Supervised Learning (Part II)

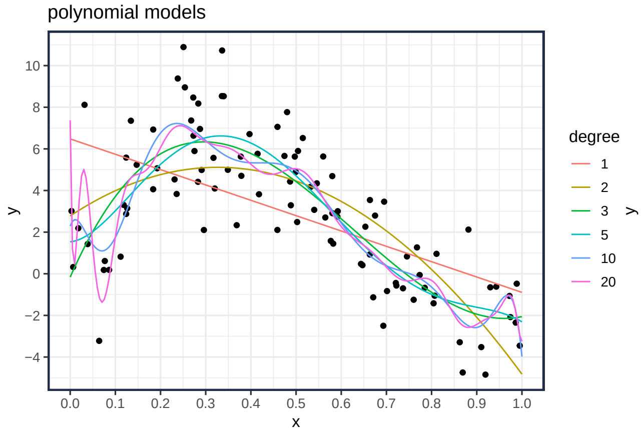
DS 6030 | Fall 2021

supervised_2.pdf

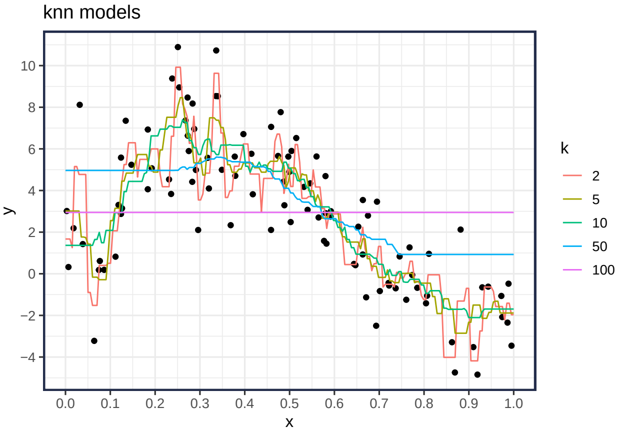
Contents

1 Training Data and Model Fits

Recall from last class that we are consider several models from two model families (knn and polynomial).



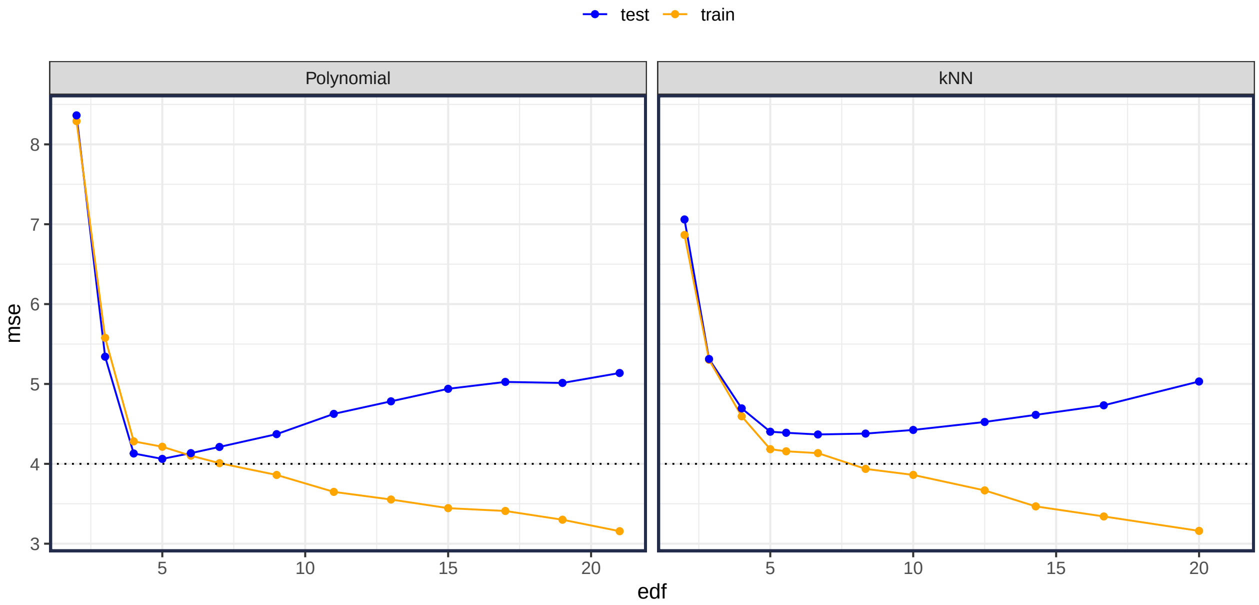
polynomial models



knn 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.



Polynomial

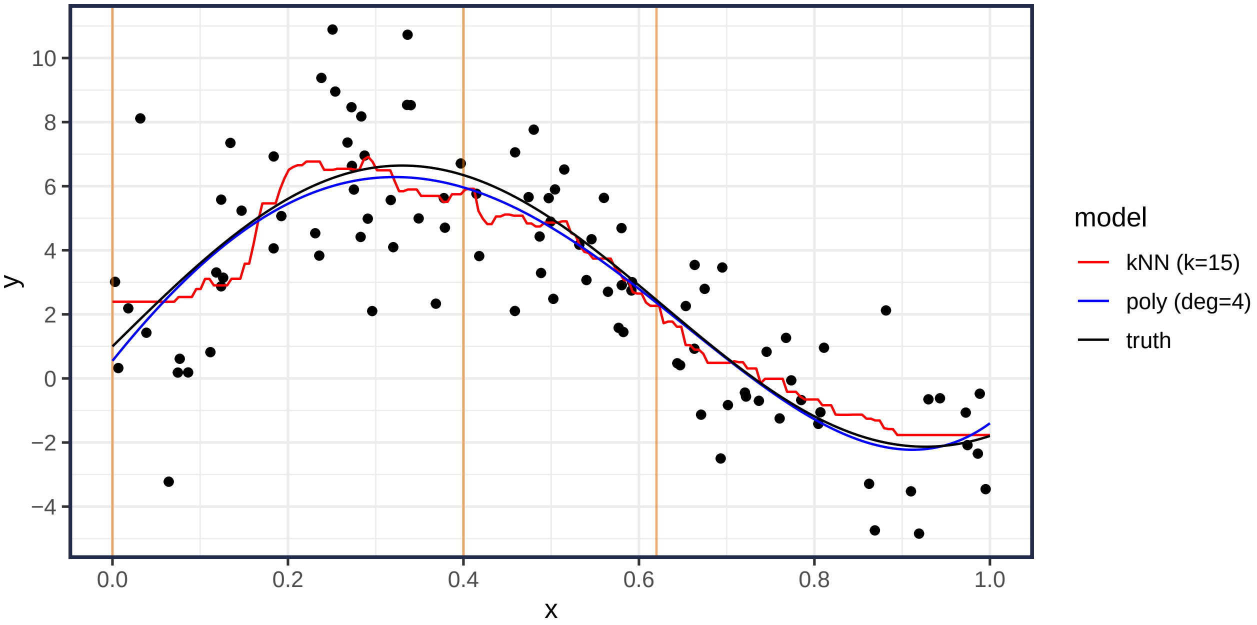
degree	edf	mse.train	mse.test
1	2	8.29	8.36
2	3	5.58	5.34
3	4	4.28	4.13
4	5	4.21	4.06
5	6	4.10	4.13
6	7	4.01	4.21
8	9	3.86	4.37
10	11	3.65	4.63
12	13	3.55	4.78
14	15	3.44	4.94
16	17	3.41	5.03
18	19	3.30	5.01
20	21	3.16	5.14

kNN

k	edf	mse.train	mse.test
50	2.00	6.87	7.06
35	2.86	5.30	5.31
25	4.00	4.59	4.69
20	5.00	4.18	4.40
18	5.56	4.16	4.39
15	6.67	4.13	4.37
12	8.33	3.94	4.38
10	10.00	3.86	4.42
8	12.50	3.67	4.52
7	14.29	3.47	4.61
6	16.67	3.34	4.73
5	20.00	3.16	5.03

Observations:

- as the complexity increases, both classes of models *overfit*.
  - overfit means model 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 $\text{MSE} = 4.06$.
- The optimal $\text{MSE} = 4$.
  - I only know this because I know how the data was generated


10
8
6
4
2
0
−2
−4
y
0.0
0.2
0.4
0.6
0.8
1.0
x
model
kNN (k=15)
poly (deg=4)
truth

3 Ensemble Models

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

model	edf	number of votes
knn (k=10)	10.0	13
poly (deg=3)	4.0	12
poly (deg=2)	3.0	8
knn (k=20)	5.0	5
poly (deg=5)	6.0	5
knn (k=5)	20.0	4
poly (deg=10)	11.0	4
knn (k=2)	50.0	1
knn (k=3)	33.3	1
knn (k=15)	6.7	1

- 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

$$\hat{f}_w(x) = \frac{13}{54} f_{\text{knn}}(x, k = 10) + \frac{12}{54} f_{\text{poly}}(x, deg = 3) + \frac{8}{54} f_{\text{poly}}(x, deg = 2) + ... + \frac{1}{54} f_{\text{knn}}(x, k = 15)$$

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

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

where M is the number of test observations.

This gives a **test** MSE of 4.21, which is better than most individual models

model	n	w	mse
poly(deg=3)	12	0.22	4.13
poly(deg=5)	5	0.09	4.13
ensemble	NA	NA	4.21
knn(k=15)	1	0.02	4.37
knn(k=20)	5	0.09	4.40
knn(k=10)	13	0.24	4.42
poly(deg=10)	4	0.07	4.63
knn(k=5)	4	0.07	5.03
poly(deg=2)	8	0.15	5.34
knn(k=3)	1	0.02	5.57
knn(k=2)	1	0.02	6.57

4 Bias-Variance Trade-off

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

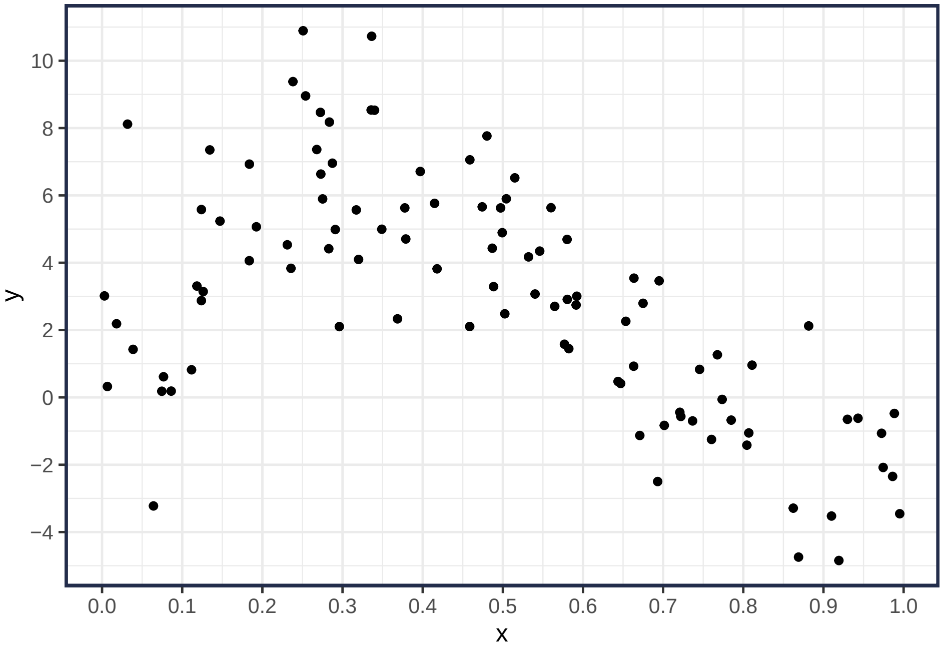
4.1 Data Generating Functions

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

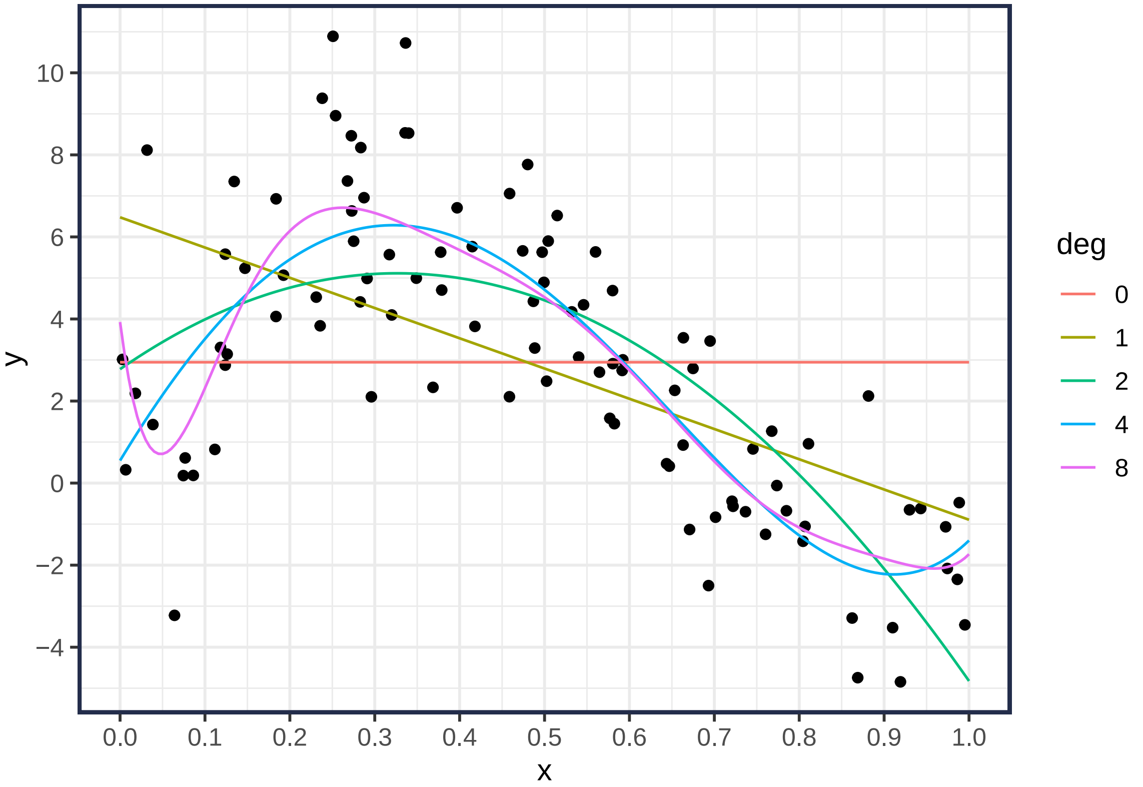
```
#- Simulation functions
sim_x <- function(n) runif(n)            # U[0,1]
f <- function(x) 1 + 2*x + 5*sin(5*x)    # true mean function
sim_y <- function(x, sd){                # generate Y|X from N{f(x),sd}
  n = length(x)
  f(x) + rnorm(n, sd=sd)
}
#- Simulation settings
n = 100                                  # number of obs
sd = 2                                   # stdev for error
```

4.2 One Realization

Last class, we explored one realization from this system.

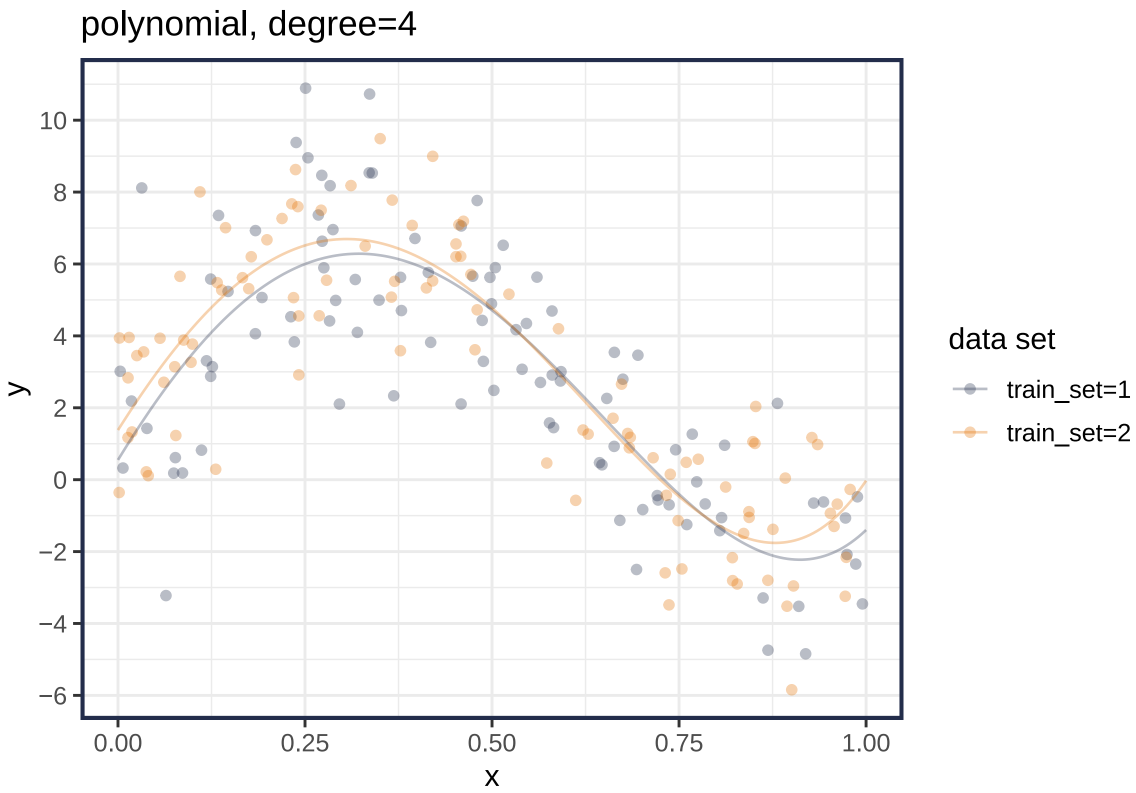


And then fit several polynomial regression models. Recall by polynomial regression I mean using a predictor function $\hat{y}(x) = f(x, d) = \sum_{j=0}^{d} x^j \beta_j$, where $d \in \{0, 1, \dots\}$ is the degree.



4.3 A second realization

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



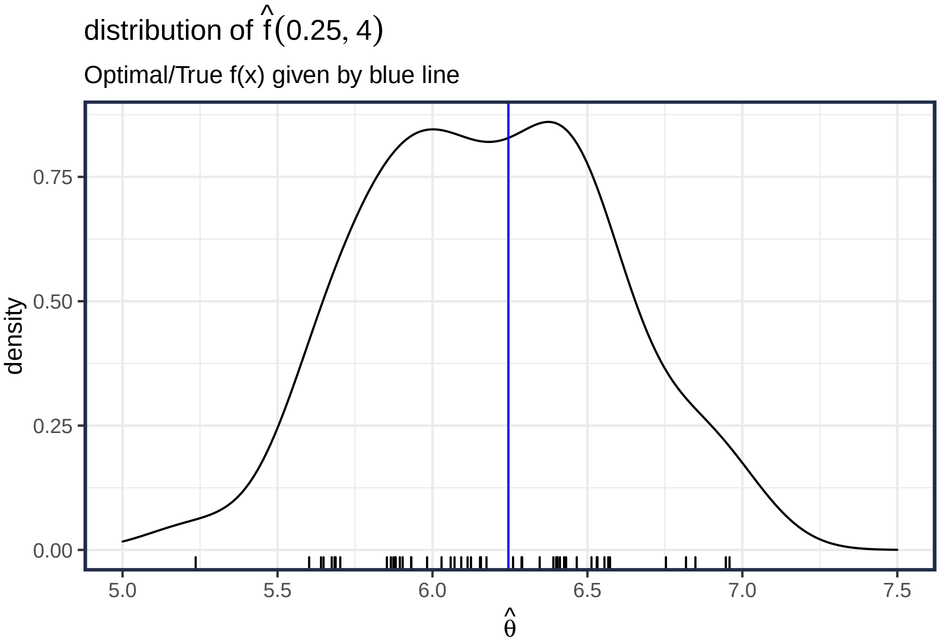
- We get another fitted curve using the new training data.
- While the two curves are visually similar, they are not identical.
- If we took more training samples, we would get more fitted curves
- What we want to study in this section is the likelihood that we will happen to get a *good* fit given a 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 $D = [(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)]$ be *training data*
- Let $\hat{\theta} = \hat{\theta}(D)$ be the estimated parameter *calculated from the training data* D
  - E.g. $\theta = f(x)$, $\hat{\theta} = \hat{f}(x \mid D)$
  - $\hat{\theta}$ is a random variable; it has a distribution.

4.4.1 Distribution of $\hat{\theta}$

- Consider the distribution of $\hat{\theta} = \hat{f}_{\text{poly}}(0.25, d = 4)$.
  - 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 estimate at $x = 0.25$.



4.4.2 Some properties of an estimator

- **Bias** of an estimator is defined as $\mathrm{E}_D[\hat{\theta}] - \theta$
- **Variance** of an estimator is defined as $\mathrm{V}_D[\hat{\theta}] = \mathrm{E}_D[\hat{\theta}^2] - \mathrm{E}_D[\hat{\theta}]^2$
- **MSE** of an estimator is defined as $\mathrm{MSE}(\hat{\theta}) = E_D[(\hat{\theta} - \theta)^2]$

$$\begin{aligned}\mathrm{MSE}(\hat{\theta}) &= E_D[(\hat{\theta} - \theta)^2] \\ &= V_D[\hat{\theta} - \theta] + E_D[\hat{\theta} - \theta]^2 \\ &= V_D[\hat{\theta}] + E_D[\hat{\theta} - \theta]^2\end{aligned}$$

- 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

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

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.

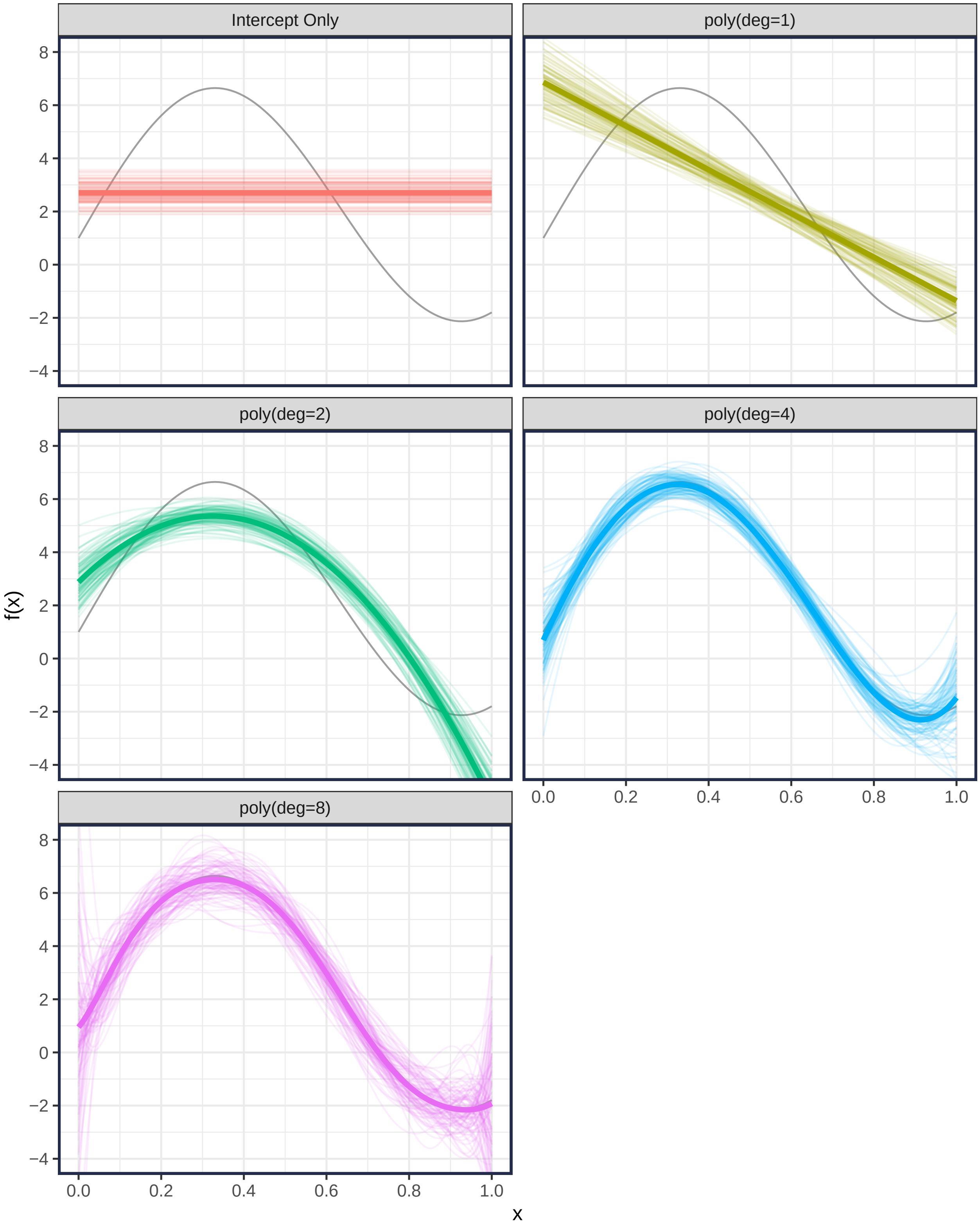
$$\begin{aligned}
\mathrm{MSE}_x(f) &= \mathrm{E}_{DY|X}[(Y - \hat{f}_D(x))^2 \mid X = x] \\
&= \mathrm{E}_{DY|X}[(Y - f(x) + f(x) - \hat{f}_D(x))^2 \mid X = x] \\
&= \mathrm{E}_{DY|X}[(Y - f(x))^2] + \mathrm{E}_{DY|X}(f(x) - \hat{f}_D(x))^2 + \mathrm{E}_{DY|X}\, 2(Y - f(x))(f(x) - \hat{f}_D(x)) \\
&=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

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

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

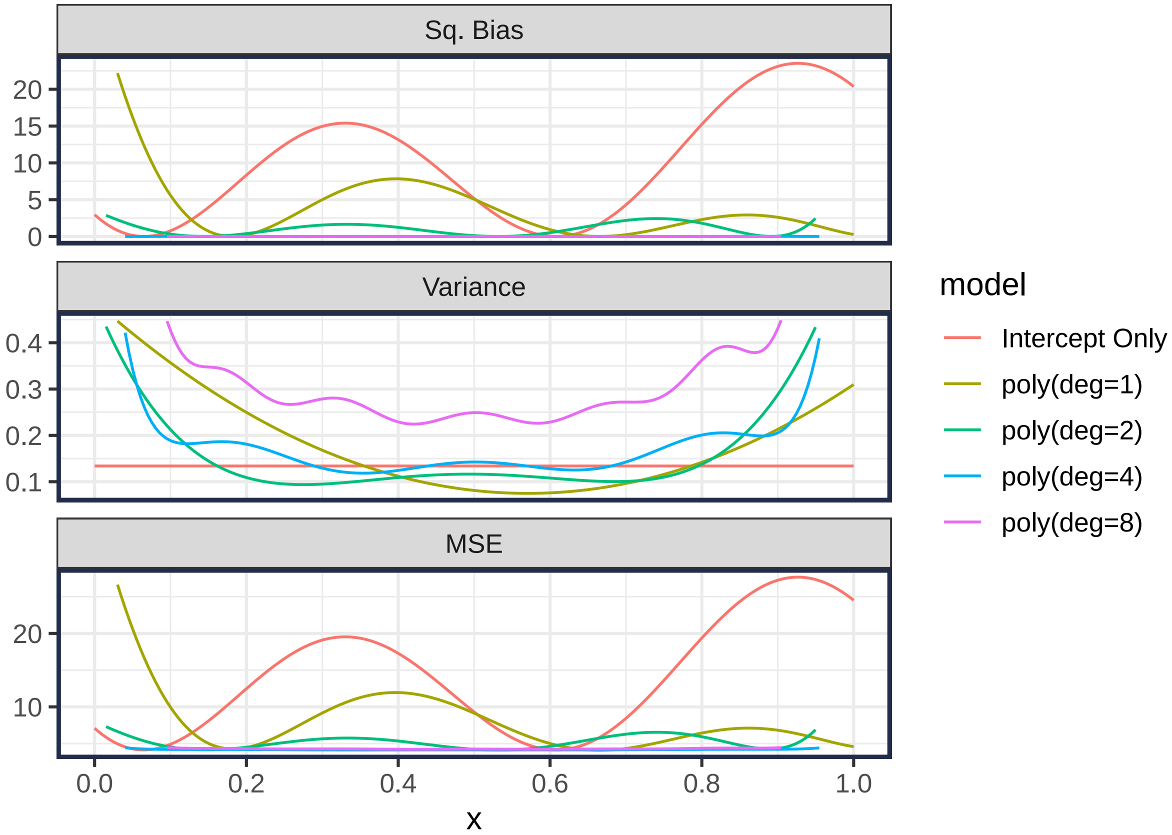
4.5.1 Simulation

I ran 2000 simulations to generate $\{\hat{f}_m(x, deg = d) : d \in \{0, 1, 2, 4, 8\}, m \in \{1, 2, \ldots, 2000\}\}$.



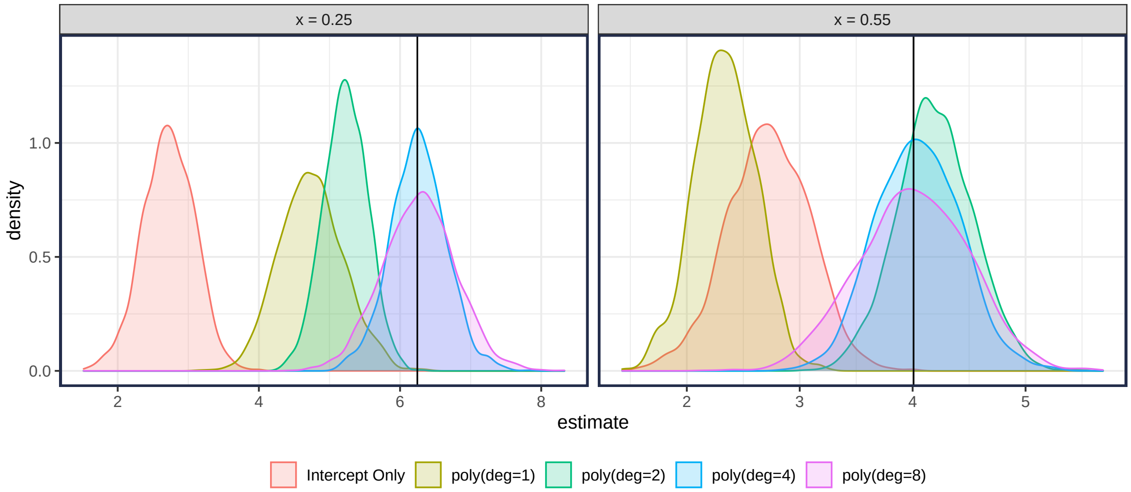
4.5.2 Observations

- This shows the bias and variance of each model.
- You can see that as the complexity (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 simulation.



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 distributions at $x = 0.25$ and $x = 0.55$.



4.5.4 Integrated MSE

The above analysis examines the $\mathrm{MSE}_x(f)$ 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. So we are usually more interested in the *integrated* MSE:

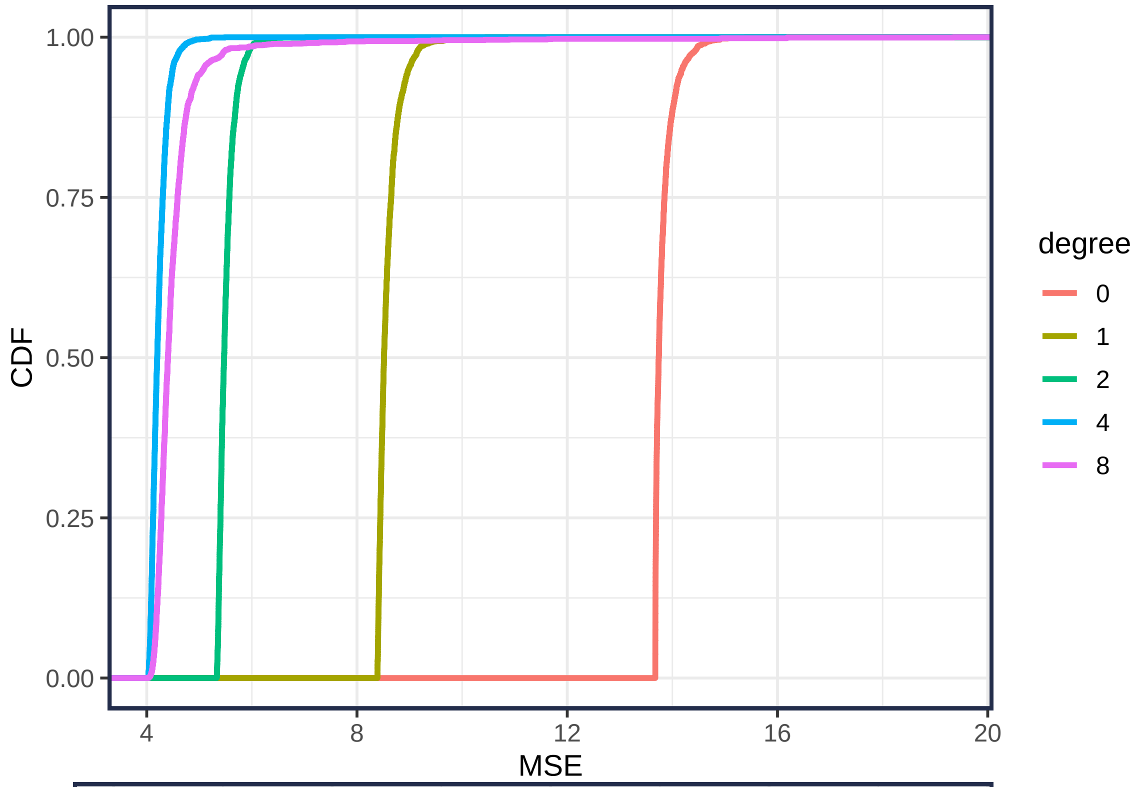
$$\begin{aligned}\mathrm{MSE}(f) &= \mathrm{E}_{DYX}[(Y - \hat{f}_D(X))^2] \\ &= E_X[\mathrm{MSE}_X(f)] \\ &= \int \mathrm{MSE}_x(f)\,\Pr(dx)\end{aligned}$$

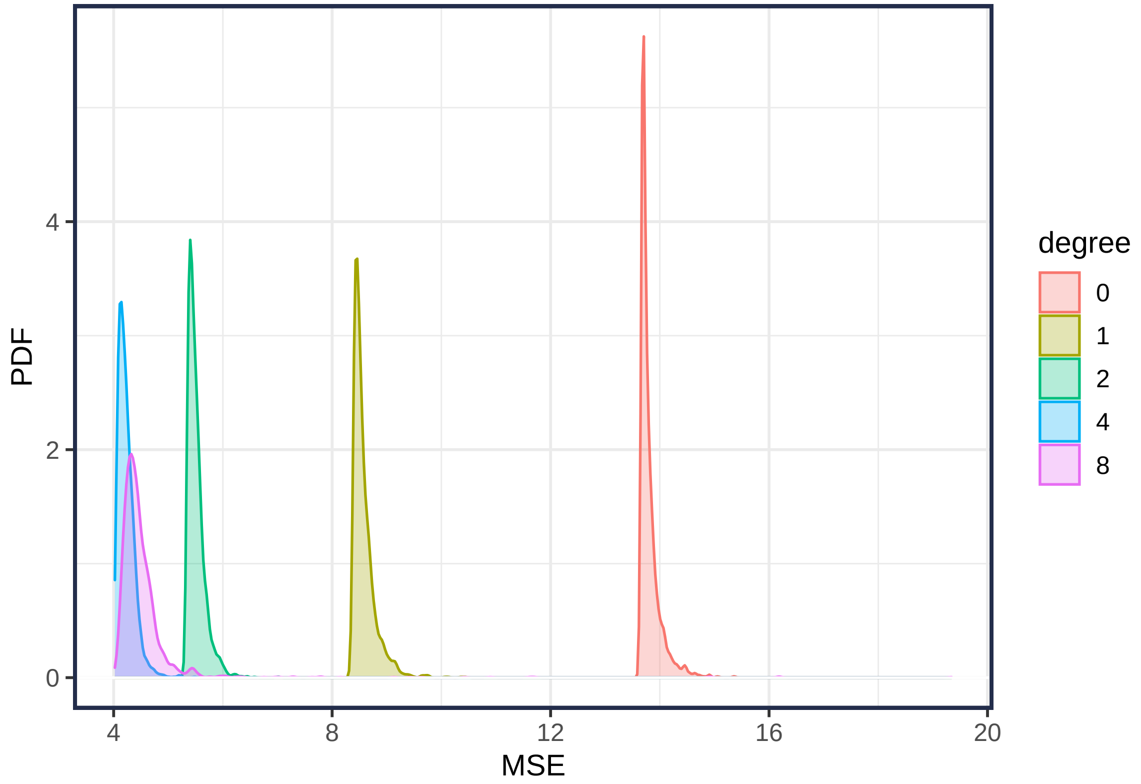
deg	bias.sq	var	mse
0	9.68	0.13	13.81
1	4.39	0.19	8.58
2	1.33	0.18	5.51
4	0.01	0.22	4.23
8	0.00	0.53	4.53

4.6 What does it all mean

The main point is that we desire to find a predictive model that has just the right *complexity*. This will depend on both the true complexity of the data as well as the sample size. A model that is too complex will have low bias, but potentially high variance. A model that is not complex enough will have high bias, but lower variance.

In a real setting, we will only observe one training data (a single curve) and will have to decide the optimal complexity. The following plot shows the estimated distribution of MSE values.





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
2	1
4	1941
8	58

- While the degree=4 model does best, degree=8 is sometimes best.
- If we examined more models, we would find that it wouldn't be too unusual to find degrees between 3 and 9 coming out as the best model.
- **Conclusion 1**: In our toy example, a polynomial with degree=4 is the best model, in principal. However, for some data (i.e., some simulations) the models with degree=8 and degree=2 performed 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 actually 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 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 complexity.
  - 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 not 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.