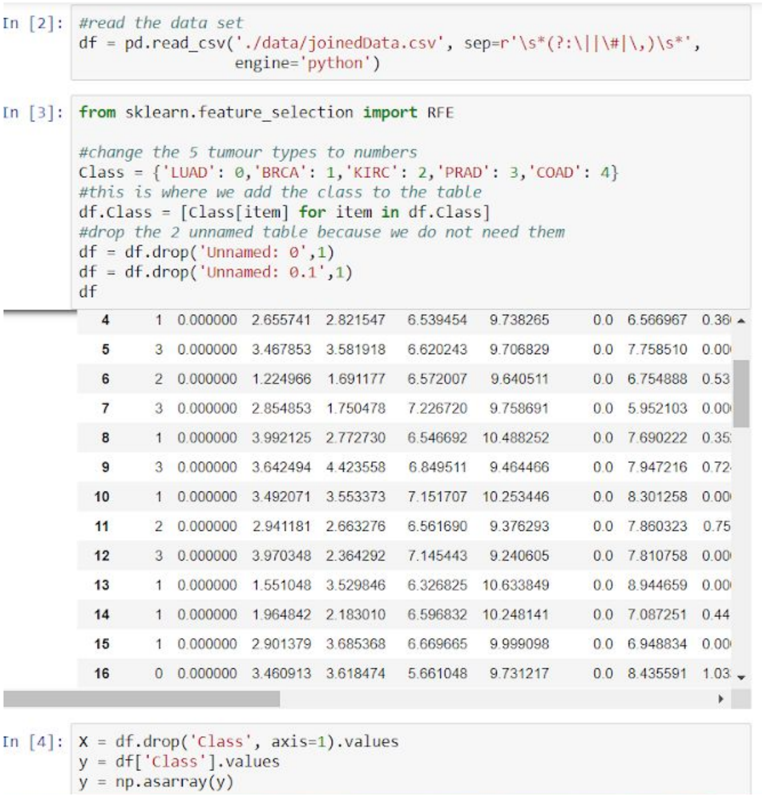
General

The current prototype that is working are all three classifiers, the KNN, DNN and OVR. The author had used tensorflow and sklearn website and libraries to achieve this, apart from numpy, pandas etc. She used the sample codes that was provided by the websites tensorflow and sklearn and changed it a bit to suit the authors needs. The reading in, standardizing and splitting the data sets are the authors work. She will then try to use less built in features and create her own code.

One-Vs-Rest

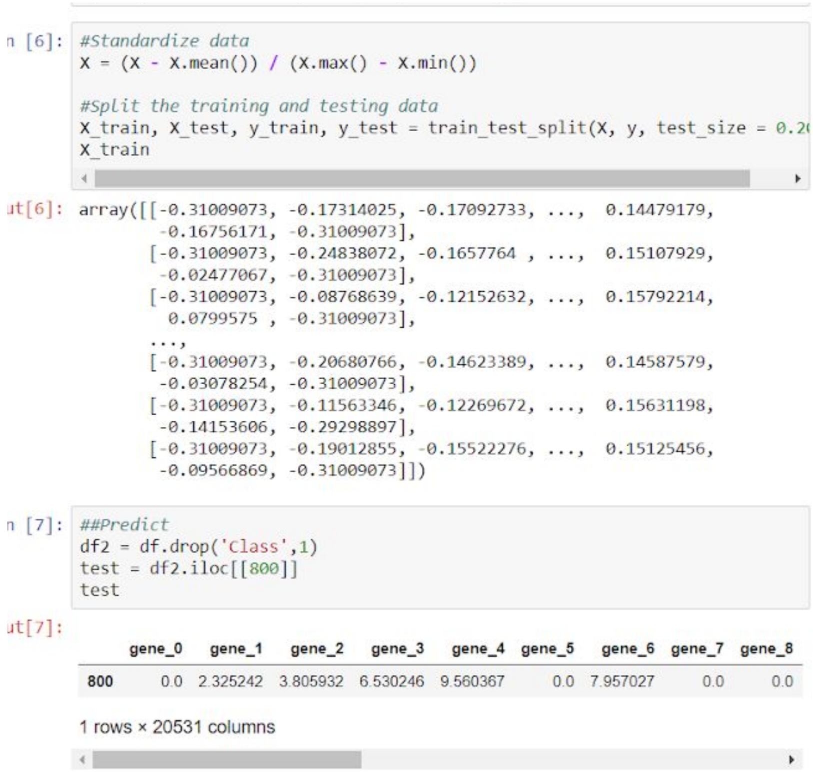


```
In [2]: #read the data set
        df = pd.read_csv('./data/joinedData.csv', sep=r'\s*(?:\||\#|\,)\s*',
                         engine='python')
```

```
In [3]: from sklearn.feature_selection import RFE

        #change the 5 tumour types to numbers
        Class = {'LUAD': 0,'BRCA': 1,'KIRC': 2,'PRAD': 3,'COAD': 4}
        #this is where we add the class to the table
        df.Class = [Class[item] for item in df.Class]
        #drop the 2 unnamed table because we do not need them
        df = df.drop('Unnamed: 0',1)
        df = df.drop('Unnamed: 0.1',1)
        df
```

```
 4   1  0.000000  2.655741  2.821547   6.539454   9.738265    0.0  6.566967  0.36
 5   3  0.000000  3.467853  3.581918   6.620243   9.706829    0.0  7.758510  0.00
 6   2  0.000000  1.224966  1.691177   6.572007   9.640511    0.0  6.754888  0.53
 7   3  0.000000  2.854853  1.750478   7.226720   9.758691    0.0  5.952103  0.00
 8   1  0.000000  3.992125  2.772730   6.546692  10.488252    0.0  7.690222  0.35
 9   3  0.000000  3.642494  4.423558   6.849511   9.464466    0.0  7.947216  0.72
10   1  0.000000  3.492071  3.553373   7.151707  10.253446    0.0  8.301258  0.00
11   2  0.000000  2.941181  2.663276   6.561690   9.376293    0.0  7.860323  0.75
12   3  0.000000  3.970348  2.364292   7.145443   9.240605    0.0  7.810758  0.00
13   1  0.000000  1.551048  3.529846   6.326825  10.633849    0.0  8.944659  0.00
14   1  0.000000  1.964842  2.183010   6.596832  10.248141    0.0  7.087251  0.44
15   1  0.000000  2.901379  3.685368   6.669665   9.999098    0.0  6.948834  0.00
16   0  0.000000  3.460913  3.618474   5.661048   9.731217    0.0  8.435591  1.03
```

```
In [4]: X = df.drop('Class', axis=1).values
        y = df['Class'].values
        y = np.asarray(y)
```



```
n [6]: #Standardize data
       X = (X - X.mean()) / (X.max() - X.min())

       #Split the training and testing data
       X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2(
       X_train
```

```
ut[6]: array([[-0.31009073, -0.17314025, -0.17092733, ...,  0.14479179,
                -0.16756171, -0.31009073],
               [-0.31009073, -0.24838072, -0.1657764 , ...,  0.15107929,
                -0.02477067, -0.31009073],
               [-0.31009073, -0.08768639, -0.12152632, ...,  0.15792214,
                 0.0799575 , -0.31009073],
               ...,
               [-0.31009073, -0.20680766, -0.14623389, ...,  0.14587579,
                -0.03078254, -0.31009073],
               [-0.31009073, -0.11563346, -0.12269672, ...,  0.15631198,
                -0.14153606, -0.29298897],
               [-0.31009073, -0.19012855, -0.15522276, ...,  0.15125456,
                -0.09566869, -0.31009073]])
```

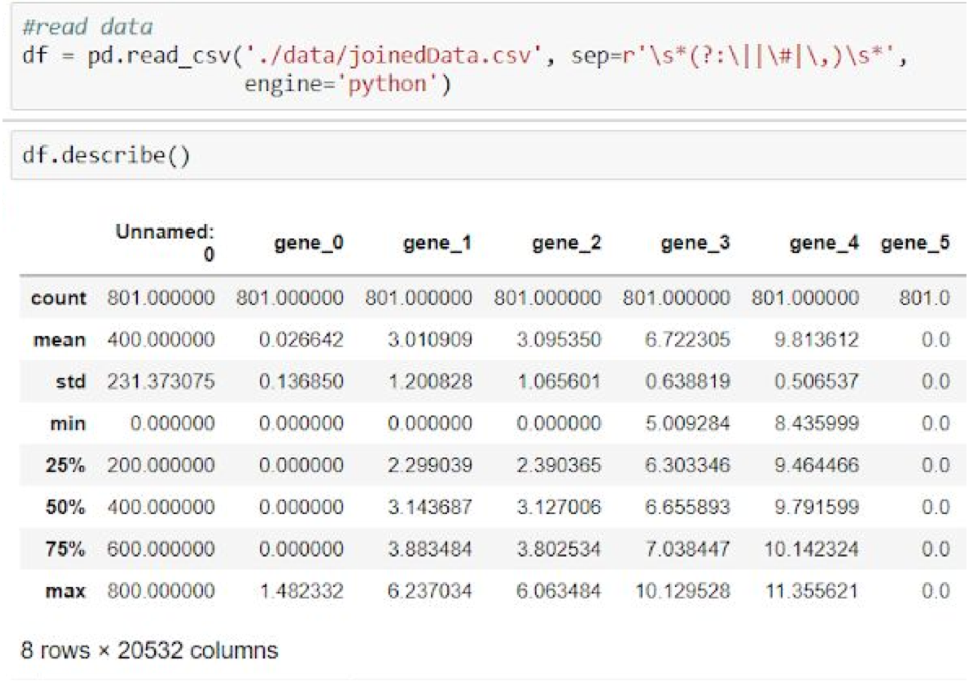
```
n [7]: ##Predict
       df2 = df.drop('Class',1)
       test = df2.iloc[[800]]
       test
```

ut[7]:

	gene_0	gene_1	gene_2	gene_3	gene_4	gene_5	gene_6	gene_7	gene_8
800	0.0	2.325242	3.805932	6.530246	9.560367	0.0	7.957027	0.0	0.0

1 rows × 20531 columns

K-Nearest Neighbour

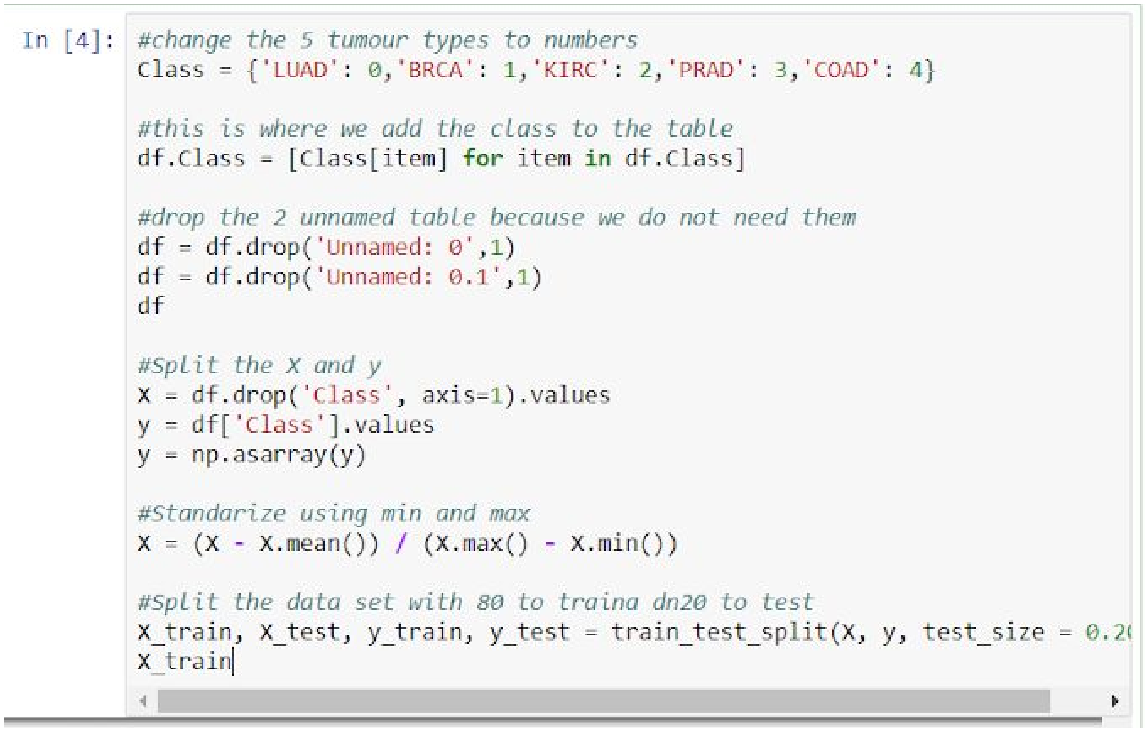


```
#read data
df = pd.read_csv('./data/joinedData.csv', sep=r'\s*(?:\||\#|\,)\s*',
                 engine='python')
```

```
df.describe()
```

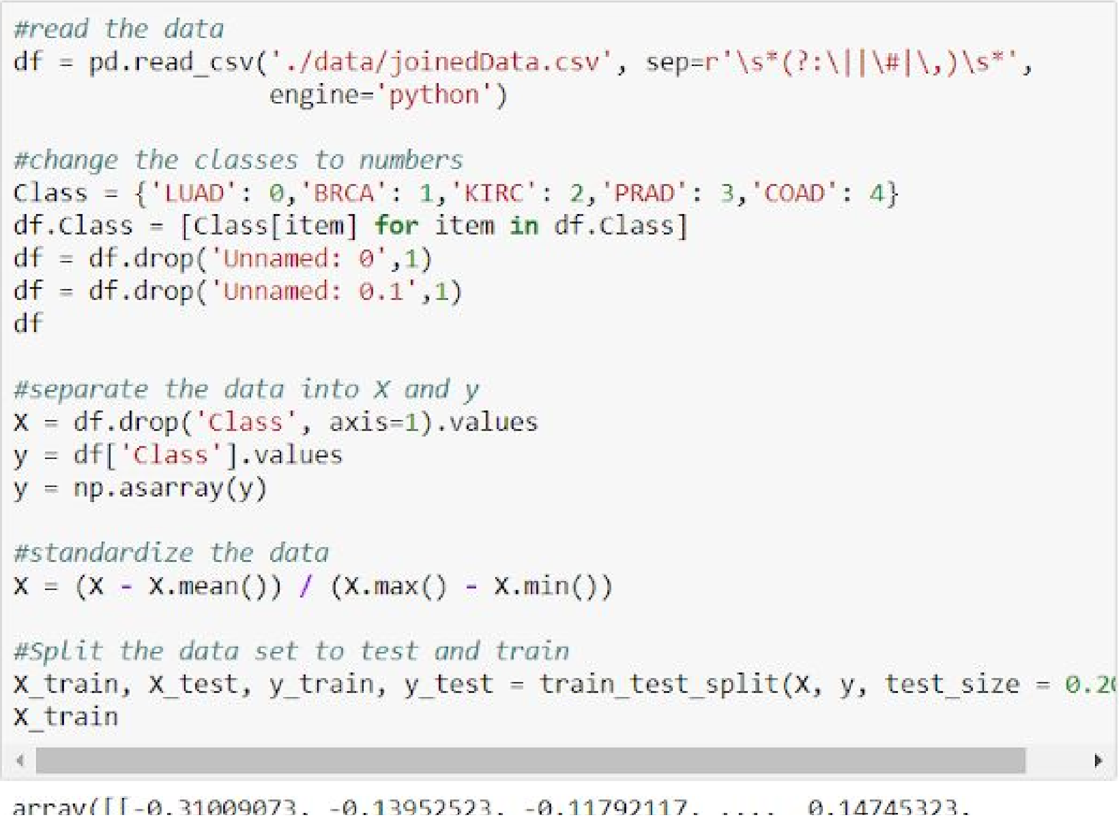
	Unnamed: 0	gene_0	gene_1	gene_2	gene_3	gene_4	gene_5
count	801.000000	801.000000	801.000000	801.000000	801.000000	801.000000	801.0
mean	400.000000	0.026642	3.010909	3.095350	6.722305	9.813612	0.0
std	231.373075	0.136850	1.200828	1.065601	0.638819	0.506537	0.0
min	0.000000	0.000000	0.000000	0.000000	5.009284	8.435999	0.0
25%	200.000000	0.000000	2.299039	2.390365	6.303346	9.464466	0.0
50%	400.000000	0.000000	3.143687	3.127006	6.655893	9.791599	0.0
75%	600.000000	0.000000	3.883484	3.802534	7.038447	10.142324	0.0
max	800.000000	1.482332	6.237034	6.063484	10.129528	11.355621	0.0

8 rows × 20532 columns



```
#change the 5 tumour types to numbers
Class = {'LUAD': 0,'BRCA': 1,'KIRC': 2,'PRAD': 3,'COAD': 4}

#this is where we add the class to the table
df.Class = [Class[item] for item in df.Class]

#drop the 2 unnamed table because we do not need them
df = df.drop('Unnamed: 0',1)
df = df.drop('Unnamed: 0.1',1)
df

#Split the X and y
X = df.drop('Class', axis=1).values
y = df['Class'].values
y = np.asarray(y)

#Standarize using min and max
X = (X - X.mean()) / (X.max() - X.min())

#Split the data set with 80 to traina dn20 to test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2
X_train
```

Neural Networks



```
#read the data
df = pd.read_csv('./data/joinedData.csv', sep=r'\s*(?:\||\#|\,)\s*',
                 engine='python')

#change the classes to numbers
Class = {'LUAD': 0,'BRCA': 1,'KIRC': 2,'PRAD': 3,'COAD': 4}
df.Class = [Class[item] for item in df.Class]
df = df.drop('Unnamed: 0',1)
df = df.drop('Unnamed: 0.1',1)
df

#separate the data into X and y
X = df.drop('Class', axis=1).values
y = df['Class'].values
y = np.asarray(y)

#standardize the data
X = (X - X.mean()) / (X.max() - X.min())

#Split the data set to test and train
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2
X_train
```

```
array([[-0.31009073, -0.13952523, -0.11792117, ...,  0.14745323,
```