Quantifying the World

Case Study 1

Presented by:

Alexander Sepenu

Nnenna Okpara

Taifur Chowdhury

Edgar Nunez - Gonzalez

Business Understanding

Objective: The objective of this case study is to identify the most influential variables on our linear regression model designed to predict the critical temperatures at which different material compositions become superconductors.

Problem Statement: We have two datasets regarding a study of the critical temperatures of superconductors. The first file contains all the variations of the measures for critical temperatures of superconductors. The second file contains the chemical composition associated with the critical temperature of the superconductors. Given this information, design a model to help us determine whether the critical temperature of a material composition yields a superconductor and the critical temperature at which the material compositions in the study become superconductors.

Data Evaluation

In [ ]:

```
## Importing libraries
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
```

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html)

In [ ]:

```
## Reading CSV files
train = pd.read_csv('train.csv')
unique = pd.read_csv('unique_m.csv')
```

In [ ]:

```
# Checking the shape of the file read
train.shape
```

In [ ]:

```
# Checking the shape of the file read
unique.shape
```

In [ ]:

```
# Removing last columns - "critical temperature" and combining both tables to create a data
train_2 = train.iloc[:, :-1]
unique_2 = unique.iloc[:, :-1]
df_main = pd.concat([train_2, unique_2], axis=1, join="inner")
```

In [ ]:

```
#Dataframe dimension should equal (88+81) 169 columns and 21,263 rows
df_main.shape
```

In [ ]:

```
#Checking the statistical features of the new dataframe
df_main.describe(include='all')
```

In [ ]:

```
# Code to remove outliers
#df_main = df_main[(df_main.critical_temp >0.0125) & (df_main.critical_temp <133.6)]
```

In [ ]:

```
df_main.describe()
```

In [ ]:

```
# Converting all cloumns to float to normalize data for ease of use
df_main = df_main[:].astype('float64')
```

In [ ]:

```
 df_main.dtypes
```

In [ ]:

```
df_main.head(10)
```

In [ ]:

```
df_main.describe()
```

EDA and Visualization

In [ ]:

```
#pip install https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
```

In [ ]:

```
import numpy as np
import pandas as pd
import pandas_profiling
from pandas_profiling import ProfileReport

profile = ProfileReport(df_main, title="Profiling Report", explorative=True)
```

In [ ]:

```
# this line of code does an in depth EDA but take about 4 hours to complete depend on compu
profile
```

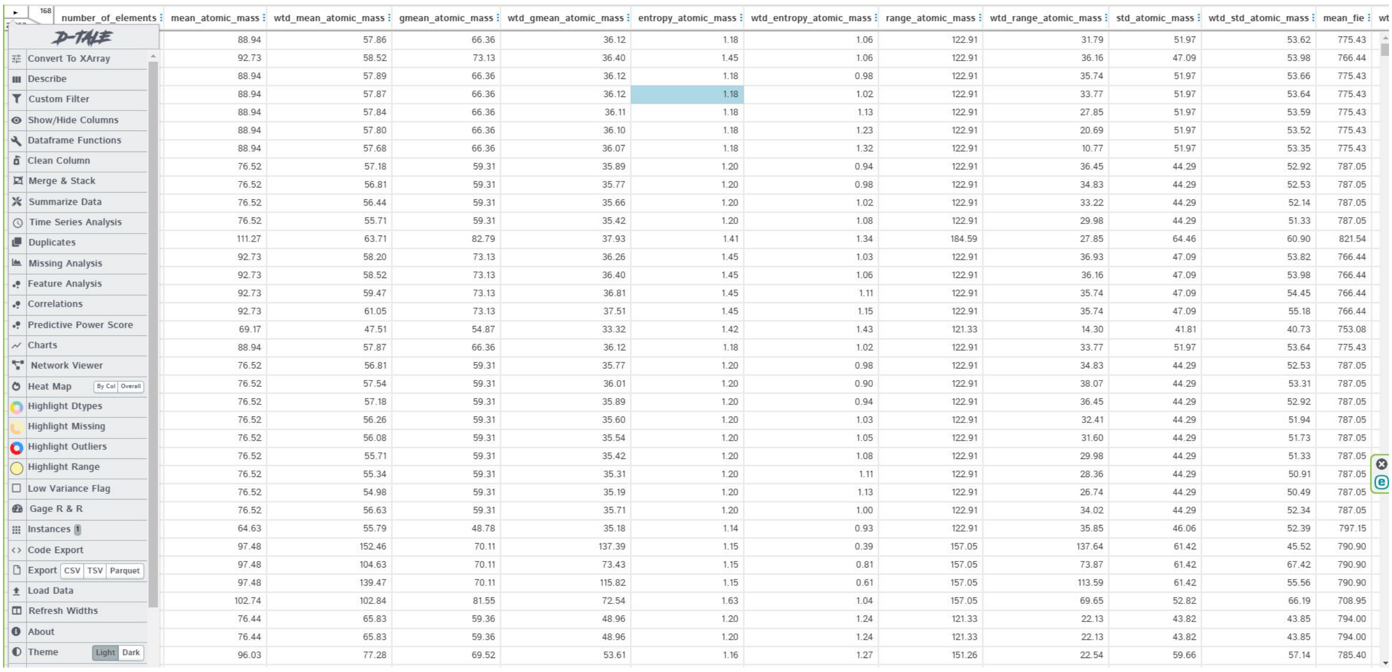
In [ ]:

```
# this line of code saves the results into your working directory which you can open in you
profile.to_file('crit_temp.html')
```

In [ ]:

```
#EDA Package to select features of relevance based on correlation with target variable
#opens in a new brower window to save the selected columns that have significance
#source: https://www.investopedia.com/terms/c/correlationcoefficient.asp#:~:text=Values%20a
import pandas as pd
import dtale
dtale.show(df_main, open_browser=True)
```

EDA Output from Dtale Package



number_of_elements	mean_atomic_mass	wtd_mean_atomic_mass	gmean_atomic_mass	wtd_gmean_atomic_mass	entropy_atomic_mass	wtd_entropy_atomic_mass	range_atomic_mass	wtd_range_atomic_mass	std_atomic_mass	wtd_std_atomic_mass	mean_fie
	88.94	57.86	66.36	36.12	1.18	1.06	122.91	31.79	51.97	53.62	775.43
	92.73	58.52	73.13	36.40	1.45	1.06	122.91	36.16	47.09	53.98	766.44
	88.94	57.89	66.36	36.12	1.18	0.98	122.91	35.74	51.97	53.66	775.43
	88.94	57.87	66.36	36.12	1.18	1.02	122.91	33.77	51.97	53.64	775.43
	88.94	57.84	66.36	36.11	1.18	1.13	122.91	27.85	51.97	53.59	775.43
	88.94	57.80	66.36	36.10	1.18	1.23	122.91	20.69	51.97	53.52	775.43
	88.94	57.68	66.36	36.07	1.18	1.32	122.91	10.77	51.97	53.35	775.43
	76.52	57.18	59.31	35.89	1.20	0.94	122.91	36.45	44.29	52.92	787.05
	76.52	56.81	59.31	35.77	1.20	0.98	122.91	34.83	44.29	52.53	787.05
	76.52	56.44	59.31	35.66	1.20	1.02	122.91	33.22	44.29	52.14	787.05
	76.52	55.71	59.31	35.42	1.20	1.08	122.91	29.98	44.29	51.33	787.05
	111.27	63.71	82.79	37.93	1.41	1.34	184.59	27.85	64.46	60.90	821.54
	92.73	58.20	73.13	36.26	1.45	1.03	122.91	36.93	47.09	53.82	766.44
	92.73	58.52	73.13	36.40	1.45	1.06	122.91	36.16	47.09	53.98	766.44
	92.73	59.47	73.13	36.81	1.45	1.11	122.91	35.74	47.09	54.45	766.44
	92.73	61.05	73.13	37.51	1.45	1.15	122.91	35.74	47.09	55.18	766.44
	69.17	47.51	54.87	33.32	1.42	1.43	121.33	14.30	41.81	40.73	753.08
	88.94	57.87	66.36	36.12	1.18	1.02	122.91	33.77	51.97	53.64	775.43
	76.52	56.81	59.31	35.77	1.20	0.98	122.91	34.83	44.29	52.53	787.05
	76.52	57.54	59.31	36.01	1.20	0.90	122.91	38.07	44.29	53.31	787.05
	76.52	57.18	59.31	35.89	1.20	0.94	122.91	36.45	44.29	52.92	787.05
	76.52	56.26	59.31	35.60	1.20	1.03	122.91	32.41	44.29	51.94	787.05
	76.52	56.08	59.31	35.54	1.20	1.05	122.91	31.60	44.29	51.73	787.05
	76.52	55.71	59.31	35.42	1.20	1.08	122.91	29.98	44.29	51.33	787.05
	76.52	55.34	59.31	35.31	1.20	1.11	122.91	28.36	44.29	50.91	787.05
	76.52	54.98	59.31	35.19	1.20	1.13	122.91	26.74	44.29	50.49	787.05
	76.52	56.63	59.31	35.71	1.20	1.00	122.91	34.02	44.29	52.34	787.05
	64.63	55.79	48.78	35.18	1.14	0.93	122.91	35.85	46.06	52.39	797.15
	97.48	152.46	70.11	137.39	1.15	0.39	157.05	137.64	61.42	45.52	790.90
	97.48	104.63	70.11	73.43	1.15	0.81	157.05	73.87	61.42	67.42	790.90
	97.48	139.47	70.11	115.82	1.15	0.61	157.05	113.59	61.42	55.56	790.90
	102.74	102.84	81.55	72.54	1.63	1.04	157.05	69.65	52.82	66.19	708.95
	76.44	65.83	59.36	48.96	1.20	1.24	121.33	22.13	43.82	43.85	794.00
	76.44	65.83	59.36	48.96	1.20	1.24	121.33	22.13	43.82	43.85	794.00
	96.03	77.28	69.52	53.61	1.16	1.27	151.26	22.54	59.66	57.14	785.40

In [ ]:

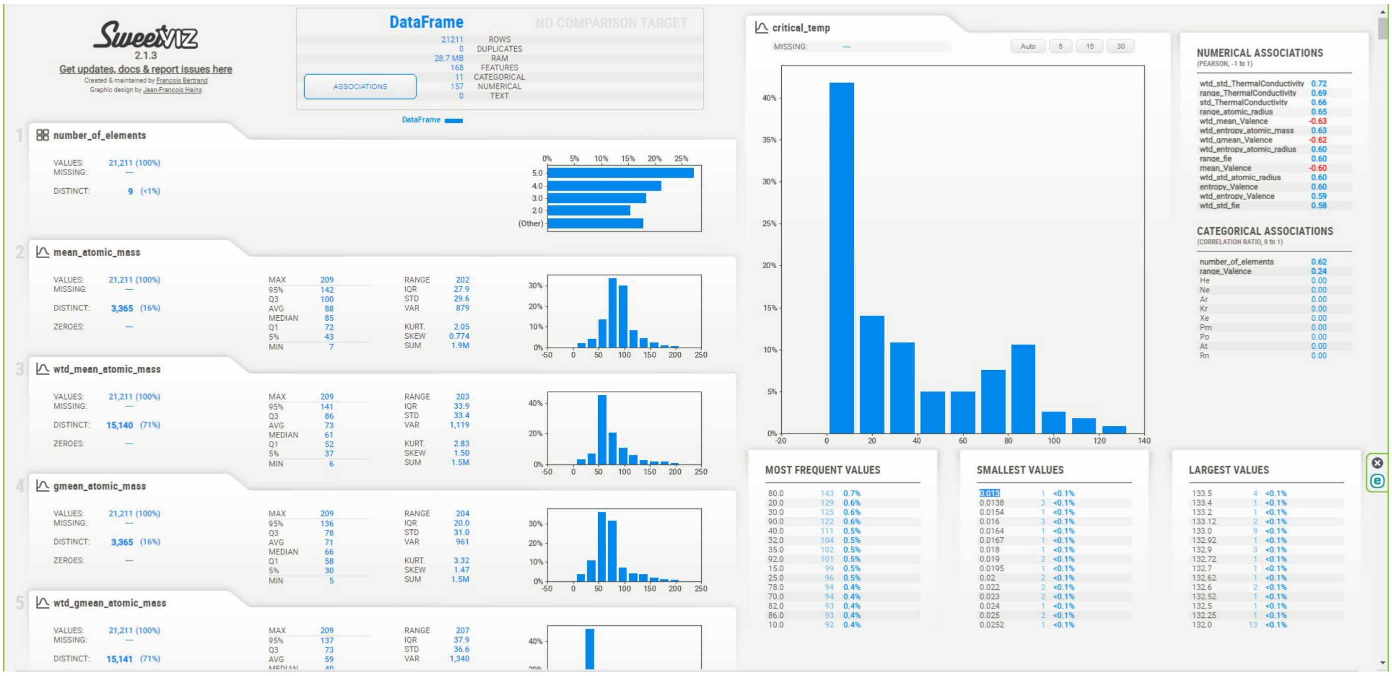
```
#display the report
import sweetviz as sv

temp_report = sv.analyze(df_main)

#display the report
temp_report.show_html('temp.html')
```

EDA Output from Sweetviz Package

Below is the EDA report of the semi conductor dataset to learn the distribution of the various features.



```
DataFrame
    21211  ROWS
        0  DUPLICATES
  28.7 MB  RAM
      168  FEATURES
       11  CATEGORICAL
      157  NUMERICAL
        0  TEXT
```

```
1  number_of_elements
   VALUES:    21,211 (100%)
   MISSING:   ---
   DISTINCT:  9 (<1%)

2  mean_atomic_mass
   VALUES:    21,211 (100%)    MAX     209    RANGE  202
   MISSING:   ---              95%     142    IQR    27.9
                               Q3      100    STD    29.6
   DISTINCT:  3,365 (16%)      AVG     88     VAR    879
                               MEDIAN  85
   ZEROES:    ---              Q1      72     KURT.  2.05
                               5%      43     SKEW   0.774
                               MIN     7      SUM    1.9M

3  wtd_mean_atomic_mass
   VALUES:    21,211 (100%)    MAX     209    RANGE  203
   MISSING:   ---              95%     141    IQR    33.9
                               Q3      86     STD    33.4
   DISTINCT:  15,140 (71%)     AVG     73     VAR    1,119
                               MEDIAN  61
   ZEROES:    ---              Q1      52     KURT.  2.83
                               5%      37     SKEW   1.50
                               MIN     6      SUM    1.5M

4  gmean_atomic_mass
   VALUES:    21,211 (100%)    MAX     209    RANGE  204
   MISSING:   ---              95%     136    IQR    20.0
                               Q3      78     STD    31.0
   DISTINCT:  3,365 (16%)      AVG     71     VAR    961
                               MEDIAN  66
   ZEROES:    ---              Q1      58     KURT.  3.32
                               5%      30     SKEW   1.47
                               MIN     5      SUM    1.5M

5  wtd_gmean_atomic_mass
   VALUES:    21,211 (100%)    MAX     209    RANGE  207
   MISSING:   ---              95%     137    IQR    37.9
                               Q3      73     STD    36.6
   DISTINCT:  15,141 (71%)     AVG     59     VAR    1,340
                               MEDIAN  40
```

critical_temp

MISSING: ---

NUMERICAL ASSOCIATIONS
(PEARSON, -1 to 1)

wtd_std_ThermalConductivity	0.72
range_ThermalConductivity	0.69
std_ThermalConductivity	0.66
range_atomic_radius	0.65
wtd_mean_Valence	-0.63
wtd_entropy_atomic_mass	0.63
wtd_gmean_Valence	-0.62
wtd_entropy_atomic_radius	0.60
range_fie	0.60
mean_Valence	-0.60
wtd_std_atomic_radius	0.60
entropy_Valence	0.60
wtd_entropy_Valence	0.59
wtd_std_fie	0.58

CATEGORICAL ASSOCIATIONS
(CORRELATION RATIO, 0 to 1)

number_of_elements	0.62
range_Valence	0.24
He	0.00
Ne	0.00
Ar	0.00
Kr	0.00
Xe	0.00
Pm	0.00
Po	0.00
At	0.00
Rn	0.00

MOST FREQUENT VALUES

80.0	143	0.7%
20.0	129	0.6%
30.0	125	0.6%
90.0	122	0.6%
40.0	111	0.5%
32.0	104	0.5%
35.0	102	0.5%
92.0	101	0.5%
15.0	99	0.5%
25.0	96	0.5%
78.0	94	0.4%
70.0	94	0.4%
82.0	93	0.4%
86.0	93	0.4%
10.0	92	0.4%

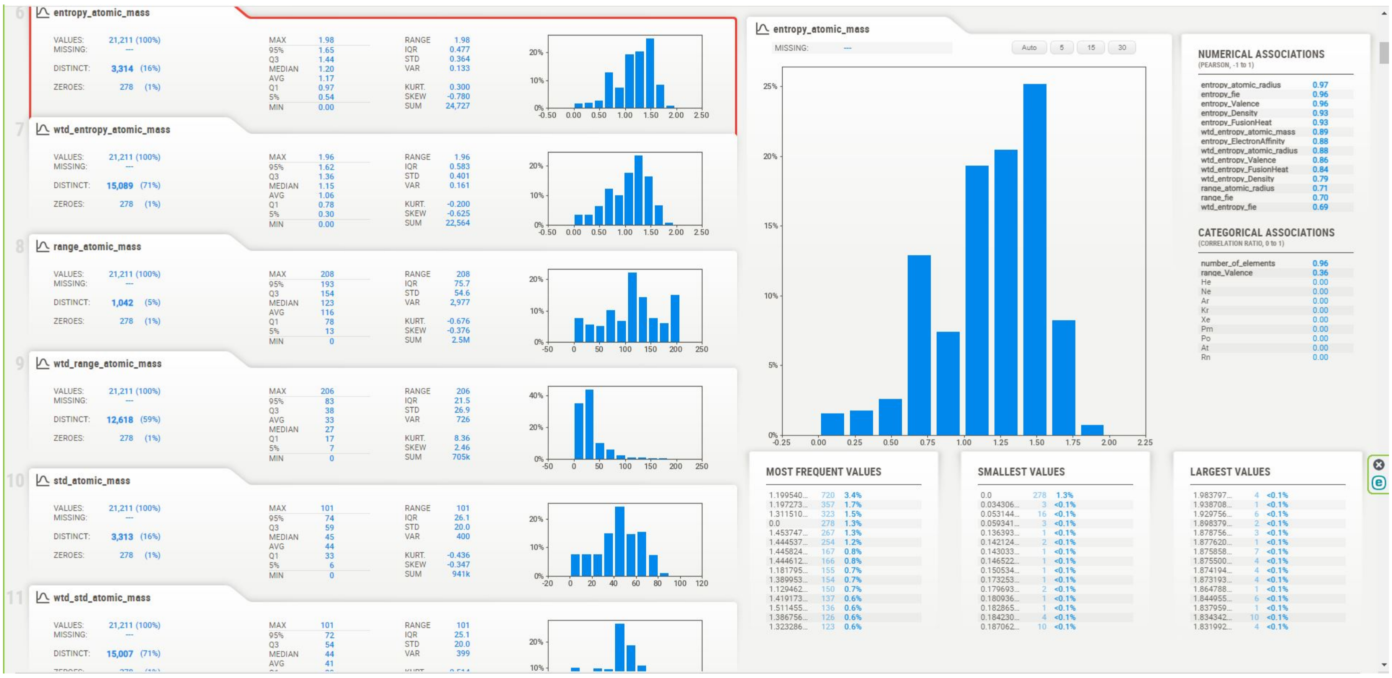
SMALLEST VALUES

0.013	1	<0.1%
0.0138	3	<0.1%
0.0154	1	<0.1%
0.016	3	<0.1%
0.0164	1	<0.1%
0.0167	1	<0.1%
0.018	1	<0.1%
0.019	2	<0.1%
0.0195	1	<0.1%
0.02	2	<0.1%
0.022	2	<0.1%
0.023	2	<0.1%
0.024	1	<0.1%
0.025	2	<0.1%
0.0252	1	<0.1%

LARGEST VALUES

133.5	4	<0.1%
133.4	1	<0.1%
133.2	1	<0.1%
133.12	2	<0.1%
133.0	9	<0.1%
132.92	1	<0.1%
132.9	3	<0.1%
132.72	1	<0.1%
132.7	1	<0.1%
132.62	1	<0.1%
132.6	2	<0.1%
132.52	1	<0.1%
132.5	1	<0.1%
132.25	1	<0.1%
132.0	13	<0.1%

EDA Output from Sweetviz Package Cont'd



```
6  entropy_atomic_mass
VALUES:    21,211 (100%)   MAX     1.98   RANGE  1.98
MISSING:   ---             95%     1.65   IQR    0.477
                           Q3      1.44   STD    0.364
DISTINCT:  3,314  (16%)    MEDIAN  1.20   VAR    0.133
                           AVG     1.17
ZEROES:    278    (1%)     Q1      0.97   KURT.  0.300
                           5%      0.54   SKEW   -0.780
                           MIN     0.00   SUM    24,727

7  wtd_entropy_atomic_mass
VALUES:    21,211 (100%)   MAX     1.96   RANGE  1.96
MISSING:   ---             95%     1.62   IQR    0.583
                           Q3      1.36   STD    0.401
DISTINCT:  15,089 (71%)    MEDIAN  1.15   VAR    0.161
                           AVG     1.06
ZEROES:    278    (1%)     Q1      0.78   KURT.  -0.200
                           5%      0.30   SKEW   -0.625
                           MIN     0.00   SUM    22,564

8  range_atomic_mass
VALUES:    21,211 (100%)   MAX     208    RANGE  208
MISSING:   ---             95%     193    IQR    75.7
                           Q3      154    STD    54.6
DISTINCT:  1,042  (5%)     MEDIAN  123    VAR    2,977
                           AVG     116
ZEROES:    278    (1%)     Q1      78     KURT.  -0.676
                           5%      13     SKEW   -0.376
                           MIN     0      SUM    2.5M

9  wtd_range_atomic_mass
VALUES:    21,211 (100%)   MAX     206    RANGE  206
MISSING:   ---             95%     83     IQR    21.5
                           Q3      38     STD    26.9
DISTINCT:  12,618 (59%)    AVG     33     VAR    726
                           MEDIAN  27
ZEROES:    278    (1%)     Q1      17     KURT.  8.36
                           5%      7      SKEW   2.46
                           MIN     0      SUM    705k

10 std_atomic_mass
VALUES:    21,211 (100%)   MAX     101    RANGE  101
MISSING:   ---             95%     74     IQR    26.1
                           Q3      59     STD    20.0
DISTINCT:  3,313  (16%)    MEDIAN  45     VAR    400
                           AVG     44
ZEROES:    278    (1%)     Q1      33     KURT.  -0.436
                           5%      6      SKEW   -0.347
                           MIN     0      SUM    941k

11 wtd_std_atomic_mass
VALUES:    21,211 (100%)   MAX     101    RANGE  101
MISSING:   ---             95%     72     IQR    25.1
                           Q3      54     STD    20.0
DISTINCT:  15,007 (71%)    MEDIAN  44     VAR    399
                           AVG     41
```

entropy_atomic_mass

MISSING: ---

NUMERICAL ASSOCIATIONS
(PEARSON, -1 to 1)

entropy_atomic_radius	0.97
entropy_fie	0.96
entropy_Valence	0.96
entropy_Density	0.93
entropy_FusionHeat	0.93
wtd_entropy_atomic_mass	0.89
entropy_ElectronAffinity	0.88
wtd_entropy_atomic_radius	0.88
wtd_entropy_Valence	0.86
wtd_entropy_FusionHeat	0.84
wtd_entropy_Density	0.79
range_atomic_radius	0.71
range_fie	0.70
wtd_entropy_fie	0.69

CATEGORICAL ASSOCIATIONS
(CORRELATION RATIO, 0 to 1)

number_of_elements	0.96
range_Valence	0.36
He	0.00
Ne	0.00
Ar	0.00
Kr	0.00
Xe	0.00
Pm	0.00
Po	0.00
At	0.00
Rn	0.00

MOST FREQUENT VALUES

1.199540...	720	3.4%
1.197273...	357	1.7%
1.311510...	323	1.5%
0.0	278	1.3%
1.453747...	267	1.3%
1.444537...	254	1.2%
1.445824...	167	0.8%
1.444612...	166	0.8%
1.181795...	155	0.7%
1.389953...	154	0.7%
1.129462...	150	0.7%
1.419173...	137	0.6%
1.511455...	136	0.6%
1.386756...	126	0.6%
1.323286...	123	0.6%

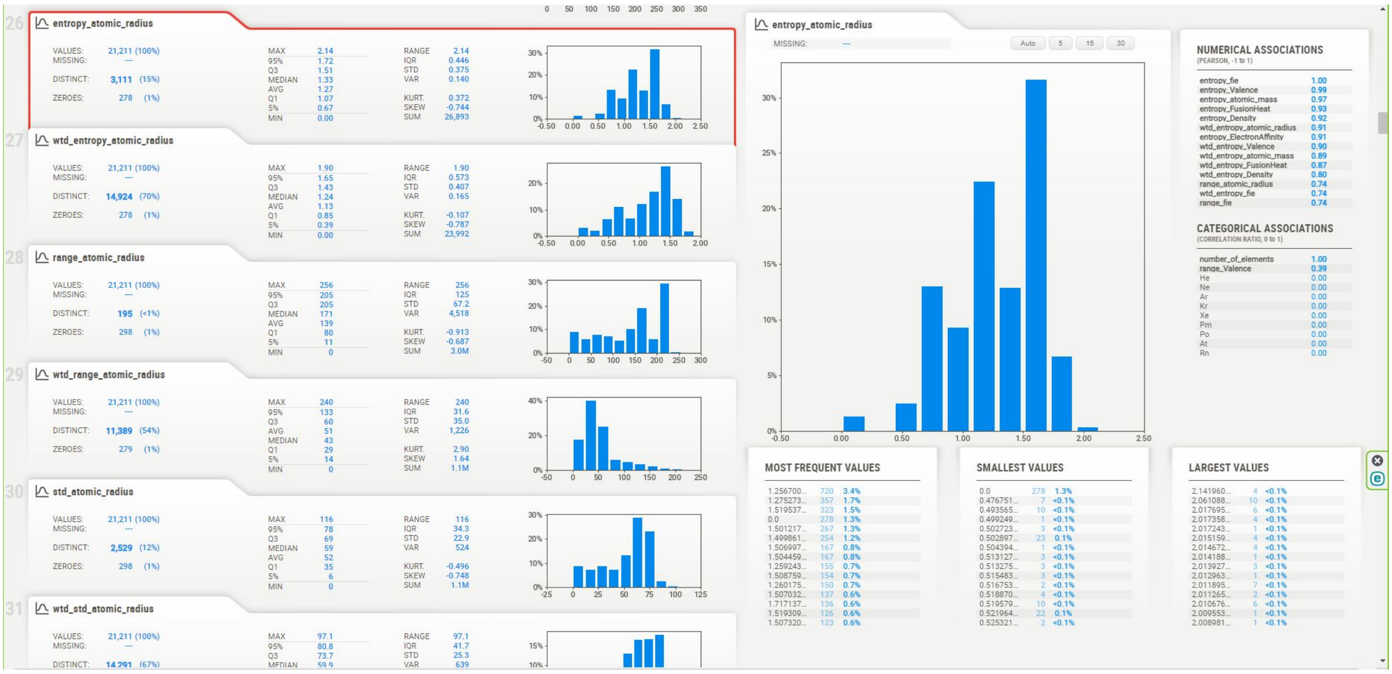
SMALLEST VALUES

0.0	278	1.3%
0.034306...	3	<0.1%
0.053144...	16	<0.1%
0.059341...	3	<0.1%
0.136393...	1	<0.1%
0.142124...	2	<0.1%
0.143033...	1	<0.1%
0.146522...	1	<0.1%
0.150534...	1	<0.1%
0.173253...	1	<0.1%
0.179693...	2	<0.1%
0.180936...	1	<0.1%
0.182865...	1	<0.1%
0.184230...	4	<0.1%
0.187062...	10	<0.1%

LARGEST VALUES

1.983797...	4	<0.1%
1.938708...	1	<0.1%
1.929756...	6	<0.1%
1.898379...	2	<0.1%
1.878756...	3	<0.1%
1.877620...	1	<0.1%
1.875858...	7	<0.1%
1.875500...	4	<0.1%
1.874194...	4	<0.1%
1.873193...	4	<0.1%
1.864788...	1	<0.1%
1.844955...	6	<0.1%
1.837959...	1	<0.1%
1.834342...	10	<0.1%
1.831992...	4	<0.1%

EDA Output from Sweetviz Package Cont'd



```
26 entropy_atomic_radius
VALUES:    21,211 (100%)   MAX     2.14   RANGE  2.14
MISSING:   ---             95%     1.72   IQR    0.446
                           Q3      1.51   STD    0.375
DISTINCT:  3,111  (15%)    MEDIAN  1.33   VAR    0.140
                           AVG     1.27
ZEROES:    278    (1%)     Q1      1.07   KURT.  0.372
                           5%      0.67   SKEW   -0.744
                           MIN     0.00   SUM    26,893

27 wtd_entropy_atomic_radius
VALUES:    21,211 (100%)   MAX     1.90   RANGE  1.90
MISSING:   ---             95%     1.65   IQR    0.573
                           Q3      1.43   STD    0.407
DISTINCT:  14,924 (70%)    MEDIAN  1.24   VAR    0.165
                           AVG     1.13
ZEROES:    278    (1%)     Q1      0.85   KURT.  -0.107
                           5%      0.39   SKEW   -0.787
                           MIN     0.00   SUM    23,992

28 range_atomic_radius
VALUES:    21,211 (100%)   MAX     256    RANGE  256
MISSING:   ---             95%     205    IQR    125
                           Q3      205    STD    67.2
DISTINCT:  195    (<1%)    MEDIAN  171    VAR    4,518
                           AVG     139
ZEROES:    298    (1%)     Q1      80     KURT.  -0.913
                           5%      11     SKEW   -0.687
                           MIN     0      SUM    3.0M

29 wtd_range_atomic_radius
VALUES:    21,211 (100%)   MAX     240    RANGE  240
MISSING:   ---             95%     133    IQR    31.6
                           Q3      60     STD    35.0
DISTINCT:  11,389 (54%)    AVG     51     VAR    1,226
                           MEDIAN  43
ZEROES:    279    (1%)     Q1      29     KURT.  2.90
                           5%      14     SKEW   1.64
                           MIN     0      SUM    1.1M

30 std_atomic_radius
VALUES:    21,211 (100%)   MAX     116    RANGE  116
MISSING:   ---             95%     78     IQR    34.3
                           Q3      69     STD    22.9
DISTINCT:  2,529  (12%)    MEDIAN  59     VAR    524
                           AVG     52
ZEROES:    298    (1%)     Q1      35     KURT.  -0.496
                           5%      6      SKEW   -0.748
                           MIN     0      SUM    1.1M

31 wtd_std_atomic_radius
VALUES:    21,211 (100%)   MAX     97.1   RANGE  97.1
MISSING:   ---             95%     80.8   IQR    41.7
                           Q3      73.7   STD    25.3
DISTINCT:  14,291 (67%)    MEDIAN  59.9   VAR    639
```

entropy_atomic_radius

MISSING: ---

NUMERICAL ASSOCIATIONS
(PEARSON, -1 to 1)

entropy_fie	1.00
entropy_Valence	0.99
entropy_atomic_mass	0.97
entropy_FusionHeat	0.93
entropy_Density	0.92
wtd_entropy_atomic_radius	0.91
entropy_ElectronAffinity	0.91
wtd_entropy_Valence	0.90
wtd_entropy_atomic_mass	0.89
wtd_entropy_FusionHeat	0.87
wtd_entropy_Density	0.80
range_atomic_radius	0.74
wtd_entropy_fie	0.74
range_fie	0.74

CATEGORICAL ASSOCIATIONS
(CORRELATION RATIO, 0 to 1)

number_of_elements	1.00
range_Valence	0.39
He	0.00
Ne	0.00
Ar	0.00
Kr	0.00
Xe	0.00
Pm	0.00
Po	0.00
At	0.00
Rn	0.00

MOST FREQUENT VALUES

1.256700...	720	3.4%
1.275273...	357	1.7%
1.519537...	323	1.5%
0.0	278	1.3%
1.501217...	267	1.3%
1.499861...	254	1.2%
1.506997...	167	0.8%
1.504459...	167	0.8%
1.259243...	155	0.7%
1.508759...	154	0.7%
1.260175...	150	0.7%
1.507032...	137	0.6%
1.717137...	136	0.6%
1.519309...	126	0.6%
1.507320...	123	0.6%

SMALLEST VALUES

0.0	278	1.3%
0.476751...	7	<0.1%
0.493565...	10	<0.1%
0.499249...	1	<0.1%
0.502723...	3	<0.1%
0.502897...	23	0.1%
0.504394...	1	<0.1%
0.513127...	3	<0.1%
0.513275...	3	<0.1%
0.515483...	3	<0.1%
0.516753...	2	<0.1%
0.518870...	4	<0.1%
0.519579...	10	<0.1%
0.521964...	22	0.1%
0.525321...	2	<0.1%

LARGEST VALUES

2.141960...	4	<0.1%
2.061088...	10	<0.1%
2.017695...	6	<0.1%
2.017358...	4	<0.1%
2.017243...	1	<0.1%
2.015159...	4	<0.1%
2.014672...	4	<0.1%
2.014188...	1	<0.1%
2.013927...	3	<0.1%
2.012963...	1	<0.1%
2.011895...	7	<0.1%
2.011265...	2	<0.1%
2.010676...	6	<0.1%
2.009553...	1	<0.1%
2.008981...	1	<0.1%

Since we have been provided with two datasets, we combined both dataset so then we can preprocess our main dataframe for the model. We have 168 columns and 21,263 rows for our mainframe. We do not have any missing values for our dataframe. For our exploratory data analysis, we have used the packages called ‘profile’, 'sweetviz' and 'dtale'. It was able to tell us how our data was distributed. From the EDA the dataset was heavily unbalanced hence would require some transformations for model building. The package has provided us the data distribution report for all the features.

Dtale package was use to select features of relevance based on their correlation with Critical_temp(target) between 1 and 0.8. This removed the cloumn that are highly colinear with out target variable. We have gathered a list of important features in a document which lists the important features for our model building. 59 features out of 168 were selected and are listed in the dtale report.

Model Preparation

To solve the problem, we have used linear regression to predict the critical temperature using those important features that are described above. This method will have use to tackle our objective of predicting the temperature values and finding out how the important features have contributed to our prediction. The following metrics were used to evaluate our models (L1 and L2) of which the best model was chosen.

Evaluation Metrics:

Root Mean Squared Error (RMSE): measures the standard deviation between residuals which are the variance between actual values and predicted values. It measures the distance of how far the actual values are from the values in regression line. The formula first divides the variance squared by number of values in the dataset (n) and then takes the squared root of the value. It is widely used for linear forecasting and regression; we will use this as primary metrics for model evaluation.

Source : https://towardsdatascience.com/what-does-rmse-really-mean-806b65f2e48e (https://towardsdatascience.com/what-does-rmse-really-mean-806b65f2e48e)

R-Squared: It is a statistical measurement of the proportion of the variance for the dependent variable that is explained by an independent variable. In other words, it tells us how well the dependent variable has been explained by the independent variables. The measurement is varied from 0% to 100% - 0% being the model explains none of the variability of the response variable around the mean, and 100% being the model explains all the variability of the response variable around the mean. It has some limitations it cannot determine whether the coefficient estimates and predictions are biased. The Higher the R-squared, the better the model.

Source: https://blog.minitab.com/en/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit (https://blog.minitab.com/en/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit)

MAE measures the prediction error. It would be used to calculate the absolute difference between the actual and predicted values. MAE does not penalize large errors. MAE is less sensitive to outliers compared to RMSE. The MAE can be calculated as follows: • MAE = 1 / N * sum for i to N abs(y_i – yhat_i) Where y_i is the i'th expected value in the dataset, yhat_i is the i'th predicted value and abs()is the absolute function.

MSE is the average squared difference between the observed actual outcome values and the values predicted by the model. The MSE is calculated as the mean or average of the squared differences between predicted and expected target values in a dataset. • MSE = 1 / N * sum for i to N (y_i – yhat_i)^2 Where y_i is the i'th expected value in the dataset and yhat_i is the i'th predicted value. The difference between these two values is squared, which has the effect of removing the sign, resulting in a positive error value.

AIC works by evaluating the model's fit on the training data and adding a penalty term for the complexity of the model. The desired result is to find the lowest possible AIC, which indicates the best balance of model fit with generalizability. This serves the eventual goal of maximizing fit on out-of-sample data. The lower the AIC, the better the model.

Sources:

http://www.sthda.com/english/articles/38-regression-model-validation/158-regression-model-accuracy-metrics-r-square-aic-bic-cp-and-more/ (http://www.sthda.com/english/articles/38-regression-model-validation/158-regression-model-accuracy-metrics-r-square-aic-bic-cp-and-more/)

https://towardsdatascience.com/evaluation-metrics-model-selection-in-linear-regression-73c7573208be (https://towardsdatascience.com/evaluation-metrics-model-selection-in-linear-regression-73c7573208be) https://machinelearningmastery.com/regression-metrics-for-machine-learning/ (https://machinelearningmastery.com/regression-metrics-for-machine-learning/)

In [ ]:

```
############################################
###### Import required Python packages #####
############################################

import os
import pandas as pd
import re
import datetime as dt
from datetime import timezone
import numpy as np
from ipywidgets import interact, interactive, IntSlider, Layout
import ipywidgets as widgets
from IPython.display import display
import h2o
from h2o.automl import H2OAutoML
from sklearn.model_selection import train_test_split
import io
import requests
import subprocess
import json
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:

```
from IPython.display import HTML

HTML('''<script>
code_show=true;
function code_toggle() {
 if (code_show){
 $('div.input').hide();
 } else {
 $('div.input').show();
 }
 code_show = !code_show
}
$( document ).ready(code_toggle);
</script>
The raw code for this IPython notebook is by default hidden for easier reading.
To toggle on/off the raw code, click <a href="javascript:code_toggle()">here</a>.''')
```

Type *Markdown* and LaTeX: α^2

In [ ]:

```
set_working_directory = widgets.Text(
    value=os.getcwd(),
    placeholder='C:/Users/korku/Documents/MY SMU COURSES/Semester 2/Spring 2022/Quantifying
    description='Directory:',
    disabled=False,
    layout=Layout(width='100%')
)

display(set_working_directory)
```

In [ ]:

```
try:
    os.chdir(set_working_directory.value)
    print('Changed directory to {}'.format(set_working_directory.value))
except Exception as e:
    print('Failed to change directory')
    print(e)
```

In [ ]:

```
##########################################
##### Data Ingestion Functions
##########################################

def compile_raw_data(filename, tab_names, subfolder, delimiter_char = ',', skip_rows = 0, f

    # Inputs:
    ## filename = 'sample.csv' | 'sample.xlsx' - the filename in the directory (including t
    ## tab_names = None | ['Sheet1,'Sheet2'] - None for csv; [comma separated list of tab n
    ## subfolder = 'source_data' - string containing the name of a folder in the working di
    ## delimiter_char = ',' | ';' - None for xlsx
    ## rows to skip = default 0 - Not used for csv; trims the user-defined number of rows f
    ## file extension = csv | xlsx

    # Description: reads in the workbook; standardizes header names;
    # Outputs: returns a dictionary of dataframes

    master_data = {}
    if subfolder:
        file_path = subfolder+'/{}'.format(filename)
    else:
        file_path = filename

    if file_ext == 'csv':
        tab_names = [re.sub('.csv','', filename)]

    for tab in tab_names:
        try:
            if file_ext == 'xlsx':
                dframe = pd.read_excel(file_path, tab, skip_rows)
            elif file_ext == 'csv' and delimiter_char == ',':
                dframe = pd.read_csv(file_path, header=0, delimiter=',')
            else:
                dframe = pd.read_csv(file_path, header=0, delimiter=';')

            sanitizer = {
                        '$':'USD',
                        '(':' ',
                        ')':' ',
                        '/':' ',
                        '-':' ',
                        ',':' ',
                        '.':' '
            }

            for key, value in sanitizer.items():
                dframe.rename(columns=lambda x: x.replace(key, value), inplace=True)

            dframe.rename(columns=lambda x: x.strip(), inplace=True)
            dframe.rename(columns=lambda x: re.sub(' +','_', x), inplace=True)

            dframe.columns = map(str.lower, dframe.columns)

            master_data.update({tab:dframe})
        except Exception as e:
            master_data.update({tab:'Failed'})

    return master_data
```

In [ ]:

```
upload_type = widgets.RadioButtons(
    options=['local', 'url'],
    description='File Location:',
    disabled=False
)

upload_url = widgets.Text(
    value='scaled_down_df.csv',
    placeholder='http://',
    description='URL:',
    disabled=False,
    layout=Layout(width='80%')
)
upload_filename = widgets.Text(
    value='scaled_down_df.csv',
    placeholder='Sample File.csv',
    description='File Name:',
    disabled=False,
    layout=Layout(width='50%')
)

file_type = widgets.RadioButtons(
    options=['csv', 'xlsx'],
    description='File Type:',
    disabled=False
)

tab_names = widgets.Text(
    value='Sheet1, Sheet2, Sheet3, etc',
    placeholder='ALL EMPLOYEES, PAST EMPLOYEES',
    description='Tab(s):',
    disabled=False,
    layout=Layout(width='50%')
)

subfolder_name = widgets.Text(
    value='source_data',
    placeholder='Subfolder name',
    description='Subfolder:',
    disabled=False,
    layout=Layout(width='50%')
)

subfolder = widgets.RadioButtons(
    options=['no','yes'],
    value='no',
    description='Subfolder:',
    disabled=False
)

skip_rows = widgets.IntSlider(
    value=0,
    min=0,
    max=10,
    step=1,
    description='Skip Rows:',
    disabled=False,
    continuous_update=True,
    orientation='horizontal',
    readout=True,
    readout_format='d'
)

delimiter = widgets.RadioButtons(
    options=[',',';'],
    value=',',
    description='Delimiter:',
    disabled=False
)

def text_field(x):
    if(x=='xlsx'):
        display(tab_names)
        tab_names.on_submit(tab_names)
        display(skip_rows)
    else:
        display(delimiter)
        print('Tab Names: Not needed for csv files')

def sub_folder(y):
    if(y=='yes'):
        display(subfolder_name)
        subfolder_name.on_submit(subfolder_name)
    else:
        print('Using {} folder'.format(os.getcwd()))

def file_location(z):
    if(z=='local'):
        display(upload_filename)
        i = widgets.interactive(text_field, x=file_type)
        display(i)
        p = widgets.interactive(sub_folder, y=subfolder)
        display(p)
    else:
        display(upload_url)

q = widgets.interactive(file_location, z=upload_type)

display(q)
```

In [ ]:

```
master_data = {}

if upload_type.value == 'url':
    url_response = requests.request("GET", upload_url.value)
    master_data['url_data'] = pd.read_csv(io.BytesIO(url_response.content))
else:
    if file_type.value == 'csv':
        tabs = None
        skiprows = 0
    else:
        tabs = [x.strip() for x in tab_names.value.split(',')]
        skiprows = skip_rows.value

    if subfolder.value == 'yes':
        subfolder = subfolder_name.value
    else:
        subfolder = None
    master_data = compile_raw_data(upload_filename.value, tabs, subfolder, delimiter_char =
```

In [ ]:

```
for key, value in master_data.items():
    try:
        print('{} table was ingested with {} rows and {} columns'.format(key,value.shape[0]
    except:
        print('{} table failed to load'.format(key))
```

In [ ]:

```
dict_keys = widgets.Select(
    options=master_data.keys(),
    description='Tables:',
    disabled=False,
    layout=Layout(width='50%')
)

display(dict_keys)
```

In [ ]:

```
master_data[dict_keys.value].info()
```

In [ ]:

```
review_variables = widgets.SelectMultiple(
    options=master_data[dict_keys.value].columns.tolist(),
    description='Variables:',
    disabled=False,
    layout=Layout(width='50%')
)

display(review_variables)
```

In [ ]:

```
review_var_list = []
for i in review_variables.value:
    review_var_list.append(i)

master_data['custom_table'] = master_data[dict_keys.value][review_var_list]

head_number = widgets.BoundedIntText(
    value=5,
    min=1,
    max=50,
    step=1,
    description='Rows:',
    disabled=False
)

def sample_view(head_number):
    sample = master_data['custom_table'].head(head_number)
    print(sample)

out = widgets.interactive_output(sample_view, {'head_number':head_number})

widgets.VBox([widgets.VBox([head_number]), out])
```

In [ ]:

```
target = widgets.Select(
    options=master_data['custom_table'].columns.tolist(),
    description='Target',
    disabled=False
)

target_type = widgets.Select(
    options=['Continuous','Categorical'],
    description='Type',
    disabled=False,
)

display(target)
display(target_type)
```

In [ ]:

```
try:
    h2o.cluster().shutdown()
    h2o.init()
except:
    h2o.init()
```

In [ ]:

```
model_df = master_data['custom_table'].dropna(subset=[target.value])
model_df = h2o.H2OFrame(model_df)
```

In [ ]:

```
model_df.describe()
```

In [ ]:

```
if target_type.value == 'Categorical':
    model_df[target.value] = model_df[target.value].asfactor()
```

Model Building and Evaluation

Data was loaded into H2o package and a train test split of 70 to 30 was done to model and evaluate the model. The General Linear Model approach was selected to build the linear regression in which the regularaization technique was used to achieve model results. Parameters were defined to proceed with modeling of which alpha was set to 1 for L1 and 0 for L2 whiles lambda was set to greater than 0 in the range

What is Regularization?

Simply, it is a technique that prevents the model from overfitting by adding extra information to it. In cases of overfitting, the model may not be able to predict the output when it deals with unseen data and adds noise to the output. Regularization reduces (or regularizes) the magnitude of the variables while at the same time keeps the number of features. In that way, the noise gets reduced. Mathematically, regularization minimizes the cost function. It introduces penalty in our loss function. Regularization penalizes the slope, m, as it increases – both in positive scale and negative scale. In that way, the magnitude of the variables gets reduced. Unless there is a very strong interaction that lowers the loss significantly, we want to keep the absolute value of the slopes small. Lambda is the strength of the penalization. As lambda increases, the parameters become smaller which reduces overfitting.

We need to find a function that penalizes weak correlations and promotes strong correlations. The following are the two forms of the functions:

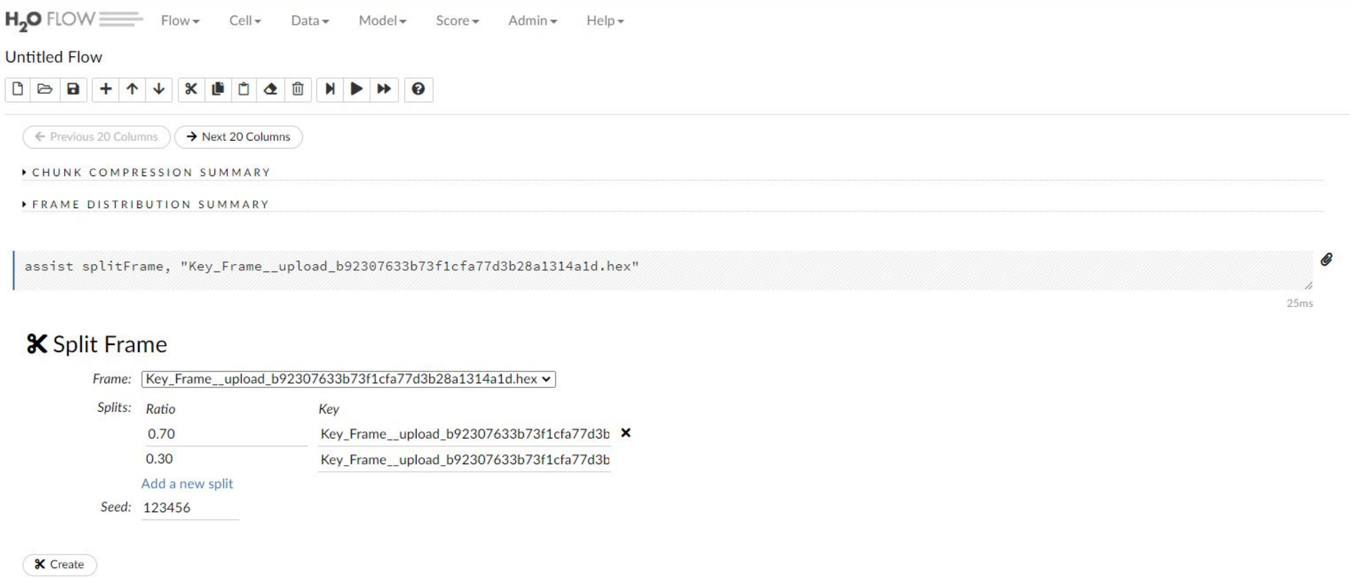
L1 Regularization: It is also called Lasso Regression or first order regularization. The function of the penalty is the absolute value of the coefficient. It primarily helps us with variable selection. It introduces sparsity our equation. As we increase the value of lambda, some of the slope to zero and others to non-zero. We end up keeping the non-zero-coefficient variables because those are the important ones. This process overall makes the primary use as feature selection.

L2 Regularization: It is also called Ridge regularization or second order regularization. The function of the penalty is the square of the coefficient. It does not induce sparsity. It only allows variable to contribute. It primarily used to prevent overfitting.

Source: https://www.javatpoint.com/regularization-in-machine-learning (https://www.javatpoint.com/regularization-in-machine-learning)

Below is a summaraized output of results from H2O detailing the comparison between L1 and L2 models in which L2(Ridge Regression) was chosen based on the metrics mentioned above because the predicted values were closer to the actual values used in the dataset.MSE, MAE, RMSE and AIC scored the lowest which means the predicted values were closer to actual values from that model. R-sqaure was actually higher in L2 compared to that from L1 due to explanabilty defined above.

Sampling Techniques used was 70 to 30 split in H2O Package



Untitled Flow

```
assist splitFrame, "Key_Frame__upload_b92307633b73f1cfa77d3b28a1314a1d.hex"
```

Split Frame

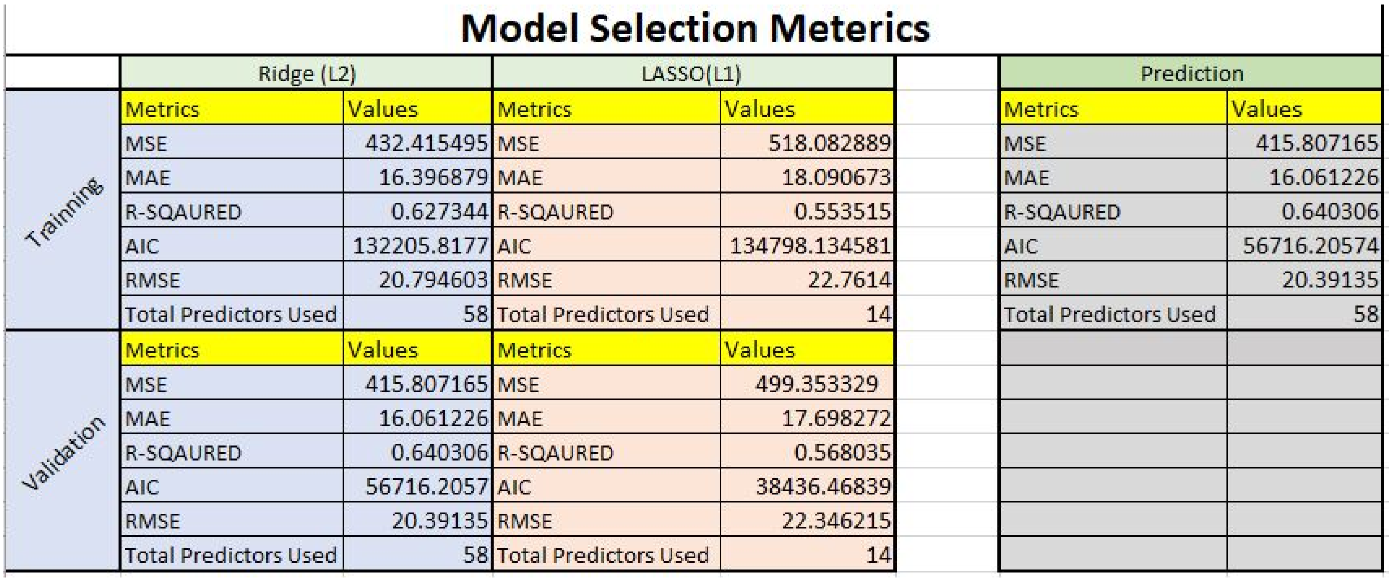
Frame: Key_Frame__upload_b92307633b73f1cfa77d3b28a1314a1d.hex

Splits: Ratio	Key
0.70	Key_Frame__upload_b92307633b73f1cfa77d3b
0.30	Key_Frame__upload_b92307633b73f1cfa77d3b

Add a new split

Seed: 123456

Table results of model metrics for comparison to choose prefered model



Model Selection Meterics

	Ridge (L2)		LASSO(L1)			Prediction	
Trainning	Metrics	Values	Metrics	Values		Metrics	Values
	MSE	432.415495	MSE	518.082889		MSE	415.807165
	MAE	16.396879	MAE	18.090673		MAE	16.061226
	R-SQAURED	0.627344	R-SQAURED	0.553515		R-SQAURED	0.640306
	AIC	132205.8177	AIC	134798.134581		AIC	56716.20574
	RMSE	20.794603	RMSE	22.7614		RMSE	20.39135
	Total Predictors Used	58	Total Predictors Used	14		Total Predictors Used	58
Validation	Metrics	Values	Metrics	Values			
	MSE	415.807165	MSE	499.353329			
	MAE	16.061226	MAE	17.698272			
	R-SQAURED	0.640306	R-SQAURED	0.568035			
	AIC	56716.2057	AIC	38436.46839			
	RMSE	20.39135	RMSE	22.346215			
	Total Predictors Used	58	Total Predictors Used	14			

Model Interpretability & Explainability

Below are the results images from the model build in which the coefficients are generated to help interpret the linear regression model.

From image5, the feature importance from the L1 model is shown in which just 14 features were used to interpret the model whereas in image6 the feature importance from the L2 model is shown with 58 variables used to achieve the best model performance.

In image6 and image7, the coefficients of entire linear regression model is shown with their signs (positve or negative) to showtheir relationship with the target variable of interest.

From these results, the top five is analyzed to interpret the model.

critical_temp = -79.3245 - 0.0074(number_of_elements) + 0.0572(mean_atomic_mass) + 0.0572(mean_atomic_mass) -0.0601(wtd_mean_atomic_mass) + 0.0569(gmean_atomic_mass) + 0.0817(wtd_gmean_atomic_mass) ... N(M) where N is cofficient and M is the variable.

Interpretation

Intercept

From the above, the model can be interpreted as holding all variable at zero, the critical_temp will have a negative effect on the superconductor at its intercept

All Betas

Also, holding the intercept and all other variables constant except for number_of_elements, the critical_temp will have a negative effect on the superconductor etc.

Image of feauture importance in L1 (LASSO)

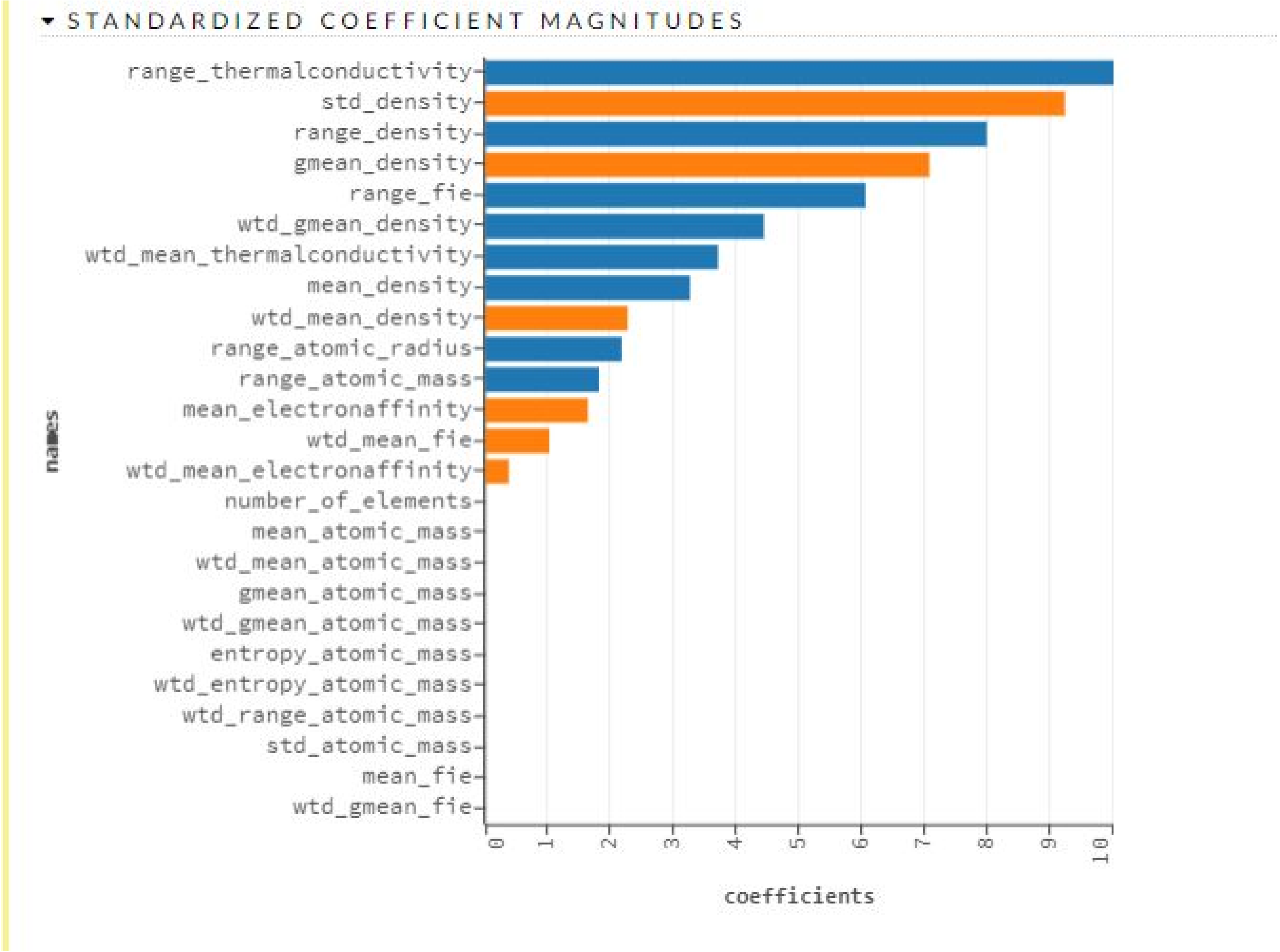
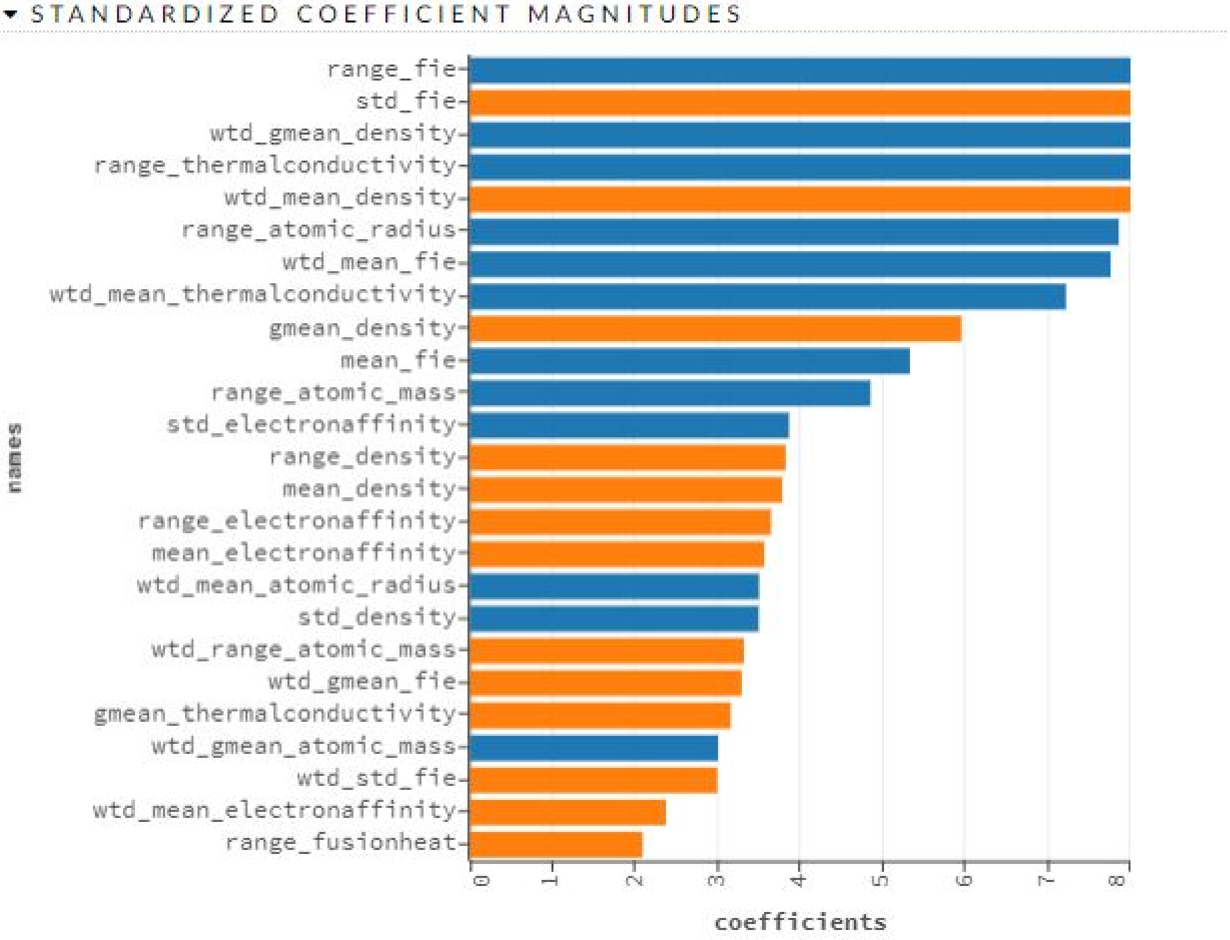
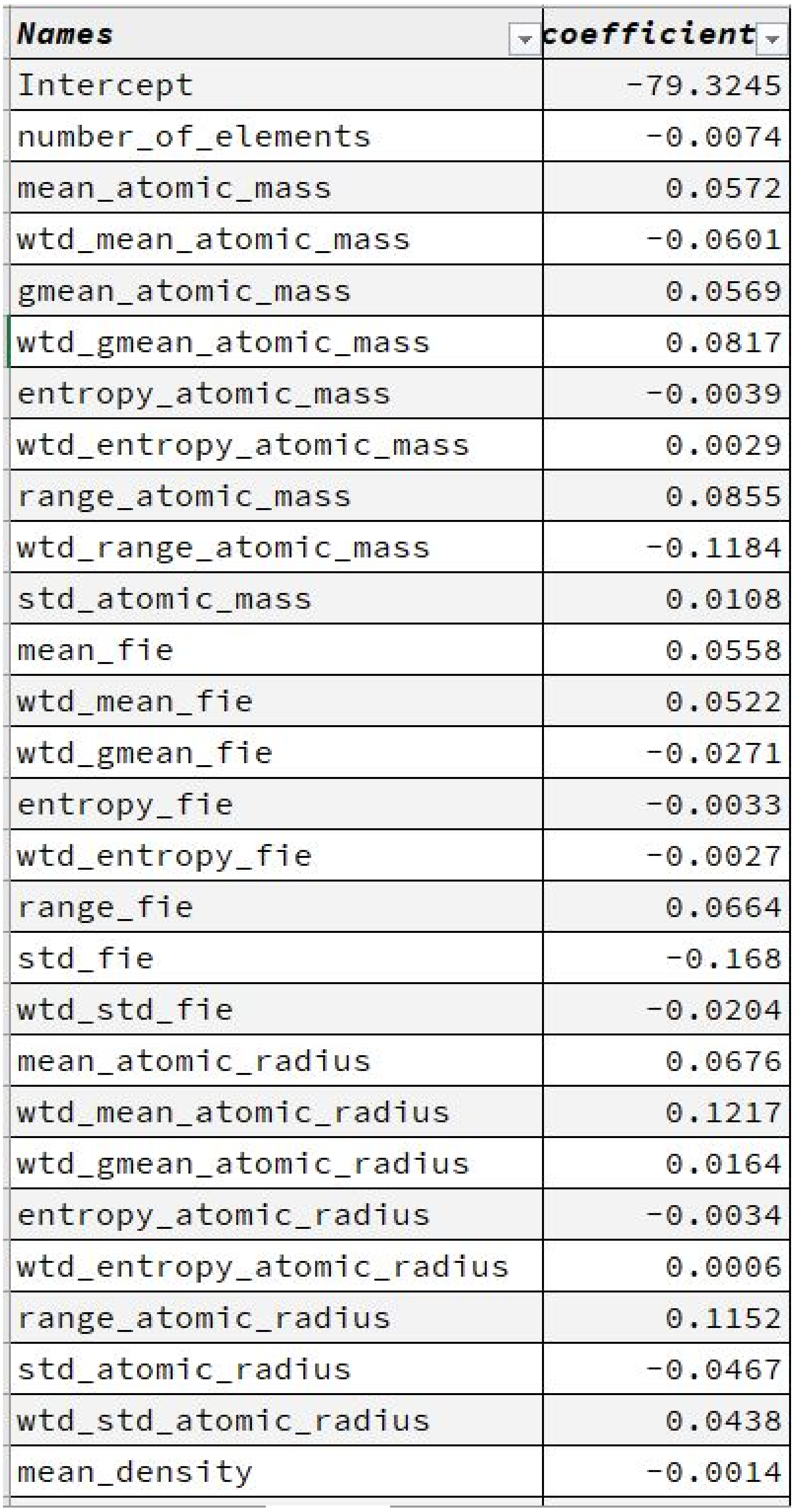


Image of feauture importance in L2 (Ridge)



▾ STANDARDIZED COEFFICIENT MAGNITUDES

Image of model coefficients

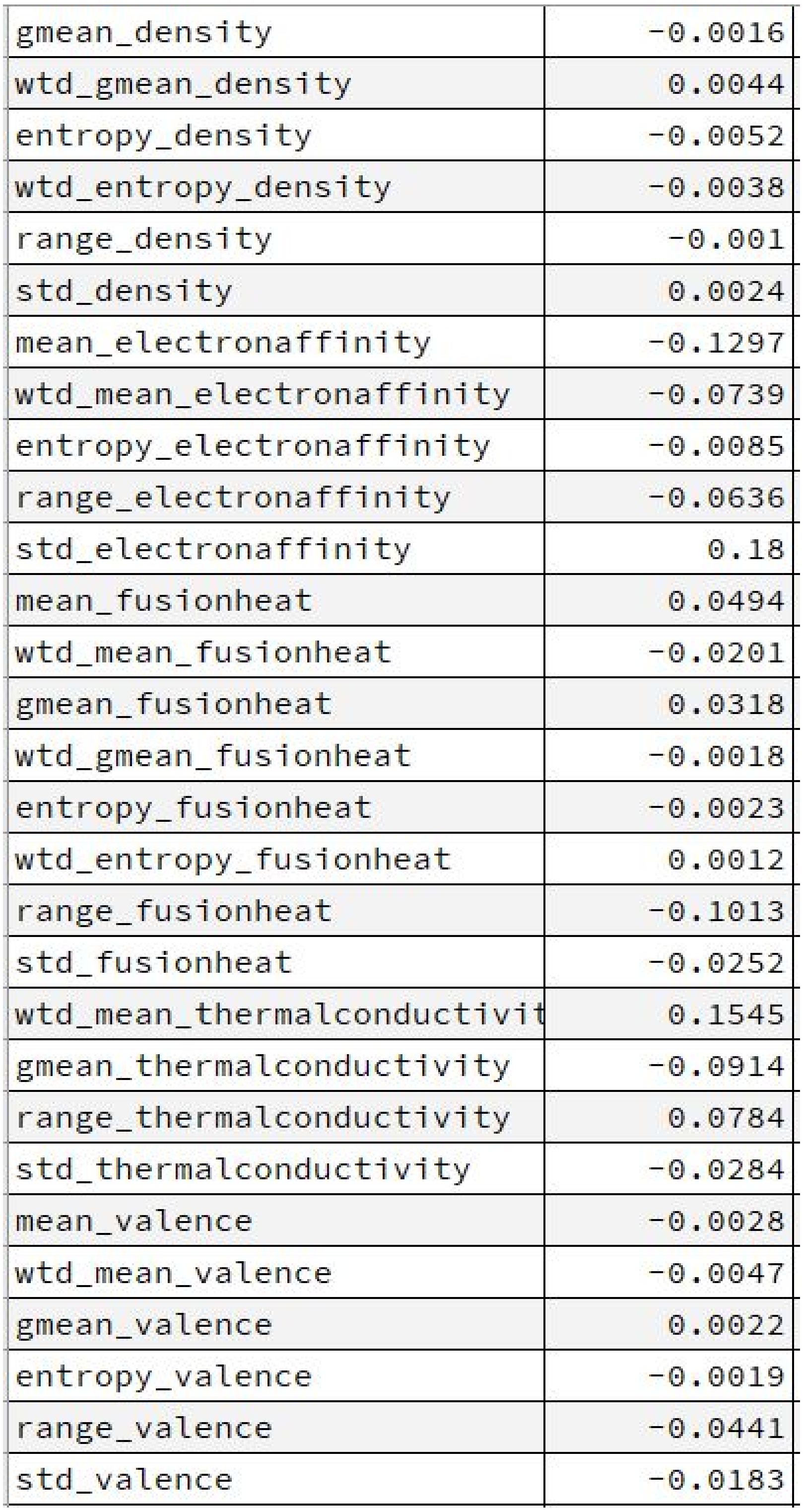


Names	coefficient
Intercept	-79.3245
number_of_elements	-0.0074
mean_atomic_mass	0.0572
wtd_mean_atomic_mass	-0.0601
gmean_atomic_mass	0.0569
wtd_gmean_atomic_mass	0.0817
entropy_atomic_mass	-0.0039
wtd_entropy_atomic_mass	0.0029
range_atomic_mass	0.0855
wtd_range_atomic_mass	-0.1184
std_atomic_mass	0.0108
mean_fie	0.0558
wtd_mean_fie	0.0522
wtd_gmean_fie	-0.0271
entropy_fie	-0.0033
wtd_entropy_fie	-0.0027
range_fie	0.0664
std_fie	-0.168
wtd_std_fie	-0.0204
mean_atomic_radius	0.0676
wtd_mean_atomic_radius	0.1217
wtd_gmean_atomic_radius	0.0164
entropy_atomic_radius	-0.0034
wtd_entropy_atomic_radius	0.0006
range_atomic_radius	0.1152
std_atomic_radius	-0.0467
wtd_std_atomic_radius	0.0438
mean_density	-0.0014

In [ ]:

```
#Image of model coefficients
from PIL import Image
myImage8 = Image.open("Model_Coefficients_2.jpg");
myImage8.show();
```

Image of model coefficients Cont'd



gmean_density	-0.0016
wtd_gmean_density	0.0044
entropy_density	-0.0052
wtd_entropy_density	-0.0038
range_density	-0.001
std_density	0.0024
mean_electronaffinity	-0.1297
wtd_mean_electronaffinity	-0.0739
entropy_electronaffinity	-0.0085
range_electronaffinity	-0.0636
std_electronaffinity	0.18
mean_fusionheat	0.0494
wtd_mean_fusionheat	-0.0201
gmean_fusionheat	0.0318
wtd_gmean_fusionheat	-0.0018
entropy_fusionheat	-0.0023
wtd_entropy_fusionheat	0.0012
range_fusionheat	-0.1013
std_fusionheat	-0.0252
wtd_mean_thermalconductivit	0.1545
gmean_thermalconductivity	-0.0914
range_thermalconductivity	0.0784
std_thermalconductivity	-0.0284
mean_valence	-0.0028
wtd_mean_valence	-0.0047
gmean_valence	0.0022
entropy_valence	-0.0019
range_valence	-0.0441
std_valence	-0.0183

Conclusion

We are proposing to use the Ridge regression (L2) to predict critical temperatures of superconducting materials. This is based on our analysis in comparing the model results of MAE, MSE, RMSE, AIC against different modeling techniques. The L2 had lower evaluation metrics than the other models.

Our prediction model using (L2) helped us predict the critical temperatures at which the majority of the material compositions in the study become superconductors. In addition, we were able to determine which are the set of variables with the most influence on our model.