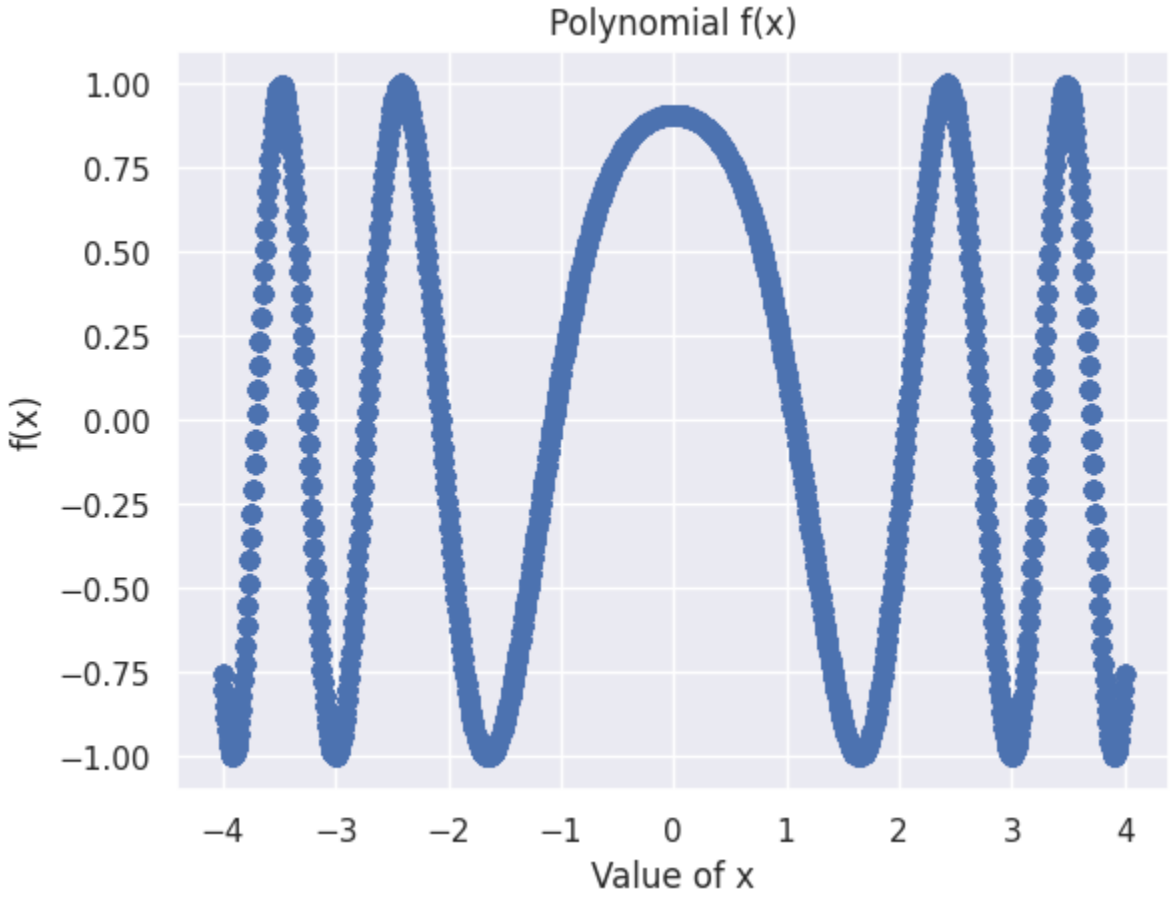
Experiment 1

In this experiment, we assume an arbitrary function $f(x)$ and compute its value on 10,000 equally-spaced data points. We try to find out the optimal complexity of the regression model that can best estimate this function.

We generate 10,000 points using the numpy.linspace for the **chosen trigonometric function : *f(x) = sin(x^2 + 2)***.

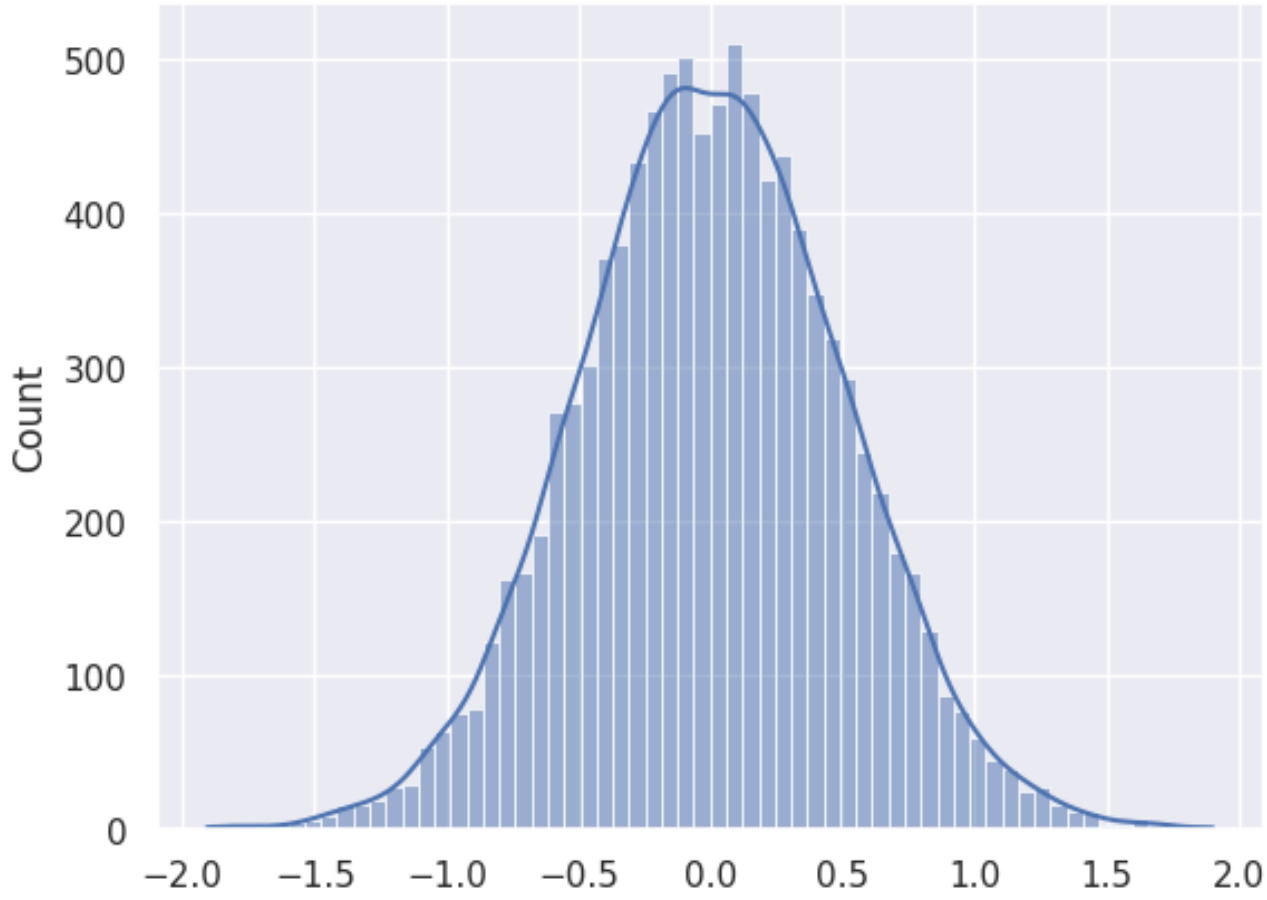
Visualising Dataset



We introduce some noise to the response variable, assuming that the noise follows Normal Distribution $\epsilon \sim N(\mu,\sigma^2)$ with zero mean and a non-zero variance (σ^2=0.50). The probability density function of the Normal Distribution is given as:

$$n(\epsilon) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{\epsilon-\mu}{\sigma}\right)^2\right]$$

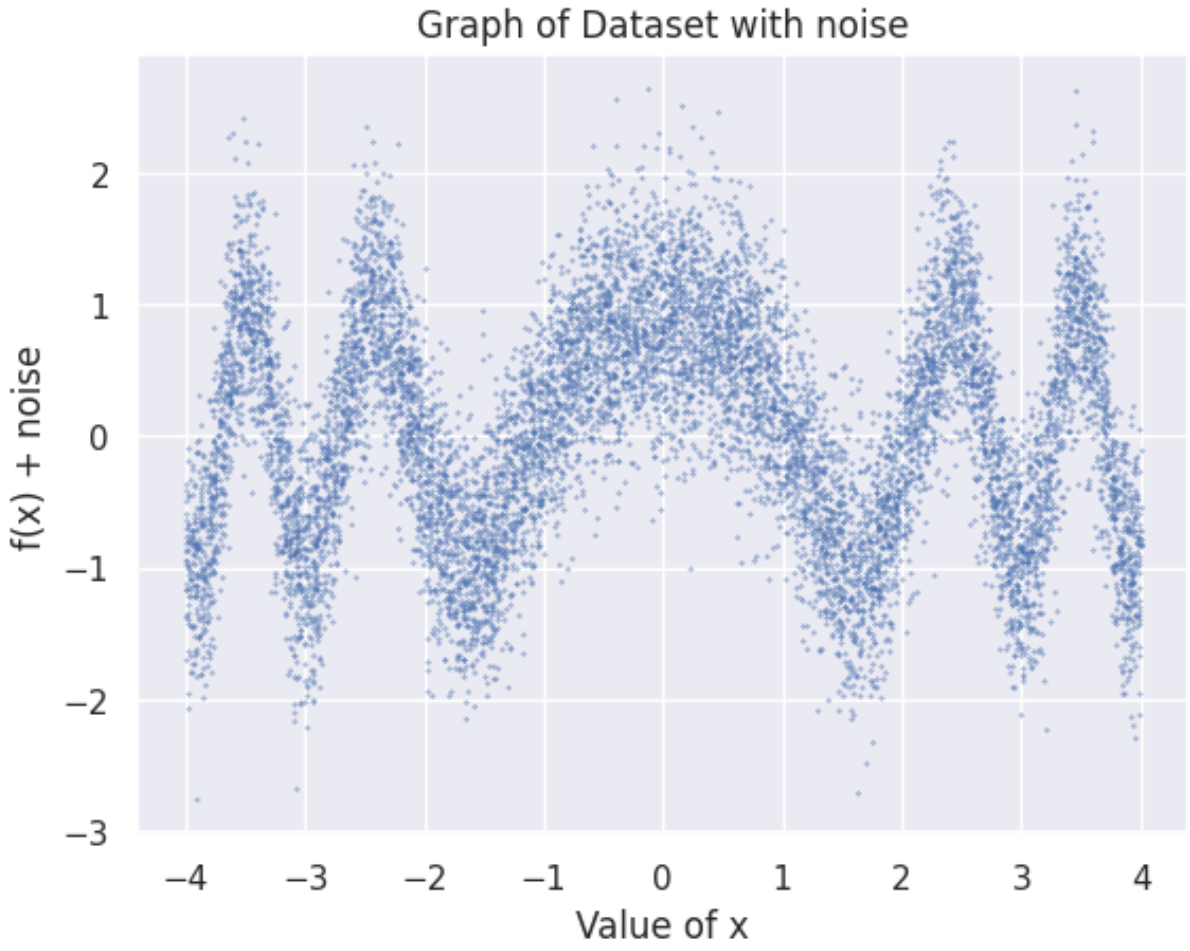
Visualizing Generated Noise $\epsilon \sim N(0, \sigma^2)$



We introduced the generated Gaussian noise to the function values f(x) to produce noisy data.

	x	f(x)	e(x)	f(x) + e(x)
0	-3.92	-0.996162	-0.521580	-1.517742
1	-3.55	0.893672	-0.410428	0.483244
2	1.00	0.141120	0.332573	0.473693
3	3.39	0.799060	0.911313	1.710373
4	0.02	0.909131	-0.720792	0.188339

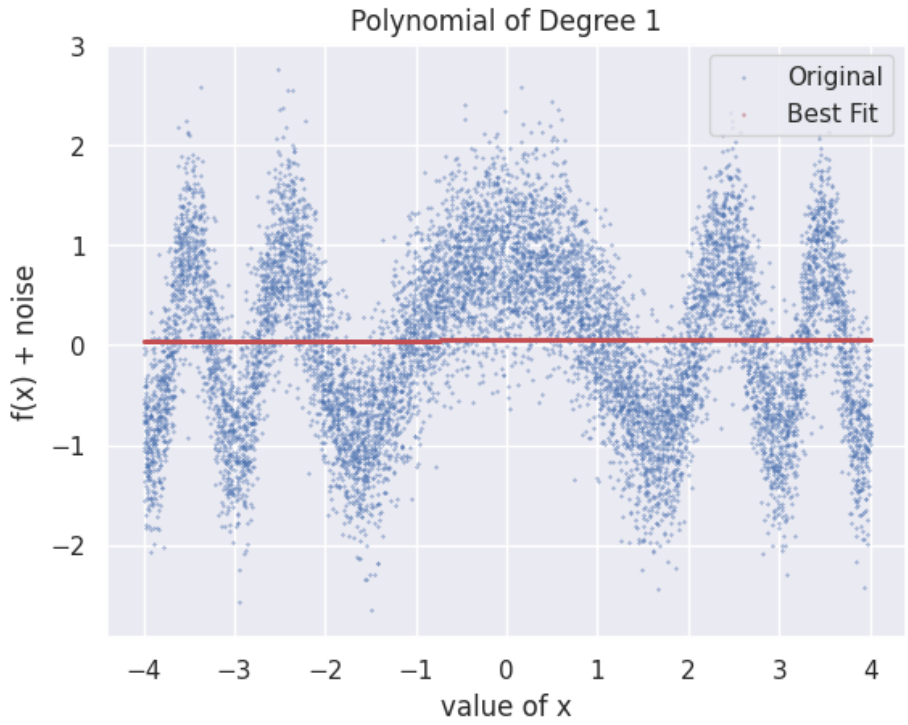
Visualizing Noisy Dataset



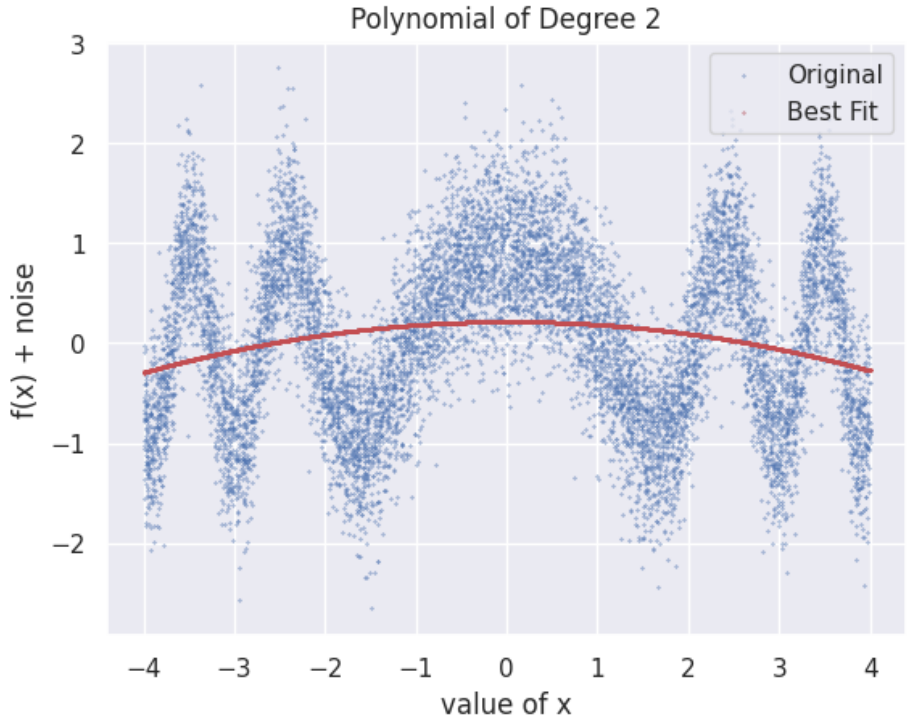
Now, we naively fit polynomials of order 1 through 10 sequentially, on the entire dataset to estimate the parameters w_0, w,…, w_d for each polynomial. When we visualize these polynomials, they give us a sense of how the best-fit polynomial of each degree looks like when they are superimposed on the noisy dataset. This is represented in the output as follows:

Fitting Polynomials without CV

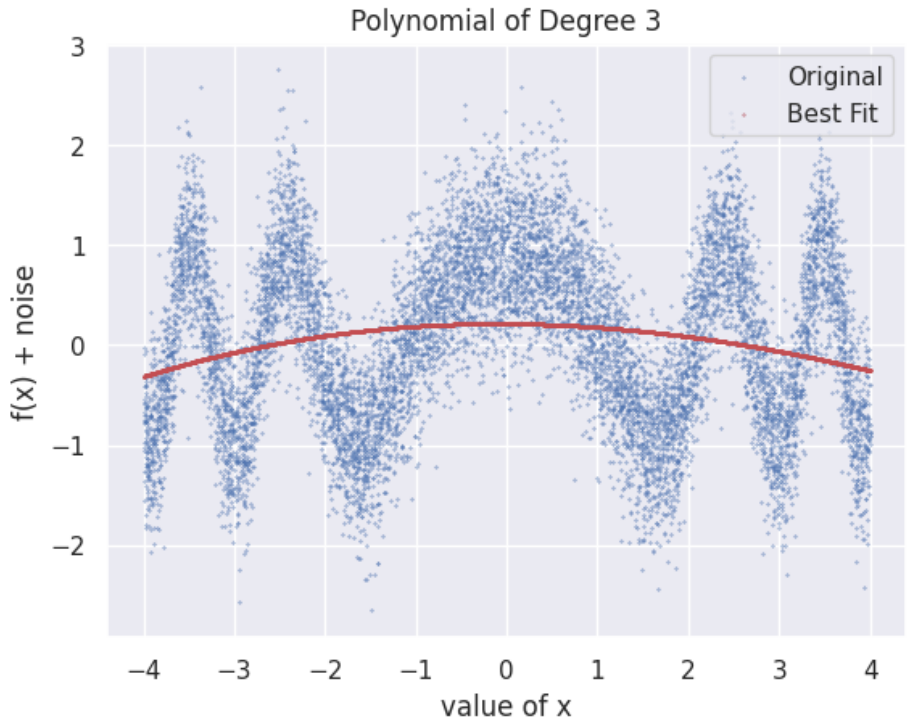
`Best Fit (Degree 1): 0.0008358 x + 0.06129`



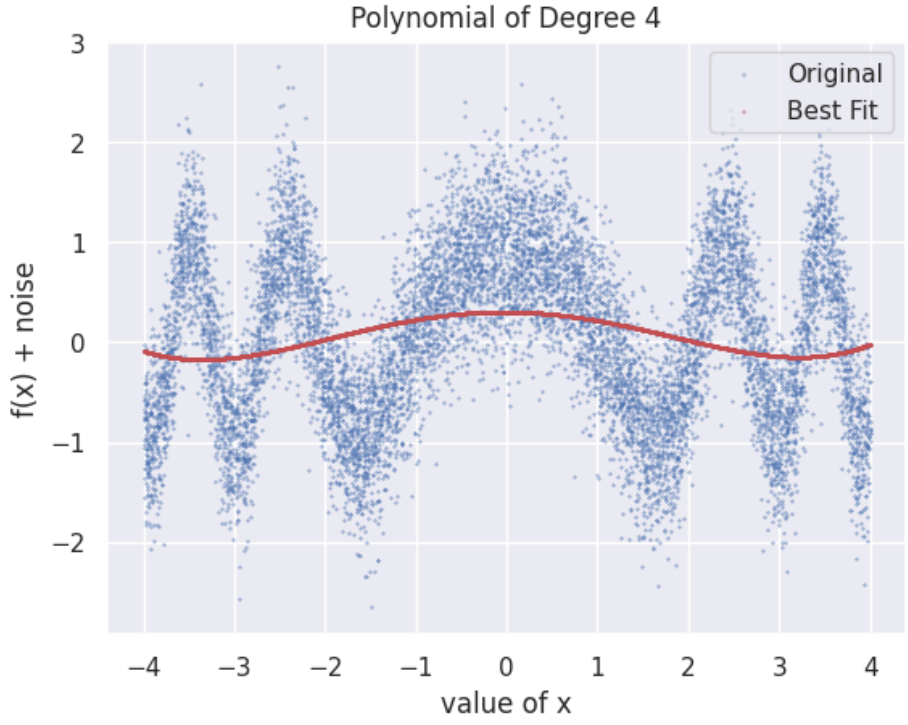
`Best Fit (Degree 2): -0.03327 x`^{2}`+ 0.001418 x + 0.239`



Best Fit (Degree 3): $0.0008202\ x^3 - 0.03116\ x^2 - 0.005707\ x + 0.2164$

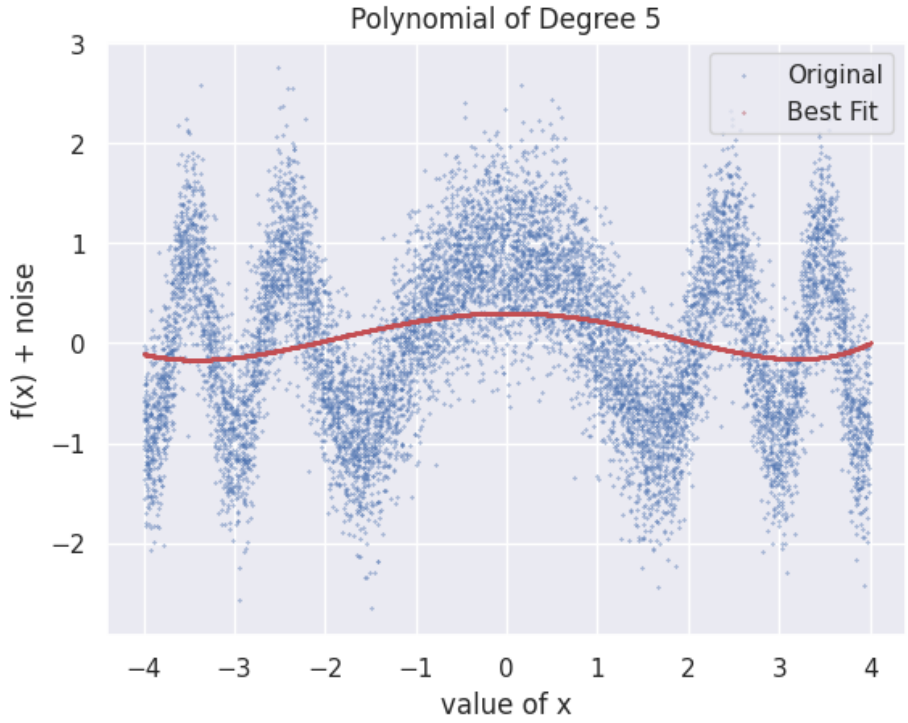


Best Fit (Degree 4):
$0.003965\ x^4 + 0.0009035\ x^3 - 0.0858\ x^2 - 0.006132\ x + 0.3047$



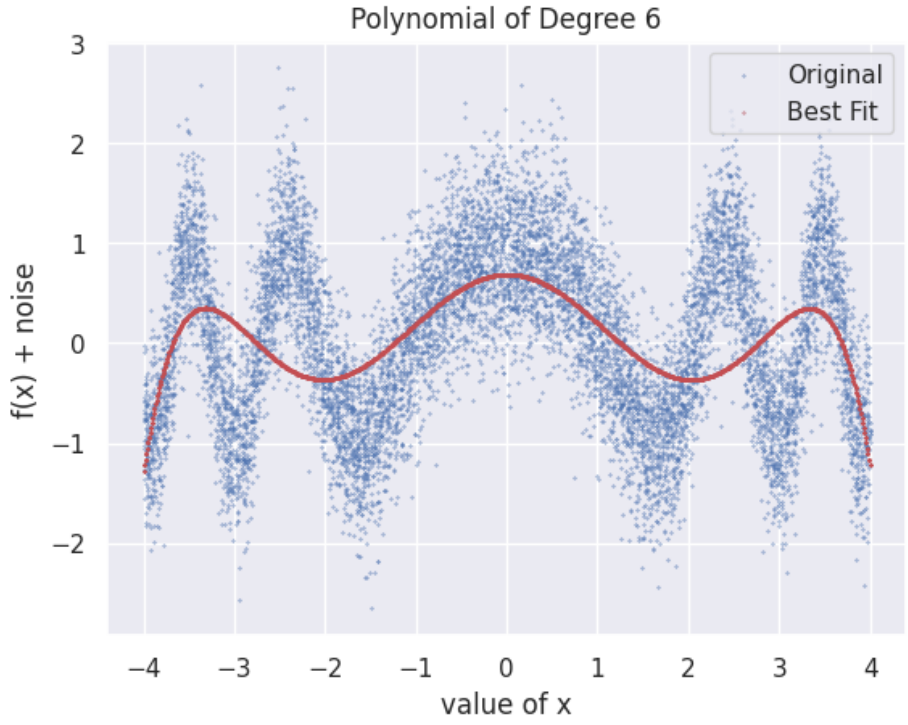
Best Fit (Degree 5):

$0.0001685\ x^5 + 0.003969\ x^4 - 0.002098\ x^3 - 0.08583\ x^2 + 0.004163\ x + 0.3048$

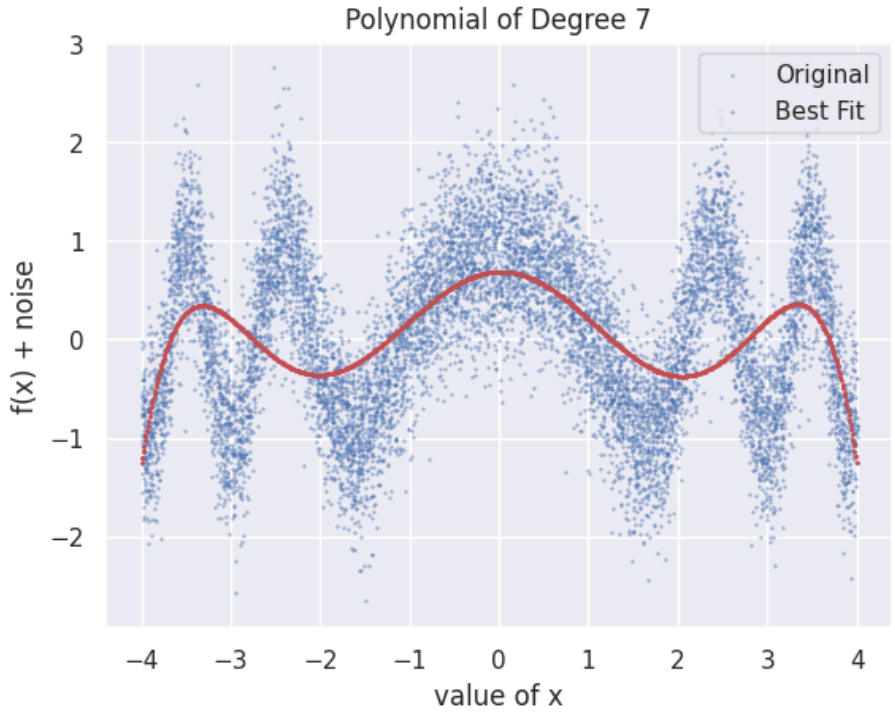


Best Fit (Degree 6):

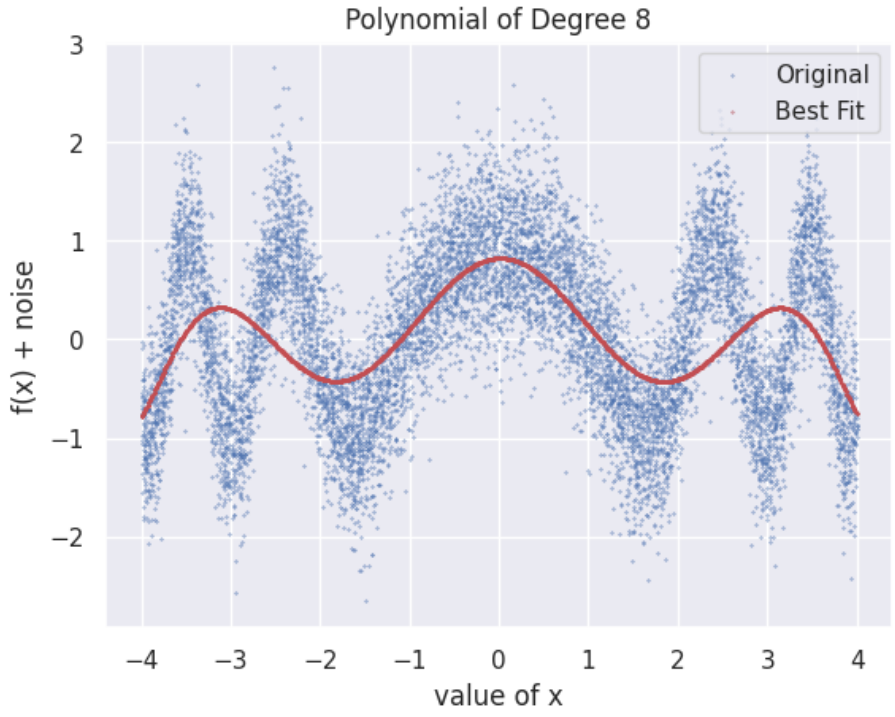
$-0.004281\ x^6 + 0.0001387\ x^5 + 0.09747\ x^4 - 0.002134\ x^3 - 0.5848\ x^2 + 0.005917\ x + 0.6887$



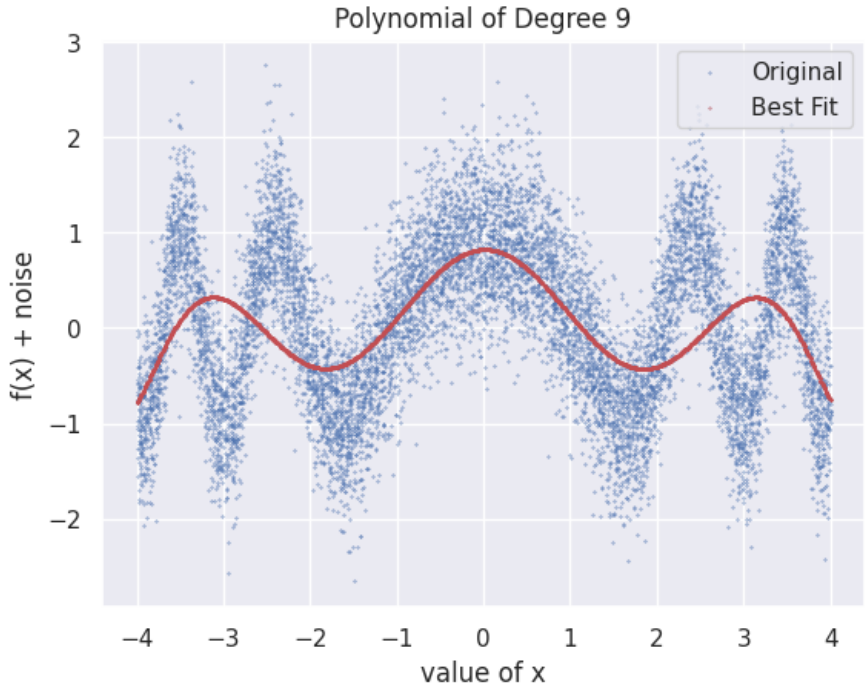
Best Fit (Degree 7): $-4.285e-05\ x^7 - 0.004282\ x^6 + 0.001249\ x^5 + 0.09748\ x^4 - 0.01024\ x^3 - 0.5849\ x^2 + 0.02042\ x + 0.6888$



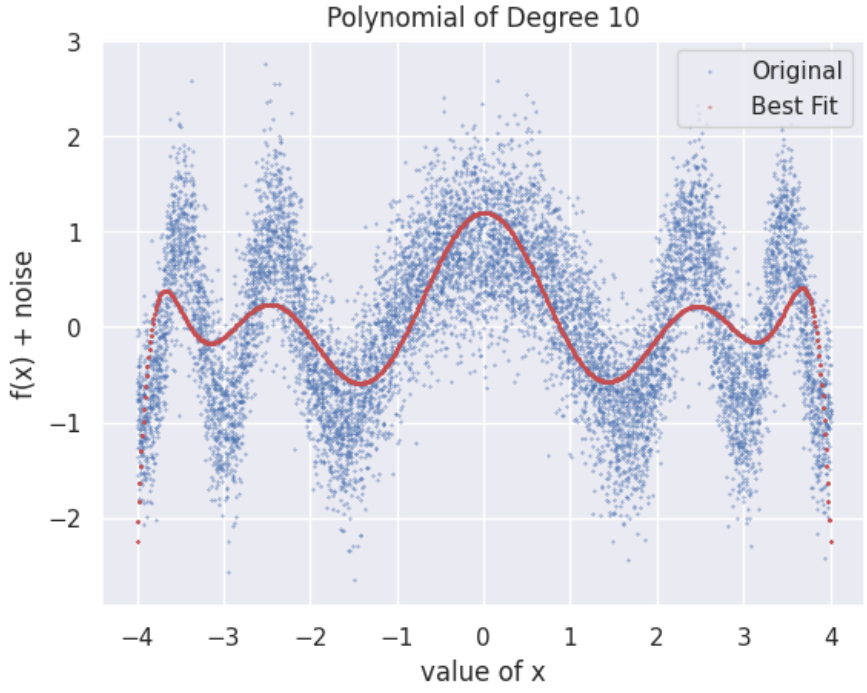
Best Fit (Degree 8): $0.0003816\ x^8 - 4.051e-05\ x^7 - 0.01568\ x^6 + 0.001227\ x^5 + 0.2027\ x^4 - 0.0104\ x^3 - 0.8908\ x^2 + 0.02172\ x + 0.8248$



Best Fit (Degree 9): $4.917e-07\ x^9 + 0.0003816\ x^8 - 5.72e-05\ x^7 - 0.01568\ x^6 + 0.001414\ x^5 + 0.2027\ x^4 - 0.01117\ x^3 - 0.8908\ x^2 + 0.02257\ x + 0.8248$



Best Fit (Degree 10): $-0.0002642\ x^{10} + 2.516e-06\ x^9 + 0.01041\ x^8 - 0.0001345\ x^7 - 0.1481\ x^6 + 0.002351\ x^5 + 0.9099\ x^4 - 0.01522\ x^3 - 2.198\ x^2 + 0.02828\ x + 1.203$



EWe now have to evaluate the complexity of the polynomial which best estimates our function. To do this, we apply k -fold cross validation. Applying k -fold cross validation, with k = 10 produces 10 train-test splits from the shuffled dataset of sizes 9000:1000 in each fold. In each fold we fit polynomials of order 1 - 10 on the training set to estimate the parameters θ to find the coefficients for the best-fit polynomial.

We calculate the following error measures for each model in each fold:

- MSE (Mean Squared Error):

$$MSE(\theta|X) = \frac{1}{N}\sum_{t=1}^{N}[r^t - g(x^t|\theta)]^2$$

- SE (Squared Error):

$$SE(\theta|X) = \frac{1}{2}\sum_{t=1}^{N}[r^t - g(x^t|\theta)]^2$$

- RSE (Relative Squared Error):

$$RSE(\theta|X) = \frac{\sum_{t=1}^{N}[r^t - g(x^t|\theta)]^2}{\sum_{t=1}^{N}[r^t - \bar{r}]^2}$$

- AbsE (Absolute Error):

$$AbsE(\theta|X) = \sum_{t}|r^t - g(x^t|\theta)|$$

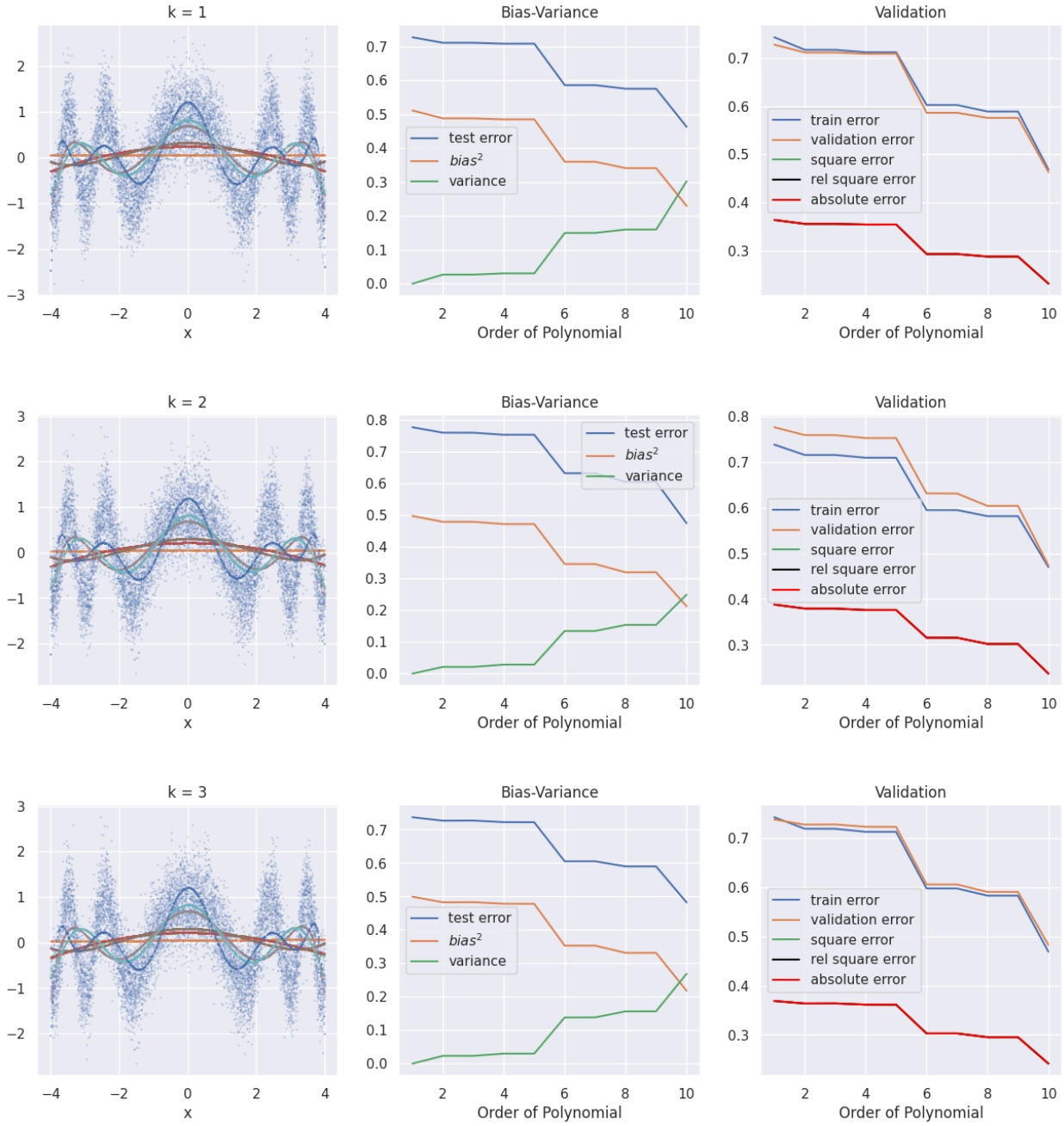
Apart from the error measures, we also calculate the bias and variance in each fold as follows:

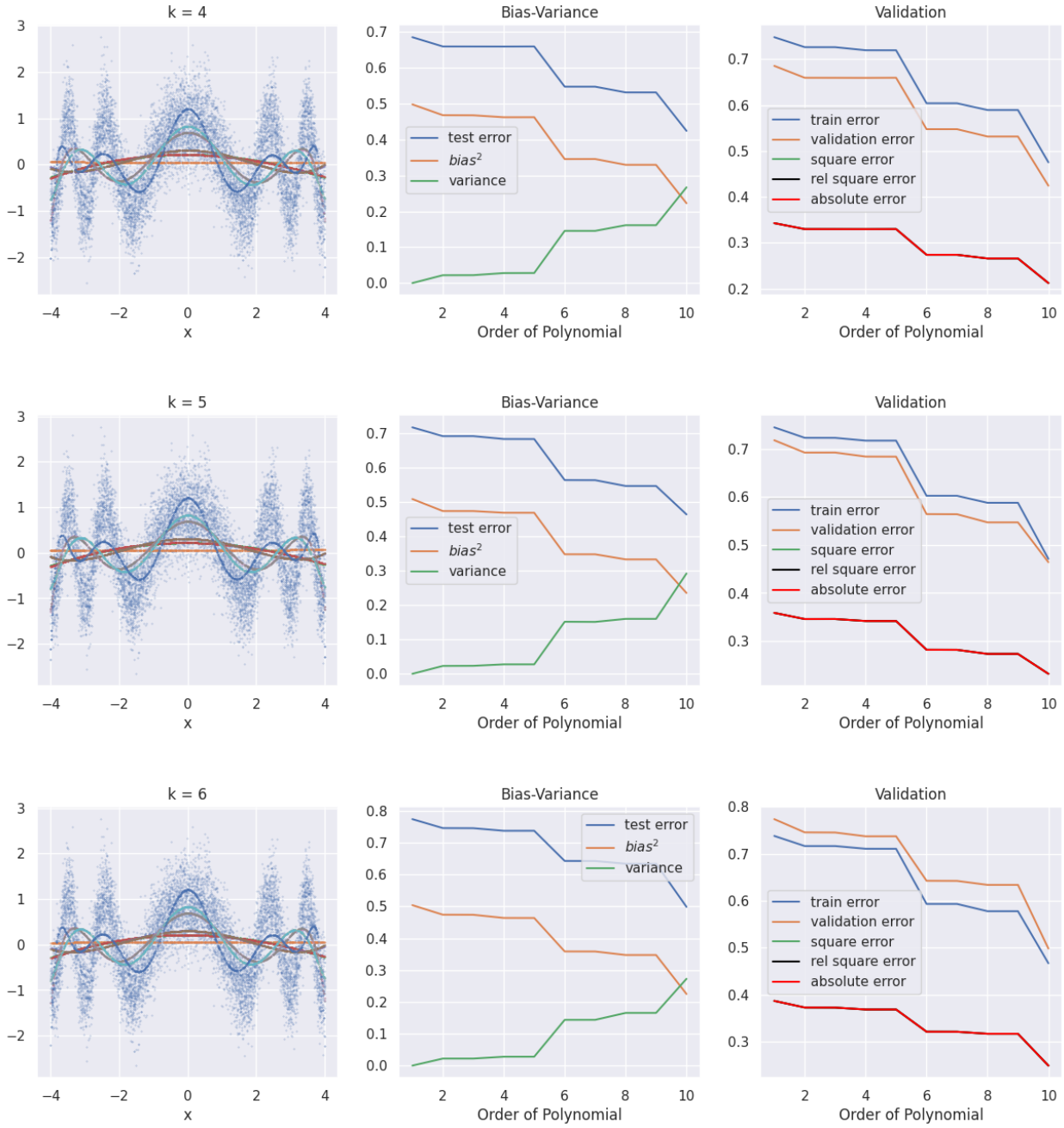
$$Bias^2(g) = \frac{1}{N}\sum_{t}[\bar{g}(x^t) - f(x^t)]^2$$

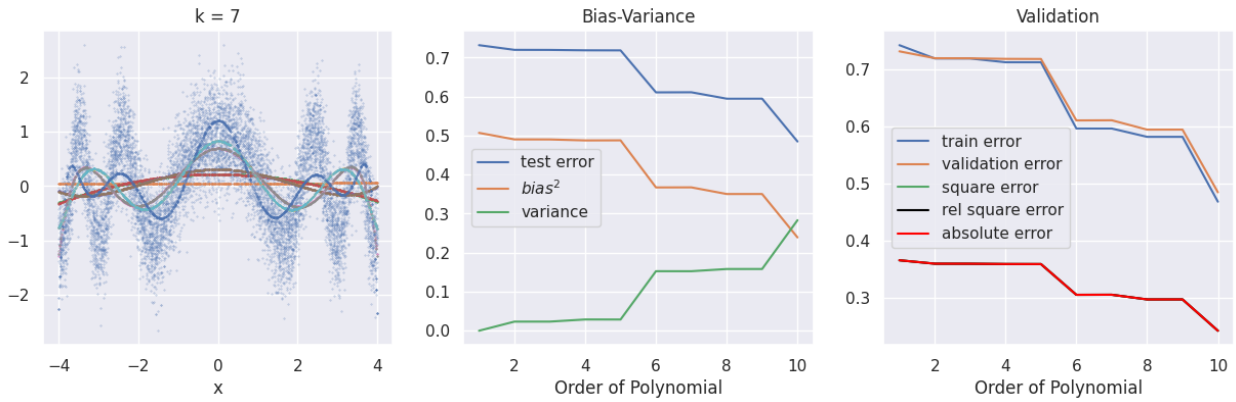
$$Var(g) = \frac{1}{NM}\sum_{t}\sum_{i}[\bar{g}(x^t) - g_i(x^t)]^2$$

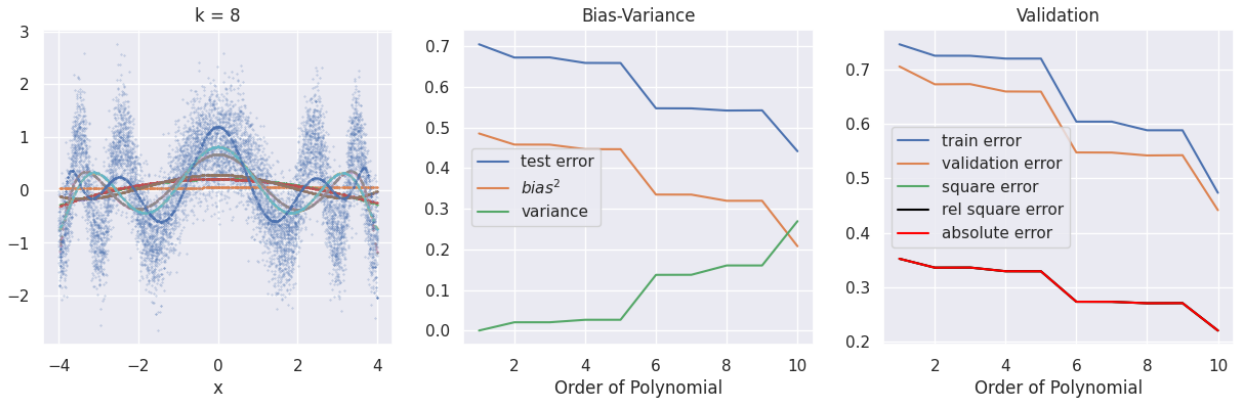
$$\bar{g}(x) = \frac{1}{M}\sum_{t} g_i(x)$$

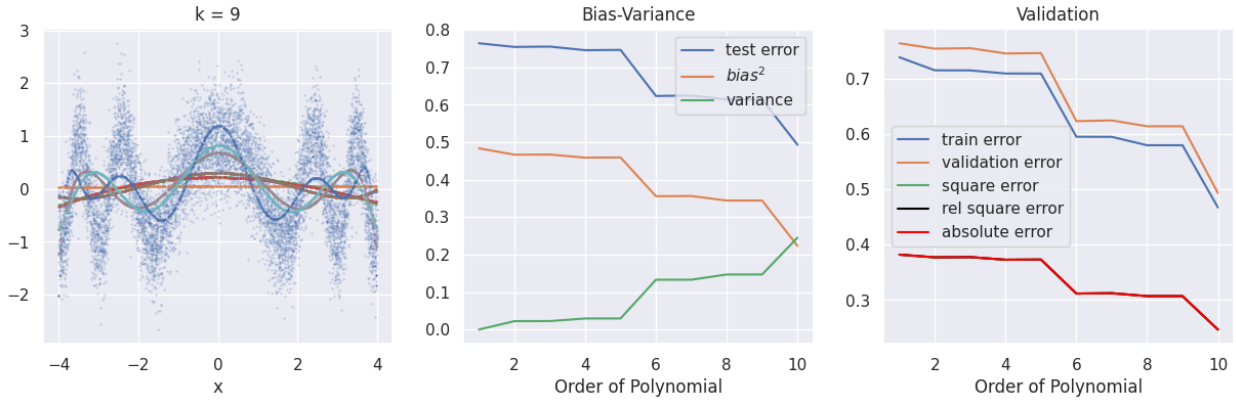
Applying k-fold CV and Fitting Polynomials

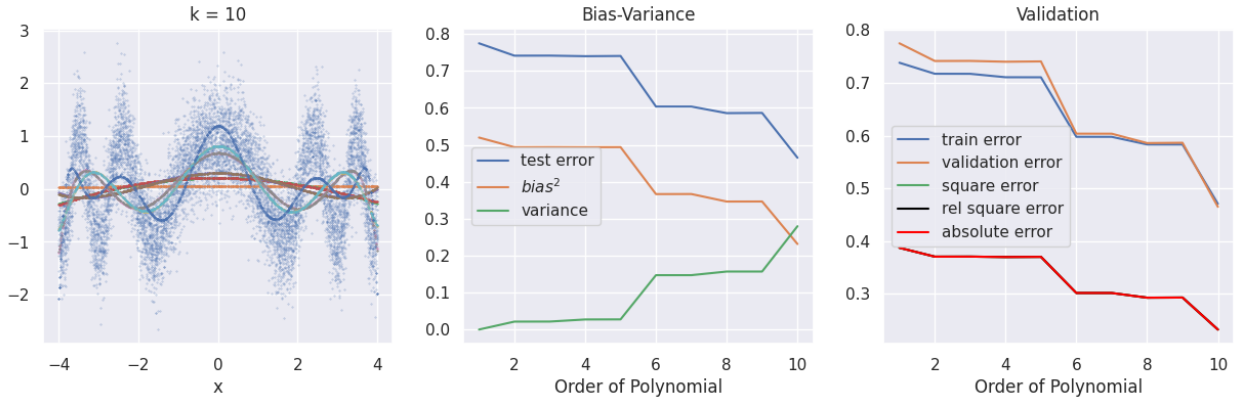



k = 4
Bias-Variance
Validation
x
Order of Polynomial
test error
bias²
variance
train error
validation error
square error
rel square error
absolute error
k = 5
Bias-Variance
Validation
x
Order of Polynomial
test error
bias²
variance
train error
validation error
square error
rel square error
absolute error
k = 6
Bias-Variance
Validation
x
Order of Polynomial
test error
bias²
variance
train error
validation error
square error
rel square error
absolute error


k = 7
x
Bias-Variance
test error
bias²
variance
Order of Polynomial
Validation
train error
validation error
square error
rel square error
absolute error
Order of Polynomial

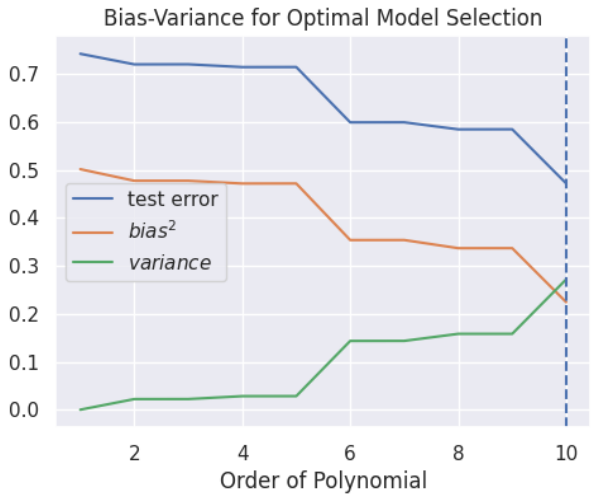

k = 8
x
Bias-Variance
test error
bias²
variance
Order of Polynomial
Validation
train error
validation error
square error
rel square error
absolute error
Order of Polynomial

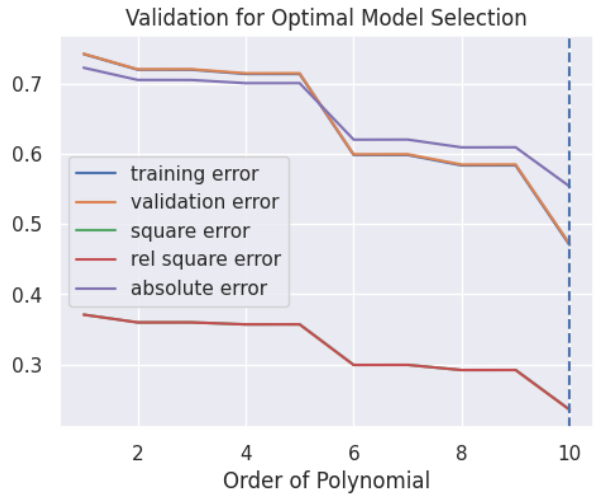

k = 9
x
Bias-Variance
test error
bias²
variance
Order of Polynomial
Validation
train error
validation error
square error
rel square error
absolute error
Order of Polynomial


k = 10
x
Bias-Variance
test error
bias²
variance
Order of Polynomial
Validation
train error
validation error
square error
rel square error
absolute error
Order of Polynomial

Finally, we visualize the trend of error measures, bias, variance over all trials (or folds) by taking the mean of all trials for each degree. We also observe the training v/s validation error averaged over all trials for each degree.

Analyzing Results from k-fold CV to select Optimal Model





Observation

Variance increases with increasing order of polynomial - this is because the performance of overfit models vary widely depending on the training set. The bias decreases with increasing model complexity as estimates from more complex models resemble the actual data points more closely.

Conclusion

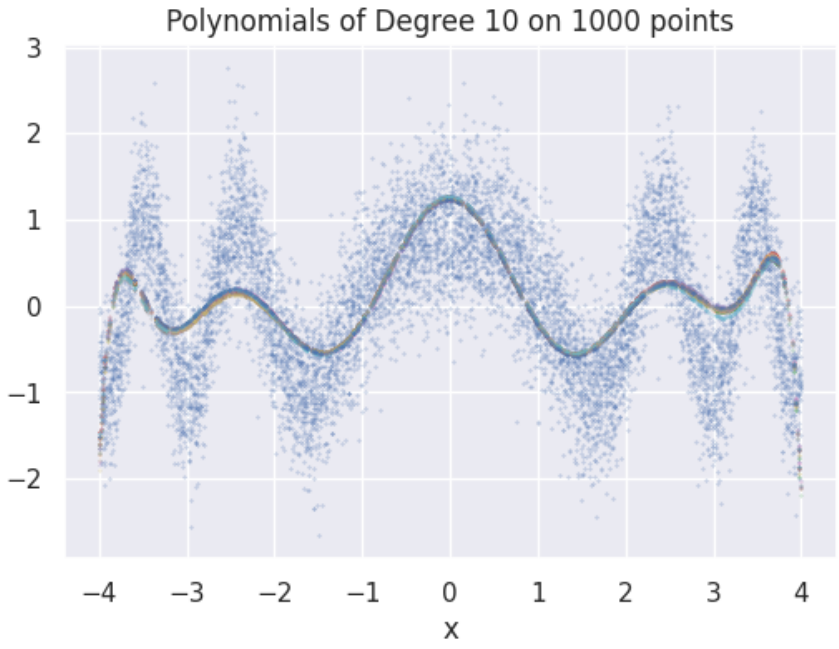
We observe that the test error is minimum for degree 10 polynomials. We also see that we have the best bias-variance trade off for degree 10 polynomials only. Also the other errors are minimum for polynomials to this degree.

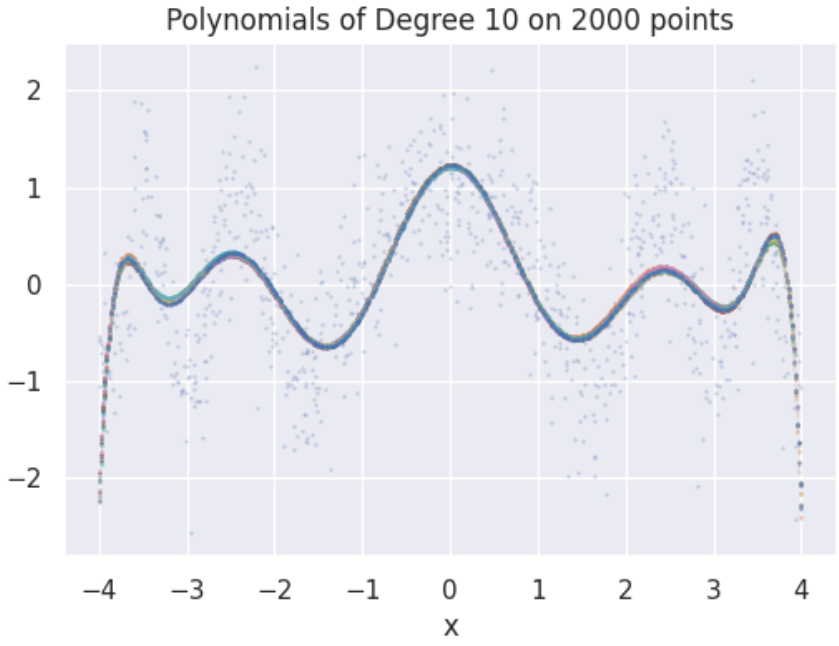
Hence, we choose a polynomial estimator of order 10 as the optimal model.

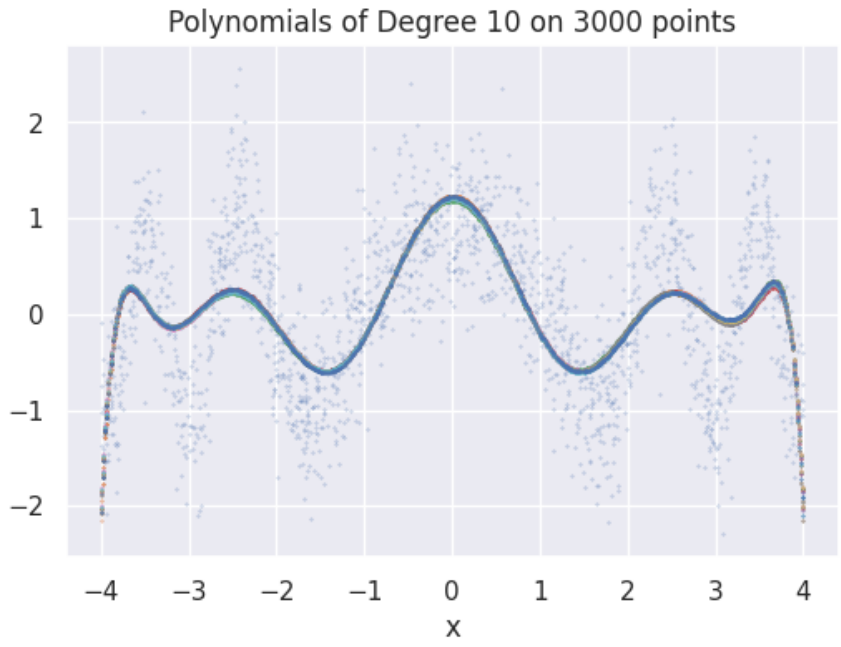
Experiment 2

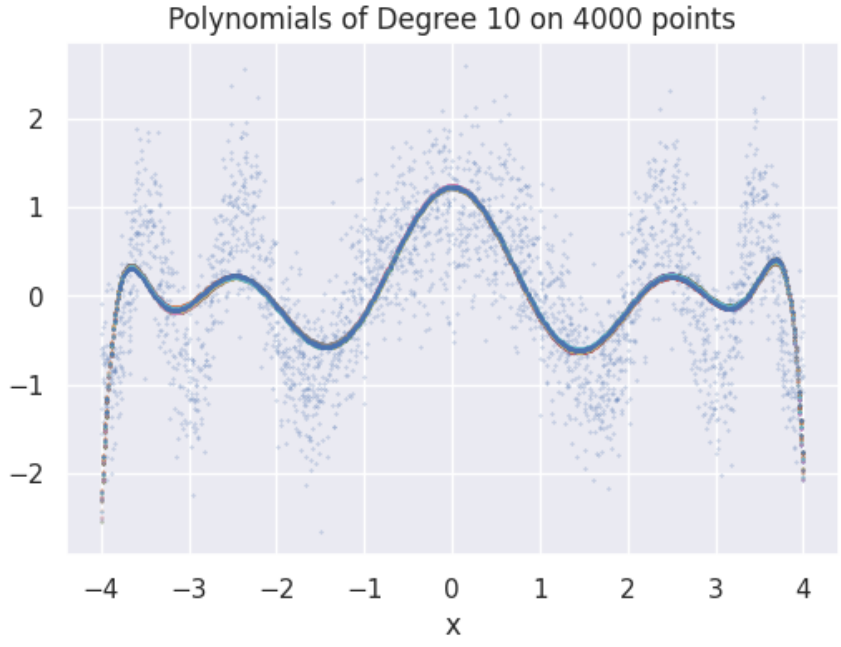
In this experiment, we consider *M* samples of varying sizes and train our optimal model on them. For each such sample, we apply k-fold CV (k =10) to evaluate the model.

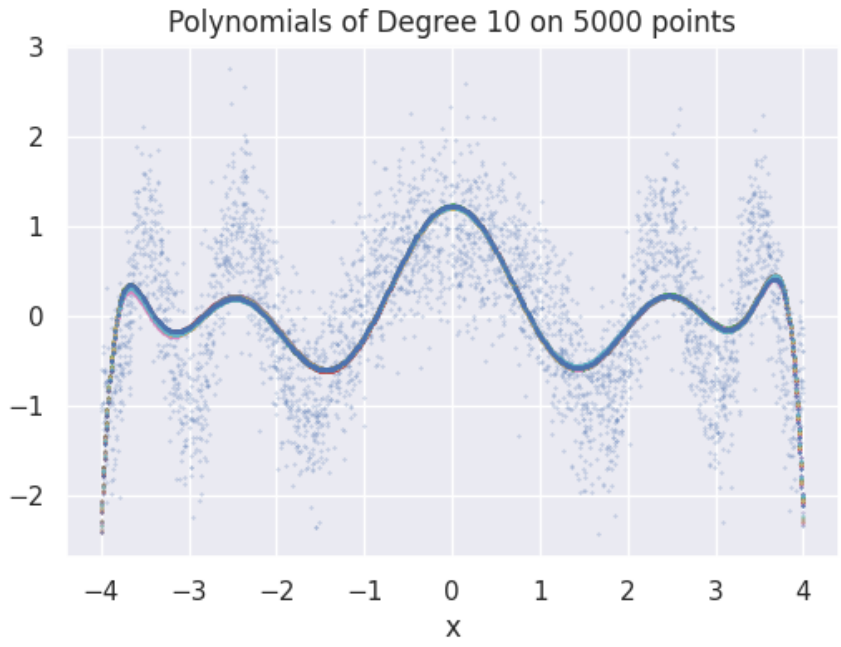
Applying k-fold CV with Different Sample Sizes

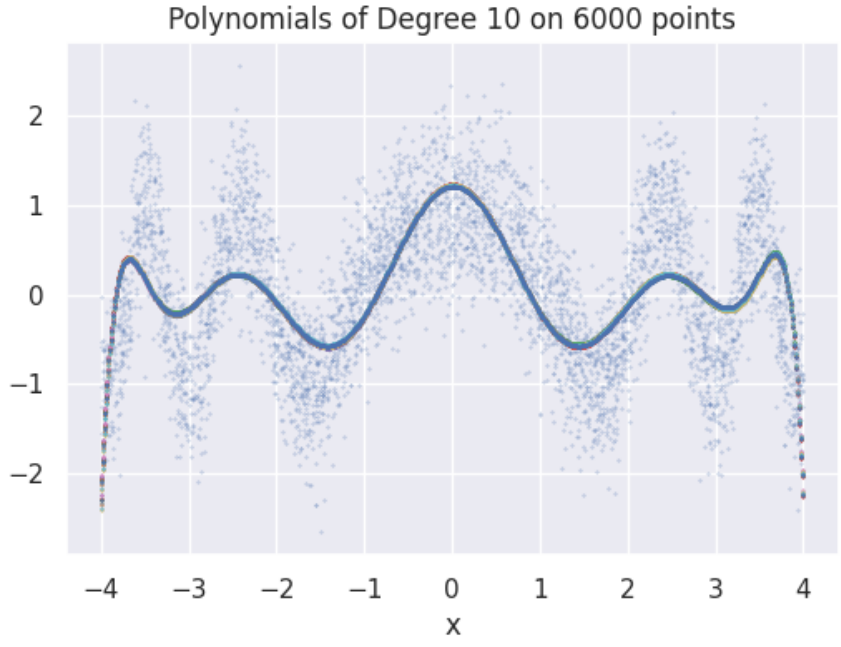


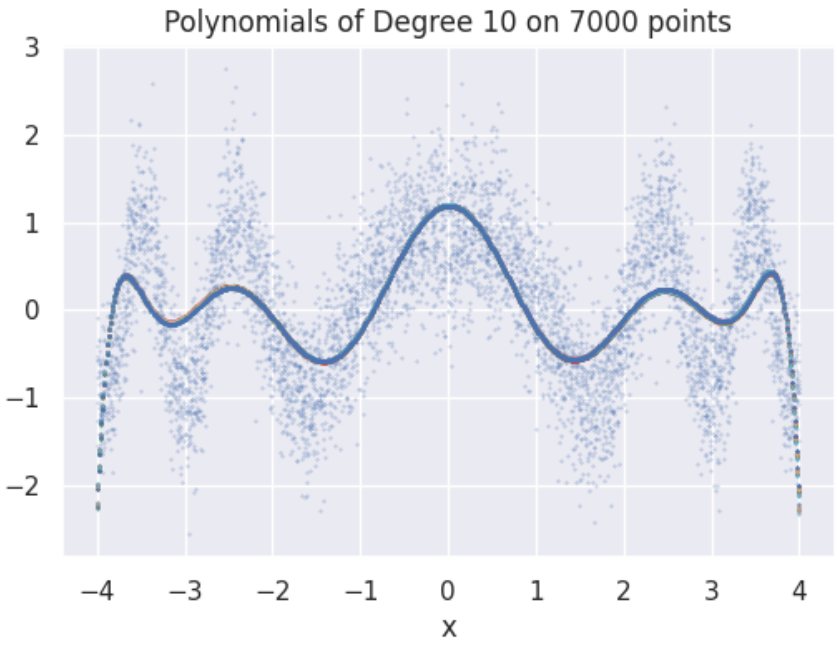


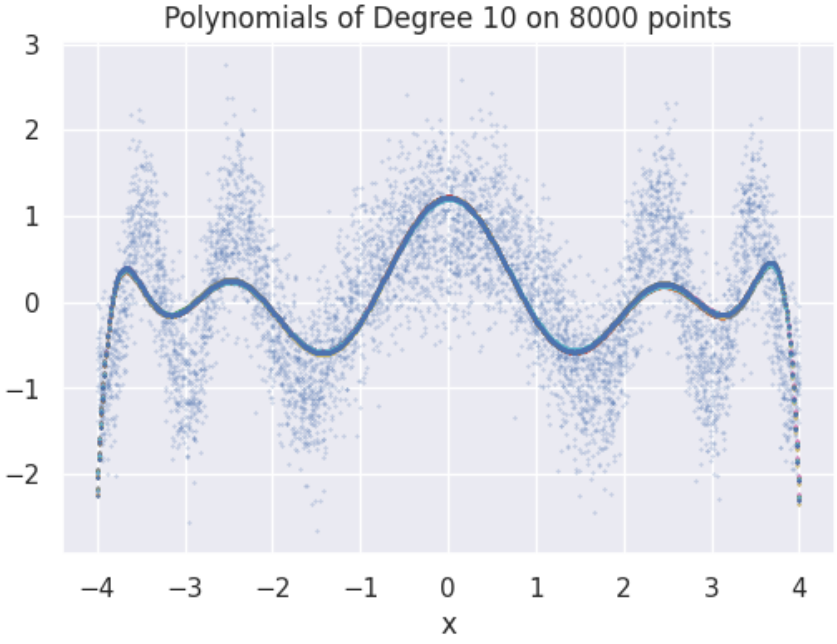

Polynomials of Degree 10 on 3000 points
2
1
0
−1
−2
−4
−3
−2
−1
0
1
2
3
4
x

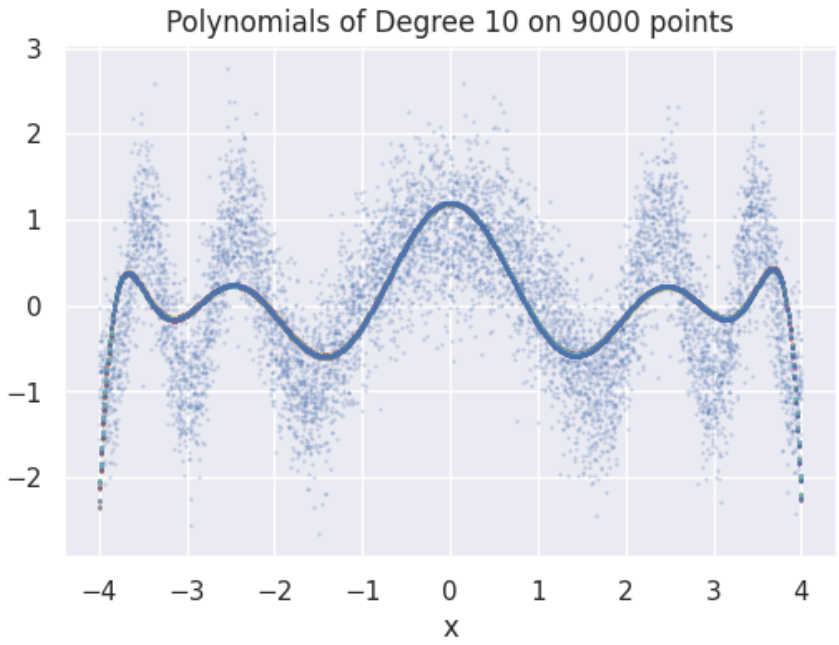

Polynomials of Degree 10 on 4000 points
2
1
0
−1
−2
−4
−3
−2
−1
0
1
2
3
4
x

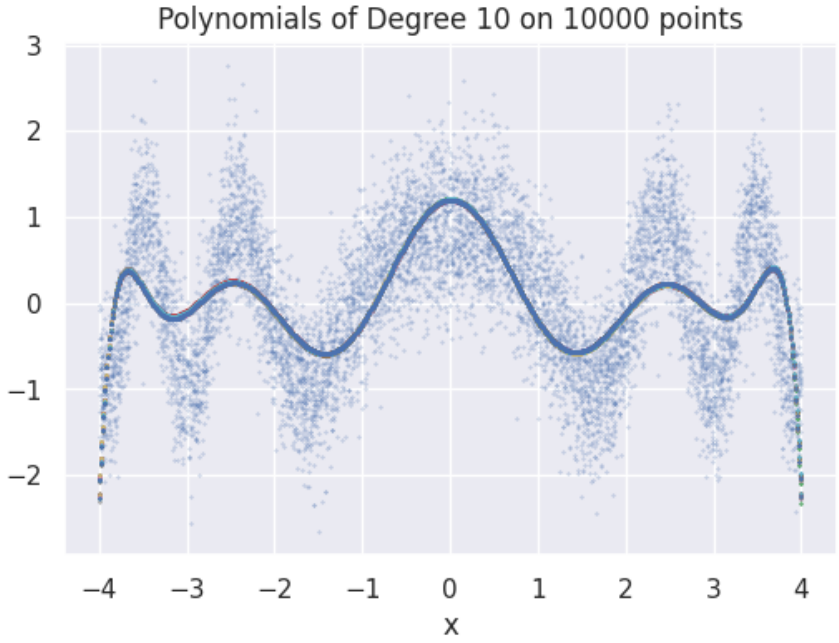

Polynomials of Degree 10 on 5000 points
3
2
1
0
−1
−2
−4
−3
−2
−1
0
1
2
3
4
x


Polynomials of Degree 10 on 6000 points
2
1
0
−1
−2
−4
−3
−2
−1
0
1
2
3
4
x


Polynomials of Degree 10 on 7000 points
3
2
1
0
−1
−2
−4
−3
−2
−1
0
1
2
3
4
x


Polynomials of Degree 10 on 8000 points
3
2
1
0
−1
−2
−4
−3
−2
−1
0
1
2
3
4
x

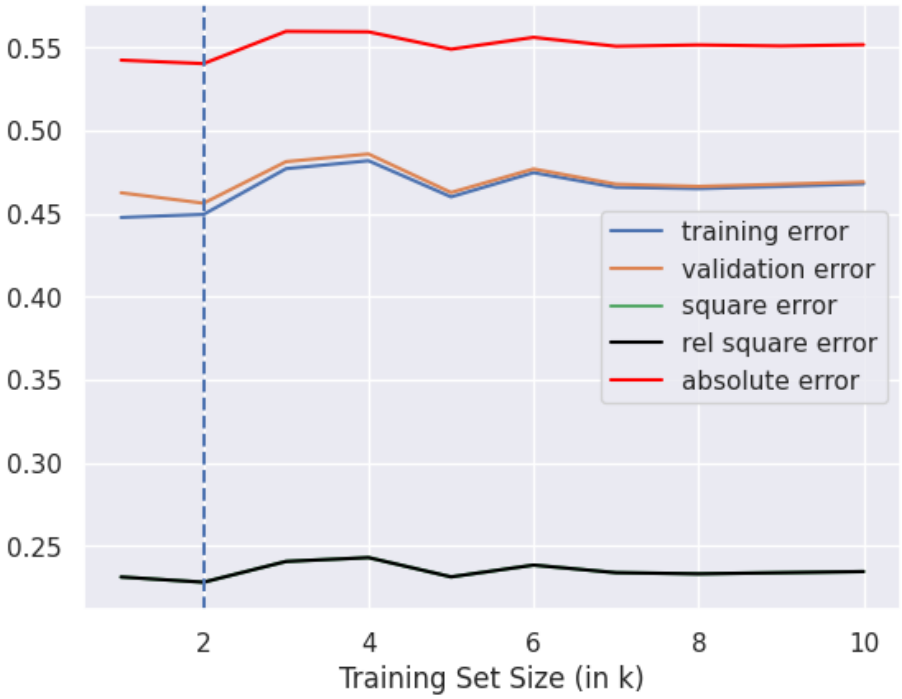




We observe the error measures, bias, variance for each such sample and visualize their behavior. We also observe how training and validation errors vary with the size of the dataset.

Analyzing Bias-Variance for Different Sample Sizes

Analyzing Error with Different Sample Sizes



Observation

As the sample size increases, the variability of the model decreases. This generalization, however, is not true for bias.

Conclusion

For our selected function $f(x)=sin(x^2+2)$, polynomials of order 10 trained with 2,000 instances are the best estimators.