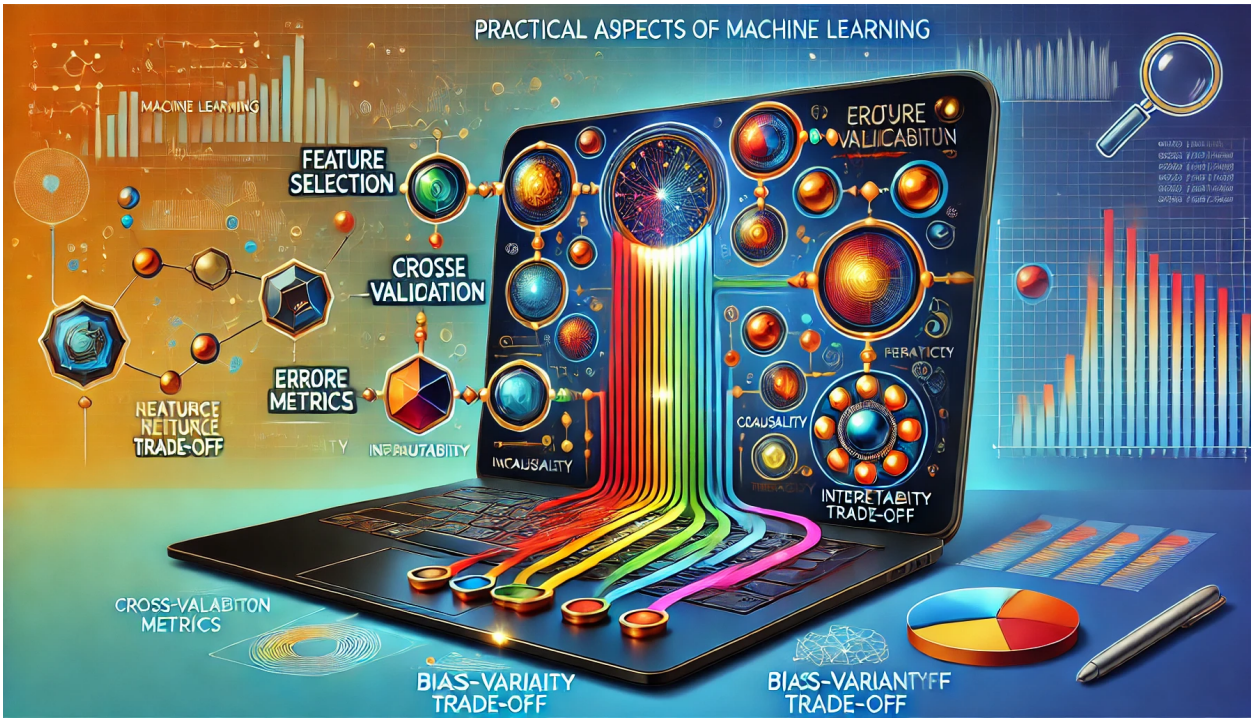
ML in Practice

Mark Asch - ML-PREP

2025



Practical ML

Program

- Method choice: which method should I use for my problem?

  $\Rightarrow$ Bias-Variance tradeoff.
  $\Rightarrow$ Interpretability and explainability.

- EDA (Exploratory Data Analysis): how to use unsupervised learning as an initial phase?

- Model tuning and validation: how can I be sure of the accuracy and robustness of my model?

- Evaluation metrics and predictive power: which metrics measure the predictive power best?

  $\Rightarrow$ Dangers of R^2 and p-values.

- Causality vs Correlation: how to distinguish between them?

$\Rightarrow$ Shapley values

- Feature selection, feature importance and feature engineering.
- PINN (Physics Informed Neural Networks): when and how should they be incorporated?
- Ethics and bias.

CODING ENVIRONMENT

Setup

Python

Please follow general instructions in 01_setup.pdf

ML FRAMEWORK

Mathematical Formulation

- Suppose that we have:

  $\Rightarrow$ a response variable (to explain), Y,
  $\Rightarrow$ p explanatory[1] variables, $X = (X_1, X_2, \ldots, X_p)$,
  $\Rightarrow$ n samples of data, giving an $(n \times p)$ matrix,

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & & \\ \vdots & & \ddots & \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$$

  $\Rightarrow$ a relationship between Y and X of the form

$$Y = f(X) + \epsilon$$

  $\Rightarrow$ where
    $\rightarrow$ f is an unknown function of (columns) $X_1, X_2, \ldots, X_p$

[1] Also called: features, attributes, covariates

→ ϵ is a random error term, independent of X, and with zero mean

- ML is then an ensemble of approaches for estimating the unknown function f with the objectives of

⇒ Prediction: $\hat{Y} = \hat{f}(X)$ where $\hat{f}$ is an estimation for f and $\hat{Y}$ is the resulting prediction

⇒ Inference: to understand how Y varies as a function of X (correlations, importances, linearity, etc.)

Initial Categories

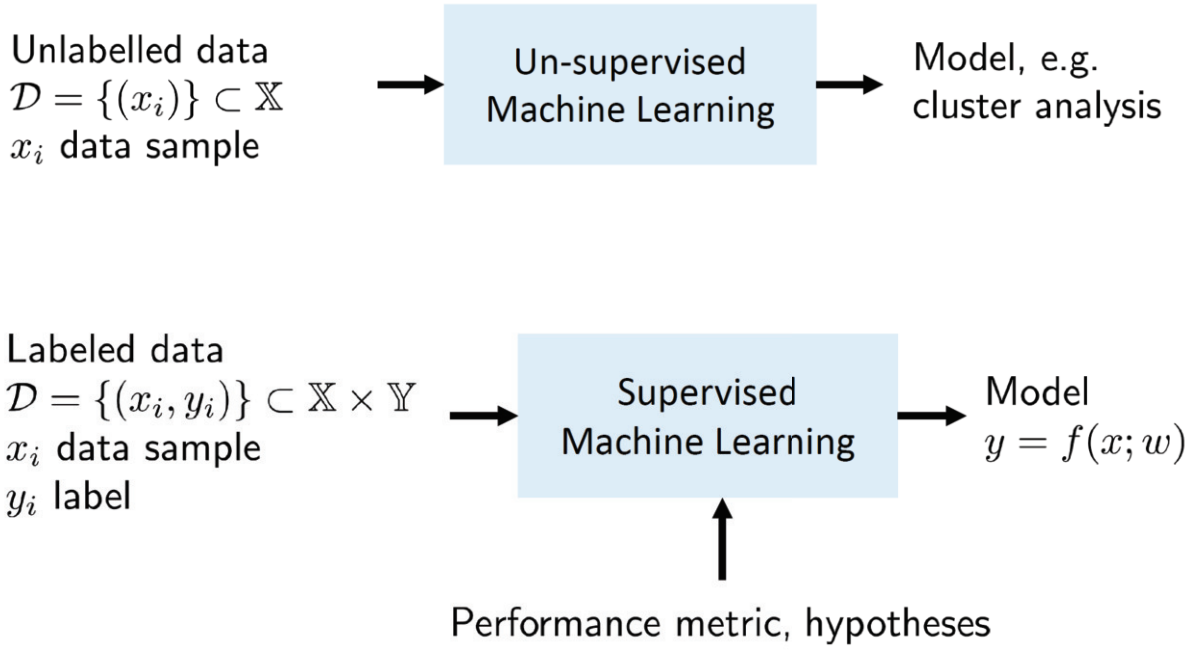
See: Overview - general, Overview - methods

1. Supervised vs. Unsupervised.

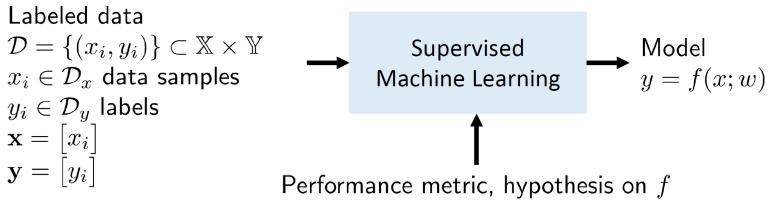
2. Regression vs. Classification.

3. Mixtures of the above?

Supervised vs. Unsupervised

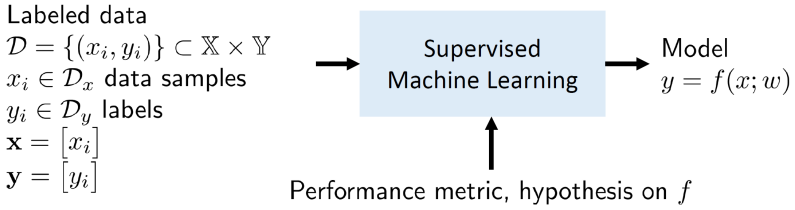


Regression vs. Classification



Model purpose – Regression

- The model f shall map $x \mapsto y$ and approximate an unknown function $\hat{f} : \mathbb{X} \to \mathbb{Y}$
- $y_i \in \mathbb{Y} \subseteq \mathbb{R}^{n_y}$
- Examples: data-driven modeling, energy forecasting, ...



Model purpose – Classification

- The model f shall map $x \mapsto y$ and approximate an unknown function $\hat{f} : \mathbb{X} \to \mathbb{Y}$
- $y_i \in \mathbb{Y} \subseteq \mathbb{N}^{n_y}$
- Examples: spam filter, fraud detection, fault detection, ...

The only difference is the space over which the response Y is defined!

METHOD CHOICE

Classes of ML Methods

- supervised learning
- unsupervised learning
- reinforcement learning (AlphaGo, AlphaFold)
- self-supervised learning (LLMs, ChatGPT)

Table of ML Methods

Class	Model	Task
Supervised	Linear regression	R
	CART (trees)	R, C
	SVM	R, C
	NN	R, C
	k-NN	C
	Naive Bayes	C
Unsupervised	k-means	clustering
	hierarchical	clustering
	PCA	patterns

- R = regression, C = classification
- CART = Classification and Regression Trees
- SVM = Support Vector Machine
- NN = Neural Network
- PCA = Principal Component Analysis

Methods - presentations

Lectures and code examples for all the above methods are available on the CMU-IMU-2023 course website.

How to choose a model?

- Once the tuning (hyper)parameters have been determined/chosen (see: cross-validation below), we still must choose between several models

  $\Rightarrow$ the choice will largely depend on the data characteristics and the type of questions we ask

- But, predicting which model will be the most pertinent, is in general quite difficult...

Scheme for "How to choose a model"

A recommended scheme for finalizing the choice is as follows:

1. Begin with a few models that are the least interpretable and the most flexible. These models have a strong chance of producing better precision.

   (a) SVM.
   (b) Random Forests (RF).
   (c) Trees with boosting (xxxBoost).

2. Study simpler models, that are less opaque.

   (a) Linear models.
   (b) (GAM/GLM)
   (c) Naive Bayes.
   (d) k-NN.
   (e) Logistic Regression.
   (f) Regression Splines (MARS).

3. Use, if possible, the simplest model (see: bias-variance trade-off) that approximates reasonably well the performance of the more complex models.

Remark 1. In ML, the algorithm learns the parameters of the model, usually by optimization of a error-loss score. The method itself is defined by hyperparameters. These hyperparameters control the learning process. They have default values, but need to be tuned for each new dataset.

EDA

Recall: EDA

See: Overview - general

Remark 2. In addition to plots and summary statistics, unsupervised learning techniques often play an important role in identifying structure in the data.

CROSS-VALIDATION and TUNING

Recall: CV and Tuning

See: Resample,, elementary train-test tutorial, a bit more advanced CV tutorial

- Take-home lessons:

  ⇒ Basic, train-test split is not recommended since a single split may not be representative, and the resulting score may not be a reliable estimate of the predictive power on unseen data (i.e. in production.)

  ⇒ Simple, k-fold CV is better, but is rarely sufficient to ensure model reliability.

  ⇒ Nested CV with inner loop to select hyperparameters, outer loop to evaluate the best model, is probably the best option.

  ⇒ Repeated k-fold CV.

  → A single run of the k-fold cross-validation procedure may result in a noisy estimate of model performance.

  → Different splits of the data may result in very different results.

- → The mean result is expected to be a more accurate estimate of the true unknown underlying mean performance of the model on the dataset, as calculated using the standard error.
- → Usual numbers of repeats include 3, 5, and 10.

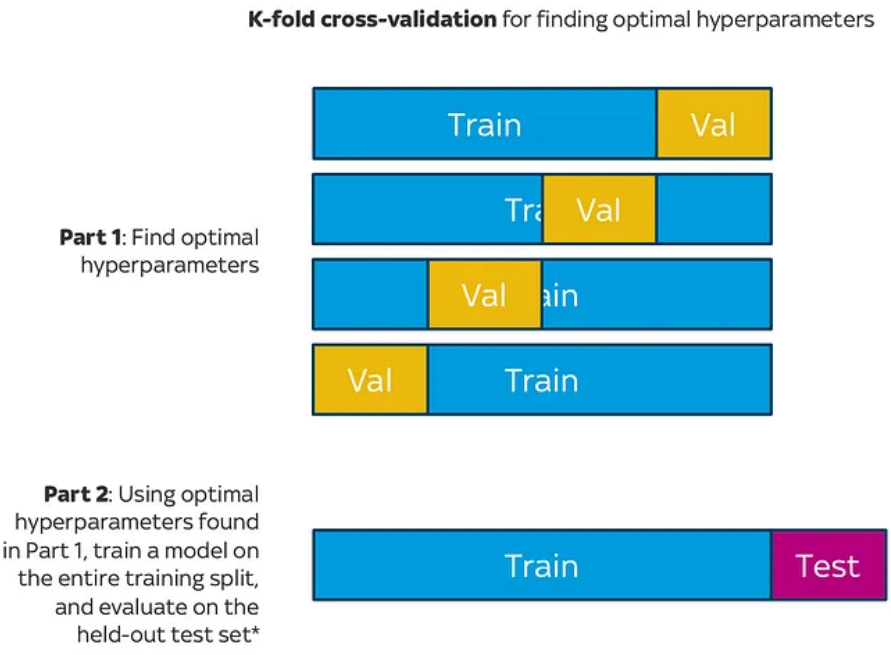
⇒ Time-series data requires special splitting, that respects the temporality (autocorrelation). For this one should use the scikit function TimeSeriesSplit

⇒ Geospatial data often requires special splitting, that respects the spatial distribution (autocorrelation). For this one should use special techniques, presented in the advanced geospatial data analysis lecture notes.

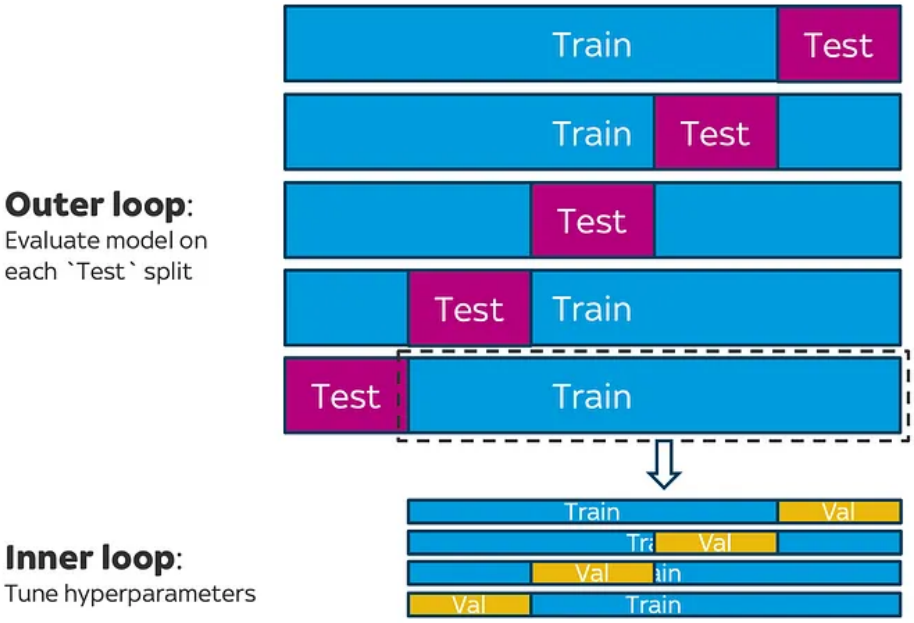
Nested CV

Single split:

k-fold CV:



Nested k-fold CV:



See Example notebook.

Resampling: Take-Home Lessons

No resampling method is uniformly better than another and the choice of a particular method should consider several aspects.

- If the sample size is small, repeated (5- or) 10-fold cross-validation is recommended for several reasons:

  $\Rightarrow$ the bias and variance properties are good and,
  $\Rightarrow$ given the sample size, the computational costs are not large.

- If the goal is to choose between models, as opposed to getting the best indicator of performance, it is advisable to use a bootstrap method since these have very low variance.

- For large sample sizes, the differences between resampling methods become less pronounced, and computational efficiency becomes the critical factor. Here,

simple 10-fold cross-validation should provide acceptable variance and low bias, and is relatively cheap to compute.

Finalized Model

- A final machine learning model is a model that you will use to make predictions on new data.
- You finalize a model by applying the chosen, tuned machine learning procedure on all of your data. That's it.
- See Example notebook.

EVALUATION METRICS

Loss Functions and Metrics

We need to distinguish between metrics for the 2 following cases. See: linear regression, model selection.

- For a given model: MSE, R^2, etc.
- For comparing different models: AIC, BIC, etc.

Definitions

- Coefficient of determination (R-squared)

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}} = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}}$$

  represents the proportion of the variance "explained" by the model. Beware of the limitations of this coefficient!

- Mean-squared Error:

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 ,$$

  where $\hat{y}_i$ is the predicted value by the model for observation i. This can be more reliable than R-squared, but is context-dependent (not between 0 and 1).

- Mean Classification Error:

$$\text{MCE} = \frac{1}{n}\sum_{i=1}^{n} I\left(y_i \neq \hat{y}_i\right),$$

where I is the indicator function and $\hat{y}_i$ is the class, or label predicted by the model for observation i.

Recall: Measures of Performance

- We compute individual performance scores for each cycle of the CV loop:

  $\Rightarrow$ For regression problems, use the mean squared error.
  $\Rightarrow$ For classification problems, use the misclassification rate.

- Then, the general score of the CV is the average of the k individual scores,

$$\mathrm{CV}_{(k)} = \frac{1}{k} \sum_{j=1}^{k} \mathrm{MSE}_j,$$

  to which we add a measure of the variance, the standard error, which is the standard deviation of the sampling distribution,

$$\mathrm{SE} = \frac{s}{\sqrt{n}},$$

where s is the empirical standard deviation, the square root of the empirical variance,

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 .$$

- Finally, an estimated 95% confidence interval of the empirical mean[2], is defined by

$$\bar{x} \pm 1.96\mathrm{SE}.$$

[2]This confidence interval is exact for a Gaussian distribution, but random i.i.d. errors and their sums are only approximately Gaussian.

Recall: Mean-Squared Error

- Since we are estimating an unknown parameter, or more often, unknown function, we need to evaluate

  $\Rightarrow$ how good our estimation is?
  $\Rightarrow$ the quality of the estimation?

- This is traditionally done by using a discrete L_2 norm, though other norms are possible and are used in specific circumstances—see below.

- Before defining the MSE, we need to recall the definitions of

  $\Rightarrow$ expectation (or expected value):

$$\mathrm{E}[X] = \sum_i x_i f(x_i),$$

where f is the (discrete) probability function.

$\Rightarrow$ bias:

$$\mathrm{Bias}(T, \theta) = \mathrm{Bias}(T) = \mathrm{E}[T] - \theta,$$

where T is a statistic used to estimate a parameter, and θ is the parameter to be estimated

$\Rightarrow$ variance:

$$\mathrm{Var}\left[X\right] = \mathrm{E}\left[(X - \mu)^2\right] = \mathrm{E}\left[X^2\right] - \mu^2,$$

where μ is the mean of X.

Definition 1 (Mean Squared Error). The mean squared error measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value. MSE is a risk function, corresponding to the expected value of the squared error loss,

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(e_i\right)^2,$$

or for an estimator

$$\mathrm{MSE}(\hat{\theta}) = \mathrm{E}_\theta\left[\left(\hat{\theta} - \theta\right)^2\right].$$

The MSE is the second moment (about the origin) of the error, and thus incorporates both the variance of the estimator (how widely spread the estimates are from one data sample to another) and its bias (how far off the average estimated value is from the true value).

The MSE either assesses the quality of a predictor (i.e., a function mapping arbitrary inputs to a sample of values of some random variable), or of an estimator (i.e., a mathematical function mapping a sample of data to an estimate of a parameter of the population from which the data is sampled). The definition of an MSE differs according to whether one is describing a predictor or an estimator.

Remark 3. The mean squared error has the disadvantage of heavily weighting outliers. This is a result of the squaring of each term, which effectively weights large errors more heavily than small ones. This property, undesirable in many

applications, has led to the use alternatives such as the mean absolute error, which is a discrete L_1 norm, or scores based on the median.

Definition 2 (Mean Absolute Error)**.** The mean absolute error is the average of the sum of absolute errors,

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i|\,,$$

or for an estimator

- A python snippet:

```
from sklearn.metrics import mean_absolute_error
import numpy as np
# Generate some sample data
y_true = np.array([1, 2, 3, 4, 5])
y_pred = np.array([1.5, 2.5, 2.8, 4.2, 4.9])
# Calculate the MAE
mae = mean_absolute_error(y_true, y_pred)
print("Mean Absolute Error", mae)
```

Remark. The MAE has two important properties.

1. It is robust to outliers. The MAD (median absolute deviation) is even more robust.

2. It has the same units as the response variable.

Bias-Variance Decomposition of MSE

Theorem 1. *The bias-variance decomposition of the MSE is given by*

$$\begin{aligned}\mathrm{MSE} &\doteq \mathrm{E}\left[\left(y - \hat{f}\right)^2\right] \\ &= \mathrm{Bias}\left[\hat{f}\right]^2 + \mathrm{Var}\left[\hat{f}\right] + \sigma^2,\end{aligned}$$

where $y = f(x) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$[3] *and* $\hat{f}(x)$ *is an approximation/model of the true (unknown) function* $f(x)$.

Proof. First expand the definition of MSE,

$$\begin{aligned}\mathrm{MSE} &\doteq \mathrm{E}\left[\left(y - \hat{f}\right)^2\right] \\ &= \mathrm{E}\left[y^2 - 2y\hat{f} + \hat{f}^2\right] \\ &= \mathrm{E}\left[y^2\right] - 2\mathrm{E}\left[y\hat{f}\right] + \mathrm{E}\left[\hat{f}^2\right].\end{aligned}$$

[3]This Gaussian noise model is the most arbitrary (least restrictive), assuming only a central tendency and low chance of extreme values.

Now look at each of the three terms.

$$\mathrm{E}\left[\hat{f}^2\right] = \mathrm{Var}\left[\hat{f}\right] + \mathrm{E}\left[\hat{f}\right]^2$$

from variance decomposition.

$$\begin{aligned}
\mathrm{E}\left[y^2\right] &= \mathrm{E}\left[(f+\epsilon)^2\right] \\
&= \mathrm{E}\left[f^2\right] + 2\mathrm{E}\left[f\epsilon\right] + \mathrm{E}\left[\epsilon^2\right] \\
&= f^2 + 2f \cdot 0 + \sigma^2
\end{aligned}$$

from definition of the noise distribution ϵ. Finally,

$$\begin{aligned}
\mathrm{E}\left[y\hat{f}\right] &= \mathrm{E}\left[(f+\epsilon)\,\hat{f}\right] \\
&= \mathrm{E}\left[f\hat{f}\right] + \mathrm{E}\left[\epsilon\right]\mathrm{E}\left[\hat{f}\right] \\
&= f\mathrm{E}\left[\hat{f}\right].
\end{aligned}$$

Putting it all together, we find

$$\begin{aligned}
\mathrm{MSE} &= \left(f^2 + \sigma^2\right) - 2f\mathrm{E}\left[\hat{f}\right] + \left(\mathrm{Var}\left[\hat{f}\right] + \mathrm{E}\left[\hat{f}\right]^2\right) \\
&= \left(f - \mathrm{E}\left[\hat{f}\right]\right)^2 + \sigma^2 + \mathrm{Var}\left[\hat{f}\right] \\
&= \mathrm{Bias}\left[\hat{f}\right]^2 + \mathrm{Var}\left[\hat{f}\right] + \sigma^2.
\end{aligned}$$

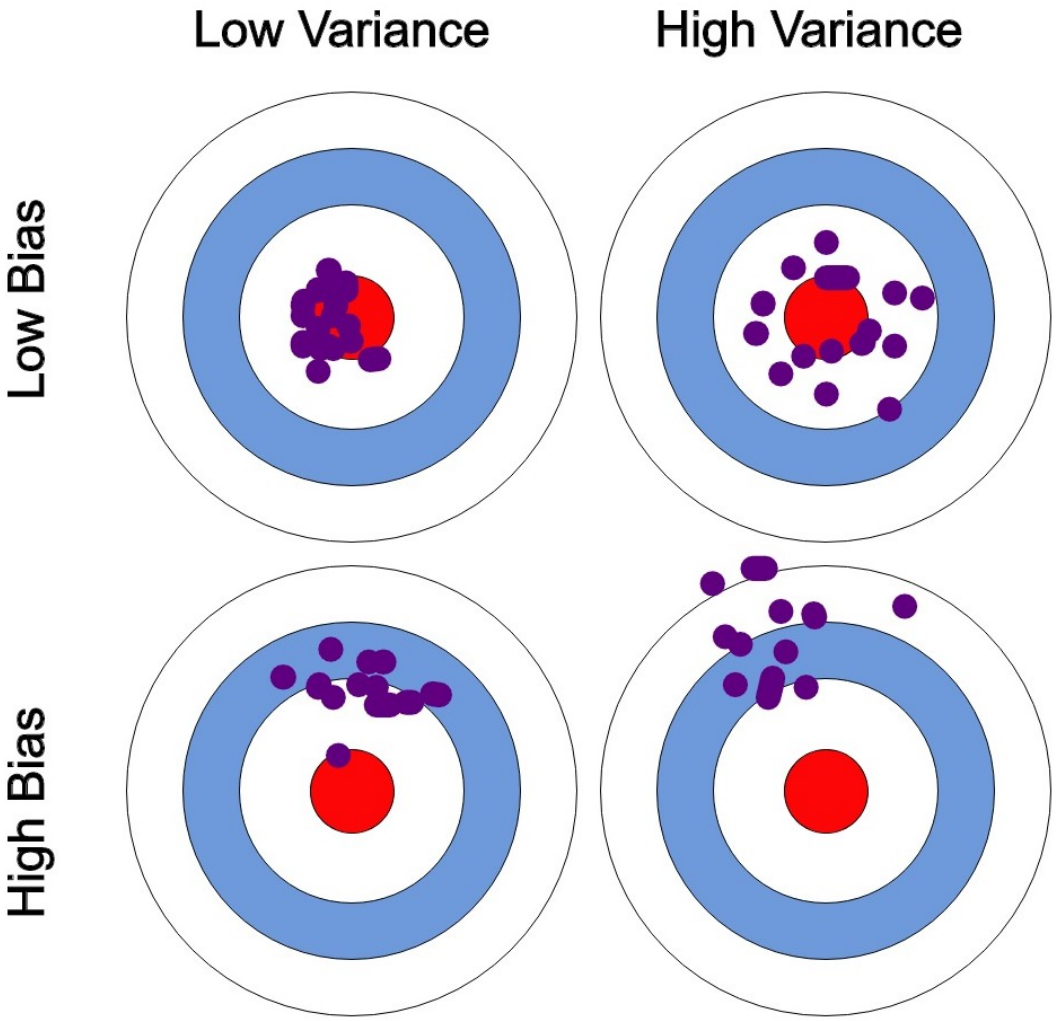
□

Recall: Bias-Variance Compromise

- Questions:

  ⇒ What is bias? (see also Ethics lecture)
  - → Low Bias: Predicted data points are close to the target. The model has fewer assumptions about the form of the target function.
  - → High-Bias: Predicted data points are far from the target. The model has more assumptions about the form of the target function.

  ⇒ What is variance?
  - → Low Variance: Data points are close to each other, and as a result are close to the function. The model suggests small changes to the estimate of the target function with changes in the training dataset.
  - → High Variance: Data points are spread out and as a result are far from the function. The model suggests large changes to the estimate of the target function with changes in the training dataset.


Low Variance
High Variance
Low Bias
High Bias

Bias-Variance Compromise II

- We can now return to the question of the tradeoff between bias and variance.

- Looking at the bias-variance decomposition of the MSE,

$$\mathrm{MSE} = \mathrm{Bias}\left[\hat{f}\right]^2 + \mathrm{Var}\left[\hat{f}\right] + \sigma^2,$$

  we observe that we need to simultaneously minimize both bias and variance. This is the reducible part of the error, whereas σ^2 is the intrinsic, irreducible part and cannot be reduced by the ML model.

- Theoretical result (see also the "compromise curve" above):

  $\Rightarrow$ Decreasing the bias will mechanically increase the variance.

  $\rightarrow$ Low bias $\Longrightarrow$ high variance.

$\Rightarrow$ Decreasing the variance will mechanically increase the bias.

$\rightarrow$ Low variance $\Longrightarrow$ high bias.

- **Conclusion:**

$\Rightarrow$ To build a good model, we need to find a good balance/compromise/tradeoff between bias and variance that minimizes the total/generalization error.

$\Rightarrow$ An optimal balance of bias and variance should neither overfit nor underfit the model. This is achieved by hyperparameter tuning on the basis of systematic grid-search (trial and error, or experience, for the ranges) and cross-validation. See above, and examples.

A simple example that brings out many points

- **Basic fact:** most of ML is about pattern recognition—see C. M. Bishop, PRML, Springer, 2006.

  $\Rightarrow$ we have a bunch of data, $\mathbf{x}$ and corresponding observations, $\mathbf{y}$

  $\Rightarrow$ we seek the relation (recognize the pattern), f, that relates $\mathbf{x}$ to $\mathbf{y}$

  $$\mathbf{y} = f(\mathbf{x}),$$

  or,

  $$f : \mathbf{x} \longmapsto \mathbf{y}$$

- **Our goal**: make predictions for new (unseen) data (inputs)

Polynomial Curve Fitting

- suppose that

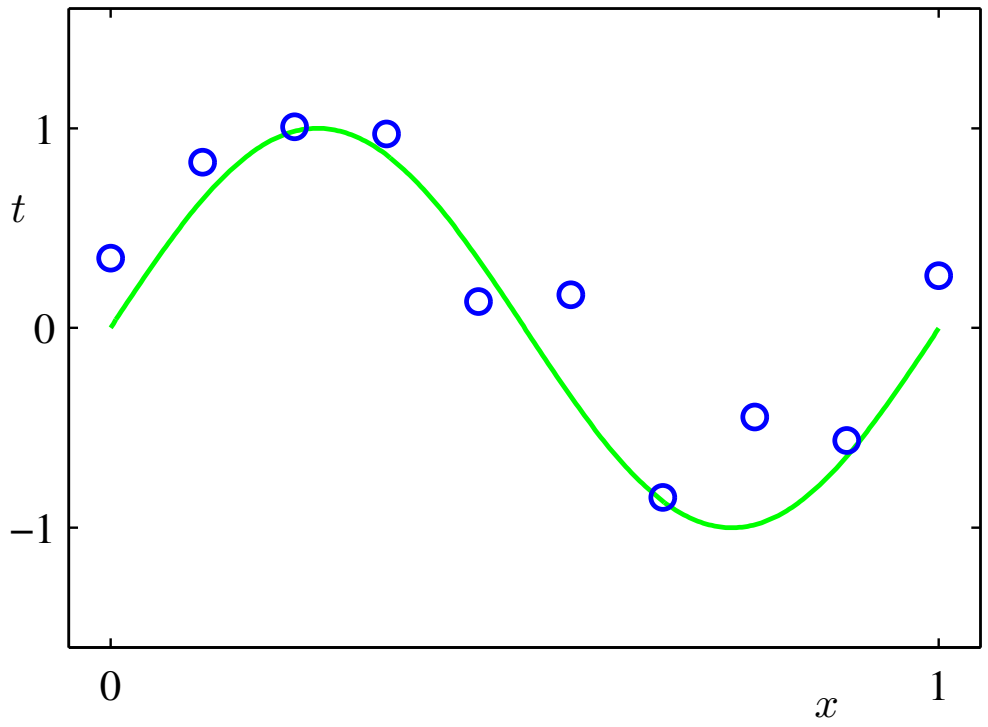
$$y = \sin(2\pi x) + \text{noise}$$

- given: a training set of N observations of x,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

and the corresponding (noisy) observations,

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$



- data generation from

$$y_i = \sin(2\pi x_i) + \epsilon,$$

where the noise term,

$$\epsilon \sim \mathcal{N}(0, \sigma)$$

is due to intrinsic/natural variability and other uncertainties from the data collection methods.

- **Objective:** from the training set, learn/discover an approximation of the relation f that will enable us to make predictions based on

  $\Rightarrow$ probability theory
  $\Rightarrow$ (polynomial) curve fitting

- PCF: suppose that the data can be represented by the polynomial/"linear model"

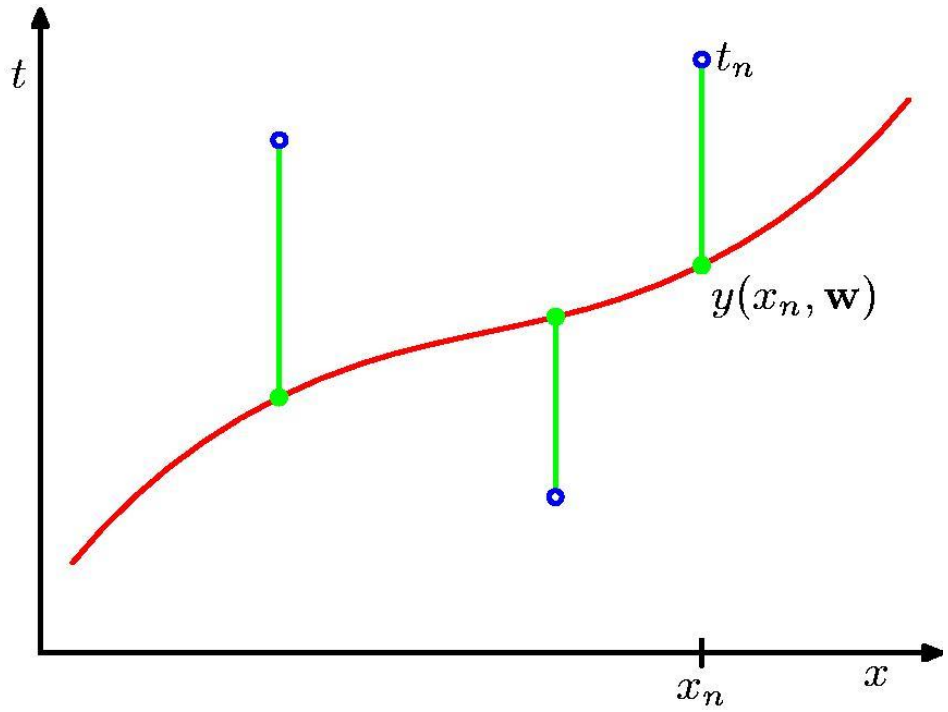
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M = \sum_{k=0}^{M} w_k x^k$$

  with weight vector (coefficients) $\mathbf{w}$.

  $\Rightarrow$ compute the coefficents by minimizing an error/loss function—this is (often) a least-squares approach, where the loss function is defined as

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{N} \left[\hat{y}(x_i, \mathbf{w}) - y_i \right],$$

  where $\hat{y}_i$ is the approximation obtained from our polynomial model.



- in the case of a quadratic loss function, the gradient is linear and the solution to the optimization problem exists and can be found in closed form using the normal form, or psuedo-inverse,

$$\mathbf{w}_* = \left(X^{\mathrm{T}} X\right)^{-1} X^{\mathrm{T}} \mathbf{y},$$

where X is the (rectangular) coefficient matrix of dimension $(N \times M)$ with $N > M$ and $\mathbf{y}$ is the data/observation vector.

- the choice of M is a model selection problem

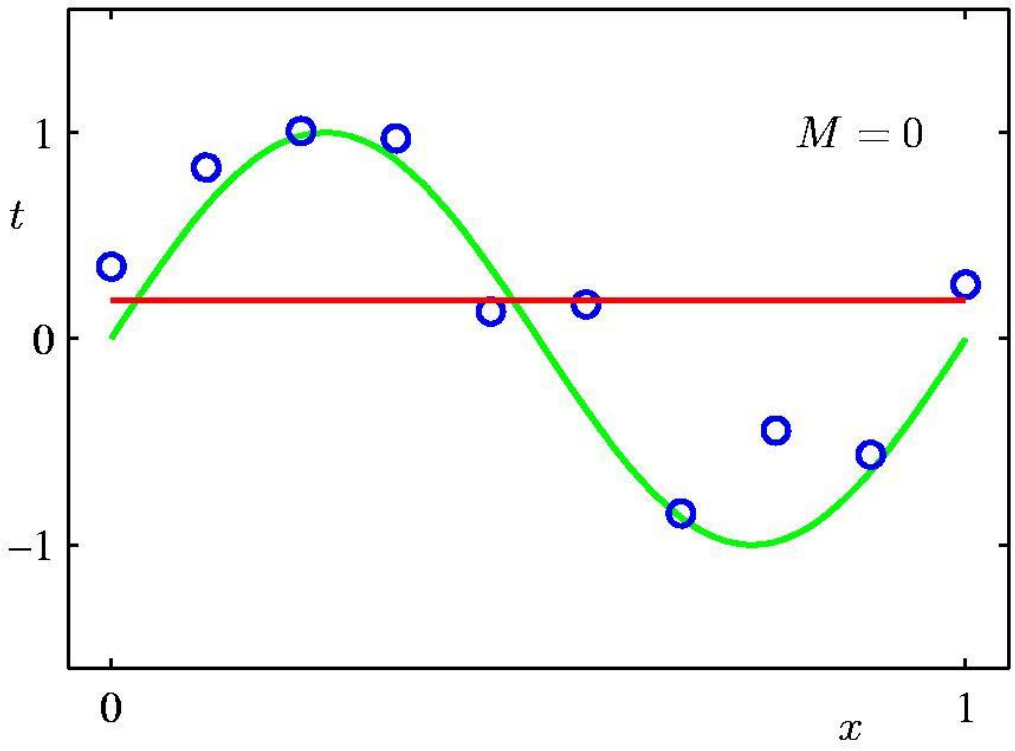
  $\Rightarrow$ we compute the solution for 4 values: $M = 0, 1, 3, 9$—see below

  $\Rightarrow$ Model $M = 0$ produces underfitting—it is too simple and does not find the trend

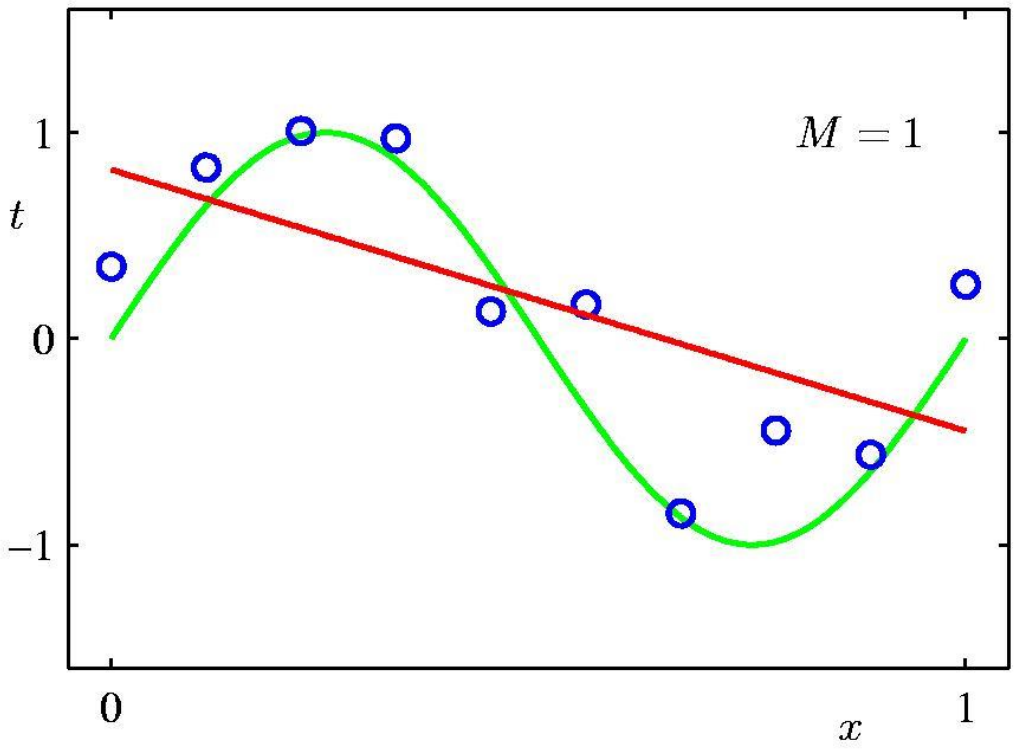
  $\Rightarrow$ Model $M = P$ produces overfitting—it fits the data points perfectly, but is "brittle" and does not generalize

- **Conclusion:** we must seek a compromise between the two.

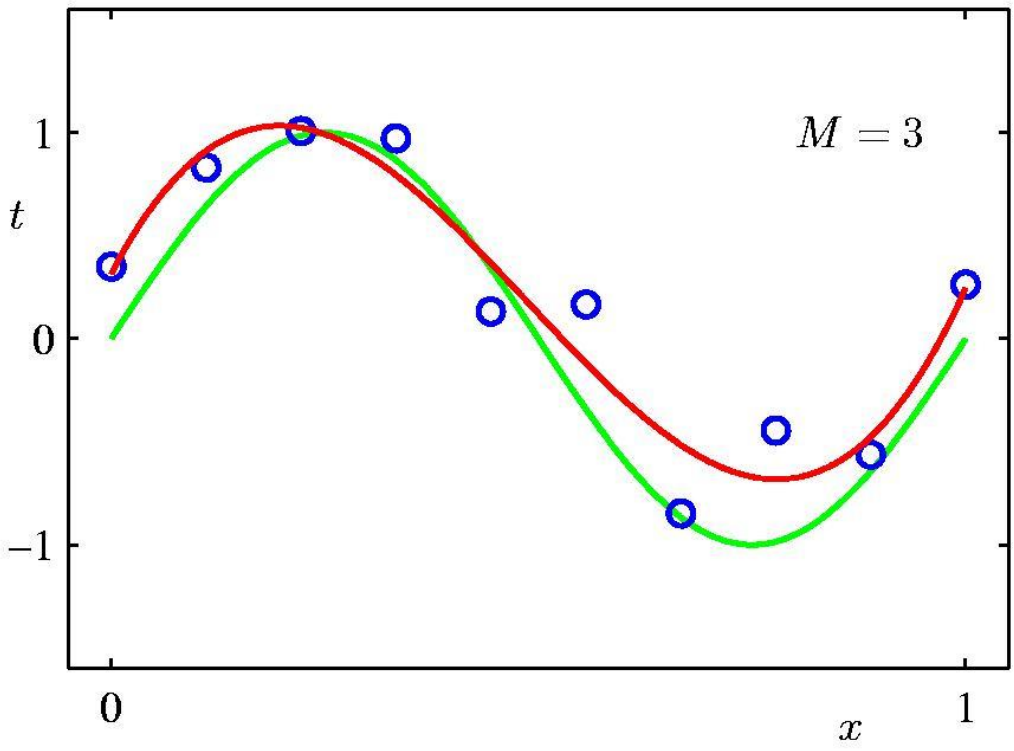
0^{th} Order Polynomial



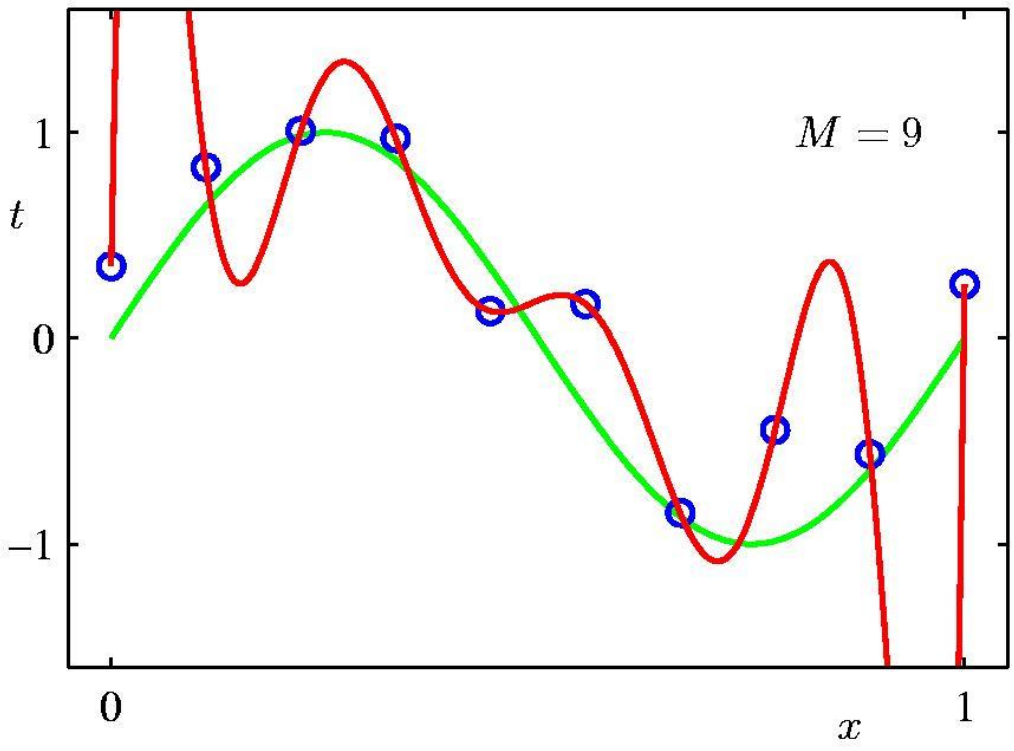
1st Order Polynomial



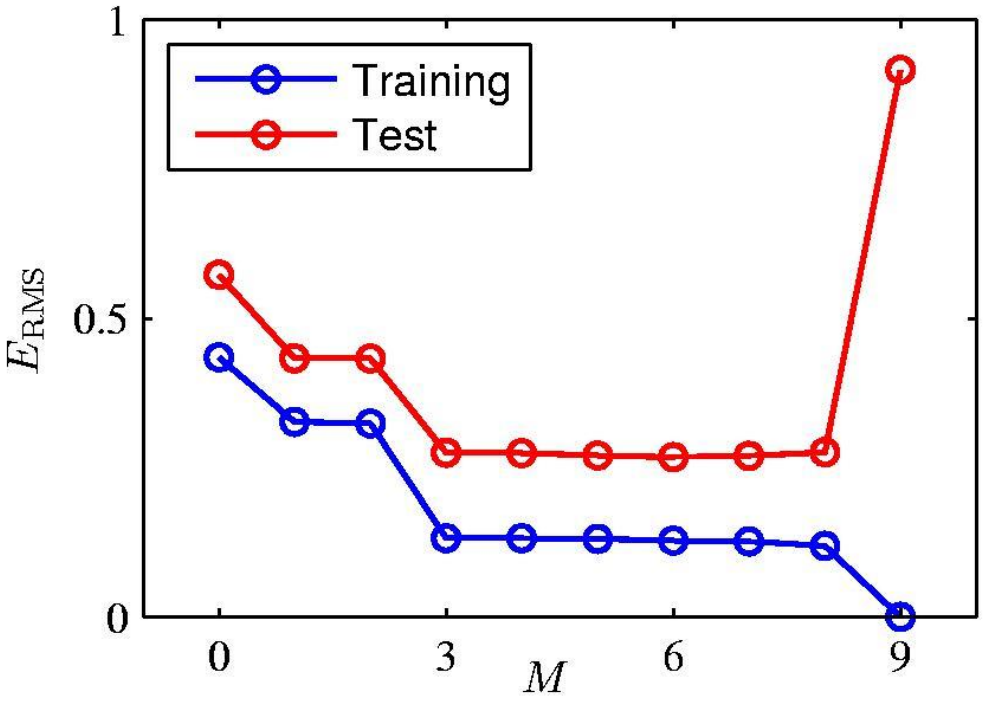
3rd Order Polynomial



9th Order Polynomial



Over-fitting

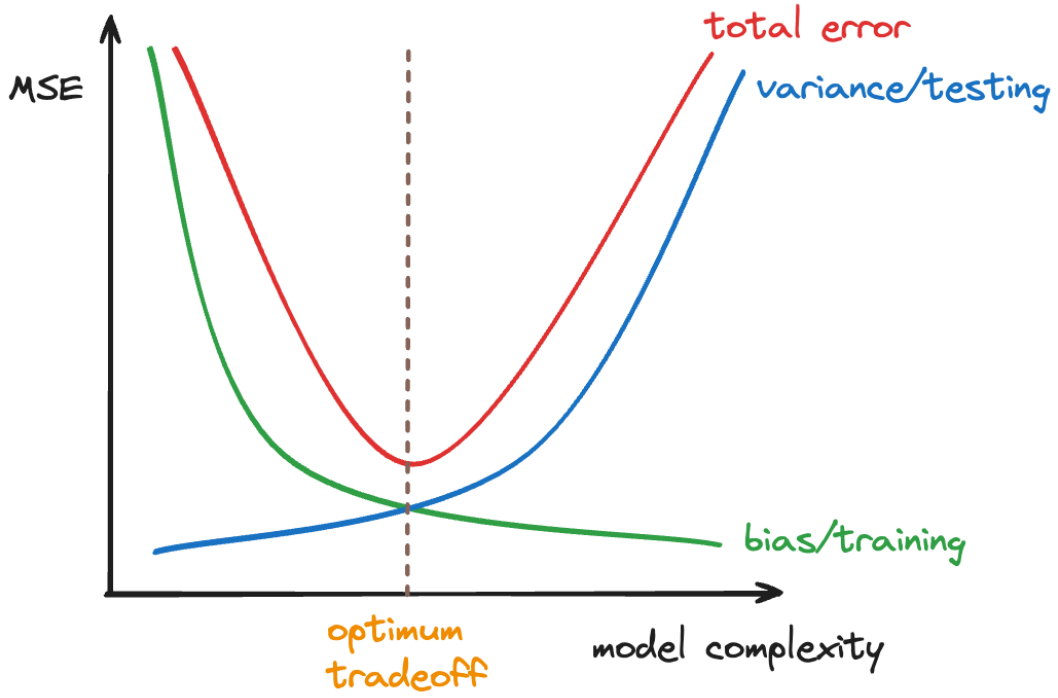


Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^{\star})/N}$

Polynomial Coefficients

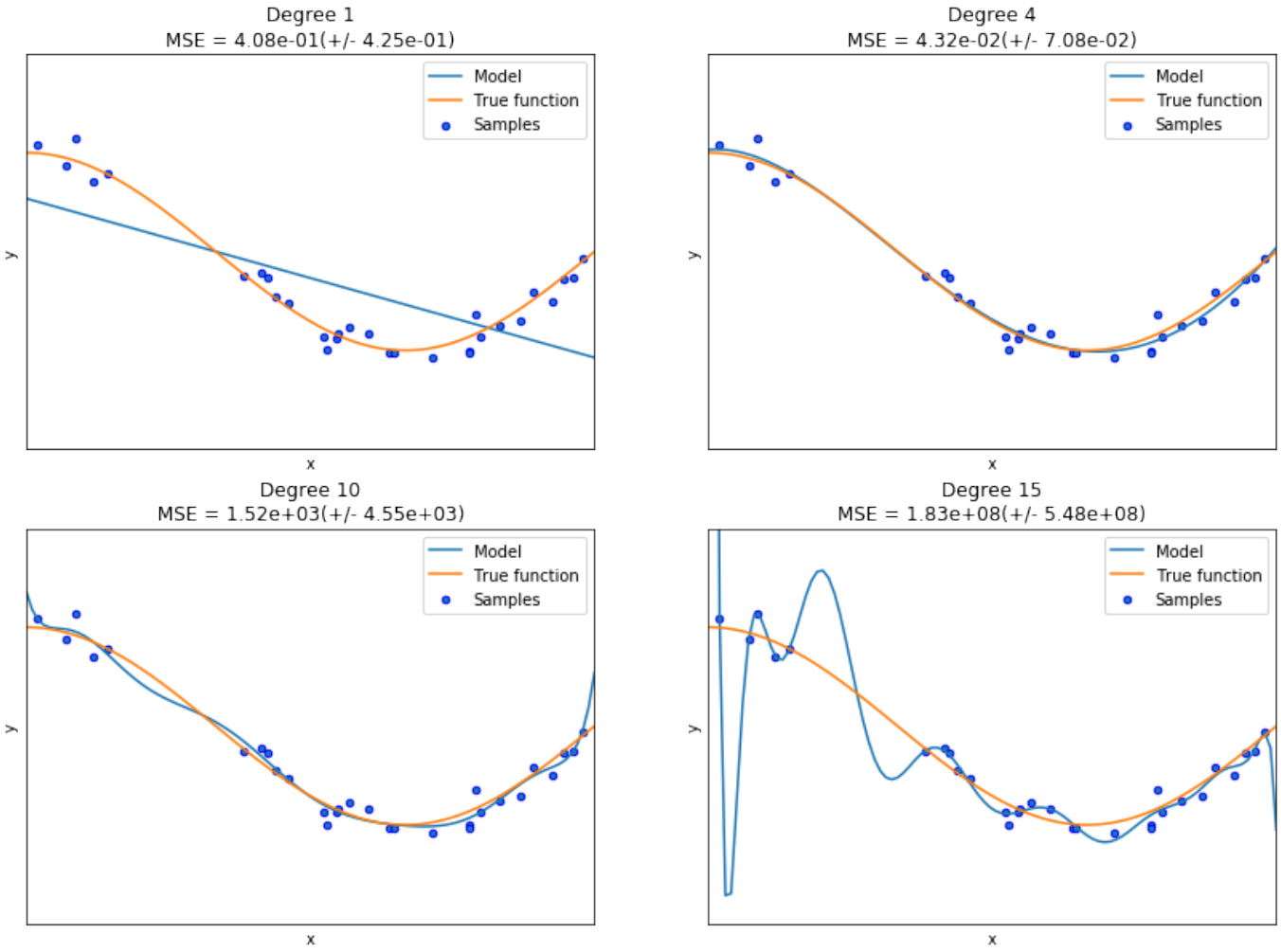
	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Bias-Variance Compromise



- We saw the dangers of overfitting in the very simple polynomial fitting example.
- To treat this, we need to make a compromise between bias and variance, i.e. forgo some precision in order to attain better predictive performance.

  $\Rightarrow$ this is a fundamental point underlying **ALL** machine learning methods



order M	MSE
1	0.408
4	0.043
10	1.52e03
15	1.83e08

Over- and Underfitting

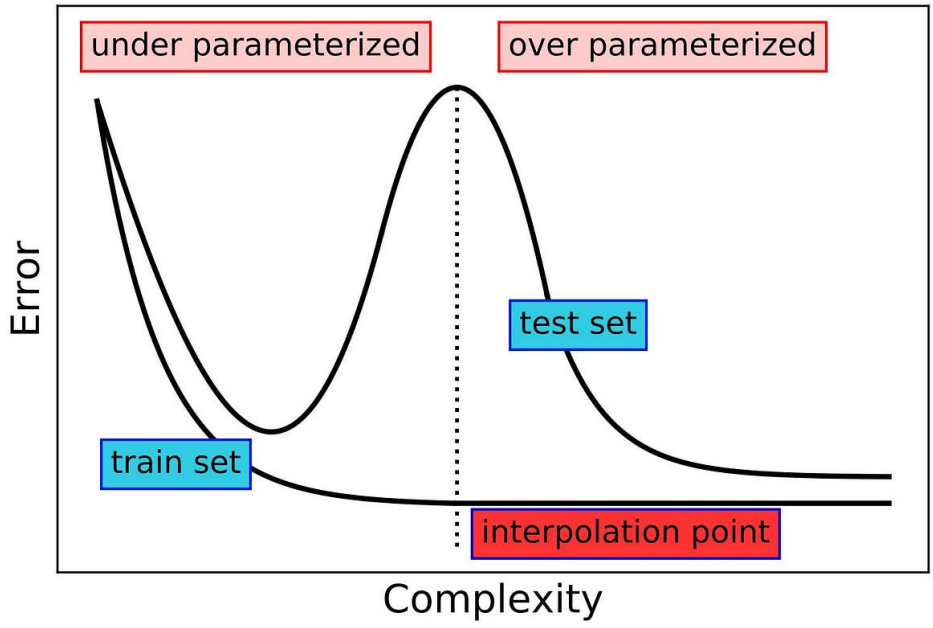
- ML Algorithm properties:
  - ⇒ Examples of low-bias machine learning algorithms: Decision Trees, k-Nearest Neighbors and Support Vector Machines.
  - ⇒ Examples of high-bias machine learning algorithms: Linear Regression, Linear Discriminant Analysis and Logistic Regression.
  - ⇒ Examples of low-variance machine learning algorithms: Linear Regression, Linear Discriminant Analysis and Logistic Regression.
  - ⇒ Examples of high-variance machine learning algorithms: Decision Trees, k-Nearest Neighbors and Support Vector Machines.

- Dealing with the overfitting problem:
  - ⇒ always be prepared to "sacrifice" some bias, to get a lower variance, since it is the variance that is the most "dangerous" in terms of predictive power/performance.

- ⇒ in general, prefer lower-order ML methods—so-called linear methods
- ⇒ increase the number of data points—this is not always feasible in reality
- ⇒ use a Bayesian approach—probably the best, but more complex to implement
- ⇒ regularize the problem—penalize terms of large magnitude, but requires extrensive tuning of the regularization parameters that usually relies on cross-validation techniques.

BV Tradeoff: Double Descent phenomenon

- Test error can present a "double descent" phenomenon in a range of ML models.
- As the number of model parameters grows relative to the number of data points, test error drops in the highly overparameterized (data undersampled) regime



- Interpretation: [OpenAI]

  $\Rightarrow$ For models at the interpolation threshold, there is effectively only one model that fits the train data, and forcing it to fit even slightly noisy or misspecified labels will destroy its global structure.

  $\Rightarrow$ There are no "good models" which both interpolate the train set and perform well on the test set.

  $\Rightarrow$ However, in the over-parameterized regime, there are many models that fit the train set and there exist such good models.

  $\Rightarrow$ Moreover, the implicit bias of stochastic gradient descent (SGD) leads it to such good models, for reasons we do not yet understand...

  $\Rightarrow$ Warning: this is a topic of ongoing research!

Recall: Precision-Recall Compromise

- What happens with binary classification problems?

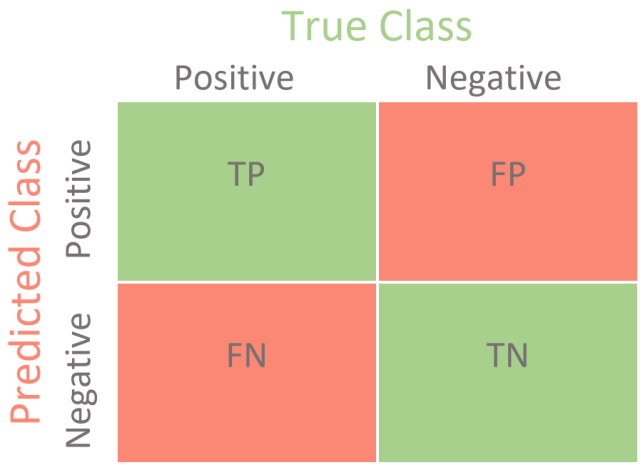
  $\Rightarrow$ the famous "truth/confusion table":

  $\rightarrow$ True positives (TP): positive COVID test, correctly diagnosed.

  $\rightarrow$ False positives (FP): positive COVID test, falsely diagnosed.

  $\rightarrow$ True negatives (TN): negative COVID test, correctly diagnosed.

  $\rightarrow$ False negatives (FN): negative COVID test, falsely diagnosed.



Definition 3. A *confusion table/matrix* , C, for a classification with n classes is an $(n \times n)$ matrix with entries

$$C_{ij} = \text{the number of observations in class } i \text{ that are predicted in class } j.$$

Then,

- C_{ii}, $i = 1, \ldots, n$, are the good classifications,
- C_{ij} with $i \neq j$ are the bad classifications.
- Now we can define precision and recall/sensitivity.

  Precision the number of true positives divided by the total number of true positives and false positives:

$$\text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}} = 1 - \text{FDR}$$

  Appropriate when minimizing false positives is the focus—FP is more serious, think of a false postive cancer diagnosis, and the heavy, unnecessary treatment that follows.

Recall the number of correct positive predictions made out of all positive predictions that could have been made:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR}$$

Appropriate when minimizing false negatives is the focus—FN is more serious, think of missing a positive COVID, who then goes on to infect many people.

Specificity is the number of true negatives divided by the total number of true negatives and false positives:

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{FPR}$$

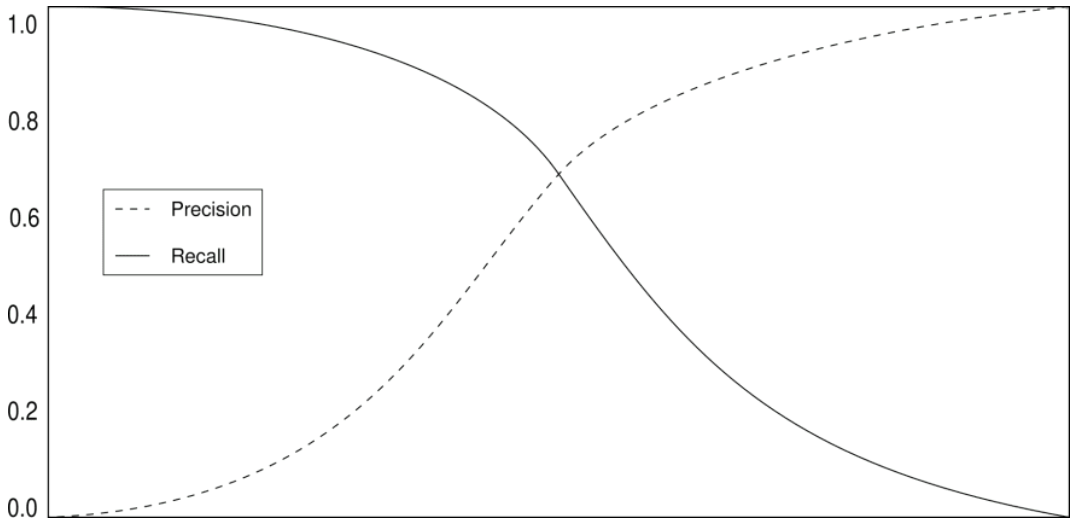
F1-Score is the harmonic mean of precision and recall:

$$\mathrm{F}_1 = 2 \times \frac{\mathrm{PPV} \times \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

Alone, neither precision or recall tells the whole story. We can have excellent precision with terrible

recall, or alternately, terrible precision with excellent recall. F-measure provides a way to express both concerns with a single score.

Precision-Recall Compromise II



- The compromise:

  ⇒ When precision increases, recall decreases, and vice-versa.

  ⇒ We cannot simultaneously increase both precision and recall.

- To balance precision and recall, a number of techniques can be used, such as

  ⇒ adjusting the decision threshold or

$\Rightarrow$ using an ensemble of models.

- Another approach is to use a metric that combines both precision and recall,

  $\Rightarrow$ such as the F1 score, which is the harmonic mean of precision and recall.

- Ultimately, the choice between precision and recall depends on the specific goals/context of the application.

  $\Rightarrow$ In some cases, such as detecting fraudulent transactions, precision may be more important than recall, as false positives can have significant consequences.
  $\Rightarrow$ In other cases, such as detecting rare medical conditions, recall may be more important, as missing a true positive can have serious consequences.

Compromises: Conclusions

1. There is always the need to compromise, or tradeoff:
   (a) Bias-Variance
   (b) Precison-Recall

2. The compromise we choose will always be subjective, and require judgement in context. There is no single, well-defined, "right" answer.

3. This will necessarily introduce an ethical/cognitive bias. See Ethics lectures.

MSE in Practice

See Overview - general

- This is the oldest (Gauss, Legendre, in the 1800's...) error (or loss) metric.
  - $\Rightarrow$ MSE, or RMSE, is adapted to optimization errors (is this the best possible fit?), and should be used exclusively for this.
  - $\Rightarrow$ MSE, or RMSE, is not adapted to evaluation of a ML model (how well is it doing?), since it is highly sensitive to outliers.

- For model performance evaluation, one should prefer the Median Absolute Deviation (MAD), or the Mean Absolute Error (MAE), though the former is more robust to outliers. In other words, the choice of metric depends on the characteristics of your data.

- To avoid orders of magnitude in errors, due to measurement units, and to obtain relative measures, the MAD and MAE can be recalculated as percentages.

R^2 in Practice

- This very widely-used regression metric has a number of shortcomings, and some people say that it should not be used at all, because:

  $\Rightarrow$ R^2 relies on data leakage
  $\Rightarrow$ R^2 decreases with increasing noise level
  $\Rightarrow$ R^2 increases with increasing model complexity

- One can use R^2 for the following instances:

  $\Rightarrow$ determine whether a change in explanatory variables improves the model fit
  $\Rightarrow$ compare models with different subsets of explanatory variables
  $\Rightarrow$ detecting co-linearity[4]

- Possible solutions

[4] High R-squared alongside insignificant coefficients can indicate multicollinearity, where independent variables are highly correlated and inflate the standard errors of the regression coefficients.

$\Rightarrow$ do not use it,

$\Rightarrow$ use an out-of-sample version where, for test performance evaluation, the TSS is calculated over the training samples only,

$$\mathrm{TSS}_{\mathrm{oos}} = \sum_{i=1}^{n} (y_i - \bar{y}_{\mathrm{train}})^2$$

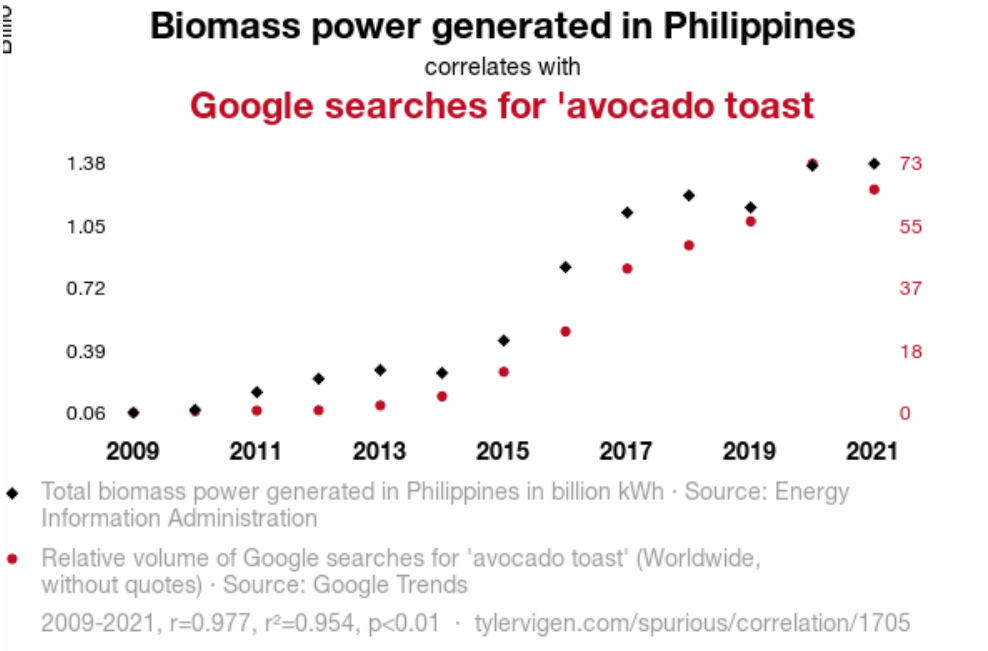
Classification Metrics

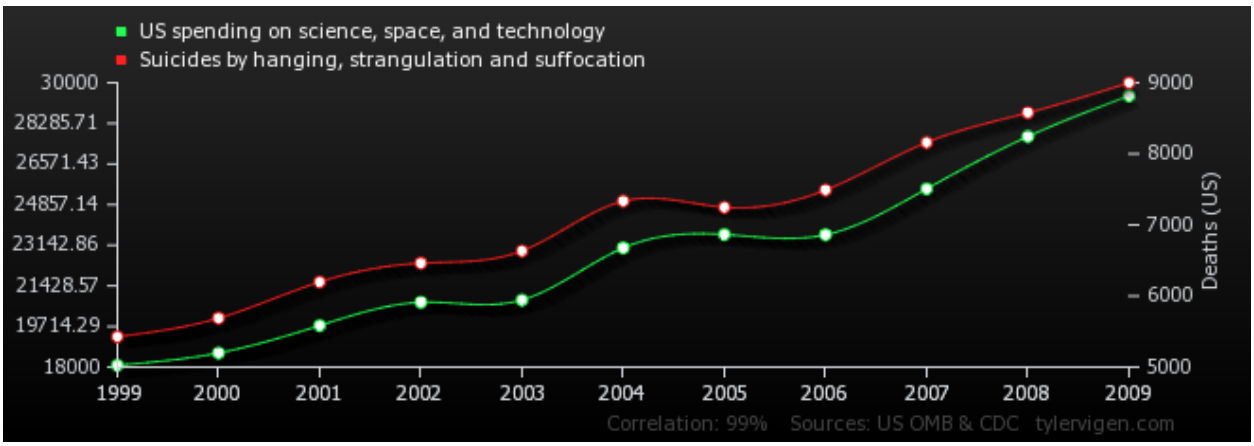
- There are a number of potential pitfalls in measuring the performance of classification models.
- The scikit tutorial is an excellent basis for illustrating several of these problems.
- See also: Overview - general.

CAUSALITY
and
CORRELATION

Principles

- Correlation can be misleading.





Warning

Correlation does not always imply causation.

- Causality can be difficult to evaluate.

  $\Rightarrow$ Confounder analysis.
  $\Rightarrow$ Partial correlation.
  $\Rightarrow$ Shapley values.

FEATURES

Features

- Feature selection: see model selection.
- Feature engineering: transformed values can be more meaningful.
- Feature importance: use ranking provided by ensemble-based methods.

  ⇒ The scikit tutorial is an excellent basis for illustrating these aspects.

  ⇒ Shapley values are powerful indicators of feature importance.

Shapley Values

Definition

Shapley values quantify the significance of individual input features' contribution to the model's predicted values.

- Analysis of explainability through SHAP regression values aims to evaluate the contribution of input variables (often called "input features") to the predictions made by a machine learning predictor model.

- The contribution of that input feature to a prediction is calculated mathematically through the construction of a so-called "explanation model".

- The explanation model evaluates the predictions of a machine learning system as the sum of the contributions of each input feature and the mean predicted value.

- Mathematically, the explanation model can be stated as

$$y = \bar{y} + \sum_i \phi_i,$$

where

⇒ y is an individual prediction,
⇒ $\bar{y}$ is the average predicted value across all predictions
⇒ ϕ_i is the contribution of input feature i to the prediction (also known as the "SHAP regression value" or "SHAP value").

- Input variables with larger magnitude SHAP values are interpreted as contributing more to a specific prediction than those with a smaller magnitude SHAP values.

- For a given case,

⇒ positive SHAP values indicate a specific feature contributes toward increasing the final predicted value y, and
⇒ negative SHAP values indicate a contribution toward decreasing the prediction.

- These SHAP values, ϕ_i, are calculated following a game theoretic approach to assess prediction contributions.

- The SHAP framework has several key desirable properties:

  $\Rightarrow$ The sum of the contributions accurately reproduces the predicted value.

  $\Rightarrow$ The contributions of input features that are not present in the machine learning model are assigned values of 0.

  $\Rightarrow$ SHAP is a machine learning model-agnostic technique and can be applied to a wide class of prediction methods.

- Despite these key advantages, there are several potential deficiencies to the application of SHAP analysis to error characterization in the Earth Sciences.

  $\Rightarrow$ SHAP analysis cannot directly yield causal insights.

  $\Rightarrow$ Direct calculation of SHAP values is very computationally expensive.

  $\rightarrow$ As such, care must be taken to properly interpret the SHAP values resulting from any particular analysis.

→ It is important to note that to improve computational performance, common implementations of SHAP analysis are available in existing machine learning libraries.

Warnings on SHAP for Causality

- Flexible predictive models like XGBoost or LightGBM are powerful tools for solving prediction problems. However, they are not inherently causal models, so interpreting them with SHAP will fail to accurately answer causal questions in many common situations.

- Unless features in a model are the result of experimental variation, applying SHAP to predictive models without considering confounding is generally not an appropriate tool to measure causal impacts used to inform policy.

- SHAP and other interpretability tools can be useful for causal inference, and SHAP is integrated into many causal inference packages, but those use cases are explicitly causal in nature.

- To that end, using the same data we would collect for prediction problems and using causal inference methods like double ML that are particularly designed to return

causal effects is often a good approach for informing policy.

- In other situations, only an experiment or other source of randomization can really answer what if questions. Causal inference always requires us to make important assumptions.

- The main point is that the assumptions we make by interpreting a normal predictive model as causal, are often unrealistic.

PINN

Physics Informed Neural Networks

See: Basics of NNs, PINNs

Remark 4. This approach, though seemingly very elegant and powerful, has serious problems of cost, robustness and precision. It often requires a lot of tuning to work. PINNs should be employed with extreme attention and great caution.

ETHICS

Moral, Ethics and Bias

See: Ethics

References

1. Please consult the list provided on the CMU-IMU-2023 website:
   CODE REFERENCES

2. Scikit-learn MOOC website

3. VSCODIUM for code editing Site, Downloads

4. G. James, D. Witten, T. Hastie, R. Tibshirani, J. Taylor. *An Introduction to Statistical Learning*. Springer. 2023.
   ISL Site