Homework Assignment 2

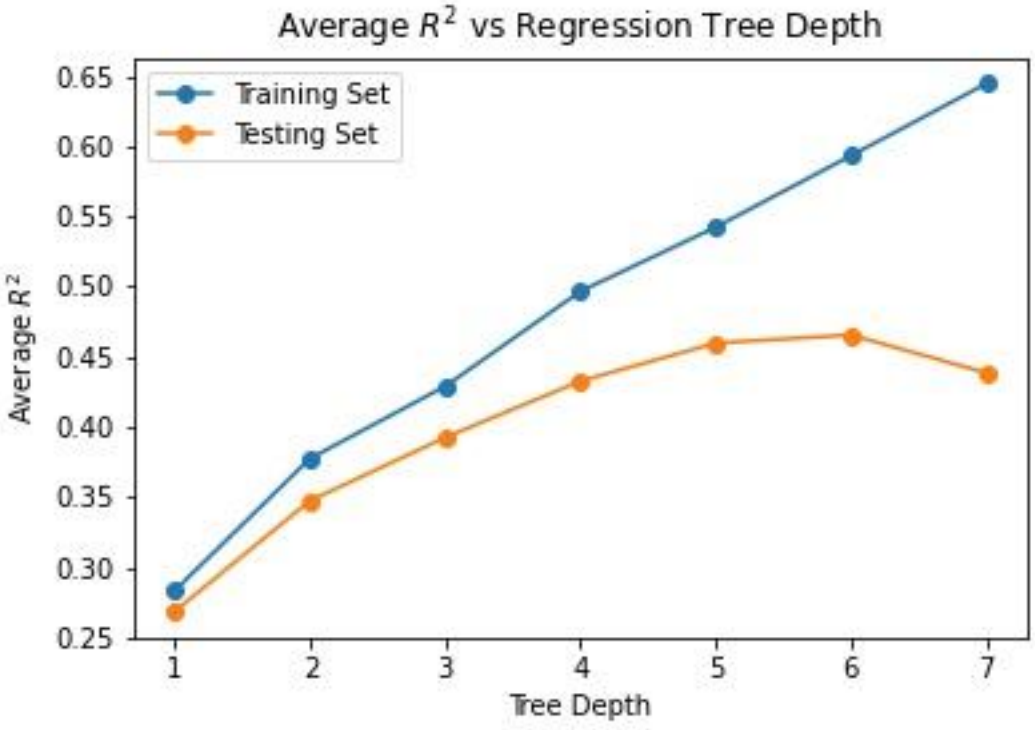
Jarod Klion

December 19th, 2021

a)

OLS Regression	
Set	Average R^2
training	0.9538
test	0.9544

b)



c)

Random Forest Regression		
Number of Trees	Average Training R^2	Average Testing R^2
10	0.9111	0.4916
30	0.9295	0.5204
100	0.9359	0.5311
300	0.9376	0.5342

```
#!/usr/bin/env python
# coding: utf-8

# In[424]:


import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import statsmodels.api as sm
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
import matplotlib.pyplot as plt


# In[55]:


#Read in the needed data for the assignment
df = pd.read_csv("abalone.csv", header=None)
X = df.iloc[:, :-1] #Select first 7 columns as predictors
y = df.iloc[:, -1:] #Select last column as the response
df


# In[86]:


#Set up the designated split
```

```
X_train, X_test, y_train, y_test = zip(*(train_test_split(X, y,
test_size=0.1, random_state = i) for i in range(1,11)))

#zip(*) unpacks the lists to each variable



# In[171]:



#Perform OLS regression on training and testing set analytically using
linalg solver

betas_train = [np.linalg.solve(X_train[i].T @ X_train[i] + 0.0001,
X_train[i].T @ y_train[i]) for i in range(10)]

betas_test = [np.linalg.solve(X_test[i].T @ X_test[i] + 0.0001,
X_test[i].T @ y_test[i]) for i in range(10)]

#print(*betas_train, sep="\n"*2)


#Show that these match the regression

reg_train = [sm.OLS(y_train[i], X_train[i]).fit() for i in range(10)]

reg_test = [sm.OLS(y_test[i], X_test[i]).fit() for i in range(10)]

#for i in range(10):

#    print("beta value: " + str(betas_train[i]), "\t", "regression
value: " + str(reg_train[i].params))



# In[514]:



R2_train = np.mean([reg_train[i].rsquared for i in range(10)])

R2_test = np.mean([reg_test[i].rsquared for i in range(10)])

print("Average training R^2: {}".format(R2_train), "\nAverage testing
R^2: {}".format(R2_test))
```

```
# In[445]:



#Define a function to create regression trees
def run_tree_reg(X_train, X_test, y_train, y_test, depth):
    tree_reg = DecisionTreeRegressor(random_state = 42, max_depth =
depth).fit(X_train, y_train)
    #this accuracy measure is the same as R^2
    acc_train = tree_reg.score(X_train, y_train)
    acc_test = tree_reg.score(X_test, y_test)
    #print("Tree depth: ", depth)
    #print("Training set accuracy: ", round(acc_train*100,4), "%")
    #print("Testing set accuracy: ", round(acc_test*100,4), "%")
    return acc_train, acc_test



# In[477]:



#Create 7 regression trees for each depth asked
tree_acc_train = np.empty(shape= (7,10))
tree_acc_test = np.empty(shape= (7,10))
depth = range(1,8)
for i in depth:
    for k in range(10):
        tree_acc_train[i-1][k], tree_acc_test[i-1][k] =
run_tree_reg(X_train[k], X_test[k], y_train[k], y_test[k], i)


#Calculate the mean for each tree depth
```

```
tree_R2_train_mean = [np.mean(tree_acc_train[i]) for i in range(7)]
tree_R2_test_mean = [np.mean(tree_acc_test[i]) for i in range(7)]



# In[447]:



#Plot train and test R^2 vs the tree depth on the same graph
fig1 = plt.figure(1)
plt.plot(depth, tree_R2_train_mean, "-o", label = "Training Set")
plt.plot(depth, tree_R2_test_mean, "-o", label = "Testing Set")
plt.xlabel("Tree Depth")
plt.ylabel("Average $R^2$")
plt.title("Average $R^2$ vs Regression Tree Depth")
plt.legend()
plt.show()
#fig1.savefig("RegressionTree.jpg")



# In[448]:



#Define a function to create random forest regressions
def run_rf_reg(X_train, X_test, y_train, y_test, trees):
    rf_reg = RandomForestRegressor(random_state = 42, n_estimators =
trees).fit(X_train, y_train)
    #this accuracy measure is the same as R^2
    acc_train = rf_reg.score(X_train, y_train)
    acc_test = rf_reg.score(X_test, y_test)
    print("# of trees: ", trees)
```

```
    print("Training set accuracy: ", round(acc_train*100,4), "%")
    print("Testing set accuracy: ", round(acc_test*100,4), "%")
    return acc_train, acc_test


# In[482]:



#Create a random forest regression for each different tree value (10,
30, 100, 300)
rf_acc_train = np.empty(shape= (4,10))
rf_acc_test = np.empty(shape= (4,10))
trees = [10, 30, 100, 300]
for i, tree in enumerate(trees):
    for k in range(10):
        rf_acc_train[i][k], rf_acc_test[i][k] = run_rf_reg(X_train[k],
X_test[k], y_train[k].values.ravel(), y_test[k].values.ravel(), tree)


#Calculate the mean for each different tree count
rf_R2_train_mean = [np.mean(rf_acc_train[i]) for i in range(4)]
rf_R2_test_mean = [np.mean(rf_acc_test[i]) for i in range(4)]



# In[509]:



print(*[f"\nNumber of trees: {tree}\n" + f"Average training R^2:
{rf_R2_train_mean[i]}\n" + f"Average test R^2: {rf_R2_test_mean[i]}" for
i, tree in enumerate(trees)])
```