

2018  
ARTIFICIAL  
INTELLIGENCE

TERM  
PROJECT 2  
Korea  
University

# UCI ML REPOSITORY DATA ANALYSIS USING SCIKIT-LEARN



2013130625 조성표

# UCI ML Repository Data Analysis using Scikit-Learn

SUNGPYO CHO

## ABSTRACT

본 프로젝트는 UCI Machine Learning Repository 의 Adult 데이터와 Breast cancer 데이터를 분석하고, 다양한 머신러닝 모델을 사용하여 비교 평가했습니다. 이를 위해 파이썬의 scikit-learn 라이브러리를 이용하여 직접 모든 코드를 구현하였습니다. 파이썬 외에도, OneR 알고리즘 사용을 위해 R 도 사용하였습니다. 데이터의 특징을 다각적인 측면에서 분석하고, Seaborn 패키지를 이용해 시각화했습니다.

머신러닝 모델 비교 평가를 위해, Adult 데이터에서는 총 8 개의 모델과 Breast cancer 데이터에서는 총 7 개의 모델을 사용하였습니다. kNN, 로지스틱 회귀, 결정트리, 그래디언트 부스팅 회귀 트리, MLP, SVM 등 여러 모델을 사용하였으며, 모델을 평가하기 위한 baseline 알고리즘으로 ZeroR 과 OneR 을 사용하였습니다. 그리고 이러한 다양한 기계학습 모델의 성능을 교차검증 데이터의 정확도, 테스트 데이터 정확도(만점=1)로 평가하였습니다. 그 결과를 표로 나타내면 다음과 같습니다.

Machine Learning (on Adult Data)	Best CV Accuracy	Test Set Accuracy
kNN(k=10)	0.832	0.824
로지스틱 회귀(C=3)	0.849	0.838
LASSO 회귀(alpha=0.003)	0.386	0.381
그래디언트 부스팅 회귀 트리	0.870	0.863
결정트리	0.849	0.836
SVM(C=1)	0.841	0.840
ZeroR	-	0.756
OneR	-	0.799

Machine Learning (on Breast Cancer Data)	Best CV Accuracy	Test Set Accuracy
kNN(k=9)	0.927	0.956
로지스틱 회귀(C=10)	0.960	0.956
결정트리	0.949	0.965
MLP	0.974	0.982
SVM	0.976	0.982
ZeroR	-	0.623
OneR	-	0.92

## 1. INTRODUCTION

본 프로젝트의 목적은 UCI Machine Learning Repository 의 Adult 데이터와 Breast cancer 데이터를 다양한 머신러닝 모델을 사용하여 학습하고, 새로운 데이터가 주어졌을 경우 정확한 예측을 하는 것입니다. Adult 데이터의 경우 개인의 여러 특성을 분석하여 수입이 5 만달러를 넘을지 아닐지 예측할 수 있으며, Breast cancer 데이터의 경우 암과 관련된 여러 수치를 분석하여 암의 발생을 예측할 수 있습니다.

이를 구현하기 위해서 Weka 가 아닌, 파이썬의 Scikit-learn 을 사용했습니다. 데이터 분석과 머신러닝 모델 학습 등을 모두 직접 구현했습니다. Scikit-learn 은 파이썬에서 가장 널리 사용되는 머신러닝 라이브러리로, 다양한 머신러닝 기법을 손쉽게 구현할 수 있는 것이 특징입니다. 그러나 OneR 알고리즘 학습과 평가에는 파이썬이 아닌 R 의 OneR 라이브러리를 사용했습니다. 구현한 파이썬 코드와 R 코드는 압축파일에 보고서와 함께 첨부했습니다.

## 2. REASONS FOR DATA SELECTION

프로젝트의 분석 대상으로 Adult 와 Breast Cancer 데이터셋을 선택한 것은 두 데이터셋 모두 attribute 수와 instance 수가 많으며, Multivariate 데이터이므로 분석할만한 가치가 있기 때문입니다. 또한 두 데이터셋 모두 매우 유명하고 기계학습에 전통적으로 쓰인 데이터로, 제 머신러닝 모델의 정확도를 다른 연구들의 결과와 비교하기 쉽다는 이점이 있습니다.

특히 Adult 데이터의 경우 수만개의 instance 가 존재하며, 대부분의 feature 들의 값이 범주형 데이터로 이루어져 있기 때문에 도전해보고 싶었습니다. 모든 값이 숫자로 된 데이터는 여러 번 다뤄봤지만, 인코딩을 통해 범주형 데이터를 변환해 본 적은 없기 때문에 더욱 매력적으로 느껴졌습니다. 또한 분석을 통해 개인의 소득에 영향을 주는 요소는 무엇인지, 어떤 계층의 사람들이 고소득을 얻는지와 같은 흥미로운 문제들의 답을 알아낼 수 있다는 것도 매력적이었습니다. 이 분석을 응용하면 사회학적, 경제학적으로 의미있는 연구 결과 또한 도출할 수 있기 때문입니다.

Breast Cancer 데이터는 암세포의 특성을 통해 암의 유발 여부를 판정할 수 있다는 점이 흥미로워 선택했습니다. 최근 Kaggle 에서는 이러한 질병을 예측하는 다양한 의료분야 competition 을 개최하고 있습니다. Breast cancer 데이터 분석은 추후 이러한 competition 에 도전하는 데에도 좋은 연습이 될 것이라고 생각했습니다.

## 3. DATA EXPLORATION

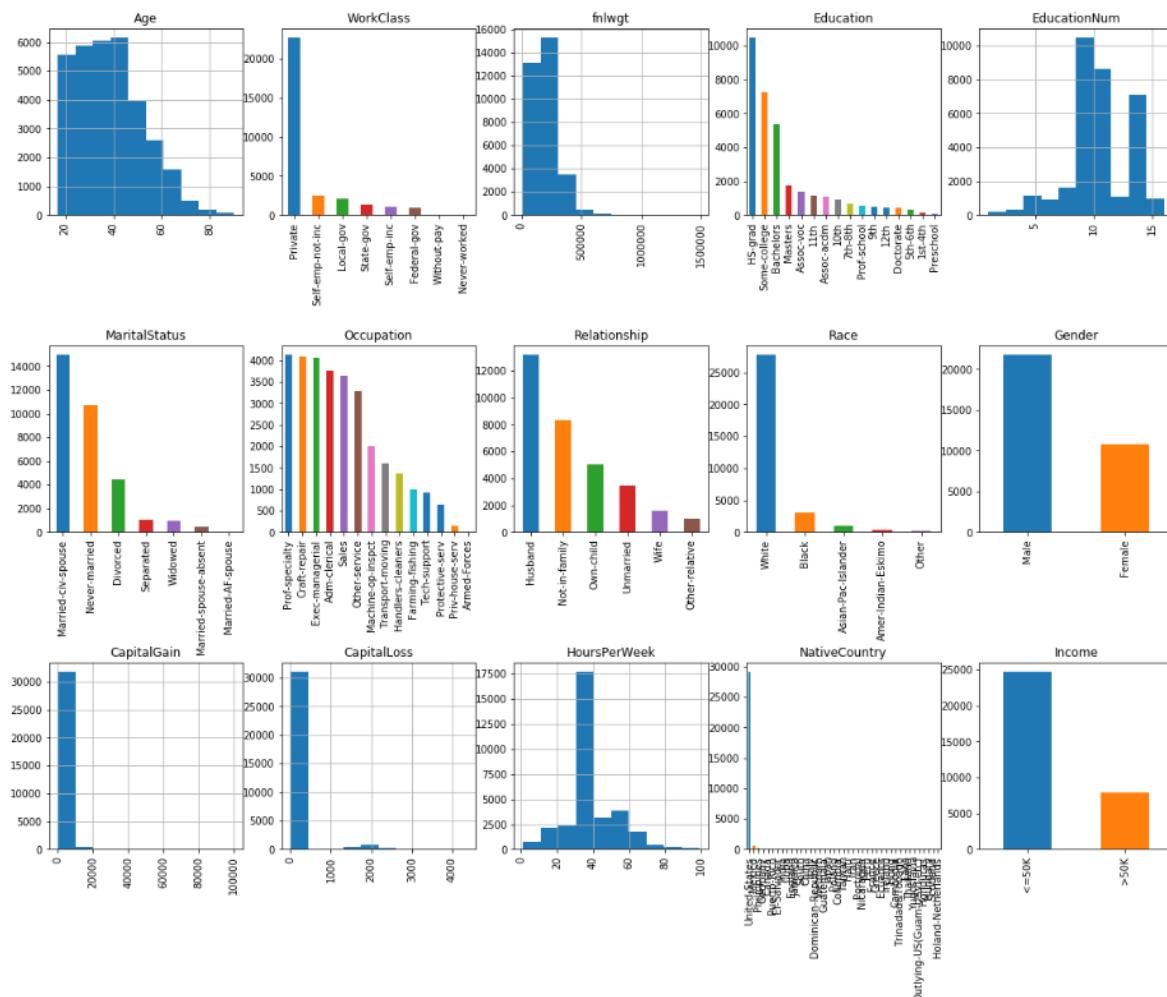
### 3-1. Adult 데이터 분석

우선 데이터를 분석하기 위해, pandas 의 read\_cv 를 사용하여 UCI ML Repository 에서 .data 파일을 읽었습니다.

Adult 데이터셋은 Age, WorkClass, fnlwgt, Education, EducationNum, MartialStatus, Occupation 등 15 개의 특성(attribute)과 32561 개의 Instance 가 존재합니다. 특히 마지막 15 번째 특성인 ‘Label’은 분류 결과로서, 해당 개인의 수입이 5 만달러를 넘는지 아닌지를 알려줍니다. 3 번째 특성인 ‘fnlwgt’는 어떤 특성인지 알기 어려운데, Chris Shoemaker 의 연구에 의해 수입과는 전혀 관련이 없는 특성이라는 것이 밝혀졌으므로 삭제했습니다.

데이터 분석 도중, 두 가지 특징을 발견할 수 있었습니다. 첫째로, 많은 특성들이 연속형(numerical) 특성이 아닌, 범주형 특성(categorical attribute) 입니다. 이 특성들을 제외하고 연속형 특성들에만 분류 알고리즘을 적용할 수는 없기 때문에, 범주형 특성들을 다른 방식으로 표현해야 했습니다. 둘째로, '?'로 표기된 missing value 가 존재합니다. 그래서 read\_csv 로 데이터를 읽어들일 때 na\_values='?'를 사용해 '?'를 NaN 으로 인식하도록 했습니다.

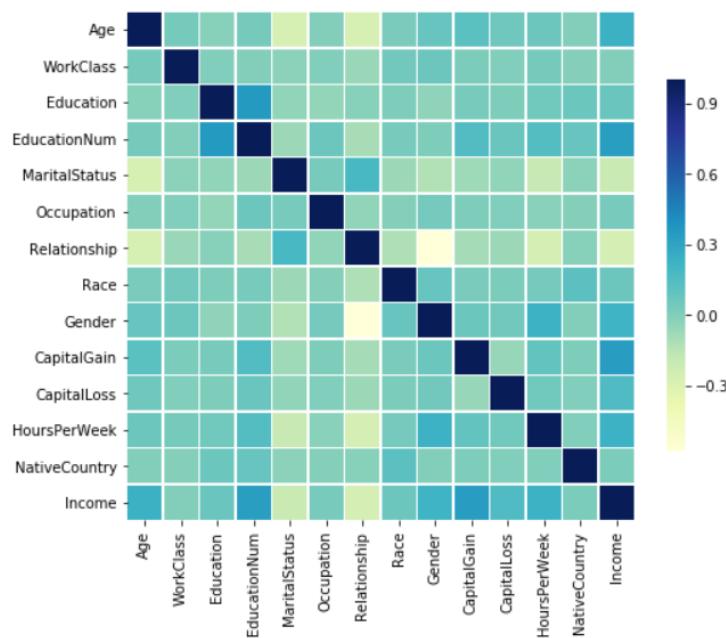
각 특성의 데이터마다 어떤 특징과 분포를 갖고 있는지 확인하기 위해, 아래와 같이 도표로도 나타내었습니다.



위 도표를 보면, 현재 데이터셋이 주로 미국 출신의 백인 남성에 편중되어 있다는 것을 알 수 있습니다. 또한, 예측 label(Income)이 <=50K 가 훨씬 많아 불균형적으로 구성되어 있다는 사실도 확인할 수 있습니다.

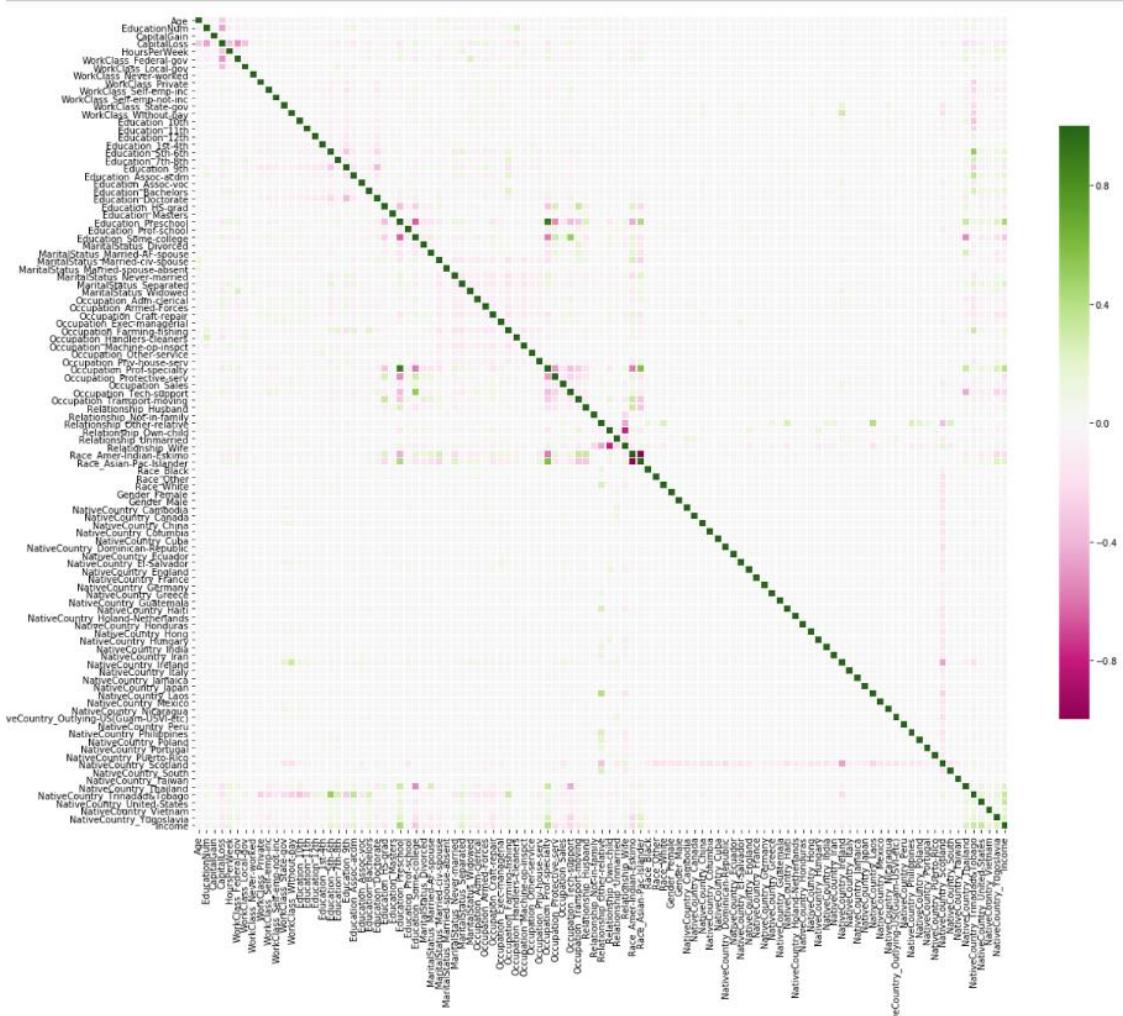
다음으로, `.isnull()`을 사용해 '?'으로 표시된 missing value 가 데이터상에 얼마나 있는지 살펴보았습니다. 그 결과, WorkClass에 1836 개, Occupation에 1843 개, NativeCountry에 583 개의 비교적 많은 missing data 가 존재했습니다. 이러한 missing value 를 지우는 것은 가치있는 정보를 삭제하는 것과 같으므로, 삭제하는 대신 다른 훈련 데이터로부터 missing value 들을 추측하여 impute 하였습니다. Impute 에는 'most frequent imputation'을 사용했습니다.

adult 데이터는 범주형 데이터를 많이 포함하고 있습니다. 이러한 범주형 데이터를 기계학습에서 처리하기 위해서는, 이를 숫자로 바꾸는 작업이 필요합니다. 그래서 Scikit-learn 의 LabelEncoder 와 OneHotEncoder 클래스를 이용한 One-Hot-Encoding 을 사용했습니다.



먼저 LabelEncoder 로 인코딩 후, 특성들 간 상관관계를 Seaborn 라이브러리를 활용해 시각화했습니다. 위 도표에서 특히 눈에 띄는 것은 Education 과 EducationNum 인데, 실제 이를 비교해보면 EducationNum 은 개인의 학력(Education)을 단순히 숫자로 표현한 수치라는 것을 알 수 있습니다.

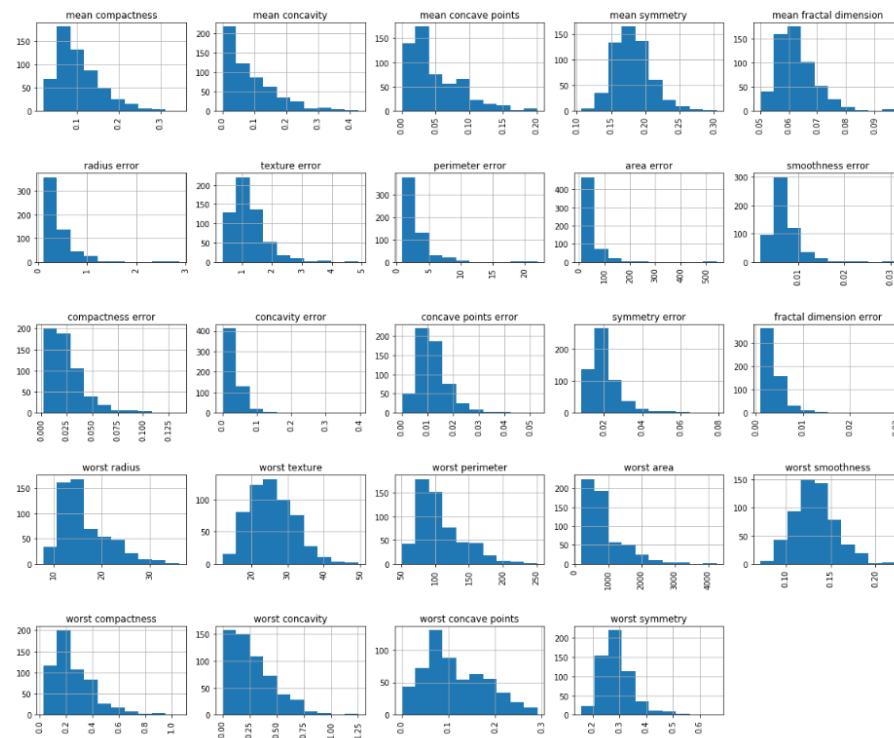
One Hot Encoding 후에는 원래 14 개(fnlwgt 제외)였던 특성이 105 개로 늘어났습니다. 그래서 이 특성들 간의 상관관계를 확인하기 위해 Seaborn 을 이용해 heatmap 을 그렸습니다. 대부분의 특성은 0 의 상관관계를 보이지만, 양의 상관관계(녹색)과 음의 상관관계(적색)가 명확한 특성들을 여럿 확인할 수 있습니다.



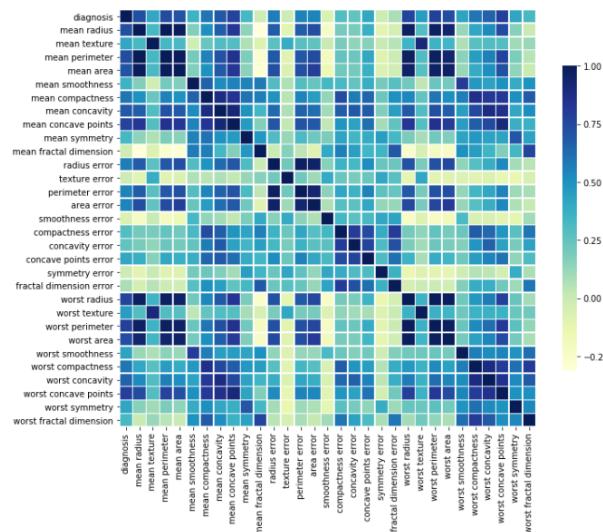
### 3-2. Breast cancer 데이터 분석

유방암 데이터 또한 Adult 데이터와 같은 방식으로 분석했습니다. 먼저, 데이터를 읽어들여 확인한 결과 32 개의 특성과 569 개의 예제가 존재한다는 것을 알 수 있었습니다. 32 개의 특성은 id, diagnosis, mean radius, mean texture 등으로, 암의 크기, 지름 등 환자의 암의 특성과 그에 따른 판정 결과를 포함합니다. Diagnosis 중 M(malignant)와 B(benign)의 개수는 각각 212, 357 로서 양성 환자가 212 명(37.26%), 음성 환자는 357 명(62.74%)이었습니다.

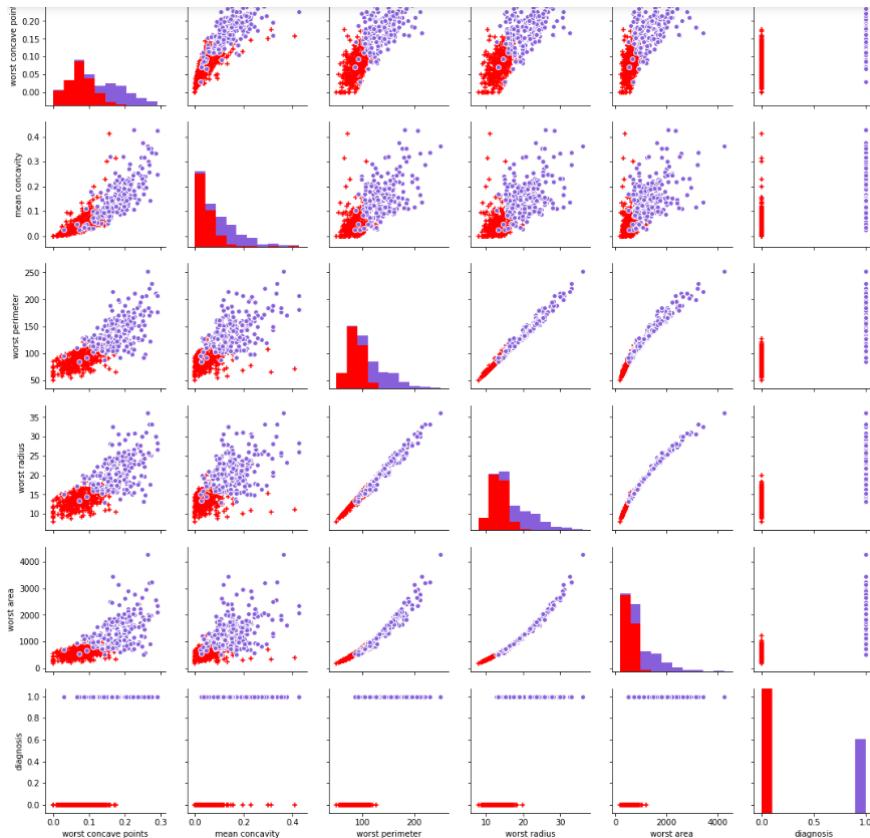
각 feature의 값은 모두 크기와 같은 숫자 측정값이기 때문에, Adult 데이터에서 했던 것과 같은 One Hot Encoding은 진행하지 않았습니다. Missing value 또한 존재하지 않습니다.



Matplotlib을 이용해 각 특성의 값이 어떤 분포를 가지고 있는지 살펴본 결과 위 도표와 같았습니다. 데이터가 가우시안 분포를 이루고 있는지 여부는 머신러닝의 결과에도 영향을 끼치기 때문에, 확인할 필요가 있습니다. 위 시각화 도표에는 feature 중 약 반절이 가우시안 분포를 보임을 확인할 수 있고, 변수들 값이 소수점부터 심지어는 몇천까지 혼재되어 있습니다. 그래서 feature scaling(normalize)을 해야 할 것으로 예상하였으나, 이를 추후 기계학습에 적용해 본 결과 오히려 성능이 더 떨어졌습니다. 이는 normalization이 모든 특성들을 '동등'하게 취급하는데, 우리의 유방암 데이터셋은 특정 특성들이 높은 상관관계를 지니고 있어서 normalization이 적합하지 않기 때문입니다. 따라서 feature scaling은 배제했습니다.



아래 도표에서는 우리가 갖고 있는 데이터셋이 대부분 양의 상관관계를 지니고 있다는 것을 확인할 수 있습니다. 이 중 특히 강한 상관관계를 갖고 있다고 보여지는 독립적인 6 개의 특징들의 관계를 그려보겠습니다. 6 개의 특징으로는 'worst concave points', 'mean concavity', 'worst perimeter', 'worst radius', 'worst area', 그리고 'diagnosis'를 선정했습니다.



위 산점도는 6 개의 특성들 사이의 관계를 표로 나타냅니다. 특히 강한 상관관계를 가지고 있었던 특성들이었던 만큼, Malignant 와 Benign 을 쉽게 구분할 수 있었다는 것을 알 수 있습니다. 이러한 산점도 행렬 도표는 데이터셋 속 변수들 간의 관계가 어떻게 형성되어있는지를 알 수 있는 쉬운 지표가 됩니다.

## 4. EXPREIMENT DESIGN METHOD

### 4-1. Adult 데이터 실험 설계

1. 전처리: Adult 데이터의 분포와 특징을 알기 위해, 데이터 전처리와 시각화를 진행합니다. 먼저 missing value 가 있는지 확인합니다. Missing value 가 얼마나 많은지에 따라 그냥 삭제할지, 혹은 적절한 imputation 을 할지 결정합니다. 본 프로젝트에서는 그 column 에서 가장 자주 등장하는 값으로 impute 했습니다.

그 후 범주형 데이터가 존재하는지 확인하고, 있다면 scikit-learn 의 LabelEncoder 와 OneHotEncoder 클래스를 이용해 One-Hot-Encoding 을 하고 없다면 따로 인코딩을 하지 않습니다. 마지막으로 특성 간의 상관관계를 시각화합니다. 이 과정에서 필요없거나 중복되는 특성이 발견되면 삭제합니다. 실제로 이 과정을 따라 ‘fnlwgt’ 와 같은 특성을 삭제했습니다.

2. 기계학습 적용: 데이터에 다양한 기계학습 모델을 적용하는 과정입니다. 데이터 간의 scale 이 필요할 경우 StandardScaler 등으로 데이터 값을 조정한 뒤, 파이프라인을 만들어 각 모델마다 다양한 hyper-parameter 를 조합하며 최상의 결과를 찾습니다. 본 프로젝트에서는 Knn, 로지스틱회귀, LASSO 회귀, 그래디언트 부스팅 회귀트리, 결정트리, 서포트 벡터 머신의 6 개 기계학습 모델을 사용했습니다.
3. 결과 비교분석: 기계학습 모델이 도출한 결과끼리 비교분석을 진행합니다. 또한, OneR 과 ZeroR 알고리즘을 baseline 으로 사용해 기계학습 모델의 성능을 평가합니다. 가장 좋은 성능을 낸 모델을 찾고, 그 이유를 분석합니다.
4. 결과 개선: 기계학습 모델의 성능이 baseline 보다 낮다면, parameter 변경 혹은 scaling 등의 여러가지 방법을 사용해 결과를 향상시킵니다.

## 4-2. Breast cancer 데이터 실험 설계

1. 전처리: 데이터 전처리와 시각화를 진행합니다. 먼저 Missing value 가 얼마나 많은가에 따라 그냥 삭제할지, 혹은 적절한 imputation 을 할지 결정합니다. Breast cancer 는 missing value 가 없기 때문에, 따로 imputation 은 하지 않았습니다.

그 후 범주형 데이터가 존재하는지 확인하고, 있다면 One-Hot-Encoding 을 하고 없다면 따로 인코딩을 하지 않습니다. 마지막으로 특성 간의 상관관계를 시각화합니다. 이 과정에서 필요없거나 중복되는 특성이 발견되면 삭제합니다. 결과적으로 ‘id’와 같이 결과 예측과 상관없다고 생각되는 특징을 삭제했습니다.

2. 기계학습 적용: 데이터에 다양한 기계학습 모델을 적용하는 과정입니다. 데이터 간의 scale 이 필요할 경우 StandardScaler 등으로 데이터 값을 조정한 뒤, 파이프라인을 만들어 각 모델마다 다양한 hyper-parameter 를 조합하며 최상의 결과를 찾습니다. Breast cancer 데이터에는 Knn, 로지스틱회귀, 결정트리, 신경망, 서포트 벡터 머신의 5 개 기계학습 모델을 사용했습니다.
3. 결과 비교분석: 기계학습 모델이 도출한 결과 비교분석을 진행합니다. 또한, OneR 과 ZeroR 알고리즘을 baseline 으로 사용해 기계학습 모델의 성능을 평가합니다. 가장 좋은 성능을 낸 모델을 찾고, 그 이유를 분석합니다.
4. 결과 개선: 기계학습 모델의 성능이 baseline 보다 낮다면, parameter 변경 혹은 scaling 등의 여러가지 방법을 사용해 결과를 향상시킵니다.

## 5. CLASSIFIER COMPARISON ANALYSIS

### 5-1. Adult 데이터에 대한 기계학습 기법 비교분석

데이터에 기계학습을 적용하기 전, 해야 할 몇 가지 일들이 있습니다. 먼저 현재 주어진 데이터가 3 만 개가 넘으므로 계산이 오래 걸리기 때문에 그 중 10000 개만 샘플링을 했습니다. 사실 10000 개도 상당히 많은 데이터일 뿐만 아니라 3 만개 전체에 기계학습을 적용할 때와 비교해 성능에서 그렇게 큰 차이가 나지 않습니다. 또, X 와 y 데이터를 나누어야 합니다. Income 이 예측하고자 하는 target label 이므로, label column 을 y 로 지정해주고, 그 외의 column 들은 모두 X 로 지정합니다.

데이터를 훈련 데이터와 테스트 데이터로 나누는(split) 과정 또한 빼놓을 수 없습니다. 본 프로젝트에서는 훈련 데이터를 80%, 테스트 데이터를 20% 비중으로 나누었습니다. 이후 X\_train, X\_test 에 StandardScaler 를 적용하였습니다. 이는 feature 사이의 값을 균일하게 조정해주어 기계학습 모델의 성능을 더욱 높이는 효과가 있기 때문입니다. 그렇다면 이제 본격적으로 8 가지의 기계학습 모델을 적용해보도록 하겠습니다.

#### 1. K Nearest Neighbors

kNN 은 가장 널리 쓰이는 지도학습의 분류기 중 하나입니다. kNN 에서 중요한 것은 k, 즉 nearest neighbor 를 몇 개로 설정해줄 것인가입니다. 보통 k=1 은 의미있는 분류 결과를 만들어내지 못한다고 알려져 있으므로, 2 부터 10 까지 k 값을

```
Test set score: 0.824
Best Parameter: {'n_neighbors': 10}
Best CV score: 0.832
<<Classification Result>>
      precision    recall   f1-score   support
       0.0      0.85     0.93     0.89     1512
       1.0      0.70     0.49     0.57      488
avg / total   0.81     0.82     0.81     2000
```

적용하여 그 결과를 비교해 보았습니다. 그러나 2, 3, 4, 5, 6, 7, 8, 9, 10 의 값을 모두 하나 하나씩 적용하는데에는 시간도 많이 걸릴 뿐더러 비효율적이므로, GridSearchCV 를 사용합니다. Scikit-learn 의 GridSearchCV 는 파라미터를 한꺼번에 병렬 계산하고 k-fold cross validation 도 같이 계산하여, 최적의 성능을 보이는 파라미터를 도출해냅니다. Knn 을 그리드 서치에 적용한 결과 k 의 값이 10 일 때 최고의 교차검증 정확도 0.832, 테스트 데이터 정확도 0.824 라는 높은 점수가 나왔습니다. 자세한 결과는 다음과 같습니다.

## 2. 로지스틱 회귀

로지스틱 회귀는 가장 기본적이며, 자주 쓰이는 방법입니다. 특히 로지스틱 회귀는 인코딩한 adult 데이터와 같은 희소행렬에 효과적이라고 알려져 있습니다. 로지스틱 회귀에서 중요한 것은 C, 즉 regularization 파라미터입니다. 보통 0.1, 0.3, 1, 3 과 같이 3 배씩 값을 조정해 비교한다고 알려져 있으므로, 'C'를 [0.1, 0.3, 1, 3, 10]로 조합하여 그 결과를 비교해 보았습니다. 그 결과 교차검증 데이터 정확도는 0.849, 테스트 데이터 정확도는 0.838으로 kNN 보다 조금 더 좋은 성능을 보였습니다. 자세한 결과는 다음과 같습니다.

```
Test set score: 0.838
Best Parameter: {'C': 0.3}
Best CV score: 0.849
<<Classification Result>>
      precision    recall   f1-score  support
          0.0       0.87      0.92      0.90     1512
          1.0       0.70      0.58      0.64      488
avg / total       0.83      0.84      0.83     2000
```

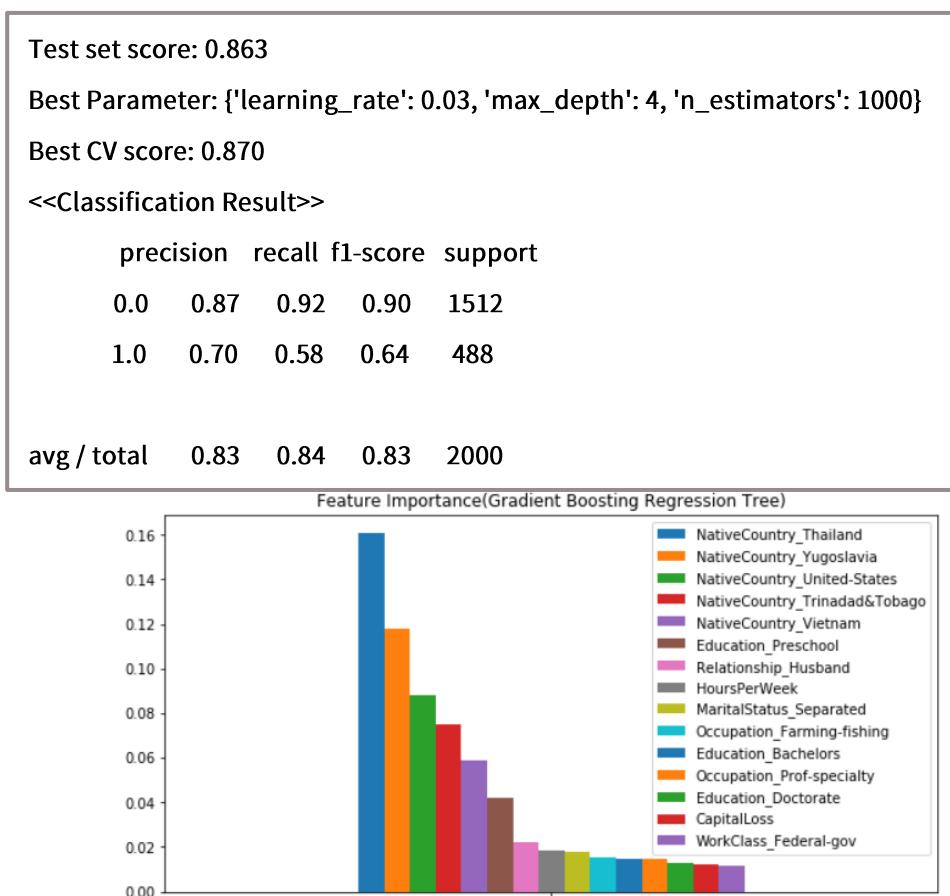
## 3. LASSO 회귀

LASSO 회귀는 Ridge 회귀와 달리 regularization 방법으로 파라미터의 절대값을 더하는 방법을 씁니다. LASSO 회귀에서 중요한 것은 alpha, 즉 regularization 파라미터입니다. 로지스틱 회귀의 C처럼 3 배씩 값을 조정해 비교한다고 알려져 있으므로, 'alpha'를 [0.001, 0.003, 0.01, 0.03]로 조합하여 그 결과를 비교해 보았습니다. LASSO는 로지스틱 회귀의 연장선상에서 시험적으로 적용해 보았는데, 41%라는 baseline 보다 낮은 accuracy 가 나왔습니다. 추후 잘못된 곳이 없는지 다시 검사하여 예측률을 개선하겠습니다.

```
Test set score: 0.381
Best Parameter: {'alpha': 0.003}
Best CV score: 0.386
```

## 4. 그래디언트 부스팅 회귀 트리

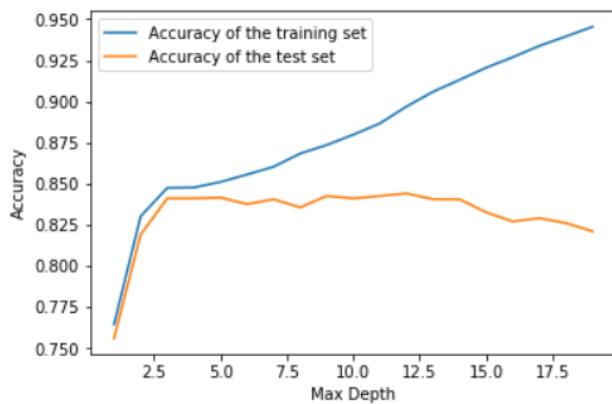
그래디언트 부스팅 회귀 트리는 여러 개의 결정 트리를 묶어 강력한 모델을 만드는 앙상블 방법입니다. 이름은 회귀이지만 회귀와 분류 모두에 사용할 수 있습니다. 이전 트리의 오차를 보완하는 'Boosting'기법과 강력한 사전 가지치기가 쓰여 메모리를 적게 사용하고 예측도 빠릅니다. 여기서도 마찬가지로 GridSearchCV를 이용한 병렬 연산을 통해 가장 좋은 성능을 보이는 hyperparameter의 조합을 계산했습니다. Hyperparameter는 'learning\_rate'를 [0.01, 0.03], 'n\_estimators'를 [100, 1000], 'max\_depth'를 [3, 4]로 설정해 조합했습니다. 교차검증은 5-fold 교차검증을 사용했습니다. 그 결과 교차검증 데이터 정확도는 0.870, 테스트 데이터 정확도는 0.863으로 가장 좋은 성능을 보였습니다. 자세한 결과는 다음과 같습니다.



Scikit\_learn에서 제공하는 결정트리의 `feature_importances_ attribute`를 이용해 그래디언트 부스팅 트리의 중요한 특성들도 시각화했습니다. 특성은 105개 존재하나 대부분 0의 값을 갖고 있으므로, 가장 중요한 15개의 특성을 순서대로 나열했습니다. 이 결정트리에 의해 가장 중요하다고 간주된 특성은 `NativeCountry_Thailand`로, 0.160912였습니다.

## 5. 결정 트리

결정 트리 모델을 학습하기 전에 앞서서, 트리의 최대깊이와 훈련 및 테스트 데이터의 정확도 사이의 관계를 도표로 시각화했습니다. 그 결과 아래와 같은 결과가 나왔습니다. 트리의 최대 깊이가 깊어질수록 training set 의 정확도는 1에 가깝게 계속 증가하지만, test set 의 정확도는 증가하기는커녕 오히려 줄어들고 있는 것을 확인할 수 있습니다. 이는 깊이가 깊은 트리는 훈련 데이터에 과적합되어 매우 잘 분류하지만, 정작 새로운 데이터(테스트 데이터)가 주어졌을 경우 잘 분류하지 못하기 때문입니다. 즉, 깊이가 깊어질수록 Overfitting 이 발생하고 있다는 것을 확인할 수 있습니다.

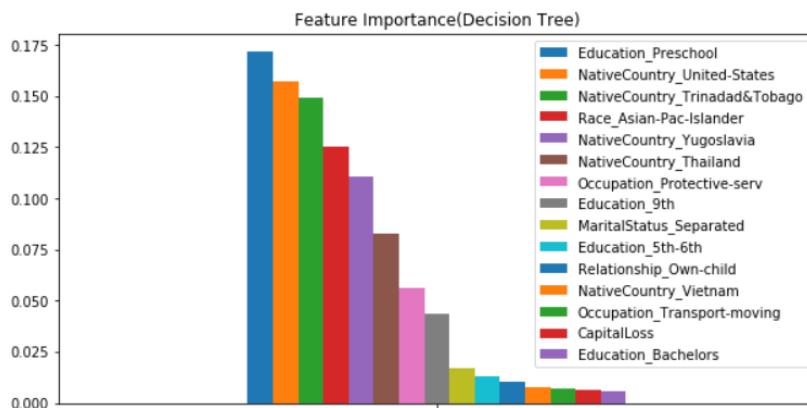


테스트 데이터가 가장 높은 정확도를 보이는 구간은 max depth 가 3~8 사이인 구간이므로, GridSearchCV 를 이용한 병렬 연산을 통해 가장 높은 정확도를 보이는 depth 의 값을 계산했습니다. 따라서 Hyperparameter 는 max\_depth 를 [3, 4, 5, 6, 7, 8], max\_feature 를 ['auto', 'sqrt', 'log2', None] 값을 설정해 조합했습니다. 교차검증은 5-fold 교차검증을 사용했습니다. 그 결과 교차검증 데이터 정확도는 0.849, 테스트 데이터 정확도는 0.836 으로 그래디언트 부스팅 회귀 트리보다는 낮았습니다. 자세한 결과는 다음과 같습니다.

```

Test set score: 0.836
Best Parameter: {'max_depth': 8, 'max_features': None}
Best CV score: 0.849
<<Classification Result>>
      precision    recall   f1-score   support
          0.0       0.89      0.90      0.89     1512
          1.0       0.67      0.65      0.66      488
avg / total       0.83      0.84      0.83     2000

```



Scikit\_learn에서 제공하는 결정트리의 `feature_importances_` attribute를 이용해 결정트리의 중요한 특성들도 시각화했습니다. 특성은 105개 존재하나 대부분 0의 값을 갖고 있으므로, 가장 중요한 15개의 특성을 순서대로 나열했습니다. 이 결정트리에 의해 가장 중요하다고 간주된 2개의 특성은 `Education_Preschool`, `NativeCountry_United-States`으로, 각각 0.172055와 0.157492였습니다. 바로 앞에서 다룬 그래디언트 부스팅 회귀트리의 도표와 비교해 보면 특성들이 대부분 일치한다는 것을 알 수 있습니다.

## 6. 서포트 벡터 머신

서포트 벡터 머신은 분류 시 다른 labeled 데이터 사이에 최대한 큰 간격(margin)을 확보하려는 분류기로, 로지스틱회귀의 비용함수에 조금 변경을 가한 모델입니다. SVM에서 중요한 것은 C, 즉 regularization 파라미터입니다. 이는 로지스틱 회귀의 C처럼 3배씩 값을 조정해 비교한다고 알려져 있으므로, 'C'를 [0.01, 0.03, 0.1, 0.3, 1, 3, 10], 'decision\_function\_shape'를 ['ovo', 'ovr']로 조합하여 그 결과를 비교해 보았습니다. 그래서 마찬가지로 GridSearchCV를 이용한 병렬 연산을 통해, 가장 좋은 성능을 보이는 hyperparameter의 조합을 계산했습니다. 교차검증은 5-fold 교차검증을 사용했습니다. 그 결과 교차검증 데이터 정확도는 0.841, 테스트 데이터 정확도는 0.840으로 로지스틱 회귀와 kNN 보다 좋은 성능을 보였습니다. 자세한 결과는 다음과 같습니다.

```

Test set score: 0.840
Best Parameter: {'C': 1, 'decision_function_shape': 'ovo'}
Best CV score: 0.841
<<Classification Result>>
      precision    recall   f1-score   support
          0.0      0.86      0.94      0.90     1501
          1.0      0.75      0.54      0.63      499
avg / total      0.83      0.84      0.83     2000

```

## 7. Baseline: ZeroR

ZeroR 알고리즘은 직접 구현했습니다. ZeroR은 정말 간단한 알고리즘이지만, 75.6%라는 정확도가 나오는 것을 확인할 수 있었습니다. 이는 현재 갖고 있는 데이터가 0(50K 이하)에 편중되어 있어 무조건 0으로 예측하더라도 대부분 맞기 때문입니다.

The Test accuracy of ZeroR is: 0.756

## 8. Baseline: OneR

OneR은 파이썬으로 제대로 구현되어 있는 것이 없고, numerical value들을 범주형 특성으로 바꾸어주어야 하기 때문에 어렵습니다. 따라서 R의 OneR 라이브러리를 사용했습니다. R 코드는 첨부되어 있는 'OneR Algorithm'(R로 작성)이라는 파일을 확인하시면 됩니다. OneR 알고리즘은 baseline이고 룰이 간단하지만, 79.91%라는 괜찮은 accuracy 수치를 보였습니다. 적용된 룰, 정확도는 다음과 같습니다.

**Rule:** If Cap\_Gain = (-100,3.97e+03] then Income = <=50K

If Cap\_Gain = (3.97e+03,1e+05] then Income = >50K

20815 of 26048 instances classified correctly (79.91%)

### 5-2. Breast cancer 데이터에 대한 기계학습 기법 비교분석

데이터에 기계학습을 적용하기 전, 먼저 X와 y 데이터를 나누어야 합니다. 'diagnosis'가 예측하고자 하는 target label이므로, diagnosis column만을 y로 지정해주고, 그 외의 column들은 모두 X로 지정합니다.

데이터를 훈련 데이터와 테스트 데이터로 나누는(split) 과정 또한 빼놓을 수 없습니다. Adult 데이터와 마찬가지로 훈련 데이터를 80%, 테스트 데이터를 20% 비중으로 나누었으나, StandardScaler는 따로 적용하지 않고 신경망과 SVM에만 적용하였습니다. 그렇다면 이제 본격적으로 7 가지의 기계학습 모델을 적용해보도록 하겠습니다. Adult 데이터 분석 때와 많은 모델이 겹치므로, 앞에서 설명한 모델별 특징은 생략하도록 하겠습니다.

#### 1. K Nearest Neighbors

kNN에서 중요한 것은 k, 즉 nearest neighbor를 몇 개로 설정해줄 것인가입니다. 그래서 k 값을 1부터 10까지 모두 조합하여 적용하여 그 결과를 비교해 보았습니다. 그러나 이 값들을 모두 하나 하나씩 적용하는데에는 시간도 많이 걸릴 뿐더러 비효율적이므로, GridSearchCV를 사용했습니다. kNN을 그리드 서치에 적용한 결과 k의 값이 9일 때 교차검증 정확도 0.927, 테스트 데이터 정확도 0.956이라는 매우 높은 점수가 나왔습니다. 자세한 결과는 다음과 같습니다.

```

Test set score: 0.956
Best Parameter: {'algorithm': 'auto', 'n_neighbors': 9}
Best CV score: 0.927
<<Classification Result>>
precision recall f1-score support
0      0.95    0.99    0.97     71
1      0.97    0.91    0.94     43

avg / total   0.96    0.96    0.96    114

```

## 2. 로지스틱 회귀

로지스틱 회귀는 가장 기본적이며, 자주 쓰이는 방법입니다. 로지스틱 회귀에서 중요한 것은 C, 즉 regularization 파라미터입니다. 보통 0.1, 0.3, 1, 3과 같이 3 배씩 값을 조정해 비교한다고 알려져 있으므로, 'C'를 [0.1, 0.3, 1, 3, 10]로 조합하여 그 결과를 비교해 보았습니다. 그 결과 C=10일 때 교차검증 데이터 정확도는 0.960, 테스트 데이터 정확도는 0.956으로 kNN 보다 더 좋은 성능을 보였습니다. 자세한 결과는 다음과 같습니다.

```

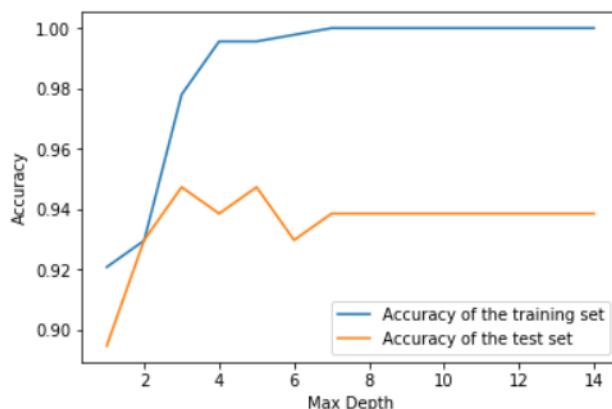
Test set score: 0.956
Best Parameter: {'C': 10}
Best CV score: 0.960
<<Classification Result>>
precision recall f1-score support
0      0.95    0.99    0.97     71
1      0.97    0.91    0.94     43

avg / total   0.96    0.96    0.96    114

```

### 3. 결정 트리

결정 트리 모델을 학습하기 전에 앞서서, 트리의 최대깊이와 훈련 및 테스트 데이터의 정확도 사이의 관계를 도표로 시각화했습니다. 그 결과 아래와 같은 결과가 나왔습니다. 트리의 최대 깊이가 깊어질수록 training set 의 정확도는 1 이 되어 완전히 fit 되지만, test set 의 정확도는 증가하기는커녕 오히려 감소하고 있는 것을 확인할 수 있습니다. 이는 깊이가 깊은 트리는 훈련 데이터에 과적합되어 매우 잘 분류하지만, 정작 새로운 데이터(테스트 데이터)가 주어졌을 경우 잘 분류하지 못하기 때문입니다. 즉, 깊이가 깊어질수록 Overfitting 이 발생하고 있다는 것을 확인할 수 있습니다.



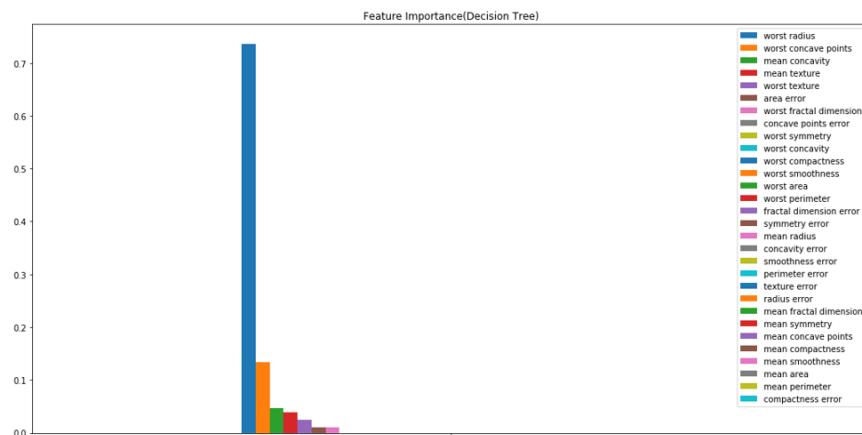
테스트 데이터가 가장 높은 정확도를 보이는 구간은 max depth 가 3~5 사이인 구간이므로, GridSearchCV를 이용한 병렬 연산을 통해 가장 높은 정확도를 보이는 depth 의 값을 계산했습니다. 따라서 Hyperparameter 는 max\_depth 를 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], max\_feature 를 ['auto', 'sqrt', 'log2', None] 값을 설정해 조합했습니다. 교차검증은 5-fold 교차검증을 사용했습니다. 그 결과 교차검증 데이터 정확도는 0.949, 테스트 데이터 정확도는 0.965 으로 로지스틱 회귀보다도 높은 성능을 보였습니다. 자세한 결과는 다음과 같습니다.

```

Test set score: 0.965
Best Parameter: {'max_depth': 3, 'max_features': 'auto'}
Best CV score: 0.949
<<Classification Result>>
      precision    recall   f1-score  support
          0       0.95     1.00     0.97      71
          1       1.00     0.91     0.95      43

avg / total   0.97    0.96    0.96    114

```



Scikit\_learn에서 제공하는 결정트리의 `feature_importances_ attribute`를 이용해 결정트리의 중요한 특성들도 시각화했습니다. 30개의 특성을 순서대로 나열했습니다. 이 결정트리에 의해 가장 중요하다고 간주된 특성은 `worst radius`로, 0.736487이었습니다. Adult 데이터 분석 때의 0.19와 같은 수치와 비교하면 굉장히 높고, 그만큼 결정트리 분류에 큰 영향을 미친다는 것을 알 수 있습니다.

## 4. 신경망(MLP)

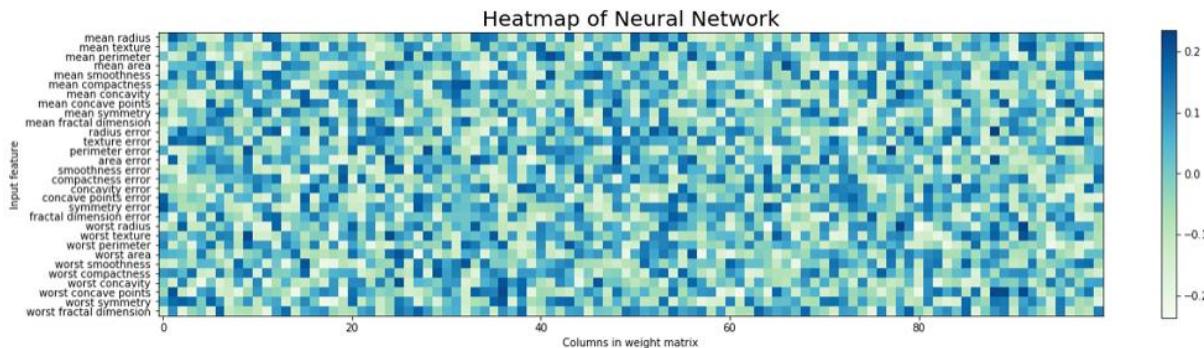
신경망은 최근 가장 유명해진 지도 학습 방법입니다. 하지만 복잡하고, 그만큼 조정해 주어야 할 hyperparameter가 많은 것이 단점입니다. 그래서 GridSearchCV를 이용한 병렬 연산을 통한 조합이 필수적입니다. 파라미터 조합 `param_mlp`를 `{'activation': ['logistic', 'tanh', 'relu'], 'solver': ['lbfgs', 'sgd'], 'alpha': [0.0001, 0.0003, 0.001, 0.003], 'max_iter': [1000, 1500, 2000]}`과 같이 주어 조합해 본 결과, `{'activation': 'tanh', 'alpha': 0.0001, 'max_iter': 1000, 'solver': 'sgd'}`

```

Test set score: 0.982
Best Parameter: {'activation': 'tanh', 'alpha': 0.0001, 'max_iter': 1000, 'solver': 'sgd'}
Best CV score: 0.974
<<Classification Result>>
      precision    recall   f1-score  support
          0       0.97     1.00     0.99      71
          1       1.00     0.95     0.98      43
avg / total       0.98     0.98     0.98     114

```

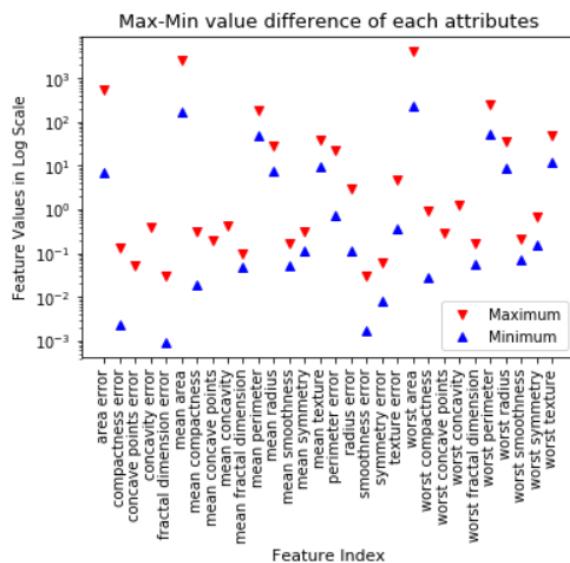
조합이 최고의 성능을 보였습니다. 교차검증은 5-fold 교차검증을 사용했습니다. 그 결과 교차검증 데이터 정확도는 0.974, 테스트 데이터 정확도는 0.982라는 엄청난 성능을 보였습니다. 자세한 결과는 다음과 같습니다.



또한 이 조합을 사용해, 위와 같이 특성들 간의 상관관계를 Heatmap 으로 나타내었습니다.

## 5. 서포트 벡터 머신

유방암 데이터에서 X 값의 scaling 을 진행하지 않고 서포트 벡터 머신을 사용할 경우, 훈련 데이터의 정확도는 1 인데 반해 테스트 데이터의 정확도는 0.623 으로 매우 낮은 Overfitting 이 발생합니다. 이는 아래 그림과 같이 각 변수들의 데이터 사이에 최소와 최대의 차가 크기 때문입니다. 그래서 서포트 벡터 머신에 데이터를 적용하기 전, StandardScaler 를 통해 데이터를 scale 해주는 과정을 거쳤습니다.



서포트 벡터 머신은 SVM 에서 중요한 것은 C, 즉 regularization 파라미터입니다. 이는 로지스틱 회귀의 C 처럼 3 배씩 값을 조정해 비교한다고 알려져 있으므로, 'C'를 [0.01, 0.03, 0.1, 0.3, 1, 3, 10], 'decision\_function\_shape'를 ['ovo', 'ovr']로

조합하여 그 결과를 비교해 보았습니다. 그래서 GridSearchCV 를 이용한 병렬 연산을 통해, 가장 좋은 성능을 보이는 hyperparameter 의 조합을 계산했습니다. 교차검증은 5-fold 교차검증을 사용했습니다. 그 결과 교차검증 데이터 정확도는

```
Test set score: 0.982
Best Parameter: {'C': 3, 'decision_function_shape': 'ovo'}
Best CV score: 0.976
<<Classification Result>>
precision    recall   f1-score  support
          0       0.97     1.00     0.99      71
          1       1.00     0.95     0.98      43
avg / total     0.98     0.98     0.98     114
```

0.976, 테스트 데이터 정확도는 0.982로 신경망과 비슷하거나 좋은 성능을 보였습니다. 자세한 결과는 다음과 같습니다.

## 6. Baseline: ZeroR

ZeroR 알고리즘은 직접 구현했습니다. ZeroR은 정말 간단한 알고리즘답게, 62.3%라는 낮은 정확도가 나오는 것을 확인할 수 있었습니다.

```
The Test accuracy of ZeroR is: 0.623
```

## 7. Baseline: OneR

OneR은 파이썬으로 제대로 구현되어 있는 것이 없고, numerical value 들을 범주형 특성으로 바꾸어주어야 하기 때문에 어렵습니다. 따라서 R 의 OneR 라이브러리를 사용했습니다. 비록 baseline 이고 아래와 같이 룰이 간단하지만, 92.09%라는 꽤 높은 accuracy 를 보였습니다. 적용된 룰, 정확도는 다음과 같습니다.

```
Rule: If worst perimeter = (50.2,110] then diagnosis = B
If worst perimeter = (110,251] then diagnosis = M
419 of 455 instances classified correctly (92.09%)
```

## 6. CONCLUSION

지금까지 UCI Machine Learning Repository 의 Adult 데이터와 Breast cancer 데이터를 분석하고, 다양한 머신러닝 모델을 사용하여 비교 평가했습니다. 이를 위해 파이썬의 scikit-learn 라이브러리와 R의 OneR 알고리즘을 사용해 코드를 구현했습니다. 또한 Matplotlib 와 Seaborn 을 이용해 머신러닝 모델의 결과에서 Feature Importance, Heatmap 등을 이끌어내며 의미있는 정보를 찾아내었습니다. 그래서 다음 표와 같은 결과를 얻었습니다.

Machine Learning (on Adult Data)	Best CV Accuracy	Test Set Accuracy
kNN(k=10)	0.832	0.824
로지스틱 회귀(C=3)	0.849	0.838
LASSO 회귀(alpha=0.003)	0.386	0.381
그래디언트 부스팅 회귀 트리	0.870	0.863
결정트리	0.849	0.836
SVM(C=1)	0.841	0.840
ZeroR	-	0.756
OneR	-	0.799

Machine Learning (on Breast Cancer Data)	Best CV Accuracy	Test Set Accuracy
kNN(k=9)	0.927	0.956
로지스틱 회귀(C=10)	0.960	0.956
결정트리	0.949	0.965
MLP	0.974	0.982
SVM	0.976	0.982
ZeroR	-	0.623
OneR	-	0.92

머신러닝 모델 비교 평가를 위해 사용한 10 가지의 다른 모델들은 모두 다른 특성과 성능을 보여주었습니다. 특히 Adult 데이터셋에서는 그래디언트 부스팅 회귀 트리가 0.863이라는 가장 높은 test set accuracy를 보였습니다. 그래디언트 부스팅 회귀 트리가 adult 데이터셋과 같은 희소행렬에 약하다는 것을 고려하면 주목할 만한 결과입니다. 또한 그래디언트 부스팅 트리는 시행횟수, 즉 부스팅 횟수를 늘릴수록 더욱 더 정확해지는데 현재 설정한 1000 번을 5000 번으로 늘린다면 정확도는 더욱 더 올라갈 것입니다. 반면 LASSO 회귀는 baseline 보다 실망스러운 결과를 보여주었습니다. 그러나 이는 과정상 오류에 의해 일어난 성능 저하로 추측됩니다. 그래서 추후 알고리즘과 처리 과정에 어떠한 문제점이 있는지 살피고, 개선하려고 합니다.

Breast Cancer 데이터셋에서는 baseline 을 제외한 5 개의 머신러닝 모델 모두 0.95 가 넘는 매우 좋은 정확도를 보여주었지만, 그 중 MLP 와 SVM 이 0.98 을 넘는 정확도를 보였습니다. 특히 SVM 과 MLP 는 StandardScaler 를 거친 후에

test set accuracy 가 비약적으로 상승하고 Overfitting 이 해소되었다는 것에 주목할 필요가 있습니다. OneR 알고리즘의 0.92라는 높은 성능 또한 두드러졌습니다.

Baseline 으로 설정된 OneR 과 ZeroR 알고리즘은 두 데이터셋의 경우 모두 최소 0.6 부터 최대 0.9 라는 괜찮은 정확도를 보여주었습니다. 특히 ZeroR 은 정말 간단한 알고리즘으로 어떠한 데이터든 0.5 이상은 나오며, OneR 은 ZeroR 보다 조금 더 복잡한 알고리즘으로 더 정확한 예측을 하기 때문에 baseline 알고리즘으로 쓰기 적합하다는 것을 알 수 있었습니다.

이번 프로젝트는 데이터 과학의 분야에 종사하고자 하는 저에게 데이터 분석 능력을 키우는데 큰 도움이 되었습니다. 직접 scikit-learn 으로 코드를 구현해보고 수많은 시행착오를 겪으며 실력이 많이 늘은 것을 체감할 수 있었습니다. 이런 프로젝트를 내주신 교수님과 조교님께 감사드리고 싶습니다. 이번에 쌓은 실력을 바탕으로 하여, 앞으로도 데이터 분석과 기계학습 분야에 전문가가 될 수 있도록 노력하겠습니다. 감사합니다.