In [17]:
```
!pip install mlxtend
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: mlxtend in c:\users\test25\appdata\roaming\python\pyt
hon312\site-packages (0.23.4)
Requirement already satisfied: scipy>=1.2.1 in c:\programdata\anaconda3\lib\site-pac
kages (from mlxtend) (1.13.1)
Requirement already satisfied: numpy>=1.16.2 in c:\programdata\anaconda3\lib\site-pa
ckages (from mlxtend) (1.26.4)
Requirement already satisfied: pandas>=0.24.2 in c:\programdata\anaconda3\lib\site-p
ackages (from mlxtend) (2.2.2)
Requirement already satisfied: scikit-learn>=1.3.1 in c:\programdata\anaconda3\lib\s
ite-packages (from mlxtend) (1.4.2)
Requirement already satisfied: matplotlib>=3.0.0 in c:\programdata\anaconda3\lib\sit
e-packages (from mlxtend) (3.8.4)
Requirement already satisfied: joblib>=0.13.2 in c:\programdata\anaconda3\lib\site-p
ackages (from mlxtend) (1.4.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\programdata\anaconda3\lib\site
-packages (from matplotlib>=3.0.0->mlxtend) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-pac
kages (from matplotlib>=3.0.0->mlxtend) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\anaconda3\lib\sit
e-packages (from matplotlib>=3.0.0->mlxtend) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\programdata\anaconda3\lib\sit
e-packages (from matplotlib>=3.0.0->mlxtend) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda3\lib\site-
packages (from matplotlib>=3.0.0->mlxtend) (23.2)
Requirement already satisfied: pillow>=8 in c:\programdata\anaconda3\lib\site-packag
es (from matplotlib>=3.0.0->mlxtend) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\programdata\anaconda3\lib\site
-packages (from matplotlib>=3.0.0->mlxtend) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\programdata\anaconda3\lib
\site-packages (from matplotlib>=3.0.0->mlxtend) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-pac
kages (from pandas>=0.24.2->mlxtend) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\site-p
ackages (from pandas>=0.24.2->mlxtend) (2023.3)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\programdata\anaconda3\lib
\site-packages (from scikit-learn>=1.3.1->mlxtend) (2.2.0)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-package
s (from python-dateutil>=2.7->matplotlib>=3.0.0->mlxtend) (1.16.0)
```

In [18]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report
```

In [19]:
```
df=pd.read_csv('cancer_data.csv')
df.head()
```

Out[19]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothn
0	842302	M	17.99	10.38	122.80	1001.0	
1	842517	M	20.57	17.77	132.90	1326.0	
2	84300903	M	19.69	21.25	130.00	1203.0	
3	84348301	M	11.42	20.38	77.58	386.1	
4	84358402	M	20.29	14.34	135.10	1297.0	

5 rows × 33 columns

In [20]:
```
print(df.columns)
```

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32'],
      dtype='object')
```

In [21]:
```
# Drop columns only if they exist (optional)
cols_to_drop = ['id', 'Unnamed: 32']
existing_cols = [col for col in cols_to_drop if col in df.columns]
if existing_cols:
    df.drop(existing_cols, axis=1, inplace=True)

# Map diagnosis
df['diagnosis'] = df['diagnosis'].map({'M': 1, 'B': 0})

df.head()
```

Out[21]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
0	1	17.99	10.38	122.80	1001.0	0.11840
1	1	20.57	17.77	132.90	1326.0	0.08474
2	1	19.69	21.25	130.00	1203.0	0.10960
3	1	11.42	20.38	77.58	386.1	0.14250
4	1	20.29	14.34	135.10	1297.0	0.10030

5 rows × 31 columns

In [22]:
```
x=df.drop('diagnosis', axis=1)
y=df['diagnosis']
```

In [23]:
```
scaler =StandardScaler()
scaled_data = scaler.fit_transform(x)
print(scaled_data[:2])
```

```
[[ 1.09706398e+00 -2.07333501e+00  1.26993369e+00  9.84374905e-01
   1.56846633e+00  3.28351467e+00  2.65287398e+00  2.53247522e+00
   2.21751501e+00  2.25574689e+00  2.48973393e+00 -5.65265059e-01
   2.83303087e+00  2.48757756e+00 -2.14001647e-01  1.31686157e+00
   7.24026158e-01  6.60819941e-01  1.14875667e+00  9.07083081e-01
   1.88668963e+00 -1.35929347e+00  2.30360062e+00  2.00123749e+00
   1.30768627e+00  2.61666502e+00  2.10952635e+00  2.29607613e+00
   2.75062224e+00  1.93701461e+00]
 [ 1.82982061e+00 -3.53632408e-01  1.68595471e+00  1.90870825e+00
  -8.26962447e-01 -4.87071673e-01 -2.38458552e-02  5.48144156e-01
   1.39236330e-03 -8.68652457e-01  4.99254601e-01 -8.76243603e-01
   2.63326966e-01  7.42401948e-01 -6.05350847e-01 -6.92926270e-01
  -4.40780058e-01  2.60162067e-01 -8.05450380e-01 -9.94437403e-02
   1.80592744e+00 -3.69203222e-01  1.53512599e+00  1.89048899e+00
  -3.75611957e-01 -4.30444219e-01 -1.46748968e-01  1.08708430e+00
  -2.43889668e-01  2.81189987e-01]]
```

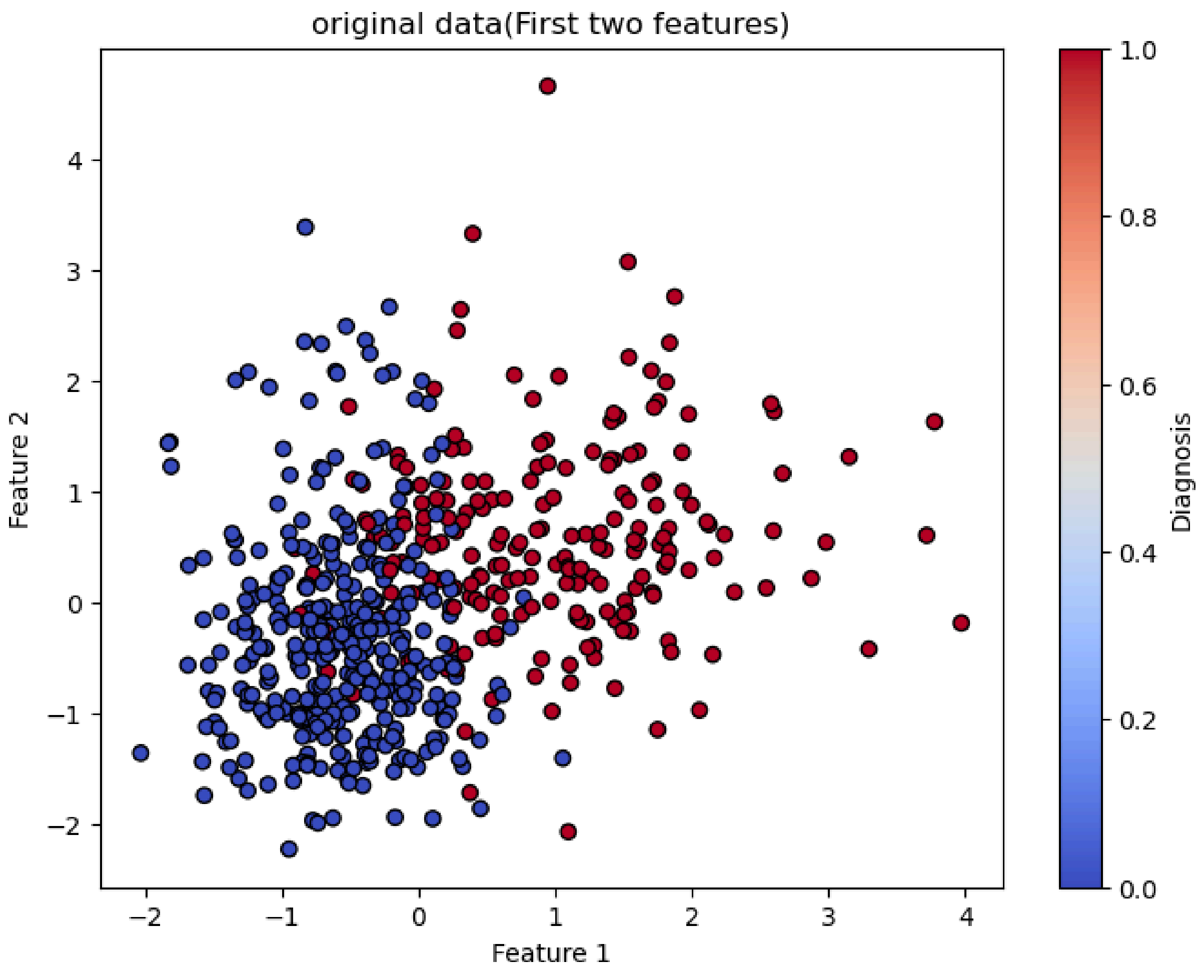
In [24]:
```
pca=PCA(n_components=2)
x_pca=pca.fit_transform(scaled_data)
print(x_pca[:2])
```

```
[[ 9.19283683  1.94858307]
 [ 2.3878018  -3.76817174]]
```

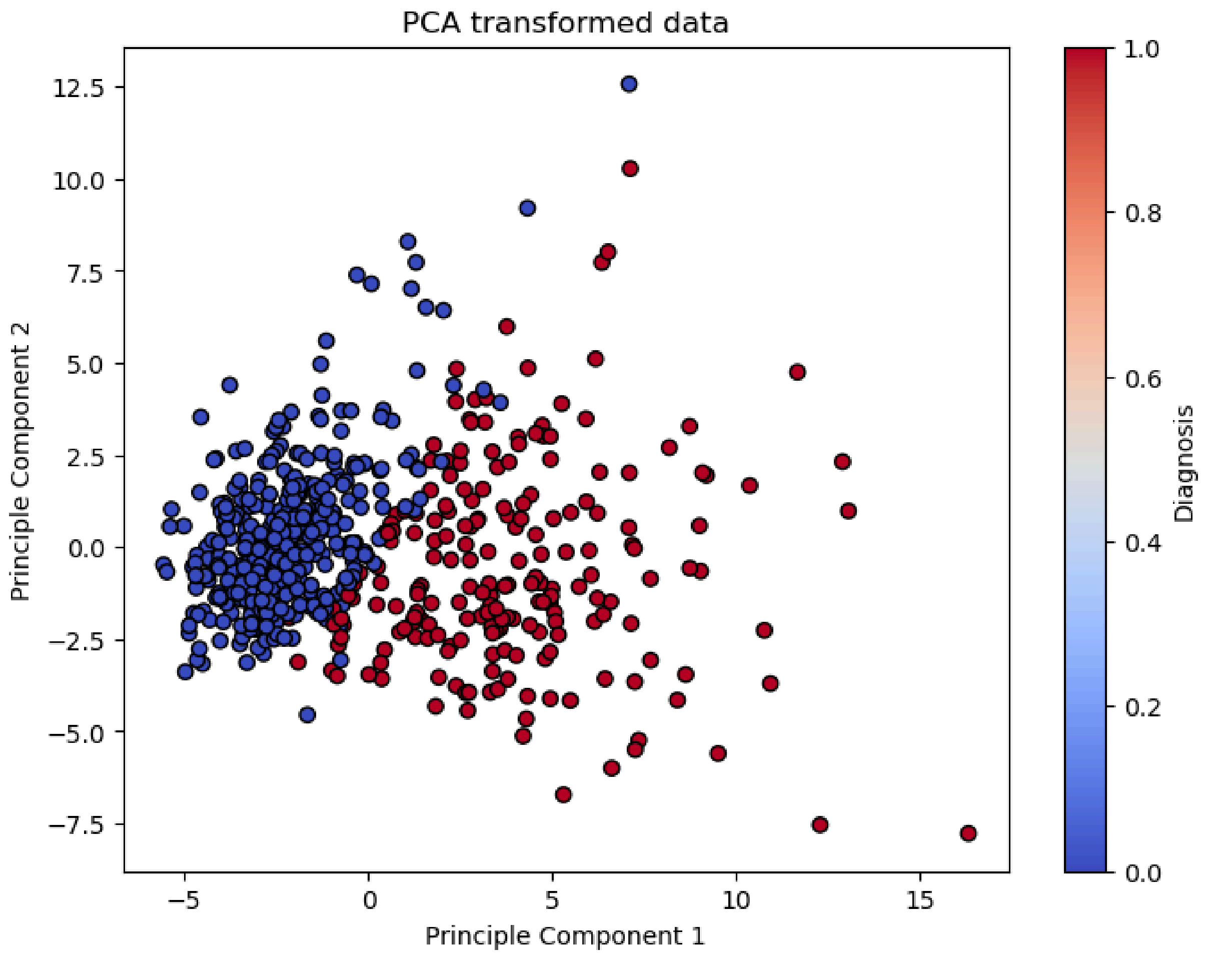
In [25]:
```
print("Explained variance:",pca.explained_variance_ratio_)
print("Cumulative:", np.cumsum(pca.explained_variance_ratio_))
```

```
Explained variance: [0.44272026 0.18971182]
Cumulative: [0.44272026 0.63243208]
```

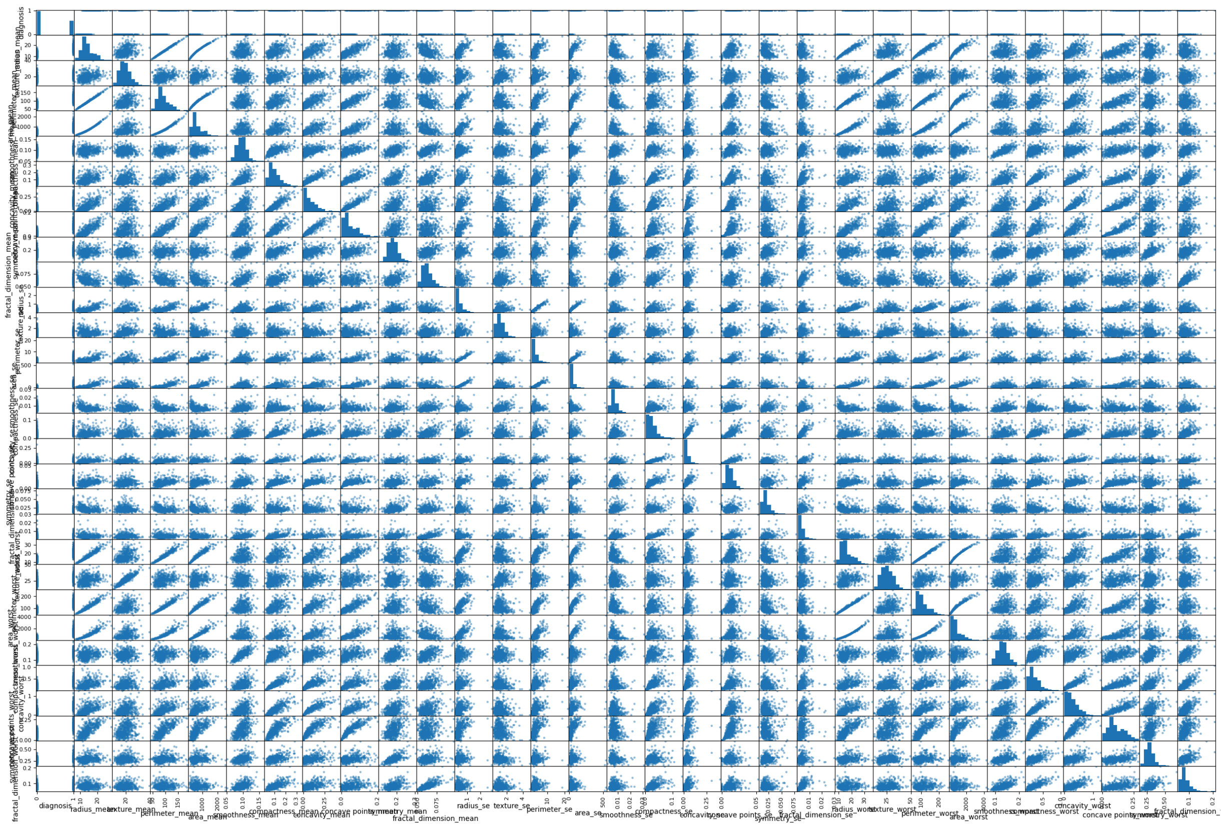
In [26]:
```
plt.figure(figsize=(8,6))
plt.scatter(scaled_data[:,0],scaled_data[:,1],c=y,cmap='coolwarm', edgecolor='k')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title("original data(First two features)")
plt.colorbar(label="Diagnosis")
plt.show()
```



In [28]:
```
plt.figure(figsize=(8,6))
plt.scatter(x_pca[:, 0], x_pca[:, 1], c=y, cmap='coolwarm', edgecolor='k')
plt.xlabel('Principle Component 1')
plt.ylabel('Principle Component 2')
plt.title("PCA transformed data")
plt.colorbar(label="Diagnosis")
plt.show()
```



In [30]:
```
from pandas.plotting import scatter_matrix
scatter_matrix(df, figsize=(30,20));
```



In [34]:
```
X_train, X_test, y_train, y_test = train_test_split(x_pca, y, test_size=0.2, random
model=LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99        71
           1       1.00      0.98      0.99        43

    accuracy                           0.99       114
   macro avg       0.99      0.99      0.99       114
weighted avg       0.99      0.99      0.99       114
```

In [36]:
```
X_train, X_test, y_train, y_test = train_test_split(df, y, test_size=0.2, random_st
model=LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        71
           1       1.00      1.00      1.00        43

    accuracy                           1.00       114
   macro avg       1.00      1.00      1.00       114
weighted avg       1.00      1.00      1.00       114
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:469: Co
nvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

In [40]:
```
x_reconstructed = pca.inverse_transform(x_pca)
reconstruction_loss = np.mean((scaled_data - x_reconstructed) ** 2)
print(f"Reconstryction loss:{reconstruction_loss:.4f} ")
```

```
Reconstryction loss:0.3676
```

In [44]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier,plot_tree
from sklearn.metrics import accuracy_score, classification_report
```

In [46]:
```
df=pd.read_csv('fruits.csv')
df.head()
```

Out[46]:

	weight	size	color_score	Fruit_label
0	150	7.5	0.50	lemon
1	170	7.5	0.50	lemon
2	190	8.0	0.74	orange
3	210	8.5	0.75	orange
4	230	8.5	0.75	orange

In [48]:
```
x=df[['weight','size','color_score']]
y=df['Fruit_label']
```

In [68]:
```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.4, random_sta
clf=DecisionTreeClassifier(criterion='entropy', random_state=42)
clf.fit(X_train, y_train)
```

Out[68]:

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', random_state=42)
```

In [70]:
```
y_pred = clf.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

       Apple       1.00      1.00      1.00         3
 Pomegranate       1.00      1.00      1.00         2
       lemon       1.00      1.00      1.00         4
      orange       1.00      1.00      1.00         1

    accuracy                           1.00        10
   macro avg       1.00      1.00      1.00        10
weighted avg       1.00      1.00      1.00        10
```

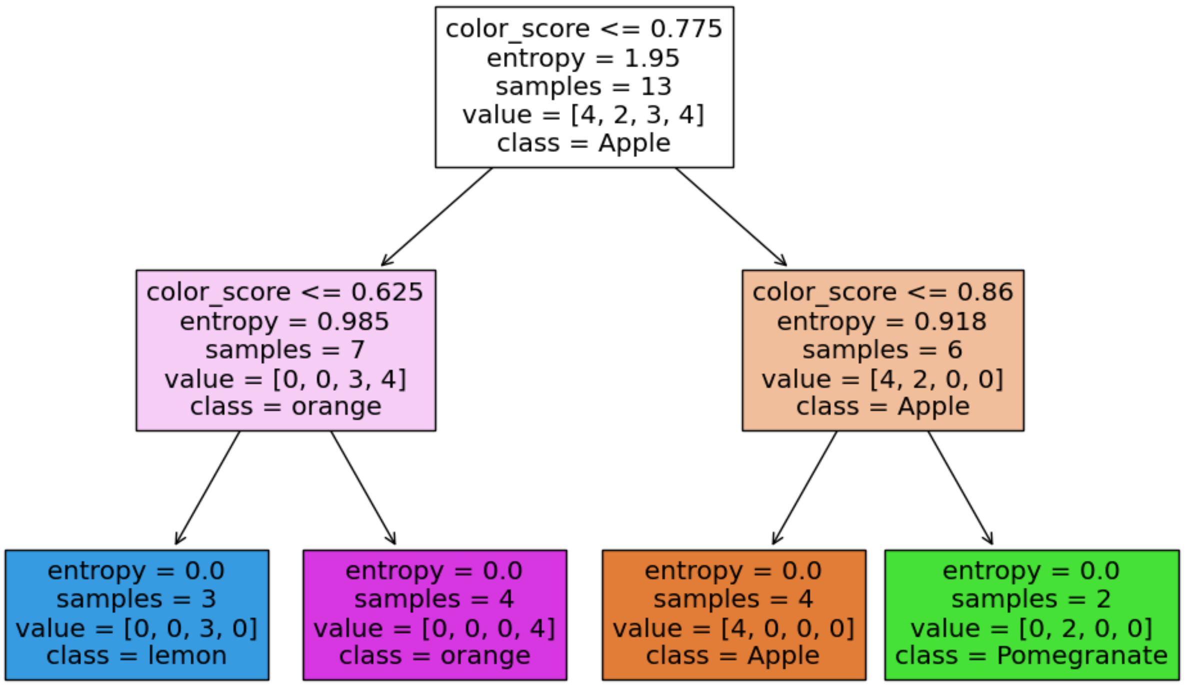
In [72]:
```
print(clf.predict([[300,9,0.9]]))
```

```
['Pomegranate']
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does
not have valid feature names, but DecisionTreeClassifier was fitted with feature nam
es
  warnings.warn(
```

In [80]:
```
plt.figure(figsize=(12,8))
plot_tree(clf, filled=True,feature_names=['weight','size','color_score'], class_nam
plt.title("Decission Tree for fruit classification")
plt.show()
```

Decission Tree for fruit classification



In [ ]: