LESSON 4: UNSUPERVISED LEARNING AND GENERATIVE METHODS

Savannah Thais

Intro to Machine Learning

Princeton Wintersession 2021

01/28/2021

Outline for Today

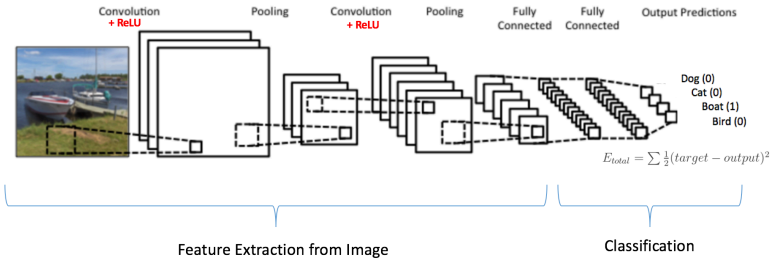
- Quick Review
- Overview of Unsupervised Models
- Use Case Walk Throughs
- Generative Models
- Coding Exercises

Knowledge Review

- What model would you choose to classify images?

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

- What do the filters in a CNN do?
  - What would this filter do?



- What is the purpose of pooling in CNNs?
  - Where is the pooling layer of this network?

- What kind of data are RNNs used for?

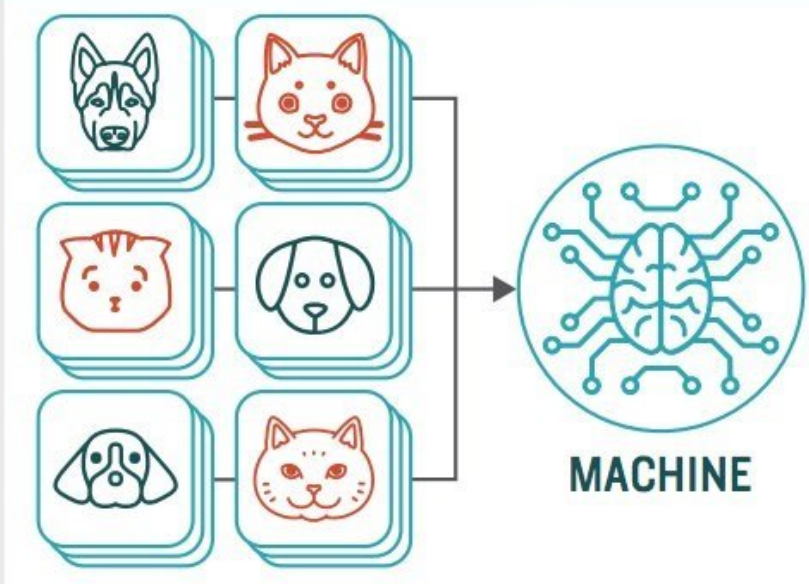
- What is a major difficulty when training RNNs?

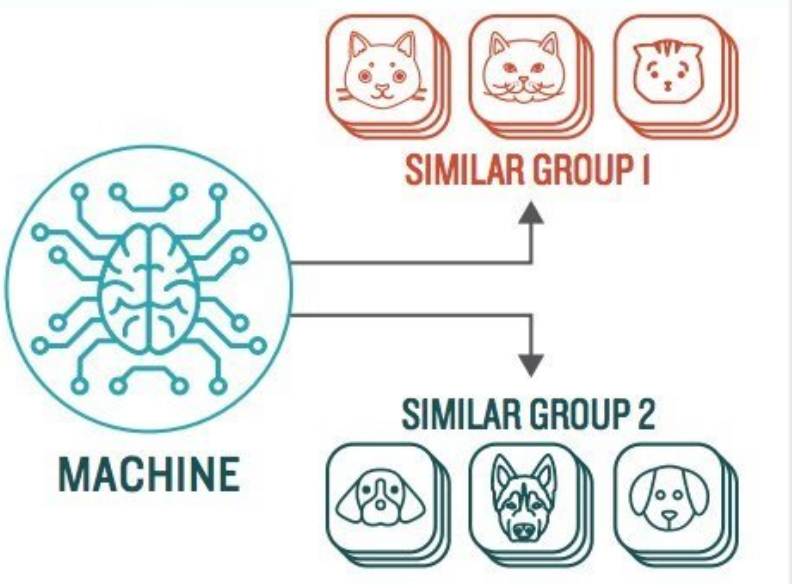
- What are h and x in this equation?

Formula for applying Activation function(tanh):

$$h_t = \tanh\left(W_{hh}h_{t-1} + W_{xh}x_t\right)$$

Unsupervised Algorithms

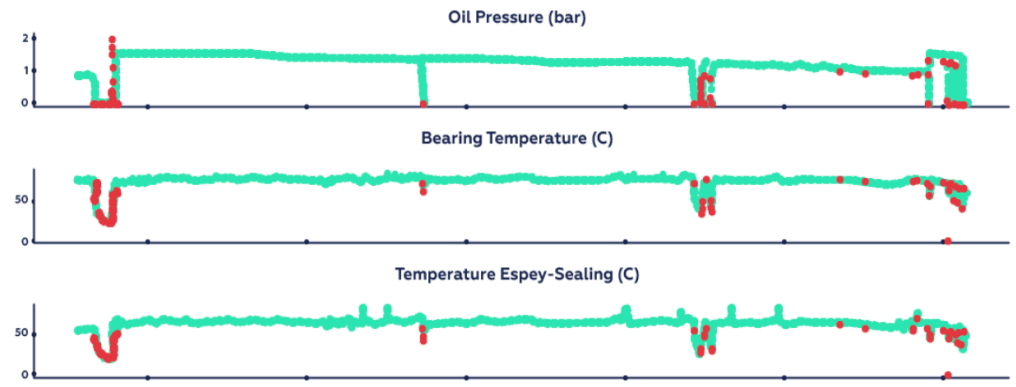


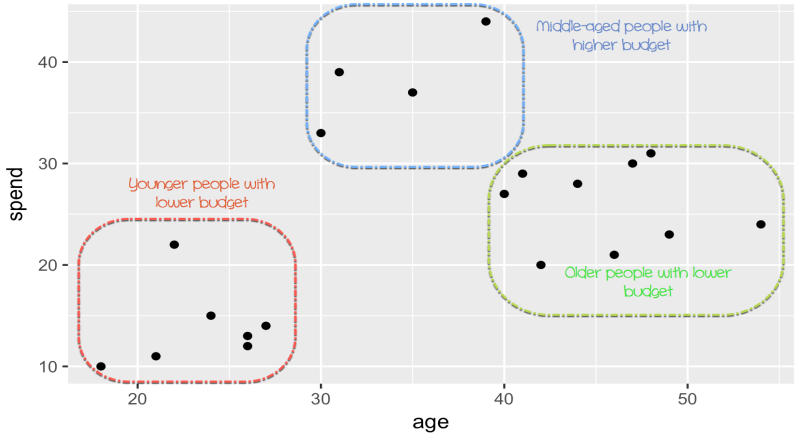


Unsupervised Learning Basics

Model is trained without examples of correct behavior

- Goal is to learn something about the underlying structure of data
  - This may be useful to downstream tasks, may not
- Central challenge is evaluating model performance
  - Real world utility depends if the patterns actually exist is statistically significant ways
- But a very powerful class of algorithms
  - Anomaly detection, data segmentation, recommender systems…



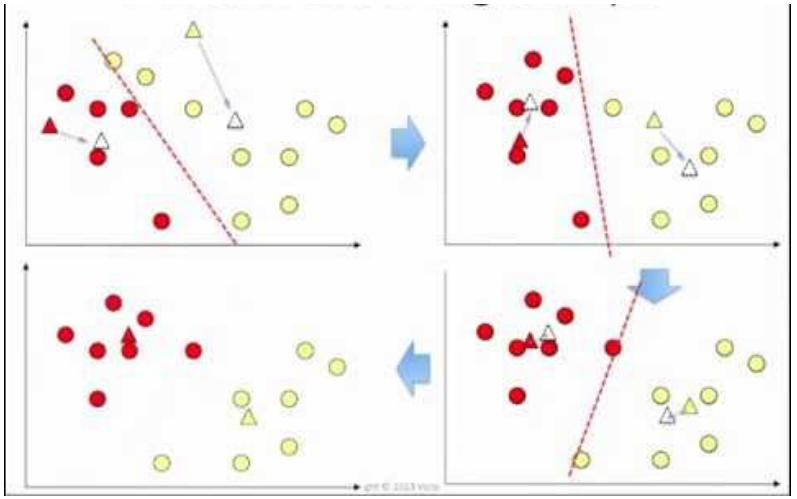


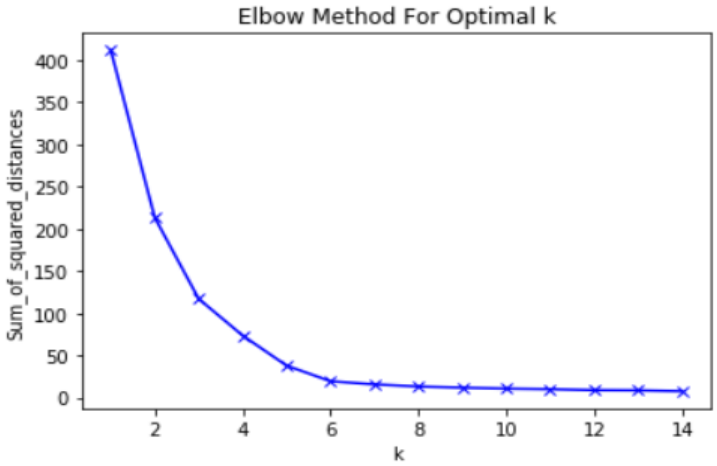
Clustering: kMeans

Clustering aims to group similar data points together (you mathematically define similarity)

kMeans clustering groups data into k classes by minimizing average distance to cluster center

1. Initial centroids randomly selected
2. All points assigned to closest centroid
3. Centroid adjusted towards mean of all assigned points
4. Stops when clusters are stable



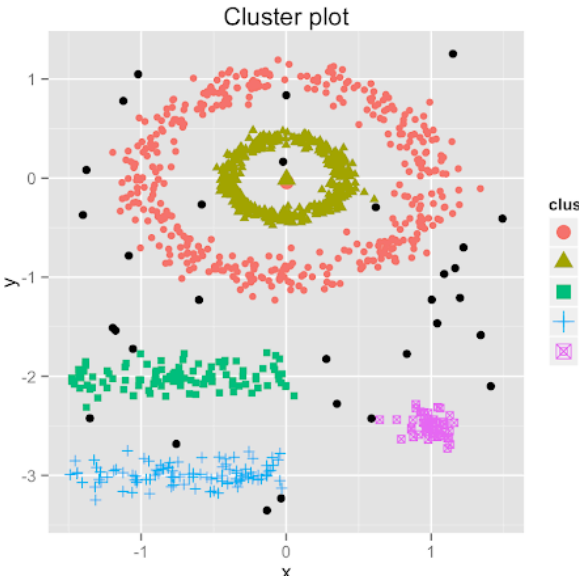


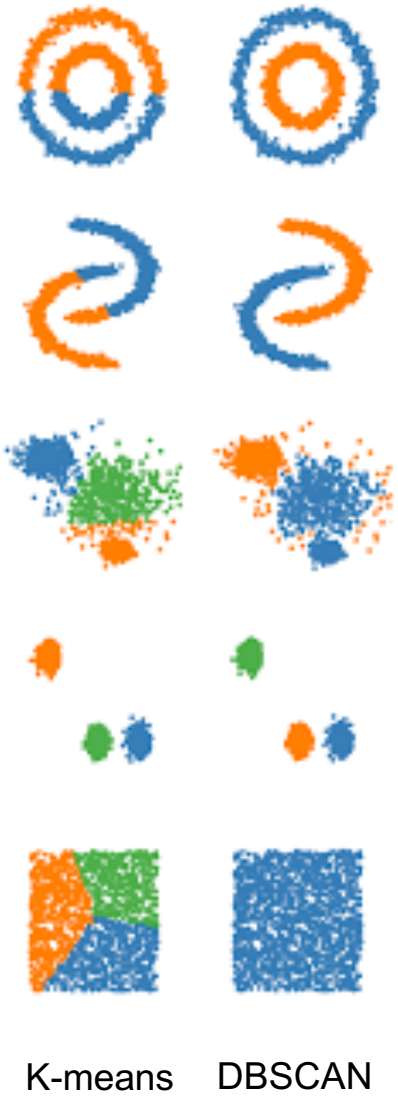
- Main challenge is optimizing k
  - Common method is elbow plot
- Define most useful distance metric
- kMeans works best for ~circular clusters

further reading

Clustering: DBSCAN

- Density-based spatial clustering of applications with noise
  1. Starting point is slected
  2. All neighbors (distance less than some specified cutoff) are mapped
  3. If enough neighbors found, all are added to a cluster
  4. Repeat 2-3 until no new points are added
  5. Restart with a new (ungrouped) starting point
- More tuneable parameters but more flexible cluster shapes
  - And outlier detection ‘for free’



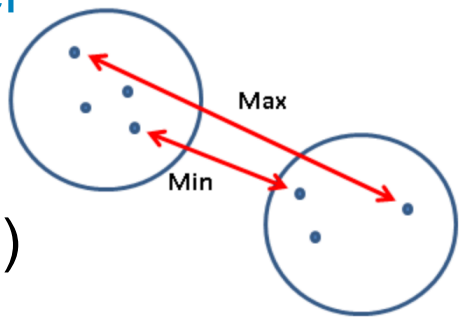


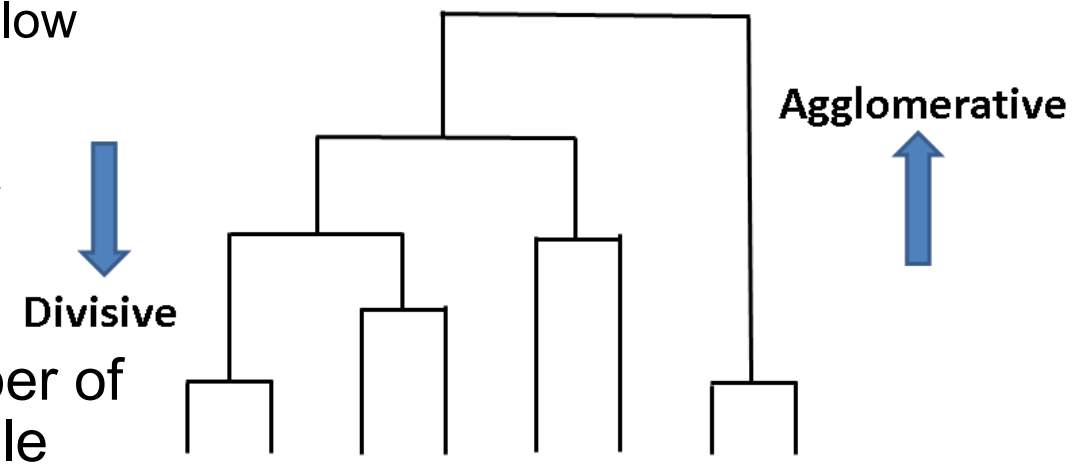
K-means DBSCAN

further reading

Clustering: Hierarchical

- Iteratively cluster or separate points based on order of similarity
  - Agglomerative: all points start as separate clusters
  - Divisive: all points start in same cluster
- Several methods for determining similarity (linkage)
  - Ward's: minimize (maximize) variance of clusters being merged (divided)
  - Average: create ~average distance of datapoints in each cluster
  - Complete: form clusters to allow maximum distance between clusters
  - Single: form clusters to allow minimum distance between clusters
- Don't have to specify number of clusters, interpretable, stable
  - But is slower (especially for large datasets)





further reading

Dimensionality Reduction: PCA

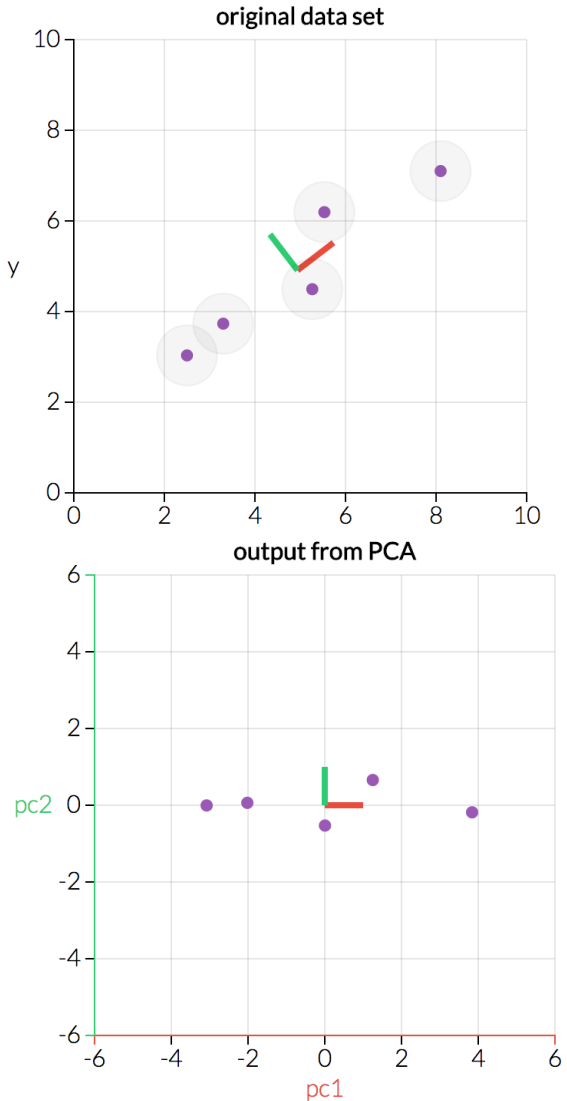
Transform and group input features into higher-level, informative sets

- Principle Component Analysis derives an orthonormal basis for the dataset
  - Each direction is linearly uncorrelated
- Find a projection matrix V that minimizes reconstruction error between original data and reduced space
  - Inversely, maximizes variance of the data

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

- Main hyperparameter: number of desired dimensions
- Typically solved with eigenvalue decomposition

further reading



Dimensionality Reduction: SVD

- Singular Value Decomposition is a generalized PCA that finds all eigenvectors of dataset covariance matrix
  - Based on theorem that all matrices can be factorized as $A=USV^T$ where U and V are orthogonal
- Eigenvectors are ranked by their 'singular value'
  - In highly correlated data many SVs will be small
- Can do dimensionality reduction with goal of level of information capture rather than number of dimensions

$$A = U S V^T = \sigma_1 u_1 v_1^T + \ldots + \sigma_n u_n v_n^T$$

S

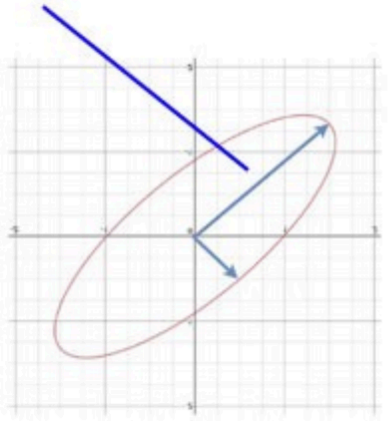
$$\begin{pmatrix} \sigma_1 & & & & \\ & \sigma_2 & & & \\ & & \ddots & & \\ & & & \sigma_r & \\ & & & & \ddots \\ & & & & 0 \end{pmatrix}$$

σ_2: *singular value*

more significant — less significant

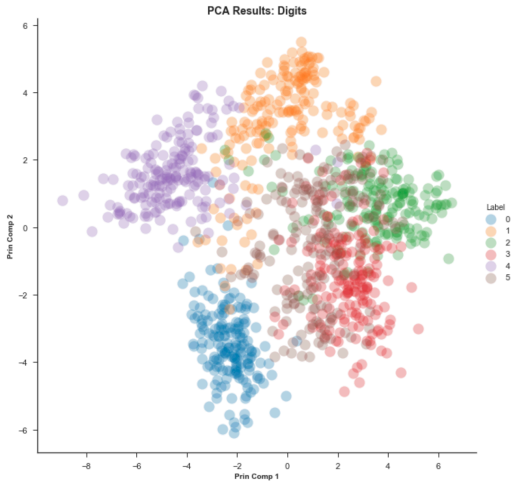
$$A = U S V^T = \sigma_1 u_1 v_1^T + \ldots + \sigma_r u_r v_r^T$$

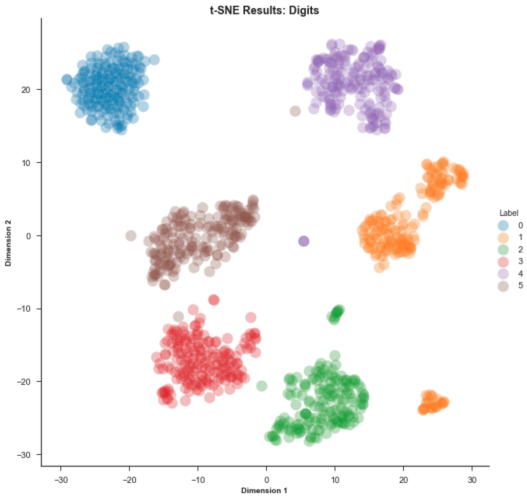
$m \times n$ $\quad m \times 1 \quad 1 \times n \quad m \times n$

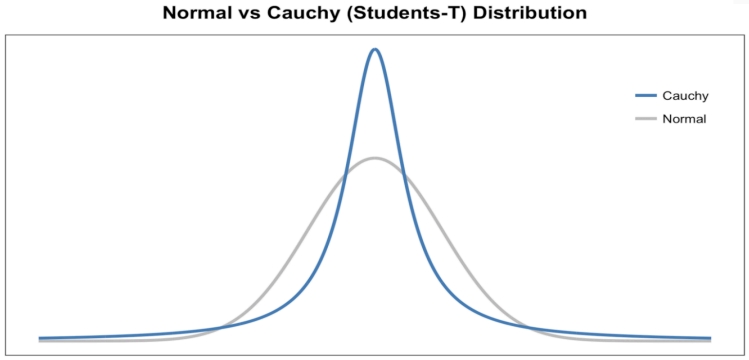


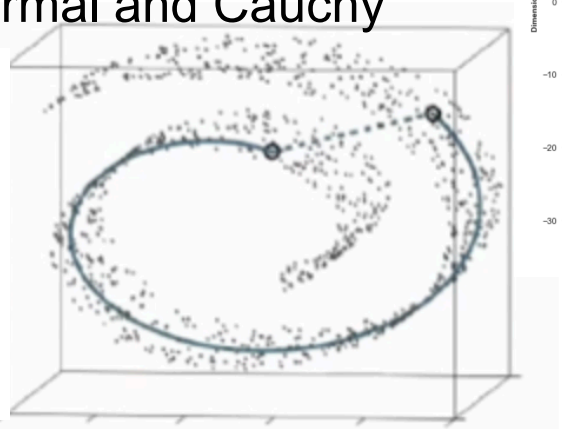
Dimensionality Reduction: tSNE

- T-Distributed Stochastic Neighbor Embedding preserves small pairwise distances (local similarities)
  - Opposite of PCA which preserves large pairwise distance (variance)
  - Useful for different types of data distributions/tasks (tSNE is typically used for visualization)
- Center a Gaussian over each point, measure density of points under the Gaussian, normalize to get probabilities (similarities)
  - Repeat with Cauchy distribution
  - Minimize difference between normal and Cauchy





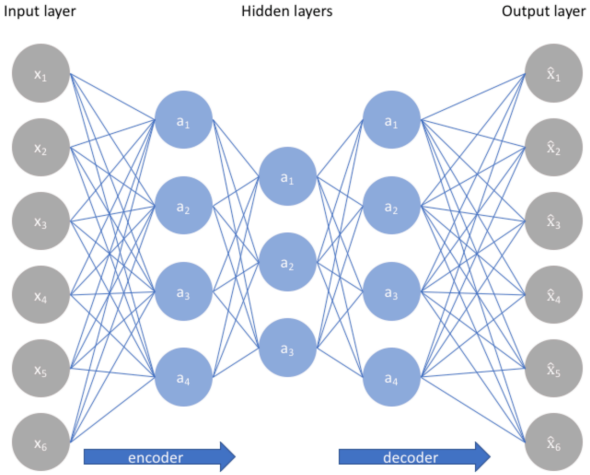


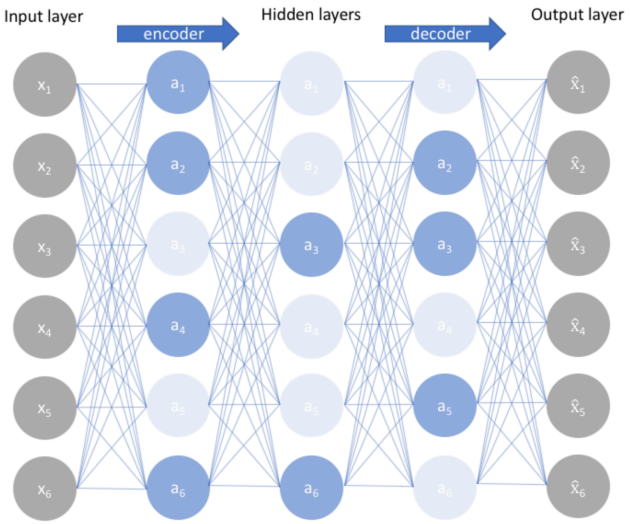


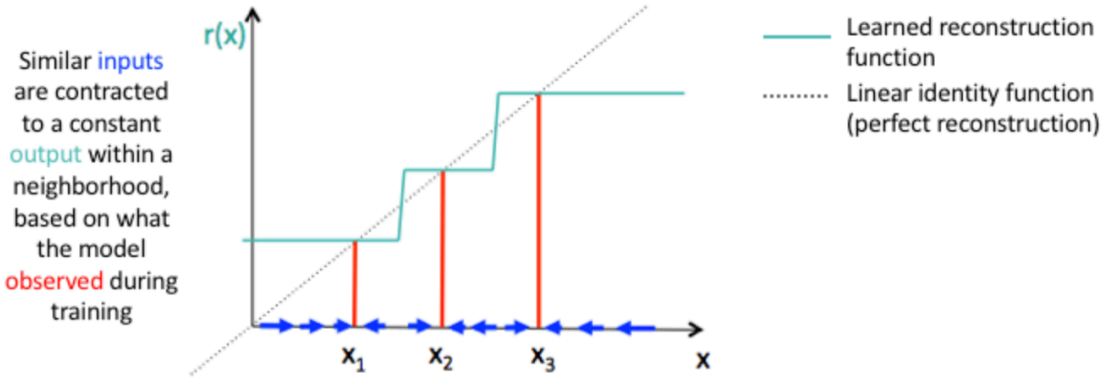
Autoencoders

Loss-ily transform data such that it can be transformed back

- Consists of an encoder that re-embeds the data and an decoder that attempts to reconstruct original data
  - Both are typically feed-forward NNs (can be part of same NN)
- Different ways to enforce loss/create a bottleneck
  - Undercomplete (reduced hidden dim)
  - Sparsity (only allow some activations)
  - Contractive (minimize derivative of hidden layer activations)



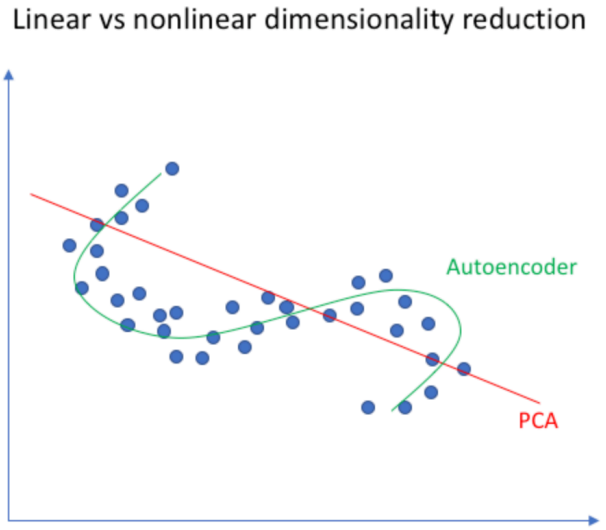


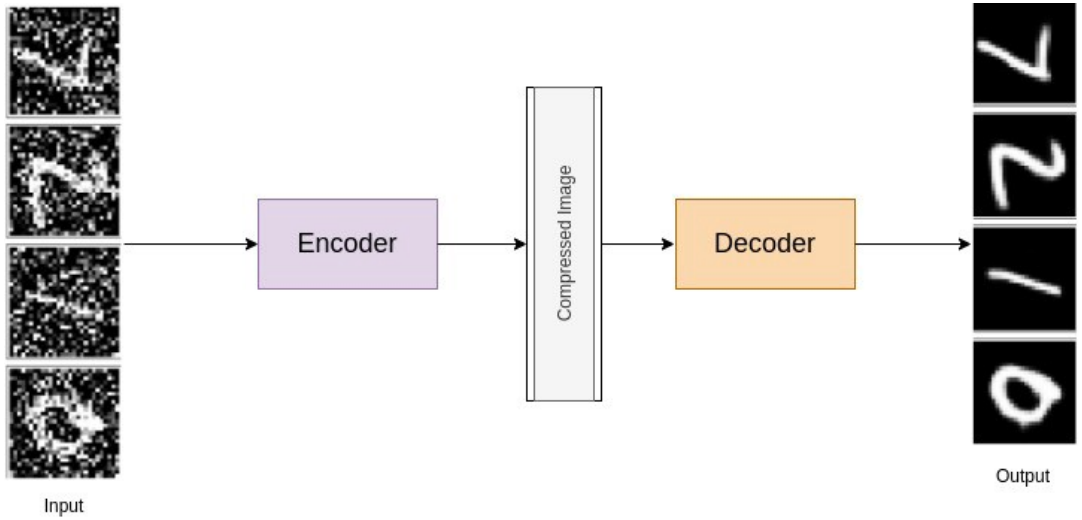


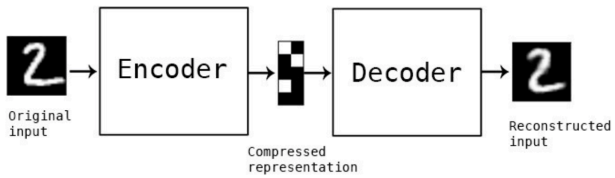
further reading

Autoencoder Uses

- Data compression
- Non-linear dimensionality reduction
  - Learn a manifold representation of the data
  - New embedding space can be useful for other tasks
- Denoise samples
  - Reconstruction goal is input with noise removed
  - Learn to map data to a manifold that describes non-noisy space





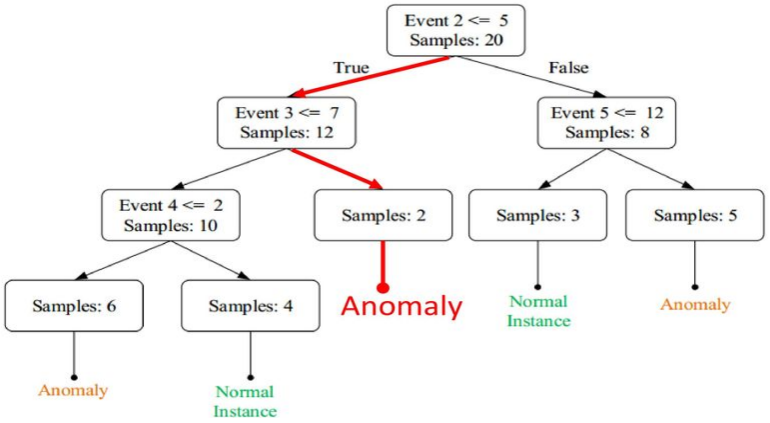


Anomaly Detection

Find outliers in data by understanding the underlying distribution

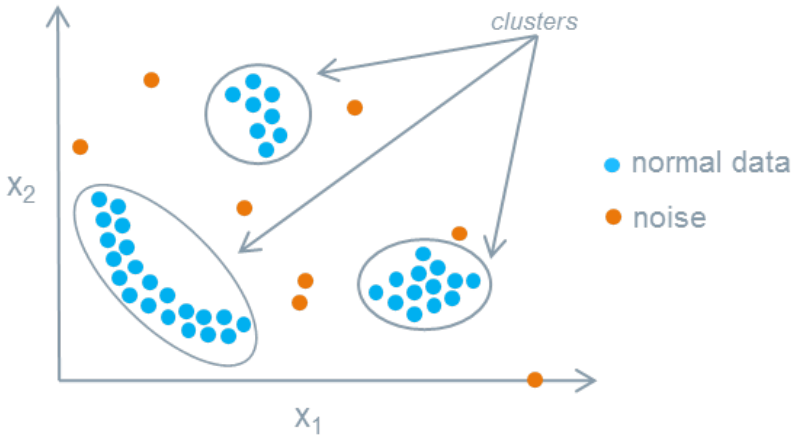
Classification Based

- If no examples of anomalies: train classifier and flag extremely low scoring data points
- If examples of anomalies: early isolation in BDTs or 'other' class



Clustering Based

- If no examples of anomalies: train classifier and flag extremely new points outside of clusters
- If examples of anomalies: use DBSCAN or similar algorithm



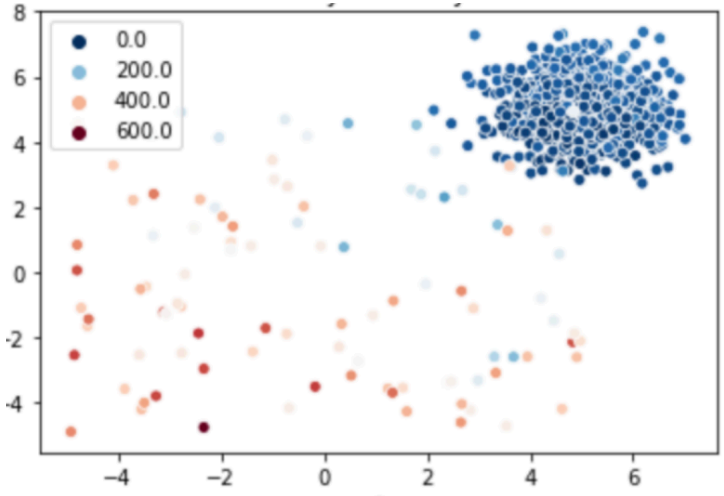
further reading

Anomaly Detection

Find outliers in data by understanding the underlying distribution

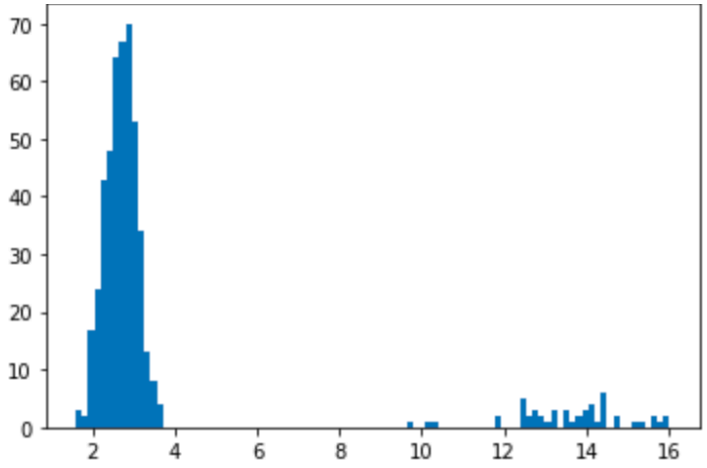
Dimension Based

- Assume full datasest is normally distributed across reduced dimensions
- Flag datapoints outside normal distribution



Autoencoder Based

- Assume trained autoencoder (on non-anomalous data) has a normal distribution of error
- Flag new datapoints outside normal distribution



further reading

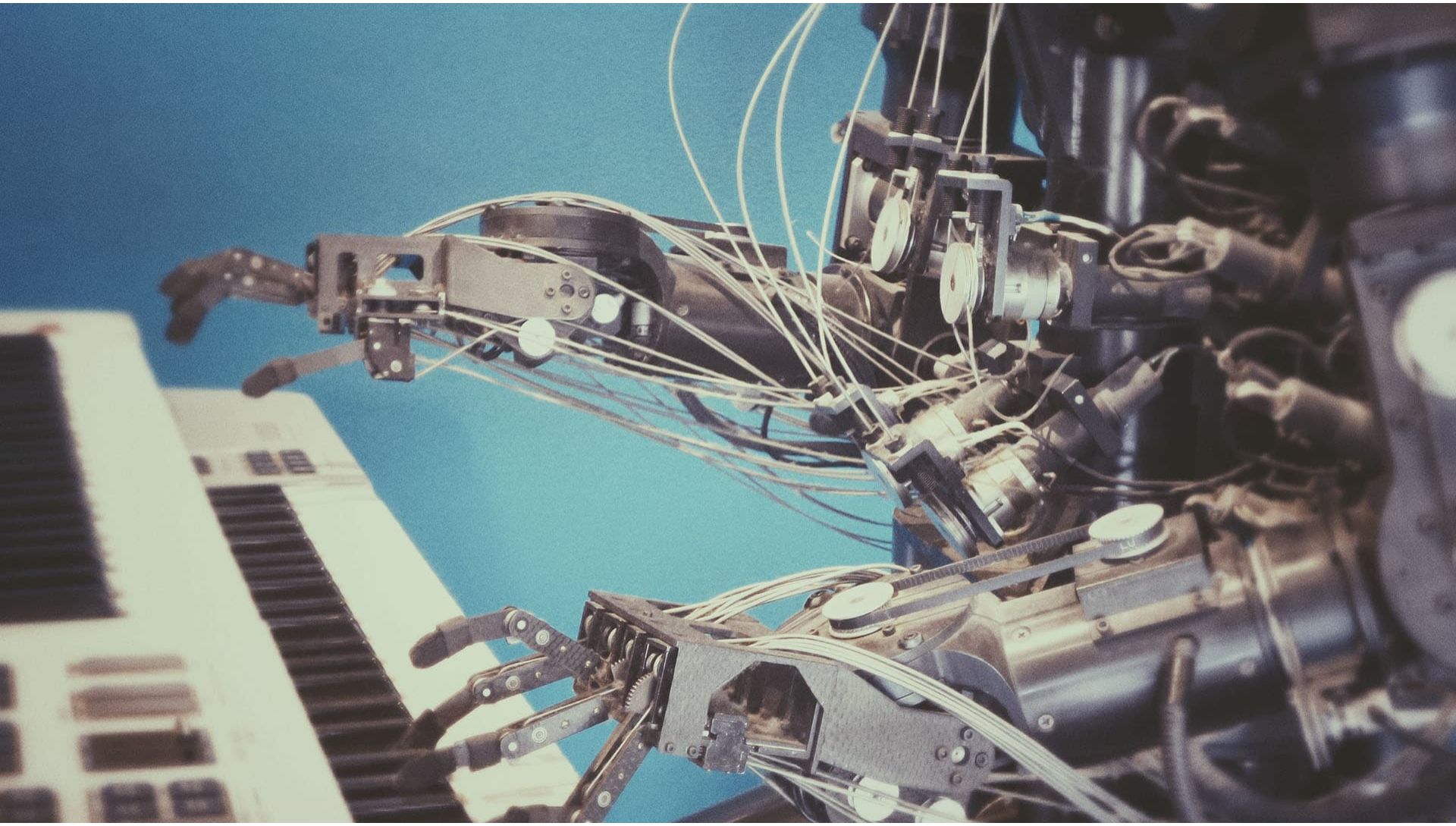
Coding Interlude!

We'll walk through real world examples of using clustering to segment customer bases and using autoencoders for health insurance fraud detection.

Break!

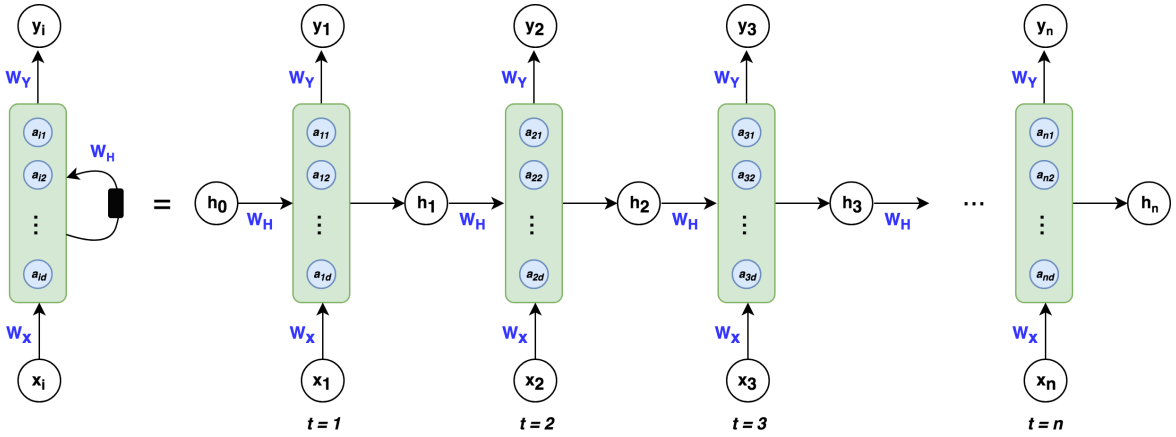
When we return we'll discuss generating new data with supervised and unsupervised models.

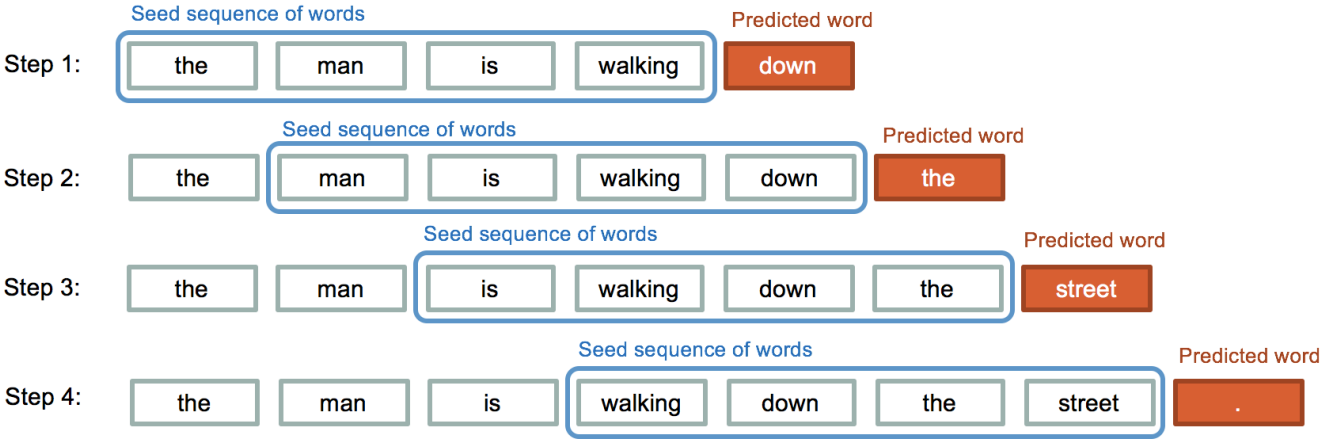
Generative Models



Recall Yesterday: Generation with RNNs

RNN trained on example sequences can use generated sequences as input to continuously create





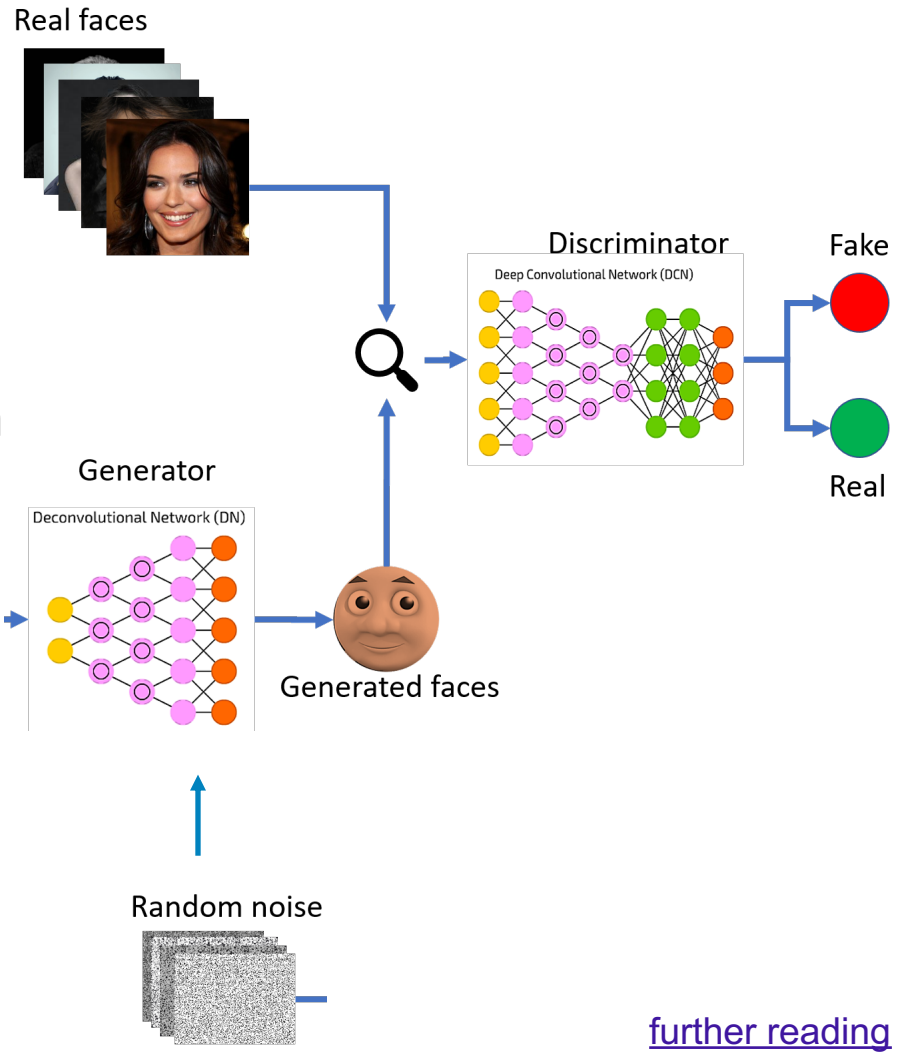
you can learn about trees...
and bees...
and knees.
and knees on trees!
and bees on threes!

you can read about anchors.
and all about ants.
you can go by cow.
mow we have a mot of inke.
and i hope that they may.
good long, you can learn about.

Generative Adversarial Networks

Two NNs with different learning goals are trained in tandem

- Generator network creates new examples from random noise
- Discriminator network distinguishes generated and real data
- Goal of tandem network is to learn the underlying probability distribution of true data
  - Without specifying features
  - Must have a loss function that balances both networks
- GANs typically trained by having both feed into a downstream task and moving networks in opposite direction during gradient descent
- Can use different network types depending on generated data type

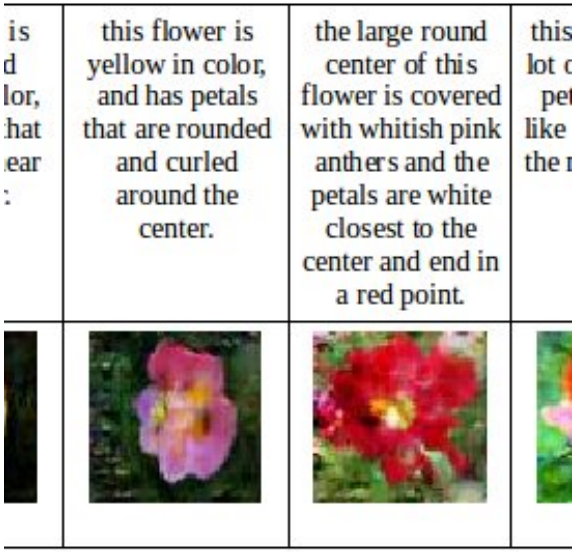


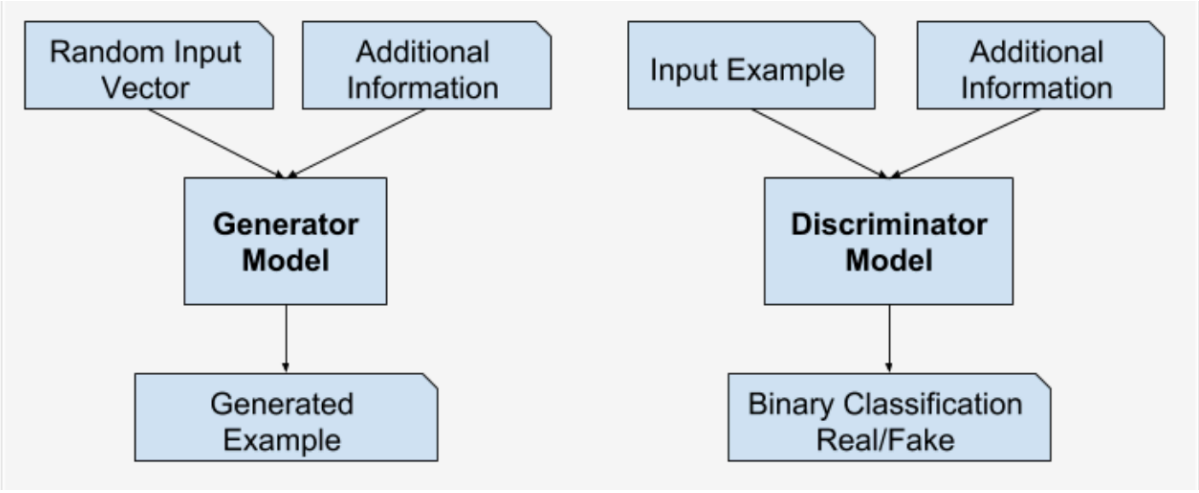
further reading

Conditional GANs

Train generative model to create new example from conditioned input vector

- Examples: specific digit or word in handwriting generator, or input image for caption generator
  - Learns separate domains of underlying probability distribution
- Both networks get an additional (conditioned) input vector
  - Can be learned or constructed





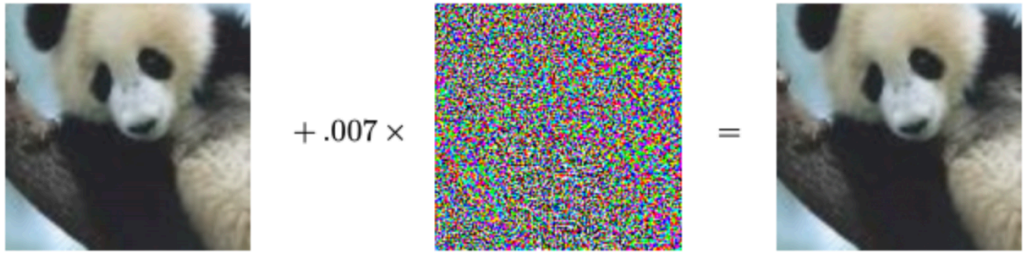
These are the foundation of DeepFakes, maybe the best demonstration of how dangerous unchecked ML can be

further reading

Fast Gradient Sign

Use the gradients of a trained NN to create adversarial or idealized examples

- Use gradient of loss at a specific datapoint to create a new datapoint that maximizes or minimizes loss function
- Other model based generative models exist
  - Exploit learned features to create new examples
- Adversarial attacks are a big concern in AI safety
  - Easiest if you have knowledge of the network, but can also brute force
  - Large area of research is designing networks resistant to adversarial examples



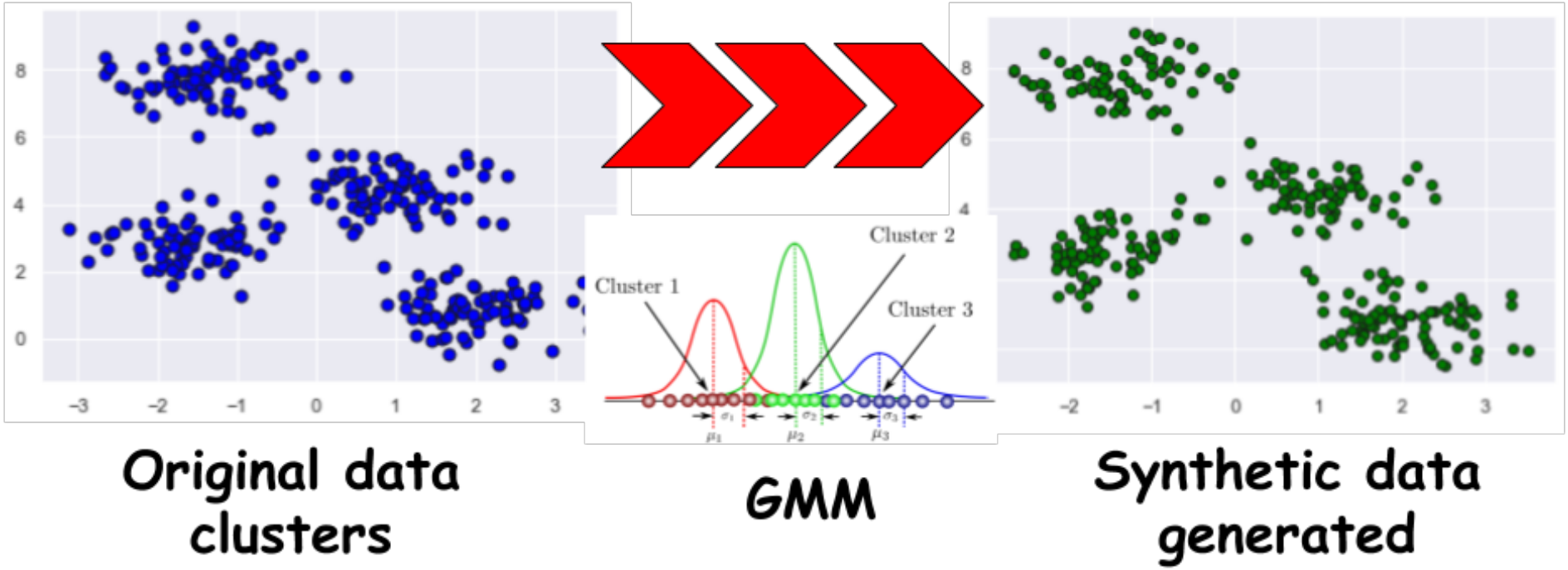
$$adv\_x = x + \epsilon * \text{sign}(\nabla_x J(\theta, x, y))$$

further reading

Gaussian Mixture Models

Sample from a learned probability distribution to generate new examples

- GMMs use Gaussian distributions to model feature distributions in different clusters
- Can use the parameters of the gaussian to generate data that is probabilistic



Coding Interlude!

We'll walk through some cool examples using ML to generate photos in new styles, real time sketching, and original songs!

Let's revisit a question from Monday

Can you think of an example of ML in your daily life (or research)?:

- Do you think it's a supervised or un-supervised model?
  - What type of model do you think was used?
  - What data could have been used to train it?
  - What might have been the learning objective?
- Can you imagine some important training considerations?

Coding Time!

Happy to answer any questions!

link to Day 4 notebook

✉ sthais@princeton.edu @basicsciencesav