In [20]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
```

In [21]:

```
df = pd.read_csv("C:/Users/bharg/Downloads/restaurant_data.csv")
```

In [22]:

```
df['Date'] = pd.to_datetime(df['Date'])
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month
df['Day'] = df['Date'].dt.day
```

```
C:\Users\bharg\AppData\Local\Temp\ipykernel_24584\2624385925.py:1: UserWarning: Parsing
dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may le
ad to inconsistently parsed dates! Specify a format to ensure consistent parsing.
  df['Date'] = pd.to_datetime(df['Date'])
```

In [23]:

```
df['ReservationHour'] = df['ReservationTime'].apply(lambda x: int(x.split(':')[0]))
df['ReservationMinute'] = df['ReservationTime'].apply(lambda x: int(x.split(':')[1]))
```

In [24]:

```
df = df.drop(columns=['Date', 'ReservationTime'])
```

In [25]:

```
X = df.drop(columns=['Feedback', 'MenuItem', 'Category', 'WaitStaff', 'TableSize'])  # D
y = df['Feedback']
```

In [26]:

```
categorical_columns = ['PaymentMethod', 'Weather', 'SpecialEvent', 'CustomerGender']
for col in categorical_columns:
    one_hot_encoder = pd.get_dummies(X[col], prefix=col)
    X = pd.concat([X, one_hot_encoder], axis=1)
    X.drop(columns=[col], inplace=True)
```

In [27]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42
```

In [28]:

```
categorical_columns = [col for col in X.columns if col.strip() == 'PaymentMethod']
```

In [29]:

```
logistic_model = LogisticRegression()
logistic_model.fit(X_train, y_train)
```

```
C:\Users\bharg\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:458: Conve
rgenceWarning: 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 (https://scikit-learn.or
g/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 (http
s://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[29]:

```
LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [30]:

```
y_pred = logistic_model.predict(X_test)
```

In [31]:

```
accuracy = accuracy_score(y_test, y_pred)
classification_report_output = classification_report(y_test, y_pred)
```

In [32]:

```
print(f"Accuracy: {accuracy}")
print("Classification Report:\n", classification_report_output)
```

```
Accuracy: 0.5
Classification Report:
               precision    recall  f1-score   support

   Excellent       1.00      0.17      0.29         6
        Fair       0.50      0.50      0.50         2
        Good       0.45      0.83      0.59         6

    accuracy                           0.50        14
   macro avg       0.65      0.50      0.46        14
weighted avg       0.69      0.50      0.45        14
```

In [33]:

```
knn_model = KNeighborsClassifier(n_neighbors=5)  # You can adjust the number of neighbor
knn_model.fit(X_train, y_train)
```

Out[33]:

```
KNeighborsClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [34]:

```
knn_predictions = knn_model.predict(X_test)
```

In [35]:
```
knn_accuracy = accuracy_score(y_test, knn_predictions)
knn_classification_report = classification_report(y_test, knn_predictions)
```

```
C:\Users\bharg\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labe
ls with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\bharg\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labe
ls with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\bharg\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labe
ls with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [36]:
```
print("K-Nearest Neighbors Classifier:")
print(f"Accuracy: {knn_accuracy}")
print("Classification Report:\n", knn_classification_report)
```

```
K-Nearest Neighbors Classifier:
Accuracy: 0.35714285714285715
Classification Report:
               precision    recall  f1-score   support

   Excellent       0.33      0.17      0.22         6
        Fair       0.00      0.00      0.00         2
        Good       0.36      0.67      0.47         6

    accuracy                           0.36        14
   macro avg       0.23      0.28      0.23        14
weighted avg       0.30      0.36      0.30        14
```

In [37]:
```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)  # You can adjust t
rf_model.fit(X_train, y_train)
```

Out[37]:
```
RandomForestClassifier(random_state=42)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [38]:
```
rf_predictions = rf_model.predict(X_test)
```

In [39]:
```
rf_accuracy = accuracy_score(y_test, rf_predictions)
rf_classification_report = classification_report(y_test, rf_predictions)
```

```
C:\Users\bharg\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labe
ls with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\bharg\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labe
ls with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\bharg\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:1344: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labe
ls with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [40]:

```
print("Random Forest Classifier:")
print(f"Accuracy: {rf_accuracy}")
print("Classification Report:\n", rf_classification_report)
```

```
Random Forest Classifier:
Accuracy: 0.6428571428571429
Classification Report:
               precision    recall  f1-score   support

   Excellent       1.00      0.50      0.67         6
        Fair       0.00      0.00      0.00         2
        Good       0.55      1.00      0.71         6

    accuracy                           0.64        14
   macro avg       0.52      0.50      0.46        14
weighted avg       0.66      0.64      0.59        14
```

In [41]:

```
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
```

Out[41]:

```
DecisionTreeClassifier(random_state=42)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [42]:

```
dt_predictions = dt_model.predict(X_test)
```

In [43]:

```
dt_accuracy = accuracy_score(y_test, dt_predictions)
dt_classification_report = classification_report(y_test, dt_predictions)
```

In [44]:

```
print("Decision Tree Classifier:")
print(f"Accuracy: {dt_accuracy}")
print("Classification Report:\n", dt_classification_report)
```

```
Decision Tree Classifier:
Accuracy: 0.9285714285714286
Classification Report:
               precision    recall  f1-score   support

   Excellent       0.86      1.00      0.92         6
        Fair       1.00      1.00      1.00         2
        Good       1.00      0.83      0.91         6

    accuracy                           0.93        14
   macro avg       0.95      0.94      0.94        14
weighted avg       0.94      0.93      0.93        14
```

In [45]:

```
import matplotlib.pyplot as plt
```

In [49]:

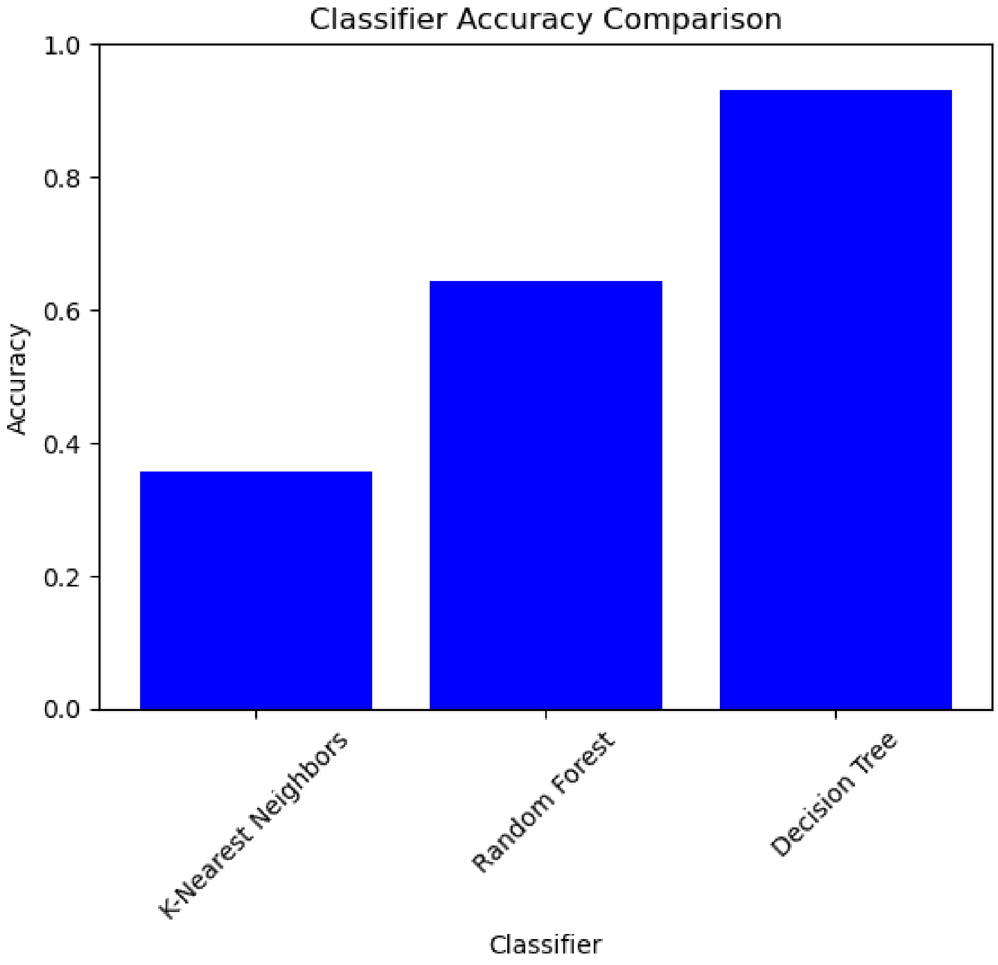
```
classifiers = ["K-Nearest Neighbors", "Random Forest", "Decision Tree"]
accuracies = [knn_accuracy, rf_accuracy, dt_accuracy]
```

In [50]:

```
plt.bar(classifiers, accuracies, color='blue')
plt.xlabel("Classifier")
plt.ylabel("Accuracy")
plt.title("Classifier Accuracy Comparison")
plt.ylim(0.0, 1.0)  # Set the y-axis limits from 0 to 1
plt.xticks(rotation=45)  # Rotate the classifier names for readability
```

Out[50]:

```
([0, 1, 2],
 [Text(0, 0, 'K-Nearest Neighbors'),
  Text(1, 0, 'Random Forest'),
  Text(2, 0, 'Decision Tree')])
```



In [ ]: