機器學習概論 HW1 Report

Part1

Split the train and test data

Using pandas to finish csv files editing

```
def read_and_split(filename):
    file = pd.read_csv(filename, header=None)
    file.columns = ['Label','Alcohol',' Malic acid','Ash',' Alcalinity of ash','Ma

    type1_filt = (file['Label'] == 1)
    type2_filt = (file['Label'] == 2)
    type3_filt = (file['Label'] == 3)

    type1 = file.loc[type1_filt]
    type2 = file.loc[type2_filt]
    type3 = file.loc[type3_filt]

    type1_test = type1.sample(n=18, frac=None ,random_state=200)
    type1_train = type1.drop(type1_test.index)
    type2_test = type2.sample(n=18, frac=None ,random_state=200)
    type2_train = type2.drop(type2_test.index)
    type3_test = type3.sample(n=18, frac=None ,random_state=200)
    type3_train = type3.drop(type3_test.index)

    training_set = pd.concat([type1_train, type2_train, type3_train])
    testing_set = pd.concat([type1_test, type2_test, type3_test])

    training_set.to_csv('training_set.csv')
    testing_set.to_csv('testing_set.csv')

    return training_set, testing_set
```

Part2

Seperate each feature into a group with Label

SPLIT INTO X AND Y

```
x_dataset = dataset[:, 1:] #features
y_dataset = dataset[:, 0] # Label
```

CLASSIFY INTO LABELS

```
for i in range(len(y_dataset)):
    if(y_dataset[i] == 1):
        type1.append(x_dataset[i])

    elif(y_dataset[i] == 2):
        type2.append(x_dataset[i])

    elif(y_dataset[i] == 3):
        type3.append(x_dataset[i])
```

For each feature, it has index for its label

```
type1_ft = np.zeros(shape=[13, len(type1)])
type2_ft = np.zeros(shape=[13, len(type2)])
type3_ft = np.zeros(shape=[13, len(type3)])
for idx, ele in enumerate(x_dataset[0:len(type1)]):
    for i in range(13):
        type1_ft[i][idx] = x_dataset[idx][i]
for idx, ele in enumerate(x_dataset[len(type1):len(type1)+len(type2)]):
    for i in range(13):
        type2_ft[i][idx] = x_dataset[len(type1)+idx][i]
for idx, ele in enumerate(x_dataset[len(type1)+len(type2):len(y_dataset)]):
    for i in range(13):
        type3_ft[i][idx] = x_dataset[len(type1)+len(type2)+idx][i]
```

Calculate mean, standard and normal distribution

Get 13 features mean, standard and normal distribution with numpy and synic.stats

$$f(x) = \frac{\exp(-x^2/2)}{\sqrt{2\pi}}$$

```
for i in range(13):
    type1_mean.append(np.mean(type1[i]))
    type2_mean.append(np.mean(type2[i]))
    type3_mean.append(np.mean(type3[i]))
```

```
for i in range(13):
    type1_std.append(np.std(type1[i]))
    type2_std.append(np.std(type2[i]))
    type3_std.append(np.std(type3[i]))
```

```
    for i in range(13):
        type1_norm.append(st.norm(type1_mean[i], type1_std[i]))
        type2_norm.append(st.norm(type2_mean[i], type2_std[i]))
        type3_norm.append(st.norm(type3_mean[i], type3_std[i]))
```

Prior

The distrubution of Labels will be the prior distribution

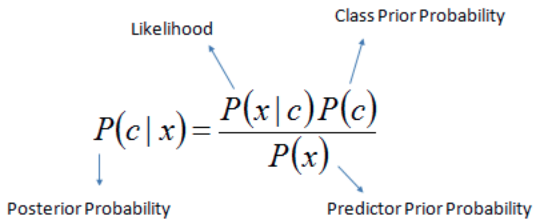
```
prior[0] = len(type1) / len(y_dataset) #0.33
prior[1] = len(type2) / len(y_dataset) #0.42
prior[2] = len(type3) / len(y_dataset) #0.24
```

Likelihood

$$\mathrm{argmax}_\theta \frac{1}{h} \int_{x_j}^{x_j+h} f(x \mid \theta)\, dx,$$

```
likelihood = sc.integrate.quad(dis[label][i].pdf, data[i+1], data[i+1]+delta )[0]
```

Posterior



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

```
post[label] = 1. * prior[label]
```

```
post[label] = post[label]*likelihood
```

MAP

```
predict = np.argmax(post)
    total+=1
    if predict == (data[0] -1):
        correct += 1
    else:
        pass
print('accuracy: ', correct/total)
```

Part3

Dimension 13 features into 2 & 3 features

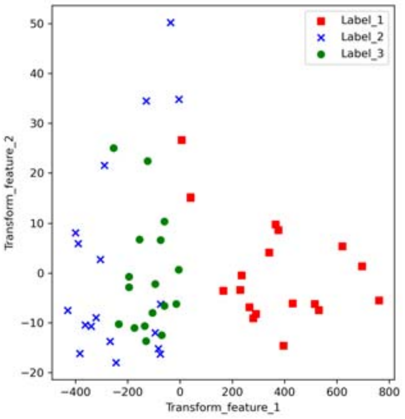
```
X_p = pca2.fit(X).transform(X)
```

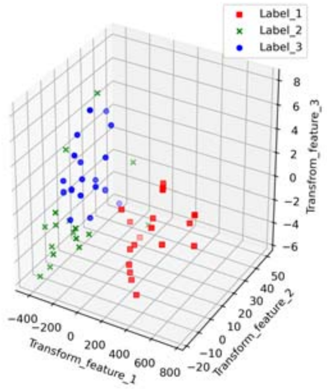
```
X_p = pca3.fit(X).transform(X)
```

Visualization

```
for c, i, target_name, m in zip('rbg', labels, types, markers):
        plt2D.scatter(X_p[y==i, 0], X_p[y==i, 1], c=c, label=target_name, marker=m
```

```
for c, i, target_name, m in zip('rgb', labels, types, markers):
        plt3D.scatter(X_p[y==i, 0], X_p[y==i, 1], X_p[y==i, 2], c=c, label=target_
```





Part4

Discussion with different Prior

First, I just try serveral cases to observe the difference with each of them.

The more labels approaches to 0 , the worse accuracy will be.

In the other hand, be more close to the training distribution of 3 labels, accuracy will be more higher.

```
#acc = 0.96
    """
    prior[0] = 0.05
    prior[1] = 0.05
    prior[2] = 0.9
    """
    #acc = 0.87
    """
    prior[0] = 0.9
    prior[1] = 0.09999999999
    prior[2] = 0.00000000001
    """
    #acc = 0.57
    """
    prior[0] = 0.000000000005
    prior[1] = 0.99999999999
    prior[2] = 0.000000000005
    """
    #acc = 0.33
    """
    prior[0] = 0.0
    prior[1] = 0.1
    prior[2] = 0.0
    """
```

Discover most important factors

Then I check critical roles in 13 features, Proline & Magnesium are 2 highest metric affect results.

```
pd.DataFrame(pca2.components_,columns=X.columns,index = ['Transform_feature_1','Tr
```

	Alcohol	Malic acid
Transform_feature_1	0.001404923002591525	-0.00094
Transform_feature_2	-0.002664669843727335	-0.00675

Proline ,Magnesium, and Alcalinity of ash are 3 highest metric

```
pd.DataFrame(pca3.components_,columns=X.columns,index = ['Transform_feature_1','Tr
```

	Alcohol	Malic acid
Transform_feature_1	0.001404923002591525	-0.00094
Transform_feature_2	-0.002664669843727335	-0.00675
Transform_feature_3	0.04599422668581956	0.138707

Just use three most important features to calculate MAP

accuracy = 0.81

```
for i in range(3):
    if i == 0:
        # 13th feature
        i = 13-1
    elif i == 1:
        # 5th feature
        i = 5-1
    else:
        # 4th feature
        i = 4-1
    likelihood = sc.integrate.quad(dis[label][i].pdf, data[i+1], data[i+1]+delta )
    post[label] = post[label]*likelihood
```

We still have over 80% accuracy