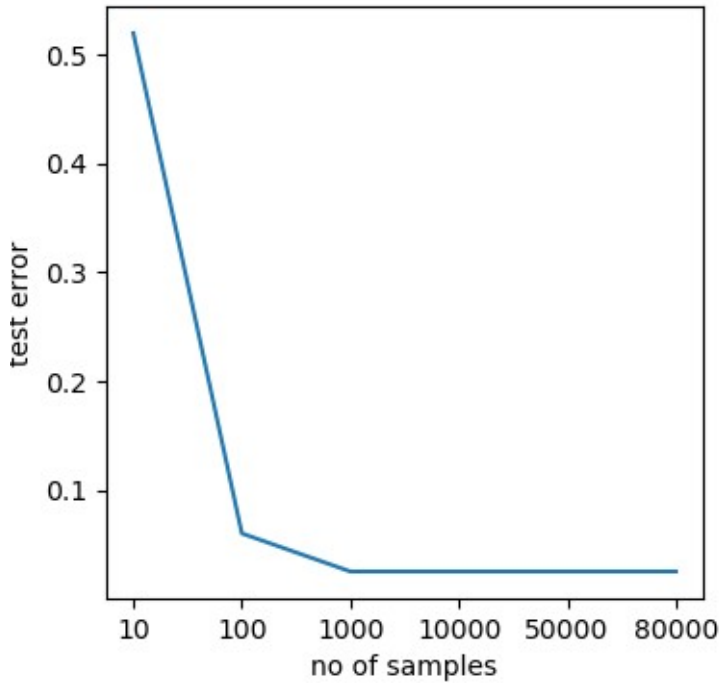
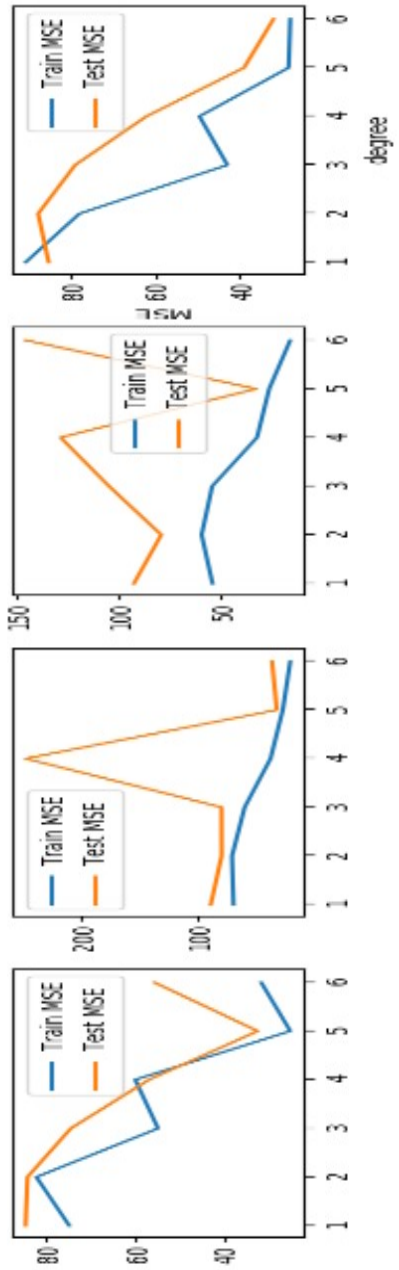
2.1
One vs One:
It can obtain better accuracy on average than one vs rest if we use more number of features because, more features help in clearly classifying into binary data. We have to make nC2 classifiers and train them and then combine the results of each of the classifiers but since we train on smaller sets of data, the overall computation overhead is unpredictable and depends on data. It should be used when there are enough samples and features for each class so as to classify the data binarily(such as a digits dataset) and it also provides us the option to make use of different features in data when classifying one class against different classes. It also helps in case the data has multiple features present of each class because one vs rest classification is less accurate in that case because, the one vs rest classifier wouldn't be able to correctly learn the classification from the data.
One vs Rest:
We have to make O(n) classifiers but work on larger data sets. It can be used with data sets with large number of features and reduced number of samples since, it helps to classify data into multiple classes at once. But it is more prone to imbalance in dataset(where we have many data points for one class and less data points for other classes) because, the classification into the classes that have less no of samples will have more error in this case.

3.1
a) Bias increases and variance decreases because as lambda increases, more and more weights go to zero which means we move away from the true function(unless it's a zero function, i.e. f(x)=0) hence bias increases and predicted values deviate less from their mean(since less no of features are used) which reduces variance.
b) Bias increases(slightly) but variance decreases because, by training on more data, the probability of predicting wrongly decreases which reduces variance, on the other hand the bias which is the calculated as the no of mismatches in this case increases, increasing the bias.
c) Adding more features which are linear combinations of original features doesn't affect the bias or the variance as lasso makes it's coefficient zero anyways (since it's mentioned in the explanation of LASSO in the assignment that, it mitigates multi-collinearity by reducing their weights to 0 and weights that are non-zero signify they are important). It's because it doesn't matter as adding such a column indirectly means changing new weights of the involved columns. So even if the weight of the new feature is not zero, the weights of the involved columns may change and their final effective weight maybe same as before. So it doesn't have any effect on the expressiveness of model as the complexity and the final weights remain the same. Hence, bias and variance are unchanged.
3.2



Bias increases(slightly) but variance decreases because, by training on more data, the probability of predicting wrongly decreases which reduces variance, on the other hand the bias which is the calculated as the no of mismatches in this case increases, increasing the bias. But, the total error decreases, since the decrease in variance is more than the increase in bias.



Theoretically, we obtain least training error with degree 6(because 6 is the maximum degree we are considering here).

For test data:
As the degree increases, the bias decreases and variance increases. But upto the expected degree at which we obtain least error, the bias decreases more as we go towards the true function and variance increases(but less comapared to decrease because we are at lower degrees) as expected because of increase in degree. As we cross the optimal degree the increase in variance overshoots the decrease in bias because, bias is almost at it’s lowest and for higher degree, variance increases very fast.

For train data:
The error is expected to decrease with increasing degree as the more complexity the model has, the more it can accomodate the exact given y values of the data points (due to overfitting).

~~60+~~

1.1 Let w be the weight vector and let $w_i \sim \text{laplace}(0, b)$

$\Rightarrow p(w|b) \propto e^{-\frac{1}{2b}\sum_i |w_i|}$

In linear regression, we assume $Y \sim \mathcal{N}(Xw, \sigma^2 I)$

$P(Y|X, w, \sigma^2) \propto (\sigma^2)^{-\frac{n}{2}} e^{\left(-\frac{1}{2\sigma^2}(y-Xw)^T(y-Xw)\right)}$

$\log(P(Y|X,w,\sigma^2)) \propto -n\log\sigma - \frac{1}{2\sigma^2}(y-Xw)^T(y-Xw)$

$\log(p(w|b)) \propto -\frac{1}{2b}\sum|w_i|$

$\log(P(Y|X,w,\sigma^2)\,P(w|b)) = \log(k \cdot (\sigma^2)^{-\frac{n}{2}} e^{-\left(\frac{1}{2\sigma^2}\right)}$

$= \log(P(Y|w,X,\sigma^2)) + \log(P(w|b))$

$= K_1\left(-n\log\sigma - \frac{1}{2\sigma^2}(y-Xw)^T(y-Xw)\right) + K_2 \cdot \left(-\frac{1}{2b}\sum|w_i|\right)$

For MAP estimate, we find w such that $\log(P(Y|X,w,\sigma^2)\,P(w|b))$ is max ($\because$ posterior $= \frac{P(Y|X,w,\sigma^2)}{P(w|b)}$)

$\Rightarrow \underset{w}{\text{argmax}}\left(K_1 \cdot \left(-n\log\sigma - \frac{1}{2\sigma^2}(y-Xw)^T(y-Xw)\right) + K_2 \cdot \left(-\frac{1}{2b}\sum|w_i|\right)\right)$

(and we know that maximizing $\log(P)$ is same maximizing P)

$= \underset{w}{\text{argmax}}\left(-K_1(y-Xw)^T(y-Xw) - K_2 \cdot \frac{1}{2b}\sum|w_i|\right)$

$= \underset{w}{\text{argmax}} \left(-(y-Xw)^T(y-Xw) - \frac{k_2}{k_1} \cdot \frac{1}{2L} \sum |w_i| \right)$

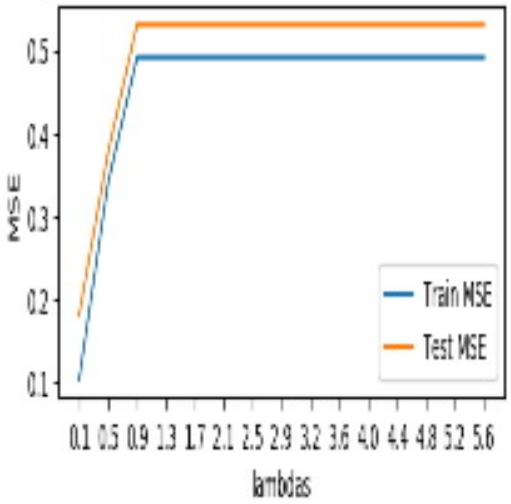
$= \underset{w}{\text{argmin}} \left(\|y-Xw\|^2 + \lambda \sum |w_i| \right)$ (combining all constants to λ)

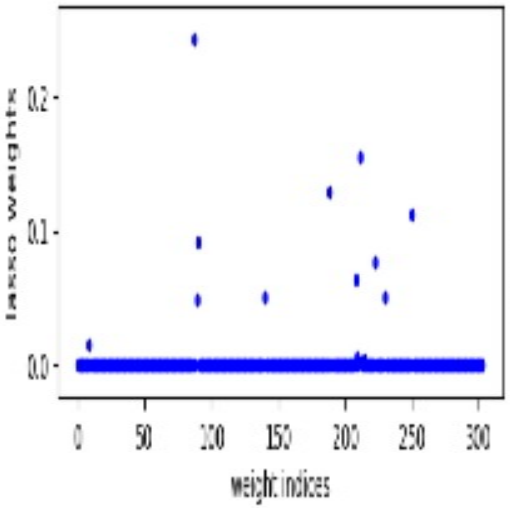
$= \underset{w}{\text{argmin}} \left(\|y-Xw\|^2 + \lambda \|w\|_1 \right)$ = Lasso regression

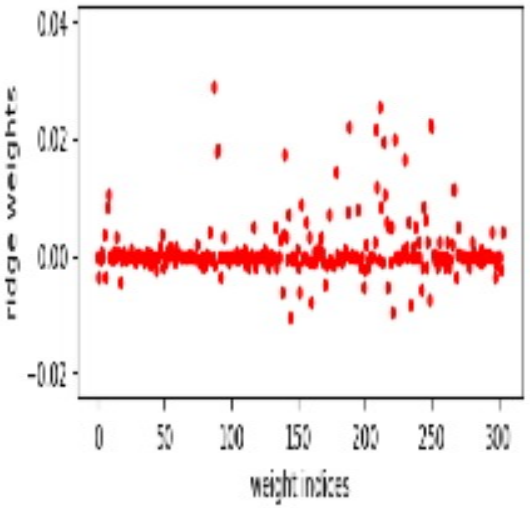
∴ Lasso regression is same as linear regression with Laplacian prior on w

1.2 b) $\lambda = 0.1$ is optimal because, it both the least test and train errors. In the plot, both test and train ~~data~~ errors increase with λ. This is due to more and more weights going to zero, making the model underfit. As λ increases even further, all the weights go to 0 and thus, the error remains constant (after ~~+3~~ $\lambda = 1.3$).

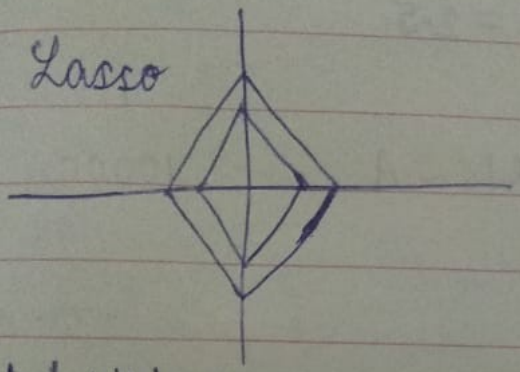
c) In Lasso, almost all weights are zero and those that are non-zeros have more value than ridge weights. In Ridge, weights have small values, but most of them are non-zero.
This is because of the nature of the penalising terms. In Lasso non-zero values are greater than ridge non-zero values, because, ridge penalises weights more heavily.


MSE
0.5
0.4
0.3
0.2
0.1
Train MSE
Test MSE
0.1 0.5 0.9 1.3 1.7 2.1 2.5 2.9 3.2 3.6 4.0 4.4 4.8 5.2 5.6
lambdas


lasso weights
0.2
0.1
0.0
0
50
100
150
200
250
300
weight indices

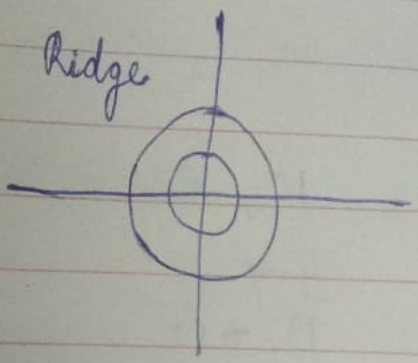

ridge weights
0.04
0.02
0.00
-0.02
0
50
100
150
200
250
300
weight indices

Lasso



$|a| + |b| = c$

Ridge



$a^2 + b^2 = c$

The more zero values in lasso can be explained by the graphs, where, in lasso, weights tend to accumulate at the corners making most of them zero, whereas, there is no such tendency in ridge graph, which is why most are non-zero.