lasso

lasso について

- least absolute shrinkage and selection operator (lasso) はモデルの係数を推定し、その推定はどの共変量をモデルに組み込むべきかを選択するのに使われます。
- lasso は、多くの潜在的な共変量のうちのいくつかがアウトカムに影響を与える場合に最も有用です。そして、影響を与える共変量のみをモデルに含めることが重要です。

目次

lasso によるフィッティングとモデル選択の方法

- lasso は、λ を所与とした時に次式を最小化する β の推定値を求める推定手法です。

$$\frac{1}{2N}(y - X\beta')'(y - X\beta') + \lambda \sum_{j=1}^{p} |\beta_j|$$

- 任意の λ に対して何らかの β が求まります。その時、いくつかの係数 β _j はゼロになるかもしれません。また、λ が大きくなるほど、ゼロとならない係数の数は減っていきます。
- lasso の目的は、ゼロとならない係数 β に対応する λ で、好ましい特徴を持ったものを決定することです。
- 最終的に選択した λ は λ^*と表記します。
- λ^*について触れる場合、ゼロとならない係数 β のセットを対象にしていることを覚えておいてください。

選択(cv):クロスバリデーション

- Stata のサンプルデータ auto.dta を使ってクロスバリデーション(CV)について説明します。
- ただし、このサンプルデータは変数の数もサンプルサイズも小さいので、本来は lasso による解析向きではありません。lasso は一般的に数百、数千個以上の変数を持つデータに対して利用します。
- サンプルデータは小さいデータですが、出力が小さく解説に便利なため、ここでは敢えて利用します。下記のコマンドでサンプルデータを読み込みます。

```
. sysuse auto
```

- **lasso** コマンドを実行する前に乱数キーを設定します。
- CV は乱数を利用しますので、再現性を確保します。

```
. set seed 1234
```

- **mpg**(マイル/ガロン)を被説明変数とするモデルを推定します。
- **lasso** のモデルとして、**linear**、**logit**、**probit**、**poisson** が用意されています。ここでは、線形モデルなので linear を利用します。
- CV のオプションは **selection(cv)**ですが、デフォルトで実行されますので、ここでは表記しません。

```
. lasso linear mpg i.foreign i.rep78 headroom weight turn gear_ratio price
trunk length
```

- **lasso** コマンドの実行後、推定結果を **autolasso** という名前で保存します。一般的な **lasso** コマンドの利用場面では、計算時間がかかりますので、推定結果を保存する習慣をつけておくと効率的です。

```
. estimates store autolasso
```

```
10-fold cross-validation with 100 lambdas ...
Grid value 1:     lambda =  4.69114   no. of nonzero coef. =       0
Folds: 1...5....10   CVF = 33.97832
Grid value 2:     lambda = 4.274392   no. of nonzero coef. =       2
Folds: 1...5....10   CVF = 31.62288
Grid value 3:     lambda = 3.894667   no. of nonzero coef. =       2
Folds: 1...5....10   CVF = 28.65489
Folds: 1...5....10   CVF = 13.36279
Grid value 44:    lambda = .0858825   no. of nonzero coef. =      10
Folds: 1...5....10   CVF = 13.39785
Grid value 45:    lambda = .0782529   no. of nonzero coef. =      11
Folds: 1...5....10   CVF = 13.45168
... cross-validation complete ... minimum found

Lasso linear model                          No. of obs        =         69
                                            No. of covariates =         15
Selection: Cross-validation                 No. of CV folds   =         10
```

ID	Description	lambda	No. of nonzero coef.	Out-of-sample R-squared	CV mean prediction error
1	first lambda	4.69114	0	-0.0018	33.97832
41	lambda before	.1135316	8	0.6062	13.3577
* 42	selected lambda	.1034458	8	0.6066	13.3422
43	lambda after	.0942559	9	0.6060	13.36279
45	last lambda	.0782529	11	0.6034	13.45168

* lambda selected by cross-validation.

- この計算ではグリッド 1 の λ=4.691140 からグリッド 45 の λ=0.078253 まで繰り返し計算を実行します。
- デフォルトではオプション selection(cv)が 100 個の λ を利用します。
- λ は一定間隔か、または対数スケールで作成されます。ここではグリッド値 47 で終了し、λ のグリッドポイントの最小値 55 では何の計算も実行しません。
- 出力結果の表からは、グリッド値 1 の λ に対しては、非ゼロの係数がゼロ個であることが分かります。
- この表には CV の過程で計算した λ を表示します。
- グリッド値が 100 の時の λ は、grid()のサブオプション ratio(#)で設定します。この比率は最後の最小 λ と最初の妻財 λ の比として求めます。ratio(#)のデフォルト値は 1e-4 です。
- 各 λ に対して係数を推定します。すべての λ を表示する場合は、lassoknots コマンドと alllambdas オプションを利用します。

```
. lassoknots, alllambdas
```

ID	lambda	No. of nonzero coef.	CV mean pred. error	Variables (A)dded, (R)emoved, or left (U)nchanged
1	4.69114	0	33.97832	U
2	4.274392	2	31.62288	A weight length
3	3.894667	2	28.65489	U
4	3.548676	2	26.0545	U
5	3.233421	2	23.8774	U
6	2.946173	2	22.07264	U
7	2.684443	2	20.57514	U
8	2.445964	2	19.30795	U
9	2.228672	2	18.23521	U
10	2.030683	2	17.43067	U
11	1.850282	2	16.78884	U
12	1.685908	2	16.32339	U
13	1.536137	2	15.97483	U
14	1.399671	2	15.70143	U
15	1.275328	3	15.48129	A 5.rep78
16	1.162031	3	15.34837	U
17	1.0588	3	15.30879	U
18	.9647388	3	15.30897	U
19	.8790341	4	15.3171	A turn
20	.8009431	5	15.32254	A gear_ratio
21	.7297895	6	15.31234	A price
22	.664957	6	15.28881	U
23	.6058841	6	15.26272	U
24	.552059	6	15.20981	U
25	.5030156	6	15.1442	U
26	.4583291	6	15.04271	U
27	.4176124	6	14.92838	U
28	.3805129	6	14.877	U
29	.3467091	6	14.83908	U
30	.3159085	7	14.77343	A 0.foreign
31	.287844	8	14.67034	A 3.rep78
32	.2622728	8	14.53728	U
33	.2389732	8	14.35716	U
34	.2177434	8	14.15635	U
35	.1983997	8	13.95308	U
36	.1807744	8	13.77844	U
37	.1647149	8	13.62955	U
38	.1500821	8	13.519	U
39	.1367492	8	13.43867	U
40	.1246008	8	13.39141	U
41	.1135316	8	13.3577	U
* 42	.1034458	8	13.3422	U
43	.0942559	9	13.36279	A 1.rep78
44	.0858825	10	13.39785	A headroom
45	.0782529	11	13.45168	A displacement

* lambda selected by cross-validation.

- λが小さくなるほど、非ゼロの係数が増えます。非ゼロの係数が変化すると、モデルの変数が増えます。ただし、場合によっては変数がモデルから除外される場合もあります。非ゼロの係数がより小さいλでゼロになることもあります。
- この例題では追加された変数が削除されていません。より多くの変数が存在するような場面では、変数の追加によって他の変数が削除されることもあります。
- 一般的にはλが単調に小さくなるにしたがって、非ゼロの係数の数は増える傾向にあります。稀にモデルに含まれる変数の数が少なくなるとλは小さくなることがあります。
- 変数が追加または削除されるようなλはノットと呼ばれます。デフォルトで **lassoknots** コマンドはノットだけを表示します。ノットでないλが CV 関数を最小化します。そのようなλはλ^*と表記します。

```
. lassoknots
```

ID	lambda	No. of nonzero coef.	CV mean pred. error	Variables (A)dded, (R)emoved, or left (U)nchanged
2	4.274392	2	31.62288	A weight length
15	1.275328	3	15.48129	A 5.rep78
19	.8790341	4	15.3171	A turn
20	.8009431	5	15.32254	A gear_ratio
21	.7297895	6	15.31234	A price
30	.3159085	7	14.77343	A 0.foreign
31	.287844	8	14.67034	A 3.rep78
* 42	.1034458	8	13.3422	U
43	.0942559	9	13.36279	A 1.rep78
44	.0858825	10	13.39785	A headroom
45	.0782529	11	13.45168	A displacement

* lambda selected by cross-validation.

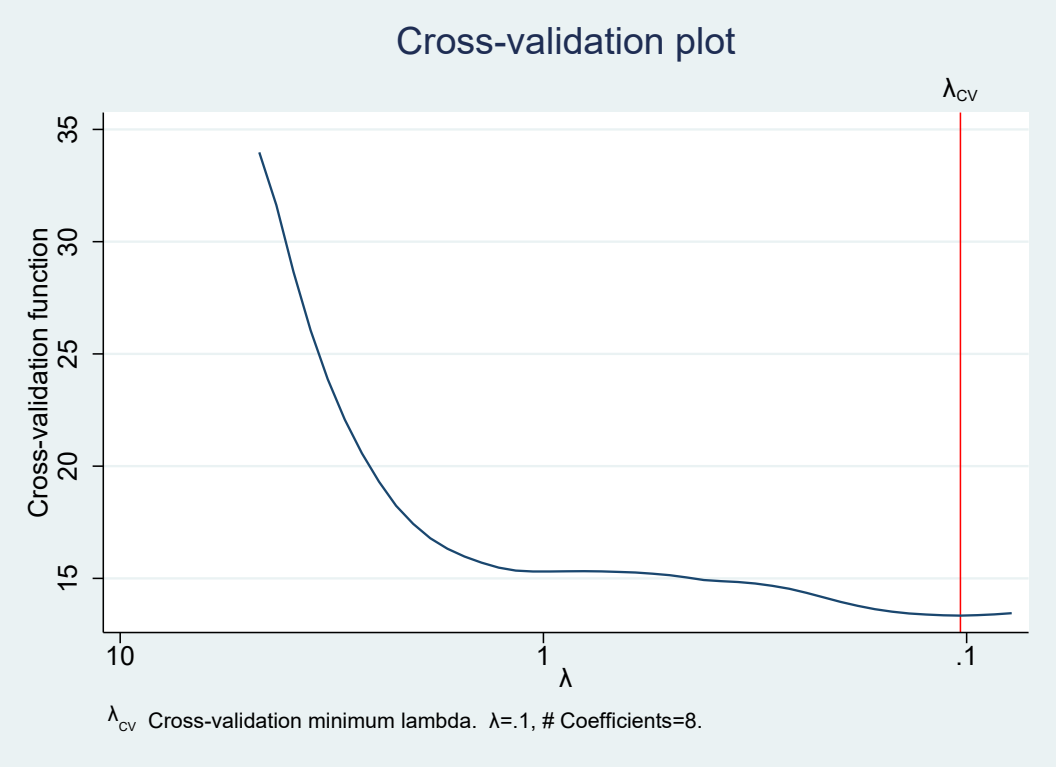
CV（クロスバリデーション）関数

- 各λに対して係数を推定したのち、CV 関数を計算します。具体的には乱数を使ってデフォルトで 10 個のフォルド(塊)にデータを分割します。
- あるフォルドを選択し、それを残りの 9 つのフォルドに回帰します。この時の新しい係数を使って選択したフォルドの予測値を求め、MSE を計算します。同じことを残りの 9 つのフォルドに対しても実行します。
- 最後にこれらの MSE の平均をとって、CV 関数の値とします。MSE の値は CV mean prediction error として表に出力します。デフォルトで **selection(cv)** は CV 関数の最小

値を検索し、計算を終了します。

- λの値として単純に CV を利用したものよりも CV の平均値が大きくなるものが 3 つ存在します。λ 自体ではなく、その CV の平均が最小となるものを λ^*と表現します。
- CV 関数をプロットする場合は cvplot コマンドを利用します。CV 関数の値についてより多くのものを調査する場合は selection(cv,alllabdas)と操作します。

```
. cvplot
```



- CV 関数の値をもっと表示するには、selection(cv, alllambdas)オプションを使って再度 lasso を実行します。

```
. set seed 1234
. lasso linear mpg i.foreign i.rep78 headroom weight turn gear_ratio price
trunk length displacement, selection(cv, alllambdas)
```

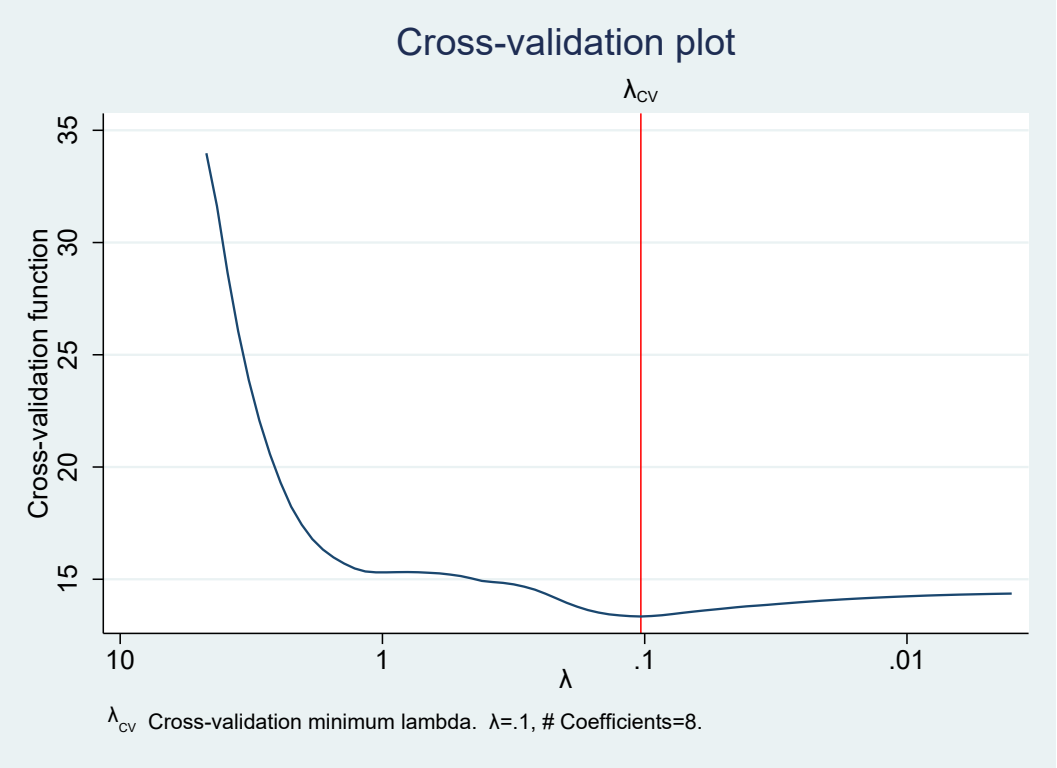
```
Evaluating up to 100 lambdas in grid ...
Grid value  1:     lambda =  4.69114   no. of nonzero coef. =       0
Grid value  2:     lambda = 4.274392   no. of nonzero coef. =       2
Grid value  3:     lambda = 3.894667   no. of nonzero coef. =       2

Grid value 75:     lambda = .0048015   no. of nonzero coef. =      13
Grid value 76:     lambda =  .004375   no. of nonzero coef. =      13
Grid value 77:     lambda = .0039863   no. of nonzero coef. =      13
... change in deviance stopping tolerance reached

10-fold cross-validation with 77 lambdas ...
Fold  1 of 10:  10....20....30....40....50....60....70...
Fold  2 of 10:  10....20....30....40....50....60....70...
Fold  3 of 10:  10....20....30....40....50....60....70...
Fold  9 of 10:  10....20....30....40....50....60....70...
Fold 10 of 10:  10....20....30....40....50....60....70...
... cross-validation complete

Lasso linear model                          No. of obs        =         69
                                            No. of covariates =         15
Selection: Cross-validation                 No. of CV folds   =         10
```

ID	Description	lambda	No. of nonzero coef.	Out-of-sample R-squared	CV mean prediction error
1	first lambda	4.69114	0	-0.0018	33.97832
41	lambda before	.1135316	8	0.6062	13.3577
* 42	selected lambda	.1034458	8	0.6066	13.3422
43	lambda after	.0942559	9	0.6060	13.36279
77	last lambda	.0039863	13	0.5765	14.36306

```
* lambda selected by cross-validation.
```

- 繰り返し計算のログの内容は、最初の `lasso` コマンド実行時とは異なっています。今回は、最初にすべてのグリッド値が表示され、次に CV のフォルドが出力されています。前回のコマンドでは、グリッド値とフォルドが交互に出力されました。
- オプション `alllambdas` を利用すると、すべての λ に対する係数ベクトルを推定し、次に CV を計算します。このように、最小値で計算を終了しない方が、やや計算処理は速くなります。
- 乱数シードを利用した場合、選択されたλ^*と CV 関数の値、そして R^2 は全く同じものになります。
- λに対して CV 関数をプロットします。

```
. cvplot
```



- `alllambdas` オプションを利用しても、実際に図のプロットに利用する λ は 100 個(グリッド)ではなく 77 個です。
- このオプションを使っても、λ の利用には他のルールが適用されます。`stop(#)`オプションの停止許容範囲が設定されます。推定値の係数が λ が変わってもほぼ変化しない場合、繰り返し計算を終了します。実質的に CV 関数が平坦になっており、より小さな λ に関して計算を実行しても意味がないと考えれることがその理由です。
- λ の最小値を調べる場合は、`stop(0)`オプションを次のように利用します。

```
. lasso linear mpg i.foreign i.rep78 headroom weight turn gear_ratio price
trunk length displacement, selection(cv, alllambdas) stop(0)
```

- λ が小さくなるほど、計算時間が長くかかります。そのため、λ の計算をできる限り早く収束させるオプションが用意されています。
- 観測値と変数が多いほど、繰り返し計算の回数は膨大なものになります。求めた最小値がグローバルな最小値であることを確認する場合は `cvtolerance(#)`を用いて、`alllambdas` よりも広い範囲を設定します。そうすると処理速度は短くなります。

係数へのペナルティと選択

- ここでは mpg の予測モデルの作成に lasso を利用する方法を説明します。
- lasso コマンドの実行後に予測値を求める場合は predict コマンドを利用します。ただし、予測値の計算にあたっては、ペナルティのかかった係数を利用する方法と、その他に、自分で選択した係数を利用する方法の 2 つがあります。
- 実際には lasso を実行すると、標準化(standardized)、罰則化(penalized)、選択(postselection)の 3 種類の係数を得ます。
- 最小化する目的関数を次に示します。

$$\frac{1}{2N}(y - X\beta')'(y - X\beta') + \lambda\sum_{j=1}^{p}|\beta_j|$$

X で示す変数を標準化し、平均 0、分散 1 とします。

$$\sum_{j=1}^{p}|\beta_j|$$

- standardized オプションは、目的関数を最小化する標準化回帰係数を推定します。
- lasso コマンドによる予測を実行する場合、係数の表現方法はあまり重要ではありませんが、標準化用の専用コマンド lassocoef を利用します。標準化した変数による係数はつぎのようになります。

```
. lassocoef, display(coef, standardized)
```

	active
0.foreign	1.352764
rep78	
3	-.2941186
5	1.25592
weight	-.3125537
turn	-.6939711
gear_ratio	1.286896
price	-.3039696
length	-2.918227
_cons	0

Legend:
b - base level
e - empty cell
o - omitted

- 標準化した係数の推定値は元の変数の係数の推定値とほぼ同じ大きさです。
- **penalize** オプションは、標準化を実行していない状態で目的関数を最小化することで係数を推定します。厳密に言うと、**standardized** オプションは標準化した変数の罰則化係数です。**penalized** オプションは非標準化変数の罰則化係数です。
- コマンドを実行して比べてみます。

```
. lassocoef, display(coef, penalized)
```

	active
0.foreign	2.939956
rep78	
3	-.5933059
5	3.430842
weight	-.0003971
turn	-.1574076
gear_ratio	2.801762
price	-.0001051
length	-.1292282
_cons	42.99934

```
Legend:
  b - base level
  e - empty cell
  o - omitted
```

- **postselection** オプションは選択した変数を利用して線形回帰を実行し、その係数を利用します。

```
. lassocoef, display(coef, postselection)
```

	active
0.foreign	4.769344
rep78	
3	-1.010493
5	4.037817
weight	-.000157
turn	-.2159788
gear_ratio	3.973684
price	-.0000582
length	-.1355416
_cons	40.79938

Legend:
  b - base level
  e - empty cell
  o - omitted

- regress コマンドを使って結果を再現します。

```
. regress mpg 0bn.foreign 3bn.rep78 5bn.rep78 weight turn gear_ratio price
length
```

Source	SS	df	MS		
				Number of obs	= 69
				F(8, 60)	= 22.14
Model	1748.04019	8	218.505024	Prob > F	= 0.0000
Residual	592.162704	60	9.86937839	R-squared	= 0.7470
				Adj R-squared	= 0.7132
Total	2340.2029	68	34.4147485	Root MSE	= 3.1416

mpg	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
foreign						
Domestic	4.769344	1.596469	2.99	0.004	1.575931	7.962757
rep78						
3	-1.010493	.8775783	-1.15	0.254	-2.765911	.7449251
5	4.037817	1.262631	3.20	0.002	1.512178	6.563455
weight	-.000157	.0021651	-0.07	0.942	-.0044878	.0041739
turn	-.2159788	.1886946	-1.14	0.257	-.5934242	.1614665
gear_ratio	3.973684	1.603916	2.48	0.016	.7653732	7.181994
price	-.0000582	.0001996	-0.29	0.772	-.0004574	.0003411
length	-.1355416	.0595304	-2.28	0.026	-.2546201	-.0164632
_cons	40.79938	9.206714	4.43	0.000	22.38321	59.21555

- p 値に注目してみましょう。有意でない係数推定値が目立ちます。lasso という手法は推定にあたって p 値は考慮しません。予測力を向上させることに重点をおいている結果が、このような p 値の生み出す原因です。
- ランダムノイズと考えられる観測値のことも考慮して優れたフィットを実現するのが CV 関数です。ランダムノイズに不要に反応してしまう状態は過剰選択 (overselecting) と呼ばれています。
- 推定結果から分かるように、**rep78** には 5 つのカテゴリがありますが、lasso はそのうち **rep78=3** と **rep78=5** だけをモデルで採用しています。

lassoselect コマンドを使って手作業で λ を選択する

- 自動選択された λ^*を変更する方法を紹介します。最初に、自動選択による lasso の結果を次のコマンドでメモリー上に保存します。

```
. estimates store name
```

- これを他の lasso コマンドの結果と比較します。ファイルとして保存する場合は次のようにします

```
. estimates save filename
```

- 保存結果を呼び出す場合は次のようにします。

```
. estimates restore autolasso
```

- **lassoknots** コマンドにオプションを使って 2 種類の **R^2** を表示します。ここでは **out-of-sample** ラベルも表示しました。

```
. lassoknots, display(cvmpe r2 osr2)
```

	ID	lambda	CV mean pred. error	Out-of-sample R-squared	In-sample R-squared
	2	4.274392	31.62288	0.0676	0.1116
	15	1.275328	15.48129	0.5435	0.6194
	19	.8790341	15.3171	0.5484	0.6567
	20	.8009431	15.32254	0.5482	0.6627
	21	.7297895	15.31234	0.5485	0.6684
	30	.3159085	14.77343	0.5644	0.7030
	31	.287844	14.67034	0.5675	0.7100
*	42	.1034458	13.3422	0.6066	0.7422
	43	.0942559	13.36279	0.6060	0.7431
	44	.0858825	13.39785	0.6050	0.7439
	45	.0782529	13.45168	0.6034	0.7449

* lambda selected by cross-validation.

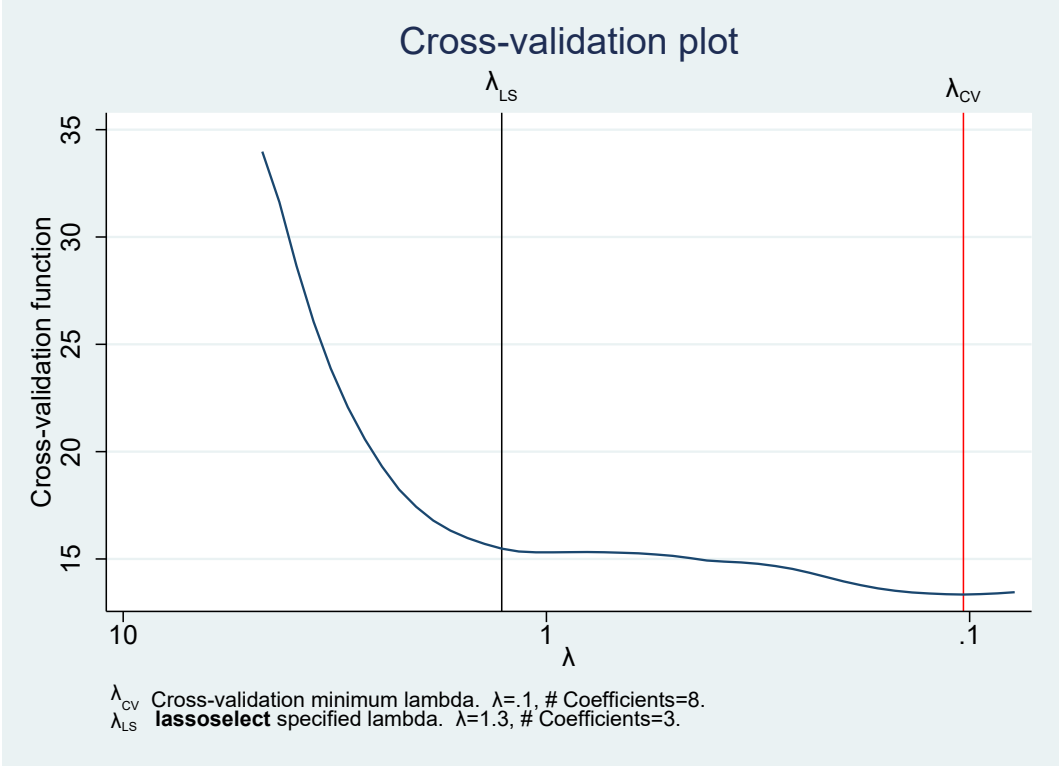
- ID=15 の λ が CV により選択されたものと同程度に良いことが分かりますので、それを使ってみましょう。

```
. lassoselect id = 15
```

```
ID = 15  lambda = 1.275328 selected
```

- 新たに選択したλ^*を使って cvplot コマンドを実行します。

```
. cvplot
```



λ_{CV} Cross-validation minimum lambda. λ=.1, # Coefficients=8.
λ_{LS} **lassoselect** specified lambda. λ=1.3, # Coefficients=3.

- 係数を表示し、先の結果と比べます。

```
. lassocoef autolasso ., display(coef, postselection)
```

	autolasso	active
0.foreign	4.769344	
rep78		
3	-1.010493	
5	4.037817	2.782347
weight	-.000157	-.0024045
turn	-.2159788	
gear_ratio	3.973684	
price	-.0000582	
length	-.1355416	-.1120782
_cons	40.79938	49.23984

```
Legend:
  b - base level
  e - empty cell
  o - omitted
```

- 先に実行した lasso コマンドの結果を autolasso に保存していました。ここでは、lassoselect コマンドを利用して新たに分析を実行し、異なる推定結果を得ました。
- ピリオド.は直近の推定結果を意味します。
- 別の分析結果との比較などに利用する場合は、estimates store コマンドで新たな名前を付けて保存しておきます。
- 名前を付けて保存した結果の内容を比較表示する時は lassocoef コマンドを利用します。