

1 **Analysis of Machine Learning Regression Estimators for Richard's Equation**
2 **Saturation Curves**

3
4 MELISSA BUTLER, University of Wyoming, USA
5

6
7 CCS Concepts: • Computer systems organization → Embedded systems; Redundancy; Robotics; • Networks → Network
8 reliability.

9 Additional Key Words and Phrases: differential equations, neural networks, initial value problem, boundary value problem, ODE
10

11 **ACM Reference Format:**

12 Melissa Butler. 2023. Analysis of Machine Learning Regression Estimators for Richard's Equation Saturation Curves. 1, 1 (May 2023),
13 20 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>
14

15 **1 PROBLEM STATEMENT**
16

17 Richard's Equation models fluid flow through semi-saturated porous media. It is highly non-linear partial differential
18 equation governing the capillary pressure and soil saturation. The non-linearity of Richard's Equation results in a
19 saturation curve with extreme slopes, making numerical methods involving derivatives unstable. We would like to see
20 if Machine Learning regression algorithms can capture the extreme curvature of the saturation, given a finite sampling
21 of numerically constructed data points, describing the one dimensional saturation curve.
22

23
24 **2 SIGNIFICANCE**
25

26 The study of fluid flow through partially saturated porous media is critical to agriculture, construction, waste disposal,
27 and other significant fields and is an extremely complex process, described by Richards equation. Richards equation is
28 of great interest due to the lack of closed form solutions and the difficulties in numerical approximations. We hope that
29 given a limited set of data points, that could be obtained in the field, an accurate approximation of the saturation curve
30 can be predicted.
31

32
33 **3 BACKGROUND**
34

35 Richard's Equation is represented by

36
37
$$\begin{cases} \partial_t \theta(u) - \partial_z (\kappa(u) \partial_z(u - z)) = 0, & \text{in } (0, L) \times (0, T) \\ u(z, 0) = u_0(z), & z \in (0, L) \\ \kappa(u) \partial_z(u - z) \Big|_{(0,t)} = g_0(t), & u(L, t) = 0. \end{cases} \quad (1)$$

38
39
40

41 The highly nonlinearity nature is seen in the dependence on the pressure head, u , by the hydraulic conductivity, κ , and
42 saturation, θ . This dependence produces rapid changes in the capillary head around the infiltration front, generating

44 Author's address: Melissa Butler, University of Wyoming, Laramie, WY, USA.

45 Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not
46 made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components
47 of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to
48 redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

49 © 2023 Association for Computing Machinery.

50 Manuscript submitted to ACM

51
52 Manuscript submitted to ACM

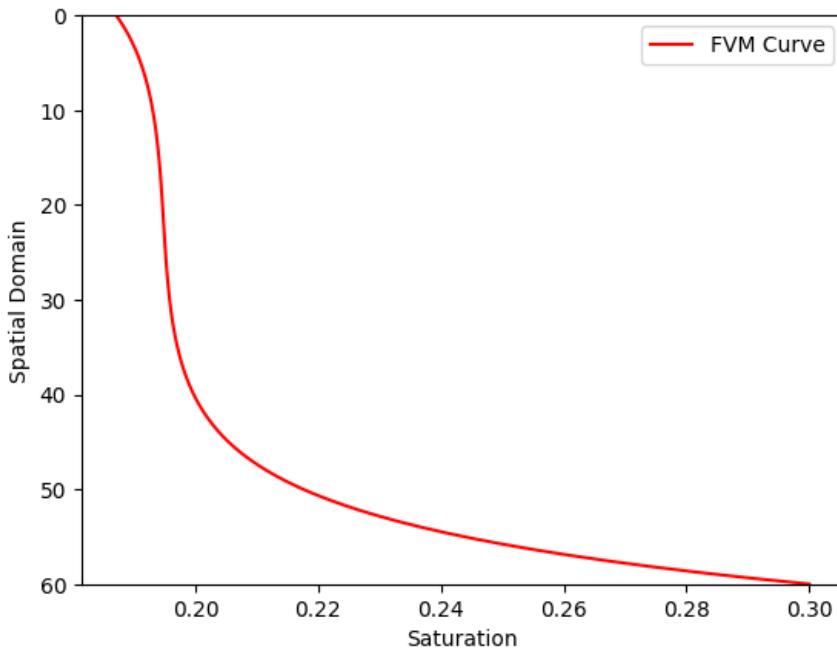


Fig. 1. FVM Curve

possibly non-differentiable zones. This, in turn, creates instability in many numerical approximation techniques. We will explore the effect of the non-differentiable zones of the saturation on machine learning regression estimators. By standard convention, we plot the spatial domain on the vertical axis in reverse. This visualizes the domain with the surface, $z = 0$, at the top and the water table, $z = 60$, at the bottom; refer to Figure 1.

4 DATASET DESCRIPTION

The Finite Volume Method (FVM) is a common numerical approximation tool for PDE's and was used to generate our data points. Our inputs consist of a discretized spatial domain, $z \in [0, 60]$, with $M = 100$ equispaced nodes,

$$\{z_i\}_{i=1}^M, 0 = z_1 < z_2 < \dots < z_{M-1} < z_M = 60.$$

The outputs are the approximate values of saturation at each node,

$$\{\theta(u_h(z_i))\}_{i=1}^M,$$

generated by the FVM. Note that Richard's Equation is also time dependent. Our FVM employed a full-discretization (spatial and temporal) and time marching was used. The extreme curvature occurs in early timesteps, with later times reaching a smoother steady state, so early timesteps were used. We can see the available data points in Figure 2.

5 METHODOLOGY

To analyze the capabilities of machine learning regression algorithms on the saturation curve, the data was split into various size training and testing sets. The training data was used to fit Polynomial Regression, KNN Regression, Decision

Manuscript submitted to ACM

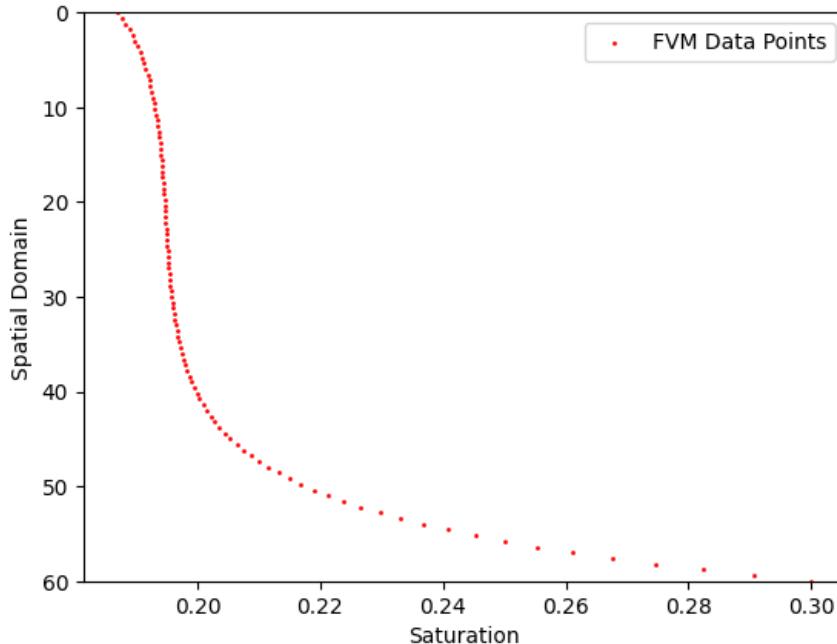


Fig. 2. FVM Generated Data Points

Tree Regression, and Random Forest Regression models. Various hyperparameters where also tested, using a grid search. The parameters used are given in Figure 3. To analyze the accuracy, the Mean Squared Error and Maximum Error were calculated. We first started with two random training splits with 70% training and 30% testing, to analyze a dense set of data, shown in Figure 8 and Figure 9. Next, to simulate potential in-field measurements of saturation, we split the data using equispaced grid points with 6 training points and 11 training points, shown in Figure 19 and Figure 18.

Estimator	Hyperparameter	Values
Polynomial Regression	Degree	3,4,5,6
K-Nearest Neighbors	k-neighbors	2,3,4,5
Decision Tree	Max Depth	3,4,5,6
Random Forest	Estimators	5,100,1000,2000

Fig. 3. Hyperparameter Values

6 RESULTS

6.1 Random Split 1 - 70% Training Data

In this random split, there were no training data points close to the water table, which resulted in larger errors for the methods normally used for classification.

	Estimator	Hyperparameter Value	Training Error	Testing Error
157 158 159 160 161	PR	3	0.000016	0.000021
		4	0.000003	0.000004
		5	0.000000	0.000000
		6	0.000000	0.000000
162 163 164 165 166	KNN	2	0.000005	0.000009
		3	0.000010	0.000017
		4	0.000019	0.000036
		5	0.000027	0.000062
167 168 169 170 171	DT	3	0.000005	0.000031
		4	0.000001	0.000020
		5	0.000000	0.000017
		6	0.000000	0.000017
172 173 174 175 176	RF	5	0.000004	0.000017
		100	0.000004	0.000011
		1000	0.000003	0.000008
		5000	0.000003	0.000008

Fig. 4. Errors for Random Split 1

6.2 Random Split 2 - 70% Training Data

In this random split, we happened to capture data points for the entire domain. This cannot be guaranteed, but also doesn't reflect real world situations, since collecting the saturation level for 70 data points is infeasible. However, it does illustrate the capabilities of the machine learning algorithms to capture high curvature.

	Estimator	Hyperparameter Value	Training Error	Testing Error
183 184 185 186 187	PR	3	0.000019	0.000011
		4	0.000004	0.000003
		5	0.000000	0.000000
		6	0.000000	0.000000
188 189 190 191	KNN	2	0.000002	0.000001
		3	0.000002	0.000001
		4	0.000004	0.000001
		5	0.000007	0.000001
192 193 194 195	DT	3	0.000007	0.000017
		4	0.000001	0.000006
		5	0.000000	0.000004
		6	0.000000	0.000003
196 197 198	RF	5	0.000002	0.000002
		100	0.000001	0.000002
		1000	0.000001	0.000002
		5000	0.000001	0.000002

Fig. 5. Errors for Random Split 2

6.3 Equispaced - 11 Training Points

For this split we used 11 equally spaced training inputs,

$$z \in \{0, 6, 12, 18, 24, 30, 36, 42, 48, 54, 60\}$$

and their corresponding saturation values. The rest of the data was used for testing. This is under the assumption, that a core sample can be taken and the saturation can be measured at various depths. This could then be extrapolated to estimate nearby water content.

Estimator	Hyperparameter Value	Training Error	Testing Error
PR	3	0.000028	0.000023
	4	0.000005	0.000005
	5	0.000000	0.000000
	6	0.000000	0.000000
KNN	2	0.000109	0.000034
	3	0.000251	0.000070
	4	0.000358	0.000126
	5	0.000495	0.000176
DT	3	0.000011	0.000054
	4	0.000002	0.000045
	5	0.000000	0.000044
	6	0.000000	0.000044
RF	5	0.000293	0.000064
	100	0.000049	0.000011
	1000	0.000050	0.000011
	5000	0.000054	0.000011

Fig. 6. Errors for 11 Equispaced

6.4 Equispaced - 6 Training Points

For this split we used 6 equally spaced inputs,

$$z \in \{0, 12, 24, 36, 48, 60\},$$

and their corresponding saturation values. The rest of the data was used for testing. This was to observe the accuracy of the model for even sparser data. Reducing the data further resulting in highly inaccurate models.

Estimator	Hyperparameter Value	Training Error	Testing Error
PR	3	0.000021	0.000037
	4	0.000001	0.000007
	5	0.000000	0.000001
	6	0.000000	0.000000
KNN	2	0.000338	0.000150
	3	0.000785	0.000209
	4	0.000955	0.000280
	5	0.001190	0.000333
DT	3	0.000001	0.000182
	4	0.000000	0.000181
	5	0.000000	0.000180
	6	0.000000	0.000180
RF	5	0.000057	0.000073
	100	0.000137	0.000057
	1000	0.000155	0.000053
	5000	0.000158	0.000053

Fig. 7. Errors for 6 Equispaced

261 7 CONCLUSION

262
263 A visual analysis of the training and testing errors for each data set is given Figure 28, Figure 29, Figure 30, and Figure 31.
264 For random sampled data, we can achieve a testing error of less than 1e-7 for polynomial regression of degree 5. This
265 occurs even in Random Split 1, where we do not capture points near the water table. Other methods were not able
266 to reach this level of accuracy for the same data set. However, when we include points along the whole domain, all
267 methods had a testing error less than 1e-5, for some hyperparameter. Polynomial regression was the best performing
268 model, with degree of 5 or greater. When training the models on equispaced data points, we again see polynomial
269 performed the best. Even with only 6 data points, a trained polynomial of degree 5 had a testing error less than 1e-7.
270 The classification methods did not perform well for the sparse data, with K-nearest neighbors performing the worst,
271 by far. The maximum error for each model agrees with the findings of the average error. A polynomial of degree 5 or
272 greater has the smallest maximum error and K-nearest neighbors has the highest.
273

274
275
276 These results are consistent with what we would expect from training these models. It was surprising how well
277 the classifier methods did when given enough data. Perhaps this could be expanded into a classification problem to
278 predict if the soil is within a target range that needs additional moisture. If the specific value of saturation is needed,
279 polynomial regression is an easily trainable and highly accurate model. The data points used in this report represent a
280 small sample of possible conditions involved with Richard's Equation. There is a great deal of uncertainty involved with
281 the constitutive relation between the capillary head and pressure. There is also uncertainty regarding the boundary
282 value, associated with the amount of precipitation and evaporation on the surface. These would be subsequent problems
283 to explore using machine learning. There is also the opportunity to use Physics Informed Neural Networks to estimate
284 values for Richard's Equation, itself.
285

286
287 In conclusion, polynomial regression method is able to capture high curvature in data with a small degree, as well as
288 incomplete and sparse data. Classification methods are able to predict the values with varying accuracy, depending on
289 the given data. If some generalizations can be made about the components of Richard's Equation, machine learning
290 models, trained on sparse field test data, could potentially be used to predict saturation levels. Either way, comparing
291 machine learning models is always a good time.
292

293 8 PRESENTATION

294 Video Presentation

295 9 CODE

296 [https://github.com/butlerm0405/ml_{fluid}dynamics.git](https://github.com/butlerm0405/ml_fluid_dynamics.git)

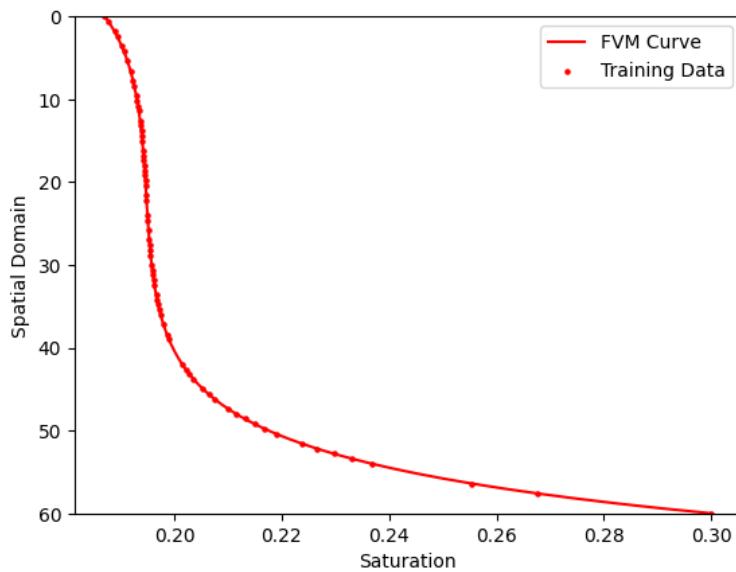
10 FIGURES

Fig. 8. Random 1 Data

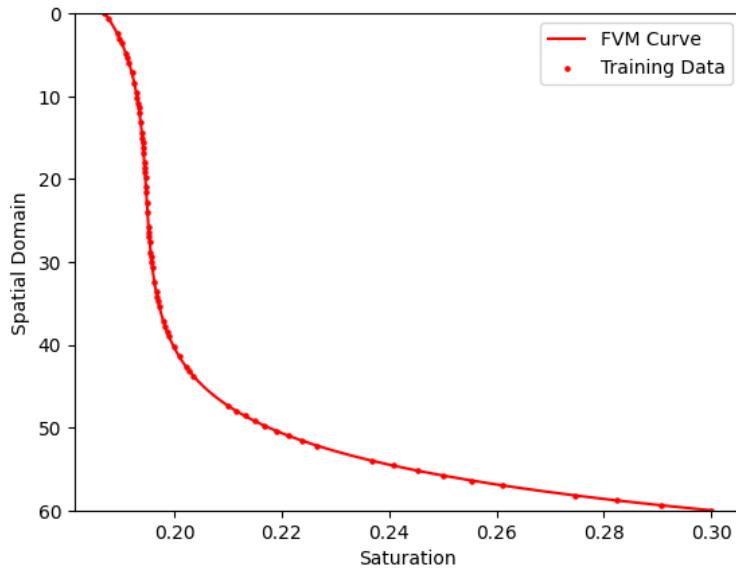


Fig. 9. Random 2 Data

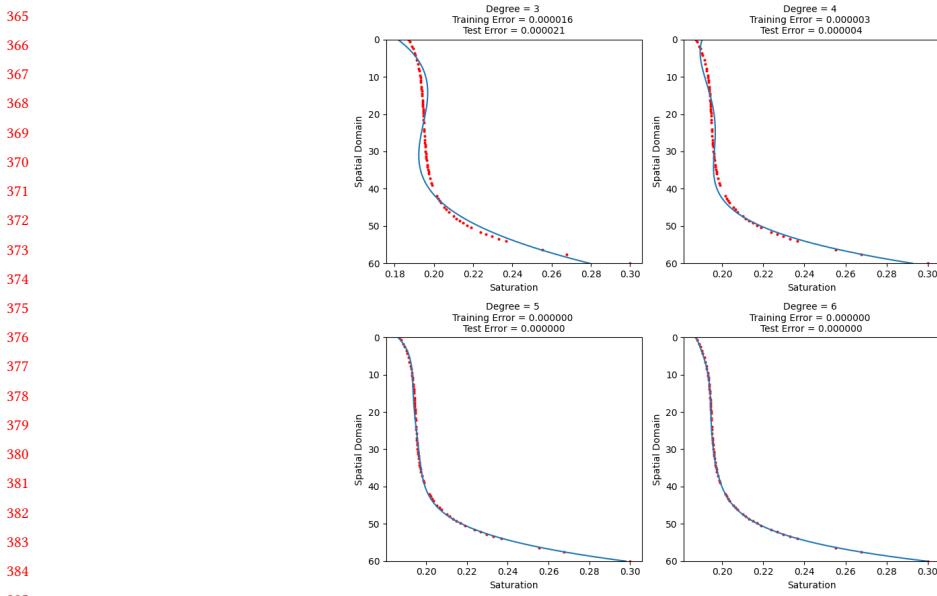


Fig. 10. Random 1 Polynomial Regression

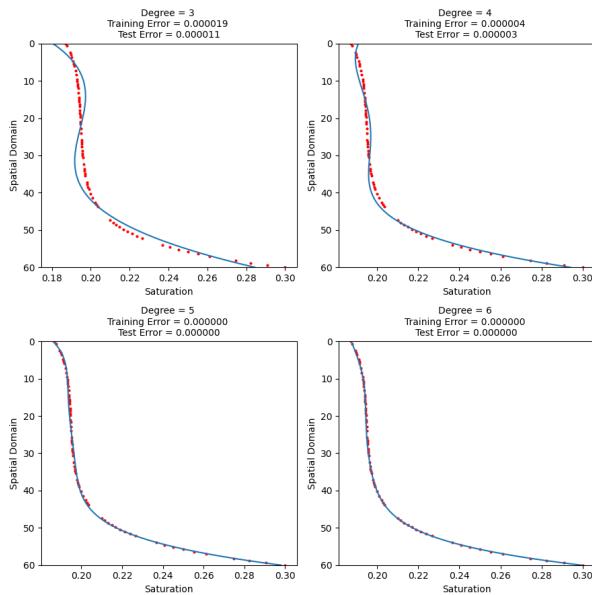


Fig. 11. Random 2 Polynomial Regression

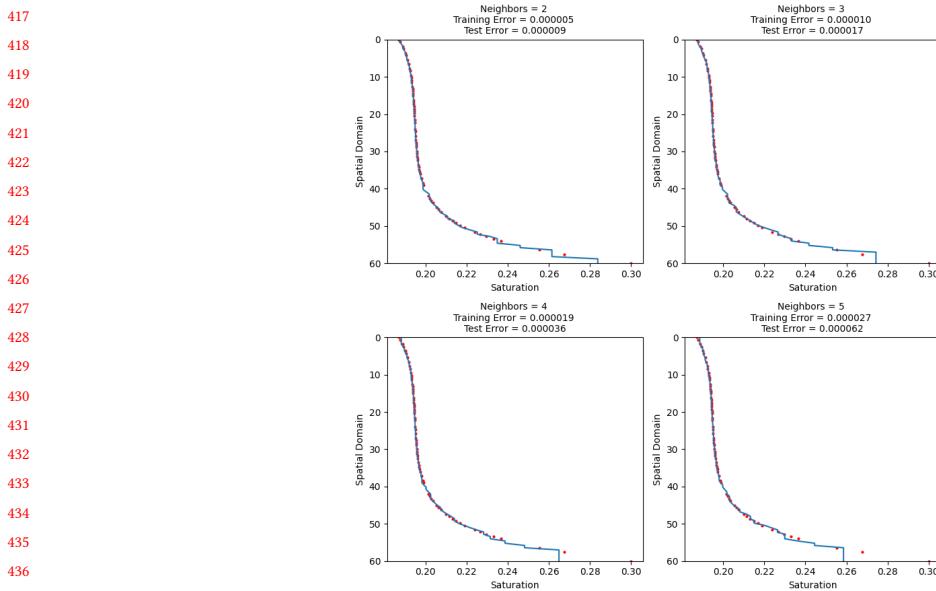


Fig. 12. Random 1 KNN

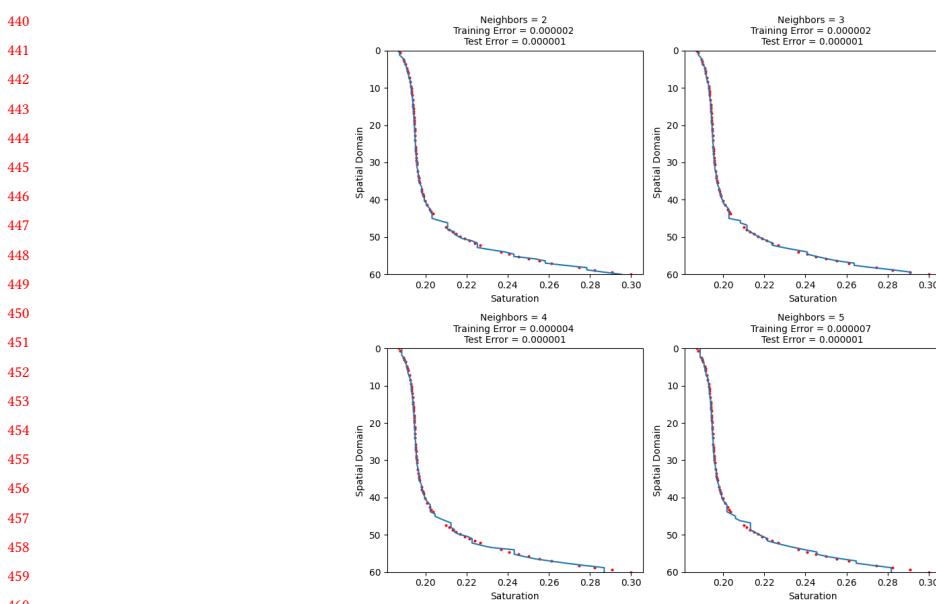


Fig. 13. Random 2 KNN

Max Depth = 3
Training Error = 0.000005
Test Error = 0.000031

Max Depth = 4
Training Error = 0.000001
Test Error = 0.000020

Max Depth = 5
Training Error = 0.000000
Test Error = 0.000017

Max Depth = 6
Training Error = 0.000000
Test Error = 0.000017

Fig. 14. Random 1 Decision Tree

The figure consists of four subplots arranged in a 2x2 grid, each showing the relationship between Saturation (x-axis, ranging from 0.18 to 0.30) and Spatial Domain (y-axis, ranging from 0 to 60). Each subplot includes two data series: Training Error (red dotted line) and Test Error (blue solid line).

- Top Left (Max Depth = 3):** Training Error = 0.000007, Test Error = 0.000017.
- Top Right (Max Depth = 4):** Training Error = 0.000001, Test Error = 0.000006.
- Bottom Left (Max Depth = 5):** Training Error = 0.000000, Test Error = 0.000004.
- Bottom Right (Max Depth = 6):** Training Error = 0.000000, Test Error = 0.000003.

In all plots, the error generally decreases as the spatial domain increases. The transition from high error to low error becomes sharper as the maximum depth increases. For Max Depth 3, the transition starts around Saturation 0.18 and ends at approximately 0.25. For Max Depth 4, it starts around 0.18 and ends at approximately 0.22. For Max Depth 5, it starts around 0.18 and ends at approximately 0.20. For Max Depth 6, it starts around 0.18 and ends at approximately 0.19.

Fig. 15. Random 2 Decision Tree

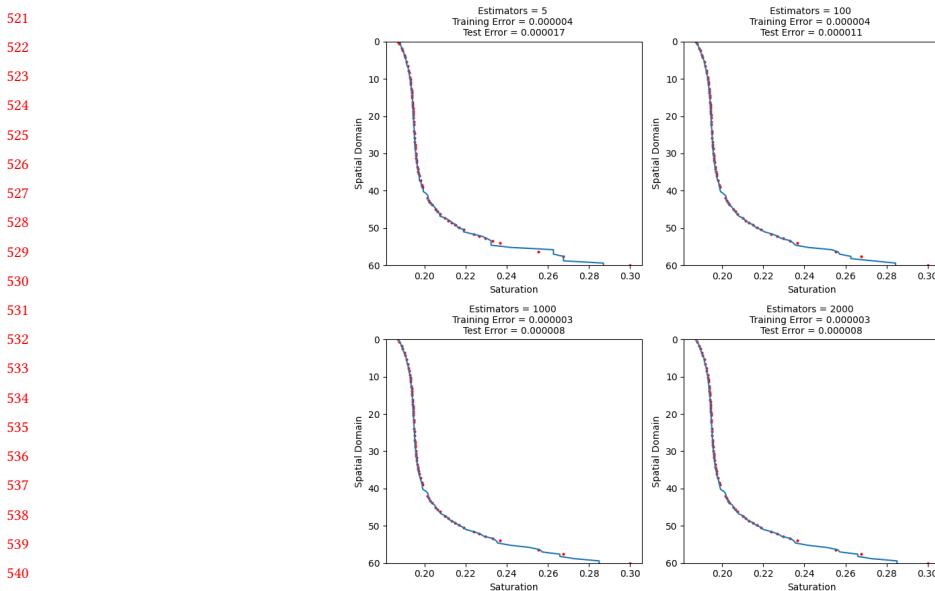


Fig. 16. Random 1 Random Forest

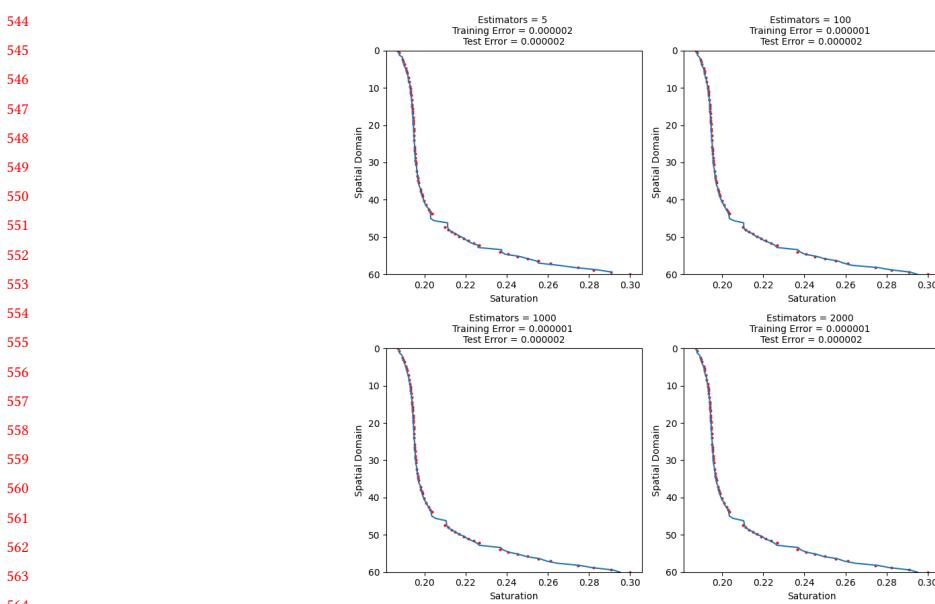


Fig. 17. Random 2 Random Forest

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

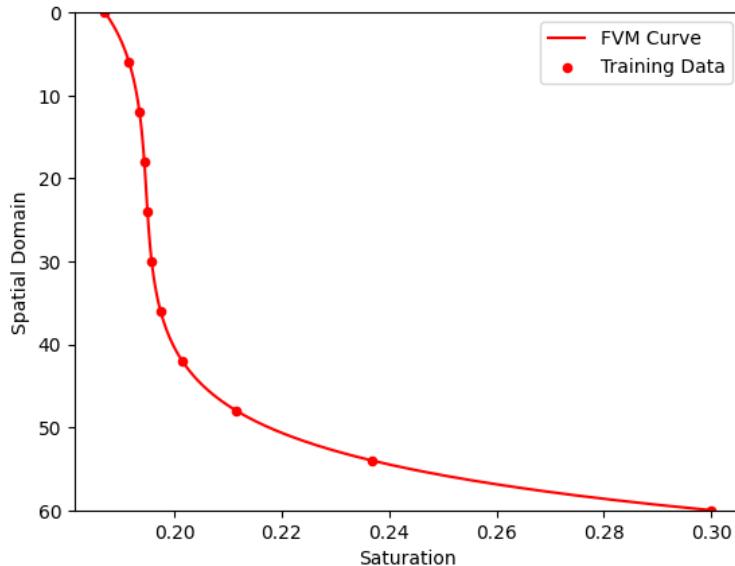


Fig. 18. 11 Equispaced Data

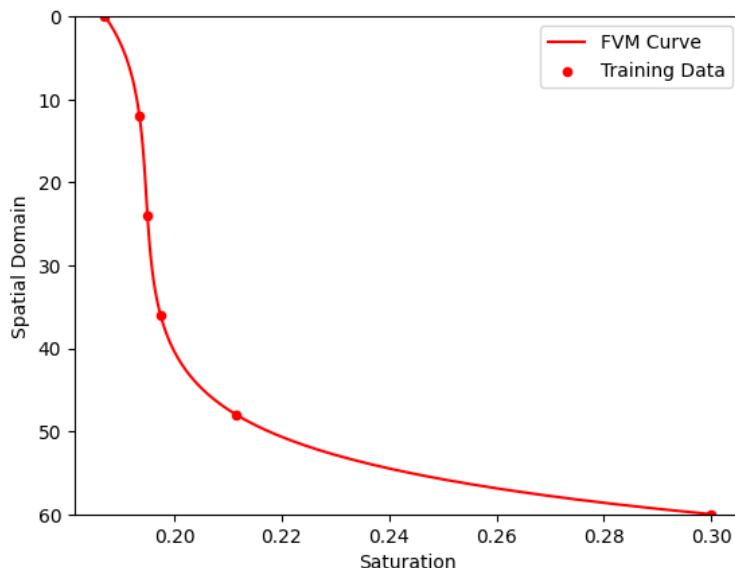


Fig. 19. 6 Equispaced Data

624

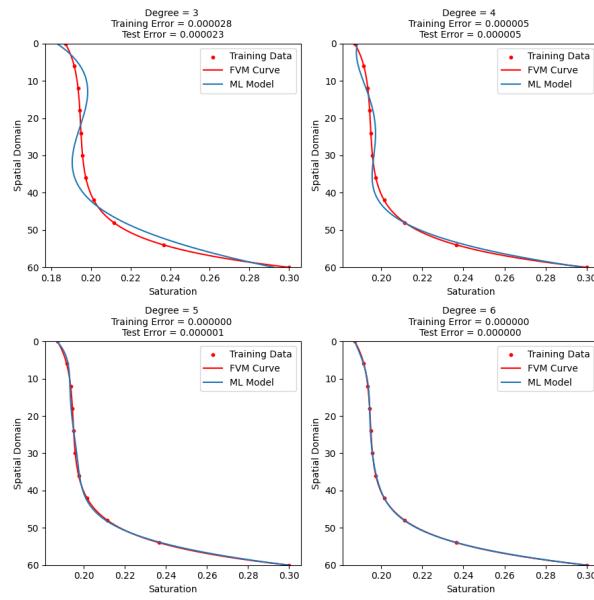


Fig. 20. 11 Equispaced Polynomial Regression

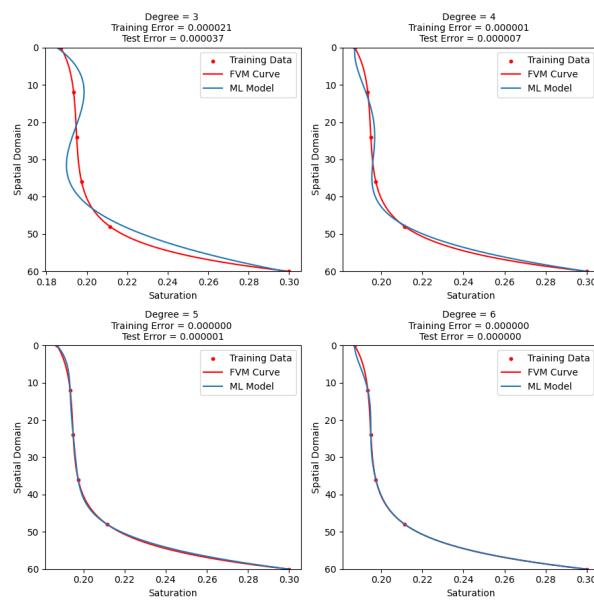


Fig. 21. 6 Equispaced Polynomial Regression

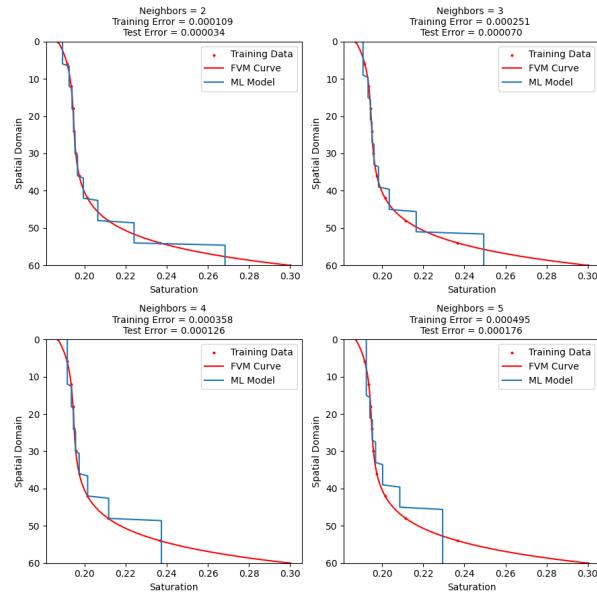


Fig. 22. 11 Equispaced KNN

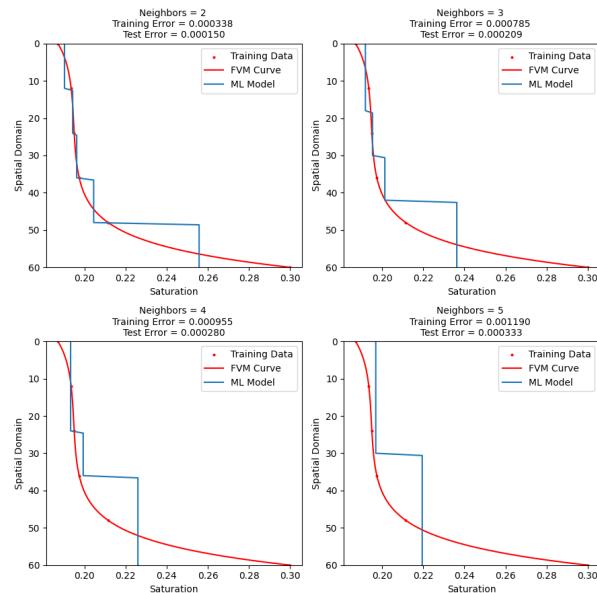


Fig. 23. 6 Equispaced KNN

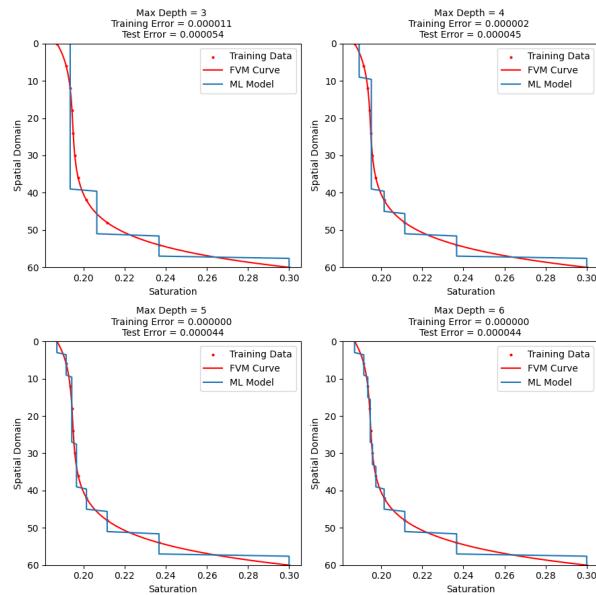


Fig. 24. 11 Equispaced Decision Tree

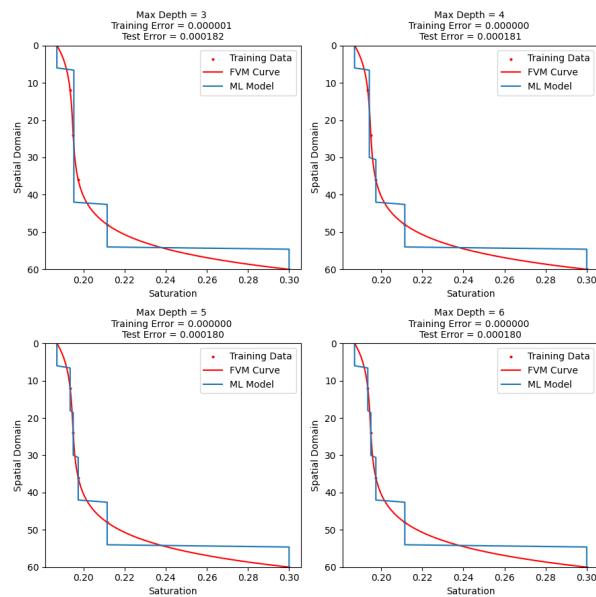


Fig. 25. 6 Equispaced Decision Tree

781 Estimators = 5
Training Error = 0.000293
Test Error = 0.000064

782 Estimators = 100
Training Error = 0.000049
Test Error = 0.000011

783

784

785

786

787

788

789

790

791 Estimators = 1000
Training Error = 0.000050
Test Error = 0.000011

792

793

794

795

796

797

798

799

800 Estimators = 2000
Training Error = 0.000054
Test Error = 0.000011

Fig. 26. 11 Equispaced Random Forest

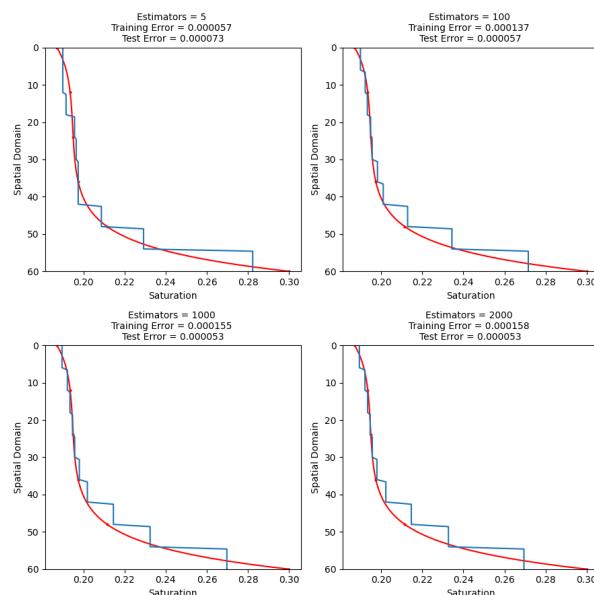


Fig. 27. 6 Equispaced Random Forest

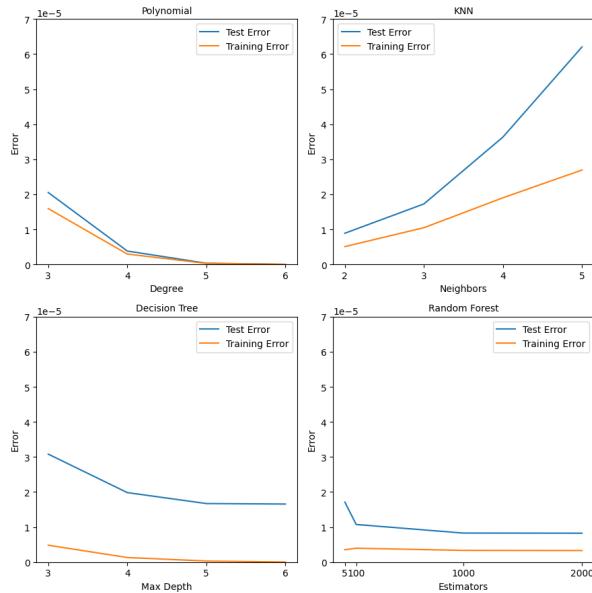


Fig. 28. Random 1 Testing and Training Error

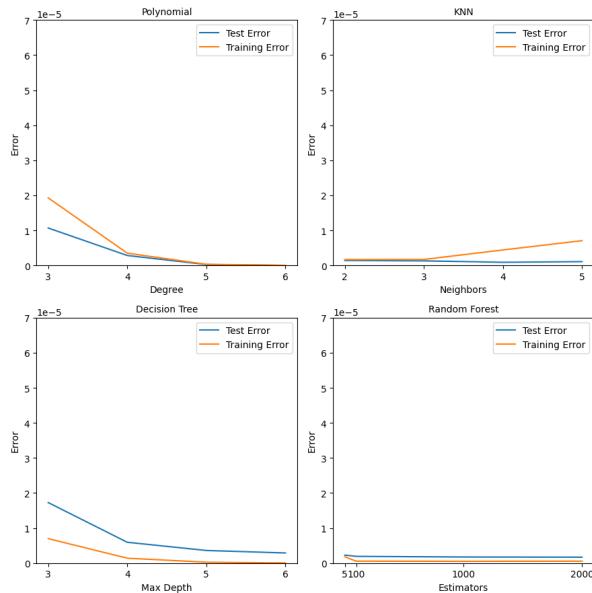


Fig. 29. Random 2 Testing and Training Error

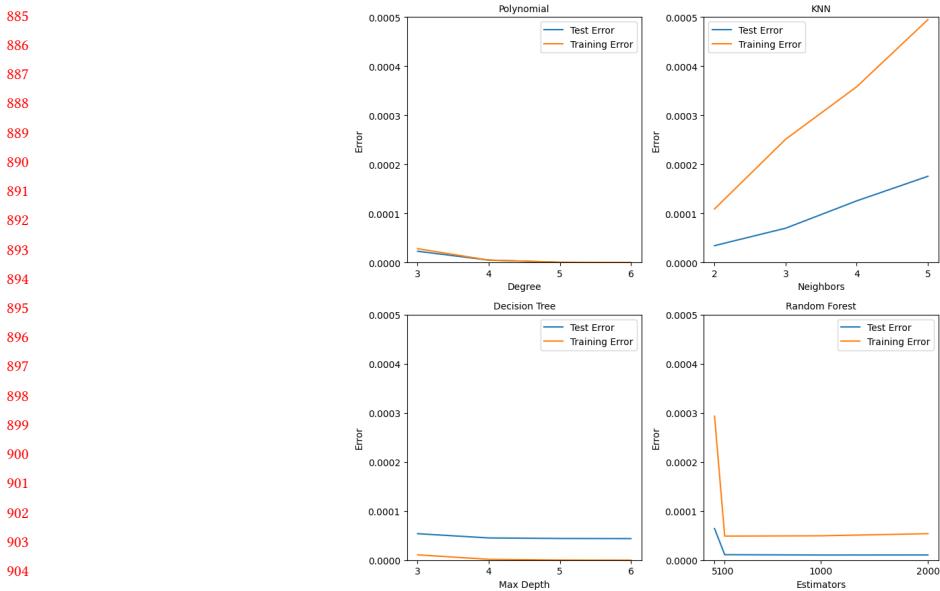


Fig. 30. 11 Equispaced Testing and Training Error

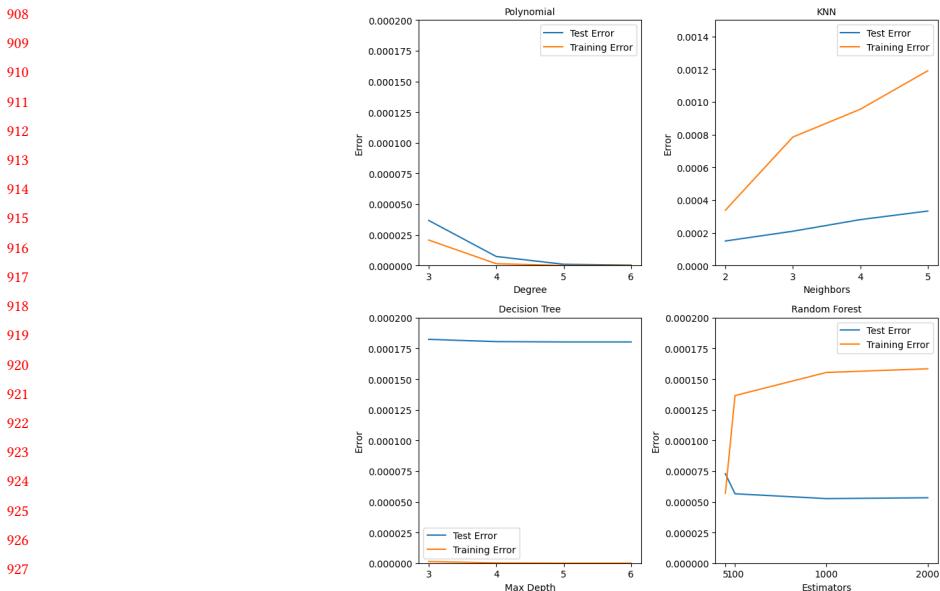


Fig. 31. 6 Equispaced Testing and Training Error

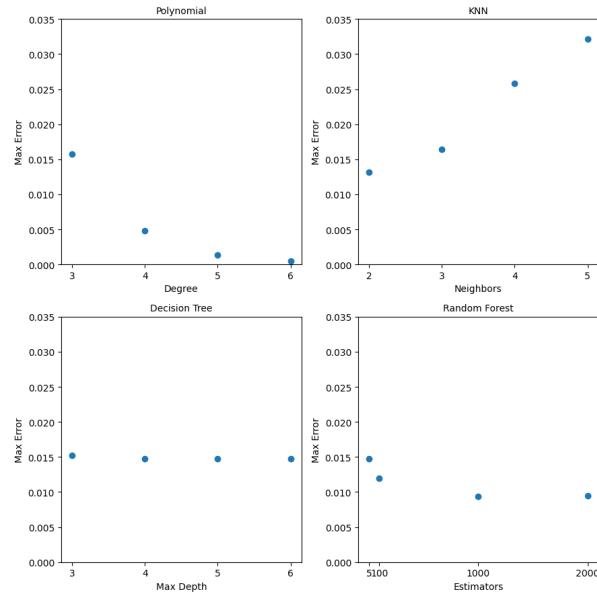


Fig. 32. Random 1 Maximum Error

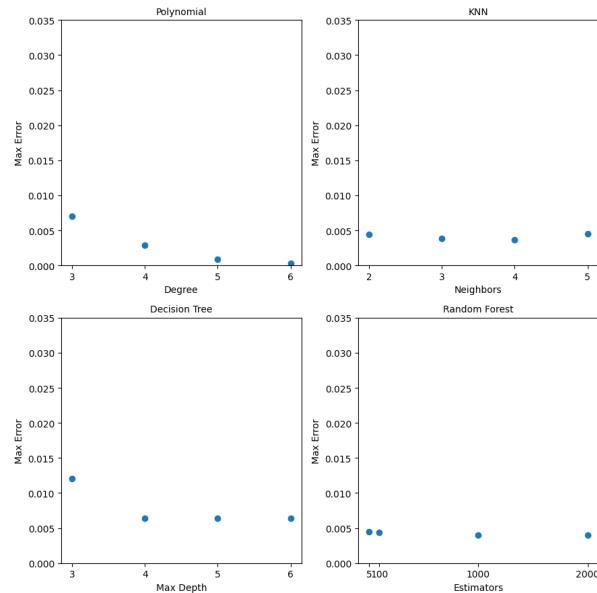


Fig. 33. Random 2 Maximum Error

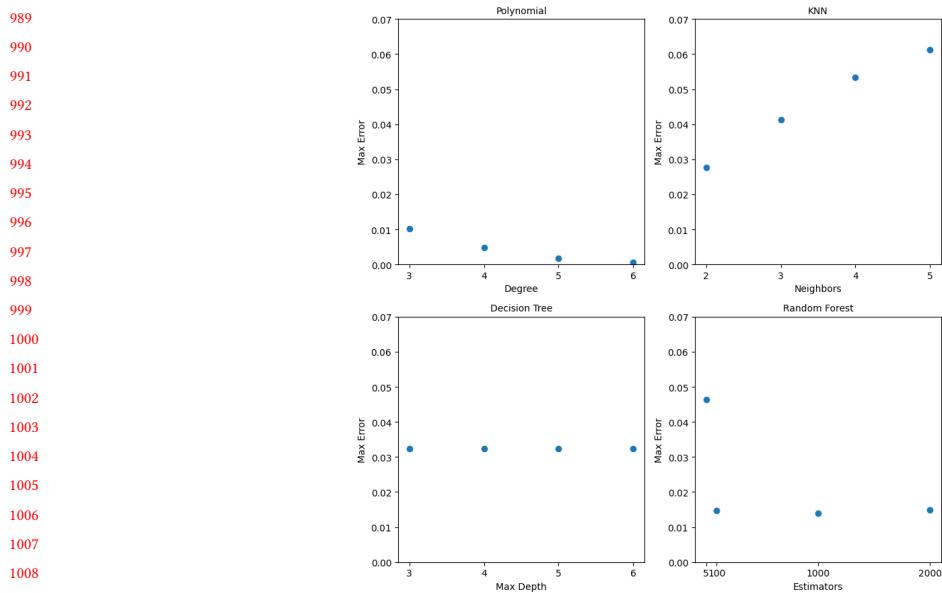


Fig. 34. 11 Equispaced Maximum Error

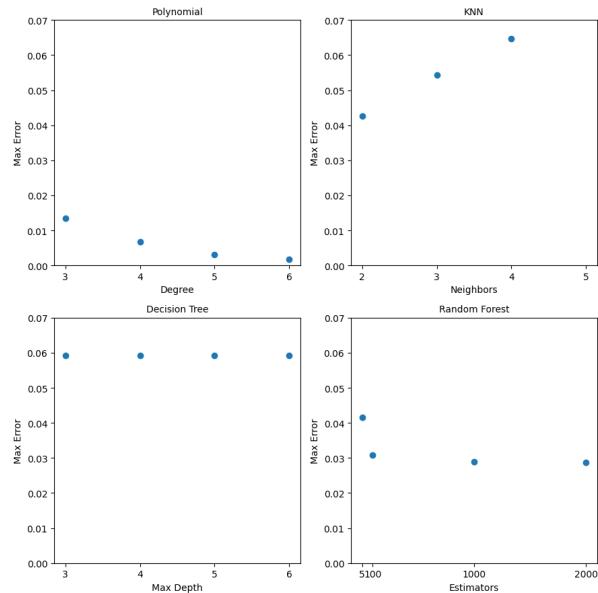


Fig. 35. 6 Equispaced Maximum Error