Bias and Variance

Consider the case where the target function $f : [-1, 1] \Rightarrow \mathbb{R}$ is given by $f(x) = \sin(\pi x)$ and the input probability distribution is uniform on [-1, 1]. Assume that the training set has only two examples (picked independently), and that the learning algorithm produces the hypothesis that minimizes the mean squared error on the examples.

```
(* Clear globals *)
Clear[Generate𝒳, DoLinearRegressionExperiment, Ein,
 GetYInterceptForm, LinearTarget, NoFeature, RiskNoiseFlip]
Clear[DoInteractiveLR, DoAutoLR]
```

```
(* Generate our 𝒳=[-1,1]×[-1,1] space with d uniformly distributed points
   (input to f) and a random 𝒳-partitioning solution function f. *)

Generate𝒳[OptionsPattern[]] :=
 Module[{
   𝒟 , f
  },
  𝒟 = RandomReal[{-1, 1}, OptionValue[𝒟Size]];
  f = OptionValue[Targetf] /@ 𝒟;  (*(Tanh[π#])&/@𝒟;*)
  {𝒟, f}
 ]

Options[Generate𝒳] = {𝒟Size → 2, Targetf → ((Sin[π #]) &)};

RegressionTarget[f_, 𝕏_] := f;(*Sign[𝕏.f];*)
NoFeature[𝕏_] := 𝕏;
RiskNoiseFlip[percent_] := If[RandomReal[] ≤ percent/100, -1, 1];
DoLinearRegressionExperiment[𝒳_, OptionsPattern[]] :=
 Module[{
   𝒟, f, 𝕪,
   𝕏, 𝕏f, 𝕏fdag,
   𝕨
  },
  {𝒟, f} = 𝒳;
  𝕏 = ({0, #}) & /@ 𝒟;  (*Function[x,Prepend[x,(*1*)0]]/@𝒟;*)
  𝕏f = OptionValue[DataFeature][𝕏];
  𝕪 = RiskNoiseFlip[OptionValue[Noise]] * OptionValue[TargetFunction][f, 𝕏];
  𝕏fdag = PseudoInverse[𝕏f];
  𝕨 = 𝕏fdag.𝕪;
  {𝕨, 𝕏, 𝕪, 𝒟, f}
 ]
Options[DoLinearRegressionExperiment] =
  {TargetFunction → RegressionTarget, DataFeature → NoFeature, Noise → 0};

RegressionsEin[𝕏_, 𝕨_, 𝕪_] :=
 Module[{

   N
  },
```

```
    N = Length[𝕪];
    1/N Norm[Sign[X.𝕨] - 𝕪]^2
   ]

 (*
  AltEin[𝕏_,𝕨_,𝕪_]:=
    Module[{
      N
     },
     N=Length[𝕪];
     1/N ∑_{n=1}^N (Transpose[𝕨].𝕏[[n]]-𝕪[[n]])^2
     ]
 *)

GetYInterceptForm[𝕨_] :=
  Module[{
     a, b, c, M, B
    },
    a = 𝕨[[2]];
    b = 𝕨[[3]];
    c = 𝕨[[1]];
    M = -a/b; (*If[b≠0,-a/b,10000];*)
    B = -c/b; (*If[b≠0,-c/b,10000];*)
    {M, B}
   ]

 (* Auto mode implements what the problem asks for:
    it runs the tests and then outputs a fomatted string bearing the results. *)
DoAutoLR[OptionsPattern[]] :=
  Module[{
     runs = 10 000, N = 100,
     𝕘, 𝕘s,
     𝕪, 𝕪s,
     𝕏, 𝕏s,
     𝑓, 𝑓s,
     𝒟,
     avg𝕘,
     (*avEin,*)
     i, n
    },
    𝑓s = {};
    𝕘s = {};
    𝕪s = {};
    𝕏s = {};
    For[i = 1, i ≤ runs, i++, (
      {𝕘, 𝕏, 𝕪, 𝒟, 𝑓} = DoLinearRegressionExperiment[
        Generate𝒳[𝒟Size → 2(*,Target𝑓→((Tanh[π#])&)*)],
```

```
      DataFeature -> OptionValue[DataFeature](*,OptionValue[DataFeature]*)];
    AppendTo[fs, f];
    AppendTo[gs, g];
    AppendTo[ys, y];
    AppendTo[Xs, X]
   )];
  avg = {Mean[gs[[All, 1]]], Mean[gs[[All, 2]]]};
  (*avEin=
    Mean[MapThread[Function[{x,y,z},ClassificationEin[x,y,z]],{Xs,gs,ys}]];*)
  n = Length[D];
  {fs, gs, D, avg, runs, n}
 ]
Options[DoAutoLR] = {DataFeature → NoFeature};
```

Finding the expected value

Assume the learning model consists of all hypotheses of the form $h(x) = ax$, find the expected value $\overline{g}(x)$ of the hypothesis produced by the learning algorithm (expected value with respect to the data set).

```
(* Clear globals *)
Clear[Experiment1]
Clear[e1avg, e1fs, e1gs, e1D, e1runs, e1n]
```

```
Experiment1[interactive_] := If[interactive,
  DoAutoLR[DataFeature → NoFeature], DoAutoLR[DataFeature → NoFeature]]
{e1fs, e1gs, e1D, e1avg, e1runs, e1n} = Experiment1[False];
(*Experiment1[True]*)
StringForm[
 "Linear regression results\n(runs=``, N=``):\naverage g offset=``\naverage
   g coefficient=``", e1runs, e1n, e1avg[[1]] × 1., e1avg[[2]] × 1.]
```

```
Linear regression results
(runs=10000, N=2):
average g offset=6.044187200081877`*^-19
average g coefficient=1.4468213610288414`
```

Finding the bias

```
(* Clear globals *)
Clear[Experiment2]
Clear[bias]
```

```
Experiment2[gbar_] :=
 Module[{bias},
  bias = 1/2 ∫_{-1}^{1} (gbar[[1]] + gbar[[2]] * x - Sin[π x])^2 dx
  (*bias:=1/2 Total[MapThread[(gbar*#1-#2)^2&,{xs,fs}]]*)
 ]
bias = Experiment2[e1avg];
StringForm["Linear regression results\nbias=``", bias × 1.]
```

```
Linear regression results
bias=0.2766889313950476`
```

Finding the variance

```
(* Clear globals *)
Clear[Experiment3]
Clear[variance]
```

```
Experiment3[gbar_, gs_] :=
 Module[{variance},
  variance = (1/Length[gs]
     (Sum_{i=1}^{Length[gs]} (1/2 ∫_{-1}^{1} ((gs[[i, 1]] + gs[[i, 2]] * x) - (gbar[[1]] + gbar[[2]] * x))^2 dx)))
  (*bias:=1/2 Total[MapThread[(gbar*#1-#2)^2&,{xs,fs}]]*)
 ]
variance = Experiment3[e1avg, e1gs];
StringForm["Linear regression results\nvariance=``", variance × 1.]
```

```
Linear regression results
variance=0.2392776493640879`
```

Check expected value of E_{out} for different learning models

Now change $\mathcal{H}$. Check calculated expected value for out-of-sample error for several learning models

```
(* Clear globals *)
Clear[Experiment4]
Clear[e4avg, e4fs, e4gs, e4𝒟, e4runs, e4n, e4EdEoutGd]
```

```
FeatureA[𝕏_] := {1, 0} & @@@ 𝕏;
FeatureB[𝕏_] := {0, #2} & @@@ 𝕏;
FeatureC[𝕏_] := {1, #2} & @@@ 𝕏;
FeatureD[𝕏_] := {0, #2^2} & @@@ 𝕏;
FeatureE[𝕏_] := {1, #2^2} & @@@ 𝕏;

{e4𝒻s, e4𝕘s, e4𝒟, e4avg, e4runs, e4n} = DoAutoLR[DataFeature → FeatureA];
bias = Experiment2[e4avg];
variance = Experiment3[e4avg, e4𝕘s];
e4EdEoutGd = bias + variance;
StringForm["Linear regression results h(x)= b\n(runs=``,
   N=``):\naverage 𝕘 offset=``\naverage 𝕘 coefficient=``\n𝔼𝒟[Eout(g^𝒟)]=``",
 e4runs, e4n, e4avg[[1]] × 1., e4avg[[2]] × 1., e4EdEoutGd × 1.]

{e4𝒻s, e4𝕘s, e4𝒟, e4avg, e4runs, e4n} = DoAutoLR[DataFeature → FeatureB];
bias = Experiment2[e4avg];
variance = Experiment3[e4avg, e4𝕘s];
e4EdEoutGd = bias + variance;
StringForm["Linear regression results h(x)= ax\n(runs=``,
   N=``):\naverage 𝕘 offset=``\naverage 𝕘 coefficient=``\n𝔼𝒟[Eout(g^𝒟)]=``",
 e4runs, e4n, e4avg[[1]] × 1., e4avg[[2]] × 1., e4EdEoutGd × 1.]

{e4𝒻s, e4𝕘s, e4𝒟, e4avg, e4runs, e4n} = DoAutoLR[DataFeature → FeatureC];
bias = Experiment2[e4avg];
variance = Experiment3[e4avg, e4𝕘s];
e4EdEoutGd = bias + variance;
StringForm["Linear regression results h(x)= ax + b\n(runs=``,
   N=``):\naverage 𝕘 offset=``\naverage 𝕘 coefficient=``\n𝔼𝒟[Eout(g^𝒟)]=``",
 e4runs, e4n, e4avg[[1]] × 1., e4avg[[2]] × 1., e4EdEoutGd × 1.]

{e4𝒻s, e4𝕘s, e4𝒟, e4avg, e4runs, e4n} = DoAutoLR[DataFeature → FeatureD];
bias = Experiment2[e4avg];
variance = Experiment3[e4avg, e4𝕘s];
e4EdEoutGd = bias + variance;
StringForm["Linear regression results h(x)= ax^2\n(runs=``,
   N=``):\naverage 𝕘 offset=``\naverage 𝕘 coefficient=``\n𝔼𝒟[Eout(g^𝒟)]=``",
 e4runs, e4n, e4avg[[1]] × 1., e4avg[[2]] × 1., e4EdEoutGd × 1.]

{e4𝒻s, e4𝕘s, e4𝒟, e4avg, e4runs, e4n} = DoAutoLR[DataFeature → FeatureE];
bias = Experiment2[e4avg];
variance = Experiment3[e4avg, e4𝕘s];
e4EdEoutGd = bias + variance;
StringForm["Linear regression results h(x)= ax^2 + b\n(runs=``,
   N=``):\naverage 𝕘 offset=``\naverage 𝕘 coefficient=``\n𝔼𝒟[Eout(g^𝒟)]=``",
 e4runs, e4n, e4avg[[1]] × 1., e4avg[[2]] × 1., e4EdEoutGd × 1.]
```

Linear regression results h(x)= b
(runs=10000, N=2):
average $\mathbb{g}$ offset=0.010487445035393929`
average $\mathbb{g}$ coefficient=0.`
$\mathbb{E}_{\mathcal{D}}[E_{out}(g^{\mathcal{D}})]$=0.7542300448857159`

Linear regression results h(x)= ax
(runs=10000, N=2):
average $\mathbb{g}$ offset=-1.46478×10^{-19}
average $\mathbb{g}$ coefficient=1.4279040025903553`
$\mathbb{E}_{\mathcal{D}}[E_{out}(g^{\mathcal{D}})]$=0.5079414245539238`

Linear regression results h(x)= ax + b
(runs=10000, N=2):
average $\mathbb{g}$ offset=0.0028060964029559825`
average $\mathbb{g}$ coefficient=0.7802982730255525`
$\mathbb{E}_{\mathcal{D}}[E_{out}(g^{\mathcal{D}})]$=1.8500665718437155`

Linear regression results h(x)= ax^2
(runs=10000, N=2):
average $\mathbb{g}$ offset=1.5827938841056742`*^-18
average $\mathbb{g}$ coefficient=-0.0814262
$\mathbb{E}_{\mathcal{D}}[E_{out}(g^{\mathcal{D}})]$=16.4040487923253`

Linear regression results h(x)= ax^2 + b
(runs=10000, N=2):
average $\mathbb{g}$ offset=0.4649135588013267`
average $\mathbb{g}$ coefficient=-0.553618
$\mathbb{E}_{\mathcal{D}}[E_{out}(g^{\mathcal{D}})]$=6584.2916889760345`