In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
import time
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import plotly.express as px
```

In [3]:
```
df = pd.read_csv('Wine.csv')
df.head()
```

Out[3]:

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39

In [4]:
```
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
```

In [5]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
```

In [6]:
```
knn = KNeighborsClassifier()
knn.fit(X_train,y_train)
```

Out[6]:
```
KNeighborsClassifier
KNeighborsClassifier()
```

In [7]:
```
start = time.time()
y_pred = knn.predict(X_test)
print(time.time()-start)
```

```
0.018793344497680664
```

In [8]:
```
accuracy_score(y_test,y_pred)
```

Out[8]:
```
0.7222222222222222
```

In [9]:
```
scaler = StandardScaler()
X_train_trf = scaler.fit_transform(X_train)
X_test_trf = scaler.transform(X_test)
```

In [10]:
```
pca = PCA(n_components=2)
X_train_trf = pca.fit_transform(X_train_trf)
X_test_trf = pca.transform(X_test_trf)
```

In [11]:
```
knn.fit(X_train_trf,y_train)
```

Out[11]:
```
KNeighborsClassifier
KNeighborsClassifier()
```

In [12]:
```
start = time.time()
y_pred_trf = knn.predict(X_test_trf)
print(time.time()-start)
```

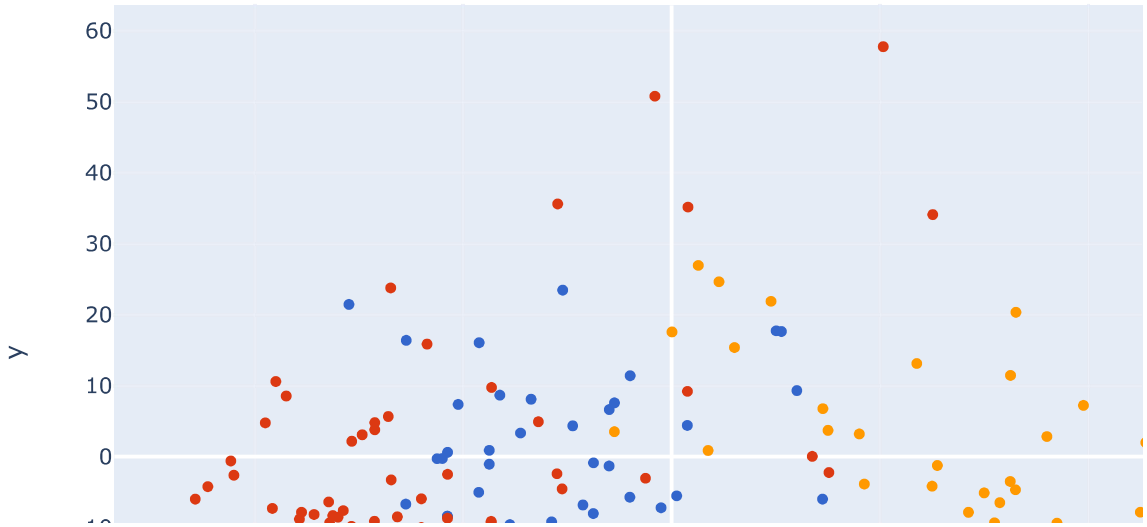
```
0.008545398712158203
```

In [13]:
```
accuracy_score(y_test,y_pred_trf)
```

Out[13]:
```
1.0
```

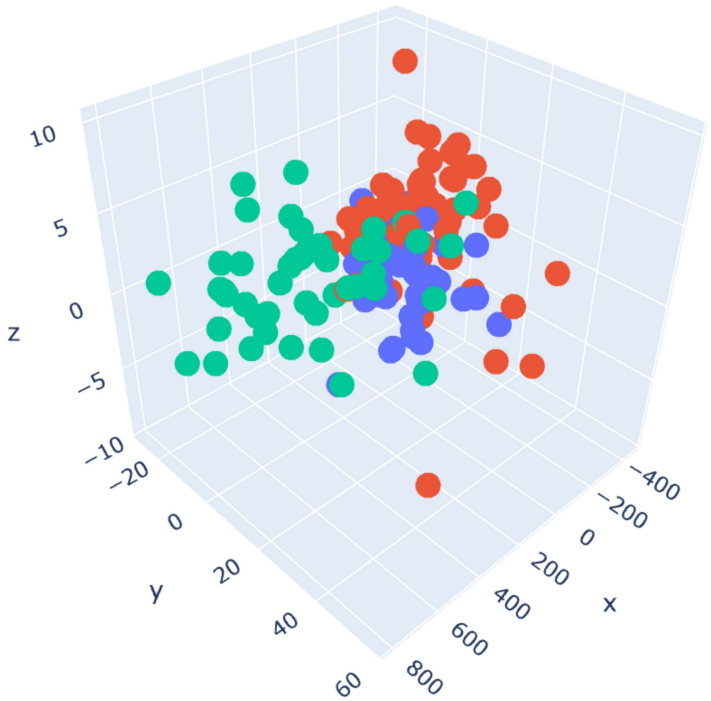
In [14]:
```
pca = PCA(n_components=2)
X_train_trf = pca.fit_transform(X_train)
X_test_trf = pca.transform(X_test)
```

In [15]:
```
y_train_trf = y_train.astype(str)
fig = px.scatter(x=X_train_trf[:,0],
                 y=X_train_trf[:,1],
                 color=y_train_trf,
                 color_discrete_sequence=px.colors.qualitative.G10
                )
fig.show()
```



In [16]:
```
pca = PCA(n_components=3)
X_train_trf = pca.fit_transform(X_train)
X_test_trf = pca.transform(X_test)
```

In [17]:
```
y_train_trf = y_train.astype(str)
fig = px.scatter_3d(df, x=X_train_trf[:,0], y=X_train_trf[:,1], z=X_train_trf[:,2],
              color=y_train_trf)
fig.update_layout(
    margin=dict(l=20, r=20, t=20, b=20),
    paper_bgcolor="LightSteelBlue",
)
fig.show()
```



In [ ]:
```

```