In [2]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_m
```

In [3]:
```
data = pd.read_csv('car_evaluation.csv')
```

In [4]:
```
# Encoding all the string data
data = data.apply(LabelEncoder().fit_transform)
```

In [5]:
```
x = data.iloc[:, :-1]
y = data.iloc[:, -1]
```

In [6]:
```
#Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, randc
```

In [7]:
```
# Create a Random Forest Classifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
rf_classifier.fit(X_train, y_train)
```

Out[7]:

RandomForestClassifier

Parameters

n_estimators	100
criterion	'gini'
max_depth	None
min_samples_split	2
min_samples_leaf	1
min_weight_fraction_leaf	0.0
max_features	'sqrt'
max_leaf_nodes	None
min_impurity_decrease	0.0
bootstrap	True
oob_score	False
n_jobs	None
random_state	42
verbose	0
warm_start	False
class_weight	None
ccp_alpha	0.0
max_samples	None
monotonic_cst	None

In [8]:
```
# make predictions on the test data
y_pred = rf_classifier.predict(X_test)
```

In [9]:
```
# evaluate the model
accuracy = accuracy_score(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
```

In [12]:
```
print(f"Accuracy:{accuracy}\n")
print(f"Confusion Matrix:\n{confusion}\n")
print(f"Classification Report:\n{classification_rep}")
```

```
Accuracy:0.9624277456647399

Confusion Matrix:
[[ 72   1   3   1]
 [  2  10   0   3]
 [  1   0 236   0]
 [  2   0   0  15]]

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.94      0.94        77
           1       0.91      0.67      0.77        15
           2       0.99      1.00      0.99       237
           3       0.79      0.88      0.83        17

    accuracy                           0.96       346
   macro avg       0.91      0.87      0.88       346
weighted avg       0.96      0.96      0.96       346
```

In [ ]: