MML 프로젝트

20192128
전자공학과
윤성철

Part 1, 구현코드

구현코드

In [346]:

```
#필요한 라이브러리를 간소화
import numpy as np
import pandas as pd
from sklearn.model_selection import ShuffleSplit
import visuals as vs
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings(action='ignore')

#주택 가격은 MEDV이므로 반응변수로 설정
%matplotlib inline
data = pd.read_csv('data.csv')
prices = data['MEDV']
features = data.drop('MEDV', axis = 1)
print("부동산 데이터는 {} 개의 데이터가 있고 {} 의 변수가 존재 한다.".format(*data.shape))

# 주택 최소값과 최대값, 평균값, 중간값, 표준편차를 확인한다.
minimum_price = np.amin(prices)
maximum_price = np.amax(prices)
mean_price = np.mean(prices)
median_price = np.median(prices)
std_price = np.std(prices)

print("주택값의 데이터:₩n")
print("최소값: ${}".format(minimum_price))
print("최댓값: ${}".format(maximum_price))
print("평균값: ${}".format(mean_price))
print("중간값 ${}".format(median_price))
print("표준편차: ${}".format(std_price))
```

출력결과

```
부동산 데이터는 506 개의 데이터가 있고 13 의 변수가 존재 한다.
주택값의 데이터:

최소값: $5.0
최댓값: $50.0
평균값: $22.532806324110698
중간값 $21.2
표준편차: $9.188011545278206
```

구현코드

In [315]:
```
%matplotlib inline
# 산점도와 히스토그램으로 데이터를 분석한다.
sns.pairplot(data, height=2.5)
plt.tight_layout()
```

출력 결과는 다음 페이지에 작성하였고,

파일이 크기에 figure들은 따로 별첨도 하였습니다.

구현코드

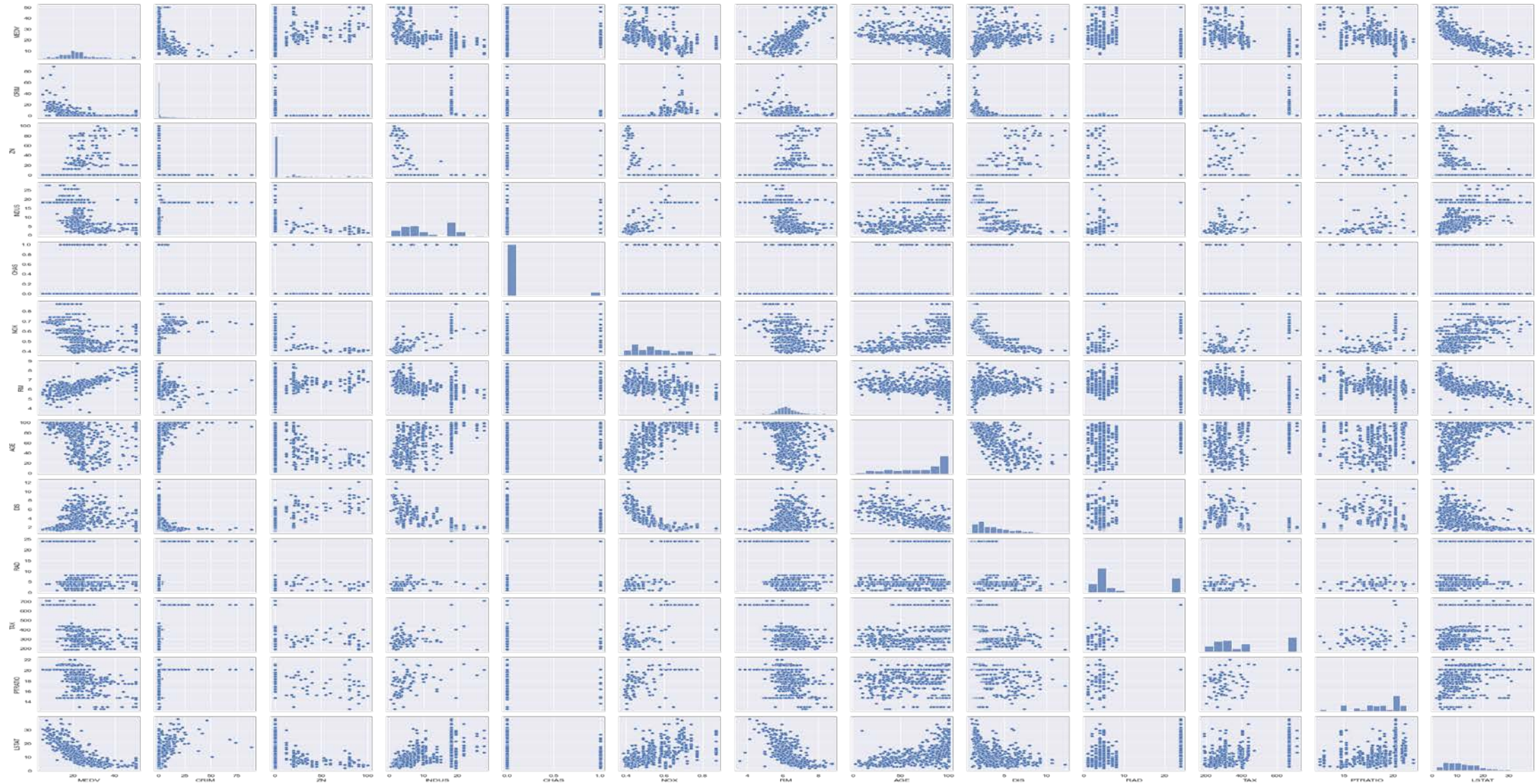

Figure 1.

구현코드

In [348]:
```
# 상관관계를 계산한다.
cm = np.corrcoef(data.values.T)
sns.set(font_scale=1.5)
# 상관관계를 나타내는 표를 나타낸다.
cm = np.corrcoef(data.values.T)
sns.set(font_scale=1)
hm = sns.heatmap(cm,
                 cbar=True,
                 annot=True,
                 square=True,
                 fmt='.2f',
                 annot_kws={'size': 8},
                 yticklabels='auto',
                 xticklabels='auto',cmap = "RdYlBu")
plt.tight_layout()
plt.show()
```

구현코드

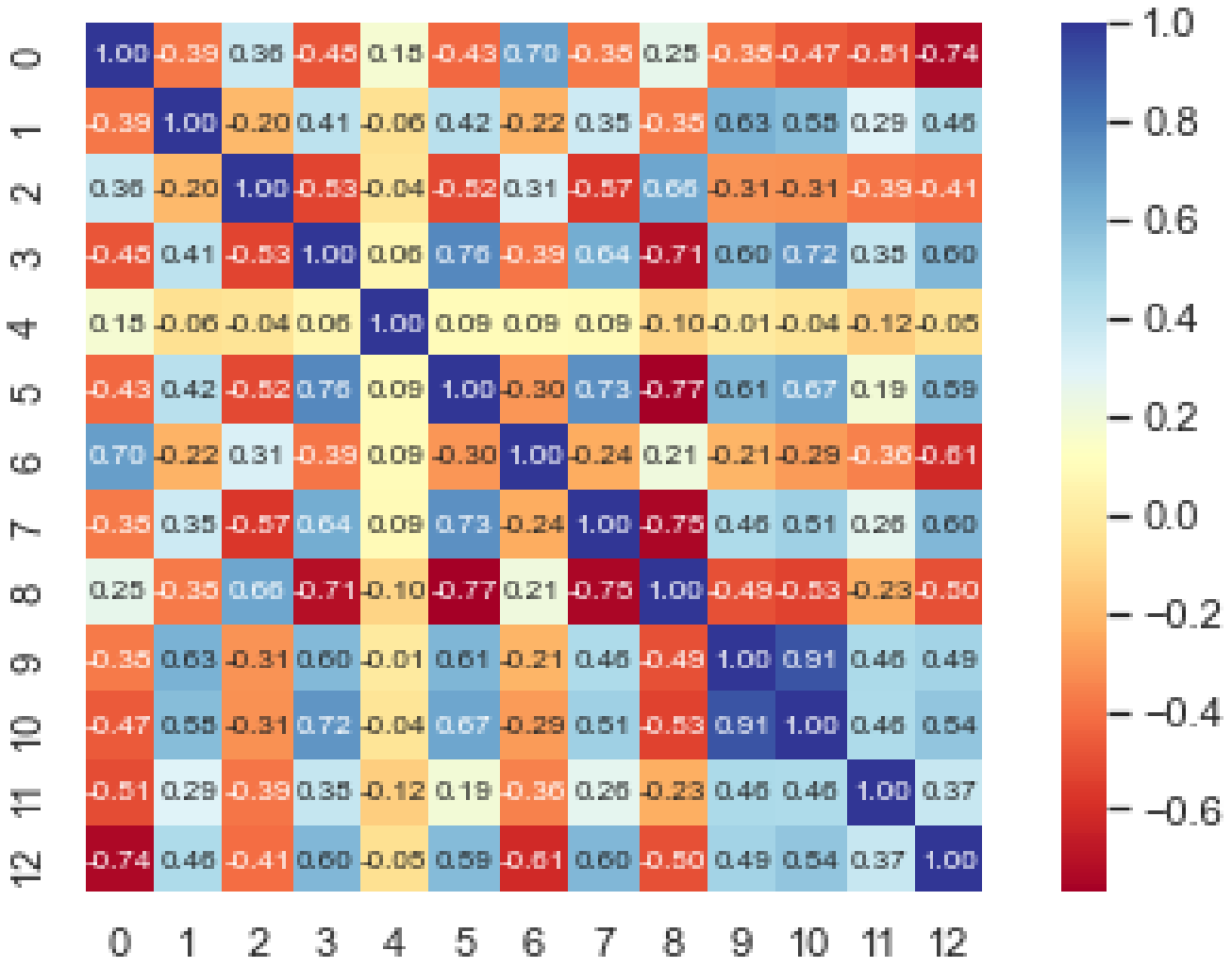

Figure 2.

0~12까지 순서대로

0. MEDV
1. CRIM
2. ZN
3. INDUS
4. CHAS
5. NOX
6. RM
7. AGE
8. DIS
9. RAD
10. TAX
11. PTRATIO
12. LSTAT

입니다.

구현코드

In [350]:
```
# 결정계수 R 정의

from sklearn.metrics import r2_score

def performance_metric(y_true, y_predict):

    score = r2_score(y_true, y_predict)
    return score

# train과 test를 분리한다.
from sklearn.model_selection import train_test_split

# 데이터를 섞은 후 분할하고 learning performance 곡선과 complexity 곡선을 생성한다.
X_train, X_test, y_train, y_test = train_test_split(features, prices, test_size=0.2, random_state = 150)
vs.ModelLearning(features, prices)
vs.ModelComplexity(X_train.values, y_train.values)
```

구현코드

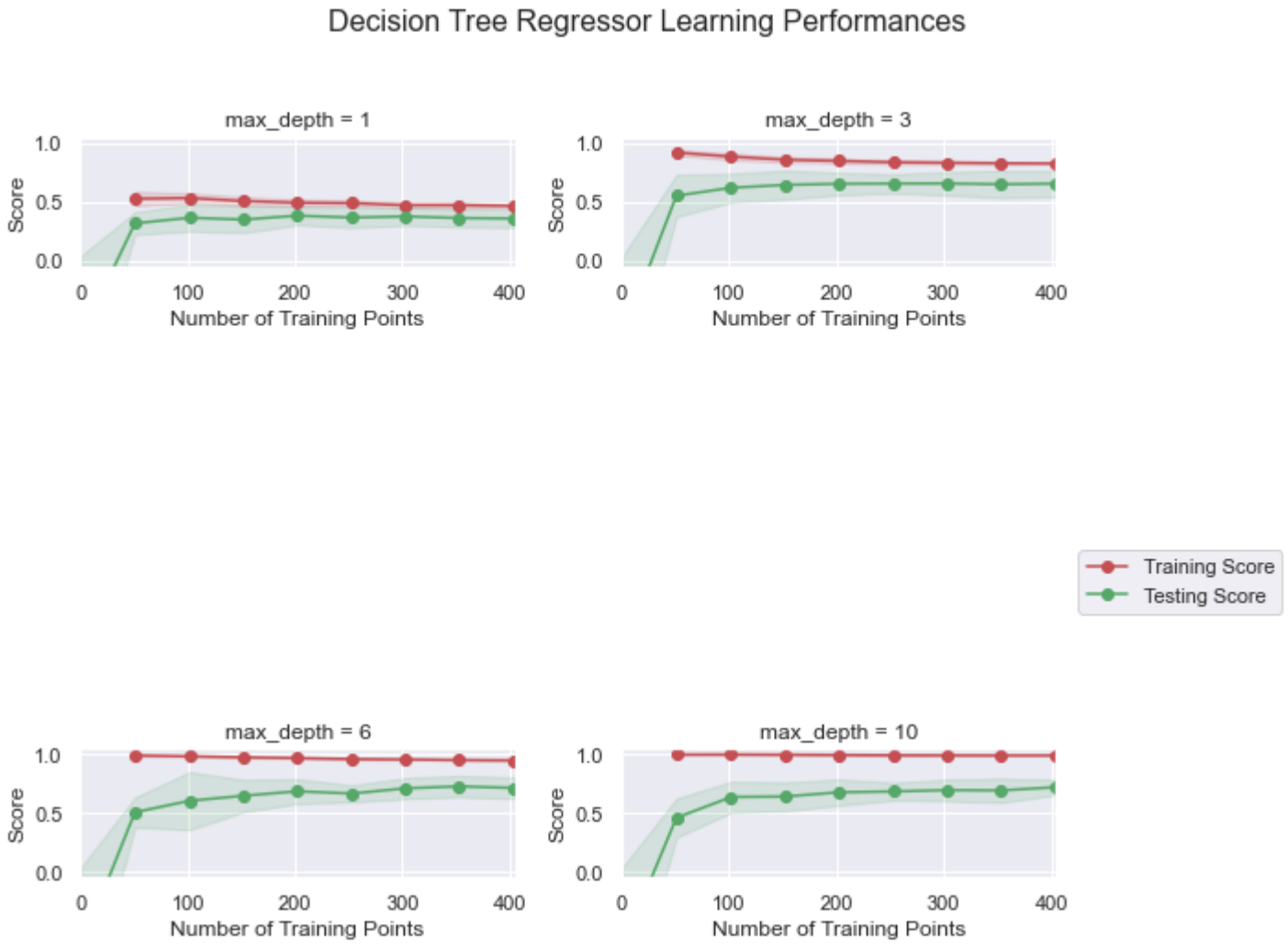

Figure 3.

Figure 4.

구현코드

In [351]:
```
param_grid = [ {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
               {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel': ['rbf']}, ]

# sklearn
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import make_scorer
from sklearn.model_selection import GridSearchCV

def fit_model(X, y):

    # 교차 검증 세트, 결정 회귀 트리를 설정하고 그것의 figure들을 설정
    cv_sets = ShuffleSplit(n_splits = 10, test_size = 0.20, random_state = 0)
    regressor = DecisionTreeRegressor()
    params = {'max_depth':[1,2,3,4,5,6,7,8,9,10]}
    scoring_fnc = make_scorer(performance_metric)
    grid = GridSearchCV(estimator=regressor, param_grid=params, scoring=scoring_fnc, cv=cv_sets)
    grid = grid.fit(X, y)
    return grid.best_estimator_
```

구현코드

In [352]:
```
#train한 것을 model화 하고 max depth의 값을 확인
reg = fit_model(X_train.values, y_train.values)
print("최적화 model의 max_depth는 {} 이다.".format(reg.get_params()['max_depth']))
```

```
최적화 model의 max_depth는 5 이다.
```

In [353]:
```
#figure값들을 입력하여 판매가격을 예측한다.
client_data = [[1,12.5,7.87,0,1,6.012,66.6,5,5,311,15,13]]
for i, price in enumerate(reg.predict(client_data)):
    print("Client의 주택 예상가격 :${:,.2f}₩n".format(price))

vs.PredictTrials(features, prices, fit_model, client_data)
```

```
Client의 주택 예상가격 :$20.78

Trial 1: $20.37
Trial 2: $20.53
Trial 3: $20.85
Trial 4: $20.39
Trial 5: $20.15
Trial 6: $20.67
Trial 7: $21.61
Trial 8: $21.58
Trial 9: $20.32
Trial 10: $21.49

Range in prices: $1.46
```

Part 2,

프로젝트 report

Part 2,

프로젝트 report

이번 프로젝트에서 주택 가격을 데이터의 값을 통해 예측하는 모델을 구현하였다.

우선, 주택 가격의 data인 MEDV를 반응 변수로 설정하고 나머지 data들 중 주택 가격과는 연관이 없다고 생각되는 data는 삭제하였다. 총 13개의 변수(반응 변수 포함)의 data를 활용하였다.

그 과정에서 mml에 필요한 library인 pandas와 sklearn을 활용하였다.

산점도와 히스토그램으로 데이터를 분석하고 heatmap을 통해서 상관관계를 나타내는 표를 만들었다.

회귀분석을 하기전에 결정계수인 R^2의 값을 구한 후 train과 test의 값을 분리하였다.

Train과 test값의 오차는 꽤 크게 나왔고, Training point가 증가할 수록 그 폭이 점차 감소하였다.

Part 2,

프로젝트 report

Complexity 곡선을 training과 validation의 두가지 그래프로 나타냈다. Max depth가 증가할 수록 training score가 1에 가깝게 다가가는 것을 확인할 수 있다.

마지막으로 model의 성능평가를 진행하였다. Model을 검증하고 피팅하는 과정을 거치고 프로젝트의 목표인 부동산 가격을 예측하였다. Client의 요구사항을 1~12까지의 변수(변수 0은 주택가격)를 입력하여 주택 가격을 예측하였다.

비례와 반비례관계는 상관 관계 figure을 확인해보면 알 수 있었다. 가장 큰 비례관계에 있는 것은 자택당 평균 방의 개수(RM)이고, 가장 큰 반비례관계에 있는 것은 빈곤층 비율(LSTAT)이었다.

변수가 많이 존재하지만, 데이터의 량이 많지 않았기에 정확한 값을 입력하지 않은 경우에는 오차가 크게 나오는 한계가 있었다. 데이터의 분산이 크기에 예측이 정확하지 않은 한계가 존재하였다.

Part 2,

프로젝트 report

저장된 모델은 여러가지 형태로 저장하였습니다. (Html, py , ipynb)