Performance Evaluation project

[1] http://andrewgelman.com/2013/03/14/everyones-trading-bias-for-variance-at-some-point-its-just-done-at-different-places-in-the-analyses/

Problem 1

- Samples x_1 to x_n, y_1 to y_n

$$\mathrm{MSE} = \frac{\sum_{i=1}^{n} (y_i - \beta x_i)^2}{n}$$

- Objective

$$\min_{\beta} \frac{\sum_{i=1}^{n} (y_i - \beta x_i)^2}{n}$$

- Optimal solution

$$\beta^* = \frac{\sum_{i=1}^{n} (x_i y_i)}{\sum_{i=1}^{n} (x_i x_i)}$$

Problem 1

- Since y_i = x_i^{0.75}, we have

$$\beta^* = \frac{\sum_{i=1}^{n}\left(x_i^{1.75}\right)}{\sum_{i=1}^{n}\left(x_i^{2}\right)}$$

$$E(x^2) = \lim_{n\to\infty}\frac{\sum_{i=1}^{n} x_i^2}{n} = \int_0^1 t^2 \mathrm{d}t = \frac{1}{3}$$

$$E(x^{1.75}) = \lim_{n\to\infty}\frac{\sum_{i=1}^{n} x_i^{1.75}}{n} = \int_0^1 t^{1.75} \mathrm{d}t = \frac{1}{2.75}$$

- Therefore, when n->inf

$$\beta^* = \frac{\frac{\sum_{i=1}^{n}\left(x_i^{1.75}\right)}{n}}{\frac{\sum_{i=1}^{n}\left(x_i^{2}\right)}{n}} = \frac{E(x^{1.75})}{E(x^2)} = \frac{12}{11}$$

Problem 1

- Method 2
- when n ->inf, MSE converges to

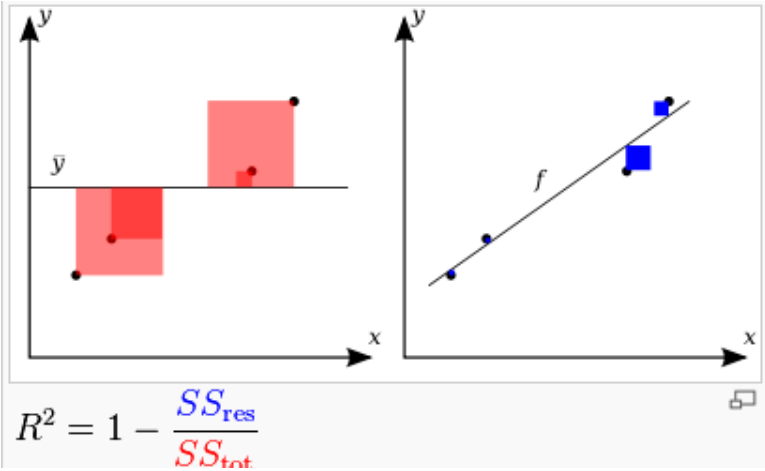
$$\int_0^1 (t^{0.75} - \beta t)^2 dt = \frac{2}{5} - \frac{8}{11}\beta + \frac{1}{3}\beta$$

$$\frac{d(\frac{2}{5} - \frac{8}{11}\beta + \frac{1}{3}\beta)}{d\beta} = 0 \Rightarrow \beta^* = \frac{12}{11}$$

$$\text{Asympotatic bias} = \frac{12}{11} * 0.5 - 0.5^{0.75}$$
$$= -0.0491$$

Problem 2a

- R^2:coefficient of determination
  - how well data points fit a statistical model
  - as the square of the correlation coefficient between the original and modeled data values



$$SS_{\text{tot}} = \sum (y_i - \bar{y})^2,$$

$$SS_{\text{res}} = \sum_i (y_i - f_i)^2,$$

Problem 2a

- R^2:coefficient of determination
  - one might keep adding variables to increase the R^2 value since it will never decrease as variables are added
- *Adjusted* R^2
  - increases only if the new term improves the model more than would be expected by chance

$$\bar{R}^2 = 1 - (1 - R^2)\frac{n-1}{n-p-1} = R^2 - (1 - R^2)\frac{p}{n-p-1}$$

Problem 2a

- *Akaike Information Criterion (AIC)*
  - deals with the trade-off between the goodness of fit of the model and the complexity of the model

    $$AIC = 2k - 2\ln(L)$$
  - the preferred model is the one with the minimum AIC value
  - Is a probability,1 <= y <= 1

Problem 2b

Method \ n	100	1000	10000	100000
k = 0.01	1 2 3 7	1 2 3	1 2 3	1 2 3
	1 2 3 4 6	1 2 3	1 2 3	1 2 3
	1 2 3	1 2 3	1 2 3	1 2 3
k = 0.05	1 2 3	1 2 3	1 2 3	1 2 3
	1 2 3	1 2 3	1 2 3	1 2 3
	1 2 3	1 2 3	1 2 3	1 2 3
Signif	1 2 3	1 2 3 4 7	1 2 3 4 5	1 2 3 4
	1 2 3 9 10	1 2 3	1 2 3 4	1 2 3 4 6
	1 2 3 8	1 2 3	1 2 3 4 7	1 2 3 4

Problem 2b

- Compare to Significant test
  - converge quickly
  - not so accurate in this sample
- but is still preferable
  - Not ideal in real world
  - Bias VS variance
  - Stable for small n and large n

Problem 2c

- Large n and large p datesets

Method	Parkinsons dataset($n > 5000, p > 15, cont - Y$)
$R^2(k = 0.01)$	2 5 9 10 12 13 14 15 16
$R^2(k = 0.05)$	9 10 12 13 15 16
$AIC(k = 0.01)$	NA
$AIC(k = 0.05)$	NA
$Significance$	1 2 9 10 12 13 14 15 16

Method	Parkinsons Dataset($n > 5000, p > 15, 0 - 1 - Y$)
$R^2(k = 0.01)$	2 3 4 10 12 13 14
$R^2(k = 0.05)$	2 3 4 12
$AIC(k = 0.01)$	2 3 11 12 13 15
$AIC(k = 0.05)$	2 3 12 15
$Significance$	1 2 4 6 10 12 13 14 15 16

Problem 2c

- Large n and small p datesets

Method	Protein Dataset($n > 5000, p < 10, cont - Y$)
$R^2(k = 0.01)$	1 3 4 8
$R^2(k = 0.05)$	1 3 4
$AIC(k = 0.01)$	NA
$AIC(k = 0.05)$	NA
$Significance$	1 2 3 4 5 6 7 8 9

Method	Page Block Dataset($n > 5000, p < 10, 0 - 1 - Y$)
$R^2(k = 0.01)$	1 2 4 5 6 8 9
$R^2(k = 0.05)$	1 2 4 5 6
$AIC(k = 0.01)$	1 2 3 4 5 6 8
$AIC(k = 0.05)$	1 2 4 5 6
$Significance$	1 2 3 4 5 6 7 8 9

Problem 2c

- Small n and large p datesets

Method	Auto Dataset($n < 1000, p > 15, cont - Y$)
$R^2(k = 0.01)$	4 8 10 13
$R^2(k = 0.05)$	8
$AIC(k = 0.01)$	NA
$AIC(k = 0.05)$	NA
$Significance$	1 4 5 8 10 11 12 13

Method	Auto Dataset($n < 1000, p > 15, 0 - 1 - Y$)
$R^2(k = 0.01)$	1 2 3 12 14 15
$R^2(k = 0.05)$	1 3 12
$AIC(k = 0.01)$	1 3 8
$AIC(k = 0.05)$	1 3 8
$Significance$	1 3 14 15

Problem 2c

- Small n and small p datesets

Method	Bike Sharing Dataset($n < 1000, p < 10, cont - Y$)
$R^2(k = 0.01)$	1 3 4 5
$R^2(k = 0.05)$	3 4 5
$AIC(k = 0.01)$	NA
$AIC(k = 0.05)$	NA
$Significance$	1 4 5

Method	Wine Dataset($n < 1000, p < 10, 0 - 1 - Y$)
$R^2(k = 0.01)$	1 3 4 6 8 9
$R^2(k = 0.05)$	6 9
$AIC(k = 0.01)$	1 3 4 6 9
$AIC(k = 0.05)$	1 3 4 6 9
$Significance$	1 3 4 6 8 9

Summary and General Findings

- When n is very large, significance test approach always gives us a complex model.
- Significance test approach is only useful when n is small.
- It is hard to compare adjusted R-squared and AIC.
- The larger k value is, the simpler model we will get. However, the k value cannot be too large.
- The value of n has more influence than the value of p in model selection.

Problem 2d

- New PAC Leave1out01

```
for (i in 1:number_of_tests) {
  # delete one observation from our sample
  x_subset[i] <- FALSE
  # avoid copying X again; use the "subset" parameter
  lmout <- lm(Y ~ X, subset=x_subset)
  my_x <- c(1, X[i, ])
  betas <- lmout$coef
  # this is faster than using "predict()"
  my_predict_y <- betas %*% my_x
  my_predict_y_int <- round(my_predict_y)
  if (my_predict_y_int == Y[i]) {
    correct_predictions <- correct_predictions + 1
  }
  x_subset[i] <- TRUE
}
return(correct_predictions/number_of_tests)
```

Problem 2d

- New PAC Leave1out01: Implementation details

- The “subset” argument of lm() is used to avoid unnecessary copy of input matrix
- Compute prediction manually without calling predict()
- Computational intensive; parallelized code very helpful

Problem 2d

- New PAC Leave1out01: Result comparison

- More aggressive than ar2() and aiclogit()!

PAC	Predictors
ar2()	1 2 3 6 7
aiclogit()	1 2 6 7
leave1out01()	2 6 7

Comparison of three PACs (Pima with k=0.01)

Parallelized Version

- Using “snow” package
- Test N predictor deletions in parallel

```
if( is.null(cls)) {
  # Run code in serial mode!
  for (i in 1:ncol(Xb2)) {
    # reconstruct/recombine X
    X <- Xb2[, c(if(i>1){seq(1,i-1)}, if(i+1<=ncol(Xb2)){seq(i+1,ncol(Xb2))})]
    pac[i] <- predacc(Y, X)
  }
} else {
  # Run code in parallel mode!
  require(parallel)
  clusterExport(cls, c("Xb2", "Yb2", "ar2", "aiclogit"))
  pac <- unlist(clusterApply(cls, 1:ncol(Xb2), prsm_kernel, predacc))
}
```

Parallelized Version

- Parallel speedup for ar2() and aiclogit() on a 2-core 4-thread HT CPU
  - Intel i3-2328m, 2.2GHz

Input size	Original time (s)	Parallelized time (s)	Speedup
500	1.170	2.251	0.520
1000	1.545	2.572	0.601
2000	1.951	2.677	0.729
5000	3.288	3.464	0.949
10000	5.299	4.584	1.156
20000	9.612	7.196	1.336
50000	22.804	14.617	1.560
100000	44.776	27.387	1.635