Selecting Top genes

This code is a part of class project for PHC6067.

Student: Vitalii Stebliankin

Instructor: Dr. Changwon Yoo

In this section we will select top genes that correlates with the disease variable.

A. Discrete

In [1]:

```
import pandas as pd
from scipy import stats
from matplotlib import pyplot as plt
import seaborn as sns

n_top_genes=20
```

In [2]:

```
# Read Data

data_discrete = "../../data/oct5-oct11_2020/Merged-discret-dropNA.csv"
df_disc = pd.read_csv(data_discrete)
df_disc.index = df_disc["Unnamed: 0"]
df_disc = df_disc.drop(["Unnamed: 0"], axis=1)

df_disc = df_disc.T

df_disc["age"] = df_disc["age"].apply(lambda x: 0 if x=="<1" else float(x))
df_disc["disease"] = df_disc["disease"].apply(lambda x: int(x))

df_disc = df_disc.apply(pd.to_numeric)

df_disc.head()
```

```
/Users/stebliankin/miniconda3/lib/python3.8/site-packages/IPython/core/interactives
hell.py:3145: DtypeWarning: Columns (1085) have mixed types.Specify dtype option on
import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

Out[2]:

Unnamed: 0	gender	age	disease	A1BG	A1CF	A2M	A2M-AS1	A2ML1	AA06	AACS	...	ZW10	ZW
GSM855942	1.0	13.0	1	1.0	1.0	2.0	1.0	1.0	1.0	1.0	...	1.0	
GSM855943	1.0	15.0	1	1.0	1.0	2.0	1.0	1.0	1.0	1.0	...	1.0	
GSM855944	1.0	13.0	1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	1.0	
GSM855945	0.0	20.0	1	1.0	1.0	2.0	1.0	1.0	1.0	1.0	...	1.0	
GSM855946	1.0	17.0	1	1.0	1.0	2.0	1.0	1.0	1.0	1.0	...	1.0	

5 rows × 17495 columns

Correlations with clinical variables

We will use Kendall Rank Correlation because "disease" variable is categorical.

In [3]:

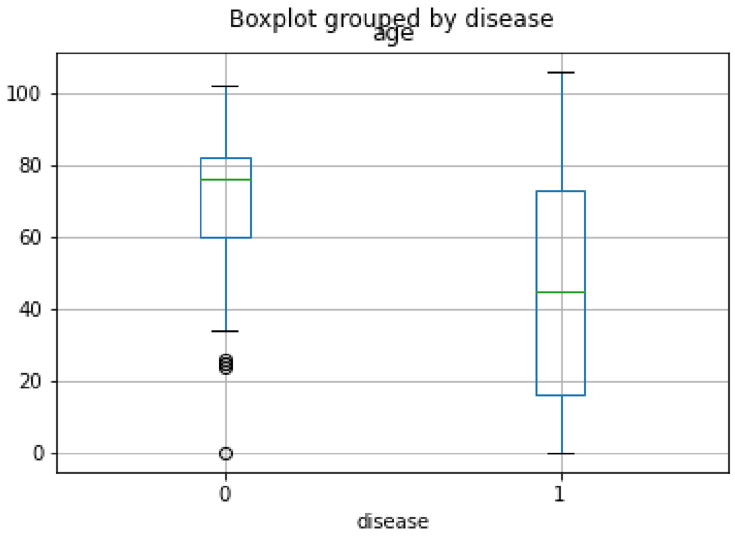
```
#
df = df_disc
clinical_vars = ["gender", "age"]
all_vars = list(df.columns)
print("Correlation with clinical variables")
tau, p_value = stats.kendalltau(list(df["disease"]), list(df["age"]))
print("Age vs Disease: tau: {}; p value: {}".format(tau, p_value))
#df["age"] = df["age"].apply(lambda x: float(x))

df.boxplot(by="disease", column="age")
# print(tau, p_value)
# for var in all_vars:
#     pass
```

```
Correlation with clinical variables
Age vs Disease: tau: -0.19341700162935796; p value: 3.2032844125140975e-16
```

```
/Users/stebliankin/miniconda3/lib/python3.8/site-packages/numpy/core/_asarray.py:8
3: VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences (whi
ch is a list-or-tuple of lists-or-tuples-or ndarrays with different lengths or shap
es) is deprecated. If you meant to do this, you must specify 'dtype=object' when cr
eating the ndarray
  return array(a, dtype, copy=False, order=order)
```

Out[3]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1359523a0>
```



Observations:

- Disease is more common in the yonger group

In [4]:

```
tau, p_value = stats.kendalltau(list(df["disease"]), list(df["gender"]))
print("Age vs Disease: tau: {}; p value: {}".format(tau, p_value))

df_fem = df[df["gender"]==0]
df_male = df[df["gender"]==1]

print("N males: {}; control: {}; disease: {}; percent disease: {}".format(len(df_ma
                                                                            len(d
                                                                           len(df
                                                                           len(df
print("N females: {}; control: {}; disease: {}; percent disease: {}".format(len(df_
                                                                            len(d
                                                                           len(df
                                                                           len(df

print("Total control subjects: {}; Total disease subjects: {}")
```

```
Age vs Disease: tau: -0.07316386891032732; p value: 0.0112278409283513 26
N males: 705; control: 84; disease: 621; percent disease: 0.8808510638297873
N females: 497; control: 37; disease: 460; percent disease: 0.9255533199195171
Total control subjects: {}; Total disease subjects: {}
```

Observation:

- Data is skewed towards disease state

Correlated genes

In [5]:
```
all_vars = df.columns
clinical_vars = ["gender", "age", "disease"]

corr_dict = {"gene":[], "tau":[], "p_value":[]}
for var in all_vars:
    if var not in clinical_vars:
        try:
            tau, p_value = stats.kendalltau(list(df["disease"]), list(df[var]))
            corr_dict["tau"].append(tau)
            corr_dict["gene"].append(var)
            corr_dict["p_value"].append(p_value)
        except ValueError:
            print(var)
```

```
1-Mar
2-Mar
1-Mar
2-Mar
```

Observation:

Some of the genes has the following names

```
1-Mar
2-Mar
1-Mar
11-Mar
2-Mar
3-Mar
4-Mar
5-Mar
6-Mar
7-Mar
8-Mar
9-Mar
```

Are those correct gene names, or they got accidentally converted to the date format in excel?

In [6]:
```
corr_df = pd.DataFrame(corr_dict)
corr_df["tau_module"] = corr_df["tau"].apply(lambda x: x if x>0 else -x)
corr_df.head()
```

Out[6]:

	gene	tau	p_value	tau_module
0	A1BG	-0.060257	3.621857e-02	0.060257
1	A1CF	-0.093715	1.163320e-03	0.093715
2	A2M	0.188855	4.737177e-11	0.188855
3	A2M-AS1	-0.000882	9.754670e-01	0.000882
4	A2ML1	-0.005997	8.342184e-01	0.005997

In [7]:

```
corr_df_signif=corr_df[corr_df["p_value"]<0.05]
print("{} out of {} significant correlations".format(len(corr_df_signif), len(corr_

corr_df_top = corr_df_signif.nlargest(n_top_genes, 'tau_module')
corr_df_top
```

```
8397 out of 17488 significant correlations
```

Out[7]:

	gene	tau	p_value	tau_module
9986	NCOA7	-0.515115	2.811533e-71	0.515115
7120	KIAA0513	-0.498604	6.730395e-67	0.498604
16605	WDR7	-0.496619	2.212448e-66	0.496619
4145	DYNC1I1	-0.430025	1.863970e-50	0.430025
13888	SLC12A5	-0.421833	1.372029e-48	0.421833
6372	HPRT1	-0.417661	5.890255e-48	0.417661
10071	NEFM	-0.413668	4.959018e-48	0.413668
9918	NAPB	-0.410721	5.657968e-46	0.410721
3962	DNAJC6	-0.402443	2.798112e-44	0.402443
6734	INPP5F	-0.398787	1.926607e-43	0.398787
11167	PHYHIP	-0.397582	2.716215e-43	0.397582
17131	ZNF365	-0.392604	3.171319e-42	0.392604
7211	KIF5A	-0.389317	1.498999e-41	0.389317
12310	RBFOX1	-0.384733	1.190957e-40	0.384733
624	ANK3	-0.383734	2.194945e-40	0.383734
14866	SV2B	-0.382647	1.290225e-40	0.382647
472	AK5	-0.382184	3.370618e-40	0.382184
13915	SLC17A7	-0.380890	5.719902e-40	0.380890
15758	TPPP	-0.379299	1.823316e-39	0.379299
13335	RUNDC3A	-0.379271	1.847061e-39	0.379271

Top correlations are with negative sign. Therefore, we will select top 20 negative and top 20 positive correlations

In [8]:

```
corr_df_top_pos = corr_df_signif.nlargest(n_top_genes, "tau")
corr_df_top_pos
```

Out[8]:

	gene	tau	p_value	tau_module
15411	TMEM123	0.312777	1.385889e-27	0.312777
4156	DYNLT1	0.311182	3.712222e-27	0.311182
10542	ODC1	0.298466	4.480667e-25	0.298466
9284	MFSD1	0.293463	2.582179e-24	0.293463
6688	ILF2	0.292873	3.051344e-24	0.292873
15531	TMEM263	0.289802	7.983992e-24	0.289802
10668	OST4	0.289653	1.036997e-23	0.289653
13628	SERBP1	0.289482	1.101309e-23	0.289482
408	AGPAT5	0.288509	1.548881e-23	0.288509
3308	CSRP2	0.285597	3.483396e-23	0.285597
10669	OSTC	0.284981	5.282949e-23	0.284981
5693	GOLPH3	0.283820	7.886252e-23	0.283820
10923	PCNA	0.283701	8.215093e-23	0.283701
15384	TMED10	0.283323	9.355140e-23	0.283323
11715	PRDX4	0.280758	2.162969e-22	0.280758
13501	SCP2	0.280476	2.477162e-22	0.280476
16650	WLS	0.280097	2.317461e-22	0.280097
14267	SMIM15	0.279822	3.094493e-22	0.279822
9039	MAPK1IP1L	0.278985	4.110108e-22	0.278985
16990	ZNF121	0.278902	4.225975e-22	0.278902

In [9]:

```
corr_df_top_neg = corr_df_signif.nsmallest(n_top_genes, "tau")
corr_df_top_neg
```

Out[9]:

	gene	tau	p_value	tau_module
9986	NCOA7	-0.515115	2.811533e-71	0.515115
7120	KIAA0513	-0.498604	6.730395e-67	0.498604
16605	WDR7	-0.496619	2.212448e-66	0.496619
4145	DYNC1I1	-0.430025	1.863970e-50	0.430025
13888	SLC12A5	-0.421833	1.372029e-48	0.421833
6372	HPRT1	-0.417661	5.890255e-48	0.417661
10071	NEFM	-0.413668	4.959018e-48	0.413668
9918	NAPB	-0.410721	5.657968e-46	0.410721
3962	DNAJC6	-0.402443	2.798112e-44	0.402443
6734	INPP5F	-0.398787	1.926607e-43	0.398787
11167	PHYHIP	-0.397582	2.716215e-43	0.397582
17131	ZNF365	-0.392604	3.171319e-42	0.392604

	gene	tau	p_value	tau_module
7211	KIF5A	-0.389317	1.498999e-41	0.389317
12310	RBFOX1	-0.384733	1.190957e-40	0.384733
624	ANK3	-0.383734	2.194945e-40	0.383734
14866	SV2B	-0.382647	1.290225e-40	0.382647
472	AK5	-0.382184	3.370618e-40	0.382184
13915	SLC17A7	-0.380890	5.719902e-40	0.380890
15758	TPPP	-0.379299	1.823316e-39	0.379299
13335	RUNDC3A	-0.379271	1.847061e-39	0.379271

In [10]:

```
selected_genes_disc = list(corr_df_top_pos["gene"]) + list(corr_df_top_neg["gene"])
print(selected_genes_disc)
```

```
['TMEM123', 'DYNLT1', 'ODC1', 'MFSD1', 'ILF2', 'TMEM263', 'OST4', 'SERBP1', 'AGPAT
5', 'CSRP2', 'OSTC', 'GOLPH3', 'PCNA', 'TMED10', 'PRDX4', 'SCP2', 'WLS', 'SMIM15',
'MAPK1IP1L', 'ZNF121', 'NCOA7', 'KIAA0513', 'WDR7', 'DYNC1I1', 'SLC12A5', 'HPRT1',
'NEFM', 'NAPB', 'DNAJC6', 'INPP5F', 'PHYHIP', 'ZNF365', 'KIF5A', 'RBFOX1', 'ANK3',
'SV2B', 'AK5', 'SLC17A7', 'TPPP', 'RUNDC3A']
```

In [11]:

```
filtered_df_disc = df[["disease", "age", "gender"]+selected_genes_disc]
filtered_df_disc
```

Out[11]:

Unnamed: 0	disease	age	gender	TMEM123	DYNLT1	ODC1	MFSD1	ILF2	TMEM263	OST4	...
GSM855942	1	13.0	1.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	...
GSM855943	1	15.0	1.0	2.0	2.0	1.0	2.0	1.0	1.0	2.0	...
GSM855944	1	13.0	1.0	2.0	2.0	2.0	1.0	2.0	2.0	2.0	...
GSM855945	1	20.0	0.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	...
GSM855946	1	17.0	1.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	...
...	...	...	...	...	...	...	...	...	...	...	...
GSM492655_y	1	45.0	1.0	2.0	2.0	2.0	1.0	2.0	1.0	2.0	...
GSM492656_y	1	65.0	1.0	2.0	2.0	2.0	1.0	1.0	1.0	2.0	...
GSM525014_y	0	40.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...
GSM525015_y	0	60.0	0.0	2.0	1.0	2.0	1.0	1.0	1.0	1.0	...
GSM525016_y	0	43.0	1.0	2.0	2.0	1.0	1.0	1.0	1.0	2.0	...

1202 rows × 43 columns

In [12]:

```
filtered_df_disc.to_csv("../../data/oct5-oct11_2020/Merged-discret-top40.csv")
```

B. Continious

In [13]:

```
# Read Data

data_discrete = "../../data/oct5-oct11_2020/Merged-continue-dropNA.csv"
df_cont = pd.read_csv(data_discrete)
```

```
df_cont.index = df_cont["Unnamed: 0"]
df_cont = df_cont.drop(["Unnamed: 0"], axis=1)

df_cont = df_cont.T

df_cont["age"] = df_cont["age"].apply(lambda x: 0 if x=="<1" else float(x))
df_cont["disease"] = df_cont["disease/0"].apply(lambda x: int(x))
df_cont = df_cont.drop(["disease/0"], axis=1)

df_cont = df_cont.apply(pd.to_numeric)

df_cont.head()
```

Out[13]:

Unnamed: 0	gender	age	A1BG	A1CF	A2M	A2M-AS1	A2ML1	AA06	A/
GSM119615	1.0	63.0	-0.347768	-0.352081	0.162496	-0.185905	-0.298362	-0.294490	1.334
GSM119616	1.0	85.0	-0.282368	-0.274081	-0.141895	-0.217301	-0.298429	-0.309166	0.543
GSM119617	1.0	80.0	-0.181743	-0.075556	0.062192	-0.097256	-0.153840	0.034077	0.045
GSM119618	1.0	80.0	-0.153203	-0.170592	0.020673	-0.117320	-0.152809	-0.049684	0.273
GSM119619	0.0	102.0	-0.306879	-0.330848	0.076087	-0.275320	-0.325281	-0.319807	1.538

5 rows × 17495 columns

In [14]:

```
df = df_cont

all_vars = df.columns
clinical_vars = ["gender", "age", "disease"]

corr_dict = {"gene":[], "r":[], "p_value":[]}
for var in all_vars:
    if var not in clinical_vars:
        try:
            r, p_value = stats.spearmanr(list(df["disease"]), list(df[var]))
            corr_dict["r"].append(r)
            corr_dict["gene"].append(var)
            corr_dict["p_value"].append(p_value)
        except ValueError:
            print("Error: can't find correlation for {}".format(var))
corr_df = pd.DataFrame(corr_dict)
corr_df["r_module"] = corr_df["r"].apply(lambda x: x if x>0 else -x)


corr_df_signif=corr_df[corr_df["p_value"]<0.05]
print("{} out of {} significant correlations".format(len(corr_df_signif), len(corr_

corr_df_top = corr_df_signif.nlargest(n_top_genes, 'r_module')
corr_df_top
```

```
Error: can't find correlation for 1-Mar
Error: can't find correlation for 2-Mar
Error: can't find correlation for 1-Mar
Error: can't find correlation for 2-Mar
13129 out of 17488 significant correlations
```

Out[14]:

	gene	r	p_value	r_module
8782	LRRC42	0.523294	2.151736e-85	0.523294

	gene	r	p_value	r_module
14255	SMEK2	0.498095	2.646110e-76	0.498095
8993	MAP1A	-0.495896	1.514149e-75	0.495896
10311	NPHP3-AS1	-0.492268	2.619625e-74	0.492268
11847	PRR7-AS1	-0.485413	5.212613e-72	0.485413
12629	RNF187	-0.483453	2.317183e-71	0.483453
17131	ZNF365	-0.481479	1.030678e-70	0.481479
14244	SMC4	0.481329	1.153775e-70	0.481329
12436	RFC2	0.480737	1.801556e-70	0.480737
9783	MYC	0.479042	6.418584e-70	0.479042
12198	RAD51AP1	0.476673	3.744897e-69	0.476673
4462	EPB41L1	-0.476312	4.892985e-69	0.476312
16861	ZCCHC9	0.475019	1.272808e-68	0.475019
10744	PAGE4	-0.472936	5.884412e-68	0.472936
14035	SLC30A5	0.472473	8.257425e-68	0.472473
10174	NIP7	0.471690	1.462761e-67	0.471690
4643	EZH2	0.470710	2.986869e-67	0.470710
13888	SLC12A5	-0.469635	6.519944e-67	0.469635
15758	TPPP	-0.468355	1.644639e-66	0.468355
624	ANK3	-0.468348	1.652736e-66	0.468348

In the continuous case, there are positive and negative correlations within the top 20. Therefore, we will select top 40 strongest correlations

In [16]:

```
corr_df_top = corr_df_signif.nlargest(40, 'r_module')
selected_genes = list(corr_df_top["gene"])
filtered_df_cont = df[["disease", "age", "gender"]+selected_genes]
filtered_df_cont.to_csv("../../data/oct5-oct11_2020/Merged-continue-top40.csv")
filtered_df_cont
```

Out[16]:

Unnamed: 0	disease	age	gender	LRRC42	SMEK2	MAP1A	NPHP3-AS1	PRR7-AS1	RN
GSM119615	0	63.0	1.0	-0.284011	-0.026434	0.804039	-0.336354	-0.310325	0.04
GSM119616	0	85.0	1.0	-0.269954	-0.123603	2.227096	-0.307239	-0.297585	0.27
GSM119617	0	80.0	1.0	-0.170731	-0.102242	1.090534	-0.157733	-0.146449	-0.03
GSM119618	0	80.0	1.0	-0.183907	-0.119025	2.356501	-0.190443	-0.186968	-0.0
GSM119619	0	102.0	0.0	-0.234011	-0.093423	0.558703	-0.327920	-0.314601	0.19
...	...	...	...	...	...	...	...	...	
GSM492665	1	78.0	0.0	0.027583	-0.007285	0.134505	-0.241592	-0.315923	-0.03
GSM492666	1	69.0	1.0	-0.093116	-0.068750	-0.004373	-0.299227	-0.298629	-0.08
GSM525014_y	0	40.0	0.0	-0.183938	-0.080257	0.659879	-0.320204	-0.292715	0.04

Unnamed: 0	disease	age	gender	LRRC42	SMEK2	MAP1A	NPHP3-AS1	PRR7-AS1	RN
GSM525015_y	0	60.0	0.0	-0.147505	-0.050501	0.411707	-0.305972	-0.302378	0.0
GSM525016_y	0	43.0	1.0	-0.149306	-0.068517	0.796029	-0.325183	-0.291170	-0.00

1201 rows × 43 columns

In [ ]:

Questions

- "Disease" distribution is imbalanced. Do we need to do anything with this?

In [ ]: