

Data Science

Feature Engineering: Feature Reduction Data Reduction



Won Kim
2022



Roadmap: Feature Engineering

- Feature Creation
- Feature Selection
- Feature Reduction



Feature Reduction vs. Feature Selection

- Both reduce the number of features.
- Feature Selection
 - Selects useful features without changing them.
- Feature Reduction
 - Transforms the features.



Feature Reduction Methods

- **Principal Component Analysis**
 - Transform data so that much of the information is concentrated in a small number of features.
- Singular Value Decomposition (will not cover)
- Linear Discriminant Analysis (will not cover)
- Autoencoder (neural network) (will not cover)
- Self-Organizing Map (SOM) (will not cover)



Primary Uses of Principal Component Analysis

- By reducing the number of features to deal with
 - speed up machine learning processing
 - improve data visualization

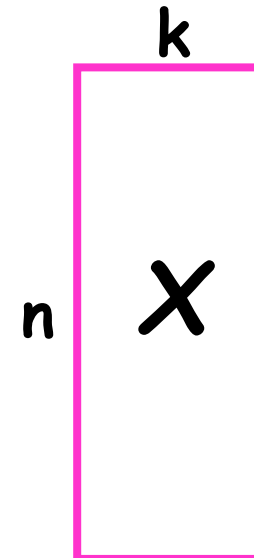
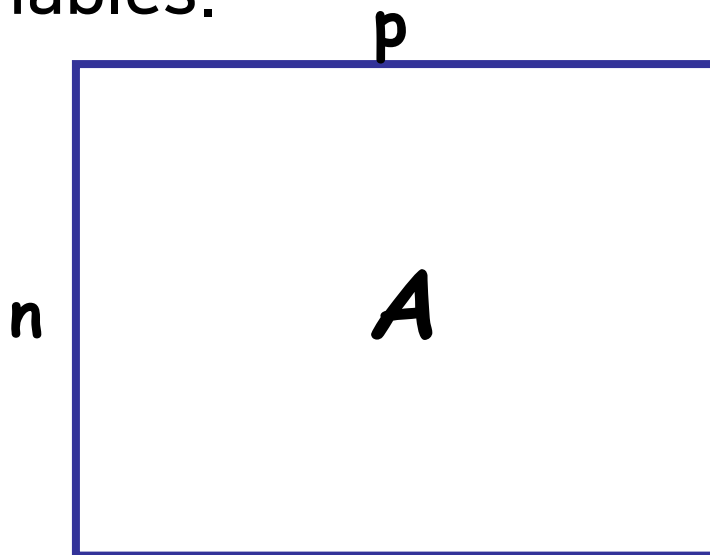


Acknowledgments

- https://web.njit.edu/~usman/courses/cs698_spring18/Principal%20Component%20Analysis.pptx
- <https://www.slideserve.com/rollin/principal-component-analysis-dimensionality-reduction>

Principal Component Analysis (PCA)

- Probably the most widely-used and well-known of the “standard” multivariate methods
- Invented by Karl Pearson (1901) and Harold Hotelling (1933)
- Summarizes data with many (p) variables by a smaller set of (k) derived (synthetic, composite) variables.



Principal Component (PC) (1/3)

- Principal component = (linear algebra) eigenvector
- Eigenvectors are directions where there is the most **variance** (i.e., the data is most spread out).
 - * variance = STD^2
(larger variance = larger area under distribution curve)
- “clumped together” vs. “spread out”

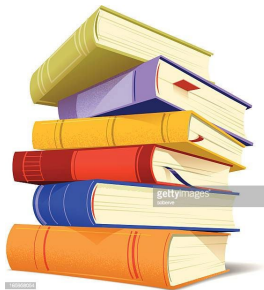


Illustration (1/3)

- Given the following 2-D dataset, how do we find the PC?
- (i.e., find the direction (straight line) in which the 2-D dataset (features x , y) is most spread out.
- The line represents a reduction of 2 features into a single feature.

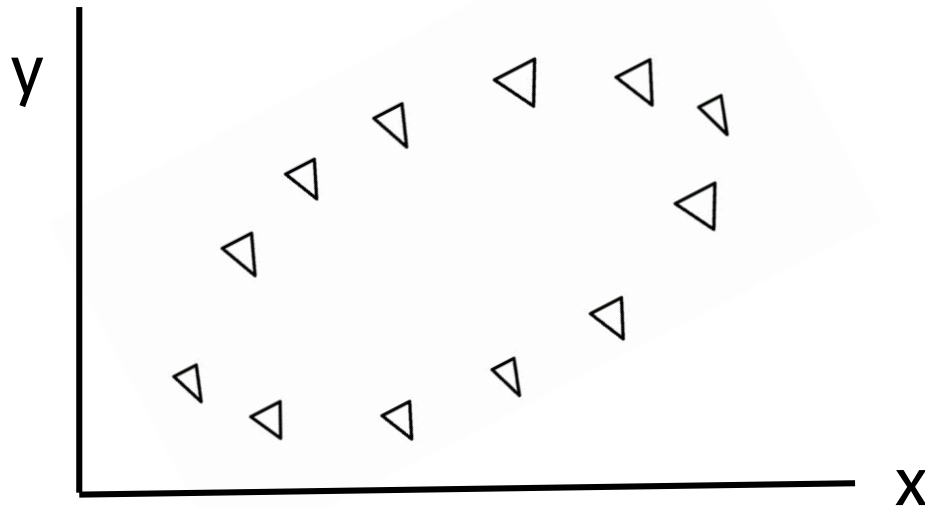


Illustration (2/3)

- Find a straight line where the data is most spread out when projected onto it.
 - It is the horizontal line in the right figure below.
 - It becomes the first Principal Component – the new coordinate system.

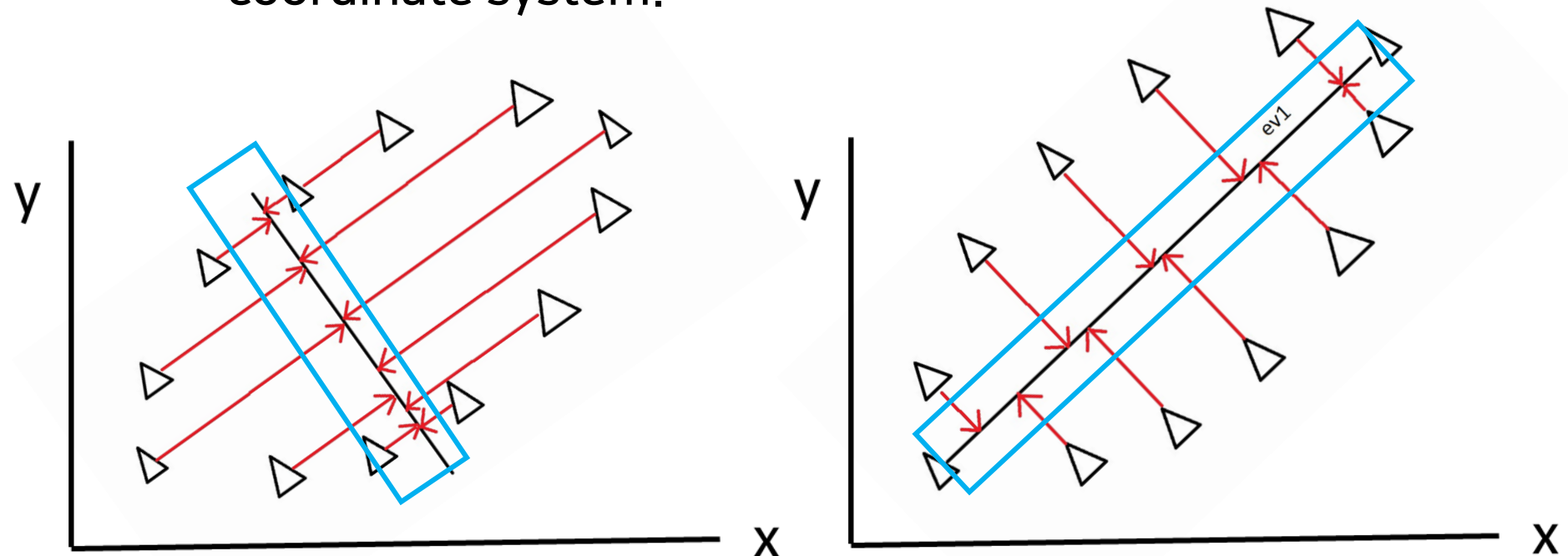
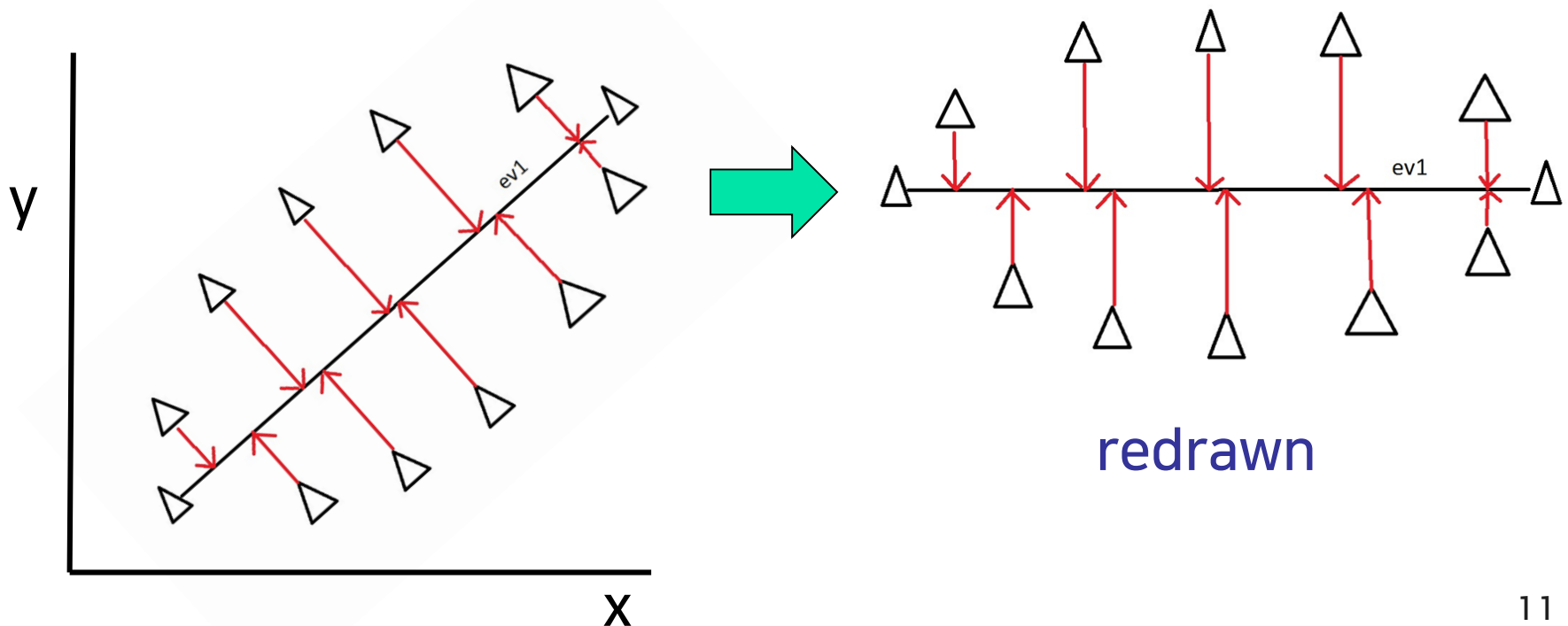


Illustration (3/3)

- ** The straight line (with the direction, called eigenvector) chosen is the **regression line** !!
- The original 2-dimensional dataset becomes 1-dimensional under the new coordinate system.



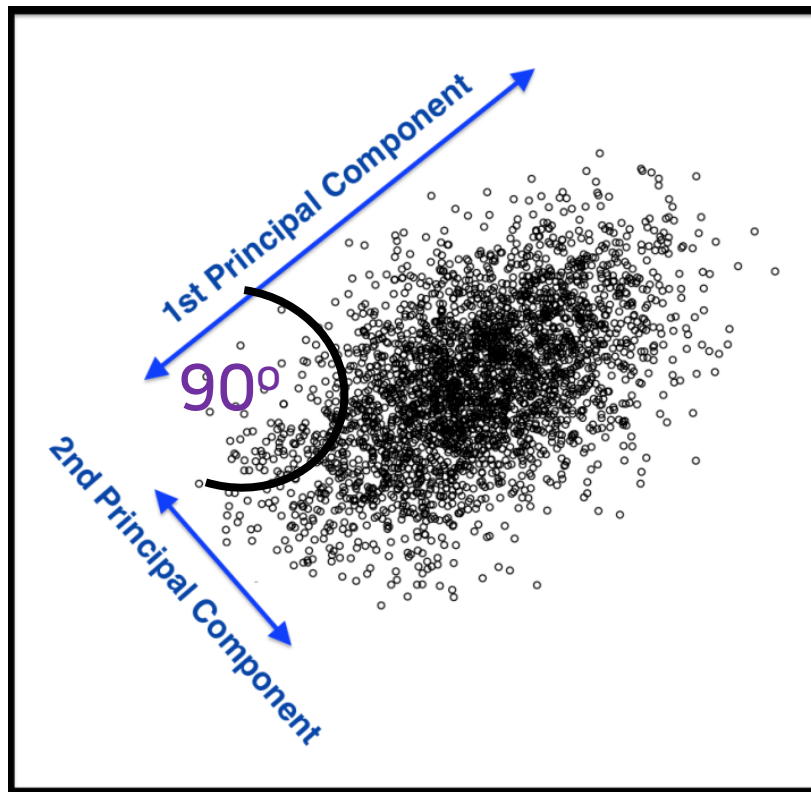


Principal Component (PC) (2/3)

- Each PC represents a reduction of one dimension (feature).
- A Principal Component can be expressed by one or more of existing variables (features).

Principal Component (PC) (3/3)

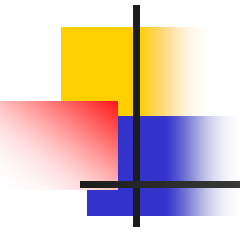
- Each PC is orthogonal (90 degrees) to all other PCs. (just like the x, y, z coordinates)
 - (The PC is a rigid rotation of the original coordinates.)





Math Behind PCA: Eigenvectors and Eigenvalues (to study later)

- The principal components are the **eigenvectors** of the **covariance matrix** of the original dataset.
- Because the covariance matrix is symmetric, the eigenvectors are **orthogonal** (90 degrees to each other).
- The principal components (eigenvectors) correspond to the **direction** (in the original n-dimensional space) with the **greatest variance** (maximum spread-out) in the data.
- Each eigenvector has a corresponding **eigenvalue**. An eigenvalue is a scalar.
- Eigenvector corresponds to a direction, and corresponding eigenvalue indicates how much **variance** there is in the data along that eigenvector.
- In other words, a **larger eigenvalue** means that the principal component explains a **larger variance** in the data.



An Intuitive Perspective



Identifying Variables – Illustration (1/5)

- <https://algobeans.com/2016/06/15/principal-component-analysis-tutorial/>
- Let us differentiate (group) food items.
- Let us use just one variable – **vitamin C level**.
- This will differentiate meat and vegetables. But different meat items will be clumped together.
- To spread out the meat items, let us add a second variable – **fat level**.



Identifying Variables – Illustration (2/5)

- Since **vitamin C level** and **fat level** are measured in different units, we need to normalize (scale) them.
- This will allow us to combine the two variables into a new variable – “**vitamin C minus fat**”.
 - (This new variable is a principal component. The ‘minus’ thing is about a correlation weight, to be discussed a little later.)

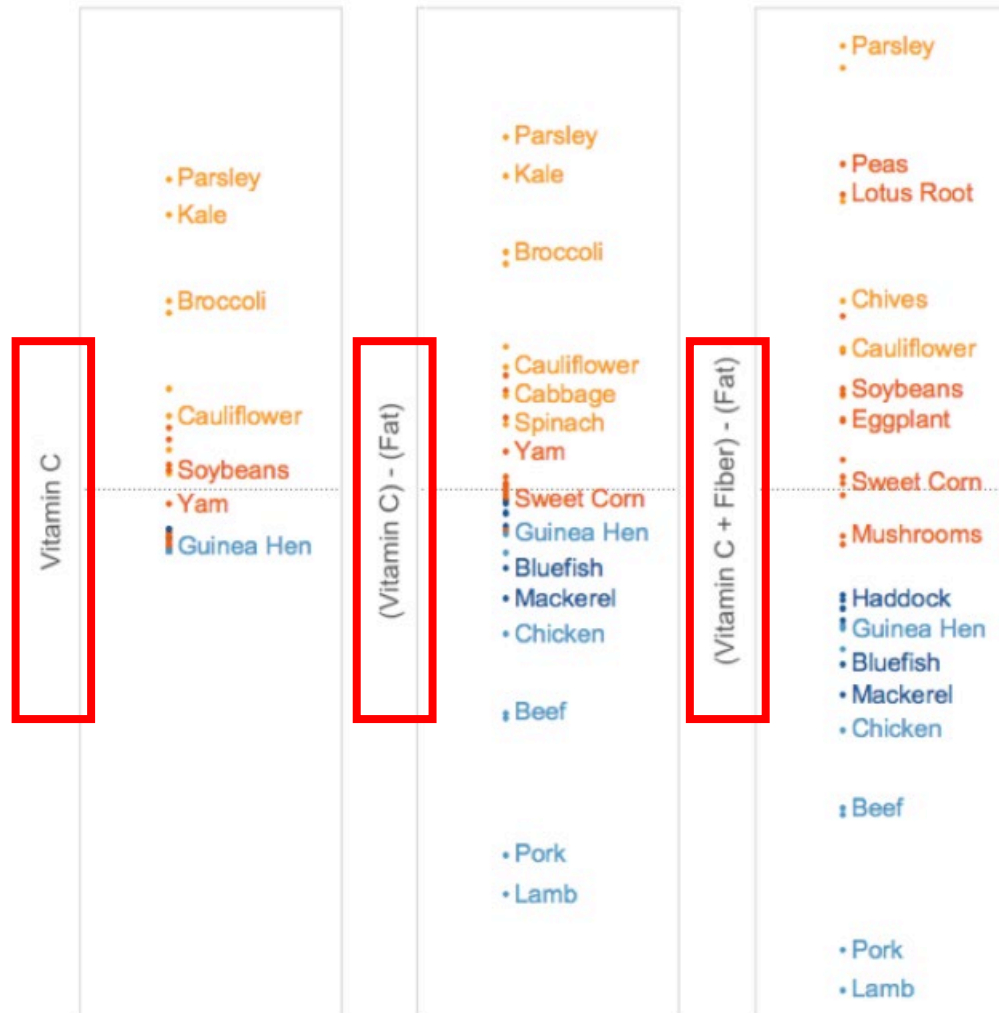


Identifying Variables – Illustration (3/5)

- The data spread can be further improved by adding a third variable – **fiber level**.
- Similarly, this can lead to a second principal component – **“(vitamin C + fiber) minus fat”**

Identifying Variables – Illustration (4/5)

- Shows how the 3 variables spread out data



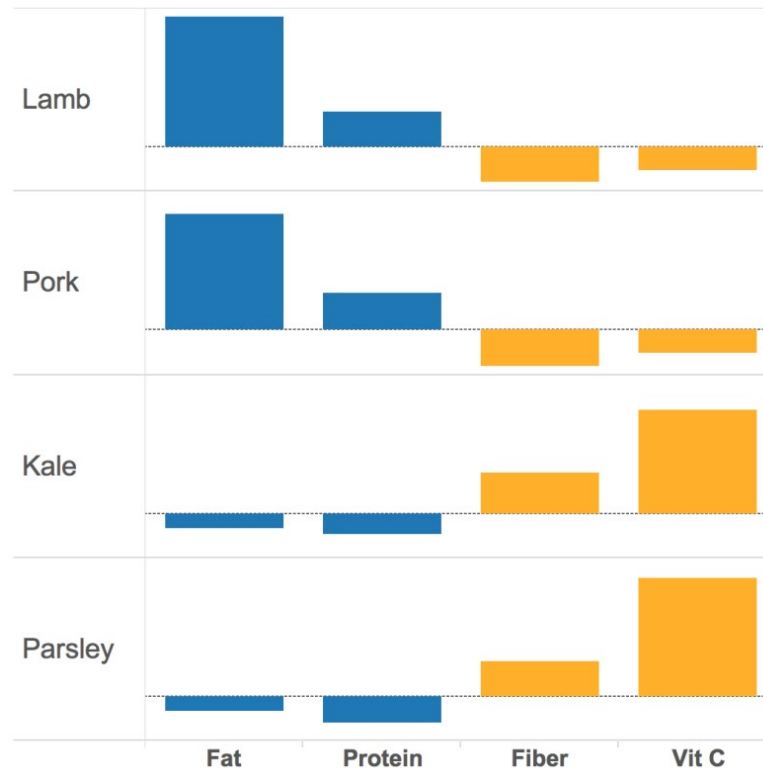


Identifying Variables – Illustration (5/5)

- We have derived the first two principal components for differentiating food items by trial and error.
- PCA does this by systematic computation.

PCA and Correlation among Variables

- PCA is based on the correlation of the variables.
- Example: Correlation among 4 variables (features) for differentiating 4 food items.
- Fat and protein are correlated (i.e., same direction); fiber and vitamin C are correlated.





PCA and Dimension Reduction

- We can combine the **fat** and **protein** variables into one principal component, and **fiber** and **vitamin C** into another principal component.
- We have reduced 4 variables into 2.



PCA Using Python (Scikit-learn)

- <https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60>
- Two Examples
 - 1. show performance improvement
 - 2. show improved visualization



Example 1: Using MNIST Dataset

- The handwritten digit images are contained in `mnist.data`, which has a shape of `(70000, 784)`.
 - 70,000 images, each with 784 features
 - a training set of 60,000 images, and a test set of 10,000 images.
- The labels (the integers 0–9) are contained in `mnist.target`.
 - The features are 784 dimensional and the labels are numbers from 0–9.
 - 784 pixels = 28 x 28

```
from sklearn.datasets import fetch_openml  
mnist = fetch_openml('mnist_784')
```




Split Data into Training and Test Sets, and Standardize Them

```
from sklearn.model_selection import train_test_split
# test_size: what proportion of original data is used
# for test set
train_img, test_img, train_lbl, test_lbl =
train_test_split( mnist.data, mnist.target,
test_size=1/7.0, random_state=0)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
# Fit on training set.
scaler.fit(train_img)
# Apply transform to both the training and test sets
train_img = scaler.transform(train_img)
test_img = scaler.transform(test_img)
```



Import and Apply PCA

```
from sklearn.decomposition import PCA  
# Make an instance of the Model  
pca = PCA(.95)
```

- Note: The code above has **.95** for the **number of components** parameter.
- It means that scikit-learn will choose the minimum number of principal components such that **95% of the variance is retained**. (* We will study this shortly. *)
- Fit PCA on training set.

```
pca.fit(train_img)
```



Apply PCA to the Training and Test Sets

```
train_img = pca.transform(train_img)
test_img = pca.transform(test_img)
```

- Now you can apply some machine learning algorithm to the transformed datasets.
- /* For this example, a logistic regression algorithm was run 5 times, each with a different variance (information) retained.

(* This part is not shown, as machine learning algorithms will be covered later in this course. *)



Performance Result

- Below are the machine learning modeling performance results.
- Note the variance retained, number of components calculated, running time, and accuracy for the 5 different runs.

Variance Retained	Number of Components	Time (seconds)	Accuracy
1.00	784	48.94	0.9158
0.99	541	34.69	0.9169
0.95	330	13.89	0.9200
0.90	236	10.56	0.9168
0.85	184	8.85	0.9156

Example 2: Using the IRIS Dataset

- The Iris Dataset (3 types of iris)

```
import pandas as pd
```

```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
```

```
# load dataset into Pandas DataFrame
```

```
df = pd.read_csv(url, names=['sepal length','sepal width','petal length','petal width','target'])
```

iris setosa



petal

sepal

iris versicolor



petal

sepal

iris virginica



petal

sepal



The Iris Dataset

- 150 rows, and 5 features (a toy dataset)
- 50 rows for each of the 3 types of iris

	sepal length	sepal width	petal length	petal width	target
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa



Standardize the Dataset

```
from sklearn.preprocessing import StandardScaler
```

```
features = ['sepal length', 'sepal width', 'petal length',  
'petal width']
```

```
# Separate out the features  
x = df.loc[:, features].values
```

```
# Separate out the target  
y = df.loc[:, ['target']].values
```

```
# Standardize the features  
x = StandardScaler().fit_transform(x)
```



Scaling Result

	sepal length	sepal width	petal length	petal width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Standardization



	sepal length	sepal width	petal length	petal width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977

PCA Projection to 2D

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data =
    principalComponents, columns =
    ['principal component 1', 'principal component 2'])
```

	sepal length	sepal width	petal length	petal width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977

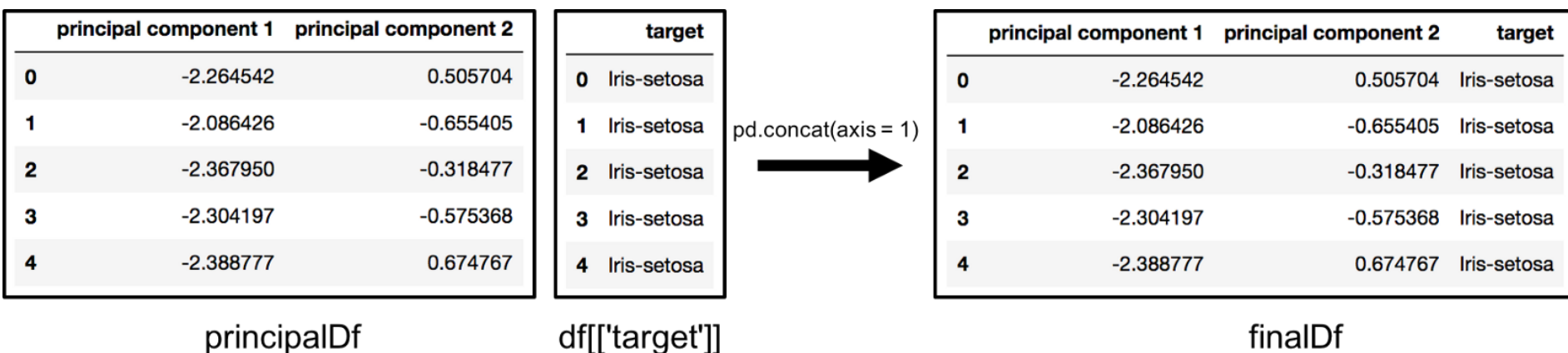
PCA
(2 components)



	principal component 1	principal component 2
0	-2.264542	0.505704
1	-2.086426	-0.655405
2	-2.367950	-0.318477
3	-2.304197	-0.575368
4	-2.388777	0.674767

Concatenate DataFrame along Axis=1 (for plotting)

```
finalDf = pd.concat([principalDf, df[['target']], axis = 1)
```

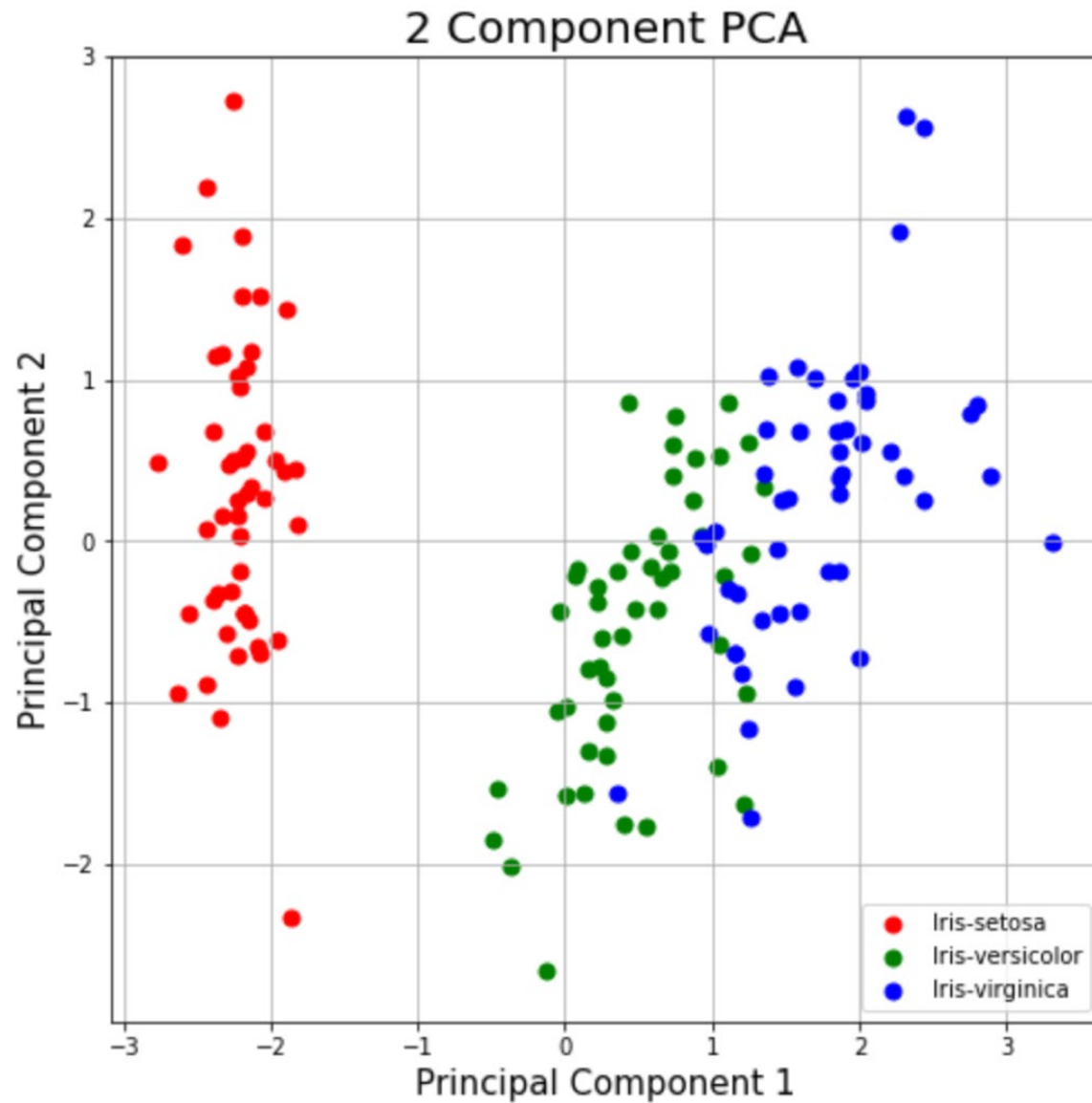




Code to Plot the Graph

```
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('2 component PCA', fontsize = 20)
targets = ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
colors = ['r', 'g', 'b']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['target'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1'],
               finalDf.loc[indicesToKeep, 'principal component 2'],
               c = color, s = 50)
ax.legend(targets)
ax.grid()
```

Result





Explained Variance Ratio (1/2)

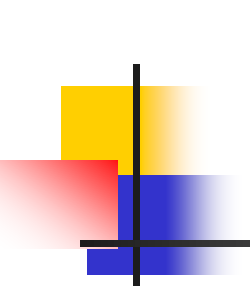
- The **explained variance ratio** tells you how much information (i.e., variance) can be attributed to each of the principal components.
- This is important, because you lose some of the variance (information) when you convert, e.g., 4 dimensional space to 2 dimensional space.



Explained Variance Ratio (2/2)

`pca.explained_variance_ratio_`

- By using the PCA class' attribute `explained_variance_ratio_`, you can see, for example, that the first principal component contains 72.77% of the variance and the second principal component contains 23.03% of the variance.
- Together, the two components contain 95.80% of the information.

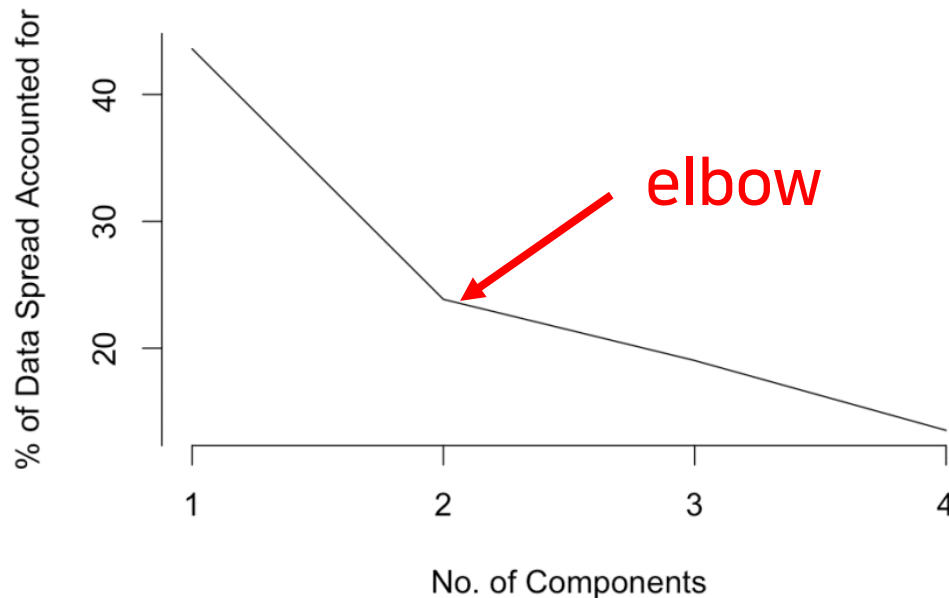


Choosing the Number of Principal Components (1/2)

- Principal components are ordered by their effectiveness in differentiating data points,
 - with the first principal component being most effective, and so on.
- To keep the results simple and generalizable, usually only the first few principal components are selected for visualization and further analysis.

Choosing the Number of Principal Components (2/2)

- The number of principal components to consider is determined by a *scree plot* (shown below).
- It shows decreasing effectiveness of subsequent principal components in differentiating data points.
- A rule of thumb is to use the number of principal components corresponding to the location of an elbow.



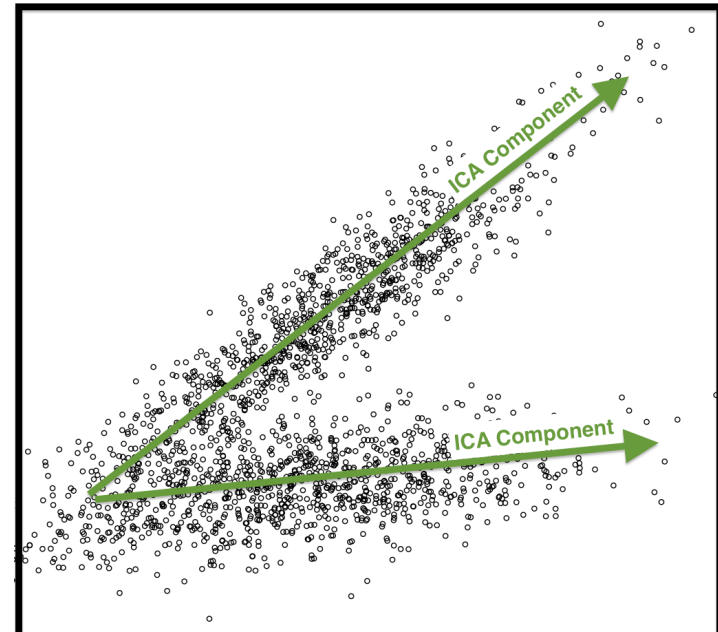
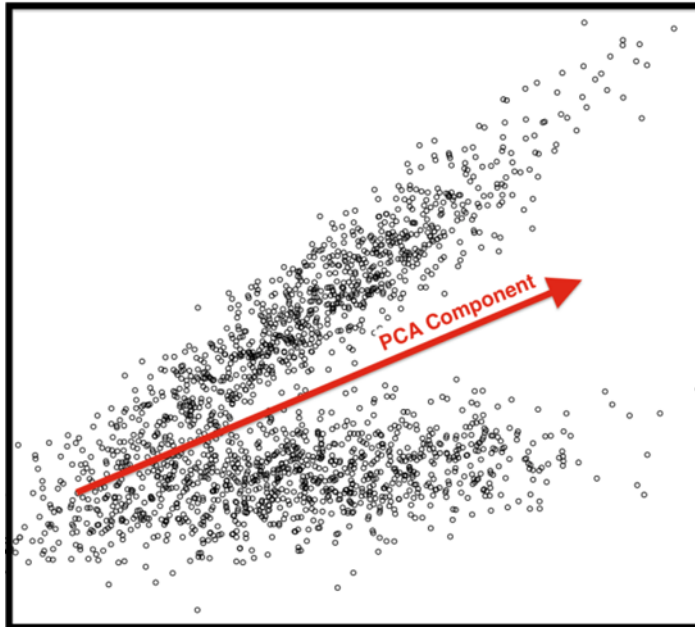


Limitations of PCA

- Maximizing the “spread” of data points sometimes is not appropriate.
- Often it is difficult to explain the combination of variables in a principal component.
- Principal components must be orthogonal; that is, they must not overlap in space (positioned 90 degrees to each other). This condition cannot always be met.

PCA and ICA (as backup to PCA)

- An alternative technique called Independent Component Analysis (ICA) is used to solve this problem.
- ICA allows its components to *overlap in space*, thus they do not need to be orthogonal. However, ICA forbids its components to *overlap in the information* they contain, aiming to reduce mutual information shared between components.





Homework

- Download the California Housing Prices dataset, and reduce using PCA.
- Submit the code and the analysis results in a single WORD file.



Roadmap: Data Preprocessing

- Data Restructuring
- Data Value Changes
- Feature Engineering
- Data Reduction



Data Reduction

- Feature Selection/Reduction
- Data Filtering
- Data Summarization
- Concept Hierarchy
- Data Compression
- Numerosity Reduction



Example Table

Name	Age	Job	Salary	Car	Hobby
Peter	25	engineer	4500	Avante	K-pop
Paul	30	marketing	3000	Matiz	game
Mary	35	marketing	5000	Grandeur	music
Neil	27	SW engineer	6000	BMW	golf



Feature Selection

- Eliminate columns/features unlikely to be useful in the analysis (e.g., Name and Job columns)

Name	Age	Job	Salary	Car	Hobby
Peter	25	engineer	4500	Avante	K-pop
Paul	30	marketing	3000	Matiz	game
Mary	35	marketing	5000	Grandeur	music
Neil	27	SW engineer	6000	BMW	golf



Data Filtering

- Eliminate data not relevant or unlikely to be useful in the analysis
- (e.g., “below age 30”)

Name	Age	Job	Salary	Car	Hobby
Peter	25	engineer	4500	Avante	K-pop
Paul	30	marketing	3000	Matiz	game
Mary	35	marketing	5000	Grandeur	music
Neil	27	SW engineer	6000	BMW	golf



Data Summarization

- Instead of detailed data, use only summarized data.
- Examples
 - Upon clustering, only the key information about each cluster may be used for further analysis (and all detailed data may be discarded)
 - Upon binning or creating a histogram, only the key information about the bins or the histogram, and the number of data contained in it may be used for further analysis



Concept Hierarchy

- When data forms a conceptual hierarchy, replace low level concepts with higher level concepts.
- (This is similar to Data Reduction through Data Summarization.)
- Example
 - In an information hierarchy for a company organized as Company – Department – Section, a Department-level analysis may suffice, rather than a Section-level analysis.



Data Compression

- Use data encoding or transformation to compress original data
- Lossless Compression
 - Original data can be reconstructed
- Lossy Compression
 - Only an approximation of the original data can be constructed
 - Wavelet transforms, and principal component analysis (PCA)



Lossless Compression Methods

- Run-length encoding
 - used in PCX, BMP, TGA, TIFF
- Area image compression
- DPCM and predictive coding
- Entropy encoding
- Adaptive dictionary algorithms, such as LZW
 - used in GIF and TIFF
- DEFLATE
 - used in PNG, MNG, and TIFF
- Chain codes



Lossy Compression Methods

- Reducing the color space to the most common colors in the image
- Chroma subsampling
- Transform coding
 - Fourier-related transform, such as the Discrete Cosine Transform (DCT)
 - wavelet transform, quantization, entropy coding
- Fractal compression.



Numerosity Reduction

- numerosity means 'a large number'
- Reduce data volume by choosing alternative, smaller data representations
- Two methods
 - parametric methods
 - non-parametric methods



Roadmap: Numerosity Reduction

- Parametric Methods
- Non-Parametric Methods

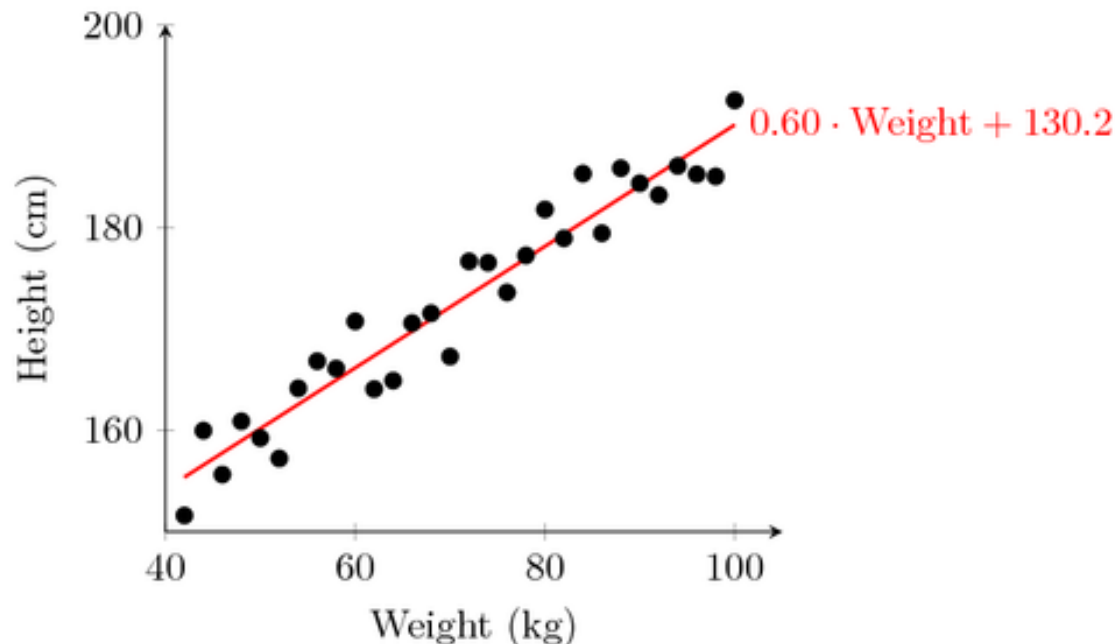


Parametric Methods

- Use a model to fit data, and store only the model parameters (not the original data).
- Example Models
 - regression models, log-linear models, Gaussian distribution models,...

Example: Linear Regression Model

- In the example below, given the dataset (weight, height) pairs, a linear regression algorithm determines that the best function is $\text{height} = 130.2 + 0.6 \cdot \text{weight}$.
- Keep only 130.2 and 0.6; and discard the dataset.





Roadmap: Numerosity Reduction

- Parametric Methods
- Non-Parametric Methods



Non-Parametric Methods

- Do not use models (model parameters) to fit data.
- Techniques
 - Discretization (or binning)
 - Clustering
 - Sampling



Roadmap: Non-Parametric Methods

- Discretization
- Clustering
- Sampling

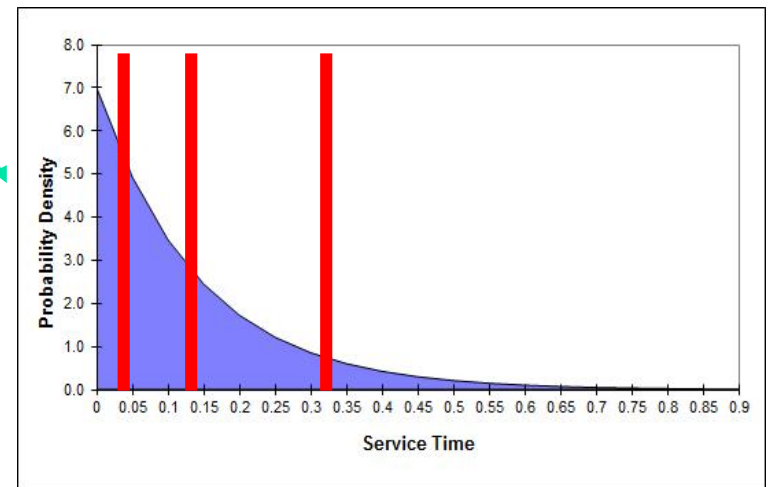
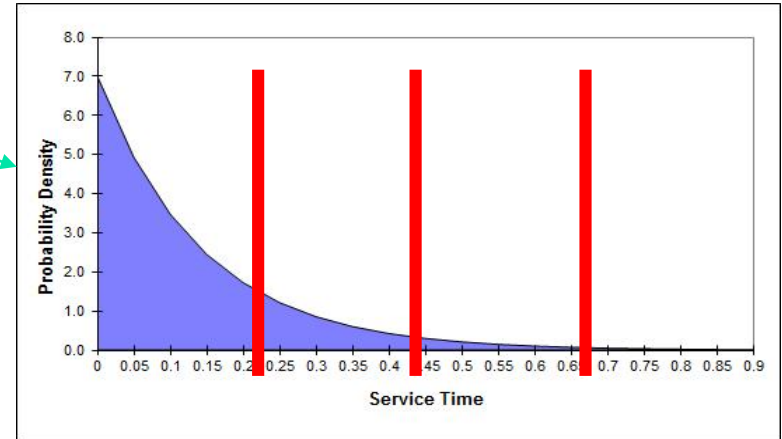


Discretization Methods (Binning)

- Reduce the number of values for a given continuous attribute (feature) by dividing the range of the attribute into intervals.
- Keep only the information for the bins (and drop the original data)

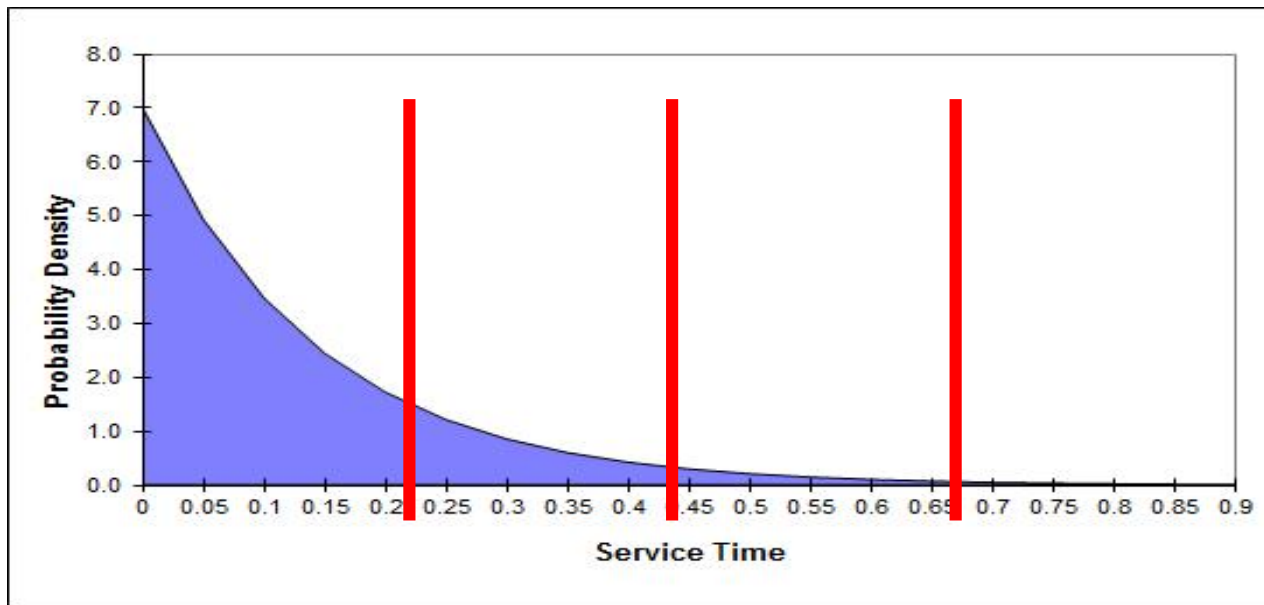
How to Split the Attribute Range

- Equal width bins
- Equal frequency bins
- Entropy based
 - Split based on information gain



Equal-Width (Distance) Partitioning

- Divides the range into N intervals of equal size (uniform grid).
- If A and B are the min and max values of the attribute, the width of intervals is $W = (B-A)/N$.
- Simple, but outliers may dominate.
- Does not handle skewed data well.





Binning for Data Smoothing

- Binning method is used to smoothing data or to handle noisy data. In this method, the data is first sorted and then the sorted values are distributed into a number of bins.
- There are three approaches to perform smoothing;
 - Smoothing by bin means : Each value in a bin is replaced by the mean value of the bin.
 - Smoothing by bin median : Each bin value is replaced by its bin median value.
 - Smoothing by bin boundary : The minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value.



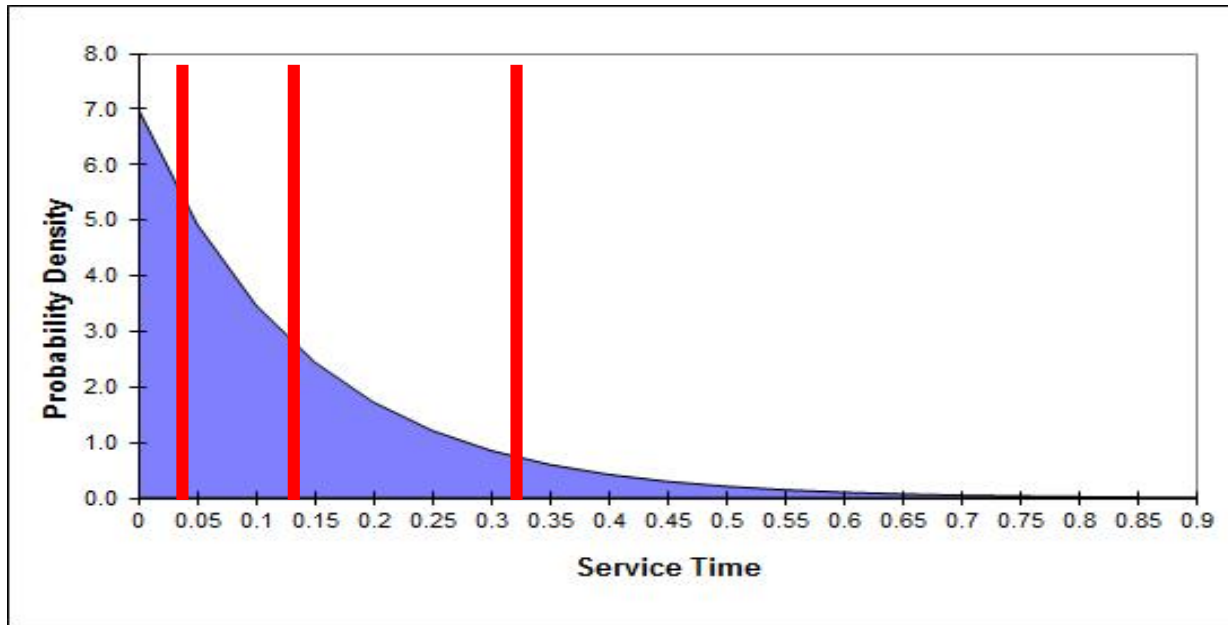
Illustration (exercise: confirm)

Sorted 12 data for prices (in dollars):

4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

- Partition into 3 (equi-width) bins: $12/4 = 3$ bins
 - Bin 1: 4, 8, 9, 15 (mean 9)
 - Bin 2: 21, 21, 24, 25 (mean 23)
 - Bin 3: 26, 28, 29, 34 (mean 29)
- Smoothing by bin means:
 - Bin 1: 9, 9, 9, 9
 - Bin 2: 23, 23, 23, 23
 - Bin 3: 29, 29, 29, 29
- Smoothing by bin boundaries: [4,15],[21,25],[26,34]
 - Bin 1: 4, 4, 4, 15
 - Bin 2: 21, 21, 25, 25
 - Bin 3: 26, 26, 26, 34

Equal Frequency (Depth) Partitioning



- Divides the range into N intervals, each containing about the same number of values.
- Good data scaling, and handles skewed data well
- But can have trouble with categorical attributes
 - (e.g., shirt size (S, M, L, XL))



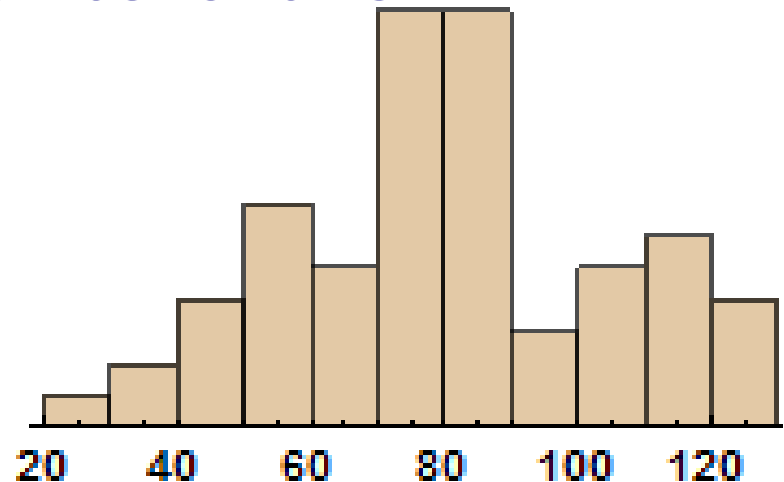
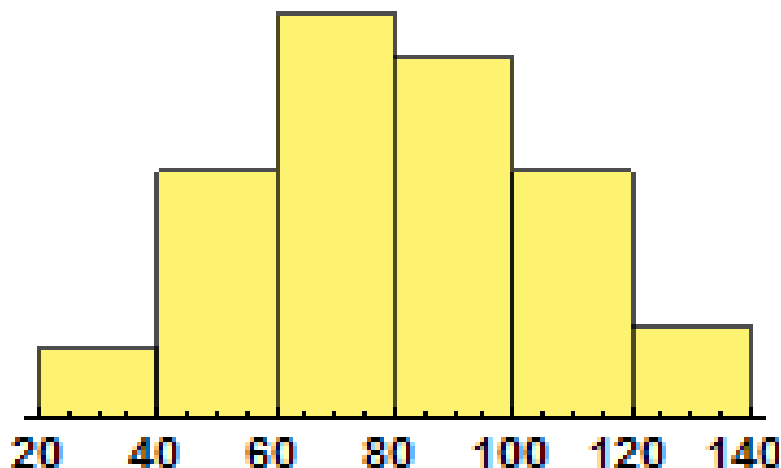
Entropy-based Partitioning

- method used by decision trees (to study later in this course)
- The approach works by finding the split with the maximal information gain (reduction in entropy) with respect to the target variable.
- Intuitively, it finds the best split so that the majority of the values in a bin correspond to the same class label.

Selecting the Number of Bins

- Different numbers of bins can make data distribution look different when sample sizes are small
- Some techniques used to select a good number of bins
 - Sturge's rule, Freedman-Diaconis rule
 - <https://medium.datadriveninvestor.com/how-to-decide-on-the-number-of-bins-of-a-histogram-3c36dc5b1cd8>

Same data, different number of bins



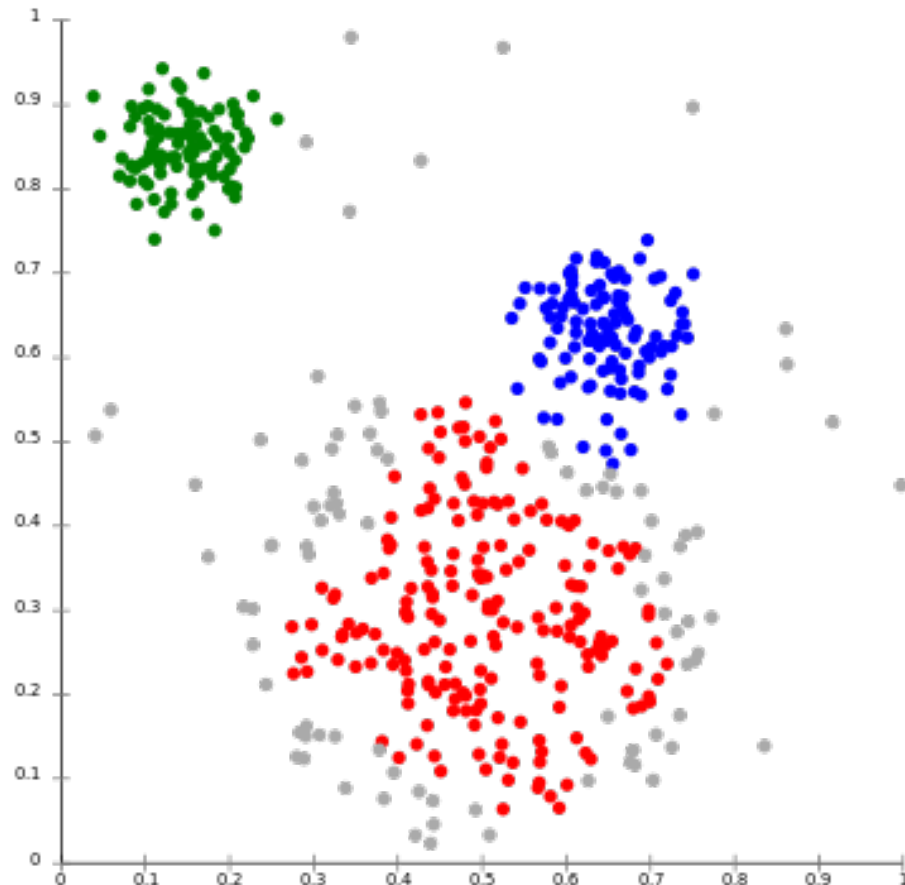


Roadmap: Non-Parametric Methods

- Discretization
- Clustering
- Sampling

Example Result of Clustering

- (* will learn two clustering algorithms in this course.*)





Data Reduction Using Clustering

- Using some clustering algorithm, create clusters of data
- Keep only the information for the cluster representations (and drop the original data)
- Keep only the data that belong to some of the clusters (and drop all the data that belong to all other clusters)



Roadmap: Non-Parametric Methods

- Discretization
- Clustering
- Sampling

Random Sampling

- Take a much smaller sample of the original data.
- Prediction using a sample data is just as good as using the original much larger data.
- Often, it is not possible to collect all original data, and only samples can be used.
- Sampling has been widely used in marketing, election, etc.





Sampling Techniques

- Simple random sampling
 - without replacement (same data can be used only once)
 - with replacement (same data can be used more than once)
- Systematic sampling
 - Select every k^{th} element: $k = \text{population size} / \text{sample size}$
- Stratified random sampling
 - Divide a population of data into strata (subgroups), and select samples randomly from each stratum
- Probability-Proportional-to-Size random sampling
 - Select samples in proportion to the size of the group
- ...



Exercise: Sampling

- 1. Do an Internet search to find out how sampling is done to predict the result of an election:
 - number of people selected (% of the population)
 - Demographics (age, gender, income, residence, ...)
 - questions asked
- 2. Suppose you have a database of people who live in Pangyo, and you want to use stratified random sampling in your market research for a food-delivery service.
 - (* You are free to decide on the type of food; and assume that you have a small budget to work with.)



End of Class
