Data: Splits, Bias, Variance, ...

Denis Baskan
Zia Muhammad
Rashik Islam
Alexei Figueroa

Studiere Stadt der Zukunft!

BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences



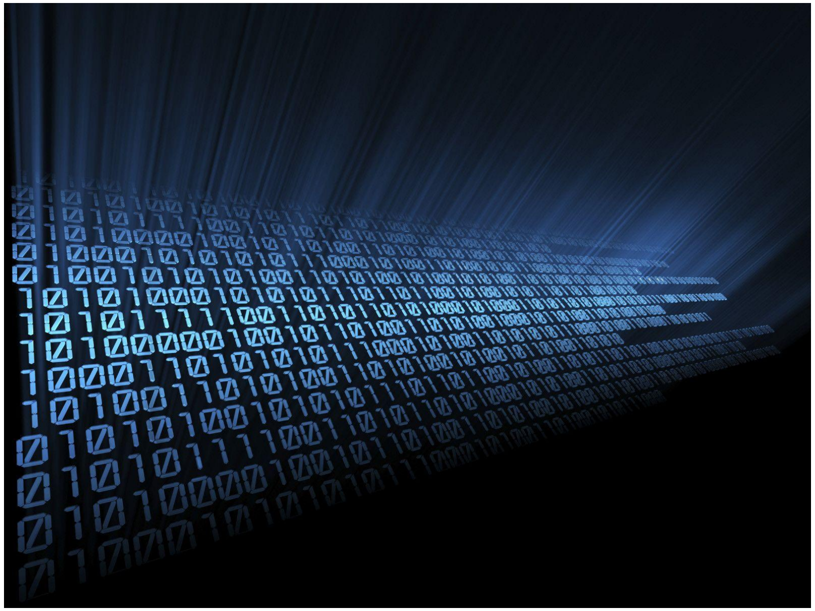



Content

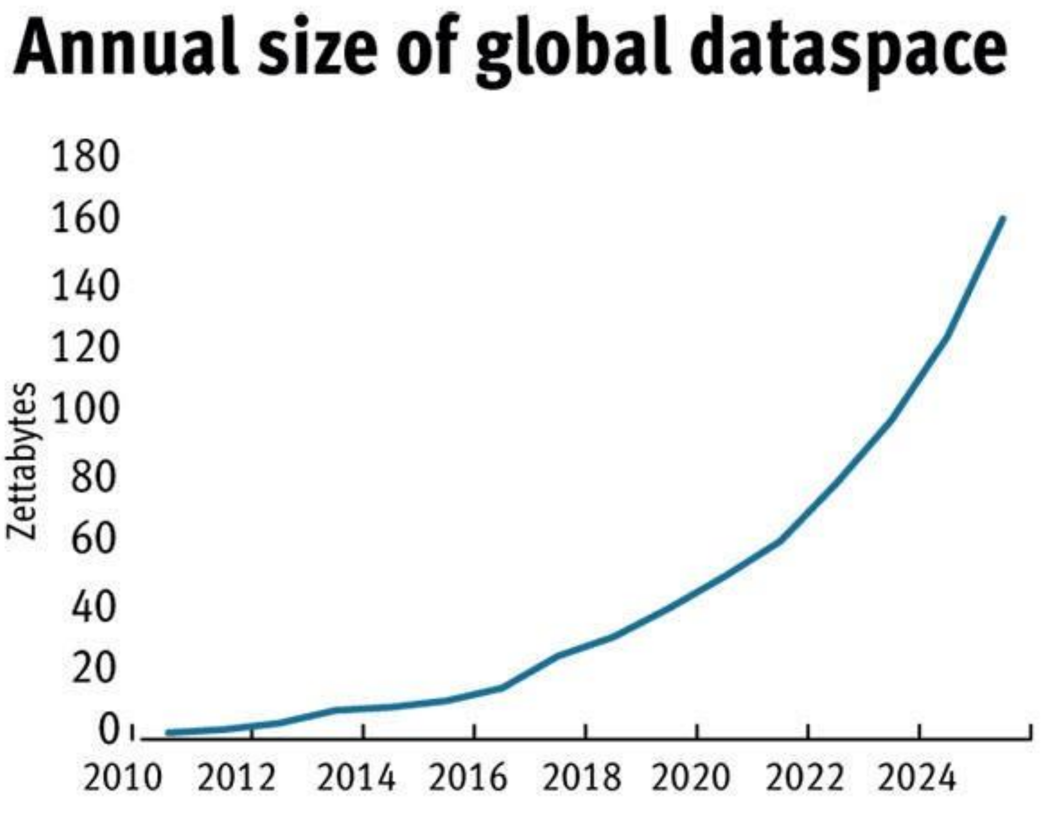
- Data
  - Amount
  - Splits
  - Distribution
- Errors
  - Irreducible Error
  - Bias and Variance Error
  - Underfitting and overfitting
  - Avoidable Bias
- BV- Tradeoff
  - In terms of MSE
  - Impact
  - Techniques to control
- BV - Ensemble Methods and NNs
  - From trees to forests
  - Neural networks

Data - Amount

- **ML & DL Methods are taking off now**

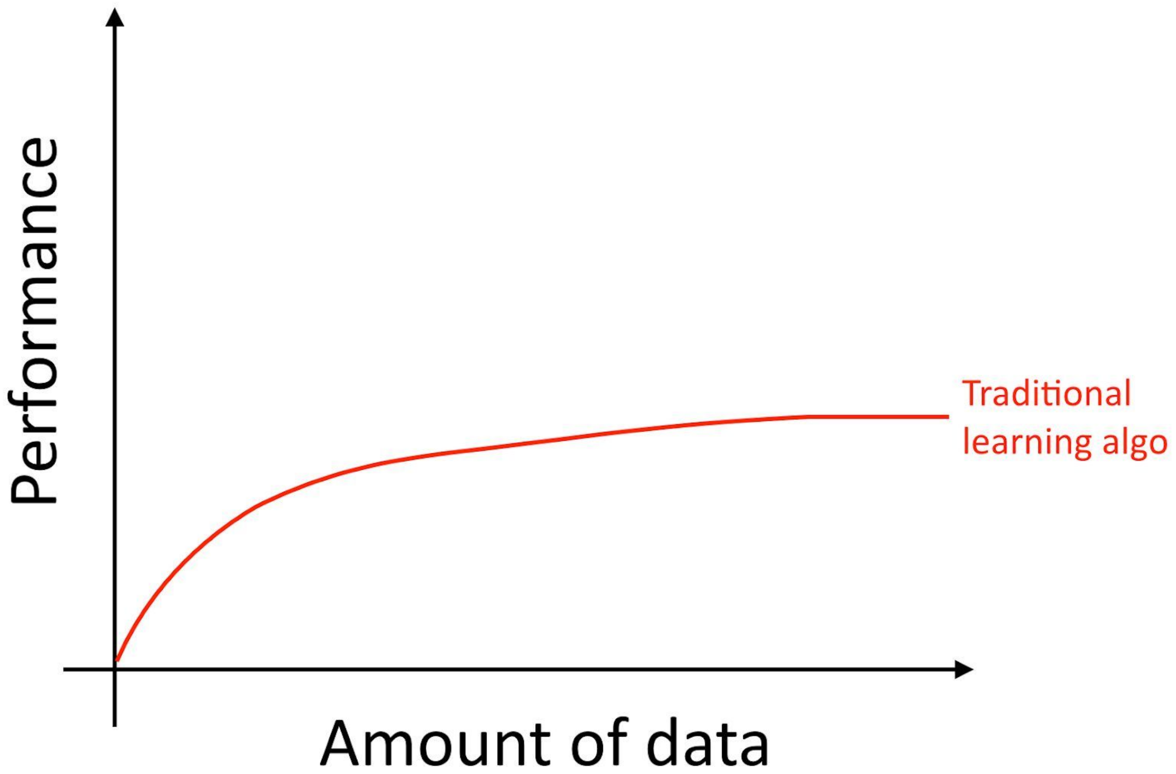


Data - Amount

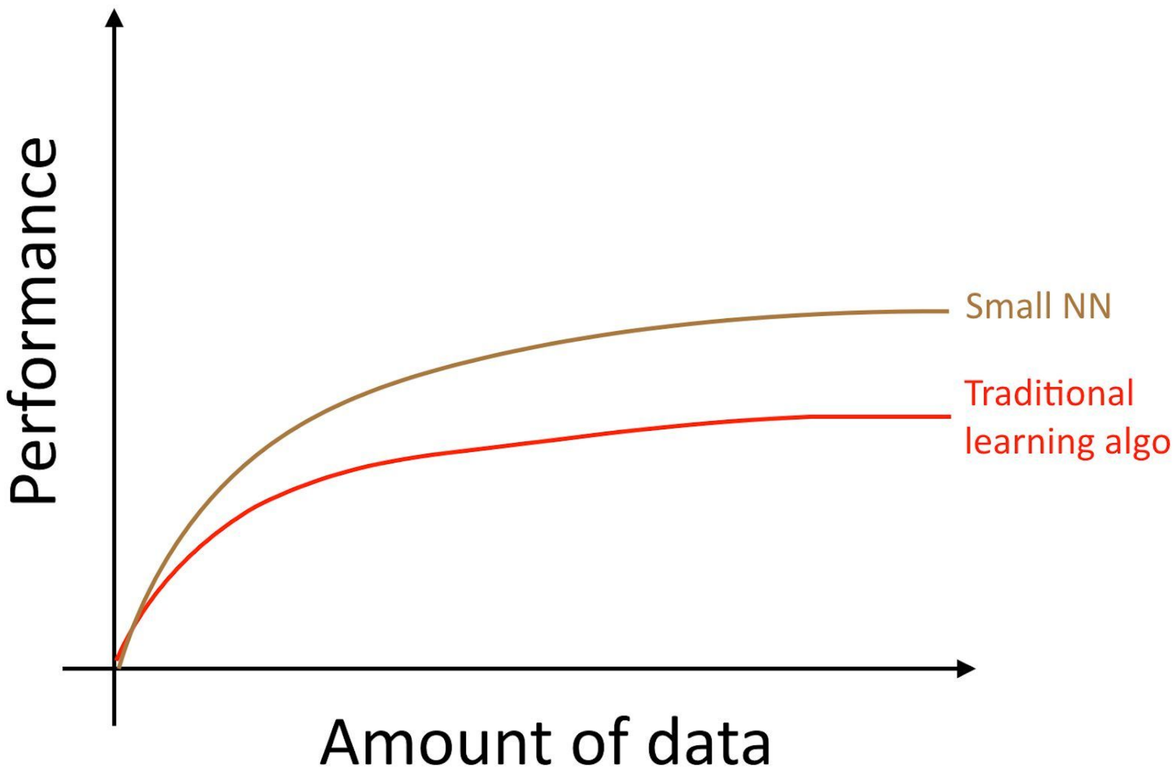


1 ZB = 1e9TB = 1,000,000,000TB

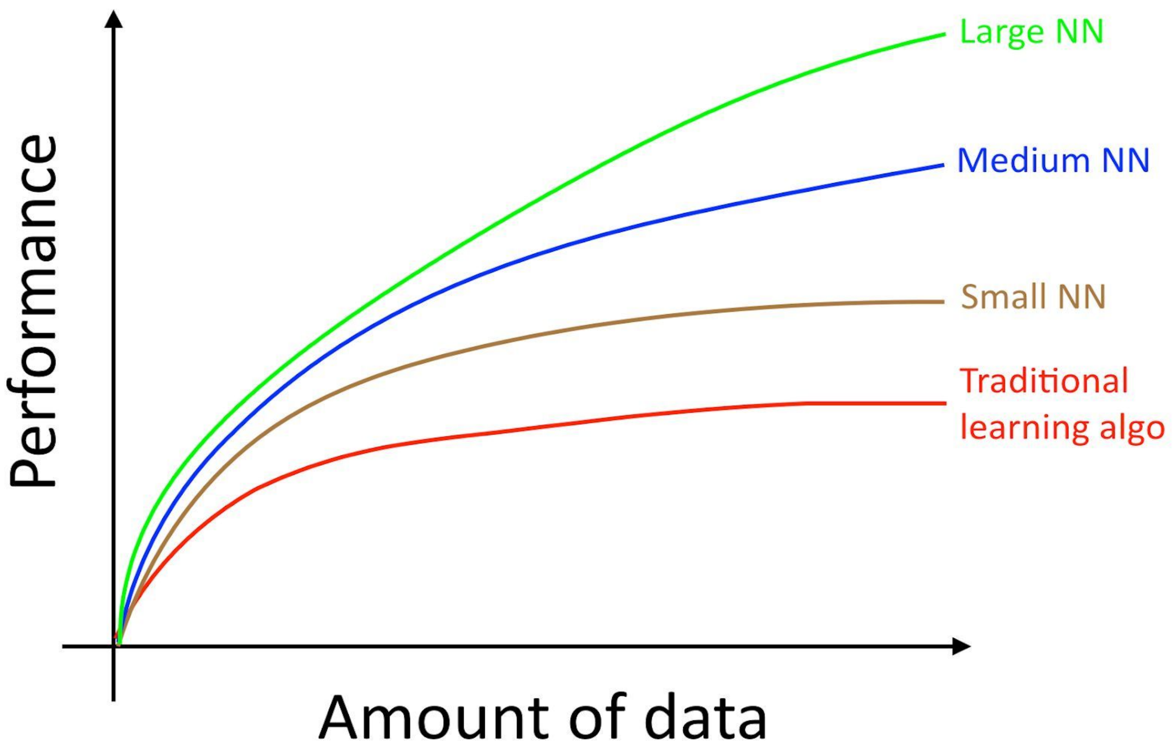
Data - Amount



Data - Amount

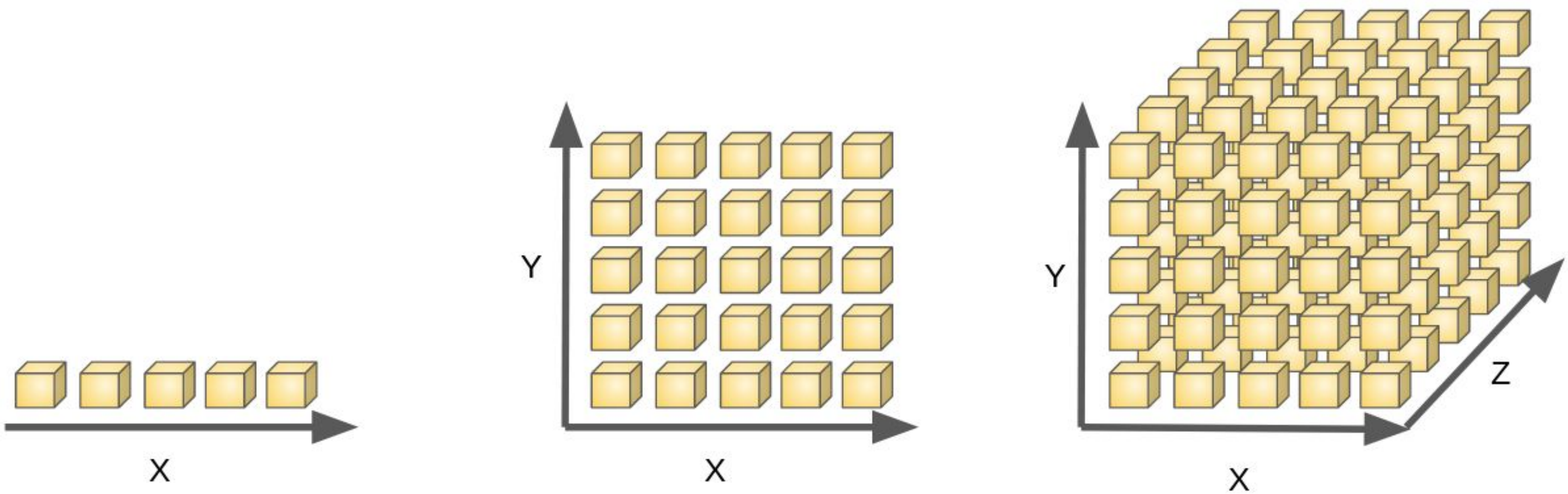

Performance
Small NN
Traditional learning algo
Amount of data

Data - Amount


Large NN
Medium NN
Small NN
Traditional learning algo
Performance
Amount of data

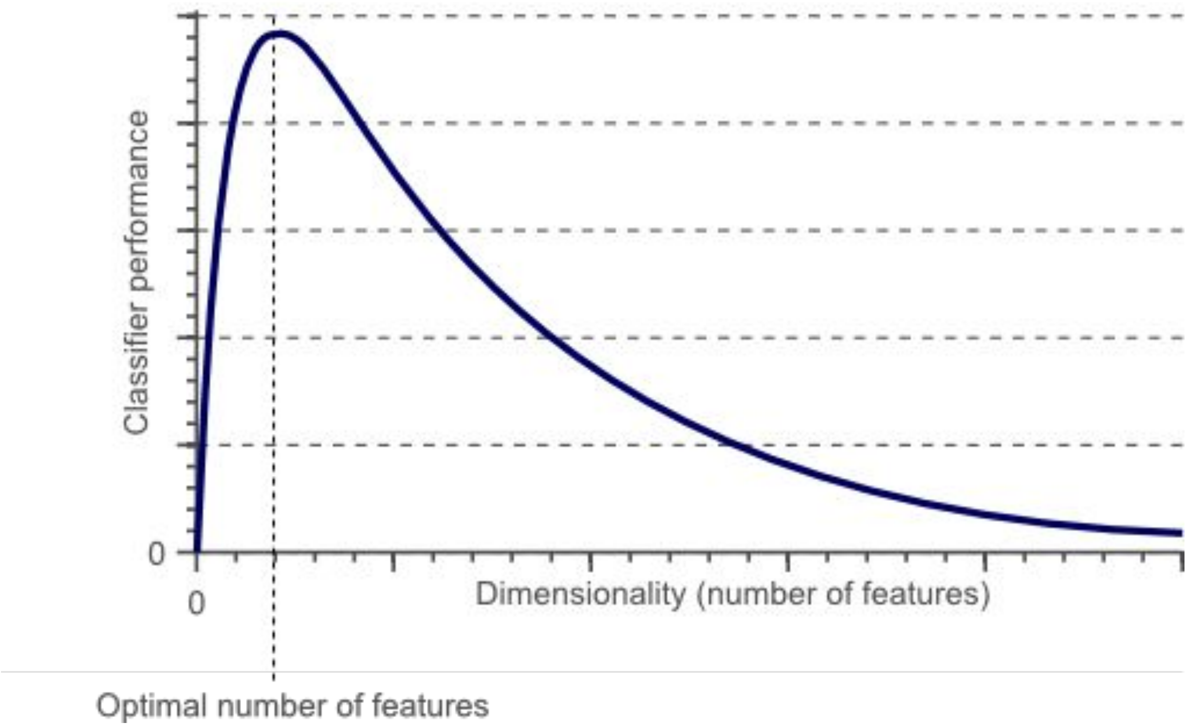
Data - Amount

- **Curse of Dimensionality**

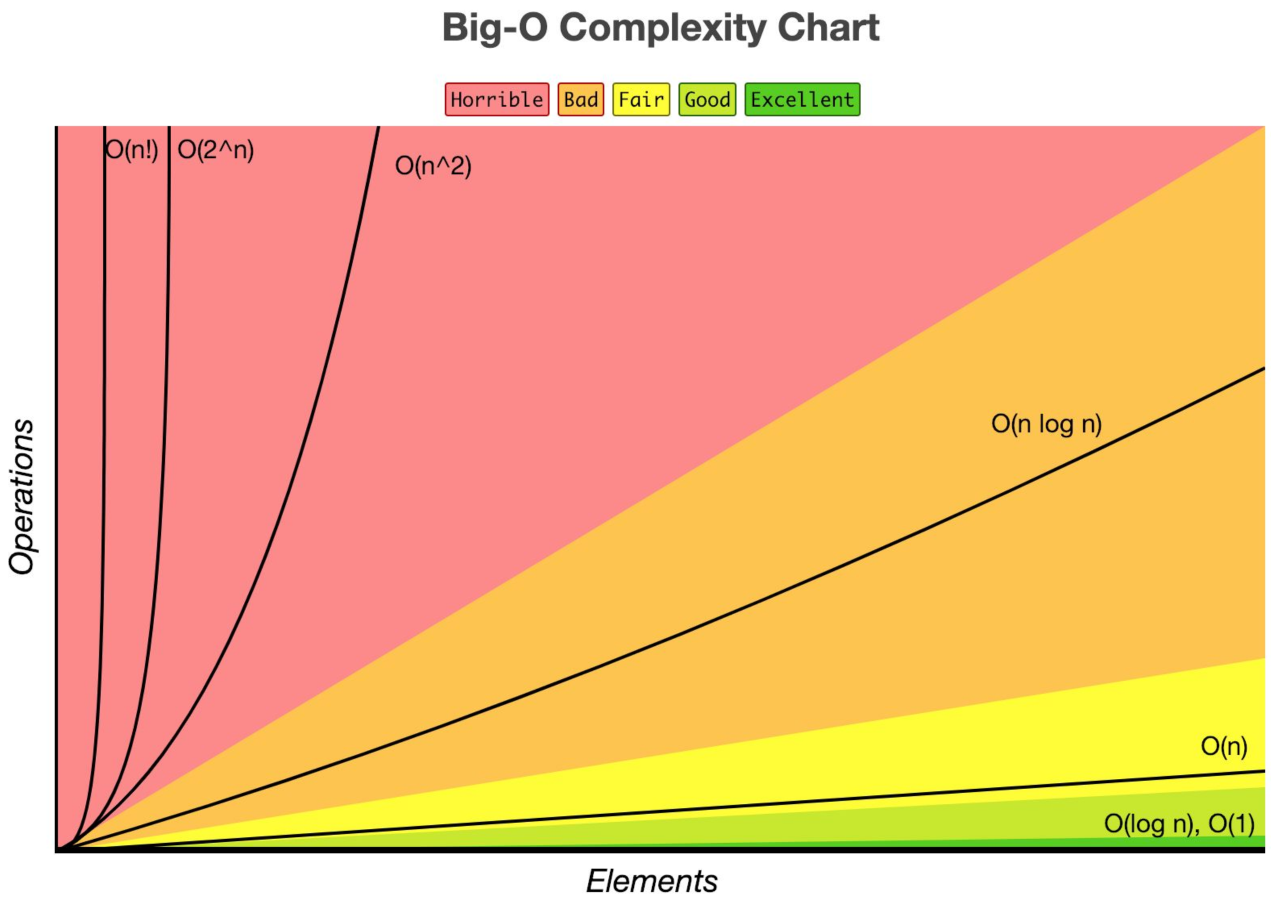


Data - Amount

- **Curse of Dimensionality**



Data - Amount


Big-O Complexity Chart
Horrible
Bad
Fair
Good
Excellent
O(n!)
O(2^n)
O(n^2)
O(n log n)
O(n)
O(log n), O(1)
Operations
Elements

Data - Amount

Algorithm	Classification/Regression	Training	Prediction
Decision Tree	C+R	$O(n^2p)$	$O(p)$
Random Forest	C+R	$O(n^2pn_{trees})$	$O(pn_{trees})$
Random Forest	R Breiman implementation	$O(n^2pn_{trees})$	$O(pn_{trees})$
Random Forest	C Breiman implementation	$O(n^2\sqrt{p}n_{trees})$	$O(pn_{trees})$
Extremly Random Trees	C+R	$O(npn_{trees})$	$O(npn_{trees})$
Gradient Boosting (n_{trees})	C+R	$O(npn_{trees})$	$O(pn_{trees})$
Linear Regression	R	$O(p^2n+p^3)$	$O(p)$
SVM (Kernel)	C+R	$O(n^2p+n^3)$	$O(n_{sv}p)$
k-Nearest Neighbours (naive)	C+R	$-$	$O(np)$
Nearest centroid	C	$O(np)$	$O(p)$
Neural Network	C+R	?	$O(pn_{l_1}+n_{l_1}n_{l_2}+\dots)$
Naive Bayes	C	$O(np)$	$O(p)$

Splits

- **3 Data Sets**
  - Training Set
    - Used for training model
  - Validation Set
    - Subset of training set
    - Used for optimizing (tuning parameters, selecting features,...)
  - Test Set
    - Evaluate your model

- **Split Ratio 70/30 or 80/20**
  - Shuffle data & stratify by classes

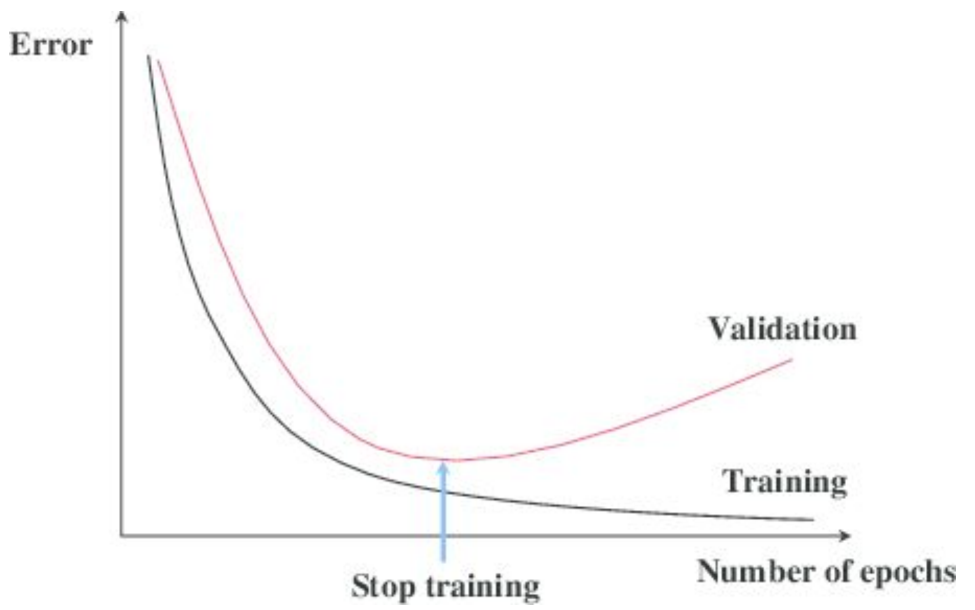
Splits

- **Validation & Test Set**
  - **Data that you expect in future**

- **Example 1 - Turnover for next year**
  - **Last 3-5 years might be helpful**
- **Example 2 - Image Classifier**
  - **User feeds images from smartphone**
  - **Model was trained on images from web**

Data Distribution

- **Good Performance on Training/Validation Set**
- **Reasons for poor performance on test set**
  - Overfit to validation set
    - Use more data
    - Train less



Data Distribution

- Good Performance on Training/Validation Set
- Reasons for poor performance on test set
  - Overfit to validation set
    - Use more data
    - Train less
  - Test set is more complex
    - Model hasn’t seen all the features
    - Model is too small / simple
  - Test set comes from different distribution
    - Example: Image is horizontally flipped

kfold cv?

https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6

Data Distribution

- **When to use different distribution?**
  - **Not enough data from customer / user**
  - **Some features might not be in data**
    - **Collect data from other sources**
    - **Split user data into 2 parts**
      - Training / Validation
      - Testing

Data Distribution

- **Why not using all (user) data?**
  - **Was the way to go in past.**
    - **Other data sources could bias your data**
    - **Other data sources could harm your model**
  - **Some features might not be in data**
    - **Collect data from other sources**
    - **Split user data into 2 parts**
      - Training / Validation
      - Testing

Data Distribution

- **Using customer data only**
  - Was the way to go in past.
    - Other data sources could bias your data
    - Other data sources could harm your model
    - Training models takes more time
  - Adding more data
    - More features
    - Small model might not capture all features
  - Inconsistent data
    - Image classifier
    - Predicting house prices

Data Augmentation

Example: Image-based methods

- Horizontal flip
- Rotate
- Scale
- Crop
- Translation
- Noise
- Blur
- Brightness / Contrast
- RGB -> Gray -> BW

Basic Error Analysis

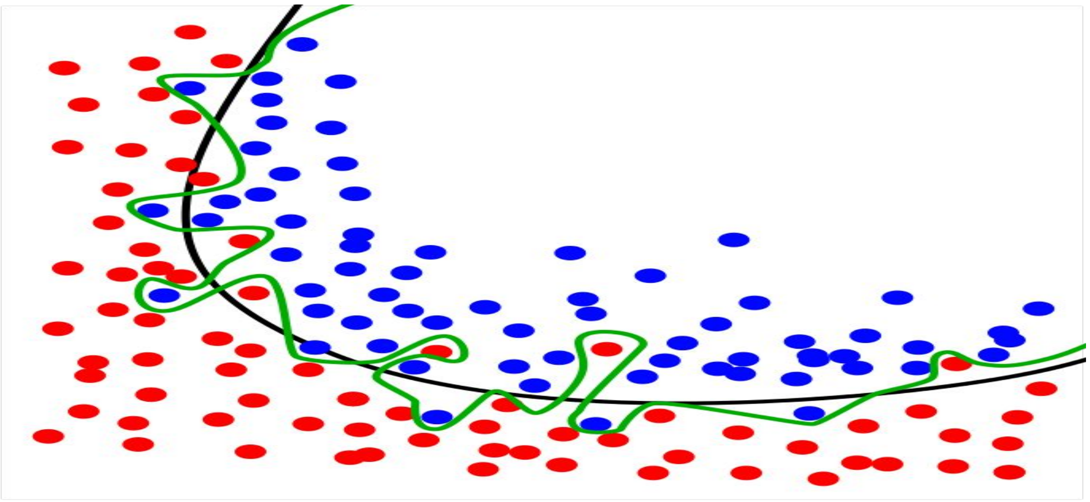
- Irreducible Error

- Reducible Error
  - Bias Error
  - Variance Error

What is Irreducible error?

Irreducible Error:

- The Measure of the amount of noise in the Data
- It is usually caused by unknown variables that may be having an influence on the output variable.
- How good we make our model, our data will have certain amount of noise or irreducible error that can not be removed.



What is Bias?

Bias:

The difference between the average prediction of our model and the correct value which we are trying to predict on training data.

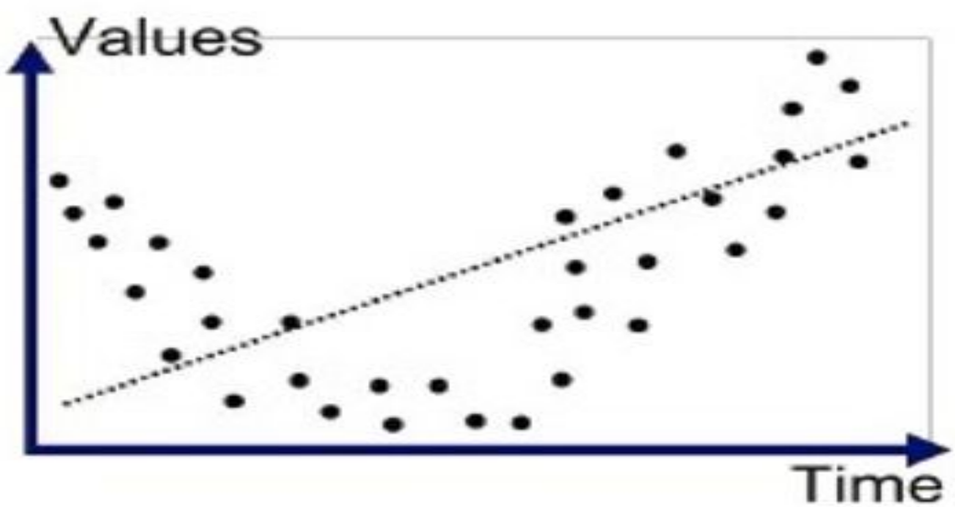
- Poorly Perform on Training data
- Low Training Accuracy
- Example:
  - Training error = 15%
  - Test error = 16%

  The bias as 15%, and variance as 1%(**Variance = Test error - Training error)** This classifier is fitting the training set poorly with 15% error, but it's error on the Test set is barely higher than the training error. This classifier therefore has **high bias** , but low variance.

Underfitting

Underfitting:

- Model can not capture underlying pattern of the data
- High Bias leads to Underfitting



Underfitted

Techniques To Reduce High Bias

Techniques To Reduce High Bias:

- Train Longer
- Train a more complex model
- Decrease Regularization
- New model architecture

Avoidable Bias

Avoidable Bias:

The difference between the training set error and the optimal error rate.

- The “avoidable bias” reflects how much worse your algorithm performs on the training set than the “optimal model.”
- Optimal error rate smallest possible error that the algorithm can reach.
- Difference (Training Error, Human-Level Performance) = Avoidable Bias
- Difference (Validation Error, Training Error) = Variance

Example: Classification Cat vs Not Cat

	Classification error (%)	
	Scenario A	Scenario B
Humans	1	7.5
Training error	8	8
Development error	10	10

	Scenario A	Scenario B
Human Error	1%	7.5%
Avoidable Bias	7%	0.5%
Variance	2%	2%

Techniques for avoidable Bias

Techniques to reduce avoidable Bias

- Increase the model size
- Reduce regularization
- Modify model architecture

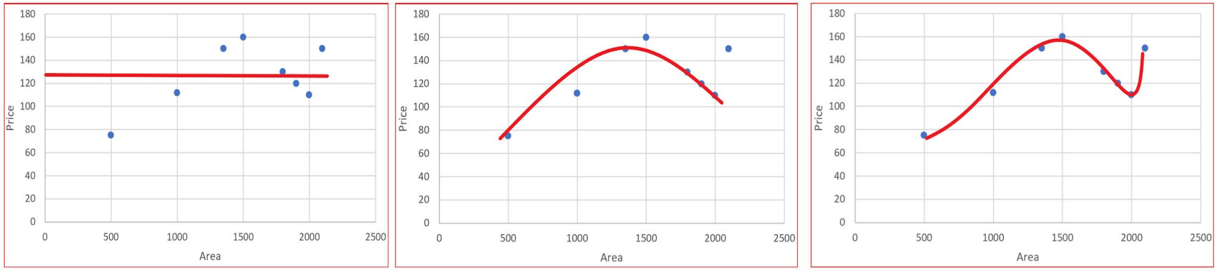
What is Variance?

Variance:

Variance is the variability of Model Prediction for a given data point.

- Low Testing Accuracy
- Example
  - Training error = 1%
  - Test error = 11%
  - Variance = Test error - Training error

  The bias as 1%, and the variance as 10% (=11% - 1%). Thus, it has **high variance** . The classifier has very low training error, but it is failing to generalize to the Test set.



High Bias - underfit | Just Fit | High Variance – overfit

Overfitting

Overfitting:

- Model capture underlying pattern too well of the training data.
- High Variance leads to Overfitting

Overfitted

How To Reduce High Variance?

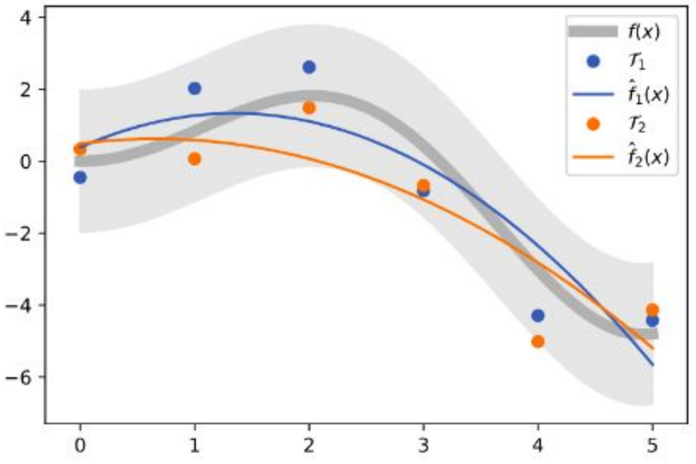
High variance is due to a model that tries to fit most of the training dataset points and hence gets more complex. To resolve high variance issue we need to work on.

- Getting more training data
- Increase Regularization term
- Modify Model Architecture(Neural network architecture)

Mean squared error

- Let the variable we are trying to predict as Y and other covariates as X. We assume there is a relationship between the two such that:
  - Y = f(X) + e
- Assume a model f^(X) of f(X)
- Expected squared error at a point x is

$$Err(x) = E\left[(Y - \hat{f}(x))^2\right]$$



Some experiments with noisy data T1 and T2

Bias-Variance In Terms of MSE

- The Err(x) can be further decomposed as

$$Err(x) = \left(E[\hat{f}(x)] - f(x)\right)^2 + E\left[\left(\hat{f}(x) - E[\hat{f}(x)]\right)^2\right] + \sigma_e^2$$

$$Err(x) = \mathrm{Bias}^2 + \mathrm{Variance} + \mathrm{Irreducible\ Error}$$

Bias-Variance Tradeoff

- **Overfitting** gives too much predictive power even to noise elements
- Attempt to reduce overfitting can also begin to **underfit**

Bias-Variance Tradeoff (Cont.)

- Low Bias and Low Variance
  - Perfect model
- Low Bias and High Variance
  - Inconsistent models
- High Bias and Low Variance
  - Consistent but inaccurate models
- High Bias and High Variance
  - Inaccurate and inconsistent models

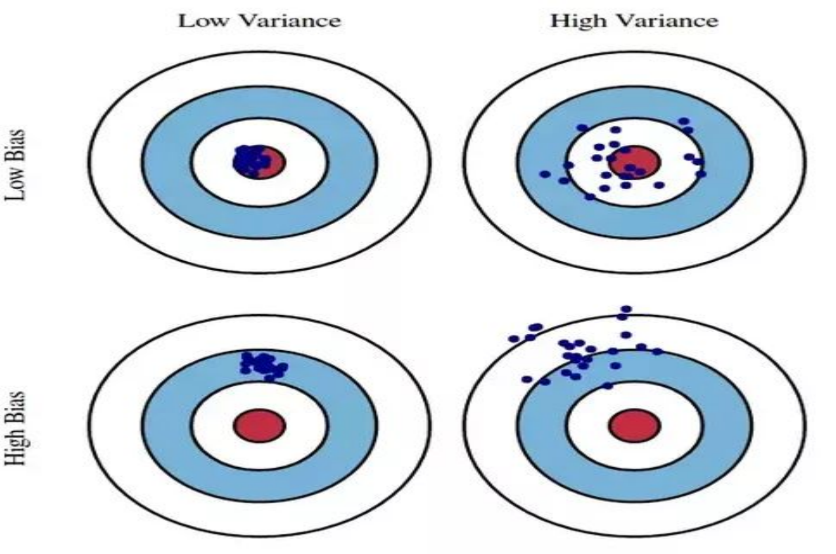
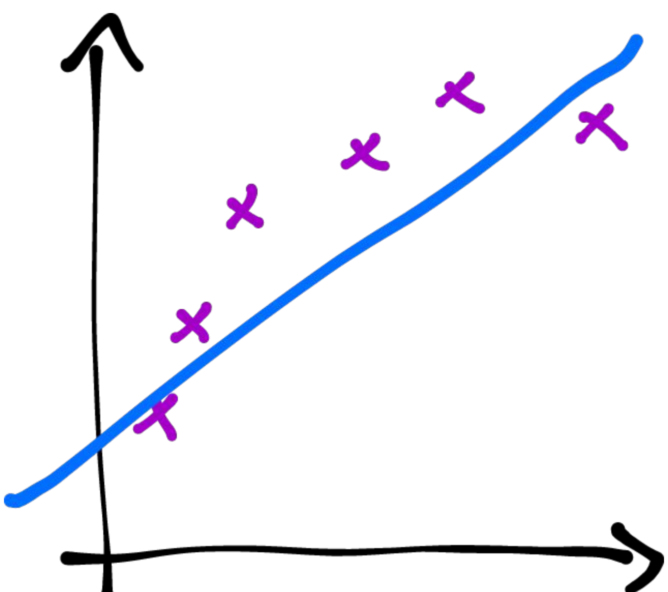


Fig. 1 Graphical illustration of bias and variance.

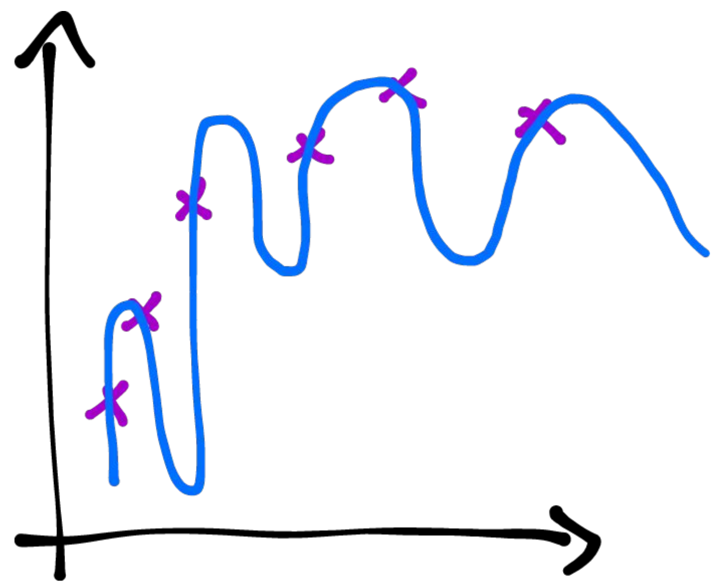
Bias-Variance Tradeoff

In terms of model complexity

- *For the case of* ***high bias****,* a linear model is used.
- *And for the case of* ***high variance****, the model used was super complex*.



© Machine Learning @ Berkeley

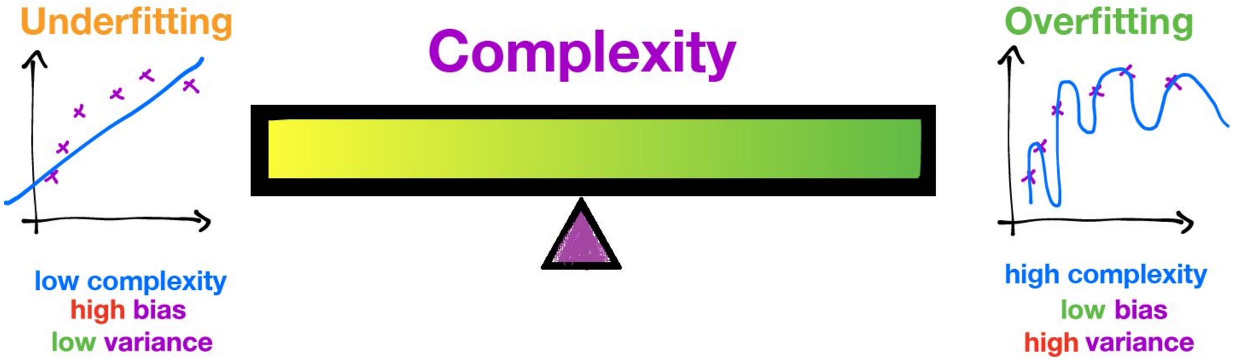


© Machine Learning @ Berkeley

Bias-Variance Tradeoff

In terms of model complexity

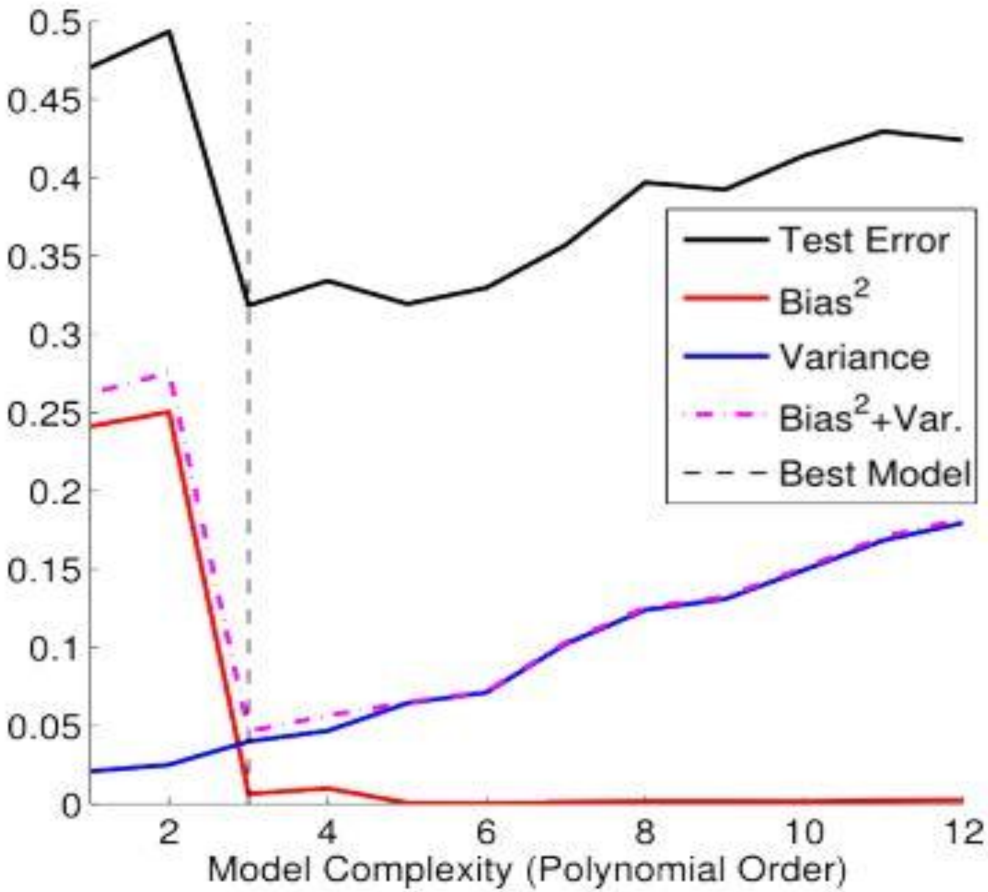
- Low complexity model- Will be prone to underfitting because of high bias and low variance
- High complexity model(Decision trees)- Will be prone to overfitting due to low bias and high variance



© Machine Learning @ Berkeley

Bias-Variance Tradeoff

In terms of model complexity



Balance between Bias-Variance

Regularization is one way to control Bias and Variance

- Which reduces the complexity in the model either by getting rid of the complex features or reducing their importance

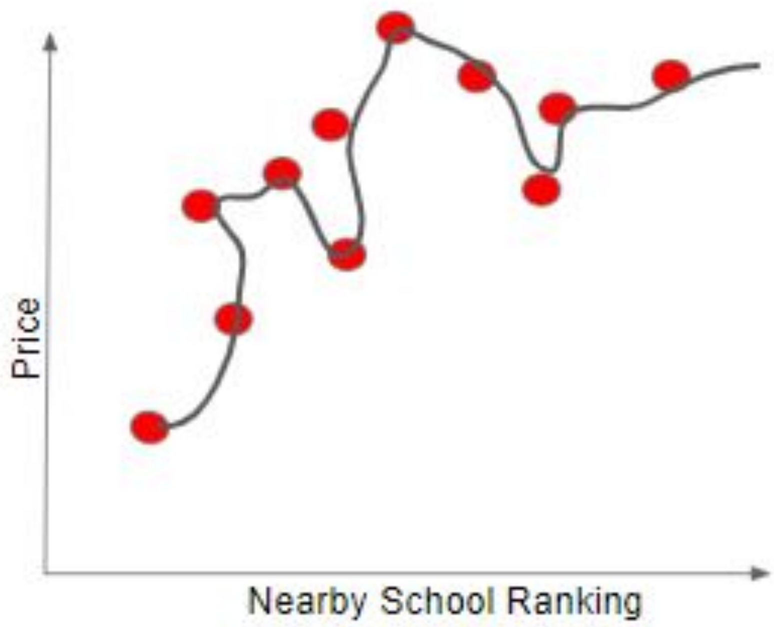
Impact of Regularization

For example if the price of a house is based on 4 features which are Location (X1), Number of bedrooms(X2), Year of Construction(X3), Nearby School ranking(X4).

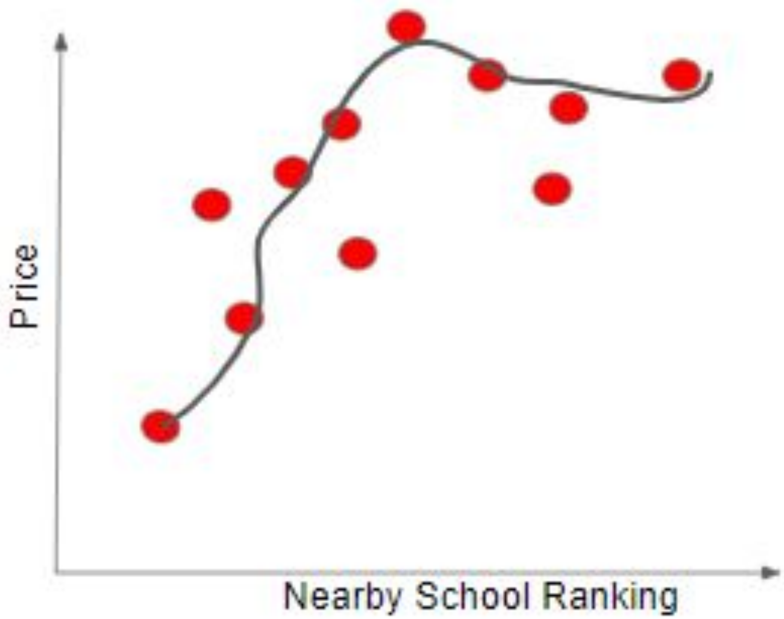
$$Y = 2.5*X_1 + 3 * X_1*X_2 + 1.4*X_3^2 + 4.5 *X_4^3 + 1.3$$

- It reduces the importance of features especially features such as X3 and X4.

Impact of Regularization(Cont.)



Without Regularization



With Regularization

Regularization process

Optimization objective of Linear Regression.

$$W^* = \text{argmin} \ (1/(2n)) \ * \ \left(\sum_{i=1}^{n} (f(Xi) - Yi)^2 + \lambda \sum_{j=1}^{m} (Wj)^2\right)$$

Regularization Process

Optimization objective of Linear Regression.

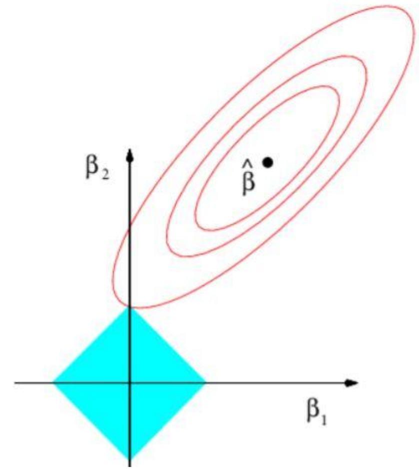
$$W^* = \text{argmin} \ (1/(2n)) * (\sum_{i=1}^{n} (f(Xi) - Yi)^2 + \lambda \sum_{j=1}^{m} (Wj)^2))$$

- Thus, λ acts as a hyperparameter to control the Bias- Variance trade-off.

Regularization Techniques

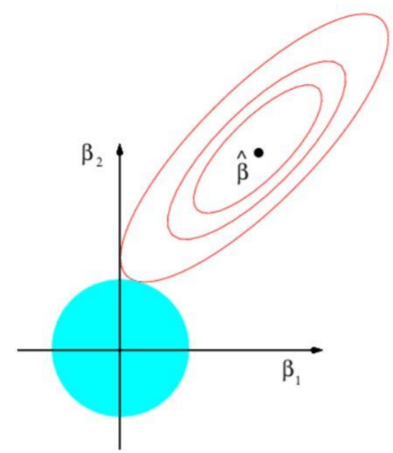
- L1 / **Lasso** Regression
  - adds absolute value of weights

$$= \sum_{i=1}^{N} \left\{ y_i - \sum_{j=0}^{M} w_j \, x_{ij} \right\}^2 + \lambda \sum_{j=0}^{M} |w_j|$$



- L2/ **Ridge** Regression
  - Squared value of weights

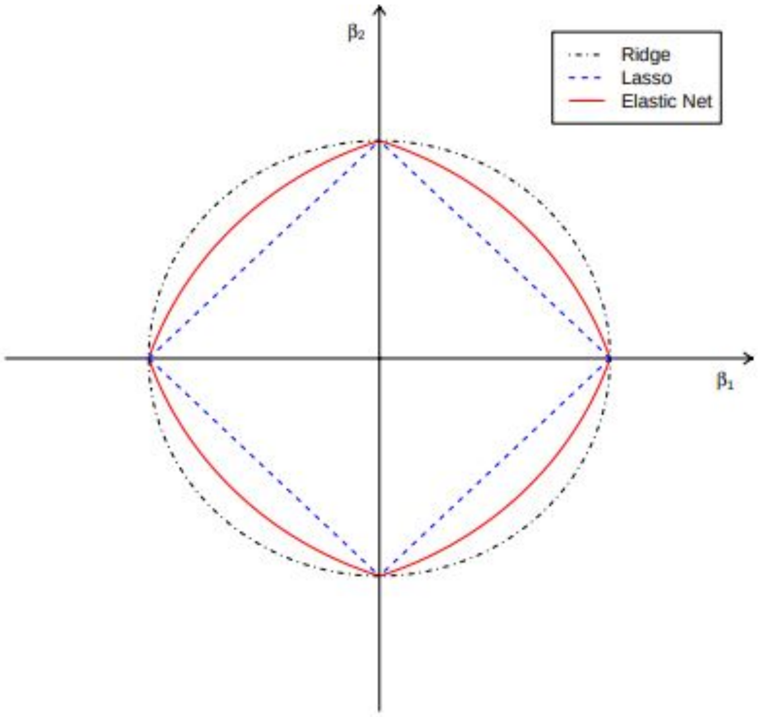
$$\lambda \sum_{j=1}^{m} (Wj)^2$$



Regularization Techniques

- **Elastic Net**
  - Elastic Net includes both L1 and L2 norm regularization terms.

$$\hat{\beta} \equiv \underset{\beta}{\operatorname{argmin}}(\|y - X\beta\|^2 + \lambda_2\|\beta\|^2 + \lambda_1\|\beta|$$



Other Techniques to control Bias-Variance Tradeoff

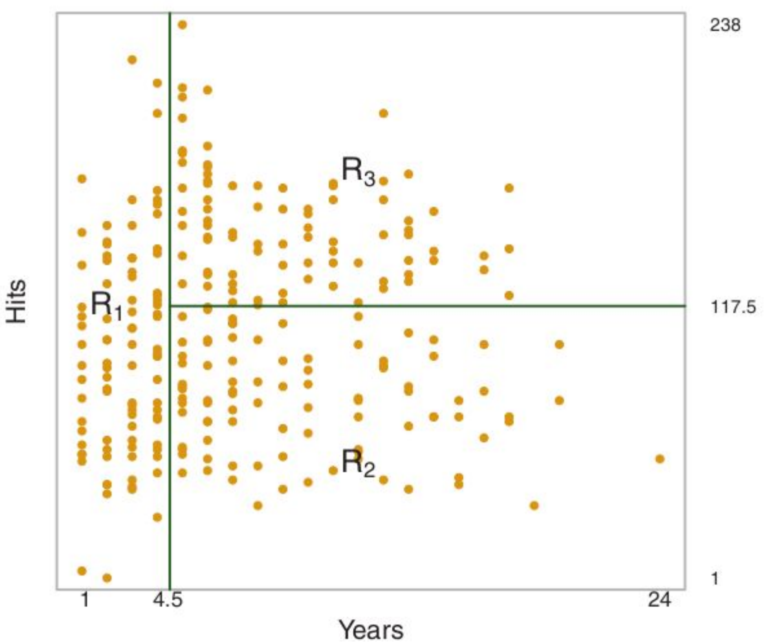
- Feature selection
- Randomization
- Increase data
- Early stopping
- Choice of Algorithm

From Trees to Forests

Random Trees:

Iteratively partition the data and minimize the RSS of the partitions (for regression)

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$



Key points:

- For every region we use the mean or mode to compute the prediction value.
- Top to bottom greedy algorithm that might not yield the best tree.

From Trees to Forests cont'd

Why do we stop?

- We could have as many partitions as to fit single observations in the data, basically memorize the training data -> Overfit

When do we stop?

- Naively set a minimum number of samples per partition
- Naively set a minimum RSS improvement per iteration

Better

Grow a large tree and prune it back -> Regularization

$$\sum_{m=1}^{|T|} \sum_{i:\, x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

From Trees to Forests cont’d

Even better : use an ensemble

- Bagging
- Random Forests
- Boosting

Combine “weak learners” with high bias into a model that has lower bias (Boosting)

Combine “strong learners” with low bias (high variance) into a general model with lower variance (Bagging)

Random forests are somewhere in the middle.

What’s the price?

- Mainly interpretability and more computation

Neural Networks

Overfitting machines

- Large number of parameters
- Variable architectures
- They can theoretically approximate any function (Cybenko, Hornik)

Deep learning just makes things even harder

- Deep architectures
- Millions(Billions) of parameters

Lu et.al proved in 2017 expressivity on Lebesgue integrable functions for width limited deep architectures, still the depth is a variable.

Neural Networks

Strategies

- Naively opt for simpler architectures
- Early stopping - stop when the validation error starts to increase
- L2 regularization - Penalize the parameters

$$R(\boldsymbol{\theta}) = \sum_{i \in Tr} (y_i - f(\mathbf{x}_i; \boldsymbol{\theta}))^2 + \lambda \sum_{j=1}^{p} \theta_j^2$$

- Ensembles
- Dropout
- Data augmentation
- SGD
- Bayesian networks

References

- gpu image: https://miro.medium.com/max/3322/1*WBMLfNKon41AFt8IpwbFFw.png
- data image: https://wallpaperbro.com/img/451165.jpg
- data exp: https://www.ft.com/__origami/service/image/v2/images/raw/https%3A%2F%2Fs3-eu-west-1.amazonaws.com%2Fic-ez-prod%2Fez%2Fimages%2F9%2F8%2F1%2F3%2F3173189-1-eng-GB%2FSectorFocus_171117_line.jpg?source=invchron
- Zettabyte: https://en.wikipedia.org/wiki/Zettabyte
- curse: https://miro.medium.com/proxy/1*kpNRfZ9DgkNPf_dA6DukLw.png
- curse acc:https://miro.medium.com/max/535/1*vah8IolNqlxNHq9ysVzYkw.png
- lin reg complexity: https://www.thekerneltrip.com/machine/learning/computational-complexity-learning-algorithms/
- complex:https://miro.medium.com/max/1663/1*0BvHowg6TmamenKHVsxngA.png
- flip h: https://i.stack.imgur.com/wM1Kc.png
- https://github.com/ajaymache/machine-learning-yearning/tree/master/full%20book
- Irreducible error: https://heartbeat.fritz.ai/bias-variance-tradeoff-to-avoid-under-overfitting-d92f1fcff352
- Bias/var: https://medium.com/ml-research-lab/chapter-3-bias-and-variance-trade-off-in-machine-learning-a449fa1e2729
- Bias/var: https://medium.com/datadriveninvestor/bias-variance-trade-off-fb5fa4c8ab56
- variance: https://www.knowledgehut.com/blog/data-science/bias-variance-tradeoff-in-machine-learning
- Avoaidable bias: http://upscfever.com/upsc-fever/en/data/deeplearning3/8.html
- https://harangdev.github.io/deep-learning/structuring-machine-learning-projects/16/
- Under/Overfitting:https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76
- bias/var image: https://cscherrer.github.io/post/bias-variance/
- bias/var tradeoff image: http://scott.fortmann-roe.com/docs/BiasVariance.html
- bias/var regression image: https://towardsdatascience.com/holy-grail-for-bias-variance-tradeoff-overfitting-underfitting-7fad64ab5d76
- bias/var complexity: https://insidebigdata.com/2014/10/22/ask-data-scientist-bias-vs-variance-tradeoff/
- regularization image1: https://medium.com/swlh/the-what-why-and-how-of-bias-variance-trade-off-55c0b6cf3d00
- regularization image2: Timothy Downey, Lecture notes ML2 WiSe 2018, Beuth University of Applied Sciences.
- regularization image3: https://web.stanford.edu/~hastie/TALKS/enet_talk.pdf
- stop learning: http://fouryears.eu/wp-content/uploads/2017/12/early_stopping.png
- Timothy Downey, Lecture notes ML2 WiSe 2018, Beuth University of Applied Sciences.
- A.Kryzhevsky et. al ImageNet classification with Deep Convolutional Neural Networks
- James et. al An introduction to statistical learning with applications in R