THE CHINESE UNIVERSITY OF HONG KONG, SHENZHEN

DDA 2020

MACHINE LEARNING

Assignment3 Report

Name: Xiang Fei

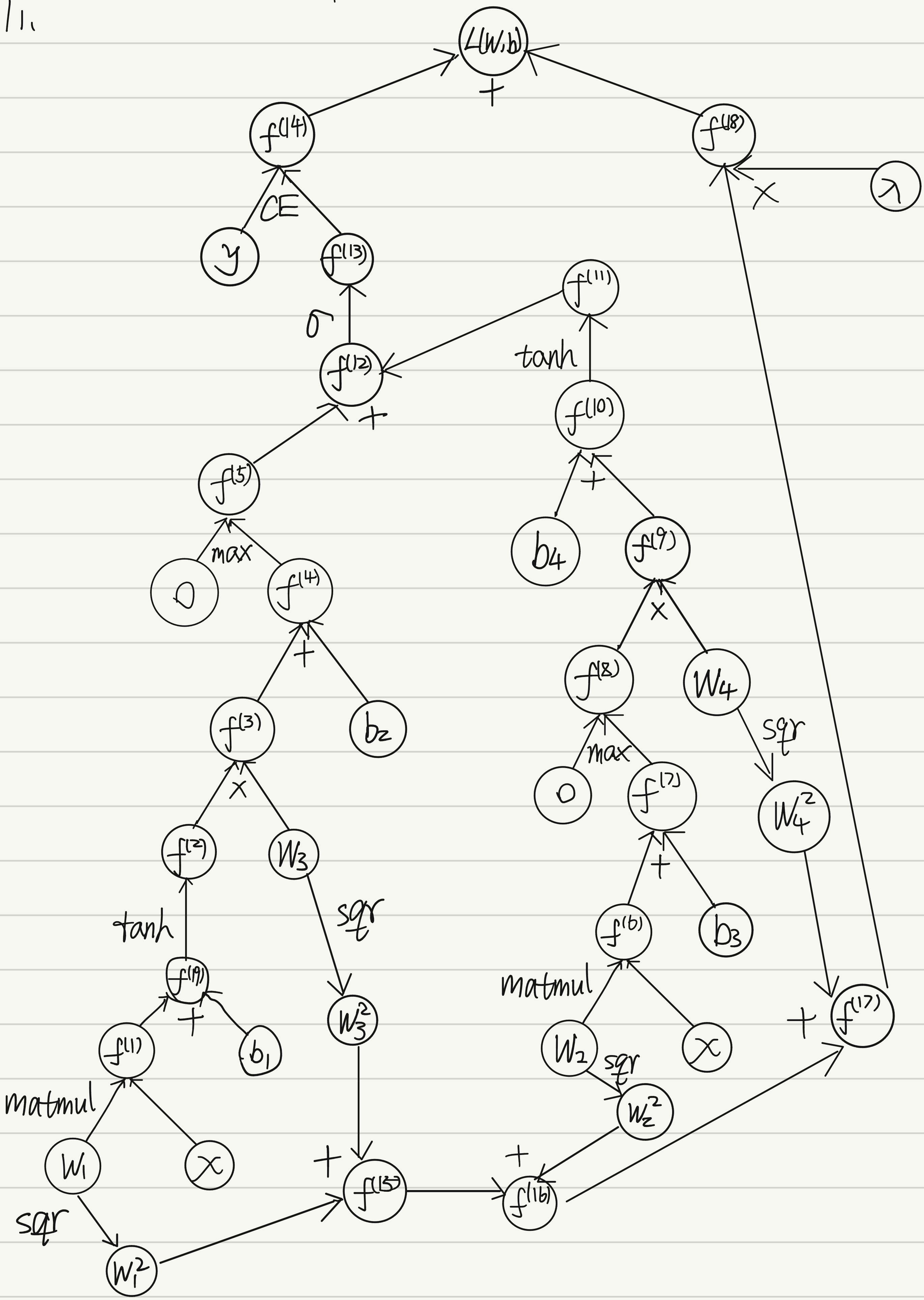
Student ID: 120090414

Contents

DDA2020 Assignment 3

Name：费祥 Student ID：120090414

T1.



$$\frac{\partial L(W,b)}{\partial f^{(14)}} = \frac{dL(W,b)}{dL} \cdot \frac{\partial L}{\partial f^{(14)}} = \frac{\partial L}{\partial f^{(14)}}$$

$$\frac{\partial L(W,b)}{\partial f^{(13)}} = \frac{\partial L}{\partial f^{(14)}} \cdot \frac{\partial f^{(14)}}{\partial f^{(13)}}$$

$$\frac{\partial L(W,b)}{\partial f^{(12)}} = \frac{\partial L}{\partial f^{(13)}} \cdot \frac{\partial f^{(13)}}{\partial f^{(12)}}$$

$$\frac{\partial L(W,b)}{\partial f^{(5)}} = \frac{\partial L}{\partial f^{(12)}} \cdot \frac{\partial f^{(12)}}{\partial f^{(5)}}$$

$$\frac{\partial L}{\partial f^{(4)}} = \frac{\partial L}{\partial f^{(5)}} \cdot \frac{\partial f^{(5)}}{\partial f^{(4)}}, \quad \frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial f^{(4)}} \cdot \frac{\partial f^{(4)}}{\partial b_2}$$

$$\frac{\partial L}{\partial f^{(3)}} = \frac{\partial L}{\partial f^{(4)}} \cdot \frac{\partial f^{(4)}}{\partial f^{(3)}}, \quad \frac{\partial L}{\partial f^{(2)}} = \frac{\partial L}{\partial f^{(3)}} \cdot \frac{\partial f^{(3)}}{\partial f^{(2)}}$$

$$\frac{\partial L}{\partial f^{(19)}} = \frac{\partial L}{\partial f^{(2)}} \cdot \frac{\partial f^{(2)}}{\partial f^{(19)}}, \quad \frac{\partial L}{\partial f^{(1)}} = \frac{\partial L}{\partial f^{(19)}} \cdot \frac{\partial f^{(19)}}{\partial f^{(1)}}$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial f^{(19)}} \cdot \frac{\partial f^{(19)}}{\partial b_1}, \quad \frac{\partial L}{\partial f^{(11)}} = \frac{\partial L}{\partial f^{(12)}} \cdot \frac{\partial f^{(12)}}{\partial f^{(11)}}$$

$$\frac{\partial L}{\partial f^{(10)}} = \frac{\partial L}{\partial f^{(11)}} \cdot \frac{\partial f^{(11)}}{\partial f^{(10)}}, \quad \frac{\partial L}{\partial b_4} = \frac{\partial L}{\partial f^{(10)}} \cdot \frac{\partial f^{(10)}}{\partial b_4}$$

$$\frac{\partial L}{\partial f^{(9)}} = \frac{\partial L}{\partial f^{(10)}} \cdot \frac{\partial f^{(10)}}{\partial f^{(9)}}, \quad \frac{\partial L}{\partial f^{(8)}} = \frac{\partial L}{\partial f^{(9)}} \cdot \frac{\partial f^{(9)}}{\partial f^{(8)}}$$

$$\frac{\partial L}{\partial f^{(7)}} = \frac{\partial L}{\partial f^{(8)}} \cdot \frac{\partial f^{(8)}}{\partial f^{(7)}}, \quad \frac{\partial L}{\partial b_3} = \frac{\partial L}{\partial f^{(7)}} \cdot \frac{\partial f^{(7)}}{\partial b_3}$$

$$\frac{\partial L}{\partial f^{(6)}} = \frac{\partial L}{\partial f^{(7)}} \cdot \frac{\partial f^{(7)}}{\partial f^{(6)}}, \quad \frac{\partial L}{\partial f^{(18)}} = \frac{dL(W,b)}{dL} \cdot \frac{\partial L}{\partial f^{(18)}} = \frac{\partial L}{\partial f^{(18)}}$$

$$\frac{\partial L}{\partial f^{(17)}} = \frac{\partial L}{\partial f^{(18)}} \cdot \frac{\partial f^{(18)}}{\partial f^{(17)}}, \quad \frac{\partial L}{\partial (W_4^2)} = \frac{\partial L}{\partial f^{(17)}} \cdot \frac{\partial f^{(17)}}{\partial (W_4^2)}$$

$$\frac{\partial L}{\partial W_4} = \frac{\partial L}{\partial f^{(9)}} \cdot \frac{\partial f^{(9)}}{\partial W_4} + \frac{\partial L}{\partial (W_4^2)} \cdot 2W_4, \quad \frac{\partial L}{\partial f^{(16)}} = \frac{\partial L}{\partial f^{(17)}} \cdot \frac{\partial f^{(17)}}{\partial f^{(16)}}$$

$$\frac{\partial L}{\partial (W_2^2)} = \frac{\partial L}{\partial f^{(16)}} \cdot \frac{\partial f^{(16)}}{\partial (W_2^2)}, \quad \frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial f^{(6)}} \cdot \frac{\partial f^{(6)}}{\partial W_2} + \frac{\partial L}{\partial (W_2^2)} \cdot 2W_2$$

$$\frac{\partial L}{\partial f^{(15)}} = \frac{\partial L}{\partial f^{(16)}} \cdot \frac{\partial f^{(16)}}{\partial f^{(15)}}, \quad \frac{\partial L}{\partial (W_3^2)} = \frac{\partial L}{\partial f^{(15)}} \cdot \frac{\partial f^{(15)}}{\partial (W_3^2)}$$

$$\frac{\partial L}{\partial W_3} = \frac{\partial L}{\partial f^{(3)}} \cdot \frac{\partial f^{(3)}}{\partial W_3} + \frac{\partial L}{\partial (W_3^2)} \cdot 2W_3$$

$$\frac{\partial L}{\partial (W_1^2)} = \frac{\partial L}{\partial f^{(15)}} \cdot \frac{\partial f^{(15)}}{\partial (W_1^2)}, \quad \frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial f^{(1)}} \cdot \frac{\partial f^{(1)}}{\partial W_1} + \frac{\partial L}{\partial (W_1^2)} \cdot 2W_1$$

$$\therefore \quad W_1 = W_1 - \alpha \frac{\partial L}{\partial W_1}, \quad W_2 = W_2 - \alpha \frac{\partial L}{\partial W_2}$$

$$W_3 = W_3 - \alpha \frac{\partial L}{\partial W_3}, \quad W_4 = W_4 - \alpha \frac{\partial L}{\partial W_4}$$

$$b_1 = b_1 - \alpha \frac{\partial L}{\partial b_1}, \quad b_2 = b_2 - \alpha \frac{\partial L}{\partial b_2}$$

$$b_3 = b_3 - \alpha \frac{\partial L}{\partial b_3}, \quad b_4 = b_4 - \alpha \frac{\partial L}{\partial b_4}$$

T2,

Conv_1: $(63+2\times2-5)/2+1 = 32$

$\therefore$ the shape of activation map: $32\times32\times1$

the number of parameters: $(5\times5\times3+1)\times10 = 760$

the computational cost: $[5\times5\times3+(5\times5\times3-1)+1]\times32\times32\times10 = 1536000$

Maxpool_1: $(32-2)/3+1 = 11$

$\therefore$ the shape of activation map: $11\times11\times1$

the number of parameters: 0

the computational cost: $2\times2\times11\times11\times10 = 4840$

Conv_2: $(11+2\times1-3)/2+1 = 6$

$\therefore$ the shape of activation map: $6\times6\times1$

the number of parameters: $(3\times3\times10+1)\times20 = 1820$

the computational cost: $[3\times3\times10+(3\times3\times10-1)+1]\times6\times6\times20 = 129600$

Maxpool_2: $(6+2\times1-2)/2+1 = 4$

$\therefore$ the shape of activation map: $4\times4\times1$

the number of parameters: 0

the computational cost: $2\times2\times4\times4\times20 = 1280$

total number of parameters: 2580

total computational cost: 1671720

T3、

Gini index:

$$G_A = 1-\left[\left(\frac{5}{18}\right)^2+\left(\frac{5}{18}\right)^2+\left(\frac{8}{18}\right)^2\right] = 0.648$$

$$G_{B_1} = 1-\left[\left(\frac{1}{8}\right)^2+\left(\frac{2}{8}\right)^2+\left(\frac{5}{8}\right)^2\right] = 0.531$$

$$G_{B_2} = 1-\left[\left(\frac{4}{10}\right)^2+\left(\frac{6}{10}\right)^2\right] = 0.48$$

$$G_{C_1} = 1-\left[\left(\frac{1}{6}\right)^2+\left(\frac{5}{6}\right)^2\right] = 0.278$$

$$G_{C_2} = 1-1^2 = 0$$

Entropy:

$$E_A = -\frac{5}{18}\log_2\frac{5}{18} - \frac{5}{18}\log_2\frac{5}{18} - \frac{8}{18}\log_2\frac{8}{18} = 1.547$$

$$E_{B_1} = -\frac{1}{8}\log_2\frac{1}{8} - \frac{2}{8}\log_2\frac{2}{8} - \frac{5}{8}\log_2\frac{5}{8} = 1.299$$

$$E_{B_2} = -\frac{4}{10}\log_2\frac{4}{10} - \frac{6}{10}\log_2\frac{6}{10} = 0.971$$

$$E_{C_1} = -\frac{1}{6}\log_2\frac{1}{6} - \frac{5}{6}\log_2\frac{5}{6} = 0.65$$

$$E_{C_2} = -1\cdot\log_2(1) = 0$$

Misclassification error:

$$M_A = 1-\frac{8}{18} = \frac{5}{9}$$

$$M_{B_1} = 1-\frac{5}{8} = \frac{3}{8}$$

$$M_{B_2} = 1-\frac{6}{10} = \frac{2}{5}$$

$$M_{C_1} = 1-\frac{5}{6} = \frac{1}{6}$$

$$M_{C_2} = 1-1 = 0$$

T4.

(a) empirical MSE:

$$\widehat{MSE} = \frac{1}{10}\times\left[(6-7)^2+(8-7)^2+(9-7)^2+(5-7)^2+(10-7)^2+(5-7)^2 + (4-7)^2+(8-7)^2+(9-7)^2+(3-7)^2\right] = 5.3$$

$$\overline{h}_D(X=3) = \frac{1}{10}\times(6+8+9+5+10+5+4+8+9+3) = 6.7$$

$$Bias^2 = \left[\overline{h}_D(X=3) - t(X=3)\right]^2 = (6.7-6.7)^2 = 0$$

$$Variance = \frac{1}{10}\sum_{i}^{10}(h_{D_i}(X=3)-\overline{h}_D(X=3))^2$$

$$= \frac{1}{10}\times\left[(6-6.7)^2+(8-6.7)^2+(9-6.7)^2+(5-6.7)^2+(10-6.7)^2+(5-6.7)^2 + (4-6.7)^2+(8-6.7)^2+(9-6.7)^2+(3-6.7)^2\right] = 5.21$$

(b) reason ①:

in fact, the theoretical equation is:

$$E_{(x,y),D}\left[(h_D(x)-y)^2\right] = E_{(x,y),D}\left[(h_D(x)-\overline{h}(x))^2\right] + E_{(x,y),D}\left[(\overline{h}(x)-t(x))^2\right] + E_{x,y}\left[(t(x)-y)^2\right]$$

and we need to use integral and the probability distribution to compute each expectation term. But in reality, we don't know the distribution and our sample is limited within finite data point.

reason ②:

here we suppose the error $\varepsilon \in N(0,\sigma^2)$ and $\sigma^2=0.5$ but in fact, the error may not satisfy the normal distribution we have supposed.

reason ③:

even in the discrete case, we have:

$$\widehat{MSE}(x,y) = \frac{1}{10}\sum_{i=1}^{10}(h_{D_i}(x) - y)^2$$

$$= \frac{1}{10}\sum_{i=1}^{10}(h_{D_i}(x) - \bar{h}(x) + \bar{h}(x) - y)^2$$

$$= \frac{1}{10}\sum_{i=1}^{10}[2(h_{D_i}(x) - \bar{h}(x))(\bar{h}(x) - y)] + \frac{1}{10}\sum_{i=1}^{10}[h_{D_i}(x) - \bar{h}(x)]^2$$

$$+ \frac{1}{10}\sum_{i=1}^{10}[\bar{h}(x) - y]^2 = \frac{1}{10}\sum_{i=1}^{10}[h_{D_i}(x) - \bar{h}(x)]^2 + \frac{1}{10}\sum_{i=1}^{10}[\bar{h}(x) - y]^2$$

$$= \frac{1}{10}\sum_{i=1}^{10}[h_{D_i}(x) - \bar{h}(x)]^2 + \frac{1}{10}\sum_{i=1}^{10}[\bar{h}(x) - t(x) + t(x) - y]^2$$

$$= \frac{1}{10}\sum_{i=1}^{10}[h_{D_i}(x) - \bar{h}(x)]^2 + \frac{1}{10}\sum_{i=1}^{10}(h(x) - t(x))^2 + \frac{1}{10}\sum_{i=1}^{10}[t(x) - y]^2$$

$$+ \frac{1}{10}\sum_{i=1}^{10}[2(\bar{h}(x) - t(x))(t(x) - y)]$$

$$= \frac{1}{10}\sum_{i=1}^{10}[h_{D_i}(x) - \bar{h}(x)]^2 + \frac{1}{10}\sum_{i=1}^{10}(h(x) - t(x))^2 + \frac{1}{10}\sum_{i=1}^{10}[t(x) - y]^2$$

$$= \text{bias}^2 + \text{Variance} + \frac{1}{10}\sum_{i=1}^{10}[t(x) - y]^2$$

and $\sigma^2 \neq \frac{1}{10}\sum_{i=1}^{10}[t(x) - y]^2$

in fact, it is similar to limited sample reason

T5、

$$\sigma(a)=\frac{1}{1+e^{-a}}$$

$$\tanh(a)=\frac{e^{a}-e^{-a}}{e^{a}+e^{-a}}$$

$$\therefore \tanh(a)=2\sigma(2a)-1$$

$$y_k(x,w)=\sigma\left(\sum_{j=1}^{M}w_{kj}^{(2)}\sigma\left(\sum_{i=1}^{D}w_{ji}^{(1)}x_i+w_{j0}^{(1)}\right)+w_{k0}^{(2)}\right)$$

$$\hat{y}_k(x,\hat{w})=\sigma\left(\sum_{j=1}^{M}\hat{w}_{kj}^{(2)}\left[2\sigma\left(2\left(\sum_{i=1}^{D}\hat{w}_{ji}^{(1)}x_i+\hat{w}_{j0}^{(1)}\right)\right)-1\right]+\hat{w}_{k0}^{(2)}\right)$$

$$=\sigma\left(\sum_{j=1}^{M}2\hat{w}_{kj}^{(2)}\sigma\left(\sum_{i=1}^{D}2\hat{w}_{ji}^{(1)}x_i+2\hat{w}_{j0}^{(1)}\right)-\sum_{j=1}^{M}\hat{w}_{kj}^{(2)}+\hat{w}_{k0}^{(2)}\right)$$

we let $2\hat{w}_{kj}^{(2)}=w_{kj}^{(2)}$

$$2\hat{w}_{ji}^{(1)}=w_{ji}^{(1)}$$

$$2\hat{w}_{j0}^{(1)}=w_{j0}^{(1)}$$

$$\hat{w}_{k0}^{(2)}-\sum_{j=1}^{M}\hat{w}_{kj}^{(2)}=w_{k0}^{(2)}$$

then $y_k(x,w)=\hat{y}_k(x,\hat{w})$, and the above is linear transformation between w and $\hat{w}$.

$\therefore$ Q.E.D.

2. Programming Question

2.1 Question restatement

In the programming problem, we need to solve a regression task using decision tree, bagging, and random forests with **sklearn** in python. First, we need to analyze the raw data. And then, for each method, we need to consider the train error and test error with different hyper-parameters. What's more, for decision tree case, we need to plot the tree structure. For random forests case, we need to do a bias-variance analysis.

2.2 Python Implementation

For data statistics, I write a function like the following.

```
def data_statistics(df):
    target = df.iloc[:,0].values
    features = np.transpose(df.iloc[:,1:].values)
    plt.hist(target,bins=20)
    plt.xlabel('Sales')
    plt.ylabel('frequency')
    plt.show()
    for i in range(10):
        plt.hist(features[i],bins=20)
        if i == 0:
            plt.xlabel('CompPrice')
        if i == 1:
            plt.xlabel('Income')
        if i == 2:
            plt.xlabel('Advertising')
        if i == 3:
            plt.xlabel('Population')
        if i == 4:
            plt.xlabel('Price')
        if i == 5:
            plt.xlabel('ShelveLoc')
        if i == 6:
            plt.xlabel('Age')
        if i == 7:
            plt.xlabel('Education')
        if i == 8:
            plt.xlabel('Urban')
        if i == 9:
            plt.xlabel('US')
        plt.ylabel('frequency')
        plt.show()
```

For regression task, I construct a class called DecisionTree like the following.

```
class DecisionTree(object):
    def __init__(self,dataframe):
        self.model = None
        self.dataframe = dataframe
        self.X_train = None
        self.Y_train = None
        self.Y_train_pred = None
        self.X_test = None
        self.Y_test = None
        self.Y_test_pred = None
        self.mse_train = None
        self.mse_test = None
        self.max_depth = None
        self.min_node_size = None
        self.processing()

    def processing(self):
        # omit details

    def fit_dt(self,max_depth=None,min_node_size=2):
        # omit details

    def fit_bagging(self,number_of_trees,max_depth=None,min_node_size=2):
        # omit details

    def fit_random_forests(self,m,number_of_trees,max_depth=None,min_node_size=2):
        # omit details

    def plot_tree(self):
        # omit details

    def bias_variance_analysis(self,max_depth=None,min_node_size=2):
        # omit details
```

precessing is used to do the data processing. **fit_dt** is used to train a decision tree. **fit_bagging** is used to to train a bagging of trees. **fit_random_forests** is used to train a random forest. **plot_tree** is used to plot the structure of a decision tree. **bias_variance_analysis** is to used to do the bias-variance analysis.

2.3 Data processing

Carseats contains 400 data points, and I simply set the first 300 rows as the training set, and the remaining 100 raws as the testing set. And the most important thing in this part is the operation to change the form of discrete features. For 'Urban' and 'US', I treat them as dummy variables which value is 0(No) or 1(Yes). For 'ShelveLoc' feature, since it has three different values, I use one-hot method to change it to three different dummy variables with 0,1 values. The code is like the following.

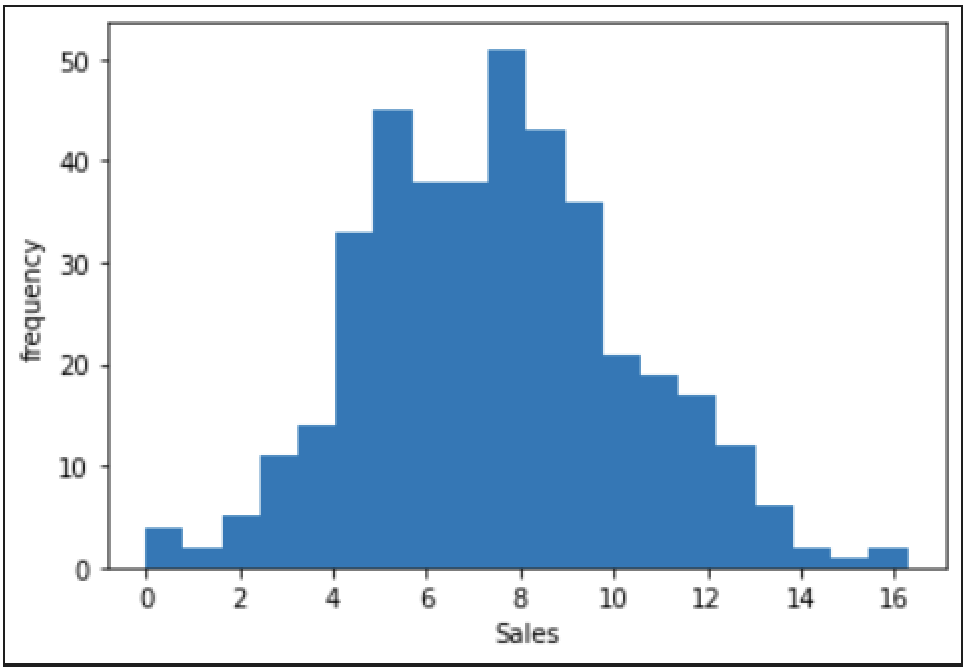
```
def processing(self):
    self.dataframe["Urban"] = self.dataframe["Urban"].replace("No",0)
    self.dataframe["Urban"] = self.dataframe["Urban"].replace("Yes",1)
    self.dataframe["US"] = self.dataframe["US"].replace("No",0)
```

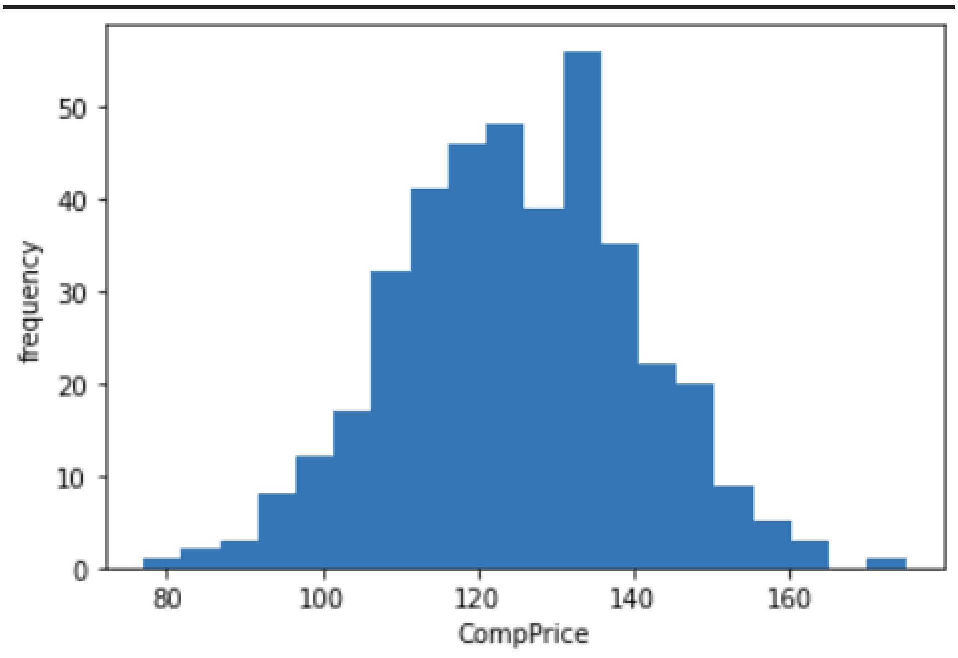
```
        self.dataframe["US"] = self.dataframe["US"].replace("Yes",1)
        self.dataframe = self.dataframe.join(pd.get_dummies(self.dataframe.ShelveLoc))
        self.dataframe = self.dataframe.drop(columns=['ShelveLoc'])
        self.Y_train = self.dataframe.iloc[:300,0].values
        self.Y_test = self.dataframe.iloc[301:,0].values
        self.X_train = self.dataframe.iloc[:300,1:].values
        self.X_test = self.dataframe.iloc[301:,1:].values
```

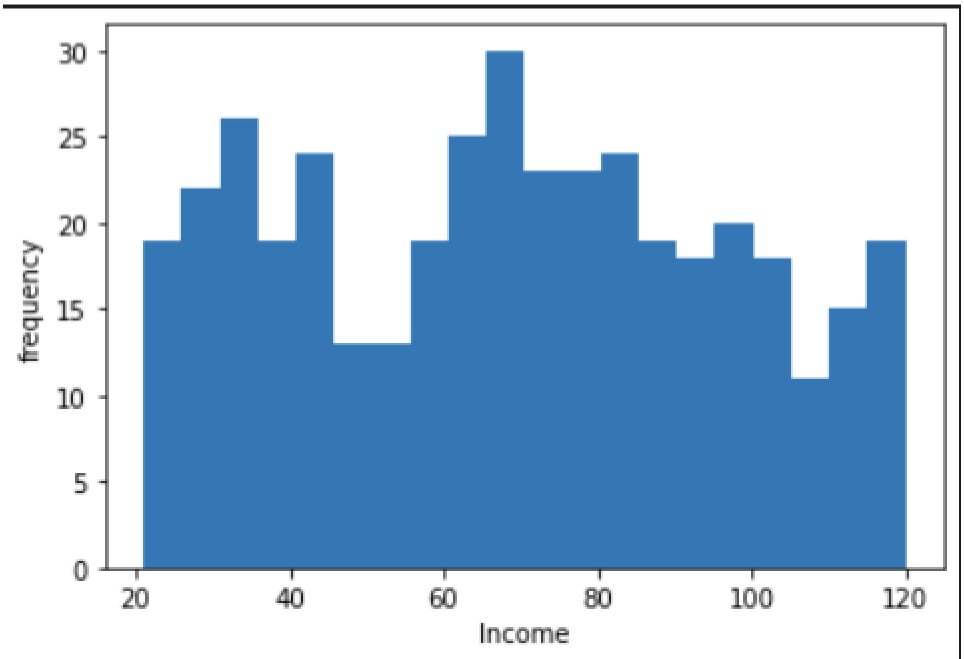
2.4 Data statistics

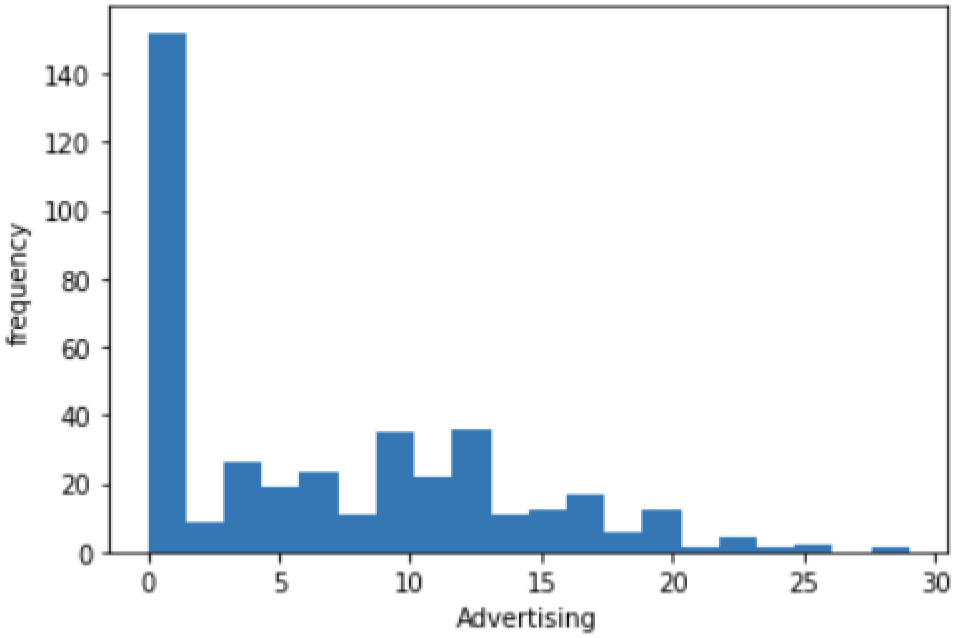
I plot the frequency of Sales and each feature.

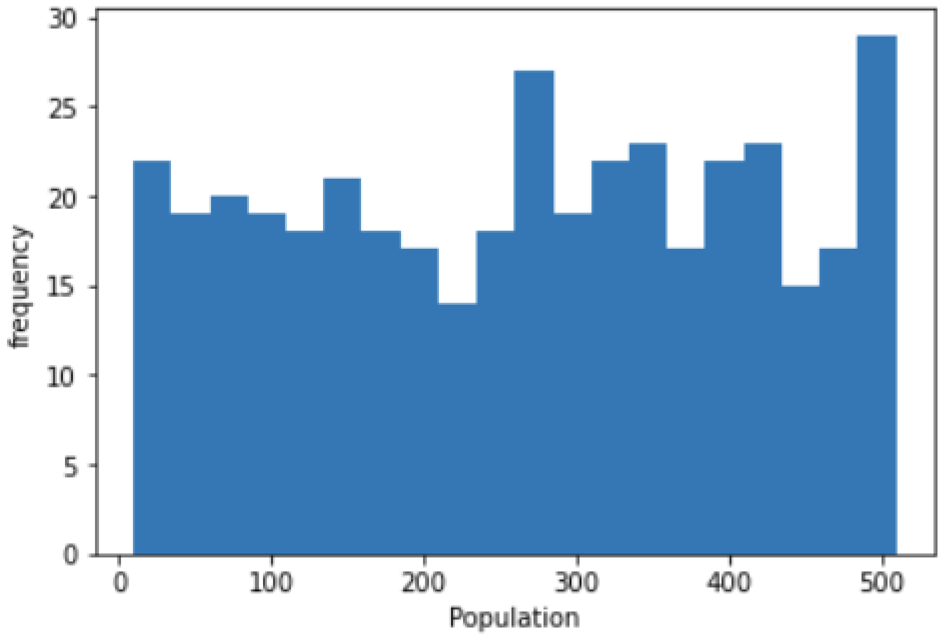
I find that Sales has more intermediate samples and less endpoint samples. CompPrice and Price also has similar distribution compared with Sales. The distribution of Income is relatively even, and so do Population, Age and Education. The left endpoint of Advertising distributes more samples. For ShelveLoc, the frequency of Bad and Good are similar, nearly half of that of Medium. And the distribution of Urban and US are similar, the frequency of No is about half of that of Yes.

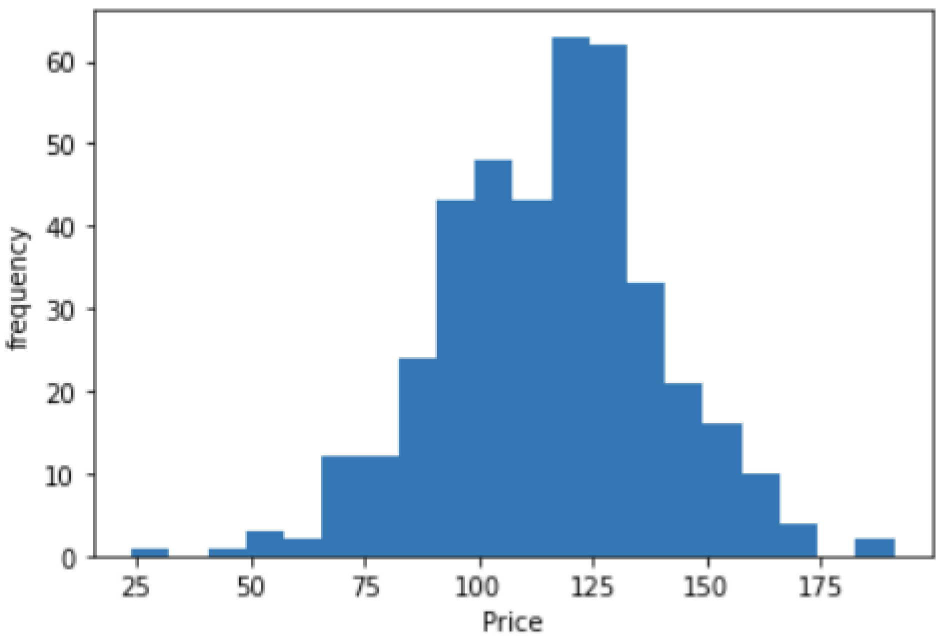

frequency
Sales

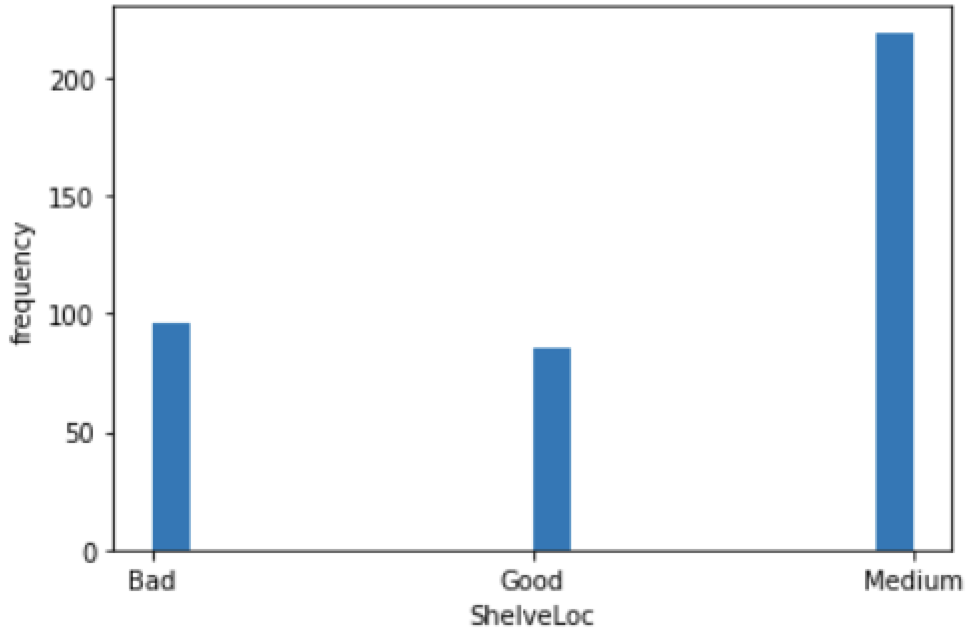

frequency
CompPrice

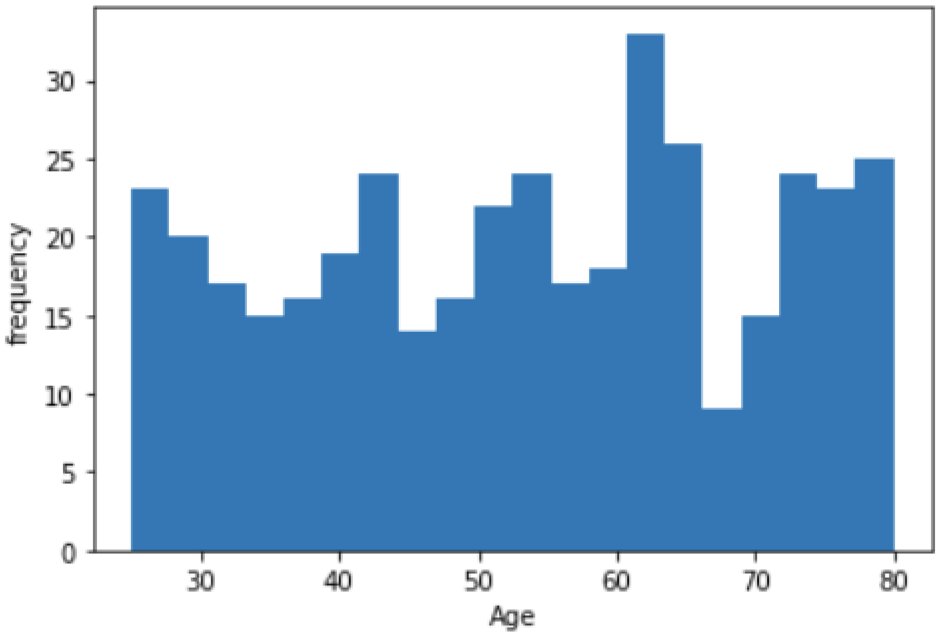

frequency
Income

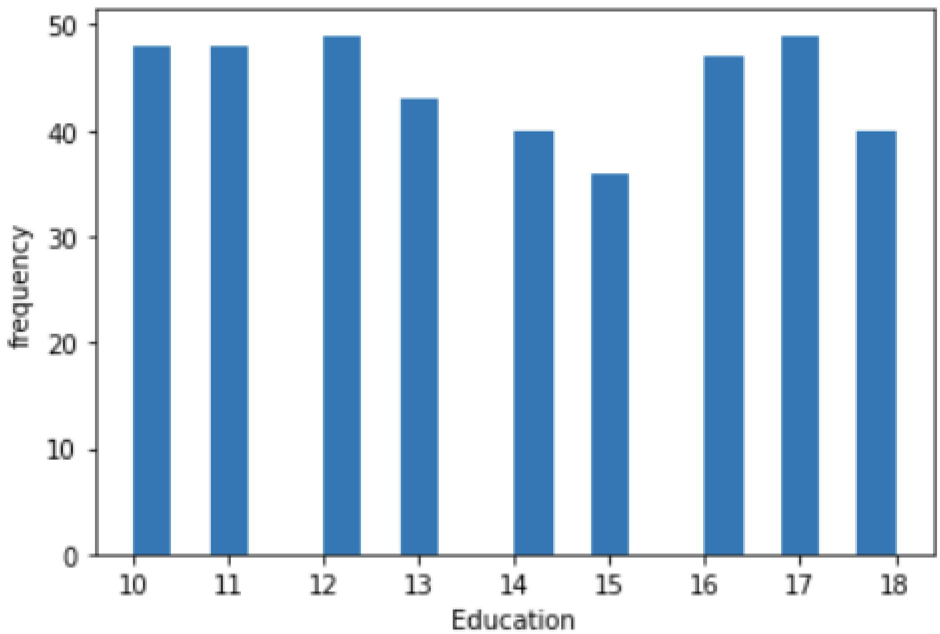

frequency
Advertising
0
20
40
60
80
100
120
140
5
10
15
25
30

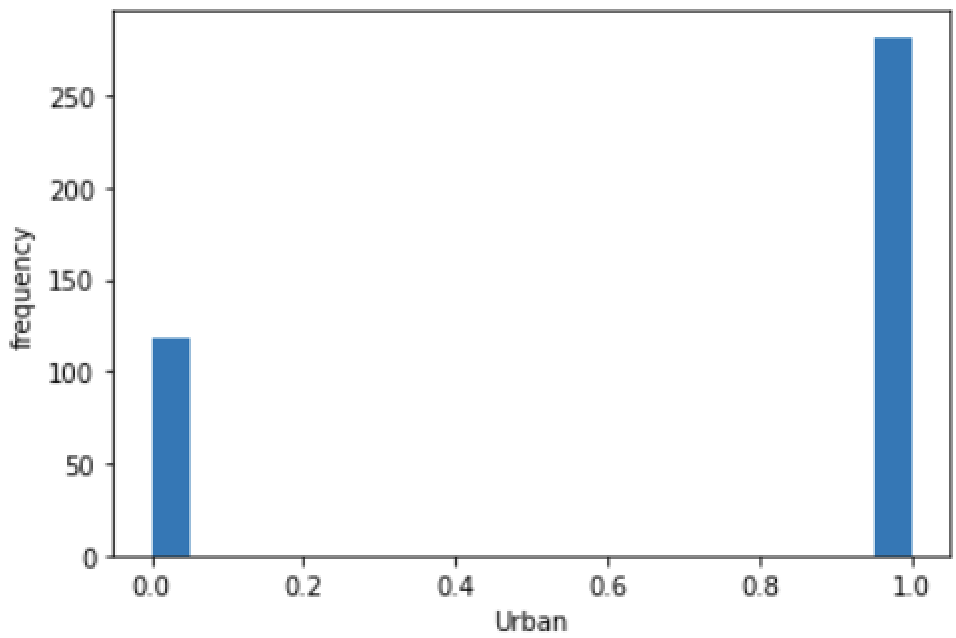

frequency
Population
0
5
10
15
20
25
30
100
200
300
400
500

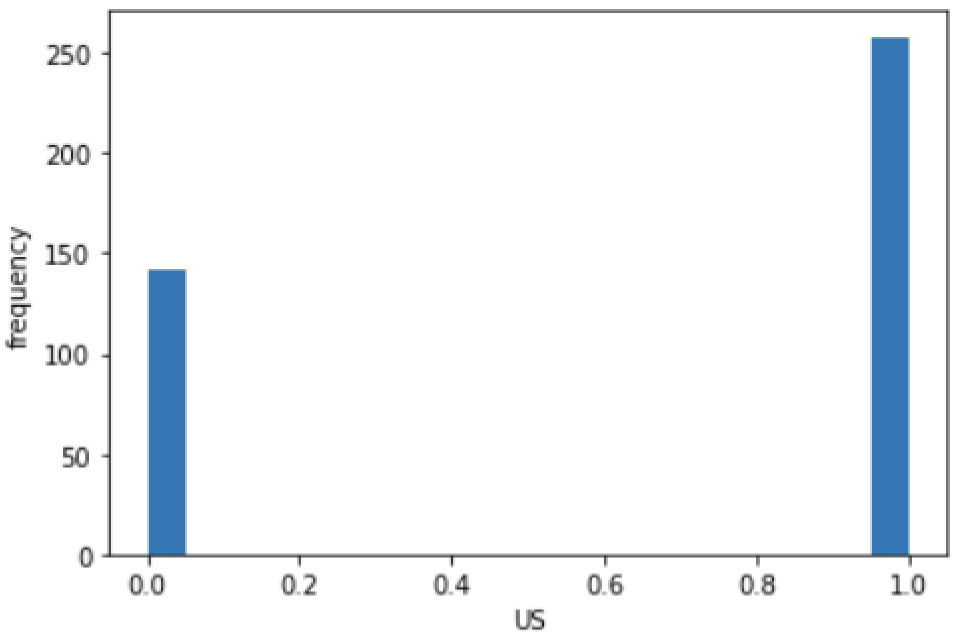

frequency
Price
0
10
20
30
40
50
60
25
50
75
100
125
150
175


frequency
200
150
100
50
0
Bad
Good
Medium
ShelveLoc


frequency
30
25
20
15
10
5
0
30
40
50
60
70
80
Age


frequency
50
40
30
20
10
0
10
11
12
13
14
15
16
17
18
Education





2.5 Decision tree

Maximum depth: [3,4,5]

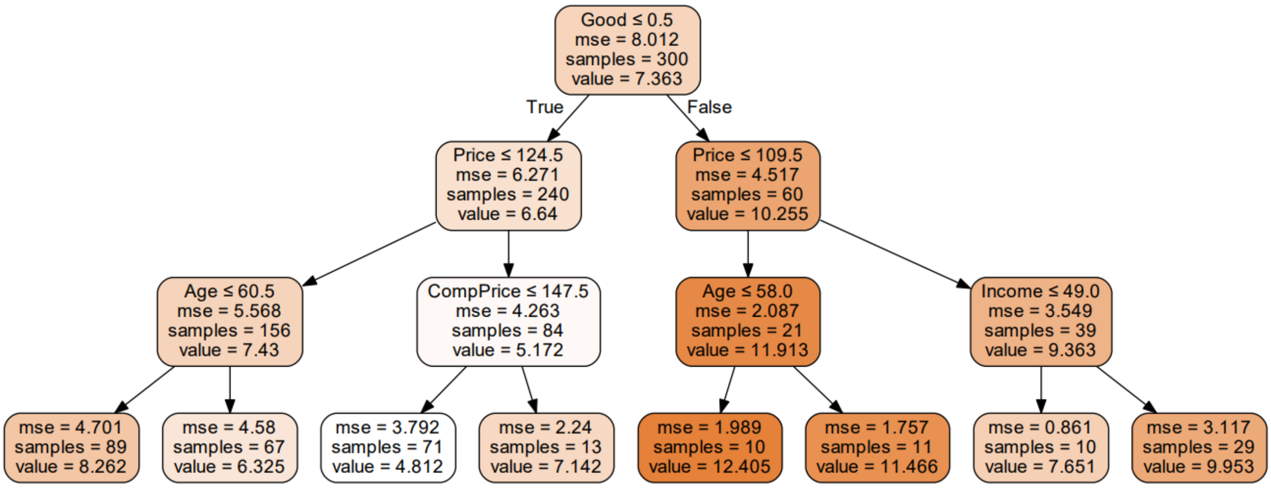
Least node size: [10,15,20]

The train error and test error is like the following:

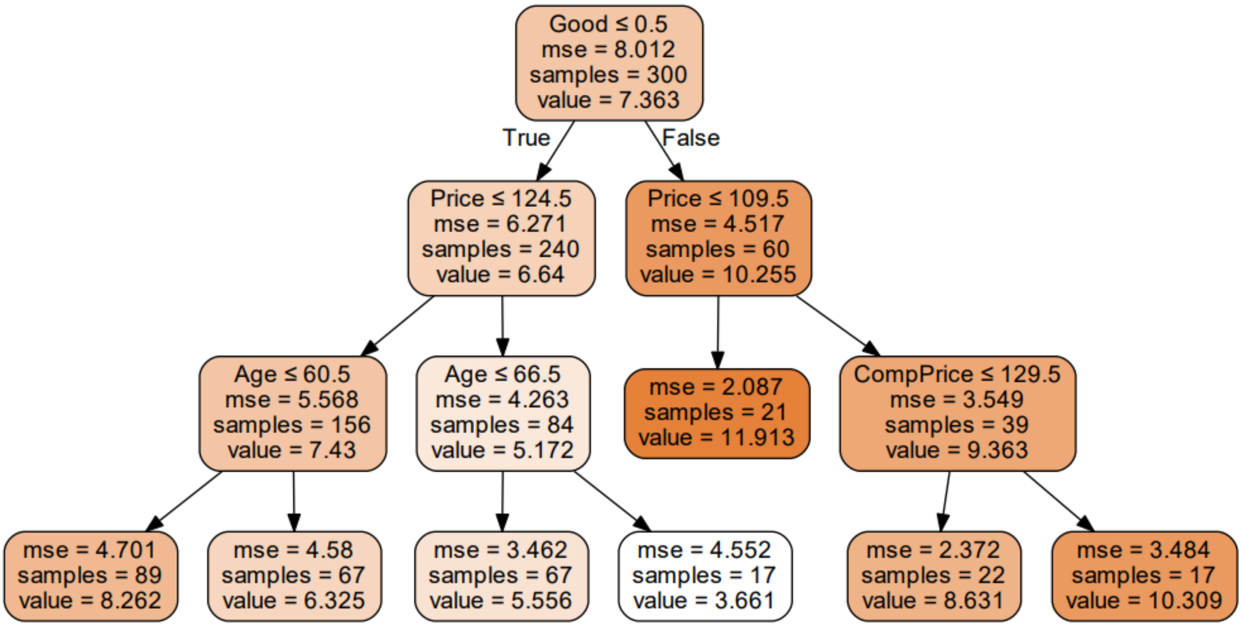
```
Maximum depth=3;Least node sizes=10: train error=3.872804
Maximum depth=3;Least node sizes=10: test error=5.706664
Maximum depth=3;Least node sizes=15: train error=3.966136
Maximum depth=3;Least node sizes=15: test error=5.749041
Maximum depth=3;Least node sizes=20: train error=4.076576
Maximum depth=3;Least node sizes=20: test error=5.412308
Maximum depth=4;Least node sizes=10: train error=3.175644
Maximum depth=4;Least node sizes=10: test error=5.472547
Maximum depth=4;Least node sizes=15: train error=3.381846
Maximum depth=4;Least node sizes=15: test error=5.231955
Maximum depth=4;Least node sizes=20: train error=3.489454
Maximum depth=4;Least node sizes=20: test error=4.936841
Maximum depth=5;Least node sizes=10: train error=2.475642
Maximum depth=5;Least node sizes=10: test error=5.171406
Maximum depth=5;Least node sizes=15: train error=2.945208
Maximum depth=5;Least node sizes=15: test error=5.190623
Maximum depth=5;Least node sizes=20: train error=3.196852
Maximum depth=5;Least node sizes=20: test error=4.993507
```

The structure of the above trees:

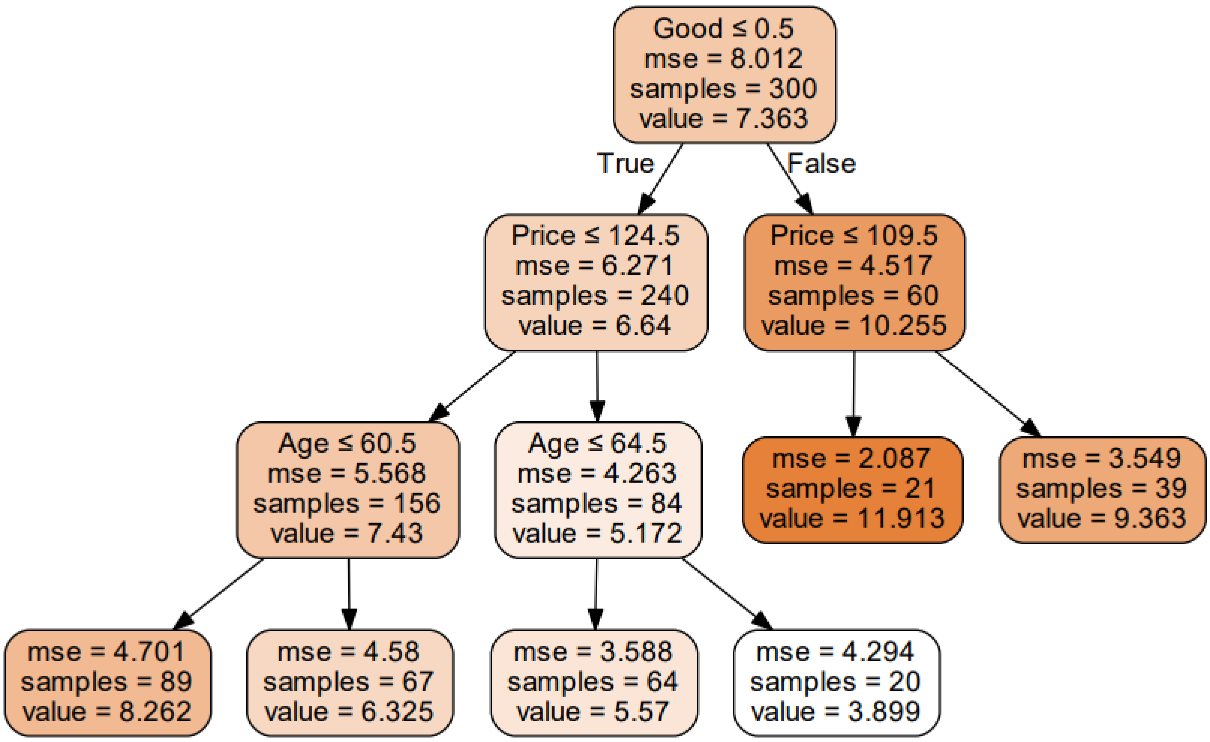
Maximum depth = 3, Least node size = 10:



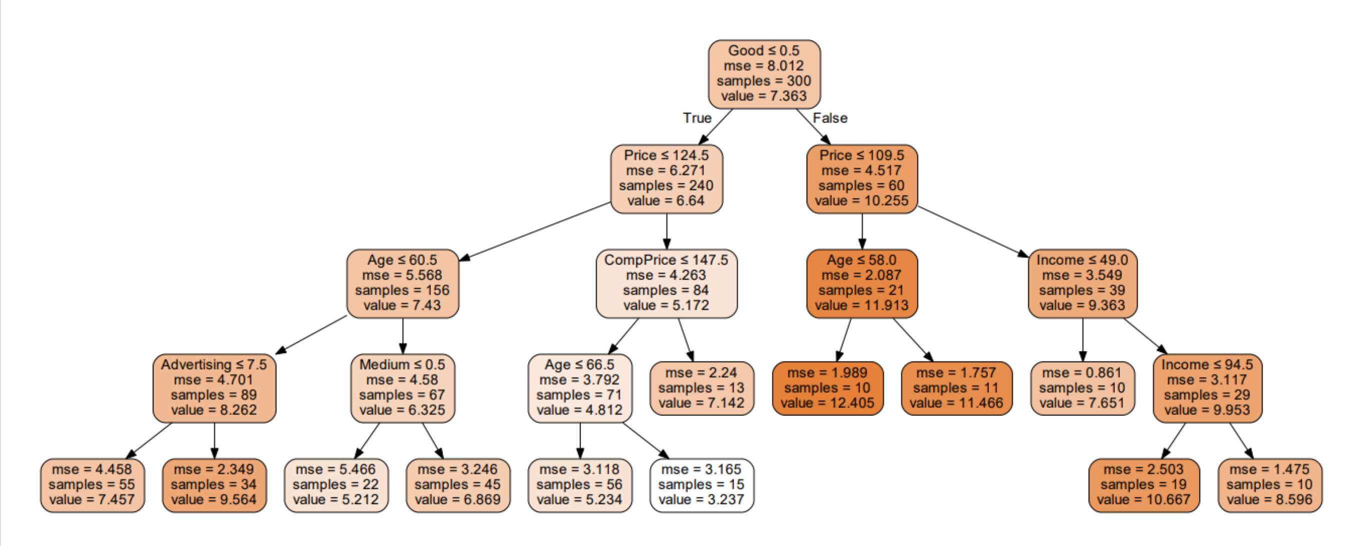
Maximum depth = 3, Least node size = 15:



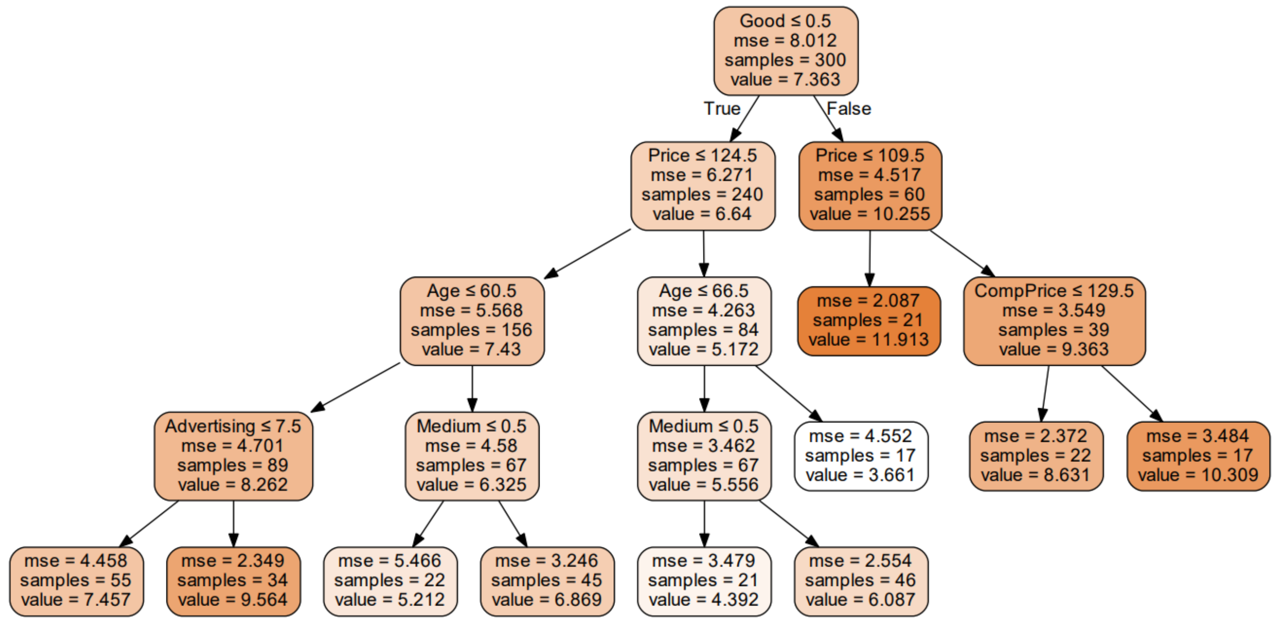
Maximum depth = 3, Least node size = 20:



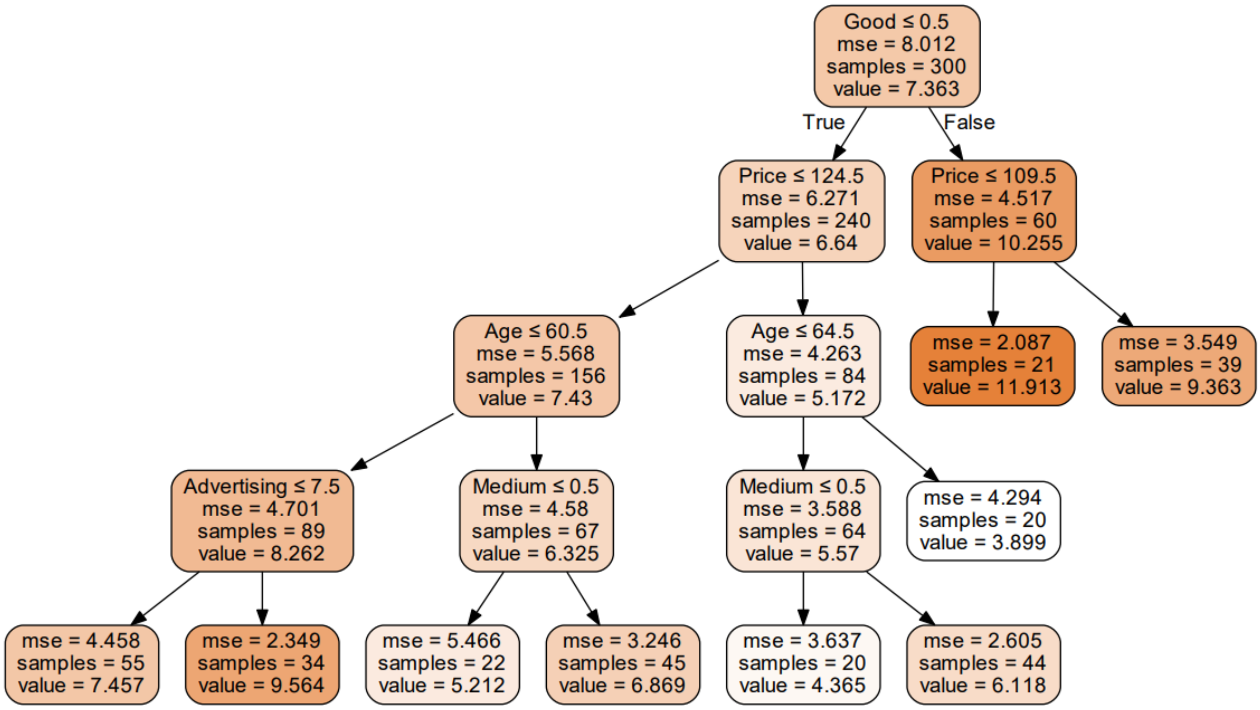
Maximum depth = 4, Least node size = 10:



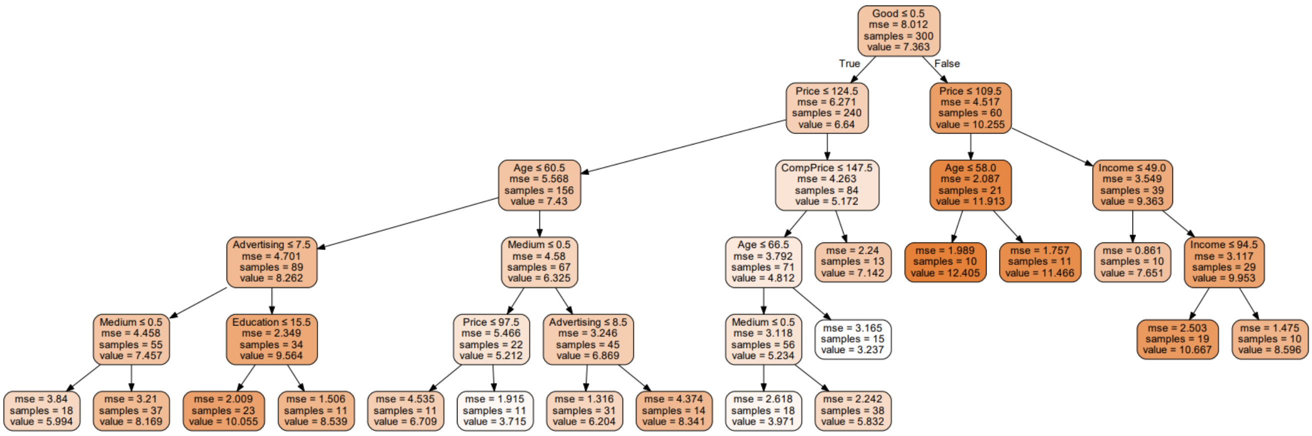
Maximum depth = 4, Least node size = 15:



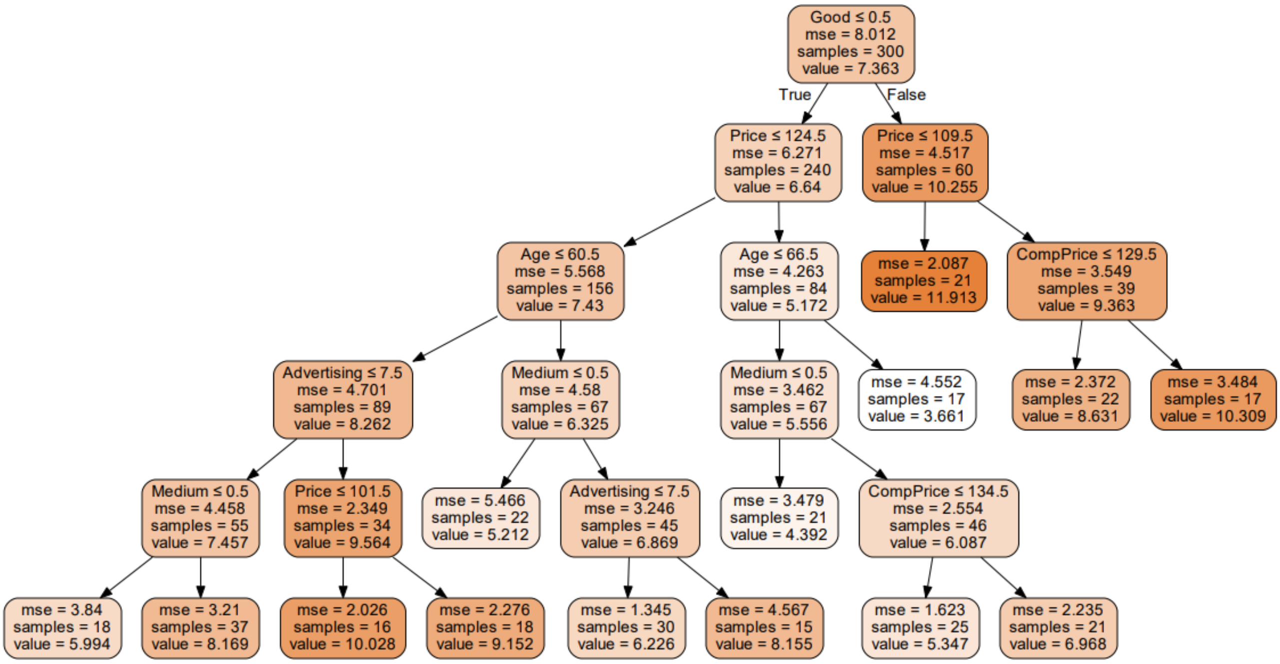
Maximum depth = 4, Least node size = 20:



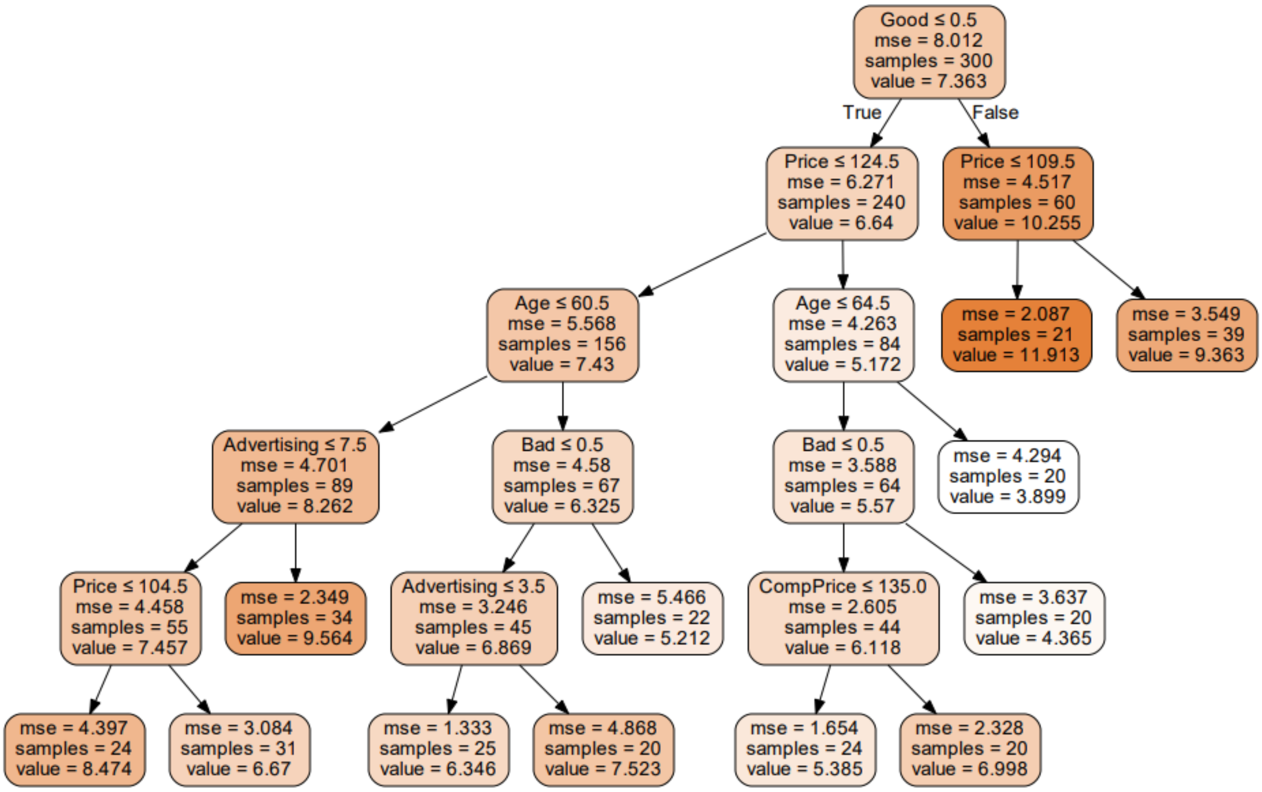
Maximum depth = 5, Least node size = 10:



Maximum depth = 5, Least node size = 15:



Maximum depth = 5, Least node size = 20:



2.6 Bagging of trees

Number of trees: [10,30,50,70,90,110,130,150]

Maximum depth: [5,6,7]

The train error and test error is like the following:

```
Number of trees=10;Maximum depth=5: train error=1.539127
Number of trees=10;Maximum depth=5: test error=3.648737
Number of trees=10;Maximum depth=6: train error=1.105240
Number of trees=10;Maximum depth=6: test error=3.530226
Number of trees=10;Maximum depth=7: train error=0.934483
Number of trees=10;Maximum depth=7: test error=3.469638
Number of trees=30;Maximum depth=5: train error=1.439148
Number of trees=30;Maximum depth=5: test error=3.237845
Number of trees=30;Maximum depth=6: train error=0.999964
Number of trees=30;Maximum depth=6: test error=3.155788
Number of trees=30;Maximum depth=7: train error=0.769961
Number of trees=30;Maximum depth=7: test error=3.114112
Number of trees=50;Maximum depth=5: train error=1.405026
Number of trees=50;Maximum depth=5: test error=3.153136
Number of trees=50;Maximum depth=6: train error=0.967992
Number of trees=50;Maximum depth=6: test error=3.079383
Number of trees=50;Maximum depth=7: train error=0.732481
Number of trees=50;Maximum depth=7: test error=2.973205
Number of trees=70;Maximum depth=5: train error=1.393394
Number of trees=70;Maximum depth=5: test error=3.038658
Number of trees=70;Maximum depth=6: train error=0.951836
Number of trees=70;Maximum depth=6: test error=2.965632
Number of trees=70;Maximum depth=7: train error=0.716784
Number of trees=70;Maximum depth=7: test error=2.865519
Number of trees=90;Maximum depth=5: train error=1.410708
Number of trees=90;Maximum depth=5: test error=3.041584
Number of trees=90;Maximum depth=6: train error=0.970692
Number of trees=90;Maximum depth=6: test error=2.923586
Number of trees=90;Maximum depth=7: train error=0.732928
Number of trees=90;Maximum depth=7: test error=2.825993
```

```
Number of trees=110;Maximum depth=5: train error=1.415131
Number of trees=110;Maximum depth=5: test error=3.054390
Number of trees=110;Maximum depth=6: train error=0.979377
Number of trees=110;Maximum depth=6: test error=2.923471
Number of trees=110;Maximum depth=7: train error=0.739093
Number of trees=110;Maximum depth=7: test error=2.819307
Number of trees=130;Maximum depth=5: train error=1.416075
Number of trees=130;Maximum depth=5: test error=3.058256
Number of trees=130;Maximum depth=6: train error=0.975695
Number of trees=130;Maximum depth=6: test error=2.951570
Number of trees=130;Maximum depth=7: train error=0.738263
Number of trees=130;Maximum depth=7: test error=2.855243
Number of trees=150;Maximum depth=5: train error=1.407957
Number of trees=150;Maximum depth=5: test error=3.032892
Number of trees=150;Maximum depth=6: train error=0.970889
Number of trees=150;Maximum depth=6: test error=2.942360
Number of trees=150;Maximum depth=7: train error=0.734949
Number of trees=150;Maximum depth=7: test error=2.851264
```

2.7 Random forests

Max features: [3,4,5]

Number of trees: [10,30,50,70,90,110,130,150]

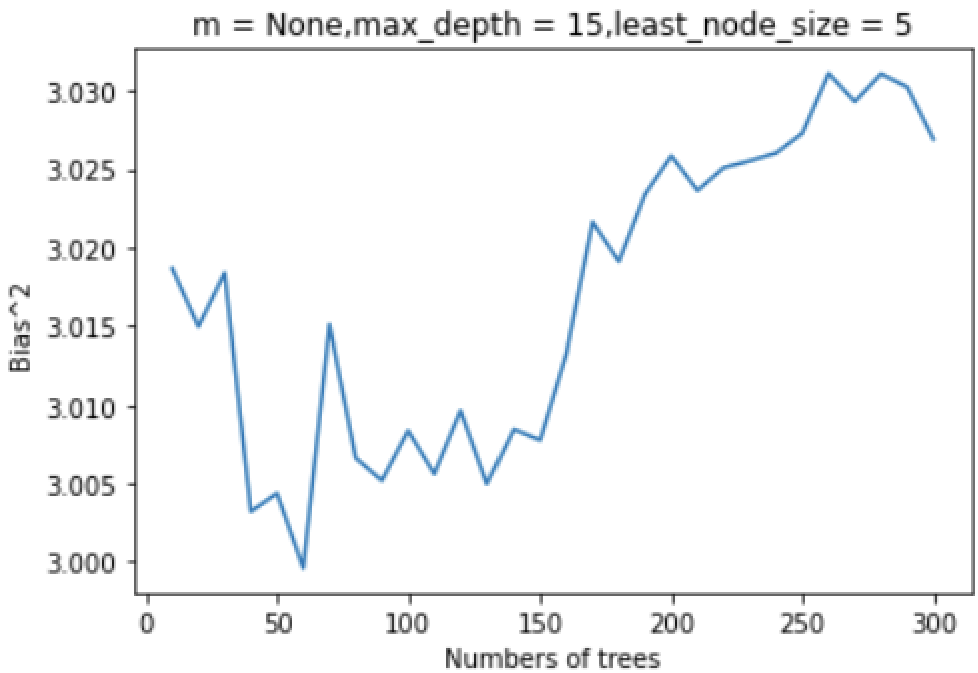
The train error and test error is like the following:

```
Max features=3;Number of trees=10: train error=1.104304
Max features=3;Number of trees=10: test error=3.493380
Max features=3;Number of trees=30: train error=0.902824
Max features=3;Number of trees=30: test error=2.902448
Max features=3;Number of trees=50: train error=0.868971
Max features=3;Number of trees=50: test error=2.884367
Max features=3;Number of trees=70: train error=0.867453
Max features=3;Number of trees=70: test error=2.889716
Max features=3;Number of trees=90: train error=0.880094
Max features=3;Number of trees=90: test error=2.911878
Max features=3;Number of trees=110: train error=0.882999
Max features=3;Number of trees=110: test error=2.897184
Max features=3;Number of trees=130: train error=0.878182
Max features=3;Number of trees=130: test error=2.908986
Max features=3;Number of trees=150: train error=0.872342
Max features=3;Number of trees=150: test error=2.911225
Max features=4;Number of trees=10: train error=0.938075
Max features=4;Number of trees=10: test error=2.909800
Max features=4;Number of trees=30: train error=0.761591
Max features=4;Number of trees=30: test error=2.787616
Max features=4;Number of trees=50: train error=0.730728
Max features=4;Number of trees=50: test error=2.714402
Max features=4;Number of trees=70: train error=0.741848
Max features=4;Number of trees=70: test error=2.728002
Max features=4;Number of trees=90: train error=0.745758
Max features=4;Number of trees=90: test error=2.786083
Max features=4;Number of trees=110: train error=0.740703
Max features=4;Number of trees=110: test error=2.750437
Max features=4;Number of trees=130: train error=0.737434
Max features=4;Number of trees=130: test error=2.741700
Max features=4;Number of trees=150: train error=0.732035
Max features=4;Number of trees=150: test error=2.730904
```

```
Max features=5;Number of trees=10: train error=0.862671
Max features=5;Number of trees=10: test error=3.193984
Max features=5;Number of trees=30: train error=0.709992
Max features=5;Number of trees=30: test error=2.816000
Max features=5;Number of trees=50: train error=0.683940
Max features=5;Number of trees=50: test error=2.772287
Max features=5;Number of trees=70: train error=0.662374
Max features=5;Number of trees=70: test error=2.728520
Max features=5;Number of trees=90: train error=0.659771
Max features=5;Number of trees=90: test error=2.687156
Max features=5;Number of trees=110: train error=0.666637
Max features=5;Number of trees=110: test error=2.708928
Max features=5;Number of trees=130: train error=0.669362
Max features=5;Number of trees=130: test error=2.715019
Max features=5;Number of trees=150: train error=0.662569
Max features=5;Number of trees=150: test error=2.696149
```

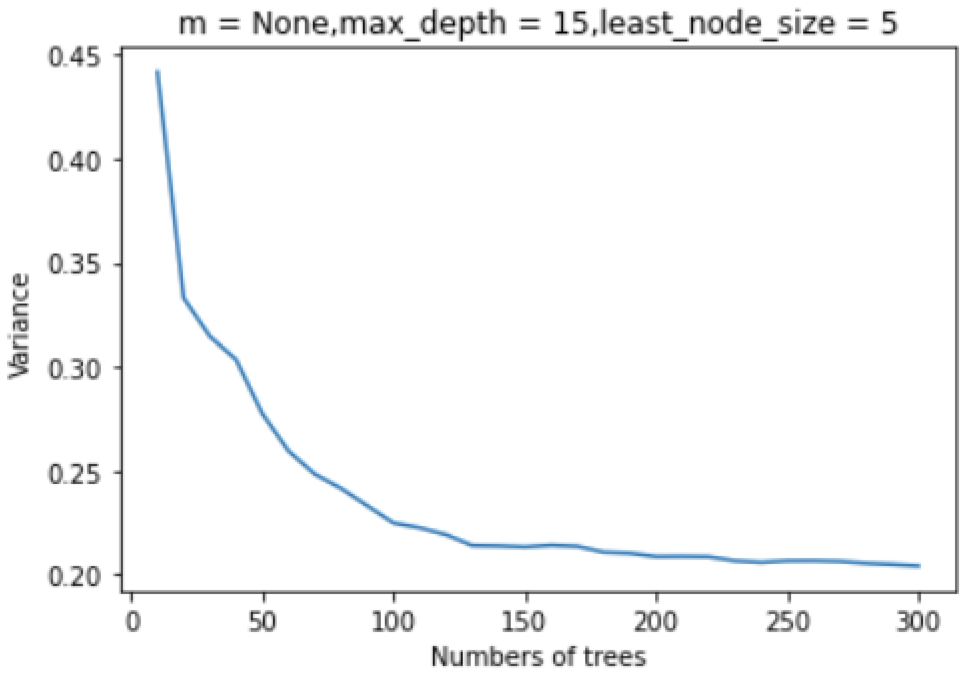
2.8 Bias-Variance analysis for random forests

Bias^2



The number of trees and Bias^2 don't have obvious relationship. When the number of trees is increasing, we can't guarantee the change of Bias^2, as different trees are independent and the overall model complexity is not increased.

Variance



When the number of trees is increasing, the variances is reduced. And when number of trees is not large, the change of variance is significant. When the number of trees become large, the reduce of variance become gently.