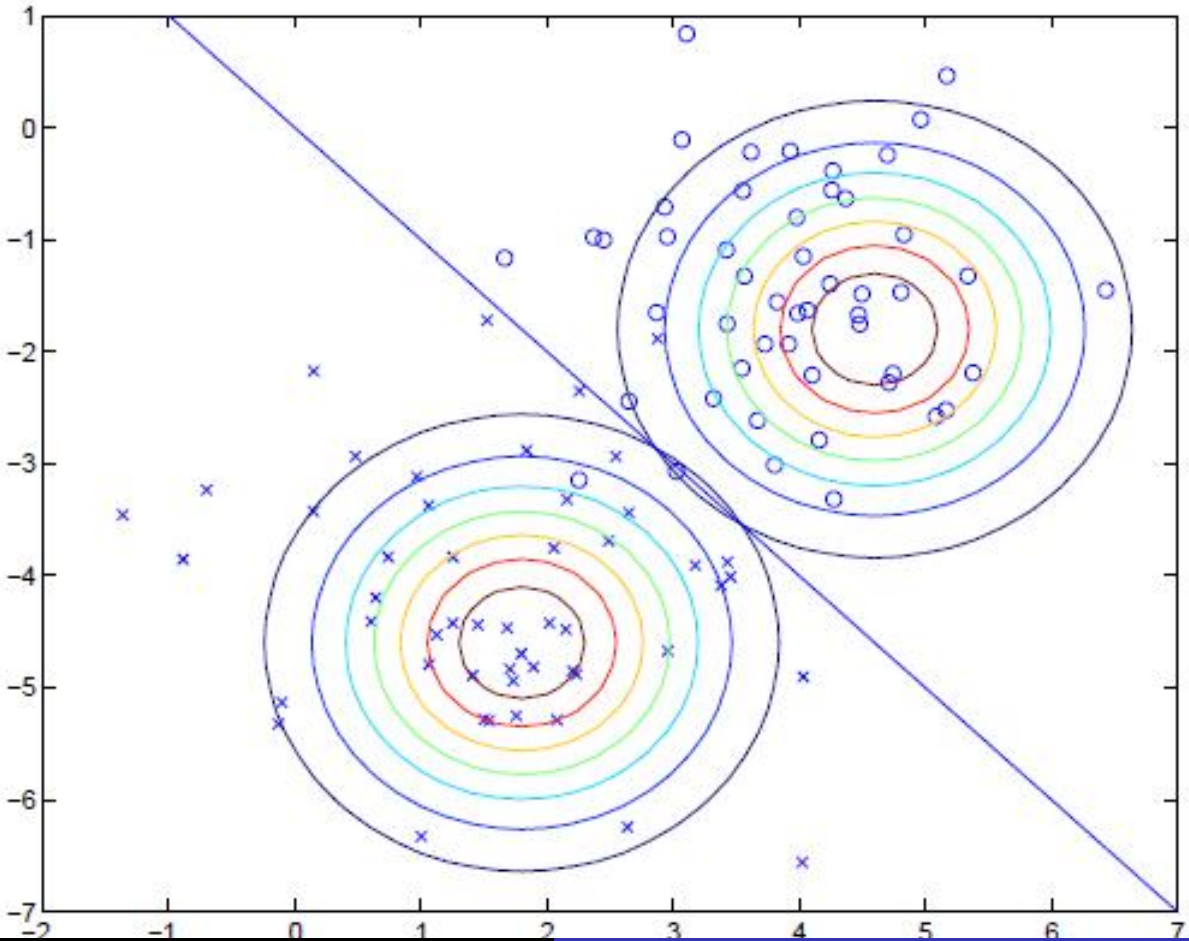
gdzie η_k to średnia (wektor) dla x. Parametry $\phi, \Sigma, \mu_1, \mu_1, \ldots, \mu_k$ szacujemy, używając metody największej wiarogodności.

GDA

Co tak naprawdę robi LDA?

GDA

Co tak naprawdę robi LDA?



GDA

Implementacja w Pythonie na przykładzie zbioru danych dotyczących irysów - Jupyter.

GDA

Dygresja - czy GDA i regresja logistyczna mają ze sobą coś wspólnego? Mówi nam o tym następujący fakt (przypadek binarny!)

Fakt

$$p\left(y=1|x;\phi,\Sigma,\mu_0,\mu_1\right)=\frac{1}{1+exp\left(-\theta^T x\right)}$$

GDA

Dygresja - czy GDA i regresja logistyczna mają ze sobą coś wspólnego? Mówi nam o tym następujący fakt (przypadek binarny!)

Fakt

$$p(y = 1|x; \phi, \Sigma, \mu_0, \mu_1) = \frac{1}{1 + exp(-\theta^T x)}$$

Dowód-szkic.

Rozpisujemy wyrażenie z faktu za pomocą wzoru Bayesa. Dzielimy licznik i mianownik ułamka przez licznik i w efekcie dostajemy żądaną postać z parametrem θ, która będzie funkcją parametrów $\phi, \Sigma, \mu_0, \mu_1$. □

GDA

Dlaczego funkcją decyzyjną w LDA jest jest właśnie funkcja liniowa?

GDA

Dlaczego funkcją decyzyjną w LDA jest jest właśnie funkcja liniowa? Oznaczmy $\pi_k = p(y = k)$. Mamy

$$p(y = k|x) = \frac{f_k(x)\,\pi_k}{p(x)},$$

gdzie $f(\cdot)$ jest gęstością standardowego rozkładu normalnego.

GDA

Dlaczego funkcją decyzyjną w LDA jest jest właśnie funkcja liniowa? Oznaczmy $\pi_k = p(y = k)$. Mamy

$$p(y = k|x) = \frac{f_k(x)\,\pi_k}{p(x)},$$

gdzie $f(\cdot)$ jest gęstością standardowego rozkładu normalnego.

$$\log p(y = k|x) = C + \log \pi_k - \frac{1}{2}(x - \eta_k)^T \Sigma^{-1}(x - \eta_k),$$

gdzie C to stała.

GDA

Dlaczego funkcją decyzyjną w LDA jest jest właśnie funkcja liniowa? Oznaczmy $\pi_k = p(y = k)$. Mamy

$$p(y = k|x) = \frac{f_k(x)\,\pi_k}{p(x)},$$

gdzie $f(\cdot)$ jest gęstością standardowego rozkładu normalnego.

$$\log p(y = k|x) = C + \log \pi_k - \frac{1}{2}(x - \eta_k)^T \Sigma^{-1}(x - \eta_k),$$

gdzie C to stała. Ale to jest równe

$$C_1 + \log \pi_k - \frac{1}{2}\eta_k^T \Sigma^{-1} \eta_k + x^T \Sigma^{-1} \eta_k,$$

C_1 stała.

GDA

Def. $\delta_k = \log \pi_k - \frac{1}{2}\eta_k^T \Sigma^{-1} \eta_k + x^T \Sigma^{-1} \eta_k$, natomiast zbiór decyzyjny dla klasy k i l definiujemy jako $\{\delta_k = \delta_l\}$, stąd liniowość ograniczeń.

GDA

Def. $\delta_k = \log \pi_k - \frac{1}{2}\eta_k^T \Sigma^{-1} \eta_k + x^T \Sigma^{-1} \eta_k$, natomiast zbiór decyzyjny dla klasy k i l definiujemy jako $\{\delta_k = \delta_l\}$, stąd liniowość ograniczeń.
Pamiętajmy, że odpowiednie parametry musimy estymować!

Naive Bayes

Algorytmu tego używamy w przypadku, gdy mamy do czynienia z **dyskretnym** problemem klasyfikacyjnym (z ciągłym też można), np. klasyfikacja wiadomości na naszej poczcie - spam, czy nie spam?

Naive Bayes

Algorytmu tego używamy w przypadku, gdy mamy do czynienia z **dyskretnym** problemem klasyfikacyjnym (z ciągłym też można), np. klasyfikacja wiadomości na naszej poczcie - spam, czy nie spam?
Pomysł: szukamy słów charakterstycznych dla spamu i te, które pojawiają się w mailu, kodujemy jako jedynkę, pozostałe jako zera.

Naive Bayes

Algorytmu tego używamy w przypadku, gdy mamy do czynienia z **dyskretnym** problemem klasyfikacyjnym (z ciągłym też można), np. klasyfikacja wiadomości na naszej poczcie - spam, czy nie spam?
Pomysł: szukamy słów charakterstycznych dla spamu i te, które pojawiają się w mailu, kodujemy jako jedynkę, pozostałe jako zera.
Problem? Taki wektor będzie miał zbyt duży wymiar! Z pomocą przychodzi **Naive Bayes**.

Naive Bayes

By modelować zatem $p(x|y)$, zakładamy, że obserwacje x_i są parami warunkowo niezależne przy ustalonym y (etykieta). Stąd *naive*. Mamy

Naive Bayes

By modelować zatem $p(x|y)$, zakładamy, że obserwacje x_i są parami warunkowo niezależne przy ustalonym y (etykieta). Stąd *naive*. Mamy

$$\begin{aligned} & p(x_1, \ldots, x_n|y) \\ & = p(x_1|y)\, p(x_2|y, x_1)\, p(x_3|y, x_1, x_2) \ldots \; p(x_n|y, x_1, \ldots x_{n-1}) \\ & = \prod_{j=1}^{n} p(x_j|y), \end{aligned}$$

a stąd znajdujemy znajdujemy klasę k, która nas interesuje (jako argmax z iloczynu prawdopobieństw przedstawionego powyżej przemnożonego przez p(y)).

Naive Bayes

Parametry do modelu, gdy x i y są binarne:

- $\phi_{j|y=1} = \frac{\sum_{i=1}^{n} \mathbf{1}\{x_j^{(i)}=1 \wedge y^{(i)}=1\}}{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=1\}}$

Naive Bayes

Parametry do modelu, gdy x i y są binarne:

- $\phi_{j|y=1} = \frac{\sum_{i=1}^{n} \mathbf{1}\{x_j^{(i)}=1 \wedge y^{(i)}=1\}}{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=1\}}$
- $\phi_{j|y=0} = \frac{\sum_{i=1}^{n} \mathbf{1}\{x_j^{(i)}=1 \wedge y^{(i)}=0\}}{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=0\}}$

Naive Bayes

Parametry do modelu, gdy x i y są binarne:

- $\phi_{j|y=1} = \frac{\sum_{i=1}^{n} \mathbf{1}\{x_j^{(i)}=1 \wedge y^{(i)}=1\}}{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=1\}}$
- $\phi_{j|y=0} = \frac{\sum_{i=1}^{n} \mathbf{1}\{x_j^{(i)}=1 \wedge y^{(i)}=0\}}{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=0\}}$
- $\phi_y = \frac{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=0\}}{n}$

Naive Bayes

Parametry do modelu, gdy x i y są binarne:

- $\phi_{j|y=1} = \frac{\sum_{i=1}^{n} \mathbf{1}\{x_j^{(i)}=1 \wedge y^{(i)}=1\}}{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=1\}}$
- $\phi_{j|y=0} = \frac{\sum_{i=1}^{n} \mathbf{1}\{x_j^{(i)}=1 \wedge y^{(i)}=0\}}{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=0\}}$
- $\phi_y = \frac{\sum_{i=1}^{n} \mathbf{1}\{y^{(i)}=0\}}{n}$

Implementacja na przykładzie zmiennych ciągłych i dyskretnych - Jupyter.