

# LASSO Models

Justin Post

# Recap

- Judge the model's effectiveness using a **Loss** function
- Often split data into a training and test set
  - Perhaps 70/30 or 80/20
- Cross-validation gives a way to use more of the data while still seeing how the model does on test data
  - Commonly 5 fold or 10 fold is done
  - Once a best model is chosen, model is refit on entire data set

# Recap

- Judge the model's effectiveness using a **Loss** function
- Often split data into a training and test set
  - Perhaps 70/30 or 80/20
- Cross-validation gives a way to use more of the data while still seeing how the model does on test data
  - Commonly 5 fold or 10 fold is done
  - Once a best model is chosen, model is refit on entire data set
- Often use both! Let's see why by introducing a model with a **tuning parameter**

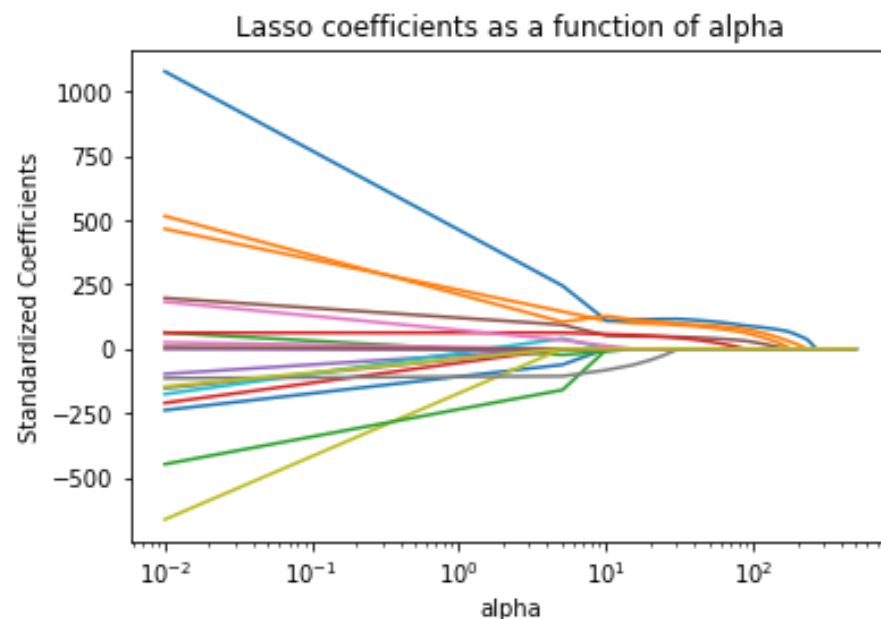
# LASSO Model

- **Least Angle Subset and Selection Operator** or LASSO
  - Similar to Least Squares but a penalty is placed on the sum of the absolute values of the regression coefficients
  - $\alpha$  ( $>0$ ) is called a tuning parameter

$$\min_{\beta' s} \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_{1i} + \dots + \beta_p x_{pi}))^2 + \alpha \sum_{j=1}^p |\beta_j|$$

# LASSO Model

- **Least Angle Subset and Selection Operator** or LASSO
  - Similar to Least Squares but a penalty is placed on the sum of the absolute values of the regression coefficients
  - Sets coefficients to 0 as you 'shrink'!



# Tuning Parameter

- When choosing the tuning parameter, we are really considering a **family of models**!
- Consider an  $\alpha = 0.1$  (small amount of shrinkage here)

```
from sklearn import linear_model
lasso = linear_model.Lasso(alpha=0.1)
lasso.fit(bike_data[["year", "log_km_driven"]].values, bike_data["log_selling_price"].values)
```

```
print(lasso.intercept_, lasso.coef_)
```

```
## -164.61209472866094 [ 0.08761607 -0.11092474]
```

# Tuning Parameter

- When choosing the tuning parameter, we are really considering a **family of models**!
- Consider an  $\alpha = 0.1$  (small amount of shrinkage here)

```
from sklearn import linear_model
lasso = linear_model.Lasso(alpha=0.1)
lasso.fit(bike_data[["year", "log_km_driven"]].values, bike_data["log_selling_price"].values)
```

```
print(lasso.intercept_, lasso.coef_)
```

```
## -164.61209472866094 [ 0.08761607 -0.11092474]
```

- Consider an  $\alpha = 1.05$  (a larger amount of shrinkage)

```
lasso = linear_model.Lasso(alpha=1.05)
lasso.fit(bike_data[["year", "log_km_driven"]].values, bike_data["log_selling_price"].values)
```

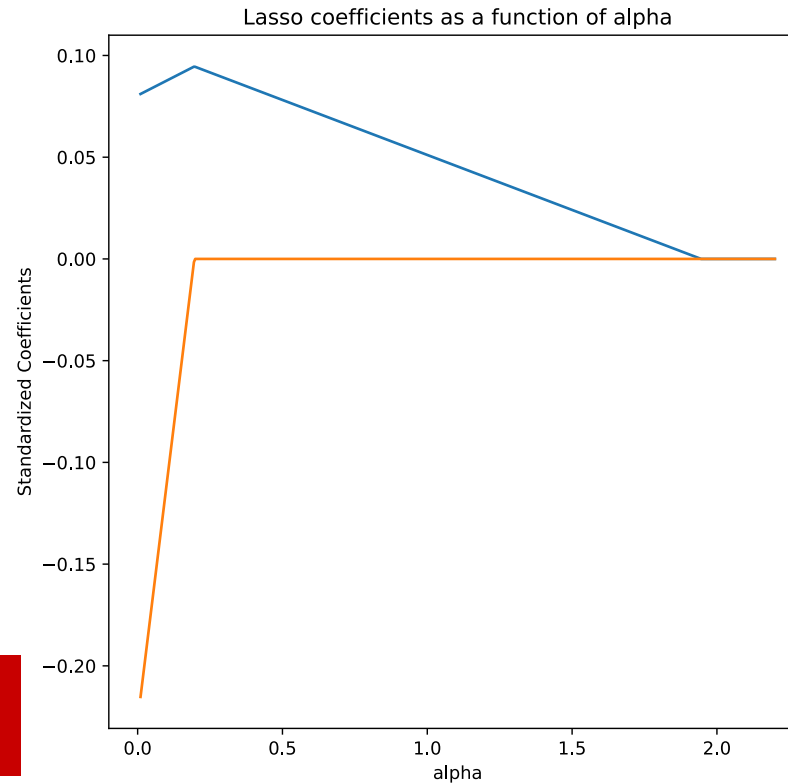
```
print(lasso.intercept_, lasso.coef_)
```

```
## -86.65630892150762 [ 0.04835598 -0.          ]
```

# LASSO Fits Visual

- Perfect place for CV to help select the best  $\alpha$ !

```
## (-0.09950000000000003, 2.3095000000000003, -0.23074900910594773, 0.1099672686377952)
```





# Using CV to Select the Tuning Parameter

- Return the optimal  $\alpha$  using `LassoCV`

```
from sklearn.linear_model import LassoCV
lasso_mod = LassoCV(cv=5, random_state=0, alphas = np.linspace(0,2.2,100)) \
    .fit(bike_data[["year", "log_km_driven"]].values,
        bike_data["log_selling_price"].values)
```

# Using CV to Select the Tuning Parameter

- Return the optimal  $\alpha$  using `LassoCV`

```
pd.DataFrame(zip(lasso_mod.alphas, lasso_mod.mse_path_), columns = ["alpha_value", "MSE_by_fold"])
```

```
##      alpha_value      MSE_by_fold
## 0      0.000000 [0.5496710875578059, 0.6805679103740427, 0.500...
## 1      0.022222 [0.5496710875578059, 0.6805679103740427, 0.500...
## 2      0.044444 [0.5496710875578059, 0.6805679103740427, 0.500...
## 3      0.066667 [0.5496710875578059, 0.6805679103740427, 0.500...
## 4      0.088889 [0.5496710875578059, 0.6805679103740427, 0.500...
## ..      ...      ...
## 95     2.111111 [0.3046546135682859, 0.3626276356362568, 0.191...
## 96     2.133333 [0.2998496943347488, 0.35411026477928204, 0.18...
## 97     2.155556 [0.29626758731408676, 0.34645956641174064, 0.1...
## 98     2.177778 [0.2939082925063055, 0.3396755405336282, 0.182...
## 99     2.200000 [0.29277214247542543, 0.33375857421410476, 0.1...
##
## [100 rows x 2 columns]
```

# Using CV to Select the Tuning Parameter

- Return the optimal  $\alpha$  using `LassoCV`

```
fit_info = np.array(list(zip(lasso_mod.alphas_, np.mean(lasso_mod.mse_path_, axis = 1))))  
fit_info[fit_info[:,0].argsort()]
```

```
## array([[0.          , 0.26832555],  
##         [0.02222222, 0.26904464],  
##         [0.04444444, 0.27086204],  
##         [0.06666667, 0.27377741],  
##         [0.08888889, 0.27779157],  
##         [0.11111111, 0.28290414],  
##         [0.13333333, 0.28911508],  
##         [0.15555556, 0.29642441],  
##         [0.17777778, 0.30483239],  
##         [0.2          , 0.30967893],  
##         [0.22222222, 0.31098715],  
##         [0.24444444, 0.31159763],  
##         [0.26666667, 0.31226415],  
##         [0.28888889, 0.31298674],  
##         [0.31111111, 0.31376537],  
##         [0.33333333, 0.31460007],  
##         [0.35555556, 0.31549081],  
##         [0.37777778, 0.31643761],  
##         [0.4          , 0.31744046],  
##         [0.42222222, 0.31844331],  
##         [0.44444444, 0.31944616],  
##         [0.46666667, 0.32044901],  
##         [0.48888889, 0.32201242],  
##         [0.5          , 0.32301527],  
##         [0.52222222, 0.32401812],  
##         [0.54444444, 0.32502097],  
##         [0.56666667, 0.32602382],  
##         [0.58888889, 0.32702667],  
##         [0.6          , 0.32802952],  
##         [0.62222222, 0.32903237],  
##         [0.64444444, 0.33003522],  
##         [0.66666667, 0.33103807],  
##         [0.68888889, 0.33204092],  
##         [0.7          , 0.33304377],  
##         [0.72222222, 0.33404662],  
##         [0.74444444, 0.33504947],  
##         [0.76666667, 0.33605232],  
##         [0.78888889, 0.33705517],  
##         [0.8          , 0.33805802],  
##         [0.82222222, 0.33906087],  
##         [0.84444444, 0.34006372],  
##         [0.86666667, 0.34106657],  
##         [0.88888889, 0.34206942],  
##         [0.9          , 0.34307227],  
##         [0.92222222, 0.34407512],  
##         [0.94444444, 0.34507797],  
##         [0.96666667, 0.34608082],  
##         [0.98888889, 0.34708367],  
##         [1.          , 0.34808652],  
##         [1.02222222, 0.34908937],  
##         [1.04444444, 0.35009222],  
##         [1.06666667, 0.35109507],  
##         [1.08888889, 0.35209792],  
##         [1.1          , 0.35310077],  
##         [1.12222222, 0.35410362],  
##         [1.14444444, 0.35510647],  
##         [1.16666667, 0.35610932],  
##         [1.18888889, 0.35711217],  
##         [1.2          , 0.35811502],  
##         [1.22222222, 0.35911787],  
##         [1.24444444, 0.36012072],  
##         [1.26666667, 0.36112357],  
##         [1.28888889, 0.36212642],  
##         [1.3          , 0.36312927],  
##         [1.32222222, 0.36413212],  
##         [1.34444444, 0.36513497],  
##         [1.36666667, 0.36613782],  
##         [1.38888889, 0.36714067],  
##         [1.4          , 0.36814352],  
##         [1.42222222, 0.36914637],  
##         [1.44444444, 0.37014922],  
##         [1.46666667, 0.37115207],  
##         [1.48888889, 0.37215492],  
##         [1.5          , 0.37315777],  
##         [1.52222222, 0.37416062],  
##         [1.54444444, 0.37516347],  
##         [1.56666667, 0.37616632],  
##         [1.58888889, 0.37716917],  
##         [1.6          , 0.37817202],  
##         [1.62222222, 0.37917487],  
##         [1.64444444, 0.38017772],  
##         [1.66666667, 0.38118057],  
##         [1.68888889, 0.38218342],  
##         [1.7          , 0.38318627],  
##         [1.72222222, 0.38418912],  
##         [1.74444444, 0.38519197],  
##         [1.76666667, 0.38619482],  
##         [1.78888889, 0.38719767],  
##         [1.8          , 0.38820052],  
##         [1.82222222, 0.38920337],  
##         [1.84444444, 0.39020622],  
##         [1.86666667, 0.39120907],  
##         [1.88888889, 0.39221192],  
##         [1.9          , 0.39321477],  
##         [1.92222222, 0.39421762],  
##         [1.94444444, 0.39522047],  
##         [1.96666667, 0.39622332],  
##         [1.98888889, 0.39722617],  
##         [2.          , 0.39822902],  
##         [2.02222222, 0.39923187],  
##         [2.04444444, 0.40023472],  
##         [2.06666667, 0.40123757],  
##         [2.08888889, 0.40224042],  
##         [2.1          , 0.40324327],  
##         [2.12222222, 0.40424612],  
##         [2.14444444, 0.40524897],  
##         [2.16666667, 0.40625182],  
##         [2.18888889, 0.40725467],  
##         [2.2          , 0.40825752],  
##         [2.22222222, 0.40926037],  
##         [2.24444444, 0.41026322],  
##         [2.26666667, 0.41126607],  
##         [2.28888889, 0.41226892],  
##         [2.3          , 0.41327177],  
##         [2.32222222, 0.41427462],  
##         [2.34444444, 0.41527747],  
##         [2.36666667, 0.41628032],  
##         [2.38888889, 0.41728317],  
##         [2.4          , 0.41828602],  
##         [2.42222222, 0.41928887],  
##         [2.44444444, 0.42029172],  
##         [2.46666667, 0.42129457],  
##         [2.48888889, 0.42229742],  
##         [2.5          , 0.42330027],  
##         [2.52222222, 0.42430312],  
##         [2.54444444, 0.42530597],  
##         [2.56666667, 0.42630882],  
##         [2.58888889, 0.42731167],  
##         [2.6          , 0.42831452],  
##         [2.62222222, 0.42931737],  
##         [2.64444444, 0.43032022],  
##         [2.66666667, 0.43132307],  
##         [2.68888889, 0.43232592],  
##         [2.7          , 0.43332877],  
##         [2.72222222, 0.43433162],  
##         [2.74444444, 0.43533447],  
##         [2.76666667, 0.43633732],  
##         [2.78888889, 0.43734017],  
##         [2.8          , 0.43834302],  
##         [2.82222222, 0.43934587],  
##         [2.84444444, 0.44034872],  
##         [2.86666667, 0.44135157],  
##         [2.88888889, 0.44235442],  
##         [2.9          , 0.44335727],  
##         [2.92222222, 0.44436012],  
##         [2.94444444, 0.44536297],  
##         [2.96666667, 0.44636582],  
##         [2.98888889, 0.44736867],  
##         [3.          , 0.44837152],  
##         [3.02222222, 0.44937437],  
##         [3.04444444, 0.45037722],  
##         [3.06666667, 0.45138007],  
##         [3.08888889, 0.45238292],  
##         [3.1          , 0.45338577],  
##         [3.12222222, 0.45438862],  
##         [3.14444444, 0.45539147],  
##         [3.16666667, 0.45639432],  
##         [3.18888889, 0.45739717],  
##         [3.2          , 0.45840002],  
##         [3.22222222, 0.45940287],  
##         [3.24444444, 0.46040572],  
##         [3.26666667, 0.46140857],  
##         [3.28888889, 0.46241142],  
##         [3.3          , 0.46341427],  
##         [3.32222222, 0.46441712],  
##         [3.34444444, 0.46541997],  
##         [3.36666667, 0.46642282],  
##         [3.38888889, 0.46742567],  
##         [3.4          , 0.46842852],  
##         [3.42222222, 0.46943137],  
##         [3.44444444, 0.47043422],  
##         [3.46666667, 0.47143707],  
##         [3.48888889, 0.47243992],  
##         [3.5          , 0.47344277],  
##         [3.52222222, 0.47444562],  
##         [3.54444444, 0.47544847],  
##         [3.56666667, 0.47645132],  
##         [3.58888889, 0.47745417],  
##         [3.6          , 0.47845702],  
##         [3.62222222, 0.47945987],  
##         [3.64444444, 0.48046272],  
##         [3.66666667, 0.48146557],  
##         [3.68888889, 0.48246842],  
##         [3.7          , 0.48347127],  
##         [3.72222222, 0.48447412],  
##         [3.74444444, 0.48547697],  
##         [3.76666667, 0.48647982],  
##         [3.78888889, 0.48748267],  
##         [3.8          , 0.48848552],  
##         [3.82222222, 0.48948837],  
##         [3.84444444, 0.49049122],  
##         [3.86666667, 0.49149407],  
##         [3.88888889, 0.49249692],  
##         [3.9          , 0.49349977],  
##         [3.92222222, 0.49450262],  
##         [3.94444444, 0.49550547],  
##         [3.96666667, 0.49650832],  
##         [3.98888889, 0.49751117],  
##         [4.          , 0.49851402],  
##         [4.02222222, 0.49951687],  
##         [4.04444444, 0.50051972],  
##         [4.06666667, 0.50152257],  
##         [4.08888889, 0.50252542],  
##         [4.1          , 0.50352827],  
##         [4.12222222, 0.50453112],  
##         [4.14444444, 0.50553397],  
##         [4.16666667, 0.50653682],  
##         [4.18888889, 0.50753967],  
##         [4.2          , 0.50854252],  
##         [4.22222222, 0.50954537],  
##         [4.24444444, 0.51054822],  
##         [4.26666667, 0.51155107],  
##         [4.28888889, 0.51255392],  
##         [4.3          , 0.51355677],  
##         [4.32222222, 0.51455962],  
##         [4.34444444, 0.51556247],  
##         [4.36666667, 0.51656532],  
##         [4.38888889, 0.51756817],  
##         [4.4          , 0.51857102],  
##         [4.42222222, 0.51957387],  
##         [4.44444444, 0.52057672],  
##         [4.46666667, 0.52157957],  
##         [4.48888889, 0.52258242],  
##         [4.5          , 0.52358527],  
##         [4.52222222, 0.52458812],  
##         [4.54444444, 0.52559097],  
##         [4.56666667, 0.52659382],  
##         [4.58888889, 0.52759667],  
##         [4.6          , 0.52859952],  
##         [4.62222222, 0.52960237],  
##         [4.64444444, 0.53060522],  
##         [4.66666667, 0.53160807],  
##         [4.68888889, 0.53261092],  
##         [4.7          , 0.53361377],  
##         [4.72222222, 0.53461662],  
##         [4.74444444, 0.53561947],  
##         [4.76666667, 0.53662232],  
##         [4.78888889, 0.53762517],  
##         [4.8          , 0.53862802],  
##         [4.82222222, 0.53963087],  
##         [4.84444444, 0.54063372],  
##         [4.86666667, 0.54163657],  
##         [4.88888889, 0.54263942],  
##         [4.9          , 0.54364227],  
##         [4.92222222, 0.54464512],  
##         [4.94444444, 0.54564797],  
##         [4.96666667, 0.54665082],  
##         [4.98888889, 0.54765367],  
##         [5.          , 0.54865652],  
##         [5.02222222, 0.54965937],  
##         [5.04444444, 0.55066222],  
##         [5.06666667, 0.55166507],  
##         [5.08888889, 0.55266792],  
##         [5.1          , 0.55367077],  
##         [5.12222222, 0.55467362],  
##         [5.14444444, 0.55567647],  
##         [5.16666667, 0.55667932],  
##         [5.18888889, 0.55768217],  
##         [5.2          , 0.55868502],  
##         [5.22222222, 0.55968787],  
##         [5.24444444, 0.56069072],  
##         [5.26666667, 0.56169357],  
##         [5.28888889, 0.56269642],  
##         [5.3          , 0.56369927],  
##         [5.32222222, 0.56470212],  
##         [5.34444444, 0.56570497],  
##         [5.36666667, 0.56670782],  
##         [5.38888889, 0.56771067],  
##         [5.4          , 0.56871352],  
##         [5.42222222, 0.56971637],  
##         [5.44444444, 0.57071922],  
##         [5.46666667, 0.57172207],  
##         [5.48888889, 0.57272492],  
##         [5.5          , 0.57372777],  
##         [5.52222222, 0.57473062],  
##         [5.54444444, 0.57573347],  
##         [5.56666667, 0.57673632],  
##         [5.58888889, 0.57773917],  
##         [5.6          , 0.57874202],  
##         [5.62222222, 0.57974487],  
##         [5.64444444, 0.58074772],  
##         [5.66666667, 0.58175057],  
##         [5.68888889, 0.58275342],  
##         [5.7          , 0.58375627],  
##         [5.72222222, 0.58475912],  
##         [5.74444444, 0.58576197],  
##         [5.76666667, 0.58676482],  
##         [5.78888889, 0.58776767],  
##         [5.8          , 0.58877052],  
##         [5.82222222, 0.58977337],  
##         [5.84444444, 0.59077622],  
##         [5.86666667, 0.59177907],  
##         [5.88888889, 0.59278192],  
##         [5.9          , 0.59378477],  
##         [5.92222222, 0.59478762],  
##         [5.94444444, 0.59579047],  
##         [5.96666667, 0.59679332],  
##         [5.98888889, 0.59779617],  
##         [6.          , 0.59879902],  
##         [6.02222222, 0.59980187],  
##         [6.04444444, 0.60080472],  
##         [6.06666667, 0.60180757],  
##         [6.08888889, 0.60281042],  
##         [6.1          , 0.60381327],  
##         [6.12222222, 0.60481612],  
##         [6.14444444, 0.60581897],  
##         [6.16666667, 0.60682182],  
##         [6.18888889, 0.60782467],  
##         [6.2          , 0.60882752],  
##         [6.22222222, 0.60983037],  
##         [6.24444444, 0.61083322],  
##         [6.26666667, 0.61183607],  
##         [6.28888889, 0.61283892],  
##         [6.3          , 0.61384177],  
##         [6.32222222, 0.61484462],  
##         [6.34444444, 0.61584747],  
##         [6.36666667, 0.61685032],  
##         [6.38888889, 0.61785317],  
##         [6.4          , 0.61885602],  
##         [6.42222222, 0.61985887],  
##         [6.44444444, 0.62086172],  
##         [6.46666667, 0.62186457],  
##         [6.48888889, 0.62286742],  
##         [6.5          , 0.62387027],  
##         [6.52222222, 0.62487312],  
##         [6.54444444, 0.62587597],  
##         [6.56666667, 0.62687882],  
##         [6.58888889, 0.62788167],  
##         [6.6          , 0.62888452],  
##         [6.62222222, 0.62988737],  
##         [6.64444444, 0.63089022],  
##         [6.66666667, 0.63189307],  
##         [6.68888889, 0.63289592],  
##         [6.7          , 0.63389877],  
##         [6.72222222, 0.63490162],  
##         [6.74444444, 0.63590447],  
##         [6.76666667, 0.63690732],  
##         [6.78888889, 0.63791017],  
##         [6.8          , 0.63891302],  
##         [6.82222222, 0.63991587],  
##         [6.84444444, 0.64091872],  
##         [6.86666667, 0.64192157],  
##         [6.88888889, 0.64292442],  
##         [6.9          , 0.64392727],  
##         [6.92222222, 0.64493012],  
##         [6.94444444, 0.64593297],  
##         [6.96666667, 0.64693582],  
##         [6.98888889, 0.64793867],  
##         [7.          , 0.64894152],  
##         [7.02222222, 0.64994437],  
##         [7.04444444, 0.65094722],  
##         [7.06666667, 0.65195007],  
##         [7.08888889, 0.65295292],  
##         [7.1          , 0.65395577],  
##         [7.12222222, 0.65495862],  
##         [7.14444444, 0.65596147],  
##         [7.16666667, 0.65696432],  
##         [7.18888889, 0.65796717],  
##         [7.2          , 0.65897002],  
##         [7.22222222, 0.65997287],  
##         [7.24444444, 0.66097572],  
##         [7.26666667, 0.66197857],  
##         [7.28888889, 0.66298142],  
##         [7.3          , 0.66398427],  
##         [7.32222222, 0.66498712],  
##         [7.34444444, 0.66598997],  
##         [7.36666667, 0.66699282],  
##         [7.38888889, 0.66799567],  
##         [7.4          , 0.66899852],  
##         [7.42222222, 0.66999137],  
##         [7.44444444, 0.67099422],  
##         [7.46666667, 0.67199707],  
##         [7.48888889, 0.67299992],  
##         [7.5          , 0.67399277],  
##         [7.52222222, 0.67499562],  
##         [7.54444444, 0.67599847],  
##         [7.56666667, 0.67699132],  
##         [7.58888889, 0.67799417],  
##         [7.6          , 0.67899702],  
##         [7.62222222, 0.67999987],  
##         [7.64444444, 0.68099272],  
##         [7.66666667, 0.68199557],  
##         [7.68888889, 0.68299842],  
##         [7.7          , 0.68399127],  
##         [7.72222222, 0.68499412],  
##         [7.74444444, 0.68599697],  
##         [7.76666667, 0.68699982],  
##         [7.78888889, 0.68799267],  
##         [7.8          , 0.68899552],  
##         [7.82222222, 0.68999837],  
##         [7.84444444, 0.69099122],  
##         [7.86666667, 0.69199407],  
##         [7.88888889, 0.69299692],  
##         [7.9          , 0.69399977],  
##         [7.92222222, 0.69499262],  
##         [7.94444444, 0.69599547],  
##         [7.96666667, 0.69699832],  
##         [7.98888889, 0.69799117],  
##         [8.          , 0.69899402],  
##         [8.02222222, 0.69999687],  
##         [8.04444444, 0.70099972],  
##         [8.06666667, 0.70199257],  
##         [8.08888889, 0.70299542],  
##         [8.1          , 0.70399827],  
##         [8.12222222, 0.70499112],  
##         [8.14444444, 0.70599397],  
##         [8.16666667, 0.70699682],  
##         [8.18888889, 0.70799967],  
##         [8.2          , 0.70899252],  
##         [8.22222222, 0.70999537],  
##         [8.24444444, 0.71099822],  
##         [8.26666667, 0.71199107],  
##         [8.28888889, 0.71299392],  
##         [8.3          , 0.71399677],  
##         [8.32222222, 0.71499962],  
##         [8.34444444, 0.71599247],  
##         [8.36666667, 0.71699532],  
##         [8.38888889, 0.71799817],  
##         [8.4          , 0.71899102],  
##         [8.42222222, 0.71999387],  
##         [8.44444444, 0.72099672],  
##         [8.46666667, 0.72199957],  
##         [8.48888889, 0.72299242],  
##         [8.5          , 0.72399527],  
##         [8.52222222, 0.72499812],  
##         [8.54444444, 0.72599097],  
##         [8.56666667, 0.72699382],  
##         [8.58888889, 0.72799667],  
##         [8.6          , 0.72899952],  
##         [8.62222222, 0.72999237],  
##         [8.64444444, 0.73099522],  
##         [8.66666667, 0.73199807],  
##         [8.68888889, 0.73299092],  
##         [8.7          , 0.73399377],  
##         [8.72222222, 0.73499662],  
##         [8.74444444, 0.73599947],  
##         [8.76666667, 0.73699232],  
##         [8.78888889, 0.73799517],  
##         [8.
```

# Using CV to Select the Tuning Parameter

- Now fit using that optimal  $\alpha$

```
lasso_best = linear_model.Lasso(lasso_mod.alpha_) #warning thrown since we are using 0, but can ignore  
lasso_best.fit(bike_data[["year", "log_km_driven"]].values, bike_data["log_selling_price"].values)
```

```
print(lasso_best.intercept_, lasso_best.coef_)
```

```
## -148.7932910778814 [ 0.0803366 -0.22686129]
```

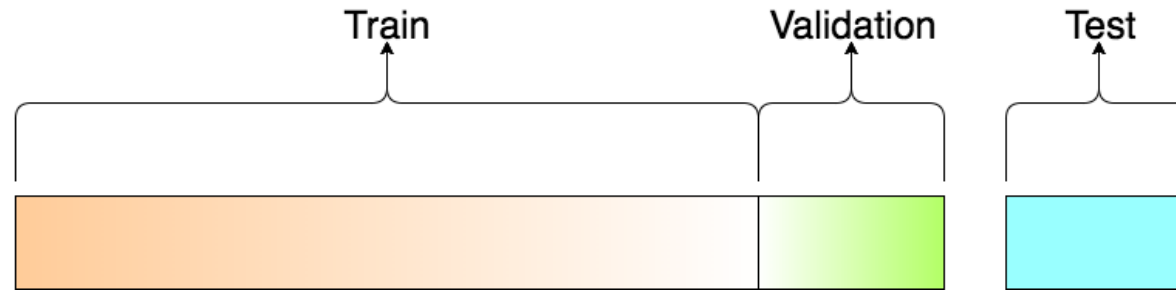
# So Do We Just Need CV?

Sometimes!

- If you are only considering one type of model, you can use just a training/test set or k-fold CV to select the best version of that model
- When you have multiple types of models to choose from, usually use both!
  - When we use the test set too much, we may have '**data leakage**'
  - Essentially we end up training our models to the test set by using it too much

# Training/Validation/Test or CV/Test

- Instead, we sometimes split into a training, validation, and test set
- CV can be used to replace the validation set!



# Training/Validation/Test or CV/Test

- Instead, we sometimes split into a training, validation, and test set
- CV can be used to replace the validation set!



- Compare only the **best** model from each model type on the test set

# Recap

- LASSO is similar to an MLR model but shrinks coefficients and may set some to 0
  - Tuning parameter must be chosen (usually by CV)
- Training/Test split gives us a way to validate our model's performance
  - CV can be used on the training set to select **tuning parameters**
  - Helps determine the 'best' model for a class of models
- With many competing model types, determine the best from each type check performance on the test set