2018.10.20

의사결정나무

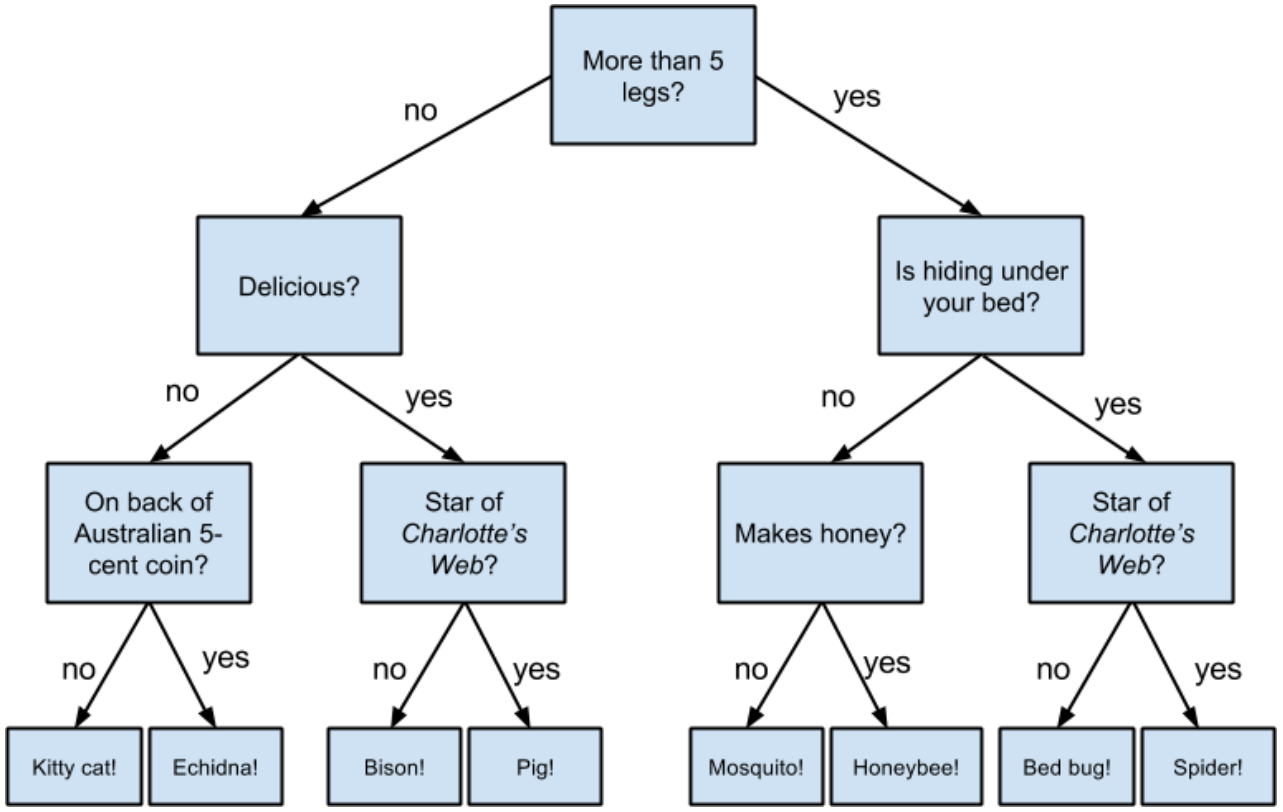
3기 양서윤

Growth Hackers

CONTENTS

Growth Hackers

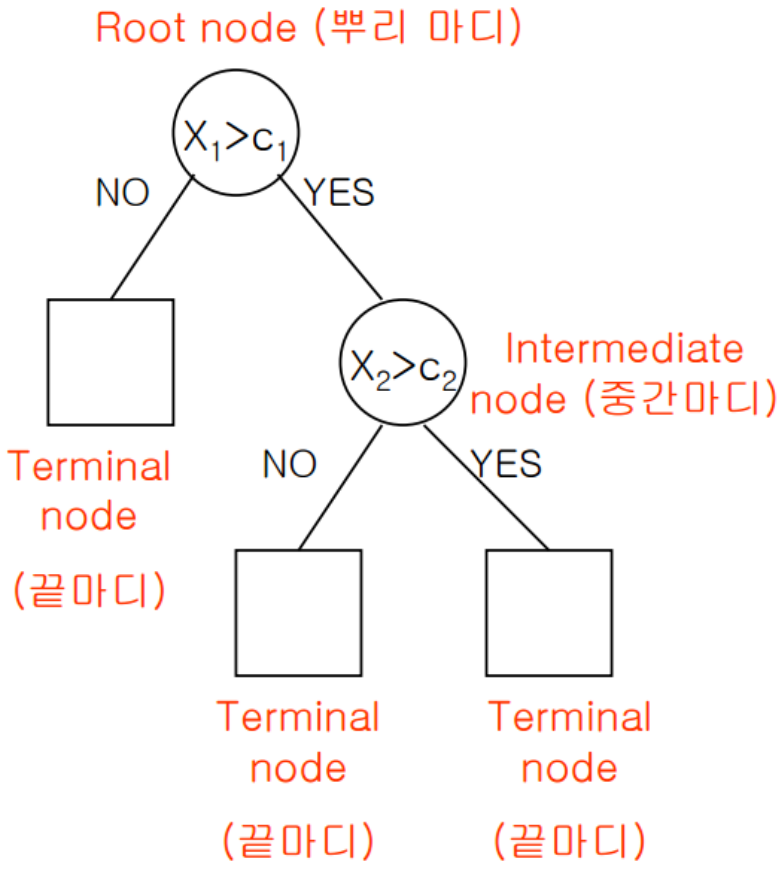
Decision Tree



- 우리 모두 알고 있어요!

Growth Hackers

Decision Tree



노드 : Root node, Intermediate node, Terminal node

Tree size : 노드의 수

Tree depth : root node에서 terminal node까지

가장 긴 경로의 길이, 트리의 깊이

엣지 : 질문의 답과 다음 질문을 연결하는 선, 가지

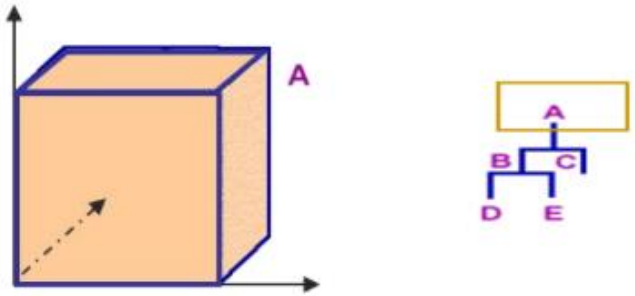
attribute : 분류 속성, 기준

Growth Hackers

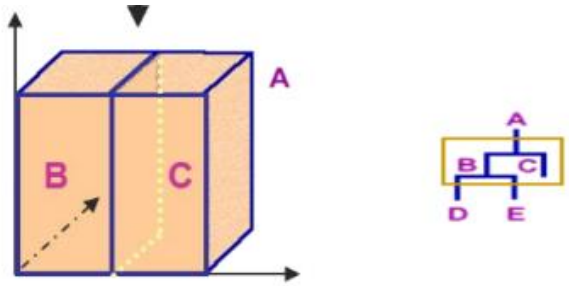
Decision Tree

설명변수가 3개인 데이터에 의사결정나무를 적용

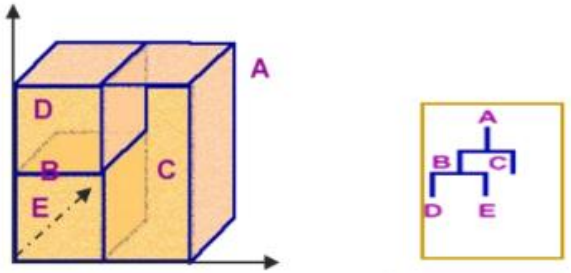
- 아무런 분기가 일어나지 않은 상태의 root node는 A



- A가 B와 C로 분할



- B가 D와 E로 분할
terminal node는 C,D,E 세 개



Growth Hackers

Decision Tree

- 집합 X에 두 종류의 원소가 있다.
  이 집합의 불순도는 얼마인가?

Ex) 흰 바둑알 W, 검정 바둑알 B

X = {B, B, B, B, W, W, W, W}

Y = {B, B, W, W, W, W, W, W}

Z = {B, B, B, B, B, B, B, B, B}

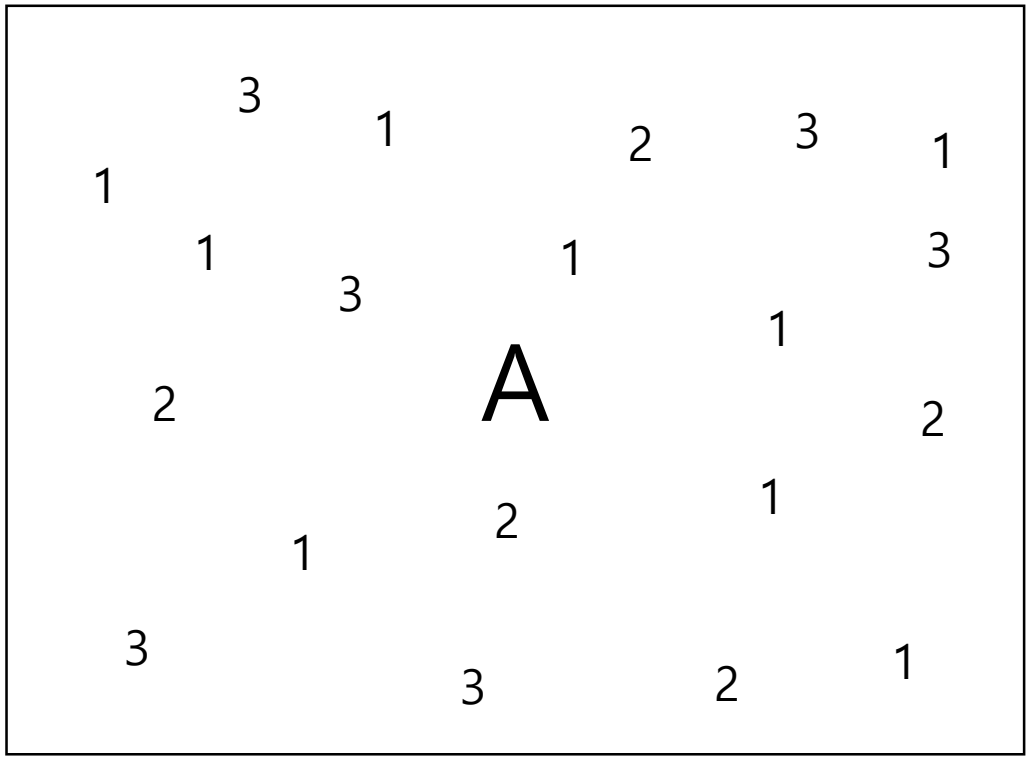
Z′ = {B, W}

⇒ 불순도 측정하는 방법?

엔트로피

지니지수

Decision Tree



1 : 9개
2 : 5개
3 : 6개
총 20개

$$p_k = \frac{n_k}{20}$$

$n_k = class\ k$에 속하는 원소의 개수

Growth Hackers

Decision Tree

지니 지수(weighted sum)

- Sckit-learn 의사결정나무 default option
- m개의 레코드를 지닌 end note A에 대한 지니 지수

$$I(A) = 1 - \sum_{k=1}^{m} p_k^2$$

- Q: $I(A)$의 범위?

$$0 \leq I(A) \leq 1 - {}^{k}/_{k^2} = \frac{k-1}{k}$$

- Q: 범주가 두 개일 때, 불순도를 최대로 만드는 비율은??

엔트로피(entropy)

- m개의 레코드를 지닌 end note A에 대한 엔트로피

$$entropy(A) = -\sum_{k=1}^{m} p_k \log_2 p_k$$

- Q: entropy(A)의 범위?

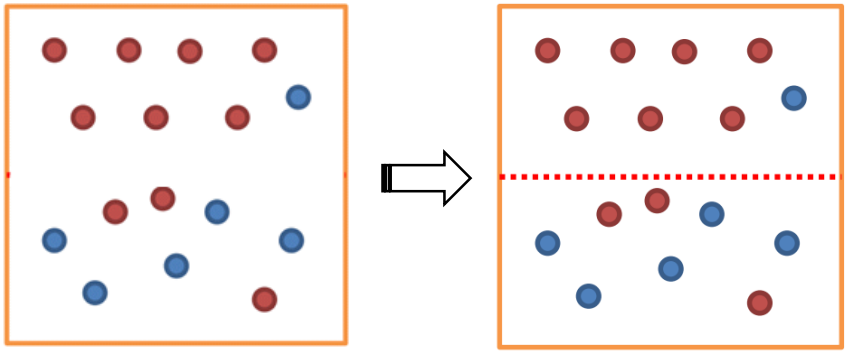
$$0 \leq entropy(A) \leq \log_2 m$$

- Q: 엔트로피가 언제 0이 될까요??

Decision Tree

정보 이득(information gain)

- 분할 후 불순도 측정



$$entropy(A) = -\sum_{k=1}^{m} p_k \log_2 p_k$$

Q: 분할 결과의 불순도를 엔트로피로 계산해볼까요?

A: $0.5 \times \left(-\frac{1}{8}\log_2\frac{1}{8} - \frac{7}{8}\log_2\frac{7}{8}\right) + 0.5 \times \left(-\frac{5}{8}\log_2\frac{5}{8} - \frac{3}{8}\log_2\frac{3}{8}\right)$

- **Information Gain** = 기존 불순도 – 분할 후 불순도

Decision Tree

알고리즘을 따라가봅시다!

Outlook, Temp, Humidity, Windy 등의
날씨 조건을 보고, 언제 테니스를 치는지에
대해 분류하는 예제

Play의 엔트로피 계산

$$E(Play) = \left(-\frac{5}{14}\log_2\frac{5}{14}\right)+\left(-\frac{9}{14}\log_2\frac{9}{14}\right)$$

$$E(Play) = 0.94$$

Outlook	Temperature	Humidity	Windy	Play?
sunny	hot	high	FALSE	No
sunny	hot	high	TRUE	No
overcast	hot	high	FALSE	Yes
rain	mild	high	FALSE	Yes
rain	cool	normal	FALSE	Yes
rain	cool	normal	TRUE	No
overcast	cool	normal	TRUE	Yes
sunny	mild	high	FALSE	No
sunny	cool	normal	FALSE	Yes
rain	mild	normal	FALSE	Yes
sunny	mild	normal	TRUE	Yes
overcast	mild	high	TRUE	Yes
overcast	hot	normal	FALSE	Yes
rain	mild	high	TRUE	No

Decision Tree

각 클래스로 분류 시 엔트로피는?

E(Play, Outlook)

		Play		
		Yes	No	Total
Outlook	sunny	3	2	5
	overcast	4	0	4
	rain	3	2	5

$$E(Play,\ Outlook) = p(sunny) \times E(3,2) + p(rainy) \times E(3,2) + p(overcast) \times E(4,0)$$
$$E(Play,\ Outlook) = \frac{5}{14} \times \left[\left(-\frac{3}{5} \times \log_2 \frac{3}{5}\right) + \left(-\frac{2}{5} \times \log_2 \frac{2}{5}\right) \right] + \cdots\cdots$$
$$E(Play,\ Outlook) = 0.36 \times 0.971 + 0.36 \times 0.971 + 0$$
$$E(Play,\ Outlook) = 0.6935$$

E(Play,Temperature)

		Play		
		Yes	No	Total
Temp..	hot	2	2	4
	mild	4	2	6
	cool	3	1	4

$$E(Play,\ Temp) = p(hot) \times E(2,2) + p(mild) \times E(4,2) + p(cool) \times E(3,1)$$
$$E(Play,\ Temp) = \frac{4}{14} \times \left[\left(-\frac{1}{2} \times \log_2 \frac{1}{2}\right) + \left(-\frac{1}{2} \times \log_2 \frac{1}{2}\right) \right] + \cdots\cdots$$
$$E(Play,\ Temp) = 0.2857 + 0.3935 + 0.2317$$
$$E(Play,\ Temp) = 0.911$$

E(Play, Humidity)

		Play		
		Yes	No	Total
Humidity	high	3	4	7
	normal	6	1	7

$$E(Play,\ Humidity) = p(high) \times E(3,4) + p(normal) \times E(6,1)$$
$$E(Play,\ Humidity) = 0.7884$$

E(Play, Windy)

		Play		
		Yes	No	Total
Windy	TRUE	3	3	6
	FALSE	6	2	8

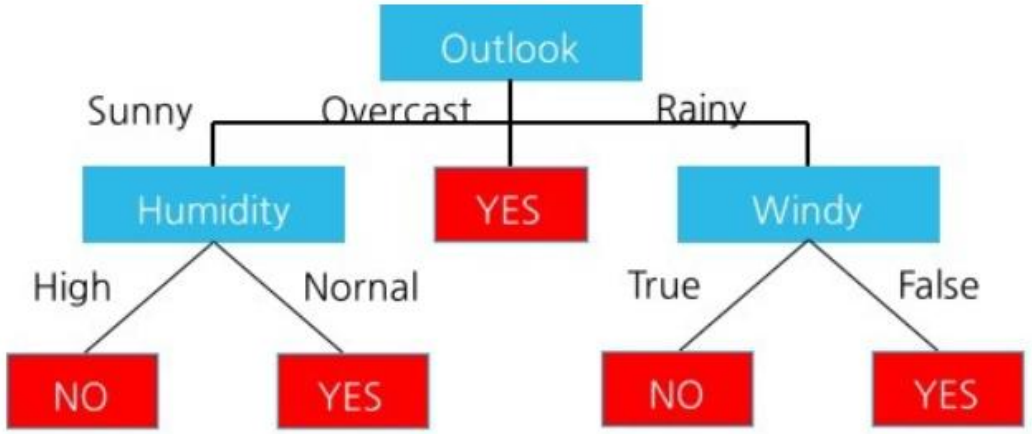
$$E(Play,\ Windy) = p(True) \times E(3,3) + p(False) \times E(6,2)$$
$$E(Play,\ Windy) = 0.8921$$

Growth Hackers

Decision Tree

분류 시 INFORMATION GAIN이 가장 큰 클래스는?

$$E(Play) - E(Play, Outlook) = 0.25$$
$$E(Play) - E(Play, Temp) = 0.02$$
$$E(Play) - E(Play, Humidity) = 0.1514$$
$$E(Play) - E(Play, Windy) = 0.047$$



Growth Hackers

Decision Tree

- **의사결정나무의 장점**
  1. 규칙을 이해하기 쉽다
  2. 연속형, 범주형 변수 모두 쉽게 처리된다
  3. 중요한 변수를 찾아준다
  4. 입력변수의 이상치에 강건(robust)하다

- **의사결정나무의 단점**
  1. 회귀모형에서 예측력이 떨어진다
  2. 나무의 크기가 크면 해석력이 떨어진다
  3. 불안정하다

의사결정트리들에
배깅, 랜덤포레스트, 부스팅
기법을 사용하면
예측 성능 향상!

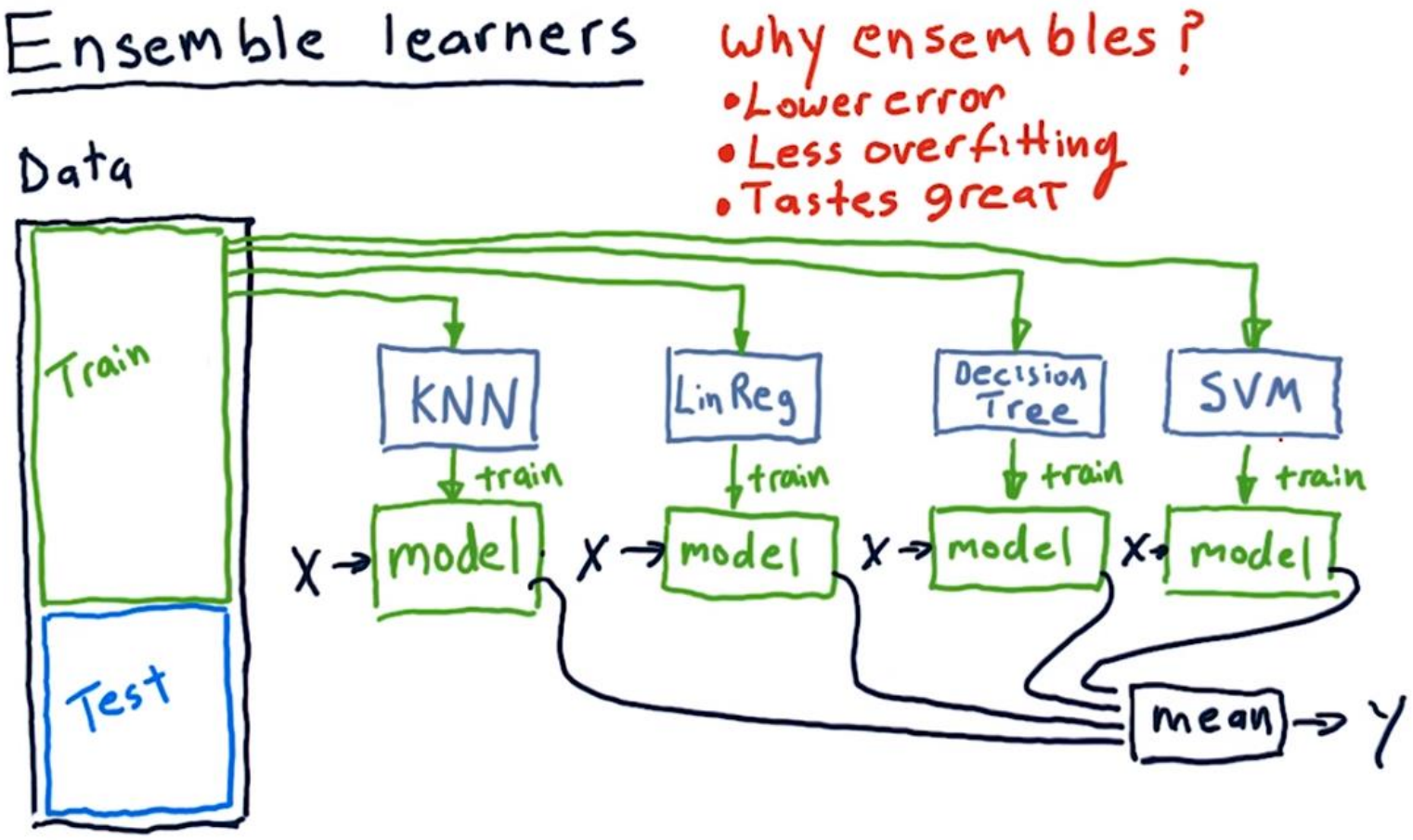
Ensemble

- **앙상블**이란 하나의 자료에 대해서 여러 개의 예측모형을 만든 후, 이를 결합하여 최종 예측모형을 만드는 방법을 말한다

- 앙상블 방법의 예
  : **Bagging, Boosting, RandomForest**

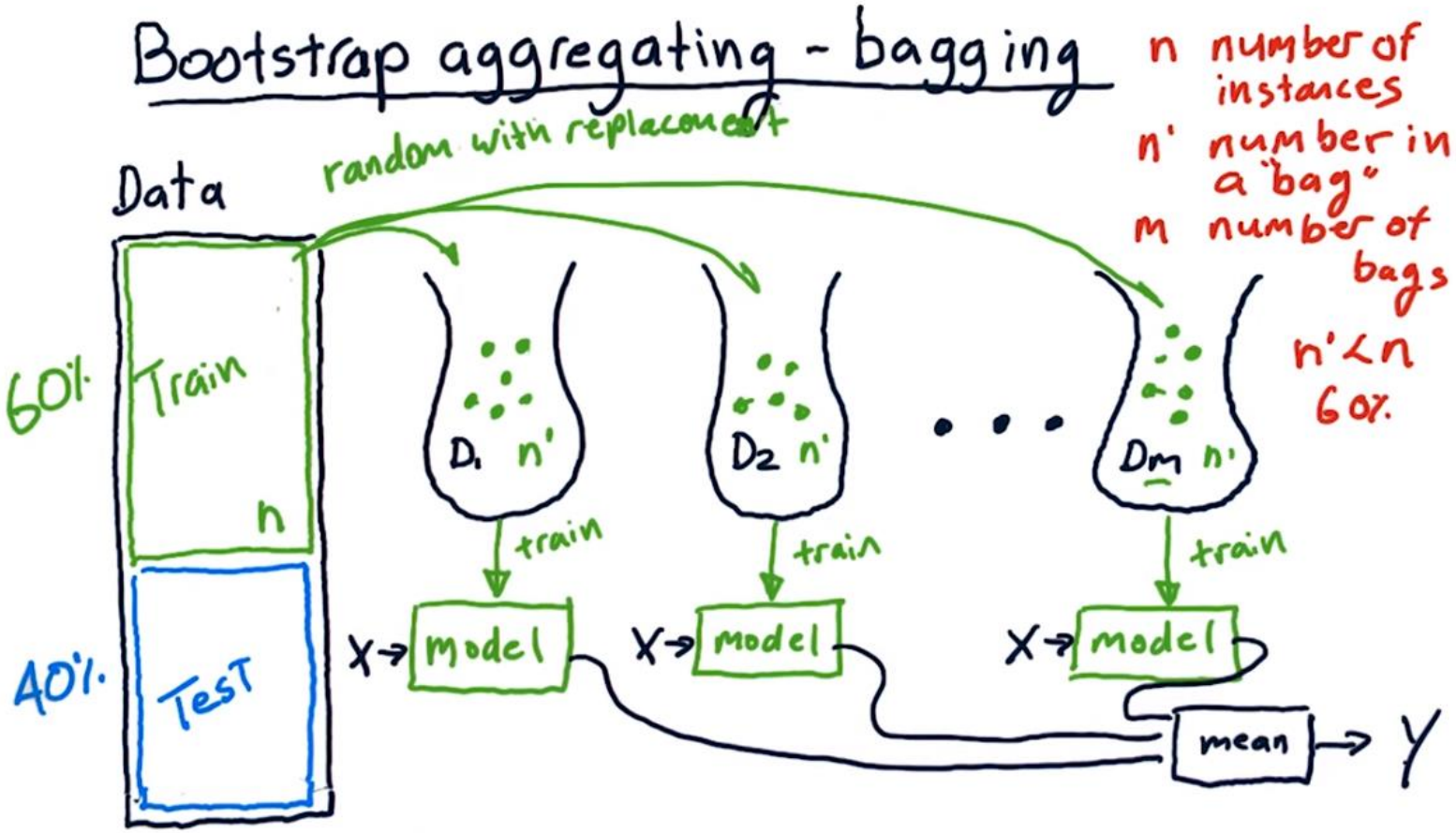
- 실증적으로 앙상블 방법이 의사결정나무보다 훨씬 좋은 예측력을 갖는 것이 밝혀졌다

Ensemble



- **Error 최소화**
  단일 모형으로 분석했을 때보다 신뢰성 높은 예측값을 얻음
- **Variance 감소**
  여러 모형의 의견을 결합하여 평균 내면 variance가 작아짐
- **Overfitting 감소**
  과적합의 가능성을 줄여줌

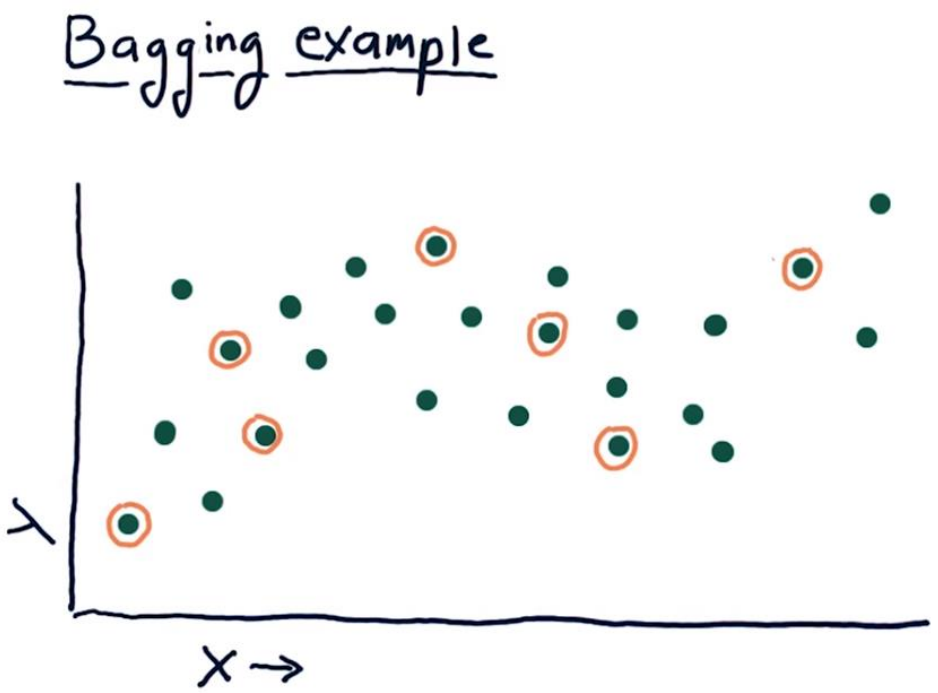
Bagging

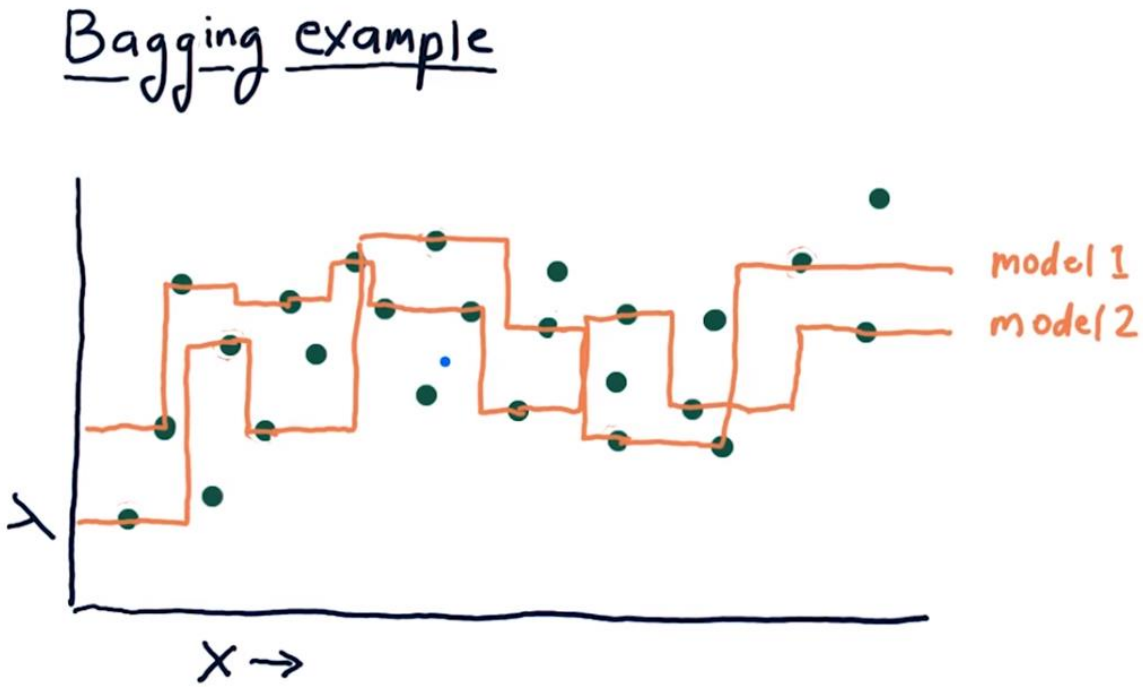


- **배깅**은 bootstrap을 사용하여 원래 훈련 자료에 대한 다수의 복사본을 만들고, 각 복사본에 별도의 의사결정트리를 적합하며 그 다음에 이 트리들을 모두 결합, 단일 예측 모형을 만든다
- **의사결정트리의 높은 분산 해결**
  : Bootstrap된 훈련 데이터에 대한 예측 결과를 평균하여 얻으면 분산 감소

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x).$$

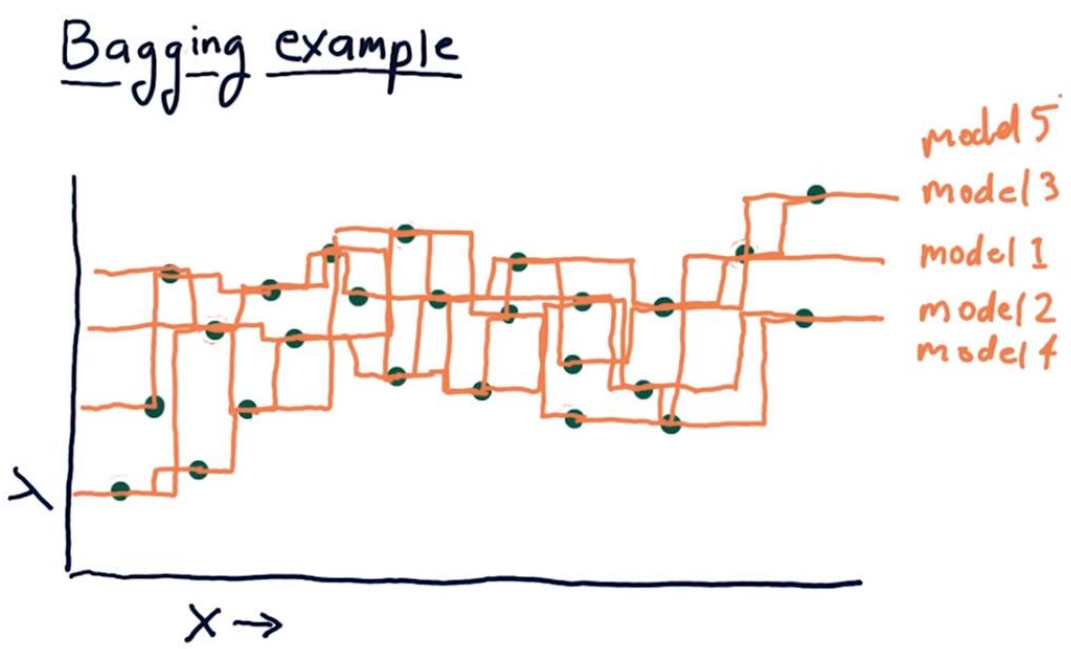
Bagging example

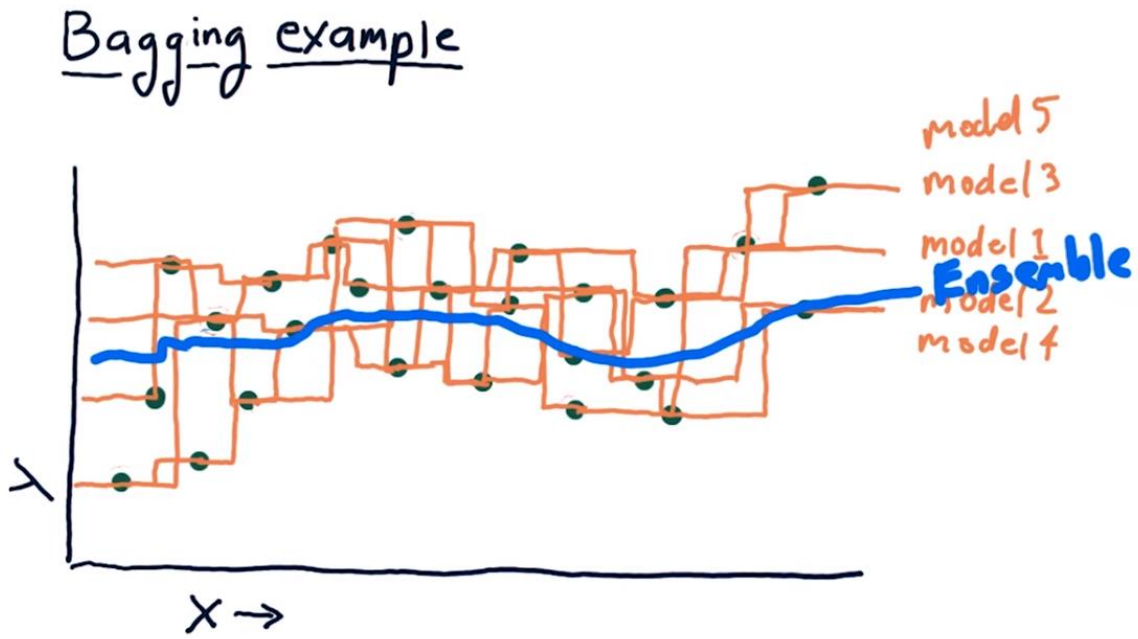




Growth Hackers

Bagging example





Growth Hackers

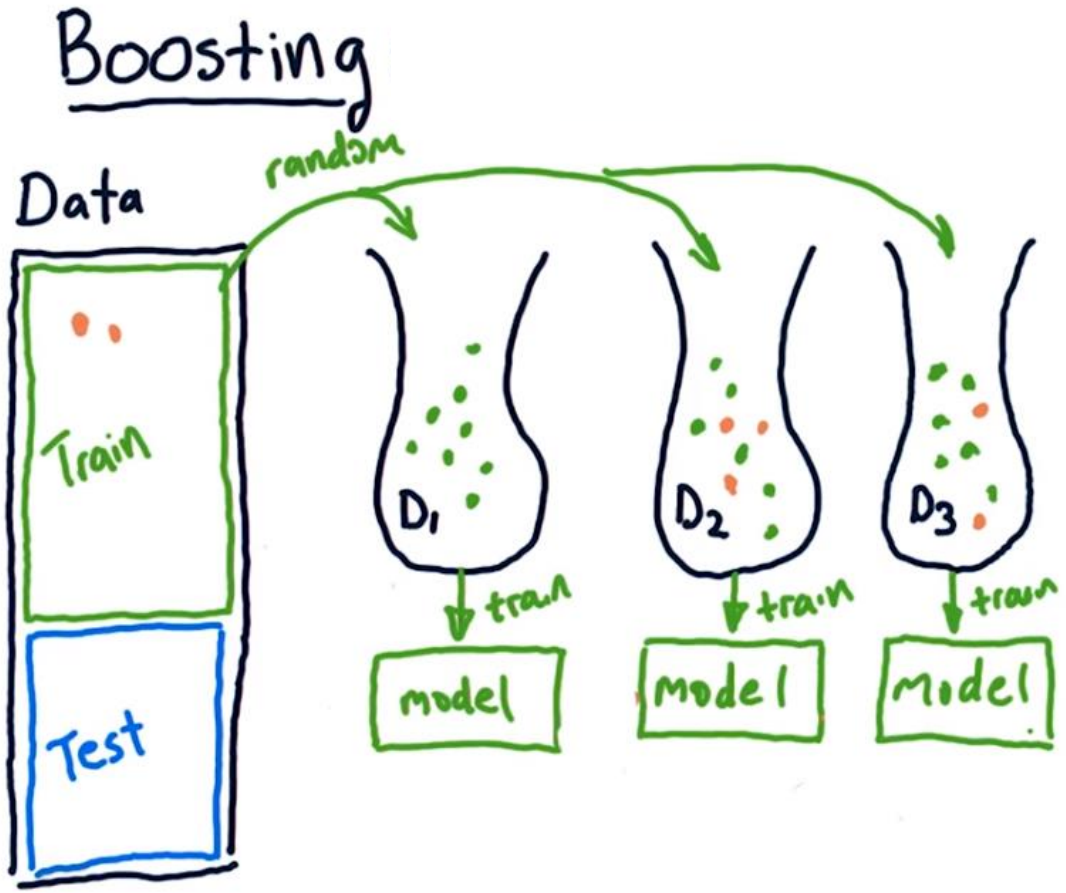
Random Forest

- **랜덤포레스트**는 배깅과 마찬가지로 bootstrap된 훈련 데이터들에 대해 다수의 의사결정트리를 만든다. 예측 성능은 배깅보다 좋다.

- 트리를 만들 때, 트리 내에서 분할이 고려될 때마다 p개 설명변수들의 전체 집합에서 **m개 설명변수들로 구성된 랜덤표본**이 분할 후보로 선택된다. (m ≈ $\sqrt{p}$, 배깅은 m=p)

- 거의 다 유사한 트리만 만들지 않도록 변수의 개수를 제한
  -> 다른 설명변수들에게 더 많은 기회를 주고, 상관성 제거

Boosting



- **부스팅**도 배깅과 유사, 다른 점은 **트리들이 순차적으로 만들어진다**는 것
- 각 트리는 이전에 만들어진 트리들로부터의 정보를 사용하여 만들어진다. 모델 적합 후 **잔차를 다음 모델의 반응변수**로 놓고 새로 적합.
- 결과 Y가 아니라 현재의 잔차들을 반응변수로 적합 -> 작은 트리들이 이 잔차들에 적합함으로, 좋은 성능을 내지 못하는 영역을 천천히 개선한다

Coding

Bagging	RandomForest	Boosting
원 데이터를 bootstrap해서 만들어진 여러 복사본에 각 모델 적합해 예측 평균내기	Bagging과 유사 단, 변수 선택이 들어감	Bagging과 유사 단, 잔차를 반응변수로 놓고 새로 적합
sklearn.ensemble. BaggingClassifier()	sklearn.ensemble. RandomForestClassifier()	xgboost. XGBClassfier()

Coding

Iris 데이터를 활용한 랜덤포레스트 & 부스팅

0. Iris 데이터 및 패키지 불러오기

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
!pip3 install xgboost
import xgboost
%matplotlib inline

from sklearn import datasets

iris = datasets.load_iris()
```

패키지 설치가 안 되어 있나요?
Terminal 창에

pip install seaborn
pip install xgboost
주피터에서는 !pip3 install xgboost

Growth Hackers

Coding

1. Iris 데이터 시각화

In [5]:
```
# Iris 데이터를 pandas의 dataframe으로 만들고, 데이터 형태를 살펴보자.
df = pd.DataFrame(iris.data, columns=iris.feature_names)
df.head()
```

Out[5]:

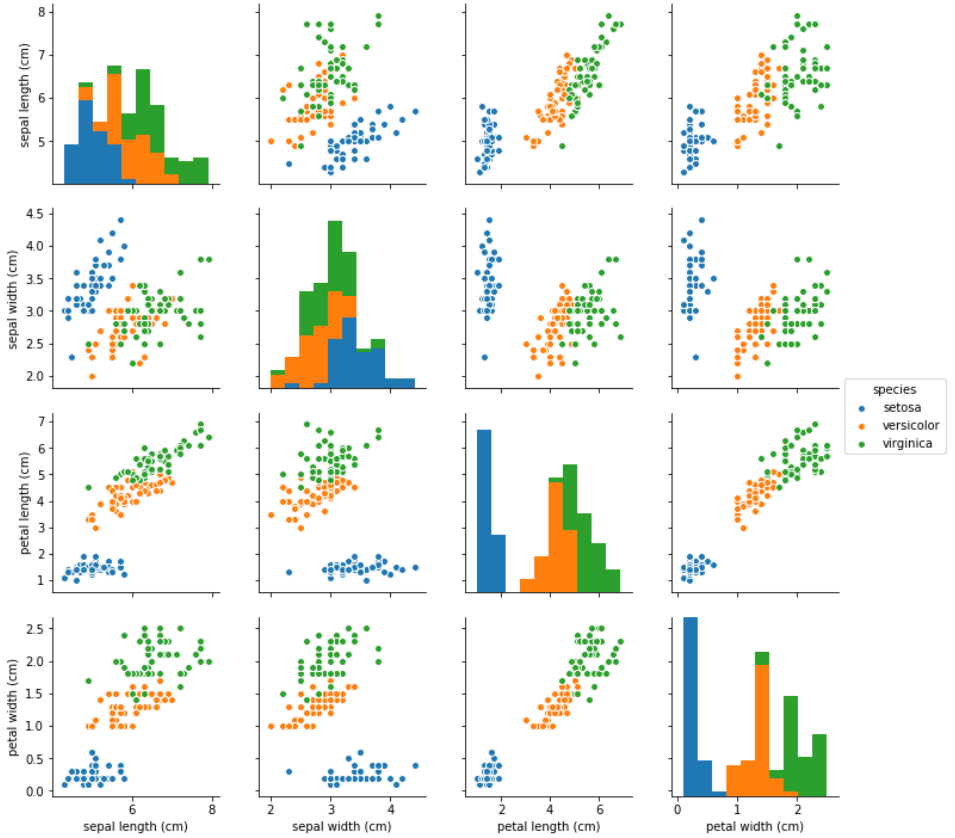
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Coding

In [6]:
```
# iris 데이터의 species는 classification을 위해 정수 형태(0, 1, 2)로 되어 있음
# 시각화를 위해 species integer를 species name으로 바꾼 새로운 열을 df에 추가
df['species'] = np.array([iris.target_names[i] for i in iris.target])

# 시각화 라이브러리인 seaborn으로 pairplot을 그려보자.
sns.pairplot(df, hue='species')
```

Out [6]: <seaborn.axisgrid.PairGrid at 0x1d2538e5c18>



Growth Hackers

Coding

2. train set, test set 만들기

In [4]:
```
# sklearn에서 train_test_split 함수 불러오기
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df[iris.feature_names], iris.target,
                                                    test_size=0.25, stratify=iris.target, random_state=123456)

# train_test_split(x, y, test_size, stratify, random_state)
# test_size : 테스트 데이터 사이즈
# train_size : 트레인 데이터 사이즈
# stratify : 클래스 라벨
# random_state : 난수 시드
```

Coding

3. 랜덤포레스트 모델 만들기

In [8]:
```
# sklearn에서 RandomForestClassifier 함수 불러오기
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=100, oob_score=True, random_state=123456)
# n_estimators : 생성할 트리의 개수
# oob_score : out-of-bag score, 예측이 얼마나 정확한가에 대한 추정을 수치로 나타낸 것

rf.fit(X_train, y_train)
# rf.fit(features, targets)
```

Out[8]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=1,
            oob_score=True, random_state=123456, verbose=0,
            warm_start=False)
```

Coding

4. test set에 적합하기

In [9]:
```
# sklearn에서 accuracy_score 함수 불러오기
from sklearn.metrics import accuracy_score

predicted = rf.predict(X_test) # rf 모델에 X_test를 넣고 그 예측값을 predicted에 저장
accuracy = accuracy_score(y_test, predicted) # 실제 데이터와 예측값이 일치하는 비율

print(f'Out-of-bag score estimate: {rf.oob_score_:.3}')
print(f'Mean accuracy score: {accuracy:.3}')
```

```
Out-of-bag score estimate: 0.946
Mean accuracy score: 0.921
```

Coding

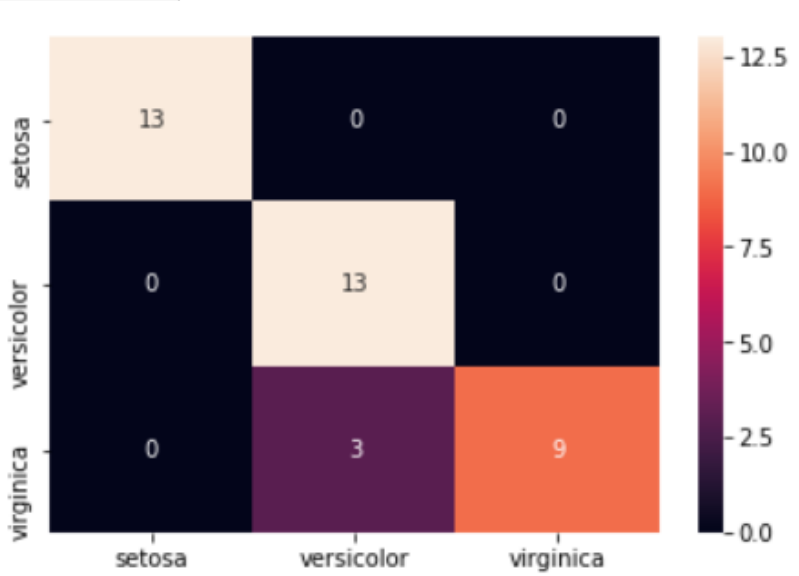
5. confusion matrix로 분류 결과 시각화

In [10]:
```
# sklearn에서 confusion_matrix 함수 불러오기
from sklearn.metrics import confusion_matrix

cm = pd.DataFrame(confusion_matrix(y_test, predicted), columns=iris.target_names, index=iris.target_names)
# confusion_matrix는 라벨이 있는 경우 분류 모델을 평가하는 방법
# column은 predicted, row는 y_test
sns.heatmap(cm, annot=True) # sns 라이브러리에 있는 heatmap으로 cm을 시각화

# setosa는 정확히 분류해 내고 있지만, versicolor와 virginica의 구별은 명확하지 않다.
```

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d254c5efd0>
```



Growth Hackers

Coding

6. xgboost 모델 만들기

In [14]:
```
xgb = xgboost.XGBClassifier(n_estimators=100, max_depth=2)
xgb.fit(X_train, y_train)
```

Out[14]:
```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_step=0,
       max_depth=2, min_child_weight=1, missing=None, n_estimators=100,
       n_jobs=1, nthread=None, objective='multi:softprob', random_state=0,
       reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
       silent=True, subsample=1)
```

Coding

7. test set에 적합하기

In [20]:
```
predicted_xgb = xgb.predict(X_test) # xgb 모델에 X_test를 넣고 그 예측값을 predicted에 저장
accuracy_xgb = accuracy_score(y_test, predicted_xgb) # 실제 데이터와 예측값이 일치하는 비율

print(f'Mean accuracy score: {accuracy_xgb:.3}')
```

```
Mean accuracy score: 0.947
```

```
C:₩Users₩yuniv₩AppData₩Local₩Continuum₩anaconda3₩lib₩site-packages₩sklearn₩preprocessing₩label.py:151: DeprecationWa
rning: The truth value of an empty array is ambiguous. Returning False, but in future this will result in an error.
Use `array.size > 0` to check that an array is not empty.
  if diff:
```

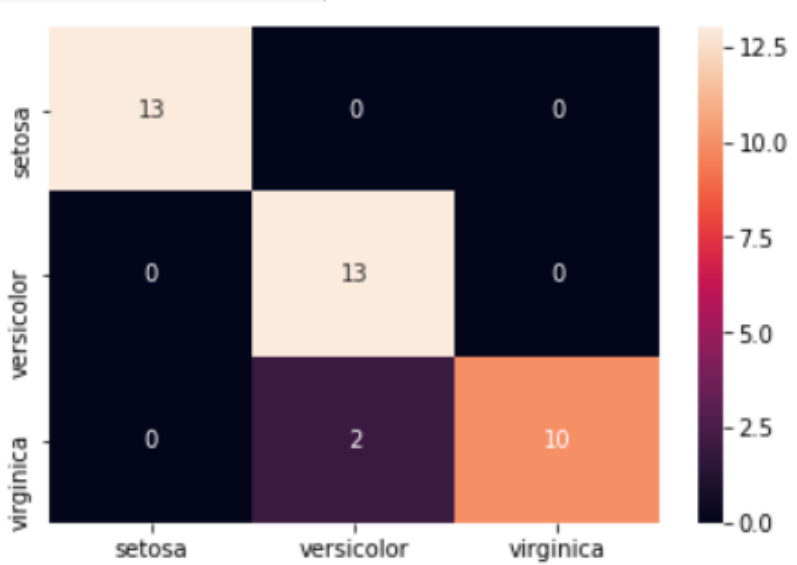
Coding

8. confusion matrix로 분류 결과 시각화 ¶

In [18]:
```
cm_xgb = pd.DataFrame(confusion_matrix(y_test, predicted_xgb), columns=iris.target_names, index=iris.target_names)
sns.heatmap(cm_xgb, annot=True) # sns 라이브러리에 있는 heatmap으로 cm_xgb를 시각화

# 랜덤포레스트보다 예측 성능이 좋아졌다
```

Out[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d25506ddd8>
```



Growth Hackers

Quest

- **데이터 : scikit-learn 기본 데이터 breast_cancer**
- **목표 : RandomForest와 Boosting을 활용한 Breast cancer 양성, 악성 예측**

- 1. import library, load data, check data
- 2. divide data into x, y
- 3. split data into train set, test set
- 4. use RandomForestClassifier and show confusion matrix
- 5. use XGBClassifier and show confusion matrix
- 6. make some simple comments about the results

Quest

- breast_cancer 데이터는 sklearn.datasets에 내장되어 있습니다 아래 코드를 참고해주세요

```
# Imports
import numpy as np
import pandas as pd
from sklearn.datasets import load_breast_cancer
```

- 제출 시 파일명은 "Session12 이름"으로 해주세요
- 제출 형식 : py 파일 + confusion matrix 2개 캡쳐본(with title)

Growth Hackers

References

- GH 2기 남정우, 이영준, 차유경 - 의사결정나무 세션 자료
- DECISION TREE 알고리즘 예제 http://jihoonlee.tistory.com/16
- BOOSTING 기법의 이해 www.slideshare.net/freepsw/boosting-bagging-vs-boosting
- ENSENBLE https://www.youtube.com/watch?v=Un9zObFjBH0&t=9s
- BAGGING https://www.youtube.com/watch?v=2Mg8QD0F1dQ&t=35s
- BAGGING EXAMPLE https://www.youtube.com/watch?v=sVriC_Ys2cw
- BOOSTING https://www.youtube.com/watch?v=GM3CDQfQ4sw
- Introduction to Statistical Learning https://www-bcf.usc.edu/~gareth/ISL/ISLR%20First%20Printing.pdf
- RANDOMFOREST PYTHON PRACTICE https://partrita.github.io/posts/random-forest-python/
- XGBOOST https://brunch.co.kr/@snobberys/137