assignment 18 (own)

March 8, 2022

1 Decision tree predicting the numerical column of price in Melbourne dataset

[24]:
```
import pandas as pd
import seaborn as sns
```

[25]:
```
from sklearn.tree import DecisionTreeRegressor
```

[26]:
```
from sklearn.model_selection import train_test_split
```

[27]:
```
from sklearn import tree
import graphviz

def plot_tree_regression(model, features):
    # Generate plot data
    dot_data = tree.export_graphviz(model, out_file=None,
                                    feature_names=features,
                                    filled=True, rounded=True,
                                    special_characters=True)

    # Turn into graph using graphviz
    graph = graphviz.Source(dot_data)

    # Write out a pdf
    graph.render("decision_tree")

    # Display in the notebook
    return graph
```

[28]:
```
df = pd.read_csv('melbourne_housing_prices.csv', sep=',')
df.head()
```

[28]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	\
0	Abbotsford	49 Lithgow St	3	h	1490000.0	S	Jellis	
1	Abbotsford	59A Turner St	3	h	1220000.0	S	Marshall	
2	Abbotsford	119B Yarra St	3	h	1420000.0	S	Nelson	
3	Aberfeldie	68 Vida St	3	h	1515000.0	S	Barry	

```
 4  Airport West  92 Clydesdale Rd      2    h   670000.0      S    Nelson

         Date  Postcode             Regionname  Propertycount  Distance  \
 0  1/04/2017      3067  Northern Metropolitan           4019       3.0
 1  1/04/2017      3067  Northern Metropolitan           4019       3.0
 2  1/04/2017      3067  Northern Metropolitan           4019       3.0
 3  1/04/2017      3040   Western Metropolitan           1543       7.5
 4  1/04/2017      3042   Western Metropolitan           3464      10.4

                   CouncilArea
 0          Yarra City Council
 1          Yarra City Council
 2          Yarra City Council
 3  Moonee Valley City Council
 4  Moonee Valley City Council
```

[29]:
```
df_model = df.dropna()
```

[30]:
```
df_model.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 48433 entries, 0 to 63020
Data columns (total 13 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Suburb         48433 non-null  object
 1   Address        48433 non-null  object
 2   Rooms          48433 non-null  int64
 3   Type           48433 non-null  object
 4   Price          48433 non-null  float64
 5   Method         48433 non-null  object
 6   SellerG        48433 non-null  object
 7   Date           48433 non-null  object
 8   Postcode       48433 non-null  int64
 9   Regionname     48433 non-null  object
 10  Propertycount  48433 non-null  int64
 11  Distance       48433 non-null  float64
 12  CouncilArea    48433 non-null  object
dtypes: float64(2), int64(3), object(8)
memory usage: 5.2+ MB
```

2 Trying to predict the price of a home based on the distance from the Melbourne Central Business District and amount of rooms

[31]:
```
df_train, df_test = train_test_split(df_model, test_size=0.3,␣
↪stratify=df_model['Regionname'], random_state=42)
```

[60]:
```
features= ['Distance','Rooms']
dt_regression = DecisionTreeRegressor(max_depth = 10) # Increase max_depth to␣
↪see effect in the plot
dt_regression.fit(df_train[features], df_train['Price'])
```

[60]:
```
DecisionTreeRegressor(max_depth=10)
```

[33]:
```
def calculate_rmse(predictions, actuals):
    if(len(predictions) != len(actuals)):
        raise Exception("The amount of predictions did not equal the amount of␣
↪actuals")

    return (((predictions - actuals) ** 2).sum() / len(actuals)) ** (1/2)
```

[58]:
```
predictionsOnTrainset = dt_regression.predict(df_train[features])
predictionsOnTestset = dt_regression.predict(df_test[features])

rmseTrain = calculate_rmse(predictionsOnTrainset, df_train.Price)
rmseTest = calculate_rmse(predictionsOnTestset, df_test.Price)

print("RMSE on training set " + str(rmseTrain))
print("RMSE on test set " + str(rmseTest))
```

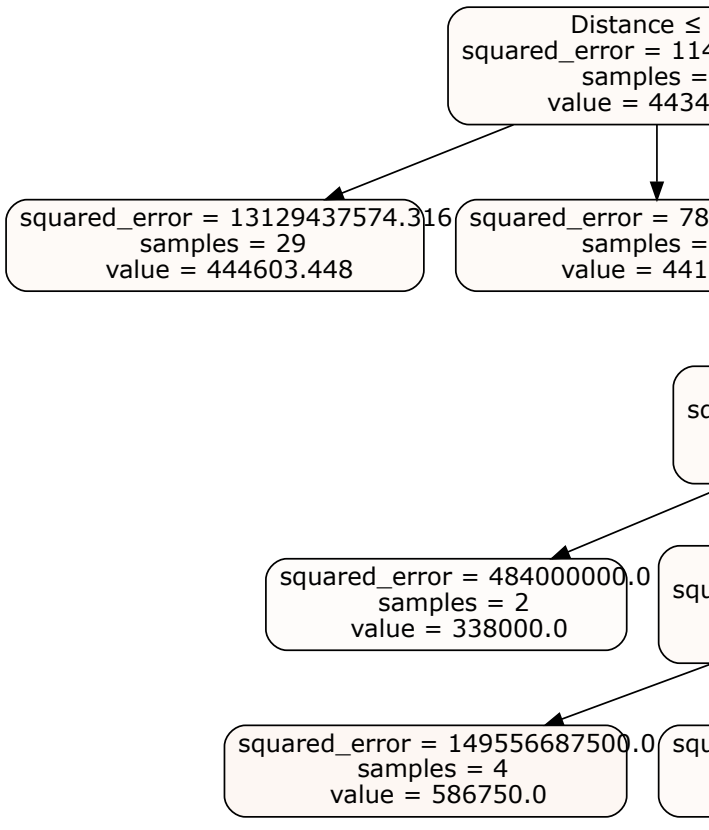
```
RMSE on training set 367953.3094318631
RMSE on test set 395359.04588201595
```

The difference between the training set and the test set is approximately 50 grams. That is not a lot imo, since that means the model isn't to overfitted. Also, a RMSE of 400 grams where most penguins are approximately 3.5 to 5.5 kilos sounds about right as well.

The tree first looks at the room and than at distance. Of course, the more rooms the more the price goes up and the lower the distance the higher the price. So under each room decision, the more to the left the higher the price (generally).

[61]:
```
plot_tree_regression(dt_regression, features)
```

[61]:


Distance ≤
squared_error = 114
samples =
value = 4434
squared_error = 13129437574.316
samples = 29
value = 444603.448
squared_error = 78
samples =
value = 441
sq
squared_error = 484000000.0
samples = 2
value = 338000.0
squ
squared_error = 149556687500.0
samples = 4
value = 586750.0
squ