初阶-技术创意 2-实验二-实验报告

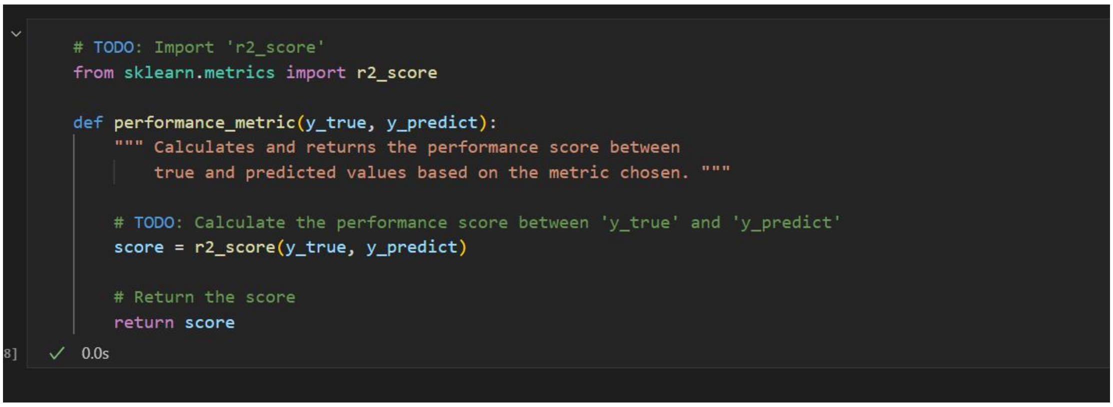
(1) 实验目的

理解如何对数据集 *fit* 和 *evaluate*。

(2) 实验仪器/设备

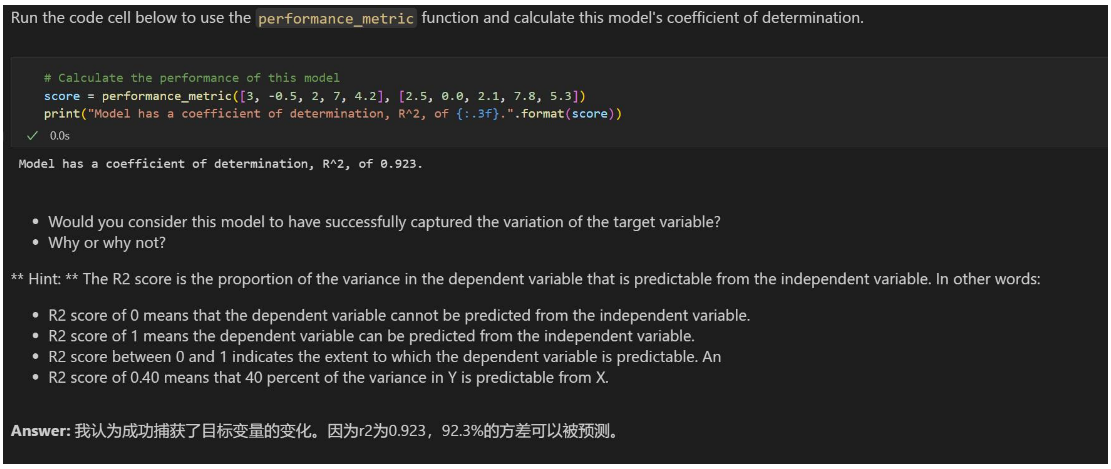
vscode

(3) 实验过程



```
# TODO: Import 'r2_score'
from sklearn.metrics import r2_score

def performance_metric(y_true, y_predict):
    """ Calculates and returns the performance score between 
        true and predicted values based on the metric chosen. """
    
    # TODO: Calculate the performance score between 'y_true' and 'y_predict'
    score = r2_score(y_true, y_predict)
    
    # Return the score
    return score
```

8] ✓ 0.0s



Run the code cell below to use the `performance_metric` function and calculate this model's coefficient of determination.

```
# Calculate the performance of this model
score = performance_metric([3, -0.5, 2, 7, 4.2], [2.5, 0.0, 2.1, 7.8, 5.3])
print("Model has a coefficient of determination, R^2, of {:.3f}.".format(score))
```

✓ 0.0s

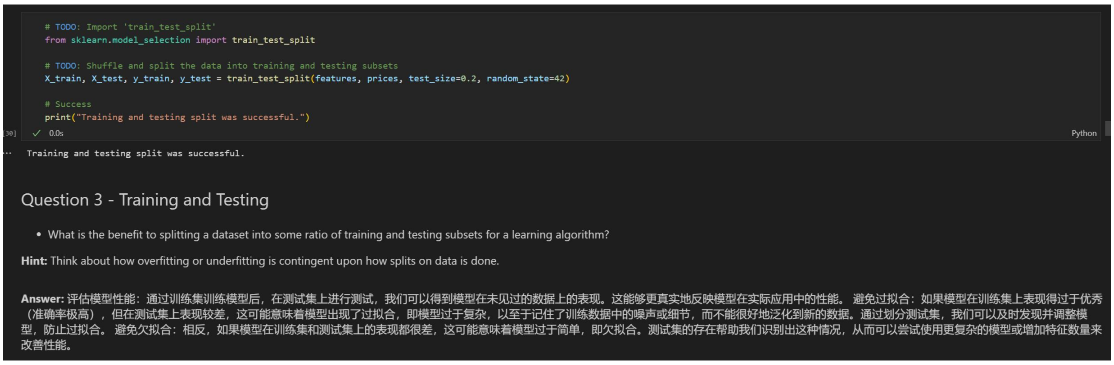
```
Model has a coefficient of determination, R^2, of 0.923.
```

- Would you consider this model to have successfully captured the variation of the target variable?
- Why or why not?

** Hint: ** The R2 score is the proportion of the variance in the dependent variable that is predictable from the independent variable. In other words:

- R2 score of 0 means that the dependent variable cannot be predicted from the independent variable.
- R2 score of 1 means the dependent variable can be predicted from the independent variable.
- R2 score between 0 and 1 indicates the extent to which the dependent variable is predictable. An
- R2 score of 0.40 means that 40 percent of the variance in Y is predictable from X.

Answer: 我认为成功捕获了目标变量的变化。因为r2为0.923，92.3%的方差可以被预测。



```python
# TODO: Import 'train_test_split'
from sklearn.model_selection import train_test_split

# TODO: Shuffle and split the data into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(features, prices, test_size=0.2, random_state=42)

# Success
print("Training and testing split was successful.")
```

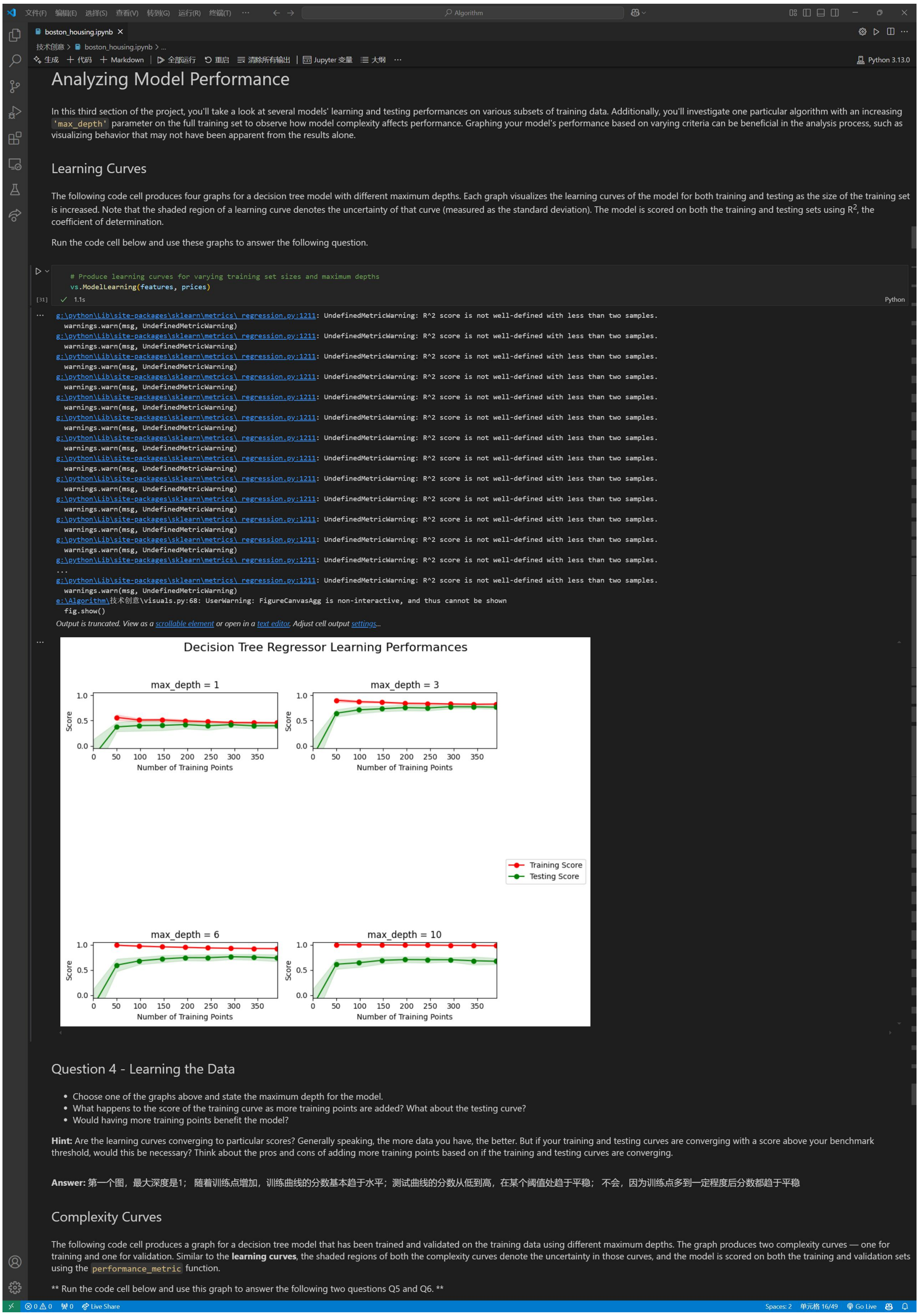
```
Training and testing split was successful.
```

Question 3 - Training and Testing

- What is the benefit to splitting a dataset into some ratio of training and testing subsets for a learning algorithm?

Hint: Think about how overfitting or underfitting is contingent upon how splits on data is done.

Answer: 评估模型性能：通过训练集训练模型后，在测试集上进行测试，我们可以得到模型在未见过的数据上的表现。这能够更真实地反映模型在实际应用中的性能。 避免过拟合：如果模型在训练集上表现得过于优秀（准确率极高），但在测试集上表现较差，这可能意味着模型出现了过拟合，即模型过于复杂，以至于记住了训练数据中的噪声或细节，而不能很好地泛化到新的数据。通过划分测试集，我们可以及时发现并调整模型，防止过拟合。 避免欠拟合：相反，如果模型在训练集和测试集上的表现都很差，这可能意味着模型过于简单，即欠拟合。测试集的存在帮助我们识别出这种情况，从而可以尝试使用更复杂的模型或增加特征数量来改善性能。



Analyzing Model Performance

In this third section of the project, you'll take a look at several models' learning and testing performances on various subsets of training data. Additionally, you'll investigate one particular algorithm with an increasing `'max_depth'` parameter on the full training set to observe how model complexity affects performance. Graphing your model's performance based on varying criteria can be beneficial in the analysis process, such as visualizing behavior that may not have been apparent from the results alone.

Learning Curves

The following code cell produces four graphs for a decision tree model with different maximum depths. Each graph visualizes the learning curves of the model for both training and testing as the size of the training set is increased. Note that the shaded region of a learning curve denotes the uncertainty of that curve (measured as the standard deviation). The model is scored on both the training and testing sets using R^2, the coefficient of determination.

Run the code cell below and use these graphs to answer the following question.

```
# Produce learning curves for varying training set sizes and maximum depths
vs.ModelLearning(features, prices)
```

```
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
...
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
e:\Algorithm\技术创意\visuals.py:68: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
  fig.show()
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Question 4 - Learning the Data

- Choose one of the graphs above and state the maximum depth for the model.
- What happens to the score of the training curve as more training points are added? What about the testing curve?
- Would having more training points benefit the model?

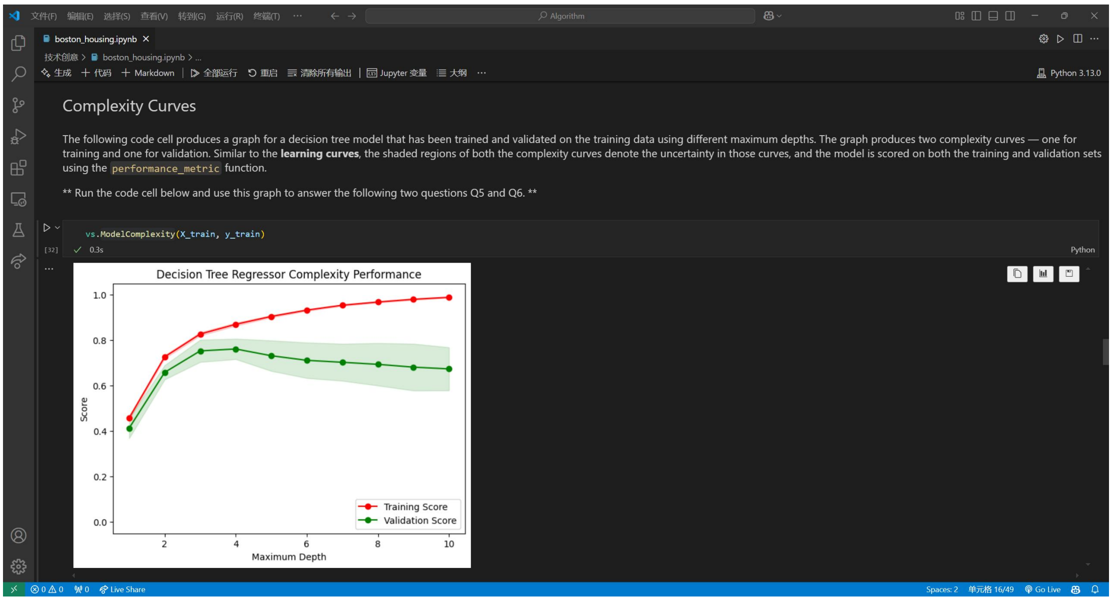
Hint: Are the learning curves converging to particular scores? Generally speaking, the more data you have, the better. But if your training and testing curves are converging with a score above your benchmark threshold, would this be necessary? Think about the pros and cons of adding more training points based on if the training and testing curves are converging.

Answer: 第一个图，最大深度是1； 随着训练点增加，训练曲线的分数基本趋于水平；测试曲线的分数从低到高，在某个阈值处趋于平稳； 不会，因为训练点多到一定程度后分数都趋于平稳

Complexity Curves

The following code cell produces a graph for a decision tree model that has been trained and validated on the training data using different maximum depths. The graph produces two complexity curves — one for training and one for validation. Similar to the **learning curves**, the shaded regions of both the complexity curves denote the uncertainty in those curves, and the model is scored on both the training and validation sets using the `performance_metric` function.

** Run the code cell below and use this graph to answer the following two questions Q5 and Q6. **


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) …
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown | 全部运行 重启 清除所有输出 | Jupyter 变量 大纲 …
Python 3.13.0
[32]
0.3s
Python
0 0 0 Live Share
Spaces: 2 单元格 16/49 Go Live

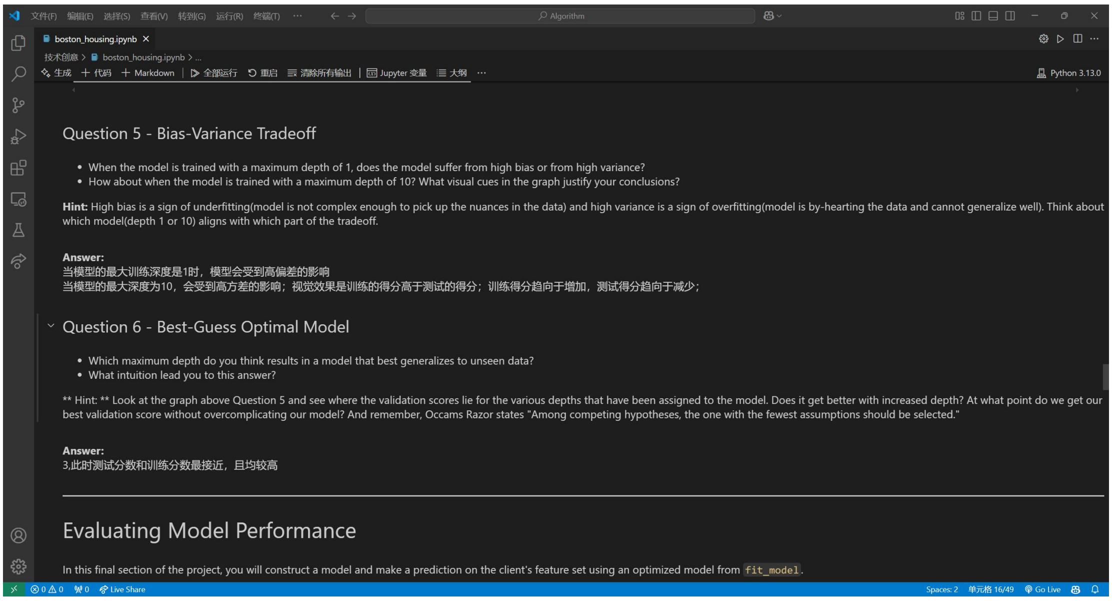
Complexity Curves

The following code cell produces a graph for a decision tree model that has been trained and validated on the training data using different maximum depths. The graph produces two complexity curves — one for training and one for validation. Similar to the **learning curves**, the shaded regions of both the complexity curves denote the uncertainty in those curves, and the model is scored on both the training and validation sets using the `performance_metric` function.

** Run the code cell below and use this graph to answer the following two questions Q5 and Q6. **

```
vs.ModelComplexity(X_train, y_train)
```

Decision Tree Regressor Complexity Performance
Score
1.0
0.8
0.6
0.4
0.2
0.0
2
4
6
8
10
Maximum Depth
Training Score
Validation Score


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) …
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown | 全部运行 重启 清除所有输出 | Jupyter 变量 大纲 …
Python 3.13.0
0 0 0 Live Share
Spaces: 2 单元格 16/49 Go Live

Question 5 - Bias-Variance Tradeoff

- When the model is trained with a maximum depth of 1, does the model suffer from high bias or from high variance?
- How about when the model is trained with a maximum depth of 10? What visual cues in the graph justify your conclusions?

Hint: High bias is a sign of underfitting(model is not complex enough to pick up the nuances in the data) and high variance is a sign of overfitting(model is by-hearting the data and cannot generalize well). Think about which model(depth 1 or 10) aligns with which part of the tradeoff.

Answer:
当模型的最大训练深度是1时，模型会受到高偏差的影响
当模型的最大深度为10，会受到高方差的影响；视觉效果是训练的得分高于测试的得分；训练得分趋向于增加，测试得分趋向于减少；

Question 6 - Best-Guess Optimal Model

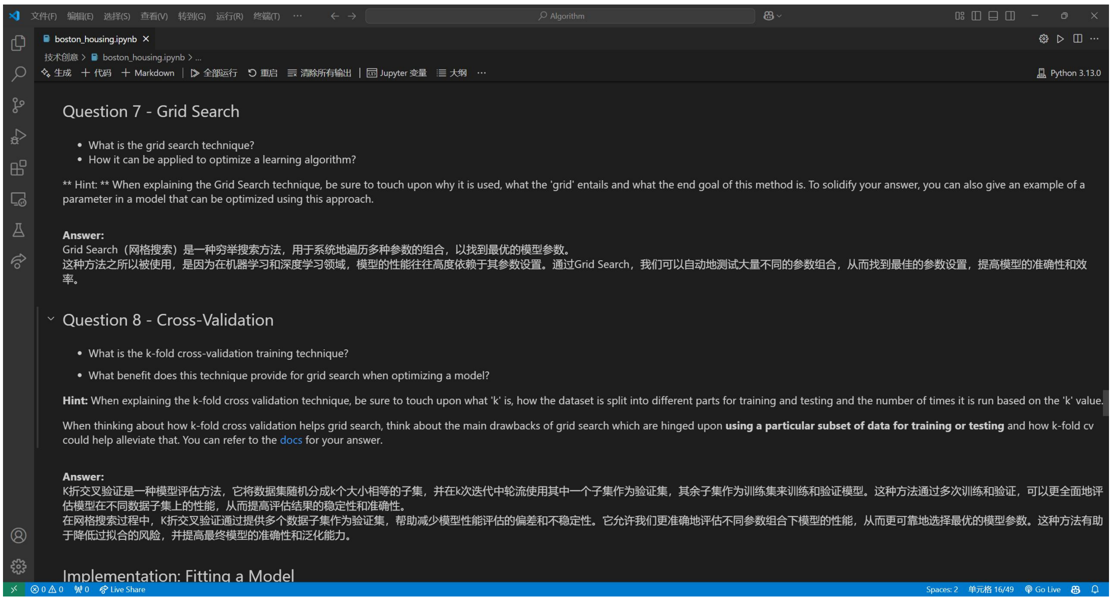
- Which maximum depth do you think results in a model that best generalizes to unseen data?
- What intuition lead you to this answer?

** Hint: ** Look at the graph above Question 5 and see where the validation scores lie for the various depths that have been assigned to the model. Does it get better with increased depth? At what point do we get our best validation score without overcomplicating our model? And remember, Occams Razor states "Among competing hypotheses, the one with the fewest assumptions should be selected."

Answer:
3,此时测试分数和训练分数最接近，且均较高

Evaluating Model Performance

In this final section of the project, you will construct a model and make a prediction on the client's feature set using an optimized model from `fit_model`.


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) …
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown | 全部运行 重启 清除所有输出 | Jupyter 变量 大纲 …
Python 3.13.0
0 0 0 Live Share
Spaces: 2 单元格 16/49 Go Live

Question 7 - Grid Search

- What is the grid search technique?
- How it can be applied to optimize a learning algorithm?

** Hint: ** When explaining the Grid Search technique, be sure to touch upon why it is used, what the 'grid' entails and what the end goal of this method is. To solidify your answer, you can also give an example of a parameter in a model that can be optimized using this approach.

Answer:
Grid Search（网格搜索）是一种穷举搜索方法，用于系统地遍历多种参数的组合，以找到最优的模型参数。
这种方法之所以被使用，是因为在机器学习和深度学习领域，模型的性能往往高度依赖于其参数设置。通过Grid Search，我们可以自动地测试大量不同的参数组合，从而找到最佳的参数设置，提高模型的准确性和效率。

Question 8 - Cross-Validation

- What is the k-fold cross-validation training technique?

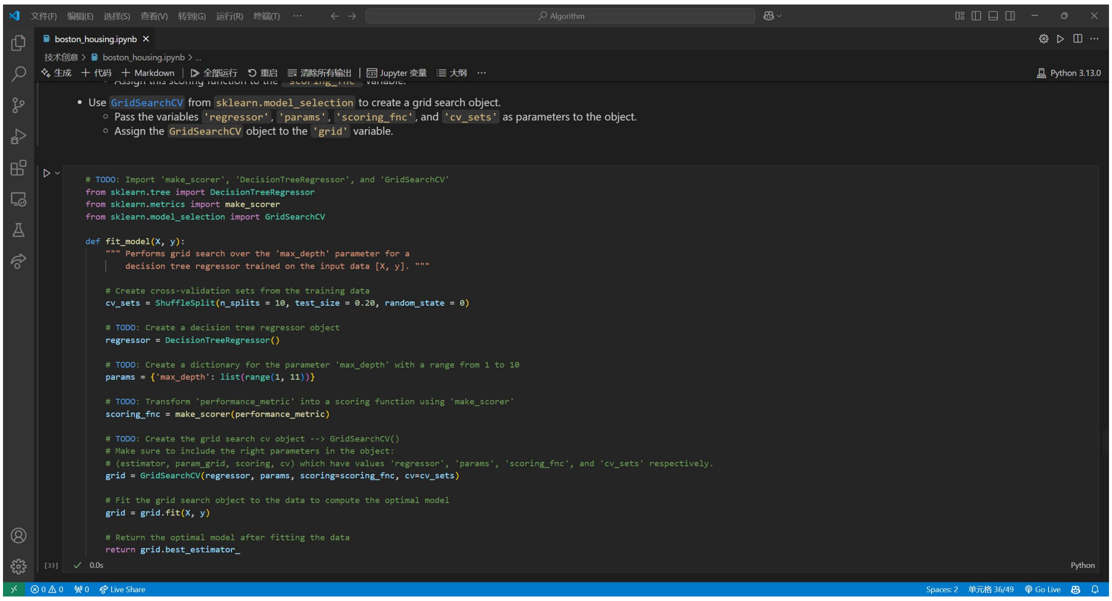
- What benefit does this technique provide for grid search when optimizing a model?

Hint: When explaining the k-fold cross validation technique, be sure to touch upon what 'k' is, how the dataset is split into different parts for training and testing and the number of times it is run based on the 'k' value.

When thinking about how k-fold cross validation helps grid search, think about the main drawbacks of grid search which are hinged upon **using a particular subset of data for training or testing** and how k-fold cv could help alleviate that. You can refer to the docs for your answer.

Answer:
K折交叉验证是一种模型评估方法，它将数据集随机分成k个大小相等的子集，并在k次迭代中轮流使用其中一个子集作为验证集，其余子集作为训练集来训练和验证模型。这种方法通过多次训练和验证，可以更全面地评估模型在不同数据子集上的性能，从而提高评估结果的稳定性和准确性。
在网格搜索过程中，K折交叉验证通过提供多个数据子集作为验证集，帮助减少模型性能评估的偏差和不稳定性。它允许我们更准确地评估不同参数组合下模型的性能，从而更可靠地选择最优的模型参数。这种方法有助于降低过拟合的风险，并提高最终模型的准确性和泛化能力。

Implementation: Fitting a Model

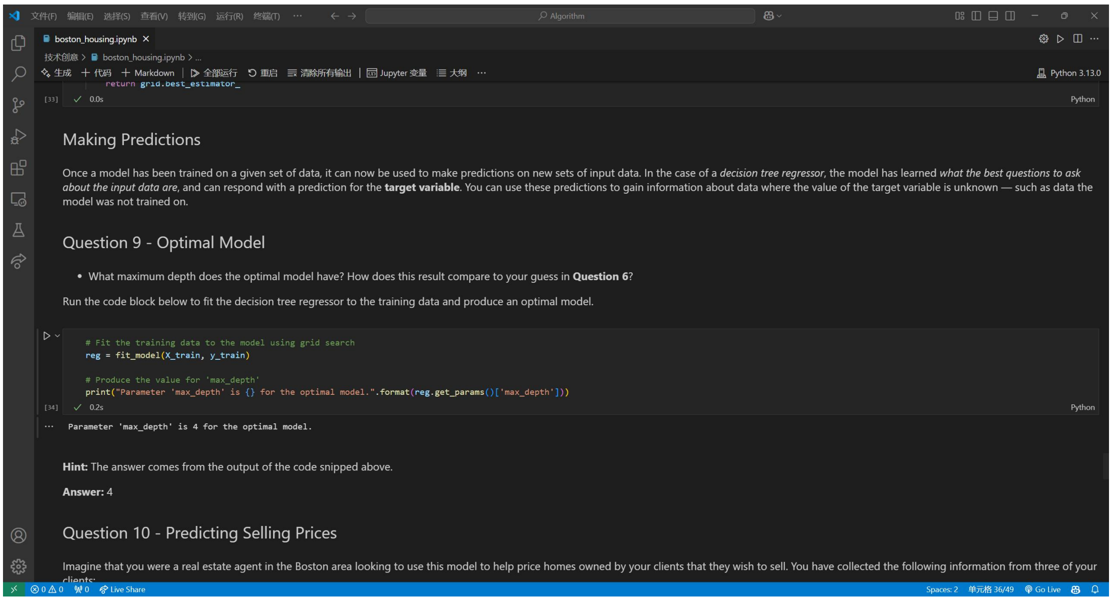

文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) ···
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown 全部运行 重启 清除所有输出 Jupyter 变量 大纲
Python 3.13.0
Python
Live Share
Spaces: 2 单元格 36/49 Go Live

- Assign this scoring function to the `'scoring_fnc'` variable.

- Use `GridSearchCV` from `sklearn.model_selection` to create a grid search object.
  - Pass the variables `'regressor'`, `'params'`, `'scoring_fnc'`, and `'cv_sets'` as parameters to the object.
  - Assign the `GridSearchCV` object to the `'grid'` variable.

```python
# TODO: Import 'make_scorer', 'DecisionTreeRegressor', and 'GridSearchCV'
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import make_scorer
from sklearn.model_selection import GridSearchCV

def fit_model(X, y):
    """ Performs grid search over the 'max_depth' parameter for a
        decision tree regressor trained on the input data [X, y]. """

    # Create cross-validation sets from the training data
    cv_sets = ShuffleSplit(n_splits = 10, test_size = 0.20, random_state = 0)

    # TODO: Create a decision tree regressor object
    regressor = DecisionTreeRegressor()

    # TODO: Create a dictionary for the parameter 'max_depth' with a range from 1 to 10
    params = {'max_depth': list(range(1, 11))}

    # TODO: Transform 'performance_metric' into a scoring function using 'make_scorer'
    scoring_fnc = make_scorer(performance_metric)

    # TODO: Create the grid search cv object --> GridSearchCV()
    # Make sure to include the right parameters in the object:
    # (estimator, param_grid, scoring, cv) which have values 'regressor', 'params', 'scoring_fnc', and 'cv_sets' respectively.
    grid = GridSearchCV(regressor, params, scoring=scoring_fnc, cv=cv_sets)

    # Fit the grid search object to the data to compute the optimal model
    grid = grid.fit(X, y)

    # Return the optimal model after fitting the data
    return grid.best_estimator_
```

[33] ✓ 0.0s


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) ···
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown 全部运行 重启 清除所有输出 Jupyter 变量 大纲
Python 3.13.0
Python
Live Share
Spaces: 2 单元格 36/49 Go Live

```python
    return grid.best_estimator_
```

[33] ✓ 0.0s

Making Predictions

Once a model has been trained on a given set of data, it can now be used to make predictions on new sets of input data. In the case of a *decision tree regressor*, the model has learned *what the best questions to ask about the input data are*, and can respond with a prediction for the **target variable**. You can use these predictions to gain information about data where the value of the target variable is unknown — such as data the model was not trained on.

Question 9 - Optimal Model

- What maximum depth does the optimal model have? How does this result compare to your guess in **Question 6**?

Run the code block below to fit the decision tree regressor to the training data and produce an optimal model.

```python
# Fit the training data to the model using grid search
reg = fit_model(X_train, y_train)

# Produce the value for 'max_depth'
print("Parameter 'max_depth' is {} for the optimal model.".format(reg.get_params()['max_depth']))
```

[34] ✓ 0.2s

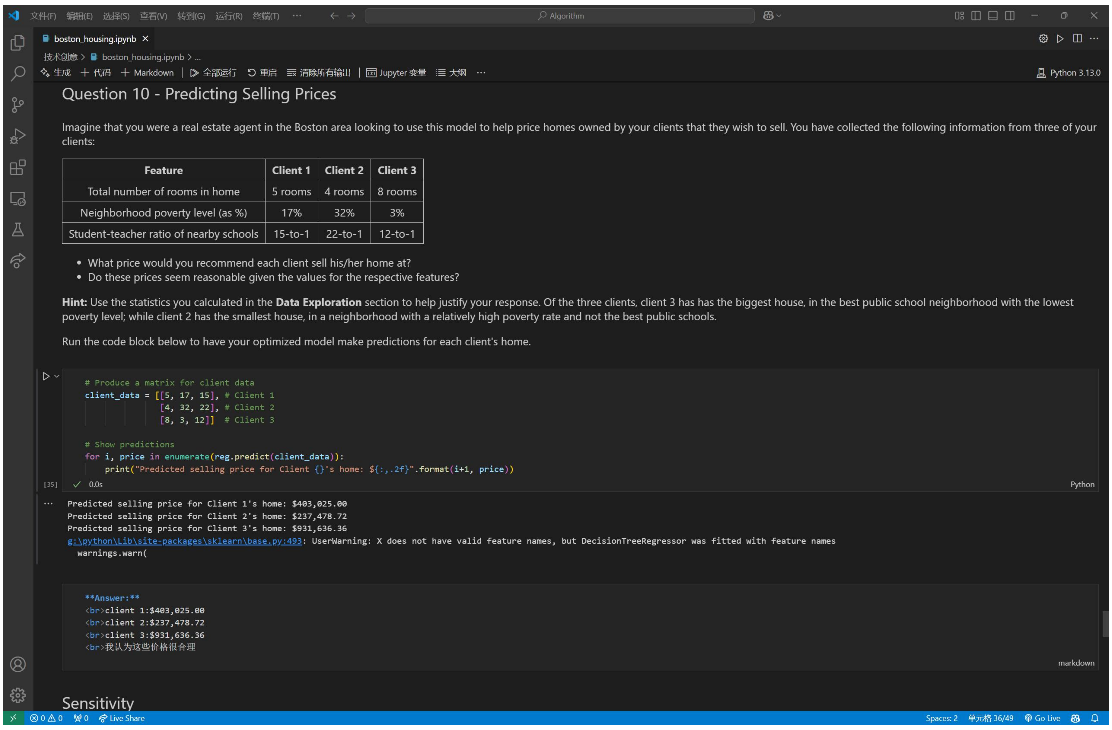
```
Parameter 'max_depth' is 4 for the optimal model.
```

Hint: The answer comes from the output of the code snipped above.

Answer: 4

Question 10 - Predicting Selling Prices

Imagine that you were a real estate agent in the Boston area looking to use this model to help price homes owned by your clients that they wish to sell. You have collected the following information from three of your clients:


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) ···
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown 全部运行 重启 清除所有输出 Jupyter 变量 大纲 ···
Python 3.13.0
Python
markdown
0 0 0 Live Share
Spaces: 2 单元格 36/49 Go Live

Question 10 - Predicting Selling Prices

Imagine that you were a real estate agent in the Boston area looking to use this model to help price homes owned by your clients that they wish to sell. You have collected the following information from three of your clients:

Feature	Client 1	Client 2	Client 3
Total number of rooms in home	5 rooms	4 rooms	8 rooms
Neighborhood poverty level (as %)	17%	32%	3%
Student-teacher ratio of nearby schools	15-to-1	22-to-1	12-to-1

- What price would you recommend each client sell his/her home at?
- Do these prices seem reasonable given the values for the respective features?

Hint: Use the statistics you calculated in the **Data Exploration** section to help justify your response. Of the three clients, client 3 has has the biggest house, in the best public school neighborhood with the lowest poverty level; while client 2 has the smallest house, in a neighborhood with a relatively high poverty rate and not the best public schools.

Run the code block below to have your optimized model make predictions for each client's home.

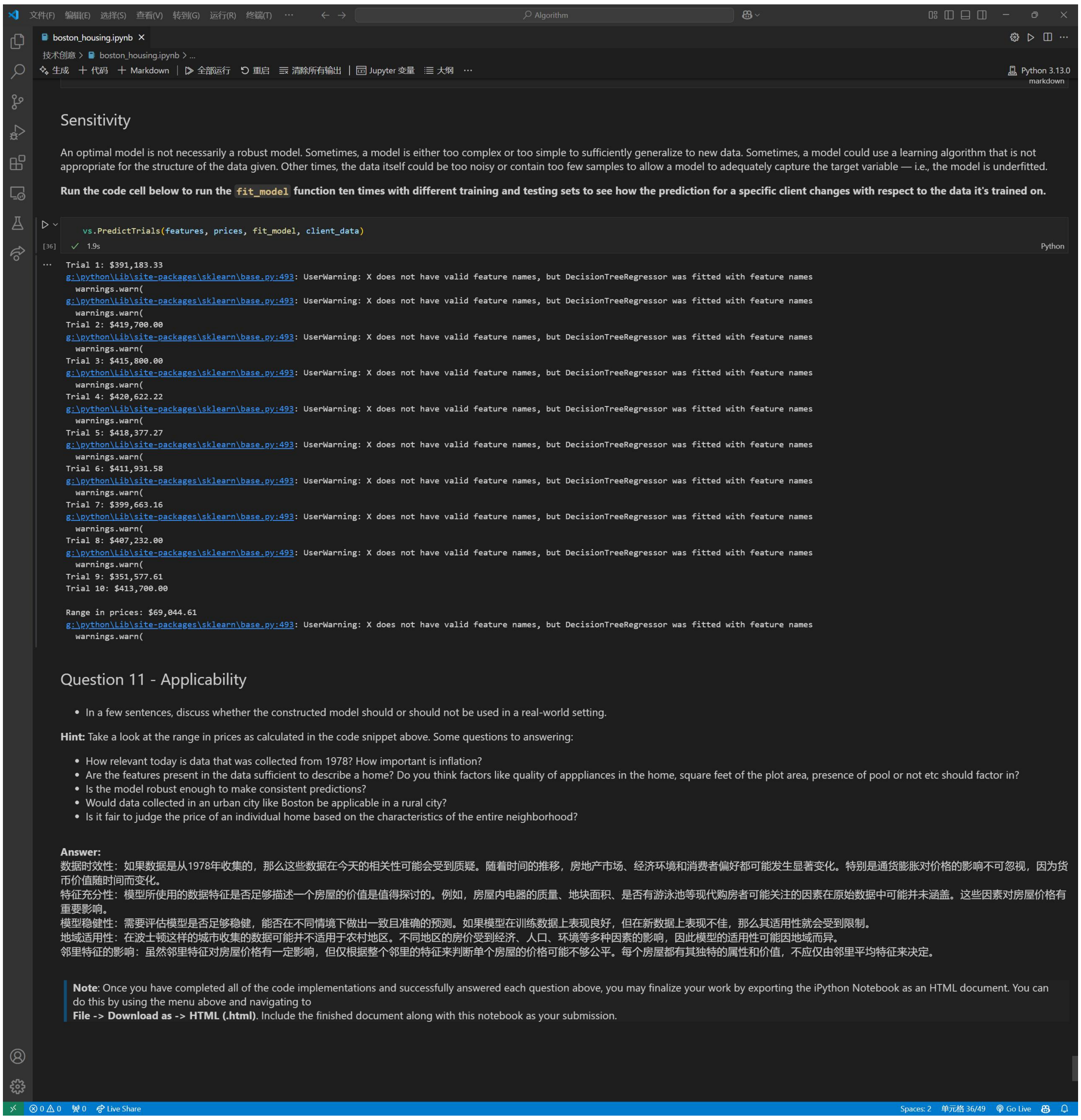
```
# Produce a matrix for client data
client_data = [[5, 17, 15], # Client 1
               [4, 32, 22], # Client 2
               [8, 3, 12]]  # Client 3

# Show predictions
for i, price in enumerate(reg.predict(client_data)):
    print("Predicted selling price for Client {}'s home: ${:,.2f}".format(i+1, price))
```

[35] ✓ 0.0s

```
Predicted selling price for Client 1's home: $403,025.00
Predicted selling price for Client 2's home: $237,478.72
Predicted selling price for Client 3's home: $931,636.36
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
```

```
**Answer:**
<br>client 1:$403,025.00
<br>client 2:$237,478.72
<br>client 3:$931,636.36
<br>我认为这些价格很合理
```

Sensitivity



Sensitivity

An optimal model is not necessarily a robust model. Sometimes, a model is either too complex or too simple to sufficiently generalize to new data. Sometimes, a model could use a learning algorithm that is not appropriate for the structure of the data given. Other times, the data itself could be too noisy or contain too few samples to allow a model to adequately capture the target variable — i.e., the model is underfitted.

Run the code cell below to run the `fit_model` function ten times with different training and testing sets to see how the prediction for a specific client changes with respect to the data it's trained on.

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

```
Trial 1: $391,183.33
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 2: $419,700.00
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 3: $415,800.00
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 4: $420,622.22
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 5: $418,377.27
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 6: $411,931.58
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 7: $399,663.16
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 8: $407,232.00
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 9: $351,577.61
Trial 10: $413,700.00

Range in prices: $69,044.61
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
```

Question 11 - Applicability

- In a few sentences, discuss whether the constructed model should or should not be used in a real-world setting.

Hint: Take a look at the range in prices as calculated in the code snippet above. Some questions to answering:

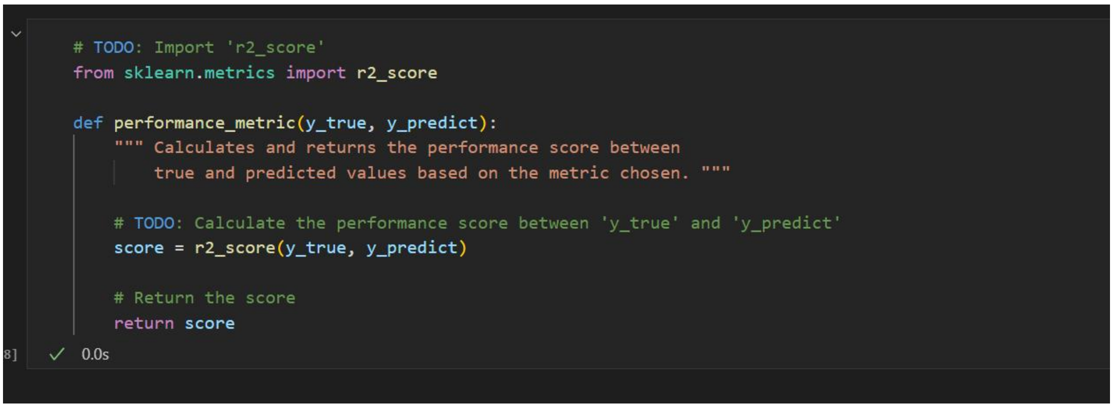
- How relevant today is data that was collected from 1978? How important is inflation?
- Are the features present in the data sufficient to describe a home? Do you think factors like quality of apppliances in the home, square feet of the plot area, presence of pool or not etc should factor in?
- Is the model robust enough to make consistent predictions?
- Would data collected in an urban city like Boston be applicable in a rural city?
- Is it fair to judge the price of an individual home based on the characteristics of the entire neighborhood?

Answer:
数据时效性：如果数据是从1978年收集的，那么这些数据在今天的相关性可能会受到质疑。随着时间的推移，房地产市场、经济环境和消费者偏好都可能发生显著变化。特别是通货膨胀对价格的影响不可忽视，因为货币价值随时间而变化。
特征充分性：模型所使用的数据特征是否足够描述一个房屋的价值是值得探讨的。例如，房屋内电器的质量、地块面积、是否有游泳池等现代购房者可能关注的因素在原始数据中可能并未涵盖。这些因素对房屋价格有重要影响。
模型稳健性：需要评估模型是否足够稳健，能否在不同情境下做出一致且准确的预测。如果模型在训练数据上表现良好，但在新数据上表现不佳，那么其适用性就会受到限制。
地域适用性：在波士顿这样的城市收集的数据可能并不适用于农村地区。不同地区的房价受到经济、人口、环境等多种因素的影响，因此模型的适用性可能因地域而异。
邻里特征的影响：虽然邻里特征对房屋价格有一定影响，但仅根据整个邻里的特征来判断单个房屋的价格可能不够公平。每个房屋都有其独特的属性和价值，不应仅由邻里平均特征来决定。

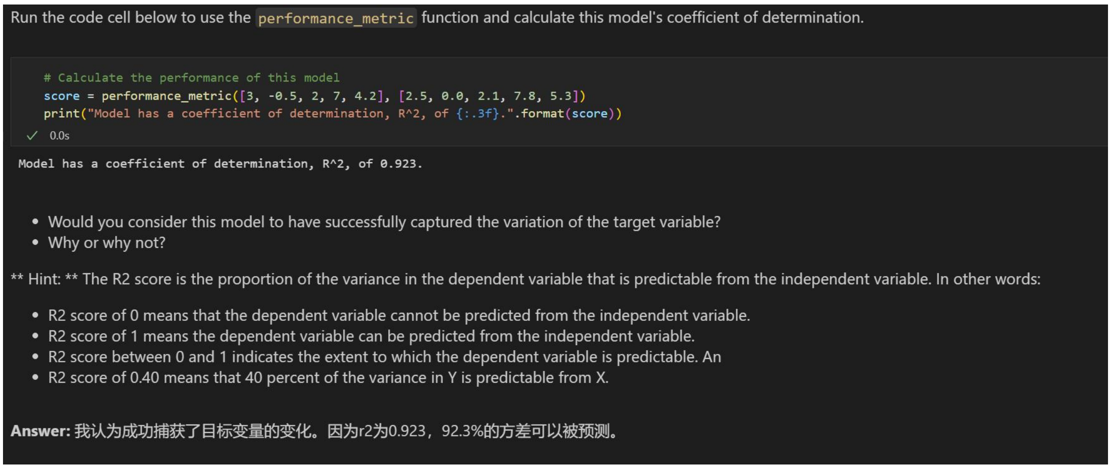
> **Note**: Once you have completed all of the code implementations and successfully answered each question above, you may finalize your work by exporting the iPython Notebook as an HTML document. You can do this by using the menu above and navigating to
> **File -> Download as -> HTML (.html)**. Include the finished document along with this notebook as your submission.

(4)实验结果及结果分析

同上



```
# TODO: Import 'r2_score'
from sklearn.metrics import r2_score

def performance_metric(y_true, y_predict):
    """ Calculates and returns the performance score between
        true and predicted values based on the metric chosen. """

    # TODO: Calculate the performance score between 'y_true' and 'y_predict'
    score = r2_score(y_true, y_predict)

    # Return the score
    return score
```



Run the code cell below to use the `performance_metric` function and calculate this model's coefficient of determination.

```
# Calculate the performance of this model
score = performance_metric([3, -0.5, 2, 7, 4.2], [2.5, 0.0, 2.1, 7.8, 5.3])
print("Model has a coefficient of determination, R^2, of {:.3f}.".format(score))
```

✓ 0.0s

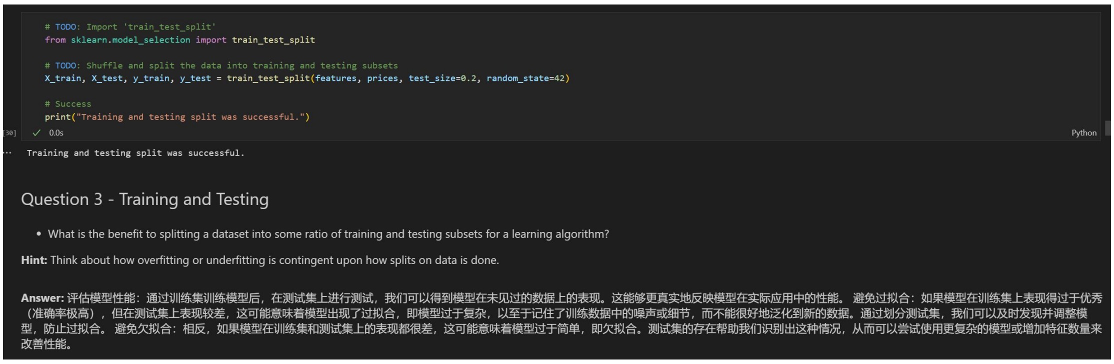
```
Model has a coefficient of determination, R^2, of 0.923.
```

- Would you consider this model to have successfully captured the variation of the target variable?
- Why or why not?

** Hint: ** The R2 score is the proportion of the variance in the dependent variable that is predictable from the independent variable. In other words:

- R2 score of 0 means that the dependent variable cannot be predicted from the independent variable.
- R2 score of 1 means the dependent variable can be predicted from the independent variable.
- R2 score between 0 and 1 indicates the extent to which the dependent variable is predictable. An
- R2 score of 0.40 means that 40 percent of the variance in Y is predictable from X.

Answer: 我认为成功捕获了目标变量的变化。因为r2为0.923，92.3%的方差可以被预测。



```
# TODO: Import 'train_test_split'
from sklearn.model_selection import train_test_split

# TODO: Shuffle and split the data into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(features, prices, test_size=0.2, random_state=42)

# Success
print("Training and testing split was successful.")
```

[30] ✓ 0.0s Python

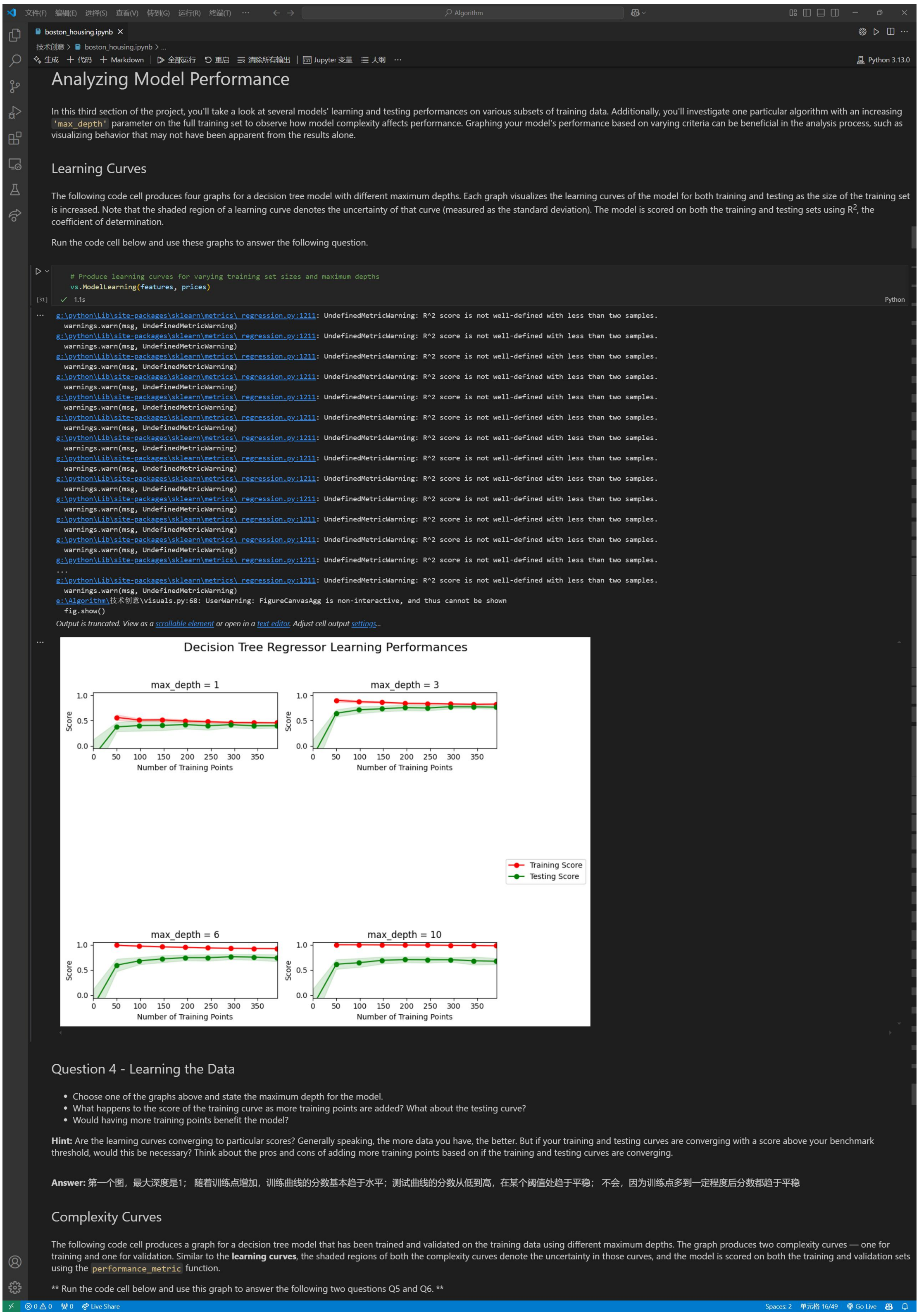
```
Training and testing split was successful.
```

Question 3 - Training and Testing

- What is the benefit to splitting a dataset into some ratio of training and testing subsets for a learning algorithm?

Hint: Think about how overfitting or underfitting is contingent upon how splits on data is done.

Answer: 评估模型性能：通过训练集训练模型后，在测试集上进行测试，我们可以得到模型在未见过的数据上的表现。这能够更真实地反映模型在实际应用中的性能。 避免过拟合：如果模型在训练集上表现得过于优秀（准确率极高），但在测试集上表现较差，这可能意味着模型出现了过拟合，即模型过于复杂，以至于记住了训练数据中的噪声或细节，而不能很好地泛化到新的数据。通过划分测试集，我们可以及时发现并调整模型，防止过拟合。 避免欠拟合：相反，如果模型在训练集和测试集上的表现都很差，这可能意味着模型过于简单，即欠拟合。测试集的存在帮助我们识别出这种情况，从而可以尝试使用更复杂的模型或增加特征数量来改善性能。



Analyzing Model Performance

In this third section of the project, you'll take a look at several models' learning and testing performances on various subsets of training data. Additionally, you'll investigate one particular algorithm with an increasing `'max_depth'` parameter on the full training set to observe how model complexity affects performance. Graphing your model's performance based on varying criteria can be beneficial in the analysis process, such as visualizing behavior that may not have been apparent from the results alone.

Learning Curves

The following code cell produces four graphs for a decision tree model with different maximum depths. Each graph visualizes the learning curves of the model for both training and testing as the size of the training set is increased. Note that the shaded region of a learning curve denotes the uncertainty of that curve (measured as the standard deviation). The model is scored on both the training and testing sets using R^2, the coefficient of determination.

Run the code cell below and use these graphs to answer the following question.

```python
# Produce learning curves for varying training set sizes and maximum depths
vs.ModelLearning(features, prices)
```

[31] ✓ 1.1s Python

```
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
...
g:\python\Lib\site-packages\sklearn\metrics\_regression.py:1211: UndefinedMetricWarning: R^2 score is not well-defined with less than two samples.
  warnings.warn(msg, UndefinedMetricWarning)
e:\Algorithm\技术创意\visuals.py:68: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
  fig.show()
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Question 4 - Learning the Data

- Choose one of the graphs above and state the maximum depth for the model.
- What happens to the score of the training curve as more training points are added? What about the testing curve?
- Would having more training points benefit the model?

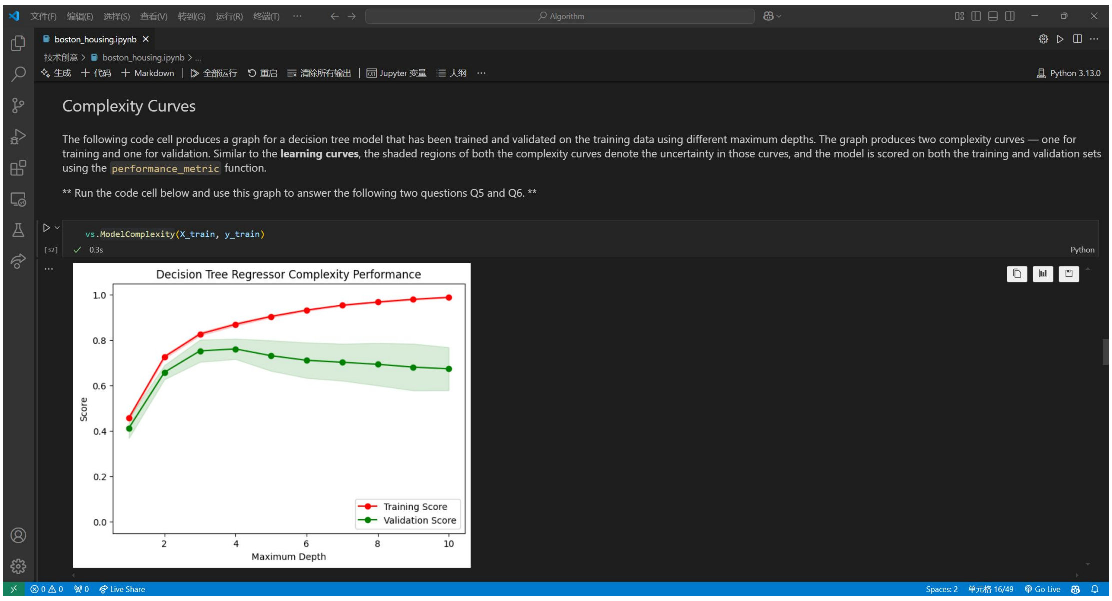
Hint: Are the learning curves converging to particular scores? Generally speaking, the more data you have, the better. But if your training and testing curves are converging with a score above your benchmark threshold, would this be necessary? Think about the pros and cons of adding more training points based on if the training and testing curves are converging.

Answer: 第一个图，最大深度是1； 随着训练点增加，训练曲线的分数基本趋于水平；测试曲线的分数从低到高，在某个阈值处趋于平稳； 不会，因为训练点多到一定程度后分数都趋于平稳

Complexity Curves

The following code cell produces a graph for a decision tree model that has been trained and validated on the training data using different maximum depths. The graph produces two complexity curves — one for training and one for validation. Similar to the **learning curves**, the shaded regions of both the complexity curves denote the uncertainty in those curves, and the model is scored on both the training and validation sets using the `performance_metric` function.

** Run the code cell below and use this graph to answer the following two questions Q5 and Q6. **


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) …
Algorithm
boston_housing.ipynb
技术创新 > boston_housing.ipynb > ...
生成 + 代码 + Markdown | 全部运行 重启 清除所有输出 | Jupyter 变量 大纲 …
Python 3.13.0
Decision Tree Regressor Complexity Performance
Score
Maximum Depth
1.0
0.8
0.6
0.4
0.2
0.0
2
4
6
8
10
Training Score
Validation Score
Spaces: 2 单元格 16/49 Go Live
Live Share

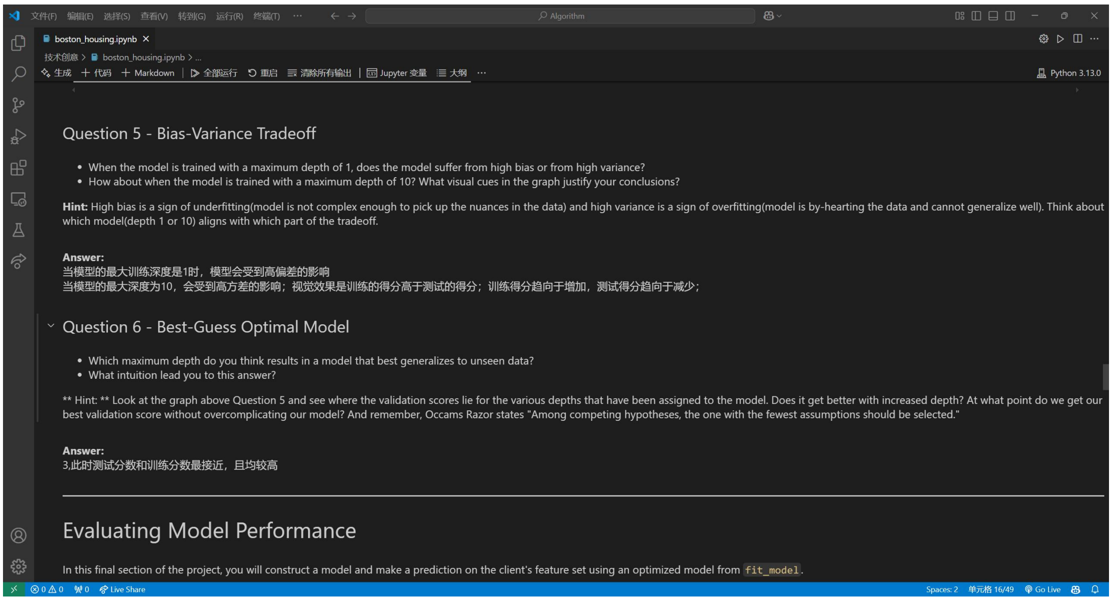
Complexity Curves

The following code cell produces a graph for a decision tree model that has been trained and validated on the training data using different maximum depths. The graph produces two complexity curves — one for training and one for validation. Similar to the **learning curves**, the shaded regions of both the complexity curves denote the uncertainty in those curves, and the model is scored on both the training and validation sets using the `performance_metric` function.

** Run the code cell below and use this graph to answer the following two questions Q5 and Q6. **

```
[32]
vs.ModelComplexity(X_train, y_train)
```

✓ 0.3s


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) …
Algorithm
boston_housing.ipynb
技术创新 > boston_housing.ipynb > ...
生成 + 代码 + Markdown | 全部运行 重启 清除所有输出 | Jupyter 变量 大纲 …
Python 3.13.0
Spaces: 2 单元格 16/49 Go Live
Live Share

Question 5 - Bias-Variance Tradeoff

- When the model is trained with a maximum depth of 1, does the model suffer from high bias or from high variance?
- How about when the model is trained with a maximum depth of 10? What visual cues in the graph justify your conclusions?

Hint: High bias is a sign of underfitting(model is not complex enough to pick up the nuances in the data) and high variance is a sign of overfitting(model is by-hearting the data and cannot generalize well). Think about which model(depth 1 or 10) aligns with which part of the tradeoff.

Answer:
当模型的最大训练深度是1时，模型会受到高偏差的影响
当模型的最大深度为10，会受到高方差的影响；视觉效果是训练的得分高于测试的得分；训练得分趋向于增加，测试得分趋向于减少；

Question 6 - Best-Guess Optimal Model

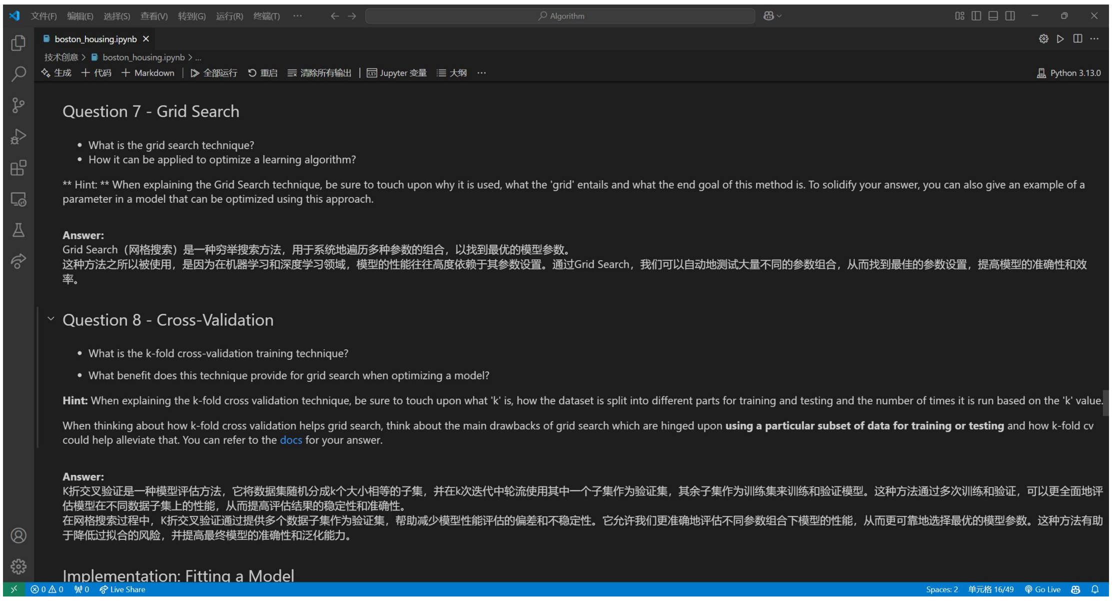
- Which maximum depth do you think results in a model that best generalizes to unseen data?
- What intuition lead you to this answer?

** Hint: ** Look at the graph above Question 5 and see where the validation scores lie for the various depths that have been assigned to the model. Does it get better with increased depth? At what point do we get our best validation score without overcomplicating our model? And remember, Occams Razor states "Among competing hypotheses, the one with the fewest assumptions should be selected."

Answer:
3,此时测试分数和训练分数最接近，且均较高

Evaluating Model Performance

In this final section of the project, you will construct a model and make a prediction on the client's feature set using an optimized model from `fit_model`.


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) …
Algorithm
boston_housing.ipynb
技术创新 > boston_housing.ipynb > ...
生成 + 代码 + Markdown | 全部运行 重启 清除所有输出 | Jupyter 变量 大纲 …
Python 3.13.0
Spaces: 2 单元格 16/49 Go Live
Live Share

Question 7 - Grid Search

- What is the grid search technique?
- How it can be applied to optimize a learning algorithm?

** Hint: ** When explaining the Grid Search technique, be sure to touch upon why it is used, what the 'grid' entails and what the end goal of this method is. To solidify your answer, you can also give an example of a parameter in a model that can be optimized using this approach.

Answer:
Grid Search（网格搜索）是一种穷举搜索方法，用于系统地遍历多种参数的组合，以找到最优的模型参数。
这种方法之所以被使用，是因为在机器学习和深度学习领域，模型的性能往往高度依赖于其参数设置。通过Grid Search，我们可以自动地测试大量不同的参数组合，从而找到最佳的参数设置，提高模型的准确性和效率。

Question 8 - Cross-Validation

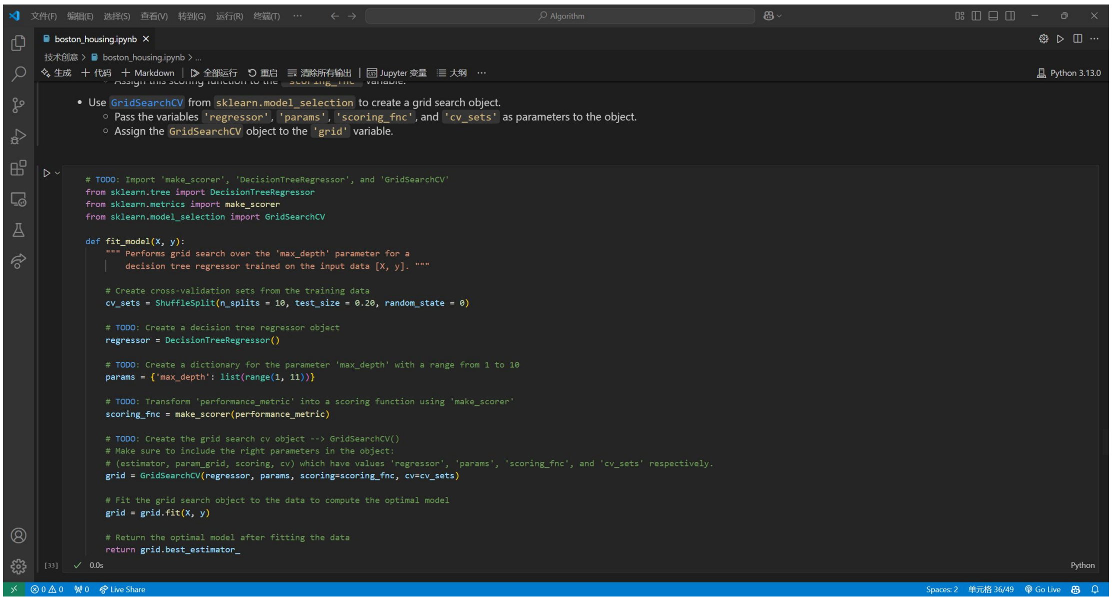
- What is the k-fold cross-validation training technique?
- What benefit does this technique provide for grid search when optimizing a model?

Hint: When explaining the k-fold cross validation technique, be sure to touch upon what 'k' is, how the dataset is split into different parts for training and testing and the number of times it is run based on the 'k' value.

When thinking about how k-fold cross validation helps grid search, think about the main drawbacks of grid search which are hinged upon **using a particular subset of data for training or testing** and how k-fold cv could help alleviate that. You can refer to the docs for your answer.

Answer:
K折交叉验证是一种模型评估方法，它将数据集随机分成k个大小相等的子集，并在k次迭代中轮流使用其中一个子集作为验证集，其余子集作为训练集来训练和验证模型。这种方法通过多次训练和验证，可以更全面地评估模型在不同数据子集上的性能，从而提高评估结果的稳定性和准确性。
在网格搜索过程中，K折交叉验证通过提供多个数据子集作为验证集，帮助减少模型性能评估的偏差和不稳定性。它允许我们更准确地评估不同参数组合下模型的性能，从而更可靠地选择最优的模型参数。这种方法有助于降低过拟合的风险，并提高最终模型的准确性和泛化能力。

Implementation: Fitting a Model

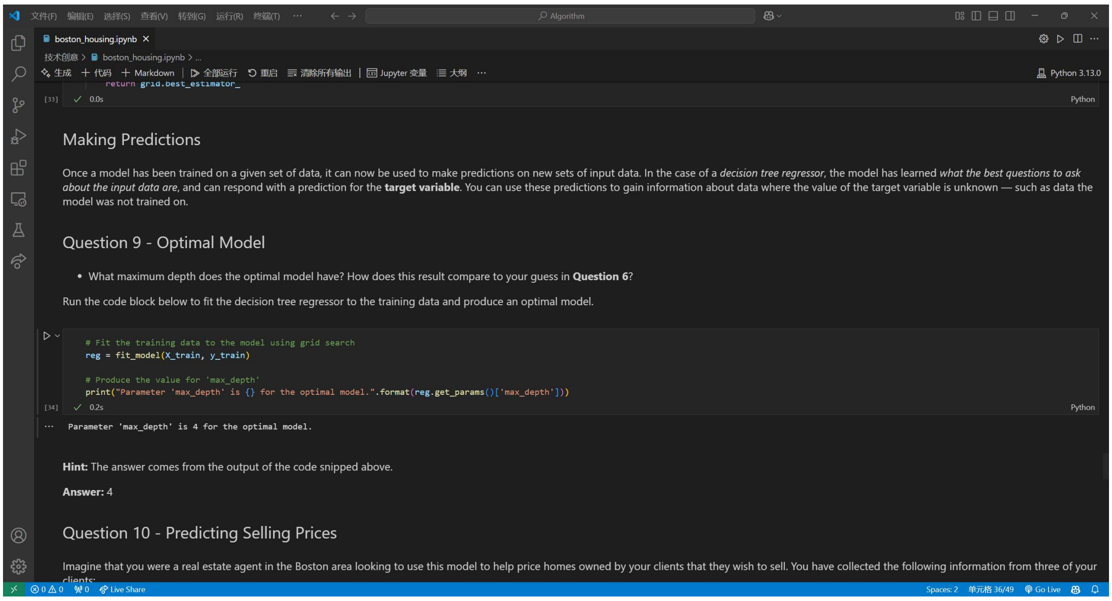

文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T)
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown 全部运行 重启 清除所有输出 Jupyter 变量 大纲
Python 3.13.0

- Assign this scoring function to the `'scoring_fnc'` variable.
- Use `GridSearchCV` from `sklearn.model_selection` to create a grid search object.
  - Pass the variables `'regressor'`, `'params'`, `'scoring_fnc'`, and `'cv_sets'` as parameters to the object.
  - Assign the `GridSearchCV` object to the `'grid'` variable.

```python
# TODO: Import 'make_scorer', 'DecisionTreeRegressor', and 'GridSearchCV'
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import make_scorer
from sklearn.model_selection import GridSearchCV

def fit_model(X, y):
    """ Performs grid search over the 'max_depth' parameter for a
        decision tree regressor trained on the input data [X, y]. """

    # Create cross-validation sets from the training data
    cv_sets = ShuffleSplit(n_splits = 10, test_size = 0.20, random_state = 0)

    # TODO: Create a decision tree regressor object
    regressor = DecisionTreeRegressor()

    # TODO: Create a dictionary for the parameter 'max_depth' with a range from 1 to 10
    params = {'max_depth': list(range(1, 11))}

    # TODO: Transform 'performance_metric' into a scoring function using 'make_scorer'
    scoring_fnc = make_scorer(performance_metric)

    # TODO: Create the grid search cv object --> GridSearchCV()
    # Make sure to include the right parameters in the object:
    # (estimator, param_grid, scoring, cv) which have values 'regressor', 'params', 'scoring_fnc', and 'cv_sets' respectively.
    grid = GridSearchCV(regressor, params, scoring=scoring_fnc, cv=cv_sets)

    # Fit the grid search object to the data to compute the optimal model
    grid = grid.fit(X, y)

    # Return the optimal model after fitting the data
    return grid.best_estimator_
```

[33] ✓ 0.0s Python

⊗ 0 ⚠ 0 0 Live Share
Spaces: 2 单元格 36/49 Go Live


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T)
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown 全部运行 重启 清除所有输出 Jupyter 变量 大纲
Python 3.13.0

```python
    return grid.best_estimator_
```

[33] ✓ 0.0s Python

Making Predictions

Once a model has been trained on a given set of data, it can now be used to make predictions on new sets of input data. In the case of a *decision tree regressor*, the model has learned *what the best questions to ask about the input data are*, and can respond with a prediction for the **target variable**. You can use these predictions to gain information about data where the value of the target variable is unknown — such as data the model was not trained on.

Question 9 - Optimal Model

- What maximum depth does the optimal model have? How does this result compare to your guess in **Question 6**?

Run the code block below to fit the decision tree regressor to the training data and produce an optimal model.

```python
# Fit the training data to the model using grid search
reg = fit_model(X_train, y_train)

# Produce the value for 'max_depth'
print("Parameter 'max_depth' is {} for the optimal model.".format(reg.get_params()['max_depth']))
```

[34] ✓ 0.2s Python

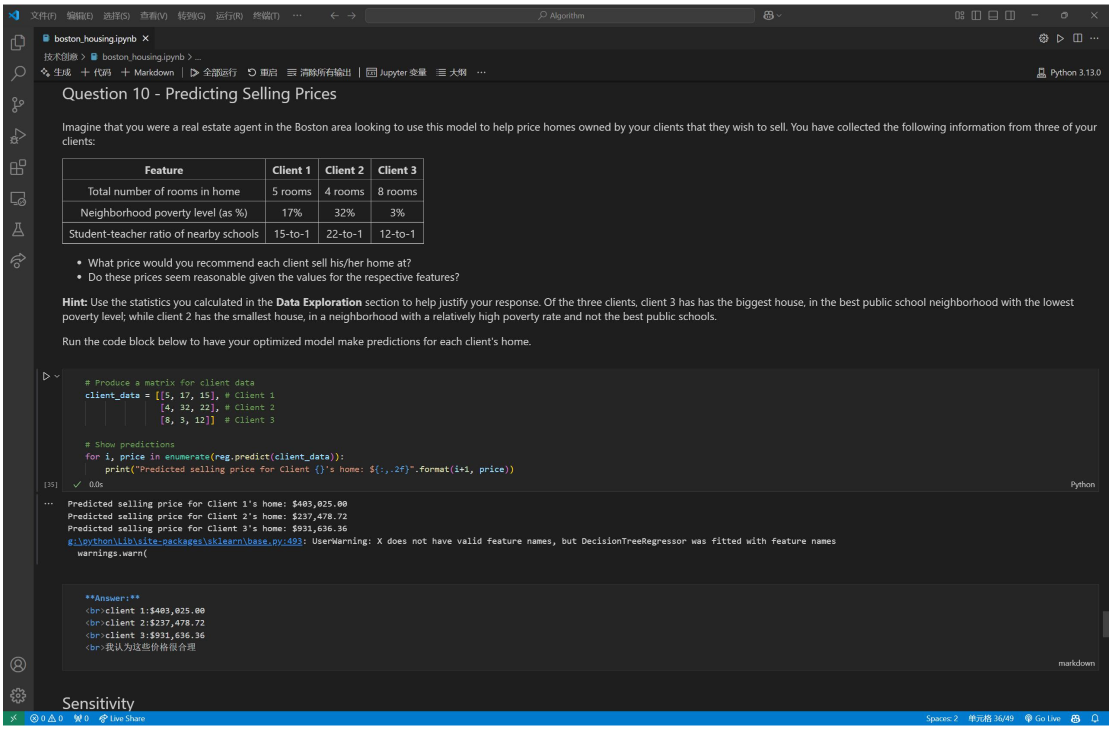
```
Parameter 'max_depth' is 4 for the optimal model.
```

Hint: The answer comes from the output of the code snipped above.

Answer: 4

Question 10 - Predicting Selling Prices

Imagine that you were a real estate agent in the Boston area looking to use this model to help price homes owned by your clients that they wish to sell. You have collected the following information from three of your clients:



Question 10 - Predicting Selling Prices

Imagine that you were a real estate agent in the Boston area looking to use this model to help price homes owned by your clients that they wish to sell. You have collected the following information from three of your clients:

Feature	Client 1	Client 2	Client 3
Total number of rooms in home	5 rooms	4 rooms	8 rooms
Neighborhood poverty level (as %)	17%	32%	3%
Student-teacher ratio of nearby schools	15-to-1	22-to-1	12-to-1

- What price would you recommend each client sell his/her home at?
- Do these prices seem reasonable given the values for the respective features?

Hint: Use the statistics you calculated in the **Data Exploration** section to help justify your response. Of the three clients, client 3 has has the biggest house, in the best public school neighborhood with the lowest poverty level; while client 2 has the smallest house, in a neighborhood with a relatively high poverty rate and not the best public schools.

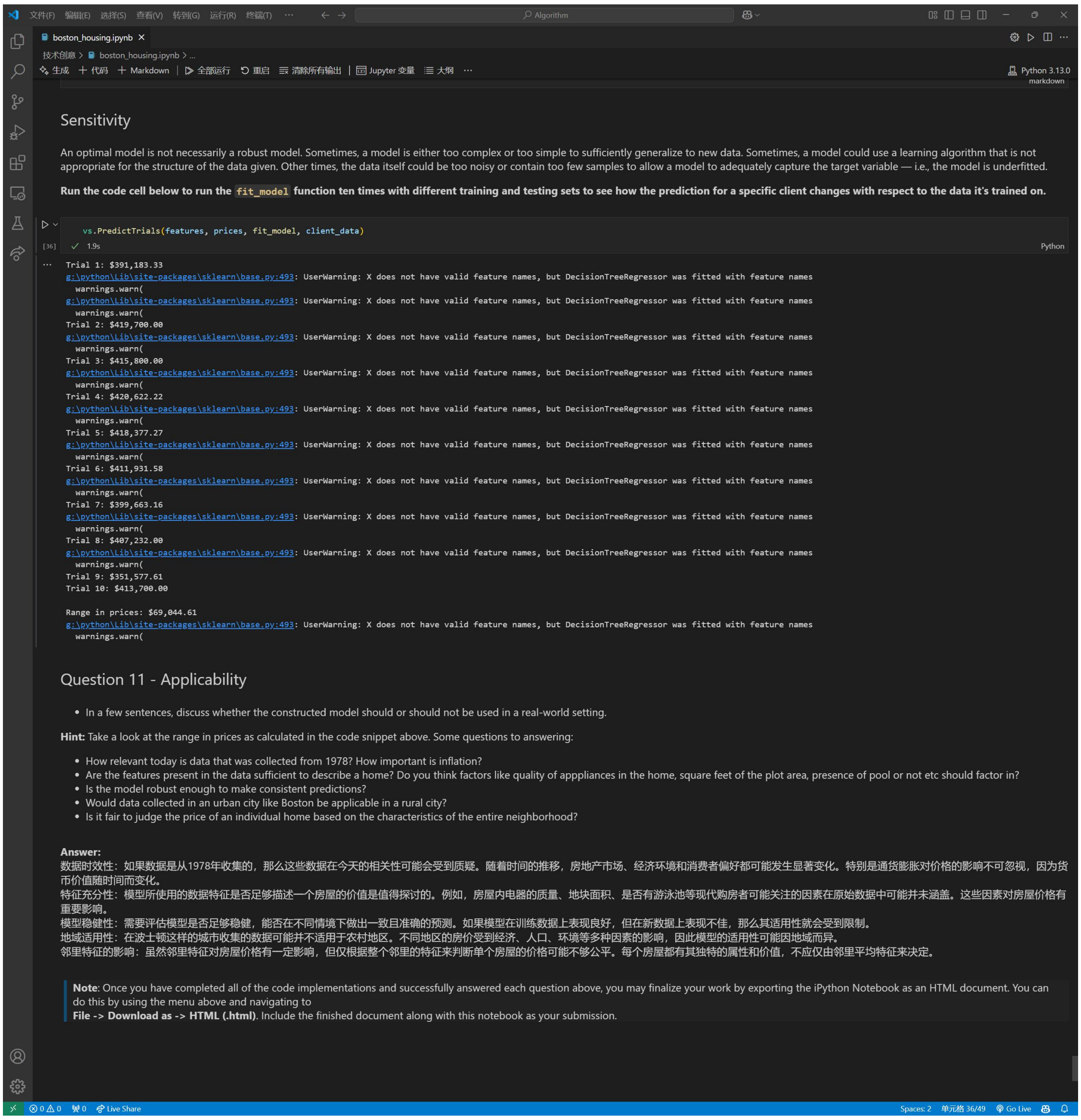
Run the code block below to have your optimized model make predictions for each client's home.

```python
# Produce a matrix for client data
client_data = [[5, 17, 15], # Client 1
               [4, 32, 22], # Client 2
               [8, 3, 12]]  # Client 3

# Show predictions
for i, price in enumerate(reg.predict(client_data)):
    print("Predicted selling price for Client {}'s home: ${:,.2f}".format(i+1, price))
```

```
Predicted selling price for Client 1's home: $403,025.00
Predicted selling price for Client 2's home: $237,478.72
Predicted selling price for Client 3's home: $931,636.36
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
```

```
**Answer:**
<br>client 1:$403,025.00
<br>client 2:$237,478.72
<br>client 3:$931,636.36
<br>我认为这些价格很合理
```

Sensitivity


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) ···
Algorithm
boston_housing.ipynb
技术创意 > boston_housing.ipynb > ...
生成 + 代码 + Markdown | 全部运行 重启 清除所有输出 | Jupyter 变量 大纲 ···
Python 3.13.0
markdown

Sensitivity

An optimal model is not necessarily a robust model. Sometimes, a model is either too complex or too simple to sufficiently generalize to new data. Sometimes, a model could use a learning algorithm that is not appropriate for the structure of the data given. Other times, the data itself could be too noisy or contain too few samples to allow a model to adequately capture the target variable — i.e., the model is underfitted.

Run the code cell below to run the `fit_model` function ten times with different training and testing sets to see how the prediction for a specific client changes with respect to the data it's trained on.

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

```
Trial 1: $391,183.33
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 2: $419,700.00
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 3: $415,800.00
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 4: $420,622.22
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 5: $418,377.27
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 6: $411,931.58
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 7: $399,663.16
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 8: $407,232.00
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
Trial 9: $351,577.61
Trial 10: $413,700.00

Range in prices: $69,044.61
g:\python\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was fitted with feature names
  warnings.warn(
```

Question 11 - Applicability

- In a few sentences, discuss whether the constructed model should or should not be used in a real-world setting.

Hint: Take a look at the range in prices as calculated in the code snippet above. Some questions to answering:

- How relevant today is data that was collected from 1978? How important is inflation?
- Are the features present in the data sufficient to describe a home? Do you think factors like quality of apppliances in the home, square feet of the plot area, presence of pool or not etc should factor in?
- Is the model robust enough to make consistent predictions?
- Would data collected in an urban city like Boston be applicable in a rural city?
- Is it fair to judge the price of an individual home based on the characteristics of the entire neighborhood?

Answer:
数据时效性：如果数据是从1978年收集的，那么这些数据在今天的相关性可能会受到质疑。随着时间的推移，房地产市场、经济环境和消费者偏好都可能发生显著变化。特别是通货膨胀对价格的影响不可忽视，因为货币价值随时间而变化。
特征充分性：模型所使用的数据特征是否足够描述一个房屋的价值是值得探讨的。例如，房屋内电器的质量、地块面积、是否有游泳池等现代购房者可能关注的因素在原始数据中可能并未涵盖。这些因素对房屋价格有重要影响。
模型稳健性：需要评估模型是否足够稳健，能否在不同情境下做出一致且准确的预测。如果模型在训练数据上表现良好，但在新数据上表现不佳，那么其适用性就会受到限制。
地域适用性：在波士顿这样的城市收集的数据可能并不适用于农村地区。不同地区的房价受到经济、人口、环境等多种因素的影响，因此模型的适用性可能因地域而异。
邻里特征的影响：虽然邻里特征对房屋价格有一定影响，但仅根据整个邻里的特征来判断单个房屋的价格可能不够公平。每个房屋都有其独特的属性和价值，不应仅由邻里平均特征来决定。

> **Note**: Once you have completed all of the code implementations and successfully answered each question above, you may finalize your work by exporting the iPython Notebook as an HTML document. You can do this by using the menu above and navigating to
> **File -> Download as -> HTML (.html)**. Include the finished document along with this notebook as your submission.

(5)实验总结

了解如何对数据集 *fit* 和 *evaluate*；初识机器学习，了解了一些机器学习方法；