

Machine Learning

Neural networks, Unsupervised learning

DSC 240

Feb 27, 2025

Instructor: Prof. Yu-Xiang Wang

Last time

- Feature expansion
- Kernel methods
- **Punchline: There is a lot more mileage in your linear learners.**

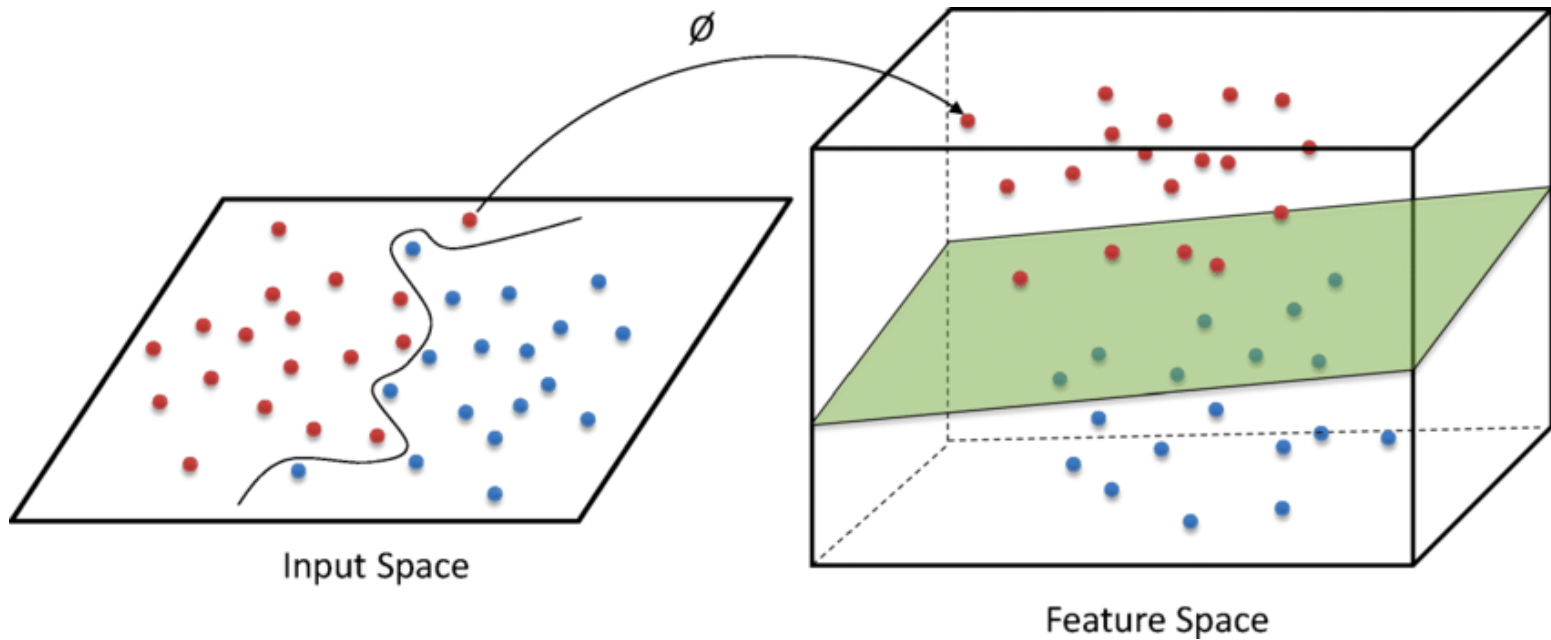
Recap: The idea of feature expansion --- “lifting” a feature vector to a higher-dimensional space.

	\mathcal{X} -space is \mathbb{R}^d	\mathcal{Z} -space is $\mathbb{R}^{\tilde{d}}$
Features	$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$	$\mathbf{z} = \Phi(\mathbf{x}) = \begin{bmatrix} 1 \\ \Phi_1(\mathbf{x}) \\ \vdots \\ \Phi_{\tilde{d}}(\mathbf{x}) \end{bmatrix}$
Data	$\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$	$\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N$
Label	y_1, y_2, \dots, y_N	y_1, y_2, \dots, y_N
Weight	no weights	$\tilde{\mathbf{w}} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{\tilde{d}} \end{bmatrix}$
	Model $g(\mathbf{z}) = \text{sign}(\tilde{\mathbf{w}}^T \Phi(\mathbf{x}))$	

Recap: Examples of feature expansion

- Polynomial expansion
- Decision-stumps (recovers random forest and boosting)
- Radial basis feature expansion
- Generic RKHS with kernel k

Recap: The general idea is that in higher dimensions it is easier for the data to be linearly separable



Recap: Kernel ridge regression

- Ridge regression

$$\begin{aligned}\hat{\theta} &= (X^T X + \lambda I_d)^{-1} X^T \mathbf{y} && \text{(See a cute proof [here](#))} \\ &= X^T (X X^T + \lambda I_n)^{-1} \mathbf{y}\end{aligned}$$

- Prediction: $\langle x, \hat{\theta} \rangle = x^T X^T (X X^T + \lambda I_n)^{-1} \mathbf{y} = \sum_{i=1}^n \langle x, x_i \rangle [(X X^T + \lambda I_n)^{-1} \mathbf{y}]_i$

- Kernel ridge regression

$$\langle x, \hat{\theta} \rangle = [k(x, x_1), \dots, k(x, x_n)] (K + \lambda I_n)^{-1} \mathbf{y}$$

- Observe that we never need to represent the infinite dimensional feature map! Only calls to $k(.,.)$ are needed!

Today

- Kernel SVM
 - And some experiments with kernel SVM
- Feature learning and neural networks
- (Possibly) Unsupervised learning

Recap: Hinge Loss and Soft-Margin Support Vector Machines (with L2-regularization)

$$\min_{\mathbf{w}, b} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i$$

$$s.t. \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \quad \forall i \in [n]$$

$$\xi_i \geq 0 \quad \forall i \in [n]$$

Equivalent to minimizing **Hinge losses**.

$$\min_{\mathbf{w}, b} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \max [1 - y_i(\mathbf{w}^T \mathbf{x} + b), 0]$$

Kernel Support Vector Machine (in the dual)

$$\begin{aligned} & \underset{\alpha \in \mathbb{R}^n}{\text{minimize}} && \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j) - \sum_{i=1}^n \alpha_i \\ & \text{subject to} && 0 \leq \alpha_i \leq C, \quad \forall i = 1, \dots, n, \\ & && \sum_{i=1}^m \alpha_i y_i = 0. \end{aligned}$$

Remark: Note that it only depends on the inner products (i.e., “kernel trick”). The feature expansion can be infinite dimensional.

Implementing kernel SVM in just a few lines with *sklearn* (also on *libsvm* and *liblinear*)

```
from sklearn import svm

clf = svm.SVC(kernel='rbf', gamma=gamma)
clf.fit(X_train, y_train)
ypred = clf.predict(x_new)
```

- How a fitted kernel SVM is making predictions?

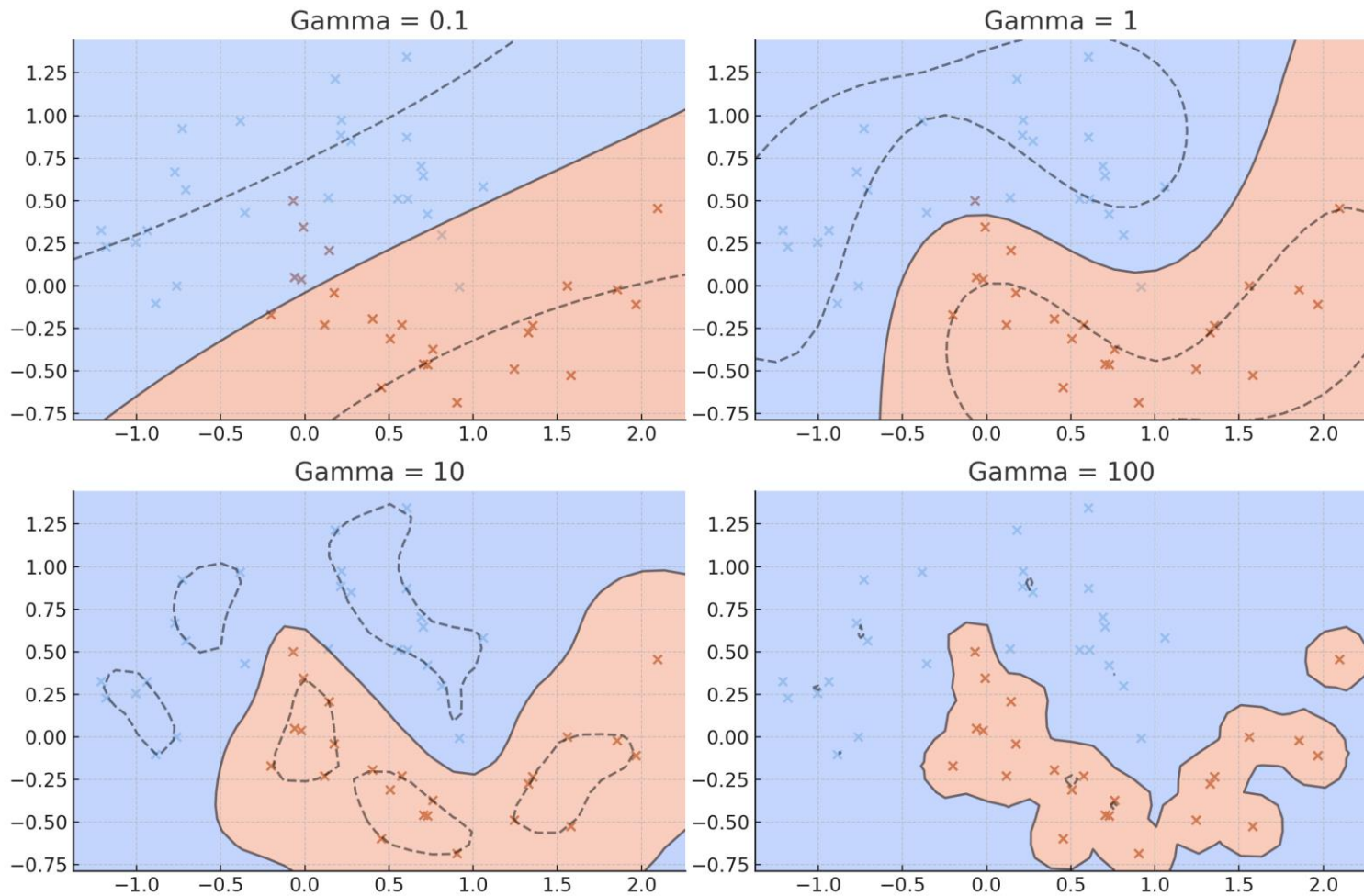
$$\hat{y}(x) = \text{sign} \left(\sum_{i=1}^n \alpha_i y_i k(x, x_i) + b \right) = \text{sign} \left(\left\langle \underbrace{\sum_{i=1}^n \alpha_i y_i k(x_i, \cdot)}_{\text{RKHS}_k}, k(x, \cdot) \right\rangle_{\text{RKHS}_k} + b \right)$$

Linear combination of your neighbors!

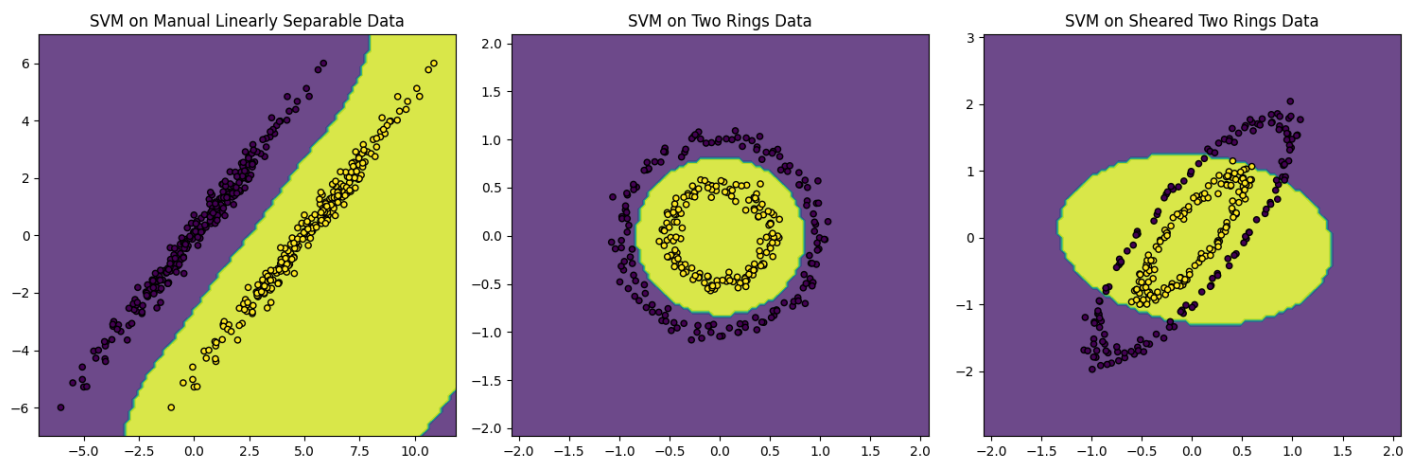
A linear classifier in an infinite dimensional space.
Here are the weights.

Illustration of how a kernel-SVM works as we adjust the kernel bandwidth

Kernel: $k(x, x') = e^{-\gamma \|x - x'\|^2}$ Feature map: $\phi(x) = e^{-\gamma \|x - \cdot\|^2}$



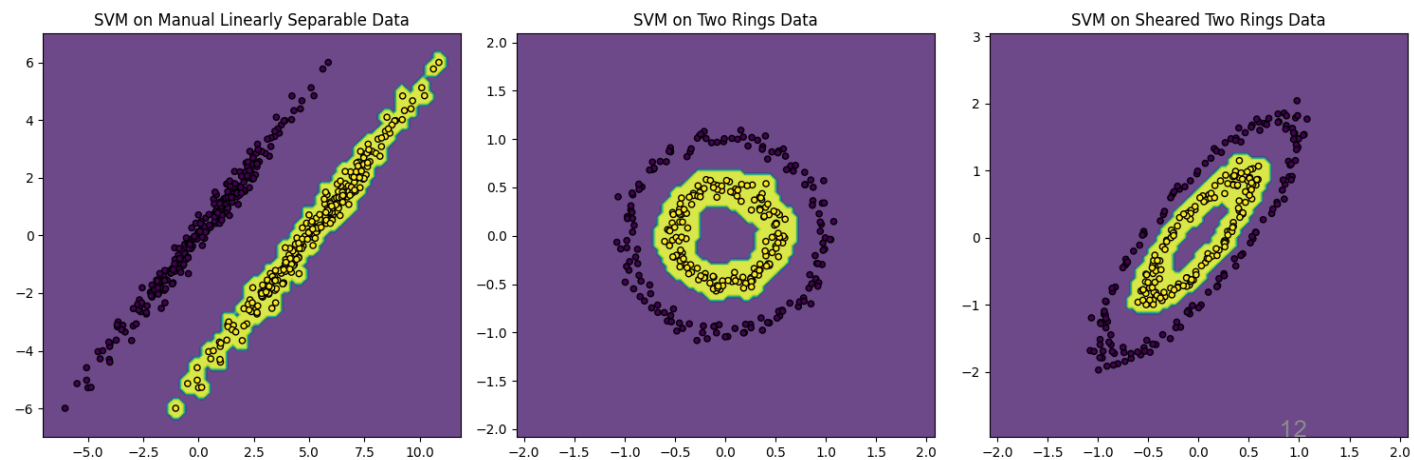
$\gamma = 0.1$



$\gamma = 10$

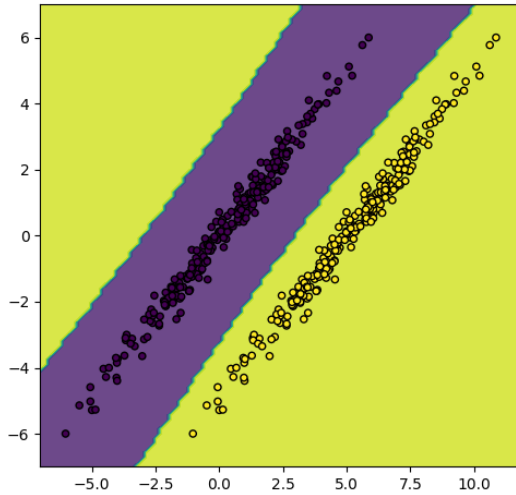


$\gamma = 100$

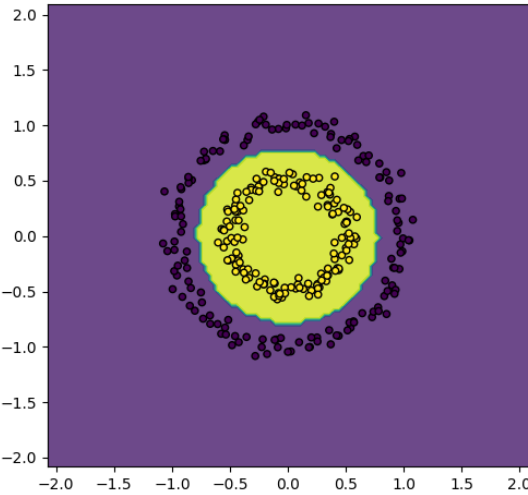


```
svm_clf = SVC(kernel='poly', gamma='auto', degree=2)
```

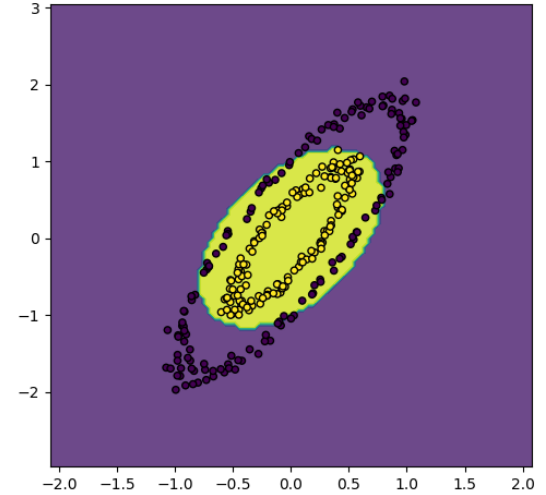
SVM on Manual Linearly Separable Data



SVM on Two Rings Data

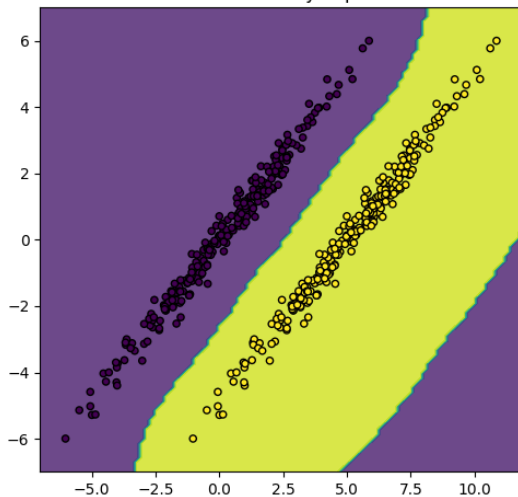


SVM on Sheared Two Rings Data

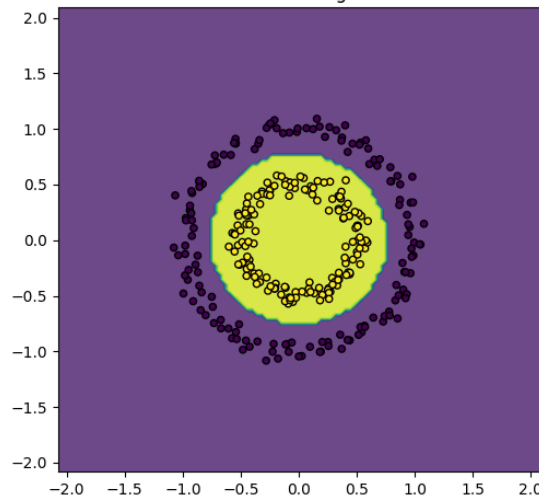


```
svm_clf = SVC(kernel='rbf', gamma='auto')
```

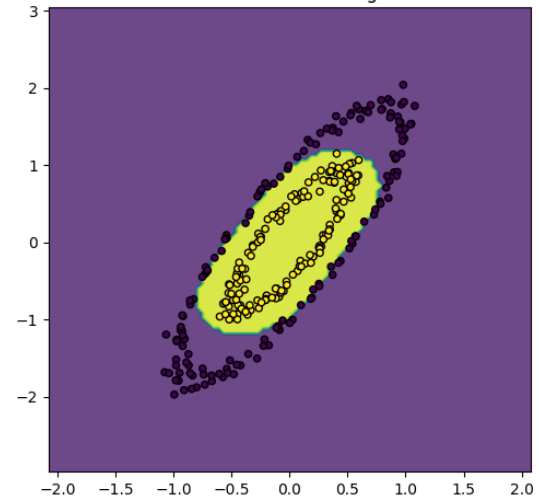
SVM on Manual Linearly Separable Data



SVM on Two Rings Data

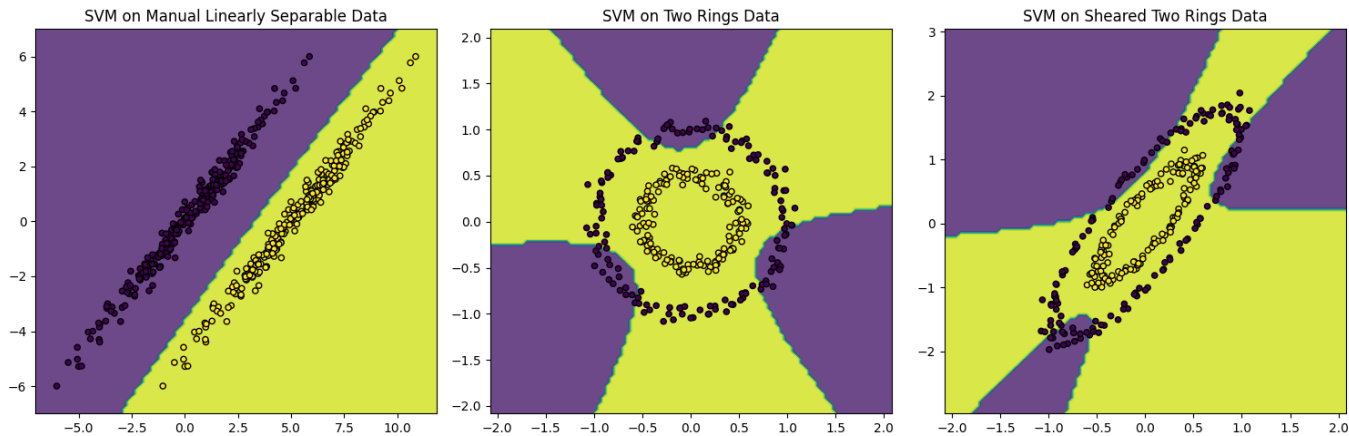


SVM on Sheared Two Rings Data

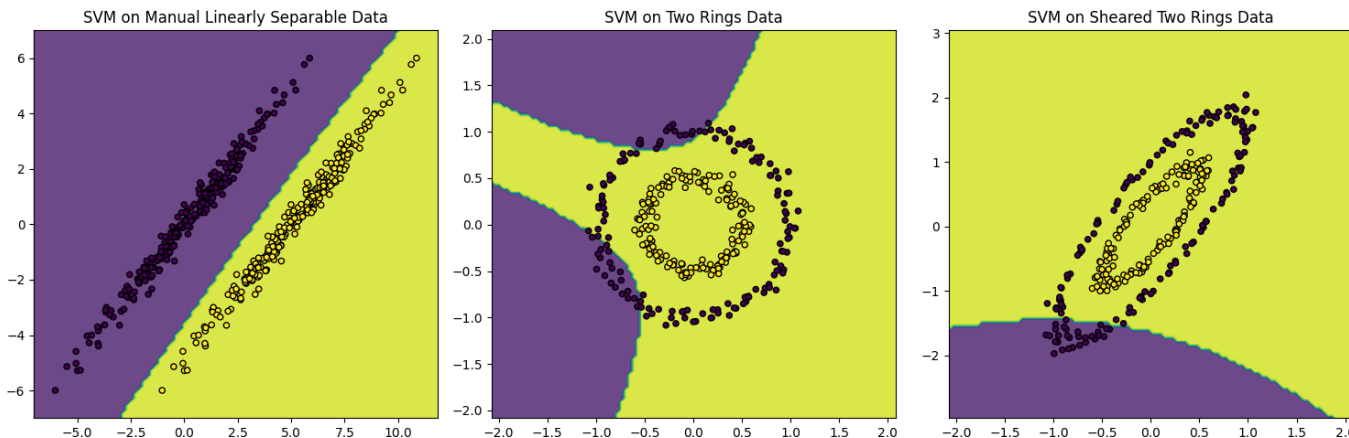


The choice of hyperparameters of these kernels can be delicate!

```
svm_clf = SVC(kernel='poly', gamma=10, degree=3)
```



```
svm_clf = SVC(kernel='poly', gamma='auto', degree=3)
```



Checkpoint: Kernel methods

- They are essentially linear models --- linear in the expanded feature space
- Systematic way to tune the kernel-bandwidth, polynomial order, allows us to reduce “approximation error” and its tradeoff with “generalization error”.
- Drawbacks:
 - Need to specify the kernel
 - Computationally efficient but not scalable ($O(n^3)$)!

Today

- Kernel SVM
 - And some experiments with kernel SVM
- Feature learning and neural networks
- (Possibly) Unsupervised learning

Neural networks: Example: AlexNet (2012)

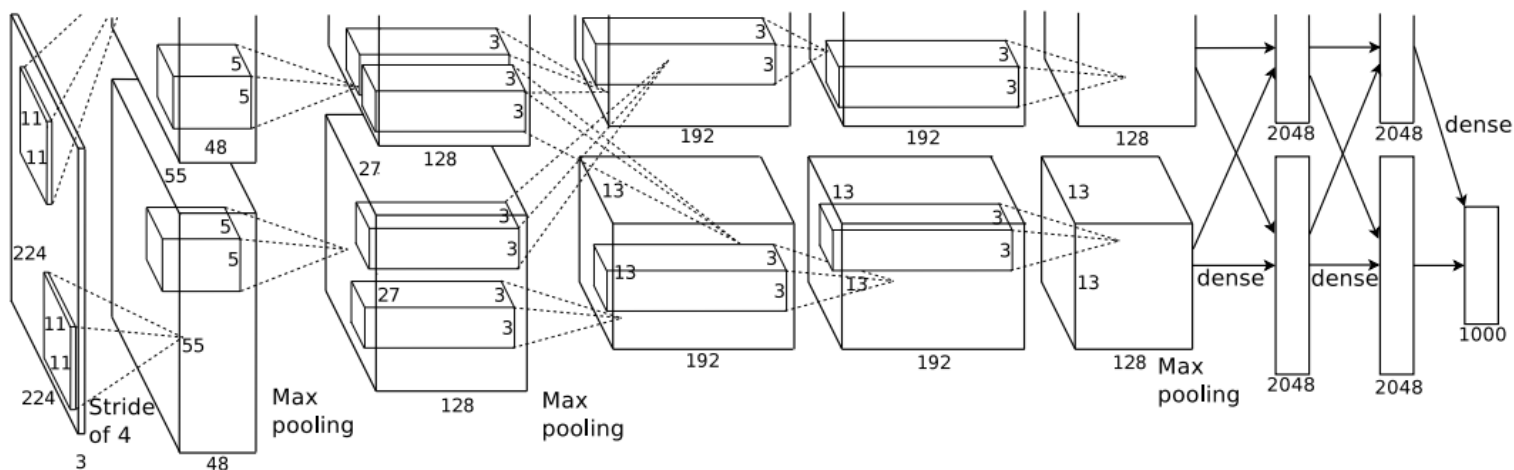


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

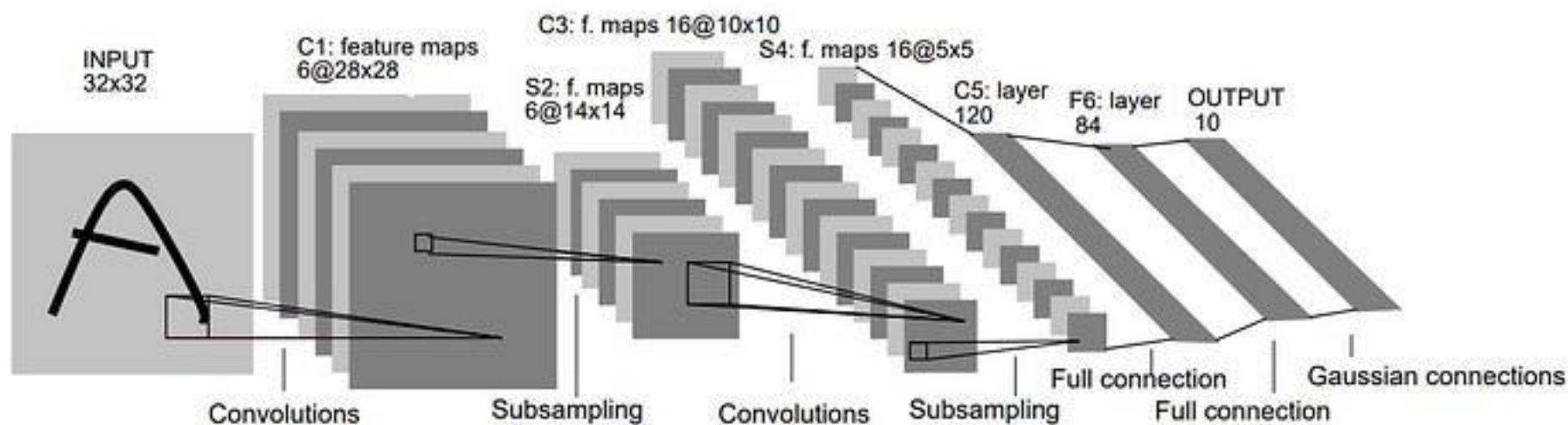
Imagenet classification with deep convolutional neural networks

[A Krizhevsky, I Sutskever... - Advances in neural ..., 2012 - proceedings.neurips.cc](#)

... a large, **deep convolutional neural network** to **classify** the 1.2 million high-resolution images in the **ImageNet** ... The **neural network**, which has 60 million parameters and 650,000 neurons, ...

☆ Save Cite Cited by 122248 Related articles All 111 versions Import into BibTeX

LeNet (1998)



Gradient-based learning applied to document recognition

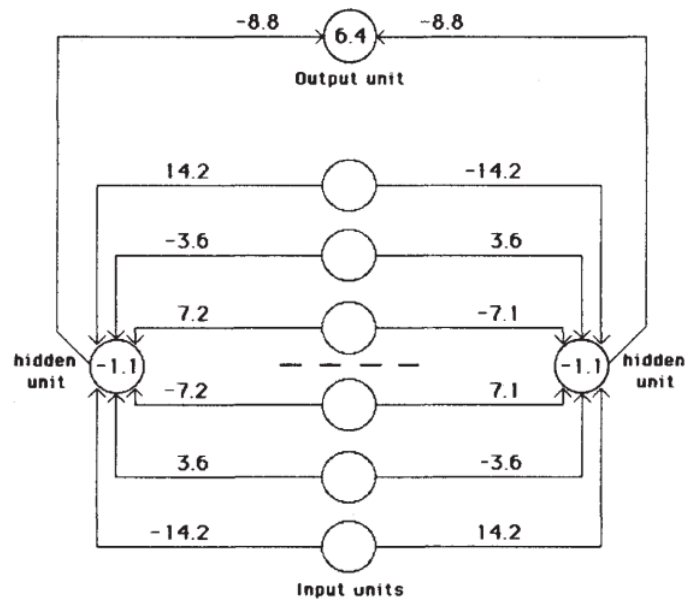
[Y LeCun, L Bottou, Y Bengio...](#) - Proceedings of the ..., 1998 - [ieeexplore.ieee.org](#)

... **gradientbased learning** technique. Given an appropriate network architecture, **gradient-based learning** algorithms can be **used** to ... methods **applied** to handwritten character **recognition** ...

☆ Save [Cite](#) Cited by 59964 [Related articles](#) [All 41 versions](#) [Web of Science: 27156](#) [Import into BibTeX](#)

Rumelhart, Hinton, Williams (1986)

- One layer of a feedforward neural networks



Learning representations by back-propagating errors

DE Rumelhart, [GE Hinton](#), [RJ Williams](#) - nature, 1986 - nature.com

... their states are completely determined by the input vector: they do not **learn representations**.)

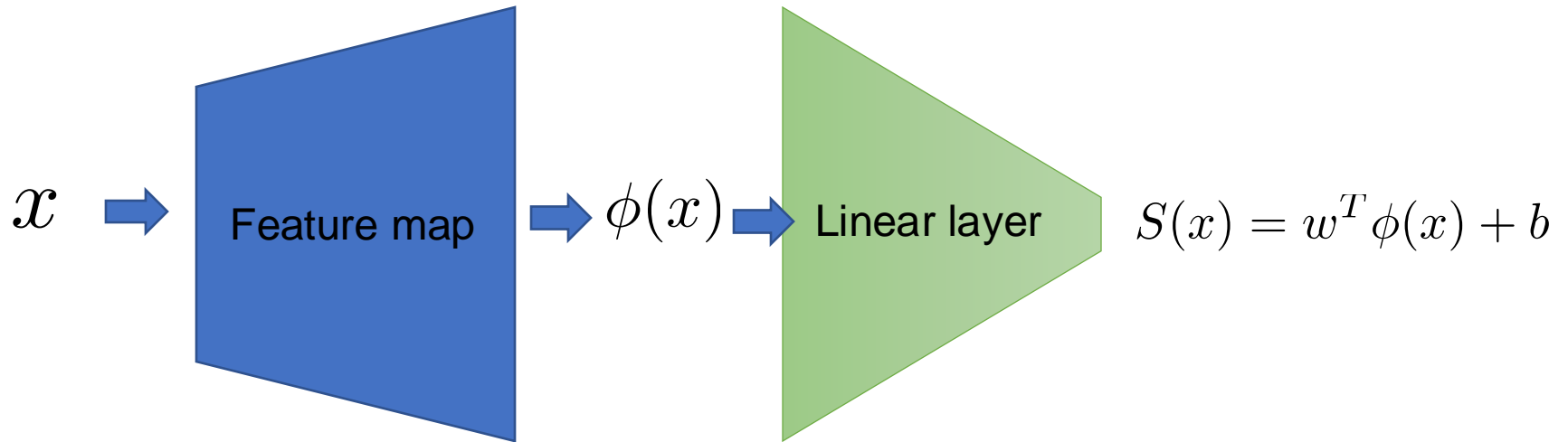
The **learning** procedure must decide under what circumstances the hidden units should be ...

☆ Save [Cite](#) Cited by 35698 [Related articles](#) [All 29 versions](#) [Import into BibTeX](#)

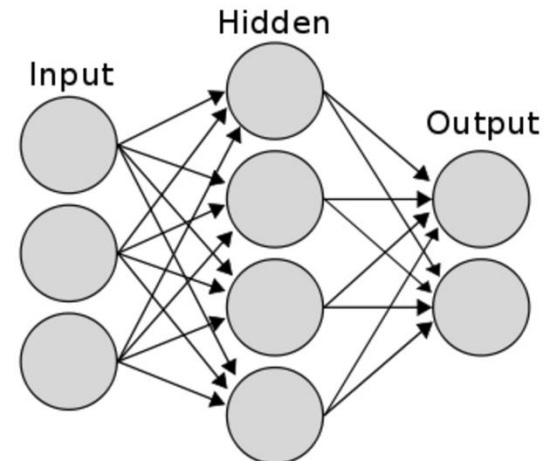
It goes back even further...

- 1943 Pitts and McCulloch: Perceptron model to mimic the brain
- 1956: Rosenblatt's Perceptron Implementation
- 1960s:
 - Ivakhnenko and Lapa: Multi-layer Perceptron (going deeper)
 - Dreyfus: Backpropagation for training (not yet the same as SGD)
 - Amari: Use SGD for training MLPs (separating non-linearly separable patterns)
- 1970s:
 - Fukushima: Convolutional Neural Networks for images
- 1982:
 - Werbos: Modern day backpropagation / SGD

From kernels to neural networks



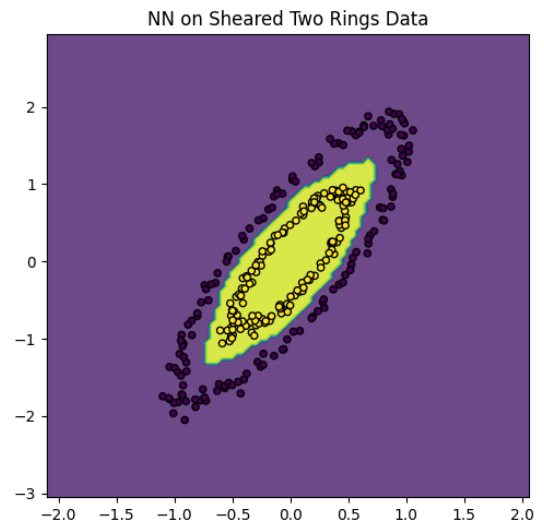
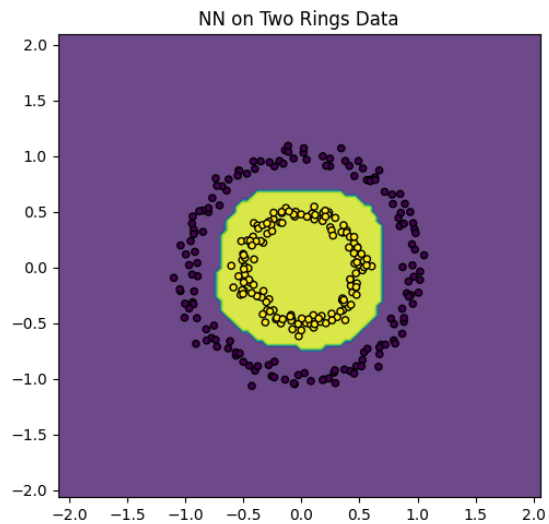
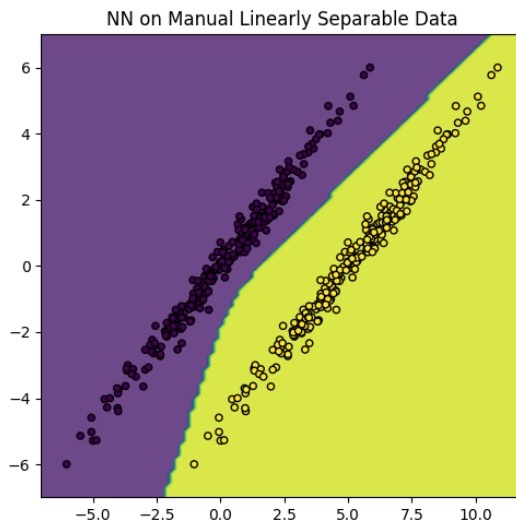
Two-layer neural networks



- Linear neural network: $S(x) = w_2^T (W_1 x + \mathbf{b}_1) + b_2$
 - Still a linear model at the end of the day, so let's add a nonlinearity σ !
- Two-layer MLP: $S(x) = w_2^T \sigma(W_1 x + \mathbf{b}_1) + b_2$
 - Linear model w.r.t. to a learnable feature map
- RBF-kernel: $S(x) = w_2^T \exp(i (W_1 x + \mathbf{b}_1)) + b_2$
 - Kernels are infinite width neural networks with fixed weights
 - By Bochner's theorem, see, e.g., [\[Rahimi and Recht, 2007\]](#)

Results of fitting MLPs on our three examples

```
from sklearn.neural_network import MLPClassifier
hidden = 100
# Neural network classifier
nn_clf = MLPClassifier(hidden_layer_sizes=(hidden,), activation='relu', max_iter=1000)
```



Modern neural networks are very complicated

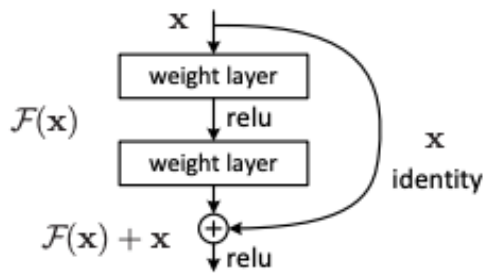
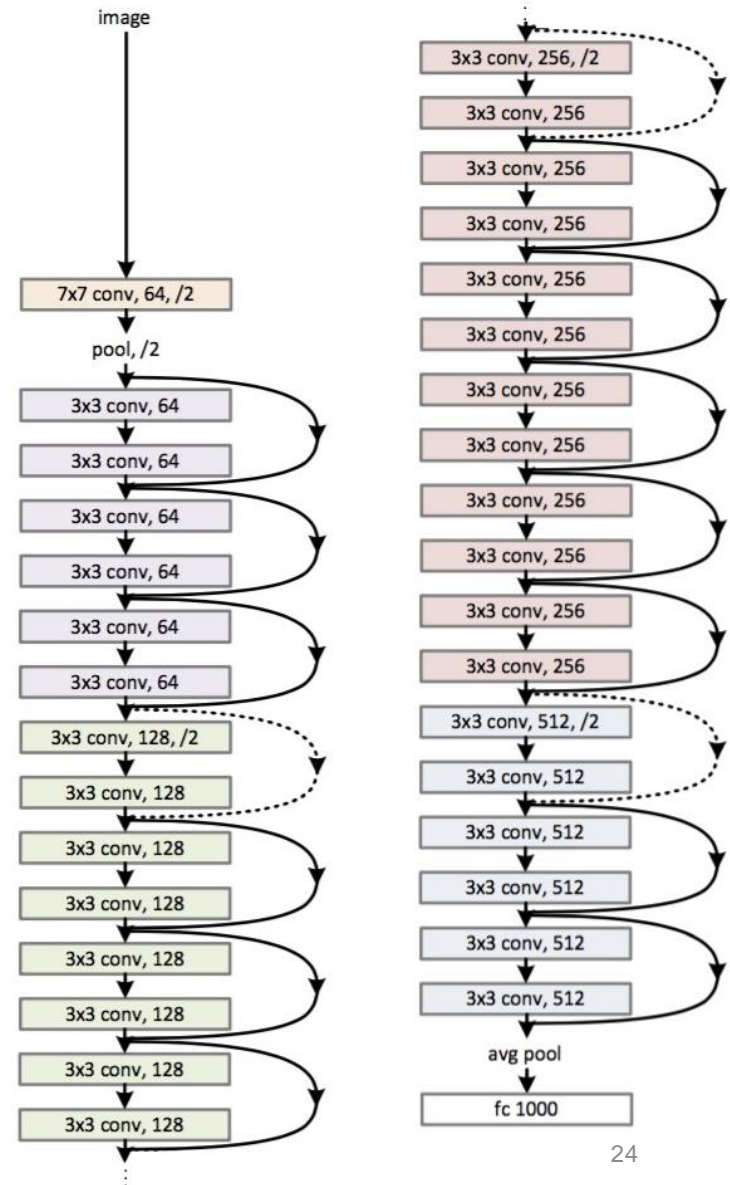


Figure 2. Residual learning: a building block.

34-layer residual



Deep residual learning for image recognition

[K He](#), [X Zhang](#), [S Ren](#), [J Sun](#) - ... and [pattern recognition](#), 2016 - [openaccess.thecvf.com](#)

... **Deeper** neural **networks** are more difficult to train. We present a **residual learning** framework to ease the training of **networks** that are substantially **deeper** than those used previously. ...

☆ Save 📄 Cite Cited by 189280 Related articles All 76 versions Import into BibTeX 📄

Modern neural networks are very complicated

Attention is all you need

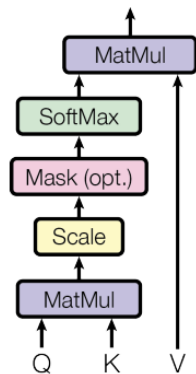
[A Vaswani, N Shazeer, N Parmar...](#) - Advances in neural ..., 2017 - proceedings.neurips.cc

... to attend to **all** positions in the decoder up to and including that position. **We need** to prevent

... **We** implement this inside of scaled dot-product **attention** by masking out (setting to $-\infty$) ...

☆ Save 📄 Cite Cited by 97503 Related articles All 62 versions Import into BibTeX 🔗

Scaled Dot-Product Attention



Multi-Head Attention

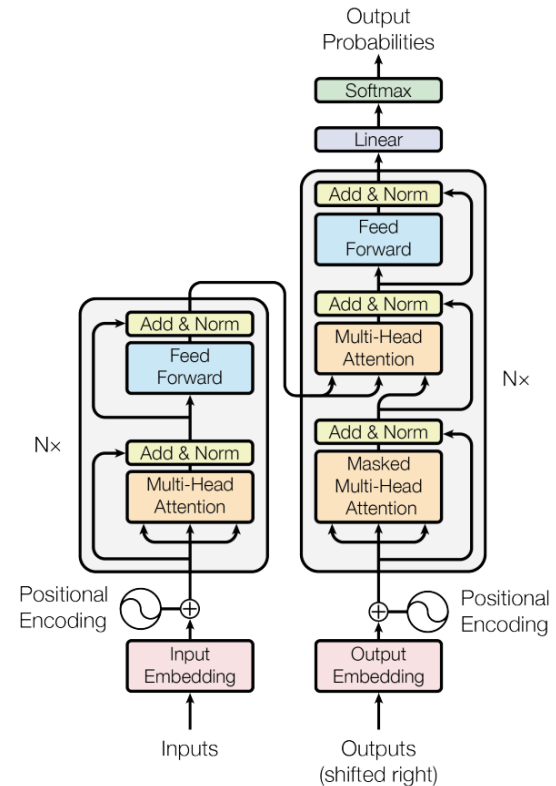
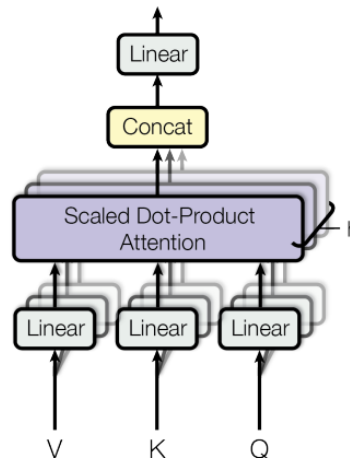


Figure 1: The Transformer - model architecture.

Solution to this? Brute-force computation with autograd and GPUs

- **Autograd:** basically chain rules, can be automated.
 - Design networks such that every block is differentiable.
- **Faster Computation:**
 - Well-packaged Deep Learning Farmework: Write Python wrapper code but running C++ underneath
 - Parallel computing: Numerical linear algebra and GPUs, scientific computing, supercomputing centers.
 - Distributed computing: Cloud computing, Map-Reduce, federated learning
- Popular tools (there are many more of these)



JAX: Autograd and XLA

Demo of autograd and implementation of a neural network

- Implementing a soft-max regression from scratch.

- Implementing a neural network from scratch.

HW4 Q1 gives you an idea of how to use “autograd”.

Learn more in electives focusing on “deep learning”, “computer vision”, “natural language processing”.

Checkpoint: Neural Networks

- Kernels are infinite width neural networks with fixed weights until the last layer, i.e., fixed feature expansion.
- Neural networks learn feature expansion end-to-end.
- Learn autograd and deep learning frameworks, e.g., pytorch.
 - You can create your neural network and start training it within minutes.

Today

- Kernel SVM
 - And some experiments with kernel SVM
- Feature learning and neural networks
- (Possibly) Unsupervised learning

Unsupervised learning

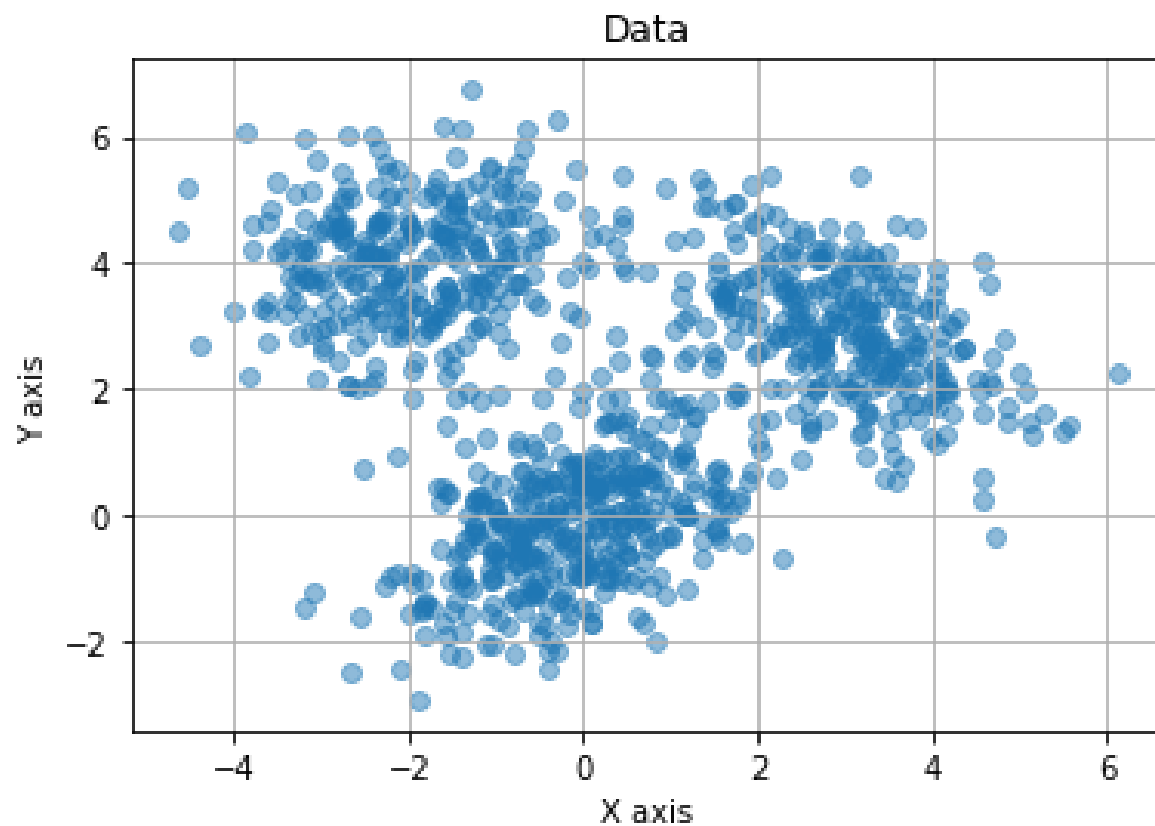
- Input space: \mathcal{X} Images, videos, text, graphs, proteins, programs, etc...
- Output space: None.
- Hypothesis space: \mathcal{H}
- Each hypothesis h is a particular way to summarize the data
- Loss function $\ell : \mathcal{H} \times \mathcal{X} \rightarrow \mathbb{R}$
- Goal of unsupervised learning: Discover simple hypothesis that captures interesting aspects of the data distribution.
 - Often achieved by minimizing the expected loss (i.e., Risk).

The goal of unsupervised learning is to **learn structures in data** without human annotations

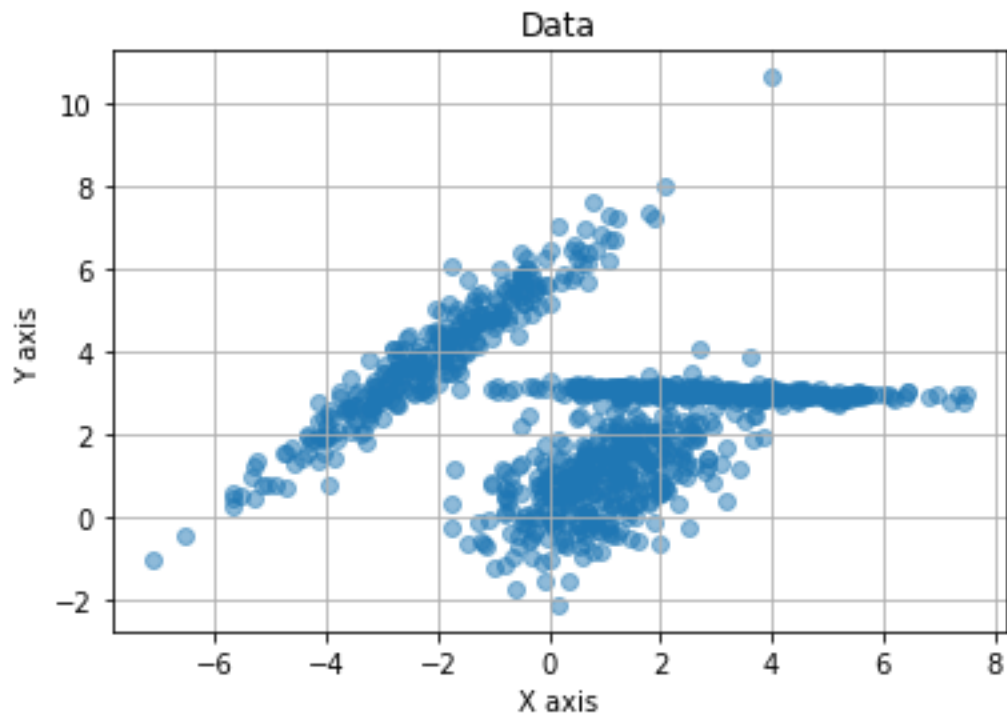


- Discussion: What kind of structures can you see?

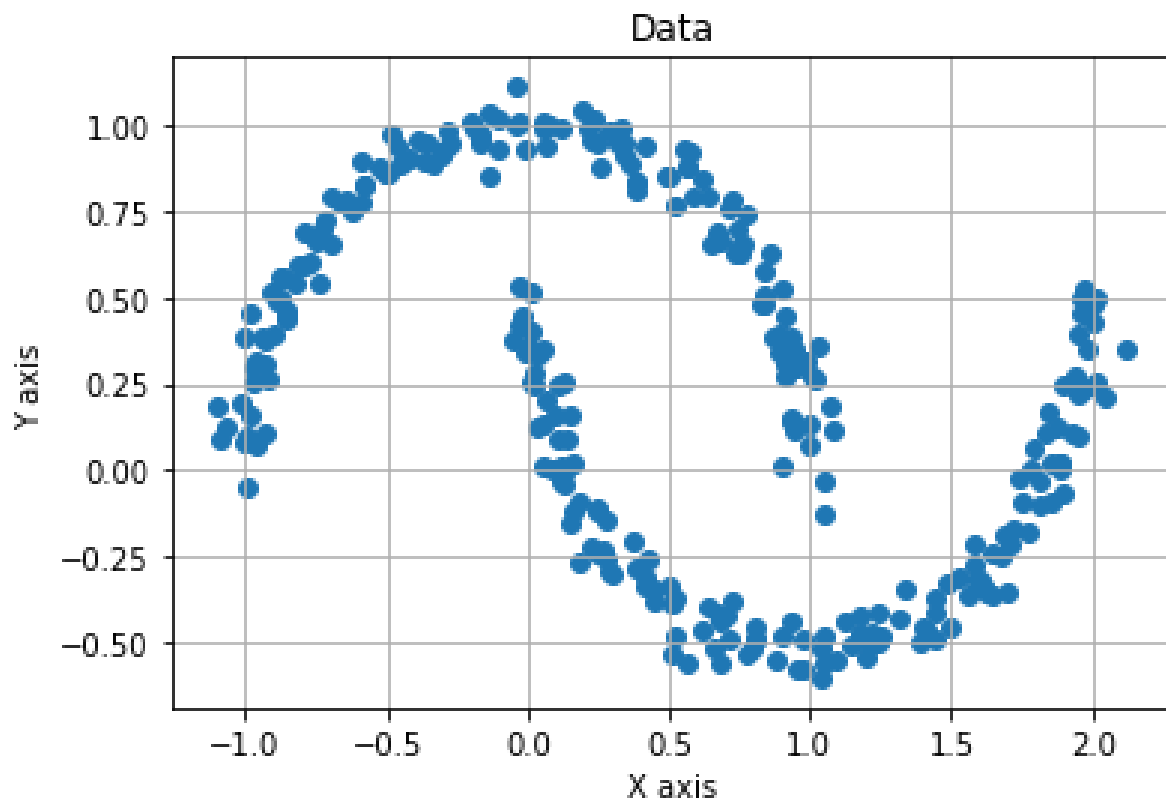
What kind of structures can you see?



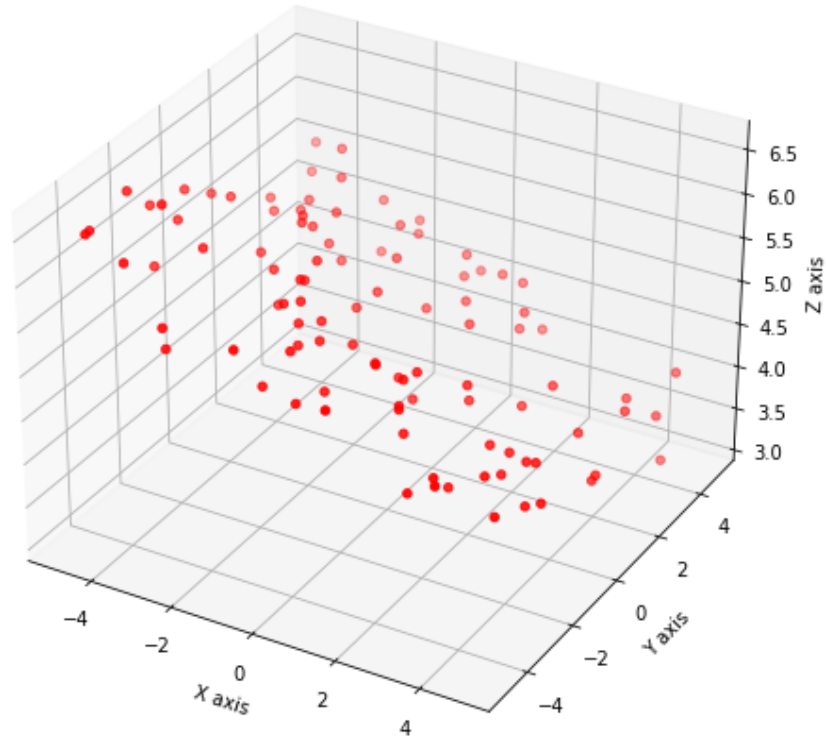
What kind of structures can you see?



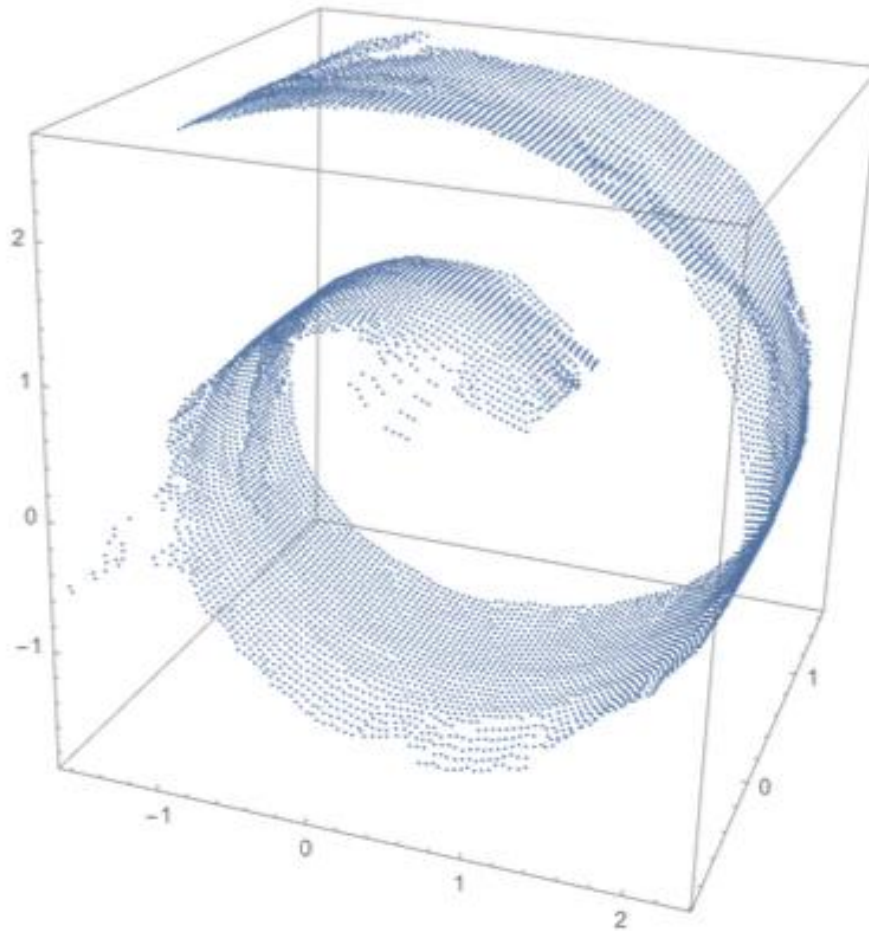
What kind of structures can you see?



What kind of structures can you see?



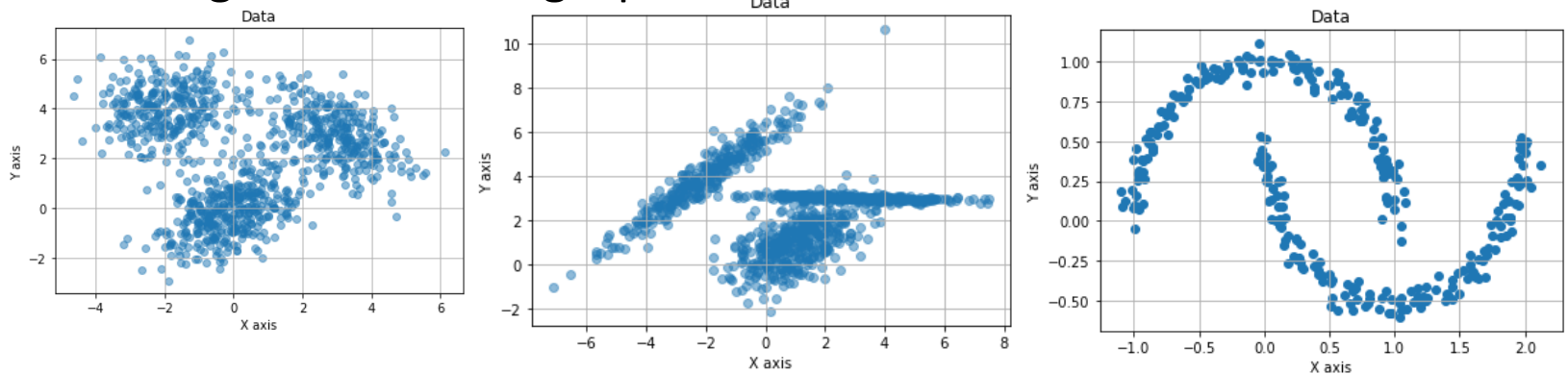
What kind of structures can you see?



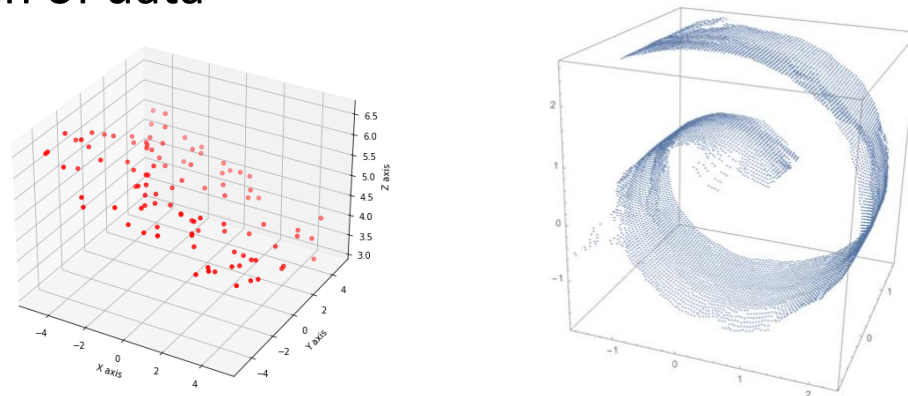
Two broad categories of unsupervised learning

(1) Clustering (2) Dimension reduction

- Clustering aims at finding a partition of the data that makes sense.



- Dimension reduction aims at identifying a more compact representation of data

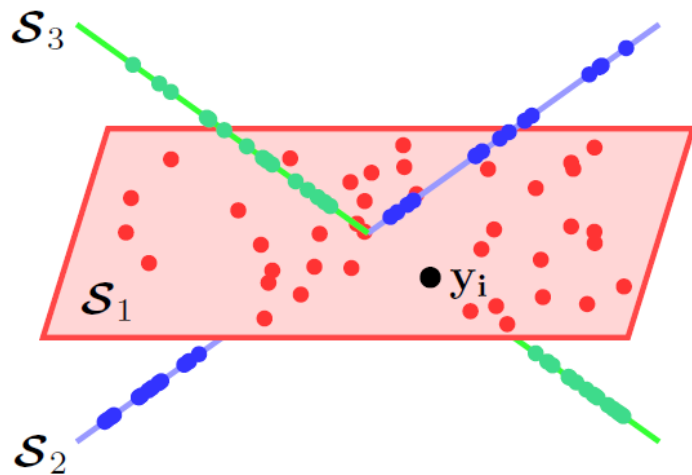


Applications: Indexing text corpus

“Arts”	“Budgets”	“Children”	“Education”
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give \$1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. “Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services,” Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center’s share will be \$200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive \$400,000 each. The Juilliard School, where music and the performing arts are taught, will get \$250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual \$100,000 donation, too.

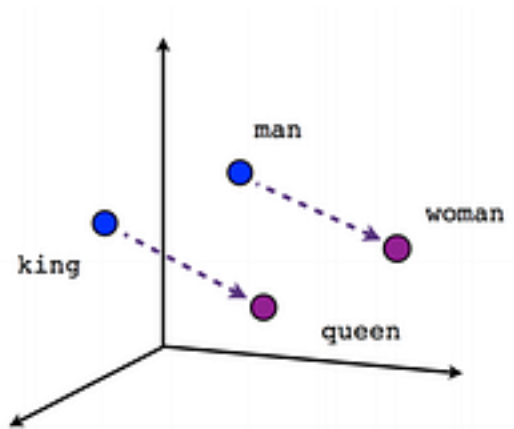
Application: Motion segmentation and subspace clustering



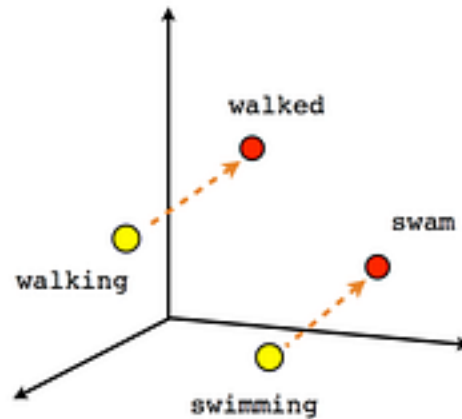
- Turns out that the tracked point trajectories of each rigid moving body captured on a video fall into a 4-dimensional subspace.
- To cluster these points, it suffices to find the subspace membership.

Applications: learn useful vector space representation of language

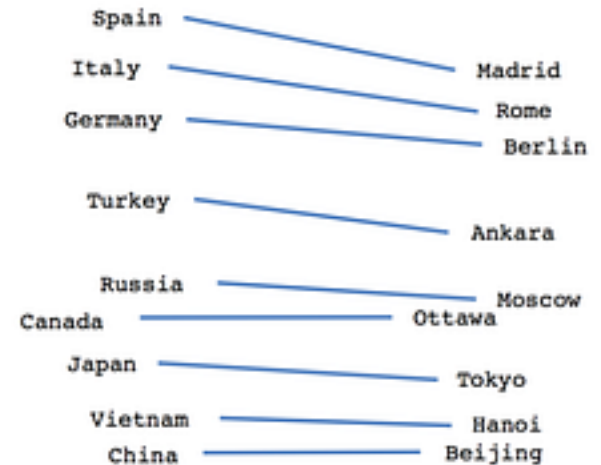
- So you can do algebra on them..



Male-Female



Verb tense



Country-Capital

Source: <https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8>

Application: Image / video compression



100 dpi low JPEG compression



File size:
248K



100 dpi medium JPEG compression



File size:
49K



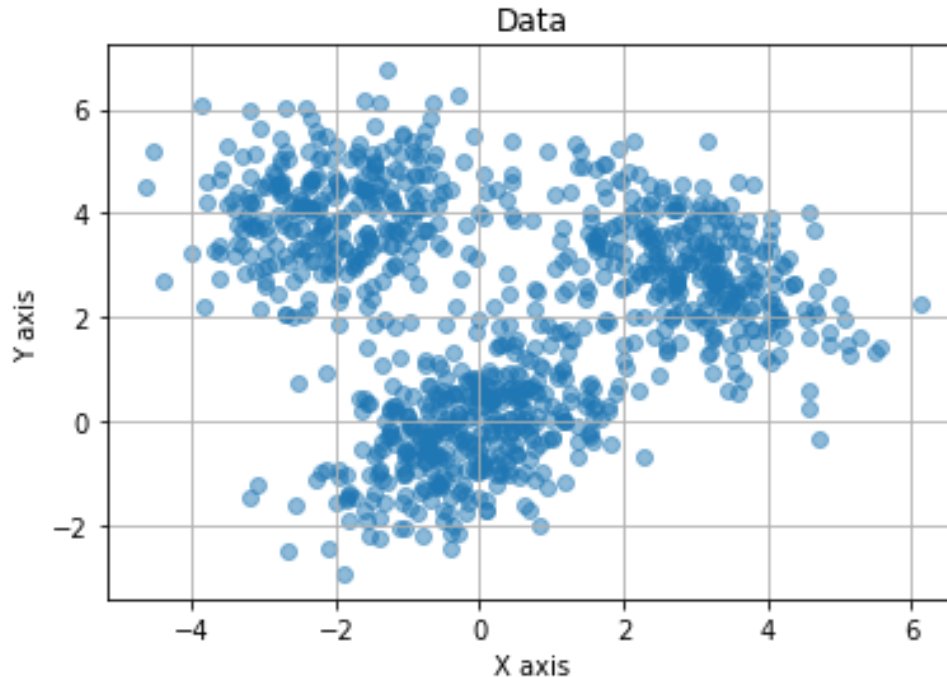
100 dpi high JPEG compression



File size:
22K

Source: <http://preservationtutorial.library.cornell.edu/intro/intro-07.html>

How do you learn the structure you see?



- Come up with a loss function to minimize?
- Come up a probabilistic model that generates the data?

The problem of k-means clustering

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

- Where $\mathbf{S} = \{S_1, S_2, \dots, S_k\}$ is a partition of the dataset $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$,
- And $\boldsymbol{\mu}_i = \frac{1}{|S_i|} \sum_{\mathbf{x} \in S_i} \mathbf{x}$, is called a **centroid** of S_i
- This optimization problem is *NP-complete*, but we have very practical algorithms. [Kmeans++](#) is guaranteed to find a solution within a factor of $O(\log k)$.

K-means clustering with Lloyd's algorithm

K is a hyperparameter

Algorithm $KMeans(D, K)$ – *K*-means clustering using Euclidean distance Dis_2

Input : data $D \subseteq \mathbb{R}^d$; number of clusters $K \in \mathbb{N}$.

Output : K cluster means $\mu_1, \dots, \mu_K \in \mathbb{R}^d$.

randomly initialise K vectors $\mu_1, \dots, \mu_K \in \mathbb{R}^d$;

repeat

 assign each $\mathbf{x} \in D$ to $\operatorname{argmin}_j Dis_2(\mathbf{x}, \mu_j)$; *← 1-Nearest neighbor assignment*

for $j = 1$ to K **do**

$D_j \leftarrow \{\mathbf{x} \in D \mid \mathbf{x} \text{ assigned to cluster } j\}$; *← Partition defined by assignment*

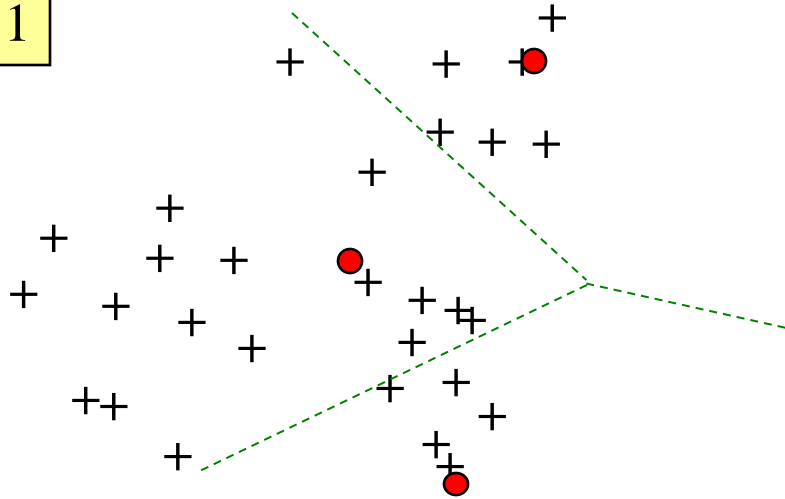
$\mu_j = \frac{1}{|D_j|} \sum_{\mathbf{x} \in D_j} \mathbf{x}$; *← Re-compute the cluster mean*

end

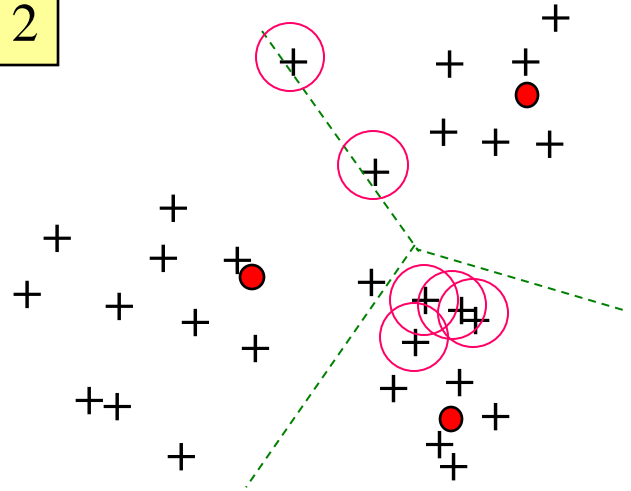
until no change in μ_1, \dots, μ_K ;

return μ_1, \dots, μ_K ;

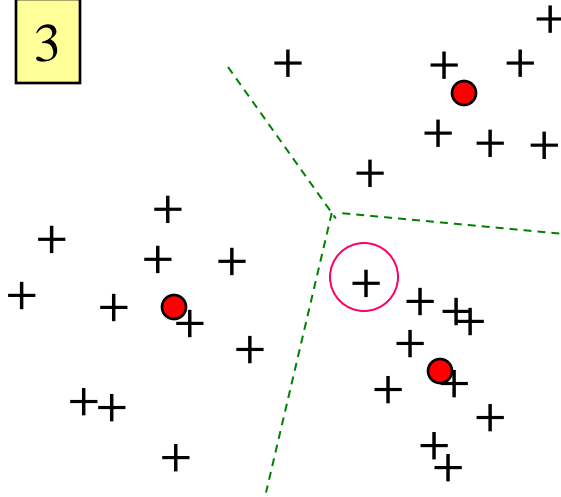
1



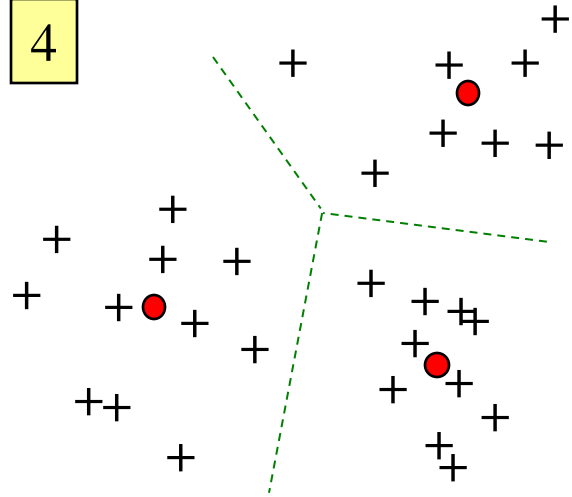
2



3



4

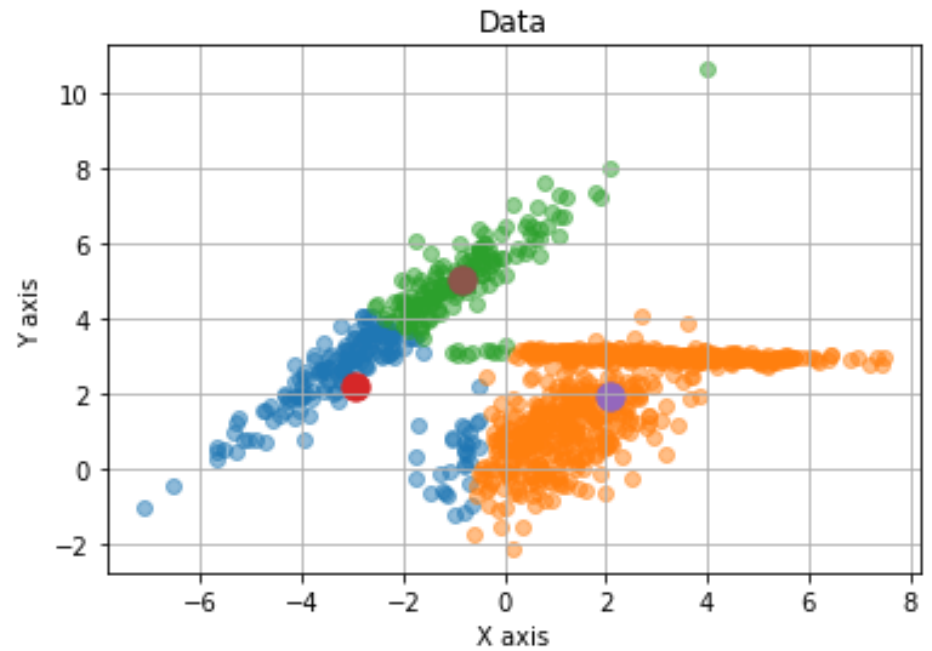
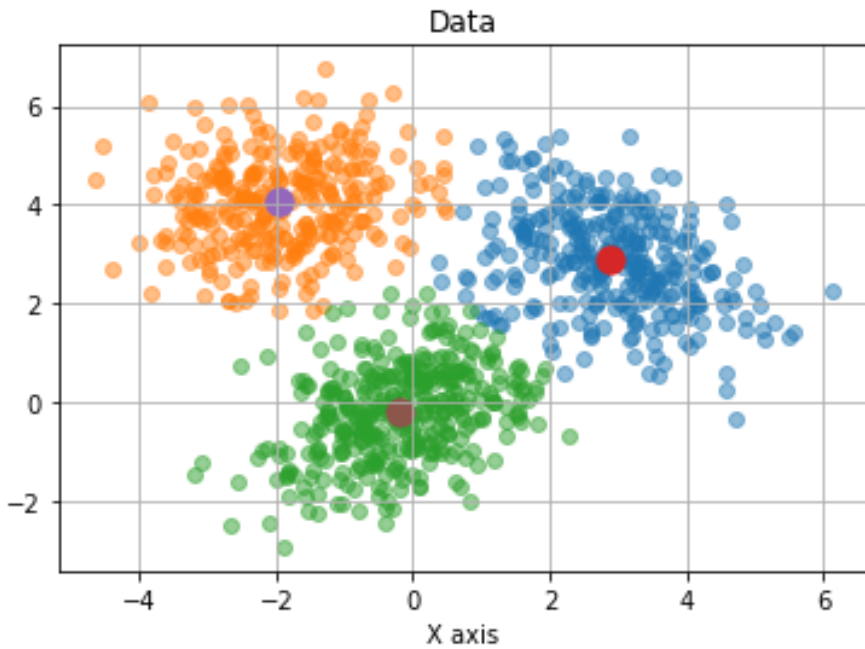


No change – finished!

Nice demo of k-means variants for
you to play with

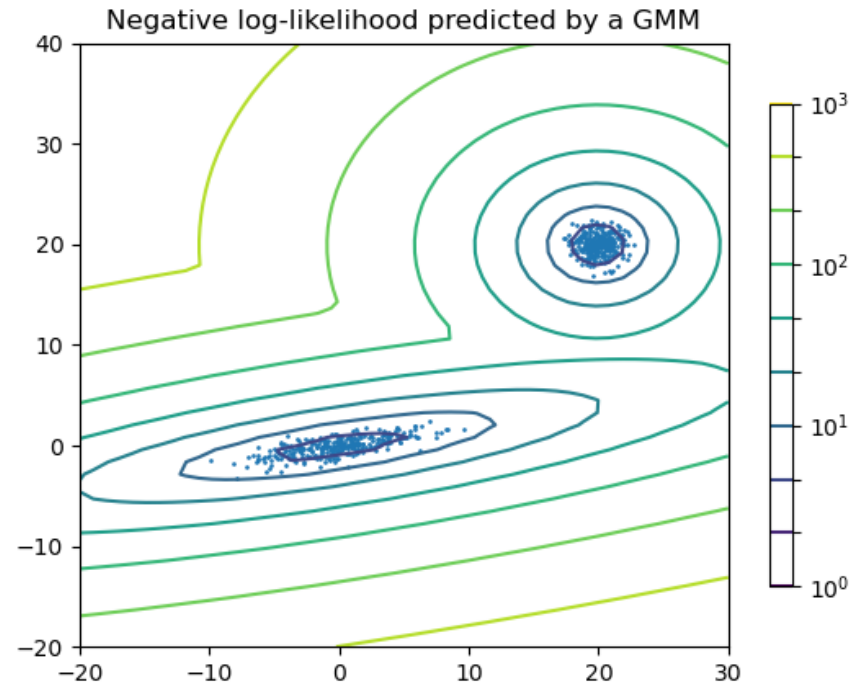
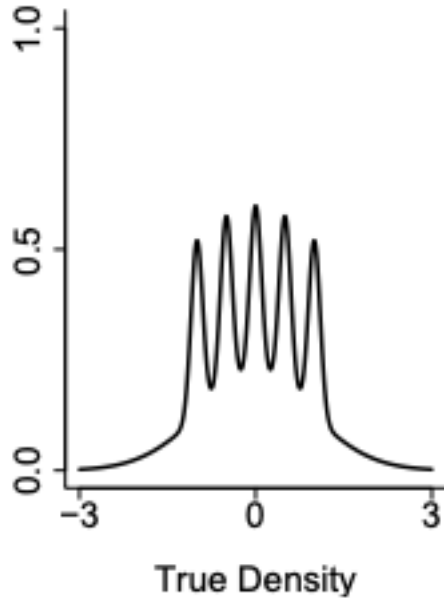
<https://www.naftaliharris.com/blog/visualizing-k-means-clustering/>

K-means on our previous examples



Gaussian mixture models

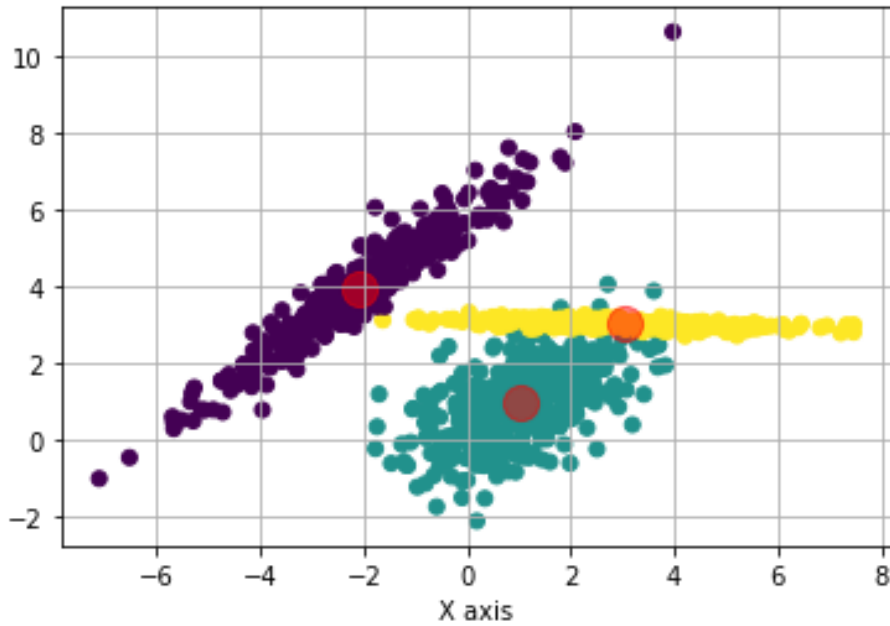
- Assume the data is generated from a mixture of Gaussian distribution



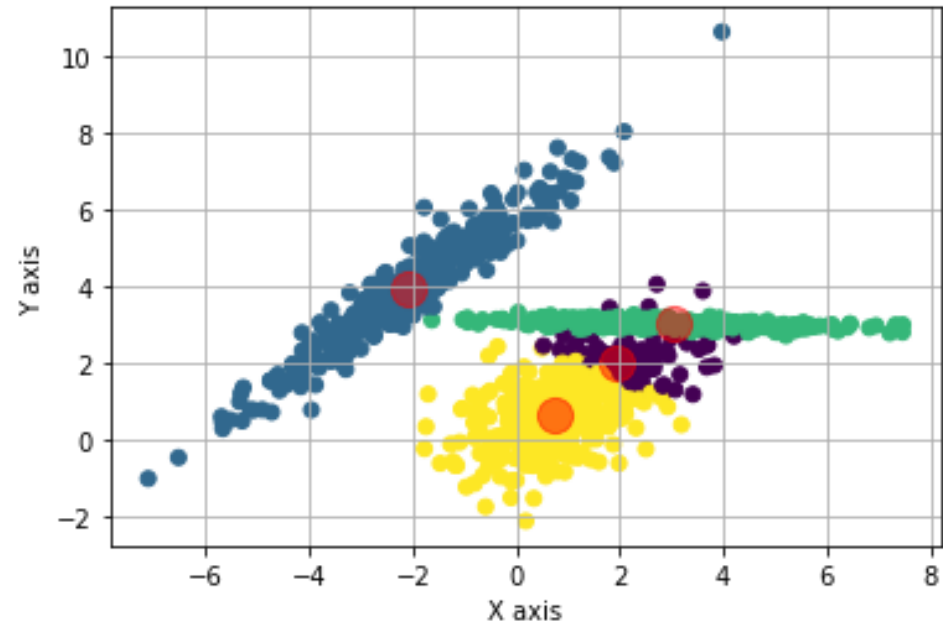
- Data generating process

Fitting Mixture of Gaussian model

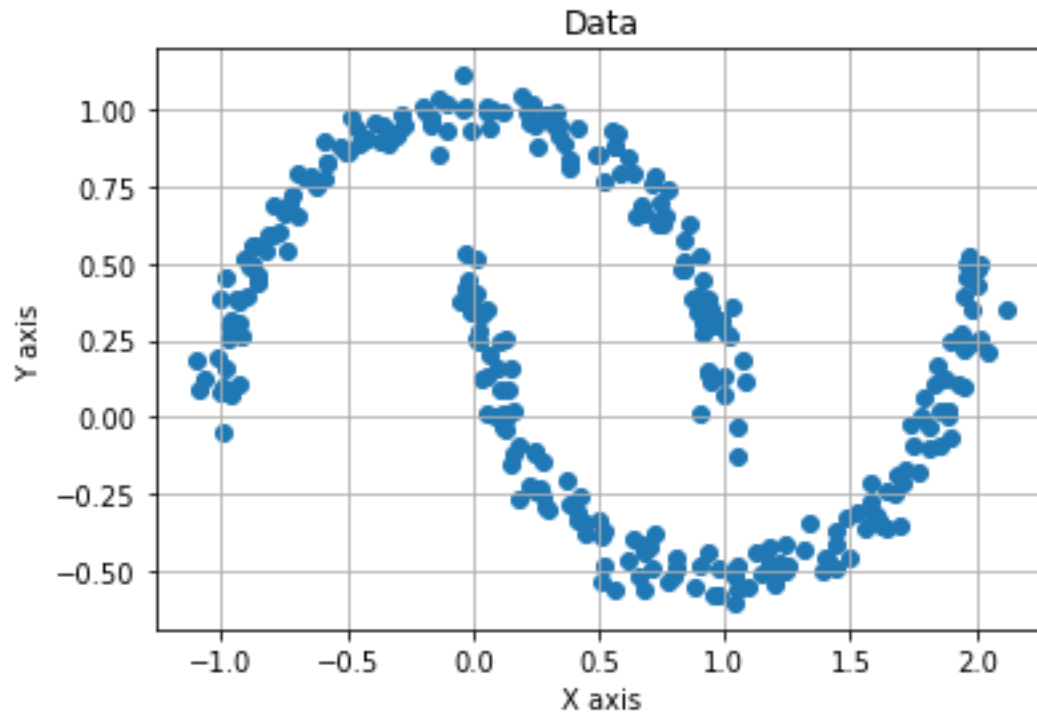
Data



Data



Discussion: what can we do for this?



- Hint: we have learned some useful tricks last week.

Checkpoint: Unsupervised Learning

- K-means problem
 - Lloyd's algorithm for solving k-means problem
 - Which distance function to use?
 - How many cluster centers (centroids) to choose?
 - How to initialize the centroids?
 - **Implement the Lloyd's algorithm in HW4 Q2**
- Gaussian Mixture models
 - A probabilistic model for clustering.
 - Soft assignment of observed data points.
 - When the covariance matrix is small and isotropic it is very similar to k-means.
 - **What's the difference from Gaussian Naïve Bayes model?**

Next lecture

- More on unsupervised learning
- Dimension reduction