

Session 4

49-781 Data Analytics for Product Managers
Spring 2018

Section 1

Interactive Terms

Interactive Terms

So far, we have been using individual predictors as main terms.

It's possible that there is a joint effect of predictors, and that it's really the co-existence of two predictors that impacts the outcome.

An **interactive effect** between predictors is an additional change to the response that occurs at particular combination of predictors.

Interaction can be between categorical, numeric or a combination of variables.
(There could also be more than two variables)

One Categorical, One Continuous

```
"", "id", "chol", "stab.glu", "hdl", "ratio", "glyhb", "location", "age", "gender", "height", "weight", "frame", "bp.1s", "bp.1d", "bp.2s", "bp.2d", "waist", "hip", "time.ppn"  
"1", 1000, 203, 82, 56, 3.5999999, 4.30999994, "Buckingham", 46, "female", 62, 121, "medium", 118, 59, NA, NA, 29, 38, 720  
"2", 1001, 165, 97, 24, 6.9000001, 4.44000006, "Buckingham", 29, "female", 64, 218, "large", 112, 68, NA, NA, 46, 48, 360  
"3", 1002, 228, 92, 37, 6.19999981, 4.63999987, "Buckingham", 58, "female", 61, 256, "large", 190, 92, 185, 92, 49, 57, 180  
"4", 1003, 78, 93, 12, 6.5, 4.63000011, "Buckingham", 67, "male", 67, 119, "large", 110, 50, NA, NA, 33, 38, 480  
"5", 1005, 249, 90, 28, 8.89999962, 7.71999979, "Buckingham", 64, "male", 68, 183, "medium", 138, 80, NA, NA, 44, 41, 300  
"6", 1008, 248, 94, 69, 3.5999999, 4.80999994, "Buckingham", 34, "male", 71, 190, "large", 132, 86, NA, NA, 36, 42, 195  
"7", 1011, 195, 92, 41, 4.80000019, 4.84000015, "Buckingham", 30, "male", 69, 191, "medium", 161, 112, 161, 112, 46, 49, 720  
"8", 1015, 227, 75, 44, 5.19999981, 3.94000006, "Buckingham", 37, "male", 59, 170, "medium", NA, NA, NA, NA, 34, 39, 1020  
"9", 1016, 177, 87, 49, 3.5999999, 4.84000015, "Buckingham", 45, "male", 69, 166, "large", 160, 80, 128, 86, 34, 40, 300  
"10", 1022, 263, 89, 40, 6.5999999, 5.78000021, "Buckingham", 55, "female", 63, 202, "small", 108, 72, NA, NA, 45, 50, 240
```

Let us fit a multiple linear regression model including a two way interaction between **age** and **frame**

Frame is a categorical variable with three levels (*small, medium and large*)

Age and Frame Only - No Interaction

```
=====
Dep. Variable:          chol    R-squared:          0.067
Model:                  OLS     Adj. R-squared:       0.059
Method:                 Least Squares   F-statistic:      9.175
Date:                  Wed, 11 Apr 2018   Prob (F-statistic): 7.07e-06
Time:                  07:22:30    Log-Likelihood:   -2013.8
No. Observations:      390        AIC:              4036.
Df Residuals:          386        BIC:              4051.
Df Model:               3
Covariance Type:       nonrobust
=====

               coef    std err          t      P>|t|      [0.025    0.975]
-----
Intercept      176.7377     8.455    20.903     0.000    160.114    193.362
frame[T.medium]  9.7609     5.324     1.833     0.068    -0.708    20.229
frame[T.small]  -4.1812     6.107    -0.685     0.494   -16.188     7.826
age             0.5916     0.139     4.256     0.000     0.318     0.865
=====
```

Age and Frame with Interaction

```
=====
Dep. Variable:          chol    R-squared:          0.079
Model:                  OLS     Adj. R-squared:      0.067
Method:                 Least Squares    F-statistic:    6.580
Date:                   Wed, 11 Apr 2018    Prob (F-statistic): 6.85e-06
Time:                   07:22:30    Log-Likelihood:    -2011.2
No. Observations:      390    AIC:              4034.
Df Residuals:          384    BIC:              4058.
Df Model:              5
Covariance Type:       nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	200.9110	14.653	13.711	0.000	172.100	229.722
frame[T.medium]	-16.3423	17.631	-0.927	0.355	-51.008	18.323
frame[T.small]	-44.9474	18.984	-2.368	0.018	-82.273	-7.621
age	0.1341	0.266	0.505	0.614	-0.388	0.657
age:frame[T.medium]	0.4997	0.335	1.493	0.136	-0.158	1.158
age:frame[T.small]	0.8511	0.378	2.252	0.025	0.108	1.594

```
=====
```

Large frame: $\text{chol} = 200.9 + 0.1341 \times \text{age}$

Medium Frame: $\text{chol} = 200.9 - 16.3423 + (0.1341 + 0.4997) \times \text{age}$

Small Frame: $\text{chol} = 200.9 - 44.9474 + (0.1341 + 0.8511) \times \text{age}$

In addition to main coefficients for age and frame, we see additional terms for interaction of age with two non-reference levels of frame.

If at least one level is significant, we should treat the categorical variable as significant.

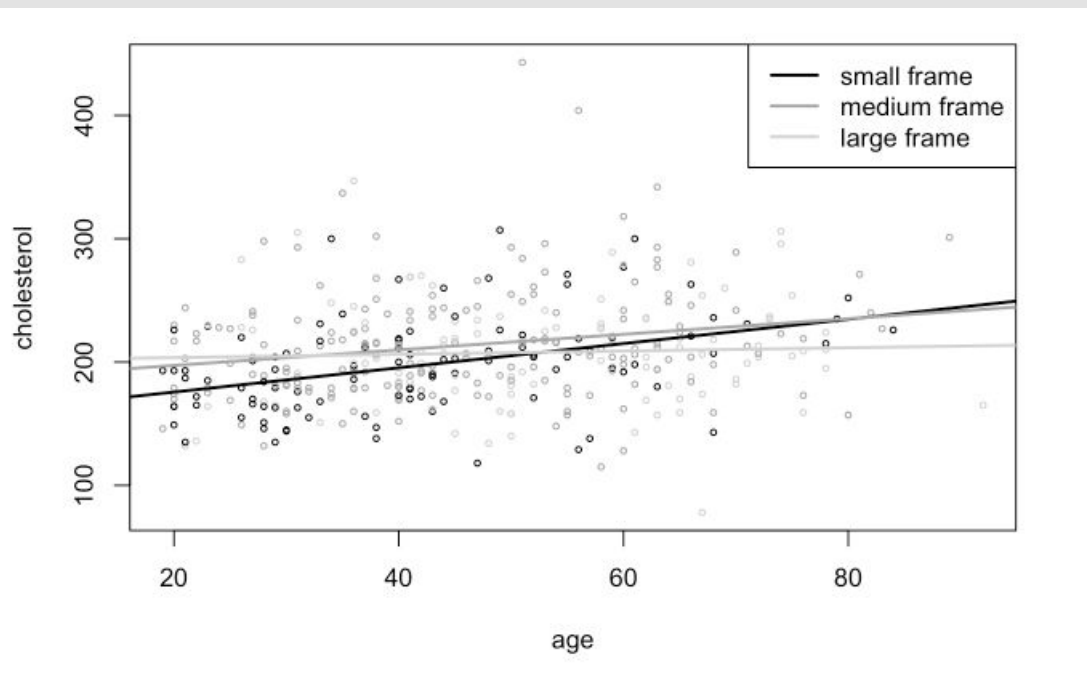
So there is statistically significant interaction between age and frame.

Plot them

Large frame: $\text{chol} = 200.9 + 0.1341 \times \text{age}$

Medium Frame: $\text{chol} = 200.9 - 16.3423 + (0.1341 + 0.4997) \times \text{age}$

Small Frame: $\text{chol} = 200.9 - 44.9474 + (0.1341 + 0.8511) \times \text{age}$



The non-parallel nature of the three lines reflects the effect of interaction.

Code (Lab later)

```
main = sm.ols(formula="chol ~ age+frame",data=d).fit()  
print(main.summary())
```

```
inter = sm.ols(formula="chol ~ age*frame",data=d).fit()  
print(inter.summary())
```


Two Categorical Variables

Gender and Frame: "chol ~ gender*frame"

```
=====
Dep. Variable:          chol    R-squared:          0.025
Model:                  OLS     Adj. R-squared:       0.012
Method:                 Least Squares    F-statistic:    1.947
Date:                   Thu, 12 Apr 2018    Prob (F-statistic): 0.0858
Time:                   17:18:42    Log-Likelihood:  -2022.3
No. Observations:      390    AIC:              4057.
Df Residuals:          384    BIC:              4080.
Df Model:               5
Covariance Type:       nonrobust
=====
```

```
=====
                                coef    std err          t      P>|t|      [0.025    0.975]
-----
Intercept                    209.8810      6.722     31.222     0.000     196.664     223.098
gender[T.male]                -3.1760      8.735     -0.364     0.716     -20.351     13.998
frame[T.medium]               4.5759      7.845      0.583     0.560     -10.849     20.001
frame[T.small]              -10.2433      8.526     -1.201     0.230     -27.007      6.520
gender[T.male]:frame[T.medium]  0.6897     10.981      0.063     0.950     -20.900     22.279
gender[T.male]:frame[T.small] -3.3146     12.634     -0.262     0.793     -28.156     21.527
=====
```

Two Continuous Variables

```
chol ~ height*weight
```

```
=====
Dep. Variable:          chol    R-squared:          0.010
Model:                  OLS     Adj. R-squared:     0.003
Method:                 Least Squares    F-statistic:    1.369
Date:                   Thu, 12 Apr 2018    Prob (F-statistic): 0.252
Time:                   17:20:39    Log-Likelihood:  -2059.2
No. Observations:      396    AIC:              4126.
Df Residuals:          392    BIC:              4142.
Df Model:              3
Covariance Type:       nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	360.5061	176.110	2.047	0.041	14.269	706.744
height	-2.5536	2.675	-0.955	0.340	-7.813	2.705
weight	-0.5511	0.987	-0.559	0.577	-2.491	1.389
height:weight	0.0097	0.015	0.651	0.516	-0.020	0.039

```
=====
```

Higher Order Interactions

chol ~ height*waist*hip

	coef	std err	t	P> t	[0.025	0.975]
Intercept	2121.8158	1762.750	1.204	0.229	-1343.950	5587.581
height	-33.7402	27.064	-1.247	0.213	-86.951	19.470
waist	-43.6014	47.000	-0.928	0.354	-136.008	48.806
height:waist	0.7946	0.718	1.107	0.269	-0.617	2.206
hip	-46.1213	42.251	-1.092	0.276	-129.192	36.949
height:hip	0.7876	0.650	1.211	0.227	-0.491	2.066
waist:hip	1.0799	1.036	1.042	0.298	-0.958	3.118
height:waist:hip	-0.0189	0.016	-1.187	0.236	-0.050	0.012

Combine Main & Interactive Terms

```
chol ~ age+weight+age*weight
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	125.9460	28.568	4.409	0.000	69.782	182.110
age	1.4679	0.607	2.420	0.016	0.275	2.661
weight	0.2959	0.161	1.843	0.066	-0.020	0.612
age:weight	-0.0047	0.003	-1.369	0.172	-0.012	0.002

Interactive Terms

Lab Review

Code for Slide Exercises

```
import numpy as np
import pandas as pd
import statsmodels.formula.api as sm

# # Multiple Logistic Regression
d=pd.read_csv("diabetes.csv")

main = sm.ols(formula="chol ~ age+frame",data=d).fit()
print(main.summary())

inter = sm.ols(formula="chol ~ age*frame",data=d).fit()
print(inter.summary())

inter = sm.ols(formula="chol ~ gender*frame",data=d).fit()
print(inter.summary())

inter = sm.ols(formula="chol ~ height*weight",data=d).fit()
print(inter.summary())

inter = sm.ols(formula="chol ~ age+weight+age*weight",data=d).fit()
print(inter.summary())
```

Section 2

Linear Model Selection

Linear Model Selection

Statisticians refer to the balancing act between **goodness-of-fit** and **complexity** as the principle of parsimony, where the goal of the associated model selection is to find a model that's as simple as possible (in other words, with relatively low complexity), without sacrificing too much goodness-of-fit.

General Guidelines

- For categorical variables, you cannot just remove some levels (You can remove the entire predictor)
- If an interaction is present in the fitted model, all lower-order interactions and main effects of the relevant predictors must remain in the model.
- In models where you've used a polynomial transformation of a certain explanatory variable, keep all lower-order polynomial terms in the model if the highest is deemed significant.

Celebrated statistician George Box (1919– 2013): “All models are wrong, but some are useful.”

Linear Model Selection

Analysis of Variance

Nested Comparisons

In nested models, the smaller model is a reduced version of the bigger, more complex model.

Height~Wr.Hnd+Sex

Height~Wr.Hnd+Sex+Smoke

Test with Anova

Analysis of Variance is useful for comparing means from different models for statistical significance.

```
Height~Wr.Hnd+Sex  
Height~Wr.Hnd+Sex+Smoke
```

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	204.0	9959.181574	0.0	NaN	NaN	NaN
1	201.0	9914.305976	3.0	44.875598	0.303265	0.823015

The result of this particular test, is a high p-value of 0.823, suggesting no evidence against H_0 .

This means that adding Smoke to the reduced model, which includes only the explanatory variables Wr.Hnd and Sex, offers no tangible improvement in fit when it comes to modeling student height.

Diabetes Data Set

cholesterol based on age and frame.

(Identify and delete any individuals with a missing value for age or for frame.)

- chol~1
- chol~age
- chol~age+frame
- chol~age*frame

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	389.0	747264.823077	0.0	NaN	NaN	NaN
1	388.0	712078.244044	1.0	35186.579033	19.630610	0.000012
2	386.0	697527.450662	2.0	14550.793383	4.058947	0.018010
3	384.0	688294.773581	2.0	9232.677081	2.575458	0.077433

Notes

(If you hadn't deleted the records containing missing values in those predictors, you would've received an error telling you that the data sets for the four models were not equal sizes.)

- including age provides a significant improvement to modeling chol;
- including a main effect for frame provides a further mild improvement;
- there's very weak evidence, if any, that including an interactive effect is beneficial to goodness-of-fit.

From this, you might prefer to use the main-effects-only model.

Quick Lab

```
d=pd.read_csv("diabetes.csv")
chol1 = sm.ols(formula="chol ~ 1",data=d).fit()
chol2 = sm.ols(formula="chol ~ age",data=d).fit()
chol3 = sm.ols(formula="chol ~ age+frame",data=d).fit()
chol4 = sm.ols(formula="chol ~ age*frame",data=d).fit()

print(sma.stats.anova_lm(chol1,chol2,chol3,chol4))
```

Linear Model Selection

Forward Selection

Model Selection: Forward Selection

Start with an intercept-only model

Perform a series of independent tests to determine which of your predictor variables significantly improves the goodness-of-fit.

Update your model object by adding that term

Execute the series of tests again for all remaining terms to determine which of those would further improve the fit.

Repeat the process until there aren't any more terms that improve the fit in a statistically significant way.

Nuclear Dataset

cost

The capital cost of construction in millions of dollars adjusted to 1976 base.

date

The date on which the construction permit was issued. The data are measured in years since January 1 1990 to the nearest month.

t1

The time between application for and issue of the construction permit.

t2

The time between issue of operating license and construction permit.

cap

The net capacity of the power plant (MWe).

pr

A binary variable where 1 indicates the prior existence of a LWR plant at the same site.

ne

A binary variable where 1 indicates that the plant was constructed in the north-east region of the U.S.A.

ct

A binary variable where 1 indicates the use of a cooling tower in the plant.

bw

A binary variable where 1 indicates that the nuclear steam supply system was manufactured by Babcock-Wilcox.

cum.n

The cumulative number of power plants constructed by each architect-engineer.

pt

A binary variable where 1 indicates those plants with partial turnkey guarantees.

=====

Dep. Variable:	chol	R-squared:	0.000
Model:	OLS	Adj. R-squared:	0.000
Method:	Least Squares	F-statistic:	inf
Date:	Wed, 11 Apr 2018	Prob (F-statistic):	nan
Time:	12:48:59	Log-Likelihood:	-2027.2
No. Observations:	390	AIC:	4056.
Df Residuals:	389	BIC:	4060.
Df Model:	0		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
Intercept	207.8385	2.219	93.647	0.000	203.475	212.202

Compare against a model with each predictor

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	562837.107402	1.0	334335.201295	17.820531	0.000207	<i>date</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	710188.747817	1.0	186983.56088	7.898614	0.00863	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	897144.924936	1.0	27.383761	0.000916	0.97606	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	697499.098161	1.0	199673.210536	8.588106	0.006414	<i>cap</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	888135.636650	1.0	9036.672047	0.305246	0.584705	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	768531.357071	1.0	128640.951626	5.021563	0.032589	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	854130.070411	1.0	43042.238286	1.511792	0.228422	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	880966.893535	1.0	6205.415162	0.551851	0.46334	<i>bw</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	829234.216944	1.0	67938.091753	2.457861	0.127427	<i>cumn</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	31.0	897172.308697	0.0	NaN	NaN	NaN	
1	30.0	591838.528850	1.0	305333.779847	15.477217	0.000458	<i>pt</i>



New Base Model with date

cost ~ date

```
=====
Dep. Variable:          cost    R-squared:          0.373
Model:                  OLS     Adj. R-squared:       0.352
Method:                 Least Squares   F-statistic:       17.82
Date:                   Wed, 11 Apr 2018   Prob (F-statistic): 0.000207
Time:                   15:03:36   Log-Likelihood:    -201.81
No. Observations:       32         AIC:              407.6
Df Residuals:           30         BIC:              410.5
Df Model:                1
Covariance Type:        nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	-6553.5666	1661.963	-3.943	0.000	-9947.748	-3159.386
date	102.2893	24.231	4.221	0.000	52.803	151.775

Compare with remaining predictors

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	547515.249347	1.0	15321.858055	0.811546	0.375084	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	494675.912803	1.0	68161.194599	3.995898	0.055061	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	373105.360239	1.0	189731.747163	14.747096	0.000616	<i>cap</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	558809.947565	1.0	4027.159837	0.208993	0.650964	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	470580.704698	1.0	92256.402704	5.685392	0.023867	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	508043.436220	1.0	54793.671182	3.127718	0.087491	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	561597.054694	1.0	1240.052708	0.064034	0.802015	<i>bw</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	558178.895546	1.0	4658.211856	0.242016	0.626457	<i>cumn</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	30.0	562837.107402	0.0	NaN	NaN	NaN	
1	29.0	472250.424653	1.0	90586.682749	5.562756	0.0253	<i>pt</i>



New Base Model with date & cap

```
cost ~ date+cap
```

OLS Regression Results

```
=====
Dep. Variable:          cost    R-squared:                0.584
Model:                  OLS     Adj. R-squared:            0.555
Method:                 Least Squares    F-statistic:        20.37
Date:                  Wed, 11 Apr 2018    Prob (F-statistic):  2.98e-06
Time:                  15:08:59    Log-Likelihood:     -195.23
No. Observations:      32    AIC:                  396.5
Df Residuals:          29    BIC:                  400.9
Df Model:               2
Covariance Type:       nonrobust
=====
```

```
=====
              coef    std err          t      P>|t|      [0.025    0.975]
-----
Intercept -6790.8792    1377.668     -4.929     0.000    -9608.527    -3973.231
date       100.7764      20.070      5.021     0.000      59.729     141.823
cap         0.4132       0.108      3.840     0.001       0.193       0.633
=====
```

Compare with remaining predictors

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	372148.158204	1.0	957.202034	0.072019	0.790387	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	359749.826705	1.0	13355.533533	1.039486	0.31667	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	354651.969151	1.0	18453.391088	1.456907	0.237521	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	278568.803905	1.0	94536.556334	9.502225	0.004575	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	324148.494509	1.0	48956.86573	4.228902	0.049169	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	365600.547942	1.0	7504.812297	0.574766	0.454705	<i>bw</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	345043.635875	1.0	28061.724364	2.277185	0.142493	<i>cumn</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	29.0	373105.360239	0.0	NaN	NaN	NaN	
1	28.0	277188.678320	1.0	95916.681918	9.688949	0.004243	<i>pt</i>



New Base Model with date, cap & pt

cost ~ date+cap+pt

```
=====
Dep. Variable:          cost    R-squared:                0.691
Model:                  OLS     Adj. R-squared:           0.658
Method:                 Least Squares   F-statistic:            20.88
Date:                  Wed, 11 Apr 2018   Prob (F-statistic):     2.64e-07
Time:                  16:14:29    Log-Likelihood:        -190.47
No. Observations:      32         AIC:                   388.9
Df Residuals:          28         BIC:                   394.8
Df Model:               3
Covariance Type:       nonrobust
=====
```

```
=====
              coef    std err          t      P>|t|      [0.025    0.975]
-----
Intercept -4553.3078    1406.112     -3.238    0.003    -7433.598    -1673.018
date       68.5245      20.428      3.355    0.002      26.680     110.369
cap         0.4191       0.094      4.439    0.000       0.226       0.612
pt        -162.7636     52.290     -3.113    0.004     -269.875     -55.652
=====
```


Compare with remaining predictors

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	28.0	277188.678320	0.0	NaN	NaN	NaN	
1	27.0	277177.159655	1.0	11.518666	0.001122	0.973525	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	28.0	277188.678320	0.0	NaN	NaN	NaN	
1	27.0	266759.995586	1.0	10428.682734	1.055535	0.313353	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	28.0	277188.678320	0.0	NaN	NaN	NaN	
1	27.0	271171.408835	1.0	6017.269485	0.599128	0.445636	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	28.0	277188.678320	0.0	NaN	NaN	NaN	
1	27.0	222616.935606	1.0	54571.742714	6.618711	0.015908	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	28.0	277188.678320	0.0	NaN	NaN	NaN	
1	27.0	259069.239609	1.0	18119.438711	1.888394	0.180684	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	28.0	277188.678320	0.0	NaN	NaN	NaN	
1	27.0	276534.594631	1.0	654.08369	0.063863	0.802406	<i>bw</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	28.0	277188.678320	0.0	NaN	NaN	NaN	
1	27.0	276726.014154	1.0	462.664166	0.045142	0.833339	<i>cumn</i>



New Base Model with date, cap, pt & ne

```
cost ~ date+cap+pt+ne
```

OLS Regression Results

```
=====
Dep. Variable:          cost    R-squared:                0.752
Model:                  OLS     Adj. R-squared:            0.715
Method:                 Least Squares    F-statistic:         20.45
Date:                  Wed, 11 Apr 2018    Prob (F-statistic):    7.51e-08
Time:                  16:16:38    Log-Likelihood:       -186.97
No. Observations:      32    AIC:                  383.9
Df Residuals:          27    BIC:                  391.3
Df Model:              4
Covariance Type:       nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	-4756.2145	1285.663	-3.699	0.001	-7394.177	-2118.252
date	71.0193	18.668	3.804	0.001	32.716	109.322
cap	0.4198	0.086	4.873	0.000	0.243	0.597
pt	-128.9438	49.498	-2.605	0.015	-230.506	-27.382
ne	99.3988	38.636	2.573	0.016	20.124	178.674

```
=====
```

Compare with remaining predictors

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	27.0	222616.935606	0.0	NaN	NaN	NaN	
1	26.0	222509.890380	1.0	107.045226	0.012508	0.91181	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	27.0	222616.935606	0.0	NaN	NaN	NaN	
1	26.0	203387.016602	1.0	19229.919004	2.458259	0.129	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	27.0	222616.935606	0.0	NaN	NaN	NaN	
1	26.0	217386.123321	1.0	5230.812285	0.62562	0.436123	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	27.0	222616.935606	0.0	NaN	NaN	NaN	
1	26.0	206852.279022	1.0	15764.656584	1.981516	0.171075	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	27.0	222616.935606	0.0	NaN	NaN	NaN	
1	26.0	222168.964132	1.0	447.971474	0.052425	0.820687	<i>bw</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	27.0	222616.935606	0.0	NaN	NaN	NaN	
1	26.0	208796.998874	1.0	13819.936732	1.720898	0.201043	<i>cum</i>

It appears that none of the remaining predictors would yield a statistically significant improvement in goodness-of-fit.

Final Model with date, cap, pt & ne

```
cost ~ date+cap+pt+ne
```

OLS Regression Results

```
=====
Dep. Variable:          cost    R-squared:                0.752
Model:                  OLS     Adj. R-squared:            0.715
Method:                 Least Squares    F-statistic:        20.45
Date:                  Wed, 11 Apr 2018    Prob (F-statistic):    7.51e-08
Time:                  16:16:38    Log-Likelihood:       -186.97
No. Observations:      32    AIC:                  383.9
Df Residuals:          27    BIC:                  391.3
Df Model:               4
Covariance Type:       nonrobust
=====
```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	-4756.2145	1285.663	-3.699	0.001	-7394.177	-2118.252
date	71.0193	18.668	3.804	0.001	32.716	109.322
cap	0.4198	0.086	4.873	0.000	0.243	0.597
pt	-128.9438	49.498	-2.605	0.015	-230.506	-27.382
ne	99.3988	38.636	2.573	0.016	20.124	178.674

```
=====
```

Summary

In forward selection, you start with an Intercept-only model and add each predictor by doing an Analysis of Variance on the model with and without the predictor.

You stop when there are no more significant predictors to add

Let's Try a Lab

```
d=pd.read_csv("nuclear.csv")
d=d.rename(index=str,columns={"cum.n":"cumn"})
dia = []
dia.append(sm.ols("cost~date+cap+pt+ne", data=d).fit())
print(dia[0].summary())
dia.append(sm.ols("cost~date+cap+pt+ne+t1", data=d).fit())
dia.append(sm.ols("cost~date+cap+pt+ne+t2", data=d).fit())
dia.append(sm.ols("cost~date+cap+pt+ne+pr", data=d).fit())
dia.append(sm.ols("cost~date+cap+pt+ne+ct", data=d).fit())
dia.append(sm.ols("cost~date+cap+pt+ne+bw", data=d).fit())
dia.append(sm.ols("cost~date+cap+pt+ne+cum", data=d).fit())
for i in np.arange(1,7):
    print(sma.stats.anova_lm(dia[0],dia[i]))
```

Linear Model Selection

Backward Selection

Backward Selection

backward selection starts with your fullest model and systematically drops terms.

Define the fullest model as that which predicts cost by main effects of all available covariates

Base Model with all predictors

```
=====
Dep. Variable:          cost    R-squared:          0.839
Model:                  OLS     Adj. R-squared:       0.763
Method:                 Least Squares   F-statistic:       10.98
Date:                   Wed, 11 Apr 2018   Prob (F-statistic): 2.84e-06
Time:                   16:30:41   Log-Likelihood:    -180.00
No. Observations:      32         AIC:               382.0
Df Residuals:          21         BIC:               398.1
Df Model:              10
Covariance Type:       nonrobust
=====
```

```
=====
              coef    std err          t      P>|t|      [0.025      0.975]
-----
Intercept  -8134.8823   2787.794    -2.918    0.008   -1.39e+04   -2337.347
date        115.4832    42.260     2.733    0.012    27.599    203.367
t1           5.9284    10.887     0.545    0.592   -16.712    28.569
t2           4.5709     2.243     2.038    0.054    -0.094     9.236
cap          0.4216     0.088     4.768    0.000     0.238     0.606
pr          -81.1211    40.769    -1.990    0.060   -165.905     3.662
ne          137.4502    38.690     3.553    0.002     56.989    217.911
ct           43.2733    34.307     1.261    0.221   -28.071    114.618
bw          -8.2384    51.884    -0.159    0.875   -116.136    99.659
cumn        -6.9886     3.822    -1.829    0.082   -14.936     0.959
pt          -19.2476    63.672    -0.302    0.765   -151.660    113.165
=====
```

Compare against a model without each predictor

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	195294.933636	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	51230.041487	7.467682	0.01247	<i>date</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	146099.150012	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	2034.257864	0.296529	0.591803	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	172546.250258	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	28481.35811	4.15166	0.05439	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	300007.623455	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	155942.731307	22.731405	0.000104	<i>cap</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	171226.152965	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	27161.260816	3.959233	0.059794	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	230645.870888	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	86580.978739	12.620705	0.001883	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	154979.922788	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	10915.03064	1.591058	0.221008	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	144237.861410	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	172.969261	0.025213	0.875354	<i>bw</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	167004.306468	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	22939.41432	3.343824	0.081698	<i>cumn</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	22.0	144691.792614	0.0	NaN	NaN	NaN	
1	21.0	144064.892149	1.0	626.900465	0.091382	0.765401	<i>pt</i>

it seems the predictor *bw* has the single least significant effect on reducing the goodness-of-fit,



New Base Model without bw

```
=====
Dep. Variable:          cost    R-squared:          0.839
Model:                  OLS     Adj. R-squared:       0.773
Method:                 Least Squares   F-statistic:       12.76
Date:                   Wed, 11 Apr 2018   Prob (F-statistic): 7.66e-07
Time:                   16:34:46   Log-Likelihood:    -180.02
No. Observations:      32     AIC:              380.0
Df Residuals:          22     BIC:              394.7
Df Model:               9
Covariance Type:       nonrobust
=====
```

```
=====
              coef      std err          t      P>|t|      [0.025      0.975]
-----
Intercept -7979.5131    2551.950     -3.127     0.005    -1.33e+04    -2687.093
date      113.1505      38.736      2.921     0.008      32.817     193.484
t1         6.6587       9.647      0.690     0.497     -13.347     26.665
t2         4.4461       2.054      2.165     0.042       0.186       8.706
cap        0.4235       0.086      4.940     0.000       0.246       0.601
pr       -80.1272     39.383     -2.035     0.054    -161.802       1.548
ne       137.1744     37.785      3.630     0.001      58.812     215.537
ct        44.2666     32.976      1.342     0.193     -24.120     112.654
cumn     -7.0778      3.696     -1.915     0.069     -14.742       0.586
pt       -22.3293     59.283     -0.377     0.710    -145.275     100.617
=====
```

Compare against a model without each predictor

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	200179.576049	0.0	NaN	NaN	NaN	
1	22.0	144237.861410	1.0	55941.714639	8.532557	0.007913	<i>date</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	147361.703135	0.0	NaN	NaN	NaN	
1	22.0	144237.861410	1.0	3123.841725	0.476467	0.497245	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	174955.359915	0.0	NaN	NaN	NaN	
1	22.0	144237.861410	1.0	30717.498505	4.685212	0.041546	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	304214.01552	0.0	NaN	NaN	NaN	
1	22.0	144237.86141	1.0	159976.154111	24.400496	0.000061	<i>cap</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	171377.366628	0.0	NaN	NaN	NaN	
1	22.0	144237.861410	1.0	27139.505219	4.139476	0.054122	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	230645.874117	0.0	NaN	NaN	NaN	
1	22.0	144237.861410	1.0	86408.012707	13.179454	0.001479	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	156052.654523	0.0	NaN	NaN	NaN	
1	22.0	144237.861410	1.0	11814.793114	1.802061	0.193153	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	168286.15768	0.0	NaN	NaN	NaN	
1	22.0	144237.86141	1.0	24048.296271	3.667986	0.068557	<i>cumn</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	23.0	145167.987705	0.0	NaN	NaN	NaN	
1	22.0	144237.861410	1.0	930.126296	0.141868	0.710039	<i>pt</i>

pt is the next most sensible main effect to drop.



Compare against a model without each predictor

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	238844.665951	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	93676.678245	14.841864	0.000811	<i>date</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	148021.428849	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	2853.441144	0.452091	0.508043	<i>t1</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	176480.064481	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	31312.076776	4.960996	0.035996	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	310762.174057	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	165594.186351	26.236268	0.000034	<i>cap</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	175152.045770	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	29984.058065	4.750588	0.039782	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	258826.519005	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	113658.5313	18.007732	0.000307	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	160769.641361	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	15601.653656	2.471881	0.129556	<i>ct</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	24.0	193650.384331	0.0	NaN	NaN	NaN	
1	23.0	145167.987705	1.0	48482.396626	7.681412	0.010856	<i>cumn</i>



t1 is the next most sensible main effect to drop.

Compare against a model without each predictor

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	25.0	517039.978932	0.0	NaN	NaN	NaN	<i>date</i>
1	24.0	148021.428849	1.0	369018.550083	59.832183	5.697409e-08	
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	25.0	176560.224962	0.0	NaN	NaN	NaN	<i>t2</i>
1	24.0	148021.428849	1.0	28538.796113	4.627243	0.041748	
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	25.0	310867.017049	0.0	NaN	NaN	NaN	<i>cap</i>
1	24.0	148021.428849	1.0	162845.5882	26.40357	0.000029	
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	25.0	175230.053526	0.0	NaN	NaN	NaN	<i>pr</i>
1	24.0	148021.428849	1.0	27208.624677	4.411571	0.046386	
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	25.0	262135.460638	0.0	NaN	NaN	NaN	<i>ne</i>
1	24.0	148021.428849	1.0	114114.031789	18.502299	0.000245	
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	25.0	163221.598289	0.0	NaN	NaN	NaN	<i>ct</i>
1	24.0	148021.428849	1.0	15200.16944	2.464536	0.129534	
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	25.0	201309.521097	0.0	NaN	NaN	NaN	<i>cumn</i>
1	24.0	148021.428849	1.0	53288.092248	8.640061	0.007164	



ct is the next most sensible main effect to drop.

Compare against a model without each predictor

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	26.0	537798.670515	0.0	NaN	NaN	NaN	
1	25.0	163221.598289	1.0	374577.072226	57.372473	6.271558e-08	<i>date</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	26.0	210296.772856	0.0	NaN	NaN	NaN	
1	25.0	163221.598289	1.0	47075.174567	7.210316	0.012685	<i>t2</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	26.0	320671.742805	0.0	NaN	NaN	NaN	
1	25.0	163221.598289	1.0	157450.144516	24.116009	0.000047	<i>cap</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	26.0	205491.553637	0.0	NaN	NaN	NaN	
1	25.0	163221.598289	1.0	42269.955348	6.47432	0.017499	<i>pr</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	26.0	290709.372718	0.0	NaN	NaN	NaN	
1	25.0	163221.598289	1.0	127487.774429	19.526793	0.000168	<i>ne</i>
	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)	
0	26.0	212897.740727	0.0	NaN	NaN	NaN	
1	25.0	163221.598289	1.0	49676.142438	7.608696	0.010703	<i>cumn</i>

All remaining predictors
seem significant

Final Model

```
=====
Dep. Variable:          cost    R-squared:          0.818
Model:                  OLS     Adj. R-squared:       0.774
Method:                 Least Squares   F-statistic:        18.74
Date:                  Wed, 11 Apr 2018   Prob (F-statistic):  3.80e-08
Time:                  16:42:17   Log-Likelihood:     -182.00
No. Observations:      32        AIC:                378.0
Df Residuals:          25        BIC:                388.3
Df Model:              6
Covariance Type:       nonrobust
=====
```

```
=====
              coef    std err          t      P>|t|      [0.025      0.975]
-----
Intercept -9701.5221    1294.355     -7.495     0.000    -1.24e+04    -7035.749
date       139.5909      18.429       7.574     0.000      101.635     177.546
t2          4.9051       1.827       2.685     0.013        1.143        8.667
cap         0.4137       0.084       4.911     0.000        0.240        0.587
pr        -88.5147      34.787     -2.544     0.017     -160.160     -16.869
ne        150.2262      33.996       4.419     0.000        80.210     220.243
cumn       -7.9194       2.871     -2.758     0.011     -13.832     -2.006
=====
```


Forward vs Backward Selection

Do they yield same models?

Forward: **cost~date+cap+pt+ne**

Backward: **cost~date+t2+cap+pr+ne+cumn**

What might cause these two to give different models?

Linear Model Selection

Lab Exercises

Use Diabetes Dataset

There are some missing values in diabetes that might interfere with model selection algorithms. Define a new version of the diabetes data frame that deletes all rows with a missing value in any of the following variables: chol, age, gender, height, weight, frame, waist, hip, location.

Use forward selection based on partial F-tests with a conventional significance level of $\alpha = 0.05$ to choose a model, starting from `dia.null`. Use the above variables.

Use backward selection based on partial F-tests with a conventional significance level of $\alpha = 0.05$ to choose a model, again starting from the full model with above variables.

Section 3

Resampling Methods

Resampling Methods

Resampling involves repeatedly drawing samples from a training set and refitting a model of interest on each sample.

They are computationally expensive because they involve fitting the same method multiple times.

Training Error vs Test Error

Given a data set, the use of a particular statistical learning method is warranted if it results in a low test error. The test error can be easily calculated if a designated test set is available.

The training error can be easily calculated by applying the statistical learning method to the observations used in its training. The training error rate often is quite different from the test error rate. The training error rate often is quite different from the test error rate.

When we do not have a designated test set that can be used to directly estimate the test error rate, some techniques can be used to estimate this quantity using the available training data.

Cross Validation

We instead consider a class of methods that estimate the test error rate by holding out a subset of the training observations from the fitting process, and then applying the statistical learning method to those held out observations.

Cross-validation can be used to estimate the test error associated with a given statistical learning method in order to evaluate its performance. The process of evaluating a model's performance is known as **model assessment**, whereas the process of selecting the proper level of flexibility for a model is known as **model selection**.

Cross Validation

Validation Sets

Validation Set Approach

Validation set approach involves randomly dividing the available set of observations into a training set and a validation set.

We fit the model on an a training set and the fitted model is used to predict the responses for observations in the validation set.

The resulting validation set error rate is typically assessed using MSE and provides an estimate of the test error rate.

Auto Dataset

We select a random subset of 196 observations from the original set as the training data.

We now build a model using only the training set.

Let us compare the predictions for the validation set and calculate the mean square error.

The estimated test MSE for the linear regression is 26.14.

Polynomial Fits

Choose Another Training Set

[1] 23.29559

Polynomial Fits

Which ones are worse?

Validation Set Approach

Validation test error rate is highly variable, depending on which observations are included in the validation set.

Since a subset of observations are used to fit the model, and statistical methods tend perform worse when trained on fewer observations, the validation error rate may tend to overestimate the test error rate for the model fit on the entire data set.

Cross Validation

Leave-One-Out
Cross-Validation (LOOCV)

Leave-One-Out Cross-Validation (LOOCV)

This approach involves splitting the set of observations into two parts. However, a single observation is used for validation, and the remaining observations are used for training.

We fit a model for $n-1$ observations, and prediction is made for the excluded observation.

The MSE on the test observation provides the estimate for test error.

We can repeat this procedure n times for n observations and compute n MSEs.

LOOCV estimate for the test MSE is the average of these n test errors.

Compare with Validation Set

In LOOCV, we repeatedly fit the method with $n-1$ data sets (or almost the same)

In contrast with Validation Set, which can yield different results owing to randomness in the test/training set selection, repeating LOOCV any number of times will yield same results.

This shows a sharp drop in estimated test MSE from linear to quadratic fits, but then the improvement beyond that is questionable.

LOOCV is a generic method and can be also be used with logistic regression.

Cross Validation

k-Fold Cross-Validation

k-Fold Cross-Validation

k-fold CV is an alternative to LOOCV. We randomly divide the set of observations into k groups or folds of approximately equal size

1. The first fold is treated as a validation set
2. The method is then fit on the remaining $k-1$ folds.
3. The MSE is computed on observations in the held-out fold
4. This process is repeated k times, each time picking a different group of observations as the validation set
5. This method gives k estimates of the test error, which are then averaged to calculate the k -fold CV estimate.

Note that LOOCV is a special case of k -fold CV when k equals n

Cross Validation

Cross Validation for Classification Problems

Cross Validation for Classification Problems

Cross-validation can also be a very useful approach in the classification setting when Y is qualitative.

Rather than using MSE to quantify test error, we instead use the number of misclassified observations.