Homework 1 by Pavel Golikov. I am working in a group with Yash Kant and Barza Nisar.

1 Question 1

a) First let us notice that the cost function that we are using is convex, which means that we can find the minimum by taking a derivative and setting it equal to 0. Thus we have:

$$\frac{\partial}{\partial m}[\frac{1}{n}\sum_{i=1}^{n}(Y_i - m)^2] = 0$$

$$-\frac{2}{n}\sum_{i=1}^{n}(Y_i - m^*) = 0$$

And after some algebraic modifications:

$$\frac{1}{n}\sum_{i=1}^{n}Y_i - \frac{1}{n}\sum_{i=1}^{n}m^* = 0$$

$$\frac{1}{n}\sum_{i=1}^{n}Y_i - m^* = 0$$

$$\frac{1}{n}\sum_{i=1}^{n}Y_i = m^*$$

b) From a) we obtain that $h_{avg}(\mathcal{D}) = m^* = \frac{1}{n}\sum_{i=1}^{n}Y_i$, and combined with the fact that bias is equal to $|E_{\mathcal{D}}[h(\mathcal{D})] - \mu|^2$, we can now substitute m^* in the bias formula for $h(\mathcal{D})$ and obtain the following for Bias:

$$|E_{\mathcal{D}}[\frac{1}{n}\sum_{i=1}^{n}Y_i] - \mu|^2$$

Since the sum of expected value is expected value of the sum, we can factor our expectation and obtain:

$$|\frac{1}{n}\sum_{i=1}^{n}E_{\mathcal{D}}[Y_i] - \mu|^2$$

Now, notice that the term $E_{\mathcal{D}}[Y_i] = \mu$ by definition, so we obtain:

$$|\frac{1}{n}\sum_{i=1}^{n}\mu - \mu|^2$$

$$|\mu - \mu|^2 = 0$$

For Variance, we have:

$$E_{\mathcal{D}}\Big[|h(\mathcal{D}) - E_{\mathcal{D}}[h(\mathcal{D})]|^2\Big]$$

Like in the bias calculation, we make a substitution: $h(\mathcal{D}) = m^* = \frac{1}{n}\sum_{i=1}^{n}Y_i$ and perform a series of algebraic manipulations, most notably, noting that expectation of sum is the sum of expectations:

$$E_{\mathcal{D}}\Big[|\frac{1}{n}\sum_{i=1}^{n}Y_i - E_{\mathcal{D}}[\frac{1}{n}\sum_{i=1}^{n}Y_i]|^2\Big]$$

$$E_{\mathcal{D}}\Big[|\frac{1}{n}\sum_{i=1}^{n}Y_i - \frac{1}{n}\sum_{i=1}^{n}E_{\mathcal{D}}[Y_i]|^2\Big]$$

$$E_{\mathcal{D}}\Big[|\frac{1}{n}\sum_{i=1}^{n}Y_i - \frac{1}{n}\sum_{i=1}^{n}\mu|^2\Big]$$

$$E_{\mathcal{D}}\Big[|\frac{1}{n}\sum_{i=1}^{n}(Y_i - \mu)|^2\Big]$$

$$\frac{1}{n^2}\sum_{i=1}^{n}E_{\mathcal{D}}[(Y_i - \mu)^2]$$

$$\frac{1}{n^2}n\sigma^2$$

$$\frac{\sigma^2}{n}$$

c) Like in part a), we have cost function being convex, which means that we can compute derivative, set it to 0, and compute the explicit formula for the estimator in terms of average and λ.

$$\frac{\partial}{\partial m}[\frac{1}{n}\sum_{i=1}^{n}(Y_i - m)^2 + \lambda|m|^2] = 0$$

$$-\frac{2}{n}\sum_{i=1}^{n}(Y_i - m) + 2\lambda m = 0$$

$$-\frac{1}{n}\sum_{i=1}^{n}(Y_i - m) + \lambda m = 0$$

$$-\frac{1}{n}\sum_{i=1}^{n}Y_i + \frac{1}{n}\sum_{i=1}^{n}m + \lambda m = 0$$

$$-\frac{1}{n}\sum_{i=1}^{n}Y_i + m + \lambda m = 0$$

$$\frac{1}{n}\sum_{i=1}^{n}Y_i = m(1 + \lambda)$$

$$m^* = \frac{h_{avg}}{1 + \lambda}$$

d) Calculating Bias from definition and noting that $h(\mathcal{D}) = \frac{1}{1+\lambda}\frac{1}{n}\sum_{i=1}^{n}Y_i$:

$$|E_{\mathcal{D}}[h(\mathcal{D})] - \mu|^2$$

$$|E_{\mathcal{D}}[\frac{1}{1+\lambda}\frac{1}{n}\sum_{i=1}^{n}Y_i] - \mu|^2$$

$$|\frac{1}{n(1+\lambda)}\sum_{i=1}^{n}E_{\mathcal{D}}[Y_i] - \mu|^2$$

Noting that $E_{\mathcal{D}}[Y_i] = \mu$:

$$|\frac{1}{n(1+\lambda)}\sum_{i=1}^{n}\mu - \mu|^2$$

$$|\frac{n\mu}{n(1+\lambda)} - \mu|^2$$

$$|\mu(\frac{1}{1+\lambda} - 1)|^2$$

$$\frac{\mu^2\lambda^2}{(1+\lambda)^2}$$

Calculating Variance:

$$E_{\mathcal{D}}\Big[|h(\mathcal{D}) - E_{\mathcal{D}}[h(\mathcal{D})]|^2\Big]$$

Like in the bias calculation, we make a substitution: $h(\mathcal{D}) = \frac{1}{1+\lambda}\frac{1}{n}\sum_{i=1}^{n} Y_i$ and perform a series of algebraic manipulations, most notably, noting that expectation of sum is the sum of expectations:

$$E_{\mathcal{D}}\Big[|\frac{1}{n(1+\lambda)}\sum_{i=1}^{n} Y_i - E_{\mathcal{D}}[\frac{1}{n(1+\lambda)}\sum_{i=1}^{n} Y_i]|^2\Big]$$

Now notice that $E_{\mathcal{D}}[\frac{1}{n(1+\lambda)}\sum_{i=1}^{n} Y_i] = \frac{1}{n(1+\lambda)}\sum_{i=1}^{n} E_{\mathcal{D}}[Y_i] = \frac{1}{n(1+\lambda)}\sum_{i=1}^{n}\mu$ and we obtain:

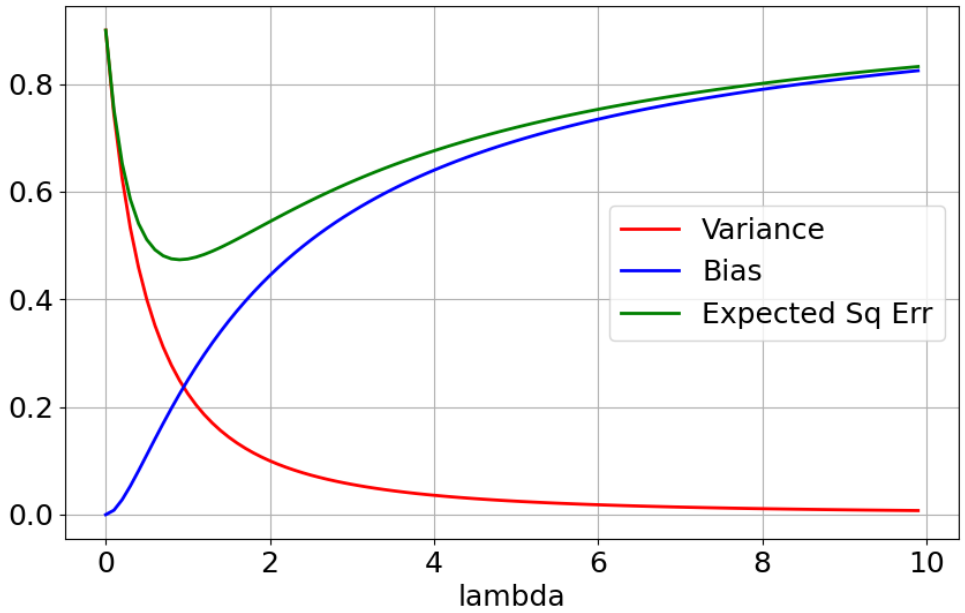
$$E_{\mathcal{D}}\Big[|\frac{1}{n(1+\lambda)}\sum_{i=1}^{n}(Y_i - \mu)|^2\Big]$$

$$\frac{1}{n(1+\lambda)}\sum_{i=1}^{n} E_{\mathcal{D}}\Big[(Y_i - \mu)^2\Big]$$

$$\frac{n\sigma^2}{n^2(1+\lambda)}$$

$$\frac{\sigma^2}{n(1+\lambda)}$$

e) Please see the plot below:



f) From the graph we can note that as λ grows, bias increases and variance decreases. This means that for too large a λ, bias will be large and variance will be small. On the other hand, for a small lambda, bias is small, but variance is large. So in order to minimize the mean squared error, we have to minimize against these two competing values. This suggests that there is a "sweet spot" when mean squared error is at a minimum, which is exactly what we see on the graph above.

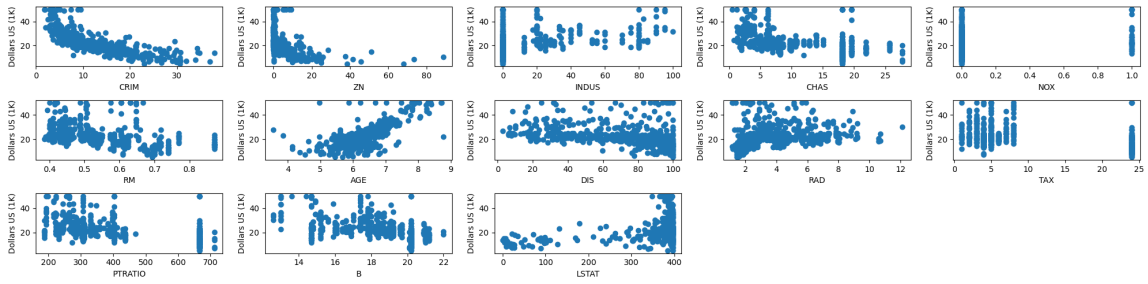
2 Question 2

b) Data set consists of 506 entries, each data entry is a 13x1 vector that contains 13 features that are used to estimate a price of a house. These features are:

CRIM - per capita crime rate by town
ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS - proportion of non-retail business acres per town.
CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
NOX - nitric oxides concentration (parts per 10 million)
RM - average number of rooms per dwelling
AGE - proportion of owner-occupied units built prior to 1940
DIS - weighted distances to five Boston employment centres
RAD - index of accessibility to radial highways
TAX - full-value property-tax rate per $10,000
PTRATIO - pupil-teacher ratio by town
B - 1000(Bk - 0.63)$\hat{2}$ where Bk is the proportion of blacks by town
LSTAT - % lower status of the population

Thus, the entire data set is a 2-dimentional matrix of size 506x13, where each row i is a vector of 13 features of a house whose price is coordinate i of the target vector. Target vector is a vector of size 506x1 and represents the prices of houses sold in the Boston Metropolitan Area. With linear regression we will train an estimator that will, given a vector of 13 features, try to predict the price of the house in the area. Thus our estimator will be a vector of size 13x1, each element of which will be a weight that will assign how important that feature (characteristic of a house) is to the price of the house.

c) Please see the plot below.



e) Please see the table below:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
Weights	-0.10	0.06	0.06	2.44	-21.47	2.80	0.00	-1.52	0.31	-0.01	-1.01	0.01	-0.57

The sign in the INDUS column of the table is positive, which means that the feature contributes positively to the output (the value of the house). Considering the column INDUS in the data set is also positive, it does not match what I have expected. I expected that having industrial properties in the area would reduce the price of the house likely due to noise, excessive movement of industrial equipment through the area etc...

f) Mean Squared Error obtained is: 19.83132367206019

g) The first error metric I would like to suggest is Root of Mean Squared Error. I like this metric because this metric preserves the original units of measurement of the problem we are trying to solve. In the case of this data set, MSE is reporting error in terms of "dollars squared". Taking square root of this metric shows us how far we are from the targets (on average) in the actual dollar amounts. Calculation of this metric is rather straight-forward - we take the square root of the MSE. In case of our predictor, we see that we are

on average 4.45 thousand dollars off when trying to predict the price of the house.

Root Mean Squared Error obtained is: 4.453237437197818.

The second error metric I would like to suggest is Mean Absolute Error (MAE). The error is computed as a sum of absolute values of individual target-data point errors. The reason I picked this error is as follows: Both MSE and RMSE punish larger errors more than smaller errors (because we are squaring the difference). It is not clear that in our case we should punish larger misprediction of house price more. In addition, MAE has the same advantage as RMSE in that it reports error in the same units as the original problem, i.e. in our case, the reported error will be in terms of "dollars" and not "dollars squared" as in MSE. According to this error metric, we see that we are on average 3.34 thousand dollars off when predicting house prices.

Mean Absolute Error obtained is: 3.3446655035984216.

e) Based on our results, we would like to break down the features into two classes: those that are affecting the house price positively (those with positive sign) and those that affect the house price negatively (those with negative sign). Positive features. Of the positive features the most significant ones are CHAS (Charles river dummy variable) and RM. CHAS suggests that more expensive houses are situated closer to the banks of the Charles river that flows through the area, which suggests people would rather live closer to water. RM - the number of rooms per dwelling, is a very intuitive factor in the house price - the larger the house, likely more expensive it will be. Both of these are very intuitive metrics. Negative features. Of the negative features, the one that stands out the most (and the one that prominently stands out in the entire set of features) is NOX - nitric oxides concentration (parts per 10 million). This is not surprising as well. People want to live in areas with clean air. This is actually the main conclusion of the original paper where the data set came from. The data was originally published by Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics Management, vol.5, 81-102, 1978. While it is not surprising that clean air will be a factor in house price, what is surprising is how much of a big role this factor plays in determining the house price. By raw value, this feature completely trumps all other features and significantly reduces the house price if the air quality is bad. Second most significant negative metric is DIS - distance from 5 of the area's most significant employment centers. This is also not surprising - people generally don't want to spend a lot of time commuting to work.

3 Question 3

To derive the appropriate formula, let us write the w^* in the matrix form as:

$$w^* = argmin_w \frac{1}{2}(y - Xw)^T A(y - Xw)$$

Now, analogous to lecture slides, let us denote the minimization objective as J and obtain:

$$J = (y - Xw)^T A(y - Xw)$$

$$J = (y^T - w^T X^T)A(y - Xw)$$

$$J = (y^T A - w^T X^T A)(y - Xw)$$

$$J = (y^T Ay - y^T AXw - w^T X^T Ay + w^T X^T AXw$$

Notice that for a scalar value c, $c^T = c$ and notice that $y^T AXw$ has size 1×1, which means that $(y^T AXw)^T = y^T AXw$. Combined with the fact that A is a diagonal matrix, which means that $A^T = A$, we get that $y^T AXw = W^T X^T Ay$ and our previous expression for J becomes:

$$J = (y^T Ay - 2w^T X^T Ay + w^T X^T AXw$$

At this point, in order to find the minimum of J, we differentiate the expression element-wise with respect to w:

$$\frac{\partial J}{\partial w} = 0 - 2X^T Ay + 2X^T AXw^* = 0$$
$$2X^T AXw^* = 2X^T Ay$$
$$w^* = (X^T AX)^{-1} X^T Ay$$

which is what we needed to show.

b) Briefly recomputing the direct solution for locally weighted regression, we obtain:

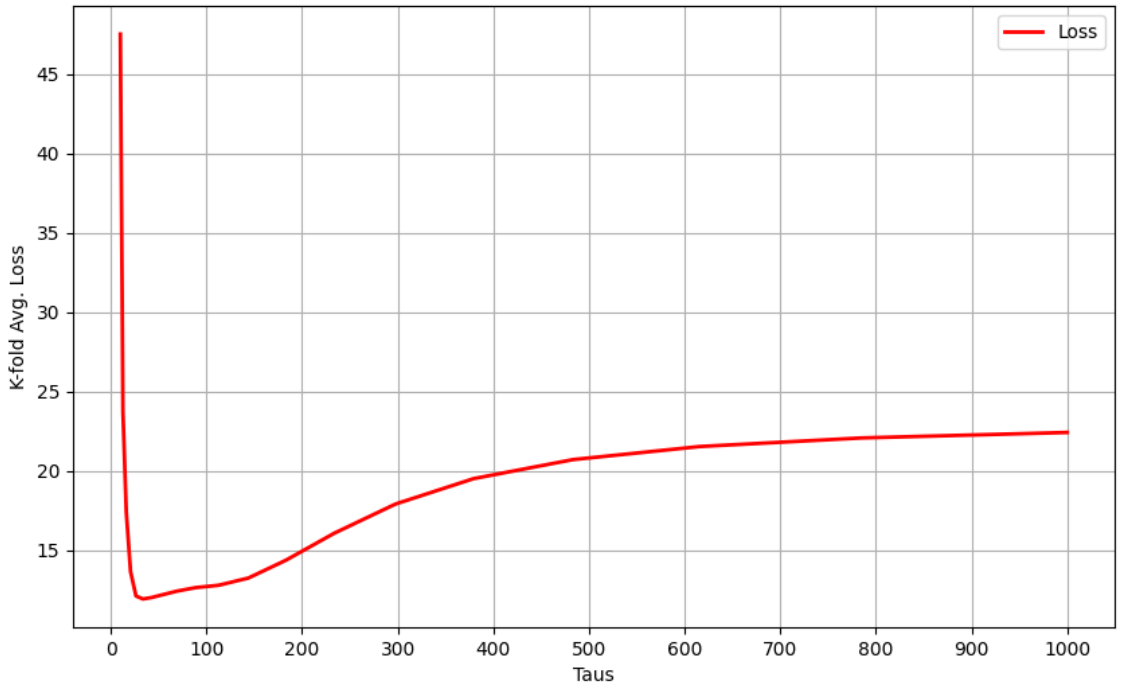
$$J = (y^T Ay - 2w^T X^T Ay + w^T X^T AXw + \lambda W^T W$$

and after solving (I am omitting details of the solution since the problem is very similar to part a):

$$w^* = (X^T AX + \lambda I)^{-1} X^T Ay$$

which will be used in the programming solution.

c) Please see the plot below:



d) First, let us remind ourselves as to the definition of a_i:

$$a_i = \frac{exp(-||x - x^{(i)}||^2/2\tau^2)}{\sum_j exp(-||x - x^{(j)}||^2/2\tau^2)}$$

Notice that as τ gets closer to ∞, the exponents in the expression for a_i get closer to 1. This is because the term $-||x - x^i||^2/2\tau^2$ becomes small and negative, which means that exponent of the term approaches 1. This means that top will approach 1 and bottom will approach N, since we have the sum of all such terms. This means that each a_i will approach $\frac{1}{n}$, meaning that our formula for w^* will approach that of ordinary linear regression.

When τ gets closer to 0, all terms $a^{(i)}$ approach 0, except for those close to training sample i, and the argmin will be the index of the closest training sample, which means that the minimization problem is solved by choosing the label of the closest point in the training set. This is essentially the behaviour of k-NN classifier with K = 1.

e) Disadvantages:

- Computational complexity (NOT the theoretical term is meant here). Each prediction involves performing linear regression model training on the entire data set. This requires a lot more computational power compared to a dot product of two vectors (features and weights). - Memory requirements. In order to make a single prediction, we have to keep the entire training set. If the training set is large, then this will put additional stress on the memory subsystem.

Advantages:

- More flexible compared to ordinary linear regression. Addition of the k-NN like behaviour allows us to model problems that are more complicated than those that ordinary linear regression can model. - We don't have to fit a function to all data points in the training data set since this model focuses on locality, and we care more about those points closer to the test data point.