Farhan Rahman
Homework Week5
NYU ID: fr2119

cccc

1 *Considering the following expression we can represent it as a sum of two functions. So we have the following*

$$L(w) = \frac{1}{2}||x - w|\,|_2^2 + \lambda||w|\,|_1$$

We can represent the above sum as,

$$(RSS(w))^{lasso} = RSS(w)^{ols} + \lambda||w|\,|_1$$

now to find the optimal weight we can use various methods but one such method is applying basic calculus to find the minima of the given function.For that both the functions $RSS(w)^{ols}$ *and* $\lambda||w|\,|_1$ *should be partially differentiated and equated to zero, to find the optimal solution. So differentiating partially w.r.t, we get ,*

$$\frac{\partial RSS(w)^{ols}}{\partial w} = 2\frac{1}{2}||x - w|\,|$$

Now for differentiating the L_1 *norm we have an ambiguity, The problem with this term is that the derivative of the absolute function is undefined at =0. The method of coordinate descent makes use of two techniques which are to perform coordinate-wise optimization, which means that at each step only one feature is considered and all others are treated as constants make use of sub derivatives and sub differentials which are extensions of the notions of derivative for non differentiable functions.The combination of these two points is important because in general, the sub differential approach to the Lasso regression does not have a closed form solution in the multivariate case. Except for the special case of orthogonal features which is discussed here. As we did previously for the OLS term, the coordinate descent allows us to isolate the* w_**:*

$$\lambda \sum_{0}^{N} w_j = \lambda|w_j| + \lambda \sum_{k \neq j}^{n} |w_k|$$

And optimizing this equation as a function of j reduces it to a univariate problem.Using the definition of the subdifferential as a non empty, closed interval [a,b] where a and b are the one sided limits of the derivative we get the following equation for the subdifferential, $\frac{\partial(\lambda|w|_1)}{\partial w}$ *On solving we get the following equations,*

$$\frac{\partial(\lambda \sum_0^n w_j)}{\partial w} =$$

$-\lambda$ *if* $w_j < 0$, $[-\lambda, \lambda]$ *if* $w_j = 0$ *and* λ *if* $w_j > 0$ *Returning to the complete Lasso cost function which is convex and non differentiable (as both the OLS and the absolute function are convex).Putting it all together we get*

$$\frac{\partial RSS(w)^{lasso}}{\partial w} = ||x - w|\,| + \frac{\partial(\lambda|w|_1)}{\partial w}$$

Computing the subdifferential of the Lasso cost function and equating to zero to find the mnimum:

$$0 = x_j - w_j - \lambda$$

if $w_j < 0$

$$[x_j - \lambda, x_j - \lambda]$$

if $w_j = 0$

$$0 = x_j - w_j + \lambda$$

if $w_j > 0$

For the second case we must ensure that the closed interval contains the zero so that j=0 is a global minimum

$$0 \in [x_j - \lambda, x_j - \lambda]$$
$$x_j - \lambda \leq 0$$
$$x_j + \lambda \geq 0$$
$$-\lambda \leq x_j \leq \lambda$$

Solving for $w_j = w_*$ *using the above obtained results, we get*

$$w_* =$$

$x_j + \lambda$ *if* $xj < -\lambda$; 0 *if* $-\lambda \leq x_j \leq \lambda$; $x_j + \lambda$ *if* $xj > \lambda$

2 a

for this we can select $|w_0|$*, Simply records every entry.*

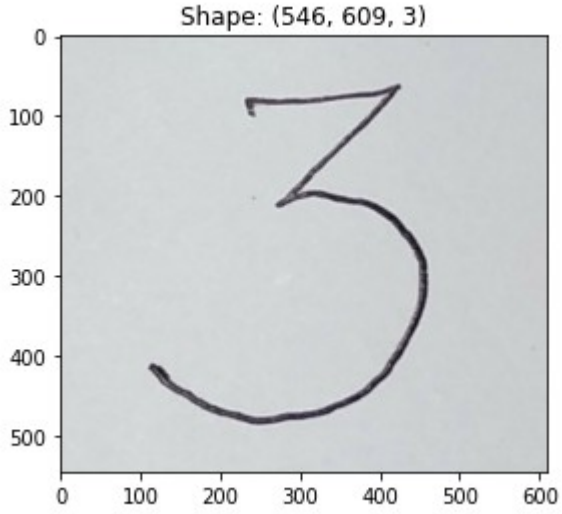
b *For sparse w , we can select* R_w *as lasso or* $|w|_1$ *as it is a model of choice when the number of features are high, Since it provides sparse solutions. We can get computational advantage as the features with zero coefficients can simply be ignored.*

c *for the coefficients of w to be very small or vary slowly over average we can use ,*$||\ w|\ |_2$ *regularization can address the multicollinearity problem by constraining the coefficient norm and keeping all the variables. L2 regression can be used to estimate the predictor importance and penalize predictors that are not important. One issue with co-linearity is that the variance of the parameter estimate is huge.Having the weights to be small gives us an advantage of using all the features without bias and giving equal weights to all the models leads to better regularization.*

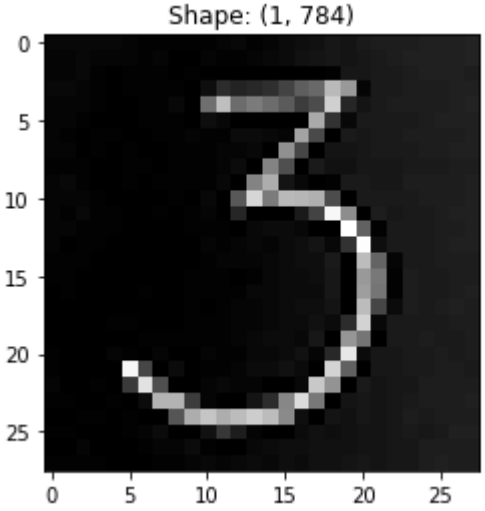
d

e *As it is given that we cannot have negative values of w,so need to find a function which only takes positive values of w , which can be* $R(w) = log(w)$ *. This ensures we only have positive entry of w and no neagative values is allowed.*

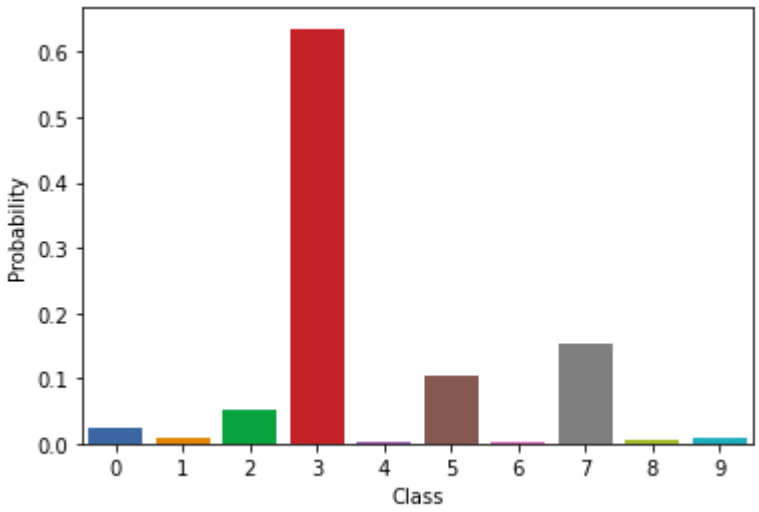
•The visualization of your test image (before pre-processing)



•The visualization of your test image (after pre-processing. Don't show the images from intermediate pre-processing stages.)



•The bar plot showing the conditional probabilities per class for your test image

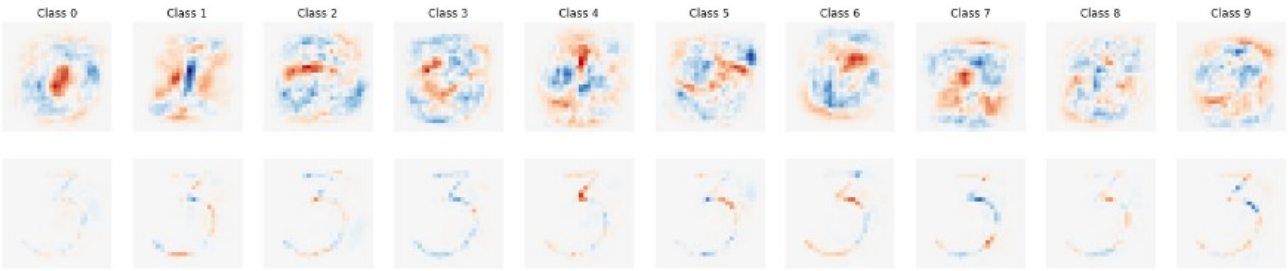


•The predicted class label for your test image

```
[ ]  test_pred = clf.predict(test_sample)
     print("Predicted class is: ", test_pred)
```

```
Predicted class is:  ['3']
```

•The figure from the "Explain the model prediction" section



In your own words, list the classes for which the logistic regression predicted a high or moderately high probability. Using the figure from the "explain the model prediction" section, explain *why* the logistic regression estimates that these classes are very likely or moderately likely.

Solution:

In my case of hand written image which is ‘3’ , from the above figure we can make out that it gives high probabilities for 2, 5, 7 and gives moderate probabilities for 0 and low for the rest of the digits. The blue color appears if there is a positive multiplication of test image writing in the part of the image and it is positively associated with belonging to the class. So as we know that many of the edge features and curves of 2,5 and 7 resemble the features of ‘3’ so we get blue or similarity between the classes whereas red occurs when there is negative association and the written part of test image does not match with the corresponding classes. If we take the case of ‘4’ , we know it has sharp features where as we expect ‘3’ to have smooth curves so it shows red color.