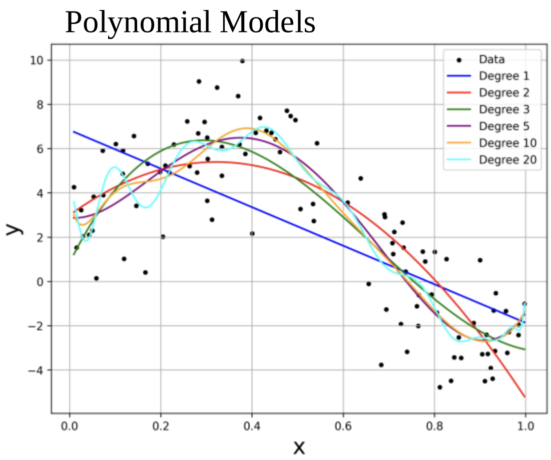
Supervised Learning (Part II)

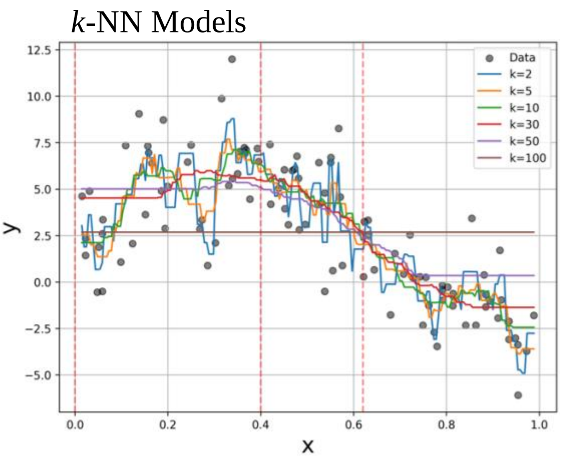
EN5422/EV4238 | Fall 2023
w02_supervised_2.pdf
(Week 2 – 2/2)

Contents

1 Training Data and Model Fits

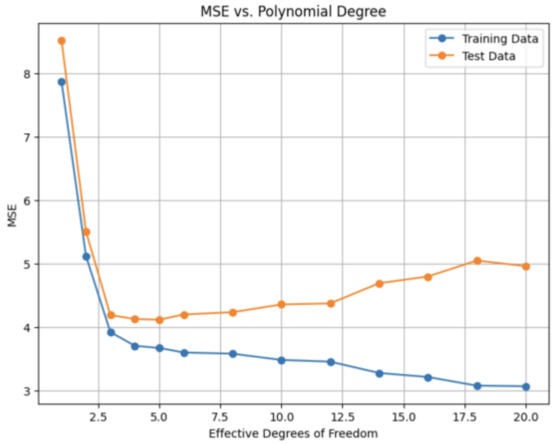
Recall from the last class that we consider several models from two model families (*k*-NN and polynomial).

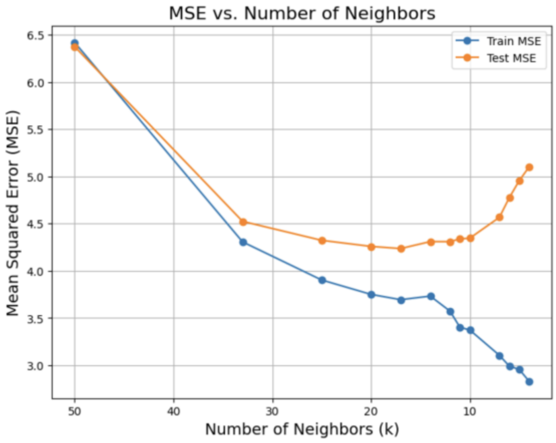

Polynomial Models

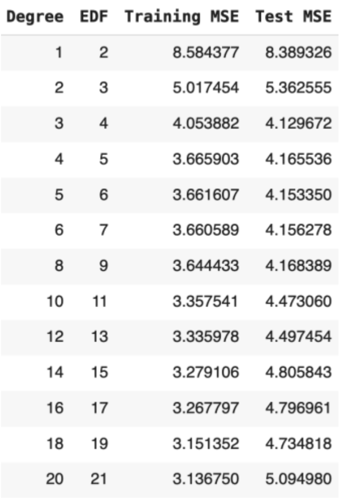

k-NN Models

2 Evaluate Simulated Test Data (or which model is best)

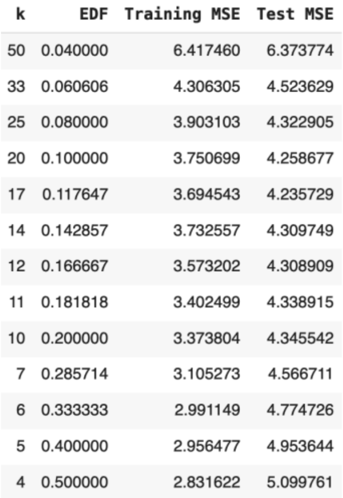
I simulated 50,000 test observations, evaluated the predictions from each model, and recorded the estimated MSE/Risk.







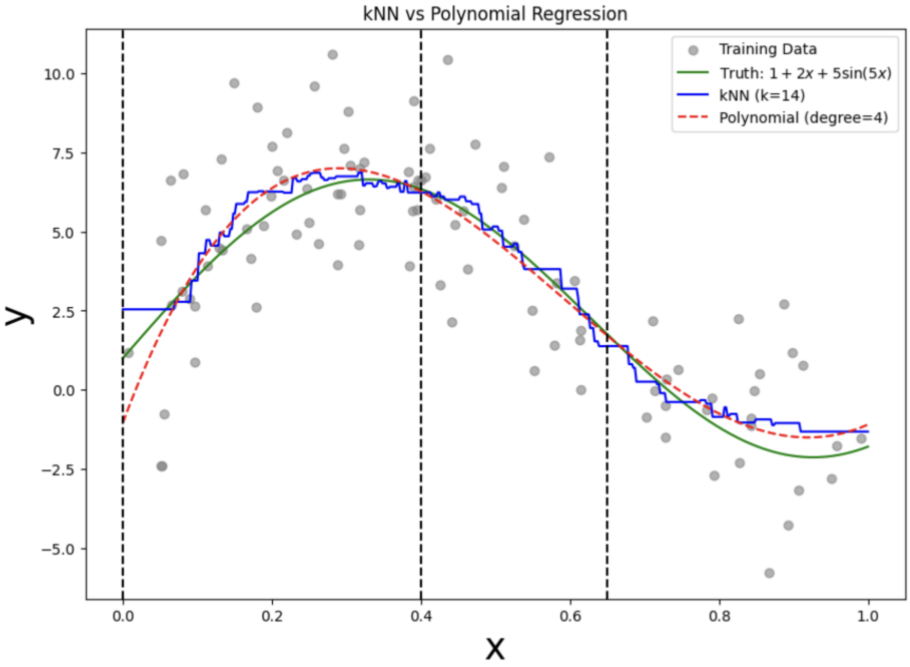
Degree	EDF	Training MSE	Test MSE
1	2	8.584377	8.389326
2	3	5.017454	5.362555
3	4	4.053882	4.129672
4	5	3.665903	4.165536
5	6	3.661607	4.153350
6	7	3.660589	4.156278
8	9	3.644433	4.168389
10	11	3.357541	4.473060
12	13	3.335978	4.497454
14	15	3.279106	4.805843
16	17	3.267797	4.796961
18	19	3.151352	4.734818
20	21	3.136750	5.094980



k	EDF	Training MSE	Test MSE
50	0.040000	6.417460	6.373774
33	0.060606	4.306305	4.523629
25	0.080000	3.903103	4.322905
20	0.100000	3.750699	4.258677
17	0.117647	3.694543	4.235729
14	0.142857	3.732557	4.309749
12	0.166667	3.573202	4.308909
11	0.181818	3.402499	4.338915
10	0.200000	3.373804	4.345542
7	0.285714	3.105273	4.566711
6	0.333333	2.991149	4.774726
5	0.400000	2.956477	4.953644
4	0.500000	2.831622	5.099761

Observations:

- As the flexibility increases, both classes of model *overfit.*
  - overfit means model is too complex.
  - underfit means model is not complex enough.
  - see discrepancy between training and test performance.
- The polynomial with degree = 4 has the best test performance with an approximate MSE = 4.12.
- The optimal MSE = 4.
  - I only know this because I know how the data was generated.



3 Ensemble Models

Last class you gave your votes for which model you thought was best:

model	edf	Number of votes
knn (k=10)	10	5
poly (deg=5)	6	3
poly (deg=3)	4	2
knn (k=5)	20	2
knn (k=20)	5	2
poly (deg=2)	3	2
knn (k=25)	4	2

- Can we use the collective *wisdom of the crowds* to help make a better prediction?
- An *ensemble model* is one that combines several models together.

The approach is to create a new ensemble model that is a weighted sum of the individual models:

$$f_w(x) = \sum_{j=1}^{p} w_j f_j(x)$$

In our specific example, we had:

$$\hat{f}_w(x) = \frac{5}{18} f_{\text{knn}}(x, k = 10) + \frac{3}{18} f_{\text{poly}}(x, \text{deg} = 5) + \frac{3}{18} f_{\text{poly}}(x, \text{deg} = 3) + \cdots + \frac{2}{18} f_{\text{knn}}(x, k = 25)$$

and the corresponding **test** performance is given by:

$$R = \frac{1}{M} \sum_{j=1}^{M} \left(y_j - \hat{f}_w(x_j)\right)^2$$

where *M* is the number of test observations.
This gives a **test** MSE of 4.17, which is better than most individual models:

model	edf	w	MSE
poly (deg=3)	3	0.13	4.13
poly (deg=5)	6	0.26	4.13
ensemble	N/A	N/A	4.17
knn (k=20)	2	0.09	4.40
knn (k=10)	7	0.30	4.42
knn (k=25)	1	0.04	4.69
knn (k=5)	2	0.09	5.03

4 Bias-Variance Trade-off

This section explore the bias-variance trade-off for the examples we covered last class. This involves examining the theoretical properties of an estimator.

4.1 Bias-Variance Trade-off

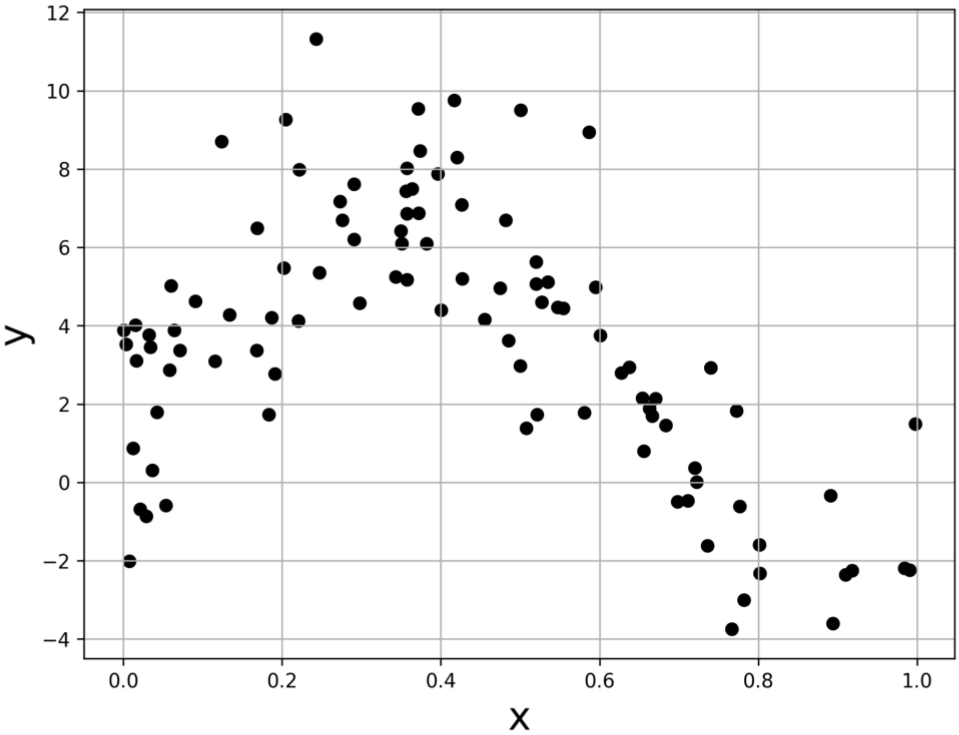
Here, we set the data generation function. $X \sim U[0,1]$ and $f(x) = 1 + 2x + 5\sin(5x)$ and $y(x) = f(x) + \epsilon$, where $\epsilon \overset{\text{iid}}{\sim} N(0,2)$.

```
# Simulation functions
def sim_x(n):
    return np.random.rand(n)
def f(x):
    return 1 + 2*x + 5*np.sin(5*x)
def sim_y(x, sd):
    n = len(x)
    return f(x) + np.random.normal(0, sd, n)
# Simulation settings
n = 100  # number of observations
```

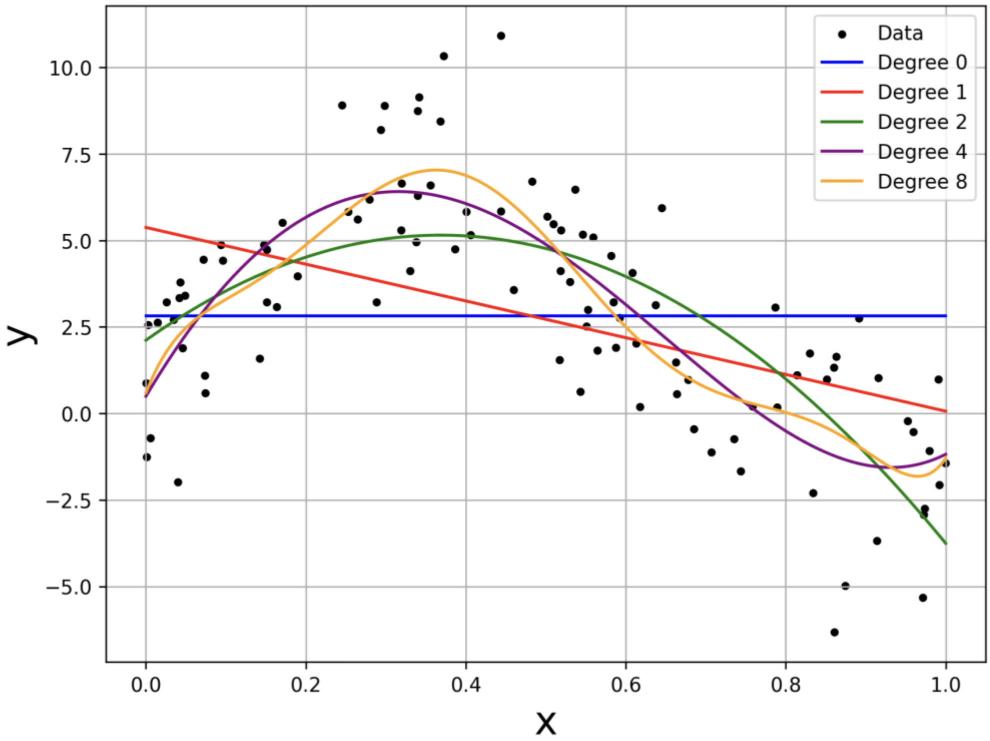
```
sd = 2  # standard deviation for error
```

4.2 One Realization

Last class, we explore one realization from this system.

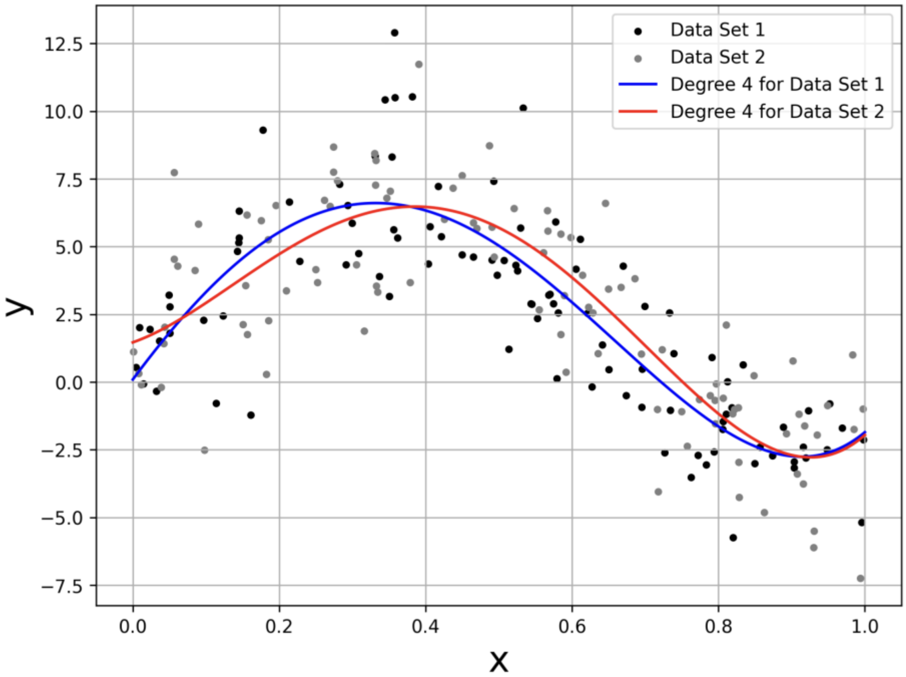


And then fit several polynomial regression models. Recall by polynomial regression, I mean using a predictor function $\hat{y}(x) = f(x, d) = \sum_{j=0}^{d} x^j \beta_j$ where $d \in \{0,1, \dots\}$ is the degree.



4.3 A second realization

Suppose we drew another training set (using same distributions and sample size n):



- We get another fitted curve using the new training data.
- While the two curves are visually similar, they are not identical.
- If we took more training samples, we would get more fitted curves.

- What we want to study in this section is the likelihood that we will happened to get a *good* fit given a single training data set.

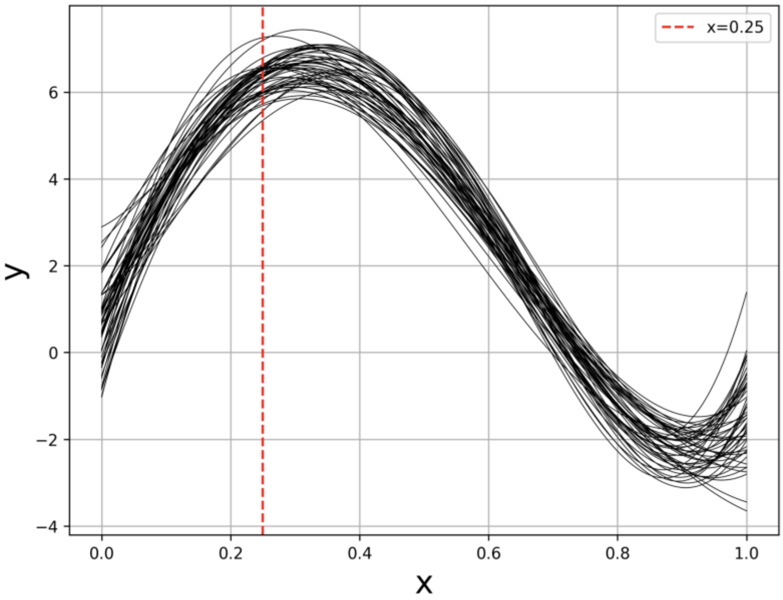
4.4 Bias, Variance, and Mean Squared Error (MSE)

- The statistical properties of an estimator can help us understand its potential performance.
- Let $D = [(X_1, Y_1), (X_2, Y_2), \dots, (X_n, Y_n)]$ be *training data.*
- Let $\hat{\theta} = \hat{\theta}(D)$ be the estimated parameter *calculated from the training data D.*
  - E.g., $\theta = f(x)$, $\hat{\theta} = \hat{f}(x|D)$
  - $\hat{\theta}$ is a random variable; it has a distribution.

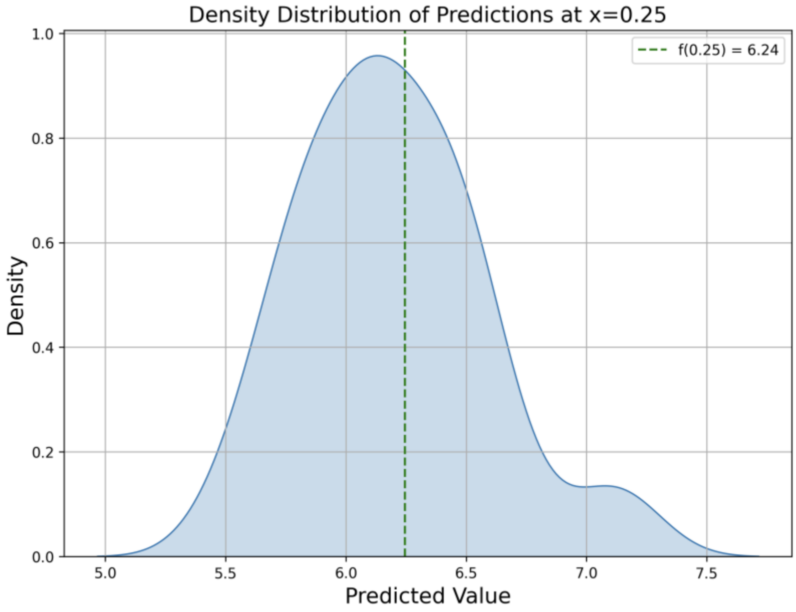
4.4.1 Bias, Variance, and Mean Squared Error (MSE)

- Consider the distribution of $\hat{\theta} = \hat{f}_{poly}(0.25, d = 4)$.
  - This is the distribution of the fit at $x = 0.25$ from a polynomial of degree 4 using different *training* sets
- I generated 50 different training data sets (each with n = 100), fit a polynomial (deg=4) model to each data set, and recorded the estimated at x = 0.25.

Predictions from 50 different training sets



distribution of $\hat{f}(0.25, 4)$
Optimal/True f(x) given by green dotted line



4.4.2 Some properties of an estimator

- **Bias** of an estimator is defined as $E_D[\hat{\theta}] - \theta$
- **Variance** of an estimator is defined as $V_D[\hat{\theta}] = E_D[\hat{\theta}^2] - E_D[\hat{\theta}]^2$
- **MSE** of an estimator is defined as:

$$\begin{aligned} MSE(\hat{\theta}) &= E_D\left[(\hat{\theta} - \theta)^2\right] \\ &= V_D[\hat{\theta} - \theta] + E_D[\hat{\theta} - \theta]^2 \\ &= V_D[\hat{\theta}] + E_D[\hat{\theta} - \theta]^2 \text{ Bias-Variance decomposition} \end{aligned}$$

Note: θ is just a constant (i.e., the true parameter value)

- Estimators are often evaluated based on MSE, being unbiased, and/or having minimum variance (out of all unbiased estimators)
- These properties are based on the *distribution of an estimate.*
  - Once we observe the training data, the resulting estimate may be great or horrible.
  - However these theoretical properties provide insight into what we can expect and how much confidence we can have in the estimates.

4.5 Estimating the Bias, Variance, and Mean Squared Error (MSE)

- Last class, we examined the Risk (e.g., MSE) *conditioning on the training data* (See Section 6.2.1).
- Now we will relax this and bring in the uncertainty in the training data D.

Under a squared error loss function $L(Y, f(X)) = (Y - f(X))^2$, the *overall* Risk (or Risk before we see any training data) at a particular $X = x$ is:

$$\begin{aligned}\mathrm{MSE}_x(f) &= \mathrm{E}_{DY|X}\left[\left(Y - \hat{f}_D(x)\right)^2 \middle| X = x\right]\\ &= \mathrm{V}[Y|X = x] + \mathrm{V}\left[\hat{f}_D(x)\middle|X = x\right] + \left(\mathrm{E}\left[\hat{f}_D(x)\middle|X = x\right] - f(x)\right)^2\\ &= irreducible\ error + model\ variance + model\ squared\ bias\end{aligned}$$

where *D* is the training data, *f* is the true model, and $\hat{f}_D(x)$ is the prediction at $X = x$ estimated from the training data *D*.

Note

$$\begin{aligned}\mathrm{MSE}_x\ (f) &= \mathrm{E}_{DY|X}\left[\left(Y - \hat{f}_D(x)\right)^2 \mid X = x\right]\\ &= \mathrm{E}_{DY|X}\left[\left(Y - f(x) + f(x) - \hat{f}_D(x)\right)^2 \mid X = x\right]\\ &= \mathrm{E}_{DY|X}[(Y - f(x))^2] + \mathrm{E}_{DY|X}\left[f(x) - \hat{f}_D(x)\right]^2 + \mathrm{E}_{DY|X}\left[2\big(Y - f(x)\big)\left(f(x) - \hat{f}_D(x)\right)\right]\\ &= \mathrm{V}[Y \mid X = x] + \mathrm{E}_{DY|X}[f(x) - \hat{f}_D(x)]^2 + 0\\ &= \mathrm{V}[Y \mid X = x] + \mathrm{V}_{DY|X}\big(\hat{f}_D(x)\big) + \mathrm{E}_{DY|X}\big(f(x) - \hat{f}_D(x)\big)^2\end{aligned}$$

Note:

- $\mathrm{E}_{DY|X}[Y] = Y$
- $\mathrm{E}_{DY|X}[f(x)] = f(x)$
- $\mathrm{E}_{DY|X}\left[\hat{f}_D\right] = f(x)$
- $\mathrm{V}[X] = \mathrm{E}[X^2] - (\mathrm{E}[X])^2$

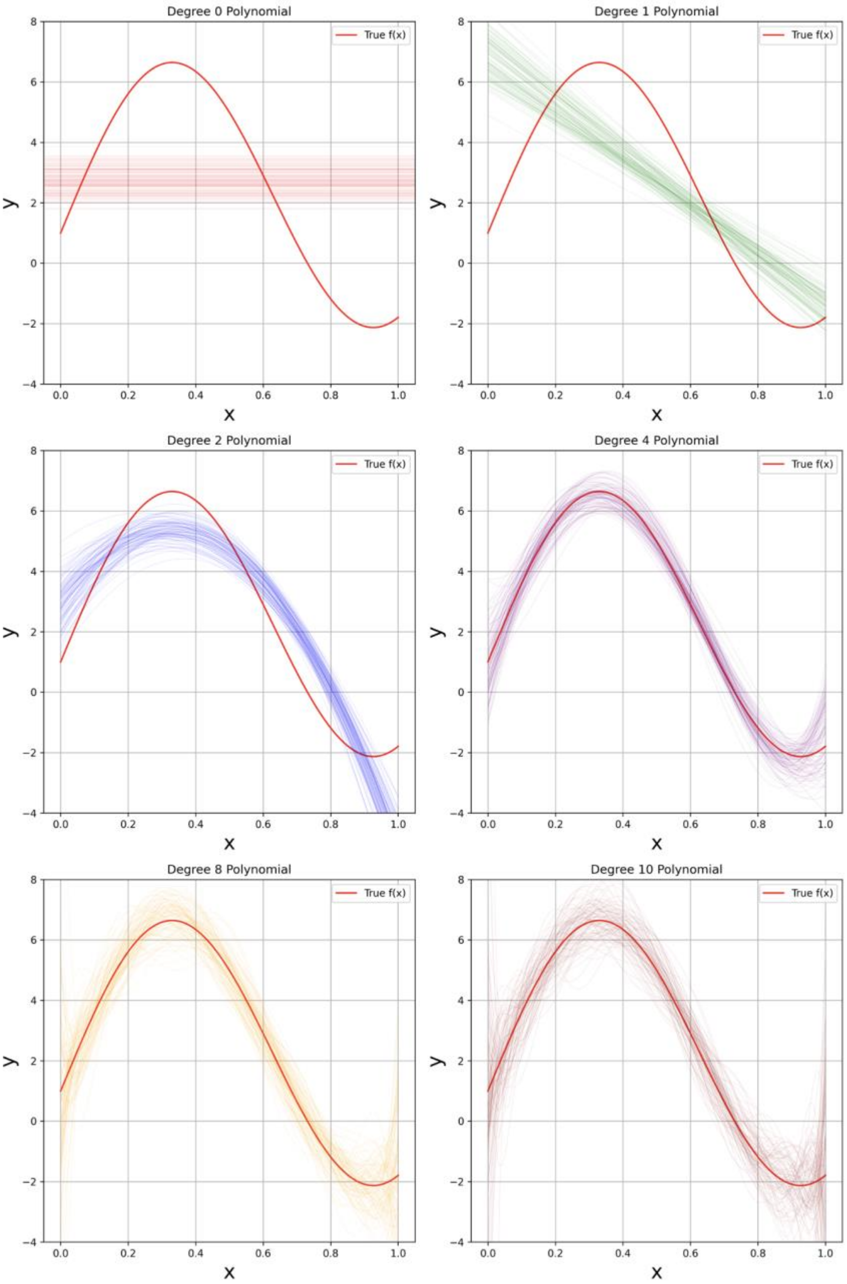
- We can estimate the model variance and bias with simulation.
  - Generate new data $D_m = \{(Y_i, X_i)\}_{i=1}^n$ for simulation $m = 1,2, \dots, M$ (use the same sample size *n*).
  - Fit the models with data D_m getting $\hat{f}_D(\cdot)$.
  - Now we can estimate the items of interest:

$$\mathrm{E}\left[\hat{f}_D(x)\right] \approx \bar{f}(x) = \frac{1}{M}\sum_{m=1}^{M} \hat{f}_{D_m}(x)$$

$$\mathrm{V}\left[\hat{f}_D(x)\right] \approx s_f^2(x) = \frac{1}{M-1}\sum_{m=1}^{M} \left(\hat{f}_{D_m}(x) - \bar{f}(x)\right)^2$$

4.5.1 Simulation

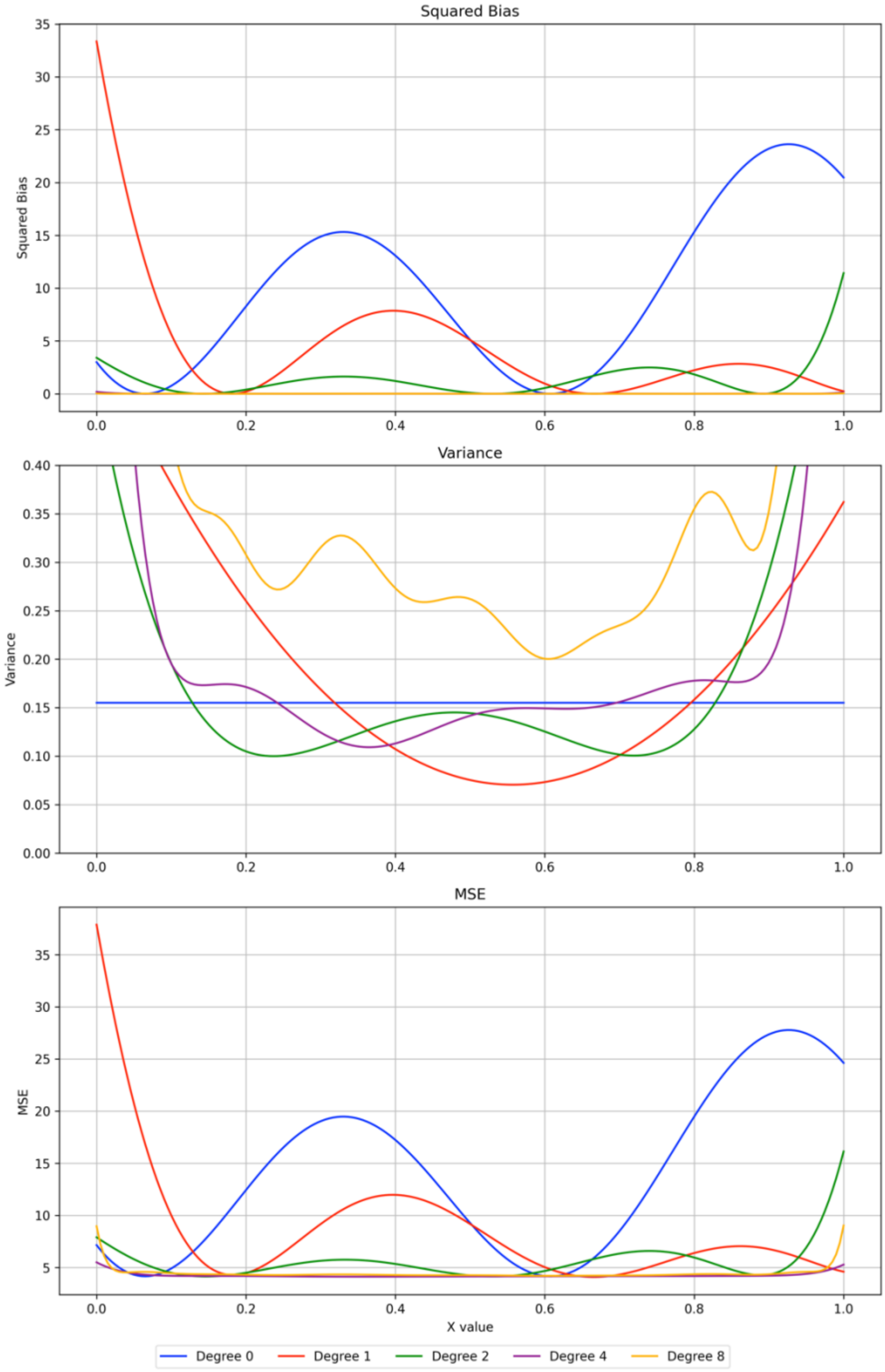
I ran 100 simulations to generate $\{\hat{f}_m(x, deg = d): d \in \{0,1,2,4,8\}, m \in \{1,2, \ldots ,100\}\}$



4.5.2 Observations

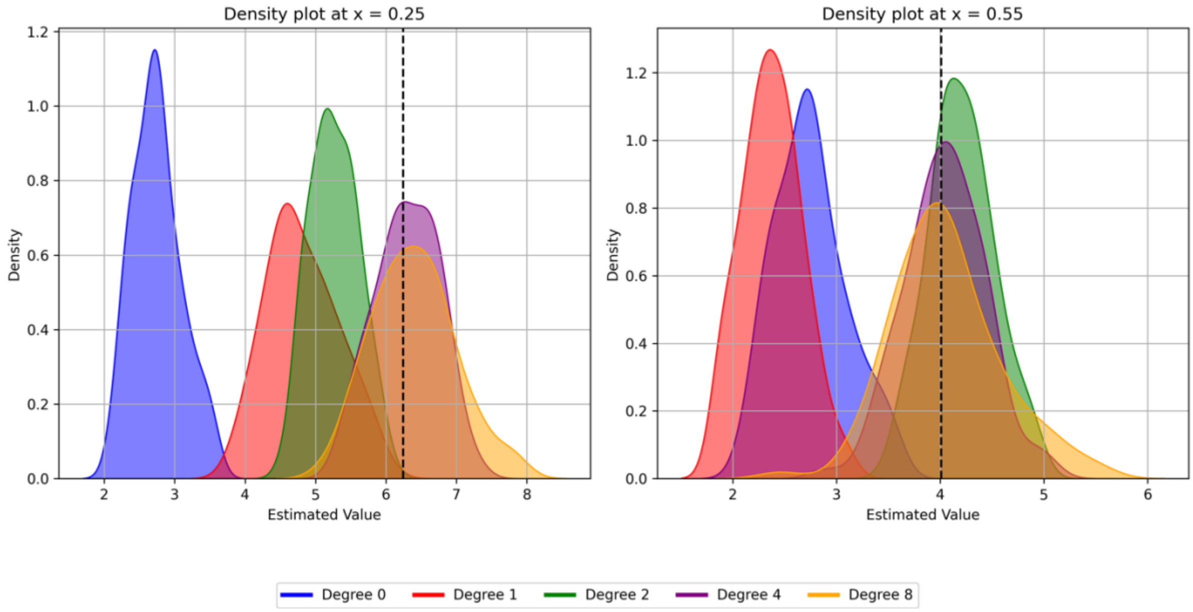
- This shows the bias and variance of each model.
- You can se that as the flexibility (e.g., degree) of the model increases, the bias decreases but the variance (especially at the edges) increases.
    - The **bias** is the difference between the true regression function (dark gray line) and the model mean (dark colored line).

- The **variation** is seen in the width of the transparent curves, one for each simulations.



4.5.3 Bias, Variance, and MSE at a single input

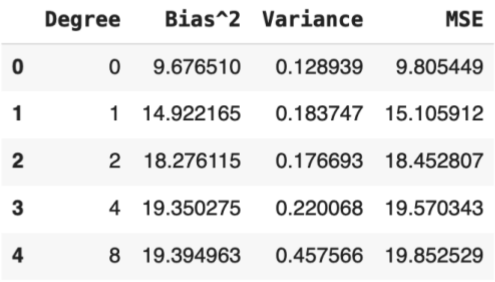
- Notice that model variance and model bias vary over *x*.
- To help see what is going on, we now look at the distribution at *x* = 0.25 and *x* = 0.55.



4.5.4 Bias, Variance, and MSE at a single input

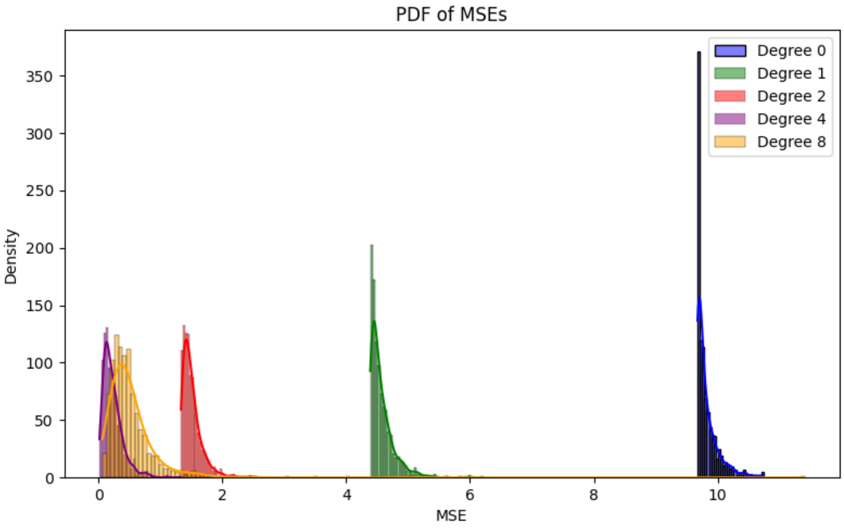
The above analysis examines the $\text{MSE}_x(f)$ over a set of $x's$. However, in a real setting, the overall test error will be based on *all* of the actual test X values.

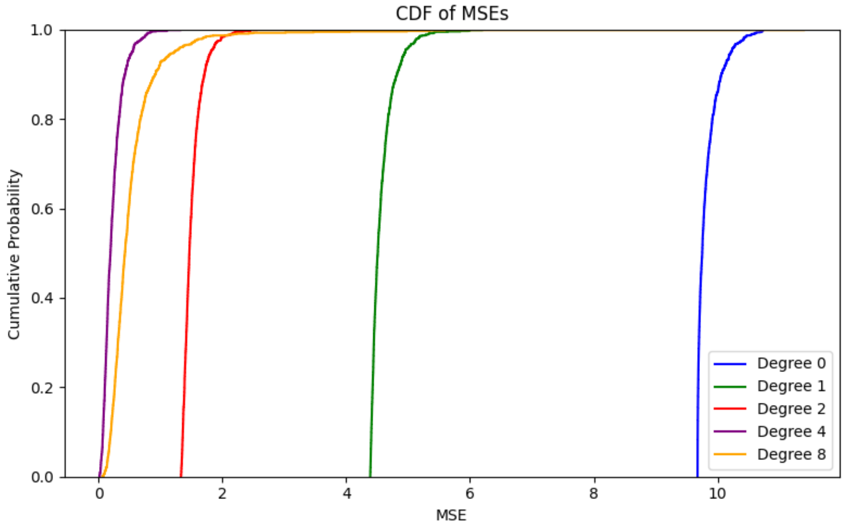
$$
\begin{aligned}
&\text{MSE}(f) = \text{E}_{DY|X}\left[\left(Y - \hat{f}_D(X)\right)^2\right] \\
&= \text{E}_X[\text{MSE}_x(f)\,\Pr(dx)] \\
&= \int \text{MSE}_x(f)\Pr\,(dx)
\end{aligned}
$$



	Degree	Bias^2	Variance	MSE
0	0	9.676510	0.128939	9.805449
1	1	14.922165	0.183747	15.105912
2	2	18.276115	0.176693	18.452807
3	4	19.350275	0.220068	19.570343
4	8	19.394963	0.457566	19.852529

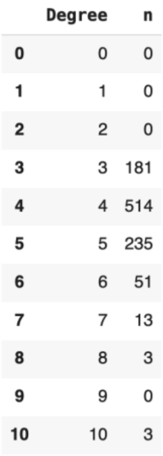
4.6 What does it all mean?





While its possible that we could just happen to get a particular training data realization that favors a model other than the globally optimal model, this is unlikely for the bad models. However, it is not uncommon for “close” models.

Below is a table of the number of simulations that each model had the best MSE:



	Degree	n
0	0	0
1	1	0
2	2	0
3	3	181
4	4	514
5	5	235
6	6	51
7	7	13
8	8	3
9	9	0
10	10	3

- While the degree=4 model does best, degree > 8 sometimes comes out best.
- **Conclusion 1**: In our toy example, a polynomial with degree=4 is the best model, in principal. However, for some data (i.e., some training data sets) the models with degree>4 and degree<4 would predict better.

- **Conclusion 2**: The above analysis is what is meant by the "bias-variance trade-off".
    - In reality, we only get to observe one realization of the training data so we can never actual estimate the bias and variance the way we did above.
    - But we can still estimate the Risk (e.g., MSE) by using resampling methods like cross-validation or statistical methods like Bayesian Information Criterion (BIC).
    - More loosely, when people mention bias-variance trade-off, they are referring to the principal that the best model is one that has just the right flexibility.
    - If the model is too complex, it is unlikely to produce a good estimate (across the entire range of inputs) because it is likely to stray far from the expected mean at certain values.
    - If the model is no complex enough, then it will not track with the expected mean across the range of input values and thus produce poor overall performance.

- **Conclusion 3**: Performance of a model can vary across the input features *X*.
    - If you are only concerned about performance in a specific range of *X*, then emphasize these during training (e.g., weight observations close to *X* more heavily during model estimation).
    - If the test *X* values are coming from a different distribution than the training *X* values, then your model may not be optimal.