CS 335 & CS 337

Adarsh Raj - 190050004

Assignment-2

September 2021

Question 1 - Perceptron

Theory - 1.1

1. One-vs-One strategy splits a multi-class classification into one binary classification problem per each pair of classes, whereas, One-vs-Rest strategy splits a multi-class classification into one binary classification problem per class.

   In one-vs-one strategy, we will have different weight vectors for each possible pair of classes, i.e, $\binom{n}{2}$ weight vectors for n classes. For every datapoint, it will be assigned the class which gets maximum number of `votes` across all pairs of classes. Whereas, for one-vs-rest perceptron, we will be having only n weight vectors corresponding to each class with all other classes combined.

   Advantages of 1-vs-1: Since, 1-vs-1 classifier, overall makes more number of classifications to predict, hence is more robust. Also, single classifier in 1-vs-1 uses a subset of data, so single classifier is faster for 1-vs-1 strategy.

   Disadvantages of 1-vs-1: It will require more time than 1-vs-rest classifier because of more weight vectors, and also it may lead to overfitting of dataset. As, 1-vs-rest perceptron approach is faster overall, it is usually preferred.

2. Given, a datapoint (f, y), which got misclassified as y'.

   Let, the old weight corresponding to class y be w_y, and old weight corresponding to class y' be $w_{y'}$.

   After update, the corresponding weights according to the perceptron update rule given are $(w_y + f)$ for y, and $(w_{y'} - f)$ for y'. Now, the corresponding previous and updated scores will be, as follows:

$$score_{prev}(f, y) = \sum_i f_i w_{y_i} = w_y^T f$$
$$score_{prev}(f, y') = \sum_i f_i w_{y'_i} = {w_{y'}}^T f$$

$$score_{update}(f, y) = (w_y + f)^T f = w_y^T f + f^T f = w_y^T f + ||f||^2$$
$$score_{update}(f, y') = (w_{y'} - f)^T f = {w_{y'}}^T f - ||f||^2$$

   From the above equations, we have:

$$score_{update}(f, y) - score_{prev}(f, y) = ||f||^2 > 0$$
$$score_{update}(f, y) - score_{prev}(f, y) = -||f||^2 < 0$$

   Hence, after the update step of given perceptron startegy, the score value for (f, y) increases and score value for (f, y') decreases.

3. Given, a 2-class dataset with $y \in \{+1, -1\}$, and loss function for the dataset as $\mathcal{L}(f, y) = \max(0, -yf^T w)$.

   Now,

$$\frac{\partial \mathcal{L}}{\partial w} = 0, \; if \; \mathcal{L} = 0$$

$$\frac{\partial \mathcal{L}}{\partial w} = -yf, \; if \; \mathcal{L} = -yf^T w$$

   Hence, the total loss function will be:

$$\mathcal{L}_w(f, y) = \sum_{yf_i^T w < 0} -yf_i$$

   Gradient Descent Update Rule (learning parameter - η):

$$w_{k+1} := w_k - \eta \frac{\partial \mathcal{L}_w(f, y)}{\partial w} = w_k + \eta \sum_{yf_i^T w_k < 0} yf_i$$

   Now, we can follow the outline of the perceptron convergence proof from the class (using same notations and assumptions), also, taking the number of misclassified points for kth iteration as $|M_k|$, we have:

$$w_{k+1}^T u = w_k^T u + \eta \sum_{yf_i^T w < 0} yf_i^T u \geq w_k u^T + \eta |M_k| \gamma$$

   Also,

$$||w_{k+1}||^2 = \big(w_k^T + \eta \sum_{yf_i^T w < 0} yf_i^T\big)\big(w_k + \eta \sum_{yf_i^T w < 0} yf_i\big)$$

$$||w_{k+1}||^2 = ||w_k||^2 + 2w_k^T(\eta \sum_{yf_i^T w < 0} yf_i^T) + ||\eta \sum_{yf_i^T w < 0} yf_i^T||^2$$

$$||w_{k+1}||^2 \leq ||w_k||^2 + \eta^2 |M_k|^2$$

   Combining above inequalities similar to class, we have:

$$w_{k+1}^T u \geq \sum_{i=1}^{k} \eta \gamma |M_i|$$

$$||w_{k+1}||^2 \leq \eta^2 \sum_{i=1}^{k} |M_i|^2$$

   Algorithm Termination: Algorithm will terminate at step j, when $|M_j| = 0$, i.e there are no more misclassified points. Also, the summation of $|M_i|$ cannot exceed, Mk, where M is the number of datapoints. Also the update

happens becuase of atleast one misclassified point, hence, $|M_i| \geq 1$. Hence the inequality can be written as:

$$\eta M\sqrt{k} \geq ||w_{k+1}|| \geq \eta\gamma \sum_{i=1}^{k} |M_i| \geq \eta\gamma k$$

$$k \leq \frac{M^2}{\gamma^2}$$

Therefore, we have proved that there is an upper bound on number of iterations, for linearly separable dataset. Hence, the gradient descent classification converges to a solution.

4. Given, the the upper bound on number of iterations required for convergence of perceptron is M, and the modified update rule as:

$$w_y = w_y + 0.5f$$
$$w_{y'} = w_{y'} - 0.5f$$

Now, if we initialise the weight parameters at zero, the upper bound will not change. This is because even though the parameters are scaled by a factor of 0.5 or any other λ, the sign of the of the difference in scores, will not be affected. Hence the number of iterations will be bounded by M, only.

For different initialization of w, the number of iterations will be dependent on the number of iterations needed to change the sign of difference of score of previous and updated datapoints. Hence, we cannot conclude a value for upper bound in this case.

5. Given, AND dataset and the AND Function, to find the upper bound on perceptron iterations, we have to consider the best separating line for the dataset. It can be seen that for data points as given in question, the best separating line will be $f_1 + f_2 - 1.5 = 0$ or $x + y - 1.5 = 0$.

Incorporating the above, and also adding a new feature $f_3 = 1$, corresponding to bias, we have the best separating line as $\sqrt{\frac{4}{17}}(f_1 + f_2 - 1.5f_3) = 0$. Hence, the best separating unit vector u can be represented as $u = \sqrt{\frac{4}{17}}(1, 1, -1.5)$.

$sign(u^T f) < 0$, for class = 0
$sign(u^T f) > 0$, for class = 1

It can be concluded that, the above holds for all datapoints f. Now, if we change the ids of classes, similar to binary classification as done in class, we will have $y = -1$ corresponding to class 0 and $y = 1$, for class 1. Hence,

$$y = sign(u^T f)$$

Now, the minimum value of $|u^T f|$ is obtained at $f = (1, 1, 1)$, (can be deduced easily), and hence $|u^T f| \geq \frac{2}{\sqrt{17}}(1 + 1 - 1.5) = \frac{1}{\sqrt{17}}$.

Following the notations of class, we have:

$$|u^T f| \geq \gamma = \frac{1}{\sqrt{17}}$$

Also, since the farthest data point is $(1, 1, 1)$, the datapoints lie inside the sphere of radius r, i.e, $||f|| \leq r = \sqrt{3}$.

Following the proof outline provided in slides we have, with zero initialisation, k updates, and using Cauchy inequality, we have the following:

$$kr^2 \geq ||w||_2^2 \geq (w^T u)^2 \geq (k\gamma)^2$$

$$k \leq \frac{r^2}{\gamma^2}$$

Now, using $r = \sqrt{3}$ and $\gamma = \frac{1}{\sqrt{17}}$, we have:

$$k \leq \frac{(\sqrt{3})^2}{(\frac{1}{\sqrt{17}})^2} = 3 * 17 = 51$$

$$k \leq 51$$

Hence, the upper bound on the number of iterations required by Perceptron algorithm for AND dataset is **51**.

Lab - 1.2

Refer to `assignment_2.ipynb` file for function `perceptron_algo()`.

A classification report is added below for the given dataset.

	precision	recall	f1-score	support
class 0	1.00	1.00	1.00	12
class 1	1.00	1.00	1.00	17
class 2	1.00	1.00	1.00	11
accuracy			1.00	40
macro avg	1.00	1.00	1.00	40
weighted avg	1.00	1.00	1.00	40

```
array([[ 9.65183993, 10.27107589],
       [-7.76125707, -1.09046762],
       [-1.89058286, -9.18060827]])
```

Figure 1: Perceptron Classification Report

Question 2 - LASSO and ISTA

Theory - 2.1

1. To prove, that LASSO gives the MAP estimate of Linear Regression, with Laplacian prior.

   The linear regression model can be represented as:

   $$y_i = w^T x_i + \epsilon_i$$

   Where, ϵ is a random variable representing Gaussian noise with distribution, $\mathcal{N}(0, \sigma^2)$, and hence $P(\epsilon_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\big(-\frac{\epsilon_i^2}{2\sigma^2}\big)$.

   Now the probability distributions of dataset and weights (prior) are,

   $$P(D|w) \propto \prod_i^N \exp\left(\frac{(y_i - w^T x_i)^2}{-2\sigma^2}\right) \propto \exp\left(-\frac{||Y - w^T X||_2^2}{2\sigma^2}\right)$$

   For laplacian prior, we have:

   $$P(w_i|0,\theta) = \frac{1}{2\theta} \exp\left(-\frac{|w_i|}{\theta}\right)$$

   $$\implies P(w) = \prod_i \frac{1}{2\theta} \exp\left(-\frac{|w_i|}{\theta}\right) \propto \exp\left(-\frac{||w||_1}{\theta}\right)$$

   Now, the posterior distribution can be represented as:

   $$P(w|D) \propto P(D|w)P(w)$$

   $$P(w|D) \propto \exp\left(-\frac{||w||_1}{\theta}\right) \exp\left(-\frac{||Y - w^T X||_2^2}{2\sigma^2}\right)$$

   $$P(w|D) = \mathcal{K} \exp\left(-\frac{||w||_1}{\theta} - \frac{||Y - w^T X||_2^2}{2\sigma^2}\right)$$

   for, $\mathcal{K}$ as a normalization constant, for the posterior distribution of weights.

   Now, MAP estimate is the value for which the probability of posterior distribution is maximum, that is:

   $$w_{MAP} = argmax_w P(w|D)$$

   Now, since log is a monotonically increasing function and $-\log$ is monotonically decreasing, we can apply negative of logarithm to determine the minimum value of R.H.S, and thus MAP estimate can be represented as:

   $$w_{MAP} = argmin_w \left[-\log\left(\mathcal{K} \exp\left(-\frac{||w||_1}{\theta} - \frac{||Y - w^T X||_2^2}{2\sigma^2}\right)\right)\right]$$

$$w_{MAP} = argmin_w \left[\frac{||w||_1}{\theta} + \frac{||Y - w^T X||_2^2}{2\sigma^2} \right]$$

$$w_{MAP} = argmin_w \left[||Y - w^T X||_2^2 + \frac{2\sigma^2 ||w||_1}{\theta} \right]$$

For $\lambda = \frac{2\sigma^2}{\theta}$, the MAP estimate can be represented as:

$$w_{MAP} = argmin_w \left[||Y - w^T X||_2^2 + \lambda ||w||_1 \right]$$

The above expression is same as the LASSO loss function with L1 norm. Hence, it can be concluded that MAP estimate of linear regression with Laplacian Prior gives same result as LASSO estimate.

2. LASSO regression gives sparse solutions then ridge regression because of the penalty term difference in both. Since, for ridge regression the penalty term increases quadratically ($||w||_2^2 < \beta$ is a hypersphere), the Ridge estimate (L2 optimum, which is basically the intersection point) can fall on the axis lines only when the minimum MSE (mean square error) is also exactly on the axis.

   Whereas, LASSO estimate is a constrained optimization problem, for which the penalty term be represented as hypercube with all its vertices on the axis ($||w||_1 < \beta$). Hence, the L1 optimum can be on the axis line as its contour is sharp and therefore there are high chances of interaction point to fall on axis. Therefore it is possible to intersect on the axis line, even when minimum MSE is not on the axis. If the intersection point falls on the axes it is known as sparse.

   Due to the above reasons, lasso regression can give more sparse solution as compared to ridge regression, for suitable values of λ.

3. In general, the LASSO regression **lacks** a closed form solution because the objective function is not differentiable (Also, given in lecture slides).

   However, it is possible to obtain closed form solutions for the special case of an orthonormal design matrix. Now, for orthonormal design matrix X, we have $X^T X = I$, where I is the identity matrix.

   Now, the least squares solution can be written as:

$$w_{LS} = (X^T X)^{-1} X^T y = X^T y$$

The weight vector for lasso regression is given by:

$$w_{Lasso} = argmin_w \left[||y - Xw||_2^2 + \lambda ||w||_1 \right]$$

$$w_{Lasso} = argmin_w \left[||Y||_2^2 - 2Y^T Xw + w^T (X^T X)w + \lambda ||w||_1 \right]$$

$$w_{Lasso} = argmin_w \left[-2Y^T Xw + w^T w + \lambda ||w||_1 \right]$$

Since, $X^T X = I$ and Y can be assumed independent of the weight vector.

Also, $Y^T X = w_{LS}^T$, hence we have:

$$w_{Lasso} = argmin_w \left[\sum_{i=1}^{N} \left((-2) w_{LS_i} w_i + w_i^2 + \lambda |w_i| \right) \right]$$

In the above expression, each term can be minimised with respect to corresponding elements of weight vector to get w_{Lasso_i}, for all i. These different elements can be combined in the matrix form to give the closed solution for LASSO regression. Hence, the closed form exists if the data matrix is orthonormal.

Lab - 2.2

1. Refer to `assignment_2.ipynb` for `ista()` function.

   For the provided ISTA function call, **number of epochs = 15000**, **learning rate = 0.001** and **lambda = 0.02**. The generated curve is added below:

```
Validation loss if 0.1829210953481074
Training Loss loss if 0.16975095790176595
(5,) 0.7502554961183839 (30, 5) (30,)
```

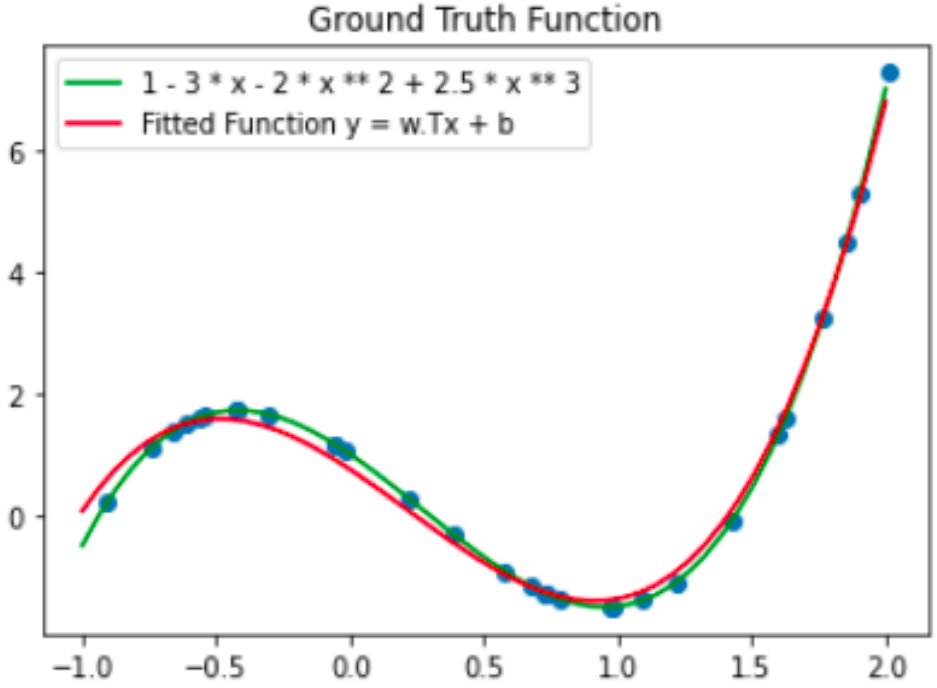


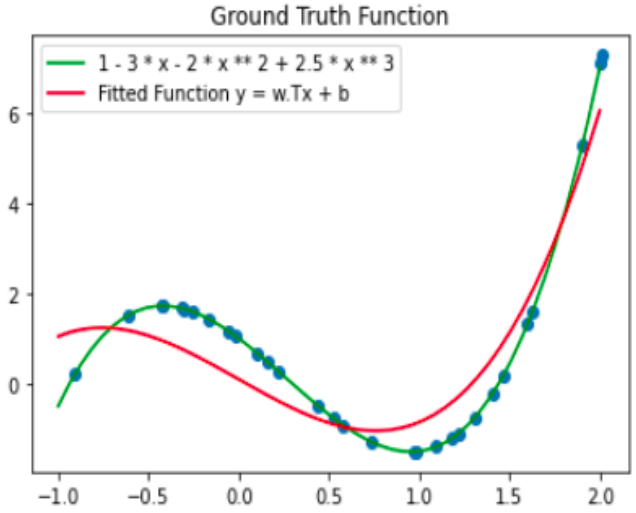
Figure 2: LASSO Regression with ISTA Algorithm

   It can be confirmed that the training loss is less than validation loss and curve is a good fit. Increasing lambda can result in more sparser solutions, for which figures are attached as answer to another subquestion 2.2.3.

2. Refer to `assignment_2.ipynb` for functions `mse_multi_var()`, `mse_regularized()`, `split_data()`, and `multivar reg closedform()`.

3. LASSO does feature selection using the ista() function, as it results much more sparser solutions than ridge regression. For **epochs = 15000**, **lr = 0.001** and **lambda = 0.3**, the plots are added corresponding to ISTA and ridge closed form.

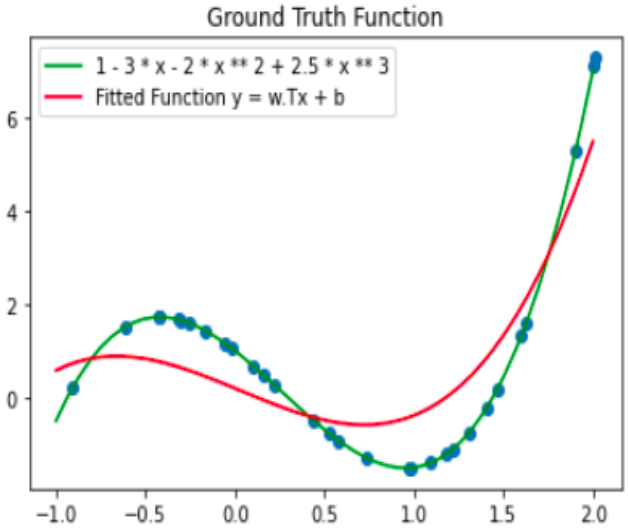
```
Validation loss if 1.3477282368360313
Training Loss loss if 1.6452831796509524
(5,) 0.09408360387688985 (30, 5) (30,)
```




```
(array([-1.12939404,  0.        ,  1.30929834,  0.        , -1.12939404]),
 0.09408360387688985)
```

Figure 3: LASSO Regression, $\lambda = 0.3$

```
Validation loss if 0.4076326289270259
Training Loss loss if 0.874657052673193
```




```
(array([-0.79511067, -0.04964275,  1.10792085, -0.04964275, -0.79511067]),
 0.19655848520003394)
```

Figure 4: Ridge Regression, $\lambda = 0.3$

Feature Selection in LASSO:

We can observe from the above plots that for lambda = 0.3, the weight vector given by ISTA algorithm has some of its elements zero, which implies that apparently the features corresponding to those weight vectors are redundant. These weight attributes are close to zero in case of ridge regression, which implies that they have negligeable effect over regression estimate.

This scenario can be explained as LASSO encourages sparsity in its solution, (refer to the explanation provided in 2.1.2). Hence, the ISTA algorithm will try to make redundant features weights on the axis and other near the vertices of hypercube. Whereas, in ridge regression due to the curvature having higher values, the required weight vector are less likely to be found on the axis, which in turn implies that the chance of getting zero weights in ridge is very low.

Question 3 - Bias and Variance Trade-Off

Theory - 3.1

1. The effect of each of the following on the bias and/or variance of the learned model:

   (a) **Increasing λ in lasso regression:**

   For this case, the **Bias increases** with the increase in lambda. This is because, the effect of penalty term is increasing which leads to more sparser solutions, and hence there will be much lesser chance to be close to the best fit with good curvature.

   Variance decreases because of high sparsity, as some weights will be increased and other will be zero and hence there will be much less curvature which in turn implies that models will be close to each other and hence variance will decrease.

   (b) **Increasing model complexity by adding more features of high degree:**

   Bias will decrease with model complexity, as the model curvature increases (due to more unknowns, as a result of more features) and the scope of finding optimal solution increases, hence the probability that the model estimated is closer to the best fit increases, which in turn decreases the bias.

   Variance will increase with model complexity generally. This is because the number of features is increasing which increases the variance contributions from different features. Since, the number of features is increasing, the chance to get a fit similar to previous decreases, which in turn increases the variance.

   (c) **Reducing dimension by choosing only those subset of features which are of more importance:**

   Bias increases here, as we have fewer features to work out with, and hence the chance of getting optimal fit decreases. Since, the less important features are removed, bias only increases slightly, accounting for the smaller portion of data removed.

   Variance decreases, as lesser features implies lesser sources for variance in the predicted values.

2. Given, irreducible noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$. For $f(\hat{x})$, we have to show that the Mean Squared Error can be written as sum of $Variance(\hat{f(x)})$ and $(Bias(\hat{f(x)}))^2$, where $\hat{f}$ is predicted function of true function f.

   We have $y_i = f(x_i) + \epsilon$, for which the predicted value is $\hat{f(x_i)}$. Now, the mean squared error over dataset can be represented as:

$$MSE = \mathbb{E}_D[(\hat{f}(x) - y)^2]$$
$$MSE = \mathbb{E}_D[(\hat{f}(x))^2] + \mathbb{E}_D[y^2] - 2\mathbb{E}_D[y\hat{f}(x)]$$

   Assuming that the predicted and true values i.e, $\hat{f}(x)$ and y are independent, and also using the fact that:

$$\mathbb{E}[X^2] = Var(X) + \mathbb{E}[X]^2$$

   Now, we have:

$$MSE = \mathbb{E}_D[\hat{f}(x)]^2 + Var(\hat{f}(x)) + \mathbb{E}_D[y]^2 + Var(y) - 2\mathbb{E}_D[y]\mathbb{E}_D[\hat{f}(x)]$$

   Now, let $\bar{f}(x) = \mathbb{E}_D[\hat{f}(x)]$, and also we have, $\mathbb{E}_D[y] = f(x)$ and $Var(y) = Var(\epsilon) = \sigma^2$. Therefore,

$$MSE = (\bar{f}(x))^2 + Var(\hat{f}(x)) + (f(x))^2 - 2\bar{f}(x)f(x) + \sigma^2$$

$$MSE = (\bar{f}(x) - f(x))^2 + Var(\hat{f}(x)) + \sigma^2$$

   Since, $Bias(\hat{f}(x)) = (\bar{f}(x) - f(x))$

$$\implies MSE = Var(\hat{f}(x)) + Bias(\hat{f}(x))^2 + \sigma^2$$

   Therefore, it is shown that MSE can be represented as some of variance and squared bias, along with guassian noise varaince.

Lab - 3.2

1. Refer to `assignment_2.ipynb` for function `gen_bias_variance()`.
2. Generated plots for Bias-Variance Trade Off depending on lambda are as follows:

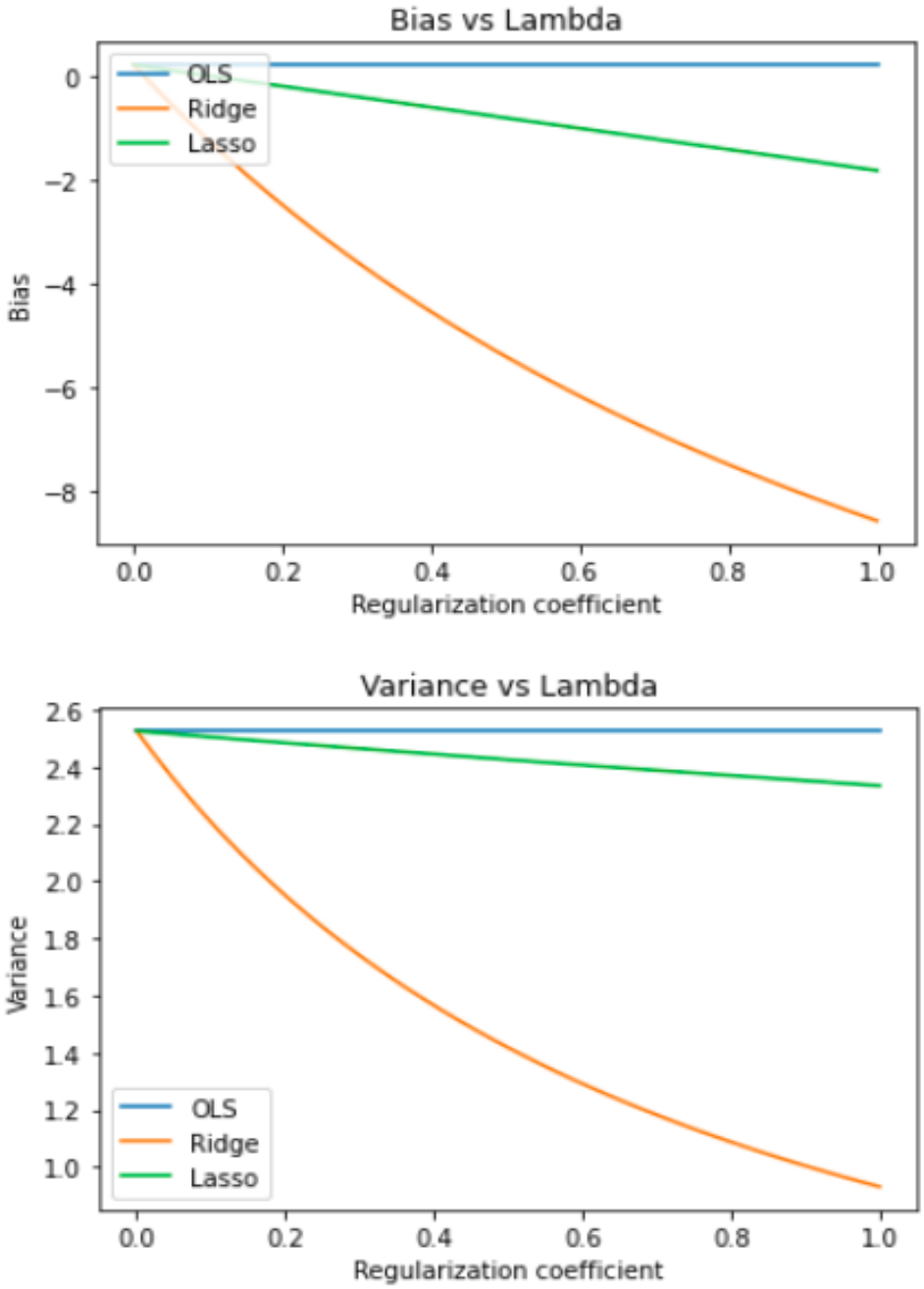


Figure 5: Bias and Variance dependence on λ for OLS, Ridge and LASSO

Analysis: For the Bias vs Lambda graph, it can be observed that all the bias values are negative. While referring to bias, we generally refer to the magnitude, hence here also, we will refer to the magnitude of Bias.

- **OLS Regression:** From, the plot it is observed that Bias increases in magnitude for Lasso and Ridge Regression, but for OLS, it is almost constant throughout. This is because, there is no involvement of λ in the cost function of OLS, hence bias does not changes with λ. Similar, argument can be made for variance also. Hence, for OLS variance also remains constant with respect to λ.
- **Ridge Regression:** Bias (in magnitude) increases and Variance decreases, with increase in λ. This is because as we are increasing lambda we are increasing the penalty term which punishes more for greater L2 norm values, which in turn decreases the curvature of the model. Due to this, the distance between good fit and model increases, hence bias increases. Also, the variance decreases because the model complexity is decreasing. This is a good example of Bias-Variance Tradeoff.
- **Lasso Regression:** Bias (in magnitude) increases with respect to increase in lambda in case of LASSO, which can be reasoned similar to that of Ridge regression. Although, after running the code multiple times, it was observed that variance decreases most of the times, but also remains constant or increases some of the times. This implies that the relationship between Variance of LASSO regression and lambda is not strong enough, unlike the case of Ridge Regression. This may be because of increased sparser solutions in LASSO, leading to randomness in the distribution.

3. Refer to `assignment_2.ipynb` for functions `ridge()`, `lasso()`, and `ols()`.