About the project

The end goal of this project is to classify patients with high protein concentration in urine and the healthy group based on SERS (Surface Enchanced Raman Spectroscopy) spectral data and biomedical data.
This project is to be released as a research paper later in 2022 or 2023. Some information might not be fully shown here as a result.

The project is divided into several Jupyter notebooks with the following names: 1) Import raw urine spectra (part 1) 2) Spectra processing (part 2) 3) Classification of patients (part 3) 4) Biomedical data (part 4) 5) Comparison of nanoparticles (part 5)

Author of all codes: Sultan Aitekenov, sultanaitekenov@gmail.com

Part of the upcoming abstract: Excessive protein excretion in human urine is an early and sensitive marker of diabetic nephropathy, primary and secondary renal disease. Kidney problems, particularly chronic kidney disease, remain among the few growing causes of mortality in the world. Therefore, it is important to develop efficient, expressive, and low-cost method for protein determination. Surface enhanced Raman spectroscopy (SERS) methods are potential candidates to achieve those criteria. In this paper, the SERS methods was developed to distinguish patients with proteinuria and the healthy group. Commercial gold nanoparticles with the diameter of 60 nm and 100 nm, and silver nanoparticles with the diameter of 100 nm were employed. Silver, gold, silicon and test slides covered with aluminium tape were utilized as substrates. Obtained spectra were analysed with several machine learning algorithms coupled with the PCA, ROC curve, and cross-validation methods.

Comparison of nanoparticles (part 5)

Motivation of this part is to compare spectra of 40 nm, 60 nm and 100 nm Au NPs on the gold substrate. 60 nm and 100 nm Au NPs might perform better. Data for the experimental set with 40 nm is limited to the first 10 patients. Patients' IDs coincide with their numbering.

Import data

Import modules

In [1]:
```
# other modules related to classification are imported later
import pandas as pd
import numpy as np
import copy
import pickle
import matplotlib.pyplot as plt
```

Raman Shift

In [2]:
```
raman_shift_400_1800=np.array(pd.read_csv('raman_shift_400_1800.csv', header=None))
wave = raman_shift_400_1800[0]
```

Processed urine spectra

In [3]:
```
# data contains a nested dictionary
f = open('processed_urine_spectra.pkl', 'rb')
processed_urine_spectra = pickle.load(f)
```

In [4]:
```
# create empty dict
comparison_spectra_AuNPs = {}


for key_set in processed_urine_spectra.keys():

    if key_set == 'Au_40nm_AuNPs' or key_set == 'Au_60nm_AuNPs' or key_set == 'Au_100r
        # create empty matrix to assign to them values later
        matrix = []

         # loop to make dict into matrix
        for key_ID in processed_urine_spectra[key_set].keys():

            if key_ID <= 10:
                    matrix.append(processed_urine_spectra[key_set][key_ID])

        # write matrix into target dictionary
        comparison_spectra_AuNPs[key_set] = np.array( matrix )
```

In [5]:
```
comparison_spectra_AuNPs.keys()
```

Out[5]:
```
dict_keys(['Au_60nm_AuNPs', 'Au_100nm_AuNPs', 'Au_40nm_AuNPs'])
```

In [6]:
```
comparison_spectra_AuNPs['Au_60nm_AuNPs'].shape
```

Out[6]:
```
(10, 2045)
```

In [7]:
```
len(comparison_spectra_AuNPs['Au_60nm_AuNPs'])
```

Out[7]:
```
10
```

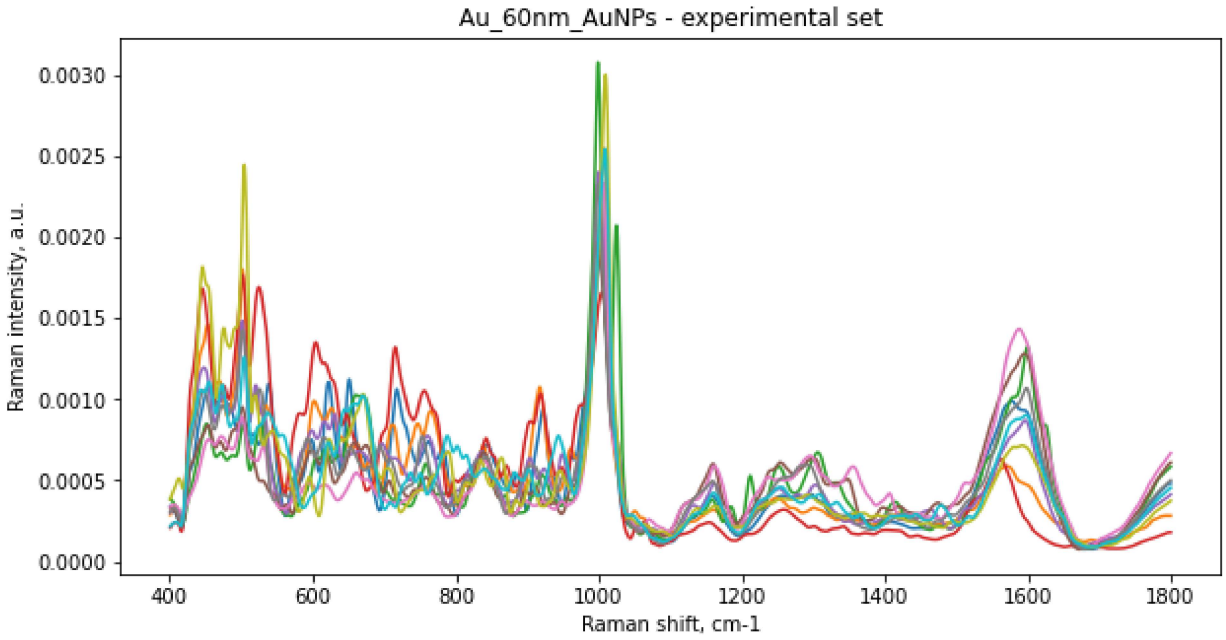
In [8]:
```
comparison_spectra_AuNPs['Au_60nm_AuNPs'][0]
```

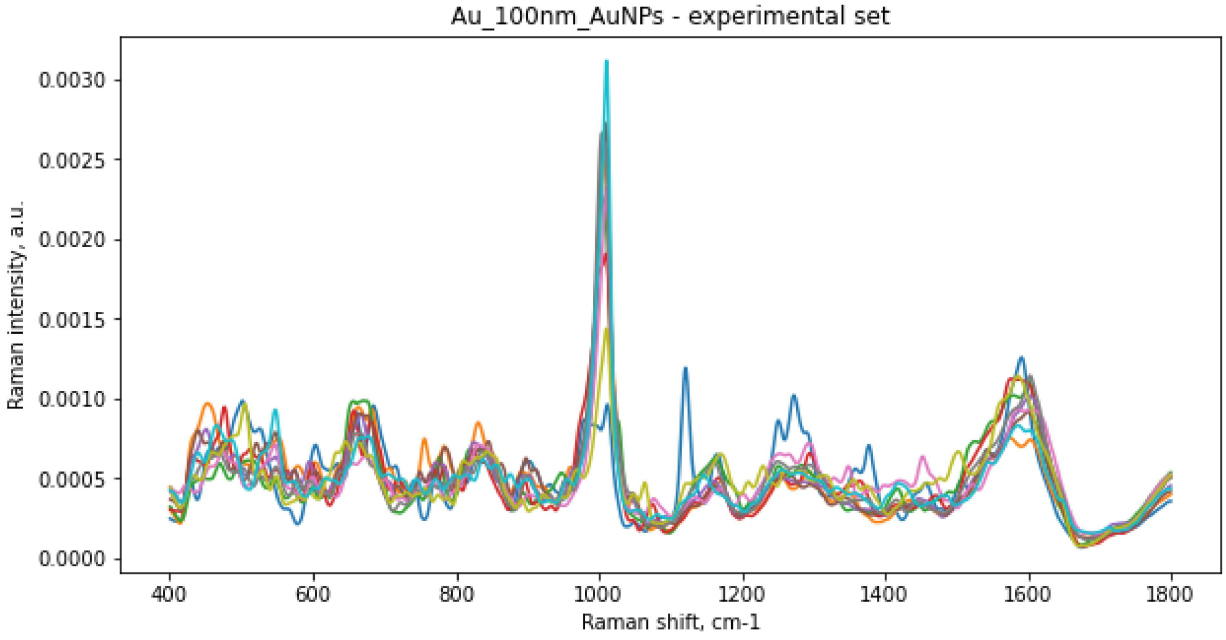
Out[8]:
```
array([0.00034195, 0.00033841, 0.00033458, ..., 0.00048184, 0.00048225,
       0.00048246])
```

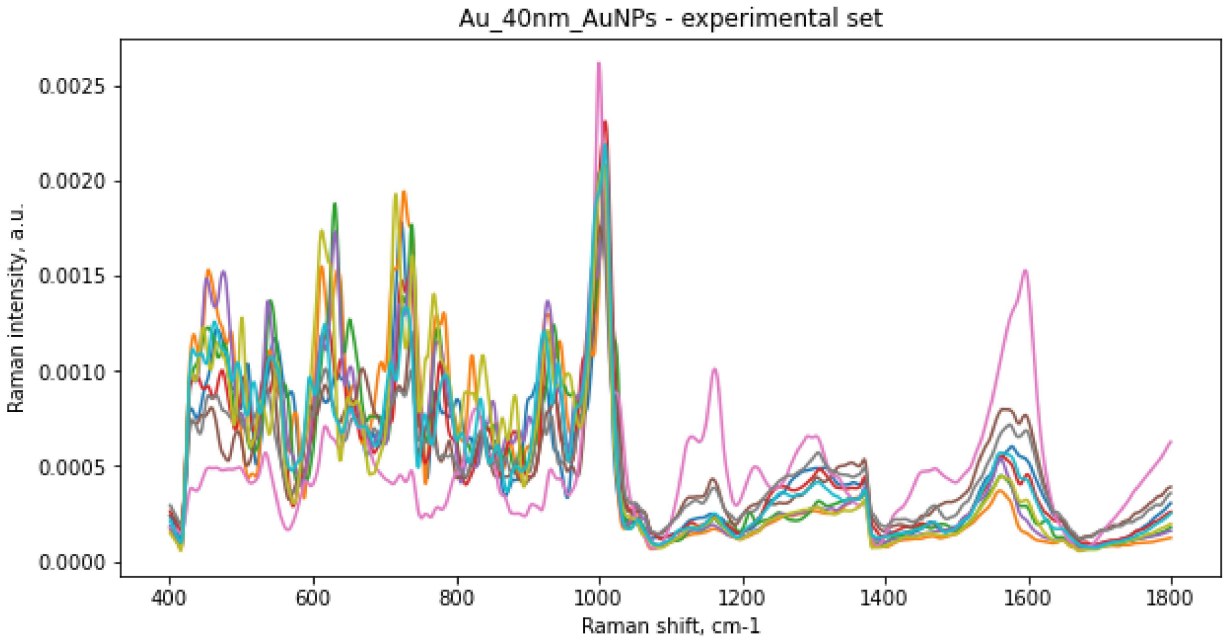
Create spectra plots

In [9]:
```
# Combine every plot for each patient
# plots every spectra for one experimental set as defined in exp_set
for key_set in comparison_spectra_AuNPs.keys():
```

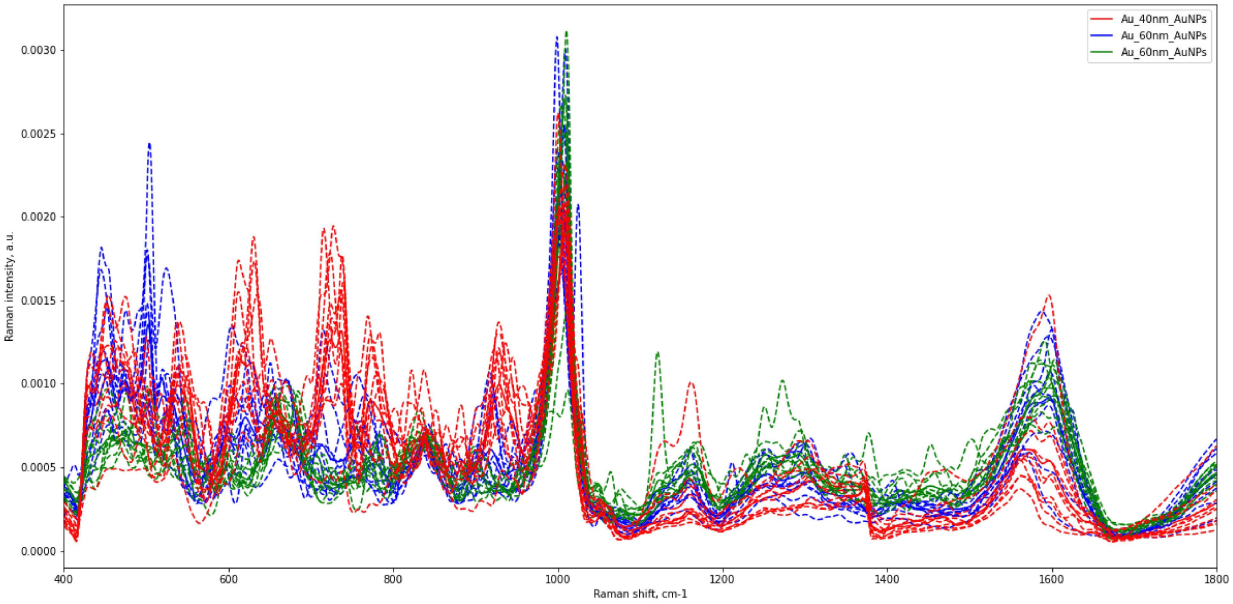
```
    plt.figure(figsize =(10,5))
    plt.xlabel('Raman shift, cm-1')
    plt.ylabel('Raman intensity, a.u.')
    plt.title(f'{key_set} - experimental set')
    for i in range( len(comparison_spectra_AuNPs['Au_60nm_AuNPs']) ):
        plt.plot(wave, comparison_spectra_AuNPs[key_set][i])
```





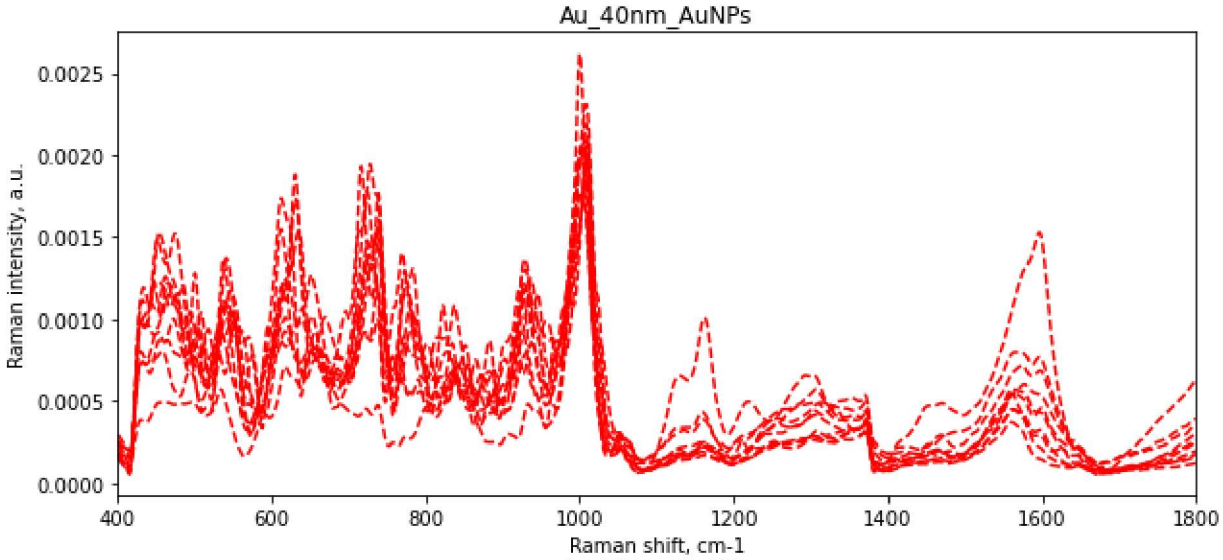


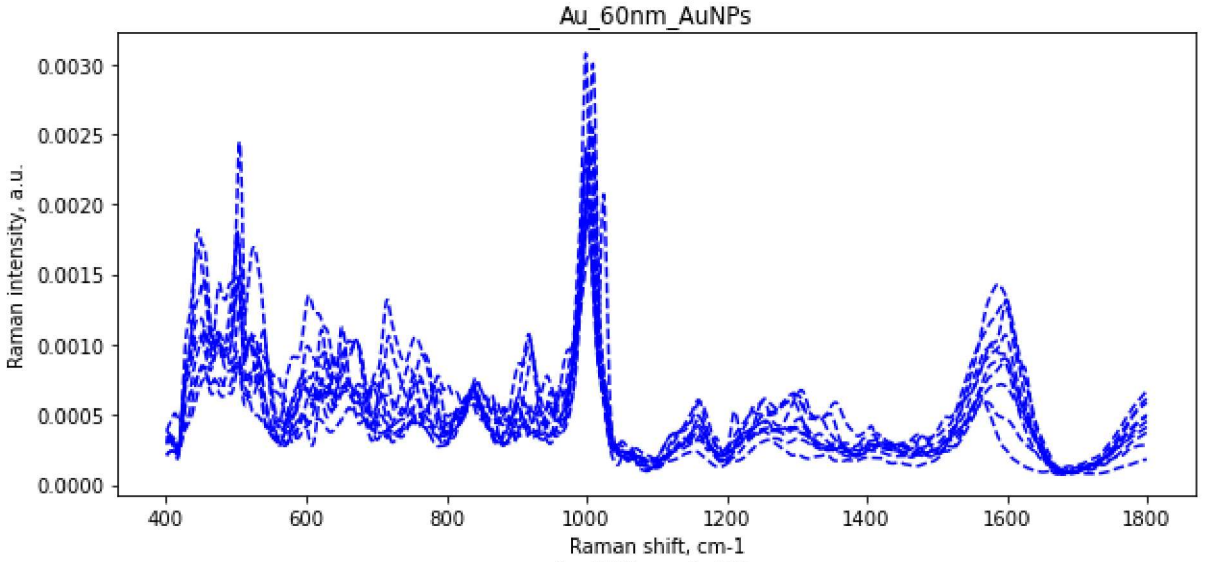
In [19]:
```
# Combine every plot for each patient
# plots every spectra for one experimental set as defined in exp_set

plt.figure(figsize =(20,10))
plt.xlabel('Raman shift, cm-1')
plt.ylabel('Raman intensity, a.u.')

# plot mean spectra
plt.plot(wave, np.mean(comparison_spectra_AuNPs['Au_40nm_AuNPs'], axis=0), 'r',
        wave, np.mean(comparison_spectra_AuNPs['Au_60nm_AuNPs'], axis=0), 'b',
        wave, np.mean(comparison_spectra_AuNPs['Au_100nm_AuNPs'], axis=0), 'g')

# plot individual spectra
for key_set in comparison_spectra_AuNPs.keys():
    if key_set == 'Au_40nm_AuNPs':
        for i in range( len(comparison_spectra_AuNPs[key_set]) ):
            plt.plot(wave, comparison_spectra_AuNPs[key_set][i], 'r--')
    elif key_set == 'Au_60nm_AuNPs':
        for i in range( len(comparison_spectra_AuNPs[key_set]) ):
            plt.plot(wave, comparison_spectra_AuNPs[key_set][i], 'b--')
    elif key_set == 'Au_100nm_AuNPs':
        for i in range( len(comparison_spectra_AuNPs[key_set]) ):
            plt.plot(wave, comparison_spectra_AuNPs[key_set][i], 'g--')

# labels and xlim
plt.legend(labels=['Au_40nm_AuNPs', 'Au_60nm_AuNPs', 'Au_60nm_AuNPs'])
plt.xlim([400,1800]);
```

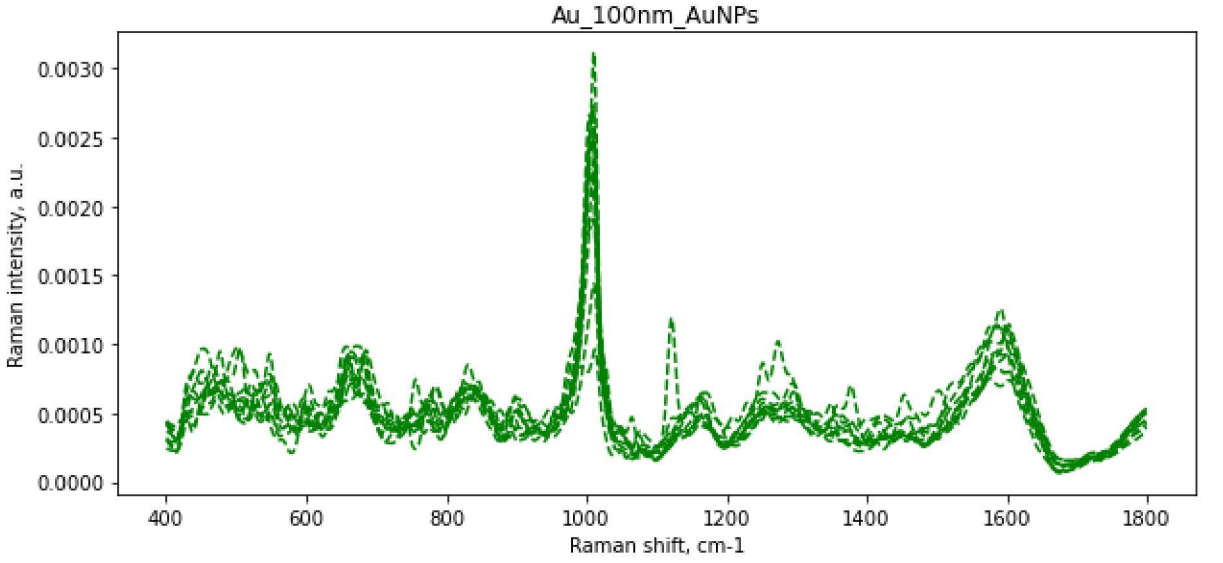


In [29]:
```
# Combine every plot for each patient
# plots every spectra for one experimental set as defined in exp_set

plt.figure(figsize =(10,15))


# plot mean spectra
plt.plot(wave, np.mean(comparison_spectra_AuNPs['Au_40nm_AuNPs'], axis=0), 'r',
        wave, np.mean(comparison_spectra_AuNPs['Au_60nm_AuNPs'], axis=0), 'b',
        wave, np.mean(comparison_spectra_AuNPs['Au_100nm_AuNPs'], axis=0), 'g')

# plot individual spectra
for key_set in comparison_spectra_AuNPs.keys():
    if key_set == 'Au_40nm_AuNPs':

        # define subplot
        plt.subplot(3,1,1)
        plt.xlabel('Raman shift, cm-1')
        plt.ylabel('Raman intensity, a.u.')
        plt.title(key_set)

        for i in range( len(comparison_spectra_AuNPs[key_set]) ):
            plt.plot(wave, comparison_spectra_AuNPs[key_set][i], 'r--')


    elif key_set == 'Au_60nm_AuNPs':

        # define subplot
        plt.subplot(3,1,2)
        plt.xlabel('Raman shift, cm-1')
        plt.ylabel('Raman intensity, a.u.')
        plt.title(key_set)

        for i in range( len(comparison_spectra_AuNPs[key_set]) ):
            plt.plot(wave, comparison_spectra_AuNPs[key_set][i], 'b--')


    elif key_set == 'Au_100nm_AuNPs':

        # define subplot
```

```
        plt.subplot(3,1,3)
        plt.xlabel('Raman shift, cm-1')
        plt.ylabel('Raman intensity, a.u.')
        plt.title(key_set)

        for i in range( len(comparison_spectra_AuNPs[key_set]) ):
            plt.plot(wave, comparison_spectra_AuNPs[key_set][i], 'g--')



plt.xlim([400,1800]);
```







In summary, from the graph above, a visual inspection shows that 40 nm Au NPs perform worse than 60 nm or 100 nm Au NPs because signal to noise ratio is higher. It is easy to see this claim if the peak at 1000 cm-1 is compared with the left side of the raman shift.