Machine Learning:

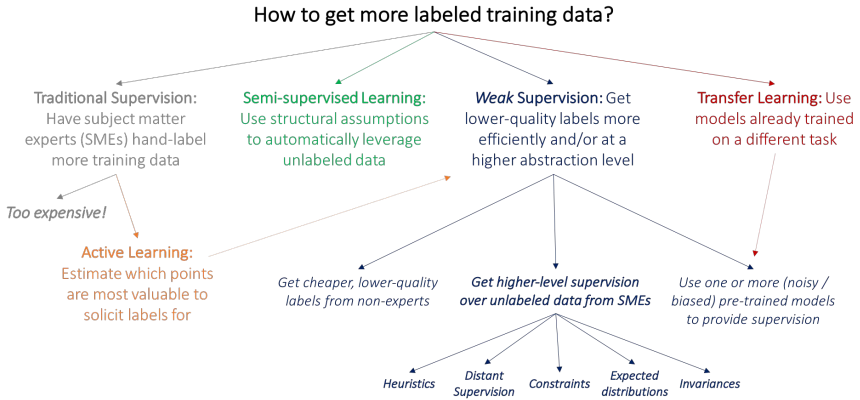
Introduction to Unsupervised Machine Learning and Autoencoders

Schedule for This Part

- Introduction to Machine Learning, Decision Trees
- Introduction to Deep Learning, Convolutional Neural Networks
- **Unsupervised Machine Learning, Autoencoders:**
  - Clustering
  - Dimensionality Reduction
  - Autoencoders
  - Variational Autoencoders
- Introduction to Graph Neural Networks

Unsupervised Learning

- Labels might be imperfect, have high cost to acquire or even don't exist.



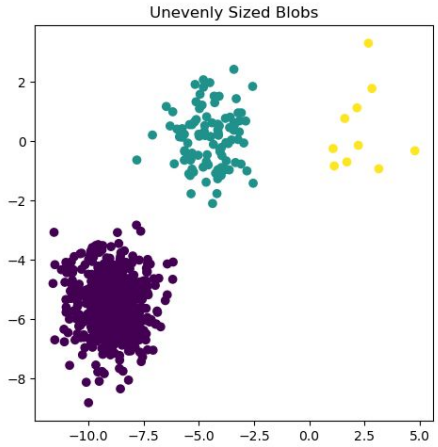
Credit

- Can we still train models without labels?

Yes, we can learn the underlying structure of the data: anomaly/novelty detection, recommender systems, clustering, dimensionality reduction.

Clustering

- Grouping similar data points together based on a similarity or distance measure

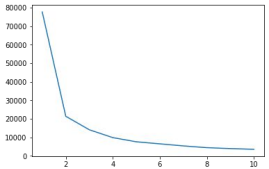


Clustering: *k*-Means

Algorithm 1 *k*-means algorithm

1: Specify the number k of clusters to assign.
2: Randomly initialize k centroids.
3: **repeat**
4: **expectation:** Assign each point to its closest centroid.
5: **maximization:** Compute the new centroid (mean) of each cluster.
6: **until** The centroid positions do not change.

- Requires the number of clusters to be specified.
- Elbow method can be used for finding *k*.



- It scales well to large number of samples.
- Works well with convex and isotropic clusters but not with irregular shapes.

Clustering: DBSCAN (Paper)

- Density-based spatial clustering of applications with noise (DBSCAN).
- Assumes clusters are areas of high density separated by areas of low density, i.e. can deal with non-convex clusters.
- Density-based spatial clustering of applications with noise:
  - Find the points in the neighborhood of every point;
  - Identify *core samples*, i.e. samples in high density areas;
  - Assign each non-core sample to a nearby cluster if within the distance.
- Insensitive to outliers: outliers don't belong to a cluster.
- More flexible cluster shapes.

Dimensionality Reduction

- Working in high-dimensional spaces can be undesirable for many reasons: redundancy of features, curse of dimensionality, challenges with visualization.
- It may be a good idea to reduce number of features.
- Simply removing features is not a good idea: information loss.
- Dimensionality reduction transforms original data from a high-dimensional space into a low-dimensional space ideally retaining meaningful properties of the original data.
- It helps in data compression (reduced storage space), removal of redundant features, noise reduction, data visualization.

Dimensionality Reduction: PCA

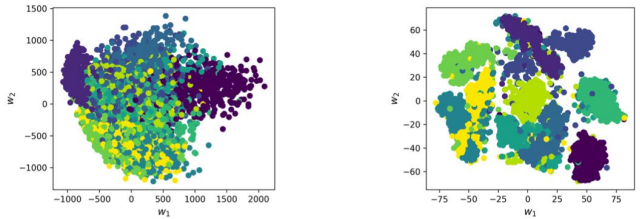
- Principal Component Analysis is likely the most popular dimensionality reduction method.
- A linear mapping of the data to a lower-dimensional space with maximized variance.
- Lossy compression - some data may be lost in the process.
- Hyperparameter: number of final dimensions.
- Each direction is linearly uncorrelated.
- Find a projection matrix **V** that minimizes reconstruction error between original data and the reduced space (inversely maximizes variance of the data).

$$\underset{\mathbf{V}^T\mathbf{V}=\mathbf{I}_k}{\operatorname{argmin}} \left\|\mathbf{X}-\mathbf{X}\mathbf{V}\mathbf{V}^T\right\|_F^2$$

- Typically solved with eigenvalue decomposition:
  - Construct the covariance matrix of the data.
  - Compute the eigenvectors.
  - Eigenvectors corresponding to the largest eigenvalues are used to reconstruct a most of variance of the original data.

Dimensionality Reduction: tSNE

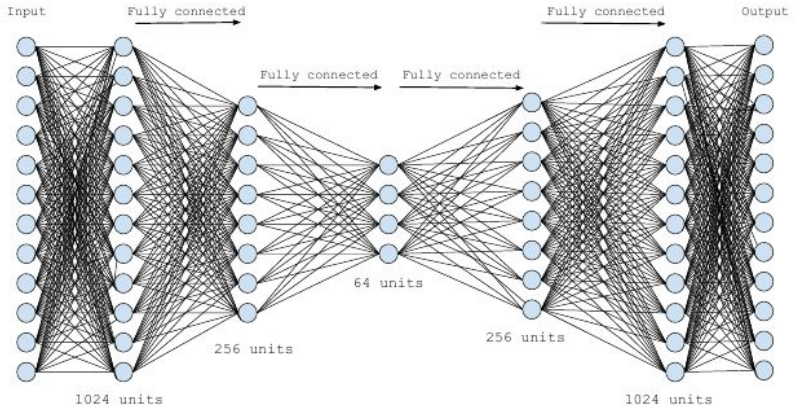
- t-distributed stochastic neighbor embedding is a method for visualizing high-dimensional data and exploratory data analysis
- Preserves small pairwise distances, i.e. local similarities (PCA preserves large pairwise distance)
- It is a nonlinear dimensionality reduction technique:
  - Measure the similarity between pairs of high-dimensional points as probability by centering a Gaussian over each point. Similar objects are assigned a higher probability.
  - Define a similar probability using Student t-distribution in the low-dimensional map.
  - Minimizes the Kullback–Leibler divergence (KL divergence) between the two.



Autoencoders

- A deep autoencoder is composed of two neural networks:
  - encoder *E* that takes an input and maps it to a usually low-dimensional representation
  - decoder *D* that tries to reconstruct the original input from the representation vector:

$$\hat{x} = D(E(x)) \text{ where } \hat{x} \sim x.$$



Autoencoders

- A deep autoencoder is composed of two neural networks:
  - encoder *E* that takes an input and maps it to a usually low-dimensional representation
  - decoder *D* that tries to reconstruct the original input from the representation vector:

$$\hat{x} = D(E(x)) \text{ where } \hat{x} \sim x.$$

- Autoencoders are parametric maps from inputs to their representations (code).
- Trained to perform an approximate identity mapping between their input and output layers.
- Simple autoencoder with just a linear activation function will mirror the PCA algorithm.
- Autoencoders enable both linear or non-linear transformations.

Autoencoders: Applications

- Data compression.
- Input denoising.

- Non-linear dimensionality reduction.
- Anomaly detection.

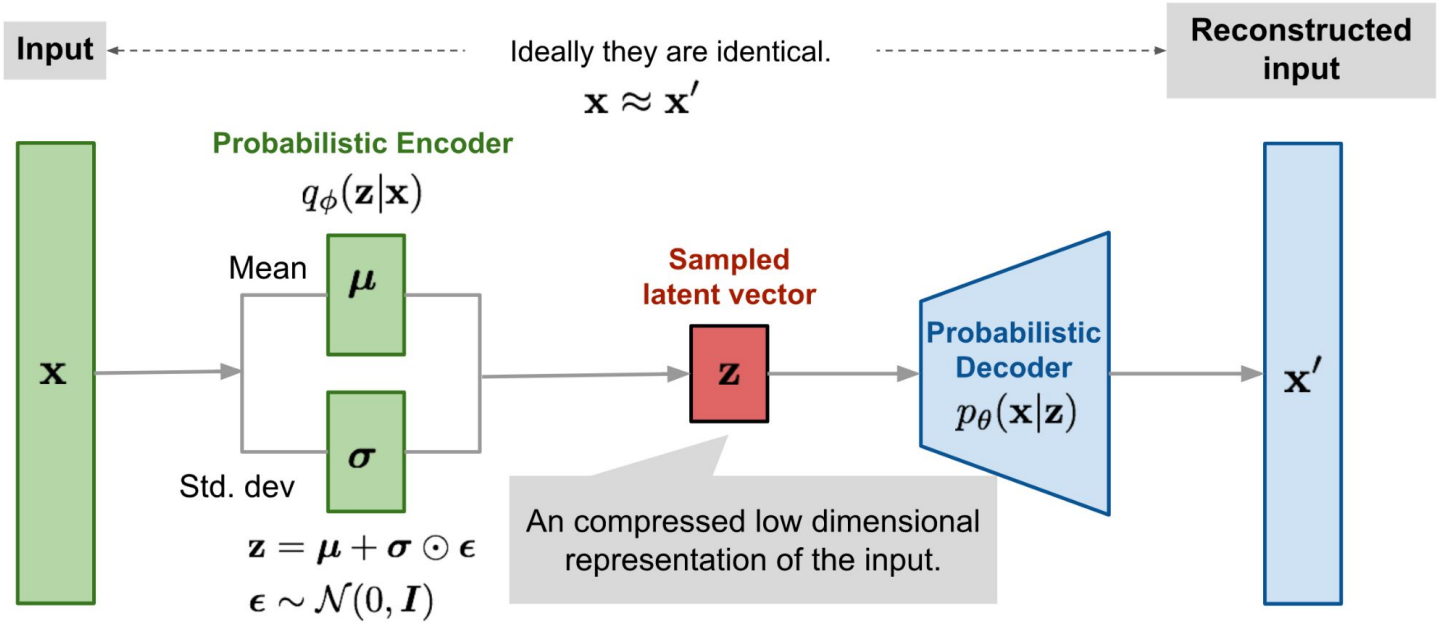
Autoencoders: Regularization

- Model must learn aspects of the input that should be distilled to learn useful representations.
- The reconstruction criterion is insufficient for learning useful representation.
- Often the capacity must be regularized beyond reduced hidden dimensionality, i.e. *undercomplete autoencoders* (as opposite to *overcomplete*).
- Going beyond simple dimensionality reduction with undercomplete autoencoder:
  - Denoising Autoencoder (Paper)
  - Contractive Autoencoder (Paper)

Variational Autoencoders

- The compressed representation can be interpreted as a latent variable z of observable data x.
- We would like to generate new $x' \sim p(x)$ samples using $z \sim q(z)$
- We cannot use vanilla autoencoders for this because the latent space can't be interpolated.
- To learn the distribution we construct two neural networks p_θ and q_ϕ, such that:
  - $\mu(x), \sigma(x) = q_\phi(x)$,
  - $x' = p_\theta(z)$, where $z \sim N(\mu(x), \sigma(x))$.
- Jointly trained p_θ and q_ϕ is called a Variational Autoencoder (VAE)

Variational Autoencoders



Variational Autoencoders

- Kullback-Leibler (KL) divergence measures distance between two distributions:

$$D_{KL}(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)}$$

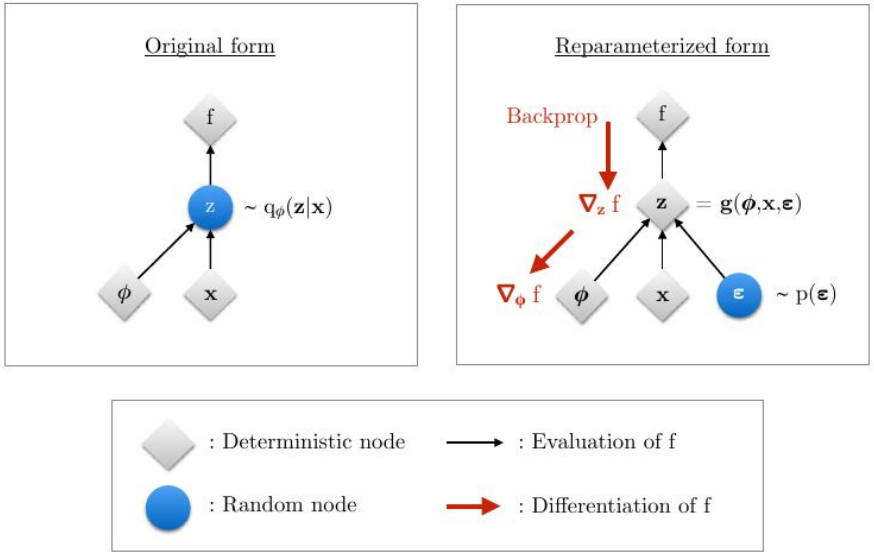
- KL divergence can be used as a regularization so that the posterior distribution tries to match the prior (in our case the Gaussian).
- The full learning objective of the VAE is:

$$\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\mathbf{x}) = \log p_{\boldsymbol{\theta}}(\mathbf{x}) - D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}))$$

  also called evidence lower bound (ELBO).

Variational Autoencoders

- We train VAEs as any other neural network: using backpropagation.
- However, we cannot backpropagate through stochastic node.
- Reparametrization trick: externalize the randomness in *z* by introduction a new random variable Ɛ.



Anomaly Detection

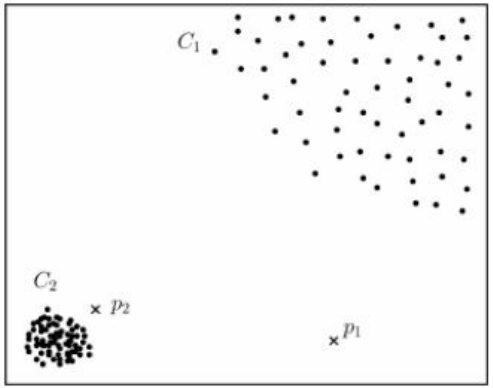


FIGURE 3.3: Two-dimensional data set with two different density clusters C_1 and C_2 and two outliers p_1 and p_2. From Chandola et al. (2009).

Anomaly Detection

- Classification: with enough examples of anomalies we can train a binary classifier.
- Statistical methods: flag data on outside the expected distribution.
- Density-based methods: segment space; flag data in low-density neighbourhood.
- Distance-based methods: segment data; flag small clusters or data away from centroids.
- Autoencoder-based: flag data with high reconstruction error.

Not Covered in These Lectures

- Recurrent Neural Networks (RNNs), LSTMs, GRUs
- Generative Adversarial Networks (GANs)
- Natural Language Processing (NLP)
- Attention and Transformers
- Reinforcement Learning
- Quantum Machine Learning
- ZOO
- List of Papers ML for HEP