DSCI352: Semester Project Final Report

Shiyi Qin

03 May, 2019

Contents

1 Project Description

1.1 Background

Solar power is one of the cleanest and most reliable forms of renewable energy available, and it has been brought into residential use since 1990s[1]. To supply usable solar power, photovoltaic (PV) systems are designed by means of photovoltaics. PV modules are made up of semiconductor materials which absorb heat from the solar rays and convert it into electric current to power people's home and business.

To optimize and extend the lifetime of the PV modules, it is important to study the degradation of these PV modules. Below are some common exposure methods and PV cell measurements used in research studies.

- Different Kinds of Indoor Accelerated Test:
  - Damp Heat (DH) Exposure
  - Humudity/Freeze (HF) Exposure
  - UV Irradiance Exposure
  - Dynamic Mechanic Load (DML)
  - Thermocycle
- Cell Measurements
  - Current-Voltage (I-V) Curve
  - Suns-Voc
  - Electroluminescence (EL) Image

1.2 Research Question

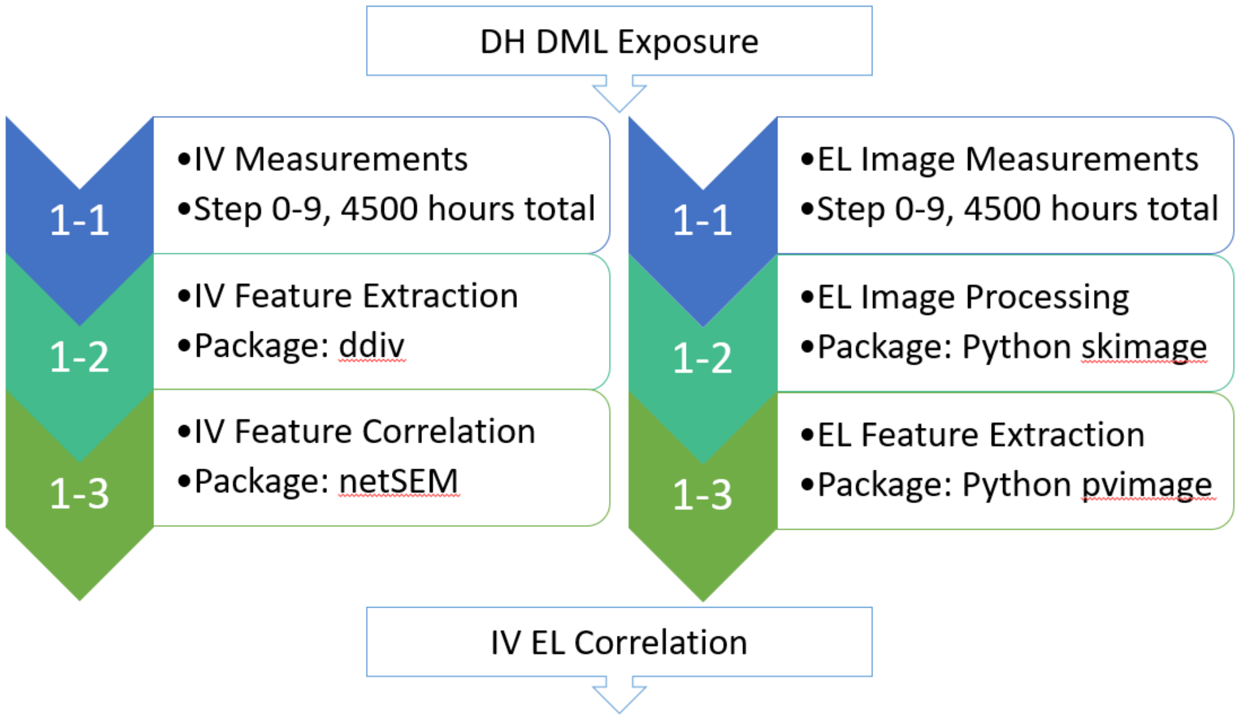
The goal of the this project is to analyze the change in the I-V features, EL images, and most importantly, to find the correlation between I-V and EL features. Additional scope includes establishing predictive models which can use I-V features to predict EL features. This can help model the degradation of the PV modules and develop more efficient PV modules in the long run.

In this project, I am going to investigate the degradation of three brands of PV modules, categorized as H, I, and J. The sample info is shown below:

```
##       sanm  ID brand     brandname   brandmodel shortname sics celltype
## 1  sa40116 CS1    CS CanadianSolar     CS6K-285        H1 mono   Al-BSF
## 2  sa40117 CS2    CS CanadianSolar     CS6K-285        H2 mono   Al-BSF
## 3  sa40118 CS3    CS CanadianSolar     CS6K-285        H3 mono   Al-BSF
## 4  sa40119 CS4    CS CanadianSolar     CS6K-285        H4 mono   Al-BSF
## 5  sa40220 QC1    QC    TrinaSolar TSM-DD05A.05        I1 mono   Al-BSF
## 6  sa40221 QC2    QC    TrinaSolar TSM-DD05A.05        I2 mono   Al-BSF
## 7  sa40222 QC3    QC    TrinaSolar TSM-DD05A.05        I3 mono   Al-BSF
## 8  sa40223 QC4    QC    TrinaSolar TSM-DD05A.05        I4 mono   Al-BSF
## 9  sa40224 SW1    SW    SolarWorld       SW-285        J1 mono     PERC
## 10 sa40225 SW2    SW    SolarWorld       SW-285        J2 mono     PERC
## 11 sa40226 SW3    SW    SolarWorld       SW-285        J3 mono     PERC
## 12 sa40227 SW4    SW    SolarWorld       SW-285        J4 mono     PERC
```

For each type of PV moddules, four samples are tested on. The PV modules are exposed to the DH chamber and DML to achieve accelerated degradation for this study. The I-V data and the electroluminescence (EL) image of each module are taken after every 500 hours of exposure. Thus, the experiment is divided into ten steps from 0 to 9 with an overall exposure time of 4500 hours.

1.3 Flow Diagram



A flow diagram is shown above.

1.4 Ideal Data Set

The ideal data set should include a sample identifier for each module per step along with the I-V features (such as Pmp, Isc, Voc, Vmp, Imp, FF, Rsh, and Rs) and EL features (such as intensity and number of cracks).
The I-V features are obtained both from the actual measurements and from the extraction algorithms using the raw I-V curves.
The EL features are extracted from the cleaned EL images which are saved as JPEG files. This process requires image processing in Python.

So far, I have successfully extracted I-V features and some EL features, as illustrated below. I am currently working on the Python code to try to extract features related to busbar width.

- 3 brands, 4 samples each, 10 steps
- 120 obs, 32 variables

```
## [1] 120  32
```

```
##  [1] "imxp"     "row_key"  "rssr"     "eimp"     "ffff"     "intemedi"
##  [7] "eisc"     "nucrcell" "exst"     "rssh"     "intevari" "pmpp"
## [13] "epmp"     "vocc"     "ersh"     "nucr"     "tamb"     "vmxp"
## [19] "intemean" "evoc"     "evmp"     "exrs"     "exff"     "poay"
## [25] "ishc"     "bbwmean"  "bbnumber" "sanm"     "ID"       "expt"
## [31] "step"     "brand"
```

1.5 Data Source

All the raw data, including I-V and EL data files, are obtained from the SDLE research center as well as its data providers. The raw data files are downloaded from the vuv-lab group Google Drive. After data cleaning and tidying, the data are ingested into HBase.

1.6 Packages

For I-v data (R):
- tidyverse - dplyr - ggplot2 - GGally - ddiv - netSEM
- SparseM

For EL image (Python):
- skimage - glob - pandas - pvimage - os - numpy - scipy

2 Databook

2.1 Metadata

```
## 'data.frame':    240 obs. of  7 variables:
##  $ dtyp  : chr  "iv" "iv" "iv" "iv" ...
##  $ mobr  : chr  "H1" "H1" "H1" "H1" ...
##  $ styp  : chr  "ss8" "ss8" "ss8" "ss8" ...
##  $ time  : int  0 500 1000 1500 2000 2500 3000 3500 4000 4500 ...
##  $ spid  : chr  "H11" "H11" "H11" "H11" ...
##  $ maty  : chr  "Mono-Si" "Mono-Si" "Mono-Si" "Mono-Si" ...
##  $ s_name: chr  "sa40116_00-00-00-01-ivdhdml" "sa40116_00-01-01-01-ivdhdml" "sa40116_00-02-02-01-ivd
```

- dtyp: meta table primary data type
- mobr: module brand
- styp: meta table secondary data type
- time: exposure time [hr]
- spid: sopar panel id
- maty: material type
- s_name: file name, lined to row_key

2.2 I-V Curve Data

```
## 'data.frame':    5832 obs. of  2 variables:
##  $ V: num  -0.274 -0.411 -0.481 -0.499 -0.504 -0.503 -0.512 -0.512 -0.525 -0.543 ...
##  $ I: num  9.33 9.31 9.29 9.27 9.27 ...
```

- I: Current [A]
- V: Voltage [V]

2.3 EL Data

Above is a raw EL image sample.

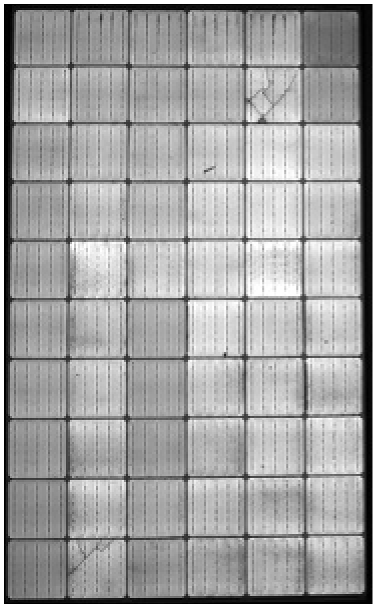


Figure 1: EL Data Sample-Raw

2.4 Main Data Set

```
## [1] 120  32
```

```
## 'data.frame':    120 obs. of  32 variables:
##  $ imxp    : num  9.08 9.08 9.03 9.02 9 ...
##  $ row_key : chr  "sa40116_00-00-00-01-ivdhdml" "sa40116_00-01-01-01-ivdhdml" "sa40116_00-02-02-01-i
##  $ rssr    : num  0.452 0.47 0.475 0.453 0.501 ...
##  $ eimp    : num  9.06 9.08 9.02 9 8.99 ...
##  $ ffff    : num  77.8 78.1 77.8 77.5 78 ...
##  $ intemedi: int  178 179 174 177 158 167 146 141 160 68 ...
##  $ eisc    : num  9.48 9.43 9.43 9.41 9.3 ...
##  $ nucrcell: int  2 2 2 2 2 2 3 3 3 3 ...
##  $ exst    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ rssh    : num  85.8 86.6 110.8 101.3 87.6 ...
##  $ intevari: num  1021 929 776 982 1745 ...
##  $ pmpp    : num  293 293 291 290 288 ...
##  $ epmp    : num  292 293 291 289 287 ...
##  $ vocc    : num  39.7 39.8 39.7 39.7 39.6 ...
##  $ ersh    : num  144 146 141 133 119 ...
##  $ nucr    : int  14 14 14 14 14 14 15 15 15 15 ...
##  $ tamb    : num  25 25 25 25 25 ...
##  $ vmxp    : num  32.2 32.3 32.3 32.1 32 ...
##  $ intemean: num  176 176 171 175 154 ...
##  $ evoc    : num  39.7 39.8 39.7 39.7 39.6 ...
##  $ evmp    : num  32.3 32.3 32.3 32.1 31.9 ...
##  $ exrs    : num  0.508 0.469 0.465 0.479 0.516 ...
##  $ exff    : num  77.7 78.1 77.7 77.4 77.9 ...
##  $ poay    : int  1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ...
##  $ ishc    : num  9.48 9.44 9.43 9.41 9.31 ...
##  $ bbwmean : num  0.109 0.109 0.108 0.107 0.106 ...
##  $ bbnumber: num  5 5 5 5 5 ...
```

```
## $ sanm    : chr  "sa40116" "sa40116" "sa40116" "sa40116" ...
## $ ID      : chr  "CS1" "CS1" "CS1" "CS1" ...
## $ expt    : int  0 500 1000 1500 2000 2500 3000 3500 4000 4500 ...
## $ step    : int  0 1 2 3 4 5 6 7 8 9 ...
## $ brand   : chr  "CS" "CS" "CS" "CS" ...
```

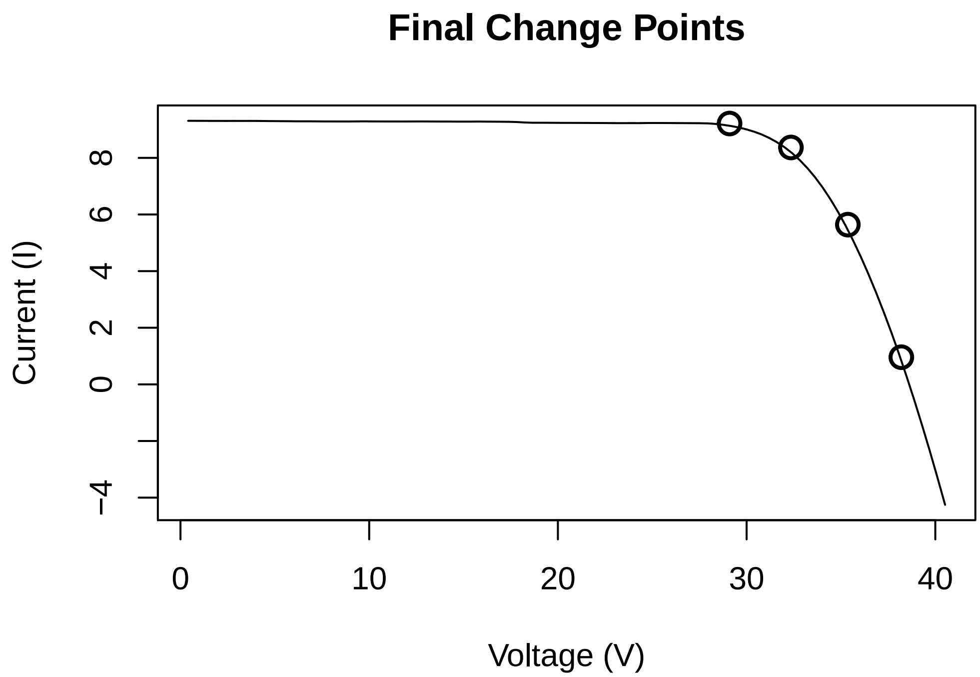
- imxp: current at max power [A]
- row_key: row identifier with sample number, step number, and measurement type
- rssr: series resistance [Ohm]
- eimp: extracted Imp [A]
- ffff: fill factor [%]
- intemedi: median intensity [pixel]
- eisc: extracted Isc [A]
- nucrcell: number of cracked cell
- exst: extracted step number
- rssh: shunt resistance [Ohm]
- intevari: variance of intensity [pixel]
- pmpp: max power [W]
- epmp: extratced max power [W]
- vocc: open circuit voltage [V]
- ersh: extracted Rsh [V/A]
- nucr: number of cracks
- tamb: ambient temperature [degree C]
- vmxp: voltage at max power [V]
- intemean: mean intensity [pixel]
- evoc: extracted Voc [V]
- evmp: extracted Vmp [V]
- exrs: extracted Rs [V/A]
- exff: extracted fill factor [%]
- poay: plane of array pyramometer [W/m^2]
- ishc: short circuit current [A]
- bbwmean: normallized busbar width mean
- bbnumber: number of busbar identified by Python function
- sanm: sample number
- ID: sample ID
- expt: exposure time [hr]
- step: step number
- brand: sample brand

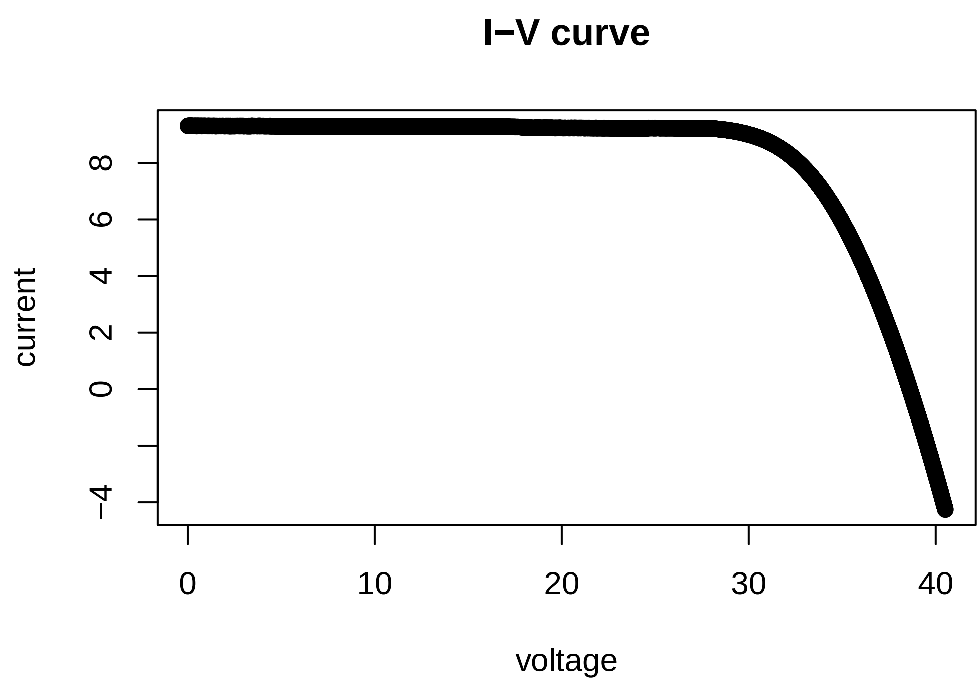
3 Data Cleaning and EDA

3.1 I-V

3.1.1 I-V Data Cleaning

I mainly used data pipelines to clean and tidy the I-V raw data as well as the measured I-V features. The extracted I-V features were generated using the “ddiv” package. An example is given below.

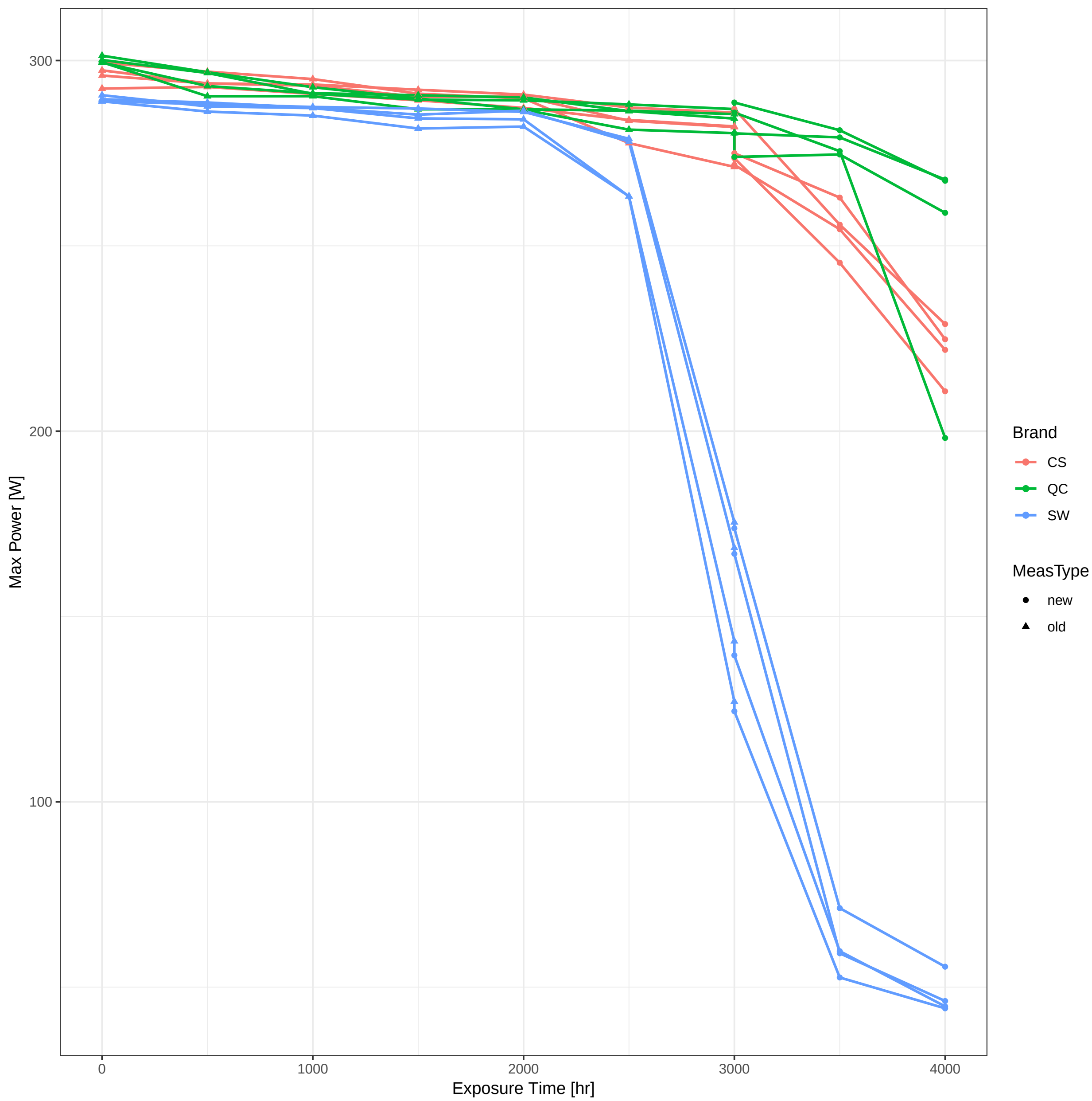


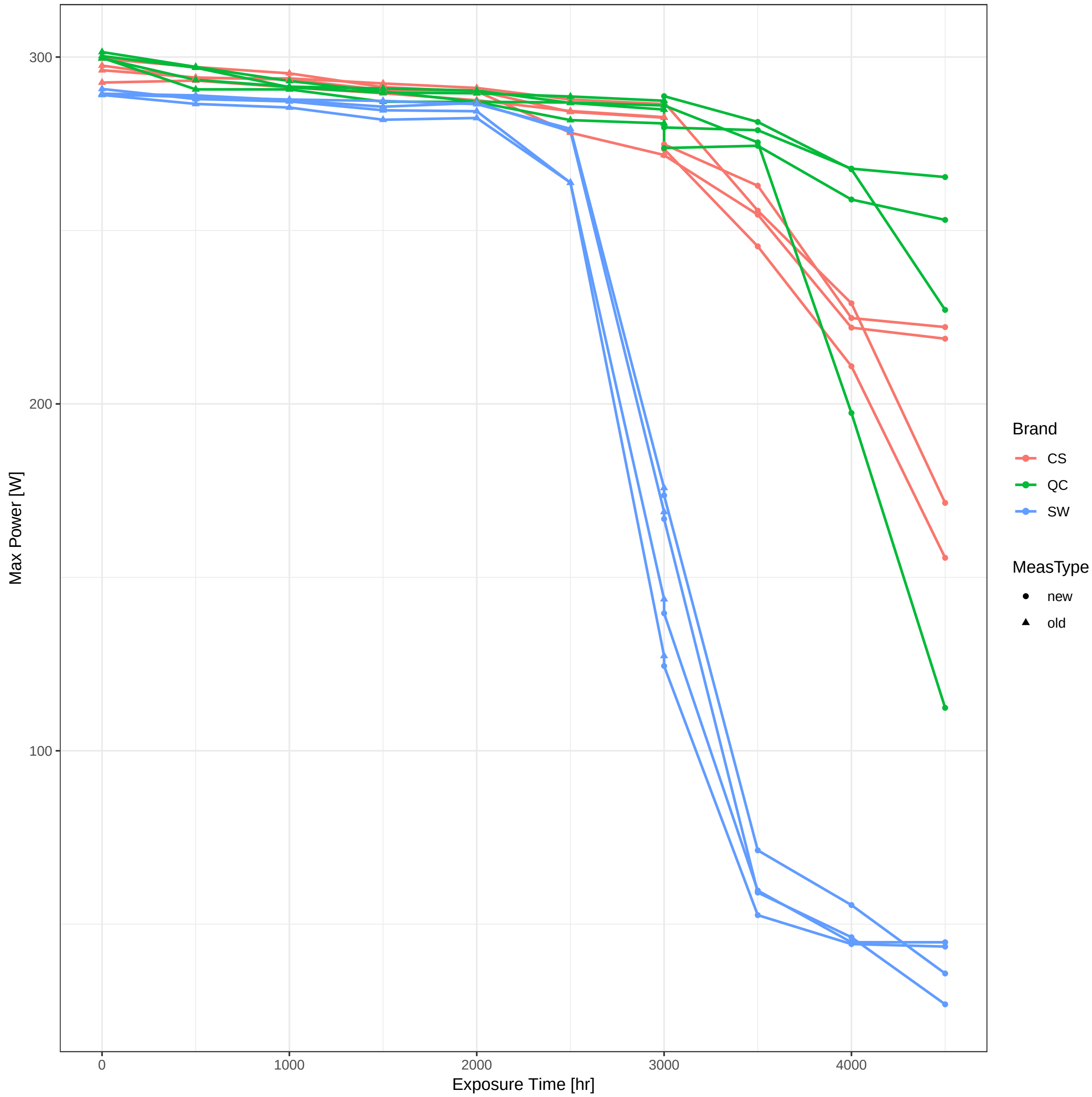



```
##       step  Isc     Rsh    Voc    Rs    Pmp   Imp    Vmp    FF Cutoff
## y381     1 9.31 453.765 38.631 0.451 271.85 8.805 30.875 75.59     NA
```

3.1.2 I-V Visualization

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

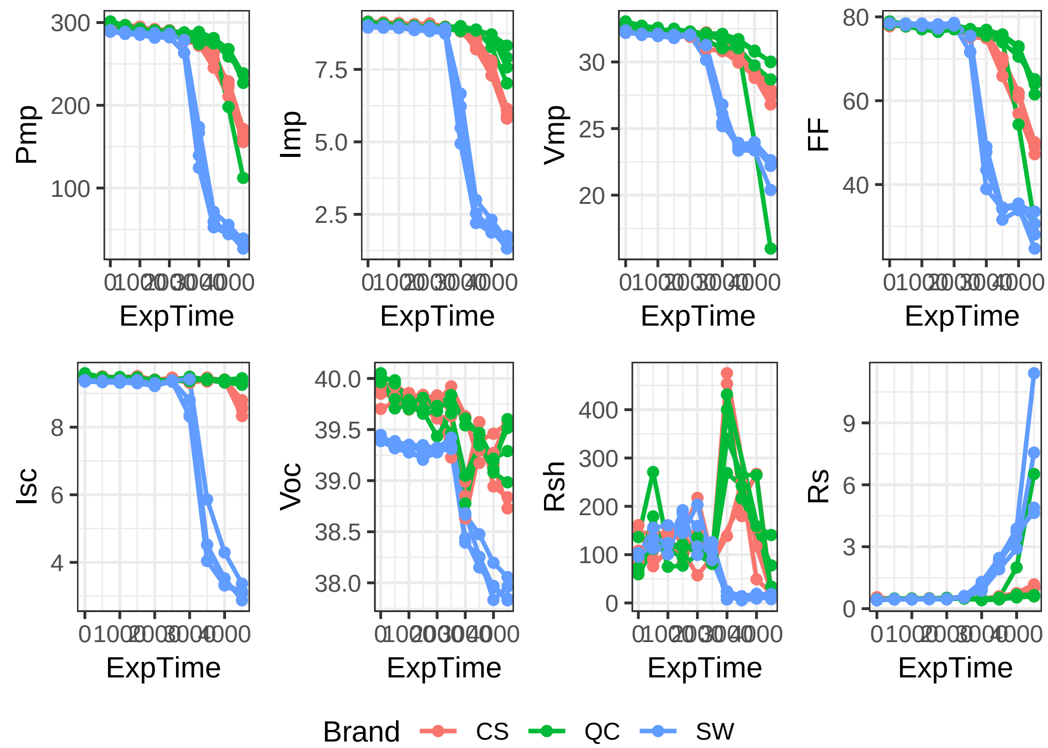




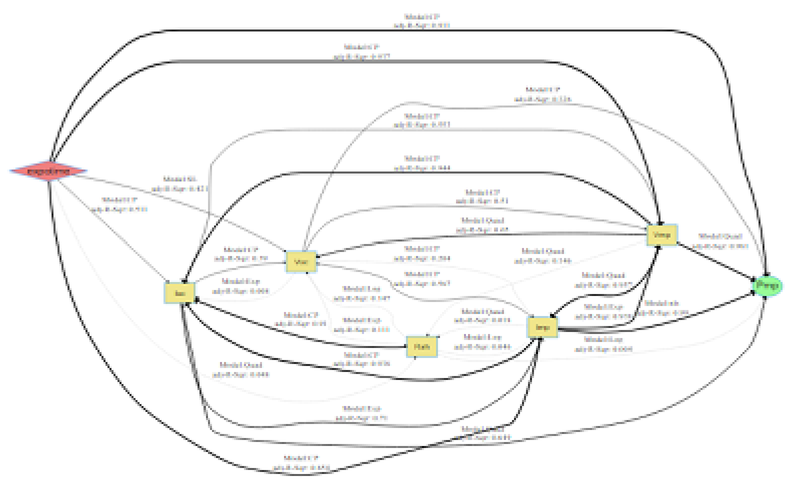
Extracted and measured features showed similar trend. It is also very obvious that SW/J performed worst among the three brands, with the greatest amount of max power decrease.

I also looked at the change in other I-V feataures in response to time. Their trends are consistent with the theory, except that the shunt resistance data has a lot of noises, so it is hard to summarize the trend.

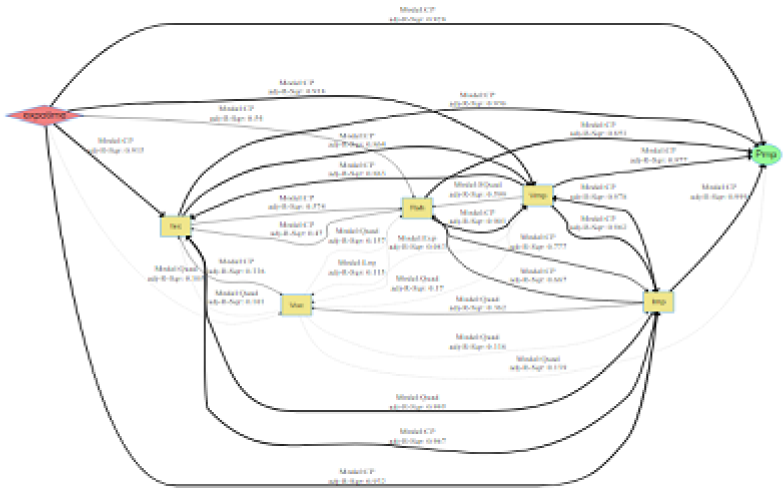
```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```



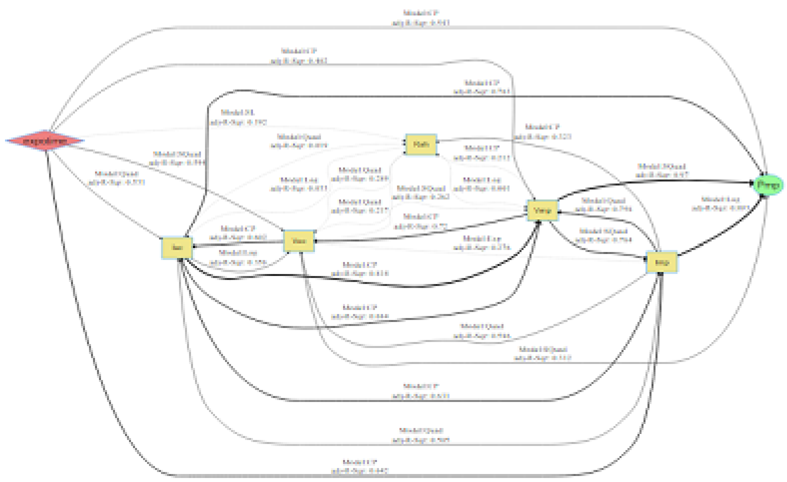
To study the correlation between I-V features. I also did netSEM analysis on each brand.



netSEM results: CS:



SW:



QC:

3.2 EL

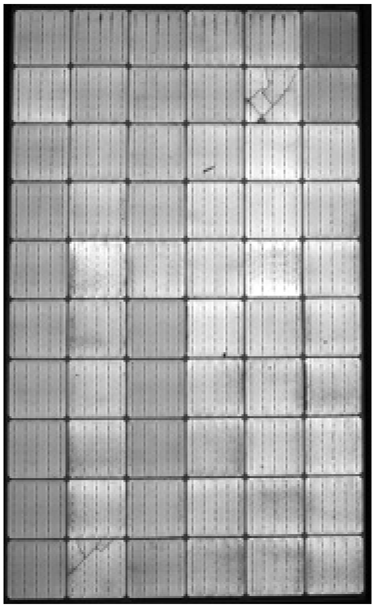
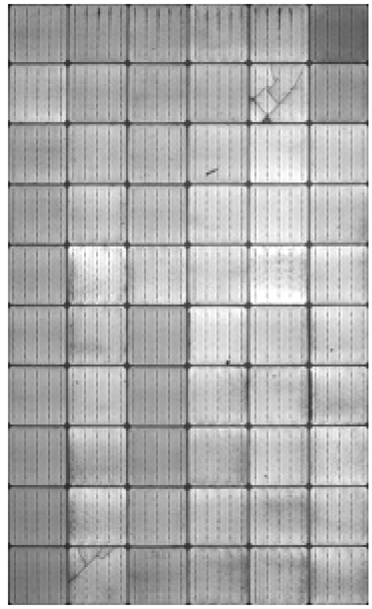
3.2.1 EL Cleaning

EL cleaning was performed in Python. Although the raw images had similar size and layout, most of the images were skewed and/or had undesired background. In order to make further analysis easier and robust, I learned Python for image background removal and planar index.

Basically, the idea was to grab the intensity of the raw image for each row and column and place an arbitrary threshold to cut out the dark background. To achieve this, the raw image needs to be denoised using ndimage.median_filter() function. Next the convex hull, a tight fitting convex boundary around the points or the shape, is determined using the threshold. The edge and corner are obtained based on the detection of the first and last row and column of the image. The interception points are calculated from there. During this process, since the border of the image may be blurred into the background, which make it hard to separate, around 60% of the dimension is kept. The position of the rest of the desired image vector is then obtained.

Finally, the modules are divided into individual cells (6*10), and each cell image is saved as an individual file for further feature extraction.

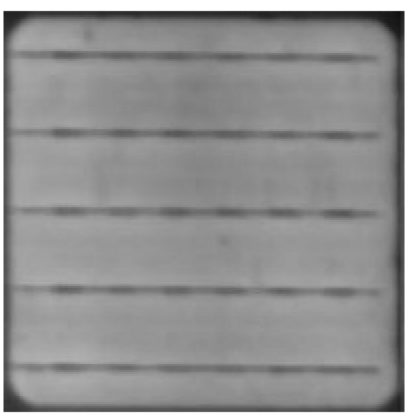
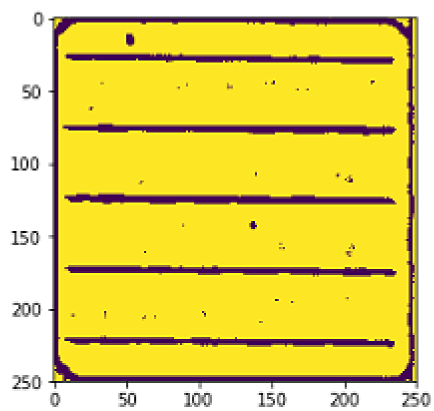
Sample images before and after planar index are shown below.

 After planar index: 

3.2.2 EL Features Extraction

3.2.2.1 Normalized Busbar Width

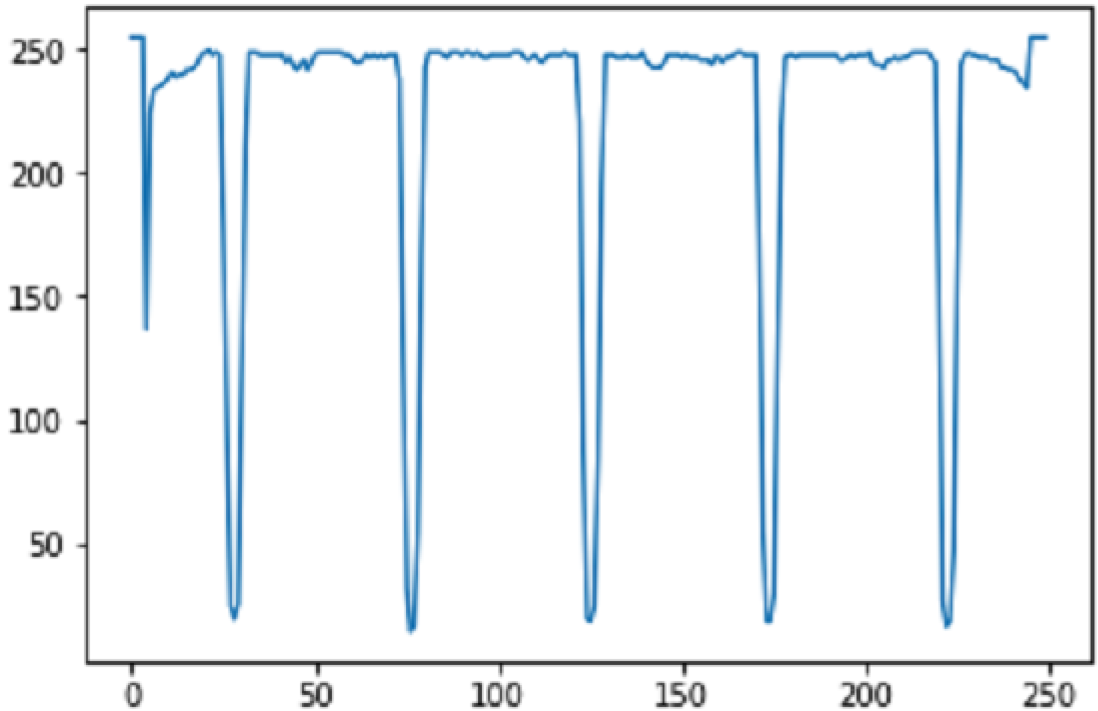
Busbars wire solar cells together to create higher voltages. Their width is a useful indicator for EL image and cell performance in general. Taking the individual cell image as an example, there are five busbars, which can be identified as the five horizontal lines here.

An image data file is essentially an array of numbers which represent the itensity at each pixel position.

```
In [20]: img
Out[20]:
array([[75, 73, 72, ..., 74, 77, 77],
       [72, 71, 70, ..., 76, 77, 77],
       [70, 71, 72, ..., 77, 76, 76],
       ...,
       [61, 64, 66, ..., 71, 69, 69],
       [63, 63, 64, ..., 69, 68, 68],
       [63, 63, 64, ..., 69, 68, 68]], dtype=uint8)
```

In Pythong, I tried image filters such as cv2.GaussianBlur() and cv2.adaptiveThreshold(), whcih helped convert the original image to the following image with sharper contrast as shown above.



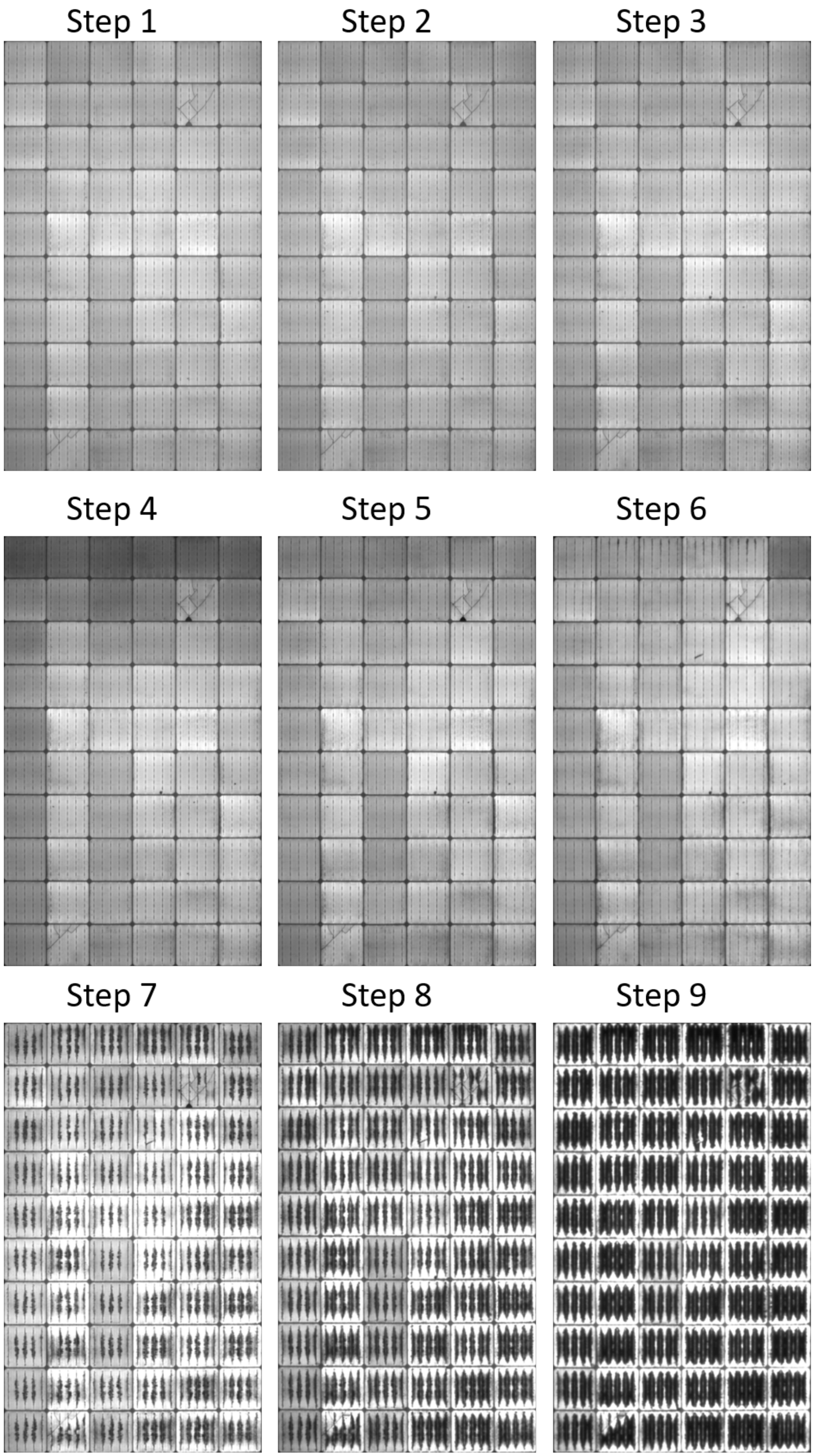
Next, I went through each column, plotted the average intensity, and identified the min_y position. Using those min_y positions as the center, the function was able to search for and calculate the normalized busbar width.

3.2.2.2 Intensity

The intensity was extracted using the r_params() function from the pvimage package. The mean, median, and variance values are extracted.

3.2.3 EL Visualization

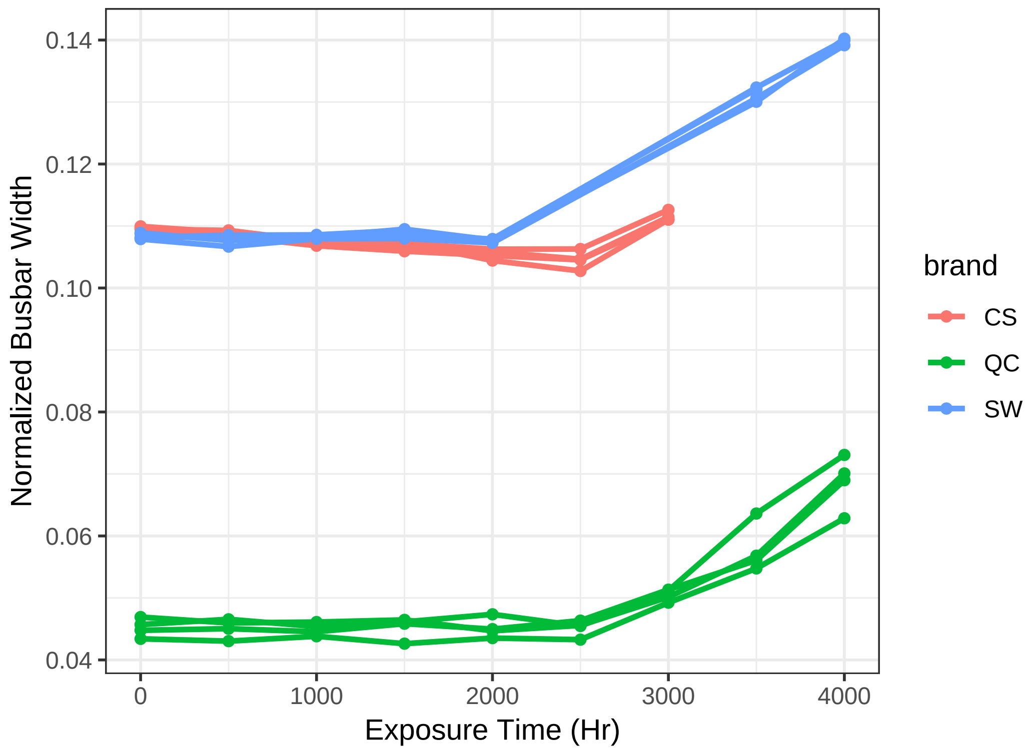
3.2.3.1 EL Step Change

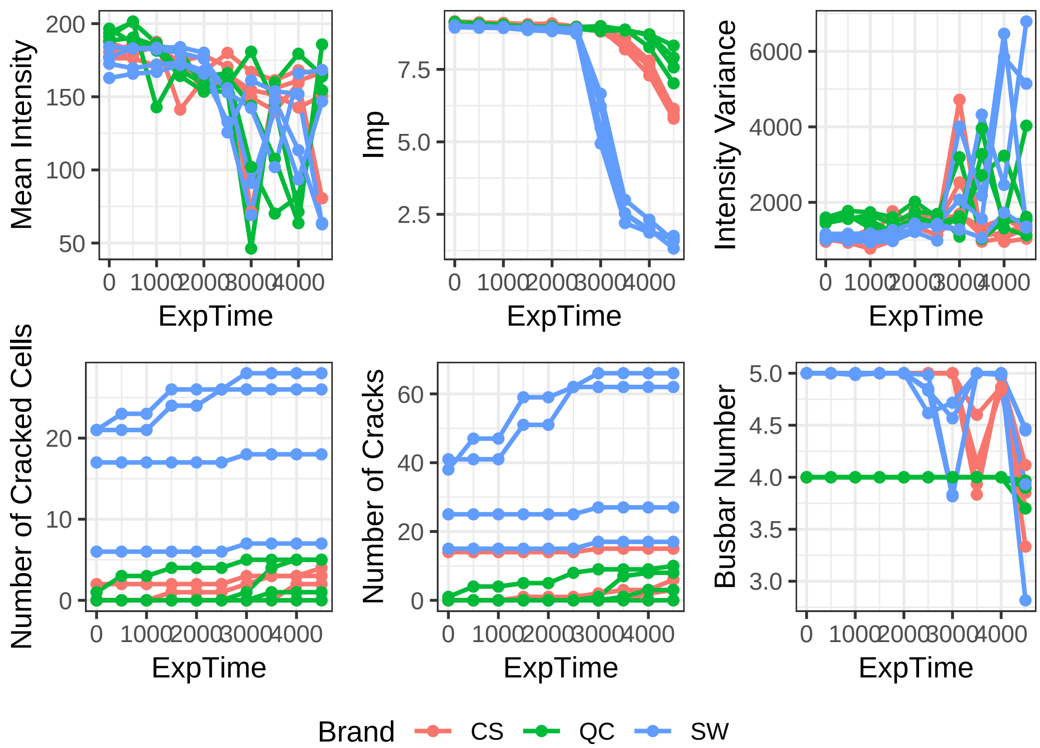


As the exposure time increases, cell darkens and more cracks appear.

3.2.3.2 EL Normalized Busbar Width vs. Exposure Time

Busbar width increases as the exposure time increases. However, towards the end of the exposure, some cells got so destroyed that the Python function was not able to identify the correct number of busbar, hence those points are NA'd. Another thing I noticed is that the normalized busbar width exhibited little change over the first couple of steps. The point of change takes place mainly after 2,500 hours.

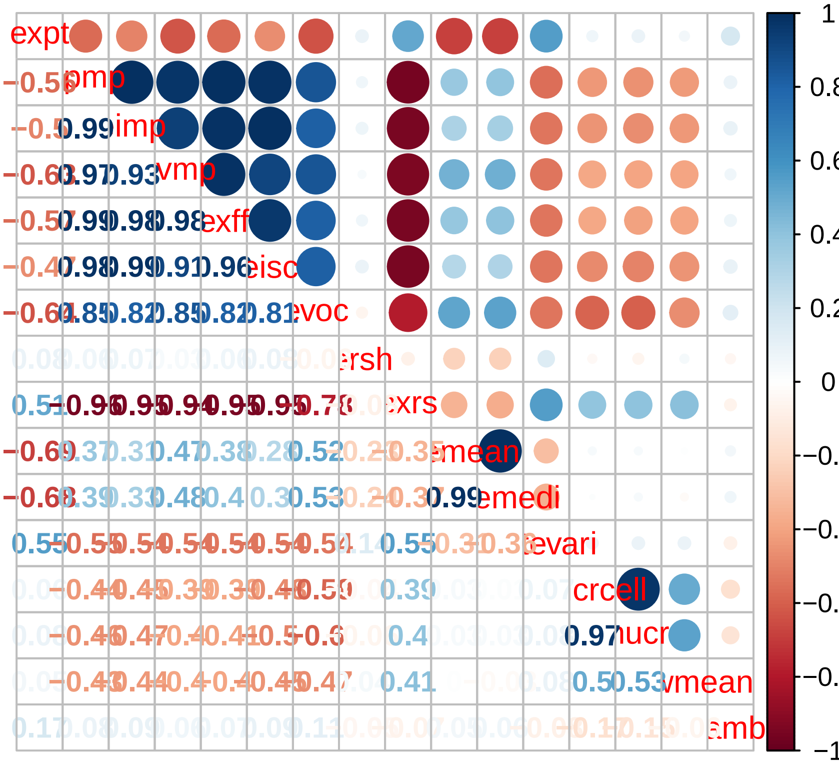




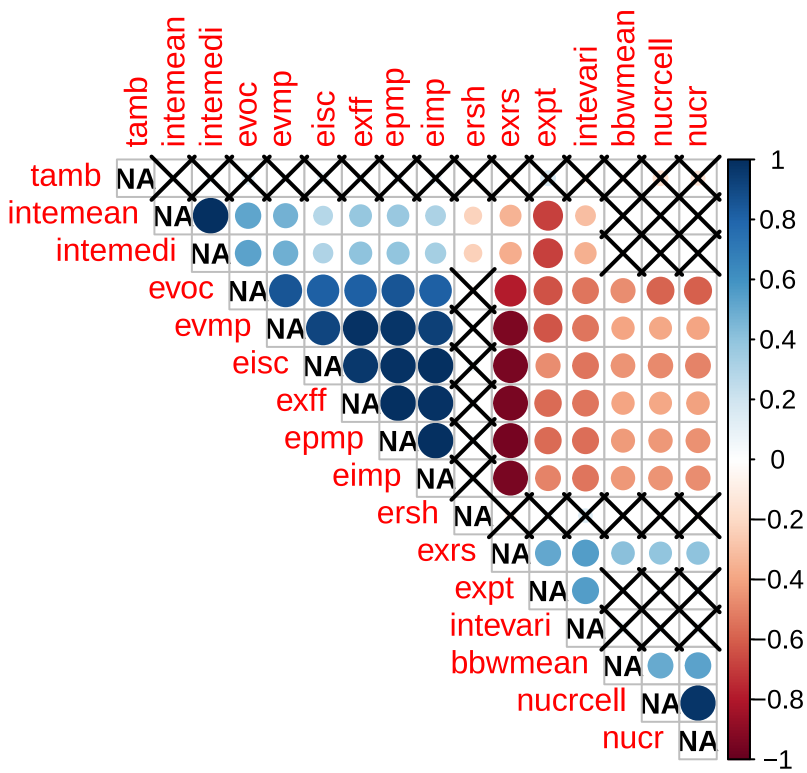
3.2.4 I-V EL Correlation Analysis

Keeping only the numerical values, I got the following correlation. Blue indicates positive correlation, and red indicates negative correlation. The larger and darker the circle, the stronger the correlation.
The I-V features are arranged at top left, and the EL features are arranged at bottom left. The triangular region at top left shows strong correlation within the I-V features themselves. However, the correlation among the EL features is not so strong. The rectangular region at top right indicates moderate I-V/EL correlation.

```
## corrplot 0.84 loaded
```

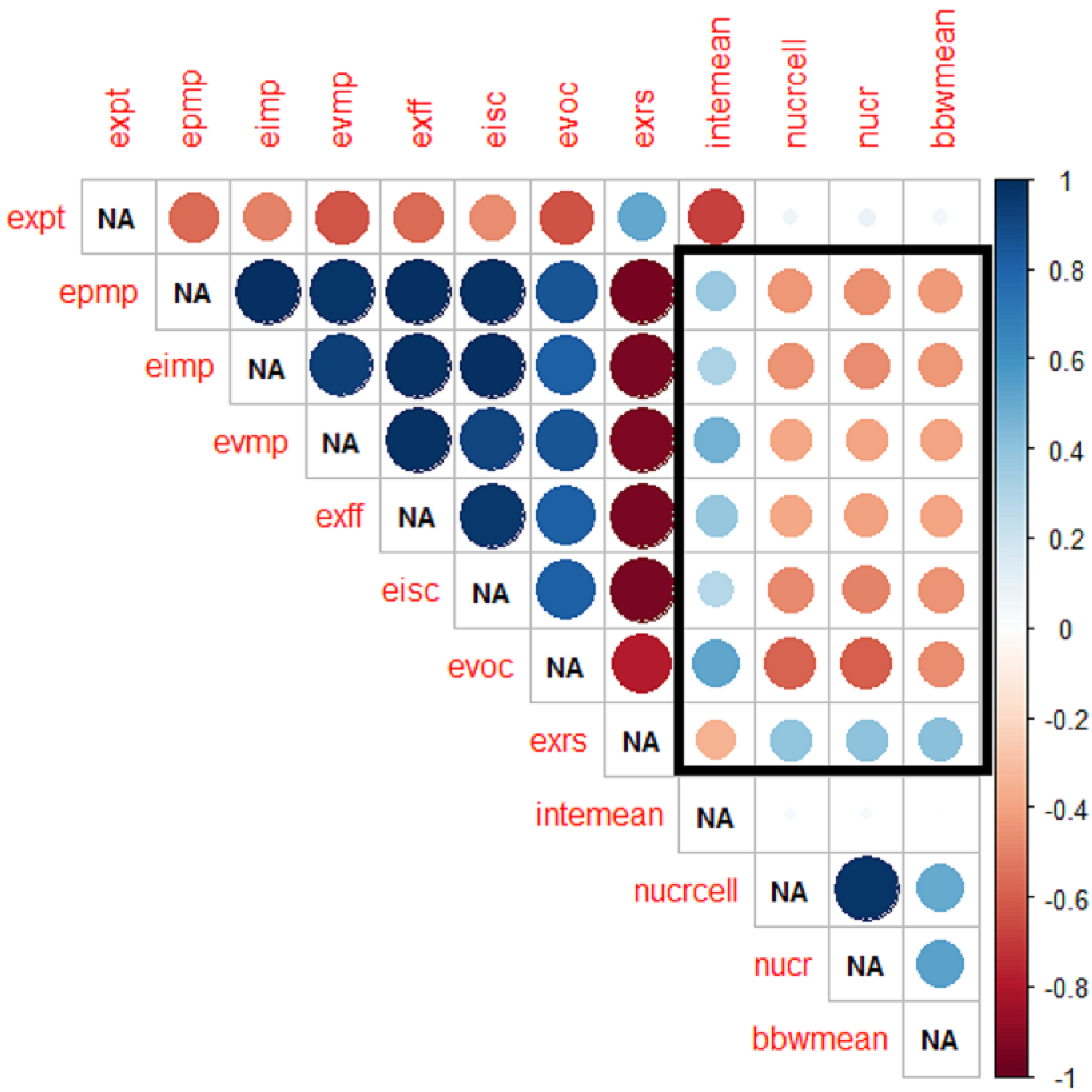


I applied hClust and significance test (CI=95%) to the above correlation data, and crossed out some insignificant points.



After removing some less important features based on the above plot, a more condensed version of the correlation plot is obtained. There are 12 features left.

- expt (exposure time)
- 7 I-V features (epmp, eimp, evmp, exff, eisc, evoc, exrs)
- 4 EL features (intemean, nucrcell, nucr, bbwmean)



The IV-EL correlation is shown in the rectangular box. All four EL features exhibited relatively strong correlation with the I-V features, which suggests further modeling building. The number of cracked cells/number of cracks have weak correlation with the exposure time, it is probably due to the random manufacturing or handling differences during the process.

4 Modeling and Statistical Learning

Based on the correlation plot, I decided to add additional scope to the project, where I tried to fit models to predict EL features, and I mainly focused on using the EL mean intensity as the response.

4.1 Linear Regression

4.1.1 Data Re-scale and Variable Selection

I started off looking at the entire data set. I performed variable normalization and subset selection. Based on the R2/Cp/BIC plots, the best subset appears to be at when number of variables equals 5, and the corresponding variables are shown below.

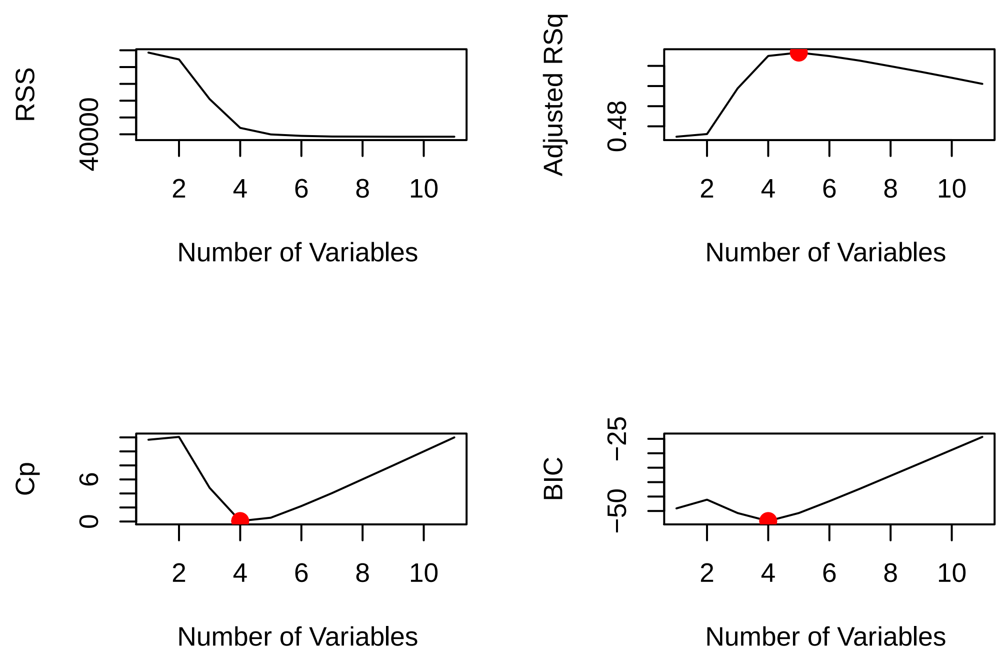
```
## -- Attaching packages --------------------------------------------------------- tidyverse 1.2.1 --
```

```
## v tibble  2.0.1     v purrr   0.3.0
## v tidyr   0.8.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ----------------------------------------------------------- tidyverse_conflicts() --
## x gridExtra::combine() masks dplyr::combine()
## x dplyr::filter()      masks stats::filter()
## x dplyr::lag()         masks stats::lag()
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```



4.1.2 Initial Linear Regression Model

With those features, I performed linear regression.

```
## (Intercept)        expt        eisc        evoc        nucr     bbwmean
```

```
##   117.24660   -42.40984   -60.21708   139.56515    47.09972    11.77766
```

```
## Warning: package 'Metrics' was built under R version 3.5.3
```

```
##
## Attaching package: 'Metrics'
```

```
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

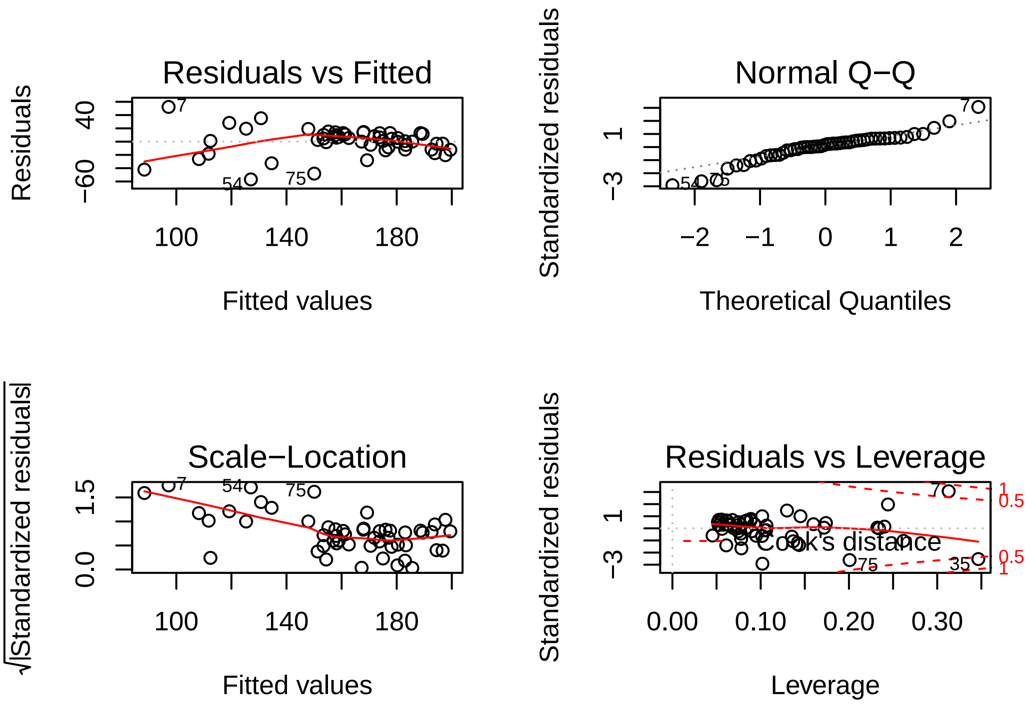
```
##
## Call:
## lm(formula = intemean ~ ., data = train5)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -56.916  -9.466   3.427  11.895  52.286
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   117.25      27.29   4.296 8.91e-05 ***
## expt          -42.41      14.42  -2.942  0.00510 **
## eisc          -60.22      22.05  -2.731  0.00893 **
## evoc          139.57      32.66   4.273 9.60e-05 ***
## nucr           47.10      14.75   3.192  0.00255 **
## bbwmean        11.78      10.96   1.075  0.28800
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.58 on 46 degrees of freedom
## Multiple R-squared:  0.6551, Adjusted R-squared:  0.6177
## F-statistic: 17.48 on 5 and 46 DF,  p-value: 1.143e-09
```

```
## [1] "Prediction Accuracy: 0.69"
```

```
## [1] "RMSE: 24.61"
```

All of those selected features are statistically significant except for the busbar width. However, the model does not fit the data very well. I got a model fitting with adjusted R-sqaured = 0.6. The prediction accuracy is 0.69 with an RMSE of around 24.6.

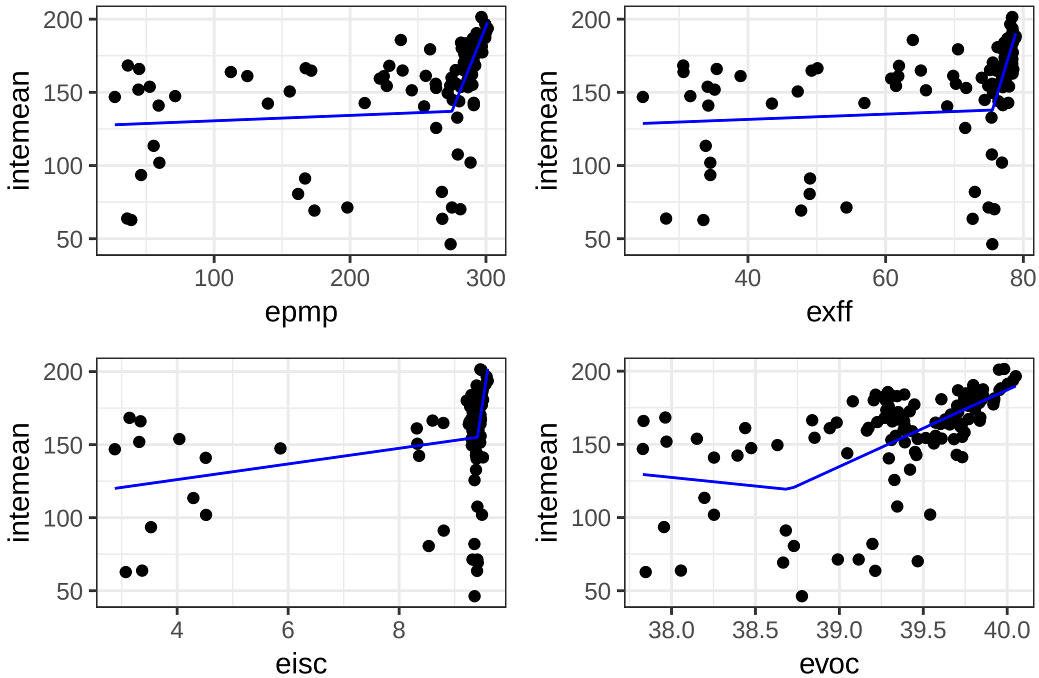
I also checked the residual plot of the linear model.



We can observe some sort of trend in the residuals, and the QQ plot also indicates the data is not normally distributed. Therefore, I started to look at other models.

4.2 Linear Regression with Changepoints

Since I observed some change point behavior from my EDA previously, where certain features remained almost constant for the a long period of exposure time and changed drastically only after a certain point. Therefore, I decided to find if there are segmented relationships that can reflect changepoints between the EL intensity and individual I-V features. This is done using the “segmented” library in R.



The x-axis represents each I-V feature, and the slope of the blue line shows how much influence this I-V variable has on the response variable, which is intensity. In general, the I-V features do not have much effect on the intensity until after a certain point towards the end. This explains why the linear regress was not a good model because these scattered data points would create a large variance.

To minimize this problem, I decided to convert intensity into different levels for classification instead of regression.

4.3 Classfication Using Only IV Features to Predict EL Intensity

4.3.1 Linear Regression - Baseline

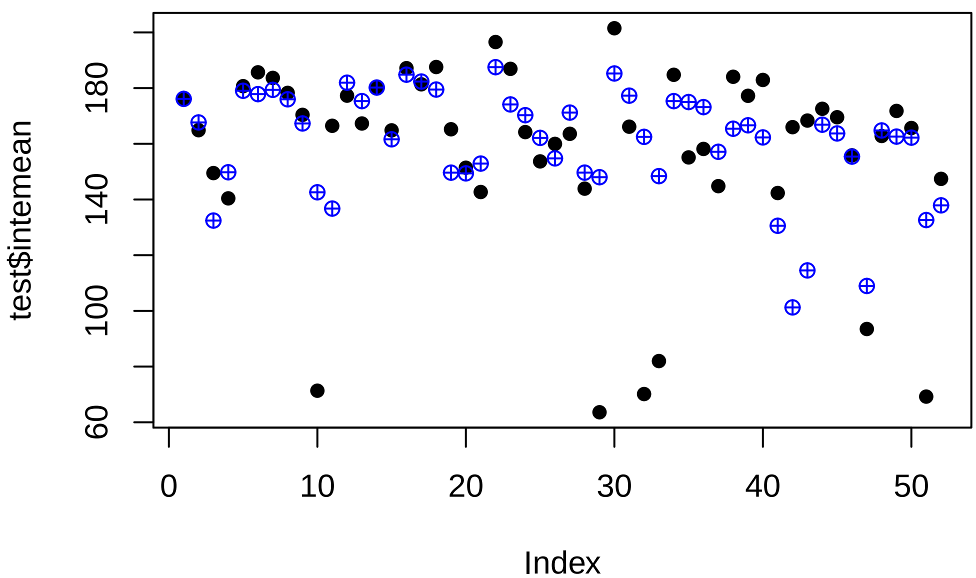
Before I dive into the classification models, I established a baseline using the linear model. This time, I excluded the exposure time and other EL feature, and tried to use only the I-V features to predict EL intensity.

```
##
## Call:
## lm(formula = intemean ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -91.232 -10.812   3.566  16.939  46.695
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   231.95      75.70   3.064  0.00325 **
## epmp          514.90     356.13   1.446  0.15334
## eimp         -889.40     470.48  -1.890  0.06345 .
```

```
## evmp         -426.56     257.37  -1.657  0.10257
## exff          360.15     312.18   1.154  0.25315
## eisc          304.11     199.93   1.521  0.13341
## evoc           93.39      34.91   2.675  0.00957 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.87 on 61 degrees of freedom
## Multiple R-squared:  0.4011, Adjusted R-squared:  0.3422
## F-statistic:  6.81 on 6 and 61 DF,  p-value: 1.464e-05
```

```
## [1] "Prediction Accuracy: 0.54"
```

```
## [1] "RMSE: 28.21"
```



The baseline prediction accuracy is 0.54, with an RMSE of 28.21. This performs much worse then the previous model where I used the entire data set.

4.3.2 Category Transformation

Median intensity was normalized between 0~255 when I extracted the data from Python. There are arbitrarily divided into the following levels. - Level 0: [0,64) - Level 1: [64,128) - Level 2: [128,192) - Level 3: [192,256)

4.3.3 Classification Models and Parameter Tuning with Cross-Validation

4.3.3.1 Random Forest

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:gridExtra':
##
##     combine
## The following object is masked from 'package:dplyr':
##
##     combine
## The following object is masked from 'package:ggplot2':
##
##     margin
## [[1]]
## [1] "Random Forest"
##
## [[2]]
##  Accuracy
## 0.7884615
##
## [[3]]
##           Reference
## Prediction  0  1  2  3
##          0  0  0  2  0
##          1  0  1  2  0
##          2  1  4 40  2
##          3  0  0  0  0
```

4.3.3.2 SVM

```
## [[1]]
## [1] "SVM"
##
## [[2]]
##  Accuracy
## 0.8461538
##
## [[3]]
##           Reference
## Prediction  0  1  2  3
##          0  0  0  0  0
##          1  0  0  0  0
##          2  1  5 44  2
##          3  0  0  0  0
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  gamma cost
##  1e-04  0.1
##
```

```
## - best performance: 0.1880952
##
## - Detailed performance results:
##   gamma cost     error dispersion
## 1 1e-04  0.1 0.1880952    0.22348
## 2 1e-03  0.1 0.1880952    0.22348
## 3 1e-02  0.1 0.1880952    0.22348
## 4 1e-04  1.0 0.1880952    0.22348
## 5 1e-03  1.0 0.1880952    0.22348
## 6 1e-02  1.0 0.1880952    0.22348
## 7 1e-04 10.0 0.1880952    0.22348
## 8 1e-03 10.0 0.1880952    0.22348
## 9 1e-02 10.0 0.1880952    0.22348
```

```
## [[1]]
## [1] "Tuned SVM"
##
## [[2]]
##  Accuracy
## 0.8461538
##
## [[3]]
##           Reference
## Prediction  0  1  2  3
##          0  0  0  0  0
##          1  0  0  0  0
##          2  1  5 44  2
##          3  0  0  0  0
```

Using random forest, I got an accuracy of 0.79. SVM worked better with a higher accuracy of 0.85. All the level 2 cases were identified correctly. I also performed parameter tuning and cross validation, but did not improve the model very well.

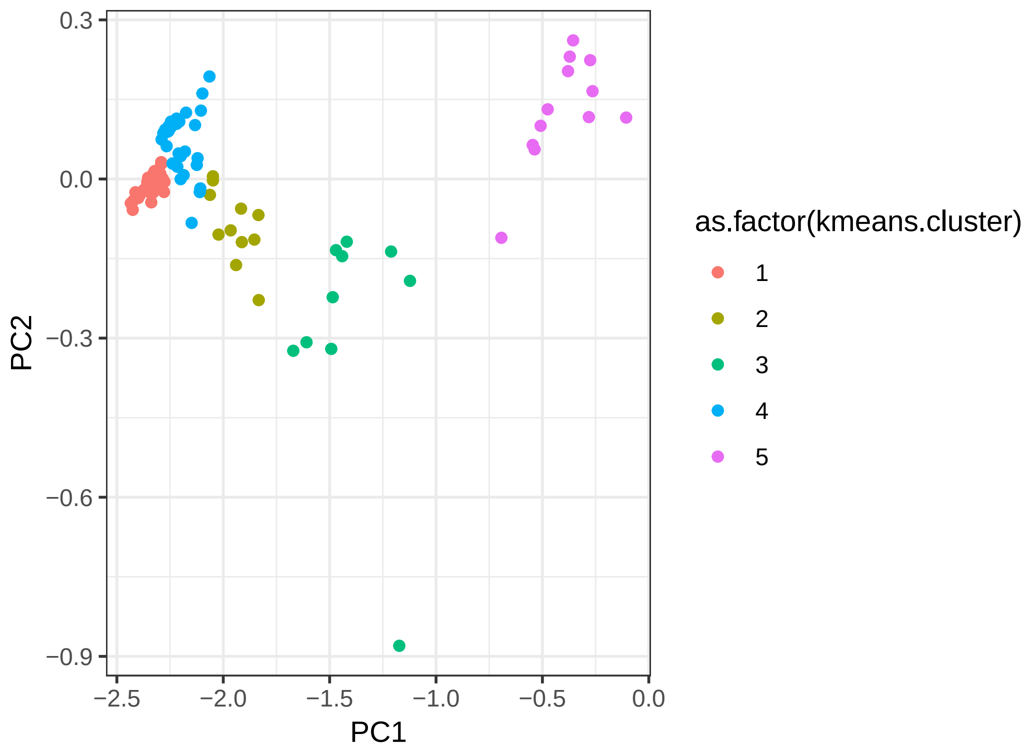
4.3.4 Unsupervised Clustering

Since classification models using arbitrary categorization method did not work very well for the Level 1 and 3 groups, I tried to use unsupervised clustering to help divide the EL intensity into different categories.

4.3.4.1 K-means

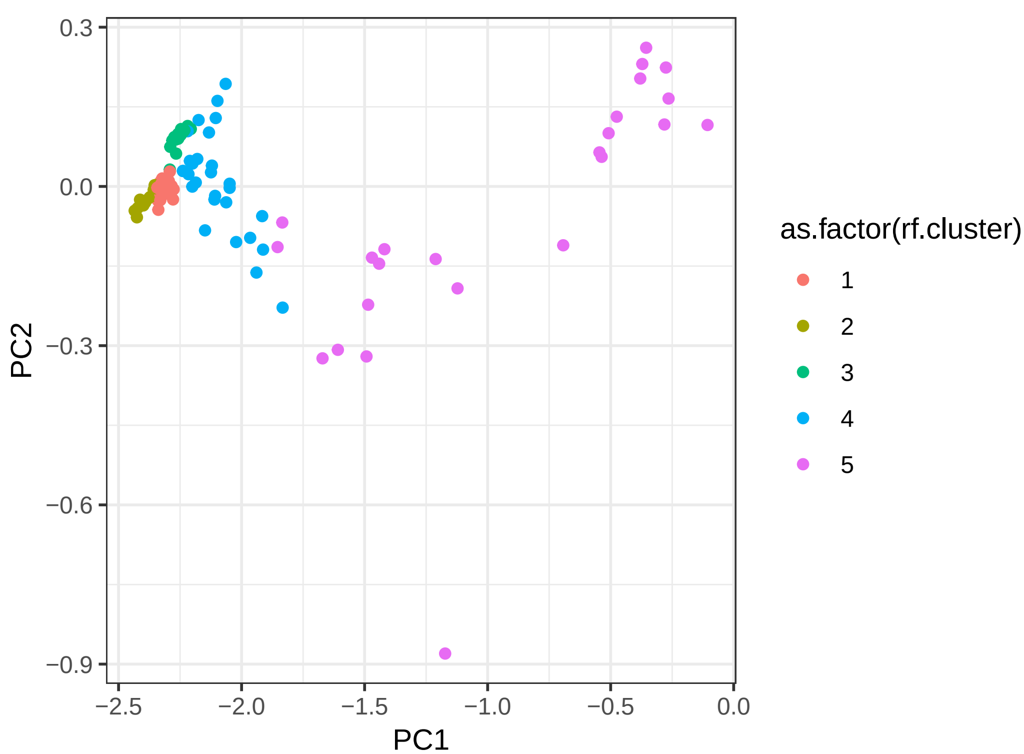
```
## Warning: package 'metricsgraphics' was built under R version 3.5.3
```

```
##  1  2  3  4  5
## 48 11 10 39 12
```



4.3.4.2 Random Forest

```
## 1 2 3 4 5
## 28 19 21 28 24
```



4.3.5 Re-training Using the Clustering Categories

```
## [[1]]
## [1] "Random Forest with k-means clustering"
##
## [[2]]
##  Accuracy
## 0.9807692
##
## [[3]]
##           Reference
## Prediction  1  2  3  4  5
##          1 22  0  0  1  0
##          2  0  3  0  0  0
##          3  0  0  4  0  0
##          4  0  0  0 18  0
##          5  0  0  0  0  4
```

```
## [[1]]
## [1] "Random Forest with random forest clustering"
##
## [[2]]
##  Accuracy
## 0.9615385
##
## [[3]]
##           Reference
## Prediction  1  2  3  4  5
##          1 11  0  0  0  0
##          2  0 10  0  0  0
##          3  0  0  8  0  0
##          4  1  0  0 13  0
##          5  0  0  0  1  8
```

Both k-means clustering and random forest clustering improved the model accuracy, to around 90%.

5 Conclusion and Future Study

Overall, It is very difficult to use only I-V features to predict EL intensity values. The linear regression model indicated non-linearly and deviation from normality. In comparison, classification model served the prediction purpose relatively well. Random forest classification model performed pretty well on the data. The accuracy reached 90% with the categories determined by clustering. However, since clustering is unsupervised, it would be hard to justify the reason for such type of categorization using PV-module theory. Further study can be done to solve this problem.

6 References:

[1]https://www.solarpowerauthority.com/history-of-solar-power-technology]