Proteomic_2.0

December 20, 2017

1 This is a workbook to examine the proteomic data.

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        %matplotlib inline
```

I modified the gene names column by running it through my script to format the names correctly with gene names that appear in UniProt.

Let's read in the data frame and examine its shape and columns.

```
In [2]: df = pd.read_csv(r'/Users/eatonaw/Desktop/Atom_Files/Proteome/Copy of GPome_HBL1_TMD8_U
```

```
In [3]: print(df.shape)
        print(df.columns)
        df.head()
```

```
(21805, 32)
Index(['Gene names', 'HBL_Ib (log2 fold-change)',
       'HBL_PRT  (log2 fold-change)', 'TMD_Ib  (log2 fold-change)',
       'TMD_PRT  (log2 fold-change)', 'U2932_Ib  (log2 fold-change)',
       'U2932_PRT  (log2 fold-change)', 'Amino acid', 'Position in protein',
       'Regulated_HBL_Ib', 'Regulated_HBL_PRT', 'Regulated_TMD_Ib',
       'Regulated_TMD_PRT', 'Regulated_U2932_Ib', 'Regulated_U2932_PRT',
       'Localization prob', 'Protein', 'Protein names', 'Unique identifier',
       'HBL1 IBR callouts', 'TMD8.IBR.callouts', 'HBL1_CSS', 'TMD8_CSS',
       'U2932_CSS', 'Gene', 'ABC mutation freq', 'GCB mutation freq',
       'Unclass mutation freq', 'Total mutation freq', 'HBL1_ibru_resist_CS',
       'TMD8_ibr_resist_CS', 'Essential.Genes'],
      dtype='object')
```

Out[3]:

	Gene names	HBL_Ib (log2 fold-change)	HBL_PRT (log2 fold-change)	\
0	ADAT1	0.047538	NaN	
1	AHNAK	-2.093571	-1.976702	
2	AHNAK	-2.126455	-1.094125	

```
3      AHNAK                  -1.914902                    -1.166859
4      AHNAK                  -1.830332                          NaN

   TMD_Ib  (log2 fold-change)  TMD_PRT  (log2 fold-change)  \
0                    0.741057                     0.122076
1                   -1.065461                    -1.701201
2                   -0.886006                    -0.740939
3                   -0.866542                    -1.276695
4                   -0.857783                          NaN

   U2932_Ib  (log2 fold-change)  U2932_PRT  (log2 fold-change) Amino acid  \
0                     -0.181591                           NaN          S
1                      2.011746                           NaN          S
2                           NaN                           NaN          S
3                           NaN                           NaN          S
4                           NaN                           NaN          T

   Position in protein Regulated_HBL_Ib       ...        TMD8_CSS U2932_CSS  \
0                162.0              NaN       ...          1.2140  -0.05778
1                135.0                +       ...          0.7516  -0.03943
2               5110.0                +       ...          0.7516  -0.03943
3               5780.0                +       ...          0.7516  -0.03943
4               4100.0                +       ...          0.7516  -0.03943

    Gene ABC mutation freq GCB mutation freq  Unclass mutation freq  \
0  ADAT1            0.0000            0.0184                 0.000
1  AHNAK            0.0475            0.0307                 0.069
2  AHNAK            0.0475            0.0307                 0.069
3  AHNAK            0.0475            0.0307                 0.069
4  AHNAK            0.0475            0.0307                 0.069

  Total mutation freq HBL1_ibru_resist_CS TMD8_ibr_resist_CS  Essential.Genes
0              0.0052            0.055523         -1.495230              NaN
1              0.0470           -0.289416         -1.841236              NaN
2              0.0470           -0.289416         -1.841236              NaN
3              0.0470           -0.289416         -1.841236              NaN
4              0.0470           -0.289416         -1.841236              NaN

[5 rows x 32 columns]
```

Let's remove the rows where the value for the 'Gene names' column is 'NaN'.

```
In [4]: df.dropna(subset = ['Gene names'], inplace = True)
```

What stands out to me are the different amino acids and position in proteins they are. We should make new features that encompass this information

```
In [5]: df['gene_aa'] = df['Gene names'] + '_' + df['Amino acid']
        df['Position in protein'] = df['Position in protein'].astype(str)
```

```
        df['gene_aa_position'] = df['Gene names'] + '_' + df['Amino acid']\
         + '_' + df['Position in protein']
        df.gene_aa_position = df.gene_aa_position.str[:-2]
```

```
In [6]: print(df.gene_aa.head())
        df.gene_aa_position.head()
```

```
0    ADAT1_S
1    AHNAK_S
2    AHNAK_S
3    AHNAK_S
4    AHNAK_T
Name: gene_aa, dtype: object
```

```
Out[6]: 0     ADAT1_S_162
        1     AHNAK_S_135
        2    AHNAK_S_5110
        3    AHNAK_S_5780
        4    AHNAK_T_4100
        Name: gene_aa_position, dtype: object
```

We now have two new columns that contains the residue and residue position in the protein.

Now, let's get a better understanding of our features.

The columns with "(log2 fold-change)" in the name is the mass spec ratio under different conditions and cell lines. For example, HBL_Ib is HBL1 in Ibrutinib. The columns with "callouts" are genes that are thought to be interesting. "CSS" designates CRISPR screen scores and the "HBL1_ibru_resist_CS" is the HBL1 score in ibrutinib.

Let's look at the data by plotting the CRISPR score on the x-axis and the phosphorylation change on the y-axis.

If a point is orange (1.0) - it is a callout gene. If it is blue (0.0), it is not a callout gene.

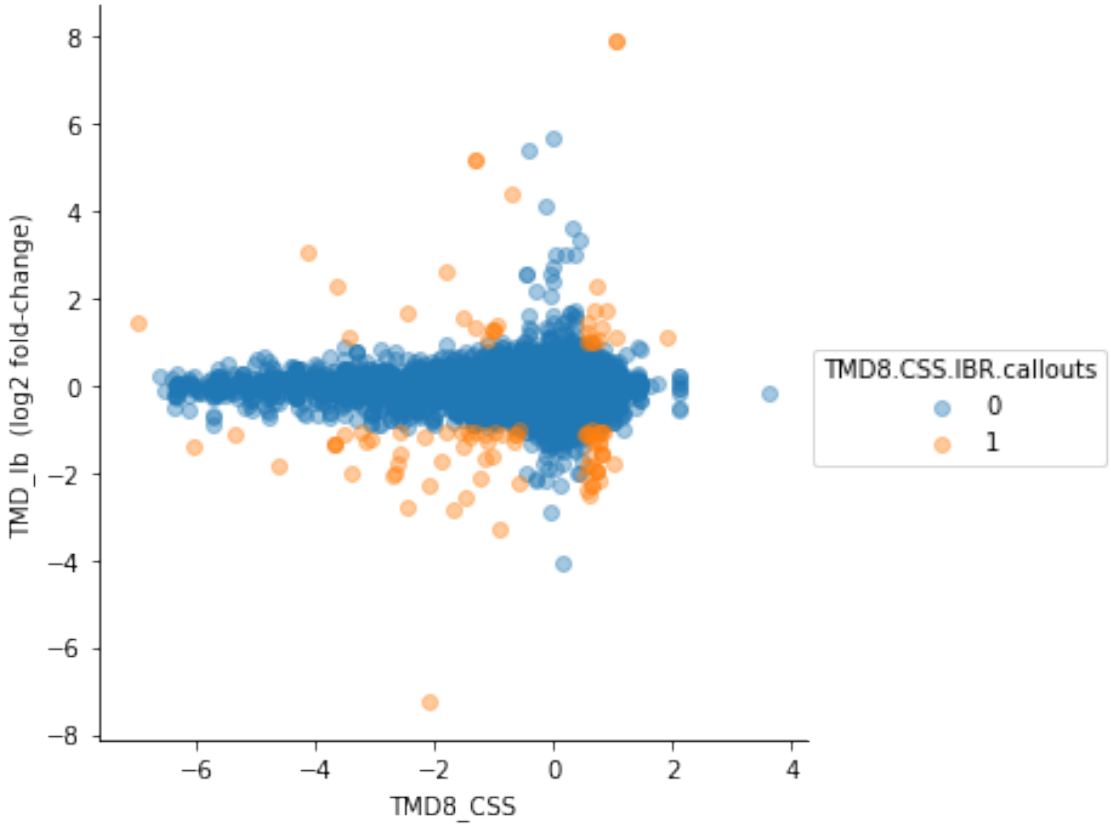
```
In [7]: df['TMD8.IBR.callouts'] = np.where(((df['TMD_Ib  (log2 fold-change)'] < -1) |\
            (df['TMD_Ib  (log2 fold-change)'] > 1)) & ((df['TMD8_ibr_resist_CS'] < -0.5) |\
            (df['TMD8_ibr_resist_CS'] > 0.5)), '1', '0')

        df['HBL1 IBR callouts'] = np.where(((df['HBL_Ib (log2 fold-change)'] < -1) |\
            (df['HBL_Ib (log2 fold-change)'] > 1)) & ((df['HBL1_ibru_resist_CS'] < -0.5) |\
            (df['HBL1_ibru_resist_CS'] > 0.5)), '1', '0')

        df['TMD8.PRT.callouts'] = np.where(((df['TMD_PRT  (log2 fold-change)'] < -1) |\
            (df['TMD_PRT  (log2 fold-change)'] > 1)) & ((df['TMD8_ibr_resist_CS'] < -0.5) |\
            (df['TMD8_ibr_resist_CS'] > 0.5)), '1', '0')
```

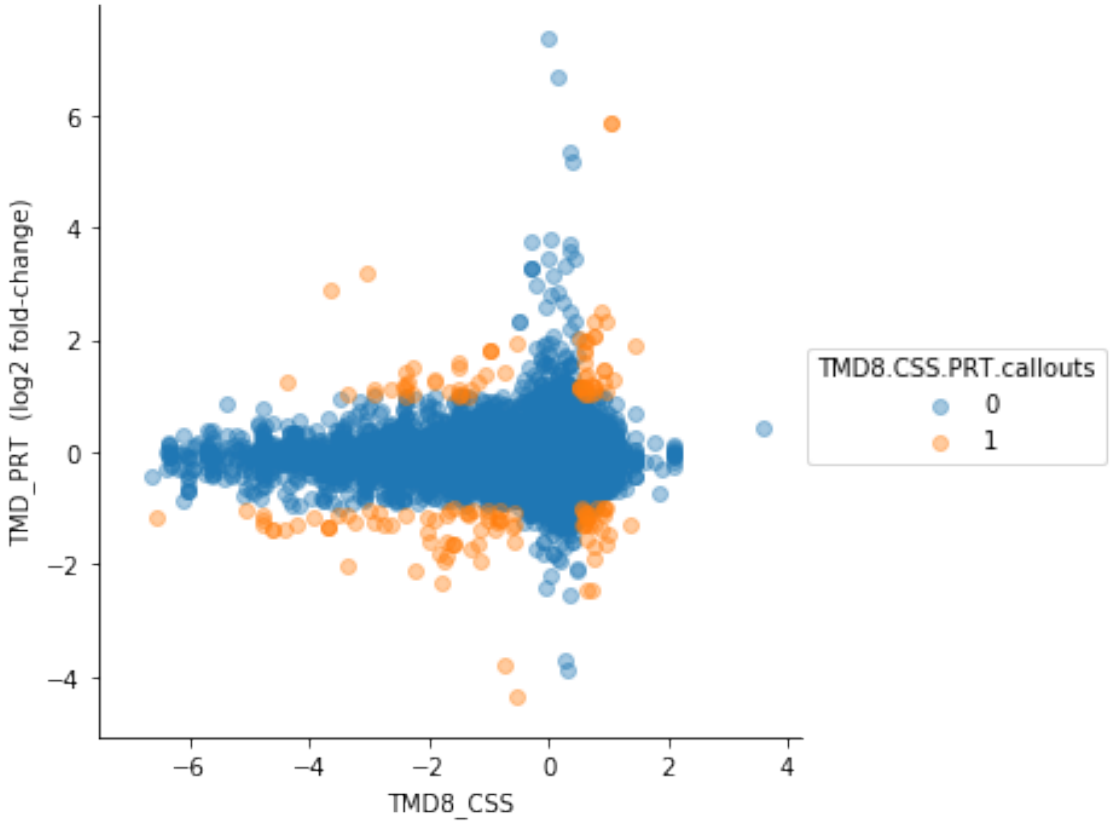
```
df['HBL1 PRT callouts'] = np.where(((df['HBL_PRT  (log2 fold-change)'] < -1) |\
    (df['HBL_PRT  (log2 fold-change)'] > 1)) & ((df['HBL1_ibru_resist_CS'] < -0.5) |\
    (df['HBL1_ibru_resist_CS'] > 0.5)), '1', '0')

df['u2932.IBR.callouts'] = np.where(((df['U2932_Ib  (log2 fold-change)'] < -1) |\
    (df['U2932_Ib  (log2 fold-change)'] > 1)) & ((df['U2932_CSS'] < -0.5) |\
    (df['U2932_CSS'] > 0.5)), '1', '0')

df['u2932.PRT.callouts'] = np.where(((df['U2932_PRT  (log2 fold-change)'] < -1) |\
    (df['U2932_PRT  (log2 fold-change)'] > 1)) & ((df['U2932_CSS'] < -0.5) |\
    (df['U2932_CSS'] > 0.5)), '1', '0')

df['TMD8.CSS.IBR.callouts'] = np.where(((df['TMD_Ib  (log2 fold-change)'] < -1) |\
    (df['TMD_Ib  (log2 fold-change)'] > 1)) & ((df['TMD8_CSS'] < -0.5) |\
    (df['TMD8_CSS'] > 0.5)), '1', '0')

df['TMD8.CSS.PRT.callouts'] = np.where(((df['TMD_PRT  (log2 fold-change)'] < -1) |\
    (df['TMD_PRT  (log2 fold-change)'] > 1)) & ((df['TMD8_CSS'] < -0.5) |\
    (df['TMD8_CSS'] > 0.5)), '1', '0')

df['HBL1 CSS IBR callouts'] = np.where(((df['HBL_Ib (log2 fold-change)'] < -1) |\
    (df['HBL_Ib (log2 fold-change)'] > 1)) & ((df['HBL1_CSS'] < -0.5) |\
    (df['HBL1_CSS'] > 0.5)), '1', '0')

df['HBL1 CSS PRT callouts'] = np.where(((df['HBL_PRT  (log2 fold-change)'] < -1) |\
    (df['HBL_PRT  (log2 fold-change)'] > 1)) & ((df['HBL1_CSS'] < -0.5) |\
    (df['HBL1_CSS'] > 0.5)), '1', '0')
```

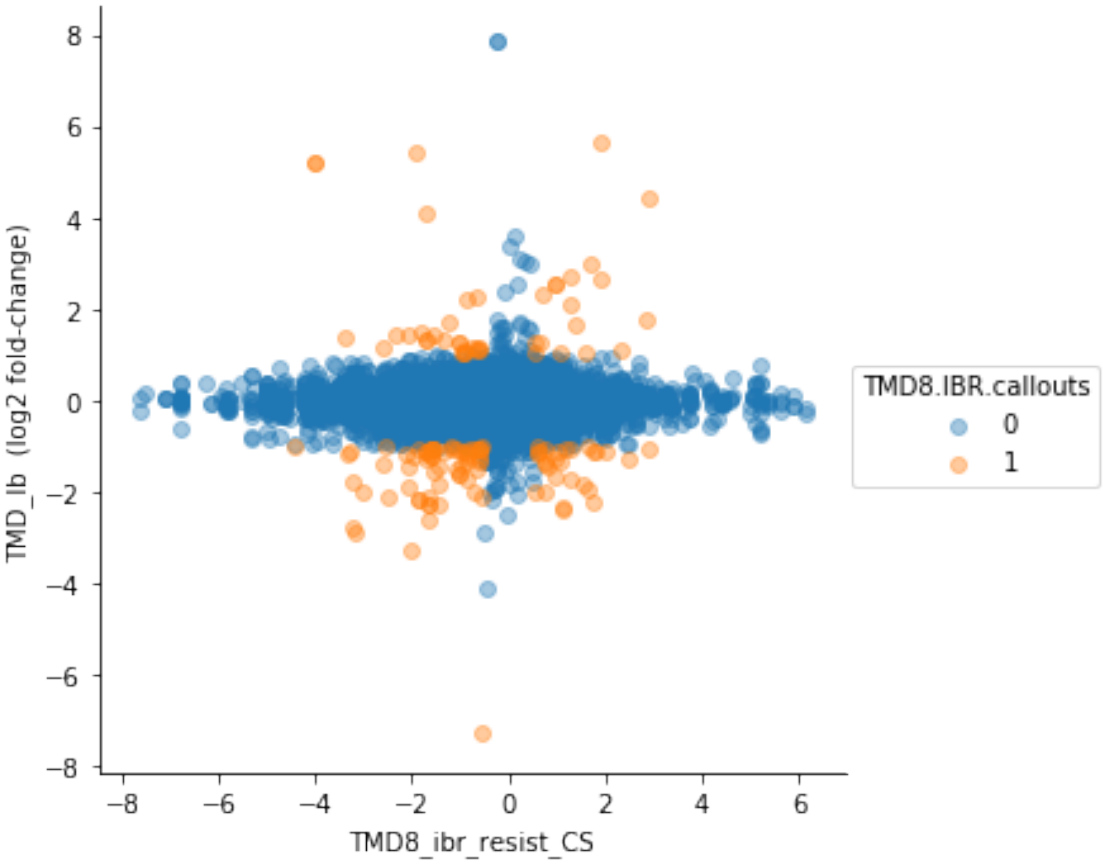
In [8]:
```
sns.lmplot(x = 'TMD8_CSS', y = 'TMD_Ib  (log2 fold-change)',\
    data = df, hue = 'TMD8.CSS.IBR.callouts', fit_reg = False, scatter_kws={'alpha':0.4

sns.lmplot(x = 'TMD8_CSS', y = 'TMD_PRT  (log2 fold-change)',\
    data = df, hue = 'TMD8.CSS.PRT.callouts', fit_reg = False, scatter_kws={'alpha':0.4

sns.lmplot(x = 'TMD8_ibr_resist_CS', y = 'TMD_Ib  (log2 fold-change)',\
    data = df, hue = 'TMD8.IBR.callouts', fit_reg = False, scatter_kws={'alpha':0.4})

sns.lmplot(x = 'TMD8_ibr_resist_CS', y = 'TMD_PRT  (log2 fold-change)',\
    data = df, hue = 'TMD8.PRT.callouts', fit_reg = False, scatter_kws={'alpha':0.4})

sns.lmplot(x = 'HBL1_CSS', y = 'HBL_Ib (log2 fold-change)',\
    data = df, hue = 'HBL1 CSS IBR callouts', fit_reg = False, scatter_kws={'alpha':0.4

sns.lmplot(x = 'HBL1_CSS', y = 'HBL_PRT  (log2 fold-change)',\
    data = df, hue = 'HBL1 CSS PRT callouts', fit_reg = False, scatter_kws={'alpha':0.4

sns.lmplot(x = 'HBL1_ibru_resist_CS', y = 'HBL_Ib (log2 fold-change)',\
```

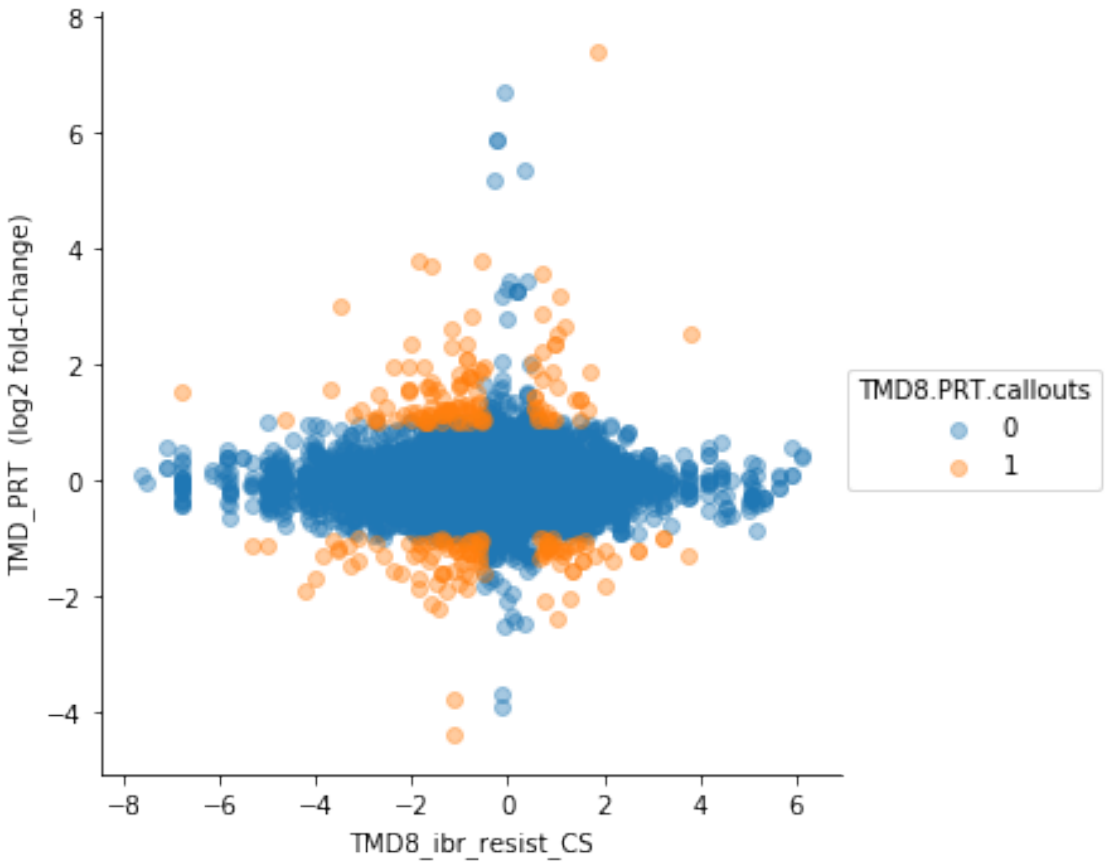
```
        data = df, hue = 'HBL1 IBR callouts', fit_reg = False, scatter_kws={'alpha':0.4})

    sns.lmplot(x = 'HBL1_ibru_resist_CS', y = 'HBL_PRT  (log2 fold-change)',\
        data = df, hue = 'HBL1 PRT callouts', fit_reg = False, scatter_kws={'alpha':0.4})

    sns.lmplot(x = 'U2932_CSS', y = 'U2932_Ib  (log2 fold-change)',\
        data = df, hue = 'u2932.IBR.callouts', fit_reg = False, scatter_kws={'alpha':0.4})

    sns.lmplot(x = 'U2932_CSS', y = 'U2932_PRT  (log2 fold-change)',\
        data = df, hue = 'u2932.PRT.callouts', fit_reg = False, scatter_kws={'alpha':0.4})
```

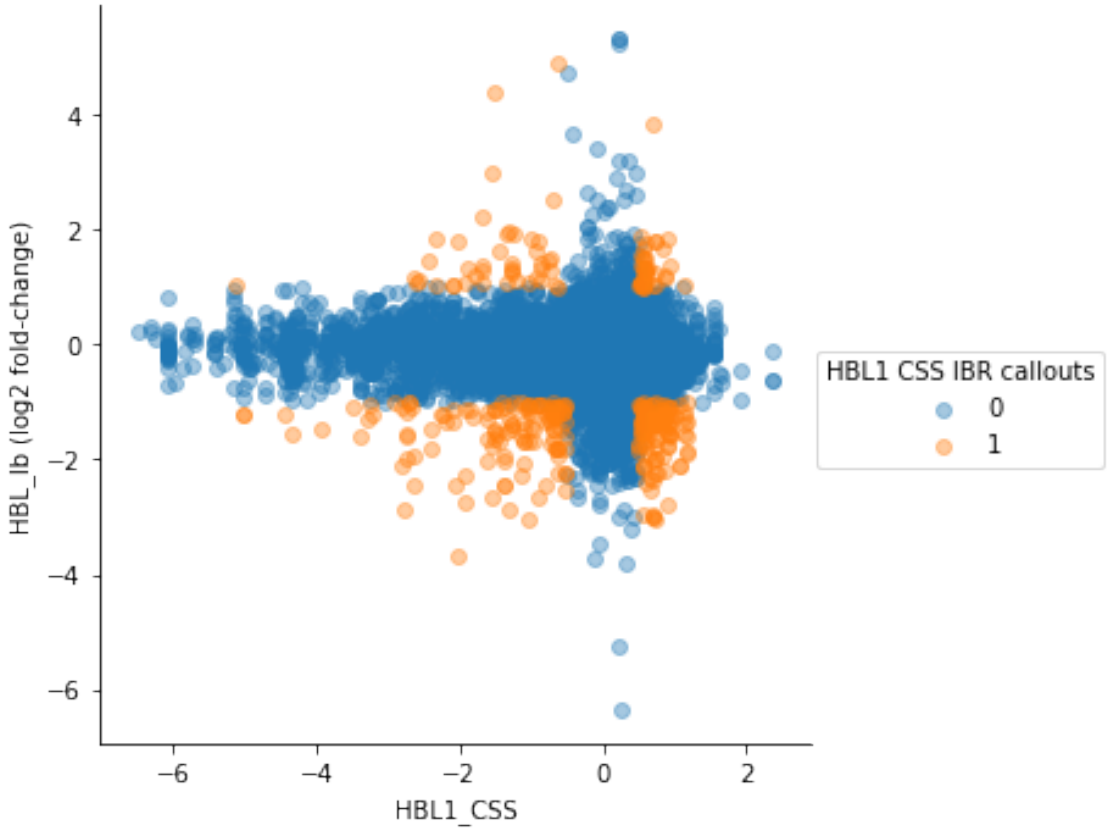
Out[8]: <seaborn.axisgrid.FacetGrid at 0x117584ef0>

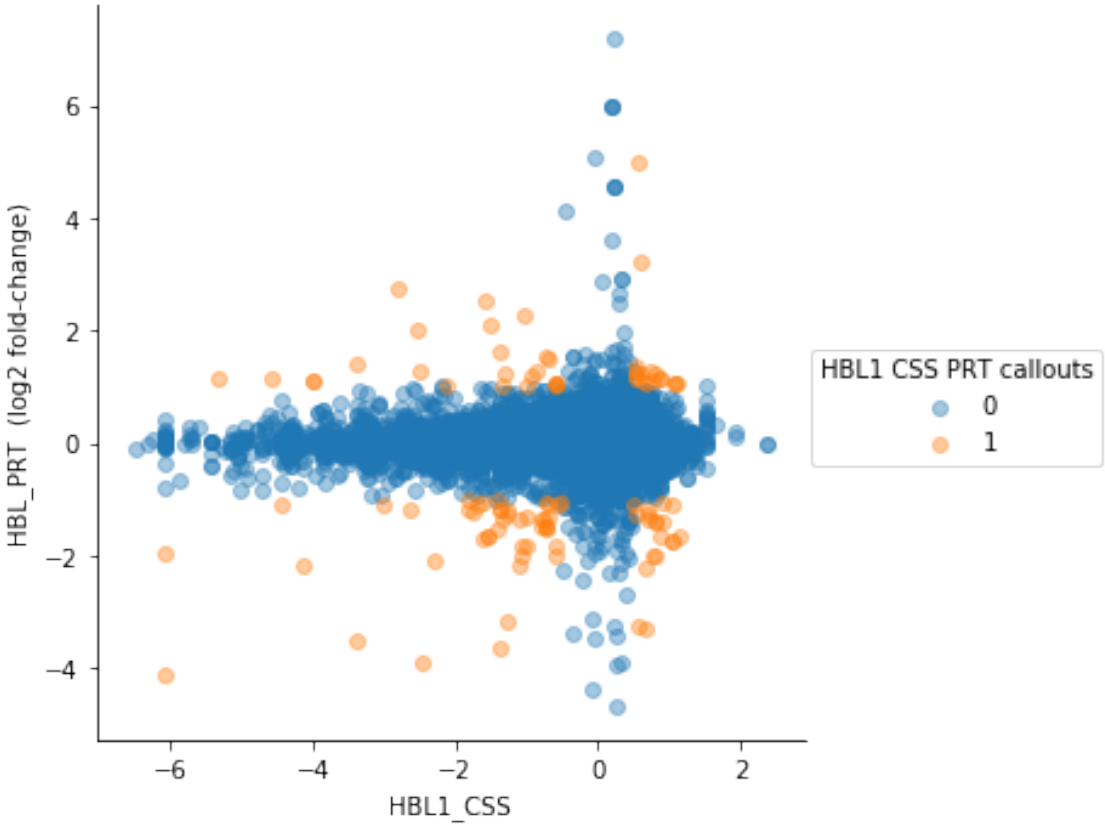


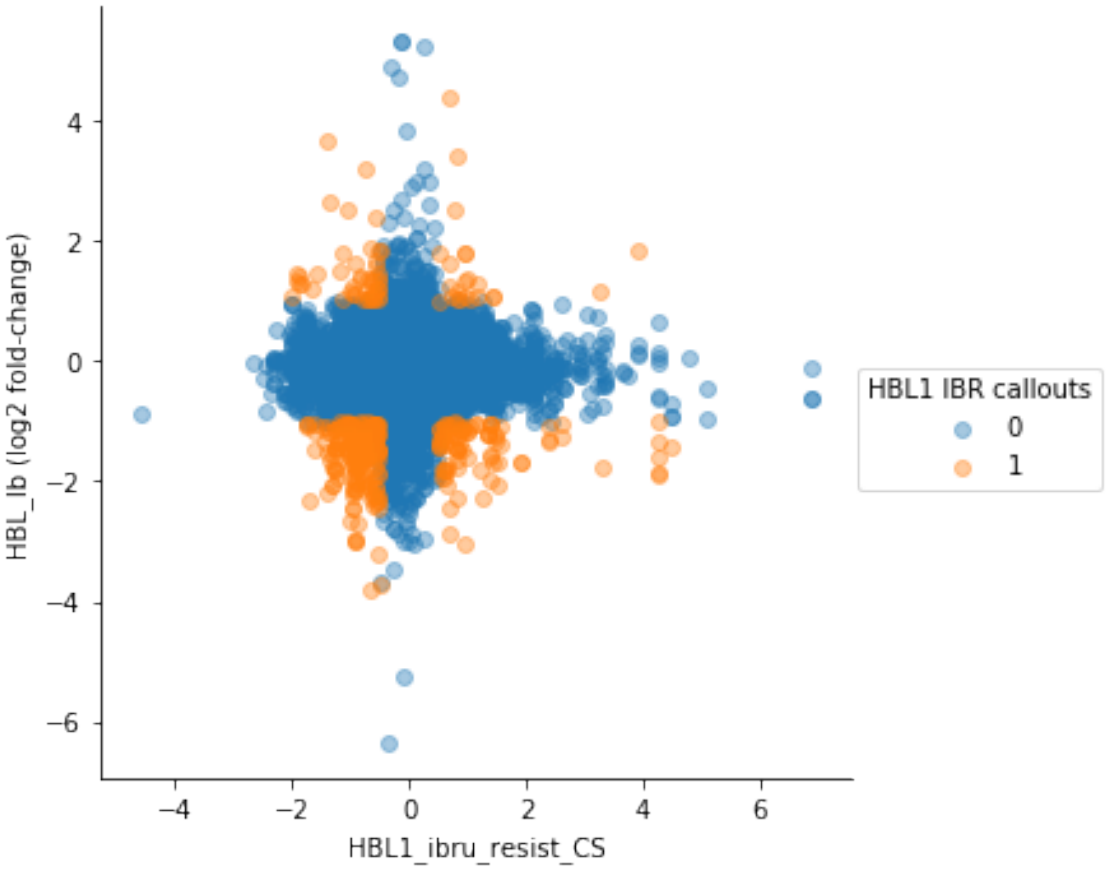

TMD_PRT (log2 fold-change)
6
4
2
0
−2
−4
−6
−4
−2
0
2
4
TMD8_CSS
TMD8.CSS.PRT.callouts
0
1

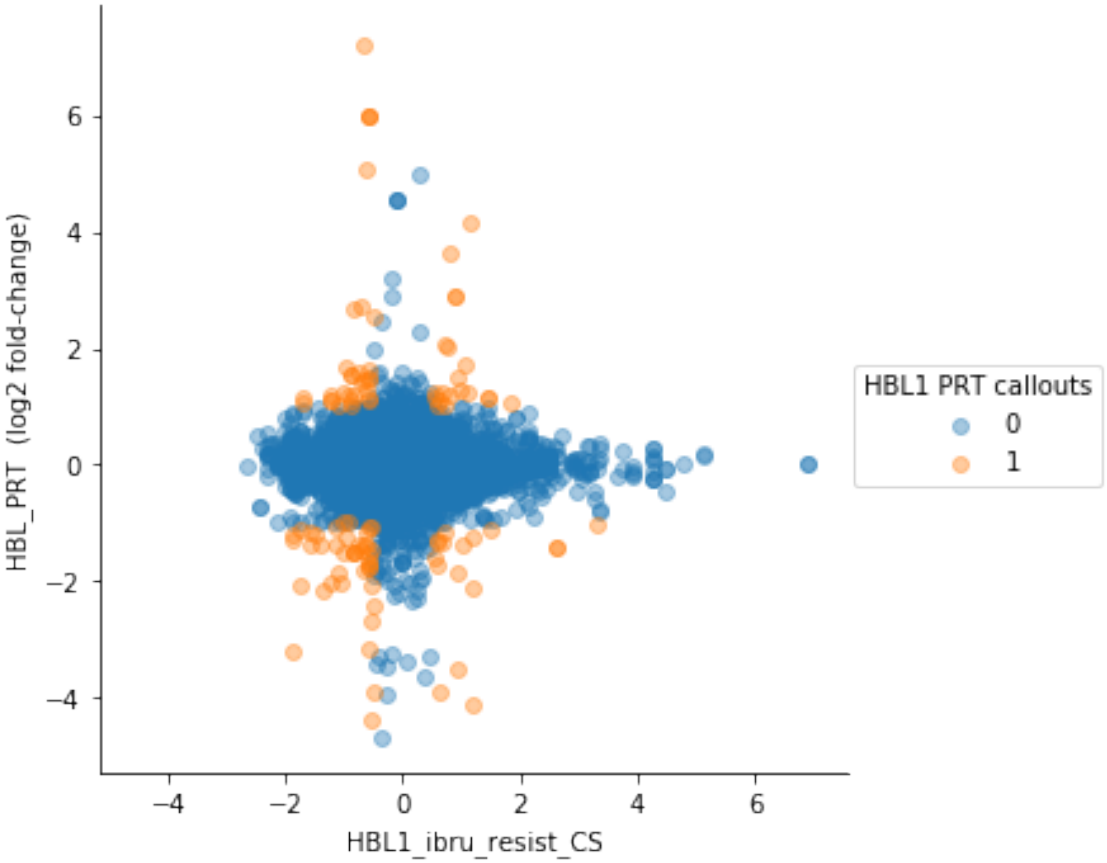

TMD_Ib (log2 fold-change)
TMD8_ibr_resist_CS
TMD8.IBR.callouts
0
1

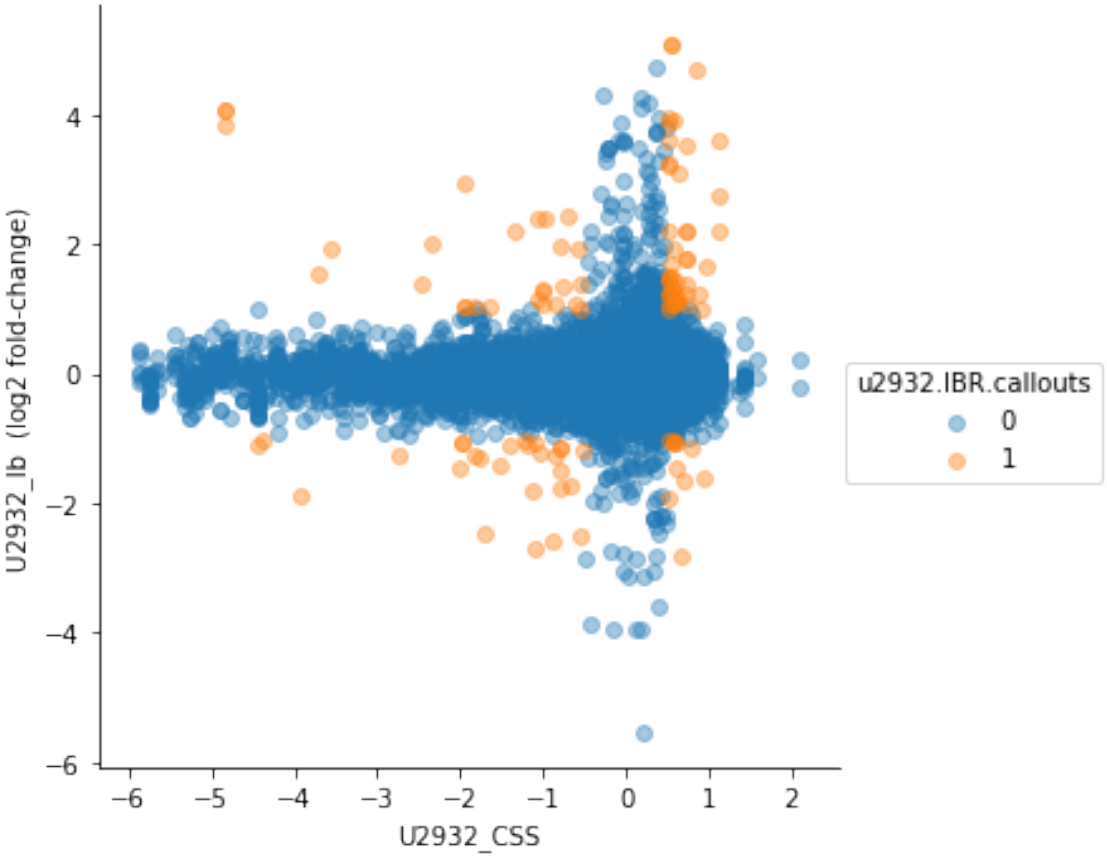

TMD_PRT (log2 fold-change)
TMD8_ibr_resist_CS
TMD8.PRT.callouts
0
1
8
6
4
2
0
−2
−4
−8
−6
−4
−2
0
2
4
6

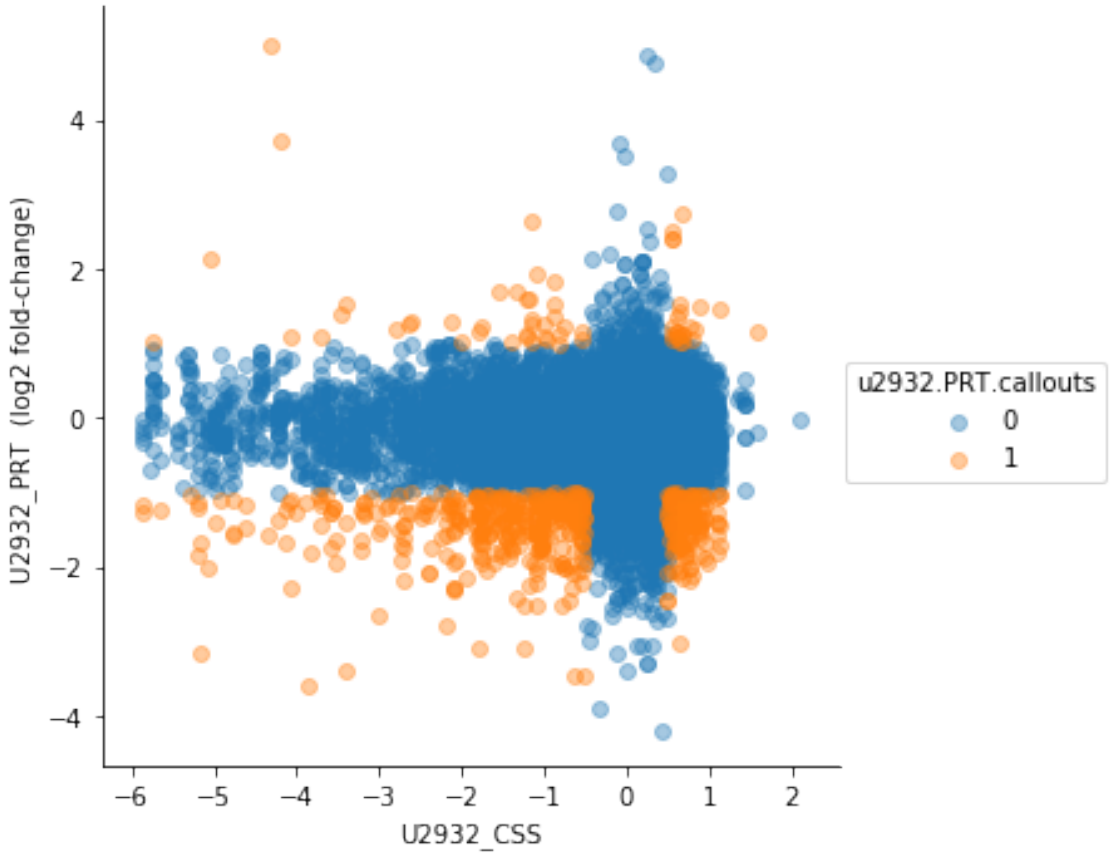

HBL_Ib (log2 fold-change)
4
2
0
−2
−4
−6
−6
−4
−2
0
2
HBL1_CSS
HBL1 CSS IBR callouts
0
1


HBL_PRT (log2 fold-change)
6
4
2
0
−2
−4
−6
−4
−2
0
2
HBL1_CSS
HBL1 CSS PRT callouts
0
1


HBL_Ib (log2 fold-change)
HBL1_ibru_resist_CS
HBL1 IBR callouts
0
1
4
2
0
−2
−4
−6
−4
−2
0
2
4
6


HBL_PRT (log2 fold-change)
6
4
2
0
−2
−4
−4
−2
0
2
4
6
HBL1_ibru_resist_CS
HBL1 PRT callouts
0
1


U2932_Ib (log2 fold-change)
U2932_CSS
u2932.IBR.callouts
0
1



Let's now output the callout columns and CRISPR scores to an excel file so that we can later put the data into StringDB, etc.

```
In [38]: import xlsxwriter

In [10]: df2 = df[['Gene names', 'HBL1 IBR callouts', 'TMD8.IBR.callouts', 'TMD8.PRT.callouts'
             'u2932.IBR.callouts', 'u2932.PRT.callouts', 'TMD8.CSS.IBR.callouts', 'TMD8.CSS.PR
             'HBL1 CSS IBR callouts', 'HBL1 CSS PRT callouts', 'HBL_PRT  (log2 fold-change)',\
             'TMD_Ib  (log2 fold-change)','TMD_PRT  (log2 fold-change)', 'U2932_Ib  (log2 fold
             'U2932_PRT  (log2 fold-change)', 'HBL1_CSS', 'TMD8_CSS', 'U2932_CSS', 'HBL1_ibru_
             'TMD8_ibr_resist_CS']]

In [39]: writer = pd.ExcelWriter('proteome_master.xlsx')
         df.to_excel(writer)
         writer.save()
```