머신러닝응용 제10강

Ensemble Learning 1.

첨단공학부 김동하교수

제10강 Ensemble Learning 1.

1	앙상블 기법을 사용하는 이유와 효과에 대해서 학습한다.
2	배깅 방법론에 대해 학습한다.
3	랜덤 포레스트 방법론에 대해 학습한다.

핵심 단어

- 앙상블 기법
- 붓스트랩 자료
- 배깅
- 랜덤 포레스트

10강. Ensemble Learning 1.

01. 앙상블 기법

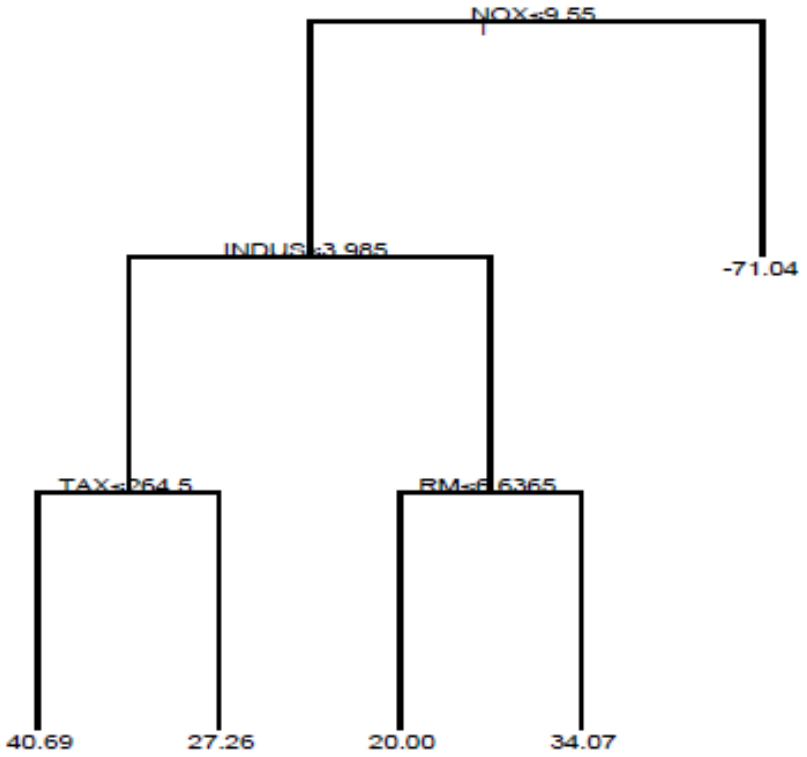
1) 학습의 불안정성

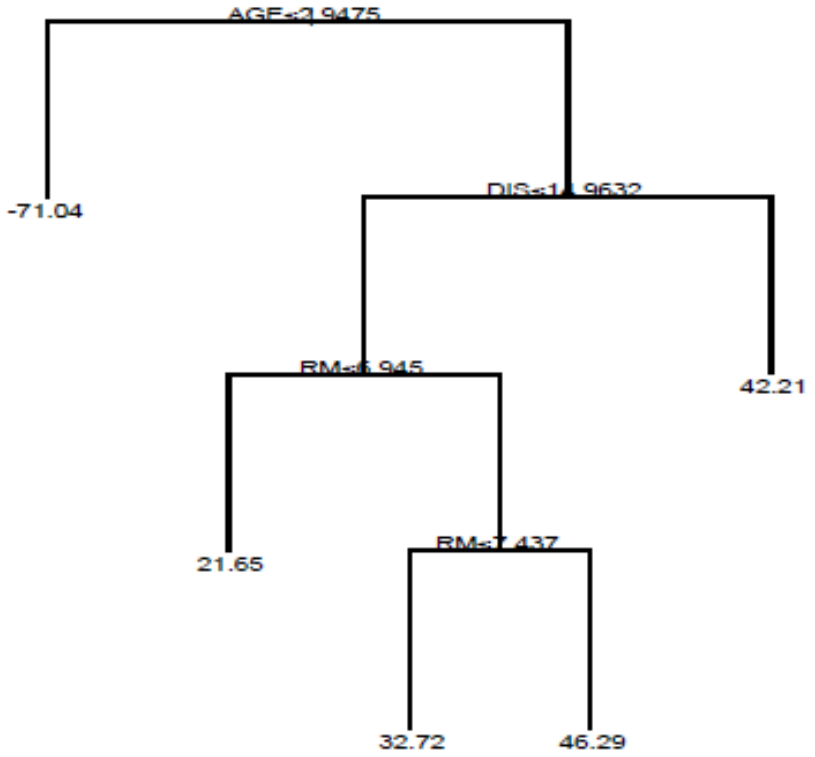
- 모든 모형은 학습의 불안정성을 가짐
  - 학습자료의 작은 변화에 의해 예측모형이 변하는 것.

- 아무리 좋은 학습 방법을 사용하더라도 복잡한 모형일수록 높은 불안정성을 가짐.
  - 선형 모형은 낮은 불안정성을 가짐.
  - 의사결정나무는 높은 불안정성을 가짐.

1) 학습의 불안정성

◆ 의사결정나무의 불안정성





출처: 박창이 등. *R을 이용한 데이터마이닝*. 교우사, 2018.

1) 학습의 불안정성

◆ 의사결정나무의 불안정성

- 동일한 자료에서 서로 다른 표본을 생성했을 때 의사결정나무가 크게 달라진다.

- 학습 방법의 불안정성
  - 예측력의 저하.
  - 예측모형의 해석을 어렵게 만듦.

2) 앙상블 기법

- 불안정한 학습방법을 안정적으로 만들어 좋은 예측력을 갖기 위한 방법.
- 주어진 자료로부터 여러 개의 예측 모형을 만든 후 결합하여 하나의 최종 예측 모형을 만드는 기법.
- 예측력을 획기적으로 향상시킬 수도 있음이 경험적으로 입증되었음.

2) 앙상블 기법

- 앙상블 기법의 예
  - 배깅 (Bagging)
  - 랜덤 포레스트 (Random Forest)
  - 부스팅 (Boosting)

10강. Ensemble Learning 1.

02. 배깅

1) 배깅 알고리즘

- Bagging (Bootstrap Aggregating)
  - Davison and Hinkley (1997)

- 주어진 자료에 대해 여러 개의 붓스트랩 (bootstrap) 자료를 생성하고, 각 붓스트랩 자료에 대해 예측 모형을 만든 후 결합하여 최종 예측 모형을 만드는 기법.

1) 배깅 알고리즘

- 붓스트랩 (Bootstrap) 자료란?

  - 주어진 자료로부터 동일한 크기의 표본을 랜덤 복원 추출로 뽑은 자료.

1) 배깅 알고리즘

◆ 학습 자료: $L = \{(x_i, y_i)\}_{i=1}^{n}$

1. B개의 붓스트랩 자료 $L^{(b)}, b = 1, \ldots, B$를 만든다.

2. 각 붓스트랩 자료 $L^{(b)}$ 에 대해 예측 모형 $\hat{f}^{(b)}(\cdot)$를 구축한다.

3. B 개의 예측 모형을 결합하여 최종 모형 $\hat{f}(\cdot)$를 만든다. 최종 모형을 만드는 방법은 다음과 같다.

1) 배깅 알고리즘

- (회귀 모형) 평균 값을 이용

$$\hat{f}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{(b)}(x)$$

- (분류 모형) 최빈값 (다수결)을 이용

$$\hat{f}(x) = argmax_k \left(\sum_{b=1}^{B} I\left(\hat{f}^{(b)}(x) = k\right) \right)$$

2) 배깅 알고리즘의 효과

- **Breiman (1996) 논문의 예시 중 일부**
  - **회귀 문제에서 의사결정나무를 활용한 경우**

Data Set	$\bar{e}_S$	$\bar{e}_B$	Decrease
Boston Housing	19.1	11.7	39%
Ozone	23.1	18.0	22%
Friedman # 1	11.4	6.2	46%
Friedman # 2	30,800	21,700	30%
Friedman # 3	.0403	.0249	38%

2) 배깅 알고리즘의 효과

Breiman (1996) 논문의 예시 중 일부

- 분류 문제에서 의사결정나무를 활용한 경우

Data Set	Samples	Variables	Classes	$\bar{e}_S$	$\bar{e}_B$	Decrease
waveform	300	21	3	29.0	19.4	33%
heart	1395	16	2	10.0	5.3	47%
breast cancer	699	9	2	6.0	4.2	30%
ionosphere	351	34	2	11.2	8.6	23%
diabetes	1036	8	2	23.4	18.8	20%
glass	214	9	6	32.0	24.9	22%
soybean	307	35	19	14.5	10.6	27%

3) 사용할 모형의 선택

- 배깅 알고리즘은 매우 단순하지만 불안정한 학습방법의 예측력을 획기적으로 향상시켜줌.
- 즉, 학습방법이 불안정할수록 배깅의 효과는 증가.
- 따라서, 의사결정나무를 활용하는 것이 가장 큰 효과를 불러올 수 있음.

3) 사용할 모형의 선택

- 가지치기를 하지 않은 나무의 사용 (더 큰 불안정성)
- 가지치기는 의사결정나무 구축시 가장 많은 계산량을 요구
- 최대로 성장시킨 의사결정나무를 사용함으로써 계산량을 대폭 줄일 수 있음.

3) 사용할 모형의 선택

- 왜 배깅에서는 가지치기가 필요 없는가?

- 가지치기를 하지 않은 채 최대로 성장시킨 나무를 사용하는 것이 배깅 알고리즘의 효과를 극대화할 수 있기 때문!

10강. Ensemble Learning 1.

03. 랜덤 포레스트

1) 랜덤 포레스트

- Breiman (2001)

- 배깅보다 더 많은 무작위성을 주어 예측 모형들을 생성한 후 이를 선형결합하여 최종 예측 모형을 만드는 방법.

- 이론적 설명은 부족하지만 일반적으로 예측력은 매우 높음.

1) 랜덤 포레스트

- 특히, 입력변수의 개수가 많을 때에 배깅이나 부스팅과 비교했을 때 비슷하거나 더 좋은 예측력을 보이는 경우가 많음.

- 랜덤 포레스트는 무작위성을 주기 위해 붓스트랩과 더불어 입력변수들에 대한 무작위 추출을 결합.

- 따라서, 서로 연관성이 약한 학습기를 여러 개 만들어 내는 기법이라 할 수 있음.

2) 랜덤 포레스트 알고리즘

◆ 다음의 과정을 B번 반복한다.

1. 훈련자료 $L = \{(x_i, y_i)\}_{i=1}^{n}$에 대해서 붓스트랩 자료 $L^{(b)}$를 생성.

2. 입력 변수들 중 $k(k \ll p)$개만 무작위로 뽑은 후에 이들을 이용해서 의사결정나무 $f_b(x)$를 생성. 이 때, 의사결정나무는 정해놓은 s 단계까지만 진행.

2) 랜덤 포레스트 알고리즘

- 최종 모형은 배깅과 같은 방식으로 만든다.
- (회귀 모형) 평균 값을 이용

$$\hat{f}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{(b)}(x)$$

- (분류 모형) 최빈값 (다수결)을 이용

$$\hat{f}(x) = argmax_k \left(\sum_{b=1}^{B} I\left(\hat{f}^{(b)}(x) = k\right) \right)$$

3) 랜덤 포레스트 조율모수 선택

- 각 의사결정나무마다 선택할 설명변수의 개수인 k를 어떻게 선택할 것인지 등 여러 가지 선택이 있음.

- 평가용 자료 또는 교차검증법(CV) 를 활용하여 선택.

4) 모형의 시각화

- 배깅, 랜덤 포레스트, 부스팅과 같이 예측 방법이 단순하지 않을 경우에는 각 변수들의 영향력을 시각화하기가 쉽지 않음.
  - 즉, 모형의 해석이 쉽지 않음.

4) 모형의 시각화

- 부분 의존성 그림 (Partial dependence plot)
  - 특정 변수가 예측 모델에 어떤 영향을 미쳤는지 알기 위한 그래프.

- 변수 중요도 그림 (Variable importance plot)
  - 설명 변수가 종속 변수를 설명하는 정도를 수치화하여 변수별 중요도를 확인할 수 있는 막대 그래프.

10강. Ensemble Learning 1.

04. Python을 이용한 실습

1) 데이터 설명

보스턴 주택 가격 데이터

- 1978년에 발표된 데이터로 미국 보스턴 지역의 주택 가격에 영향을 미치는 요소들을 정리

- 총 13가지의 요소들과 주택 가격으로 이루어져 있음.

1) 데이터 설명

- 앙상블 기법들을 이용하여 주택 가격을 예측하는 회귀 모형을 만들자.
  - 기존의 의사결정나무의 성능과 비교해보자.

2) 환경설정

◆필요한 패키지 불러오기

```
import pandas as pd
import numpy as np
import os
from sklearn.datasets import load_boston
from sklearn.model_selection import train_test_split

from sklearn.metrics import confusion_matrix
#from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import DecisionTreeRegressor
#from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import RandomForestRegressor
#from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import BaggingRegressor
```

3) 데이터 불러오기

◆데이터 불러오기

```
boston = load_boston()

X_boston = pd.DataFrame(boston.data, columns=boston.feature_names)
y_boston = pd.DataFrame(boston.target, columns=['MEDV']).iloc[:,0]
```

3) 데이터 불러오기

◆데이터 분할하기

```
X_train, X_test, y_train, y_test = \
train_test_split(X_boston, y_boston,
                 test_size = 0.3, random_state=123)
```

5) 의사결정나무 적합하기

- 최대 깊이는 5
- 시험 데이터에 적합하여 성능을 측정하자.
  - 성능 측도는 평균 제곱합

```
tree = DecisionTreeRegressor(max_depth = 5,
                             random_state = 0)
tree.fit = tree.fit(X_train, y_train)
```

5) 의사결정나무 적합하기

- 최대 깊이는 5
- 시험 데이터에 적합하여 성능을 측정하자.
  - 성능 측도는 평균 제곱합

```
tree_pred = tree.predict(X_test)
print((y_test-tree_pred).pow(2).mean())
```

```
17.602204873990754
```

6) 배깅 적합하기

- 1000개의 모형을 사용하자.

```
tree = DecisionTreeRegressor()
bag_model = BaggingRegressor(tree, n_estimators=1000,
                             max_samples=0.8, random_state=1)
bag_model.fit(X_train, y_train)
```

6) 배깅 적합하기

◆ 시험 데이터에서의 성능 확인하기.

```
bag_pred = bag_model.predict(X_test)
print((y_test-bag_pred).pow(2).mean())
```

```
14.551381714144748
```

7) 랜덤 포레스트 적합하기

- 1000개의 모형을 사용하자.

```
rf_model = RandomForestRegressor(n_estimators = 1000,
                                 max_features = 3,
                                 min_samples_leaf = 3)
rf_model.fit(X_train, y_train)
```

7) 랜덤 포레스트 적합하기

◆ 시험 데이터에서의 성능 확인하기.

```
rf_pred = rf_model.predict(X_test)
print((y_test-rf_pred).pow(2).mean())
```

```
16.304695711806463
```

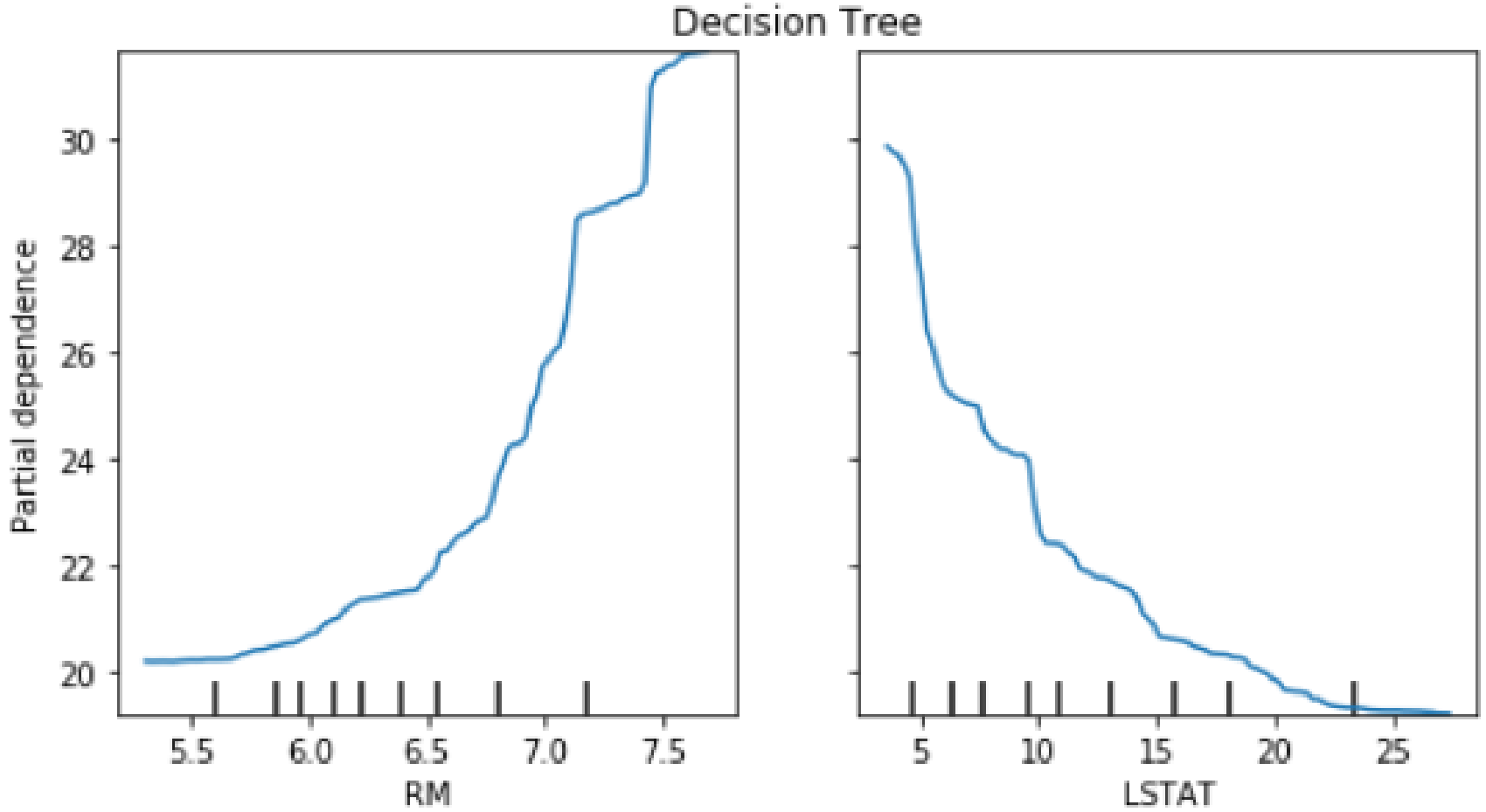
8) 모형 결과 시각화하기

◆ 부분 의존성 그림 그리기.

```
from sklearn.inspection import plot_partial_dependence
fig, ax = plt.subplots(figsize=(8, 4))
ax.set_title("Decision Tree", fontsize=12)
tree_disp = plot_partial_dependence(rf_model, X_train, ["RM","LSTAT"], ax=ax)
```

8) 모형 결과 시각화하기

부분 의존성 그림 그리기.



8) 모형 결과 시각화하기

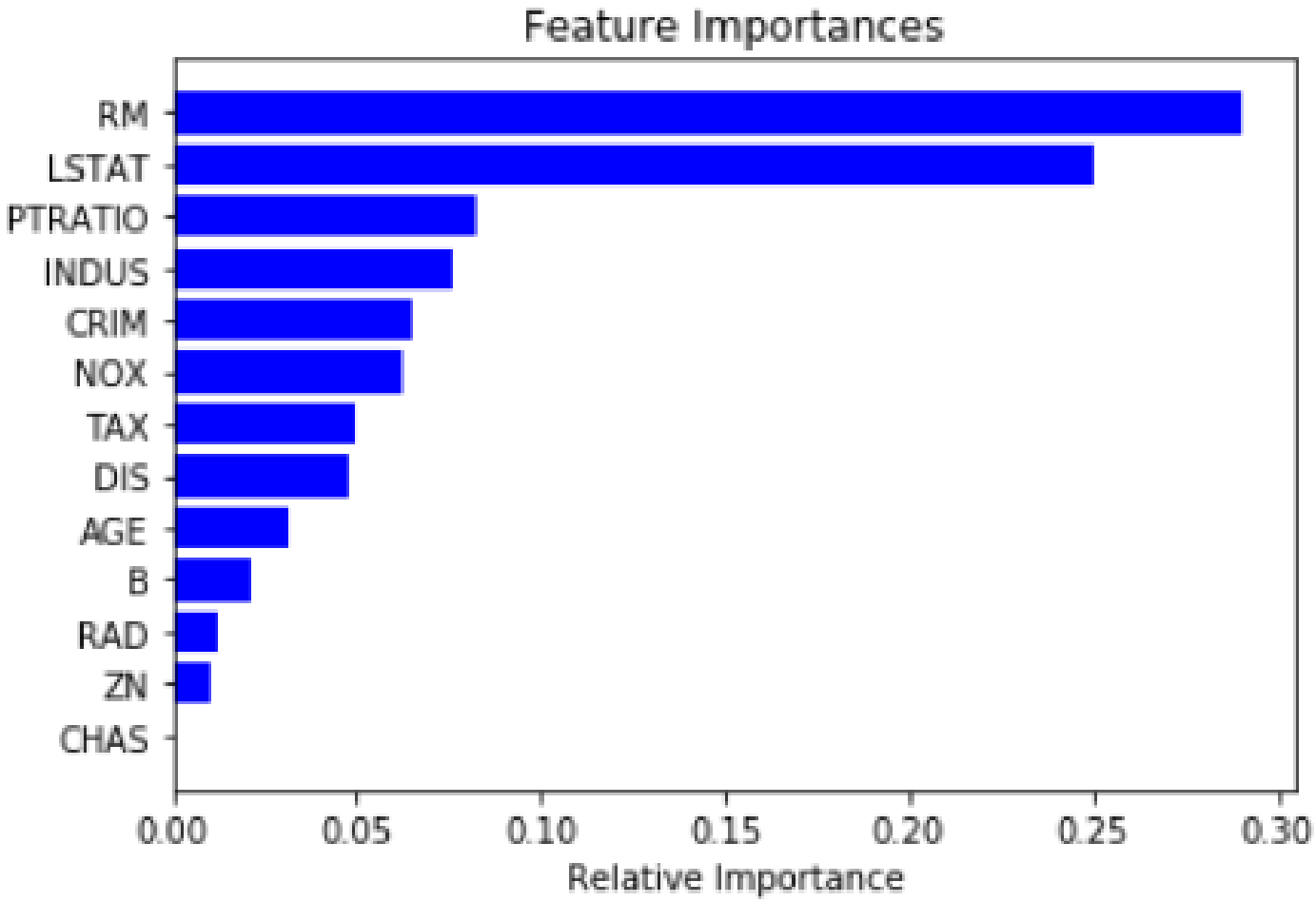
◆ 변수 중요도 그림 그리기.

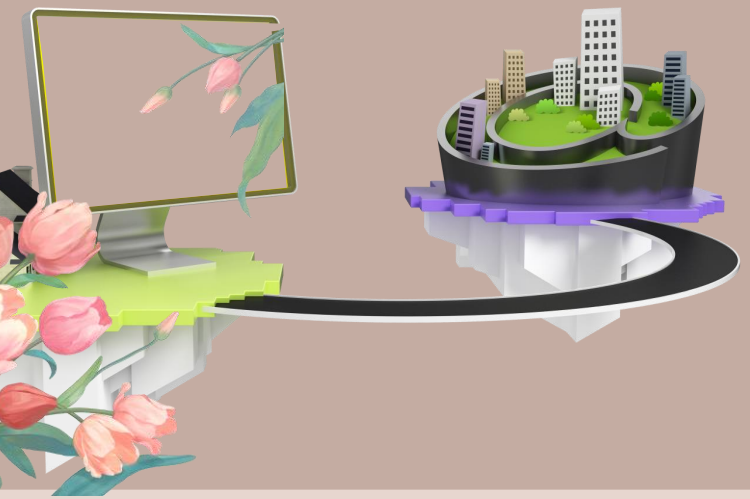
```
importances = rf_Model.feature_importances_
indices = np.argsort(importances)

plt.title('Feature Importances')
plt.barh(range(len(indices)), importances[indices], color='b', align='center')
plt.yticks(range(len(indices)), [boston.feature_names[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()
```

8) 모형 결과 시각화하기

변수 중요도 그림 그리기.





다음시간안내

제11강

Ensemble Learning 2.