1. **Review and Fresh up**

   在 ML 中从新赋予回归一些解释:

   - Regression 的本质就是估计均值 (mean)，与单纯估计一个 population 的均值不同的是我们令/假设 population 的均值是关于 x 的函数，写成表达式即为
   - lab 中的 simulation 说明，自变量间的 correlation 越高，则其 confidence interval 越宽。

   We have already encountered with two methods to estimate the parameter of a model which are the OLS and Maximum Likelihood. For OLS, the result we get for $\hat{\beta}$ is

   $$\hat{\beta} = \arg\min_{\beta} \sum_{i}^{n} (y_i - x_i^T \beta)^2 = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

   where **arg** refers to the value of β that minimize the summation.

2. 概论

   从来不存在适用于所有 dataset 的模型，即不存在通解，具体 dataset 具体分析。一个重要的指标是 MSE

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

   MSE 只是一个数值上衡量的概念，其可以应用于 training 和 testing dataset。我们往往更应该关注 testing dataset 的各种指标。通常，我们得到的 training set MSE << testing set MSE。另外，Training set MSE 随 degree of freedom 的增加而减小 (always the case)，而 Testing set MSE 往往先减小在增大，呈 U 字型。当我们得到的 testing error 特别小时我们应当慎用模型。

   Machine learning 中并不存在定理说明 lowest training MSE 的 model 一定可以产生 lowest testing MSE。

   当我们有下列组合: very *Small Training MSE* and very *Large Testing MSE* 的时候，我们称这个模型 *Overfitted*，即过拟合。这往往发生在 less flexible model（小自由度 dof）具有小的 MSE 时（注这里的自由度指的是类似线性模型的自由度，即参数个数。dof 越大说明参数越多。这种情况下我们说自由度大小描述模型的复杂程度）。简而言之，越是 ture model 简单的模型往往越容易被过拟合，因为模型实际上并不复杂而我们却容易用大量的 training data 进行训练，从而使得模型中囊括了一写实际中并不存在的 parttern，进而测试集的 MSE 特别大。

   上述的 testing MSE 中的 $\hat{y}_i$ 也可以被看成是一种 r.v——尽管我们可能无法给其赋予概率上的意义check。因为不同的训练集会产生不同的 $\hat{f}$。通过证明我们可以得到结论

   $$R_n = \mathbb{E}\left[(y_0 - \hat{f}(x_0))^2\right] = Var(\hat{f}(x_0)) + Bias^2(\hat{f}(x_0)) + Var(\epsilon)$$

   这里的 $Var(\hat{f}(x_0))$ 的意思是，在给定的 x_0 处，当测试集的变化导致了 $\hat{f}(x)$ 的变化，从而导致的 $\hat{f}(x_0)$ 的变化的幅度。通常高自由度的模型 $Var(\hat{f}(x_0))$ 越大，而 Bias 则通

常随自由度增加而减小。上式只是指的单个点的 Expected MSE，overall 的 Expected MSE 可以通过取上式在测试集的各个点期望的均值得到，即

$$Ave\left(\mathbb{E}\left[(y_i - \hat{f}(x_i))^2\right]\right)$$

for every $x_i \in$ Test set。观察分解式可得，前两项都 ≥ 0，因此 *Expected Test MSE* 一定 $\geq Var(\epsilon)$，即大于 irreducible error 的 Var。对于 $\hat{f}(x_0)$ 来说，由于随着自由度的增加，其 Var 和 Bias 一减一增，因此最终 R_n 的增减由两者的相对变化速度决定。因此这里存在 Bias 和 Var 的 trade-off。个人理解：这句话必须在 R_n 已经被 minimized 的前提下成立。

有两种极端的情况：一是 extreme Var & zero Bias，如令 $\hat{f}$ 穿过训练集的每一个点，即没有 Bias 但有特别大 var（在更换训练集后有极大变化；二是extreme Bias & no Var，如无论什么样的 training set 我们都拟合同样的一条水平的直线。所以说拟合模型的难处在于找到一个方法使得 Var 和 Bias 都比较小。

值得指出，复杂的模型并非总是比简单的模型效果好；现实情况中，我们往往无法得知真实的模型 $\hat{f}$ 从而无法计算 Var 和 Bias，但是 trad-off 的思想仍然重要。

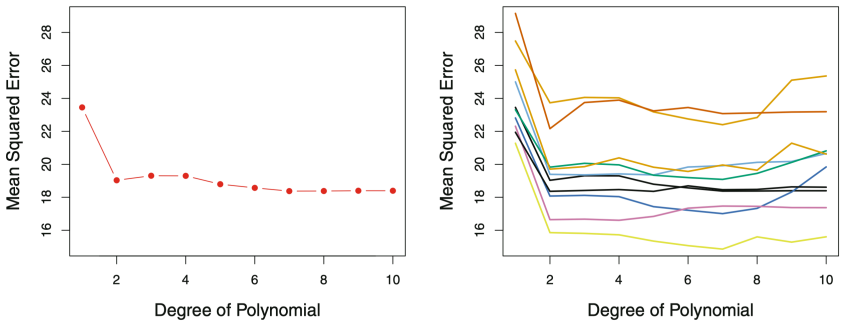
3. **Cross Validation** 交叉检验

无论何种交叉检验，其目的是为了，1. 在合理的测试集不存在的情况下通过可以的得到的训练集来估计 Testing Error (Rate)，这可以通过人为将已经得到的数据集划分为训练集和测试集达到；2. 可能关注的是能使得 test MSE 最小的位置（如多项式回归中 degree 在多大时 test MSE 最小，or the flexibility level corresponding to the minimum MSE。)3. 最后得到的 CV 值只是 variable selection process 中的其中一个选择标准，其与 AIC, $adjR^2$ 等并列。

(a) **The Validation Set Approach**

手段是将一整个集合随机划分成两个 set: 测试集和训练集。测试集用于训练模型，而测试集用来测试模型效果（i.e MSE for Numerical, Test error Rate for Qualitative）。但是这有两个缺点:

- 这样计算出的 test MSE 对于测试集、训练集特别敏感，即当测试集变化时，MSE 的 varibility 很大，如下面右图，尽管趋势相似，其相对大小差距大;



- ML 算法天然的，在 data 越多的时候表现效果越好。当天然划分了两个集合后，训练集只用了一部分数据拟合模型，因此模型往往不如用整个 dataset 训练出来的好，进而会overestimate Test error。

我们由此引入了 LOOCV 解决上述问题

(b) **Leave-One-Out Cross validation （LOOCV）**

与 (a) 中方法类似，区别是我们将只用其中一个点作为 test set，用剩余的 $(n-1)$ 个点作为 training set。我们定义某一个点 x_i 的 MSE 为

$$MSE_i = (y_i - \hat{y}_i)^2$$

然后对数据集内的每一个点都 leave-out 一次，即进行 n 次计算，然后取 n 次计算的均值 LOOCV

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} MSE_i$$

将这作为 test MSE 的 estimate。相比于（a）LOOCV 有两个好处:

- LOOCV is far less bias. LOOCV 用于训练模型的测试集中已经包含了 $(n-1)$ 个（几乎所有的）可得数据，因此模型不会倾向于 overestimate test error。总结下可以说，训练集中的数据越多，越 unbias。
- LOOCV generate no randomness due to the split of training and test set. 由于我们已经囊括了所有的分割情况，无论重复多少次，计算的结果都是一样的。

对于 Polynomial 和 least square regression，估计 test MSE 有一个 shortcut，即

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1-h_i}\right)^2$$

这里 h_i 是 leverage $\in (1/n, 1)$。值得注意，上述的第二点计算结果并非完全一样。因为并非用来拟合模型的方法都像 OLS 一样，通过设置导数等于 0 从而每一次回归结果都一样。因此得到的曲线也不完全一样。

上述的 CV 值我们称之为 ordinary Cross Validation statistics，与之相对的，还有广义的 generalized Cross Validation Statistics (GCV), 即

$$GCV = \frac{1}{n}\sum_{i=1}^{n}\frac{\hat{e}_i^2}{(1-df/n)^2} = \frac{MSE}{(1-df/n)^2}$$

where we define $df = \mathbf{tr}(H)$ which we call the effective degree of freedom。有时间可以找论文看一下

(c) **K-fold Cross Validation**

K-Fold 指将数据集随机分成数据数量大致相同的 k 组，从第一组开始作为测试集，剩余的 (k-1) 组作为训练集。重复 k 次计算，每次得到 MSE_i，最终估计的 MSE 值为

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i$$

可以看出当设 $k = n$ 时，k-fold 和 LOOCV 是完全一样的方法。相比 LOOCV，k-fold 的好处在于computational efficiency。

如果不考虑 computational efficiency，我们可以比较一下 k-fold 在 k 不同时，或者是 k-fold 与 LOOCV 的优缺点：往往我们发现 K-fold 比 LOOCV 更加准确的估计了 test error

- K-fold 比 LOOCV 的 bias 大，但是是中等程度的 bias。逻辑是 training set 中数据越多 Bias 越小，因此 K-fold 的 bias 并不是特别大
- LOOCV 的 var 要远大于 K-Fold 逻辑是 k-fold 的模型之间的相关性要比 LOOCV 要小，因为其 training set 的数据重合度要小，而相关性高的数据的 mean 的均值的 Var 要大于相关性小的均值的 Var，因此 LOOCV 的 Var 远大于 K-fold。个人理解：我们在估计 Test MSE 时候用的上述方法，是一个统计量（estimate 估计值），我们一定要将其看成是一个 r.v 的一个取值。所谓’相关性大则 Var 大‘说的是当我换了一个数据集之后，由于相关性大，不同的数据集拟合、预测出的 n 个值会在某一很近的范围之内，而与另一个数据集拟合的数据群相距很远，从而造成均值的差异很大。

4. **Subset Selection of Predictors** 模型中变量的选择

(a) **Best Subset Selection Method**

变量选择的步骤:

- Let $\mathcal{M}_0$ denote the null model containing no predictor.
- Using adj R^2 (largest), or RSS (smallest) to select the best from all of the $\binom{p}{k}$ models, denote them by $\mathcal{M}_k$, where $k \in [1, p]$, where p is the max num of candidates of predictor, so we get $p+1$ candidates in total.
- Select the best from $\mathcal{M}_0, \mathcal{M}_1, ..., \mathcal{M}_p$ based on CV prediction error, $C_p(AIC), BIC$ or adjR^2.

第二步骤基于 training set，减少了可以选择的数目 for each subset; 第三部则是基于 test set。这种方法十分不 computational efficient，当 candidates 的数目超过 40，电脑也无法快速处理。

(b) **Stepwise Selection** 分步选择

i. *Forward Selection*

M_0 中没有任何 predictor，一次加一个变量，且都会产生最大的边际进步。具体步骤如下

- Start Sith $\mathcal{M}_0$, assume now we have already k parameter in the model which is now $\mathcal{M}_k$, $k \in [0, p]$. Consider the possible $p-k$ models that could augment the model to $\mathcal{M}_{k+1}$. Chose the best based on Smallest RSS or Largest adjR^2 base on training set.
- Select form $\mathcal{M}_0$ to $\mathcal{M}_p$, based on test set using CV error, $C_p(AIC)$, BIC or $adjR^2$.

1. 注意, stepwise selection 并不保证得到的结果是 Best subset 中选择出的最好的模型。2. 即使数据个数 n<p, 仍然可以用 forward selection, 但是我们无法囊括所有的 candidates p 进入模型，最多只能 n 个（i.e $\mathcal{M}_0, \mathcal{M}_1, ..., \mathcal{M}_{p-1}$）因为每一个模型都是通过 Least Square 得到的，而点比变量少则无法得到唯一解。

ii. *Backward Selection*

与 forward 一样，但是是从全 model 出发，每次递减一个最无用的 predictor 得到一个模型，然后比较。总共 search $1+p(p+1)/2$ 个模型。模型要求 n>p 目的是为了 fit 全模型。

(c) **Hybrid Approach**

可以叫做改进版的 forward 和 backward。在增减变量的同时，也拿走最无用的变量。其保留了 computational efficiency 且囊括了 Best subset selection 的特征。

(d) **Optimal 模型的选择**

这涉及到上述模型选择的最后一步，即通过比较 test MSE

(e) **Lab: Model Selection & Establish**

5. **Ridge Regression**

与 OLS 不同，Ridge regression 通过 minmize 一个不同的 quantity 来估计参数 β，即

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j+1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2=RSS+\lambda\sum_{j=1}^{p}\beta_j^2$$

this is equvilent to

$$\frac{1}{n}\sum_{i=1}^{n}(y_i-x_i^T\beta)\ s.t.\sum_j\beta_j^2<s$$

where the second version is the minimization under constrain, and the first version is the Lagrangian version. $\lambda\geq 0$，叫做 tuning parameter，是在 fit model 前单独确定的，其用来控制和里两项对 regression 的相对影响， 第二项也被称为 *shrinkage penalty*。当 $\lambda=0$ 时，ridge regression 与 OLS 相同，而当 $\lambda\longrightarrow\infty$ 时，shrinkage penalty 上升从而 $\hat{\beta}_\lambda^R\longrightarrow 0$，这与 null model 相对应（i.e 不包含任何变量）。与 OLS 不同，岭回归会通过 CV 对每一个 λ 给出一个回归模型（具体在下面部分）。注意我们不会 shrink intercept，因此上式单独拿出了 β_0，有时我们也将 $\hat{\beta}_0$ 直接设定为 $\hat{\beta}_0=\bar{y}=\sum_{i=1}^{n}y_i/n$.

有时我们还会观察图像 $\parallel\hat{\beta}_\lambda^R\parallel_2\ /\ \parallel\hat{\beta}\parallel_2$，分母指的是 OLS 的 estimator，双杠表示 norm，即与原点 0 的距离。通常，λ 上升，$\parallel\hat{\beta}_\lambda^R\parallel_2$ 减少，从而比值减少，比值范围是 [0,1]，null model 到 OLS。与 OLS 不同的是，岭回归对于 predictor 的 scale 非常敏感，换句话说 $X_j\hat{\beta}_\lambda^R$ 取决于 λ, scale of the jth predictor and all other predictors. 因此在 fit model 之前我们先 *standardizing the predictors*（training set 中的各个 x 值）by

$$\tilde{x}_{ij}=\frac{x_{ij}}{\sqrt{1/n\sum_{i=1}^{n}(x_{ij}-\bar{x}_j)^2}}$$

denominator is the estimated s.d of jth predictor.consequently all of the standardised predictors will have a s.d of one. 啥意思？$\hat{\beta}$ 的比值 $\parallel\hat{\beta}_\lambda^R\parallel_2\ /\ \parallel\hat{\beta}\parallel_2$ 越大（越接近一）说明 model 越 flexible可以理解成岭回归比 OLS 的限制多，也因此 flexibility 更小。而

比值接近 1 说明向 OLS 靠拢，自然 flexibility 更大。

岭回归的一个好处是即使 training set 的数量 n 小于 predictors 数量 p，岭回归仍然可以进行运算而 OLS 不行。岭回归在 OLS 有 high variance 的时候拟合效果最好。其次，岭回归 has substantial computational advantages over the best subset selection

One nice thing is the ridge regression has a closed form solution which is

$$\hat{\boldsymbol{\beta}}_{\boldsymbol{\lambda}}^{\boldsymbol{R}} = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X} + \boldsymbol{\lambda}\boldsymbol{I})^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{y}$$

6. **The Lasso**

Ridge Regression 的一个缺点是他最后会将所有的变量囊括到模型中。这不会对 prediction accuracy 造成影响，但是在 p 很大的时候对模型的解释造成难度。

对岭回归改进，我们有 Lasso 回归，即 min

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j+1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

the constrained version is

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i^T\beta)\ s.t.\sum_{j}|\beta_j| < s$$

用 lasso penalty 代替 ridge penalty。Notice here the we use l_1 norm, which is $\| \beta \|_1 = \sum|\beta_j|$. Lasso 的 l_1 penalty 会 force 一些估计的系数变为 0，因此与 best subset 类似，lasso 会进行 variable selection。因此 lasso 产生的系数相对于 ridge 来说更容易 interpret。当 lasso 确定的模型只包含部分 variable 时，我们称 lasso yields sparse model。未完待续

7. **Basis Functions**

最简单的线性模型中的 *covariates* 的形式都是 x 的一次方。结合下面两种情况，我们抽象出 basis function

$$y_i = \beta_0 + \sum_{j=1}^{K}\beta_j b_j(x_i) + \epsilon_i$$

中间的 $b_j(\cdot)$ 都是关于 x 的函数，将上述模型称为 standard linear model with predictor $b_j(x_i)$。前面讨论的一切方法都适用于此。

(a) **Polynomial Regression**

多项式回归的形式为

$$y_i = \sum \beta_p x_i^p + \epsilon_i$$

这与 linear model 的拟合过程并无区别（i.e 都是关于参数线性），之前介绍的 OLS 仍然可以用。通常 d 的取值不会于 4。有时还会伴随一个 logistic 回归。此处的 x^k 可以看做是通过一个 *features expanxion function* 得到的。

(b) **Piecewise constant regression**

多项式回归会将 pattern deploy 到整个模型上。Piecewise constant 回归则是将整个 x 的 domain 划分成几段，然后用 stepwise 的 indicator function 将 x 归入划分的 subinterval 中，也就是将 x 从 numerical 变成了 categorical。这样，回归方程变成了

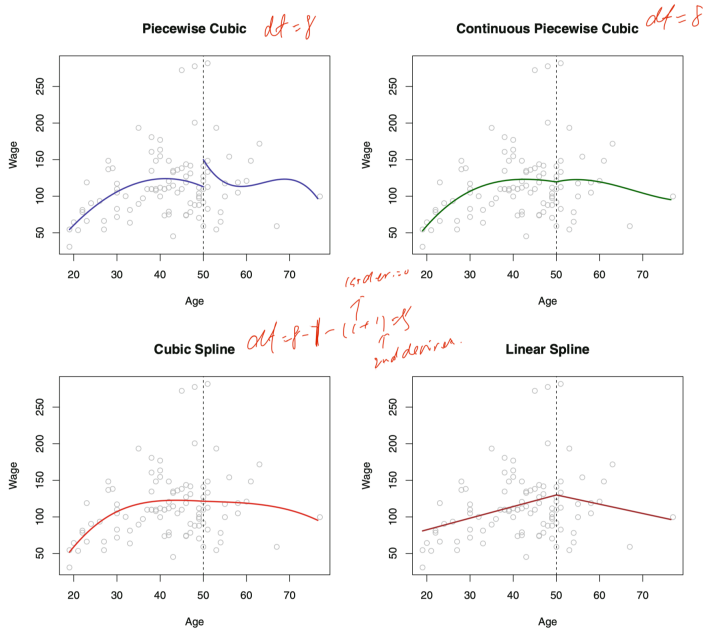
$$y_i = \sum_{j=1}^{k} \beta_j C_j(x_i) + \epsilon_i$$

如上式将 x 划分了 $(k+1)$ 段，即 $[c_j, c_{j+1})$，而 $C_j(x_i)$ 取值为 0 或 1。因此回归得到的曲线应当是一条折线，每一取值 β_j 可以解释成 represents the average increase in the response for $X \in [c_j, c_{j+1})$ comparing to $X < c_1$。Here we exclude β_0 as a predictor, the reason is the same as we only need 2 dummy variable to encode a qualitative variable with 3 levels.

(c) **Regression Spline**

基本原理及步骤是先确定一个要回归的函数，然后将 x 的 domain 分段，在不同段上回归不同的参数。分段的点称为 knots。更进一步我们可以添加限制条件，即

- 令断点处连续，$d.o.f - 1$
- 令断点处 1st derivative 和 2nd derivative 相等，$d.o.f - 2$



上图一步一步描述了限制条件的添加。第四张图特指在每一部分都通过 OLS 拟合一条直线，当然断点相连 dof 减一。

假设有 cubic spline（i.e 最高次三次），K 个 knots，则有（K+1）段 subinterval，每一 knot 减掉 3 个 dof，共有 $4(K+1) - 3K = k + 4$ 个 dof。

同样，我们可以将 spline 回归转化成一种 standard basis 的形式。未完待续 7.4.3/4/5

8. **Smoothing Splines**

9. **Decision Trees**

决策树，其可以用在 regression，也可以用在 classification 问题中。

(a) **Regression tree vis Stratification of feature space (top-down)**

步骤是将 feature space （所有可能取到的值）分解成 J 个 distinct and non-overlapping space regions。这就是我们的 model。之后进行预测，如果去测的 X 值落到了 jth region 中，则我们用该 region response variable 的 mean 作为预测值。

Feature space 的划分方法此处介绍的是 *recursive binary splitting*, 是一种 *top-down, greedy* 的算法。理论上讲，我们应该找到一种 split 使得

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

however this requires to compute every possible combination of split which is computationally infeasible.

recursive binary splitting detail:

- 对每一个 Predictor X_i 的每一个可能的 split 都进行划分并计算两个子集的 RSS（i.e 个人理解如果有 n 个 training set points 则进行 n-1 次的划分）
- 每一个 X_i 中都保留最小的 RSS 的 cutpoint 并选取该点为 cutpoint

数学表达为:

$$R_1(j,s)=\{X|X_j<s\},\quad R_2(j,s)=\{X|X_j\geq s\}$$

and seek the value j and s that minimize

$$\sum(y_i-\hat{y}_{R_1})^2+\sum(y_i-\hat{y}_{R_2})^2$$

where the $\hat{y}_{R_i})^2$ are the predicted value which in this case we use the sub-group mean response. 上述的过程一直重复直到达到某 criteria 停止（i.e have maybe 5 points left。

(b) **Regression tree via Tree pruning (down-top)**

自上而下分割 space 的方法很容易导致 overfit （i.e 在训练集表现良好但是在测试集表现不佳）。某种程度上讲，这个可以通过设定 reduction in RSS at a high theorhold 来解决，但是这会引起另一个问题，比如之前的一个 split 使得 RSS 减少较少，但是会在之后的 split 中产生较好的结果，则该方法短视。因此为得到一个 split 较少的 tree 且表现会更好的 tree 我们通过 pruning，即先得到一个大的 tree 后自下而上裁剪。

裁剪方法为 Cost complexity pruning(weakest link pruning) Not finished

10. **Bootstrap**

当我们只有一个 dataset 且无法从 population 中得到更多时，用 bootstap 来产生更多的 dataset。