JUNE 29, 2020

DIGITAL ENGINEERING PROJECT

HYPERSPECTRAL IMAGE SEGMENTATION AND ANALYSIS TOWARDS AUTOMATED NON-DESTRUCTIVE MEASUREMENTS OF OLIGOSACCHARIDES AND OTHER TRAITS

Saied Akbar Ahmady
Ramkishore Cheekatla
Ramanpreet Kaur
Amit Kumar
Sudheer Kandula
Devesh Sanghi

1 TABLE OF CONTENTS

2 INTRODUCTION

Cereals are considered as one of most import sources of food throughout the world and some of the important cereals grown widely are rice, wheat, barley, maize, millets etc. and in this project we are interested to measure agronomical traits and more specifically the compositions traits of cereals based on hyperspectral images.

We have around 480 number of hyperspectral images from each VNIR and SWIR which are taken by two cameras each with 160 bands (400-1000 nm) and 256 bands (1000-2500 nm) respectively capturing each spike. And at the meanwhile we have the lab report for those spikes, more specifically based on grains of those spikes capturing their sugar contents of each compound class, oligosaccharide, monosaccharide and oligosaccharide e.g. nystose, glucose, fructose and maltose.

In this document we are going to discuss:

- Implementation of hyperspectral image segmentation and analysis
- Development of novel AI-based approaches
- Oligosaccharides and other traits as important quality parameters
- Enabling automated non-destructive measurements

3 DATASET

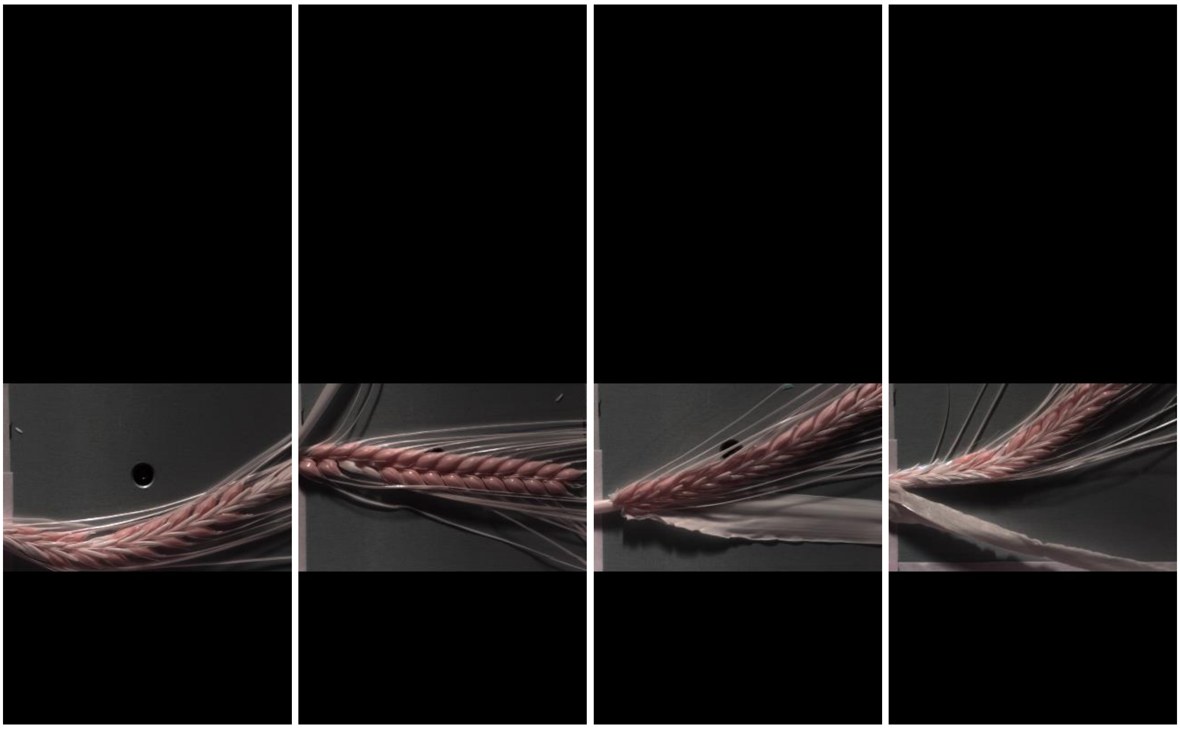
There are generally 3 replicates for 156 genotypes for each VNIR (400-1000 nm) and SWIR (1000-2500 nm) hyperspectral images. VNIR has 160 and SWIR 256 channels respectively.

In this section of this document we will cover the segmentation approaches and the extracted dataset from each image type and how we are going to merge them. And next under implementation section, we will try to build AI models first on each individual image type, VNIR and SWIR, and then will extend the models by feeding the overall dataset combined from both VNIR and SWIR.

3.1 SEGMENTATION OF GRAINS

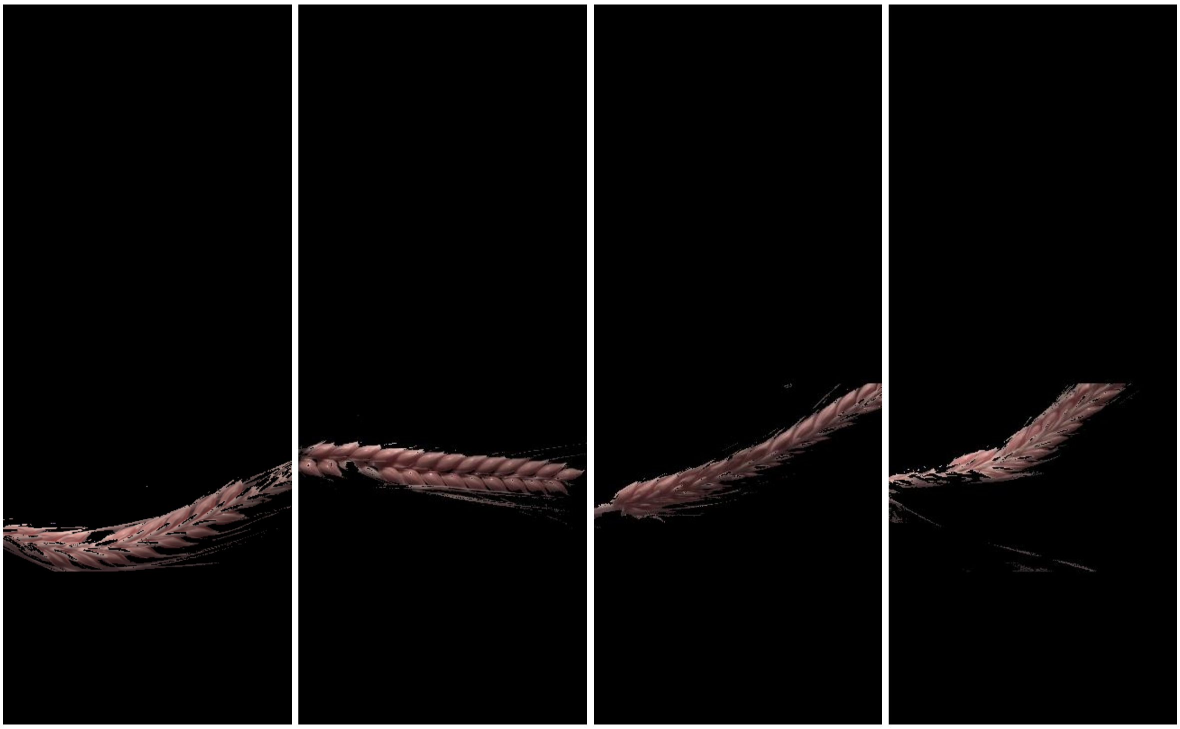
3.1.1 SWIR images

Three channels are used to make an RGB image and as it seems the grains looks much reddish and is enough different than the leaves and the background. In order to save time and not do unnecessary computation, only the area where crops located, are considered for segmentation. Channels used are 55, 41, 12



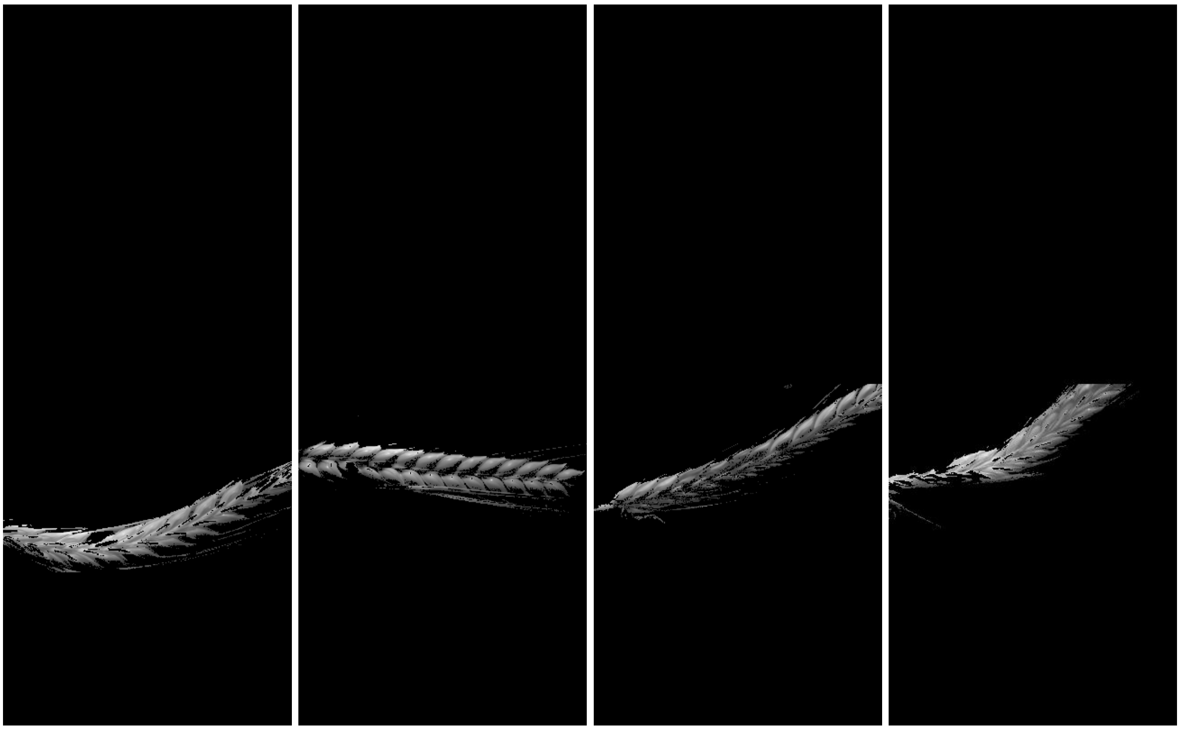
We used OpenCV/python for segmentation purpose

- **Changing Color-space**
    - **Color Conversion**
      We use the function **cv2.cvtColor(input_image, flag)** where flag determines the type of conversion (e. g. cv2.COLOR_BGR2HSV).
- **Applying mask**
    - We use the function **cv2.inRange(HSV_converted_image, lower_bounds, upper_bounds)**
    - HSV color space
        - Hue range is [0,179],
        - Saturation range is [0,255]
        - Value range is [0,255].
        - lower_bounds (H_{lower}, S_{lower}, V_{lower})
        - upper_bounds (H_{upper}, S_{upper}, V_{upper})
    - lower_bounds = (0, 60, 50)
    - upper_bounds = (100, 255, 255)

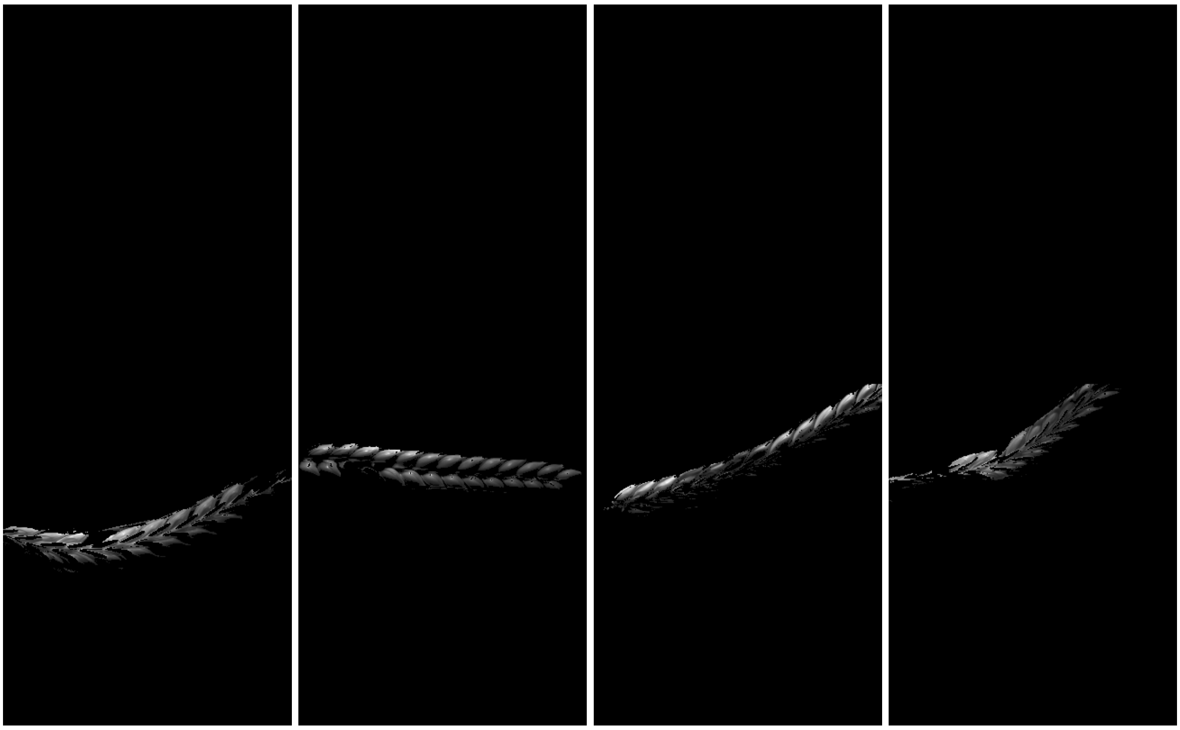


There is no need for RGB image anymore and instead only single channel is sufficient for later use. As u see the images there are some noises and unwanted remaining from leaves and need to be cleared and only grain is what we need.

In the following only single channel of this RGB is represented which is kept for further process.

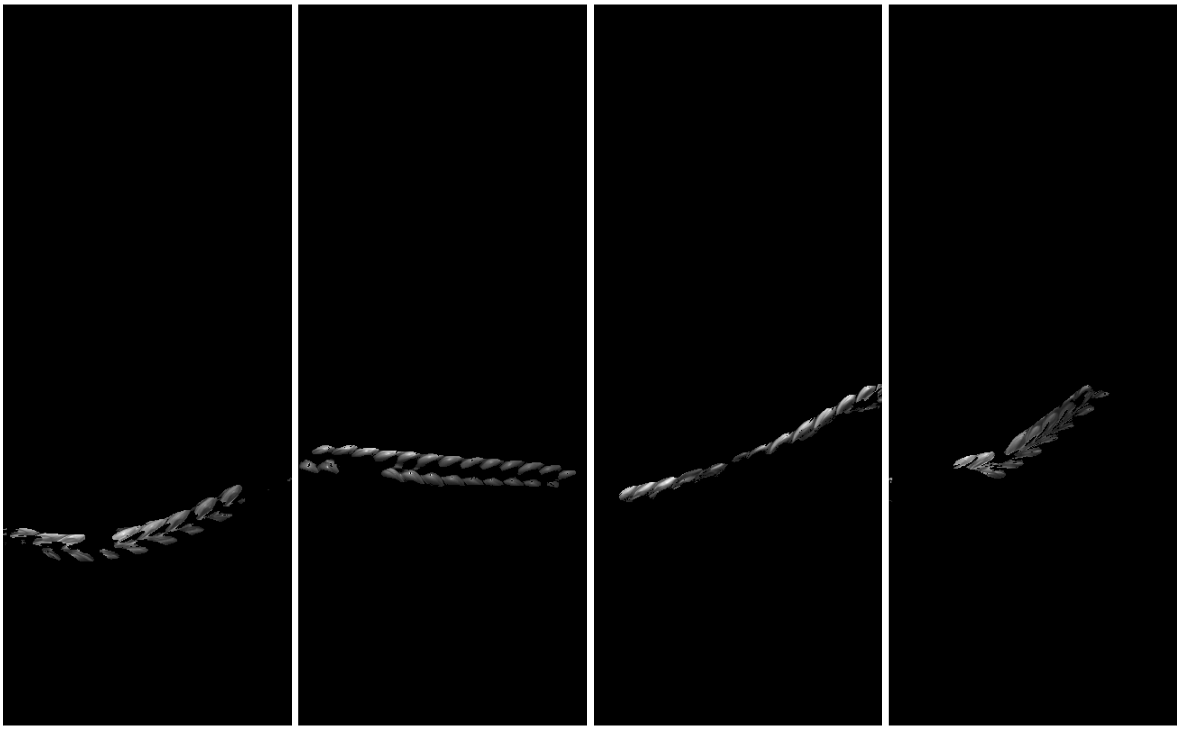


We wanted to make every grain well separated, so by squaring the values of each pixel and again all the values pushed to the range of 0 and 255 for illustration purpose. We could easier locate a global threshold to remove some pixels from between the grains and the results is as follows:

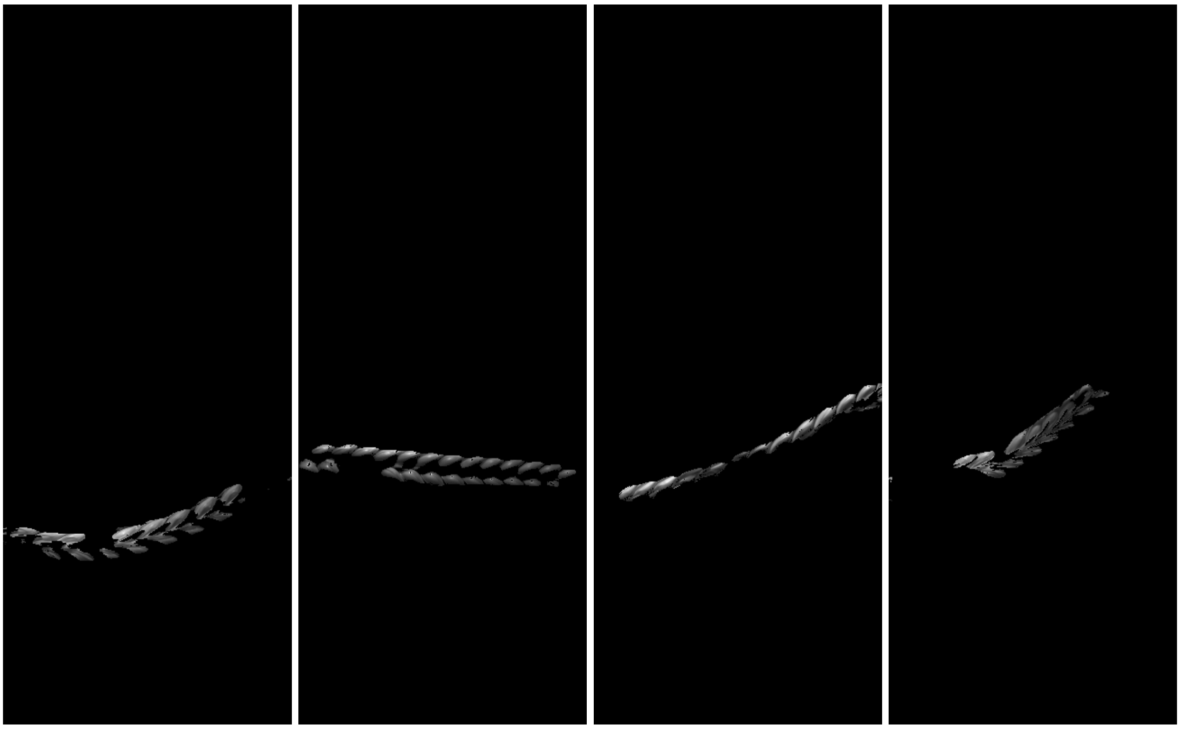


Using structuring elements on the gray scale images we tried to remove noises. such that for every pixel with in its radius (distance based on number pixels based on Moore neighborhood) and minimum number of neighbors is considered such that if number of points was less than minimum points, the point was simply switched to zero.

Radius = 5 and Minimum points = 50 results followings

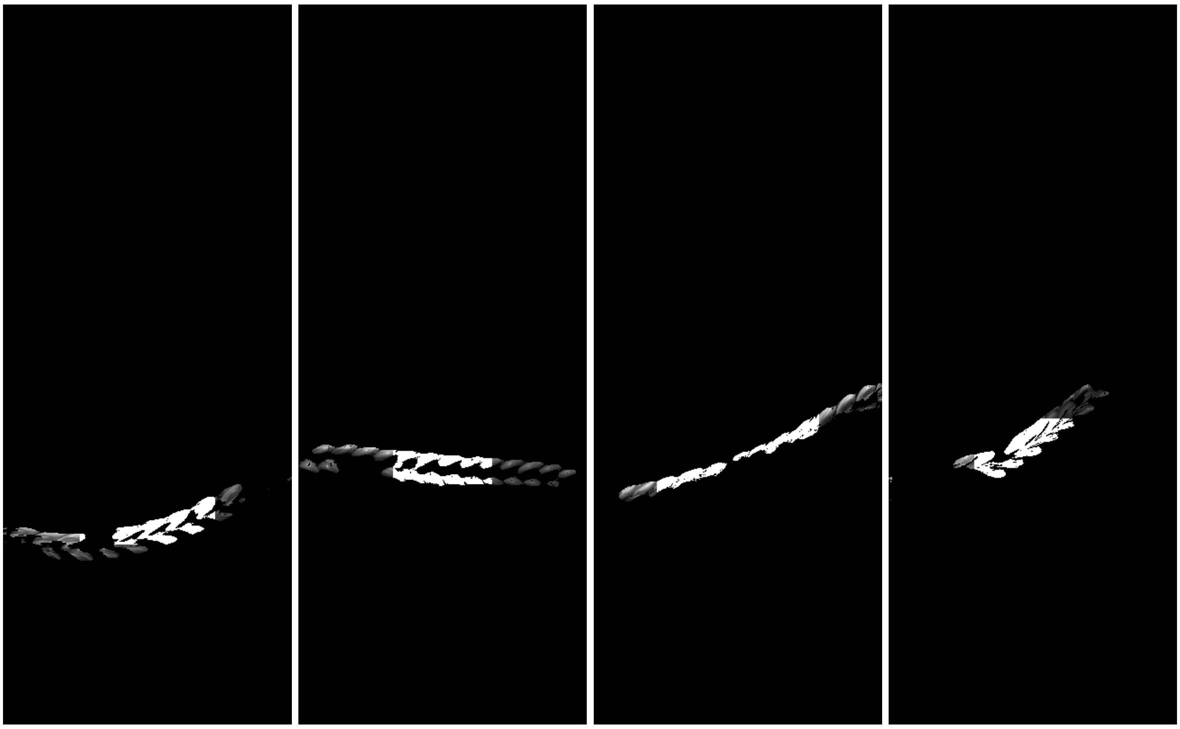


And as a result of two iterations we got much more clean grains as follows:



Each grain consist of around 20 X 20 pixels (SWIR) and since we need to get 5 to 6 grains from middle of each spike, totally we need around 2000 pixels from each image. In order to get middle grains, we started from middle point of area where spikes are located and expanded its area until we cover 2000 pixels.

So as a result we would have below highlighted pixels in the following images for further steps



So for every image (SWIR) we would have 2000 pixels to act as 2000 variables for 256 values. The mean value of middle grains of spike (highlighted) are taken for each 256 channel of images and the following is the extracted SWIR dataset.

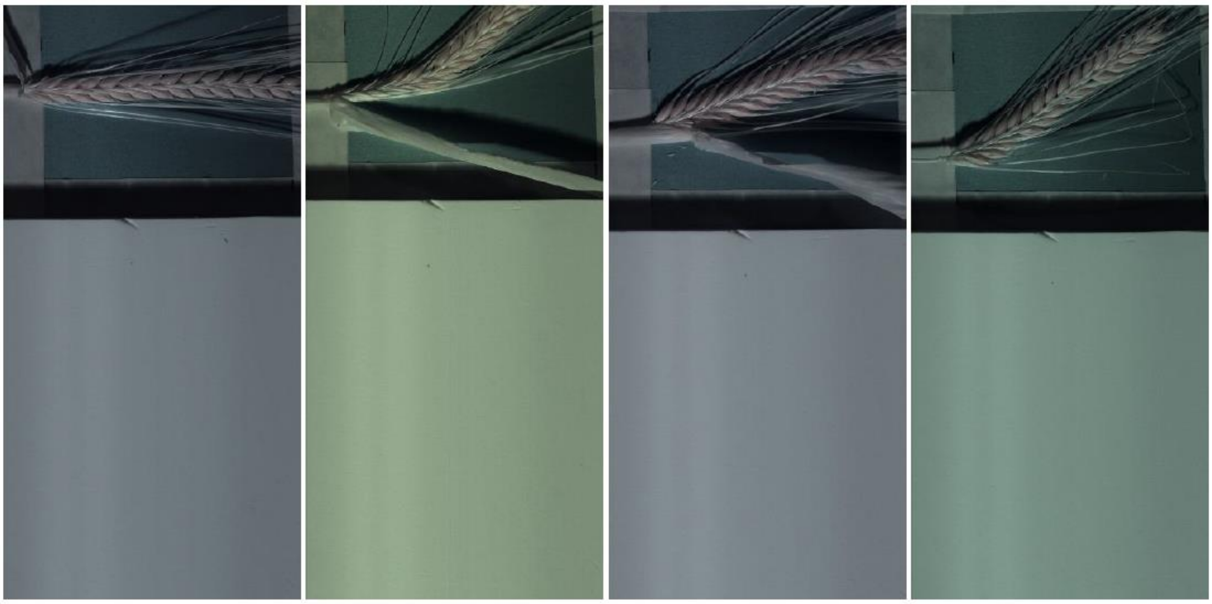
SWIR Dataset:

	Img_name	Band_001	Band_002	Band_003	Band_004	...	Band_253	Band_254	Band_255	Band_256
0	38367	33859.3	34107.3	34253.9	34424.6	...	924.64	909.74	884.22	877.07
1	38368	30172.2	30337.9	30461.6	30564.1	...	647.7	617.49	616	597.36
2	38369	28906.6	29083.7	29221.1	29342.1	...	799.39	789.29	771.96	762.62
3	38370	32421.5	32647.6	32797.9	32972.5	...	908.89	873.83	861.53	830.3
4	38371	29320.4	29456.8	29543.3	29632.5	...	778.7	754.6	740.38	715.73
...	...	...	...	...	...	...	...	...	...	...
474	38866	28353.4	28483.5	28593.4	28749.2	...	913.69	878.76	862.77	842.19
475	38867	30904.4	31081.1	31203.5	31335.4	...	854.56	831.28	800.25	785.89
476	38868	33446.1	33654.6	33811.2	33969.9	...	991.73	974.95	943.59	937.33
477	38869	36046.5	36220.8	36315.4	36413.5	...	874.15	834	817.39	790.82
478	38870	30630.9	30804.7	30948.4	31129.3	...	910.45	877.73	866.21	836.27

3.1.2 VNIR images

For VNIR we used same procedure as SWIR but the only difference is that we used overall six channels for segmentation purpose such that we made two RGB images and segmented each and at the end we got the union of both segmented images. This is because VNIR had relatively color variation in images and within single RGB images we could not segment all images properly. Below channels with lower bounds and upper bounds are used for segmentation of VNIR images.

```
Channels 23, 31, 39
lower_bounds = (0, 0, 70)
upper_bounds = (12, 255, 255)
Channels 159, 140, 120
lower_bounds = (0, 0, 0)
upper_bounds = (60, 255, 255)
```



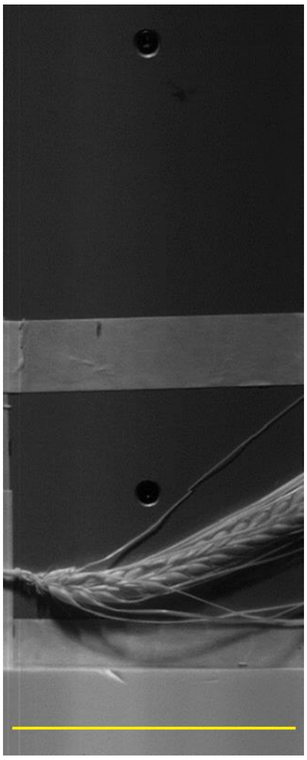
As a result of segmentation of VNIR images following dataset was extracted.

VNIR Dataset:

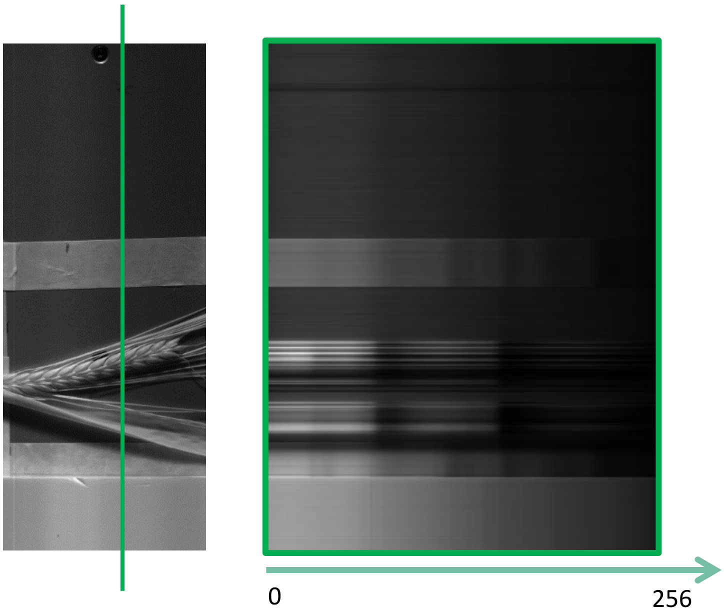
	Img_name	Band_001	Band_002	Band_003	Band_004	...	Band_157	Band_158	Band_159	Band_160
0	38367	364.42	351.9	355.18	351.7	...	19783.2	19889.9	20108.5	19968
1	38368	401.69	399.24	388.89	390.64	...	20535.2	20616.9	20816.7	20759.7
2	38369	439.95	447.82	447.28	458.13	...	22512	22706	22975.2	22834.8
3	38370	443.4	432.23	423.2	421.79	...	26510.8	26766.2	26926.6	27018.1
4	38371	468.22	453.31	446.54	430.26	...	25878.3	26040.6	26232.1	26332.7
...	...	...	...	...	...	...	...	...	...	...
458	38866	384.54	376.32	362.53	365.26	...	16894.3	17045.6	17134.3	17174.2
459	38867	570.88	566.19	574.22	578.39	...	23310	23478.7	23663.1	23609.7
460	38868	540.91	527.65	532.66	548.16	...	25167.6	25386.1	25489.5	25569.3
461	38869	535.88	523.54	514.52	525.02	...	26962	27125.7	27313.7	27357.5
462	38870	361.55	340.64	336.99	328.86	...	22438.3	22585.5	22782.1	22733.5

3.2 DAYLIGHT INTENSITY

We considered the average values of part of the reference board (highlighted in below image; rows 780:785, and columns 10:310) as daylight intensity for that particular image.



As far as we took this value as daylight intensity level, we noticed that this value is not stable along all channels of that particular image.



In addition you can take a look on below image.

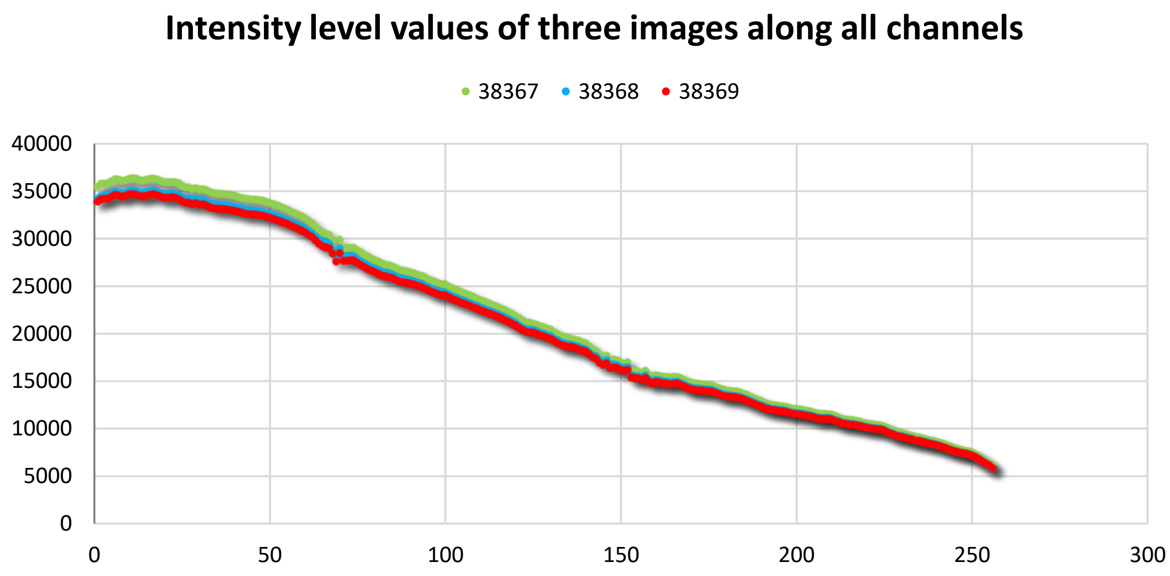
The spectrum values of grains drastically decrease while the reference board is somewhat stable.

3.2.1 Preparing dataset by adding difference of intensity level on each images

3.2.1.1 SWIR daylight intensity

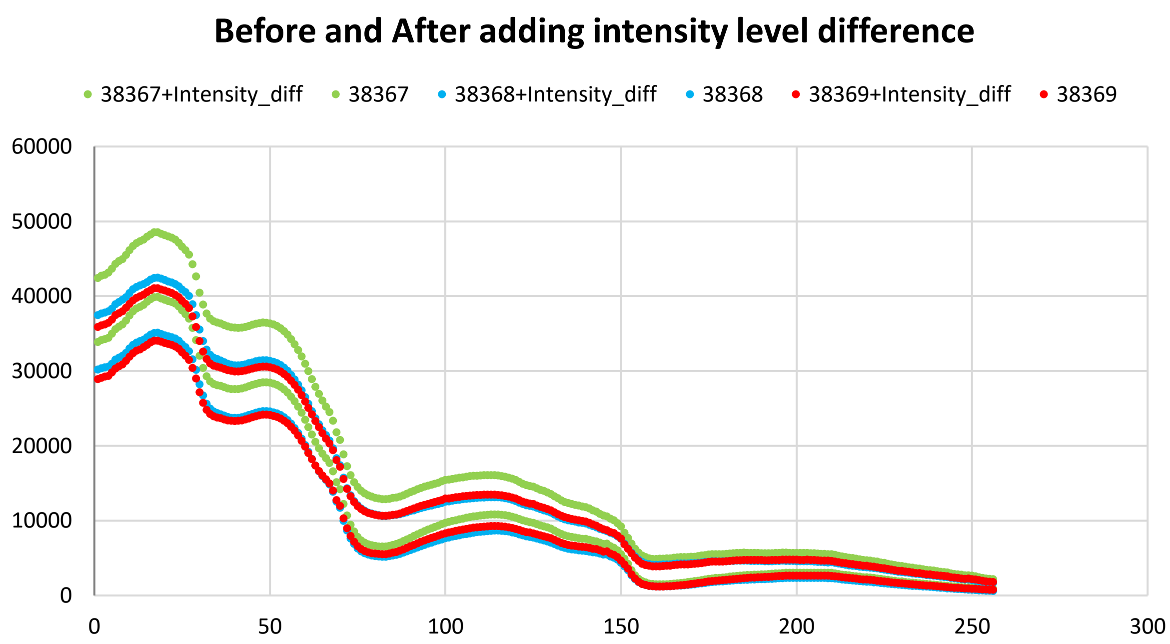
We have taken average value of 780:785 rows, and 10:310 columns of each channel as the reference for the daylight intensity level.

In below charts, intensity level of images along all the channels are represented.



After preparing this data, we took the minimum intensity value of each channel (referring image number) as reference and added the difference of the intensity level on top of respective channel of each image.

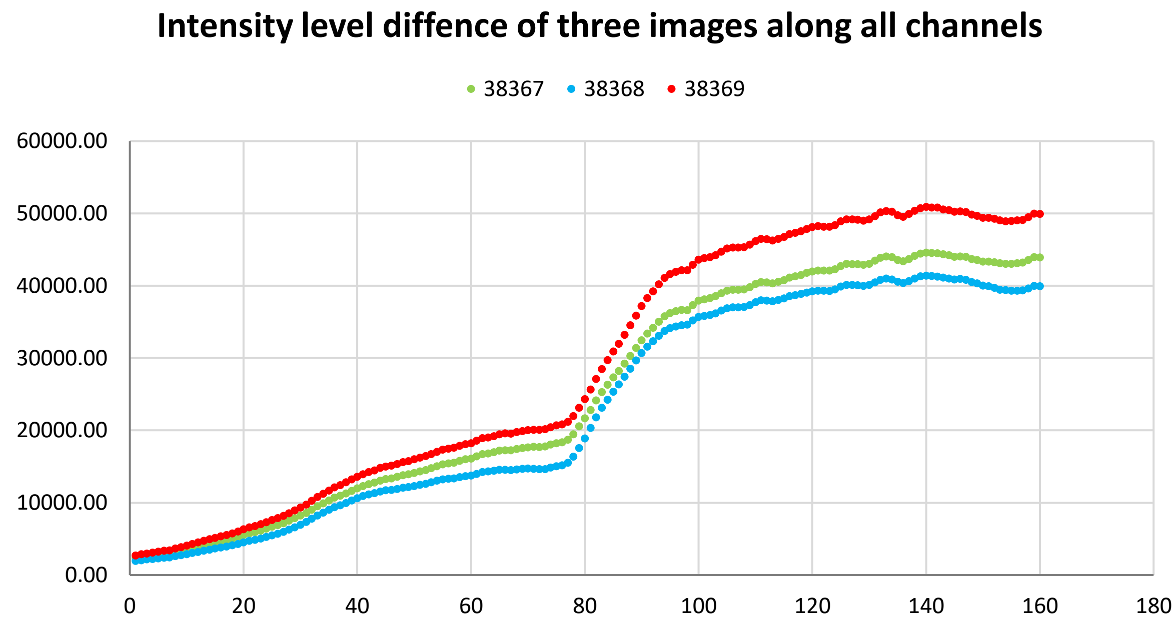
Below chart is the result of adding intensity level difference on three images.



3.2.1.2 VNIR daylight intensity

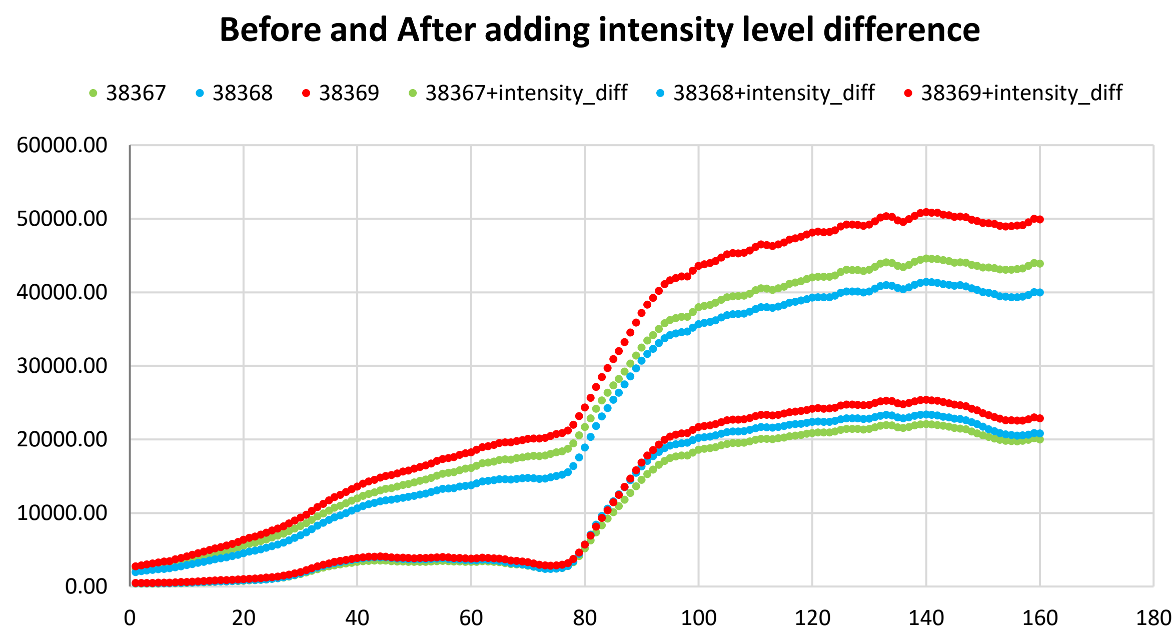
Average value of 280:285, 10:310 area of resized VNIR images is used as reference for intensity level.

In below charts, intensity level of images along all the channels are represented.



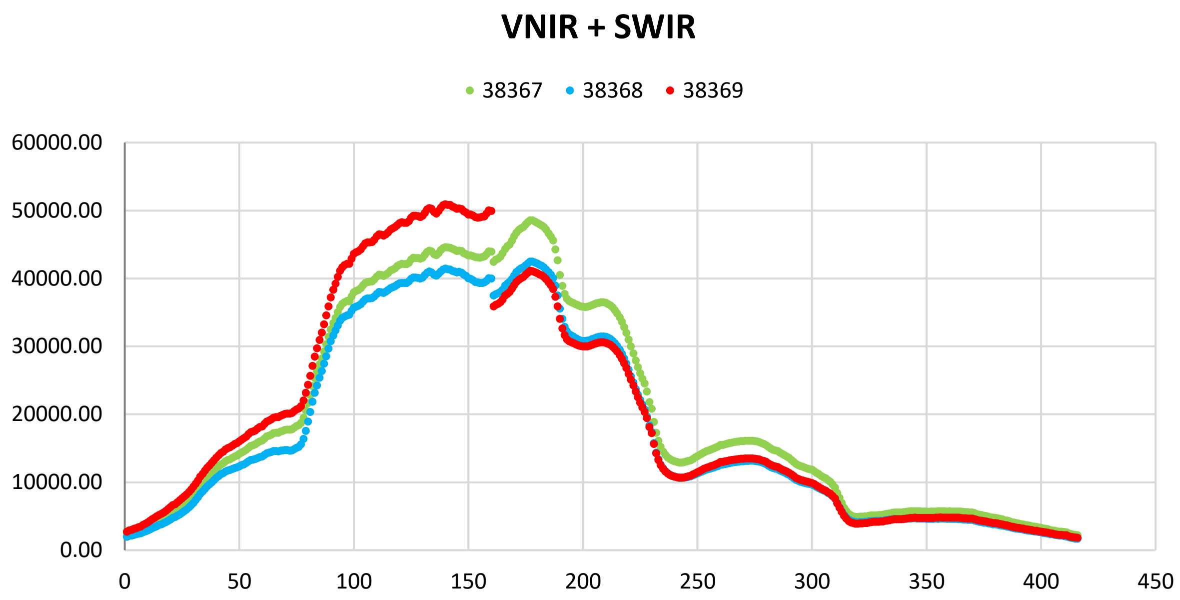
After preparing this data, we took the minimum intensity value of each channel (referring image number) as reference and added the difference of the intensity level on top of respective channel of each image.

Below chart is the result of adding intensity level difference on three images.

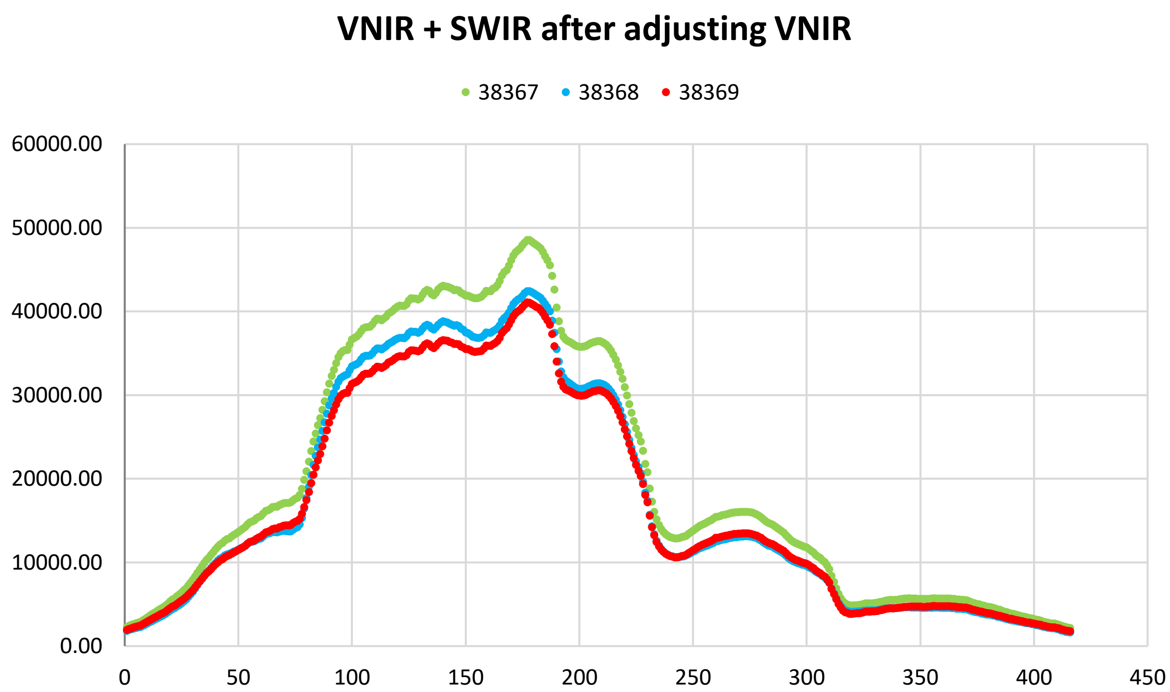


3.2.1.3 *VNIR+SWIR*

Combining VNIR 0 to 160 channels with SWIR 161 to 416 channels as follows.



Obviously there is a sharp difference between channels 160 and 161. We are keeping SWIR intact and proceed with adjusting VNIR so that both 160 and 161 values match each other and have a smooth curve.

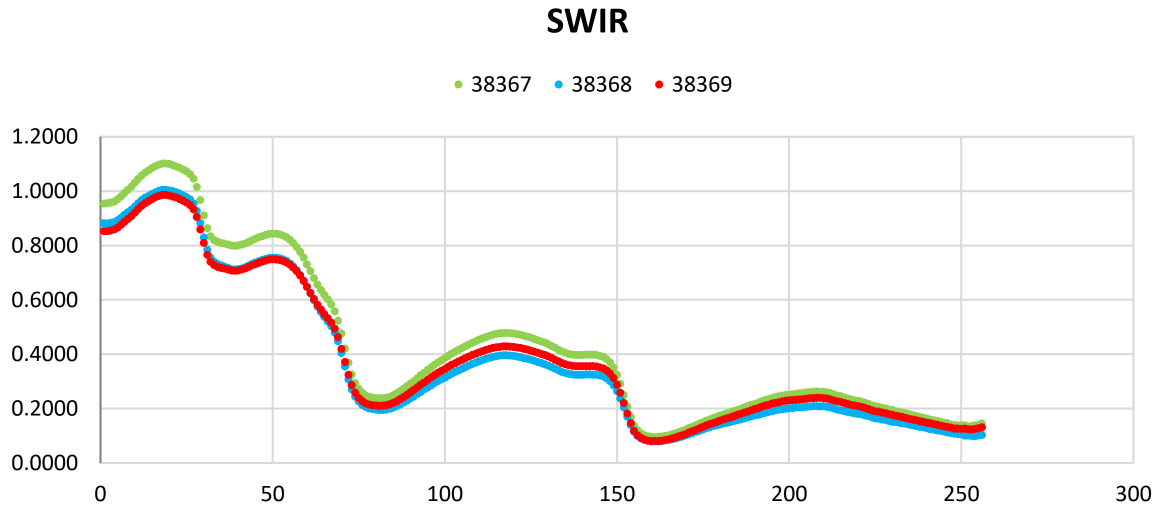


3.2.2 Preparing dataset by dividing spectrum values on intensity level

Previously we have added differences of daylight intensity elementwise on each channel. This time we are going to prepare the dataset by dividing the spectrum values elementwise on light intensity of each respective channel of each particular image and see if this way can get a better predictive model.

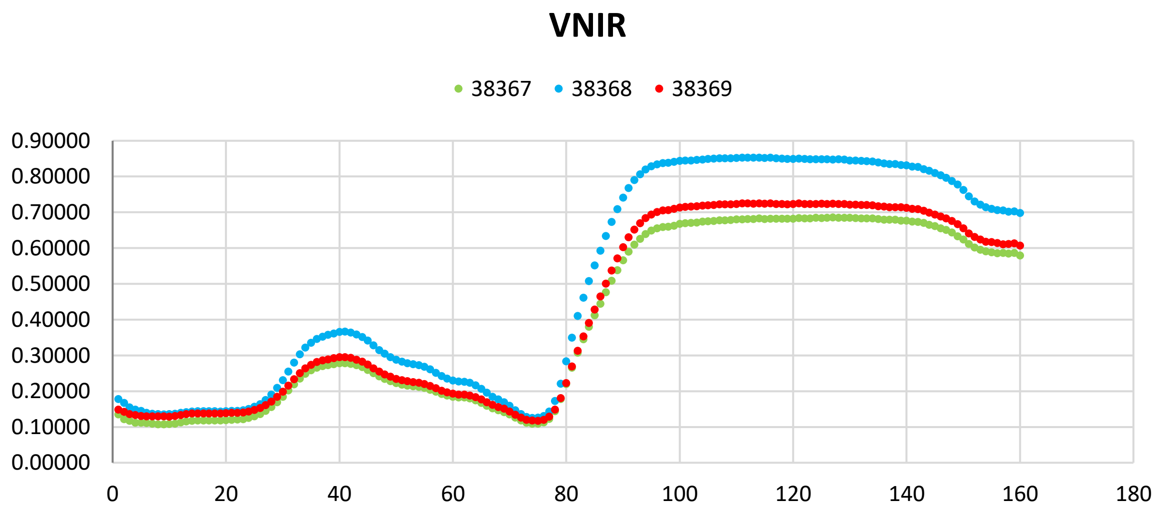
3.2.2.1 SWIR daylight intensity

Below chart is the outcome of dividing the spectrum values elementwise on light intensity of each respective channel of each particular SWIR image.



3.2.2.2 VNIR daylight intensity

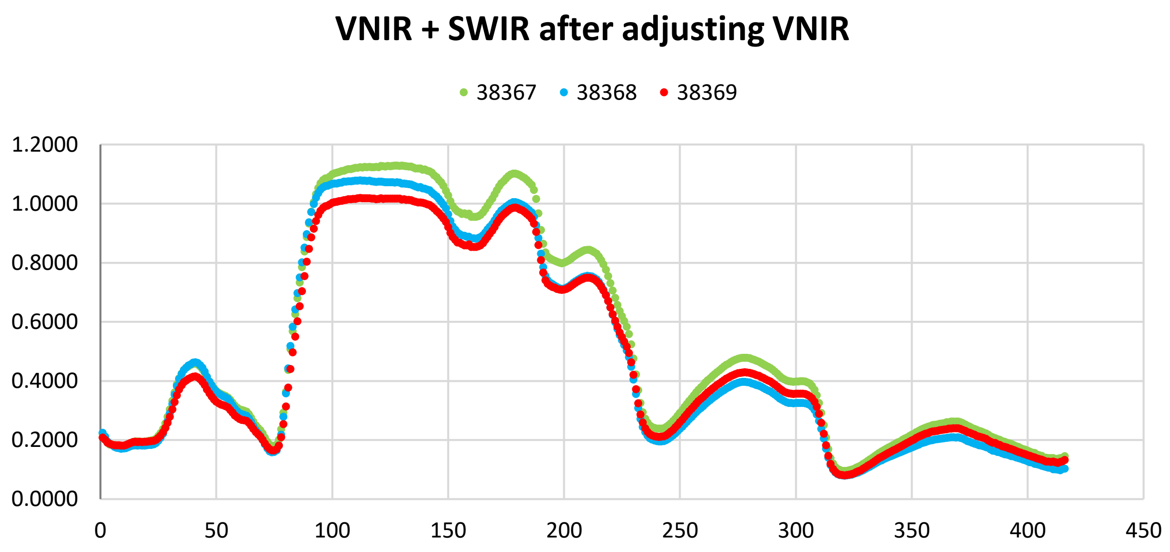
Below chart is the outcome of dividing the spectrum values elementwise on light intensity of each respective channel of each particular VNIR image.



3.2.2.3 VNIR + SWIR

Same as before we are combining VNIR 0 to 160 channels with SWIR 161 to 416 channels as follows.

We are keeping SWIR intact and proceed with adjusting VNIR so that both 160 and 161 values match each other and have a smooth curve.

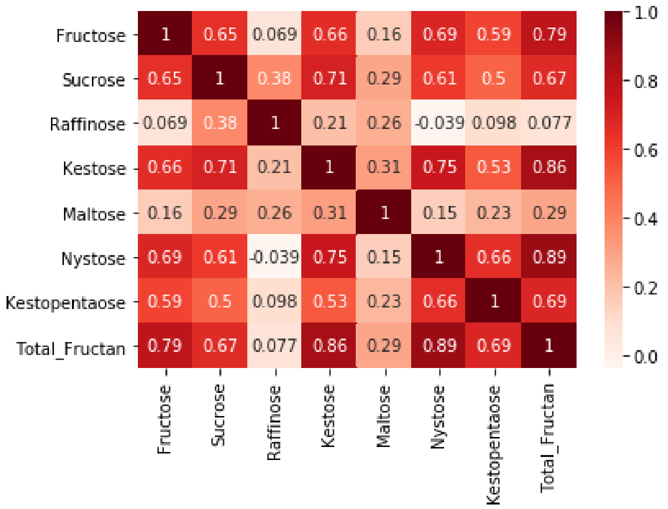


3.3 SUGAR CONTENTS (TARGET VALUES)

There are generally 3 replicates for 156 genotypes and totally nine target values taken from lab and we are going to build supervised algorithms to predict sugar values. Below is how the it looks like.

	Img_name	Print_Info	Variety_ID	Variety	Batch	Sample	Glucose	Fructose	Sucrose	Raffinose	Kestose	Maltose	Nystose	Kestopentaose	Total_Fructan
0	38367	Tucson-R1_L1P2	147	Tucson	17	391	0.759051	0.938528	11.580359	7.191522	2.508236	1.132254	0.454942	0.493446	7.451642
1	38368	Cabaret-R1_L1P3	26	Cabaret	1	1	0.525990	0.389049	14.139681	6.957827	2.297757	0.607435	0.838198	0.068990	7.209828
2	38369	Hassan-R1_L1P4	73	Hassan	1	2	0.485947	0.364791	11.247442	5.236240	2.417849	0.497984	1.378649	0.221425	8.311020
3	38370	Goldie-R1_L1P5	66	Goldie	1	3	0.878424	0.829733	13.167245	7.756482	3.367840	1.267761	2.235094	1.118777	14.602783
4	38371	Foxtrot-R1_L1P6	61	Foxtrot	1	4	0.263267	0.262206	6.567556	4.641757	1.070230	0.222498	0.261981	0.163116	3.234317
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
454	38866	Turnberry-R3_L21P21	148	Turnberry	15	330	0.406581	0.529273	12.583333	6.917725	2.591700	0.455390	0.960509	0.327040	6.638330
455	38867	Paramount-R3_L21P22	109	Paramount	17	373	0.930095	2.564277	14.531401	4.688469	2.893060	0.623882	1.716241	0.695075	15.680997
456	38868	Chariot-R3_L21P23	37	Chariot	21	474	1.607091	1.793080	14.674025	6.146295	4.023812	0.412128	2.319341	0.406548	21.371271
457	38869	Croydon-R3_L21P24	47	Croydon	16	346	0.760457	0.767068	13.195732	6.929923	3.330329	0.620863	3.080164	0.559146	12.783268
458	38870	Alis-R3_L21P25	7	Alis	21	475	0.693933	0.851303	10.310571	9.238705	1.639396	0.294228	0.486502	0.051405	5.062948

These sugar values for each image are not well correlated to each other and we are supposed to build regression models to predict the values based on the SWIR and VNIR dataset we prepared already.



4 CONCEPT

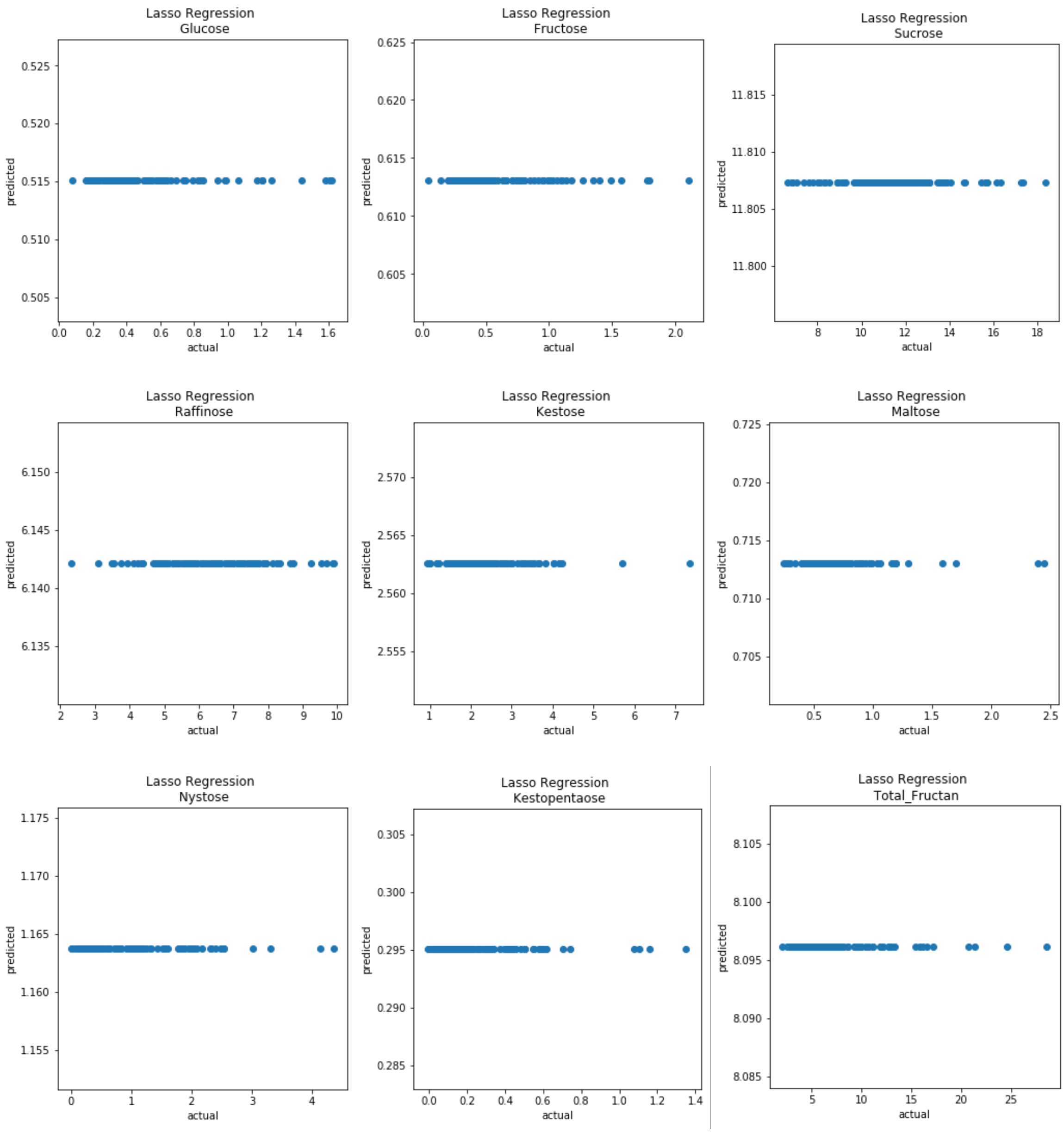
We are going to try different methods until to find the best AI model for predicting sugar contents values. First we will see which normalization technique, referring to light intensity of images can result in better prediction and subsequently we will take that and try additional methods.

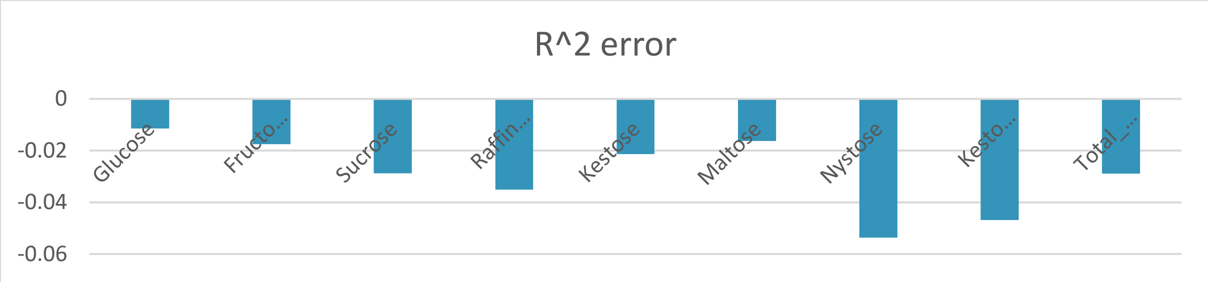
The methods that we will try are include of some regularization techniques e.g. ridge, lasso and elastic-net as well as some additional methods as PLSR, RFR, SVR and deep neural networks.

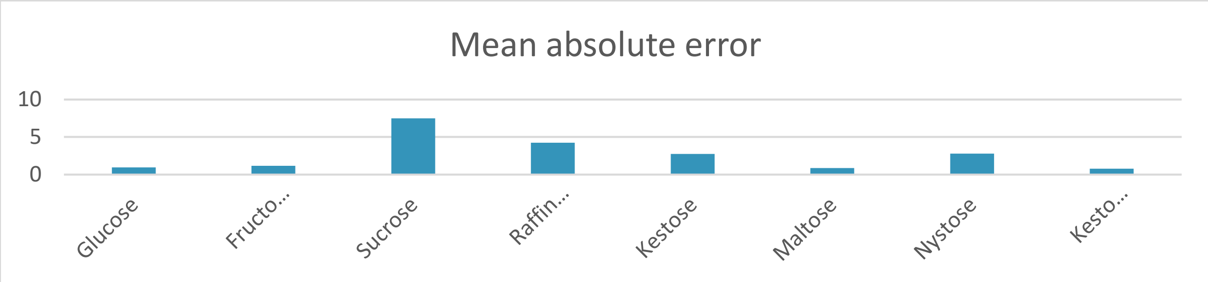
5 IMPLEMENTATION

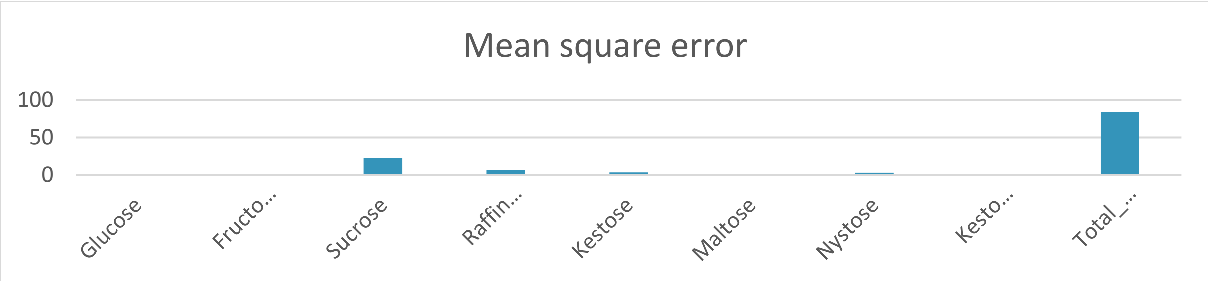
5.1 RANDOMLY SPLITTING DATASET NORMALIZED BY DIVIDING DATA OVER DAYLIGHT INTENSITY LEVEL

5.1.1.1 Lasso





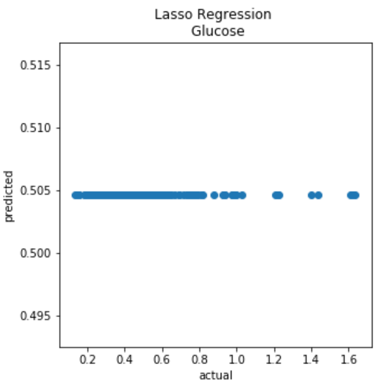


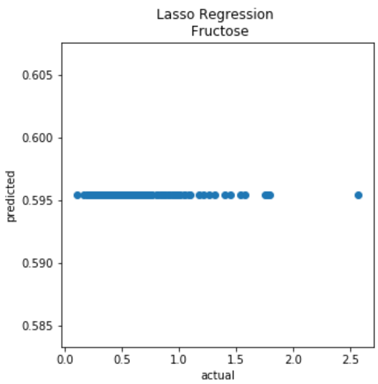


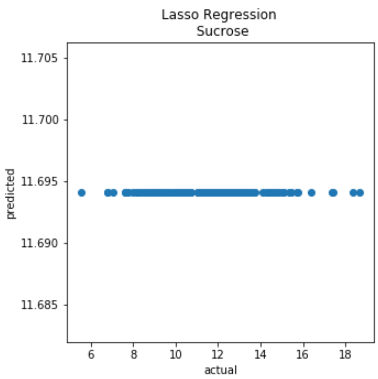
5.2 MIN MAX RANGE_0-1

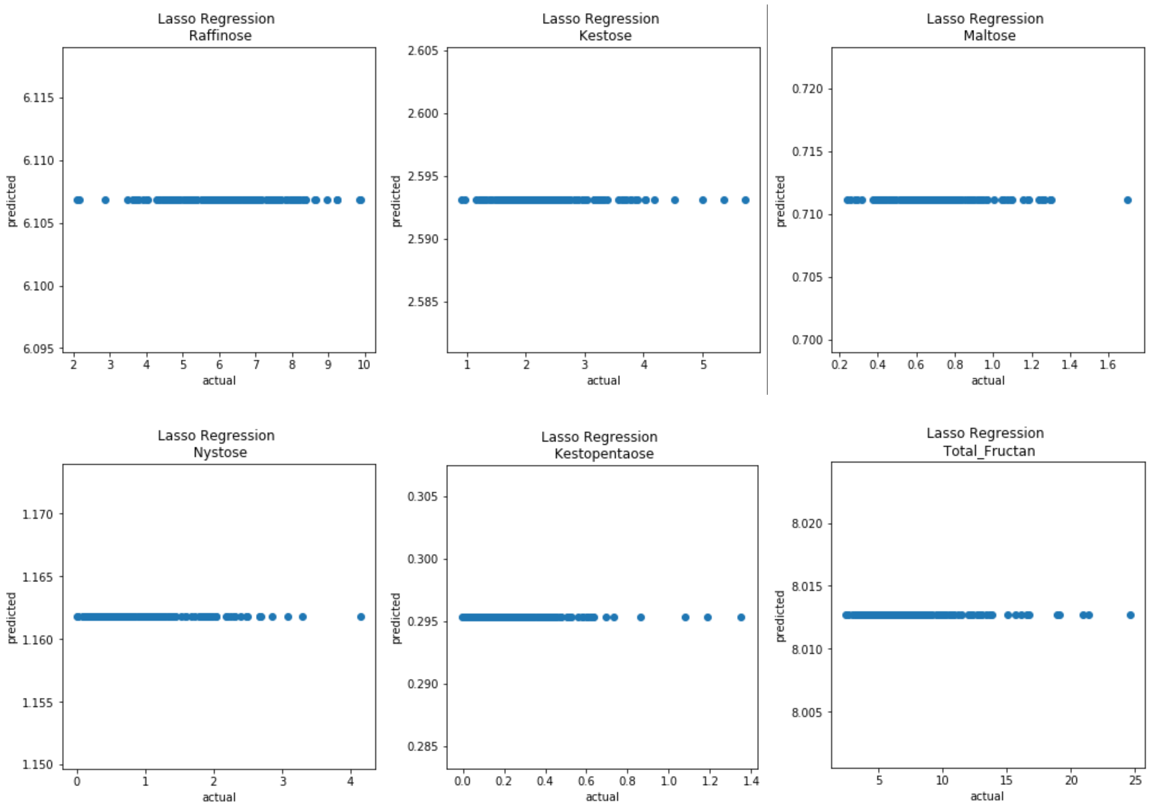
5.2.1 Split the dataset Based on Genotype

5.2.1.1 Lasso

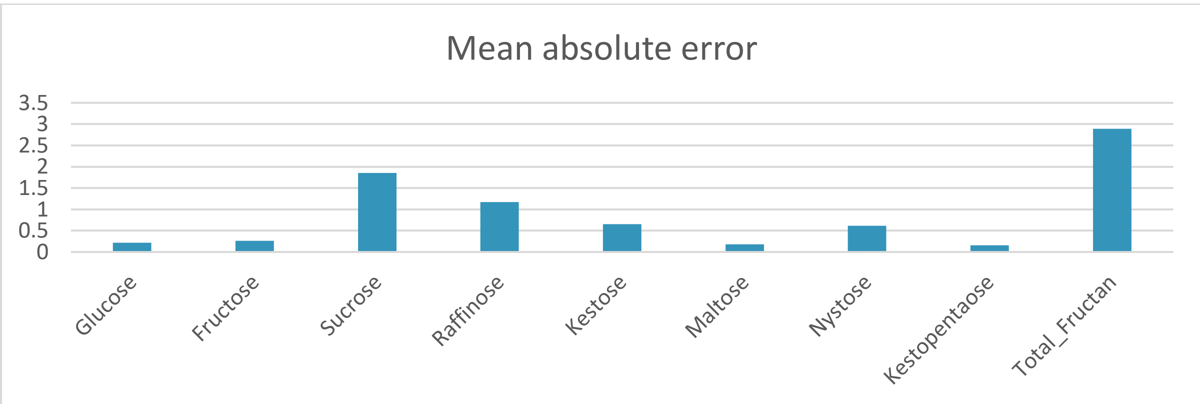


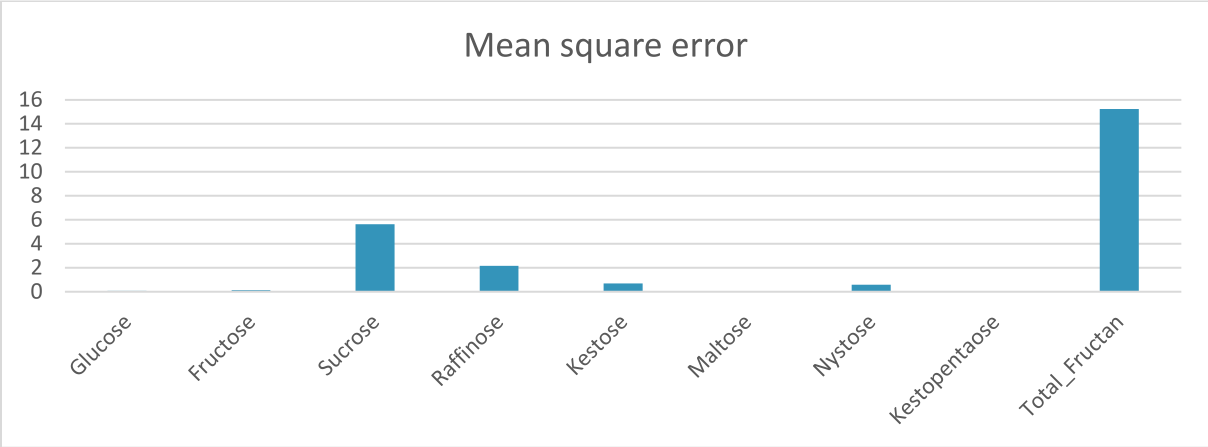





Lasso Regression
Raffinose
Lasso Regression
Kestose
Lasso Regression
Maltose
Lasso Regression
Nystose
Lasso Regression
Kestopentaose
Lasso Regression
Total_Fructan
predicted
actual


R^2 error
0
-0.01
-0.02
-0.03
-0.04
-0.05
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan

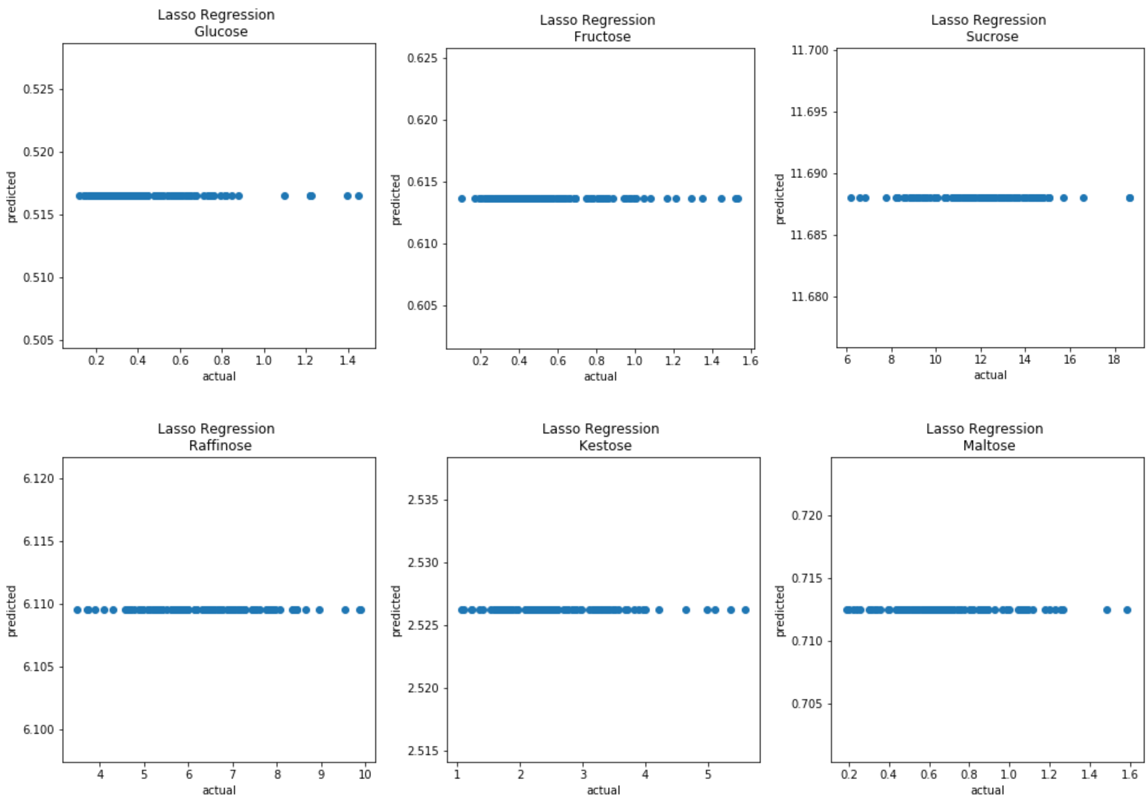

Mean absolute error
3.5
3
2.5
2
1.5
1
0.5
0
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan

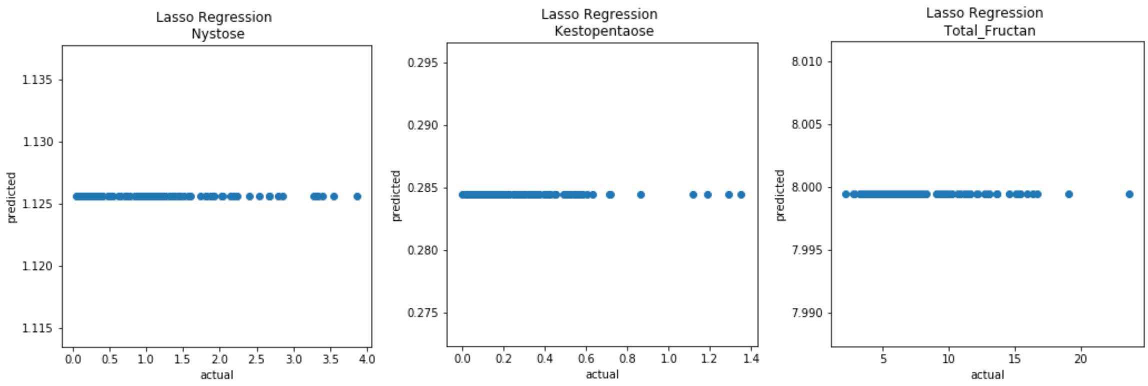


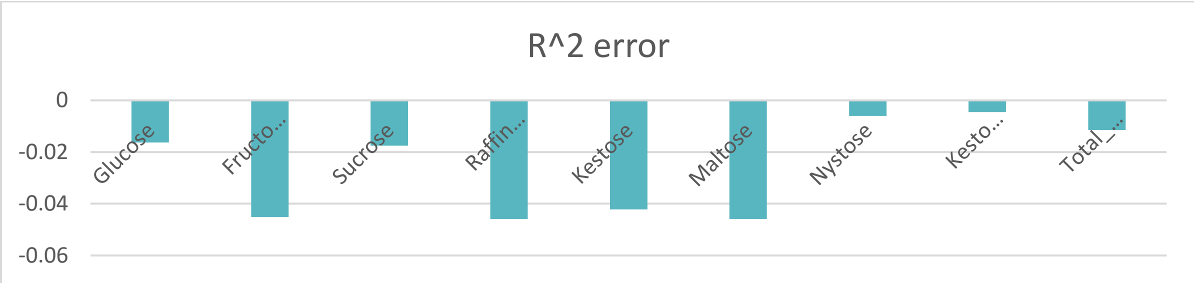
Based on the outcome of Lasso there is no point to try Ridge, Elastic net and other methods further.

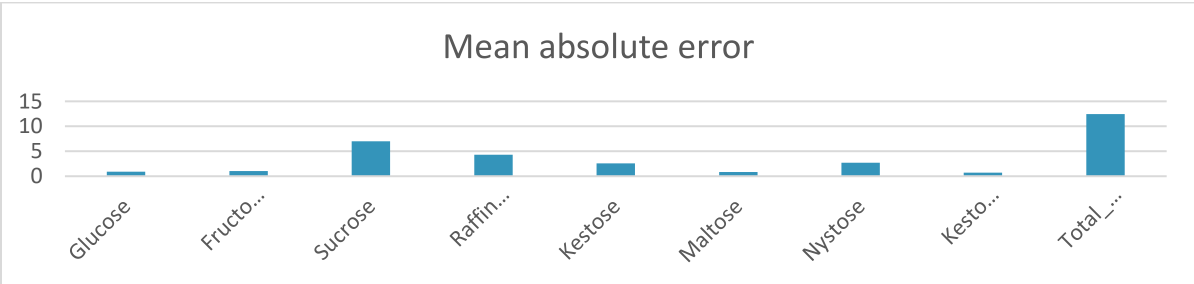
5.2.2 Randomly splitting dataset

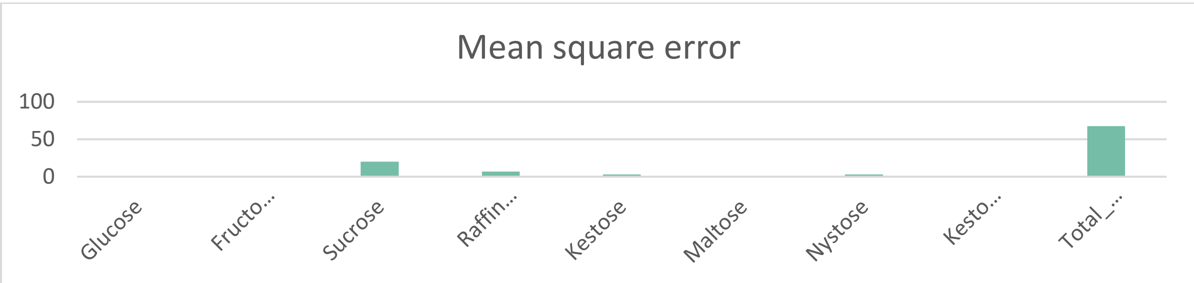
5.2.2.1 Lasso







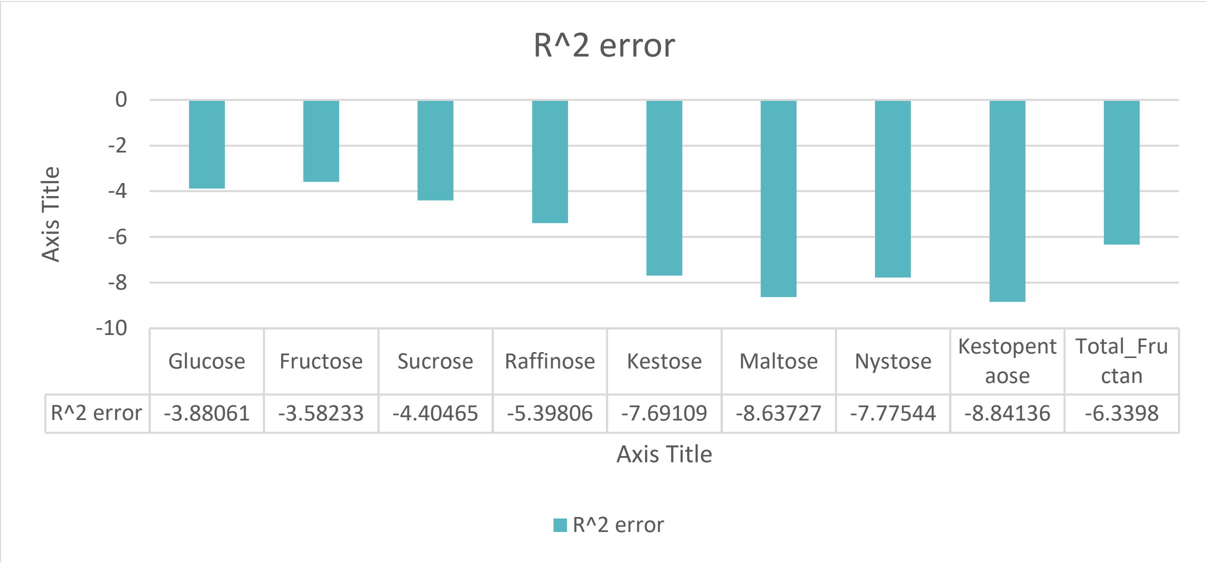


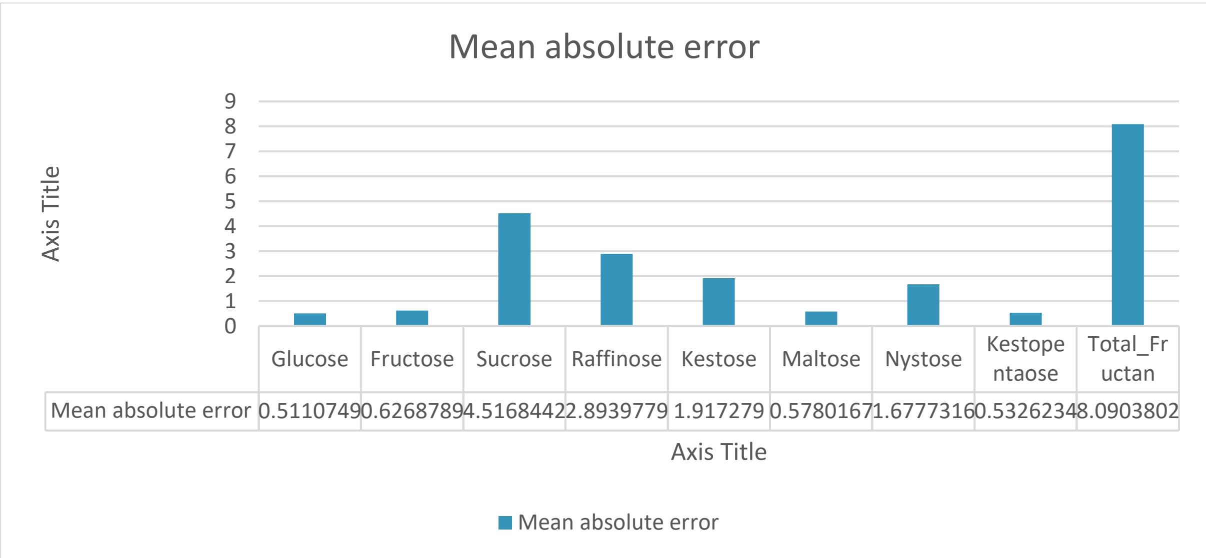


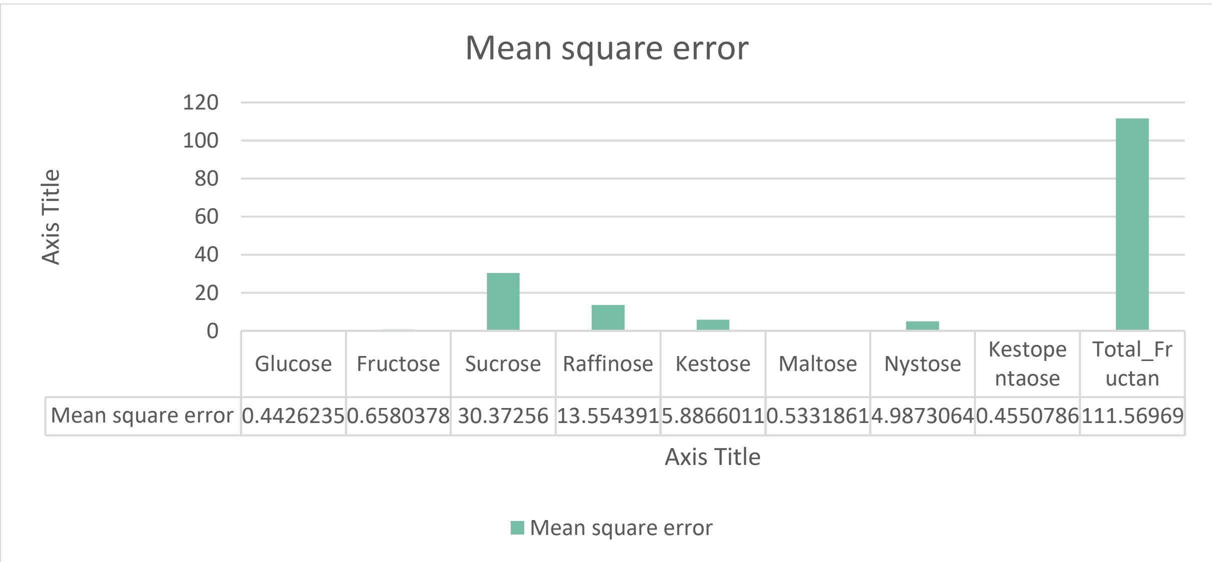
5.3 Without Scaling Features

5.3.1 Split the dataset Based on Genotype

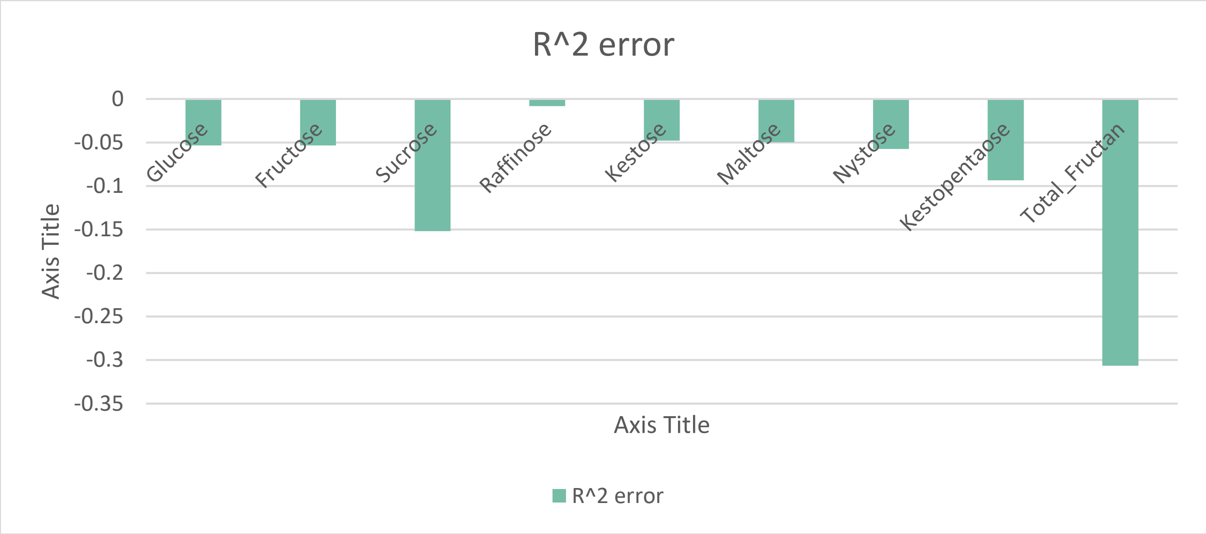
5.3.1.1 Ridge Regression


R^2 error
0
-2
-4
-6
-8
-10
Axis Title
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan
R^2 error
-3.88061
-3.58233
-4.40465
-5.39806
-7.69109
-8.63727
-7.77544
-8.84136
-6.3398
Axis Title
R^2 error

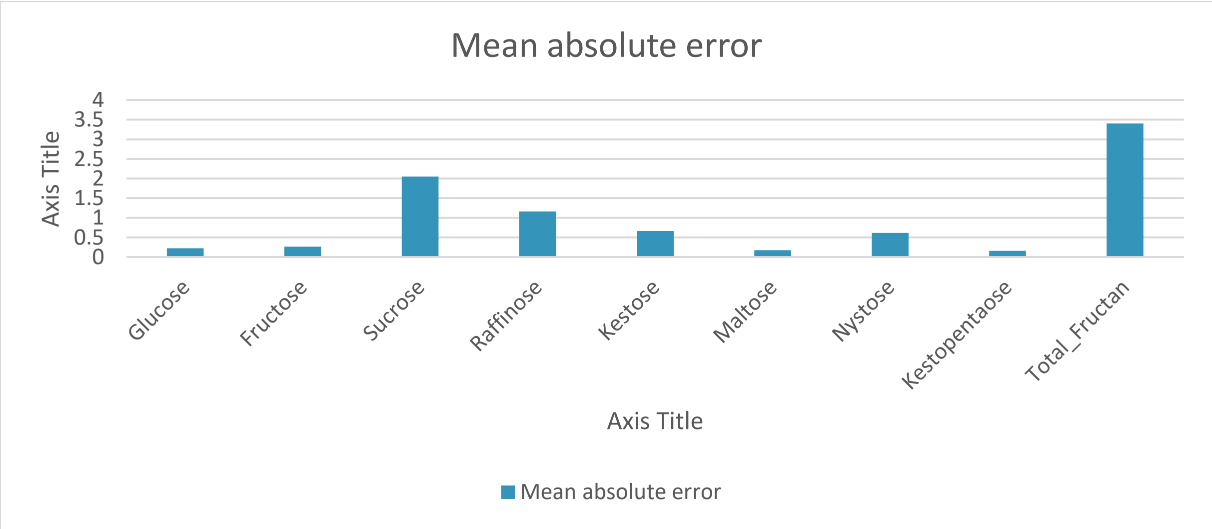

Mean absolute error
9
8
7
6
5
4
3
2
1
0
Axis Title
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan
Mean absolute error
0.5110749
0.6268789
4.5168442
2.8939779
1.917279
0.5780167
1.6777316
0.5326234
8.0903802
Axis Title
Mean absolute error


Mean square error
120
100
80
60
40
20
0
Axis Title
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan
Mean square error
0.4426235
0.6580378
30.37256
13.554391
5.8866011
0.5331861
4.9873064
0.4550786
111.56969
Axis Title
Mean square error

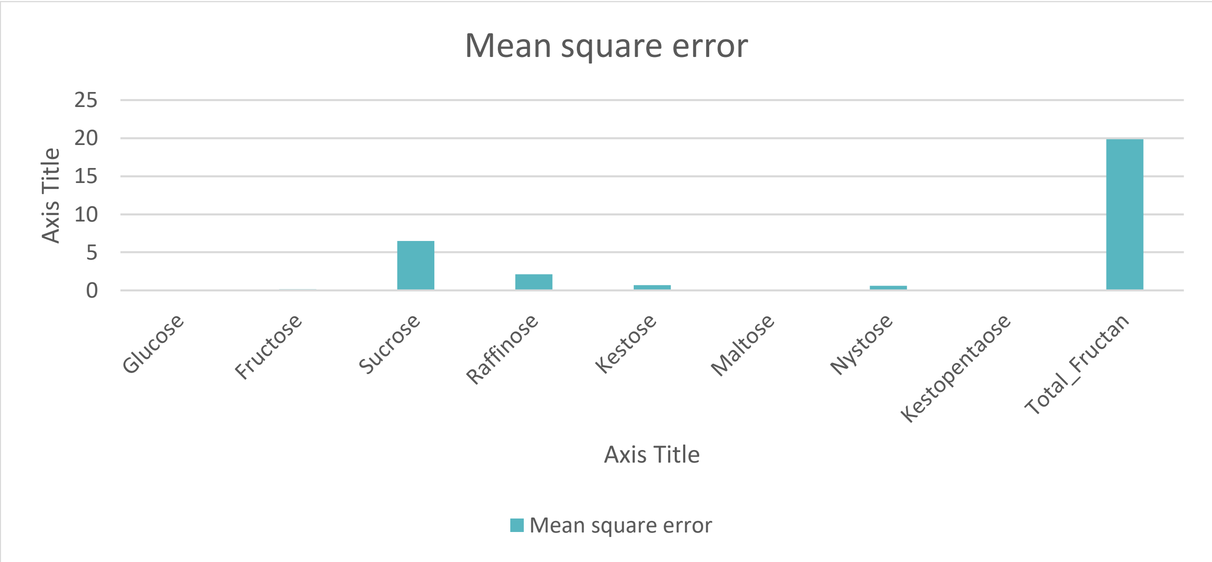
5.3.1.2 Lasso



R^2 error
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
-0.35
Axis Title
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan
Axis Title
R^2 error

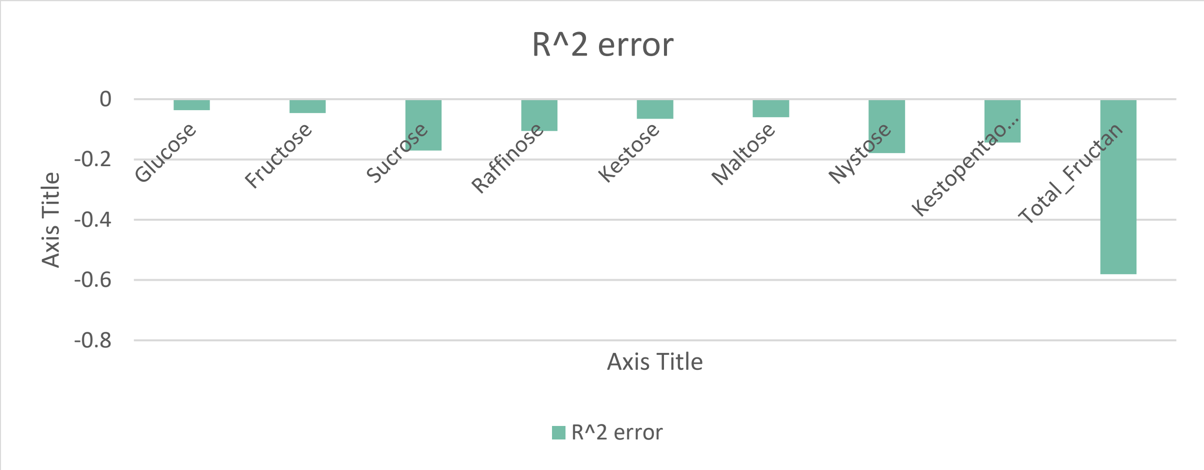


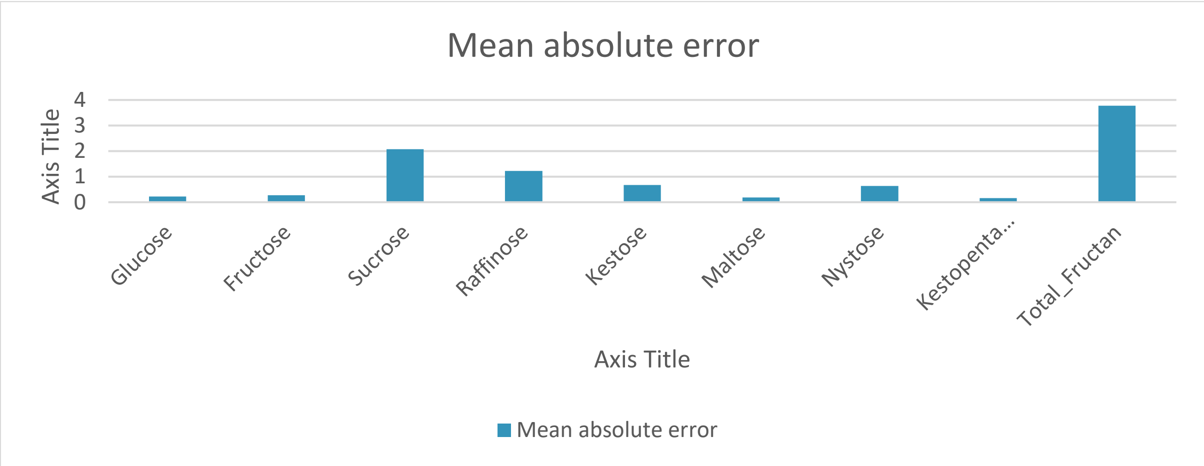
Mean absolute error
4
3.5
3
2.5
2
1.5
1
0.5
0
Axis Title
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan
Axis Title
Mean absolute error

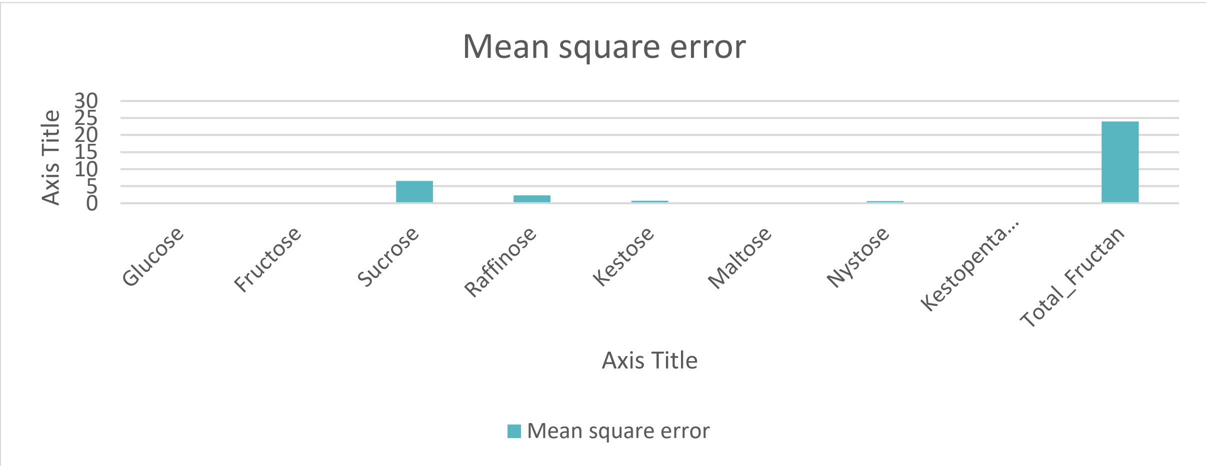


Mean square error
25
20
15
10
5
0
Axis Title
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan
Axis Title
Mean square error

5.3.1.3 *Elastic Net*



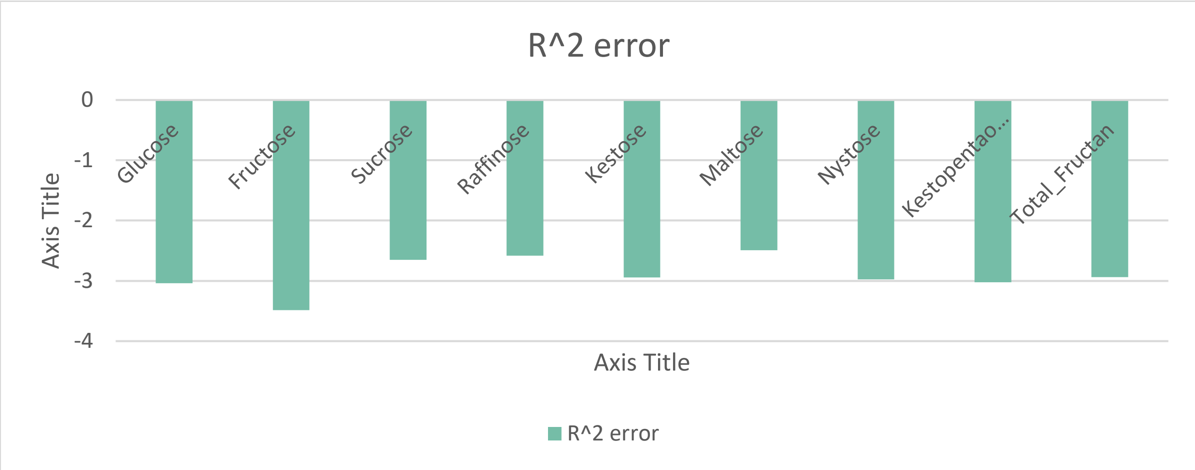


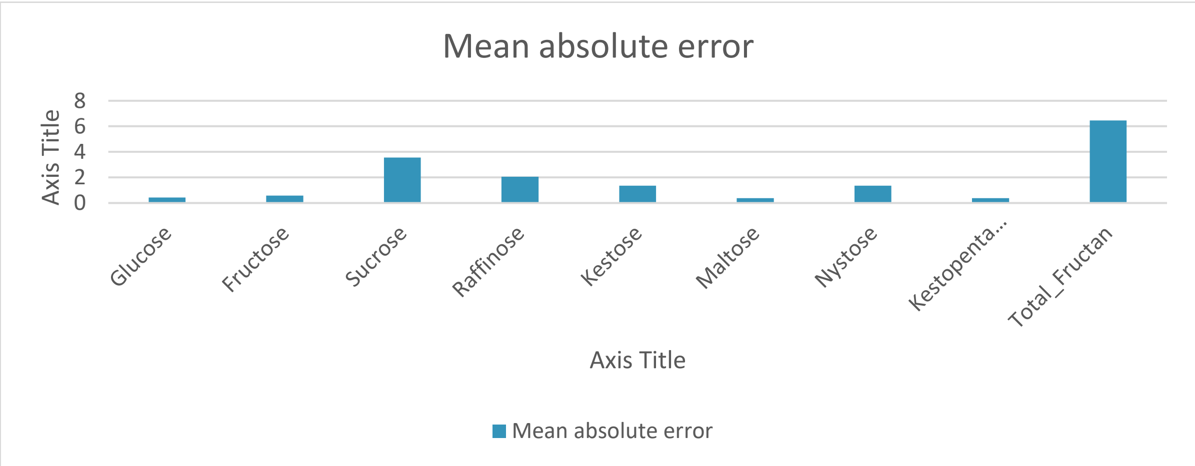


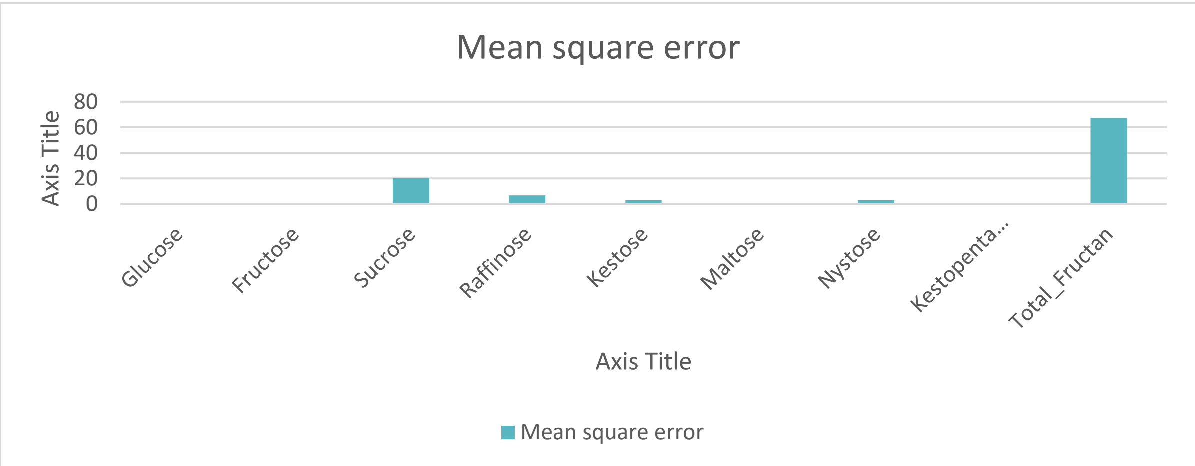
5.3.2 Randomly splitting dataset in to train and test

taking means of 10 replicates

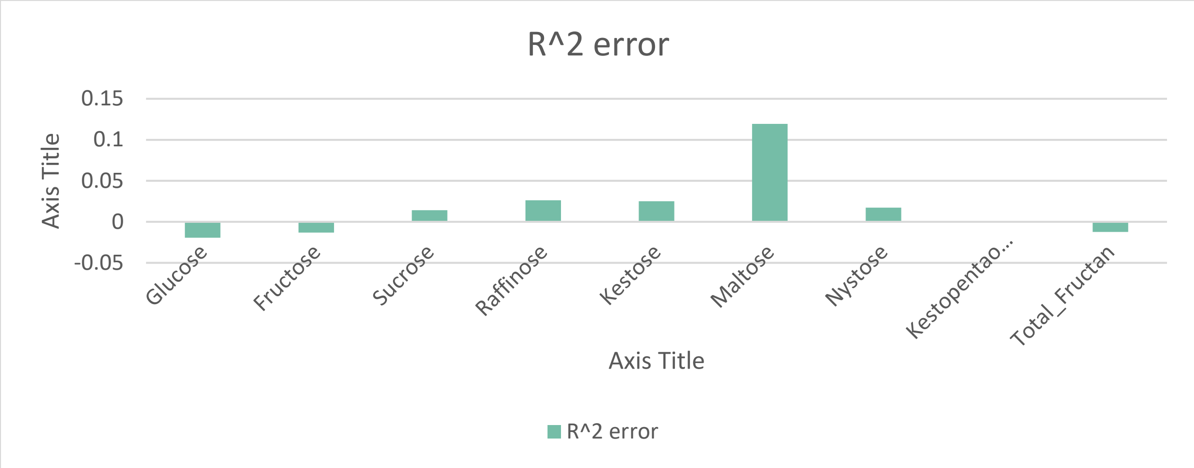
5.3.2.1 *Ridge Regression*

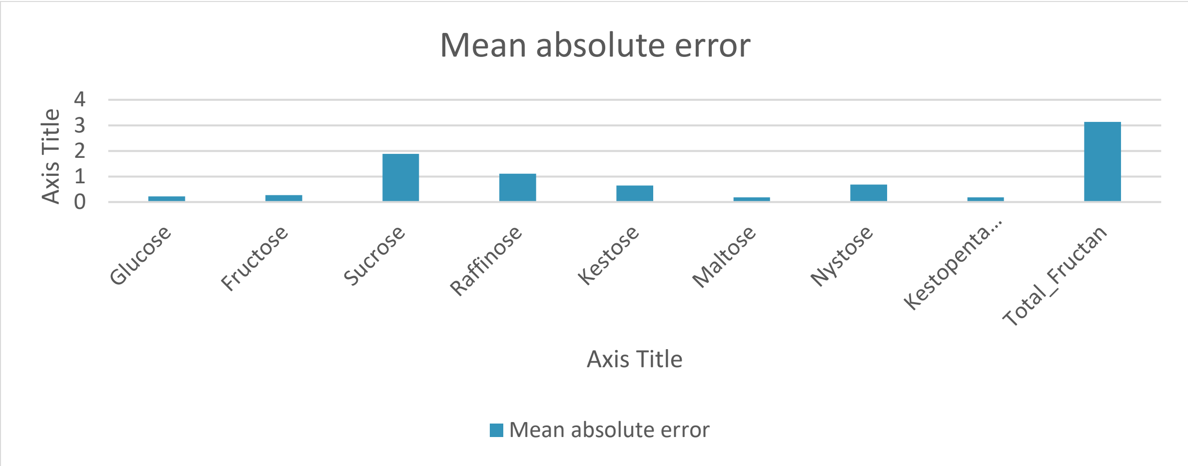


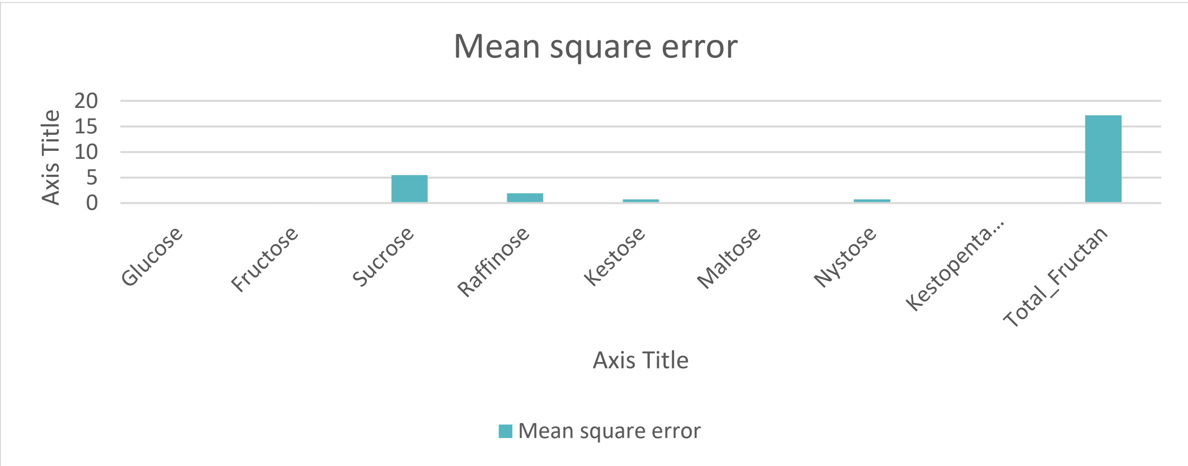




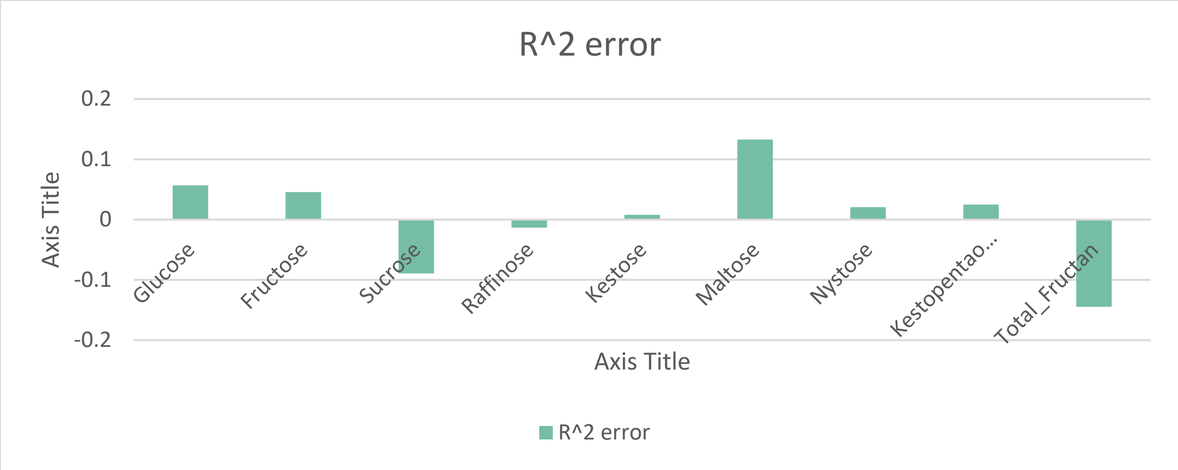
5.3.2.2 *Lasso*

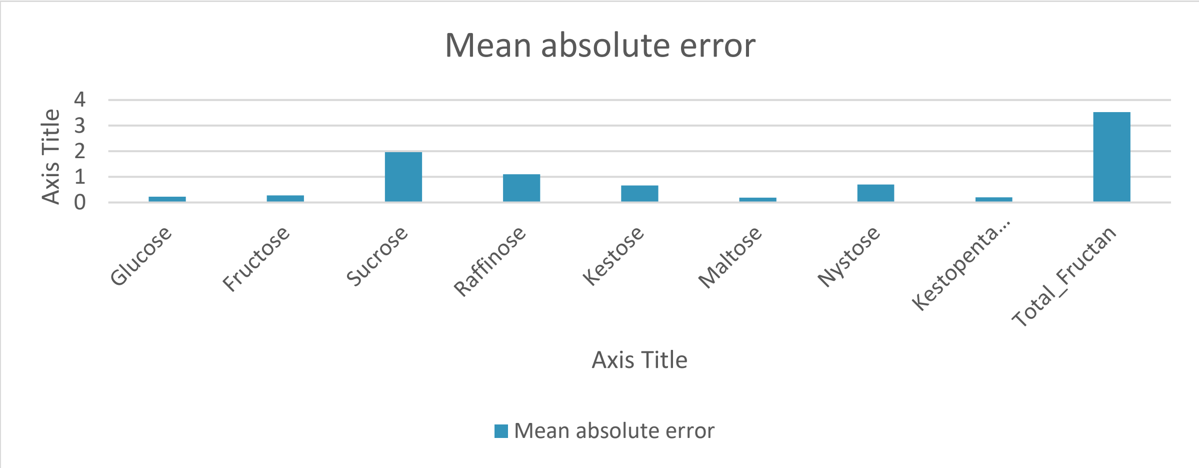


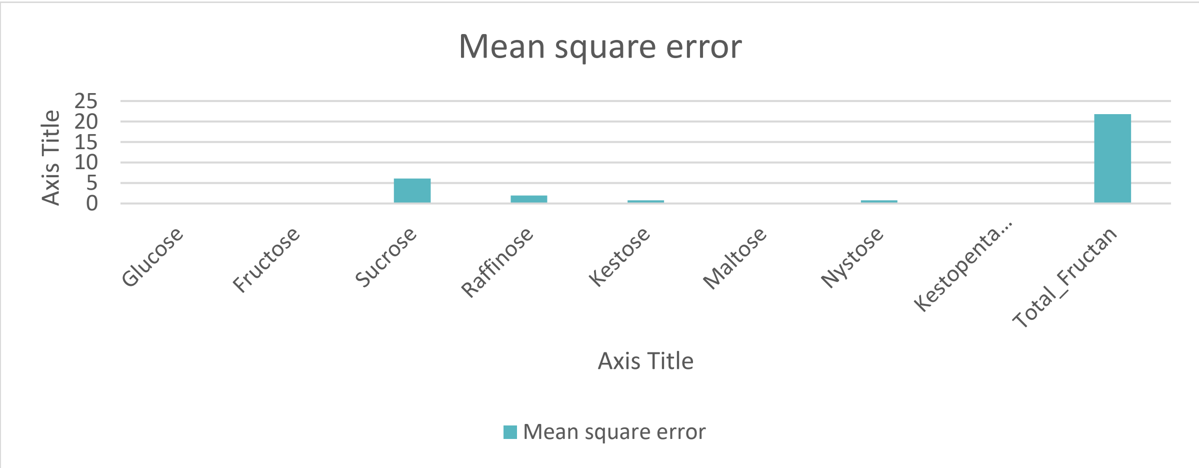




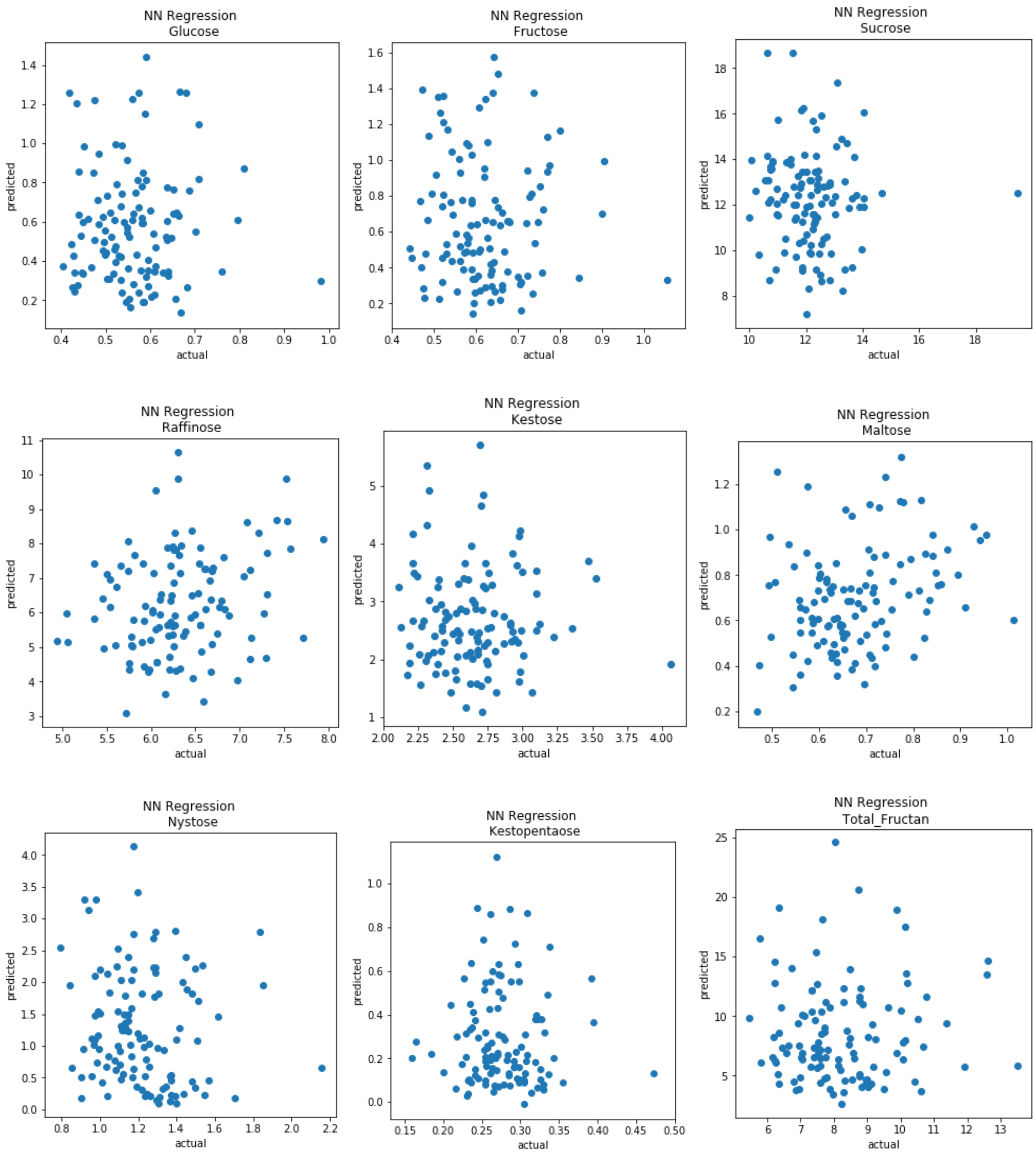
5.3.2.3 Elastic Net



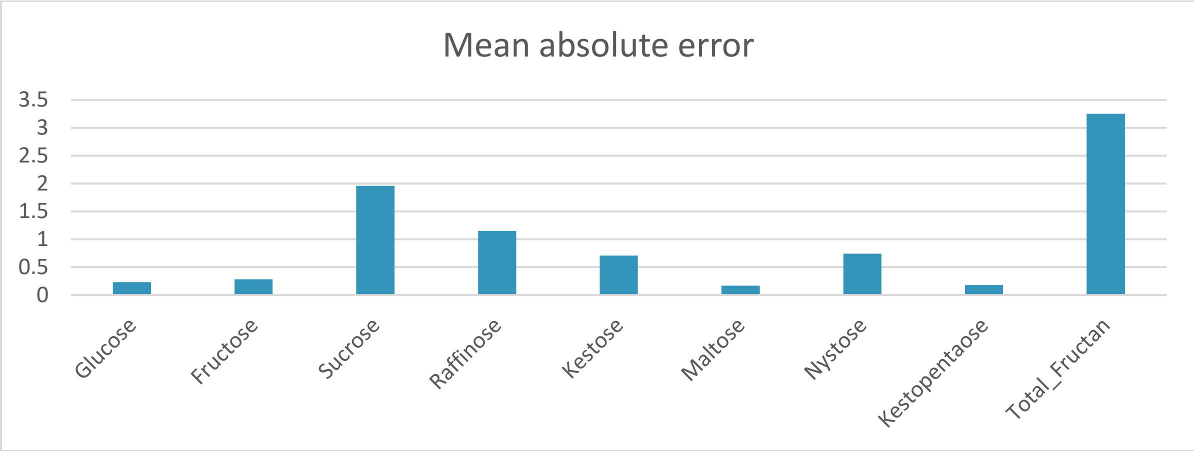


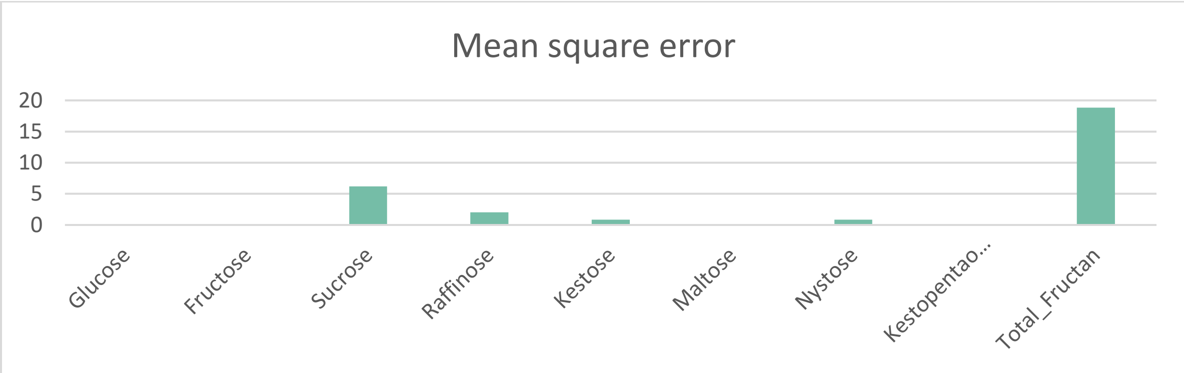


5.3.2.4 Deep Neural Networks


NN Regression
Glucose
NN Regression
Fructose
NN Regression
Sucrose
NN Regression
Raffinose
NN Regression
Kestose
NN Regression
Maltose
NN Regression
Nystose
NN Regression
Kestopentaose
NN Regression
Total_Fructan
predicted
actual


R^2 error
0
-5
-10
-15
-20
-25
-30
Glucose
Fructose
Sucrose
Raffinose
Kestose
Maltose
Nystose
Kestopentaose
Total_Fructan



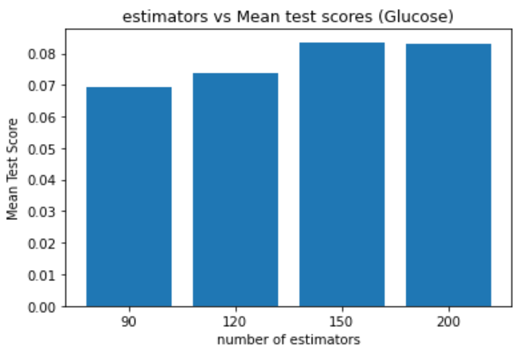


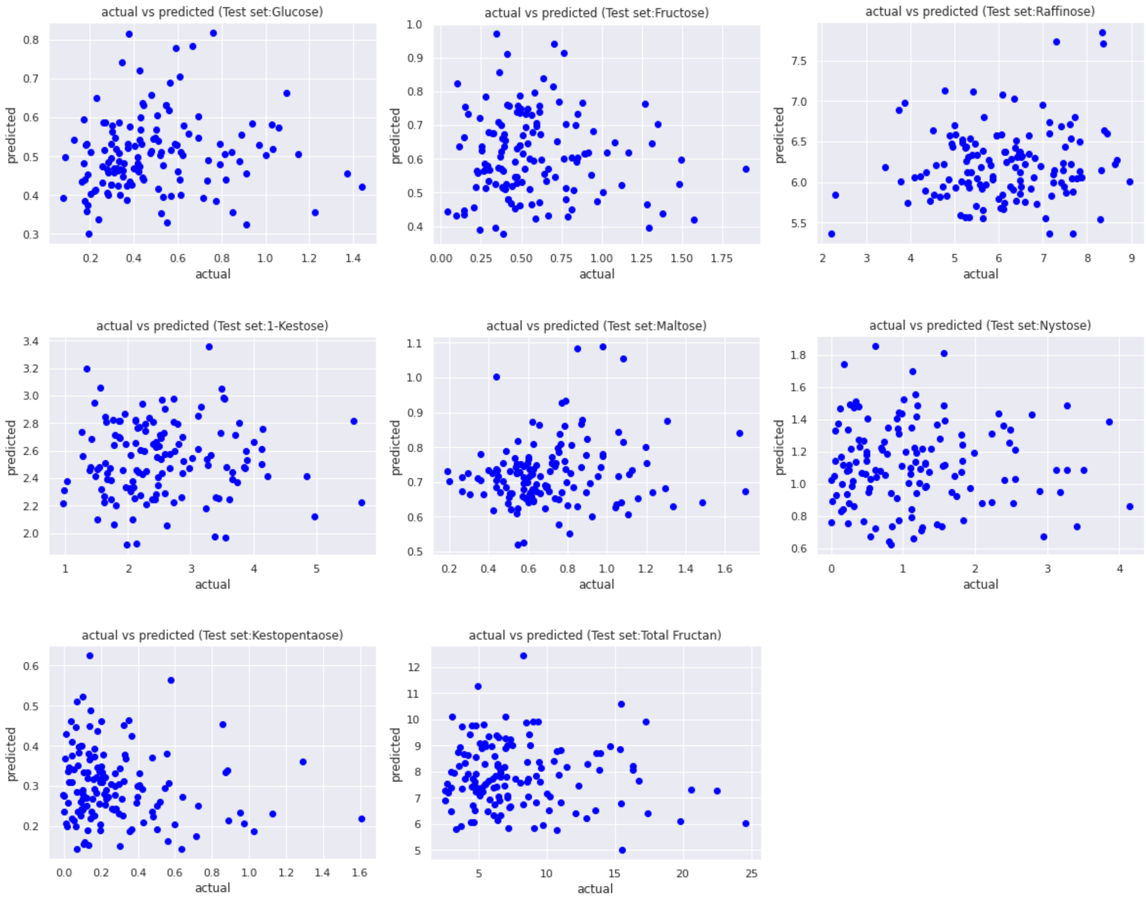
5.3.2.5 Random Forest Regression

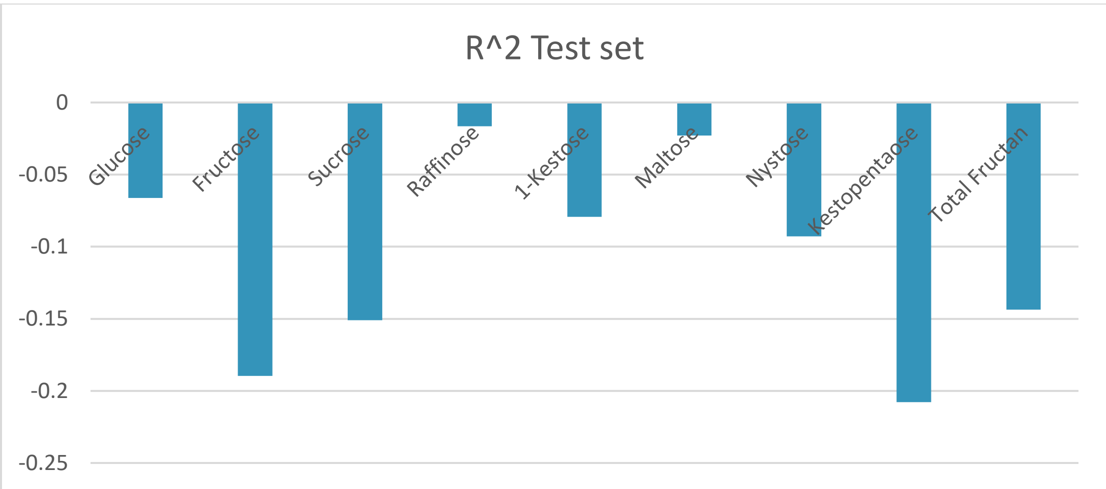
Choose the best parameters using GridSearchCV:

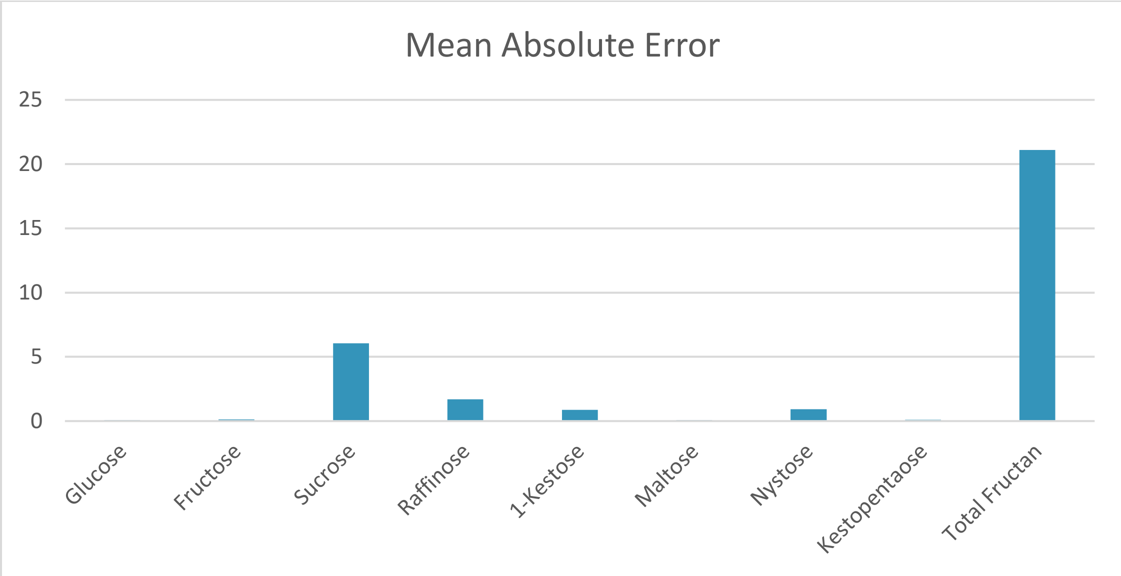
Considering Single Target Glucose:

Here the best score 0.0692 is found for number of estimators = 90




actual vs predicted (Test set:Glucose)
actual vs predicted (Test set:Fructose)
actual vs predicted (Test set:Raffinose)
actual vs predicted (Test set:1-Kestose)
actual vs predicted (Test set:Maltose)
actual vs predicted (Test set:Nystose)
actual vs predicted (Test set:Kestopentaose)
actual vs predicted (Test set:Total Fructan)
predicted
actual

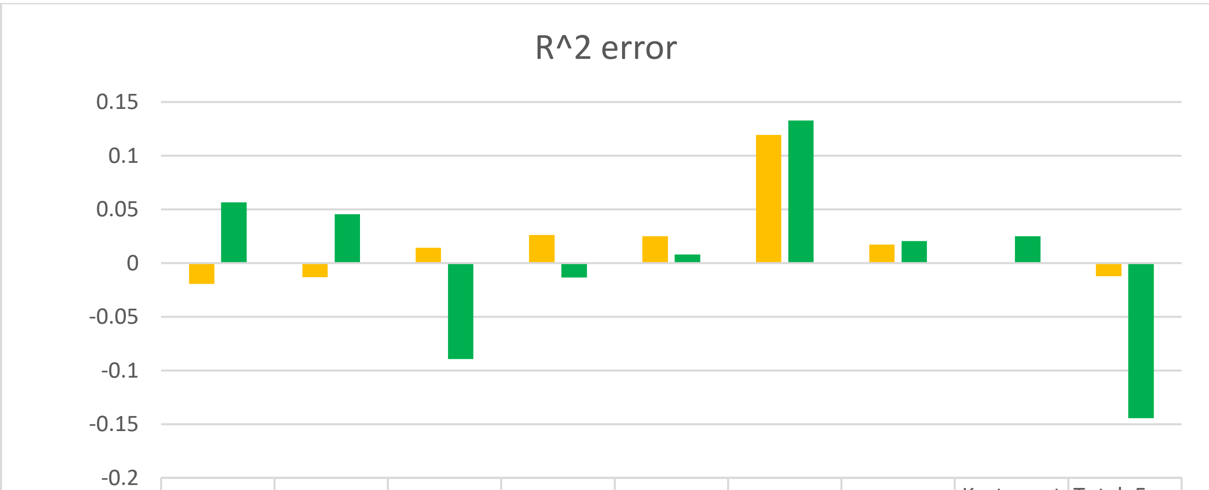

R^2 Test set
0
-0.05
-0.1
-0.15
-0.2
-0.25
Glucose
Fructose
Sucrose
Raffinose
1-Kestose
Maltose
Nystose
Kestopentaose
Total Fructan



6 EVALUATION AND CONCLUSION

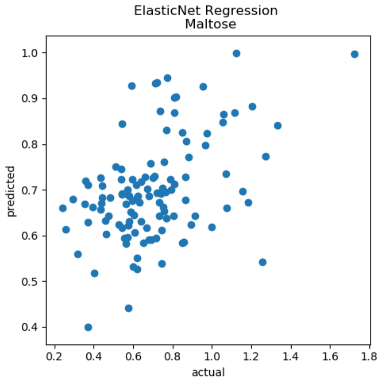
Among all the models run on each dataset using different normalization approaches, Elastic Net performs slightly better at least for Maltose, even though the results are not as good as expected.

In case of Ridge, Lasso and Elastic Net, since the slope was zero, tuning penalty term didn't help.



	Glucose	Fructose	Sucrose	Raffinose	Kestose	Maltose	Nystose	Kestopentaose	Total_Fructan
Lasso	-0.01924	-0.01317	0.014208	0.026226	0.025089	0.119368	0.017393	-0.00063	-0.01234
Elastic Net	0.056633	0.04544	-0.08924	-0.01355	0.007923	0.132907	0.02043	0.025075	-0.14442

Lasso Elastic Net


ElasticNet Regression
Maltose
predicted
actual
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8