

机器学习工程师纳米学位

模型评价与验证

项目 1: 预测波士顿房价

欢迎来到机器学习工程师纳米学位的第一个项目！在此文件中，有些示例代码已经提供给你，但你还需要实现更多的功能来让项目成功运行。除非有明确要求，你无须修改任何已给出的代码。以'练习'开始的标题表示接下来的内容中有需要你实现的功能。每一部分都会有详细的指导，需要实现的部分也会在注释中以'**TODO**'标出。请仔细阅读所有的提示！

除了实现代码外，你还必须回答一些与项目和实现有关的问题。每一个需要你回答的问题都会以'**问题 X**'为标题。请仔细阅读每个问题，并且在问题后的'**回答**'文字框中写出完整的答案。你的项目将会根据你对问题的回答和撰写代码所实现的功能来进行评分。

提示：Code 和 Markdown 区域可通过 **Shift + Enter** 快捷键运行。此外，Markdown可以通过双击进入编辑模式。

开始

在这个项目中，你将利用马萨诸塞州波士顿郊区的房屋信息数据训练和测试一个模型，并对模型的性能和预测能力进行测试。通过该数据训练后的好的模型可以被用来对房屋做特定预测---尤其是对房屋的价值。对于房地产经纪等人的日常工作来说，这样的预测模型被证明非常有价值。

此项目的数据集来自[UCI机器学习知识库 \(https://archive.ics.uci.edu/ml/datasets/Housing\)](https://archive.ics.uci.edu/ml/datasets/Housing)。波士顿房屋这些数据于1978年开始统计，共506个数据点，涵盖了麻省波士顿不同郊区房屋14种特征的信息。本项目对原始数据集做了以下处理：

- 有16个'MEDV' 值为50.0的数据点被移除。这很可能是由于这些数据点包含遗失或看不到的值。
- 有1个数据点的 'RM' 值为8.78. 这是一个异常值，已经被移除。
- 对于本项目，房屋的 'RM'， 'LSTAT'， 'PTRATIO' 以及 'MEDV' 特征是必要的，其余不相关特征已经被移除。
- 'MEDV' 特征的值已经过必要的数学转换，可以反映35年来市场的通货膨胀效应。

运行下面区域的代码以载入波士顿房屋数据集，以及一些此项目所需的Python库。如果成功返回数据集的大小，表示数据集已载入成功。

In [1]:

```
# Import libraries necessary for this project
# 载入此项目所需要的库
import numpy as np
import pandas as pd
import visuals as vs # Supplementary code
from sklearn.cross_validation import ShuffleSplit

# Pretty display for notebooks
# 让结果在notebook中显示
%matplotlib inline

# Load the Boston housing dataset
# 载入波士顿房屋的数据集
data = pd.read_csv('housing.csv')
prices = data['MEDV']
features = data.drop('MEDV', axis = 1)

# Success
# 完成
print "Boston housing dataset has {} data points with {} variables each.".format(*d
```

Boston housing dataset has 489 data points with 4 variables each.

分析数据

在项目的第一个部分，你会对波士顿房地产数据进行初步的观察并给出你的分析。通过对数据的探索来熟悉数据可以让你更好地理解 and 解释你的结果。

由于这个项目的最终目标是建立一个预测房屋价值的模型，我们需要将数据集分为特征(**features**)和目标变量(**target variable**)。特征 'RM', 'LSTAT', 和 'PTRATIO', 给我们提供了每个数据点的数量相关的信息。目标变量: 'MEDV', 是我们希望预测的变量。他们分别被存在features和prices两个变量名中。

练习：基础统计运算

你的第一个编程练习是计算有关波士顿房价的描述统计数据。我们已为你导入了numpy，你需要使用这个库来执行必要的计算。这些统计数据对于分析模型的预测结果非常重要的。 在下面的代码中，你要做的是：

- 计算prices中的'MEDV'的最小值、最大值、均值、中值和标准差；
- 将运算结果储存在相应的变量中。

In [2]:

```
# TODO: Minimum price of the data
#目标: 计算价值的最小值
minimum_price = np.min(prices)

# TODO: Maximum price of the data
#目标: 计算价值的最大值
maximum_price = np.max(prices)

# TODO: Mean price of the data
#目标: 计算价值的平均值
mean_price = np.mean(prices)

# TODO: Median price of the data
#目标: 计算价值的中值
median_price = np.median(prices)

# TODO: Standard deviation of prices of the data
#目标: 计算价值的标准差
std_price = np.std(prices)

# Show the calculated statistics
#目标: 输出计算的结果
print "Statistics for Boston housing dataset:\n"
print "Minimum price: ${:,.2f}".format(minimum_price)
print "Maximum price: ${:,.2f}".format(maximum_price)
print "Mean price: ${:,.2f}".format(mean_price)
print "Median price ${:,.2f}".format(median_price)
print "Standard deviation of prices: ${:,.2f}".format(std_price)
```

Statistics for Boston housing dataset:

Minimum price: \$105,000.00
Maximum price: \$1,024,800.00
Mean price: \$454,342.94
Median price \$438,900.00
Standard deviation of prices: \$165,171.13

问题1 - 特征观察

如前文所述，本项目中我们关注的是其中三个值: 'RM'、'LSTAT' 和 'PTRATIO'，对每一个数据点:

- 'RM' 是该地区中每个房屋的平均房间数量；
- 'LSTAT' 是指该地区有多少百分比的房东属于是低收入阶层（有工作但收入微薄）；
- 'PTRATIO' 是该地区的中学和小学里，学生和老师的数目比（学生/老师）。

凭直觉，上述三个特征中对每一个来说，你认为增大该特征的数值，'MEDV'的值会是增大还是减小呢？每一个答案都需要你给出理由。

提示：你预期一个 'RM' 值是6的房屋跟 'RM' 值是7的房屋相比，价值更高还是更低呢？

回答:

- 观察数据列 'RM' 后发现随着该地区中每个房屋的平均房间数量的增大，价格是增大的，理论上价格个房间总面积有直接的关系，同一维度上，总面积越大，价格会增大，房间数量的增多直观的反应就是面积有所增大，'MEDV' 的值会相应增大

- 观察数据列 'LSTAT' 后发现随着该地区低收入阶层的房东的比例增大（按照10%为一个阶梯来看），'MEDV' 的值会是减小的，低收入阶层的比例一定程度上反映了当地的购买能力，而购买能力弱的客户，买房子的欲望会更加淡化，当地的房子会更不好销售，也促使房子的 'MEDV' 的值减小
- 观察数据列 'PTRATIO' 后发现随着该地区中学和小学里学生和老师的数目比的增大，'MEDV' 的值是减小的，学生/老师的比例一定程度反应这里的师资水平，师资水平相应可以代表这个地区的发达程度，学生和老师的数目比大，意味着师资力量相对较弱，因此 'MEDV' 的值是减小的

建模

在项目的第二部分中，你需要了解必要的工具和技巧来让你的模型进行预测。用这些工具和技巧对每一个模型的表现做精确的衡量可以极大地增强你预测的信心。

练习：定义衡量标准

如果不能对模型的训练和测试的表现进行量化地评估，我们就很难衡量模型的好坏。通常我们会定义一些衡量标准，这些标准可以通过对某些误差或者拟合程度的计算来得到。在这个项目中，你将通过运算 决定系数 (http://stattrek.com/statistics/dictionary.aspx?definition=coefficient_of_determination) R^2 来量化模型的表现。模型的决定系数是回归分析中十分常用的统计信息，经常被当作衡量模型预测能力好坏的标准。

R^2 的数值范围从0至1，表示目标变量的预测值和实际值之间的相关程度平方的百分比。一个模型的 R^2 值为0说明它完全无法预测目标变量；而一个 R^2 值为1的模型则可以对目标变量进行完美的预测。从0至1之间的数值，则表示该模型中目标变量中有百分之多少能够用特征来解释。_模型也可能出现负值的 R^2 ，这种情况下模型所做预测还不如直接计算目标变量的平均值。

在下方代码的 `performance_metric` 函数中，你要实现：

- 使用 `sklearn.metrics` 中的 `r2_score` 来计算 `y_true` 和 `y_predict` 的 R^2 值，作为对其表现的评判。
- 将他们的表现评分储存到 `score` 变量中。

In [3]:

```
# 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
```

问题2 - 拟合程度

假设一个数据集有五个数据且一个模型做出下列目标变量的预测：

真实数值	预测数值
3.0	2.5
-0.5	0.0

2.0	2.1
7.0	7.8
4.2	5.3

你会觉得这个模型已成功地描述了目标变量的变化吗？如果成功，请解释为什么，如果没有，也请给出原因。

运行下方的代码，使用performance_metric函数来计算模型的决定系数。

In [4]:

```
# 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)
```

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

回答: 我认为已经成功的描述了目标的变化，R^2基本趋近于1，按照之前的提示对目标变量已经进行了相对好的预测

练习: 数据分割与重排

接下来，你需要把波士顿房屋数据集分成训练和测试两个子集。通常在这个过程中，数据也会被重新排序，以消除数据集中由于排序而产生的偏差。在下面的代码中，你需要：

- 使用 sklearn.cross_validation 中的 train_test_split，将features和prices的数据都分成用于训练的数据子集和用于测试的数据子集。
 - 分割比例为：80%的数据用于训练，20%用于测试；
 - 选定一个数值以设定 train_test_split 中的 random_state，这会确保结果的一致性；
- 最终分离出的子集为X_train,X_test,y_train,和y_test。

In [5]:

```
# TODO: Import 'train_test_split'
from sklearn.cross_validation 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)

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

Training and testing split was successful.

问题 3- 训练及测试

将数据集按一定比例分为训练用的数据集和测试用的数据集对学习算法有什么好处？

提示：如果没有数据来对模型进行测试，会出现什么问题？

答案:

- 如果没有数据来对模型进行测试，那么就不知道模型的好坏，并不知道预测的真实数据的评分多少

- 将数据集按一定比例分为训练用的数据集和测试用的数据集后，通过训练用的数据集得出的学习算法，通过测试数据检验学习算法的好坏，可以一定程度上了解到训练出的学习算法是否符合预期

分析模型的表现

在项目的第三部分，我们来看一下几个模型针对不同的数据集在学习和测试上的表现。另外，你需要专注于一个特定的算法，用全部训练集训练时，提高它的 'max_depth' 参数，观察这一参数的变化如何影响模型的表现。把你模型的表现画出来对于分析过程十分有益。可视化可以让我们看到一些单看结果看不到的行为。

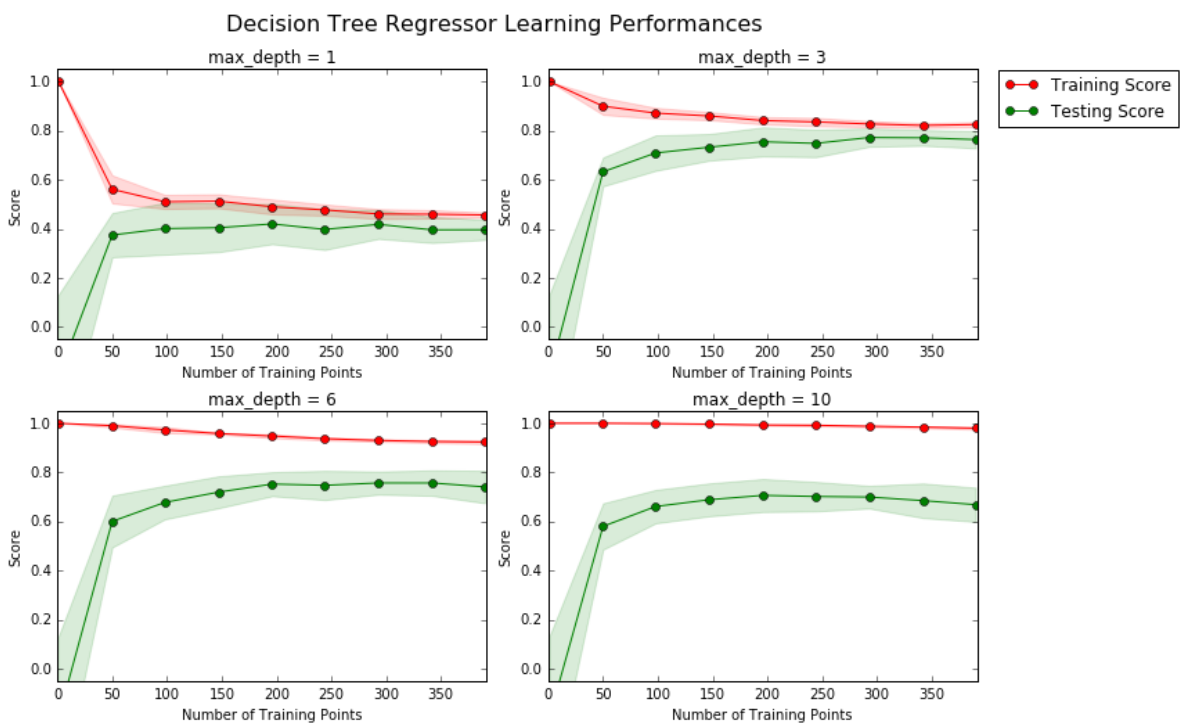
学习曲线

下方区域内的代码会输出四幅图像，它们是一个决策树模型在不同最大深度下的表现。每一条曲线都直观的显示了随着训练数据量的增加，模型学习曲线的训练评分和测试评分的变化。注意，曲线的阴影区域代表的是该曲线的不确定性（用标准差衡量）。这个模型的训练和测试部分都使用决定系数 R^2 来评分。

运行下方区域中的代码，并利用输出的图形回答下面的问题。

In [6]:

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



问题 4 - 学习数据

选择上述图像中的其中一个，并给出其最大深度。随着训练数据量的增加，训练曲线的评分有怎样的变化？测试曲线呢？如果有更多的训练数据，是否能有效提升模型的表现呢？提示：学习曲线的评分是否最终会收敛到特定的值？

答案:

- 我选择max_depth=3的图像
- 随着训练数据量的增加，训练曲线的评分逐渐趋近于1，测试曲线在0.6到0.8直接逐渐稳定下来，根据max_depth深度增大，图形表现的结果来看，即使有更多的训练数据，也不会有明显提升模型的表现。

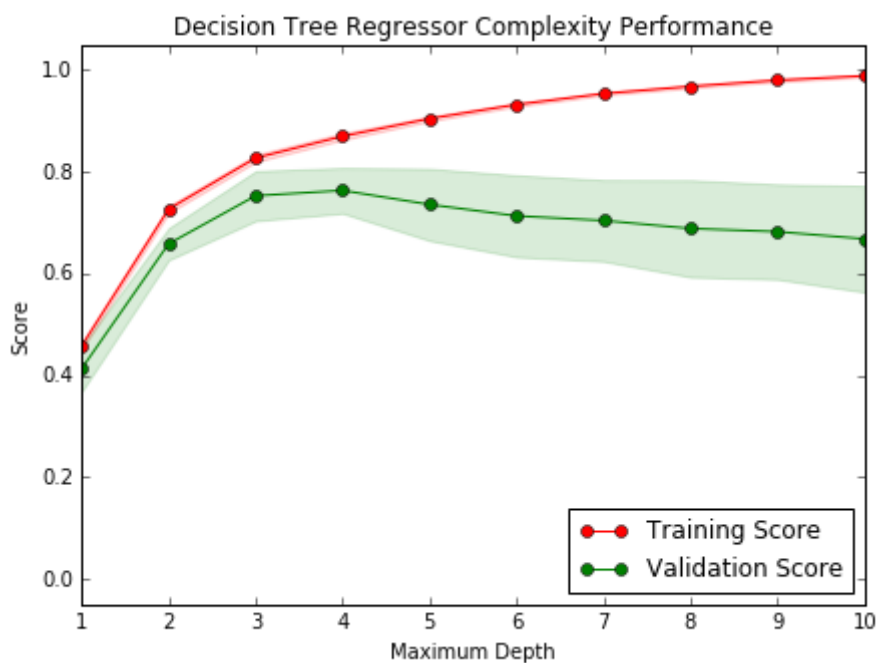
复杂度曲线

下列代码内的区域会输出一幅图像，它展示了一个已经经过训练和验证的决策树模型在不同最大深度条件下的表现。这个图形将包含两条曲线，一个是训练的变化，一个是测试的变化。跟学习曲线相似，阴影区域代表该曲线的不确定性，模型训练和测试部分的评分都用的 performance_metric 函数。

运行下方区域中的代码，并利用输出的图形并回答下面的两个问题。

In [8]:

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



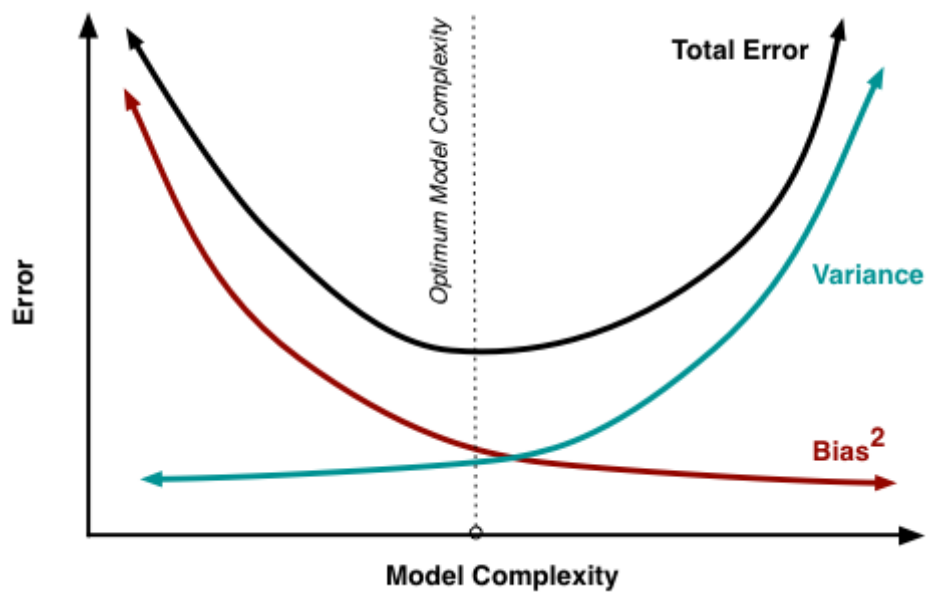
问题 5- 偏差与方差之间的权衡取舍

当模型以最大深度 1 训练时，模型的预测是出现很大的偏差还是出现了很大的方差？当模型以最大深度10训练时，情形又如何呢？图形中的哪些特征能够支持你的结论？

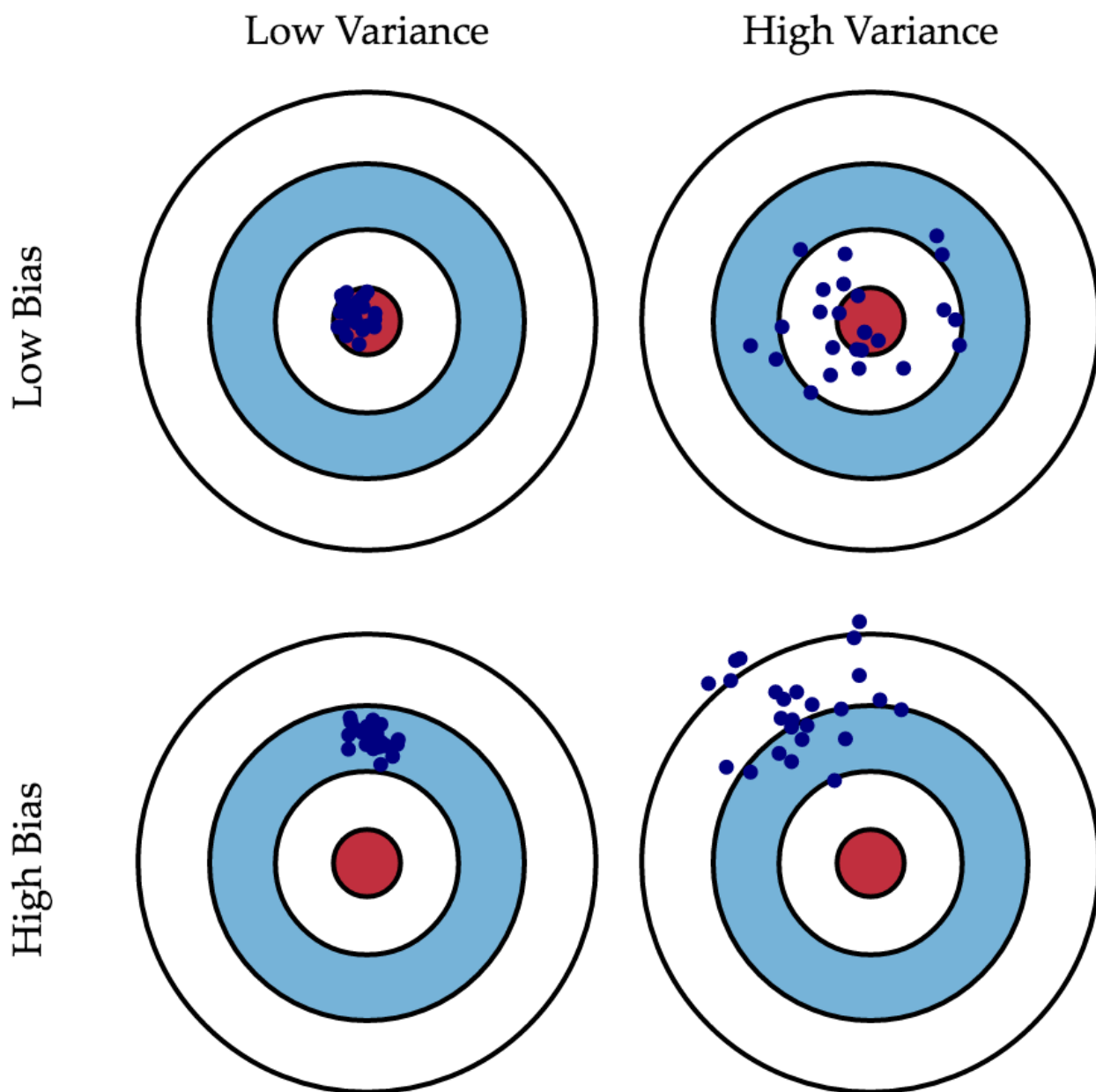
提示： 你如何得知模型是否出现了偏差很大或者方差很大的问题？

答案: 首先偏差描述的是预测值与真实值之间的差距，方差是描述预测值的变化范围

感谢评审者的评审，从评审意见中看到此图感觉非常直观，特贴出此图



这张靶心图很好的阐述了方差和偏差的区别



根据原图形的表现，当模型以最大深度1训练时，模型的预测是出现很大的偏差，因为训练数据和测试数据都里真实值较远，而预测值基本稳定，测试数据比较集中也就是所谓的偏差大，方差小，当模型以最大深度10训练时，模型的预测出现了很大的方差，因为测试数据比较分散，从图上来看，训练数据趋近于1，测试数据比较分散且在0.6~0.8之间，表示有一定的偏差，且数据分散，表示方差比较大。

问题 6- 最优模型的猜测

你认为最大深度是多少的模型能够最好地对未见过的数据进行预测？为什么你会得出了这个答案？

答案: 根据复杂度曲线所表现的结果，最大深度是4的时候能够最好的对未见过的数据进行预测。因为此时的偏差相对较小，预测值基本稳定在0.7~0.8之间，且测试数据也在这个区间范围内，这个深度是相对偏差和方差都相对较小的点。相对于深度3训练数据的效果更好，而测试数据差别不大的情况下，选择深度为4是比较合适的。

评价模型表现

在这个项目的最后，你将自己建立模型，并使用最优化的`fit_model`函数，基于客户房子的特征来预测该房屋的价值。

问题 7- 网格搜索（Grid Search）

什么是网格搜索法？如何用它来优化学习算法？

回答: 网格搜索法是一种常用的超参数寻优算法，可以根据参数范围先设定一组值，然后通过网格搜索法穷举各种参数组合，找到最匹配当前数据的参数组合，以此优化学习算法。

问题 8- 交叉验证

什么是K折交叉验证法（*k-fold cross-validation*）？优化模型时，使用这种方法对网格搜索有什么好处？

提示：跟为何需要一组训练集的原因差不多，网格搜索时如果不使用交叉验证会有什么问题？

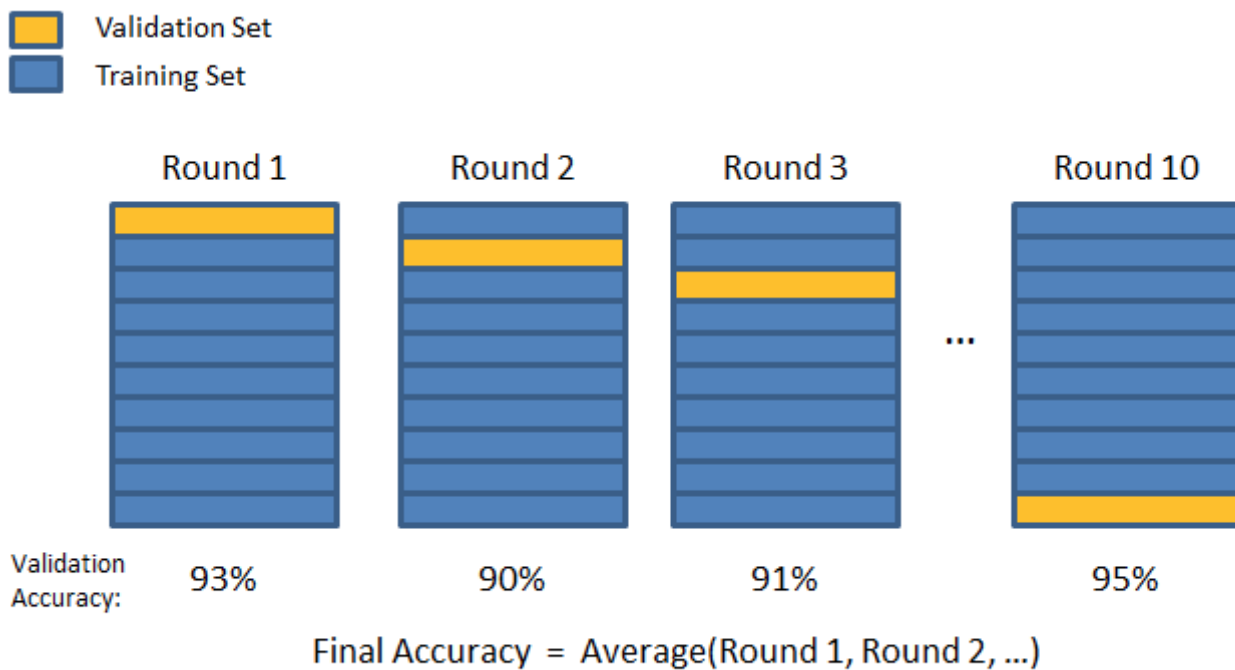
答案:

根据评审者的意见，我重新梳理了一下之前的文字，对k-fold cross-validation进行说明

基本步骤如下：

1. 获取原始数据集
2. 将原始数据集平均分成K份，标记为：1,2,3...K
3. 第1轮：取第1到第K-1份数据作为训练数据对模型进行训练，取第K份数据作为测试集测试模型，获取准确率A-1
4. 第2轮：取第1到第K-2和第K作为训练数据对模型进行训练，取第K-1份作为测试数据测试模型，获取准确率A-2
5. 第3轮：取第1到第K-3和第K-1到第K份作为训练数据多模型进行训练，取第K-2份作为测试数据测试模型，获取准确率A-3
6. ...依次类推
7. 第K轮：取第2到K份作为训练数据进行训练，取第1份作为测试数据测试模型，获取准确率A-k
8. 得出最后的模型，模型的准确率就是这K轮的平均值

从下图可以更直观的反应上述步骤：



从上图中可以看出，使用K折交叉验证法，所有的数据都被充分的用于训练模型，且所有数据也被用于测试模型，也就是说数据被充分利用了，如果网格搜索时不使用交叉验证，那么网格搜索在不完备的训练数据所优化的参数并不完备。

评审者提到的这篇文章很好的说明了这个问题[链接 \(http://www.anc.ed.ac.uk/rbf/intro/node16.html\)](http://www.anc.ed.ac.uk/rbf/intro/node16.html)

练习：训练模型

在最后一个练习中，你将需要将所学到的内容整合，使用决策树演算法训练一个模型。为了保证你得出的是一个最优模型，你需要使用网格搜索法训练模型，以找到最佳的 'max_depth' 参数。你可以把 'max_depth' 参数理解为决策树算法在做出预测前，允许其对数据提出问题的数量。决策树是监督学习算法中的一种。

在下方 `fit_model` 函数中，你需要做的是：

- 使用 `sklearn.tree` 中的 `DecisionTreeRegressor` (<http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>) 创建一个决策树的回归函数；
 - 将这个回归函数储存在 'regressor' 变量中；
- 为 'max_depth' 创建一个字典，它的值是从1至10的数组，并储存在 'params' 变量中；
- 使用 `sklearn.metrics` 中的 `make_scorer` (http://scikit-learn.org/stable/modules/generated/sklearn.metrics.make_scorer.html) 创建一个评分函数；
 - 将 `performance_metric` 作为参数传至这个函数中；
 - 将评分函数储存在 'scoring_fnc' 变量中；
- 使用 `sklearn.grid_search` 中的 `GridSearchCV` (http://scikit-learn.org/stable/modules/generated/sklearn.grid_search.GridSearchCV.html) 创建一个网格搜索对象；
 - 将变量 'regressor', 'params', 'scoring_fnc', 和 'cv_sets' 作为参数传至这个对象中；
 - 将 `GridSearchCV` 存到 'grid' 变量中。

In [7]:

```
# TODO: Import 'make_scorer', 'DecisionTreeRegressor', and 'GridSearchCV'
from sklearn.metrics import make_scorer
from sklearn.tree import DecisionTreeRegressor
from sklearn.grid_search 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(X.shape[0], n_iter = 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': (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)}

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

    # TODO: Create the grid search object
    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_
```

做出预测

当我们用数据训练出一个模型，它现在就可用于对新的数据进行预测。在决策树回归函数中，模型已经学会对新输入的数据提问，并返回对目标变量的预测值。你可以用这个预测来获取数据未知目标变量的信息，这些数据必须是不包含在训练数据之内的。

问题 9- 最优模型

最优模型的最大深度（maximum depth）是多少？此答案与你在问题 6 所做的猜测是否相同？

运行下方区域内的代码，将决策树回归函数代入训练数据的集合，以得到最优化的模型。

In [8]:

```
# 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'])

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

Answer: Parameter 'max_depth' is 4 for the optimal model. 此答案与问题6所做的猜测基本相同。

问题 10 - 预测销售价格

想像你是一个在波士顿地区的房屋经纪人，并期待使用此模型以帮助你的客户评估他们想出售的房屋。你已经从你的三个客户收集到以下的资讯：

特征	客户 1	客户 2	客户 3
房屋内房间总数	5 间房间	4 间房间	8 间房间
社区贫困指数（%被认为是贫困阶层）	17%	32%	3%
邻近学校的学生-老师比例	15：1	22：1	12：1

你会建议每位客户的房屋销售的价格为多少？从房屋特征的数值判断，这样的价格合理吗？

提示：用你在分析数据部分计算出来的统计信息来帮助你证明你的答案。

运行下列的代码区域，使用你优化的模型来为每位客户的房屋价值做出预测。

```
In [9]:  
  
# 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
```

答案:

- 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

从房屋特征的数值判断，这样的价格相对合理

敏感度

一个最优的模型不一定是一个健壮模型。有的时候模型会过于复杂或者过于简单，以致于难以泛化新增添的数据；有的时候模型采用的学习算法并不适用于特定的数据结构；有的时候样本本身可能有太多噪点或样本过少，使得模型无法准确地预测目标变量。这些情况下我们会说模型是欠拟合的。执行下方区域中的代码，采用不同的训练和测试集执行 fit_model 函数10次。注意观察对一个特定的客户来说，预测是如何随训练数据的变化而变化的。

In [12]:

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

```
Trial 1: $391,183.33
Trial 2: $419,700.00
Trial 3: $415,800.00
Trial 4: $420,622.22
Trial 5: $413,334.78
Trial 6: $411,931.58
Trial 7: $399,663.16
Trial 8: $407,232.00
Trial 9: $351,577.61
Trial 10: $413,700.00
```

Range in prices: \$69,044.61

问题 11 - 实用性探讨

简单地讨论一下你建构的模型能否在现实世界中使用？

提示： 回答几个问题：

- 1978年所采集的数据，在今天是否仍然适用？
- 数据中呈现的特征是否足够描述一个房屋？
- 模型是否足够健壮来保证预测的一致性？
- 在波士顿这样的大都市采集的数据，能否应用在其它乡镇地区？

答案:

- 1978采集的数据，在今天可能无法适用了，因为上述条件中没有考虑通货膨胀等因素
- 数据中呈现的特征，在一个时间段内可以一定程度描述一个房屋，但不是绝对的。有很多其他因素也会对房屋价格有直接影响
- 在特定时间周期内，从敏感度章节来看，模型的波动范围较小，我认为相对健壮了，可以保证预测的一致性
- 大都市和乡镇会有差异，因为人口规模还有科技水平等其他影响因素，不能应用在其他乡镇地区。

结论，我建构的模型只能特定时间和地理范围内可以在现实世界使用，这个模型不能适用于大部分现实世界的真实情况