

CSE 5523: HW2+3



Outline

- You are to implement:
 - Linear logistic regression
 - Pocket algorithm (improved perceptron)
 - Linear soft-margin SVM
 - Linear Naïve Bayes
 - Linear Gaussian discriminative analysis
 - Nonlinear Naïve Bayes
 - Nonlinear Gaussian discriminative analysis

Data

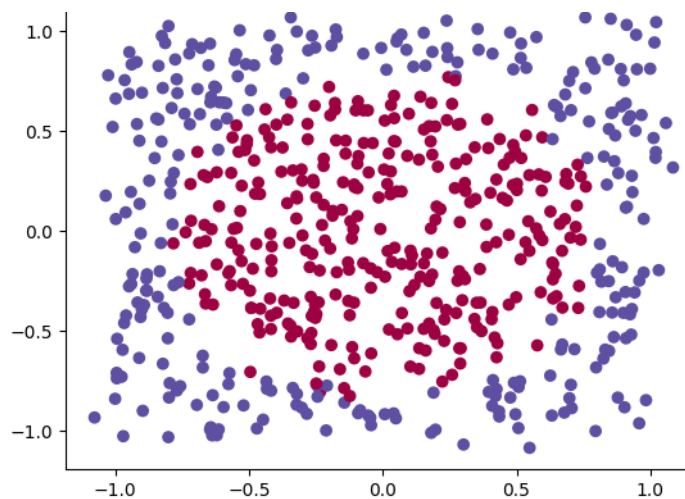
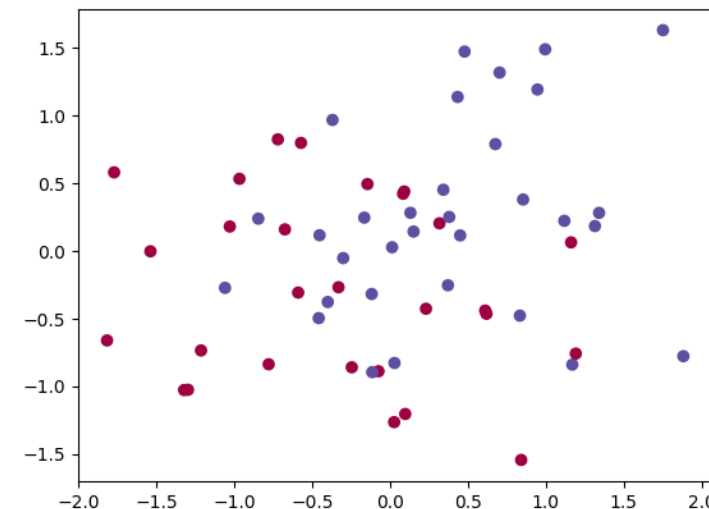
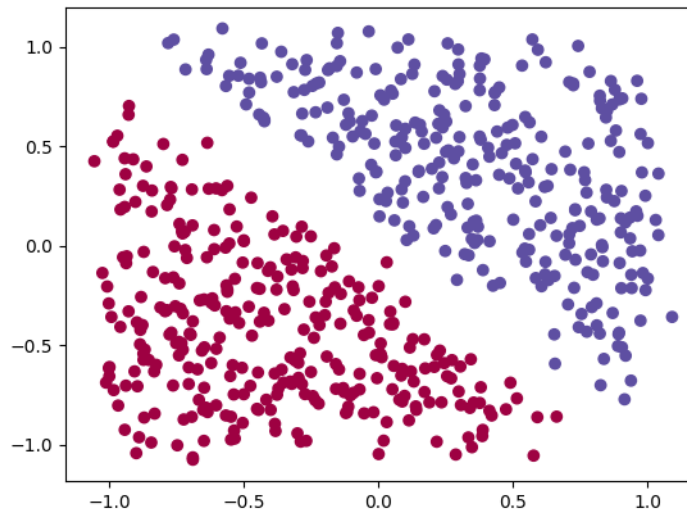
- Four data source

- 2D linear
- 2D noisy linear
- 2D quadratic (circle)
- MNIST (<5 vs. >=5)

- $X \in \mathbb{R}^{D \times N}$:

- A column as an instance

- $Y \in \{+1, -1\}^{N \times 1}$



Data

- The data \mathbf{X} are not appended with “1” yet.
- For feature transform for a 2D data instance $\mathbf{x} \in \mathbb{R}^2$, we do

- $\phi(\mathbf{x}) = \begin{bmatrix} x[1] \\ x[2] \\ x[1]^2 \\ x[2]^2 \\ x[1] \times x[2] \end{bmatrix}$

- Again, you need to append “1” to the data $\phi(\mathbf{x})$ if you want to solve $\tilde{\mathbf{w}}$ directly
 - In the homework, we have done $\phi(\mathbf{x})$ for you!

Accuracy

- Data = $\{ (\mathbf{x}_i, y_i \in \{+1, -1\}) \}_{i=1}^N$
- Accuracy = $\frac{1}{N} \sum_{i=1}^N \mathbf{1}[\hat{y}_i == y_i]$, where \hat{y}_i is the prediction based on \mathbf{x}_i

Logistic regression



Logistic regression

- **Training data:** $D_{tr} = \left\{ \left(\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{+1, -1\} \right) \right\}_{i=1}^N$
- **Model:** $\text{sign}(\mathbf{w}\mathbf{x} + b) = \text{sign}(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})$
- **Objective:** $E(\tilde{\mathbf{w}}) = \frac{1}{N} \sum_{i=1}^N \log \left(1 + e^{-y_i \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_i} \right)$
 - Please add a $\frac{1}{N}$ for normalization

Gradient descent (GD) for logistic regression

- Initialize $\tilde{\mathbf{w}}$
- For $t = 1:T$
 - $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}} - \eta \nabla_{\tilde{\mathbf{w}}} E(\tilde{\mathbf{w}})$; $E(\tilde{\mathbf{w}})$: the loss function you want to minimize!
 - No need to stop earlier
- Note:
 - If $E(\tilde{\mathbf{w}}) = \frac{1}{N} \sum_n e_n(\tilde{\mathbf{w}})$; for example, $e_n(\tilde{\mathbf{w}})$ is the loss on the i -th example
 - $\nabla_{\tilde{\mathbf{w}}} E(\tilde{\mathbf{w}}) = \frac{1}{N} \sum_n \nabla_{\tilde{\mathbf{w}}} e_n(\tilde{\mathbf{w}})$

Gradient descent (GD) for logistic regression

- $e_n(\tilde{\mathbf{w}}) = \log \left(1 + e^{-y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n} \right)$ for $y_i \in \{+1, -1\}$

$$\begin{aligned}\nabla_{\tilde{\mathbf{w}}} e_n(\tilde{\mathbf{w}}) &= \frac{1}{1 + e^{-y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n}} \times \nabla_{\tilde{\mathbf{w}}} \left(1 + e^{-y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n} \right) \\&= \frac{1}{1 + e^{-y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n}} \times (-y_n \tilde{\mathbf{x}}_n) \times e^{-y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n} \\&= \rho(y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) \times (-y_n \tilde{\mathbf{x}}_n) \times e^{-y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n} \\&= \rho(y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) \times (-y_n \tilde{\mathbf{x}}_n) \times \frac{1 - \rho(y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}})}{\rho(y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}})} \\&= -y_n \left(1 - \rho(y_n \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) \right) (\tilde{\mathbf{x}}_n)\end{aligned}$$

Gradient descent (GD) for logistic regression

- $e_n(\tilde{\mathbf{w}}) = -[y_n \log p(+1|\mathbf{x}_n; \boldsymbol{\theta}) + (1 - y_n) \log(1 - p(+1|\mathbf{x}_n; \boldsymbol{\theta}))]$ for $y_i \in \{+1, 0\}$

$$\begin{aligned}\nabla_{\tilde{\mathbf{w}}} e_n(\tilde{\mathbf{w}}) &= -[y_n \nabla_{\tilde{\mathbf{w}}} \log p(+1|\mathbf{x}_n; \boldsymbol{\theta}) + (1 - y_n) \nabla_{\tilde{\mathbf{w}}} \log(1 - p(+1|\mathbf{x}_n; \boldsymbol{\theta}))] \\ &= -[y_n \nabla_{\tilde{\mathbf{w}}} \log \rho(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}) + (1 - y_n) \nabla_{\tilde{\mathbf{w}}} \log(1 - \rho(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}))] \\ &= -[y_n \times (1 - \rho(\mathbf{w}^T \mathbf{x}_n)) \mathbf{x}_n - (1 - y_n) \times \rho(\mathbf{w}^T \mathbf{x}_n) \mathbf{x}_n] \\ &= -(y_n - \rho(\mathbf{w}^T \mathbf{x}_n)) \mathbf{x}_n\end{aligned}$$

Pocket algorithm



Pocket algorithm

- **Training data:** $D_{tr} = \left\{ \left(\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{+1, -1\} \right) \right\}_{i=1}^N$
- **Model:** $\text{sign}(\mathbf{w}\mathbf{x} + b) = \text{sign}(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})$

Pocket algorithm

- Initialize $\tilde{\mathbf{w}}$ and $\tilde{\mathbf{w}}^{\text{best}}$
- For $t = 1:T$
 - Loop for all training examples $\tilde{\mathbf{x}}_n$ (random order!)
 - Predict $\hat{y}_n = \text{sign}(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n)$
 - If $\hat{y}_n \neq y_n$
 - Update: $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}} + \eta(y_n \tilde{\mathbf{x}}_n)$
 - Evaluate $\tilde{\mathbf{w}}$ on the “training data” and calculate the training accuracy
 - If training accuracy by $\tilde{\mathbf{w}}$ is “higher” than the training accuracy by $\tilde{\mathbf{w}}^{\text{best}}$
 - $\tilde{\mathbf{w}}^{\text{best}} \leftarrow \tilde{\mathbf{w}}$
- Output $\tilde{\mathbf{w}}^{\text{best}}$

Soft-margin SVM algorithm



Soft-margin SVM

- **Training data:** $D_{tr} = \left\{ \left(\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{+1, -1\} \right) \right\}_{i=1}^N$
- **Model:** $\text{sign}(\mathbf{w}\mathbf{x} + b)$
- **Objective:** $E(\mathbf{w}, b) = \frac{1}{N} \sum_n \max\{1 - y_n(\mathbf{w}^T \mathbf{x}_n + b), 0\} + \frac{1}{2} \lambda \mathbf{w}^T \mathbf{w}$
 - Please add a $\frac{1}{N}$ for normalization
 - λ : regularization coefficients (i.e., reg_coeff)

Gradient descent (GD) for soft-margin SVM

- Initialize $\tilde{\mathbf{w}}$
- For $t = 1:T$
 - $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}} - \eta \nabla_{\tilde{\mathbf{w}}} E(\tilde{\mathbf{w}})$; $E(\tilde{\mathbf{w}})$: the loss function you want to minimize!
 - No need to stop earlier
- Note:
 - If $E(\mathbf{w}, b) = \frac{1}{N} \sum_n e_n(\mathbf{w}, b) + \frac{1}{2} \lambda \mathbf{w}^T \mathbf{w}$
 - $\nabla_{\mathbf{w}} E(\mathbf{w}, b) = \frac{1}{N} \sum_n \nabla_{\mathbf{w}} e_n(\mathbf{w}, b) + \lambda \mathbf{w}$
 - $\nabla_b E(\mathbf{w}, b) = \frac{1}{N} \sum_n \nabla_b e_n(\mathbf{w}, b)$

Gradient descent (GD) for soft-margin SVM

- $\nabla_{\mathbf{w}} e_n(\mathbf{w}, b) = \begin{cases} -y_n \mathbf{x}_n, & \text{if } y_n(\mathbf{w}^T \mathbf{x}_n + b) < 1 \\ 0, & \text{otherwise} \end{cases}$
- $\nabla_b e_n(\mathbf{w}, b) = \begin{cases} -y_n, & \text{if } y_n(\mathbf{w}^T \mathbf{x}_n + b) < 1 \\ 0, & \text{otherwise} \end{cases}$

Naïve Bayes



Naïve Bayes

- **Training data:** $D_{tr} = \{ (\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{+1, -1\}) \}_{i=1}^N$
- **Goal:** construct $p(Y = c|\mathbf{x})$ for $\hat{y} = \max_{c \in \{+1, -1\}} p(Y = c|\mathbf{x})$
- **Bayes' rules:** $p(Y = c|\mathbf{x}) = p(\mathbf{x}|Y = c)p(Y = c)$
 - $p(Y = c)$: Bernoulli
 - $p(\mathbf{x}|Y = c) = p(x[1], x[2], x[3], \dots, x[D] | Y = c)$
$$= p(x[1] | Y = c)p(x[2] | Y = c) \dots p(x[D] | Y = c) = \prod_{d=1}^D p(x[d] | Y = c)$$
 - $p(x[d] | Y = c)$: one-dimensional Gaussian

Nonlinear Naïve Bayes

- $p(x[d] \mid Y = +1)$ and $p(x[d] \mid Y = -1)$ have their own standard deviations $\sigma_{d,+1}$ and $\sigma_{d,-1}$
- See slides 11 or 12 for how to compute them
- You can represent $[\sigma_{1,+1}, \sigma_{2,+1}, \dots, \sigma_{N,+1}]^T$ as a vector

Linear Naïve Bayes

- $p(x[d] | Y = +1)$ and $p(x[d] | Y = -1)$ share the same standard deviation σ_d
- Built upon the previous slide, given $\sigma_{d,+1}$, $\sigma_{d,-1}$ and let N_{+1} , N_{-1} be the number of training examples per class, $\sigma_d^2 = \frac{N_{+1} \times \sigma_{d,+1}^2 + N_{-1} \times \sigma_{d,-1}^2}{N}$
- You can represent $[\sigma_1, \sigma_2, \dots, \sigma_N]^T$ as a vector

Prediction (please do “log” to prevent overflow)

$$\max_{c \in \{+1, -1\}} p(Y = c | \mathbf{x}) = \max_{c \in \{+1, -1\}} p(\mathbf{x} | Y = c) p(Y = c)$$

$$= \max_{c \in \{+1, -1\}} p(\mathbf{x} | Y = c) \prod_{d=1}^D p(x[d] | Y = c)$$

$$= \max_{c \in \{+1, -1\}} \log p(\mathbf{x} | Y = c) + \sum_{d=1}^D \log p(x[d] | Y = c)$$

Gaussian discriminant analysis



GDA

- **Training data:** $D_{tr} = \{ (\mathbf{x}_i \in \mathbb{R}^D, y_i \in \{+1, -1\}) \}_{i=1}^N$
- **Goal:** construct $p(Y = c|\mathbf{x})$ for $\hat{y} = \max_{c \in \{+1, -1\}} p(Y = c|\mathbf{x})$
- **Bayes' rules:** $p(Y = c|\mathbf{x}) = p(\mathbf{x}|Y = c)p(Y = c)$
 - $p(Y = c)$: Bernoulli
 - $p(\mathbf{x}|Y = c)$: multi-dimensional Gaussian

Nonlinear GDA

- $p(\mathbf{x} | Y = +1)$ and $p(\mathbf{x} | Y = -1)$ have their own covariance matrices Σ_{+1}, Σ_{-1}
- See slides 10, 11 for how to compute them
- See also your homework # 2

Linear GDA

- $p(\mathbf{x} | Y = +1)$ and $p(\mathbf{x} | Y = -1)$ share the same covariance matrix Σ
- Built upon the previous slide, given Σ_{+1}, Σ_{-1} and let N_{+1}, N_{-1} be the number of training examples per class, $\Sigma = \frac{N_{+1} \times \Sigma_{+1} + N_{-1} \times \Sigma_{-1}}{N}$
- See your homework # 2 for how to compute it

Prediction (please do “log” to prevent overflow)

$$\max_{c \in \{+1, -1\}} p(Y = c | \mathbf{x}) = \max_{c \in \{+1, -1\}} p(\mathbf{x} | Y = c) p(Y = c)$$

$$= \max_{c \in \{+1, -1\}} \log p(\mathbf{x} | Y = c) + \log p(Y = c)$$