PROJECT REPORT IDS
WINE CLASSIFICATION

Group Members:

Keshav Maheshwari, 19ucs060
Tapomay Singh Khatri, 19ucs064
Prasanna Mishra, 19ucs005
Vishnu Bhat, 19ucs058

INTRODUCTION

This is our group project for the IDS course. We have chosen a dataset that performs chemical analysis to determine the origin of wines. The dataset was chosen from https://archive.ics.uci.edu/ml/datasets/Wine

Refer github link for the Implemented code

https://github.com/gmmkmtgk/IDS_Project/tree/main

OBJECTIVE

Applying different ML Classification algorithms on the data set and getting inferences from the data using Python.

SYSTEM REQUIREMENTS

1. Python3 needs to be installed on the PC.
2. Important statistical libraries such as NumPy, pandas, scikit-learn, mathplotlib, etc.
3. The code can be run in Jupyter-notebook or VS Code if installed otherwise, Google collaboratory can be used.

PART 1: DATA SET INFORMATION

These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The data is Multivariate.

The attributes are:

1) Alcohol

8) Nonflavanoid phenols

2) Malic acid

3) Ash

4) Alkalinity of ash

5) Magnesium

6) Total phenols

7) Flavanoids

9) Proanthocyanins

10)Color intensity

11)Hue

12)OD280/OD315 of diluted wines

13)Proline

```
In [4]: wine=pd.read_csv('wine.csv')
```

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD315 of diluted wines	Proline
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
173	3	13.71	5.65	2.45	20.5	95	1.68	0.61	0.52	1.06	7.70	0.64	1.74	740
174	3	13.40	3.91	2.48	23.0	102	1.80	0.75	0.43	1.41	7.30	0.70	1.56	750
175	3	13.27	4.28	2.26	20.0	120	1.59	0.69	0.43	1.35	10.20	0.59	1.56	835
176	3	13.17	2.59	2.37	20.0	120	1.65	0.68	0.53	1.46	9.30	0.60	1.62	840
177	3	14.13	4.10	2.74	24.5	96	2.05	0.76	0.56	1.35	9.20	0.61	1.60	560

178 rows × 14 columns

Fig 1. Overview of Training Set of Wine Dataset in Jupyter Notebook

PART 2: ATTRIBUTE AND CLASS INFORMATION

There are 13 attributes in total. All attributes are continuous data. Dataset was verified to contain no missing attribute values.

```
In [7]: wine.isnull().sum()
```

```
Class                         0
Alcohol                       0
Malic acid                    0
Ash                           0
Alcalinity of ash             0
Magnesium                     0
Total phenols                 0
Flavanoids                    0
Nonflavanoid phenols          0
Proanthocyanins               0
Color intensity               0
Hue                           0
OD280/OD315 of diluted wines  0
Proline                       0
dtype: int64
```

Fig 2. No missing attribute value in the given dataset.

There are a total of 3 classes and a total of 178 instances.:

- Class 1 - 59 instances
- Class 2 - 71 instances
- Class 3 - 48 instances

PART 3: DATA PREPROCESSING

We have calculated the measures of central tendencies and it can be summarized in the contingency table below

```
In [8]: wine.describe()
```

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000
mean	1.938202	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.029270	0.361854	1.590899	5.058090	0.957449
std	0.775035	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.998859	0.124453	0.572359	2.318286	0.228572
min	1.000000	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.340000	0.130000	0.410000	1.280000	0.480000
25%	1.000000	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.205000	0.270000	1.250000	3.220000	0.782500
50%	2.000000	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.135000	0.340000	1.555000	4.690000	0.965000
75%	3.000000	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.875000	0.437500	1.950000	6.200000	1.120000
max	3.000000	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.080000	0.660000	3.580000	13.000000	1.710000

Fig 3. Measures of central tendency of Data

PART 4: PRELIMINARY ANALYSIS OF DATA

We can observe that the data is sorted based on class labels. So we infer that we must randomize it before splitting.

By observing the data description, we can infer that the features are not closely related to each other. For e.g. Proline values dominate overall central tendency measures over attributes such as Ash content.

Hence, we can infer that there is a need for normalizing the attribute values to be contained in a similar domain.

We decided to normalize the data after splitting it into test and training set

PART 5: TRAINING DATA vs TEST DATA

Here we are using a simple holdout method where we are keeping 25% of the data for test and 75% for training. The data has been randomized before split as observed in the preliminary stage.

```
In [9]: Train,Test=TTS(wine,test_size=0.25,random_state=4)

In [10]: Train
```

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD315 of diluted wines	Proline
5	1	14.20	1.76	2.45	15.2	112	3.27	3.39	0.34	1.97	6.75	1.05	2.85	1450
101	2	12.60	1.34	1.90	18.5	88	1.45	1.36	0.29	1.35	2.45	1.04	2.77	562
141	3	13.36	2.56	2.35	20.0	89	1.40	0.50	0.37	0.64	5.60	0.70	2.47	780
25	1	13.05	2.05	3.22	25.0	124	2.63	2.68	0.47	1.92	3.58	1.13	3.20	830
148	3	13.32	3.24	2.38	21.5	92	1.93	0.76	0.45	1.25	8.42	0.55	1.62	650
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
87	2	11.65	1.67	2.62	26.0	88	1.92	1.61	0.40	1.34	2.60	1.36	3.21	562
104	2	12.51	1.73	1.98	20.5	85	2.20	1.92	0.32	1.48	2.94	1.04	3.57	672
129	2	12.04	4.30	2.38	22.0	80	2.10	1.75	0.42	1.35	2.60	0.79	2.57	580
174	3	13.40	3.91	2.48	23.0	102	1.80	0.75	0.43	1.41	7.30	0.70	1.56	750
122	2	12.42	4.43	2.73	26.5	102	2.20	2.13	0.43	1.71	2.08	0.92	3.12	365

133 rows × 14 columns

Fig 4. 75% of the Dataset is Training Data

	Class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD315 of diluted wines	Proline
163	3	12.96	3.45	2.35	18.5	106	1.39	0.70	0.40	0.94	5.28	0.68	1.75	675
142	3	13.52	3.17	2.72	23.5	97	1.55	0.52	0.50	0.55	4.35	0.89	2.06	520
14	1	14.38	1.87	2.38	12.0	102	3.30	3.64	0.29	2.96	7.50	1.20	3.00	1547
45	1	14.21	4.04	2.44	18.9	111	2.85	2.65	0.30	1.25	5.24	0.87	3.33	1080
81	2	12.72	1.81	2.20	18.8	86	2.20	2.53	0.26	1.77	3.90	1.16	3.14	714
157	3	12.45	3.03	2.64	27.0	97	1.90	0.58	0.63	1.14	7.50	0.67	1.73	880
26	1	13.39	1.77	2.62	16.1	93	2.85	2.94	0.34	1.45	4.80	0.92	3.22	1195
74	2	11.96	1.09	2.30	21.0	101	3.38	2.14	0.13	1.65	3.21	0.99	3.13	886
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
76	2	13.03	0.90	1.71	16.0	86	1.95	2.03	0.24	1.46	4.60	1.19	2.48	392
90	2	12.08	1.83	2.32	18.5	81	1.60	1.50	0.52	1.64	2.40	1.08	2.27	480
18	1	14.19	1.59	2.48	16.5	108	3.30	3.93	0.32	1.86	8.70	1.23	2.82	1680
170	3	12.20	3.03	2.32	19.0	96	1.25	0.49	0.40	0.73	5.50	0.66	1.83	510

Fig 5. 25% of the Dataset is Test Data

```
In [12]: Train.shape
Out[12]: (133, 14)

In [13]: Test.shape
Out[13]: (45, 14)

In [14]: X_train=Train.drop(['Class'],axis=1)

In [15]: X_train.head()
```

	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD315 of diluted wines	Proline
5	14.20	1.76	2.45	15.2	112	3.27	3.39	0.34	1.97	6.75	1.05	2.85	1450
101	12.60	1.34	1.90	18.5	88	1.45	1.36	0.29	1.35	2.45	1.04	2.77	562
141	13.36	2.56	2.35	20.0	89	1.40	0.50	0.37	0.64	5.60	0.70	2.47	780
25	13.05	2.05	3.22	25.0	124	2.63	2.68	0.47	1.92	3.58	1.13	3.20	830
148	13.32	3.24	2.38	21.5	92	1.93	0.76	0.45	1.25	8.42	0.55	1.62	650

Fig 6. Checking the first 5 values of Training Set (for randomness)

```
In [16]: Y_train=Train['Class']

In [17]: Y_train.head()

Out[17]: 5      1
         101    2
         141    3
         25     1
         148    3
         Name: Class, dtype: int64

In [18]: X_test=Test.drop(['Class'],axis=1)

In [19]: X_test.head()
```

	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD315 of diluted wines	Proline
163	12.96	3.45	2.35	18.5	106	1.39	0.70	0.40	0.94	5.28	0.68	1.75	675
142	13.52	3.17	2.72	23.5	97	1.55	0.52	0.50	0.55	4.35	0.89	2.06	520
14	14.38	1.87	2.38	12.0	102	3.30	3.64	0.29	2.96	7.50	1.20	3.00	1547
45	14.21	4.04	2.44	18.9	111	2.85	2.65	0.30	1.25	5.24	0.87	3.33	1080
81	12.72	1.81	2.20	18.8	86	2.20	2.53	0.26	1.77	3.90	1.16	3.14	714

Fig 7. Checking the first 5 values of Test Set (for randomness)

PART 6: NORMALIZATION OF TRAINING AND TEST DATA

We performed two types of scaling:

1. Standard Scaling:

```
In [20]: Y_test=Test['Class']

In [21]: scaled_X = StandardScaler().fit_transform(X_train.values)

In [22]: X_train_ss = pd.DataFrame(scaled_X, index=X_train.index, columns=X_train.columns)

In [23]: X_train_ss.describe()
```

	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity
count	1.330000e+02	1.330000e+02	1.330000e+02	1.330000e+02	1.330000e+02	1.330000e+02	1.330000e+02	1.330000e+02	1.330000e+02	1.330000e+02
mean	2.376545e-15	1.168656e-16	1.128588e-15	2.621128e-16	-2.537653e-16	4.073600e-16	-2.671213e-16	-4.482630e-16	8.013640e-17	-2.003410e-17
std	1.003781e+00	1.003781e+00	1.003781e+00	1.003781e+00	1.003781e+00	1.003781e+00	1.003781e+00	1.003781e+00	1.003781e+00	1.003781e+00
min	-2.404430e+00	-1.256211e+00	-3.532455e+00	-2.654497e+00	-2.083407e+00	-2.155149e+00	-1.777509e+00	-1.803919e+00	-2.085055e+00	-1.595428e+00
25%	-7.535553e-01	-6.340028e-01	-5.842564e-01	-7.212708e-01	-8.408371e-01	-8.411924e-01	-8.227737e-01	-7.947865e-01	-6.078380e-01	-7.738213e-01
50%	8.420182e-02	-4.499693e-01	-2.269469e-02	-4.757084e-02	-1.505202e-01	1.202392e-01	1.218047e-01	-2.902204e-01	-6.267462e-02	-1.956534e-01
75%	8.480392e-01	6.980488e-01	6.441598e-01	5.382552e-01	5.397966e-01	8.092651e-01	8.226210e-01	6.348175e-01	6.231761e-01	5.129283e-01
max	2.277154e+00	3.046666e+00	3.030797e+00	3.028016e+00	4.267508e+00	2.491770e+00	3.036794e+00	2.316705e+00	3.489680e+00	3.499405e+00

Fig 8. Normalization using Standard scaling

2. Min-Max scaling:

```
In [24]: scaled_train=MinMaxScaler().fit_transform(X_train.values)
         X_train_s=pd.DataFrame(scaled_train, index=X_train.index, columns=X_train.columns)

In [25]: X_train_s.describe()
```

	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthocyanins	Color intensity	Hue	OD280/OD3 of dilut win
count	133.000000	133.000000	133.000000	133.000000	133.000000	133.000000	133.000000	133.000000	133.000000	133.000000	133.000000	133.0000
mean	0.513593	0.291947	0.538217	0.467134	0.328048	0.463780	0.369214	0.437778	0.374019	0.313146	0.394804	0.4932
std	0.214410	0.233281	0.152940	0.176644	0.158053	0.216010	0.208500	0.243599	0.180059	0.197019	0.195085	0.261(
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	0.352632	0.144603	0.449198	0.340206	0.195652	0.282759	0.198312	0.244898	0.264984	0.161263	0.252033	0.2527
50%	0.531579	0.187373	0.534759	0.458763	0.304348	0.489655	0.394515	0.367347	0.362776	0.274744	0.406504	0.5531
75%	0.694737	0.454175	0.636364	0.561856	0.413043	0.637931	0.540084	0.591837	0.485804	0.413823	0.520325	0.695(
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000

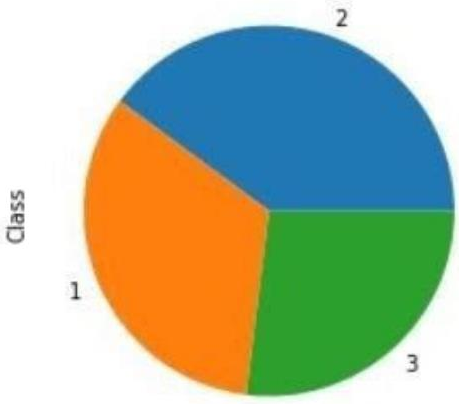
Fig 9. Training Data Min-Max normalization (0 - 1)

We have decided to use Min-Max Scaling over Standard Scaling. Since the values are much closer to each other in min-max and since we know that classifiers such as SVM depend on how good the scaling is performed, min-max dominates over standard scaling.

PART 6: RE-ANALYSIS OF DATA AFTER PARTITIONING (TRAINING AND TEST SETS) AND NORMALIZING

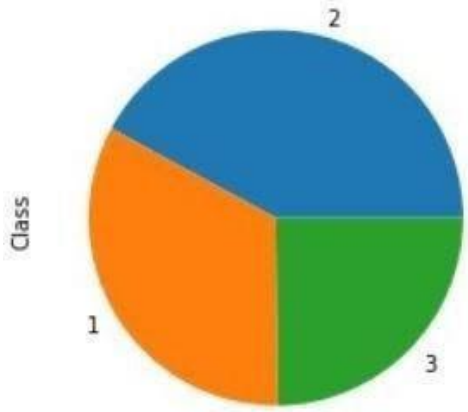
1. Class distribution

```
In [28]: wine.Class.value_counts().plot(kind='pie') #Class distribution in original dataset
```



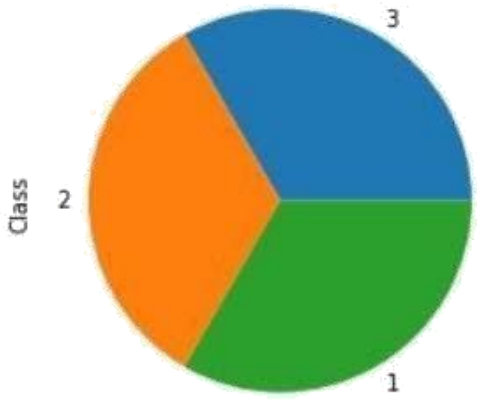
Class distribution in the original dataset

```
In [29]: Y_train.value_counts().plot(kind='pie') #Class distribution in training dataset
```



Class distribution in the training dataset

```
In [30]: Y_test.value_counts().plot(kind='pie') #Class distribution in testing dataset
```

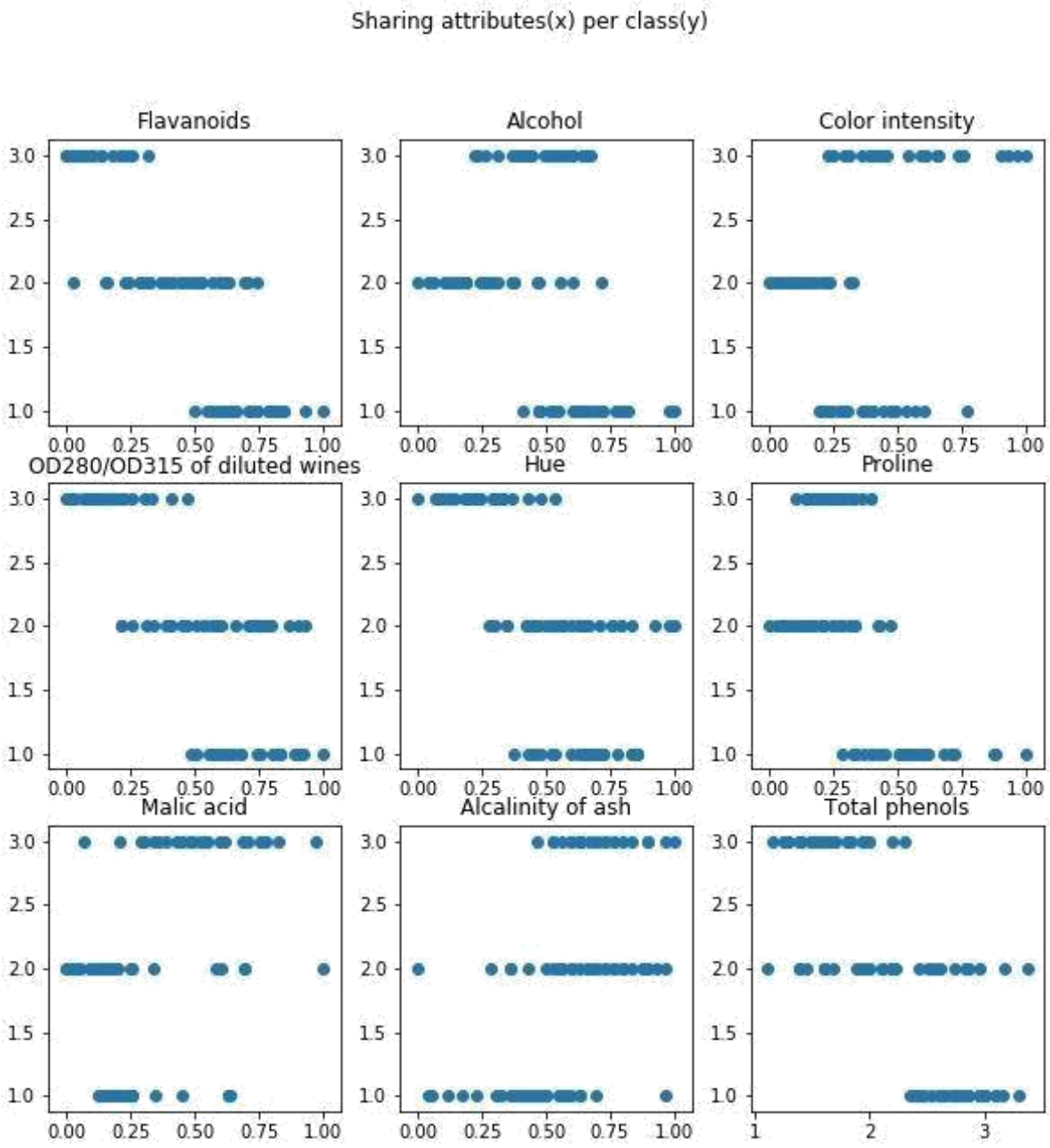


Class distribution in testing dataset

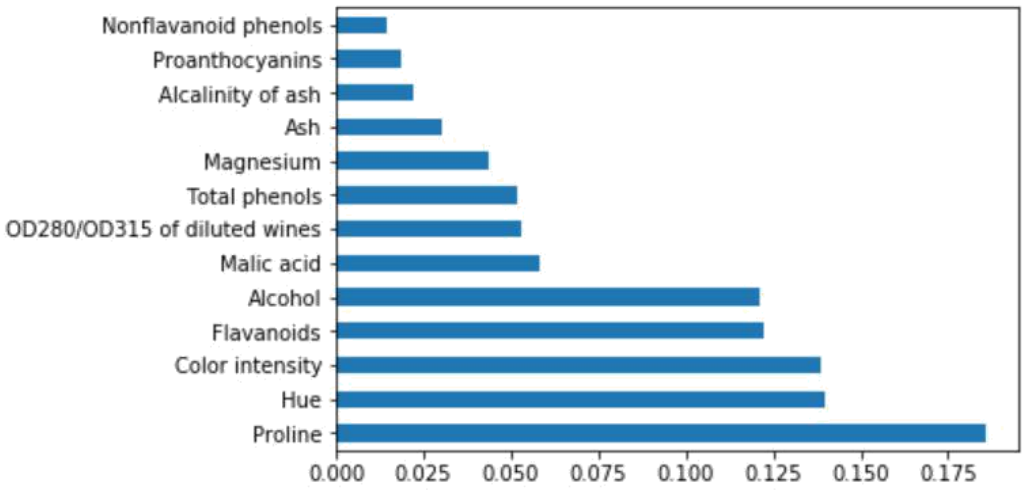
We can see that test-class distribution is roughly equivalent in all three datasets. This means accuracy is a good way of measuring classifiers (due to the absence of bias).

Class as a function of different attributes

```
In [31]: att_fig, (ax) = plt.subplots(3, 3)
att_fig.suptitle('Sharing attributes(x) per class(y)')

ax[0,0].scatter(X_train_s['Flavanoids'],Y_train)
ax[0,0].set_title('Flavanoids')

ax[0,1].scatter(X_train_s['Alcohol'],Y_train)
ax[0,1].set_title('Alcohol')

ax[0,2].scatter(X_train_s['Color intensity'],Y_train)
ax[0,2].set_title('Color intensity')

ax[1,0].scatter(X_train_s['OD280/OD315 of diluted wines'],Y_train)
ax[1,0].set_title('OD280/OD315 of diluted wines')

ax[1,1].scatter(X_train_s['Hue'],Y_train)
ax[1,1].set_title('Hue')

ax[1,2].scatter(X_train_s['Proline'],Y_train)
ax[1,2].set_title('Proline')

ax[2,0].scatter(X_train_s['Malic acid'],Y_train)
ax[2,0].set_title('Malic acid')

ax[2,1].scatter(X_train_s['Alcalinity of ash'],Y_train)
ax[2,1].set_title('Alcalinity of ash');

ax[2,2].scatter(X_train['Total phenols'],Y_train)
ax[2,2].set_title('Total phenols');

#plt.savefig('att_fig')
```



These are the sub-plots of various important attributes and the relation between attribute-values and label classes.



Attributes and their influence on classification have been calculated in order to drop those attributes which are least important like Nonflavanoid phenols, Ash.

PART 7: CLASSIFICATION AND CHOOSING APPROPRIATE CLASSIFIER

We have used SVM (with the linear kernel), Naive Bias Classifiers, and Random Forest (decision tree) classifiers.

	SVM	Naive Bias	Random Forest
Accuracy	0.978	0.956	0.978
Precision	0.979	0.961	0.979
Recall	0.978	0.956	0.978

CONCLUSION

Although it may appear counter-intuitive, we conclude naive bias classifiers may be the best classifier in this case. This is because the classifiers are showing extremely high accuracy and we must try to avoid overfitting.

REFERENCES:

1. https://scikit-learn.org/stable/modules/svm.html
2. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForest Classifier.html
3. https://scikit-learn.org/stable/modules/naive_bayes.html
4. https://matplotlib.org/stable/users/index.html
5. https://towardsdatascience.com/feature-selection-techniques-in-machine-learningwith-python-f24e7da3f36e
6. https://pandas.pydata.org/docs/