

# REPORT

## Adult Cencus Income EDA & Prediction



• 과 목 명 : 인 공 지 능

• 담 당 교 수 : 박 소 현 교수님

• 제 출 일 : 2024-06-11

• 학 과 : 컴퓨터공학전공

• 학 번 : 2019212978

• 성 명 : 고 영 민

## 목 차

1. 개요 .....	2
2. 데이터 세트 설명 .....	3
1. 데이터 소개 .....	3
2. 어노테이션 포맷 .....	6
3. 데이터 전처리 .....	9
4. 데이터 시각화 .....	14
5. 모델 구현 및 성능 평가 .....	21
1. 모델 구현 .....	21
2. 모델 성능 평가 .....	23
3. 하이퍼 파라미터 튜닝 및 성능 재평가 .....	25
6. 결론 및 소감 .....	27
7. References .....	28

## 1. 개요

### I. 선정 데이터 세트

- Adult Census Income

### II. 선정 이유

- 본 데이터 세트는 소득 수준을 예측하는 데 중점을 두고 있어 경제적 성공을 예측하는 다양한 요인들을 분석할 수 있다. 그리고 성별, 인종, 교육 수준, 결혼 상태 등 15 가지의 다양한 변수에 대해 약 32,000 개의 레코드를 포함하고 있어, 충분히 큰 규모로 구성되어 있다는 점이 특징이다. 따라서 각 변수 간의 상호작용을 분석하여 사회적 불평등이나 경제적 격차와 같은 사회학적 이슈를 심층적으로 탐구할 수 있다는 점에서 흥미가 생겨 본 데이터 세트를 선정하게 되었다. 또한, 소득이 5 만 달러 이상인지 이하인지에 관한 이진 분류 문제를 포함하고 있어, 다양한 머신러닝 알고리즘을 통해 분류 문제를 다룰 수 있는 좋은 계기가 될 것이라 판단하여 본 데이터 세트를 선택하게 되었다.

### III. 데이터 세트 크기

- 32561 행 x 15 열

### IV. 데이터 세트 출처

- 캐글(Kaggle)의 Adult Census Income 데이터 세트[1]

### V. 분석 도구

- Pandas, NumPy, Seaborn, Matplotlib, ProfileReport, scikit-learn

## 2. 데이터 세트 설명

### 1 데이터 소개

전반적인 데이터 확인														
df.head()														
	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	40	United States
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	United States
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	40	United States
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	United States
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	United States

그림 1. 상위 5 개 데이터 출력

Overview

Alerts 7

Reproduction

Dataset statistics

Number of variables	15
Number of observations	32561
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	23
Duplicate rows (%)	0.1%
Total size in memory	3.7 MiB
Average record size in memory	120.0 B

Variable types

Numeric	6
Categorical	9

그림 2. ProfileReport 라이브러리를 활용한 DataFrame 의 Overview

그림 2. ProfileReport 라이브러리를 활용한 DataFrame 의 Overview

- "Adult Census Income" 데이터 세트는 개인의 소득 수준을 예측하는 데 중점을 두고 있으며, 약 32,000 개의 레코드로 구성되어 있다. 이 데이터 세트에는 나이, 성별, 인종, 교육 수준, 결혼 상태, 직업, 가족 관계, 출신 국가 등 다양한 인구 통계학적 정보가 포함되어 있다. 또한, 자본 이득 및 손실, 주당 근로 시간 등 경제적 변수도 포함되어 있다. 소득 변수는 연간 소득이 50,000 달러를 초과하는지 여부를 나타내며, 이는 이진 분류 문제를 구성한다. 이 데이터 세트는 경제적 성공을 예측하는 다양한 요인들을 분석하고, 사회적 불평등이나 경제적 격차와 같은 사회학적 이슈를 심층적으로 탐구하는 데 유용할 것으로 판단된다.

## 데이터 요약표 확인

df.describe()

	age	fnlwgt	education.num	capital.gain	capital.loss	hours.per.week
count	32561.000000	3.256100e+04	32561.000000	32561.000000	32561.000000	32561.000000
mean	38.581647	1.897784e+05	10.080679	1077.648844	87.303830	40.437456
std	13.640433	1.055500e+05	2.572720	7385.292085	402.960219	12.347429
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.178270e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783560e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.370510e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.000000

그림 3. 데이터 요약표

## - 나이 (age)

- 개수 (count): 32,561 명의 데이터
- 평균 (mean): 평균 나이는 38.58 세
- 표준편차 (std): 나이의 표준편차는 13.64 세로, 나이 분포가 상당히 넓음
- 최소값 (min): 최저 나이는 17 세
- 중앙값 (50%): 중앙값은 37 세로, 절반의 사람들은 37 세 이하
- 최대값 (max): 최고 나이는 90 세

## - 최종 가중치 (fnlwgt)

- 개수 (count): 32,561 명의 데이터
- 평균 (mean): 평균 가중치는 189,778.4
- 표준편차 (std): 표준편차는 105,550 으로, 가중치의 변동이 큼
- 최소값 (min): 최저 가중치는 12,285
- 중앙값 (50%): 중앙값은 178,356
- 최대값 (max): 최고 가중치는 1,484,705

## - 교육 수준 숫자 (education.num)

- 개수 (count): 32,561 명의 데이터
- 평균 (mean): 평균 교육 수준은 10.08
- 표준편차 (std): 표준편차는 2.57 로, 교육 수준의 변동이 적당한 수준
- 최소값 (min): 최저 교육 수준은 1
- 중앙값 (50%): 중앙값은 10
- 최대값 (max): 최고 교육 수준은 16

## - 자본 이득 (capital.gain)

- 개수 (count): 32,561 명의 데이터
- 평균 (mean): 평균 자본 이득은 1,077.65
- 표준편차 (std): 표준편차는 7,385.29 로, 자본 이득의 변동이 큼
- 최소값 (min): 최저 자본 이득은 0
- 중앙값 (50%): 중앙값은 0
- 최대값 (max): 최고 자본 이득은 99,999

## - 자본 손실 (capital.loss)

- 개수 (count): 32,561 명의 데이터

- 평균 (mean): 평균 자본 손실은 87.30
- 표준편차 (std): 표준편차는 402.96 으로, 자본 손실의 변동이 큼
- 최소값 (min): 최저 자본 손실은 0
- 중앙값 (50%): 중앙값은 0
- 최대값 (max): 최고 자본 손실은 4,356
- **주당 근로 시간 (hours.per.week)**
- 개수 (count): 32,561 명의 데이터
- 평균 (mean): 평균 주당 근로 시간은 40.44 시간
- 표준편차 (std): 표준편차는 12.35 시간으로, 근로 시간의 변동이 꽤 있음
- 최소값 (min): 최저 주당 근로 시간은 1 시간
- 중앙값 (50%): 중앙값은 40 시간
- 최대값 (max): 최고 주당 근로 시간은 99 시간

## 2 어노테이션 포맷

항목명	데이터 타입	설명
age	int64	나이
workclass	object	고용 형태
fnlwgt	int64	특정 개인이 대표하는 인구 수를 나타내는 가중치(Final Weight)
education	object	교육 수준
education-num	int64	수치화 된 교육 수준 1: Preschool (유치원) 5: 5th grade (5 학년) 9: 9th grade (9 학년) 11: High school graduate 13: Some college (대학교 중퇴) 16: Bachelor's degree (학사 학위) 18: Master's degree (석사 학위) 20: Doctorate (박사 학위)
marital-status	object	결혼 상태
occupation	object	직업
relationship	object	가족 관계
race	object	인종
sex	object	성별
capital-gain	int64	자본 이득
capital-loss	int64	자본 손실
hours-per-week	int64	주당 근로 시간
native-country	object	출신 국가
income	object	연간 소득 (예측 목표 변수, '>50K' 또는 '<=50K')

### I. age (나이)

- 설명: age 속성은 개인의 나이를 나타내는 특성이며 데이터 타입은 int64 이다.
- 예시: 25, 40, 65

### II. workclass (고용 형태)

- 설명: workclass 속성은 개인의 고용 형태를 나타내는 특성이며 데이터 타입은 object 이다.
- 예시: Private (민간), Self-emp-not-inc (비법인 자영업), Government (정부)

### III. **fnlwgt (최종 가중치, Final Weight)**

- 설명: fnlwgt 속성은 인구 조사에서 특정 개인이 대표하는 인구 수의 가중치를 나타내는 특성이며 데이터 타입은 int64 이다.
- 예시: 1,000, 50,000, 100

### IV. **education**

- 설명: education 속성은 개인이 받은 교육 수준을 나타내는 특성이며 데이터 타입은 object 이다.
- 예시: Bachelors (학사), HS-grad (고등학교 졸업), Masters (석사)

### V. **education.num (수치화된 교육 수준)**

- 설명: education.nu 속성은 교육 수준을 숫자로 나타낸 특성이며 데이터 타입은 int64 이다.
- 예시: 9 (9 학년), 13 (대학교 중퇴), 16 (학사 졸업)

### VI. **marital.status (결혼 상태)**

- 설명: marital.status 개인의 결혼 상태를 나타내는 특성이며 데이터 타입은 object 이다.
- 예시: Never-married (결혼하지 않음), Married-civ-spouse (결혼, 배우자 있음), Divorced (이혼)

### VII. **occupation (직업)**

- 설명: occupation 속성은 개인의 직업을 나타내는 특성이며 데이터 타입은 obejct 이다.
- 예시: Tech-support (기술 지원), Craft-repair (수리 기술자), Sales (영업)

### VIII. **relationship (가족 관계)**

- 설명: relationship 속성은 가구 내에서의 관계를 나타내는 특성이며 데이터 타입은 object 이다.
- 예시: Husband (남편), Not-in-family (가족 아님), Own-child (자녀)

### IX. **race (인종)**

- 설명: race 속성은 개인의 인종을 나타내는 특성이며 데이터 타입은 object 이다.
- 예시: White (백인), Black (흑인), Asian-Pac-Islander (아시아/태평양 섬)

### X. **sex (성별)**

- 설명: sex 속성은 성별을 나타내는 특성이며 데이터 타입은 obejct 이다.
- 예시: Male (남성), Female (여성)

### XI. **capital.gain (자본 이득)**

- 설명: capital.gain 속성은 자본 자산의 매매를 통해 얻은 이익을 나타내는 특성이며 데이터 타입은 int64 이다.
- 예시: 0, 5000, 14084



**XII. capital.loss (자본 손실)**

- 설명: capital.loss 속성은 자본 자산의 매매를 통해 얻은 이익을 나타내는 특성이며 데이터 타입은 int64 이다.
- 예시: 0, 1500, 1902

**XIII. hours.per.week (주당 근로 시간)**

- 설명: hours.per.week 속성은 개인이 일주일에 일하는 시간을 나타내는 특성이며 데이터 타입은 int64 이다.
- 예시: 40, 50, 20

**XIV. native.country (출신 국가)**

- 설명: native.country 속성은 개인의 출신 국가를 나타내는 특성이며 데이터 타입은 object 이다.
- 예시: United-States (미국), Mexico (멕시코), Philippines (필리핀)

**XV. income (소득)**

- 설명: 연간 소득이 50,000 달러를 초과하는지 여부를 나타내는 특성이며 데이터 타입은 object 이다.
- 예시: >50K (초과), <=50K (이하)

### 3. 데이터 전처리

#### 3.1. 데이터 정제

##### 결측치(missing values) 처리

```
df[df == '?'] = np.nan
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
#   Column                Non-Null Count  Dtype
---  ---
0    age                  32561 non-null  int64
1    workclass            30725 non-null  object
2    fnlwgt               32561 non-null  int64
3    education            32561 non-null  object
4    education.num        32561 non-null  int64
5    marital.status       32561 non-null  object
6    occupation           30718 non-null  object
7    relationship         32561 non-null  object
8    race                 32561 non-null  object
9    sex                  32561 non-null  object
10   capital.gain         32561 non-null  int64
11   capital.loss         32561 non-null  int64
12   hours.per.week       32561 non-null  int64
13   native.country       31978 non-null  object
14   income               32561 non-null  object
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
```

그림 4. 데이터가 '?'인 경우 결측치로 데이터를 치환

```
missing_values_percentage = df.isnull().mean() * 100

missing_values_percentage_sorted = missing_values_percentage.sort_values(ascending = False)

plt.figure(figsize=(10, 6))
sns.barplot(x=missing_values_percentage_sorted, y=missing_values_percentage_sorted.index)
plt.title('Percentage of Missing Values in Each Column')
plt.xlabel('Percentage of Missing Values')
plt.ylabel('Columns')
plt.show()
```

그림 5. 결측치 시각화 코드

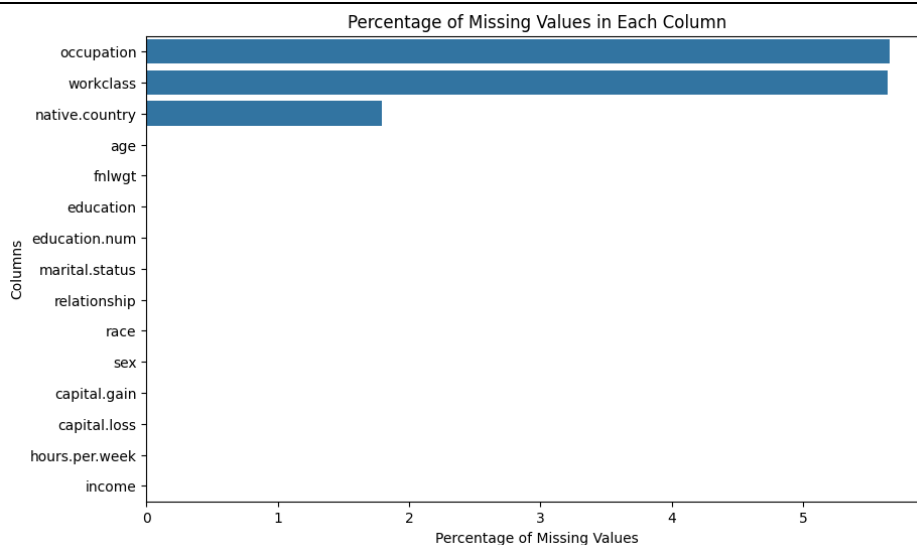
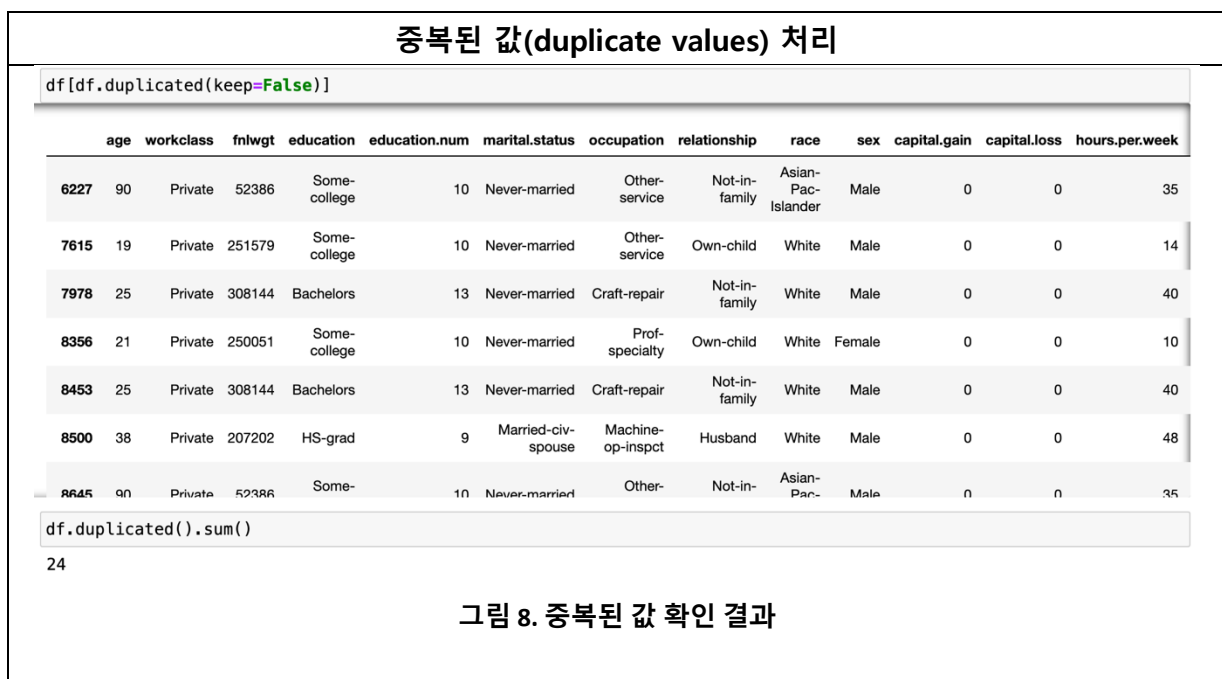
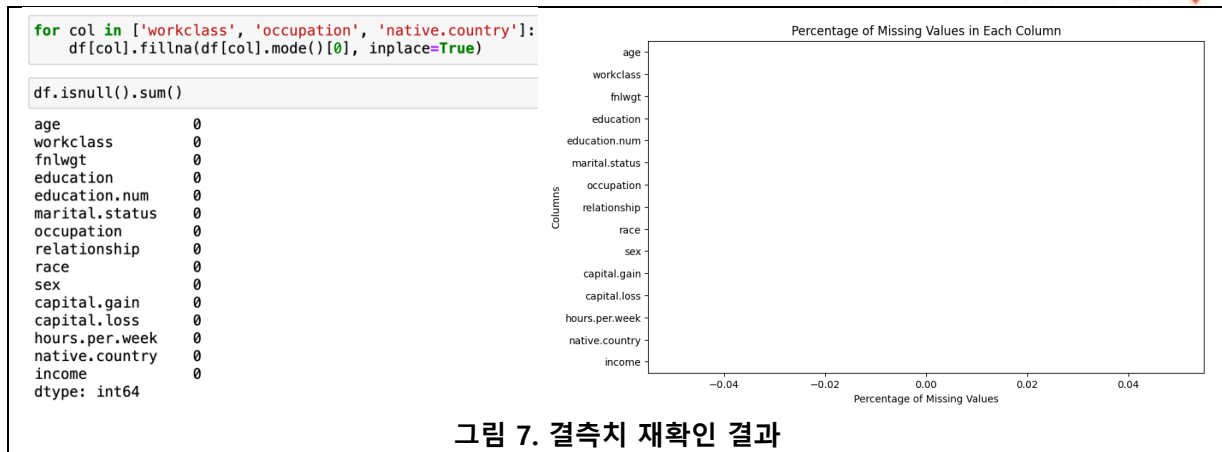


그림 6. 결측치 시각화 결과

- 이 데이터 세트에는 occupation, workclass, native.country 세 가지 속성에서 결측치가 5% 이내로 존재함. 이 속성들은 범주형 변수이기 때문에 평균과 같은 대표값으로 대체할 수 없으므로, 각 속성의 결측치를 최빈값으로 대체함.



- 이 데이터 세트에는 24 개의 중복된 값이 존재하는 것으로 나타났음. 따라서, 중복된 값을 제거함.

```
# Drop duplicate rows from the DataFrame
df.drop_duplicates(inplace=True)
```

```
df.shape
```

```
(32537, 15)
```

그림 9. 중복된 값 삭제

- 삭제 전 32,561 개의 레코드 중 24 개의 중복된 값이 제거되어, 최종적으로 32,537 개의 레코드가 남음.

### 3.2. 데이터 변환

#### 분석에 크게 기여하지 않는 특성 확인

```
# Inspect useless features
df.nunique().sort_values()
```

```
sex                2
income             2
race               5
relationship        6
marital.status      7
workclass           8
occupation         14
education          16
education.num       16
native.country     41
age                73
capital.loss       92
hours.per.week     94
capital.gain       119
fnlwgt            21648
dtype: int64
```

그림 10. 불필요 특성 파악

- fnlwgt 는 가중치일 뿐이며, 직접적인 설명 변수가 아니기 때문에 fnlwgt 속성을 삭제함.

```
df = df.drop(columns=['fnlwgt']) # 데이터 분석에 크게 기여하지 않는 경우가 많으므로 이를 제거
df.head()
```

	age	workclass	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
0	90	3	11	9	6	9	1	4	0	0	4356	40	38	
1	82	3	11	9	6	3	1	4	0	0	4356	18	38	
2	66	3	15	10	6	9	4	2	0	0	4356	40	38	
3	54	3	5	4	0	6	4	4	0	0	3900	40	38	
4	41	3	15	10	5	9	3	4	0	0	3900	40	38	

그림 11. fnlwgt 속성 삭제

#### 데이터 세트의 속성 이름 변경

```
df = df.rename(columns={
    'marital.status': 'marital_status',
    'education.num': 'education_num',
    'native.country': 'native_country',
    'capital.gain': 'capital_gain',
    'capital.loss': 'capital_loss',
    'hours.per.week': 'hours_per_week'
})
```

```
df.columns
```

```
Index(['age', 'workclass', 'education', 'education_num', 'marital_status',
       'occupation', 'relationship', 'race', 'sex', 'capital_gain',
       'capital_loss', 'hours_per_week', 'native_country', 'income'],
      dtype='object')
```

그림 12. 데이터 세트의 속성 이름 변경

- 식별성이 떨어지는 6 개의 열 이름에 대해 각각 언더바(\_)로 대체하여 열 이름을 변경함.

## 이상치 제거

### 1. Check the distributions of numerical data

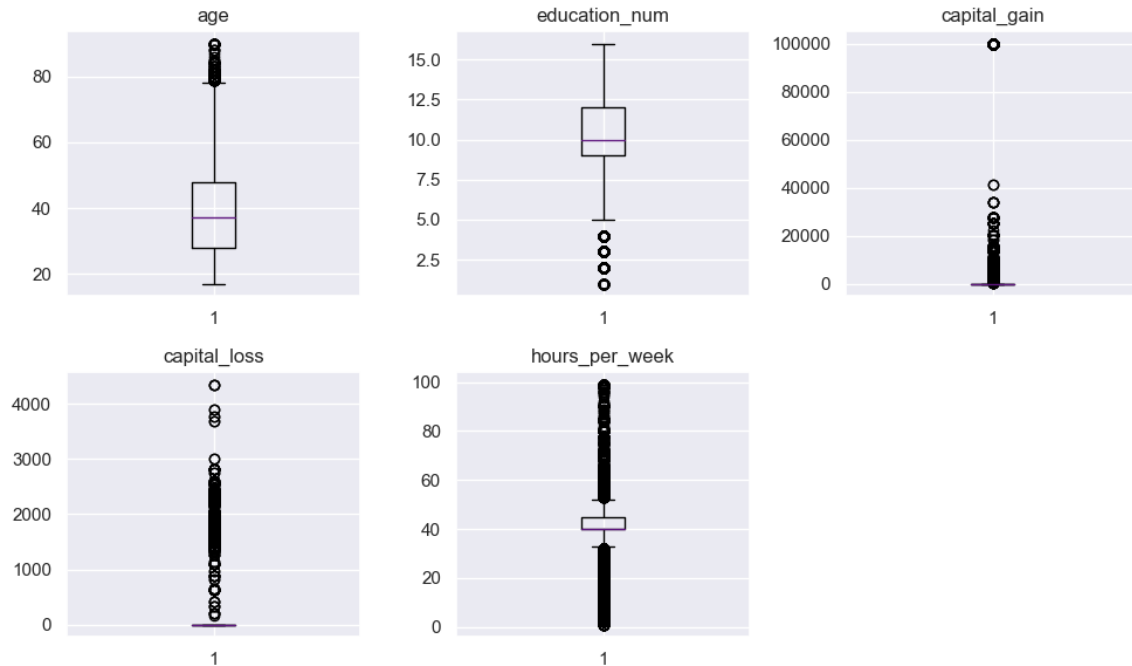


그림 13. 수치형 데이터의 상자 그림

### 2. Drop outliers

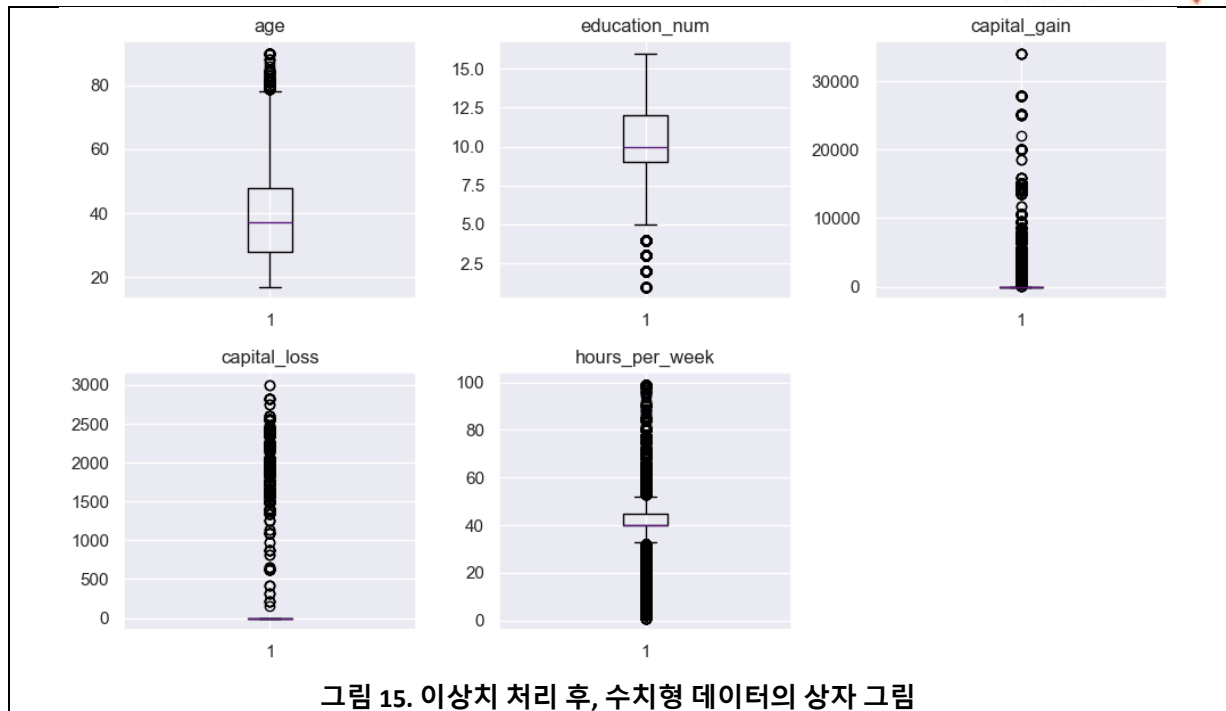
```
condition = (df['capital_gain'] > 40000) | (df['capital_loss'] > 3500)
df = df[~condition]

numeric_columns = ['age', 'education_num', 'capital_gain', 'capital_loss', 'hours_per_week']

plt.figure(figsize=(10, 6))
for i, column in enumerate(numeric_columns, 1):
    plt.subplot(2, 3, i)
    plt.boxplot(df[column])
    plt.title(column)
    plt.grid(True)

plt.tight_layout()
plt.show()
```

그림 14. 이상치 처리 코드



- 과목별 상자그림을 확인한 결과, IQR 기준으로는 너무 많은 이상치가 관찰되었다. 그러나 실제 소득 분석을 위해서는 해당 값들을 다 제거하는 경우 제대로 된 분석이 어렵기 때문에 capital\_gain 이 40,000 을 넘고 capital\_loss 가 3,500 을 넘는 경우를 이상치로 간주하고 삭제하였음.

### 3.3. 데이터 정규화

데이터 정규화
데이터 정규화는 모델 구축을 위한 "5. 모델 구현 및 성능 평가 단계"에서 진행하였음.

## 4. 데이터 시각화

### 4.1. 수치형 데이터들의 상관관계 확인 with Heat Map

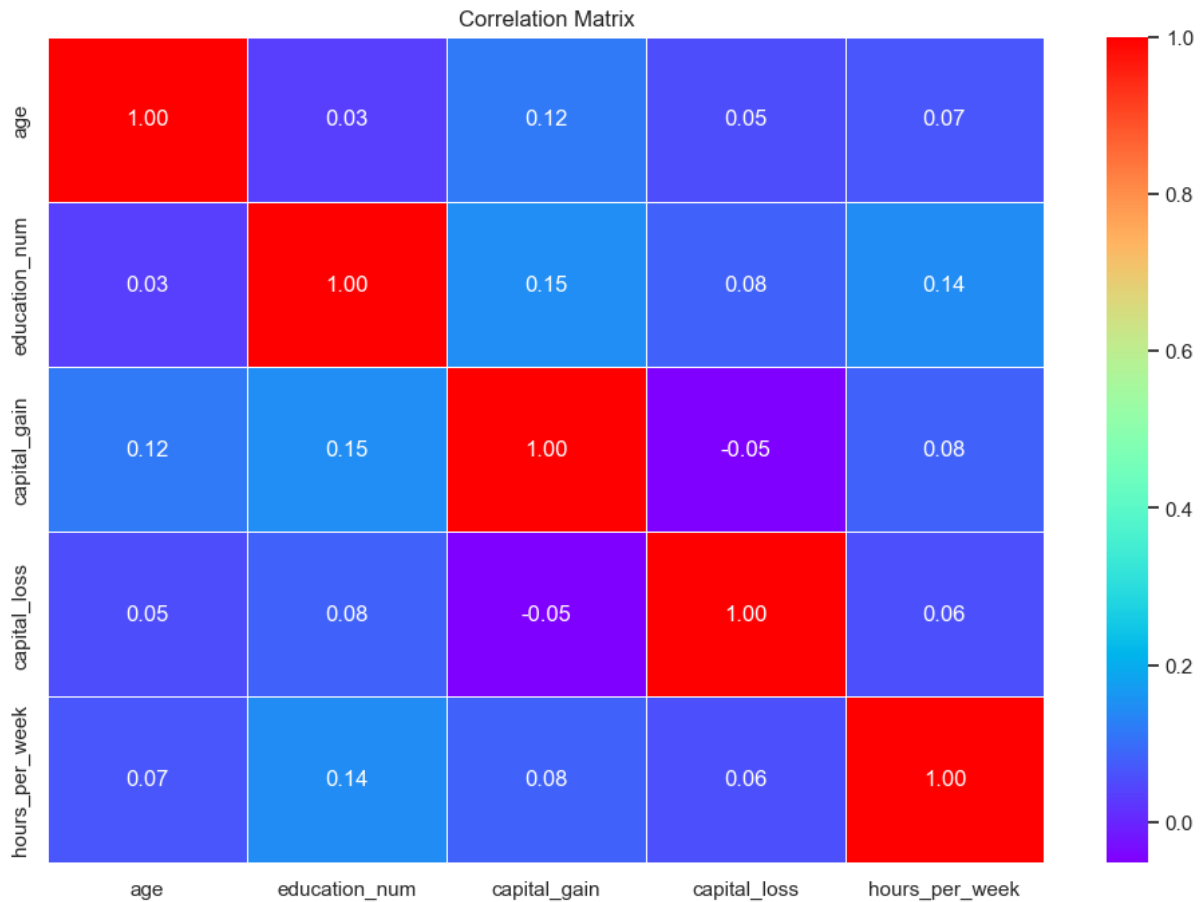
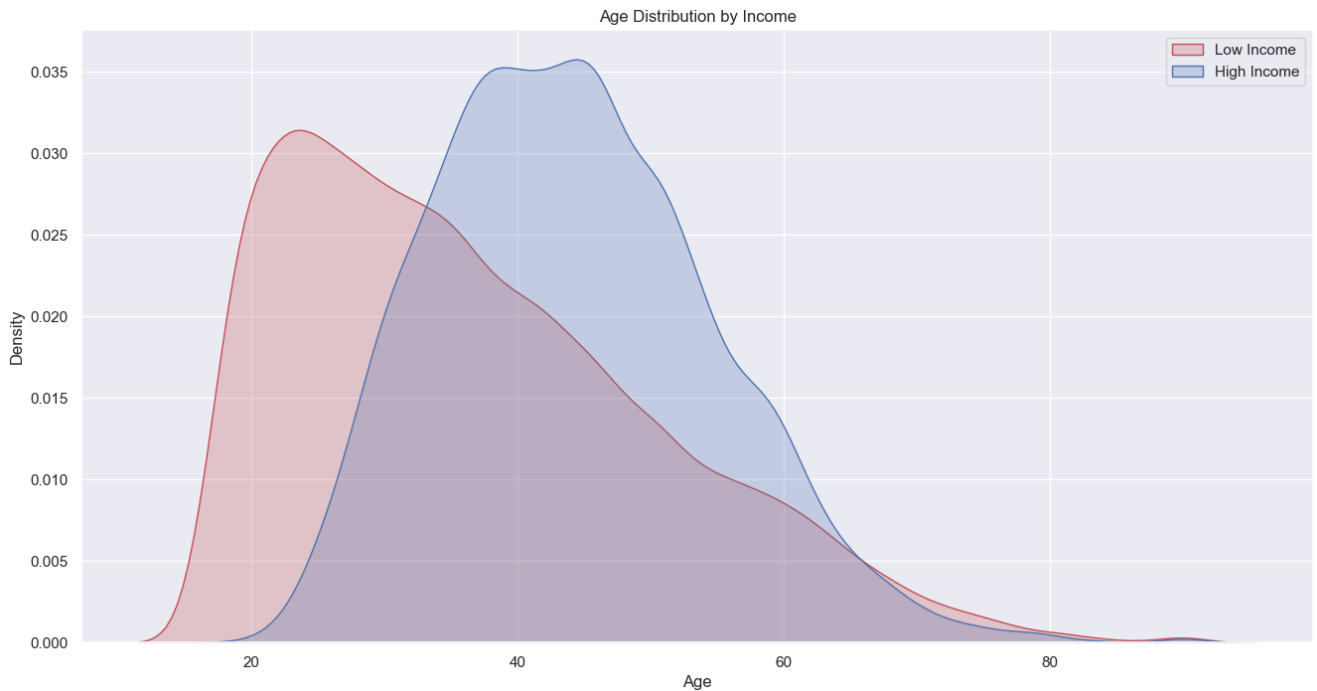


그림 16. 수치형 데이터 간의 히트맵

히트맵은 데이터의 상관관계를 시각적으로 표현하는 시각화 기법이다. 본 분석에서 사용한 히트맵은 상관관계가 높을수록 붉은색에 가깝고, 상관관계가 낮을수록 파란색에 가까워진다. 이 분석에서는 수치형 데이터에 대한 히트맵을 통해 각 속성 간의 상관관계를 확인했다. 그 결과, 데이터 간의 상관관계가 거의 없음을 확인했으며, 이로 인해 모델링 과정에서 속성 간의 상관성이 높아서 일반화 성능이 저하되는 문제는 없을 것으로 판단된다.

## 4.2. 소득 수준에 따른 연령 분포 확인 with KDE Plot



- 저소득 그룹: 젊은 연령대(20 대~40 대)의 비율이 높고, 60 세 이상의 비율은 낮다.  
이는 저소득층의 노후 대비가 부족하다는 것을 의미한다.

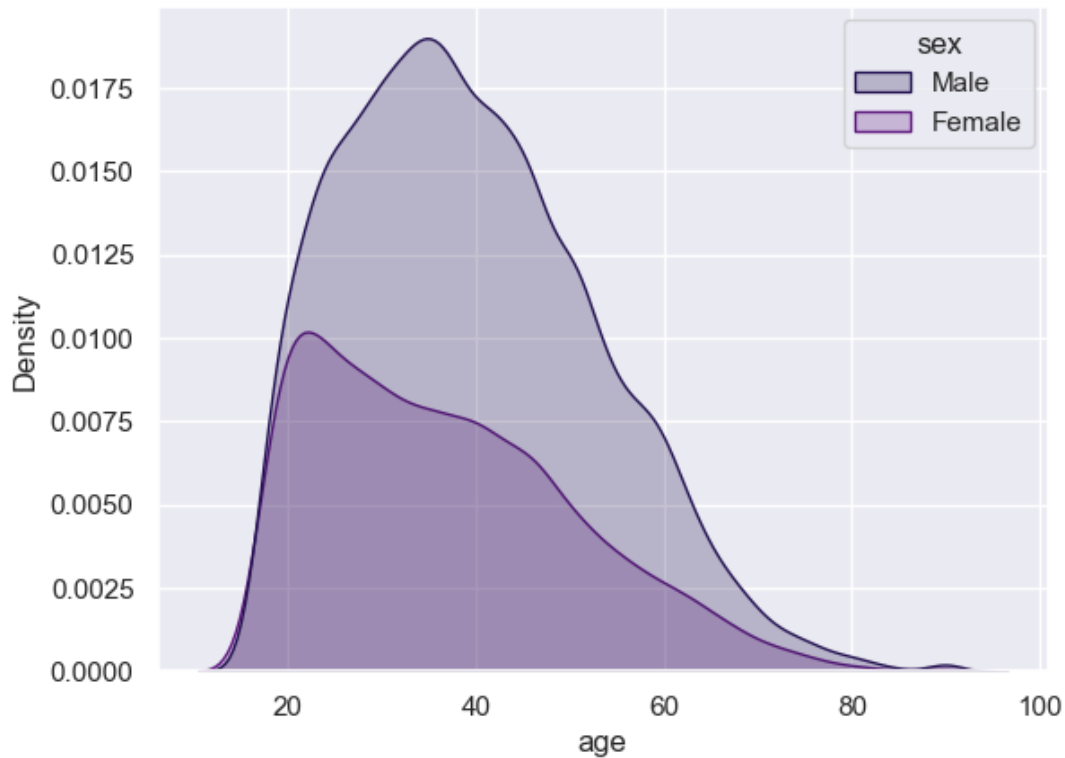
- 고소득 그룹: 40 대~60 대의 비율이 높고, 20 대 이하와 80 세 이상의 비율은 낮다.  
이는 고소득층이 경력을 쌓는 연령대에 집중되어 있고, 노후 대비가 비교적 잘 되어 있다는 것을 의미한다.

- 소득 수준과 연령 분포의 관계: 연령과 함께 평균 소득이 증가하는 경향이 있다.  
즉, 나이가 들수록 소득이 높아지는 경향이 있다.

즉, 소득 수준과 연령 분포 사이에 명확한 관계가 있음



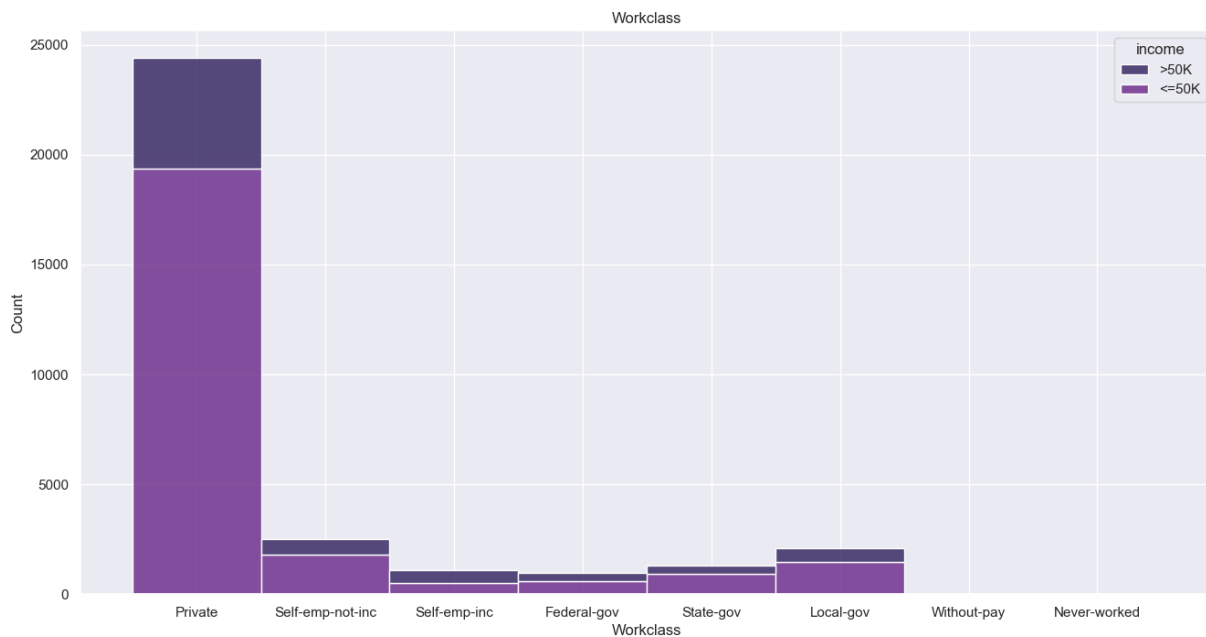
#### 4.3. 연령(age)에 따른 성별(sex) 분포 확인 with KDE Plot



본 데이터 세트에서는 남성의 데이터가 전반적으로 더 많이 포함되어 있지만, 연령에 따른 성별 분포 형태는 대체적으로 유사하다. 이는 남성과 여성 간의 연령 분포 차이가 크지 않다는 것이다.

또한, 데이터가 오른쪽으로 긴 꼬리를 가진 분포를 보이고 있어 왜도가 0보다 크다는 사실도 확인할 수 있다. 이는 전반적으로 연령이 평균 연령보다 낮은 사람들이 데이터에 더 많이 포함되어 있음을 의미한다.

#### 4.4. 직업 유형(workclass)에 따른 소득 수준(income)의 분포 with Histogram



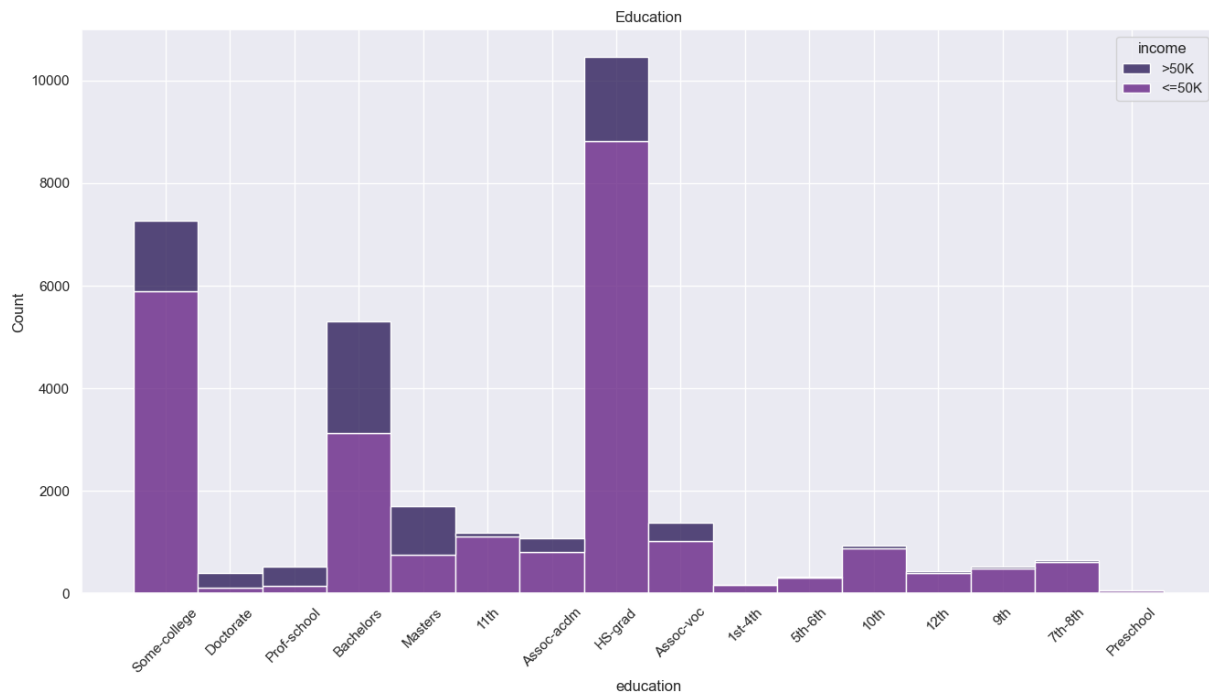
본 히스토그램을 통해 각 직업 유형 내에서 높은 소득 수준(>50K)과 낮은 소득 수준(<=50K)의 비율을 확인할 수 있다.

소득이 가장 높은 직업은 개인 사업주로 나타났다. 이 직업의 누적 인원 수는 약 18,000 명이며, 이 중 50,000 달러 이상 소득자는 약 12,000 명이다. 반면, 소득이 가장 낮은 직업은 무급 근로자와 근무 경력 없는 사람이다.

연방 정부 직원, 주 정부 직원, 지방 정부 직원의 소득은 비교적 안정적이다. 이 직업들의 누적 인원 수는 비슷하고, 50,000 달러 이상 소득자의 비율도 비슷하다.

즉, 사업을 하는 사람의 경우 50,000 달러 이상 소득자의 경우, 다른 직업의 경우보다 더 높은 비율을 차지한다는 것을 확인할 수 있다.

#### 4.5. 교육 수준(education)에 따른 소득 수준(income)의 분포 with Histogram

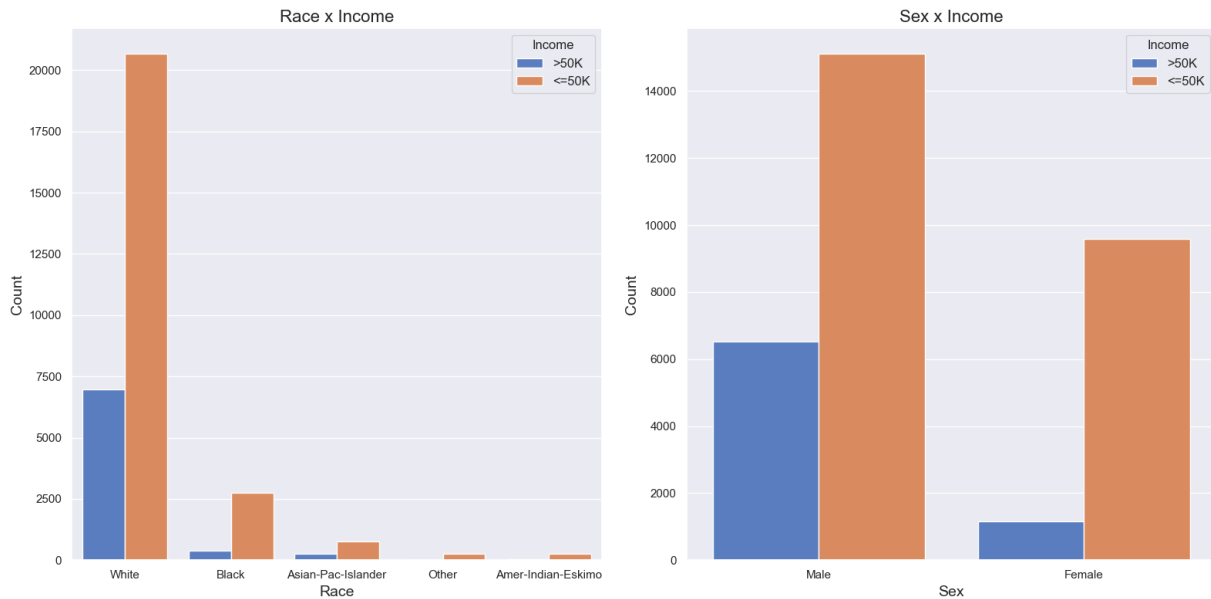


본 히스토그램을 통해 각 직업 유형 내에서 높은 소득 수준(>50K)과 낮은 소득 수준(<=50K)의 비율을 확인할 수 있다.

본 데이터 세트에서 가장 많은 비율을 차지하는 교육 수준은 고등학교 졸업이고 다음은 대학교 중퇴이다. 그렇지만 소득의 비율로만 봤을 때, 50,000 달러를 초과하는 소득을 받는 개인의 교육 수준은 박사 학위(Doctorate)이다. 반면, 가장 낮은 소득을 받는 개인의 교육 수준은 미취학(Preschool)이다.

위와 같은 결과를 통해, 교육 수준이 높을수록 소득이 높을 가능성이 높다는 점을 확인할 수 있다.

#### 4.6. 다양한 인종 그룹 및 성별에 따른 소득 분포 with Count Plot



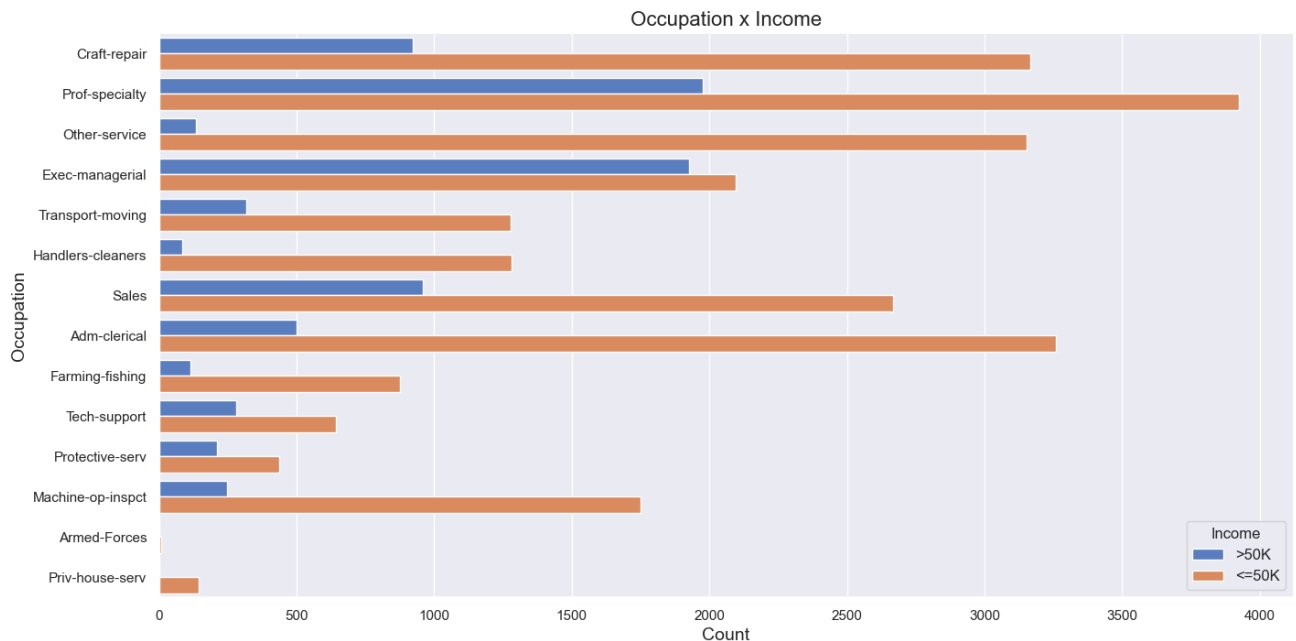
첫 번째 그래프에서 주황색 막대는 연소득이 50,000 달러 이하, 파란색 막대는 연소득이 55,000 달러 초과인 사람들을 나타내는데, 백인 그룹에서 연소득이 50,000 달러 이하인 사람들의 수가 압도적으로 많다. 그리고 흑인, 아시아-태평양 섬, 기타, 아메리카 원주민-에스키모 그룹에서는 대부분의 사람들이 연소득이 50,000 달러 이하인 것으로 나타난다.

특히 백인 그룹에서는 연소득이 50,000 달러 초과인 사람들의 수가 다른 그룹에 비해 비교적 더 많은 것으로 보인다.

남성 그룹에서는 연소득이 50,000 달러 이하인 사람들의 수가 많지만, 연소득이 50,000 달러 초과인 사람들도 상당한 수를 차지하고 있다.

여성 그룹에서는 연소득이 50,000 달러 이하인 사람들이 대부분을 차지하고 있으며, 연소득이 50,000 달러 초과인 사람들은 상대적으로 적은 수를 보이고 있다.

#### 4.7. 직업과 소득 분포 분석



Exec-managerial (임원-관리직)과 Prof-specialty (전문직) 직업군에서 연소득이 50,000 달러를 초과하는 사람들이 비교적 많은 편임을 확인할 수 있다.

Craft-repair (기술-수리) 직업군에서도 연소득이 50,000 달러를 초과하는 사람들이 다수 존재한다. 반면, Other-service (기타 서비스), Handlers-cleaners (처리-청소), Adm-clerical (행정-사무직) 직업군에서는 대부분의 사람들이 연소득이 50,000 달러 이하이다.

Priv-house-serv (가사 서비스)와 Armed-Forces (군인) 직업군은 소득이 50,000 달러를 초과하는 경우가 없다.

## 5. 모델 구현 및 성능 평가

### 1. 모델 구현

#### · 전반적인 과정

여러 범주형 데이터를 Label Encoder 를 사용하여 수치로 변환하고, Standard Scaler 로 데이터를 정규화한다. 그리고 로지스틱 회귀, 의사결정 트리 분류기, 릿지 분류, 서포트 벡터 머신으로 구성된 파이프 라인을 구축하고 GridSearchCV 를 사용하여 각각(LogisticRegression, DecisionTreeClassifier, RidgeClassifier, SVC)을 테스트하고, 각 분류기에 대한 여러 하이퍼 파라미터 조합을 평가한다.

#### - Step 1. Label Encoder 를 사용하여 범주형 데이터를 수치로 변환

```
label_encoder = LabelEncoder()
categorical_columns = ['income', 'workclass', 'education', 'marital_status', 'occupation', 'relationship',
                      'race', 'sex', 'native_country']
df[categorical_columns] = df[categorical_columns].apply(label_encoder.fit_transform)
df.head()
```

	age	workclass	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country
9	41	3	15	10	4	2	4	4	1	0	3004	60	38
10	45	3	10	16	0	9	4	2	0	0	3004	35	38
11	38	5	14	15	4	9	1	4	1	0	2824	45	38
12	52	3	9	13	6	7	1	4	0	0	2824	20	38
13	32	3	12	14	5	3	1	4	1	0	2824	55	38

#### - Step 2. 훈련 데이터와 테스트 데이터를 8:2 로 나누기

```
x_train, x_test, y_train, y_test = train_test_split(df.drop(columns=['income']), df['income'],
                                                    test_size=0.2, random_state=42)
total_size = len(df)
train_size = len(x_train)
test_size = len(x_test)

train_percent = (train_size / total_size) * 100
test_percent = (test_size / total_size) * 100

plt.figure(figsize=(8, 6))
sns.barplot(x=['Train', 'Test'], y=[train_percent, test_percent], palette='pastel')
plt.title('Size of Train and Test Sets', fontsize=16)
plt.xlabel('Dataset', fontsize=14)
plt.ylabel('Percentage', fontsize=14)
plt.ylim(0, 100)
plt.show()
```



### - Step 3. Standard Scaler 로 훈련 및 테스트 데이터 정규화

```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

### - Step 4. 머신러닝 모델 파이프라인 정의 및 그리드 서치 실행

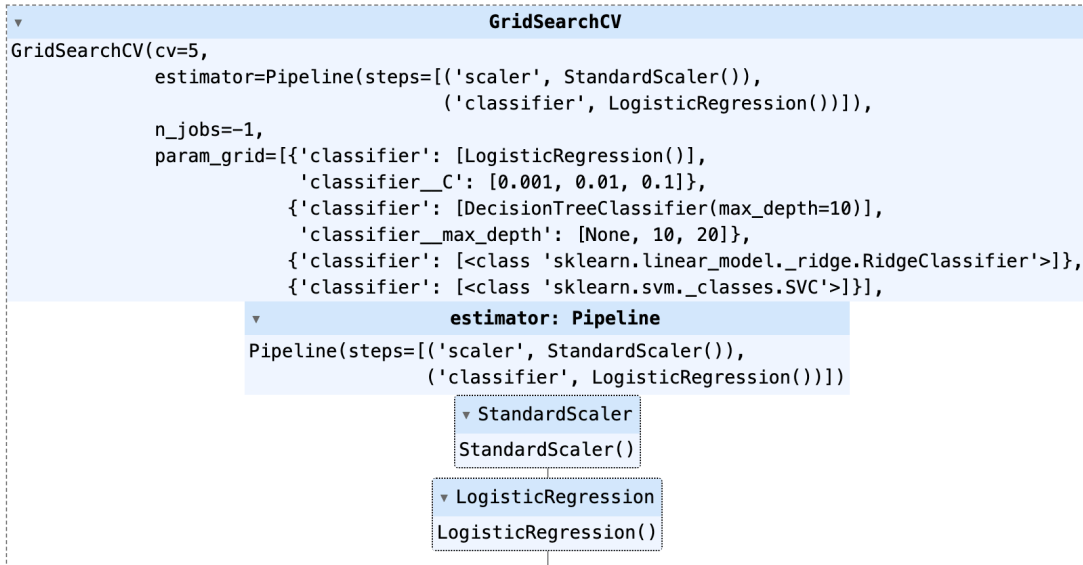
```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

```
pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', LogisticRegression())
])
```

```
param_grid = [
    {
        'classifier': [LogisticRegression()],
        'classifier__C': [0.001, 0.01, 0.1],
    },
    {
        'classifier': [DecisionTreeClassifier()],
        'classifier__max_depth': [None, 10, 20],
    },
    {
        'classifier': [RidgeClassifier],
    },
    {
        'classifier': [SVC],
    },
]
```

```
grid_search = GridSearchCV(pipe, param_grid, cv=5, verbose=2, n_jobs=-1)
grid_search.fit(x_train, y_train)
```

Fitting 5 folds for each of 8 candidates, totalling 40 fits



```
print(f"Best Model: {grid_search.best_estimator_['classifier']}")
print(f"Best Parameters: {grid_search.best_params_}")
print(f"Best Score: {grid_search.best_score_}")
```

```
Best Model: DecisionTreeClassifier(max_depth=10)
Best Parameters: {'classifier': DecisionTreeClassifier(max_depth=10), 'classifier__max_depth': 10}
Best Score: 0.8524312501589486
```

## 2. 모델 성능 평가

```
print("Predictions for 15 samples in the test set:")
correct = 0
incorrect = 0
for i in range(15):
    if y_test.iloc[i] == y_pred[i]:
        print(f"Correct! : Sample {i+1}: Actual - {y_test.iloc[i]}, Predicted - {y_pred[i]}")
        correct += 1
    else:
        print(f"Incorrect! : Sample {i+1}: Actual - {y_test.iloc[i]}, Predicted - {y_pred[i]}")
        incorrect += 1

total = correct + incorrect

print(f'Correct: {correct}, Incorrect: {incorrect} / Total: {total}')
print(f'Estimated accuracy: {correct / total :.2f}')
```

```
Predictions for 15 samples in the test set:
Correct! : Sample 1: Actual - 0, Predicted - 0
Correct! : Sample 2: Actual - 0, Predicted - 0
Correct! : Sample 3: Actual - 0, Predicted - 0
Correct! : Sample 4: Actual - 0, Predicted - 0
Correct! : Sample 5: Actual - 0, Predicted - 0
Correct! : Sample 6: Actual - 0, Predicted - 0
Correct! : Sample 7: Actual - 0, Predicted - 0
Incorrect! : Sample 8: Actual - 1, Predicted - 0
Correct! : Sample 9: Actual - 0, Predicted - 0
Incorrect! : Sample 10: Actual - 1, Predicted - 0
Correct! : Sample 11: Actual - 0, Predicted - 0
Correct! : Sample 12: Actual - 0, Predicted - 0
Correct! : Sample 13: Actual - 1, Predicted - 1
Correct! : Sample 14: Actual - 1, Predicted - 1
Correct! : Sample 15: Actual - 1, Predicted - 1
Correct: 13, Incorrect: 2 / Total: 15
Estimated accuracy: 0.87
```



```
best_model = grid_search.best_estimator_  
y_pred = best_model.predict(x_test)  
  
accuracy = accuracy_score(y_test, y_pred)  
precision = precision_score(y_test, y_pred)  
recall = recall_score(y_test, y_pred)  
f1 = f1_score(y_test, y_pred)  
  
print(f"Test Set Accuracy: {accuracy}")  
print(f"Precision: {precision}")  
print(f"Recall: {recall}")  
print(f"F1-Score: {f1}")
```

Test Set Accuracy: 0.8589743589743589  
Precision: 0.7493584260051326  
Recall: 0.5855614973262032  
F1-Score: 0.6574108818011257

### 3. 하이퍼 파라미터 튜닝 및 성능 재평가

```
param_grid = [
    {
        'classifier': [LogisticRegression()],
        'classifier__C': [0.01, 0.1, 1],
        'classifier__penalty': ['l2'],
        'classifier__solver': ['liblinear'],
    },
    {
        'classifier': [DecisionTreeClassifier()],
        'classifier__max_depth': [10, 20],
        'classifier__min_samples_split': [2, 5],
        'classifier__min_samples_leaf': [1, 2],
    },
    {
        'classifier': [RidgeClassifier()],
        'classifier__alpha': [0.1, 1.0, 10.0],
    },
    {
        'classifier': [SVC()],
        'classifier__C': [0.1, 1, 10],
        'classifier__kernel': ['linear', 'rbf'],
    },
    {
        'classifier': [RandomForestClassifier()],
        'classifier__n_estimators': [50, 100],
        'classifier__max_depth': [10, 20],
        'classifier__min_samples_split': [2, 5],
        'classifier__min_samples_leaf': [1, 2],
    },
    {
        'classifier': [KNeighborsClassifier()],
        'classifier__n_neighbors': [3, 5],
        'classifier__weights': ['uniform', 'distance'],
    },
]
```

하이퍼 파라미터 튜닝 전	하이퍼 파라미터 튜닝 후
<b>Best Model:</b> DecisionTreeClassifier(max_depth=10)	<b>Best Model:</b> RandomForestClassifier(max_depth=20, min_samples_leaf=2)
<b>Best Parameters:</b> {'classifier': DecisionTreeClassifier(max_depth=10), 'classifier__max_depth': 10}	<b>Best Parameters:</b> {'classifier': RandomForestClassifier(max_depth=20, min_samples_leaf=2), 'classifier__max_depth': 20, 'classifier__min_samples_leaf': 2, 'classifier__min_samples_split': 2, 'classifier__n_estimators': 100}
<b>Best Score:</b> 0.8523926102912414	<b>Best Score:</b> 0.8627043835330174

하이퍼 파라미터 튜닝 전	하이퍼 파라미터 튜닝 후
Test Set Accuracy: 0.86	Test Set Accuracy: 0.87
Precision: 0.75	Precision: 0.77
Recall: 0.58	Recall: 0.61
F1-Score: 0.65	F1-Score: 0.68

## 6. 결론 및 소감

### · 결론

탐색적 데이터 분석 결과, 특정 직업군과 인종, 성별, 교육수준, 연령 등 다양한 요인이 소득 수준에 큰 영향을 미친다는 사실을 확인할 수 있었다. 임원-관리직과 전문직에서 상대적으로 높은 소득을 가진 사람들이 많았으며, 여성과 소수 인종 그룹에서는 낮은 소득을 가진 사람들이 많았다. 이는 사회적, 경제적 불평등이 존재함을 시사한다.

모델링에서는 그리드 서치를 진행하여 가장 최적의 분류기와 하이퍼파라미터를 찾을 수 있었다. 처음 선택된 모델은 Decision Tree Classifier 모델이었으며, 상대적으로 높은 정확도와 안정적인 성능을 보였다. 하지만 더 나은 성능의 모델을 구현하고 싶어 다양한 알고리즘과 하이퍼 파라미터를 추가하여 다시 그리드 서치를 진행하여 랜덤 포레스트 분류 모델에서 유의미한 성능 향상을 확인할 수 있었다.

### · 소감

첫째, 이번 프로젝트를 통해 데이터 분석과 머신러닝 모델링에 대한 실질적인 경험을 쌓을 수 있었으며, 다양한 시각화를 통해 데이터를 다각도로 분석하는 방법을 배울 수 있었다. 중간 프로젝트 때는 직접적인 모델링 과정은 포함되지 않아서 아쉬웠는데 본 프로젝트에서는 데이터 수집, 전처리, 변환, 정규화, 시각화, 모델링, 평가, 하이퍼 파라미터 튜닝의 전 과정을 직접 진행 해보는 유의미한 경험을 할 수 있었다.

둘째, 이번 프로젝트를 통해 분석한 결과를 보면 사회적, 경제적 불평등을 명확히 확인할 수 있었다. 이번 경험을 바탕으로 향후 비슷한 주제를 선정하여 국내 인구 통계 데이터를 직접 분석하고 모델링까지 진행해보고 싶다.


마지막으로, 이번 프로젝트를 진행했던 코드를 캐글에 공유하여 upvote 를 5 개 이상을 받아 동메달을 획득하여 현재까지 공유했던 노트북 중 총 5 개 노트북에서 모두 동메달을 획득하였다. 이로 인해 캐글의 Expert 로 승급하게 되었는데 이 프로젝트 덕분에 데이터 분석에 대한 동기가 더욱 유발될 수 있었던 것 같다.

## Adult Census Income

642
New Notebook
Download (461 kB)


Data Card
Code (526)
Discussion (12)
Suggestions (0)

All
Your Work
Shared With You
Bookmarks
Hotness




**core-fa19-regression**  
Updated 4y ago  
6 comments · Will you be rich? +1

16




**Adult Census Income**  
Updated 1d ago  
0 comments · Adult Census Income

2




**Adult Census Income EDA & Prediction**  
Updated 8d ago  
2 comments · Adult Census Income

6
Bronze




**Adult Census Income**  
Updated 9d ago  
0 comments · Adult Census Income

4




**Admit HLCC fastai**  
Updated 5y ago  
1 comment · Adult Census Income

4




**ML Labs - 10 - UCI Adult Census Income**  
Updated 14d ago  
0 comments · Adult Census Income

2




**Starter: Adult Census Income 485fe2c0-9**  
Updated 5y ago  
0 comments · Adult Census Income

1



**Starter: Adult Census Income 14168246-c**  
Updated 5y ago  
0 comments · Adult Census Income

1



**Starter: Adult Census Income 387ee8a6-e**

0



psleon8245

# PSLeon

Student
Busan, Busan, South Korea
Joined 2 years ago · last seen in the past day



Notebooks Expert  
4,070 of 59,925

## 7. References

[1] PSLeon, Student Study Performance, <https://www.kaggle.com/datasets/uciml/adult-census-income>, 2024.05.23