Machine Learning Assignment – 1

Name: Kalahasti V V Deepesh

Roll Number: IMT2019508

Email ID: deepesh.vv@iiitb.org

Classification Problem:

Problem Statement:

The objective of the dataset is to predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. The dataset consists of only female patients that are at least 21 years old. All features are numerical in nature.

The 8 features given are:

Pregnancies, Glucose, Blood pressure, Skin Thickness, Insulin, BMI, age.

Objective:

Apply Logistic Regression with gradient descent, Logistic Regression with Newton's optimization and Univariate Naïve Bayes algorithms to create a Machine Learning Model that can be used to classify test input into two classes i.e., diabetic and non-diabetic.

Steps undertaken to create the ML model:

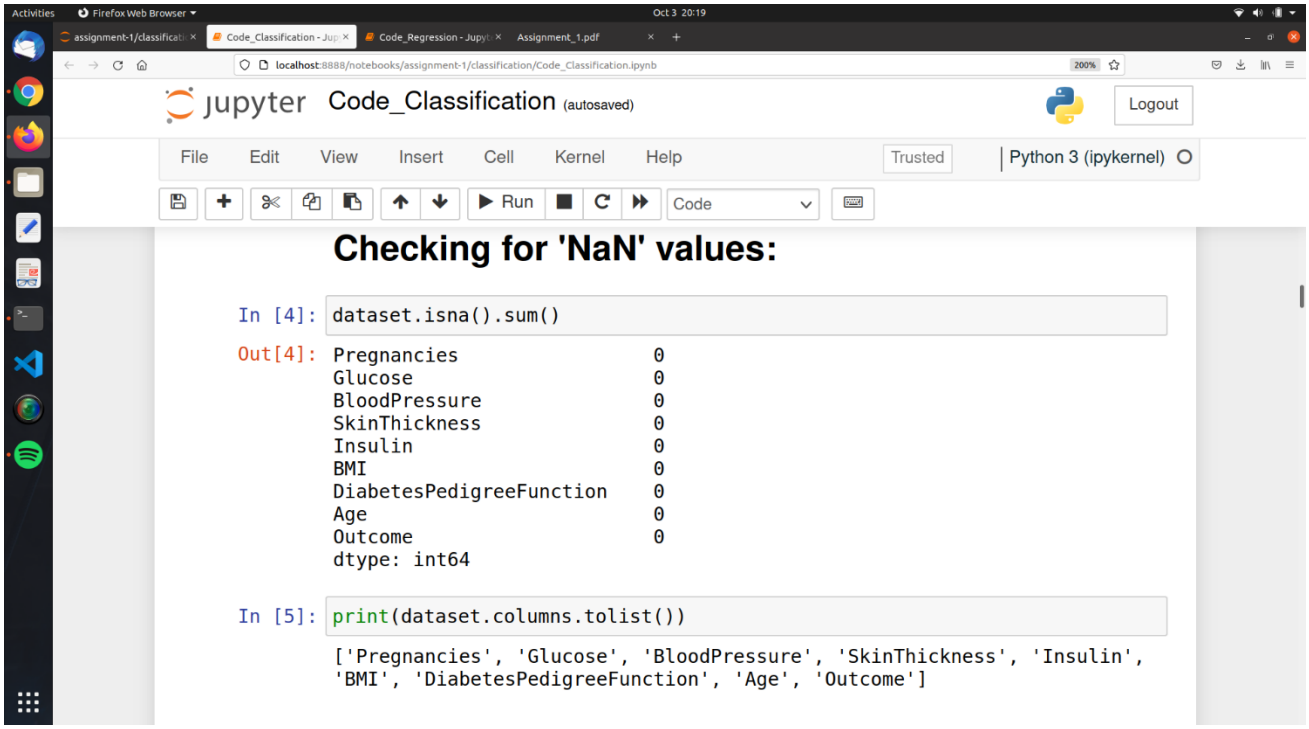
Data preprocessing:

The first step in trying to create an ML model is to make sure that your dataset does not contain invalid values, un-expected values and making sure that all of the column in the dataset contain numerical data as ML algorithms can only work on numerical data.

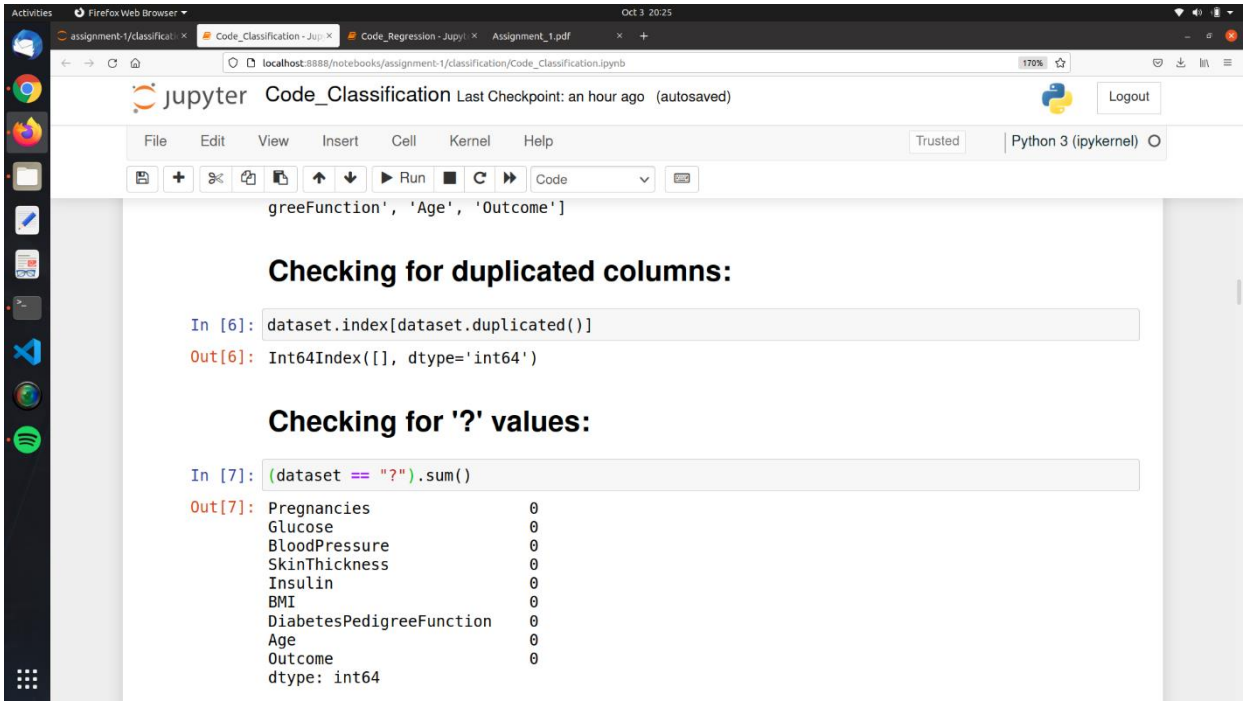
Steps taken to clean up the given dataset:

The dataset given to us is called 'diabetes.csv'.

1. First, we need to check for any 'Nan' values in the dataset. This can be done using 'pandas' library command.

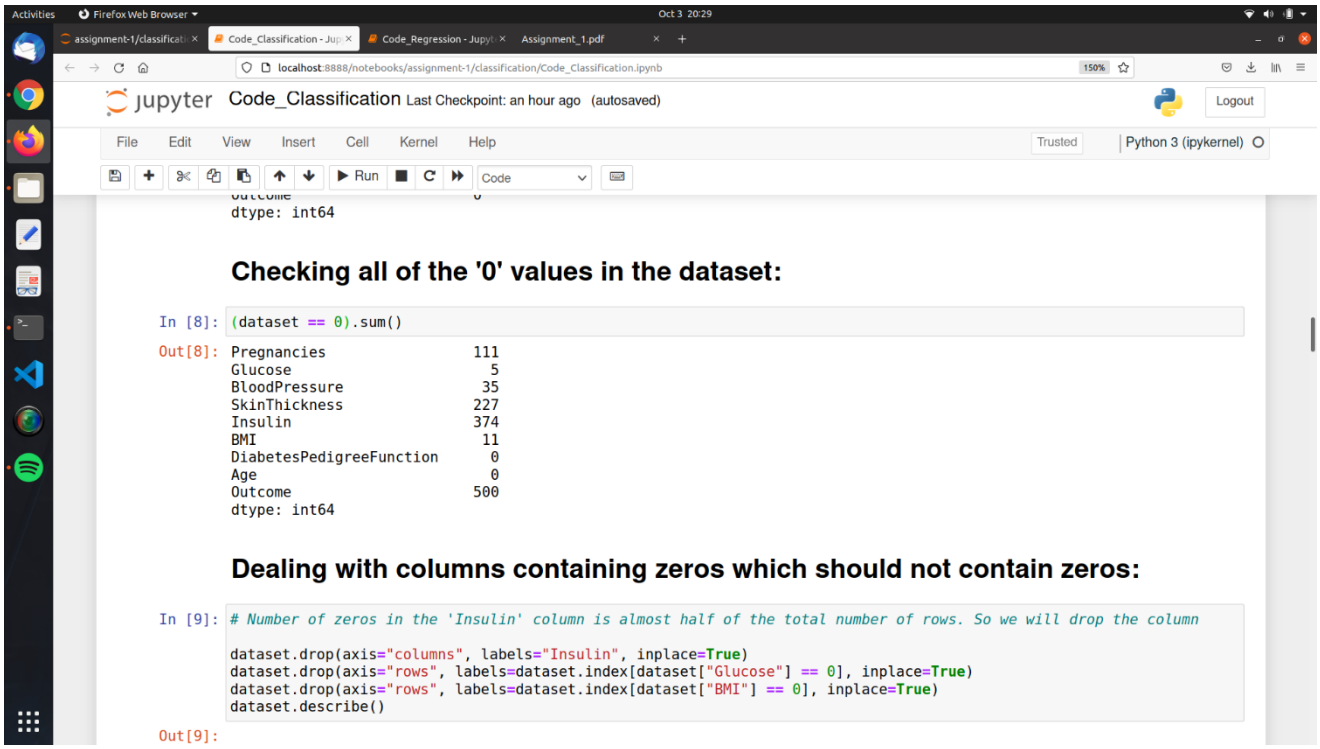


2. From the above picture we can see that there are no ‘Nan’ values in the dataset.
3. We check for any duplicate columns in the dataset. This is because duplicate columns do not provide any new data for analysis. We also check for any ‘?’ values in the dataset.



4. From the above picture we can see that there are no duplicated columns and ‘?’ values in the dataset.

5. After observing some columns in the dataset, we can see that some columns contain ‘0’ values which should not be possible.
6. The columns Glucose, Blood pressure, Skin Thickness, Insulin, BMI, age cannot contain ‘0’ values because no real person will have ‘0’ values for these particular features.



7. From the above picture, we can see there are many columns with ‘0’ values which should not contain ‘0’ values.
8. In the Insulin column, almost half of the values are ‘0’ values. This means that the column is almost useless for analysis. So we drop this column.
9. The columns Glucose, BMI have very less number of ‘0’ so we can just drop the rows containing ‘0’ values without losing too much value. For the other column we can replace the ‘0’ values with mean values during the time of model creation.

Logistic Regression with gradient descent:

As we have finished the data preprocessing, we can move on with creating out ML model.

The first model to be crated is Logistic Regression with gradient descent.

Steps undertaken to create logistic regression with gradient descent model:

1.Logistic is based on the sigmoid function. The sigmoid function is given by:

$$sigmoid(z) = \frac{1}{1 + e^{-z}}$$

Here z = w.x + b where x is the feature matrix and 'w', 'b' are parameters.

2. To create our we first split the values given to us into train sets and test sets. The implementation of this is in the Code file.

3. We take 80% of the values for training and 20% of the values for testing.

4. The gradient descent algorithm works by first taking an initial value of the parameters 'w' and 'b' and then using the gradient descent to slowly reduce the cost and reach optimum parameters.

5. The gradient descent equation is given by:

$$w_{next} = w_{prev} - \alpha * J'(w_{prev})$$

Where α is the learning rate, $J(w) is\ the\ cost\ function$.

6. This process is repeated iteratively until we reach optimum parameters and the cost function reaches a minimum.

7. The implementation of this gradient descent function is shown in the code.

Results from Logistic Regression with gradient descent:

I have used this model multiple times for various test cases and the accuracy is almost always in the range of 55% – 70%.

Logistic Regression with Newton's method for optimization:

The process of implementing logistic regression with newton's method is very similar to that of gradient descent with one key difference. The descent equation is given by:

$$w_{next} = w_{prev} - H^{-1} * J'(w_{prev})$$

Here H is called the Hessian matrix, it is given by the second of the cost function $J(w)$. For logistic regression particularly the Hessian matrix is given by:

$$H = ((1 - \hat{y})^T . \hat{y}) * (x^T . x)$$

Applying this equation in the descent calculation we can speed up our gradient descent so that it reaches the minima faster.

Results from Logistic Regression with Newton's method for optimization:

I have used this model multiple times for various test cases and the accuracy is almost always in the range of 60% – 70%.

This is shown in the code.

Observations from logistic Regression:

1. I have selected almost all features from the dataset from the dataset for analysis except the Insulin feature because the column had too many '0' values for it to be useful.
2. I have selected these because the implementation is not different for selecting one feature or multiple features, hence I selected multiple features as I thought it would give better results.
3. The accuracy of my model can be further improved by taking care of some cases where the input is much larger that the mean. If we replace these values with the mean, then the overall accuracy could imporove.

Univariate Naïve Bayes Classifier:

The naïve bayes classifier is a probabilistic classifier that works on the basis of Bayes theorem which is given is given by:

$$P\left(\frac{c}{x}\right) = \frac{P\left(\frac{x}{c}\right) * P(c)}{P(x)}$$

We can create a naïve bayes classifier using a gaussian model. The probability 'y' given x in a gaussian model is given by:

$$y = \frac{1}{\sqrt{2\pi} * \sigma} * e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

Therefore to create a univariate naïve bayes classifier from a feature variable 'x', we just need to calculate the Mean and the standard deviation of the feature vectors.

This mean and standard deviation will be the parameters of our Naïve Nayes Model. If we obtain a new 'x' variable i.e. test case then we can just put in our formula to get our predicted value.

Steps undertaken to create the ML model:

1. We take each column separately and sperate the train cases into the two sets based on the corresponding output variable.
2. We then calculate the mean and standard deviation for these sets.
3. We repeat this for all of the columns.
4. For our test cases, we calculate all of the probabilities and multiply them as this is a naïve model.
5. Then if the probability that it belongs to class '1' is greater that its probability to belong to class0 then we put it in class1 and vice-versa.

Results from Univariate Naïve Bayes Classifier:

I have used this model multiple times for various test cases and the accuracy is almost always in the range of 65% – 75%.

Observations:

1. Implementation of Univariate Naïve Bayes Classifier is simpler than that of Logistic Regression as we just have to find the mean and standard deviation of the features.

2. I has selected all of the columns together to calculate the probability. However the accuracy might have improved if I didn't choose some columns.
3. A multivariate might improve the improve the accuracy of the model. Mainly the age and column and the Blood pressure column are very closey related according to other data sources online. Hence I believe using multivariate gaussian would improve our accuracy.

Regression Problem:

Problem Statement:

The dataset consists of 21 features that represent immunization factors, mortality factors, economic factors, social factors and other health related factors in 193 countries over a span of 2000-2015.

The features given to us are:

Adult Mortality, alcohol, percentage expenditure, Hepatitis B, Measles, Under five deaths, Polio, Diphtheria, HIV/AIDS, Thinness, Income composition of resources.

Steps undertaken to create ML models:

Data preprocessing:

The data pre-processing steps undertaken for this dataset are very similar to that of the data-preprocessing steps done in the classification problem.

None of the columns contain too many Nan values compared to the total number of values so we replace all of the columns containing a larger portion of Nan values with the median of the features.

For columns which contain lesser number of Nan values we just drop the rows containing the Nan values as we wont lose too much data.

We perform one hot encoding on the ‘Status’ column as it only has two types of labels.

Hence we have now finished the data pre-processing step.

Univariate Linear Regression:

For performing univariate linear regression, I selected the 'Income composition of resources column as it seemed to correlate most with outcome feature.

The feature contained some '0' values which would have negatively affected the model therefore I replaced the '0' values with the mean.

Closed form:

Making the model through closed form is very simple. If we take 'm' and 'b' to be the parameters from y = mx+b then:

$$m, b = (x^T . x)^{-1} . x^T . y$$

Gradient descent:

Calculating through gradient for linear regression is very similar to calculating it for logistic regression. Here the formula for gradient is given by:

$$dm = \left(\frac{1}{m}\right) * (-2 * \ y^T . x + 2m * \ x^T . x + 2bx)$$

$$db = \left(\frac{1}{m}\right) * (-2y + 2mx + 2b)$$

We calculate the optimal parameters iteratively by using the above gradients.

Newton's method optimization:

Creating the model through for linear regression is also very similar to that of logistic regression. Here the Hessian matrix is given by:

$$H = 2 * \ (x^T . x)$$

Multivariate Linear Regression:

For calculating multivariate linear regression, I chose 4 features which are correlated the most with the output variable. They are:

BMI, Polio, Diphtheria and Income Composition of Resources.

The process for calculating model through closed form, gradient descent, newton's is very similar to that of univariate.

Observations:

Calculating through univariate resulted in a higher accuracy value than through multivariate.