UNSUPERVISED LEARNING

FA590 STATISTICAL LEARNING IN FINANCE

Dr. Zonghao Yang

Stevens Institute of Technology

Previous Lecture

- Learn the principles of gradient descent
- Describe the architecture of a neural network, including the roles of input, hidden, and output layers, and understand the function of different activation functions such as ReLU, sigmoid, and tanh
- Articulate the hyperparameters of a neural network and build intuition on how to tune the hyperparameters based on learning curves

Learning Objectives

- Understand the differences between supervised and unsupervised learning
- Apply clustering techniques such as hierarchical clustering, K-means, and DBSCAN to segment data into meaningful groups
- Explore Principal Components Analysis (PCA) and its extensions in the context of asset pricing to create low-dimensional representation of data

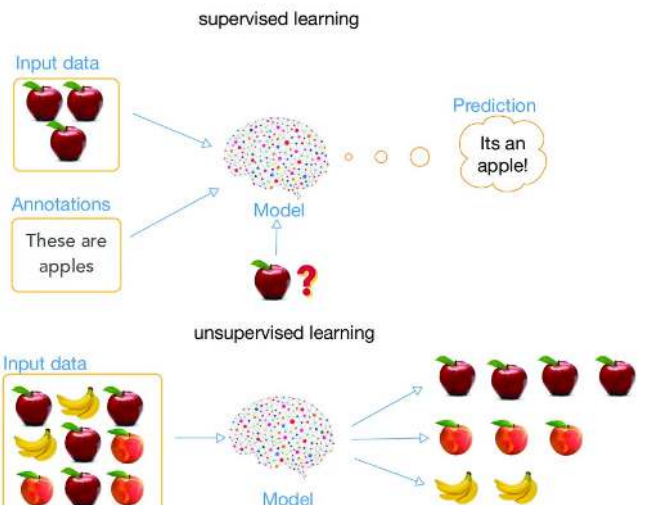
Part I

Cluster Analysis

Supervised vs. Unsupervised

- Until now, we have considered **supervised learning** problems
- In supervised learning, we train an algorithm to predict the target y, given features X
- In unsupervised learning, there is no target
  - The algorithm detects patterns within X itself

Illustration



CLUSTERING

- **Clustering** techniques are a typical example of unsupervised learning
  - Segment the data into a set of homogeneous clusters of data points
- Example: Identify groups of customers with similar behaviors, preferences, or characteristics

Cluster Analysis

- Cluster analysis measures the distance between data points, and forms clusters according to these distances
- **Hierarchical clustering**
  - **Agglomerative** methods begin with n clusters and sequentially merge similar clusters until a single cluster is obtained
  - Useful when the goal is to arrange the clusters into a natural hierarchy
- **Non-hierarchical clustering**
  - **K-means algorithm** Using a pre-specified number of clusters, the method assigns data points to each cluster
  - **Density-based algorithms** No need to pre-specify the number of clusters

Distance Metrics

- d_{ij}: A **distance metric** between data i and data j
  - Non-negative: $d_{ij} \geq 0$
  - Self-proximity: $d_{ii} = 0$
  - Symmetry: $d_{ij} = d_{ji}$
  - Triangle inequality: $d_{ij} \leq d_{jk} + d_{kj}$
- Typical distance metric: Euclidean distance

$$d_{ij} = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}$$

- **Features typically ought to be normalized**

Measuring Distance between Clusters

- Assume we have two clusters $A = \{A_1, A_2, \ldots, A_m\}$ and $B = \{B_1, B_2, \ldots, B_n\}$
- Denote Δ_{AB} as the distance between cluster A and B
- **Minimum Distance** $\Delta_{AB} = \min_{(i,j)\in(A,B)} d_{ij}$
- **Maximum Distance** $\Delta_{AB} = \max_{(i,j)\in(A,B)} d_{ij}$
- **Average Distance** $\Delta_{AB} = \frac{1}{nm} \sum_{(i,j)\in(A,B)} d_{ij}$
  - Fairly robust with respect to outliers and number of data points
  - Good for spherical clusters

Measuring Distance between Clusters (cont.)

- **Centroid Distance**
  - The centroid of cluster *A* is just its 'center of mass,' i.e., the average across all data points in cluster *A*
  - The centroid distance between cluster *A* and *B* is then just the distance between their respective centroids

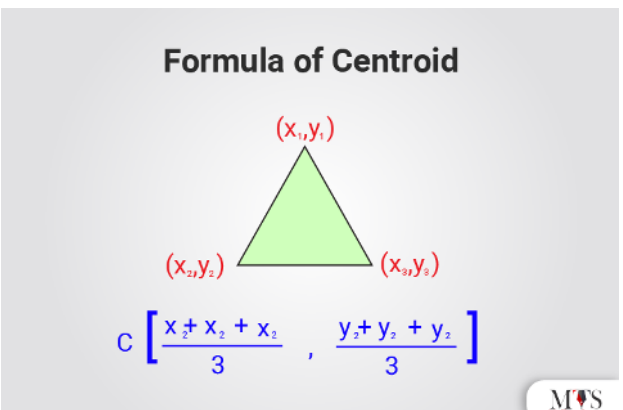
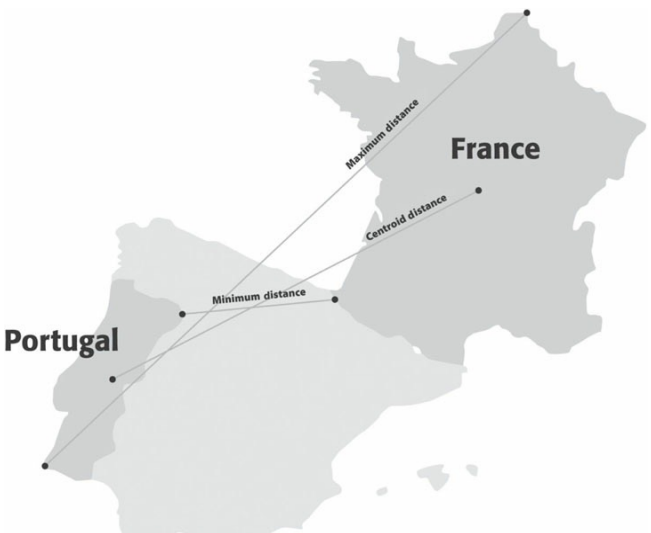
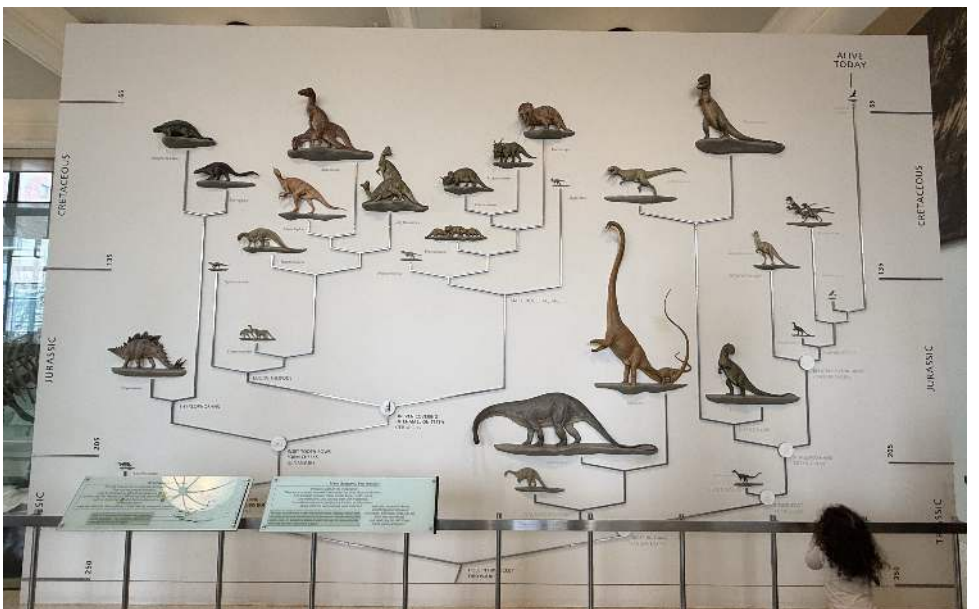


Illustration of Different Distance Notions



Dinosaurs at American Museum of Natural History

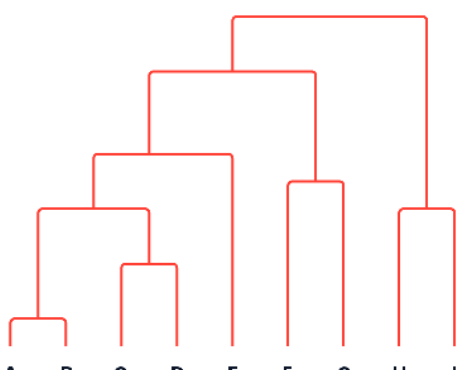


Hierarchical Agglomerative Clustering

1. Start with n clusters (each data point = one cluster)
2. At every step, the two clusters with the smallest distance are merged, i.e.,
   - A single data point is added to existing clusters, or
   - Two existing clusters are combined
3. Stop once there is only one cluster left

Dendrogram

Dendrogram A dendrogram is a tree-like diagram that summarizes the process of clustering



THE K-MEANS ALGORITHM

- The **K-means** algorithm is not hierarchical
- Pre-specify a desired number of clusters, K, and assign each data point to one of the K clusters so as to minimize the dispersion within the clusters (as homogeneous as possible)
- Within-cluster dispersion is the sum of distances of data points from their cluster centroid

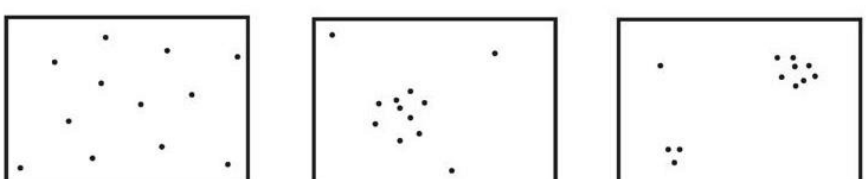


Figure. Cluster Dispersions

K-MEANS CLUSTERING ALGORITHM

1. Start with K initial clusters (user chooses K)
2. At every step, each data point is reassigned to the cluster with the “closest" centroid
3. Recompute the centroids of clusters that lost or gained a data point, and repeat Step 2
4. Stop when moving any more data points between clusters increases cluster dispersion

Visualizing K-Means[1]

[1] Visualizing the K-means algorithm: https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

Choosing the Number of Clusters (K)

- The choice of the number of clusters can either be driven by external considerations (e.g., previous knowledge, practical constraints, etc.), or
- Try a few different values for K and compare the resulting clusters

- In many cases, there is no information to be used for the initial partition
  - In these cases, the algorithm can be rerun with different randomly generated starting partitions to reduce chances of the heuristic producing a poor solution

DBSCAN

- **Density-Based Spatial Clustering of Applications with Noise (DBSCAN)** offers several advantages over K-means
  - No need to pre-specify the amount of clusters
  - Individual data points can be discarded as outliers (not part of a cluster)
  - Very fast
- Disadvantage: Knowledge of an intrinsic **length-scale** ε is required
- Intuition: Points within a fixed distance, denoted ε, are grouped together

DBSCAN ALGORITHM

1. Select a data point at random
2. Are there more than *minPts* in its ε-neighborhood?
   (i) Yes: assign to cluster
   (ii) No: mark the selected data point as visited, but not a cluster (for now)
3. Continue with step (1) until all data points have been visited

Visualizing DBSCAN[2]

[2] Visualizing DBSCAN: https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

HYPERPARAMETERS OF DBSCAN

- ε Neighborhood size around a data point
  - A small ε may result in smaller clusters and more points being classified as noise
  - A large ε may merge distinct clusters into one, reducing the algorithm's sensitivity to density variations
  - **Selection** Require domain knowledge
- *minPts* the minimum number of data points to form a dense region
  - A small *minPts* can lead to over-detection of clusters, including spurious small clusters
  - A large *minPts* makes it harder to form clusters, potentially leaving more points as noise
  - **Selection** A common heuristic is to set MinPts to $D + 1$, where D is the number of dimensions in the data

Validating Clusters

- **Cluster Interpretability** Is the interpretation of the resulting clusters reasonable?
- **Cluster Stability** Do cluster assignments change significantly if some of the inputs are altered slightly?
  - Hierarchical clustering tends to have low stability, i.e., reordering data or dropping a few data points can lead to a different solution
- **Cluster Separation** Examine the ratio of between-cluster variation to within-cluster variation to see whether the separation is reasonable
- **Number of Clusters** The number of resulting clusters must be useful, given the purpose of the analysis

Part II

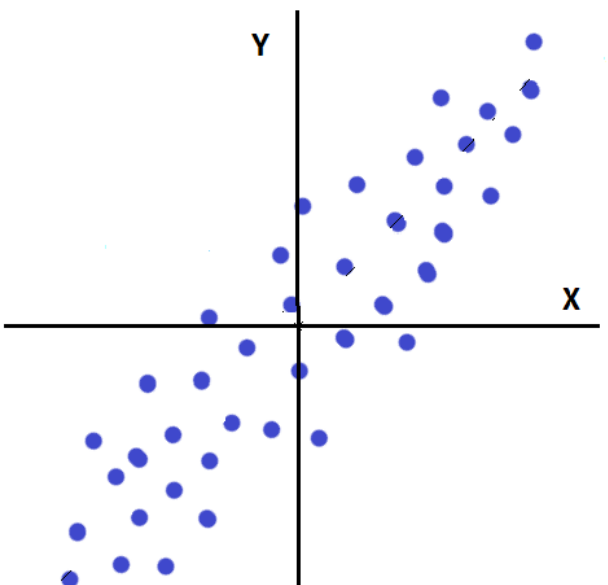
Principal Component Analysis

Dimensionality Reduction

- **Dimensionality Reduction** Suppose $X \in R^p$; Find a “useful" transformation $\phi : R^p \to R^k$ ($k < p$) that helps in the downstream prediction task
- Feature selection is one ad hoc example of dimensionality reduction
- Instead of removing the features, dimensionality reduction techniques use a transformation to summarize the information among features while reducing dimensionality
- **Principal Component Analysis (PCA)**

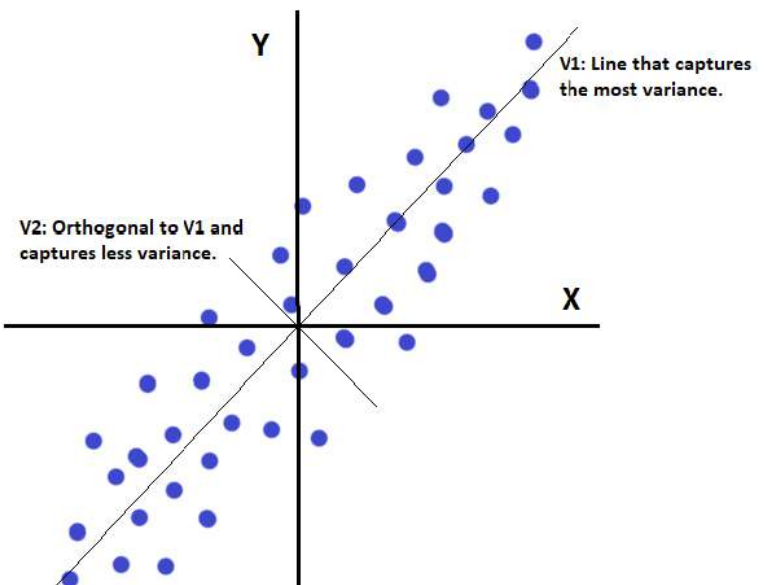
PCA: Build Intuition

What is the best one-dimensional representation of the following two-dimension sample space?



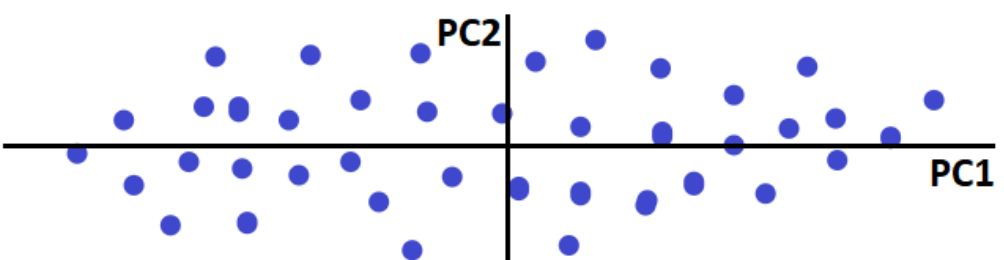
PCA: Build Intuition

What is the best one-dimensional representation of the following two-dimensional sample space?



PCA: Build Intuition

Rotate the presentation of the sample according to the new axises



Principal Components Analysis

- PCA produces a low-dimensional representation of a dataset
- PCA finds a sequence of linear combinations of the variables that have maximal variances, and are mutually uncorrelated
- A tool for data visualization

Figure. 2D Visualization of Digits Dataset

PRINCIPAL COMPONENTS ANALYSIS: DETAILS

- The first principal component of a set of features $X_1, X_2, \ldots, X_p$ is the normalized linear combination of the features that has the largest variance

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \cdots + \phi_{p1} X_p, \text{ where } \sum_{j=1}^{p} \phi_{j1}^2 = 1$$

- We refer to the elements $\phi_{11}, \ldots, \phi_{p1}$ as the loadings of the first principal component
- The principal component loading vector

$$\phi_1 = (\phi_{11} \ \phi_{21} \ \ldots \ \phi_{p1})^T$$

Computation of Principal Components

- Suppose we have a $n \times p$ data set $\mathbf{X}$
- Assume that each of the variables in $\mathbf{X}$ has been centered to have mean zero (that is, the column means of $\mathbf{X}$ are zero)
- We then look for the linear combination of the sample feature values of the form

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \cdots + \phi_{p1} x_{ip}$$

  for $i = 1, \ldots, n$ that has the largest sample variance, subject to the constraint that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$
- The sample variance of the z_{i1} can be written as

$$\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2$$

COMPUTATION: CONTINUED

- The first principal component loading vector solves the optimization problem

$$\text{maximize}_{\phi_{11},\ldots,\phi_{p1}} \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{j1}x_{ij}\right)^2 \quad \text{subject to} \quad \sum_{j=1}^{p}\phi_{j1}^2 = 1$$

- This problem can be solved via a singular-value decomposition of the matrix $\mathbf{X}$
  - ϕ_1 is the eigenvector corresponding to the largest eigenvalue of the matrix $\frac{1}{n}\mathbf{X}\mathbf{X}^\top$
- We refer to Z_1 as the first principal component, with realized values $z_{11}, \ldots, z_{n1}$

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + \cdots + \phi_{p1}X_p$$

Further principal components

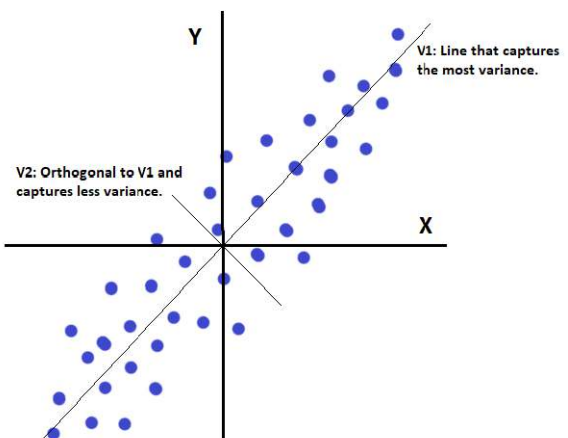
- The second principal component scores $z_{12}, z_{22}, \ldots, z_{n2}$ take the form

$$z_{i2} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \cdots + \phi_{p2} x_{ip},$$

  where $\phi_2 = (\phi_{12}\ \phi_{22}\ \ldots\ \phi_{p2})^T$
- The second principal component Z_2 has maximal variance among all linear combinations that are *uncorrelated* with Z_1
- Constraining Z_2 to be uncorrelated with Z_1 is equivalent to constraining the direction ϕ_2 to be orthogonal (perpendicular) to the direction ϕ_1
- There are at most $\min(n-1, p)$ principal components

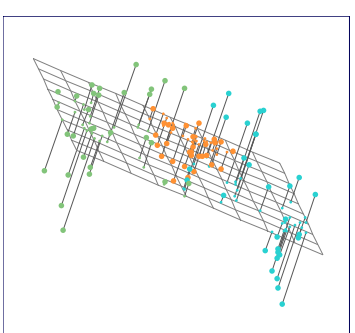
GEOMETRY OF PCA

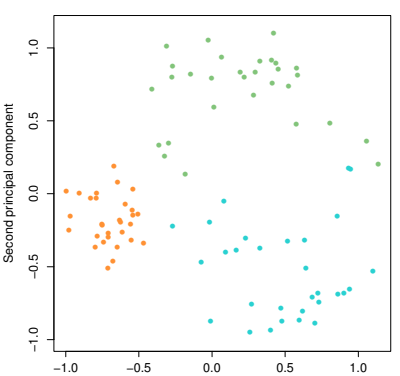
- The loading vector ϕ_1 with elements $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ defines a direction in feature space along which the data vary the most
- If we project the n data points $x_1, \ldots, x_n$ onto ϕ_1, the projected values are the principal component scores $z_{11}, \ldots, z_{n1}$ themselves



The Closest Hyperplane Interpretation

- The first principal component loading vector ϕ_1 defines the line in *p*-dimensional space that is *closest* to the *n* observations
- The notion of principal components as the dimensions that are closest to the *n* observations extends beyond just the first principal component
- For instance, the first two principal components of a data set span the plane that is closest to the *n* observations





SCALING OF THE VARIABLES MATTERS

- If the variables are in different units, scaling each to have standard deviation equal to one is recommended

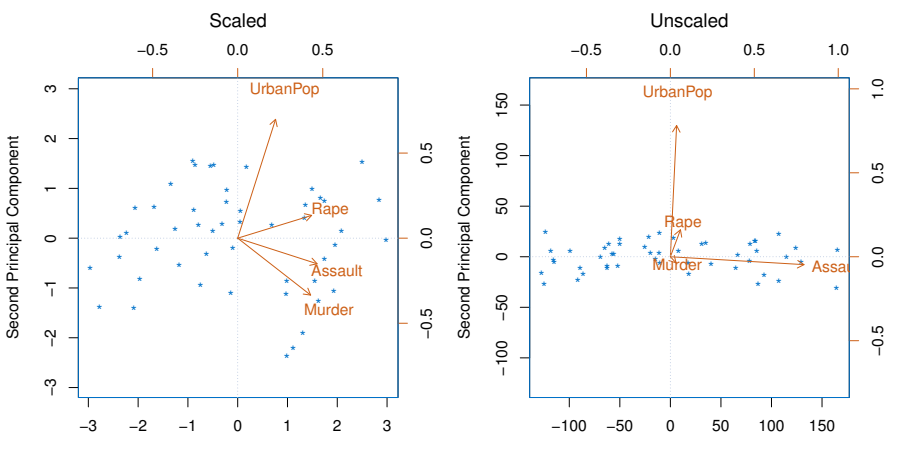


Figure. PCA with and without Feature Scaling

Proportion Variance Explained

- The total variance present in a data set is defined as

$$\sum_{j=1}^{p} \text{Var}(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

- The variance explained by the mth principal component is

$$\text{Var}(Z_m) = \frac{1}{n} \sum_{i=1}^{n} z_{im}^2$$

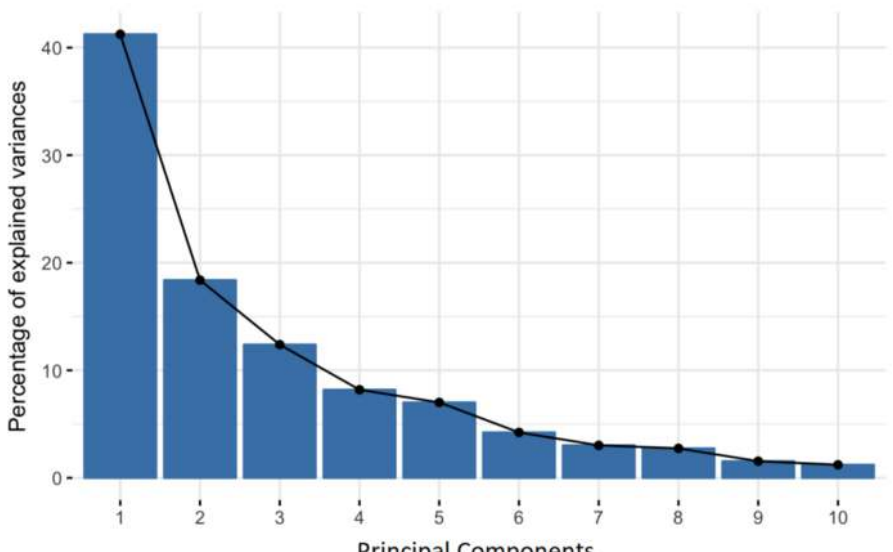
- It can be shown that

$$\sum_{j=1}^{p} \text{Var}(X_j) = \sum_{m=1}^{M} \text{Var}(Z_m)$$

- The proportion of variance explained for the mth principal component is defined as $\text{Var}(Z_m) / \sum_{j=1}^{p} \text{Var}(X_j) \in [0, 1]$

The Number of Principal Components

- Look for an “elbow" of the scree plot
- Validation results



Principal Component Regression

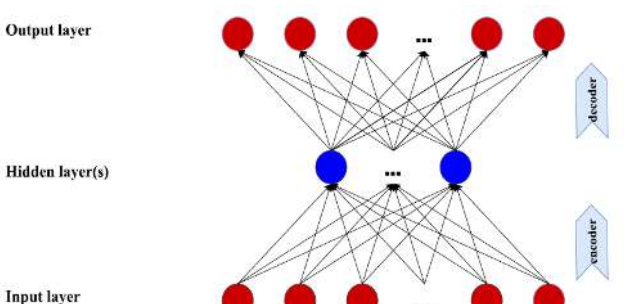
- Suppose you have determined to use E ($E < p$) principal components to represent your data $\mathbf{X}$
- **Principal Component Regression** Fitting a regression model using the E principal components
- Use OLS regression as an example

$$y_i = \beta_0 + \sum_{e=1}^{E} \beta_e z_{ie} + \epsilon_i, \quad i = 1, \ldots, n$$

- PCR loses the interpretability of linear regression models

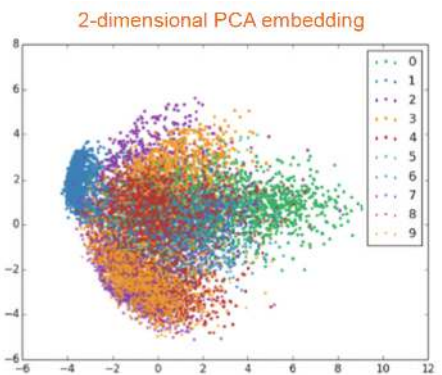
AUTOENCODER

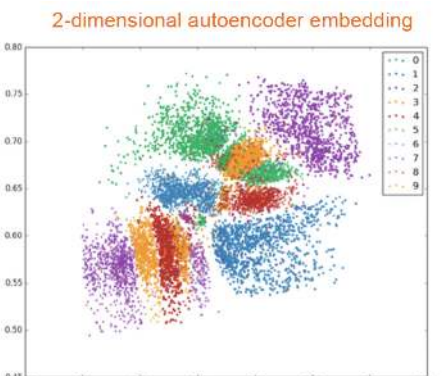
- **Autoencoder** [3] is a network architecture used to learn efficient codings of unlabeled data (unsupervised or self-supervised)
- **Encoder** Take the input and compress it into a low-dimensional vector (the embedding)
- **Decoder** Invert the computation of the first half of the network and reconstruct the original input
- The autoencoders are capable of creating sparse representations of the input data and therefore are used in image compression, denoising



[3] Simple Explanation of Autoencoder: https://youtu.be/3jmcHZq3A5s?si=8TCwlap0_mptxwKh

EXAMPLE: HANDWRITTEN DIGITS





The autoencoder separates the hand-written digits much more clearly

Dimensionality Reduction in Asset Pricing

- **Factor Zoo** The proliferation of a large number of factors identified in empirical research
  - Which factors really provide independent information about average returns? Which are subsumed by others?
  - How many of these new factors are really important?

- Finding the common risk factors that explain the cross-section returns from the factor zoo is a dimensionality reduction problem

Instrumented Principal Complement Analysis

Instrumented Principal Complement Analysis (IPCA) is a latent factor asset pricing model, which incorporates vast conditioning information into factor model estimates[4]

$$r_{i,t+1} = \beta(z_{i,t})' f_{t+1} + \mu_{i,t+1}$$
$$\beta(z_{i,t}) = z'_{i,t}\Gamma + \nu_{i,t}$$

- Let $t = 1, \ldots, T$ index time and $i = 1, \ldots, N$ index firms
- f_t and $\beta(z_{i,t-1})$ are the factors and their loadings ($K \times 1$ vectors)
- z is a $P \times 1$ vector ($P >> K$), containing firm characteristics
- Γ is a (linear) mapping that incorporates the information in the P firm characteristics into K latent factors
- $u_{i,t}$ and $\nu_{i,t}$ are the idiosyncratic errors

[4] Kelly, B. T., Pruitt, S., & Su, Y. (2019). Characteristics are covariances: A unified model of risk and return. Journal of Financial Economics.

Optimization Procedure for IPCA

$$\text{IPCA:} \quad r_{i,t+1} = \beta(z_{i,t})' f_{t+1} = z_{i,t}' \Gamma f_{t+1}$$

$$\text{Estimation:} \quad \min_{\Gamma, F} \sum_{t=1}^{T-1} (r_{t+1} - Z_t \Gamma f_{t+1})' (r_{t+1} - Z_t \Gamma f_{t+1})$$

- The optimization does not admit an analytical solution in general, but is speedily solved numerically by an alternating least squares algorithm
- It iterates between minimizing over Γ while holding $\{f_t\}$ fixed, and minimizing over $\{f_t\}$ while holding Γ fixed, until convergence
- Importantly, the two partial optimization sub problems are simple linear regressions

AUTOENCODER ASSET PRICING MODELS

$$r_{i,t+1} = \beta(z_{i,t})' f_{t+1} + \mu_{i,t+1}$$

Can we extend IPCA using autoencoder in order to model factor exposures as a flexible nonlinear function of covariates?

Conditional Autoencoder Model

$$r_{i,t+1} = \beta(z_{i,t})' f_{t+1} + \mu_{i,t+1}$$