ML_HW2

Mohammad Javad Pesarakloo
810100103

April 27, 2024

Question 1

First, we find a relation for $E[\hat{p}(x)]$:

$$E[\hat{p}(x)] = E[\frac{1}{v_n}\sum_{i=1}^{n}\frac{1}{n}\phi(\frac{x-x_i}{h_n})] = \frac{1}{n}\sum_{i=1}^{n}E[\frac{1}{v_n}\phi(\frac{x-x_i}{h_n})]$$

As expected value of X, is same for all samples, we simply choose the k-th element and omit the summation like this:

$$\frac{1}{n}\sum_{i=1}^{n}E[\frac{1}{v_n}\phi(\frac{x-x_i}{h_n})] = E[\frac{1}{v_n}\phi(\frac{x-x_k}{h_n})]$$

$$= \int \underbrace{\frac{1}{v_n}\phi(\frac{x-x_k}{h_n})}_{\delta(x-x_k)}p(x_k)dx_k \tag{1}$$

Now let's calculate variance:

$$\sigma_n^2(x) = Var(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{v_n}\phi(\frac{x-x_i}{h_n})]) = \frac{1}{n^2}\sum_{i=1}^{n}Var(\frac{1}{v_n}\phi(\frac{x-x_i}{h_n}))$$

with the same reason as above, we use k-th element to omit summation:

$$= \frac{1}{n}Var(\frac{1}{v_n}\phi(\frac{x-x_k}{h_n})) = \frac{1}{n}E[\frac{1}{v_n^2}\phi^2(\frac{x-x_k}{h_n})] - \frac{1}{n}\underbrace{E^2[\frac{1}{v_n}\phi(\frac{x-x_k}{h_n})]}_{\hat{p}_n^2}$$

$$= \frac{1}{n}\int\frac{1}{v_n^2}\phi^2(\frac{x-x_k}{h_n})p(x_k)dx_k - \frac{1}{n}\hat{p}_n^2(x)$$

by considering $sup(\phi)$ as upper bound of the kernel function, we can consider the following phrase as an upper bound for the above relation:

$$\leq \frac{1}{n}\frac{sup(\phi)}{v_n}\int\underbrace{\frac{1}{v_n}\phi(\frac{x-x_k}{h_n})}_{\delta(x-x_k)}p(x_k)dx_k$$

replacing the equation using equation 1, we get the following phrase:

$$\leq \frac{sup(\phi)E[\hat{p}(x)]}{nv_n}$$

Question 2

X is one-dimensional thus $V_n = h_n$:

$$E[p_n(x)] = E[\frac{1}{n}\sum_{i=1}^{n}\frac{1}{h_n}\phi(\frac{x-x_i}{h_n})] = E[\frac{1}{h_n}\phi(\frac{x-x_i}{h_n})]$$

$$= \int_{-\infty}^{+\infty} \frac{1}{h_n}\phi(\frac{x-x_i}{h_n})p(x_i)dx_i$$

Given the kernel function, for $xi > x$ we have $x - x_i < 0$ and the value of ϕ is zero;thus we can rewrite the above integration like:

$$I = \int_{-\infty}^{x} \frac{1}{h_n}\phi(\frac{x-x_i}{h_n})p(x_i)dx_i = \int_{-\infty}^{x} \frac{1}{h_n}e^{\frac{x_i-x}{h_n}}p(x_i)dx_i$$

Now we have three scenarios:

- $x < 0 \xrightarrow{x_i<x} p(x_i) = 0 \Rightarrow \hat{p}_n(x) = 0$
- $0 \leq x \leq a$

$$I = \int_{0}^{x} \frac{1}{h_n}e^{\frac{x-x_i}{h_n}}\frac{1}{a}dx_i = \frac{1}{a}e^{\frac{x_i-x}{h_n}} \xrightarrow[x_i=0]{x_i=x} = \frac{1}{a}(1 - e^{\frac{-x}{h_n}})$$

- $x \geq a$

$$I = \int_{0}^{a} \frac{1}{h_n}e^{\frac{x_i-x}{h_n}}\frac{1}{a}dx_i = \frac{1}{a}e^{\frac{x_i-x}{h_n}} \xrightarrow[x_i=0]{x_i=a} = \frac{1}{a}(e^{\frac{a-x}{h_n}} - e^{\frac{-x}{h_n}})$$

$$= \frac{1}{a}(e^{\frac{a}{h_n}} - 1)e^{\frac{-x}{h_n}}$$

For bias of a query x to be less than 1% we have:

$$\frac{x - \frac{1}{a}(1 - e^{\frac{-x}{h_n}})}{x} \leq 0.01$$

$$\Rightarrow 1 - \frac{1}{ax}(1 - e^{\frac{-x}{h_n}}) \leq 0.01$$

$$\Rightarrow 1 - e^{\frac{-x}{h_n}} \geq 0.99x \Rightarrow e^{\frac{-x}{h_n}} \leq 1 - 0.99x$$

$$\xrightarrow{\ln} \frac{-x}{h_n} \leq \ln(1 - 0.99x)$$

$$\Rightarrow h_n \leq \frac{x}{-\ln(1 - 0.99x)}$$

Question 3

$$\beta = argmin_\beta \sum_{i=1}^{n}(Y_i - X_i\beta)^2 + \lambda||\beta||_2^2$$

$$= \beta = argmin_\beta \underbrace{\sum_{i=1}^{n}(A\beta - Y)^T(A\beta - Y) + \lambda||\beta||_2^2}_{RR}$$

$$\frac{dRR}{d\beta} = 2A^T(A\beta - Y) + 2\lambda\beta = 0$$

$$\Rightarrow A^TA\beta - A^TY + \lambda\beta = \beta(A^TA + \lambda I) - A^TY = 0$$

$$\Rightarrow \beta(A^TA + \lambda I) = A^TY$$

$$\xrightarrow{\times(A^TA+\lambda I)^{-1}} \beta = (A^TA + \lambda I)^{-1}A^TY$$

L1 Regularization, also known as Lasso regularization (Least Absolute Shrinkage and Selection Operator), adds the sum of the absolute values of the coefficients as a penalty term to the loss function. This method can lead to sparse solutions with some coefficients being shrunk to zero, effectively performing feature selection.
L2 Regularization, also known as Ridge regularization, adds the sum of the squared values of the coefficients as a penalty term to the loss function. Unlike L1, L2 regularization tends to shrink coefficients evenly but does not necessarily zero them out, leading to non-sparse solutions. Here are three key differences between L1 and L2 Regularizations:

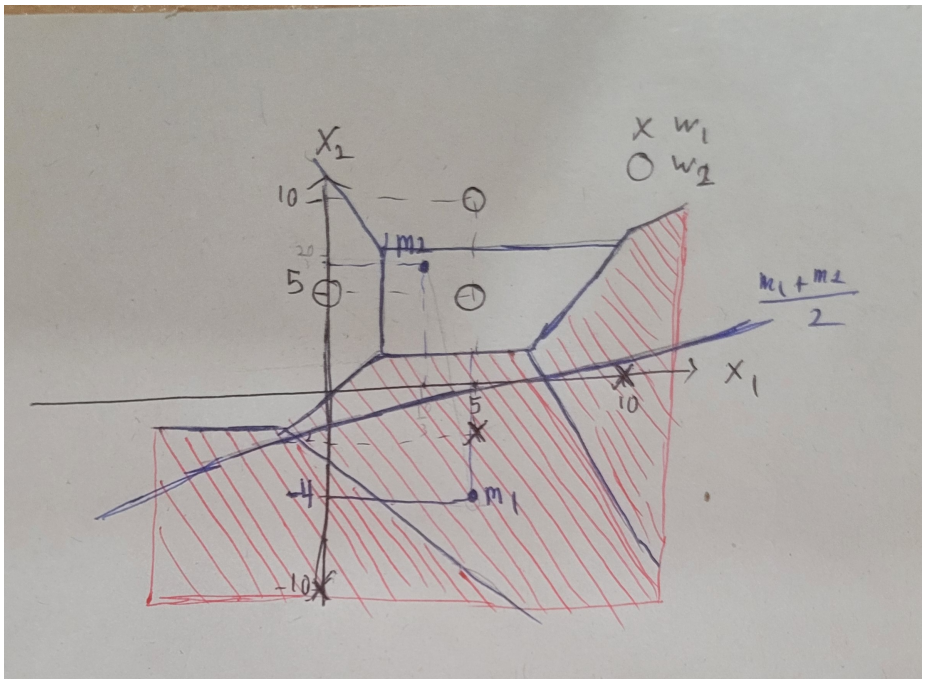
- Sparsity:L1 Regularization tends to produce sparse models where some coefficients can become exactly zero. This means that L1 can eliminate some features entirely, which can be useful for feature selection while L2 Regularization generally results in non-sparse models where coefficients are shrunk towards zero but not exactly zero.

- Feature Selection:L1 Regularization is more likely to zero out less important features, thus performing implicit feature selection.L2 Regularization does not perform feature selection; it only penalizes the magnitude of the coefficients without setting them to zero.

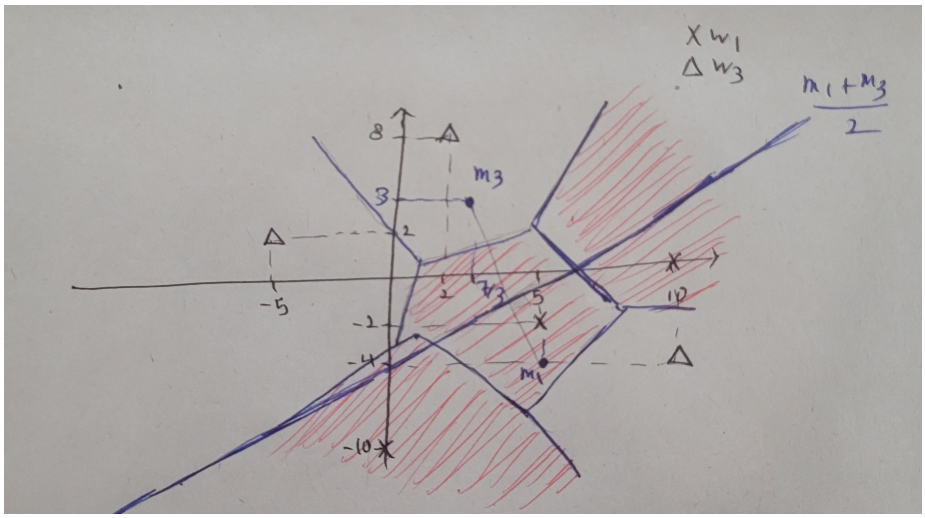
- Handling Collinearity:L1 Regularization may not be the best choice when dealing with collinear or codependent features because it might arbitrarily select one feature over another.L2 Regularization is better suited for handling collinear features as it tends to shrink all related features together and reduces the variance of these estimates.

Question 4

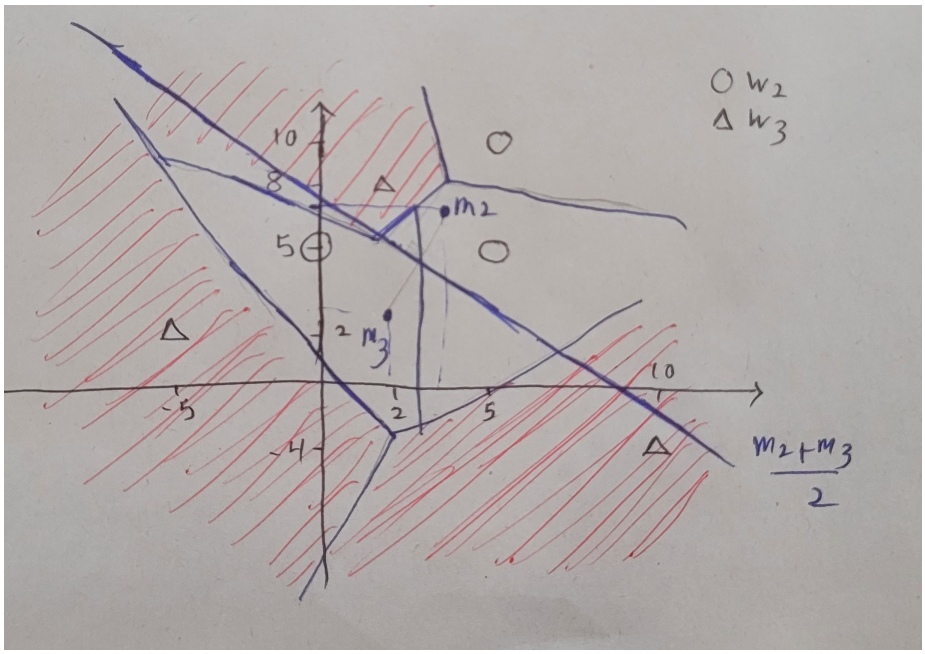
1



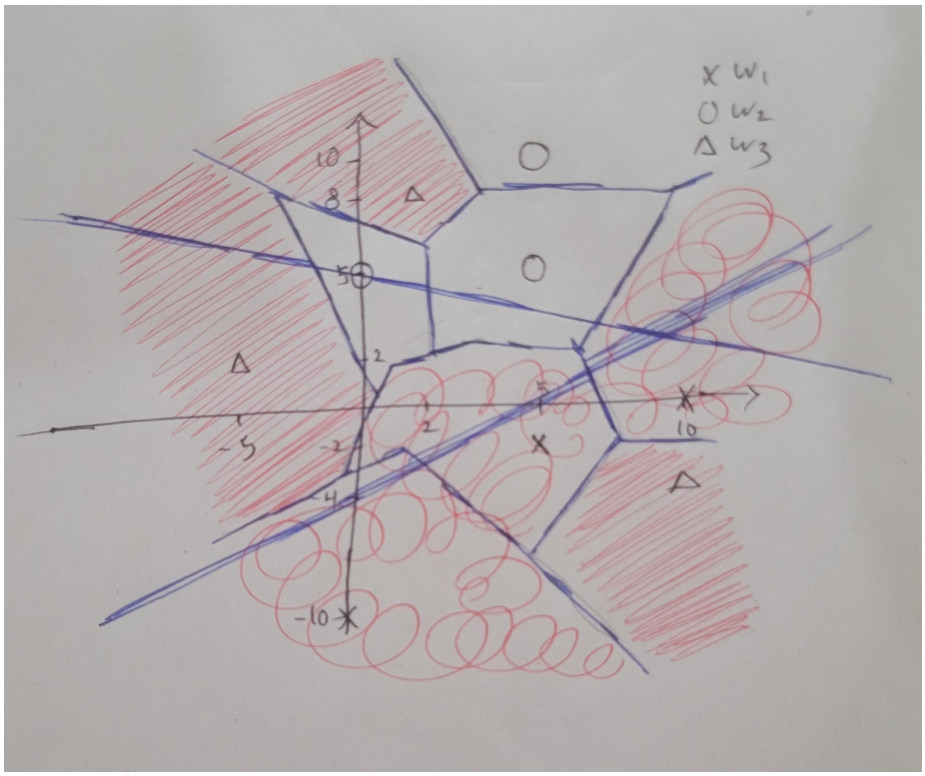
2



3



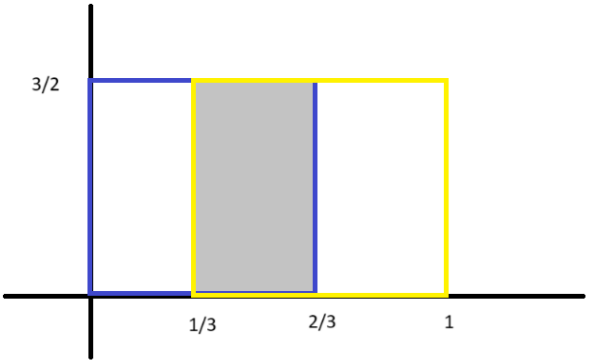
4



Question 5

1

Assume $p(w_1) = p(w_2) = 0.5$.Then the posterior probabilities intersection defines the bayes decision boundary.The following figure suggests that it can be any x between $\frac{1}{3}$ and $\frac{2}{3}$;for simplicity we consider $x = \frac{1}{3}$ as bayes decision boundary: Then the bayes error is :

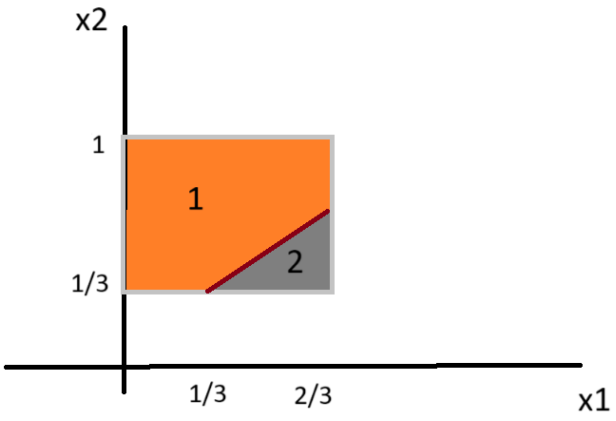


$$\int_0^1 min[p(w_1)p(x|w_1), p(w_2)p(x|w_2)]dx$$

$$\int_{\frac{1}{3}}^{\frac{2}{3}} p(w_1)p(x|w_1) = 0.5 \times \frac{1}{3} \times \frac{3}{2} = 0.25$$

2

Position of the decision boundary in this case is $\frac{x_1+x_2}{2}$.To find error, it matters which one is bigger,so we consider two cases and get weighted average on them.Distribution of probability of x_1 and x_2 is :


x2
1
1/3
1
2
1/3
2/3
x1

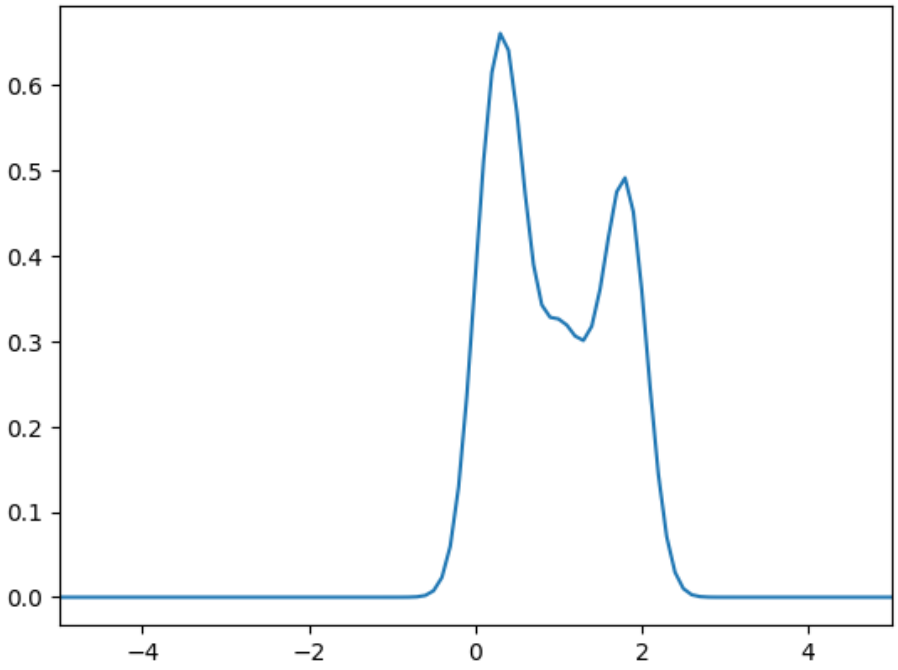
- $x_1 \leq x_2$ with probability $\frac{7}{8}$.In this case the bayes error is 0.25
- $x_1 > x_2$ with probability $\frac{1}{8}$.In this case the bayes error is 0.75

$$p_1(e) = \frac{7}{8} \times 0.25 + \frac{1}{8} \times 0.75 = 0.3125$$

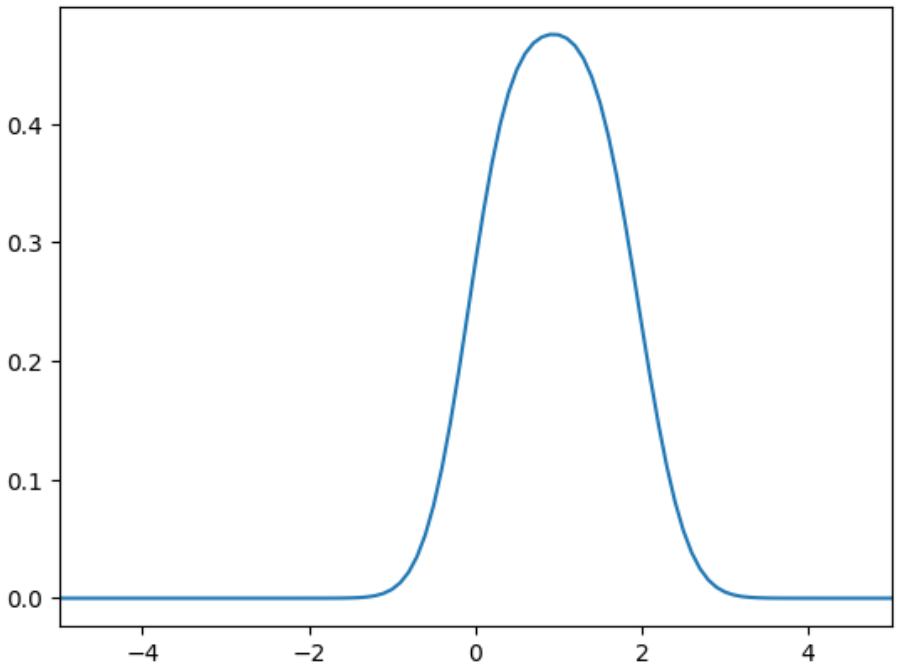
Question 6

Codes of this Question is in 6.ipynb jupyter notebook file.The estimated pdf in different cases is as follows:

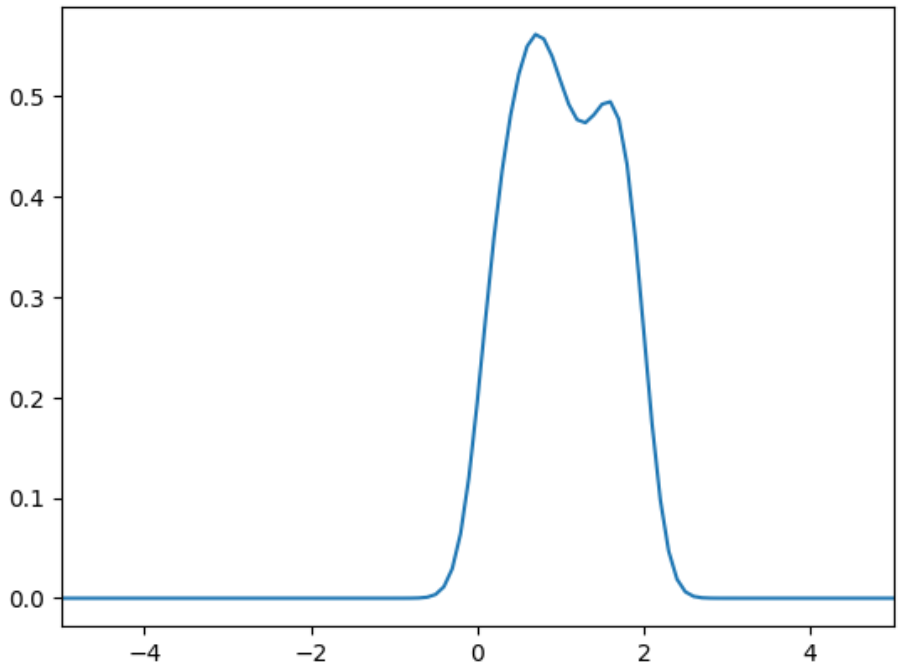
$V_n = 0.05,\ n = 32$



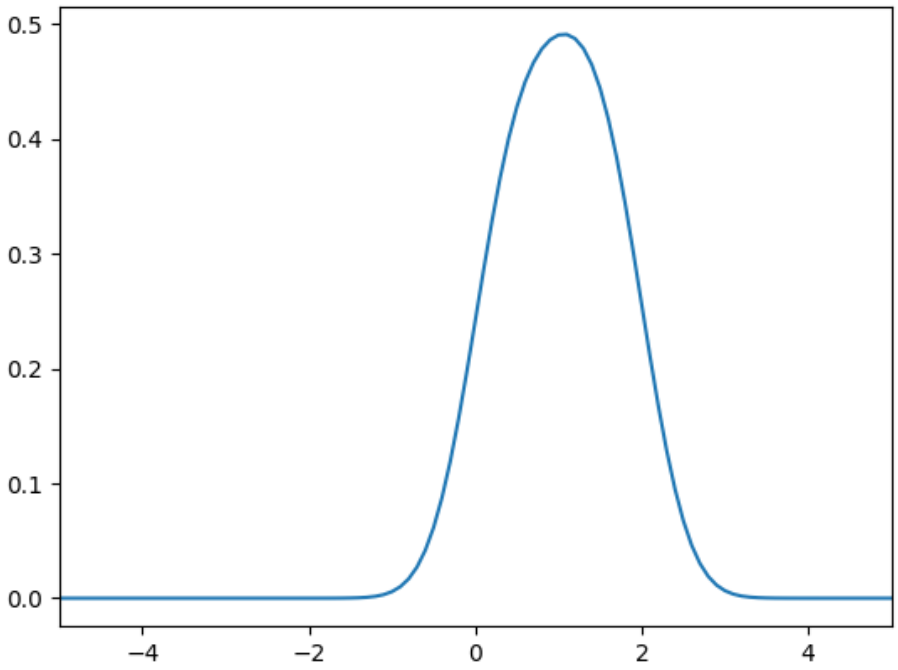
$V_n = 0.2,\ n = 32$


0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4

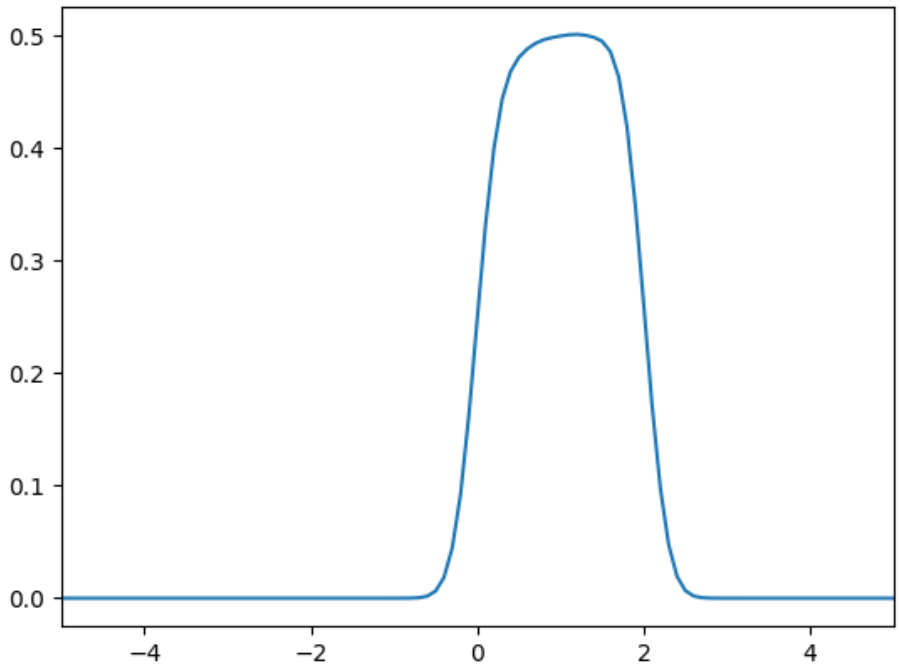
$V_n = 0.05,\ n = 256$



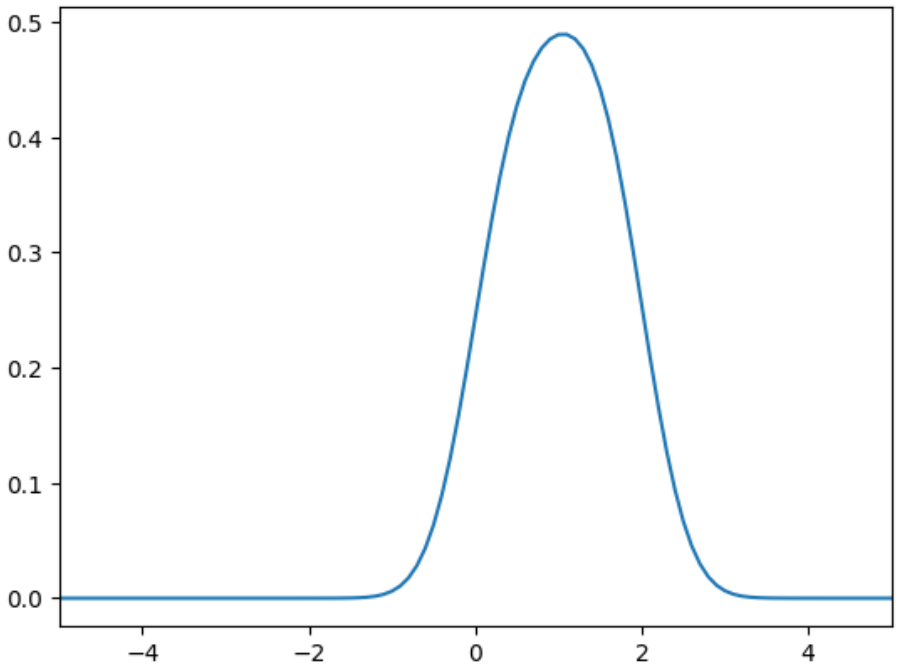
$V_n = 0.2,\ n = 256$


0.5
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4

$V_n = 0.05,\ n = 5000$

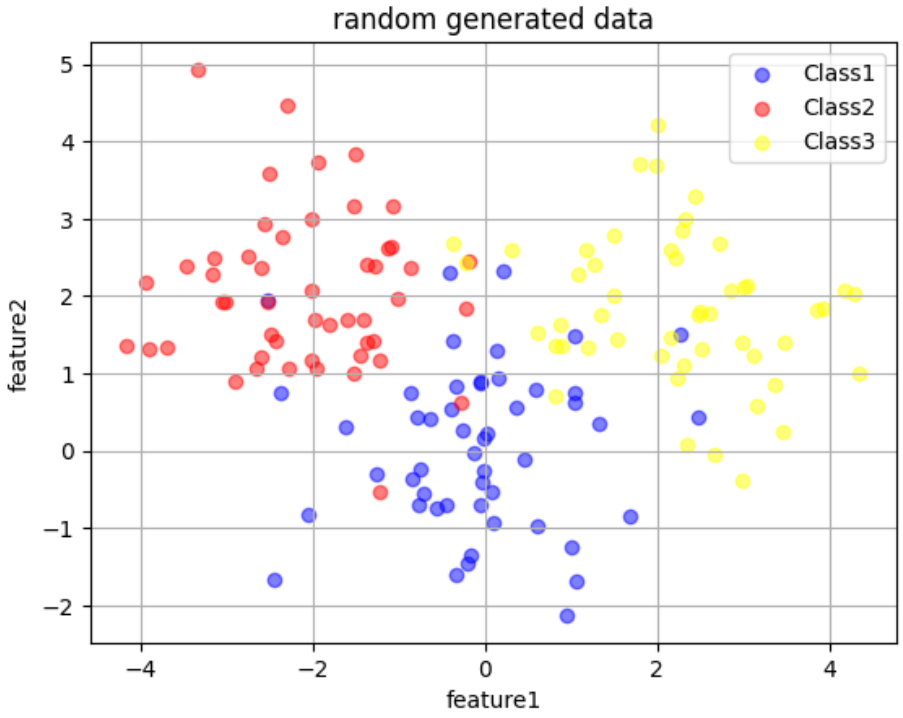


$V_n = 0.2,\ n = 5000$


0.5
0.4
0.3
0.2
0.1
0.0
−4
−2
0
2
4

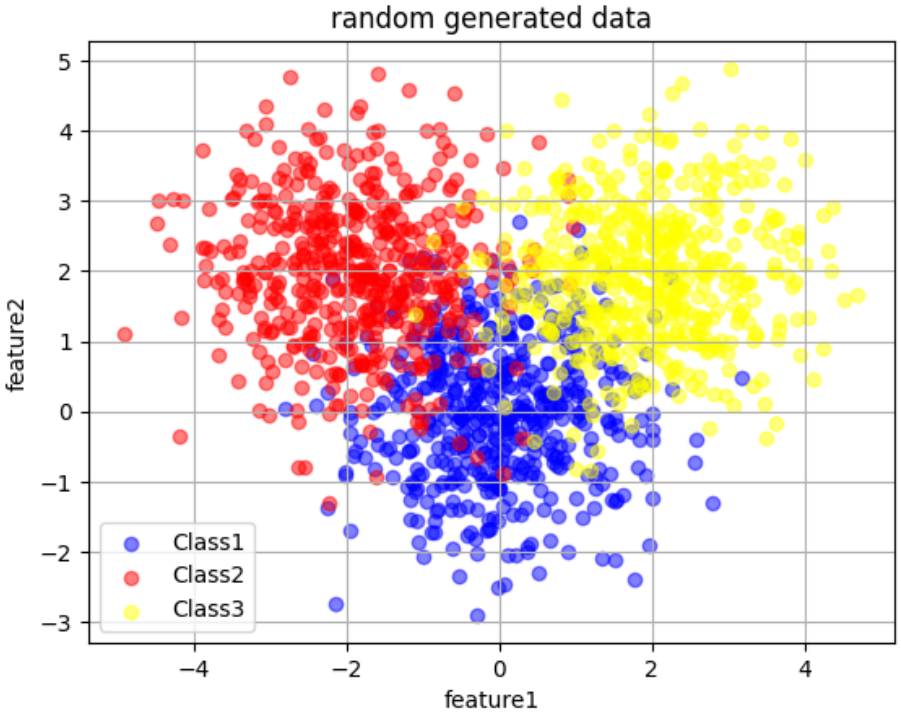
Question 7

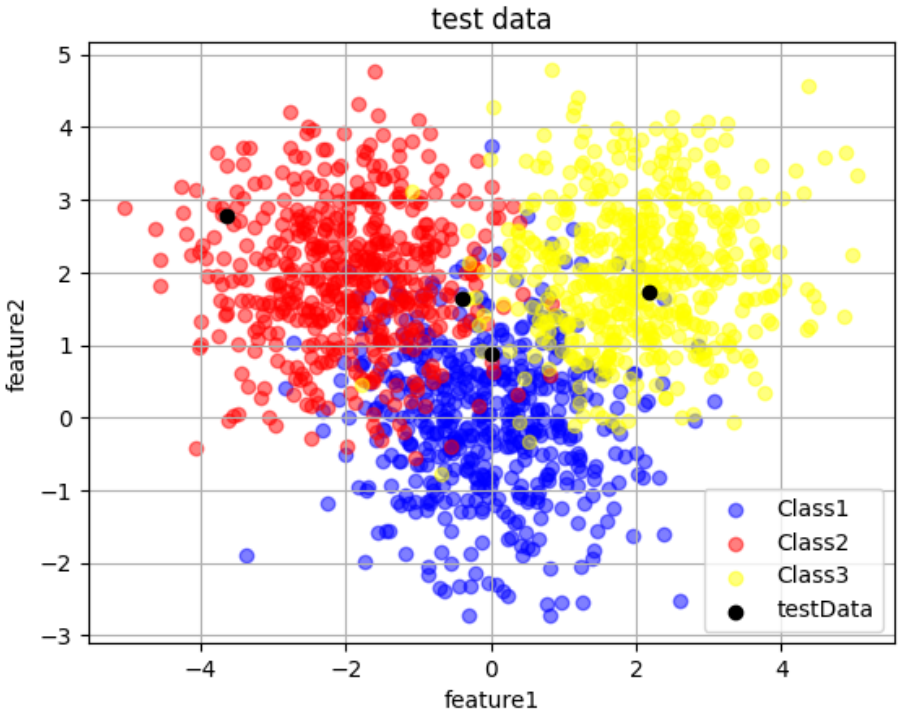
Codes of this Question is in 7.ipynb jupyter notebook file.At first, let's visualize generated data per classes:
now let's generate some test samples, the test true labels are 1,2,3,2:




test data
Class1
Class2
Class3
testData
feature1
feature2
5
4
3
2
1
0
−1
−2
−4
−2
0
2
4

The prediction of the model in this case for both $V_n = 1$ and $V_n = 0.1$ is 1, 1, 3, 2 which comes from the second test point which is exactly on the boundary of classes as you can see in the above figure. Now lets generate more samples to fine tune density estimation:
and here is the visualization of test samples:

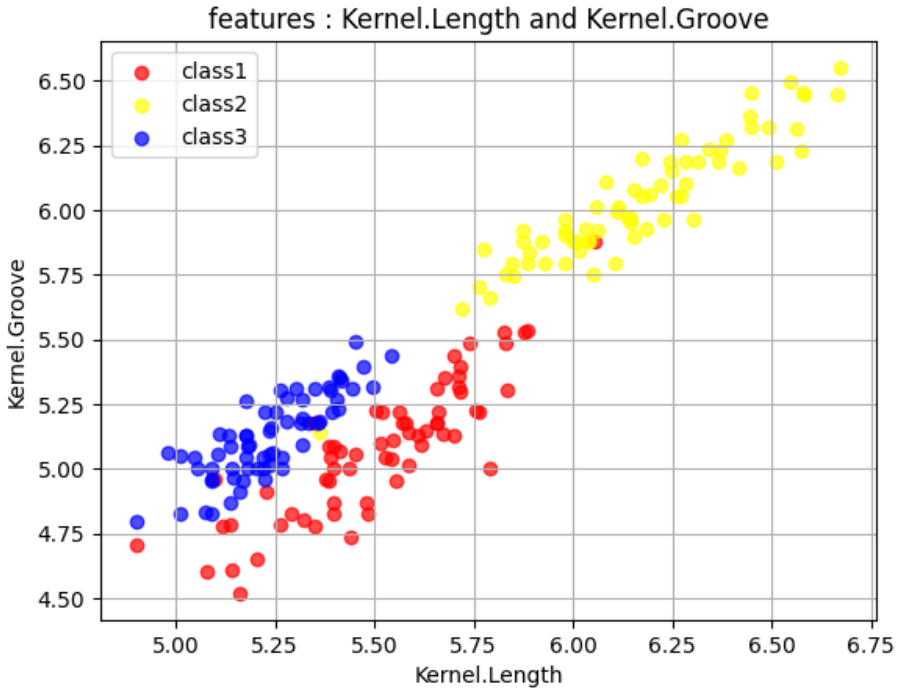



test data
feature2
feature1
Class1
Class2
Class3
testData

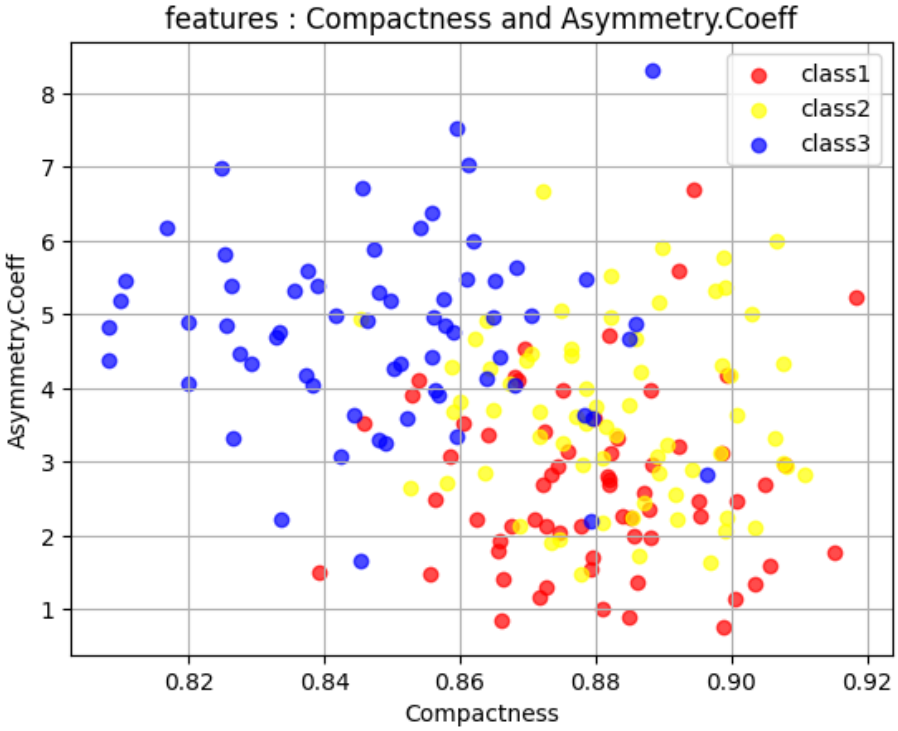
The model in this case was 100% accurate on both cases of V_n due to high number of samples.

Question 8

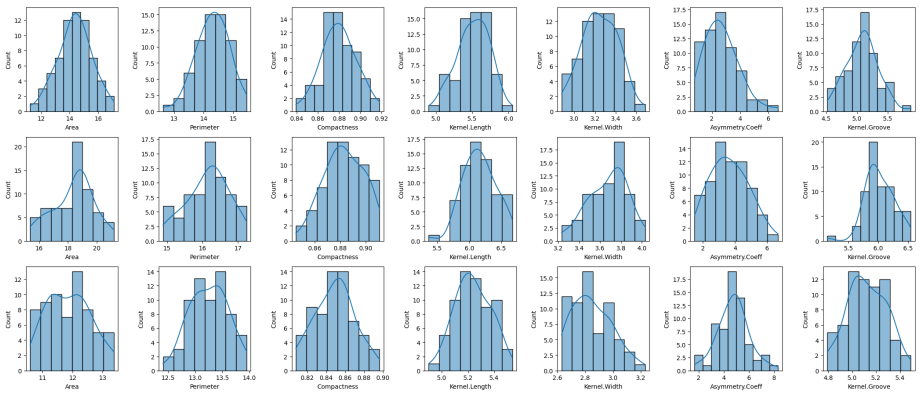
Codes of this Question is in 8.ipynb jupyter notebook file.At frist I read csv file as a pandas dataframe and seperated rows according to target variable.By choosing 2 features and visualizing scatter plot for each of them, I got 21 scatter plots that shows how good each set of two features separate 3 classes.For example a good separation can be:



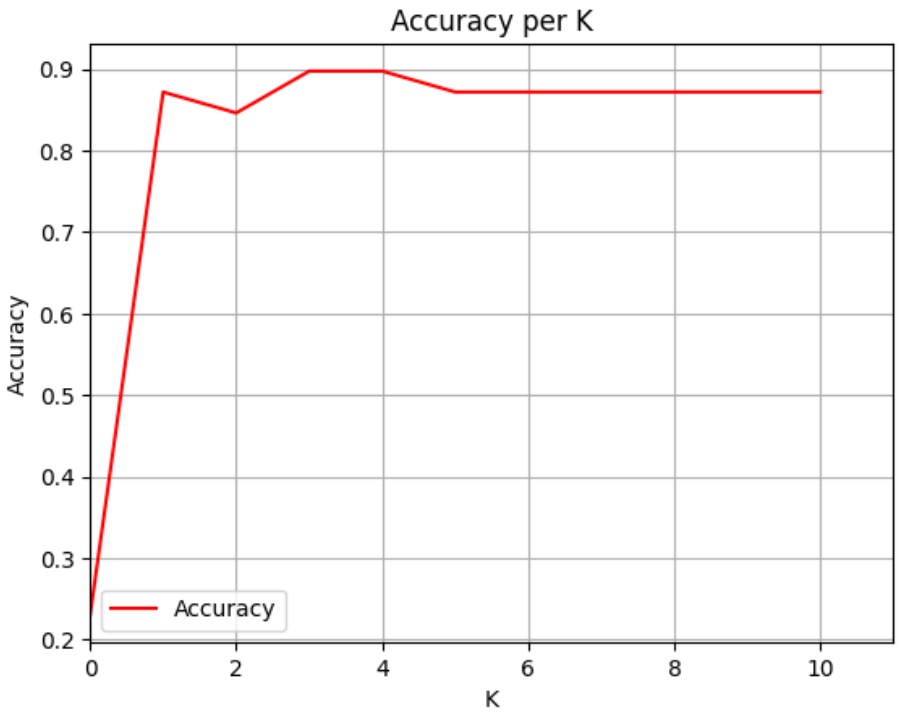
while the following feature set is not a decent separation of classes:
By using histogram plot, we can visualize pdf of each feature per class to see



whether it looks like the target variable or not:


Count
Area
Perimeter
Compactness
Kernel.Length
Kernel.Width
Asymmetry.Coeff
Kernel.Groove

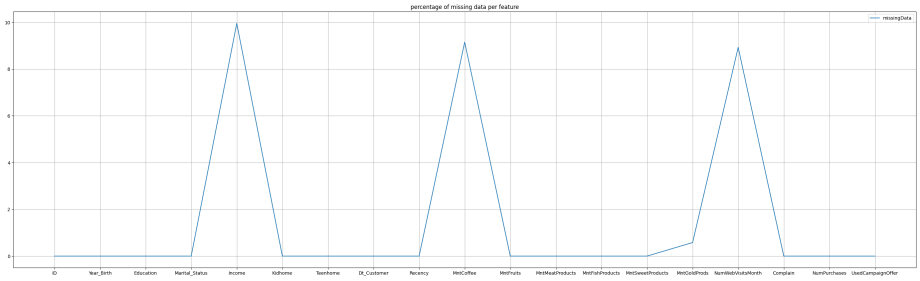
Then I verified that no data is missing, all features are numerical and I can normalize them to use logistic regression model for classification.My model converged with $learningrate = 0.0001$ and 1000 iterations and got 82% accuracy.precision, recall, f1-score and confusion matrix is available in the notebook. for the next part which we are supposed to use KNN classifier, I wrote a KNN class which can give response to queries according to specified k parameter.By testing different parameters and plotting the accuracy figure per k, we obtain that k=4 is the best choice with accuracy of 89%:



Question 9

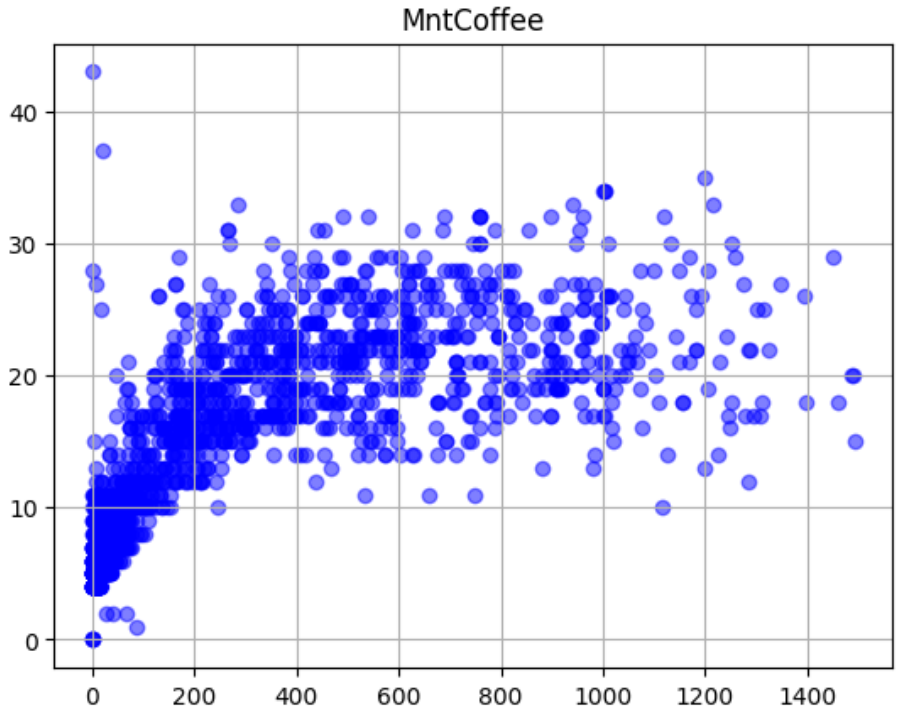
Codes of this Question is in 8.ipynb jupyter notebook file.After reading csv file as a pandas dataframe, we observe that there are lots of missing datas.let's visualize what proportion of data is lost per each feature:

To continue analyzing data, we need to **handle missing values**.One way is



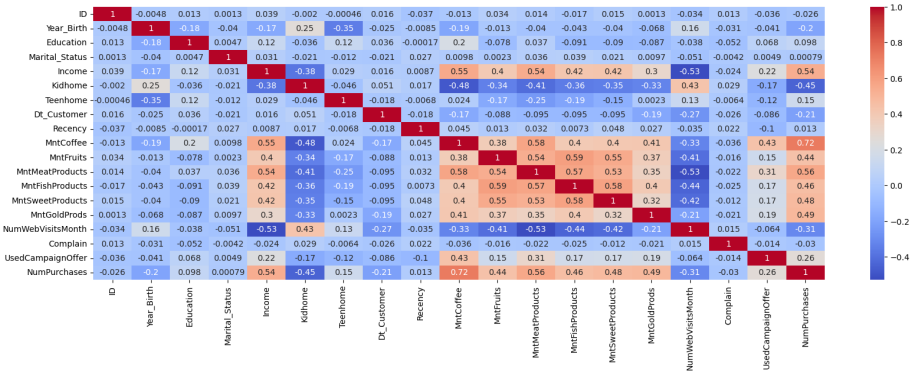
to replace missing value of each feature with mean of all available samples.Here to avoid complexity, we easily omit all missing data.

The next preprocessing we need to do is encoding categorial datas.After that we are ready to normalize them and feed them to different models;but before rushing to normalization, let's do some analysis on features.The first thing we can do is plotting scatter plot of each feature with target variable;doing this step can lead us to figuring out which feature has a good linear contribution to target variable.looking at the scatter plots in the notebook, we obtain that **MntCoffee** can be a good choice:


MntCoffee
40
30
20
10
0
0
200
400
600
800
1000
1200
1400

Another important thing to note is correlation of features;we can distinguish if features are correlated by analyzing the correlation matrix:

Using this matrix, we can find features with higher correlation with target



variable and select them as feature space of a regression model.

Now we want to fit a simple $y = ax+b$ line using the most-linear-looking feature to calculate the target variable.To find optimal **a** and **b**, we should minimize the **RootMeanSquareError(RMSE)** function:

$$RMSE = (\frac{1}{n}\sum_{i=1}^{n}(ax_i + b - y_i)^2)^{\frac{1}{2}}$$

considering error term as e(for simplicity in notations) we have:

$$\frac{d_{RMSE}}{d_a} = \frac{1}{2}e^{\frac{-1}{2}}(\frac{1}{n}\sum_{i=1}^{n}2x_i(ax_i + b - y_i)) = 0 \tag{2}$$

$$\frac{d_{RMSE}}{d_b} = \frac{1}{2}e^{\frac{-1}{2}}(\frac{1}{N}\sum_{i=1}^{n}2(ax_i + b - y_i)) = 0$$

$$\Rightarrow \sum_{i=1}^{n}2(ax_i + b - y_i) = 0 \Rightarrow b = \frac{\sum_{i=1}^{n} y_i - ax_i}{n}$$

Now if we replace value of b in equation (2), we have:

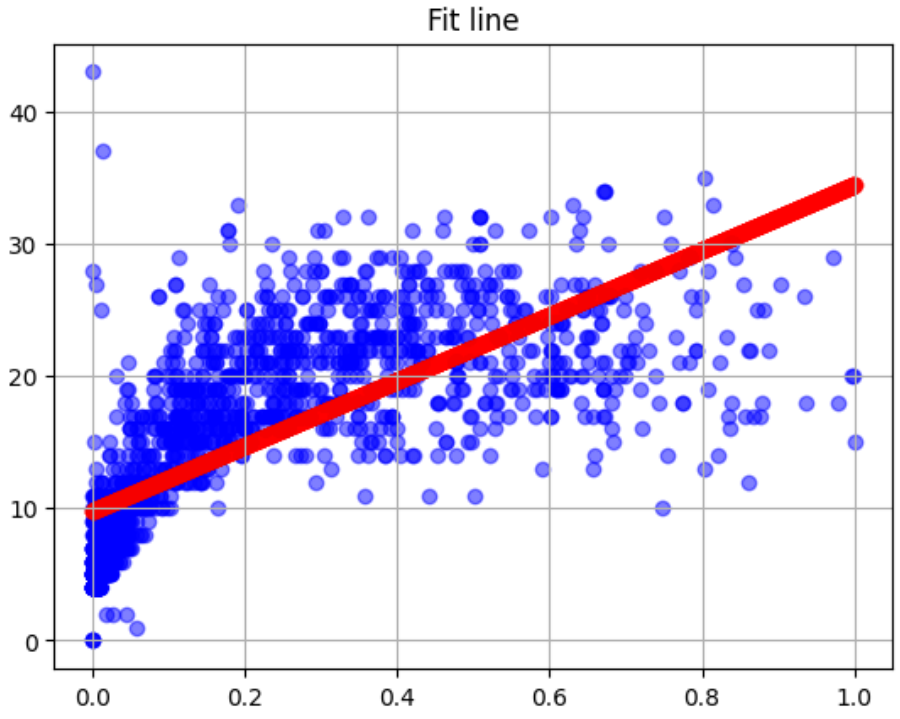
$$\sum_{i=1}^{n}2x_i(ax_i + \frac{\sum_{i=1}^{n} y_i - ax_i}{n} - y_i)$$

$$= \sum_{i=1}^{n}(2ax_i^2 + \frac{2}{n}\sum_{i=1}^{n}y_i - \frac{2ax_i}{n}\sum_{i=1}^{n}x_i - 2x_iy_i)$$

$$= a\sum_{i=1}^{n}[2x_i^2 - \frac{2x_i}{n}\sum_{i=1}^{n}x_i] + \sum_{i=1}^{n}[\frac{2x_i}{n}\sum_{i}^{n}y_i - 2x_iy_i] = 0$$

$$a = \frac{\sum_{i=1}^{n}[2x_i y_i - \frac{2x_i}{n}\sum_{i=1}^{n} y_i]}{\sum_{i=1}^{n}[2x_i^2 - \frac{2x_i}{n}\sum_{i=1}^{n} x_i]} \tag{3}$$

Now we can calculate a and b according to our train data and fit the line.As mentioned earlier, the feature **MntCoffee** can be a good choice for this section.The line fit to this feature is:



To evaluate this model, I used RootMeanSquareError and R2 score which were 5.672103385124087 and 0.46934973650947065 respectively.
Now we want to use multiple linear regression model and select multiple features;as number of dimensions of feature space increases, solving the equations like the previous part is almost impossible,so we use numerical optimization methods like gradient descent.To select three best features, we get back to correlation matrix to see which features are mostly correlated to the target variable;we can easily see that **Income**, **MntMeatProducts** and **MntCoffee** have highest correlations with target variable.If we train a mulitple linear regression model on these three features, we get:

$$RMSE = 5.272599515946127$$

$$R2score = 0.5414679690753761$$