**인공지능 기말 프로젝트**

Table of Contents

1. 실험요약
2. 실험 설계 및 방법
3. 결과 분석
   1. 각각 실험 데이터 분석 및 결론
   2. 비교분석 및 결론
4. 결론
5. 출처
6. 실험요약

Wine data와 mushroom data의 데이터를 먼저 python으로 분석하기 전에 데이터만으로 분석을 해보았다. 그런 후, 각각을 파이썬으로 분석을 한다음 각 세 개의 모델을 가지고 학습을 시켜 cross -validation을 통해 accuracy를 구하였다.

그리고 나온 accuracy와 training accuracy를 비교하여 fit을 구하였고 또한 training한 모델의 값을 가지고 data의 성질을 알아볼 수 있었다.

1. 실험 설계 및 방법
   1. 데이터 분석

2.1.1 python사용하기 전 분석

2.1.2 python으로 분석

* 1. 알고리즘 학습

2.2.1 mushroom data set학습

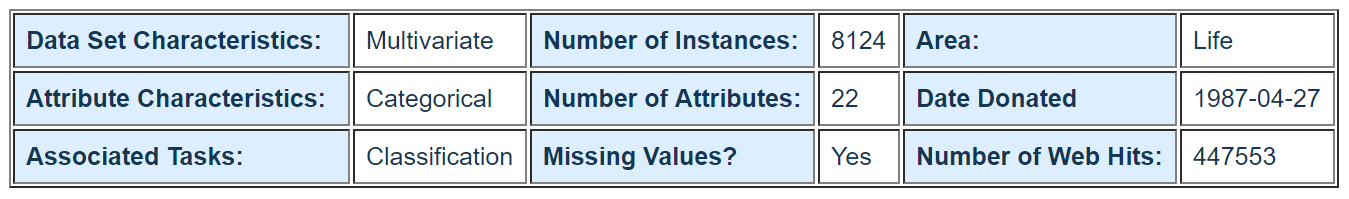
2.2.2 wine data set 학습

* 1. 알고리즘 최종평가(Cross-Validation 사용)
  2. **데이터 분석**

**2.1.1.A python 사용하기 전 Mushroom Data 분석**

먼저 python에 mushroom data를 불러오기 전에 mushroom data에 대해 찾아보았다. Mushroom data set은 UCI Machine Learning Repository에서 제공하는 data set이다. Data set information을 찾아보니 이 data set은 Agaricus와 Lepiota로 분류되는 23개 종류의 버섯에 대한 설명을 담은 set이라고 나와있다. Agraicus는 한국말로 주름버섯속이라는 버섯이고 주름버섯과(Agaricaceae)에 속한다. 1816년 브라질에서 처음 발견되었고 하위분류로는 양송이가 대표적이다. (출처 wikepedia[1]) Lepiota는 한국말로 갓버섯속이라고 부르며 주름버섯속과 같이 주름버섯과에 속한다. (출저 wikepedia[2])

각 종은 edible(식용), poisonous(독성) 혹은 unknown edibility(식용인지 알수 없는 상태) 그리고 not recommended(추천하지 않음)으로 나뉘어 진다. 버섯이 식용인지 알아내는 쉬운 규칙은 없다라고 나온다. 아래는 이 데이터 셋의 구조를 표 형식으로 표현하였다.

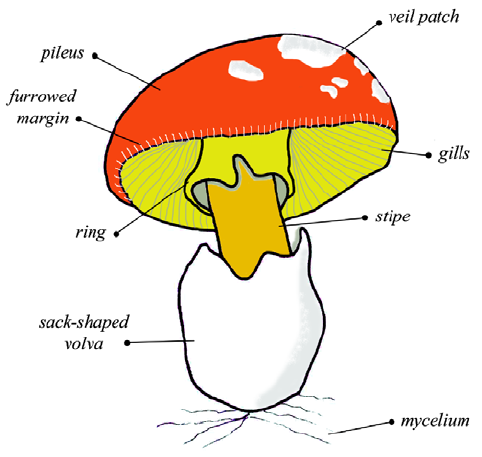


이 data set은 multivariate, 변수가 여러 개이다. 그리고 instance는 총 8124개가 있고 변수 타입은 categorical이다. Attribute는 총 22개가 있으며 Missing value가 존재한다.

아래는 각 attribute의 변수다.

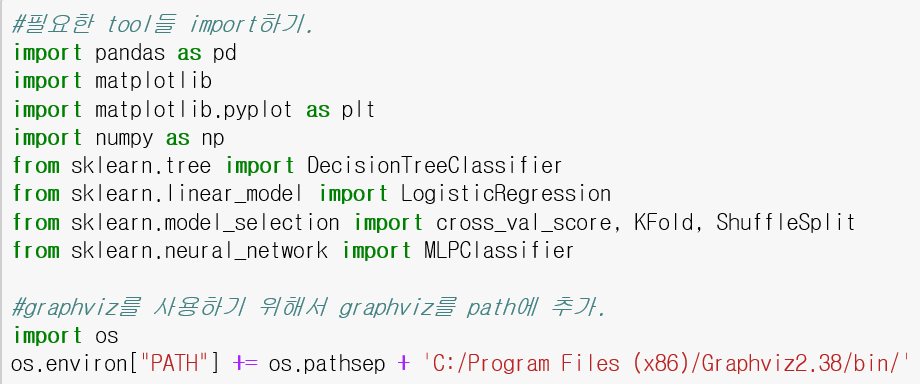
| **변수명** | **변수 설명** |
| --- | --- |
| Class | edible = e, poisonous = p |
| cap-shape(갓 모양) | bell = b, conical = c, convex = x, flat = f, knobbed = k, sunken = s |
| cap-surface(갓 표면) | fibrous = f, grooves = g, scaly = y, smooth = s |
| cap-color(갓 색깔) | brown = n, buff = b, cinnamon = c, gray = g, green = r, pink = p, purple = u, red = e, white = w, yellow = y |
| Bruises(상처) | bruises = t, no = f |
| Odor(냄새) | almond = a, anise = l, creosote = c, fishy = y, foul = f, musty = m, none = n, pungent = p, spicy = s |
| gill-attachment(주름) | attached = a, descending = d, free = f, notched = n |
| gill-spacing(주름 사이 간격) | close = c, crowded = w, distant = d |
| gill-size(주름 크기) | broad = b, narrow = n |
| gill-color(주름 색깔) | black = k, brown = n, buff = b, chocolate = h, gray = g, green = r, orange = o, pink = p, purple = u, red = e, white = w, yellow = y |
| stalk-shape(줄기 모양) | enlarging = e, tapering = t |
| stalk-root(줄기) | bulbous = b, club = c, cup = u, equal = e, rhizomorphs = z, rooted = r, missing = ? |
| stalk-surface-above-ring (ring위의 줄기) | fibrous = f, scaly = y, silky = k, smooth = s |
| stalk-surface-below-ring(ring 아래의 줄기) | fibrous = f, scaly = y, silky = k, smooth = s |
| stalk-color-above-ring(ring위의 줄기 색깔) | brown = n, buff = b, cinnamon = c, gray = g, orange = o, pink = p, red = e, white = w, yellow = y |
| stalk-color-below-ring(ring아래의 줄기 색깔) | brown = n, buff = b, cinnamon = c, gray = g, orange = o,pink = p, red = e, white = w, yellow = y |
| veil-type(veil 종류) | partial = p, universal = u |
| veil-color(veil 색깔) | brown = n, orange = o, white = w, yellow = y |
| ring-number(ring 개수) | none = n, one = o, two = t |
| ring-type(ring 종류) | cobwebby = c, evanescent = e, flaring = f, large = l, none = n, pendant = p, sheathing = s, zone = z |
| spore-print-color(sopre 색깔) | black = k, brown = n, buff = b, chocolate = h, green = r, orange =o, purple = u, white = w, yellow = y |
| Population(인구) | abundant = a, clustered = c, numerous = n, scattered = s, several = v, solitary = y |
| Habitat(서식지) | grasses = g, leaves = l, meadows = m, paths = p, urban = u, waste = w, woods = d |

Attribute를 보면서 ring, gill, stalk등이 정확히 무엇인지 몰라 아래의 그림을 찾아보았고 Ring, veil, gill, stalk은 버섯의 갓, 주름 등인 것을 알았다.

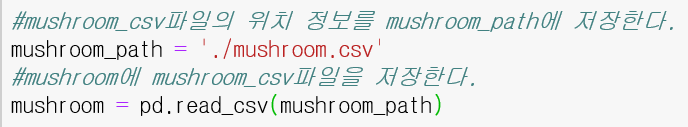
출저 [3]

**2.1.1.B python 사용해서 Mushroom Data 분석**

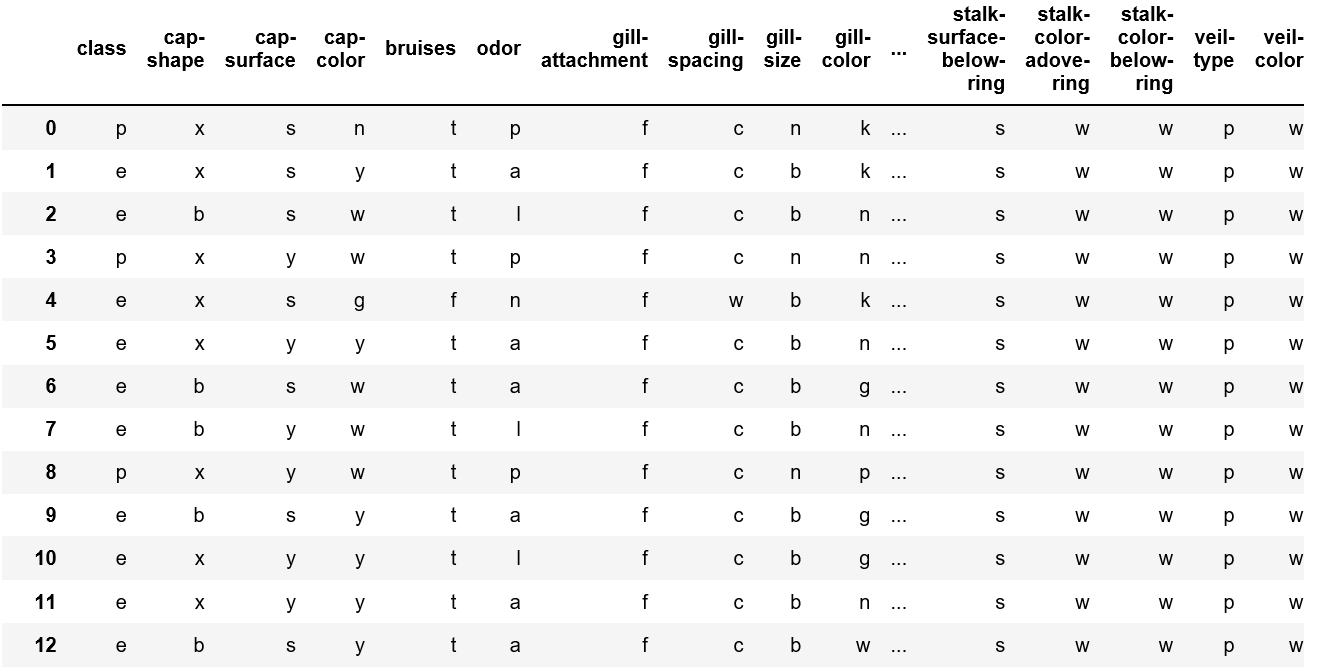
먼저, 필요한 tool을 가져와야 한다. Data를 다루기 위해서 사용되는 pandas와 numpy를 import해준다. 그리고 나중에 data를 그래프로 보기 위해 사용되는 matplotlib, 모델을 사용하기 위해 DecisionTreeClassifier, LogisticRegression, MLPClassifier등을 import한다. 마지막으로 decisionTree를 시각화 하기 위해 사용되는 graphviz를 위해 os를 import하고 graphviz의 위치를 환경변수에 추가한다.



그런 뒤, mushroom.csv파일의 위치를 mushroom\_path에 저장하고 pd.read\_csv로 mushroom.csv를 읽어서 mushroom에 저장한다.



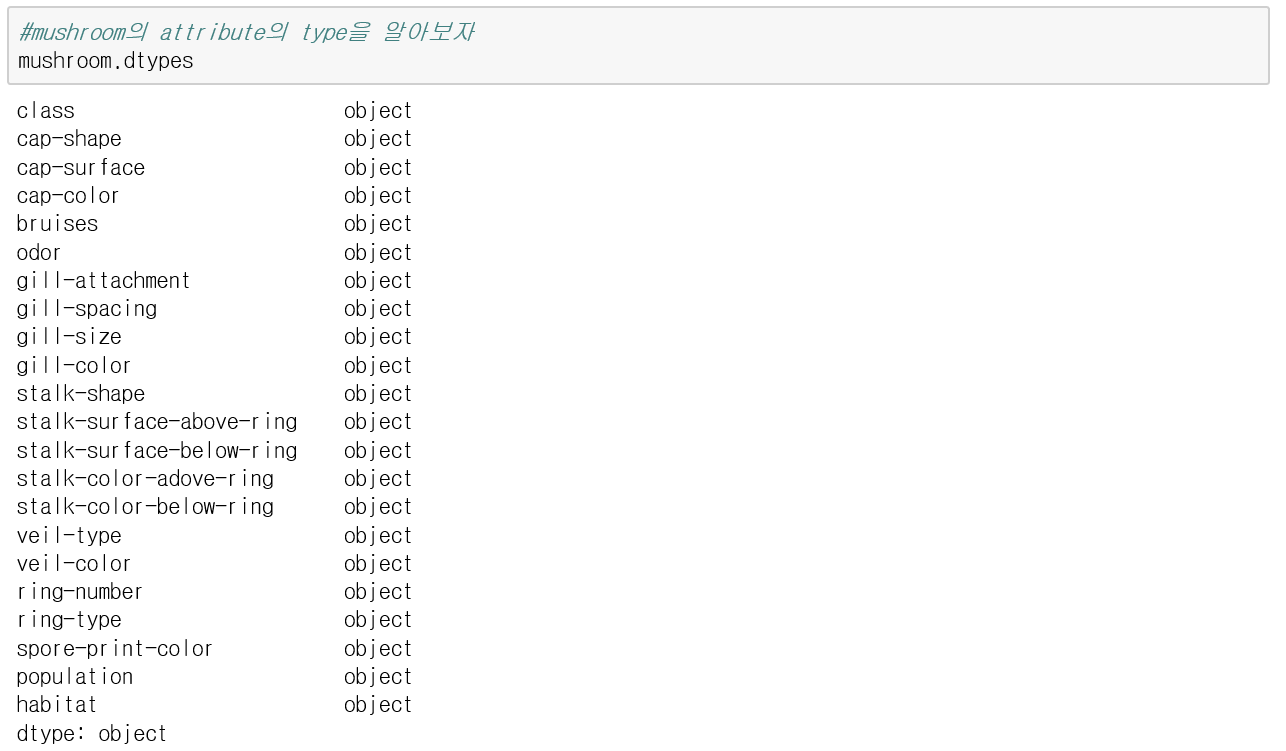
읽어온 Mushroom은 아래와 같다.



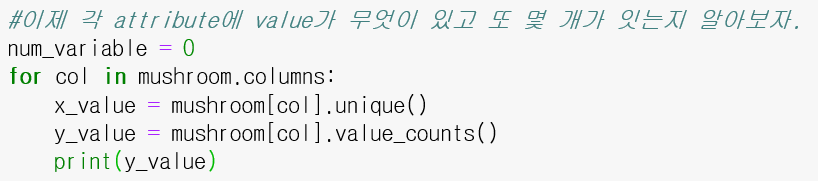
mushroom data에 대해 좀 더 알아보도록 하였다.

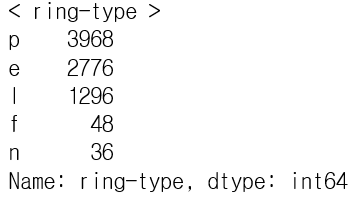
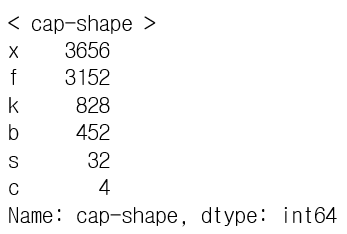
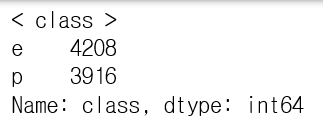
먼저 mushroom.describe()를 통해 mushroom data에 대해 알아보았으나 각 attribute의 unique

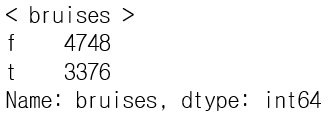
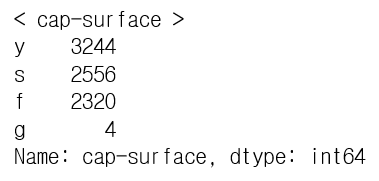
개수말고는 유의미한 정보를 알지 못하였다. Mushroom.dtypes를 사용하여 대신 각 data파일이 object data라는 것을 볼 수 있었다.

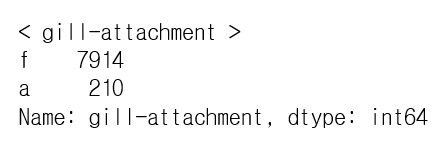
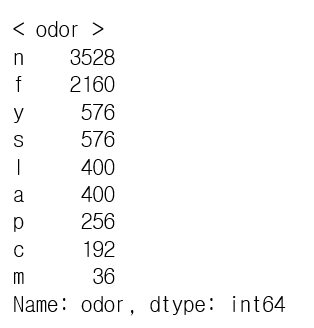
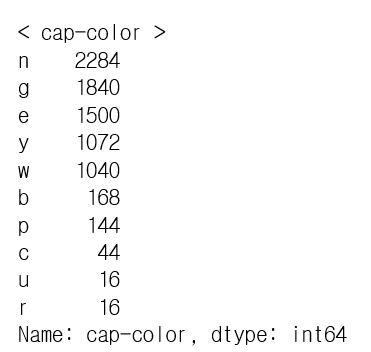


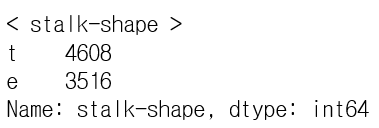
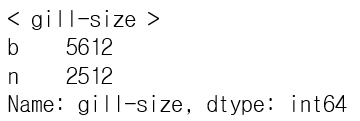
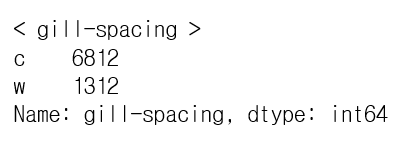
이제 각 attribute에 value가 무엇이 있고 또 몇 개가 있는지 알아보고 그 내용을 bar graph로 표현해보았다. 코드는 아래와 같다.

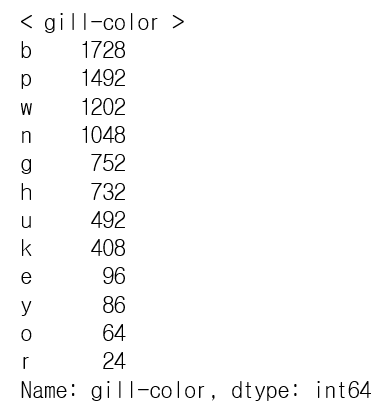
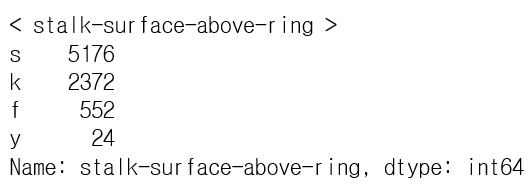
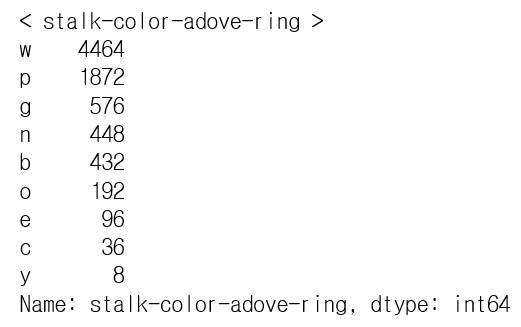
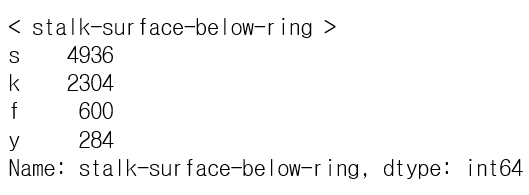


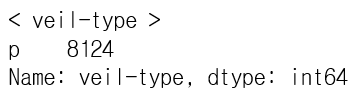
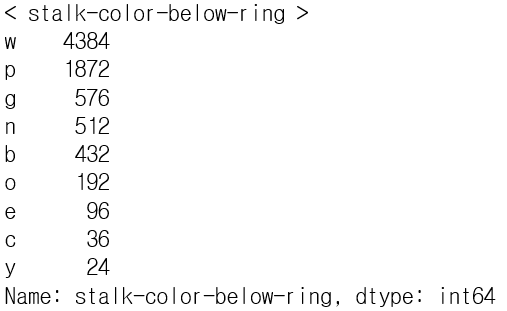


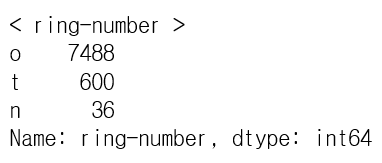
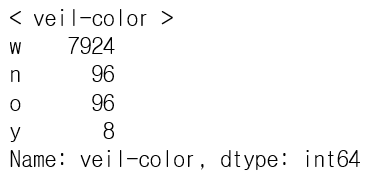


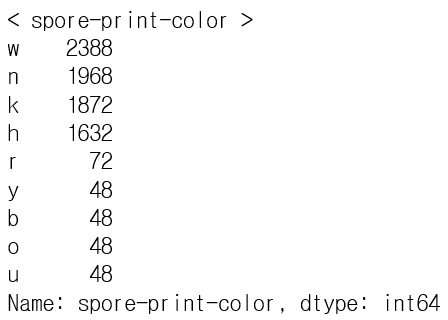
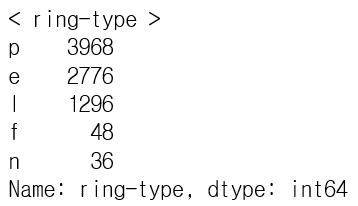


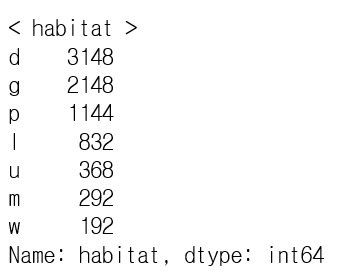
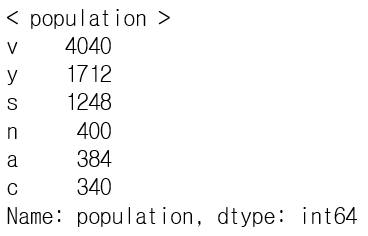










Label로 사용할 class의 경우 edible의 4208개 그리고 poisonous가 3916개가 있다. 또한, 특이한 점 중 하나가 veil-type을 보면 partial(p)와 universal(u)가 있는데 모든 attribute은 partial밖에 없다.

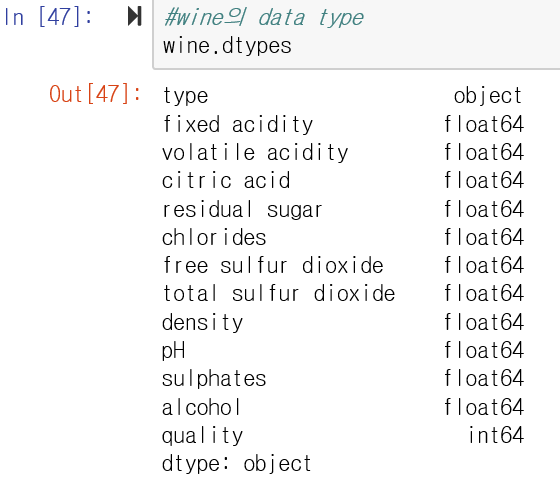
**2.1.1.A python사용하기 전 Wine Quality Data set 분석**

Wine quality data set의 경우 인터넷에서 찾아본 결과 똑 같은 data set을 찾을 수 없었다. UCI Machine Learning Repository에서 같은 이름의 wine quality data set이 있었지만 attirubte이 12개밖에 없었다. 우리의 wine quality data set의 경우 UCI의 data set에 type이 더 추가되었고 instance의 개수도 UCI에는 4898개가 있지만 우리의 데이터에는 6498개가 있다. 그래서 직접 DATA를 분석해보도록 하였다.

| **변수명** | **변수 설명** |
| --- | --- |
| types | 와인 종류 |
| Fixed acidity | 고정산 |
| Volatile acidity | 휘발성산 |
| Citric acid | 시트르산 |
| Residual sugar | 잔당 |
| Chlorides | 염화물 |
| Free sulfulr dioxide | 황 화학물 |
| Total sulfur dioxide | 황 화학물 |
| Density | 밀도 |
| PH | 산성도 |
| Sulphates | 황산염 |
| Alocohol | 도수 |
| Quality | 품질 |

**2.1.1.B python사용해서 Wine Quality Data set 분석**

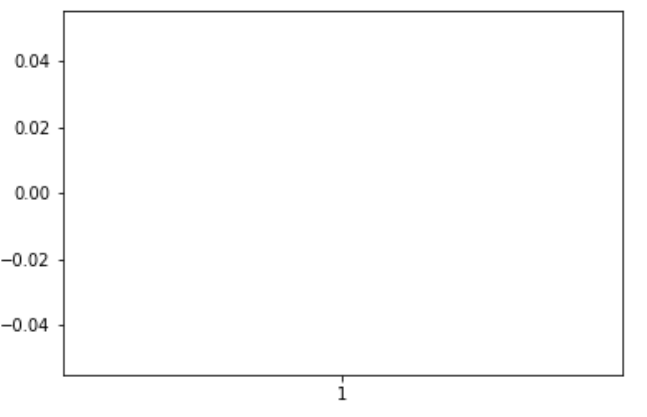
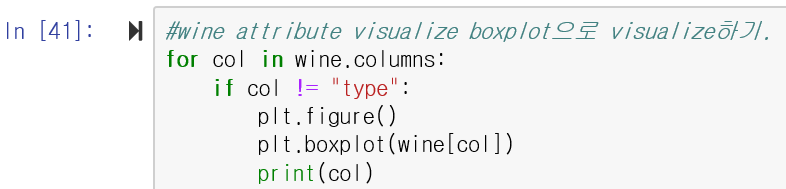
먼저 wine data의 data type을 본 결과 label인 type만 object이고 나머지는 float과 int형 data였다.



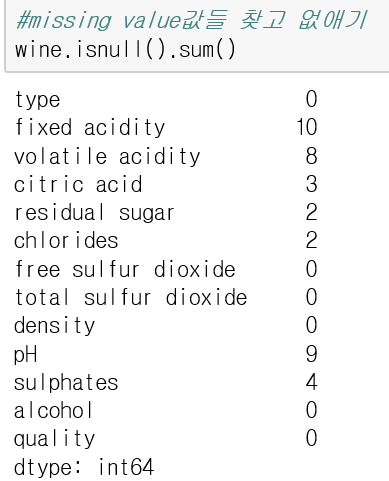
그리고, wine.describe()를 통해 wine의 특징을 살펴보았다. 각 attribute의 mean, std, min, max등을 볼 수 있다.



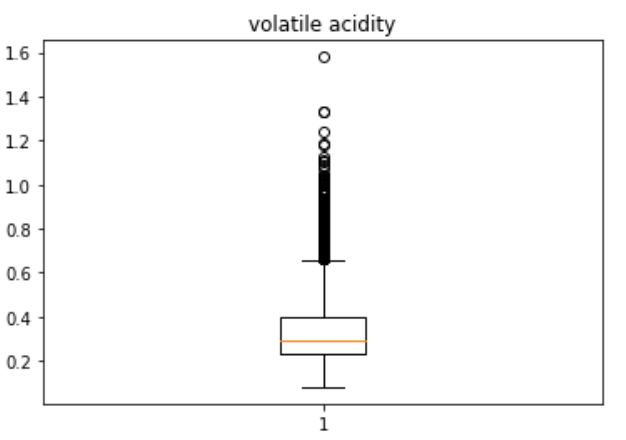
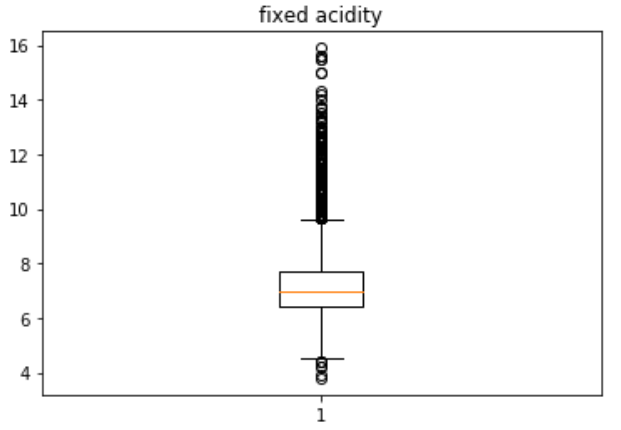
그래서 아래의 코드를 사용해서 boxplot을 만들었으나 몇몇 그래프는 아무 box도 볼 수 없었다.

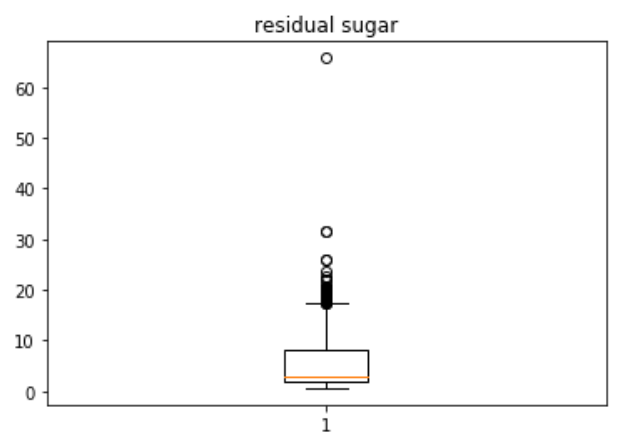
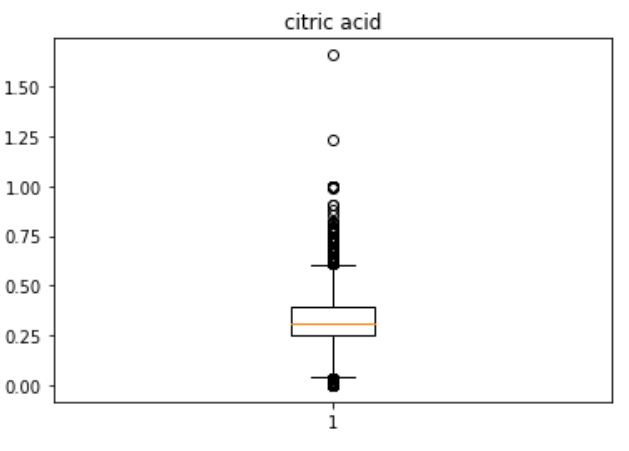


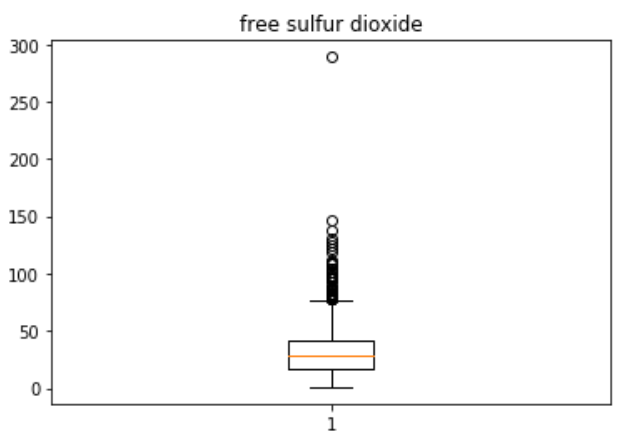
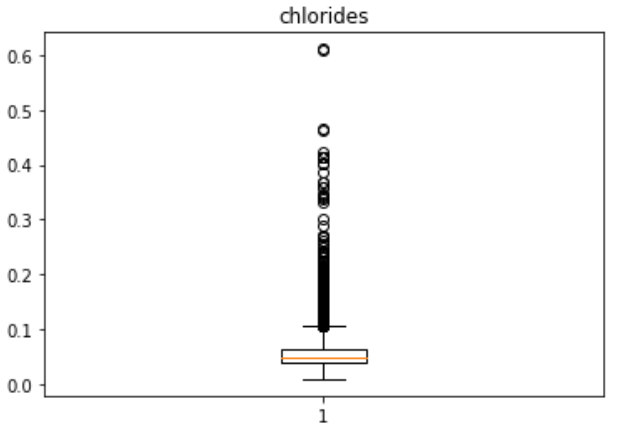
찾아본 결과 data에 missing value가 있어서 문제가 생겼었다. 그래서 missing value값들을 없애고 boxplot을 그렸다. 먼저 각 attribute에 missing value의 개수를 알아보았다.

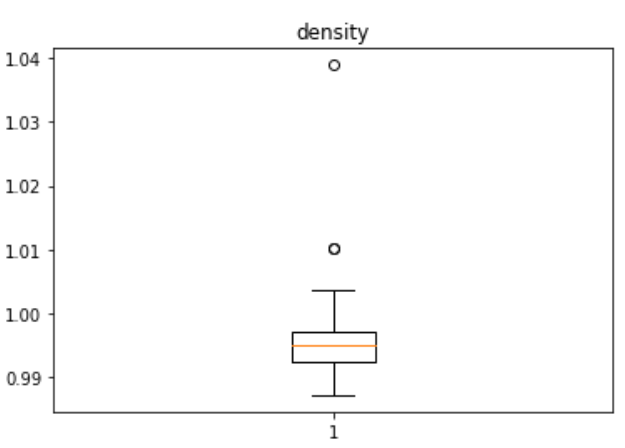
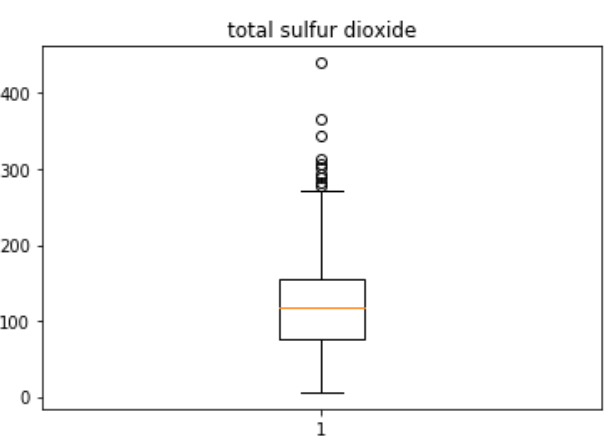


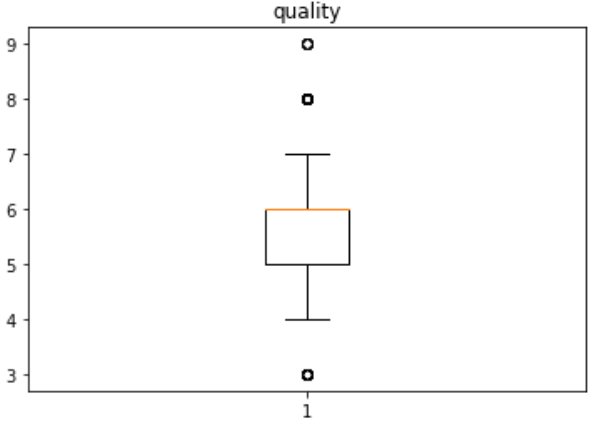
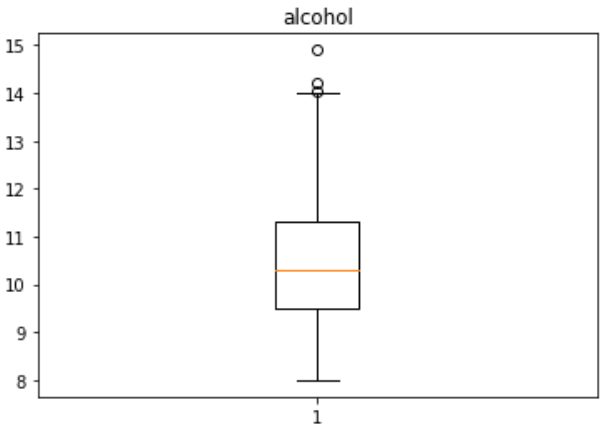
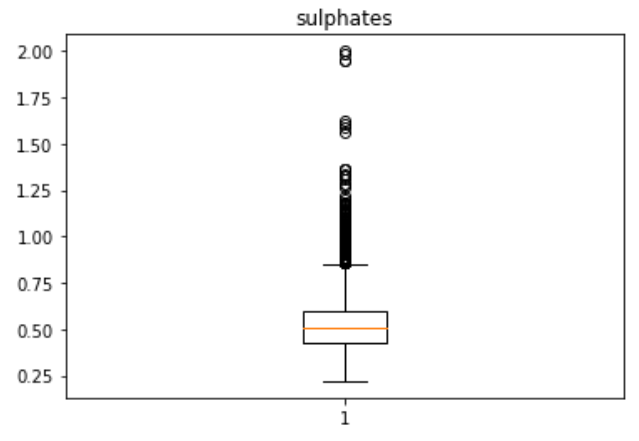
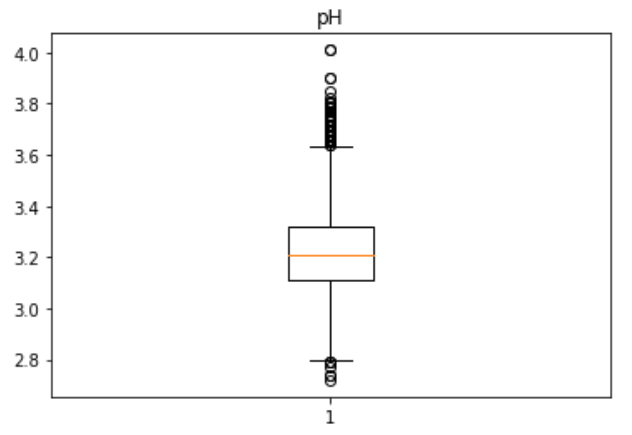
Missing value가 많지 않으므로 missing value를 모두 삭제하기로 하였다. 삭제 한 후, 다시 bar plot을 그려보니 결과가 제대로 나온 것을 확인 할 수 있었다.







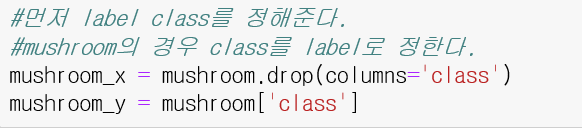




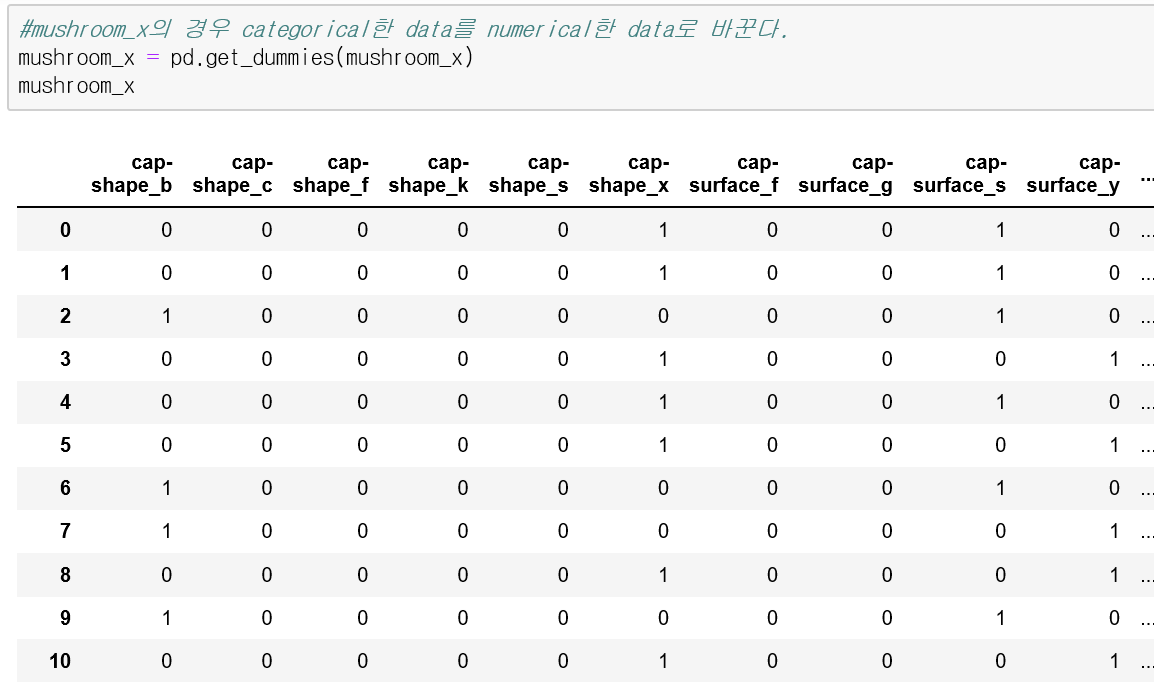
* 1. **알고리즘 학습**

**2.2.1 Mushroom Data 학습**

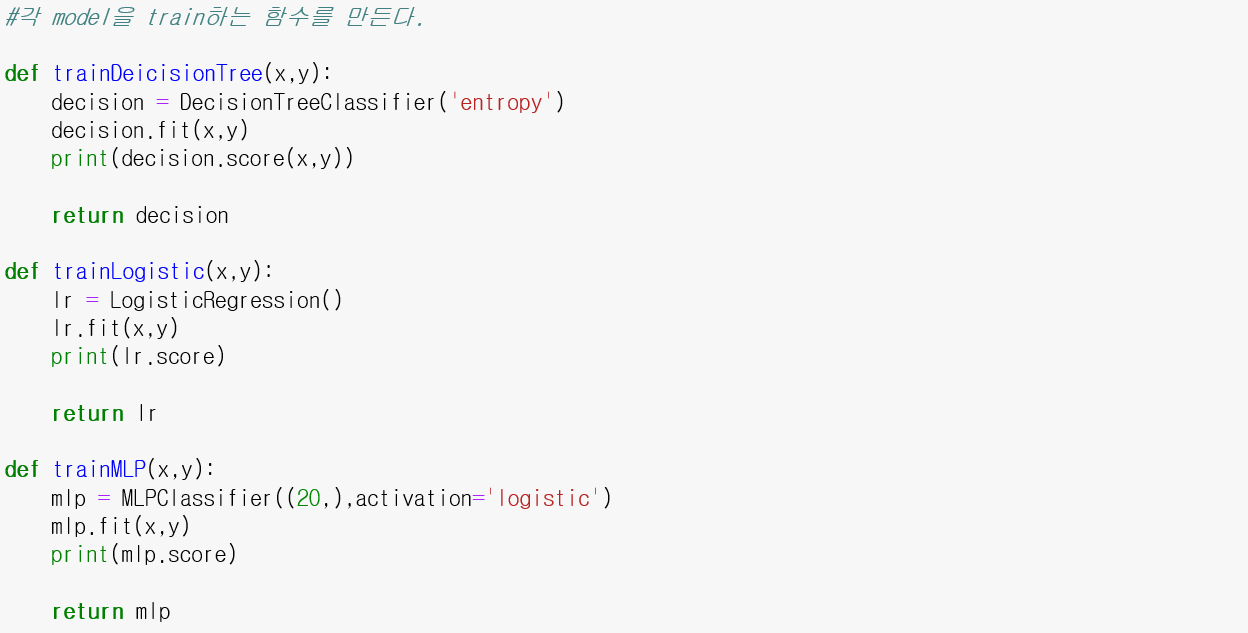
Class를 label로 정하고 mushroom\_y에 저장한다. 나머지 attribute은 mushroom\_x에 저장한다.



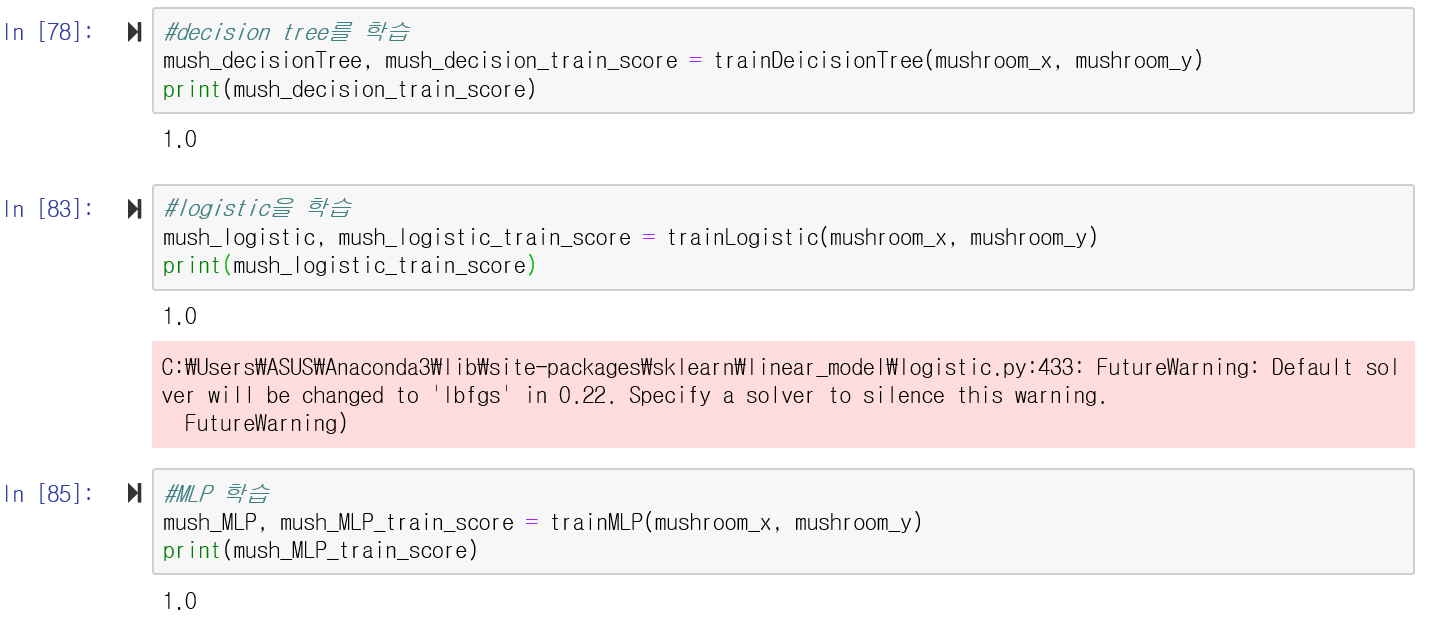
Mushroom\_x의 경우 모델에 사용하기 위해 numerical한 data로 바꿔준다.



그리고 버섯 뿐만이 아니라 와인에서도 똑같이 학습을 진행시켜줘야 하므로 각 model을 학습시키는 함수를 만들었다.

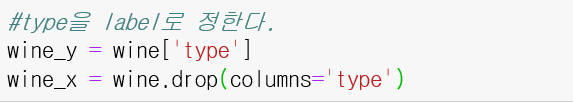


그리고 mushroom\_x와 mushroom\_y를 가지고 각 model을 학습시켰다. 그런 다음, training에 사용되었던 data를 가지고 model에 적용시켜 accuracy를 찾아보았다.

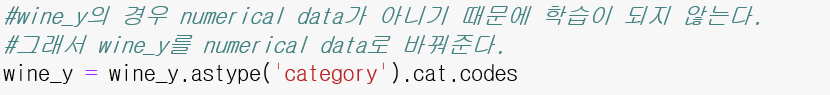


**2.2.2 Wine data 학습**

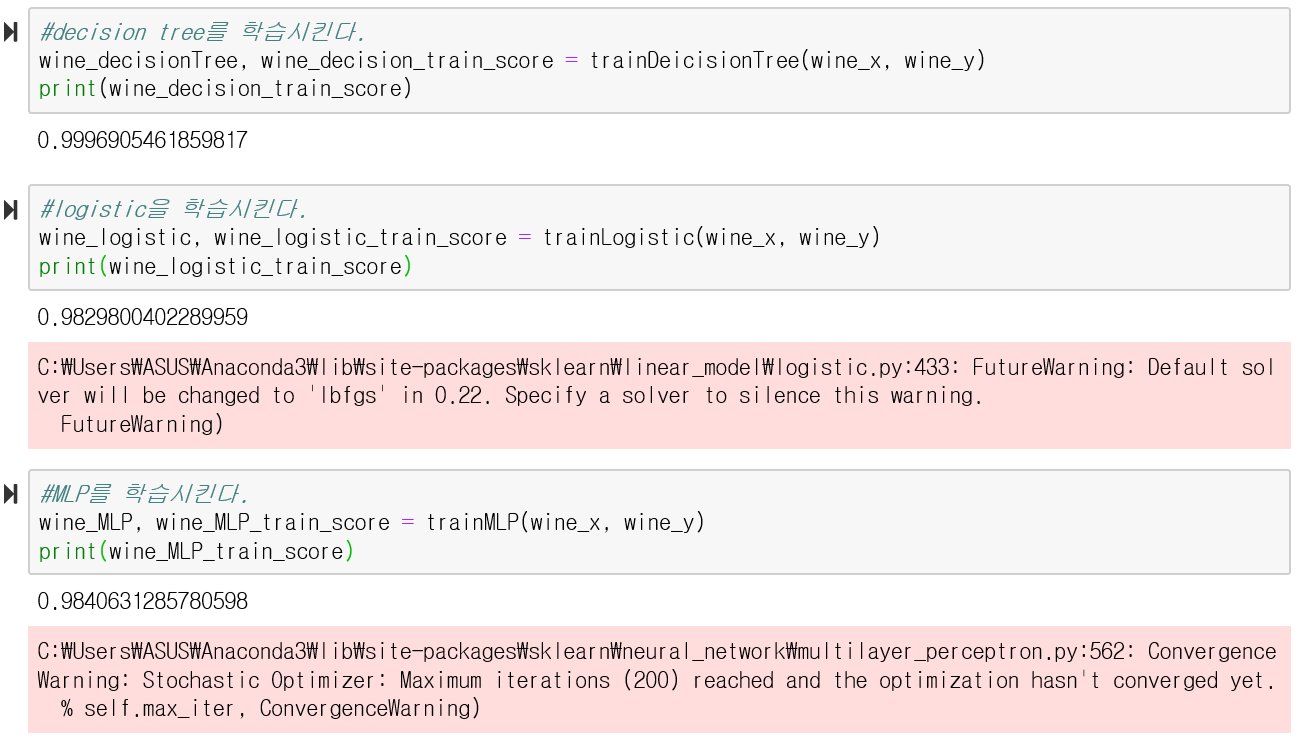
Wine data의 경우 type을 label로 정한다. 그리고 나머지 attribute은 wine\_x로 정한다.



Wine\_y는 categorical한 data이다. 그러므로, numerical한 data로 바꿔주어야 사용이 가능하다. 아래는 wine\_y를 numerical data로 바꿔주는 코드이다.



이제 wine\_x와 wine\_y를 가지고 각 모델에 학습을 시켜보았다.



* 1. **알고리즘 최종평가(Cross-validation사용)**

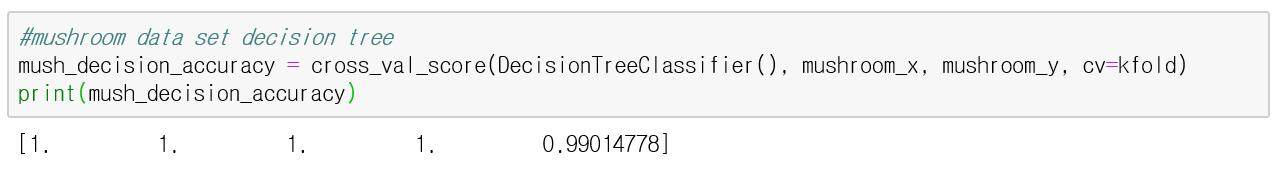
이제 학습시킨 모델들을 cross-validation을 사용하여 accuracy를 평가한다. 먼저, cross-validation을 사용하기 위해 KFold를 선언한다.



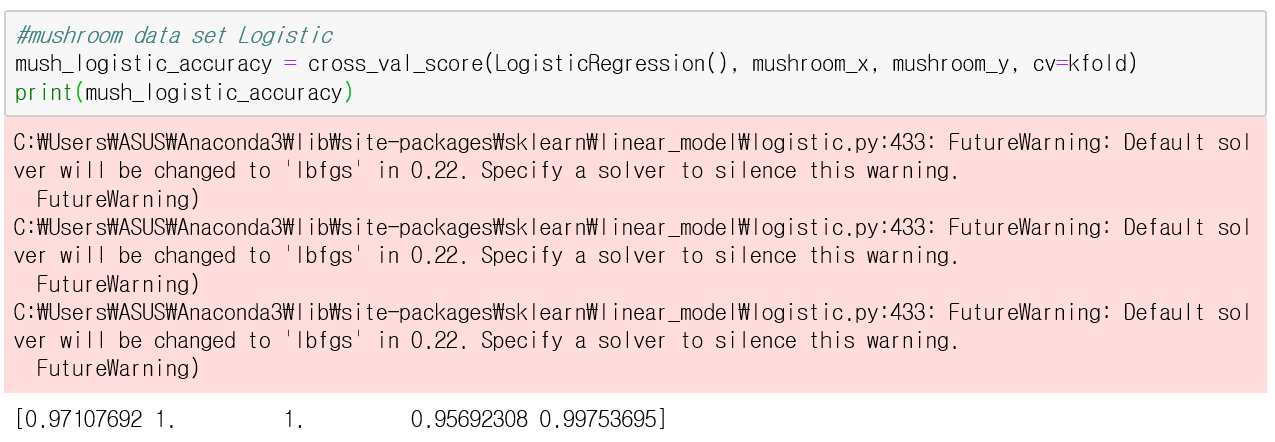
그리고 각 모델을 cross-validdation을 사용하여 평가한다.

**Mushroom data set**

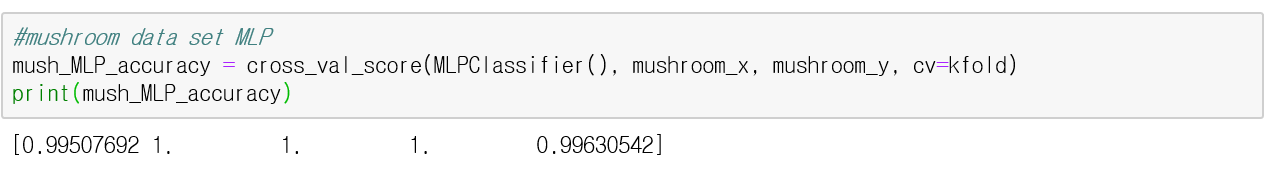
DecisionTree



Logistic

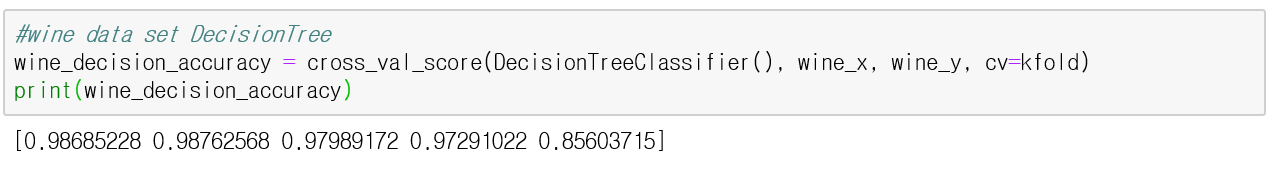


MLP

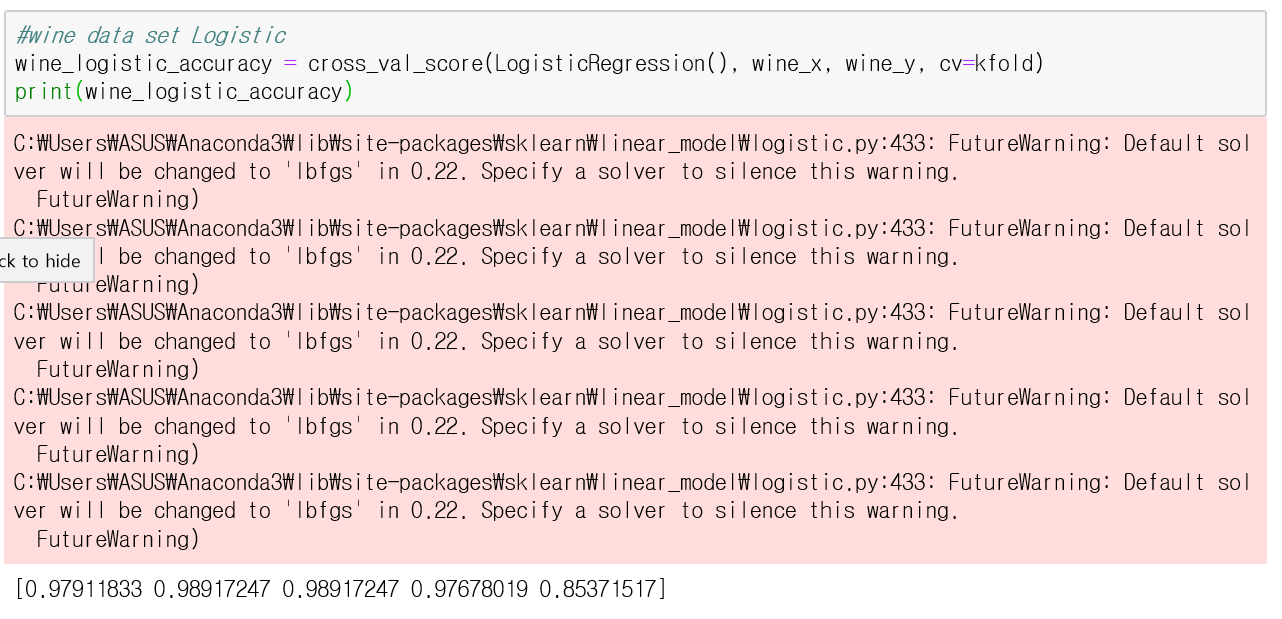


**Wine data set**

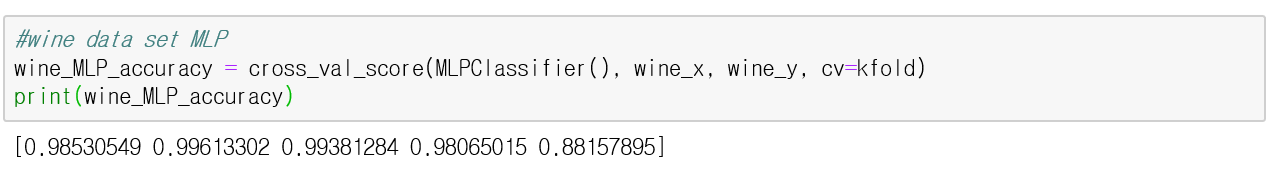
DecisionTree



Logistic



MLP



* 1. 결과분석
     1. 각각 실험 데이터 분석 및 결론
        1. Fit 분석
        2. Dataset 분석
     2. 비교분석

**3.1.1 각 실험 데이터 분석 및 결론**

**3.1.1.A Mushroom Data Fit 분석**

먼저 mushroom 모델의 accuracy의 평균을 구하였다. 그런 다음, test set으로 accuracy를 구한 것과 cross-validiation의 평균의 차이를 구해보았다.



세 모델의 accuracy는 모두 다 0.95 이상이므로 아주 높은 성능을 보여준다. 그리고 세 모델의 차이는 다 0.01이하이므로 새로운 데이터가 주어져도 training data와 성능이 비슷하다는 의미이다. 즉, 이 모델은 underfit하거나 overfit하지 않은 모델이다.

**3.1.1.A wine Data Fit 분석**

Wine dataset도 mushroom dataset과 똑같이 진행하였다. 먼저 wine 모델의 accuracy의 평균을 구하고 test set으로 accuracy를 구한 것과 cross-validiation의 평균의 차이를 구해보았다.

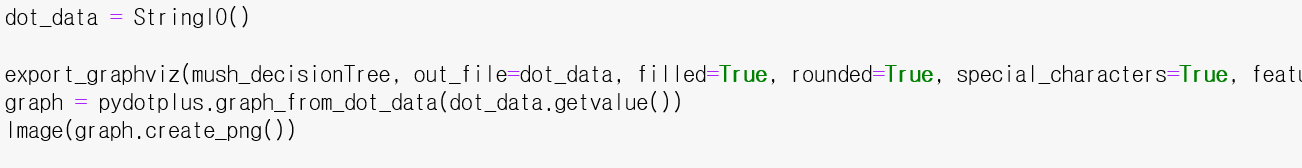


Wine model또한 mushroom model과 똑 같은 결과를 보였다. 먼저, accuracy는 모두 다 0.95 이상이므로 아주 높은 성능을 보였고 차이은 다 0.01이하이므로 새로운 데이터가 주어져도 training data와 성능이 비슷하다는 의미이다. 즉, 이 모델은 underfit하거나 overfit하지 않은 모델이다.

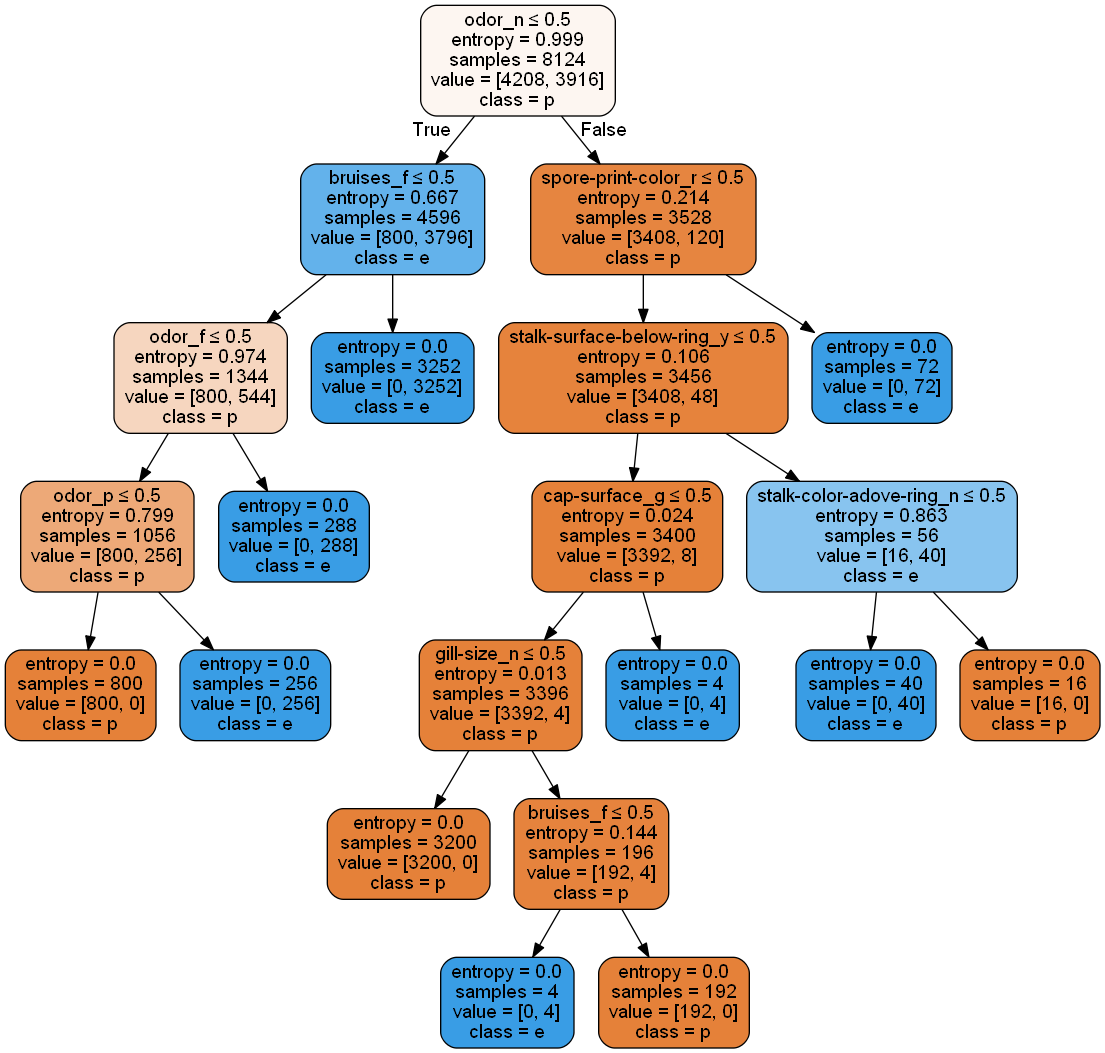
**3.1.1.B Mushroom Dataset 분석**

이제 각 data들을 모델의 결과를 통해 분석해보도록 하겠다. 먼저, mushroom의 경우 Decision\_tree와 Logistic을 통해 모델에 대해 알아 볼 수 있다. Decision tree는 graphviz를 사용하여 visualize한다.

mushroom dataset의 decision tree를 visualize해보자.



아래는 mushroom data set의 decision tree를 visualize한 것이다. Decision tree의 시작 노드는 odor\_n이다. Mushroom data를 학습시킬 때 one\_hot\_encoding으로 변환시키면서 각 categorical data의 value값은 odor\_n처럼 attribute\_value로 변환되고 만약 이 값을 가지고 있으면 1을 가지는 형식으로 바뀌었다.



이 Decision tree을 보면 먼저 odor\_n에서부터 시작한다. 그리고 각 leaf node는 edible인지 poisonous한지 알려주는 노드이다. 위 decision tree를 좀 더 쉽게 분석해보았다.

Edible :

odor\_none->True, bruises\_f -> False

odor\_none -> True, bruises\_f -> True -> odor\_f -> False

odor\_none -> True, bruises\_f -> True, -> odor\_f -> True, odor\_p -> False

위 세 개를 분석해보면, 먼저 냄새가 있고 상처가 있으면 식용이다. 만약 냄새가 있고 상처가 없고 foul한(역한) 냄새가 나면 그 버섯은 식용이 가능성이 높다. 마지막으로 냄새가 있고 상처가 없으며 역한 냄새가 나지 않고 자극적인 냄새가 난다면 그 버섯은 식용이다.

Poisonous :

odor\_none ->True, bruises\_f -> True, odor\_f -> True, odor\_p -> True

만약 버섯이 냄새가 있는데 상처가 없고 그 냄새가 자극적인 냄새거나 역한 냄새가 아니라면 그 버섯은 독버섯일 가능이 높다.

Edible:

odor\_none->False, spore-print-color\_r -> False

odor\_none->False, spore-print-color\_r -> True, stalk-color-above-ring\_n ->True,

odor\_none->False, spore-print-color\_r -> True, cap\_suface-g -> true, gill-size\_n -> false, bruises\_f -> false

odor\_none->False, spore-print-color\_r -> True, cap\_suface-g -> false

버섯이 냄새가 없을 경우, edible한 버섯의 특징은 위와 같다. 먼저, 포자의 색깔이 초록색이면 식용가능하다. 만약 냄새가 없고 초록색도 아니라면 ring 아래의 줄기가 scaly(벗거져 있다)하고 표면에 groove(홈)이 없고 gill\_size가 narrow하고 bruises(손상) 없다면 edible하다. 만약, 냄새가 없고 초록색도 아니고 줄기가 벗겨져 있지도 않으면서 줄기 색깔이 brown이지도 않으면 식용이다.

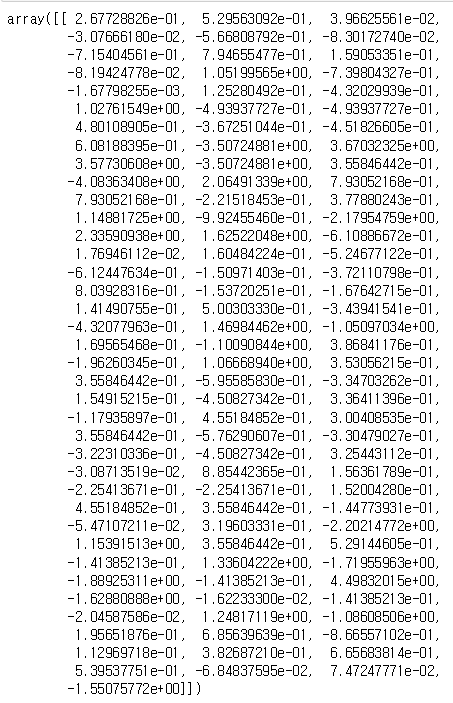
Poisonous :

버섯이 냄새가 없을 경우는 edible의 반대 상황이다.

즉 버섯의 경우, 냄새가 있을 때와 없을 때 식용과 독의 조건이 다르다. 냄새가 있을 경우 자극적인 냄새가 있을 때 식용이 가능하다. 이는, 아무래도 식용가능 한 버섯의 경우 자기 방어를 위해 냄새를 자극적으로 진화한 것으로 추정된다. 냄새가 없을 경우, 버섯은 줄기 색깔이 화려하거나 상처가 없을 경우 독버섯일 확률이 높다. 이는, 독버섯의 경우 색을 화려하게 하여 독버섯임을 알리는 역할을 하고 독버섯이 상처가 없는 이유는 동물들이 독버섯을 아는 경우 손을 대지 않기 때문이라고 추측이 된다.

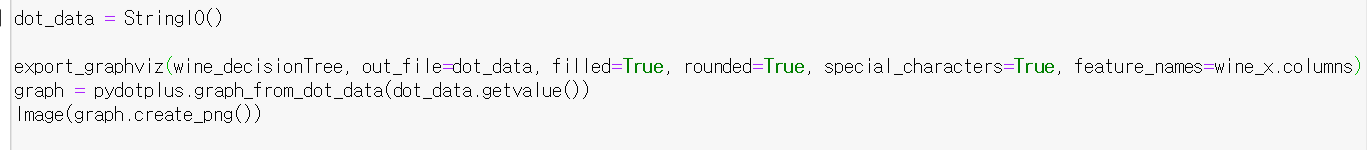
Logistic의 경우 coef\_를 통해 알아볼 수 있다. 하지만, 데이터 셋이 categorical이기 때문에 one\_hot\_encoding을 하면 변수가 22개가 아니라 112가 되어버려 평가가 어렵다.



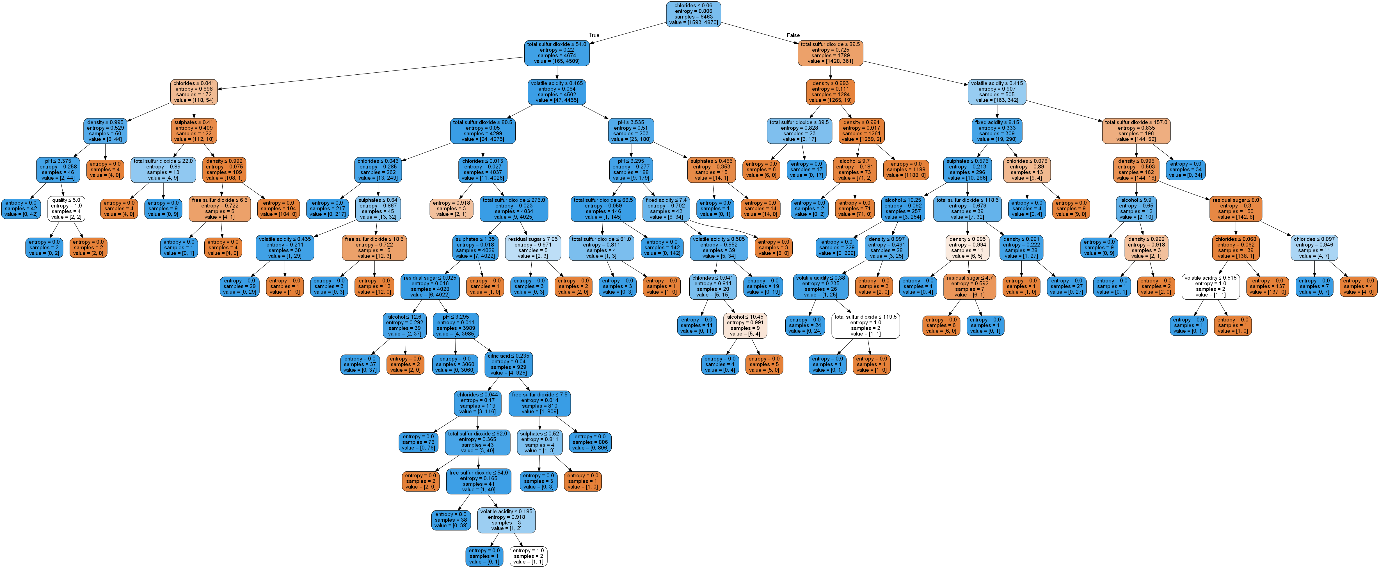


**3.1.1.B wine Dataset 분석**

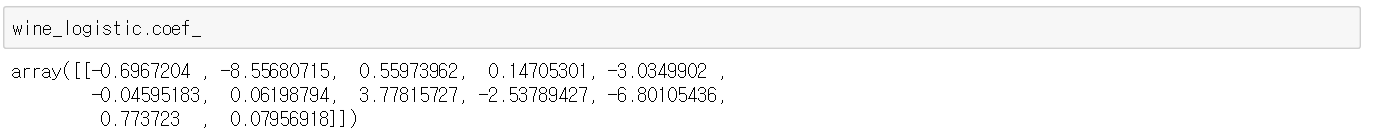
Wine data set은 아래와 같이 visualize한다.



하지만, wine data set은 numerical data이기 때문에 decision tree로 visualize하면 판단하기가 어려워진다.



그래서 대신, coef\_로 분석을 해보았다.



Coef\_로 분석을 해본 결과 fixed\_acidity와 chlorides sulphate가 높을수록 red에 더 가깝고 density가 더 높을수록 white 와인에 더 가까운 것을 볼 수 있다.

**3.1.2 비교분석**

Mushroom data와 wine data는 둘 다 overfit하거나 underfit하지 않고 제대로 학습이 되었다. 또한 두 모델 다 아주 높은 accuracy를 가지고 있다. 하지만, mushroom data의 경우 categorical한 데이터이므로 decision tree로 분석하는 것이 훨씬 더 효과적이였다. 반대로, wine data의 경우 numerical data이기 때문에 decision tree보다는 coef로 분석하는 것이 훨씬 더 효과적이였다.

결론

각 wine data와 mushroom data를 비교분석 해보았다. Mushroom data의 경우 categorical한 data이므로 one\_hot encoding을 해주어야 했고 mushroom data의 경우 label인 type이 categorical한 data여서 numerical data로 변환해주어야 했다. 또한, data type의 차이가 효과적인 모델의 차이로 이어진다는 것을 알게 되었다. 즉, categorical data의 경우 분석을 할 때 decision tree가 훨씬 효과적이였고 반대로 numerical data의 경우logistic이 훨씬 더 효과적인 것을 알 수 있었다.

****출처****

[1] <https://ko.wikipedia.org/wiki/%EC%A3%BC%EB%A6%84%EB%B2%84%EC%84%AF%EC%86%8D>

[2] <https://en.wikipedia.org/wiki/Lepiota>

[3] <https://www.researchgate.net/figure/Basidioma-of-a-gilled-mushroom-Amanita-caesarea-note-the-veil-patches-remnants-of_fig1_281625151>

[4] graphviz문제

<https://blog.naver.com/PostView.nhn?blogId=resumet&logNo=221450970851>

[5] visualize decisionTree

<https://medium.com/@rnbrown/creating-and-visualizing-decision-trees-with-python-f8e8fa394176>