비즈니스를 위한 데이터마이닝

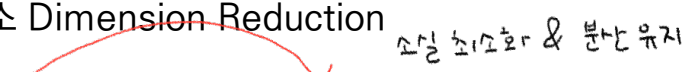
가톨릭대학교 경영학과
이홍주

차원의 저주 Curse of dimensionality

- 기계학습에서 데이터가 수천 심지어 수백만 개의 속성(Feature)을 가지고 있는 경우가 있음
- 많은 속성은 학습을 느리게 할 뿐만 아니라 좋은 모형을 생성하는 것이 어려움
- 고차원 데이터 집합은 데이터간의 거리를 측정하면 많은 차원으로 인해 거리가 멀어지고, 데이터가 공간상에 희박하게 존재하게 됨
- 학습 데이터가 서로 멀리 떨어져 있어서 학습이 어려우며, 새로운 데이터도 학습 데이터와 멀리 떨어져 있을 가능성이 높음
- 차원의 저주 해결책: 학습 데이터의 밀도가 충분히 높아질 때까지 학습 데이터의 크기를 키우는 것
- 일정 밀도에 도달하기 위해 필요한 학습 데이터는 차원 수가 커짐에 따라 기하급수적으로 늘어남

데이터 요약과 차원 축소

- 데이터 요약 Data Summarization
  - 데이터 요약은 데이터 탐색의 중요한 구성 요소
  - 요약 통계(평균, 중앙값 등) 및 시각화를 통한 요약
  - 기초 통계: 평균, 중앙값, 최소값, 최댓값, 표준편차, 개수 및 백분율
- 차원 축소 Dimension Reduction
  - 고차원의 원본 데이터를 저차원의 부분 공간으로 투영하여 데이터를 축소하는 방법
  - 정확도의 희생을 최소로 하여 독립 변수 또는 입력 변수의 차원을 축소하는 방법을 찾는 것
  - 요인 선택(factor selection) 또는 특징 추출(feature extraction)
  - 데이터의 정보를 더 작은 하위 집합으로 압축하는 데 유용
  - 유사한 범주를 결합하여 범주형 변수를 줄일 수 있음



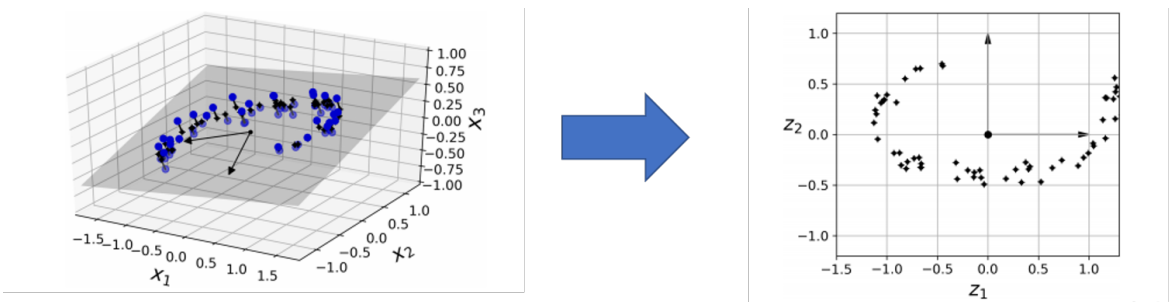
차원 축소 방법

p.120

- 주어진 데이터에 도메인 지식을 적용해 범주를 제거하거나 결합하기
  - 파악!
- 데이터 요약을 사용해 변수 간 중복 정보를 검출하고 불필요한 변수 및 범주를 제거하거나 합치기
- 데이터 변환 기술을 사용해 범주형 변수를 수치형 변수로 변환하기
  - column 만들어주니까~!
- 주성분 분석(PCA) 같은 자동화된 차원 축소 기술을 사용하기
  - cf. 히스토그램
  - 주성분 분석은 원래 수치 데이터셋을 더 적은 변수에 대부분의 원래 정보를 포함하는 원래 데이터의 더 작은 가중 평균 집합으로 변환

차원 축소: 투영

- 대부분의 데이터는 모든 차원에 걸쳐 균일하게 분포하지 않음
- 대부분의 속성은 거의 변화가 없는 반면, 몇몇 속성들은 서로 강하게 연관되어 있는 경우가 많음
- 데이터가 고차원 공간 안의 특정 저차원 부분 공간에 놓여 있을 수 있음
- 모든 데이터를 이 부분 공간에 투영



(출처: 핸즈온 머신러닝)

데이터 요약

- 기초 통계: 평균, 중앙값, 최소값, 최댓값, 표준편차, 개수 및 백분율

```
bostonHousing_df = pd.read_csv('BostonHousing.csv')
bostonHousing_df = bostonHousing_df.rename(columns={'CAT.
      MEDV': 'CAT_MEDV'})

# Compute mean, standard dev., min, max, median, length,
  and missing values for all variables
pd.DataFrame({'mean': bostonHousing_df.mean(),
    'sd': bostonHousing_df.std(),
    'min': bostonHousing_df.min(),
    'max': bostonHousing_df.max(),
    'median': bostonHousing_df.median(),
    'length': len(bostonHousing_df),
    'miss.val': bostonHousing_df.isnull().sum(),
})
```

데이터 요약

- 상관계수
  - 변수의 중복 탐지에 좋음
  - 상관 계수 표에 대한 히트맵을 사용해 강한 상관관계가 있는 변수들을 쉽게 식별 가능 기초 통계
- 범주 개수

```
> bostonHousing_df.corr().round(2)
         CRIM    ZN  INDUS  CHAS   NOX    RM   AGE   DIS
CRIM     1.00 -0.20   0.41 -0.06  0.42 -0.22  0.35 -0.38
ZN      -0.20  1.00  -0.53 -0.04 -0.52  0.31 -0.57  0.66
INDUS    0.41 -0.53   1.00  0.06  0.76 -0.39  0.64 -0.71
CHAS    -0.06 -0.04   0.06  1.00  0.09  0.09  0.09 -0.10
NOX      0.42 -0.52   0.76  0.09  1.00 -0.30  0.73 -0.77
RM      -0.22  0.31  -0.39  0.09 -0.30  1.00 -0.24  0.21
AGE      0.35 -0.57   0.64  0.09  0.73 -0.24  1.00 -0.75
DIS     -0.38  0.66  -0.71 -0.10 -0.77  0.21 -0.75  1.00
```

> bostonHousing_df.CHAS.value_counts()

0 471
1 35

35개 지역은 CHAS 값이 "1"(찰스강 경계에 인접)

Name: CHAS, dtype: int64

데이터 요약

- 그룹화 Groupby

```
# Create bins of size 1 for variable using the method pd.cut. Default
# creates a categorical variable, e.g. (6,7]. labels=False determines
# integers instead, e.g. 6.
bostonHousing_df['RM_bin'] = pd.cut(bostonHousing_df.RM,
   range(0, 10), labels=False)

bostonHousing_df.groupby(['RM_bin', 'CHAS'])['MEDV'].mean()
```

방 숫자

RM_bin	CHAS	
3	0	25.300000
4	0	15.407143
5	0	17.200000
	1	22.218182
6	0	21.769170
	1	25.918750
7	0	35.964444
	1	44.066667
8	0	45.700000
	1	35.950000

In neighborhoods where houses averaged 3 rooms and did not border the Charles, median value was 25.3 ($000)

데이터 요약

- Pivot table

```
import pandas as pd

data = pd.DataFrame({
    '지역': ['서울', '서울', '부산', '부산','서
    울','부산'],
    '제품': ['사과', '바나나', '사과', '바나나',
    '사과', '바나나'],
    '판매량': [30, 20, 25, 15, 20, 25]
})

pivot = pd.pivot_table(data, index='지역',
    columns='제품', values='판매량',
    aggfunc='sum')
print(pivot)
```

지역	제품	판매량
서울	사과	30
서울	바나나	20
부산	사과	25
부산	바나나	15
서울	사과	20
부산	바나나	25

	사과	바나나
서울	50	20
부산	25	40

인덱스 & 컬럼 설정

데이터 요약

```
# use pivot_table() to reshape data and generate pivot table
pd.pivot_table(bostonHousing_df, values='MEDV',
   index=['RM_bin'],
  columns=['CHAS'], aggfunc=np.mean, margins=True)
```

행/열에 총합 추가

CHAS	0	1	All
RM_bin			
3	25.300000	NaN	25.300000
4	15.407143	NaN	15.407143
5	17.200000	22.218182	17.551592
6	21.769170	25.918750	22.015985
7	35.964444	44.066667	36.917647
8	45.700000	35.950000	44.200000
All	22.093843	28.440000	22.532806

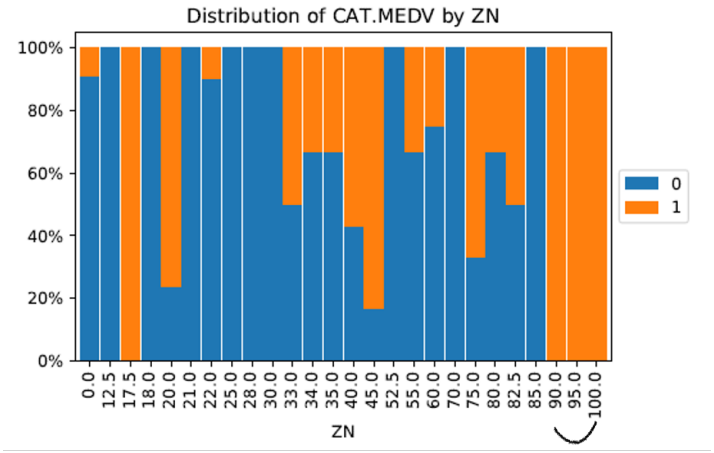
찰스강과 접하지 않는 방이 8개인 지역은 MEDV = 45.7

범주 축소

- 범주형 변수 하나에 m개의 범주가 있는 경우, 일반적으로 이를 m개 또는 m−1개의 더미 변수(dummy variables)로 변환한다
- 각 더미 변수는 값 0 또는 1을 가짐
  - 0 = 해당 범주가 아님(“no”)
  - 1 = 해당 범주임(“yes”)
- 문제: 더미 변수가 너무 많아질 수 있음
- 해결책: 서로 비슷한 범주를 결합해 변수 수를 줄임

범주 축소

- X축 ZN, Y축 CAT.MEDV
- ZN 범주 중 CAT.MEDV가 모두 0이거나 1인 범주들이 존재
- 해당 범주들을 합칠 수 있음



범주 축소

```
# use method crosstab to create a cross-tabulation of two
    variables
tbl = pd.crosstab(bostonHousing_df.CAT_MEDV,
   bostonHousing_df.ZN)

# convert numbers to ratios
propTbl = tbl / tbl.sum()
propTbl.round(2)

# plot the ratios in a stacked bar chart
ax = propTbl.transpose().plot(kind='bar', stacked=True)
ax.set_yticklabels(['{:,.0\%}'.format(x) for x in ax.get_yticks()])
plt.title('Distribution of CAT.MEDV by ZN')
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()
```

범주 축소

- 분기별 Toys R Us의 수입
- 1,2,3분기와 4분기가 다른 패턴일 보임
- 1,2,3분기를 하나의 범주로 합칠 수 있음

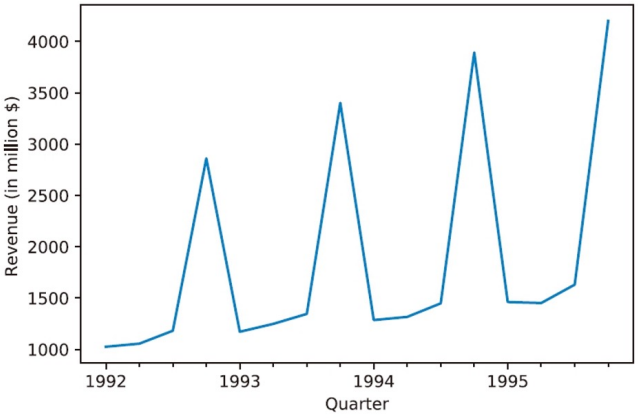


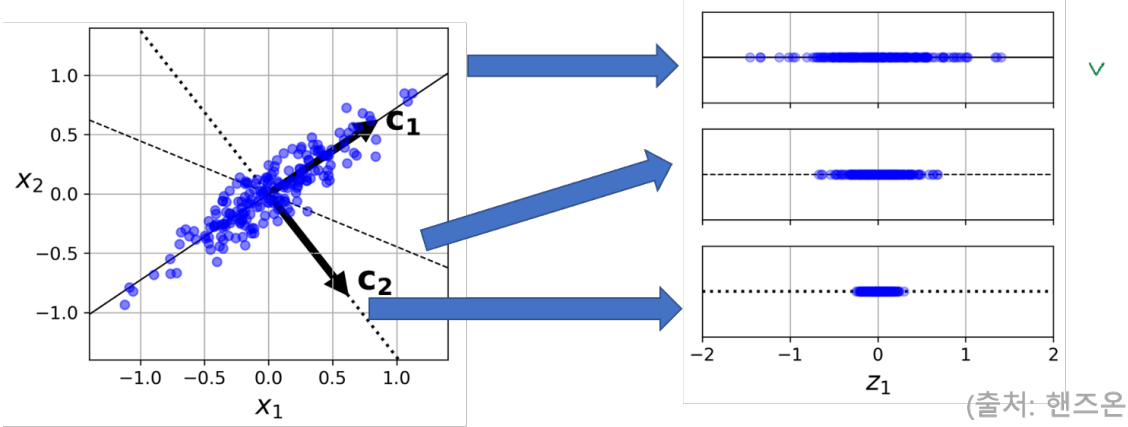
그림 4-2 1992년부터 1995년까지 미국 토이저러스(Toys R Us)의 분기별 수입

범주 축소

- 범주형 변수의 범주들이 구간을 나타내기도 함
  - 예) 나이 (20대, 30대 등), 수입 (5천만원이하, 5천만원이상) 등
- 구간의 중간값을 사용해서 범주를 수치형 변수로 변환할 수 있음
  - 예) 나이(20대 -> 25, 30대 -> 35), 수입 (2천5백원만원, 7천5백만원) 등
- 수치형 변수로 변환되면 더미 변수가 필요 없으며 하나의 컬럼으로만 표현됨

주성분 분석 Principal Component Analysis

- 수치형 변수를 줄이는 방안
- 원 데이터의 분포를 최대한 보존하면서 고차원 공간의 데이터들을 저차원 공간으로 변환하는 기법
  - 데이터가 가지고 있는 분산이 최대로 보존되는 축을 선택하는 것이 정보가 가장 적게 손실됨
  - 원본 데이터와 투영된 것 사이의 평균 제곱 거리를 최소화하는 축



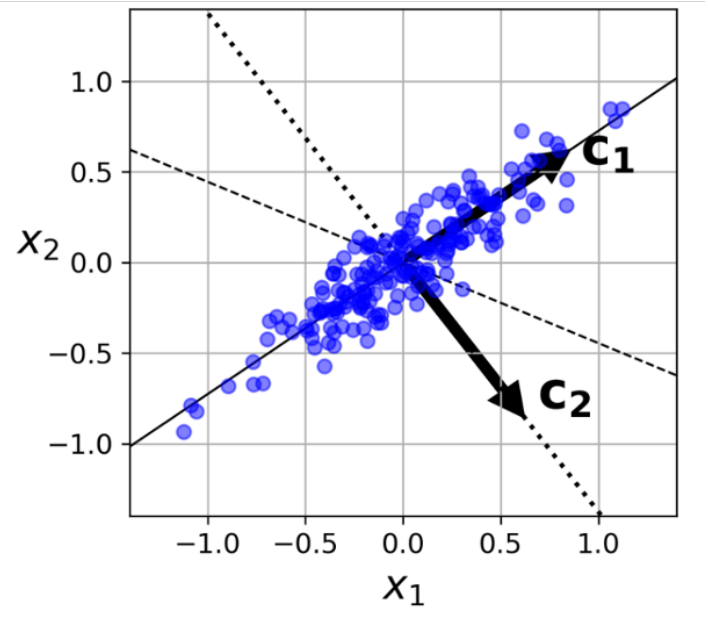
가장 분산 잘 보존 → 그다음 이게 수직이 되는 선 찾으면 됨 (3차원이라면!)

(출처: 핸즈온 머신러닝)

주성분 분석

n 행렬연산

- 데이터에서 분산이 최대인 축을 찾음
- 첫 번째 축에 직교(orthogonal)하고 남은 분산을 최대한 보존하는 두 번째 축을 찾음
- 반복적으로 d 차원의 축까지 찾아감
- C_1이 첫 번째 축이고, C_2가 두 번째 축이됨



주성분 분석 예제

- 시리얼 데이터
  - calories와 rating은 음의 상관관계 (−0.69)

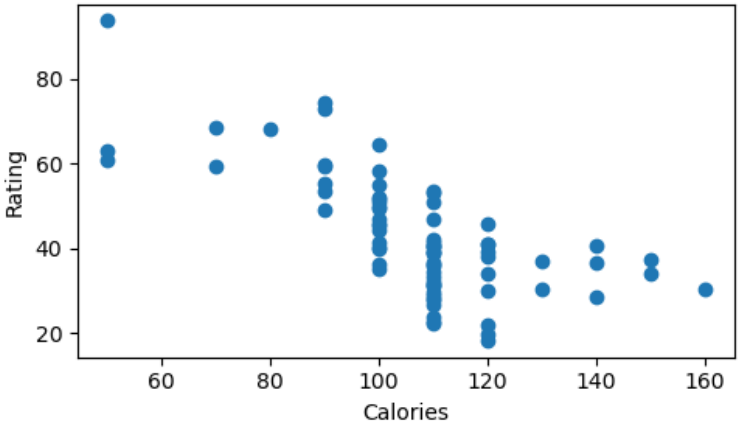


표 4-9 아침 식사용 시리얼 데이터셋의 변수 설명

mfr	시리얼 제조 업체(American Home Food Products, General Mills, Kellogg 등)
type	저온용 또는 고온용
calories	1회분에 대한 칼로리
protein	단백질(g)
fat	지방(g)
sodium	나트륨(mg)
fiber	식이섬유(g)
carbo	복합 탄수화물(g)
sugars	설탕(g)
potass	칼륨(mg)
vitamins	비타민과 미네랄: FDA의 권장 비율로서 0, 25 또는 100을 나타냄
shelf	디스플레이 선반(바닥에서부터 1, 2, 3으로 세어나감)
weight	1회분의 무게(Ounces)
cups	1회분에 제공되는 컵의 수
rating	소비자 보고서에 의한 시리얼 평점

주성분 분석 예제

```
cereals_df = pd.read_csv('../data/
   Cereals.csv')
pcs = PCA(n_components=2)
pcs.fit(cereals_df[['calories', 'rating']])
pcsSummary =
   pd.DataFrame({'Standard deviation':
   np.sqrt(pcs.explained_variance_),
                  'Proportion of variance':
   pcs.explained_variance_ratio_,
                  'Cumulative
   proportion':
   np.cumsum(pcs.explained_variance_r
   atio_)})
pcsSummary = pcsSummary.transpose()
pcsSummary.columns = ['PC1', 'PC2']
pcsSummary.round(4)
```

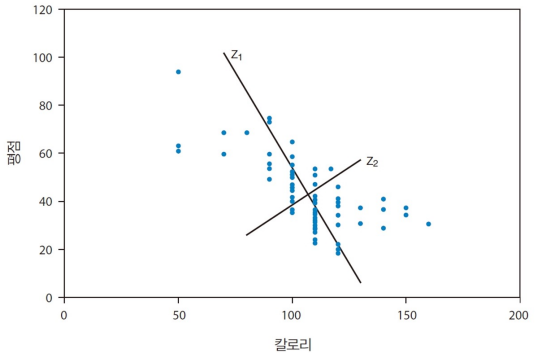


그림 4-3 2개의 주성분 방향이 있는 77개의 아침 식사용 시리얼에 대한 칼로리(calories)와 평점(rating)의 산점도

	PC1	PC2
Standard deviation	22.3165	8.8844
Proportion of variance	0.8632	0.1368
Cumulative proportion	0.8632	1.0000

주성분 분석

- 주성분은 어떻게 찾는가?
- 특잇값 분해 (Singular Value Decomposition, SVD)라는 표준 행렬 분해를 활용하여 하나의 행렬을 세 개 행렬의 곱으로 분해가능

$$X = U\Sigma V^T$$

- 모든 주성분의 단위 벡터가 V^T 에 다음과 같이 담겨 있음

$$\mathbf{V} = \begin{pmatrix} | & | & & | \\ \mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_n \\ | & | & & | \end{pmatrix}$$

n차원 선도벡

주성분 분석 예제

```
pcs = PCA()
pcs.fit(cereals_df.iloc[:, 3:].dropna(axis=0)) # axis=0 row
pcsSummary_df = pd.DataFrame({'Standard deviation':
  np.sqrt(pcs.explained_variance_),
                  'Proportion of variance':
  pcs.explained_variance_ratio_,
                  'Cumulative proportion':
  np.cumsum(pcs.explained_variance_ratio_)})
pcsSummary_df = pcsSummary_df.transpose()
pcsSummary_df.columns = ['PC{}'.format(i) for i in range(1,
  len(pcsSummary_df.columns) + 1)]
pcsSummary_df.round(4)
```

주성분 분석 예제 시각화

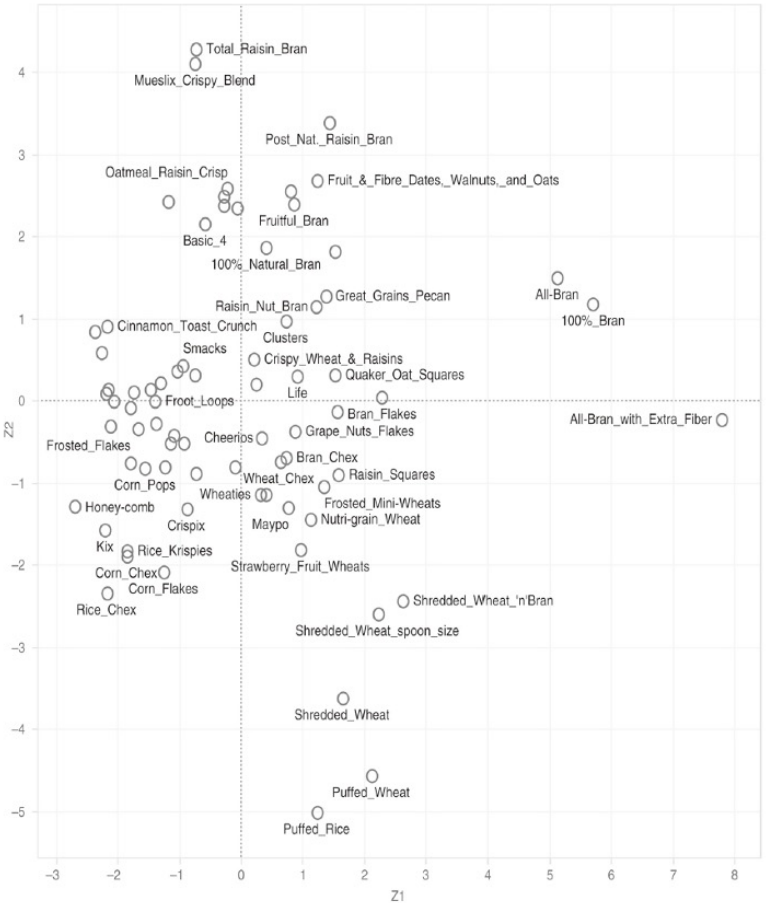


그림 4-4 정규화한 아침 식사용 시리얼 데이터에 대한 첫 번째와 두 번째 주성분 점수의 산점도(타블로(Tableau)로 생성)