Model variability and Bias-Variance Trade-off in Decision Tree regressor

January 21, 2021

[1]:
```python
#import the libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

[2]:
```python
#import the datasets
df=pd.read_csv('insurance.csv')
```

[3]:
```python
df.head()
```

[3]:

	age	sex	bmi	children	smoker	region	expenses
0	19	female	27.9	0	yes	southwest	16884.92
1	18	male	33.8	1	no	southeast	1725.55
2	28	male	33.0	3	no	southeast	4449.46
3	33	male	22.7	0	no	northwest	21984.47
4	32	male	28.9	0	no	northwest	3866.86

[4]:
```python
df.tail()
```

[4]:

	age	sex	bmi	children	smoker	region	expenses
1333	50	male	31.0	3	no	northwest	10600.55
1334	18	female	31.9	0	no	northeast	2205.98
1335	18	female	36.9	0	no	southeast	1629.83
1336	21	female	25.8	0	no	southwest	2007.95
1337	61	female	29.1	0	yes	northwest	29141.36

[5]:
```python
df.describe(include='all')
```

[5]:

	age	sex	bmi	children	smoker	region	\
count	1338.000000	1338	1338.000000	1338.000000	1338	1338	
unique	NaN	2	NaN	NaN	2	4	
top	NaN	male	NaN	NaN	no	southeast	
freq	NaN	676	NaN	NaN	1064	364	
mean	39.207025	NaN	30.665471	1.094918	NaN	NaN	

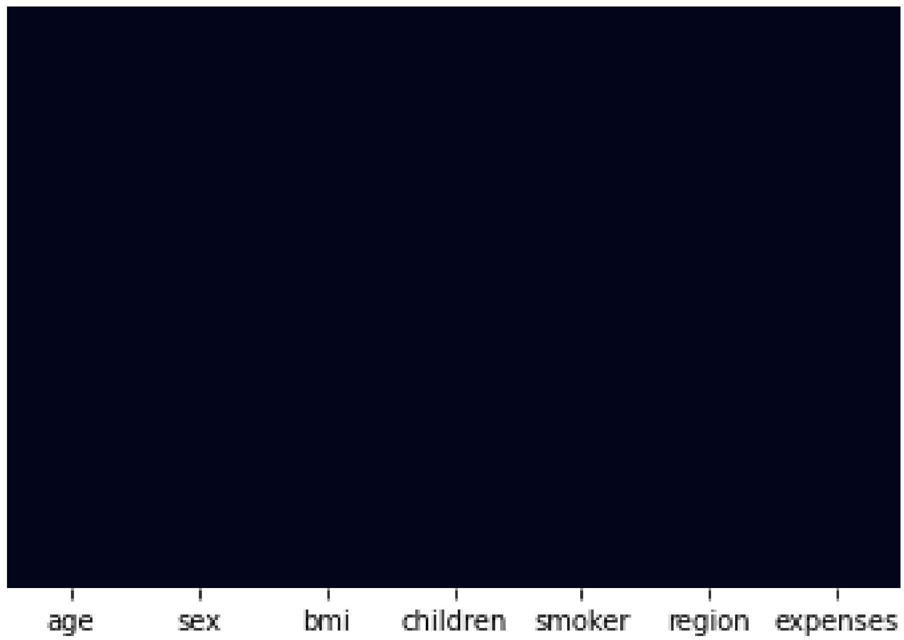
```
std       14.049960   NaN    6.098382    1.205493    NaN       NaN
min       18.000000   NaN   16.000000    0.000000    NaN       NaN
25%       27.000000   NaN   26.300000    0.000000    NaN       NaN
50%       39.000000   NaN   30.400000    1.000000    NaN       NaN
75%       51.000000   NaN   34.700000    2.000000    NaN       NaN
max       64.000000   NaN   53.100000    5.000000    NaN       NaN

            expenses
count    1338.000000
unique           NaN
top              NaN
freq             NaN
mean    13270.422414
std     12110.011240
min      1121.870000
25%      4740.287500
50%      9382.030000
75%     16639.915000
max     63770.430000
```

[6]:
```python
#check for the null values
df.isnull().sum()
```

[6]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
expenses    0
dtype: int64
```

[7]:
```python
#visualize the null values
sns.heatmap(df.isnull()==True, cbar=False, yticklabels=False)
#It shows that there is no null values in the datasets
```

[7]: <matplotlib.axes._subplots.AxesSubplot at 0x2509871af70>



[8]:
```
df.shape
```

[8]:
```
(1338, 7)
```

[9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1338 non-null   int64
 1   sex       1338 non-null   object
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   object
 5   region    1338 non-null   object
 6   expenses  1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

[10]:
```
# x and y variables
x=df.drop('expenses', axis=1)
```

[11]:
```
y=df['expenses'].values
```

[12]:
```
print(y)
```

```
[16884.92  1725.55  4449.46 …  1629.83  2007.95 29141.36]
```

[13]:
```
#Now convert the catagorical variables to numeric variables
cols=['sex', 'smoker', 'region']
x= pd.get_dummies(data=x, columns=cols, drop_first=True)
```

[14]:
```
x.head()
```

[14]:
```
   age   bmi  children  sex_male  smoker_yes  region_northwest  \
0   19  27.9         0         0           1                 0
1   18  33.8         1         1           0                 0
2   28  33.0         3         1           0                 0
3   33  22.7         0         1           0                 1
4   32  28.9         0         1           0                 1

   region_southeast  region_southwest
0                 0                 1
1                 1                 0
2                 1                 0
3                 0                 0
4                 0                 0
```

[15]:
```
#scale the features age and bmi to the same scale as of the other features
from sklearn.preprocessing import MinMaxScaler
```

[16]:
```
scaler=MinMaxScaler()
scale_cols=['age', 'bmi']
x[scale_cols]=scaler.fit_transform(x[scale_cols])
```

[17]:
```
x.head()
```

[17]:
```
        age       bmi  children  sex_male  smoker_yes  region_northwest  \
0  0.021739  0.320755         0         0           1                 0
1  0.000000  0.479784         1         1           0                 0
2  0.217391  0.458221         3         1           0                 0
3  0.326087  0.180593         0         1           0                 1
4  0.304348  0.347709         0         1           0                 1

   region_southeast  region_southwest
0                 0                 1
1                 1                 0
2                 1                 0
3                 0                 0
4                 0                 0
```

[18]:
```
from sklearn.tree import DecisionTreeRegressor
```

[19]:
```
from sklearn.model_selection import train_test_split
```

[20]:
```
x_train, x_test, y_train, y_test=train_test_split(x, y, test_size=0.25,␣
↪random_state=0)
```

[21]:
```
tree_reg=DecisionTreeRegressor(max_depth=2, criterion='mse')
tree_reg.fit(x_train, y_train)
```

[21]:
```
DecisionTreeRegressor(max_depth=2)
```

[22]:
```
y_pred_reg=tree_reg.predict(x_test)
```

[23]:
```
y_pred_reg.T
```

[23]:
```
array([12428.30292135, 12428.30292135, 41512.0223301 , 12428.30292135,
       12428.30292135,  5416.65981982,  5416.65981982, 12428.30292135,
        5416.65981982,  5416.65981982,  5416.65981982, 12428.30292135,
       12428.30292135,  5416.65981982, 21502.9989    , 12428.30292135,
       12428.30292135,  5416.65981982,  5416.65981982, 41512.0223301 ,
       21502.9989    , 12428.30292135, 12428.30292135, 21502.9989    ,
        5416.65981982,  5416.65981982,  5416.65981982,  5416.65981982,
        5416.65981982, 12428.30292135,  5416.65981982, 41512.0223301 ,
       12428.30292135, 12428.30292135, 21502.9989    ,  5416.65981982,
       12428.30292135, 41512.0223301 , 41512.0223301 ,  5416.65981982,
        5416.65981982,  5416.65981982, 21502.9989    , 41512.0223301 ,
       41512.0223301 ,  5416.65981982, 12428.30292135,  5416.65981982,
        5416.65981982, 12428.30292135,  5416.65981982,  5416.65981982,
       21502.9989    , 41512.0223301 , 12428.30292135,  5416.65981982,
        5416.65981982, 12428.30292135, 12428.30292135, 12428.30292135,
        5416.65981982, 41512.0223301 , 12428.30292135, 12428.30292135,
       12428.30292135, 12428.30292135, 41512.0223301 , 41512.0223301 ,
        5416.65981982,  5416.65981982, 12428.30292135, 12428.30292135,
       21502.9989    , 12428.30292135, 12428.30292135, 12428.30292135,
       12428.30292135, 12428.30292135, 21502.9989    , 41512.0223301 ,
       12428.30292135, 41512.0223301 ,  5416.65981982, 12428.30292135,
       41512.0223301 , 21502.9989    ,  5416.65981982,  5416.65981982,
       12428.30292135, 41512.0223301 ,  5416.65981982, 12428.30292135,
        5416.65981982, 12428.30292135,  5416.65981982,  5416.65981982,
       41512.0223301 , 41512.0223301 ,  5416.65981982, 12428.30292135,
        5416.65981982,  5416.65981982,  5416.65981982, 41512.0223301 ,
       21502.9989    ,  5416.65981982, 12428.30292135,  5416.65981982,
       12428.30292135, 41512.0223301 , 12428.30292135,  5416.65981982,
       12428.30292135, 41512.0223301 , 41512.0223301 ,  5416.65981982,
        5416.65981982, 12428.30292135, 12428.30292135, 12428.30292135,
       41512.0223301 , 12428.30292135, 12428.30292135,  5416.65981982,
       12428.30292135,  5416.65981982, 21502.9989    , 21502.9989    ,
       41512.0223301 ,  5416.65981982, 12428.30292135,  5416.65981982,
```

```
 5416.65981982, 12428.30292135, 41512.0223301 , 41512.0223301 ,
21502.9989    , 12428.30292135, 21502.9989    ,  5416.65981982,
 5416.65981982, 12428.30292135, 12428.30292135, 12428.30292135,
 5416.65981982, 12428.30292135, 12428.30292135,  5416.65981982,
 5416.65981982, 12428.30292135,  5416.65981982, 41512.0223301 ,
12428.30292135,  5416.65981982,  5416.65981982,  5416.65981982,
 5416.65981982,  5416.65981982, 12428.30292135, 12428.30292135,
 5416.65981982, 12428.30292135, 12428.30292135, 12428.30292135,
 5416.65981982,  5416.65981982, 21502.9989    ,  5416.65981982,
 5416.65981982,  5416.65981982,  5416.65981982,  5416.65981982,
12428.30292135,  5416.65981982,  5416.65981982,  5416.65981982,
 5416.65981982,  5416.65981982, 21502.9989    ,  5416.65981982,
12428.30292135,  5416.65981982, 12428.30292135,  5416.65981982,
 5416.65981982, 21502.9989    ,  5416.65981982,  5416.65981982,
12428.30292135, 12428.30292135, 41512.0223301 ,  5416.65981982,
 5416.65981982, 21502.9989    ,  5416.65981982,  5416.65981982,
 5416.65981982,  5416.65981982,  5416.65981982,  5416.65981982,
12428.30292135, 41512.0223301 , 12428.30292135, 21502.9989    ,
 5416.65981982, 41512.0223301 ,  5416.65981982, 12428.30292135,
 5416.65981982,  5416.65981982, 12428.30292135, 12428.30292135,
 5416.65981982,  5416.65981982,  5416.65981982,  5416.65981982,
 5416.65981982,  5416.65981982, 12428.30292135,  5416.65981982,
 5416.65981982,  5416.65981982, 12428.30292135,  5416.65981982,
12428.30292135, 12428.30292135, 12428.30292135, 12428.30292135,
 5416.65981982,  5416.65981982,  5416.65981982, 12428.30292135,
12428.30292135,  5416.65981982,  5416.65981982,  5416.65981982,
12428.30292135, 41512.0223301 ,  5416.65981982, 12428.30292135,
 5416.65981982, 41512.0223301 ,  5416.65981982, 12428.30292135,
12428.30292135, 12428.30292135,  5416.65981982, 12428.30292135,
 5416.65981982,  5416.65981982, 21502.9989    , 41512.0223301 ,
 5416.65981982,  5416.65981982,  5416.65981982,  5416.65981982,
12428.30292135,  5416.65981982,  5416.65981982,  5416.65981982,
12428.30292135, 21502.9989    , 41512.0223301 , 12428.30292135,
 5416.65981982, 12428.30292135, 41512.0223301 , 12428.30292135,
41512.0223301 ,  5416.65981982, 41512.0223301 ,  5416.65981982,
12428.30292135,  5416.65981982, 41512.0223301 ,  5416.65981982,
12428.30292135, 12428.30292135,  5416.65981982, 12428.30292135,
12428.30292135,  5416.65981982, 12428.30292135,  5416.65981982,
 5416.65981982,  5416.65981982,  5416.65981982, 21502.9989    ,
 5416.65981982,  5416.65981982,  5416.65981982, 41512.0223301 ,
12428.30292135,  5416.65981982, 12428.30292135, 12428.30292135,
41512.0223301 ,  5416.65981982,  5416.65981982, 12428.30292135,
 5416.65981982,  5416.65981982, 21502.9989    , 21502.9989    ,
12428.30292135,  5416.65981982,  5416.65981982, 12428.30292135,
12428.30292135, 21502.9989    , 21502.9989    ,  5416.65981982,
21502.9989    ,  5416.65981982,  5416.65981982,  5416.65981982,
12428.30292135,  5416.65981982, 12428.30292135, 41512.0223301 ,
```

```
        5416.65981982,  5416.65981982,  5416.65981982, 12428.30292135,
        5416.65981982, 12428.30292135, 41512.0223301 , 21502.9989    ,
       12428.30292135, 41512.0223301 , 12428.30292135,  5416.65981982,
       12428.30292135,  5416.65981982, 21502.9989    ])
```

0.0.1 How to Check the Model Variabiity and Bias-Variance Trade-Off??

[24]:
```python
from sklearn.model_selection import cross_validate, KFold
```

[25]:
```python
K_Fold=KFold(n_splits=10, shuffle=True, random_state=42)
depth={}

for i in range(2,11):
    tree_cv=cross_validate(DecisionTreeRegressor(max_depth=i), x, y, cv=K_Fold,␣
 ↪scoring=['r2'])
    depth['depth_' +str(i)]=tree_cv['test_r2']
depth
```

[25]:
```
{'depth_2': array([0.82797876, 0.83627513, 0.82846619, 0.79627887, 0.88576425,
        0.82437852, 0.78101358, 0.73007204, 0.80832654, 0.83971832]),
 'depth_3': array([0.85628937, 0.87237849, 0.85760018, 0.81877829, 0.91590791,
        0.85343578, 0.83150173, 0.75706664, 0.80167779, 0.84678683]),
 'depth_4': array([0.85300297, 0.86905526, 0.86086231, 0.84318317, 0.91869788,
        0.84800633, 0.82755151, 0.74791778, 0.81648643, 0.85371445]),
 'depth_5': array([0.84993277, 0.8802852 , 0.87258964, 0.8492467 , 0.89978505,
        0.85098472, 0.80967563, 0.74832063, 0.81547351, 0.85134251]),
 'depth_6': array([0.84655847, 0.85905411, 0.88005639, 0.82556283, 0.89192687,
        0.84481696, 0.79283807, 0.7439385 , 0.80581394, 0.84064556]),
 'depth_7': array([0.84696145, 0.830672  , 0.87047463, 0.79272067, 0.88495914,
        0.83069584, 0.77334081, 0.73076654, 0.77484306, 0.78097405]),
 'depth_8': array([0.82215768, 0.83557225, 0.83119287, 0.75026821, 0.8515351 ,
        0.79491397, 0.72079798, 0.71526931, 0.78600994, 0.73315571]),
 'depth_9': array([0.7837532 , 0.80745989, 0.81908771, 0.64583951, 0.84800343,
        0.74552177, 0.64208233, 0.73292473, 0.75064992, 0.67133338]),
 'depth_10': array([0.77312684, 0.78511287, 0.78250649, 0.66597412, 0.84996297,
        0.67961468, 0.55739369, 0.68449381, 0.70355455, 0.69324033])}
```

[26]:
```python
#Now convert it into dataframe
df=pd.DataFrame(depth)
```

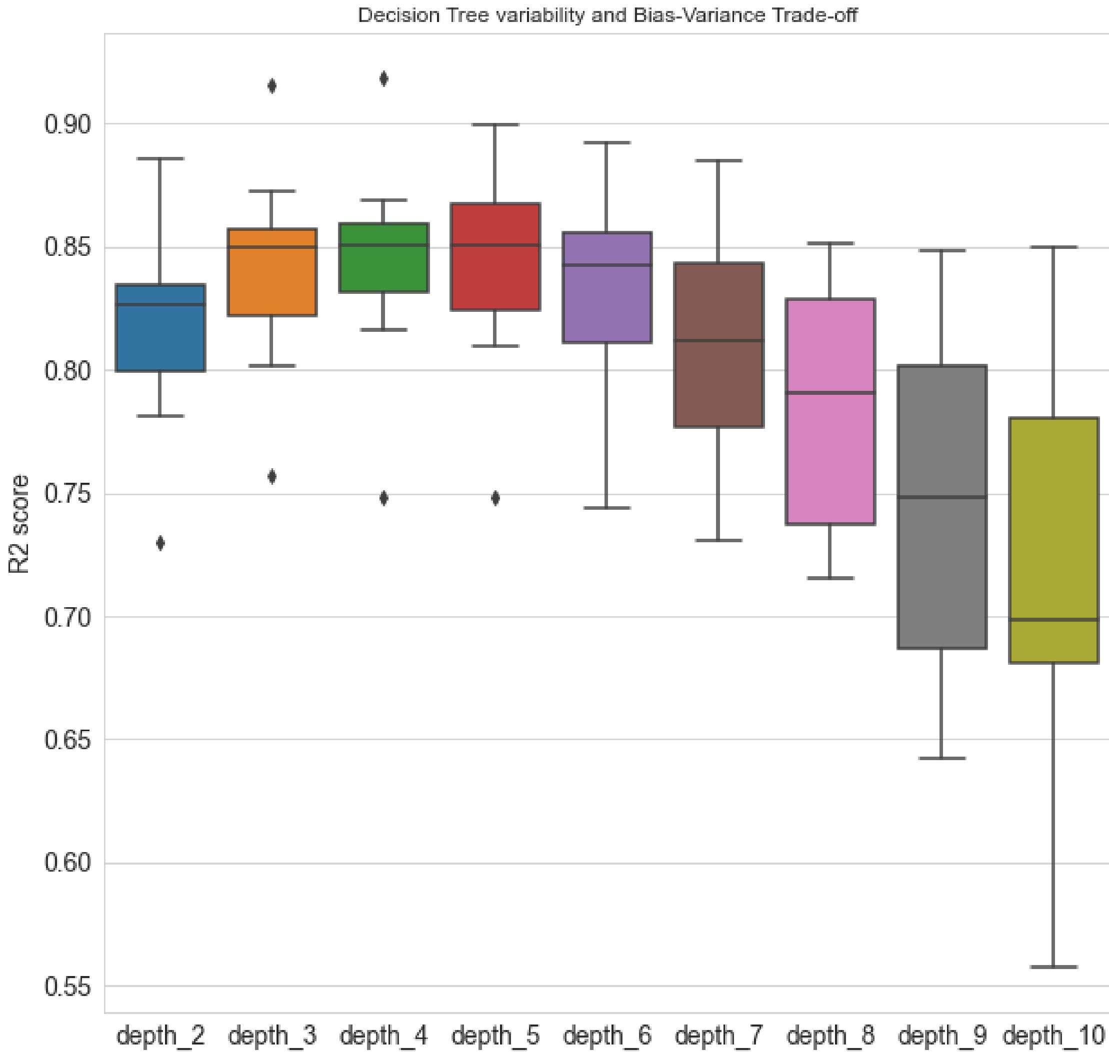
[27]:
```python
df
```

[27]:
```
    depth_2   depth_3   depth_4   depth_5   depth_6   depth_7   depth_8  \
0  0.827979  0.856289  0.853003  0.849933  0.846558  0.846961  0.822158
1  0.836275  0.872378  0.869055  0.880285  0.859054  0.830672  0.835572
2  0.828466  0.857600  0.860862  0.872590  0.880056  0.870475  0.831193
3  0.796279  0.818778  0.843183  0.849247  0.825563  0.792721  0.750268
```

```
4  0.885764  0.915908  0.918698  0.899785  0.891927  0.884959  0.851535
5  0.824379  0.853436  0.848006  0.850985  0.844817  0.830696  0.794914
6  0.781014  0.831502  0.827552  0.809676  0.792838  0.773341  0.720798
7  0.730072  0.757067  0.747918  0.748321  0.743938  0.730767  0.715269
8  0.808327  0.801678  0.816486  0.815474  0.805814  0.774843  0.786010
9  0.839718  0.846787  0.853714  0.851343  0.840646  0.780974  0.733156

    depth_9  depth_10
0  0.783753  0.773127
1  0.807460  0.785113
2  0.819088  0.782506
3  0.645840  0.665974
4  0.848003  0.849963
5  0.745522  0.679615
6  0.642082  0.557394
7  0.732925  0.684494
8  0.750650  0.703555
9  0.671333  0.693240
```

[28]:

```python
# Now choose the best depth through visualozing boxplots
plt.figure(figsize=(10,10))
sns.set_style('whitegrid')
sns.boxplot(data=df)
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)
plt.ylabel("R2 score", fontsize=14)
plt.title("Decision Tree variability and Bias-Variance Trade-off")
plt.show()
```



[29]:
```
# Reference:
# machine-learning_Dr.B. (2020, March 18). Lesson 17 Decision Trees.
# YouTube. https://www.youtube.com/watch?v=KIuB9nsVKqY&t=709s
```