@ Clustering

# Unsupervised Learning

* Goal : learning underlying hidden **structure** of the data
* Ex. Clustering, Dimension reduction, Density estimation
* It is often easier to obtain unlabeled data than labeled data. Because Labeling often require human labor(cost)

# Clustering

* Make subgroups by subjective criteria
* Hierarchical clustering: Create a hierarchical decomposition of the set of objects using some criterion (bottom-up)
* Partitional clustering : Construct various partitions and then evaluate them by some criterion (top-down)

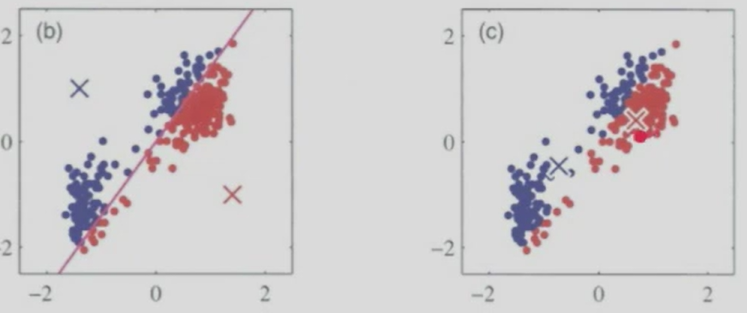
# Hierarchical clustering

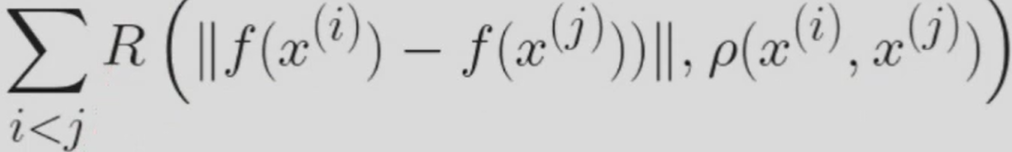
* level 0 : no cluster (singleton)
* for level=1 to L : Estimate group’s distance between cluster and merger clusters

P) Closet pair (single-link for different cluster), farthest pair (hard to connect far cluster)

# K-means algorithm

1. Pick K random points as cluster centers
2. Change the cluster center to the average of its assigned points
3. Iteration until average center is not moved



# Choosing the number of clusters

* Number of subgroup is meaningful for cost function
* Elbow method : Use SSE(Sum of Squared Errors) and select Elbow point
* Silhouette method : Compare ‘silhouette value(Density, Separation)

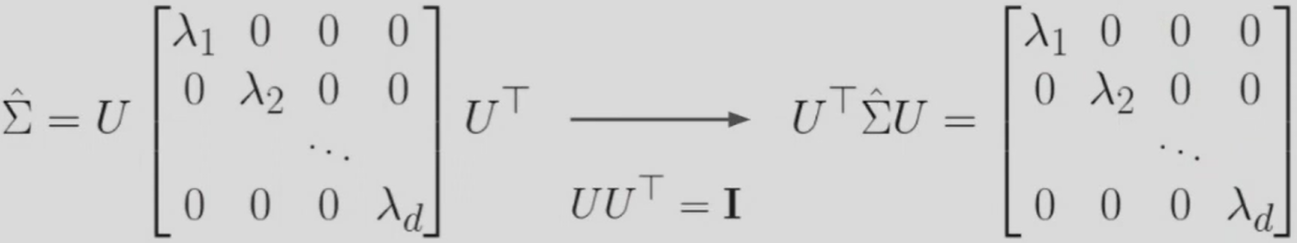
@ Dimension Reduction

# Manifold learning

* In reality, most data points are found in a subspace. (Empty space is majority)
* Manifold learning focuses on finding fundamental subspace that the data is observed
* MDS(Multidimensional Scaling) : Use similarity(distance) for minimizing the distortion by the embedding

# PCA (Principal Component Analysis)

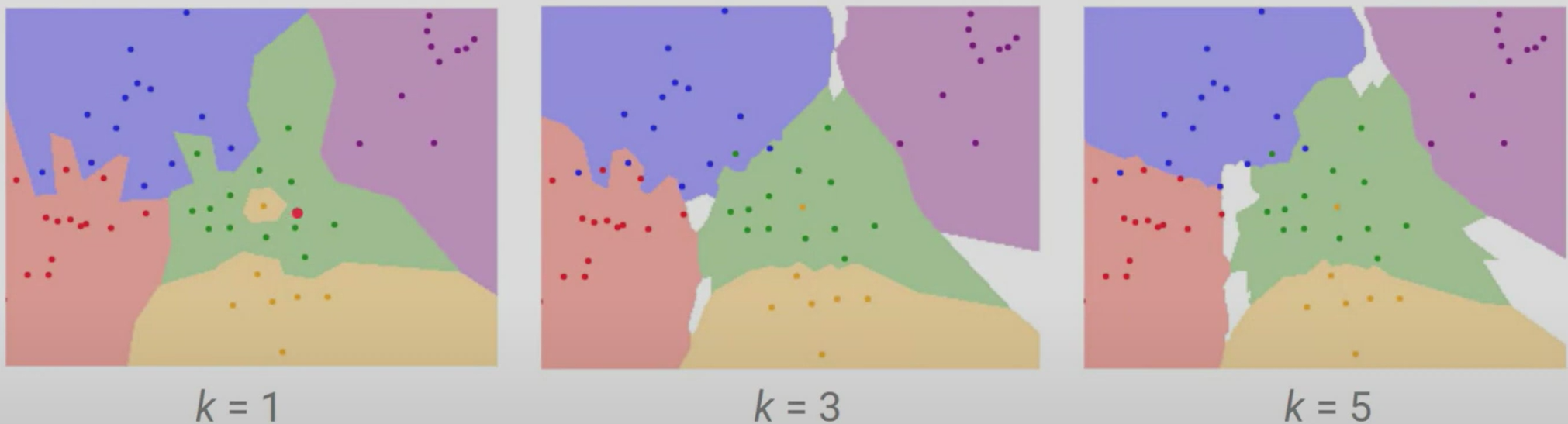
* Goal : Preserve the amount of info(var) in the original data using the reduced dimensionality
* Larger var means a large amount of info like axis, distribution.

1. Set center zero and normalize the data
2. Estimate the co-var matrix
3. Perform the **EVD** and order them by eigenvalues in decreasing order
4. If we choose the eigenvectors and discard the rest, we get a k-dimensional space such that original data loses least amount of info

@ Nearest Neighbors & Softmax Classifiers

# Nearest Neighbors

* Classification comparing each point with near neighbors.
* No need to training : Input data makes classification theirselves
* Prediction uses all data, so time complexity is **O(N)**
* lower k causes overfitting, and higher k causes equal area : N(neighbors) ares same

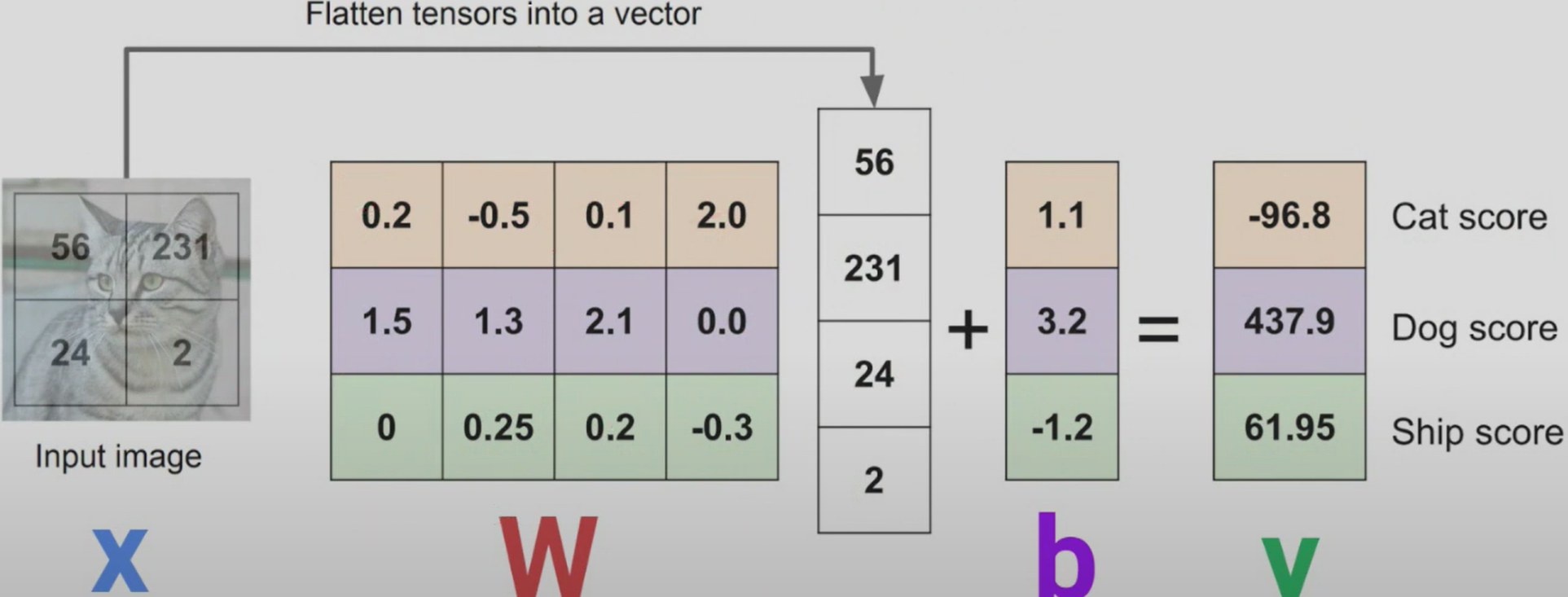


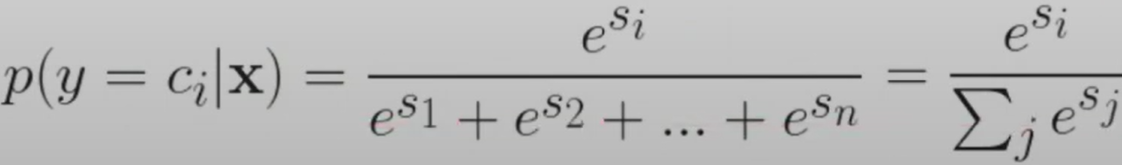
# P) Issues of Nearest Neighbors Classifiers

* DIstance matrices on pixels are not always informative. : Noise, unbalance of data distribution, no related features
* Computation cost : There is no trained model, so computation is always needed.
* Curse of dimension : High dimension data makes **sparse matrix**, and it makes computation much harder

# S) Linear Classifier

* Parametric approach : Set mapped function : input(X), label score(y)
* Weights are computed for classification
* Learn the **template** from training data > Performing the template matching with a new example(test data)
* Similar to kNN in that both compare distances, but different from kNN in that linear classifier compares only to K classes, while kNN does to N training examples



# Softmax Classifier

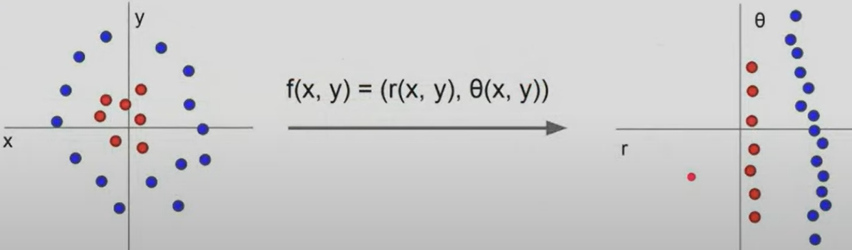
* Use expansion of sigmoid function
* Output is probability for multi-classification
* Cross Entropy(loss func) = −∑(​yi)log(pi) (yi=encoded value, pi=probability for class)

# Kullback-Leibler(KL) Divergence

@ Neural Network and Backpropagation

# Issues with Linear Classifiers

* P) Geometrically : :Linear classifiers can only draw linear decision boundaries
* S) **Extract** some features and make linearly **separable** (Polar Coordinates)



* Examples in image features : Color histogram(distribution),