In [5]:
```
import numpy as np
import pandas as pd

wine= pd.read_csv("C:\\Users\\ckhan\\OneDrive\\Desktop\\wine.csv")
```

In [6]:
```
wine
```

Out[6]:

	Wine	Alcohol	Malic.acid	Ash	Acl	Mg	Phenols	Flavanoids	Nonflavanoid.phenols	Proanth	Color.int	Hue	OD	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

In [7]:
```
wine.shape
```

Out[7]:
```
(178, 14)
```

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

Out[8]:

	Wine	Alcohol	Malic.acid	Ash	Acl	Mg	Phenols	Flavanoids	Nonflavanoid.phenols	Proanth	Color.int	Hue	OD	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

In [9]:
```
wine.groupby('Wine').size()
```

Out[9]:
```
Wine
1    59
2    71
3    48
dtype: int64
```

In [10]:
```
feature_columns = ['Alcohol','Malic.acid','Ash','Acl','Mg','Phenols','Flavanoids','Nonflavanoid.phenols','Proanth','Color.int','Hue','OD','Proline']
X = wine[feature_columns].values
y = wine['Wine'].values
```

In [11]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 1)
```

In [12]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [13]:
```
# Fitting clasifier to the Training set
# Loading libraries
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.model_selection import cross_val_score
```

In [14]:
```
# Instantiate learning model (k = 3)
classifier = KNeighborsClassifier(n_neighbors=3)
```

In [15]:
```
# Fitting the model
classifier.fit(X_train, y_train)
```

Out[15]:
```
KNeighborsClassifier(n_neighbors=3)
```

In [16]:
```
# Predicting the Test set results
y_pred = classifier.predict(X_test)
```

In [17]:
```
cm = confusion_matrix(y_test, y_pred)
cm
```

Out[17]:
```
array([[21,  0,  2],
       [ 2, 13,  4],
       [ 1,  5,  6]], dtype=int64)
```

In [18]:
```
accuracy = accuracy_score(y_test, y_pred)*100
print('Accuracy of our model is equal ' + str(round(accuracy, 2)) + ' %.')
```

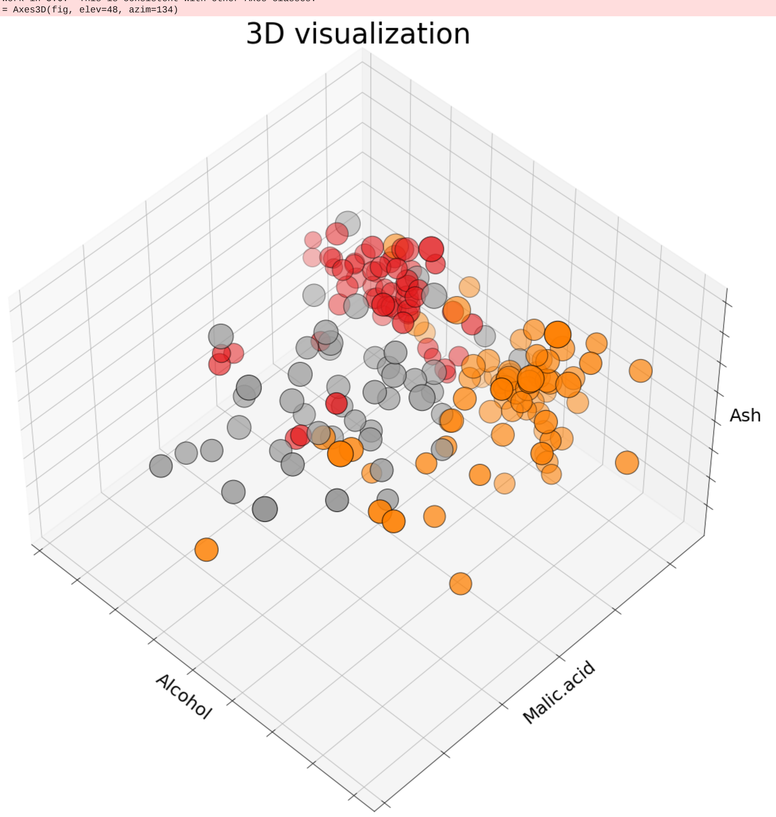
```
Accuracy of our model is equal 74.07 %.
```

In [24]:
```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(1, figsize=(20, 15))
ax = Axes3D(fig, elev=48, azim=134)
ax.scatter(X[:, 0], X[:, 1], X[:, 2], c=y,
           cmap=plt.cm.Set1, edgecolor='k',s = X[:, 3]*50 )

ax.set_title("3D visualization", fontsize=40)
ax.set_xlabel("Alcohol", fontsize=25)
ax.w_xaxis.set_ticklabels([])
ax.set_ylabel("Malic.acid", fontsize=25)
ax.w_yaxis.set_ticklabels([])
ax.set_zlabel("Ash", fontsize=25)
ax.w_zaxis.set_ticklabels([])


plt.show()
```

```
C:\Users\ckhan\AppData\Local\Temp/ipykernel_9536/1320220776.py:3: MatplotlibDeprecationWarning: Axes3D(fig) adding itself to the figure is deprecated since 3.4. Pass the keyword argument auto_add_to_figure=False and use fig.add_axes(ax) to suppress this warning. The default value of auto_add_to_figure will change to False in mpl3.5 and True values will no longer work in 3.6.  This is consistent with other Axes classes.
  ax = Axes3D(fig, elev=48, azim=134)
```



In [31]:
```
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
```

In [33]:
```
# Model Precision: what percentage of positive tuples are labeled as such?
print("Precision:",metrics.precision_score(y_test, y_pred,average = 'weighted'))

# Model Recall: what percentage of positive tuples are labelled as such?
print("Recall:",metrics.recall_score(y_test, y_pred,average = 'weighted'))
```

```
Precision: 0.7379115226337448
Recall: 0.7407407407407407
```

In [ ]:
```

```