Pattern Recognition
Lecture 03-2

Gradient Descent & Kernel Trick

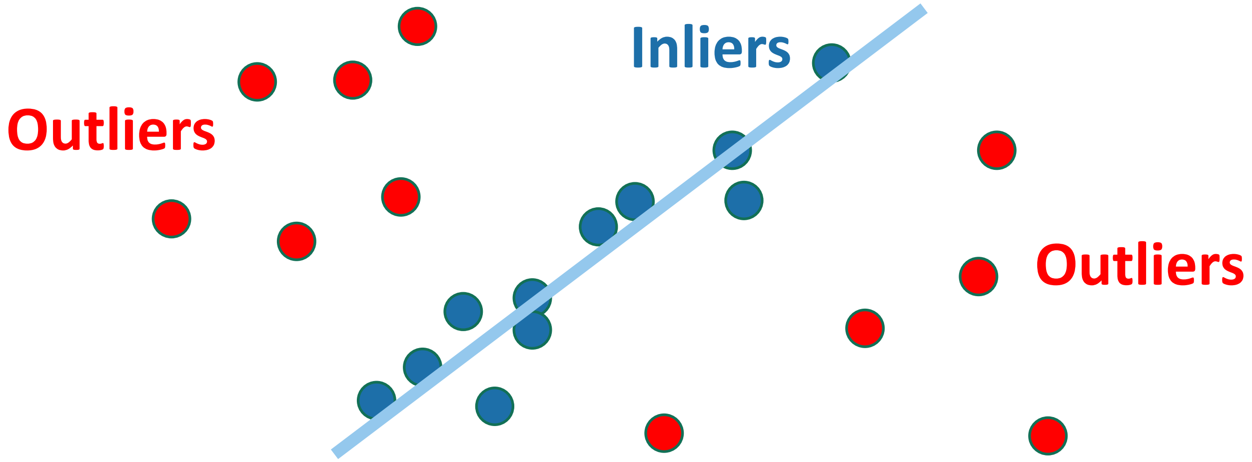
Prof. Jongwon Choi
Chung-Ang University
Fall 2022

This Class

- Gradient Descent
- Robust Regression
- Regularization
- **RANSAC**
- **Kernel Trick**

RANSAC

- In computer vision, a widely-used generic framework for robust fitting is random sample consensus (RANSAC)
- This is designed for the scenario where:
  - You have a large number of outliers
  - Majority of points are “Inliners” (Very easy to get low errors on the inliers)

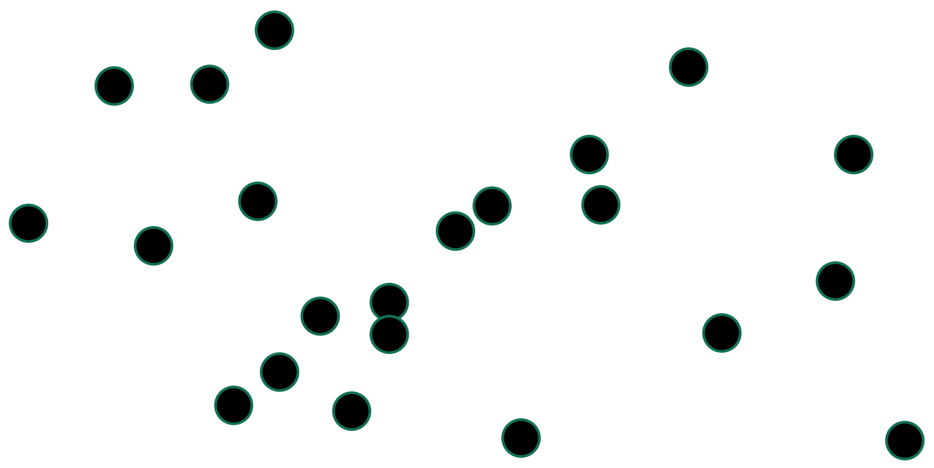


RANSAC

- 1. Sample a small number of training exmples
  - Minimum number needed to fit the model
  - For linear regression with 1 feature, just 2 examples
- 2. Fit the model based on the samples
  - Fit a line to these 2 points
  - With ‘d’ features, you’ll need ‘d+1’ examples
- 3. Test how many points are fit well based on the model
- 4. Repeat until we find a model with the most inliers
- 5. Re-fit the model based on the estimated “inliers”

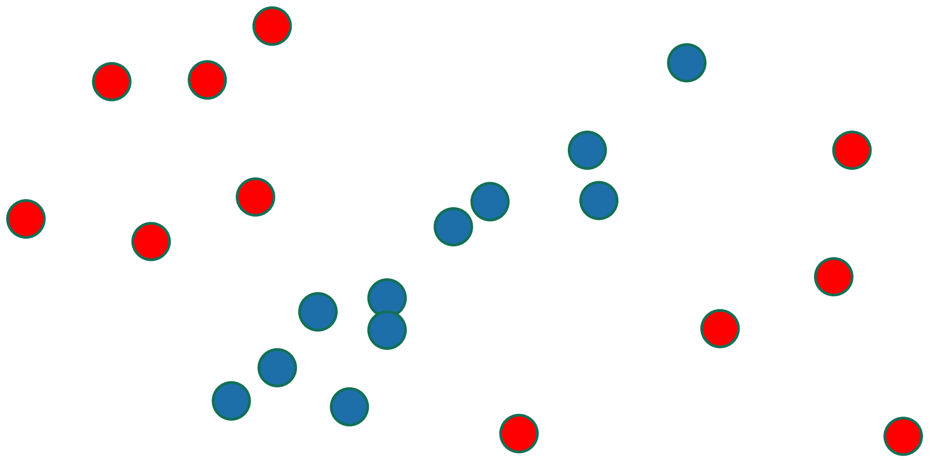
RANSAC

- 1. Sample a small number of training exmples
- 2. Fit the model based on the samples
- 3. Test how many points are fit well based on the model



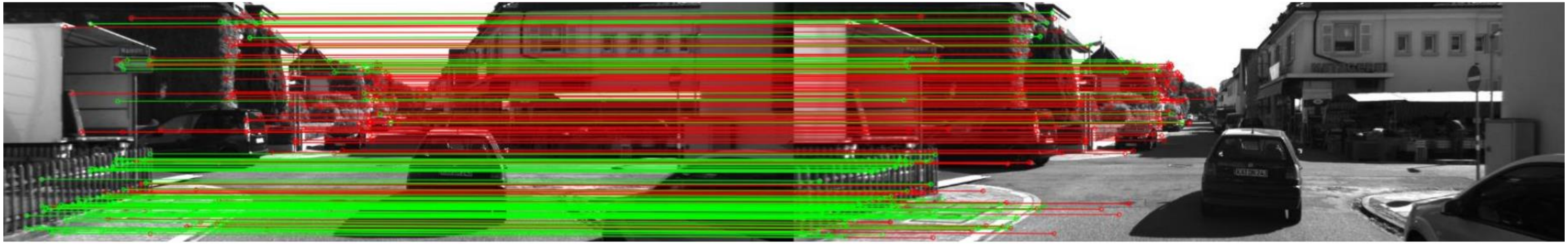
RANSAC

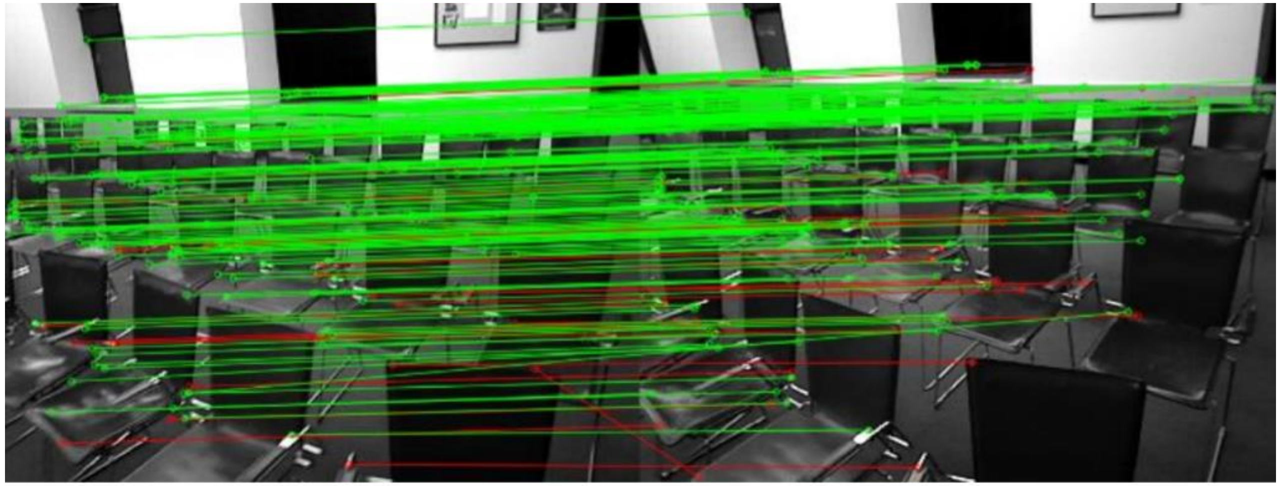
- 4. Repeat until we find a model with the most inliers
- 5. Re-fit the model based on the estimated “inliers”



RANSAC - Example

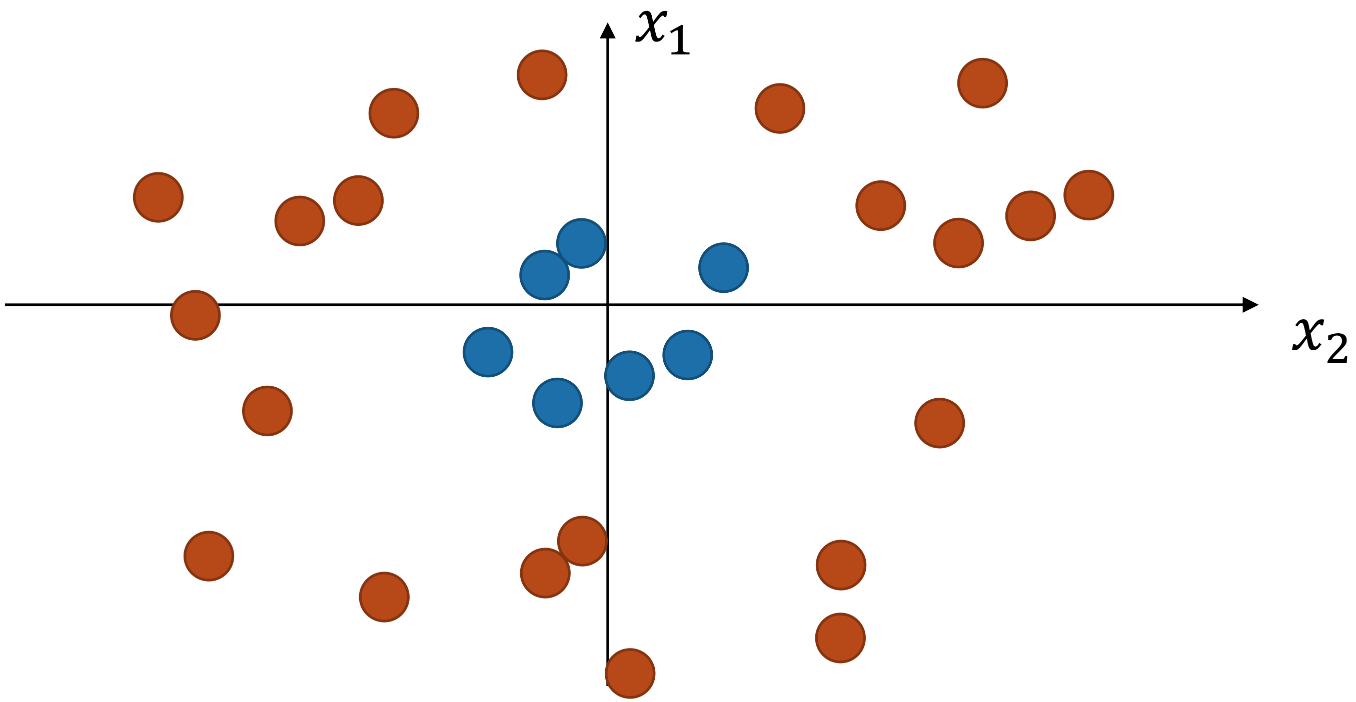
- Structure from motion – Feature matching





Kernel Trick

- Support Vector Machines for Non-separable
  - What should we do for separating the following case?:



Kernel Trick

- Support Vector Machines for Non-separable
  - What should we do for separating the following case:
    - It may be separable under change of basis (ex. $y_i = 2_1 x_{i1}^2 + w_2 \sqrt{2} x_{i1} x_{i2} + w_3 x_{i2}^2$

Multi-dimensional Polynomial Basis

- Recall the fitting polynomials when we only have 1 feature:
  - $\hat{y}_i = w_0 + w_1 x_i + w_2 x_i^2$
- Then, how can we do this when we have a lot of features?
  - It's simple! Consider all of them!
  - $1, x_{i1}, x_{i2}, x_{i3}$
  - $x_{i1}^2, x_{i2}^2, x_{i3}^2$
  - $x_{i1}x_{i2}, x_{i1}x_{i3}, x_{i2}x_{i3}$

Kernel Trick

- If we go to degree p=5, we'll have $O(d^5)$ terms!!
- For large 'd' and 'p', storing a polynomial basis is intractable
  - Z has $k = O(d^p)$ columns, so it does not fit in memory
- Fortunately, we can use all of them with the "kernel trick"

Kernel Trick – L2-regularized Least Squares

- $f(v) = \frac{1}{2}\|\mathbf{Zv} - \mathbf{y}\|^2 + \frac{\lambda}{2}\|\mathbf{v}\|^2$
  - Z : kernel basis
  - v : kernel weights
- Then, $\mathbf{v} = \underbrace{(\mathbf{Z}^T\mathbf{Z} + \lambda\mathbf{I})}_{k \times k}{}^{-1}\mathbf{Z}^T\mathbf{y} = \mathbf{Z}^T\underbrace{(\mathbf{Z}\mathbf{Z}^T + \lambda\mathbf{I})}_{n \times n}{}^{-1}\mathbf{y}$
- This is faster if n<<k:
  - Cost is $O(n^2k + n^3)$ instead of $O(nk^2 + k^3)$
  - But for the polynomial basis, this is still too slow since $k = O(d^p)$

Kernel Trick – L2-regularized Least Squares

- Given test data $\widetilde{\mathbf{X}}$, Predict $\hat{\mathbf{y}}$ by forming $\tilde{\mathbf{Z}}$ and the using:
  - $\hat{\mathbf{y}} = \tilde{\mathbf{Z}}\boldsymbol{v} = \tilde{\mathbf{Z}}\mathbf{Z}^T(\mathbf{Z}\mathbf{Z}^T + \lambda\mathbf{I})^{-1}\mathbf{y} \equiv \widetilde{\mathbf{K}}(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y}$
  - $\widetilde{\mathbf{K}} \in \mathcal{R}^{t \times n}$
  - $\mathbf{K} + \lambda\mathbf{I} \in \mathcal{R}^{n \times n}$
- Notice that if you have $\mathbf{K}$ and $\widetilde{\mathbf{K}}$ then you do not need $\mathbf{Z}$ and $\tilde{\mathbf{Z}}$
- **Key idea of kernel trick for certain bases:**
  - **We can efficiently compute K and $\widetilde{\mathrm{K}}$**
  - **even though Z and $\tilde{\mathrm{Z}}$ are interactable.**

Kernel Trick – Gram Matrix

- $\mathbf{K} = \mathbf{Z}\mathbf{Z}^T$
  - Contains the dot products between all training examples

- $\widetilde{\mathbf{K}} = \tilde{\mathbf{Z}}\mathbf{Z}^T$
  - Contains the dot products between train and test examples

- Kernel function: $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{z}_i^T \mathbf{z}_j$

Kernel Trick

- Kernel function: $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{z}_i^T \mathbf{z}_j$
  - Instead of the acquisition of $\mathbf{z}_i$ and $\mathbf{z}_j$,
  - Directly estimate the Gram matrices from $\mathbf{x}_i$ and $\mathbf{x}_j$

- Linear Kernel: $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j = \mathbf{z}_i^T \mathbf{z}_j$
- Polynomial Kernel: $k(\mathbf{x}_i, \mathbf{x}_j) = \left(1 + \mathbf{x}_i^T \mathbf{x}_j\right)^p = \mathbf{z}_i^T \mathbf{z}_j$
- Gaussian-RBF Kernel: $k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) = \mathbf{z}_i^T \mathbf{z}_j$

Maximum Likelihood Estimation (MLE)

- $\hat{\mathbf{w}} = \arg\max_{w} p(\mathbf{D}|\mathbf{w})$
  - Example. Coin throwing data = {H,H,T}
    - Likelihood p(HHT | w=0.5) = 0.125
    - Likelihood p(HHT | w=0.0) = 0.0
    - Likelihood p(HHT | w=2/3) = 0.144 (Maximum)
- Minimizing the Negative Log-Likelihood (NLL)
  - $\hat{\mathbf{w}} = \arg\max_{\mathrm{w}} p(\mathbf{D}|\mathbf{w}) = \arg\min_{\mathrm{w}}(-\log p(\mathbf{D}|\mathbf{w}))$
  - Because 'log' is monotonically increasing
  - Then, the likelihood can be represented by addition when i.i.d data are given
    - $\hat{\mathbf{w}} = \arg\min_{w}(-\log \prod_{i=1}^{n} p(\mathbf{D}_i|\mathbf{w})) = \arg\min_{\mathrm{w}}(-\sum_{i=1}^{n} \log p(\mathbf{D}_i|\mathbf{w}))$

Maximum A Priori (MAP)

- $\hat{\mathbf{w}} = \arg\max_{w} p(\mathbf{w}|\mathbf{D})$

- Through Bayes rule and NLL,
  - $p(\mathbf{w}|\mathbf{D}) = \frac{p(\mathbf{D}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{D})} \propto p(\mathbf{D}|\mathbf{w})p(\mathbf{w})$
  - $\hat{\mathbf{w}} = \arg\max_{\mathbf{w}} p(\mathbf{D}|\mathbf{w}) = \arg\min_{\mathbf{w}}(-\log p(\mathbf{D}|\mathbf{w})p(\mathbf{w}))$
  - When i.i.d data are given,
  - $\hat{\mathbf{w}} = \arg\min(-\sum_{i=1}^{n} \log p(\mathbf{D}_i|\mathbf{w}) - \log p(\mathbf{w}))$

MLE vs. MAP

Whose?

- MLE : $\arg\max_{w} p(\mathbf{D}|\mathbf{w})$
  - Classify it based on the hair length, color, shape, etc.
- MAP : $\arg\max_{w} p(\mathbf{w}|\mathbf{D})$
  - Classify it based on the hair length, color, shape, etc. while considering the ratio of cat and dog at the place (Of course, it is very hard to obtain)

This Class

- Gradient Descent
- Robust Regression
- Regularization
- **RANSAC**
- **Kernel Trick**