KEDAR DESHPANDE IMT2020523

Machine Learning Assignment 1

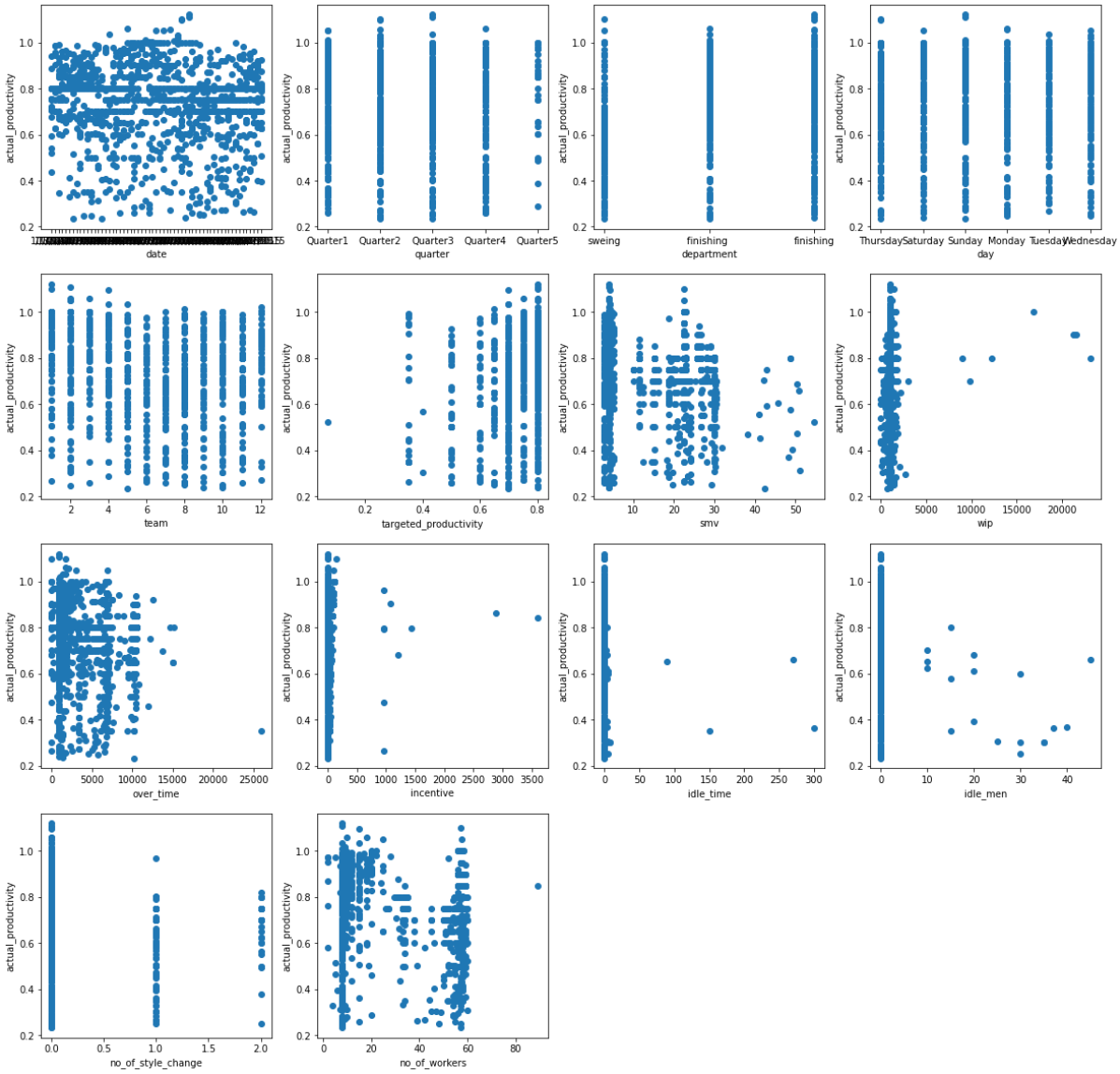
Linear Regression

Dataset- Garment workers productivity

Data analysis and preprocessing:

The column of actual_productivity is our Y (output). First we check for null values or '?' values in the dataset. There were 506 null values in wip column. I found that the median, mode and mean of the column were all close values. So I replaced the null values with the median. Checked for duplicate rows.

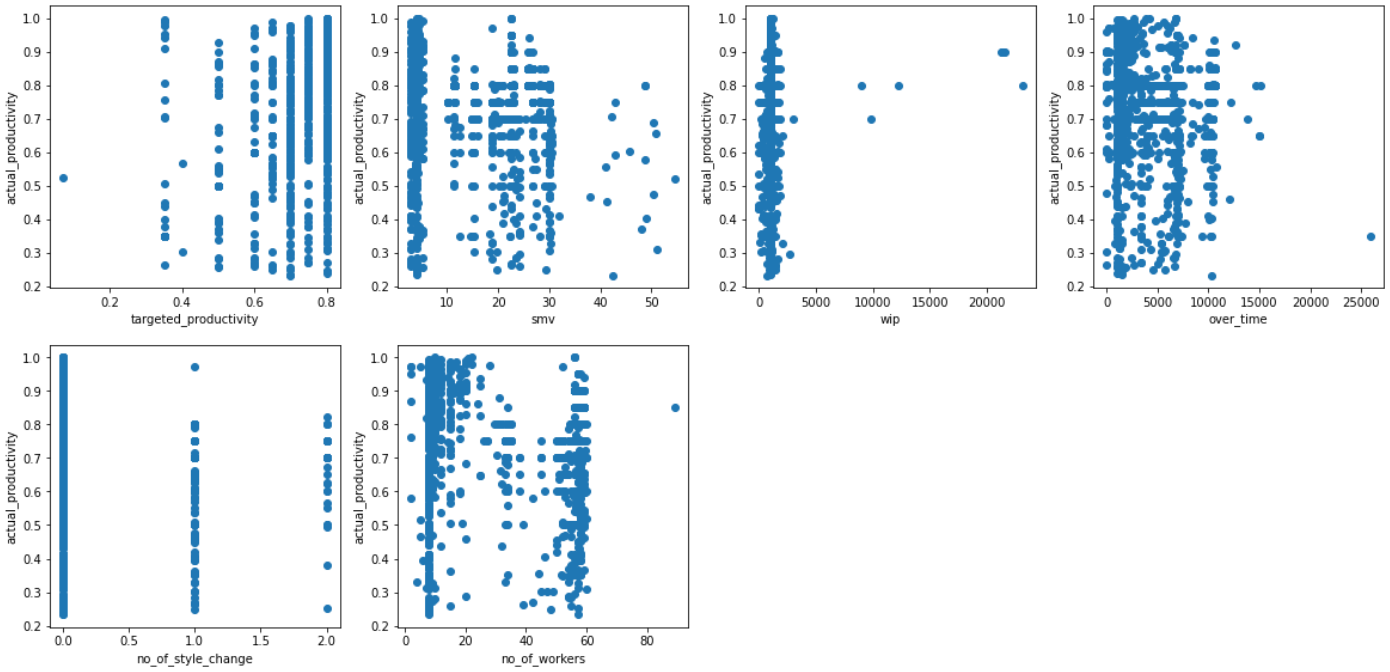
The scatter plots of each column and Y looks like:



I have dropped the categorical columns 'date', 'quarter', 'department', 'day', and 'team' as they don't seem to influence the output much from the plots. I also drop the rows where actual_productivity >1 as it's impossible. Then I drop the columns 'incentive','idle_time','idle_men' as they are concentrated around a single point only.

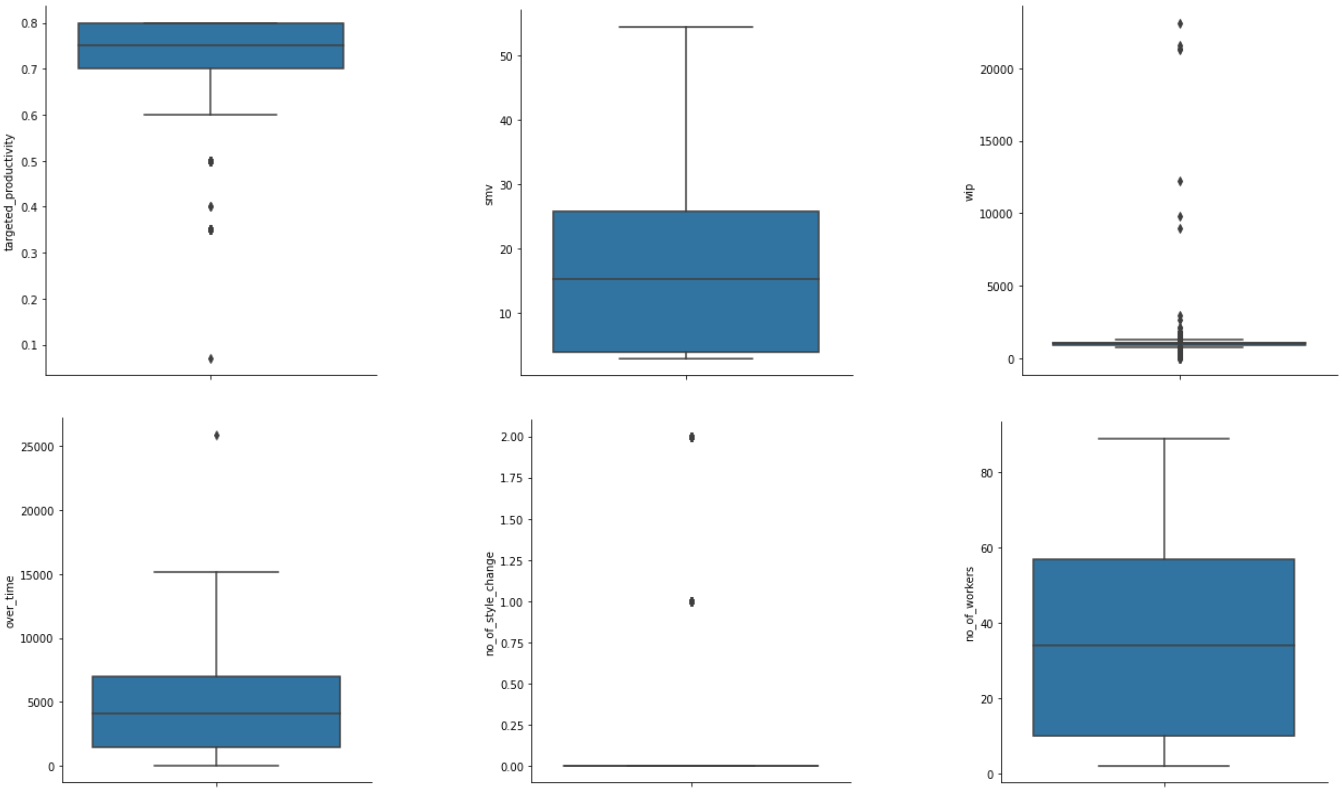
As we have dropped a few columns, we should check for duplicate rows again. There were few duplicate rows found, and were removed.

Final scatter plots:



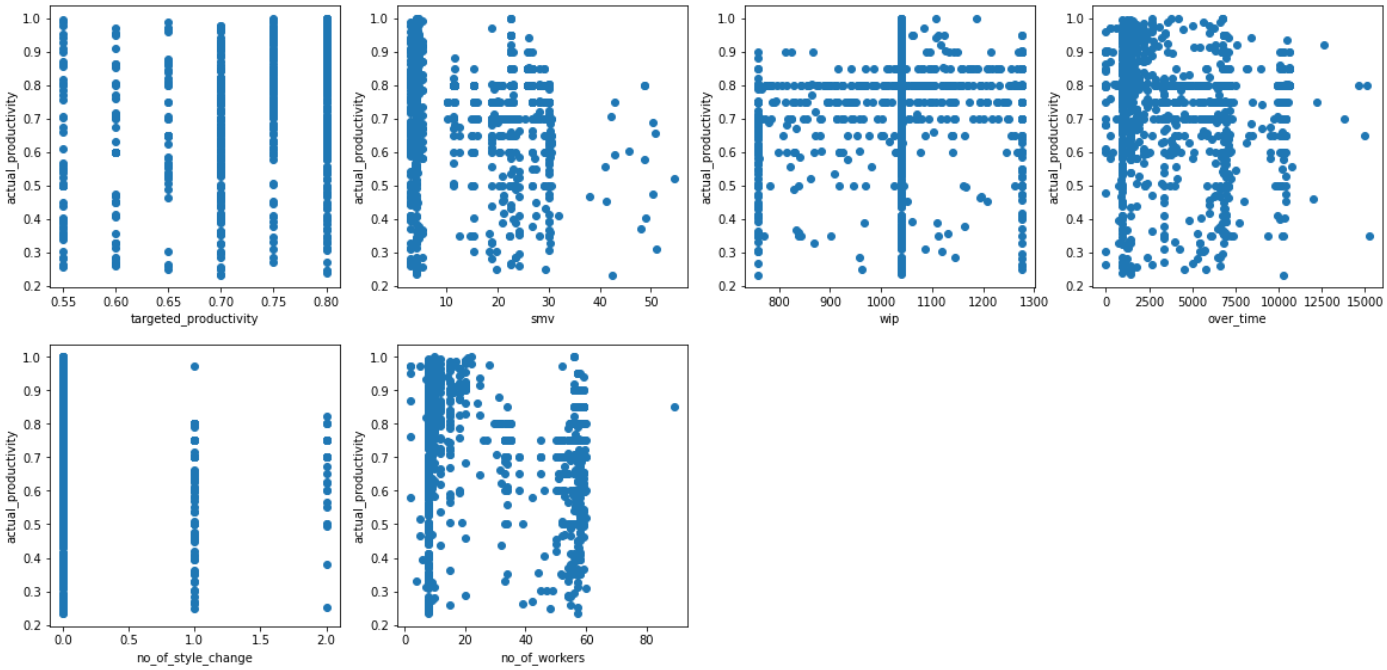
Now checked for outliers. Found them and brought them in bounds.

Following boxplots shows the outliers:



I did not deal with the outliers in the column ‘no_of_style_changes’ as there are only three unique values in that column. Checked and removed duplicate rows again.

New scatter plots:

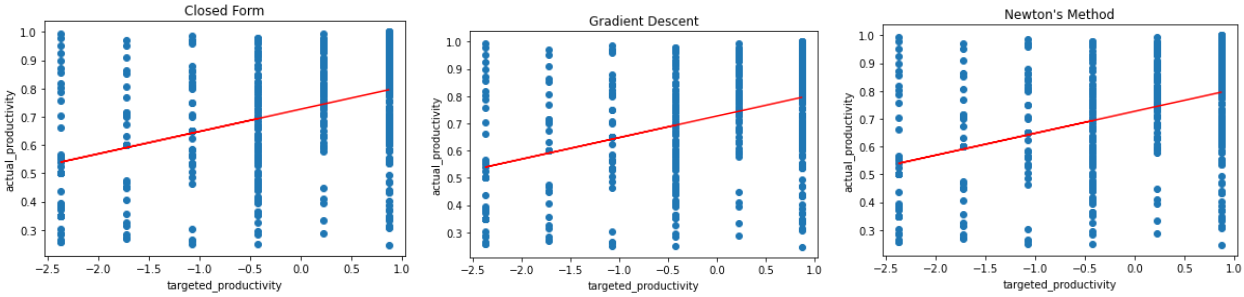


Next step was to normalize the data. After that I split the dataset into train and test (x and y each). Added a column of ones to the X matrix for w0 term. Also got separate X matrices for each feature. Wrote the implementations of **closed form, gradient descent,** and **newton's method of optimization**. Wrote the functions for MSE and MAE.

Got the weight coefficients and the results are the following:

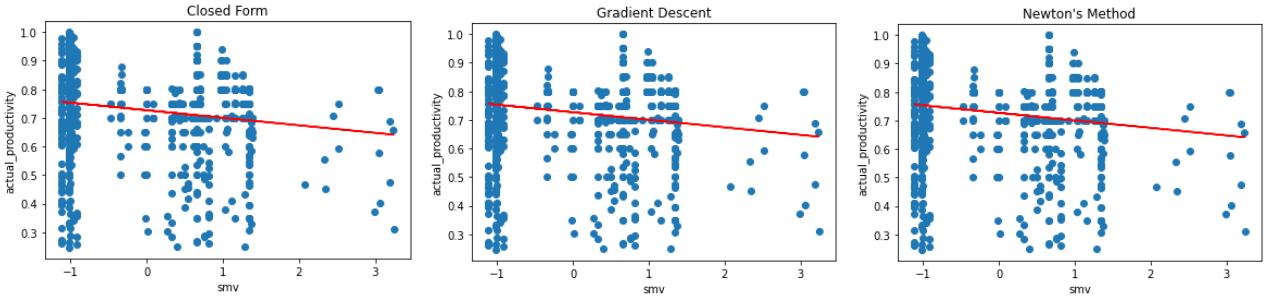
Univariate Linear Regression:

Targeted_productivity:



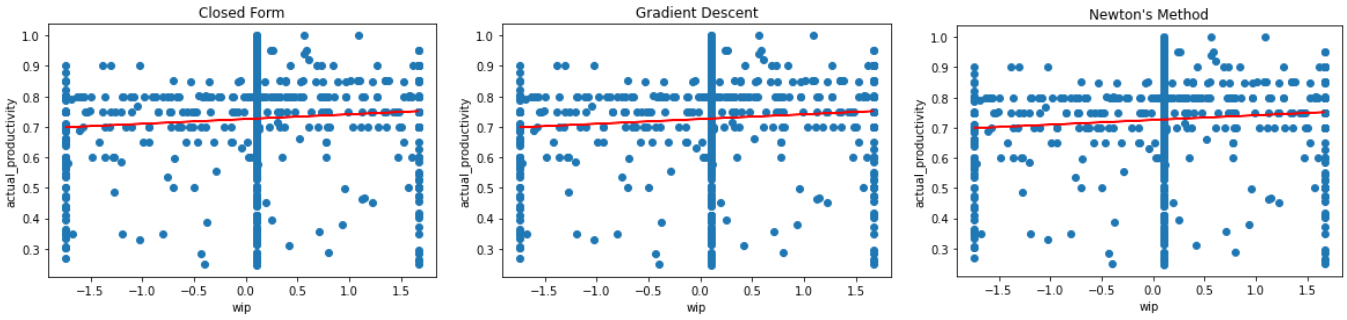

```
Errors:
1. Closed Form
          Train error:            Test error:
MSE       0.022553542290067636     0.02654536528763574
MAE       0.10207835551862805      0.11110676754956011
2. Gradient Descent
          Train error:            Test error:
MSE       0.022553542290067636     0.026545365287635746
MAE       0.10207835551862783      0.11110676754955993
3. Newton's Method
          Train error:            Test error:
MSE       0.02255354229006764      0.02654536528763575
MAE       0.10207835551862778      0.11110676754955989
```

Smv:



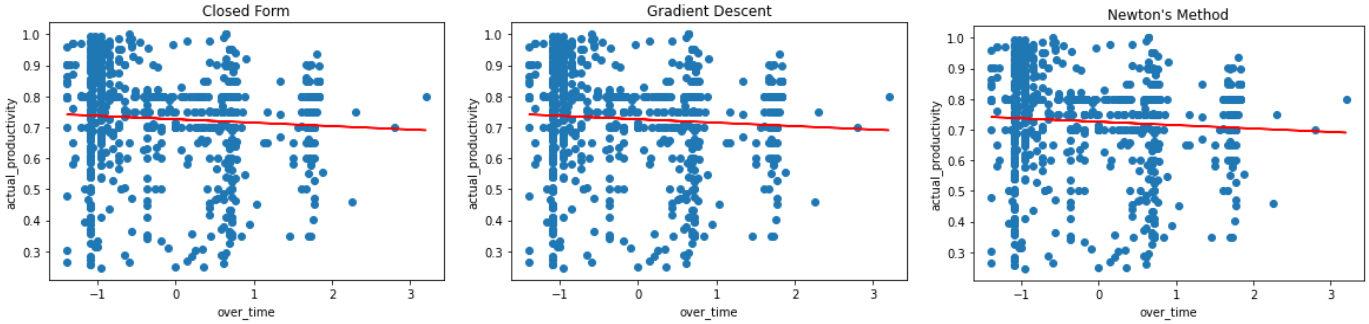

```
Errors:
1. Closed Form
         Train error:            Test error:
MSE      0.027788072277168712     0.030799431652556115
MAE      0.1299835398309292       0.13769285623902525
2. Gradient Descent
         Train error:            Test error:
MSE      0.027788072277168712     0.030799431652556105
MAE      0.12998353983092933      0.13769285623902536
3. Newton's Method
         Train error:            Test error:
MSE      0.027788072277168705     0.030799431652556126
MAE      0.12998353983092908      0.1376928562390252
```

Wip:



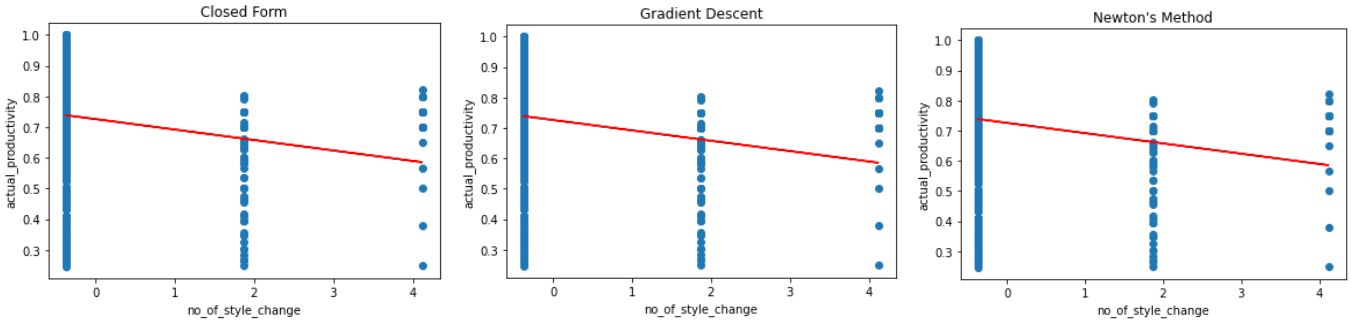

```
Errors:
1. Closed Form
         Train error:            Test error:
MSE      0.02824939535038426      0.030396887698685093
MAE      0.13031976383987484      0.13722347618130792
2. Gradient Descent
         Train error:            Test error:
MSE      0.028249395350384253     0.030396887698685083
MAE      0.13031976383987498      0.13722347618130806
3. Newton's Method
         Train error:            Test error:
MSE      0.02824939535038426      0.030396887698685093
MAE      0.13031976383987484      0.13722347618130792
```

Overtime:



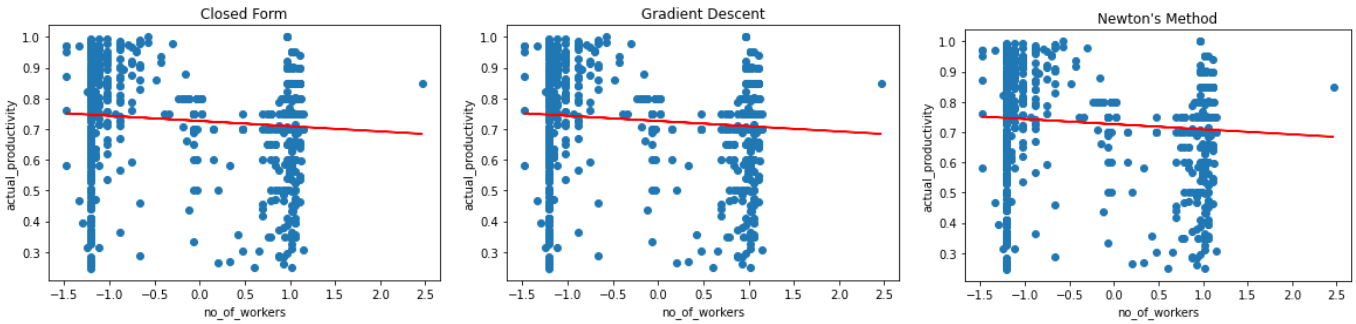
```
Errors:
1. Closed Form
         Train error:            Test error:
MSE      0.02837402570443052     0.03126892161725486
MAE      0.13118048657943154     0.1398558512710879
2. Gradient Descent
         Train error:            Test error:
MSE      0.02837402570443052     0.03126892161725486
MAE      0.13118048657943154     0.1398558512710879
3. Newton's Method
         Train error:            Test error:
MSE      0.02837402570443053     0.03126892161725488
MAE      0.1311804865794314      0.13985585127108777
```

No of style changes:



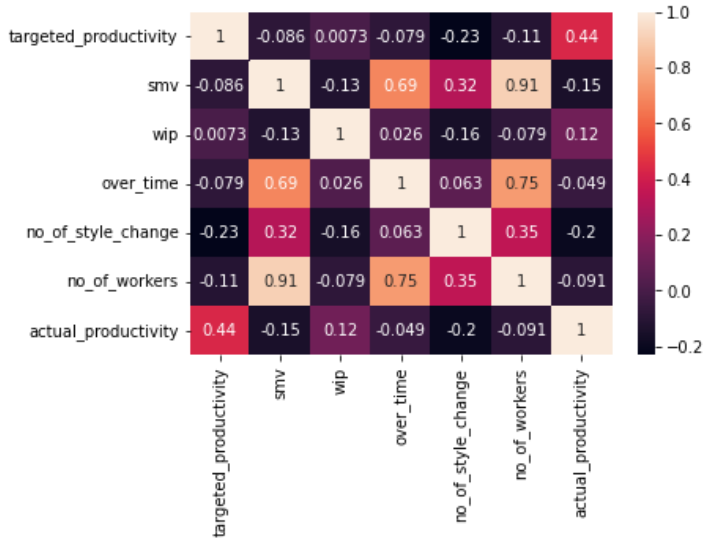

```
Errors:
1. Closed Form
         Train error:            Test error:
MSE      0.027369529551661366    0.02987984818013648
MAE      0.12869655233499344     0.13341780686239946
2. Gradient Descent
         Train error:            Test error:
MSE      0.02736952955166137     0.029879848180136455
MAE      0.12869655233499366     0.13341780686239965
3. Newton's Method
         Train error:            Test error:
MSE      0.027369529551661366    0.029879848180136472
MAE      0.12869655233499344     0.13341780686239946
```

No of workers:



```
Errors:
1. Closed Form
         Train error:            Test error:
MSE      0.028204892520006252    0.031078018532704678
MAE      0.13072064150380153     0.13883739610242538
2. Gradient Descent
         Train error:            Test error:
MSE      0.028204892520006252    0.031078018532704678
MAE      0.13072064150380153     0.13883739610242538
3. Newton's Method
         Train error:            Test error:
MSE      0.028204892520006252    0.03107801853270469
MAE      0.13072064150380136     0.13883739610242526
```

Multivariate Linear Regression:



This image shows us the correlation between the columns. I will drop the column of no_of_workers as it is highly correlated with smv (0.91) and smv is more correlated with the output. Checked and removed duplicate rows again.

Performed the train test split again (as we removed duplicate rows). Got the respective weights from all three methods and the results are:

```
Errors:
1. Closed Form
         Train error:            Test error:
MSE      0.022801745651421138    0.023172080470245344
MAE      0.10883500704184693     0.1081823885969133
2. Gradient Descent
         Train error:            Test error:
MSE      0.02280174565142113     0.023172080470245337
MAE      0.1088350070418471      0.10818238859691343
3. Newton's Method
         Train error:            Test error:
MSE      0.02280174565142113     0.02317208047024534
MAE      0.10883500704184684     0.10818238859691322
```

PTO

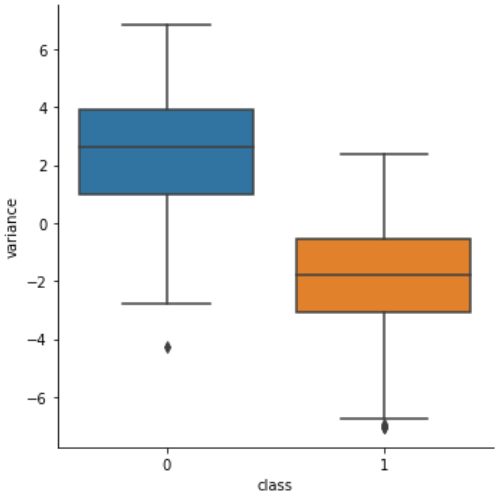
Classification

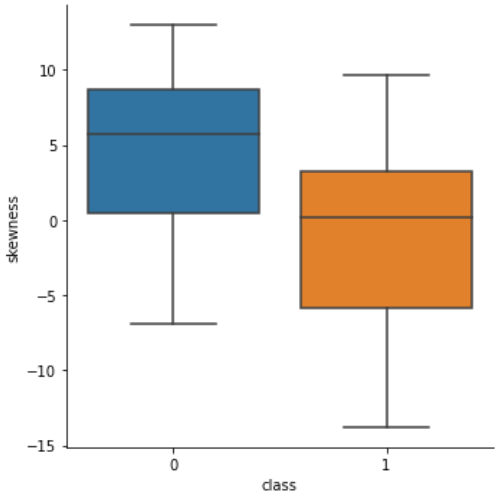
Dataset- Bank note authentication

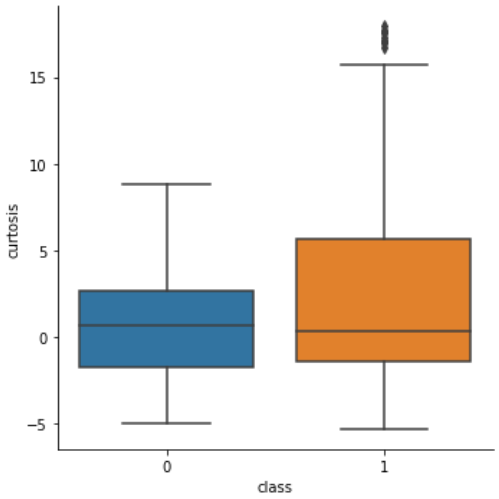
4 columns (except class): variance, skewness, curtosis, entropy

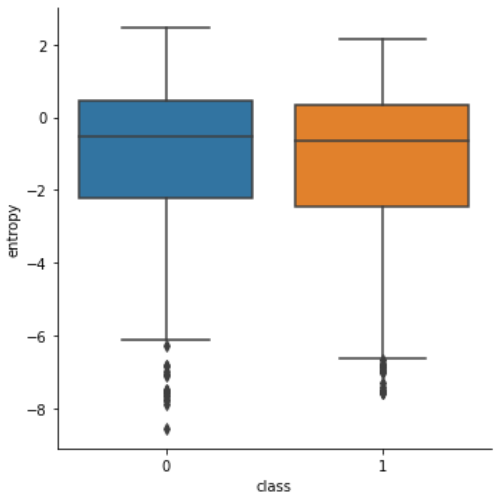
Data analysis and preprocessing:

Checked for null and '?' values. No such values found. Duplicate rows were removed. Found outliers and removed them. Following box plots show the outliers:

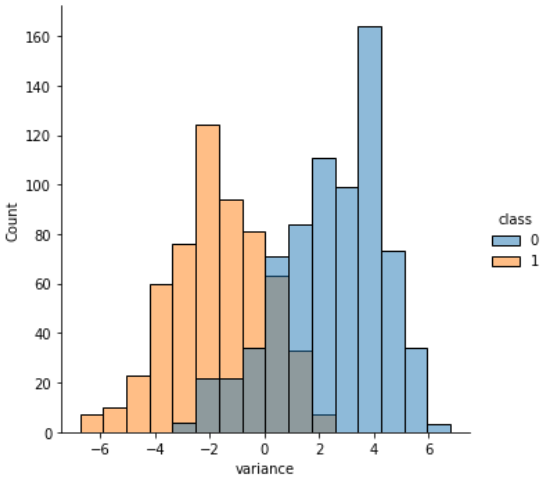


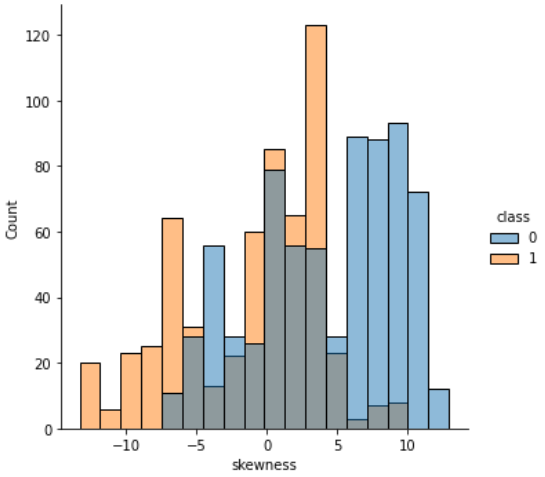


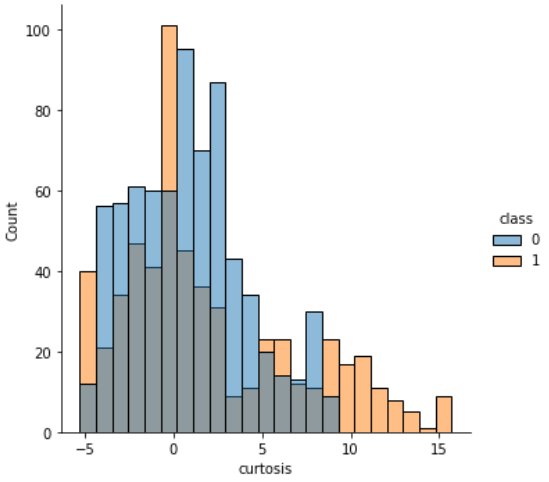


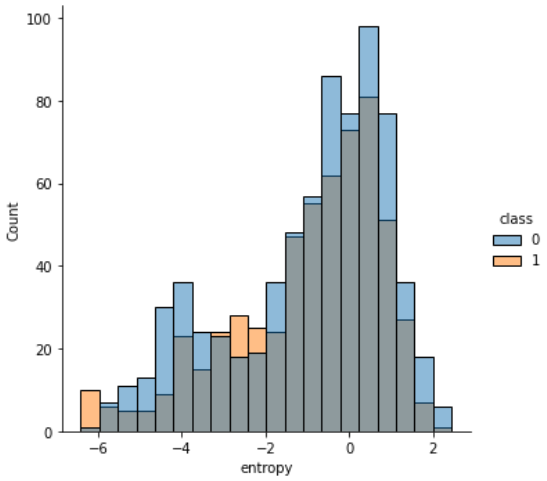


Histograms of each column:









Looks like gaussian.

Next, I normalized the data and performed train test split (70-30). Added a column of 1 to the X matrix for the w0 term. Wrote the functions to perform logistic regression on the given data using gradient descent.

Multivariate logistic regression:

Results represent: no of correct guesses, no of wrong guesses, accuracy, confusion matrix, F1 score. (For all methods)

```
findAccuracy(predict(W,train_x_final),train_y)
```

```
(906, 3, 99.66996699669967, [[419, 1], [2, 487]], 0.9964328180737217)
```

Train results

```
findAccuracy(predict(W,test_x_final),test_y)
```

```
(387, 3, 99.23076923076923, [[156, 2], [1, 231]], 0.9904761904761905)
```

Test results

Univariate Logistic regression:

Got separate X matrices for each feature/column. And got the weights.

Variance

```
[44] Wgd=gradientDescent(np.array([1]*2),train_x1,train_y,4000,0.1)
     findAccuracy(predict(Wgd,train_x1),train_y)
```

```
     (784, 125, 86.24862486248625, [[353, 67], [58, 431]], 0.8495788206979543)
```

Train results

```
[45] findAccuracy(predict(Wgd,test_x1),test_y)
```

```
     (335, 55, 85.8974358974359, [[139, 19], [36, 196]], 0.8348348348348348)
```

Test results

Skewness

```
[46] Wgd=gradientDescent(np.array([1]*2),train_x2,train_y,4000,0.1)
     findAccuracy(predict(Wgd,train_x2),train_y)
```

```
     (314, 595, 34.54345434543454, [[98, 322], [273, 216]], 0.24778761061946902)
```

Train results

```
[47] findAccuracy(predict(Wgd,test_x2),test_y)
```

```
     (126, 264, 32.30769230769231, [[33, 125], [139, 93]], 0.2)
```

Test results

Curtosis

```
[48] Wgd=gradientDescent(np.array([1]*2),train_x3,train_y,4000,0.1)
     findAccuracy(predict(Wgd,train_x3),train_y)

     (489, 420, 53.79537953795379, [[0, 420], [0, 489]], 0.0)
```

Train results

```
[49] findAccuracy(predict(Wgd,test_x3),test_y)

     (232, 158, 59.48717948717949, [[0, 158], [0, 232]], 0.0)
```

Test results

Entropy

```
[50] Wgd=gradientDescent(np.array([1]*2),train_x4,train_y,4000,0.1)
     findAccuracy(predict(Wgd,train_x4),train_y)

     (489, 420, 53.79537953795379, [[0, 420], [0, 489]], 0.0)
```

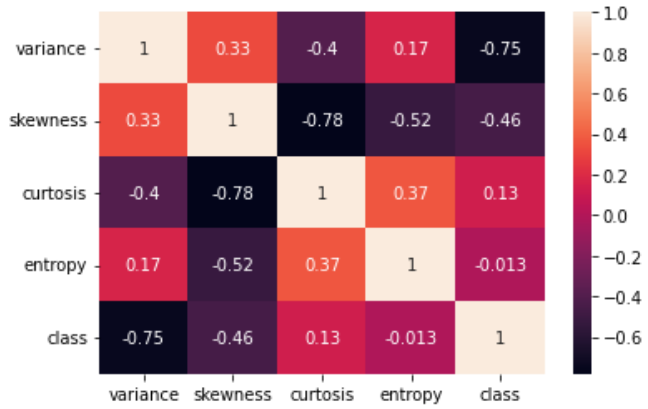
Train results

```
[51] findAccuracy(predict(Wgd,test_x4),test_y)

     (232, 158, 59.48717948717949, [[0, 158], [0, 232]], 0.0)
```

Test results

Correlation between columns:



We can see as the correlation with the output (class) decreases, our model performs more and more poorly.

Model based on entropy always predicts 0, shows how bad the model is.

PTO

Univariate Naïve Bayes:

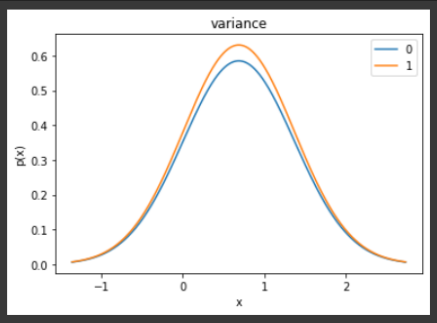
Variance

Variance

```
[29] testModel("variance",train_x,train_y)
```

```
(787, 122, 86.57865786578658, [[357, 63], [59, 430]], 0.854066985645933)
```

```
plotGaussian(m0[0],math.sqrt(v0[0]),m1[0],math.sqrt(v1[0]),"variance")
```



Train results

```
[32] testModel("variance",test_x,test_y)
```

```
(336, 54, 86.15384615384616, [[141, 17], [37, 195]], 0.8392857142857143)
```

Test results

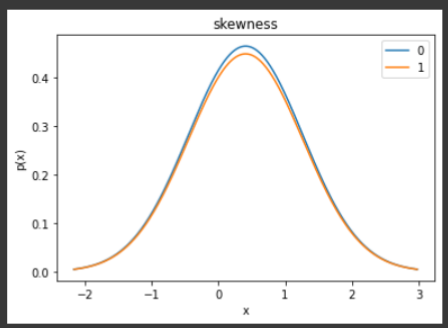
Skewness

Skewness

```
[33] testModel("skewness",train_x,train_y)
```

```
(595, 314, 65.45654565456546, [[236, 184], [130, 359]], 0.6005089058524173)
```

```
plotGaussian(m0[1],math.sqrt(v0[1]),m1[1],math.sqrt(v1[1]),"skewness")
```



Train results

```
[35] testModel("skewness",test_x,test_y)
```

```
(255, 135, 65.38461538461539, [[91, 67], [68, 164]], 0.5741324921135647)
```

Test results

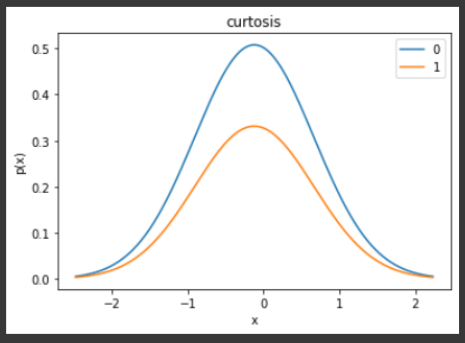
Curtosis

Curtosis

```
[36] testModel("curtosis",train_x,train_y)
```

```
(562, 347, 61.826182618261825, [[131, 289], [58, 431]], 0.4302134646962233)
```

```
plotGaussian(m0[2],math.sqrt(v0[2]),m1[2],math.sqrt(v1[2]),"curtosis")
```




```
[38] testModel("curtosis",test_x,test_y)
```

```
(253, 137, 64.87179487179488, [[49, 109], [28, 204]], 0.4170212765957467)
```

Train results

Test results

Entropy

Entropy

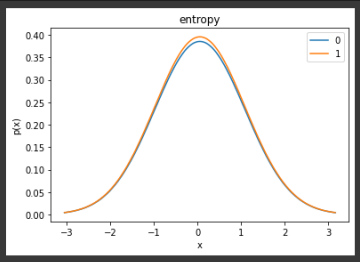
```
[39] testModel("entropy",train_x,train_y)
```

```
(489, 420, 53.79537953795379, [[0, 420], [0, 489]], 0.0)
```

As we saw earlier, entropy had very less correlation with class, it is now evident with the poor perfomance of our model.

Our model only predicts 0 here.

```
plotGaussian(m0[3],math.sqrt(v0[3]),m1[3],math.sqrt(v1[3]),"entropy")
```




```
[41] testModel("entropy",test_x,test_y)
```

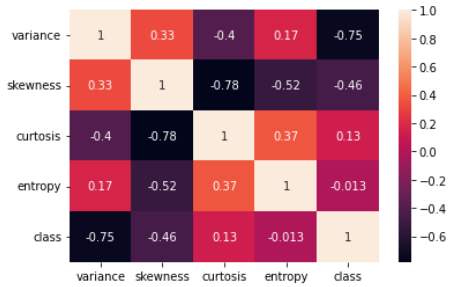
```
(232, 158, 59.48717948717949, [[0, 158], [0, 232]], 0.0)
```

Train results

Test results

Multivariate gaussian:

I have not implemented this but try to come with a few combinations of features that could give good results with multivariate gaussian. We try to have the features as mutually independent as possible. So we should have those features together that have less correlation among themselves.



So some good choices will be (for two features let's say):

- variance and entropy
- variance and skewness (would be best because corr with output is high)
- curtosis and entropy

etc.

Choosing skewness and curtosis would be a bad idea.

For 3 features:

- variance, entropy and curtosis
- variance, skewness and entropy

Thanks