Machine Learning, Fall 2020: Project 2

1.Linear Regression (15pt)
In this exercise, you will implement a linear regression model to predict the house price. For this exercises use the dataset from the link below. Only use a single feature for you regression model and explain your reasons for selecting that feature. Please explain the data setting and experimental setup similar to Project 1.The key components of your linear regression model are the cost function and gradient decent method to update the weights . https://www.kaggle.com/mayanksrivastava/predict-housing-prices-simple-linear-regression/ data

Dataset Details

It contains 21613 training data points and 21 features that might help us predict the selling price of a house.

Data Preprocessing

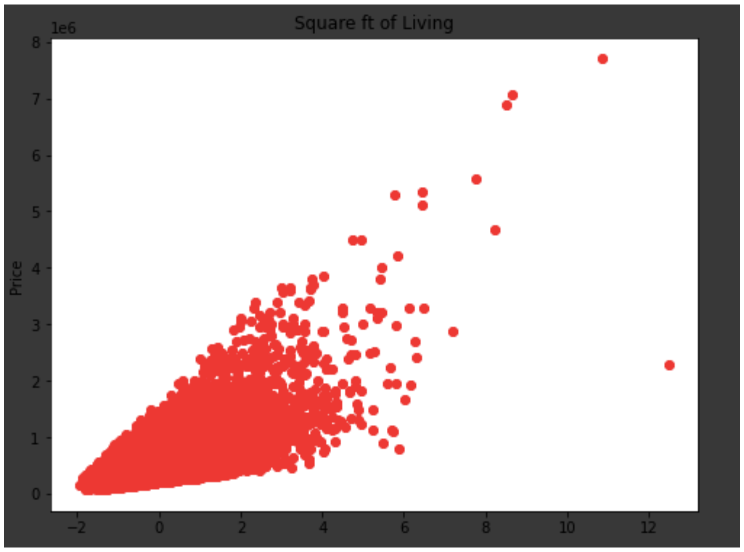
For training and testing purpose I am taking square ft of living and price data only as it is the most significant correlated feature present in the database which affects the price of any house.

Standardization(applied on sqft of living feature)
The algorithm should not be biased towards variables with higher magnitude. To overcome this problem, we can bring down all the variables to the same scale. One of the most common technique to do so is normalization where we calculate the mean and standard deviation of the variable. Then for each observation, we subtract the mean and then divide by the standard deviation of that variable:
x = (x-u)/sigma

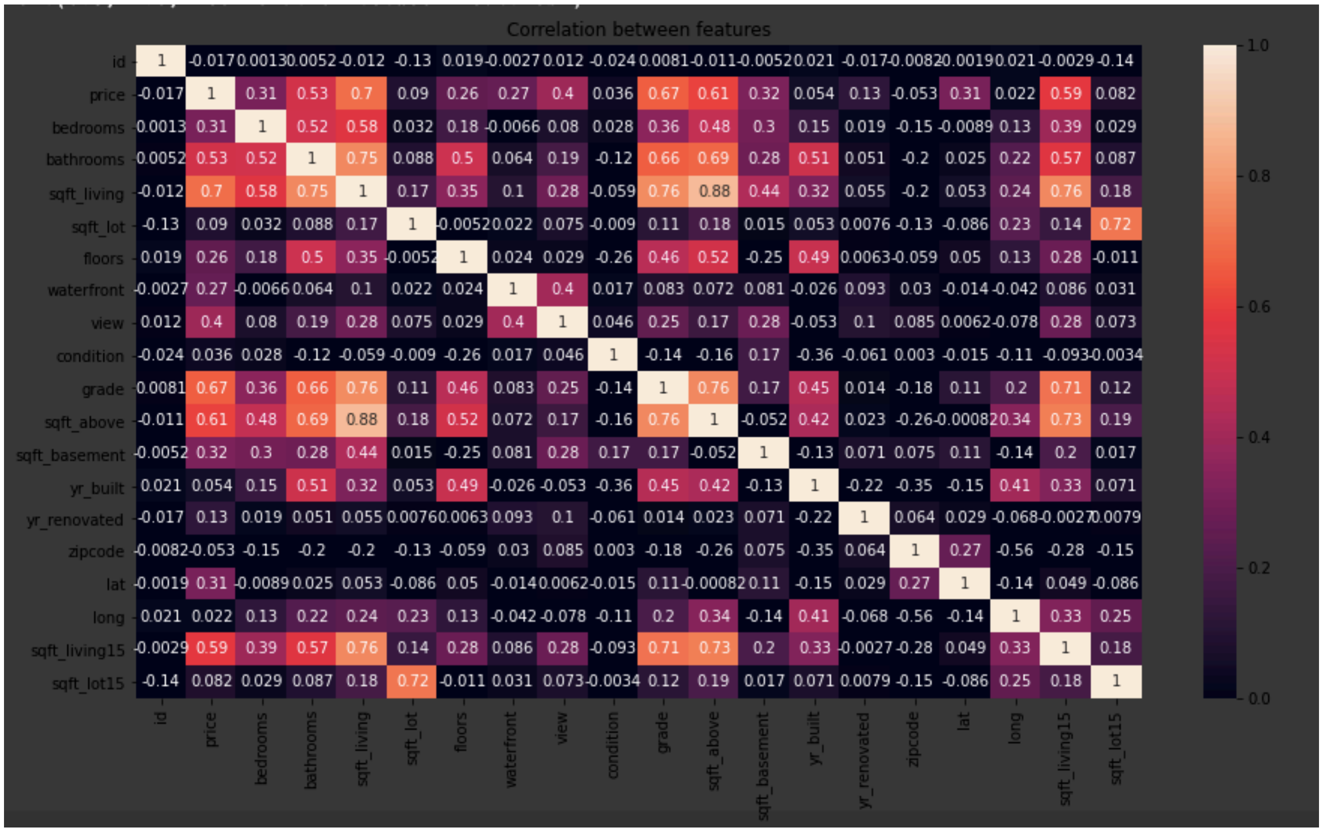
$$X' = \frac{X - \mu}{\sigma}$$

Where x is the original feature vector, 'u' is the mean of that feature vector, and sigma is its standard deviation.

Data Visualization



Correlation Between Features
An significant part of the method of data processing is the testing of associations. This analysis is one of the tools used to assess which attributes most influence the target variable, and is used in the prediction. Here we found that sqf_living is main feature which plays important role in prediction of outcome variable in this dataset that is why we are using only that feature to train the model and do predictions.



Algorithm Description

In statistics, linear regression is a linear approach to modelling the relationship between a dependent variable and one or more independent variables. Let **X** be the independent variable and **Y** be the dependent variable. We will define a linear relationship between these two variables as follows:

$$Y = mX + c$$

This is the equation for a line that you studied in high school. **m** is the slope of the line and **c** is the y intercept. we will use this equation to train our model with a given dataset and predict the value of **Y** for any given value of **X**. Our challenge today is to determine the value of **m** and **c**, such that the line corresponding to those values is the best fitting line or gives the minimum error.

Loss Function/Cost Function:

The loss is the error in our predicted value of **m** and **c**. Our goal is to minimize this error to obtain the most accurate value of **m** and **c**.

We will use the Mean Squared Error function to calculate the loss.

$$E = \frac{1}{n}\sum_{i=0}^{n}(y_i - (mx_i + c))^2$$

The Gradient Descent Algorithm

Gradient descent is an iterative optimization algorithm to find the minimum of a function. Here that function is our Loss Function.

Let’s try applying gradient descent to **m** and **c** and approach it step by step:

1. Initially let m = 0 and c = 0. Let L be our learning rate. This controls how much the value of **m** changes with each step. L could be a small value like 0.0001 for good accuracy.
2. Calculate the partial derivative of the loss function with respect to m, and plug in the current values of x, y, m and c in it to obtain the derivative value **D**.

$$D_m = \frac{1}{n}\sum_{i=0}^{n} 2(y_i - (mx_i + c))(-x_i)$$

$$D_m = \frac{-2}{n}\sum_{i=0}^{n} x_i(y_i - \bar{y}_i)$$

Derivative with respect to m

D_m is the value of the partial derivative with respect to **m**. Similarly lets find the partial derivative with respect to **c**, Dc :

$$D_c = \frac{-2}{n} \sum_{i=0}^{n} (y_i - \bar{y}_i)$$

Derivative with respect to c

3. Now we update the current value of **m** and **c** using the following equation:

$$m = m - L \times D_m$$

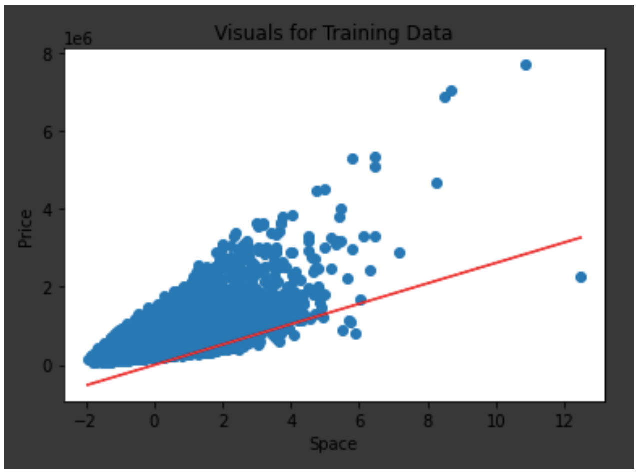
$$c = c - L \times D_c$$

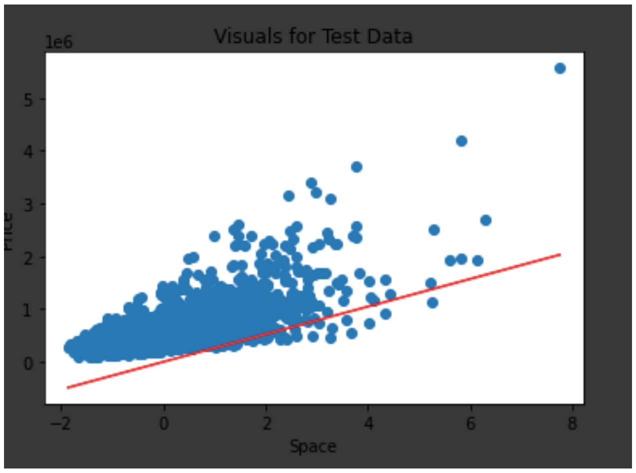
4. We repeat this process until our loss function is a very small value or ideally 0 (which means 0 error or 100% accuracy). The value of **m** and **c** that we are left with now will be the optimum values.

Algorithm Results

Ipnyb File is present in Linear. Regression folder
Model is trained on 1000 epochs with 0.01 Learning rate
Runtime of the program is 6.45s
Mean Absolute Error: 532974.2786085622

I have tried with multiple epochs and learning rate but still the Mean Absolute Error is coming to be really high which needs to be small or minimised to get better predictions. I have tried learning rates 0.01,0.001,0.0001 and epochs 1500,2000,5000 as well.





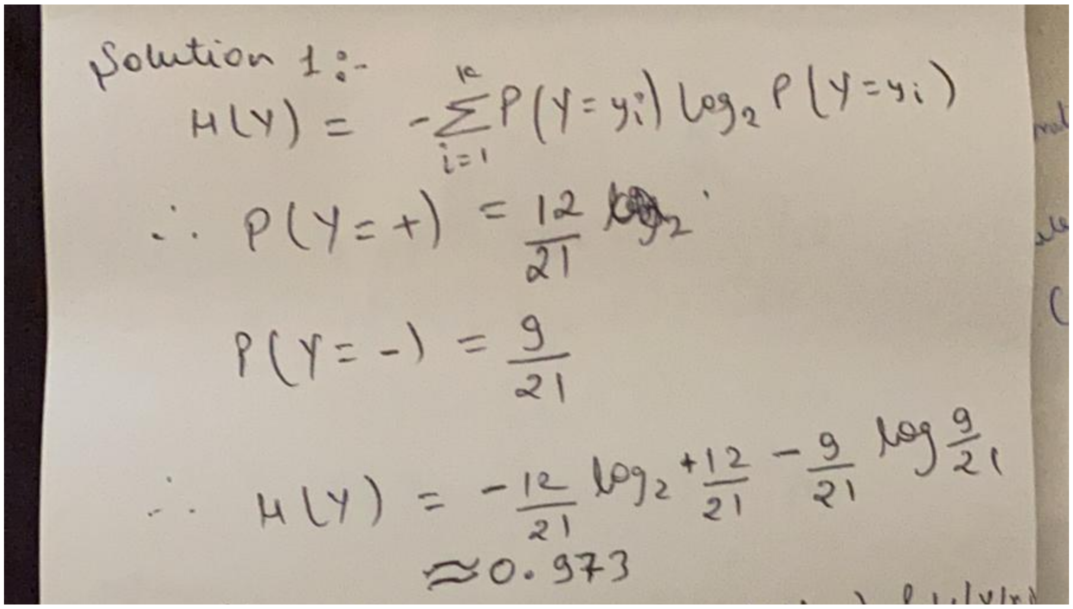
2 Decision Trees (15 pt)

2.1 ID3

Consider the following set of training examples for the unknown target function $< X_1, X_2 > \rightarrow Y$.

Y	X_1	X_2	Count
+	T	T	3
+	T	F	4
+	F	T	4
+	F	F	1
-	T	T	0
-	T	F	1
-	F	T	3
-	F	F	5

1. What is the sample entropy $H(Y)$ for this training data (with logarithms base 2)?


Solution 1:-

$$H(Y) = -\sum_{i=1}^{k} P(Y=y_i)\log_2 P(Y=y_i)$$

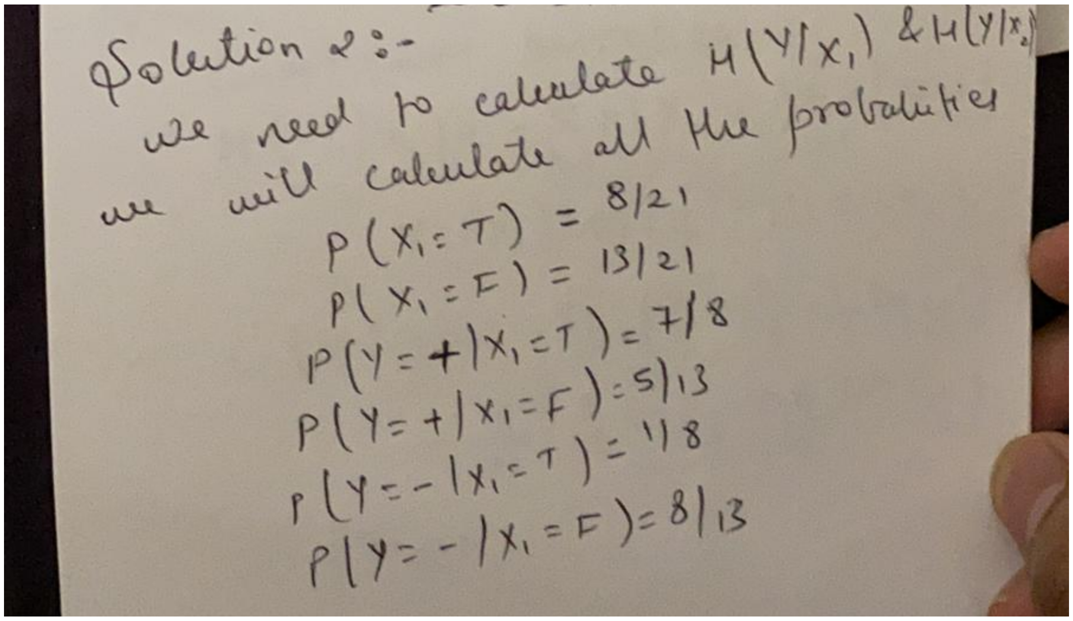
$$\therefore P(Y=+) = \frac{12}{21}\cancel{\log}_2$$

$$P(Y=-) = \frac{9}{21}$$

$$\therefore H(Y) = -\frac{12}{21}\log_2 + \frac{12}{21} - \frac{9}{21}\log\frac{9}{21}$$

$$\approx 0.973$$

2. What are the information gains $IG(X_1) \equiv H(Y) - H(Y|X_1)$ and $IG(X_2) \equiv H(Y) - H(Y|X_2)$ for this sample of training data?


Solution 2:-

we need to calculate $H(Y/X_1)$ & $H(Y/X_2)$

we will calculate all the probabilities

$$P(X_1=T) = 8/21$$

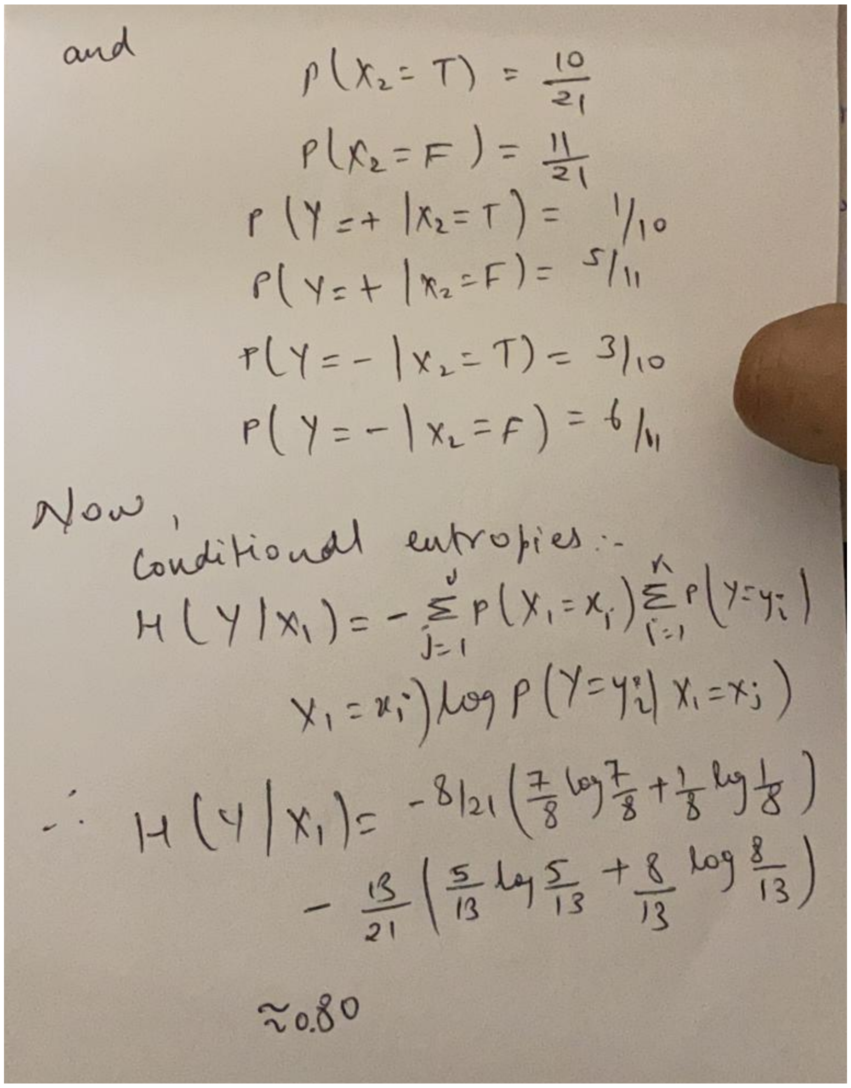
$$P(X_1=F) = 13/21$$

$$P(Y=+|X_1=T) = 7/8$$

$$P(Y=+|X_1=F) = 5/13$$

$$P(Y=-|X_1=T) = 1/8$$

$$P(Y=-|X_1=F) = 8/13$$



and

$$p(X_2 = T) = \frac{10}{21}$$

$$p(X_2 = F) = \frac{11}{21}$$

$$p(Y = + | X_2 = T) = 1/10$$

$$p(Y = + | X_2 = F) = 5/11$$

$$p(Y = - | X_2 = T) = 3/10$$

$$p(Y = - | X_2 = F) = 6/11$$

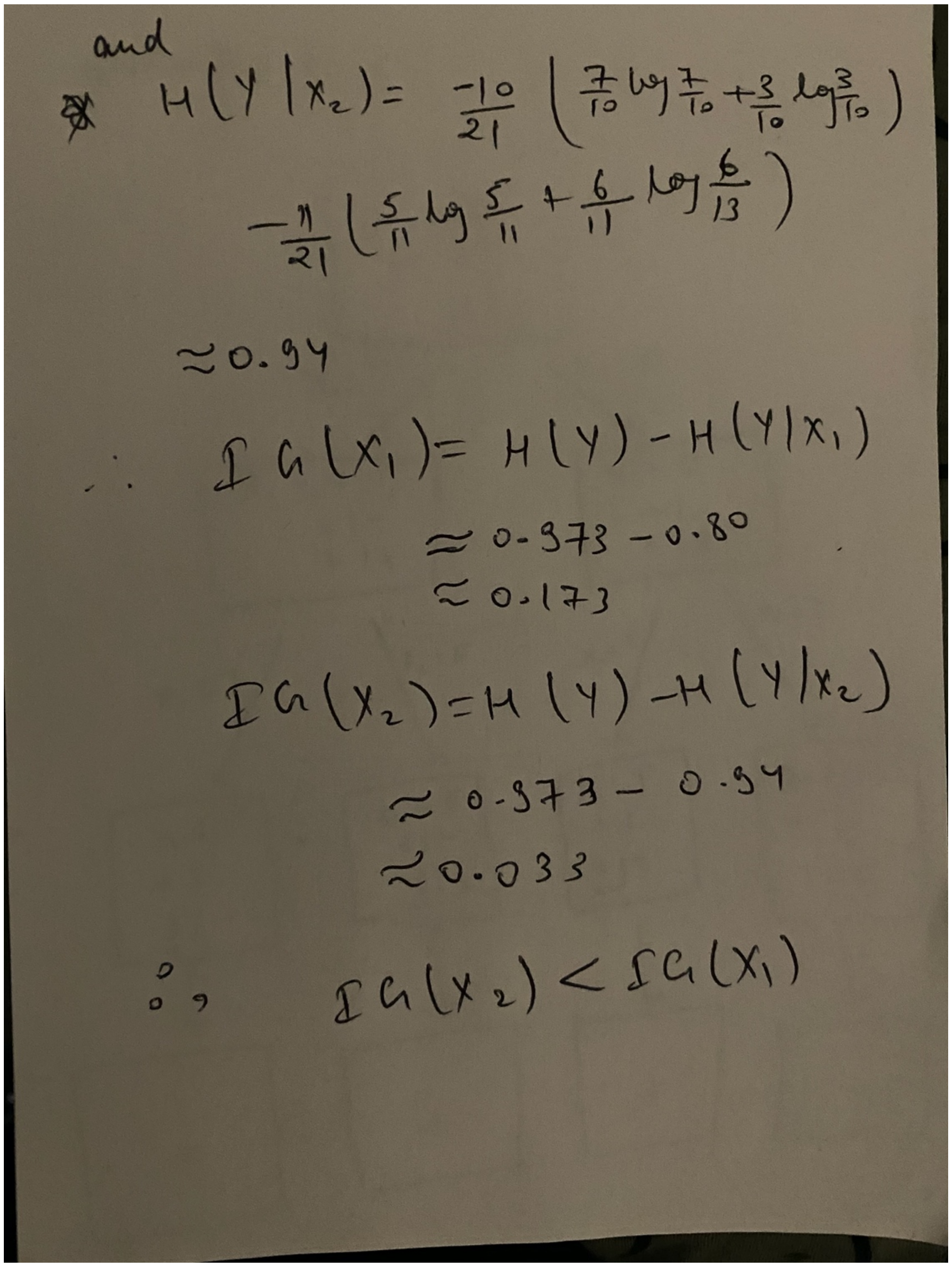
Now,

conditional entropies :-

$$H(Y|X_1) = -\sum_{j=1}^{v} p(X_1 = x_i) \sum_{i=1}^{k} p(Y = y_i | X_1 = x_i) \log p(Y = y_i | X_1 = x_j)$$

$$\therefore H(Y|X_1) = -8/21\left(\frac{7}{8}\log\frac{7}{8} + \frac{1}{8}\log\frac{1}{8}\right) - \frac{13}{21}\left(\frac{5}{13}\log\frac{5}{13} + \frac{8}{13}\log\frac{8}{13}\right)$$

$$\approx 0.80$$



and

$$H(Y|X_2) = \frac{-10}{21}\left(\frac{7}{10}\log\frac{7}{10} + \frac{3}{10}\log\frac{3}{10}\right)$$

$$-\frac{11}{21}\left(\frac{5}{11}\log\frac{5}{11} + \frac{6}{11}\log\frac{6}{13}\right)$$

$$\approx 0.94$$

$$\therefore \quad IG(X_1) = H(Y) - H(Y|X_1)$$

$$\approx 0.973 - 0.80$$

$$\approx 0.173$$

$$IG(X_2) = H(Y) - H(Y|X_2)$$

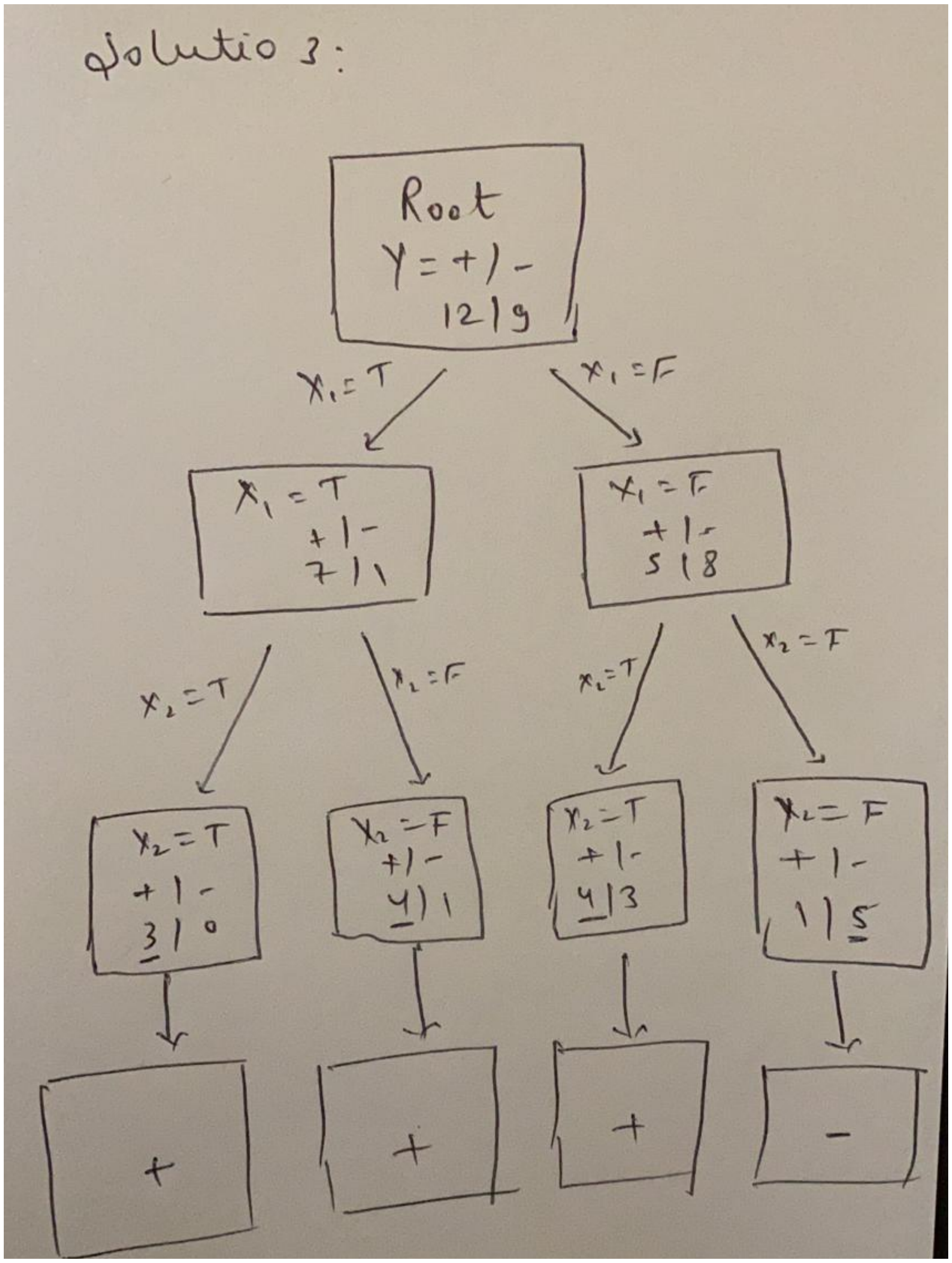
$$\approx 0.973 - 0.94$$

$$\approx 0.033$$

$$\therefore \quad IG(X_2) < IG(X_1)$$

IG(x1) is greater than IG(x2) so we will split the tree first by x1 and then we will use x2.

3. Draw the decision tree that would be learned by ID3 (without postpruning) from this sample of training data.



3 Perceptron (35 pt)

Please use the notebook provided for you to complete the perceptron exercise.

4 Support Vector Machine (35 pt)

In this problem, you will repeat the format of Project 1 but using an SVM. On the Breast Cancer Wisconsin (Diagnostic) Data Set. See associated link. http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29

Dataset Details

Breast Cancer Wisconsin (Diagnostic) Data Set

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image

Attribute Information:

ID number
Diagnosis (M = malignant, B = benign)
Ten real-valued features are computed for each cell nucleus:
radius (mean of distances from center to points on the perimeter)
texture (standard deviation of gray-scale values)
perimeter
area
smoothness (local variation in radius lengths)
compactness (perimeter^2 / area - 1.0)
concavity (severity of concave portions of the contour)
concave points (number of concave portions of the contour)
symmetry
fractal dimension ("coastline approximation" - 1)

Data Preprocessing

Converted Class variable 2,4 to 1 and -1 where 1 stands for Benign and -1 stands for Malignant. For doing svm we need the classes to be in 1 and -1 value only.

Standardization

The algorithm should not be biased towards variables with higher magnitude. To overcome this problem, we can bring down all the variables to the same scale. One of the most common technique to do so is normalization where we calculate the mean and standard deviation of the variable. Then for each observation, we subtract the mean and then divide by the standard deviation of that variable:
x = (x-u)/sigma

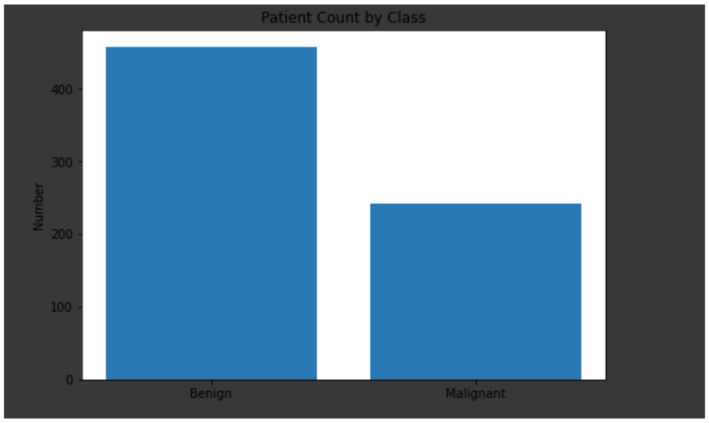
$$X' = \frac{X - \mu}{\sigma}$$

Where x is the original feature vector, 'u' is the mean of that feature vector, and sigma is its standard deviation.

Data Visualization

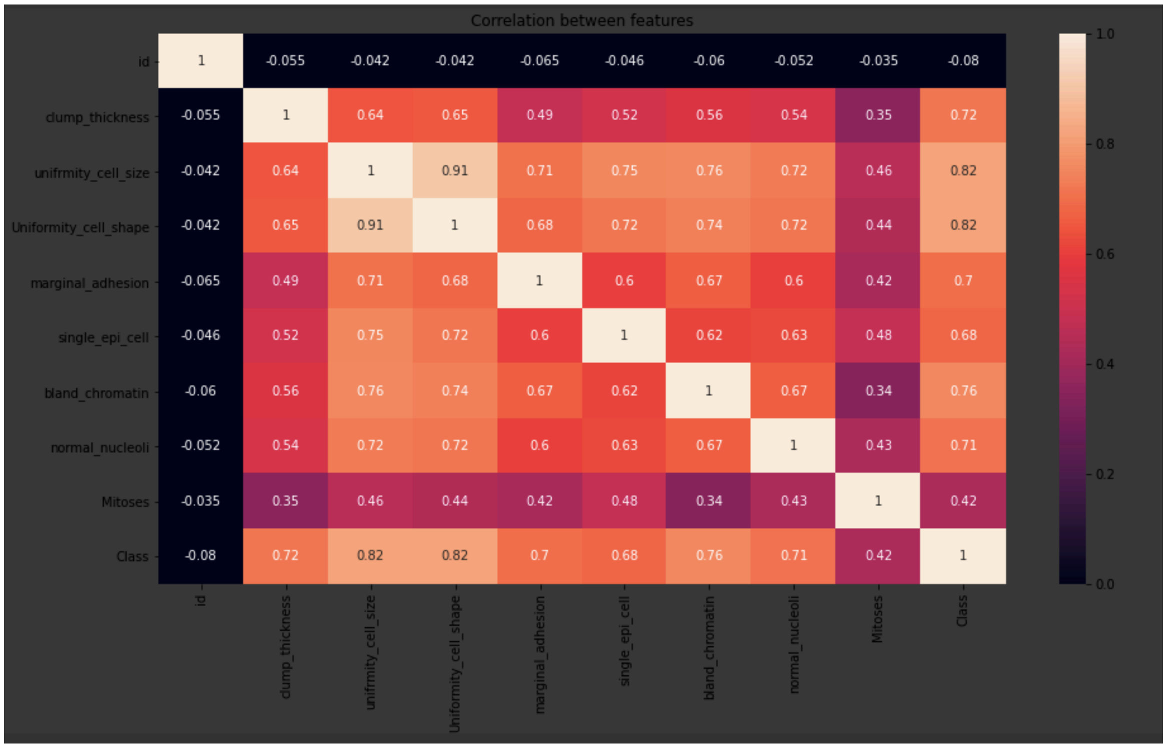
Class

Class variable (2 or 4) here 2 is for Benign and 4 is for Malignant. We have 458 of Benign and 241 for Malignant.



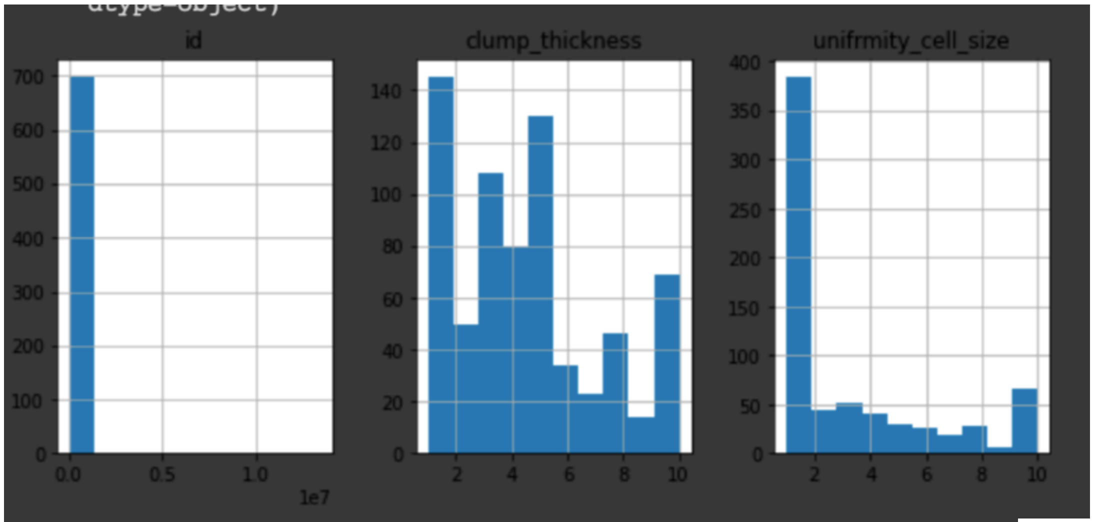
Correlation Between Features

An significant part of the method of data processing is the testing of associations. This analysis is one of the tools used to assess which attributes most influence the target variable, and is used in the prediction. Here uniformity of cell size ,uniformity of cell shape , bland chromatin are highly correlated features to predict the class. Also, other features provided in the dataset are also correlated and can be used.

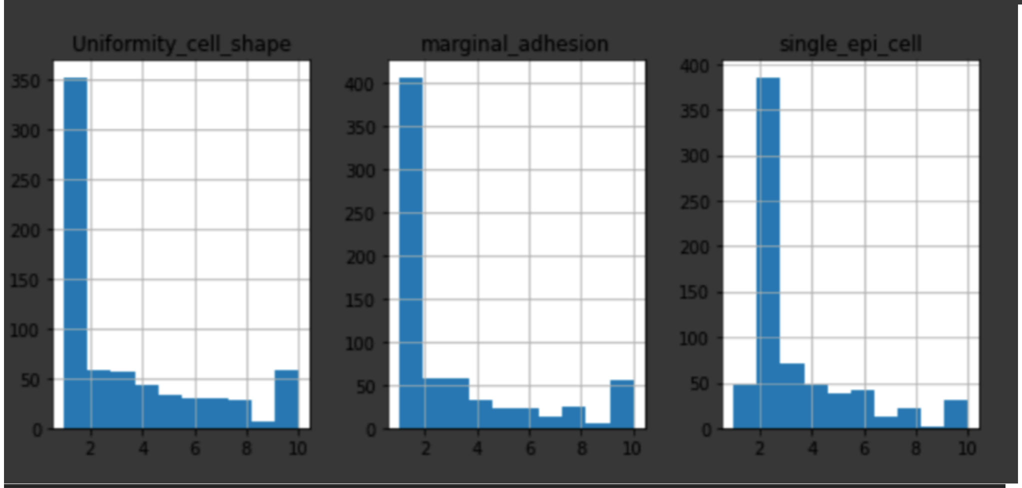


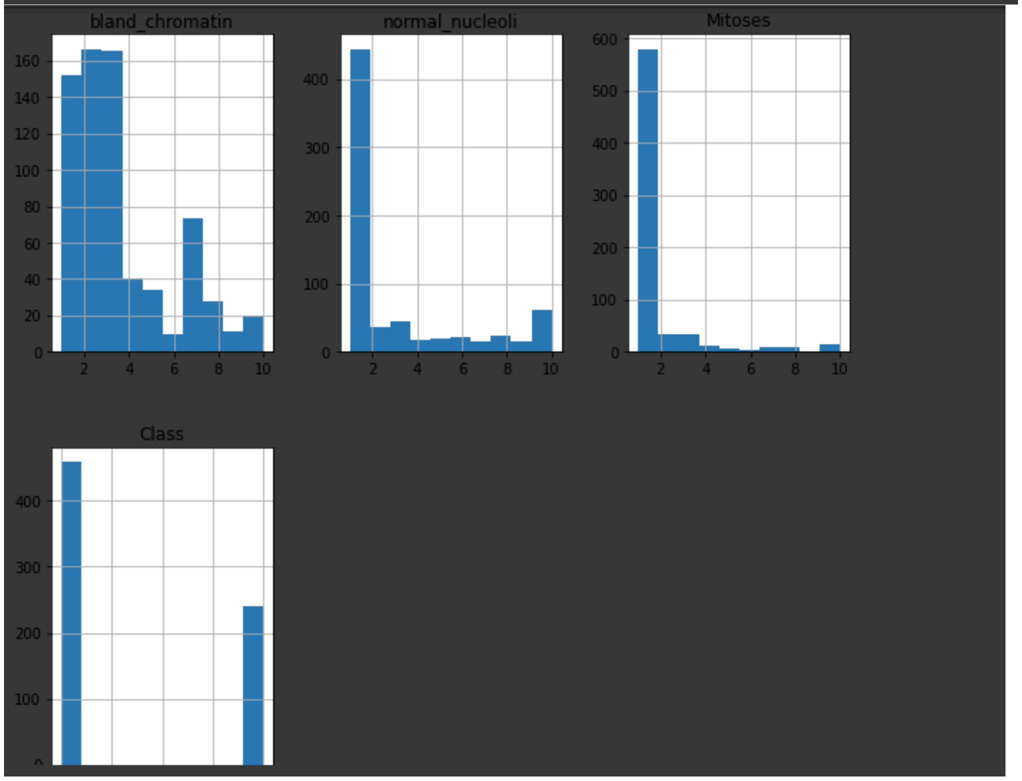
Complete Data Distribution

This is a complete representation of the distribution of data provided to us in histogram format



```
dtype=object)
```





Train-Test Split of Data
The train-test split is a technique for testing a machine learning algorithm's output. This involves a dataset being taken and divided into two parts. First part is used to train/fit the model and is referred to as the training dataset. The second part is not used to train the algorithm instead, it is used to give input element of the dataset, then predictions are made and the predicted values are compared. This second dataset is known as test dataset.
In this dataset I have choose a 20% split percentage. That is 80% would be my training set and 20% would be my test set. I have taken this split by considering several parameters like computational cost in both training and testing the model. As well as representativeness of training set.

Algorithm Description

SVM Algorithm
The SVM (Support Vector Machine) is a supervised machine learning algorithm typically used for binary classification problems. It’s trained by feeding a dataset with labeled examples (x_i, y_i).

At first approximation what SVMs do is to find a separating line(or hyperplane) between data of two classes. SVM is an algorithm that takes the data as an input and outputs a line that separates those classes if possible. According to the SVM algorithm we find the points closest to the line from both the classes.These points are called support vectors. Now, we compute the distance between the line and the support vectors. This distance is called the margin. Our goal is to maximize the margin. The hyperplane for which the margin is maximum is the optimal hyperplane. Thus SVM tries to make a decision boundary in such a way that the separation between the two classes(that street) is as wide as possible.

Calculating the Error
To calculate the error of a prediction we first need to define the objective function of the SVM.

Hinge Loss Function

To do so, we need to define the loss function, to calculate the prediction error. We will use hinge loss for our SVM:

$$c(x, y, f(x)) = (1 - y * f(x))_+$$

c is the loss function, xx the sample, yy is the true label, f(x)f(x) the predicted label.
This means the following:

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

So consider, if y and f(x) are signed values (+1,−1)(+1,−1):

- the loss is 0, if y∗f(x)y∗f(x) are positive, respective both values have the same sign.
- loss is 1−y∗f(x)1−y∗f(x) if y∗f(x)y∗f(x) is negative

Objective Function
As we defined the loss function, we can now define the objective function for the SVM:

$$\min_w \lambda \parallel w \parallel^2 + \sum_{i=1}^{n} \big(1 - y_i \langle x_i, w \rangle\big)_+$$

To minimize this function, we need the gradients.
As we have two terms, we will derive them seperately using the sum rule in differentiation.

$$\frac{\delta}{\delta w_k} \lambda \parallel w \parallel^2 \quad = 2\lambda w_k$$

$$\frac{\delta}{\delta w_k} \big(1 - y_i \langle x_i, w \rangle\big)_+ \quad = \begin{cases} 0, & \text{if } y_i \langle x_i, w \rangle \geq 1 \\ -y_i x_{ik}, & \text{else} \end{cases}$$

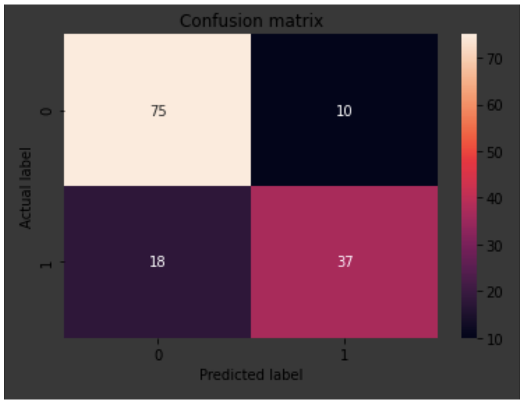
This means, if we have a misclassified sample xixi, respectively yi⟨xi,w⟩ < 1yi⟨xi,w⟩ < 1, we update the weight vector w using the gradients of both terms, if yi⟨xi,w⟩≥1yi⟨xi,w⟩≥1 we just update w by the gradient of the regularizer. To sum it up, our SGD for the SVM looks like this:

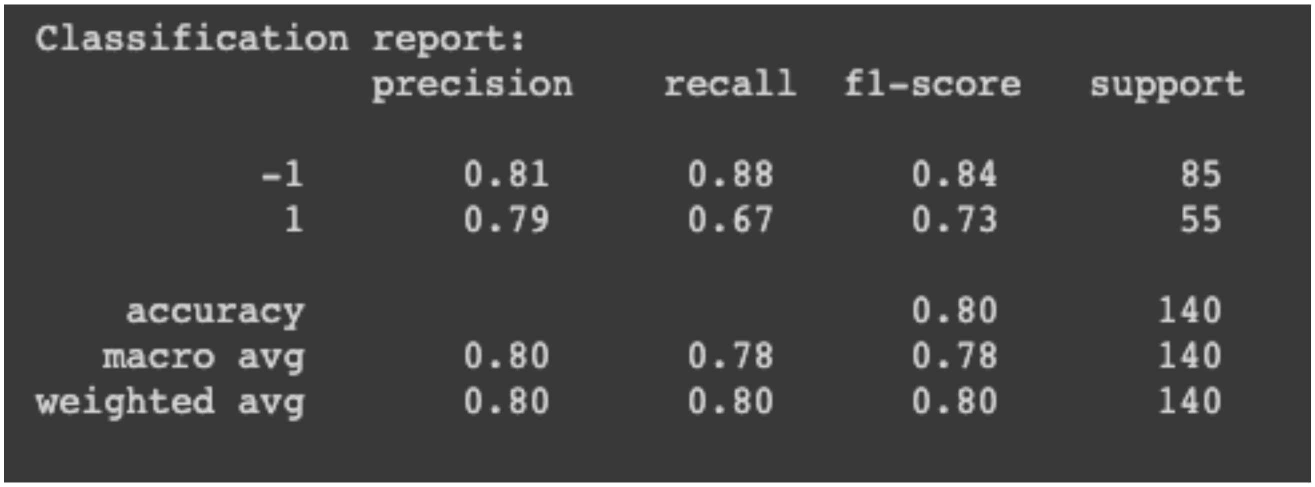
$$\text{if } y_i\langle x_i, w\rangle < 1: w = w + \eta(y_i x_i - 2\lambda w) \text{ else: } w = w + \eta(-2\lambda w)$$

Note here the difference to the perceptron. The weight vector is updated by the regularizer gradient, even if the samples is classified correctly.

Algorithm Results

Ipnyb File is present in SVM folder
Model is trained on 2000 epochs with 0.01 Learning rate
Runtime of the program is 8.002s
Accuracy: 80%
Confusion Matrix





```
Classification report:
              precision    recall  f1-score   support

          -1       0.81      0.88      0.84        85
           1       0.79      0.67      0.73        55

    accuracy                           0.80       140
   macro avg       0.80      0.78      0.78       140
weighted avg       0.80      0.80      0.80       140
```