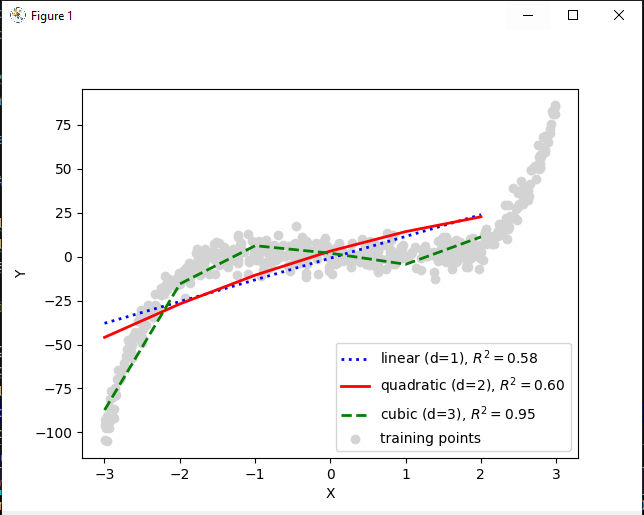
First, I tried my own version of polynomial regression using SKLearn. By learning how to use the Polynomial Features transformer class, adding a second-degree polynomial term, and comparing with a linear regression model. After constructing the polynomial feature to fit a nonlinear relationship, I created a more concrete example by applying the concepts to the data in the Housing dataset. After executing the code, I saw modeling the relationship between the second degree (quadratic) and the third degree (cubic) polynomials and comparing them to a linear fit. The cubic fit captures the relationships better than the linear and the quadratic fit. By adding more polynomial features, the complexity of the model increased the chance of overfitting. Just by looking at the scatterplot lead me to believe that a logarithmic transformation of the feature variable and the square root may project the data onto a linear feature space suitable for a linear regression fit.



I determined that the best degree polynomials were 2nd, 3rd, 4th , 5th were the best of the learning set of parameters where the loss starts very high but rapidly drops to nearly zero after the 5th degree.

