1. Ridge Regression

   After augmentation, we would have:

$$X_{aug} = \begin{bmatrix} \sqrt{\lambda} I \\ X \end{bmatrix}, \text{ and } \; y_{aug} = \begin{bmatrix} 0 \\ y \end{bmatrix}$$

   Substitute them into the closed form solution for Ordinary Least Square (OLS):

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

   We would get:

$$\hat{\beta} = \left(X_{aug}^T X_{aug}\right)^{-1} X_{aug}^T y_{aug}$$

$$\hat{\beta} = \left(\begin{bmatrix} \sqrt{\lambda} I & X^T \end{bmatrix} \begin{bmatrix} \sqrt{\lambda} I \\ X \end{bmatrix} \right)^{-1} \begin{bmatrix} \sqrt{\lambda} I & X^T \end{bmatrix} \begin{bmatrix} 0 \\ y \end{bmatrix}$$

   Multiply out:

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

   This is exactly the closed form solution for Ridge regression.

   Therefore, the ridge regression estimates can be obtained by ordinary least squares regression on an augmented data set.

2. Predicting Appliance Energy usage using Linear Regression

a) I preprocessed the data using standardization (Z-score normalization) with each feature's mean and variance from the training set.

I choose to use standardization because it can transform each feature to nearly a normal distribution with a mean of 0 and a standard deviation of 1. Scaling the features into a certain range would improve numerical stability of the model. Z-score normalization specifically is robust to outliers, meaning the scaling process is less affected by outliers in the feature.

In addition, I fit the scaler on the train set and use the same scaler to transform train, val, and test sets. Only fitting on the train set makes sure my scaler is not prone to data leakage. And transform all data sets using the same scaler makes the data comparable across different sets.

e)

RMSE and R^2 of Linear Regression Models Trained on Energy Data Train Set and Train+Val Set Separately

	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
lr_train	98.231038	0.18675401	97.539866	0.00267901	572.15434	-38.6418
lr_train+val	99.383971	0.16755192	88.707679	0.17511562	246.93778	-6.38418

In terms of RMSE and R^2, lr_train performs slightly better than lr_train+val on the training set. In comparison, lr_train+val performs much better than lr_train on both the validation set and the testing set.

The numbers suggest that by training the model on both training and validation sets, although we compromise the model's performance on the training set, we would actually get better performances on both validation and testing sets, meaning that training on the training and validation sets helps the model to generalize and avoid over fitting.

h)

RMSE and R^2 of Ridge Model Trained on Energy Data Train Set with Different Alpha

alpha	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
0.00001	98.23104	0.186754	97.53987	0.002679	572.1541	-38.6418
0.0001	98.23104	0.186754	97.53987	0.002679	572.1519	-38.6415
0.001	98.23104	0.186754	97.53994	0.002678	572.1304	-38.6385
0.01	98.23104	0.186754	97.54057	0.002665	571.9217	-38.6096
0.1	98.23104	0.186754	97.54695	0.002534	570.51	-38.4143

1	98.23143	0.186748	97.60955	0.001254	611.4299	-44.271
10	98.24434	0.186534	98.03467	-0.00747	1525.621	-280.851
100	98.34669	0.184838	97.88723	-0.00444	2009.34	-487.916
1000	99.46426	0.166206	94.70484	0.059811	505.8422	-29.9854
10000	103.1439	0.103373	94.91782	0.055578	1217.489	-178.497
100000	107.3773	0.028262	96.78143	0.018128	251.3395	-6.64977
1000000	108.7198	0.00381	97.51996	0.003086	93.11746	-0.05

Based on the validation data performance, I would choose 1000 as the optimal alpha for Ridge model, since alpha=1000 gives the lowest RMSE and highest R^2.

RMSE and R^2 of Lasso Model Trained on Energy Data Train Set with Different Alpha

alpha	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
0.1	98.28591	0.185845	98.50861	-0.01723	2337.527	-660.668
0.5	98.48888	0.182479	98.02288	-0.00722	1109.911	-148.177
1	98.84195	0.176607	96.53104	0.023202	484.3065	-27.4032
2	99.61242	0.16372	94.53612	0.063158	433.8377	-21.792
3	100.7321	0.144815	93.80706	0.077552	339.4507	-12.9534
4	101.6625	0.128945	94.06882	0.072397	394.681	-17.8634
5	102.7109	0.110886	94.71462	0.059617	444.6545	-22.9427
6	103.6439	0.09466	95.47645	0.044428	465.7287	-25.2659
10	104.8402	0.07364	96.2423	0.029037	315.0001	-11.0157
20	107.0081	0.034932	97.26116	0.00837	102.0177	-0.26031
30	108.9276	0	97.68449	-0.00028	90.88704	-0.0003
40	108.9276	0	97.68449	-0.00028	90.88704	-0.0003
50	108.9276	0	97.68449	-0.00028	90.88704	-0.0003
100	108.9276	0	97.68449	-0.00028	90.88704	-0.0003

Based on the validation set performances, I would choose 3 as the optimal alpha for Lasso model, since alpha=3 gives the least RMSE and largest R^2.

j)

RMSE and R^2 of Ridge and Lasso Trained on Train+Val Sets with Optimal Alphas Respectively

model	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
Ridge	99.9558	0.157945	89.63219	0.157832	974.2052	-113.929
Lasso	101.5296	0.13122	91.67381	0.11903	454.148	-23.9759

Take the rows of optimal alphas from 2h) as a reference:

Ridge:

Alpha	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
1000	99.46426	0.166206	94.70484	0.059811	505.8422	-29.9854

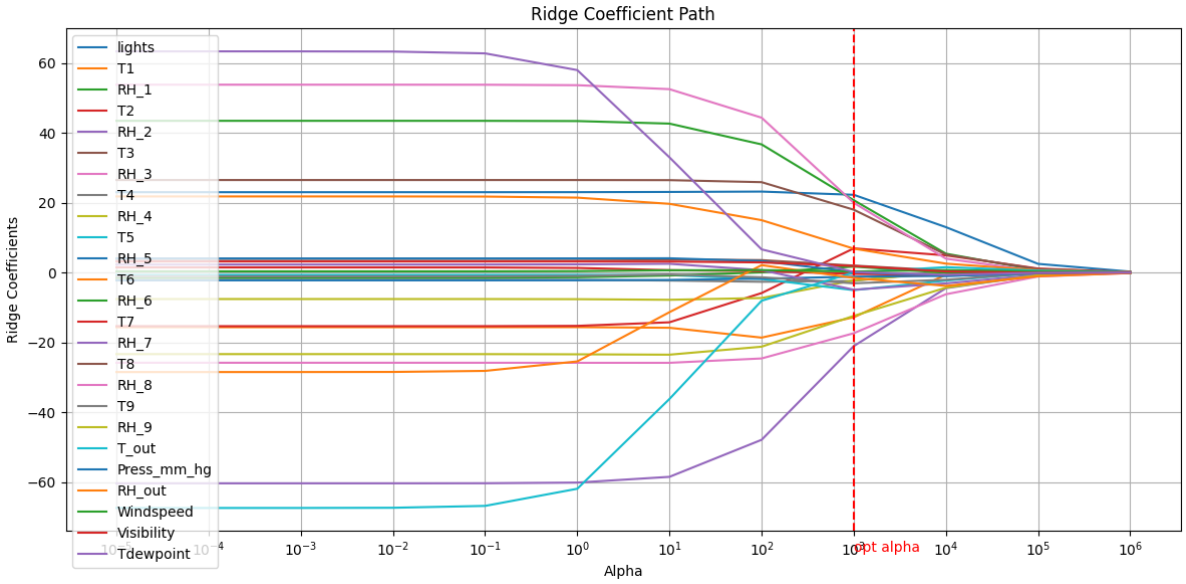
Lasso:

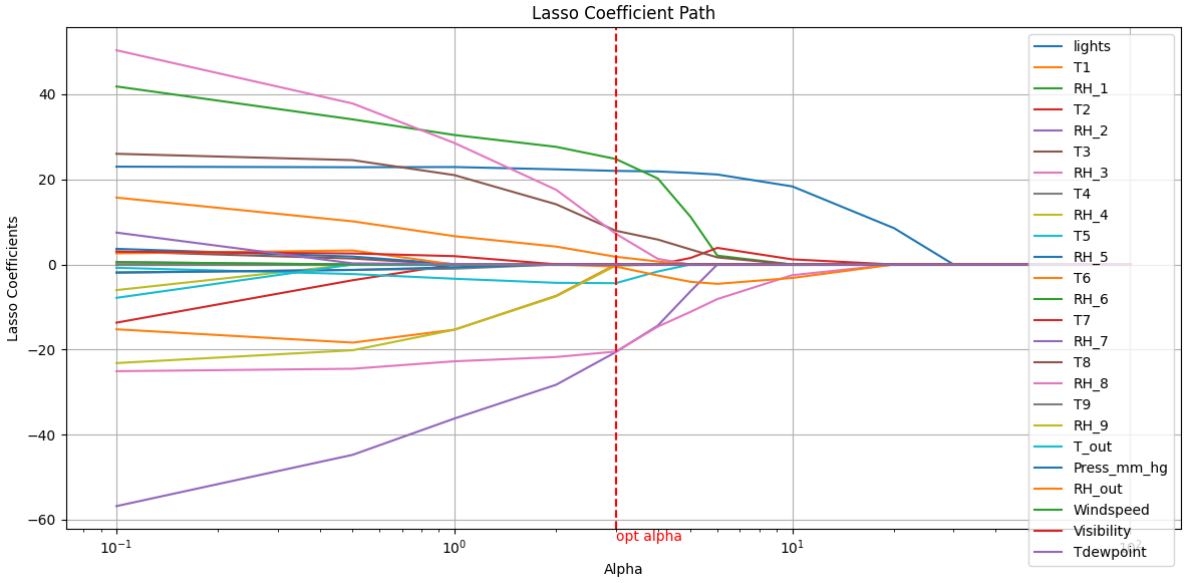
Alpha	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
3	100.7321	0.144815	93.80706	0.077552	339.4507	-12.9534

By comparing the performances, we can see that when trained on train+val sets with

their optimal alphas respectively, both Ridge and Lasso performs slightly better on validation set than models trained only on train set. However, they both perform worse on train set and test set than the models trained only on train set. This could suggest that our models do not generalize very well when trained on train+val sets, or the optimal alphas are not universal and might be changing over different training data.

k)





l) The 1st observation is that the selection range of alpha values for Ridge is much larger than that of Lasso. All coefficients of Ridge shrink to 0 when alpha is near 10^6. In comparison, all coefficients of Lasso shrink to 0 before alpha reaches 100.

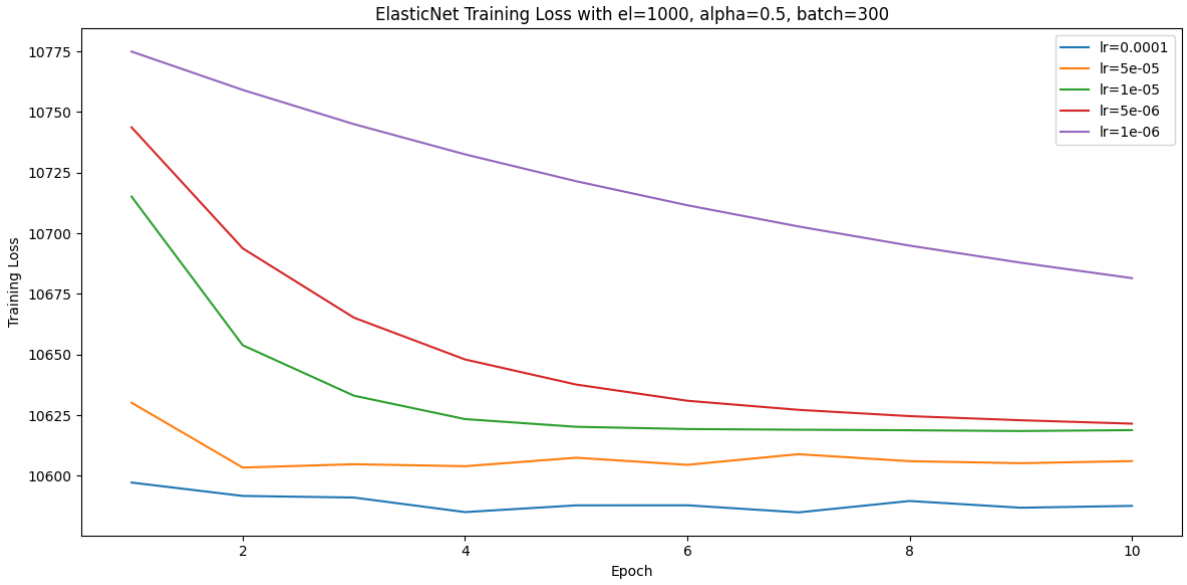
The 2nd observation is that all models used could not fit the data very well. Even when

predicting on the train set, the largest R^2 we could get through out the experiment is around 0.19. And R^2 always stay negative when predicting on the test set, regardless of the model and training sample size.

The 3rd observation is that in both Ridge and Lasso, when alpha values are large enough to drive all coefficients to nearly 0, the models' performances on the testing set actually become better.

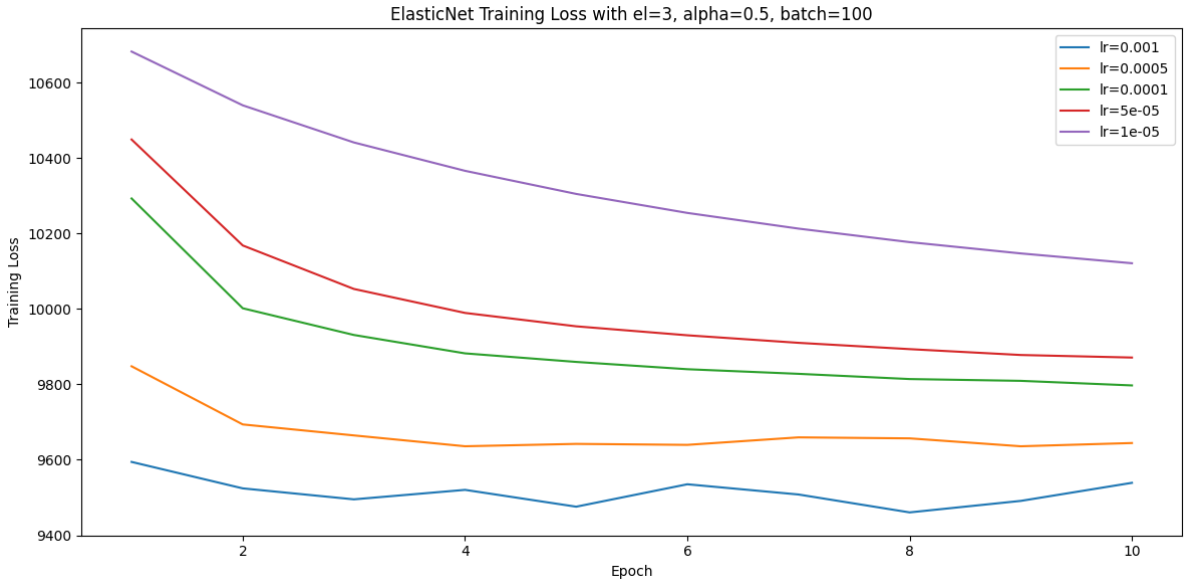
3. Predicting Appliance Energy Usage using SGD

f) For the optimal regularization parameter of Ridge (1000), the training loss for 10 epochs is like this:



From the figure, it seems that the best learning rate among the tested learning rates is lr = 0.000005

For the optimal regularization parameter of Lasso (3), the training loss for 10 epochs is like this:



From the figure, the best learning rate among the tested learning rates seems to be lr = 0.0001

Train two elastic net models with different regularization parameters and their optimal

learning rates respectively. Here are their performances:

ElasticNet with Ridge and Lass Optimal λ and lr respectively Trained on Train Set

model	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
el=1000, lr=0.000005	144.893403	-0.769382	128.90932	-0.741964	350.674714	-13.891412
el=3, lr=0.0001	139.85688	-0.648512	137.20182	-0.973288	684.179514	-55.684865

g) Use Ridge's optimal λ and lr:

ElasticNet (el=1000, lr=0.000005) with Different Alphas Trained on Trained Set

alpha	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
0	144.0063	-0.74778	127.3354	-0.69969	403.2164	-18.6881
0.1	144.2364	-0.75337	127.6451	-0.70797	395.9263	-17.9826
0.2	144.4413	-0.75836	128.2386	-0.72389	381.7131	-16.6442
0.3	144.6156	-0.7626	128.6285	-0.73438	321.112	-11.4865
0.4	144.7677	-0.76631	128.6376	-0.73463	329.9937	-12.1868
0.5	144.8939	-0.76939	128.9511	-0.74309	361.4404	-14.8198
0.6	145.0099	-0.77223	129.2186	-0.75033	372.81	-15.8307
0.7	145.1107	-0.77469	129.3967	-0.75516	388.6202	-17.2885
0.8	145.1941	-0.77673	129.6249	-0.76136	418.7469	-20.2339
0.9	145.2747	-0.77871	129.7272	-0.76414	417.8569	-20.1438
1	145.3406	-0.78032	130.0442	-0.77277	431.78	-21.5763

Use Lasso's optimal λ and lr:

ElasticNet (el=3, lr=0.0001) with Different Alphas Trained on Trained Set

alpha	train-rmse	train-r2	val-rmse	val-r2	test-rmse	test-r2
0	139.7894	-0.64692	126.9356	-0.68903	298.4667	-9.78745
0.1	139.7152	-0.64517	133.9045	-0.87958	454.2606	-23.9883
0.2	139.7902	-0.64694	133.2873	-0.86229	283.1666	-8.70982
0.3	139.8007	-0.64719	129.0654	-0.74619	436.1353	-22.034
0.4	139.9931	-0.65173	124.048	-0.61306	451.859	-23.7248
0.5	139.8892	-0.64927	127.6655	-0.70851	507.9788	-30.2477
0.6	140.0167	-0.65228	129.2612	-0.75149	542.9751	-34.7015
0.7	139.8948	-0.64941	134.1067	-0.88526	623.7377	-46.1119
0.8	139.9421	-0.65052	131.4369	-0.81095	747.7493	-66.7079
0.9	140.0179	-0.65231	126.2668	-0.67128	784.4059	-73.509
1	140.0554	-0.65319	133.7481	-0.87519	850.4237	-86.5785

h) From the performance data of all models mentioned above trained on training data only, it seems that when comparing performances on different data sets among train, val, and test sets, different models would show the best results. When comparing on val set, the Lasso model with alpha=3 seems to have the best RMSE and R^2. However, when comparing on test set, the ElasticNet with el=3 and lr=0.0001 seems to have the

best RMSE and R^2. In terms of which model is the best, I would prefer the performances on the test set, which would suggest that the ElasticNet with el=3 and lr=0.0001 is the best.

The differences between the SGD-variants of Ridge and LASSO and the standard implementations:

The major difference would be how the coefficients are updated. In the standard implementation, the coefficients are updated using the full training dataset at each iteration. In comparison, in SGD-variants, each time the coefficients are updated, a random sample of the training dataset is selected to calculate the gradient. The SGD-variants tend to converge faster than the standard implementations on large datasets, but we may witness an oscillation of the loss in the training process.

i) From the performances shown above, the best performance on test set belongs to ElasticNet with el=3, lr=0.0001, and alpha=0.2. The best performance on validation set belongs to ElasticNet with el=3, lr=0.0001, and alpha=0.4. Here are the final coefficients of the two models trained on training set with batch=100, epoch=50:

	test_coef	val_coef
lights	23.39723	22.89743
T1	-18.8448	-19.2191
RH_1	35.25339	30.79319
T2	-3.03167	-0.10362
RH_2	-45.0693	-39.1963
T3	25.97866	24.16329
RH_3	42.4887	35.48758
T4	-2.3336	-2.72269
RH_4	-6.5004	-6.23421
T5	-1.56845	-3.46254
RH_5	-1.61488	-1.82661
T6	13.79435	11.63557
RH_6	0.350537	0.063813
T7	0.19838	0.611446
RH_7	0.324953	-1.93539
T8	3.535449	3.314967
RH_8	-24.0354	-23.4506
T9	-1.4031	-1.97262
RH_9	-20.3902	-18.909
T_out	-4.18393	-2.84908

Press_mm_hg	3.04417	2.47971
RH_out	3.695809	1.705949
Windspeed	0.339317	0.268016
Visibility	2.662689	2.395437
Tdewpoint	4.149386	2.206297

For those coefficients with noticeable differences (e.g. RH_1, T2, RH_2, RH_3, etc.), the test_coef often shows a larger magnitude. Also, the model of test_coef has better performance on test set than the model of val_coef. These features might have stronger correlations with the label values in test set. Therefore, the model of test_coef assign those features with larger importance and thus have better performance on test set.