# Using the Anaconda Distribution for Python

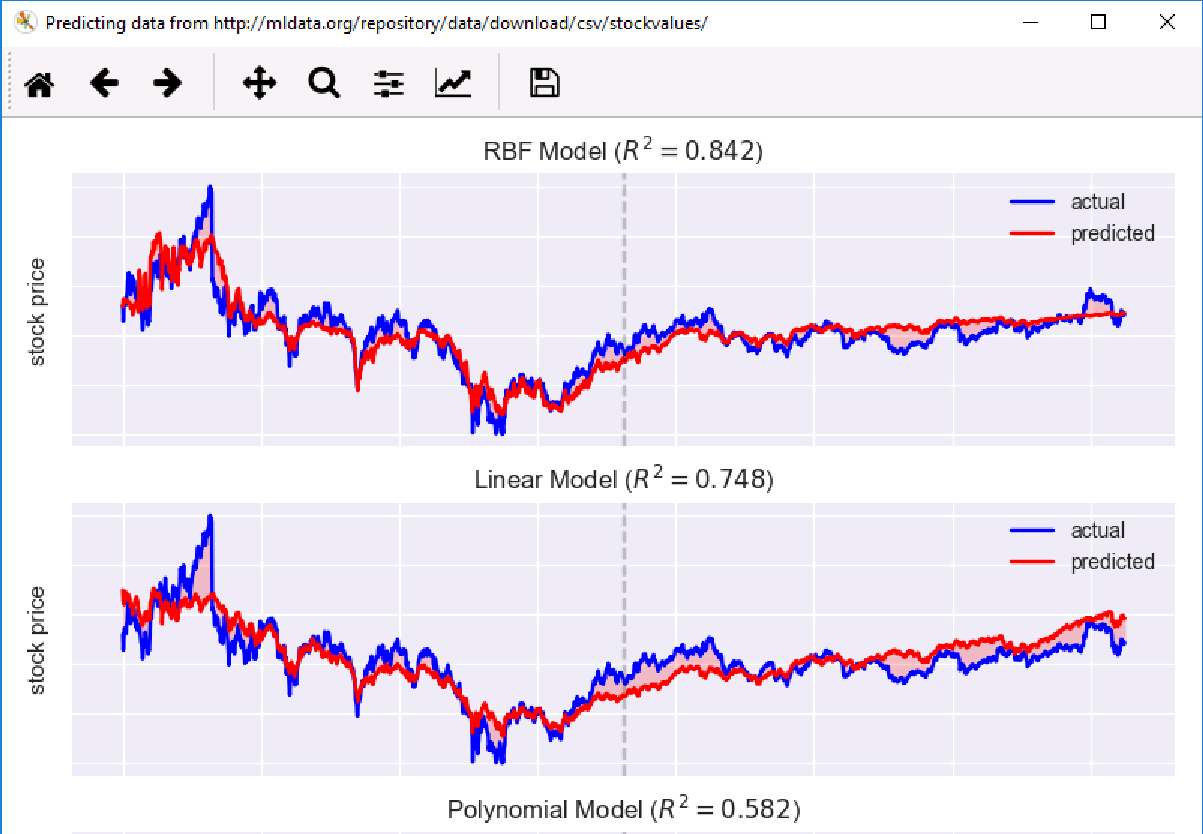
## Overview

Python is a powerful tool for scientific computing, data analysis, and machine learning. In this lab, we will learn how simple it is to perform a machine learning experiment using the Anaconda distribution installed via the Visual Studio Data Science workload. In doing so, we will interact with some of the fundamental Python tools for VS to set-up our environment and execute the experiment.

To get our ML script, we will use the Cookiecutter Explorer tool. Cookiecutter Explorer enables developers to search for hundreds of meaningful application templates on Github from within VS, and generate customized boilerplate from them to jump start their development.

## The experiment

Predictive Analysis involves using existing trend data to train models to predict future movement of that trend. In this lab, we will use open data to predict stock prices using three different regression models. Our goal is to find the regression method that leads to most accurate predictions.

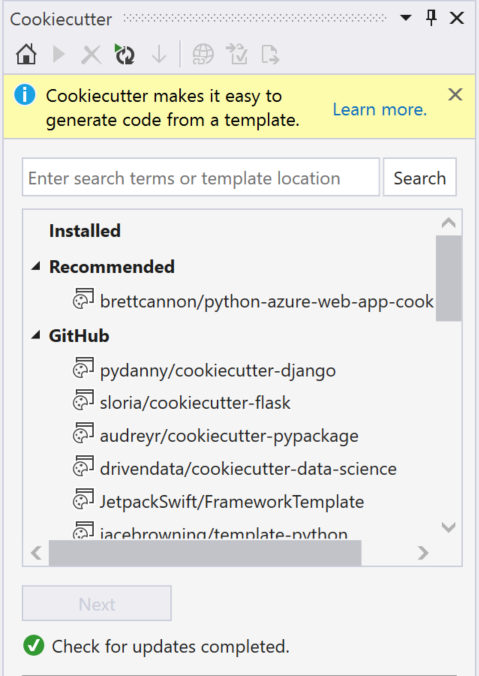


Our Python program will break the stock data into two sets: a training set and a test set. The test set will be used to compare predicted values with actual values.

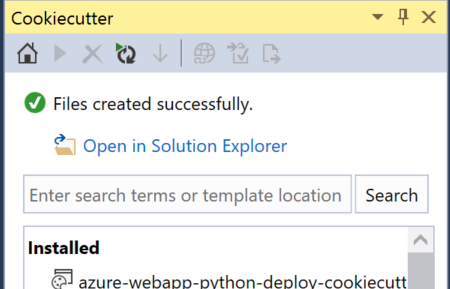
## Let's get started!

### Part I: Acquiring the experiment script

1. Launch Visual Studio 2017 Preview (During //build/ timeframe, the final version of this preview will be available as Visual Studio 2017 Update 2)
2. Open the Cookiecutter Explorer (**Tools** -> **Python Tools** -> **Cookiecutter Explorer**)



1. Create a simple regression from a template. Search for the term “regression”. Choose the **electir/build-regression** template. Hit **Next** to install the template, and follow the flow of the tool.
   1. The tool will ask you to provide some context which will be inserted in the code generated.
   2. Once the project is generated, open it in the solution explorer. If there is an error, try to generate the project again from the installed template.



## Part II: Preparing the Script

1. Import requisite packages by adding the following lines of code under the imports section:

Data will be read into a **Pandas** Data Frame:

from pandas import read\_table

The Pandas Data Frame will be transmuted into **NumPy** arrays before model calc.NumPy is optimized for matrix calculations.

import numpy as np

A powerful plotting library

import matplotlib.pyplot as plt

A powerful ML package for Python

from sklearn.svm import SVR

1. In the **get\_features\_and\_labels()** method, add the following lines of code to create training and test sets. There is a comment in the code indicating where this code should be inserted.

from sklearn.cross\_validation import train\_test\_split

X\_train, \_, y\_train, \_ = train\_test\_split(X, y, test\_size=0.5)

X\_test, y\_test = X, y

1. In the **evaluate\_learner()** method, add code to train each of the three regression learners.

For Radial Basis Function:

svr = SVR(kernel='rbf', gamma=0.1)

svr.fit(X\_train, y\_train)

y\_pred = svr.predict(X\_test)

r\_2 = svr.score(X\_test, y\_test)

yield 'RBF Model ($R^2={:.3f}$)'.format(r\_2), y\_test, y\_pred

For Linear Kernel:

svr = SVR(kernel='linear')

svr.fit(X\_train, y\_train)

y\_pred = svr.predict(X\_test)

r\_2 = svr.score(X\_test, y\_test)

yield 'Linear Model ($R^2={:.3f}$)'.format(r\_2), y\_test, y\_pred

For Polynomial Kernel

svr = SVR(kernel='poly', degree=2)

svr.fit(X\_train, y\_train)

y\_pred = svr.predict(X\_test)

r\_2 = svr.score(X\_test, y\_test)

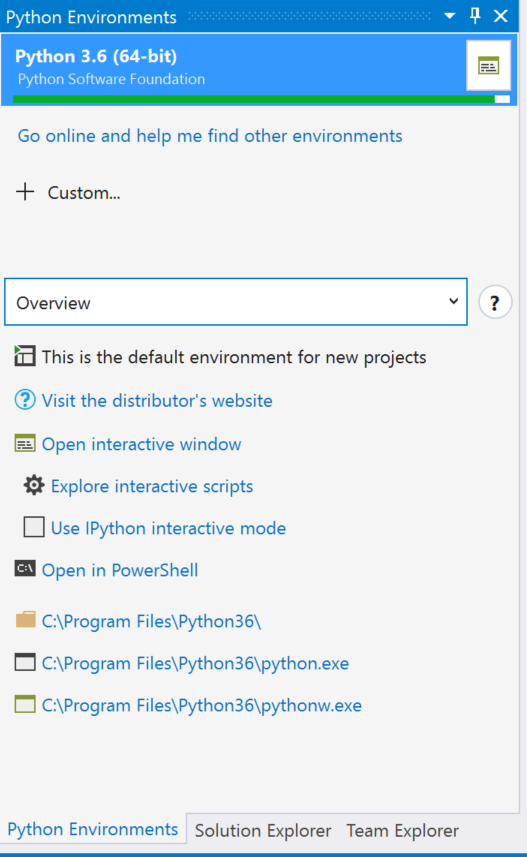
yield 'Polynomial Model ($R^2={:.3f}$)'.format(r\_2), y\_test, y\_pred

1. At this point, we have all the code to run the experiment. Take some time to look through other methods of the script. Next, we will configure the environment.

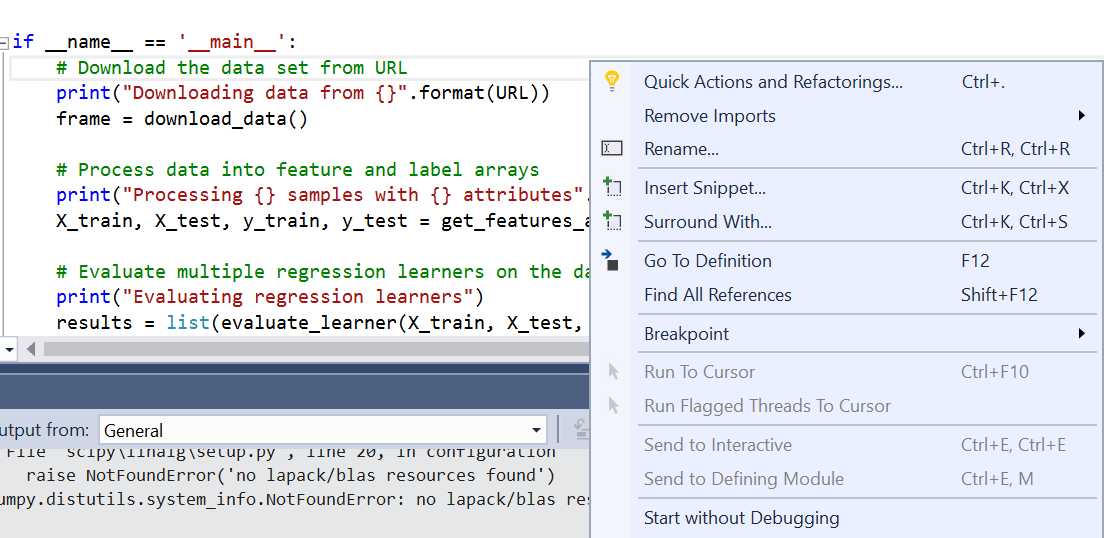
### Part III: Run Script with Anaconda

The Anaconda distribution was installed using the Data Science and Analytical Applications workload in Visual Studio 2017. Anaconda is a distribution of choice for problems like this lab – it comes packed with over 750 python packages for scientific computing and data science.

1. Open the Environments Window (**Tools** -> **Python Tools** -> **Environment**)

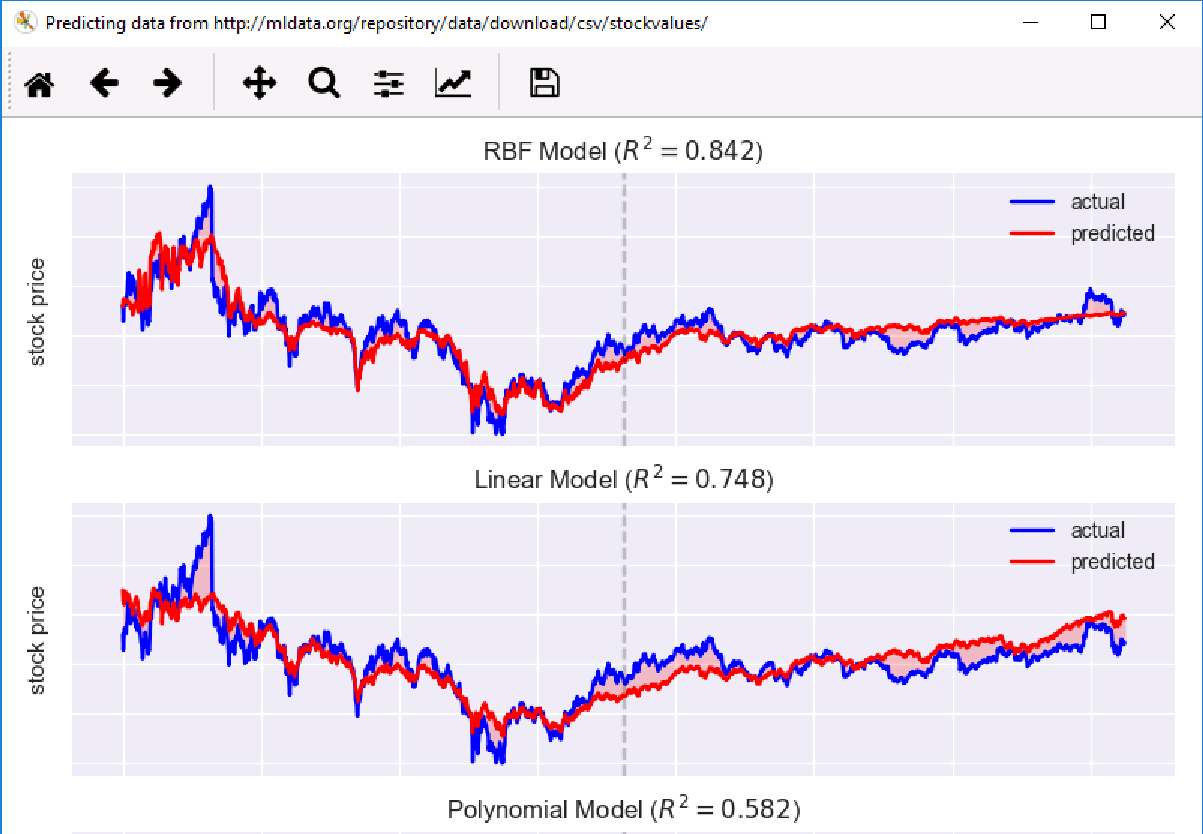


1. Select the Anaconda environment as the default for all projects. If the environment is active, it will have a bold font.
2. Run the script by **right click -> start without debugging** in the editor window.
   1. If the program fails, wait a minute or so and repeat this step. Sometimes the remote open data server may not respond.
   2. Sometimes it may take a while for the program to show an output – this is because it is fetching data sets from a remote open data server.



Once the script runs through the end, you should see a plot of precited values. The R2 value is a measure of accuracy of the prediction. The measurement compares the predicted values against the test of actual values. The closer this value is to 1, the better the predicted values curve fits with the curve of actual values.

Which regression technique best suits this problem?



As you can see using Anaconda precludes the need to customize the environment – it already comes loaded with many popular Python packages for scientific computing.

Cookiecutters can help provide customizable boiler plate code for apps, dev-ops scripts, and ML scripts. Feel free to search for other templates to get a grasp of the cookiecutter ecosystem. If you are an Azure developer, we have templates to jumpstart HPC, IoT, and Web development on the Azure stack.

If you use Jupyter Notebooks for your data science work, with Python tools in Visual Studio you can fine tune your notebooks using local python/mixed-mode debugging, and the profiler. Simply export your IPYNB file as python file and open it in Visual Studio.

## Further Reading

* [Data Science Workload in Visual Studio blogpost](https://blogs.msdn.microsoft.com/visualstudio/2017/05/10/build-intelligent-apps-faster-with-visual-studio/)
* [Python Tools release notes](https://blogs.msdn.microsoft.com/pythonengineering/2017/03/07/python-support-in-vs2017/)
* [Cookiecutter Explorer blogpost](https://blogs.msdn.microsoft.com/pythonengineering/2016/11/02/creating-apps-from-templates-cookiecutter-in-visual-studio-15/)