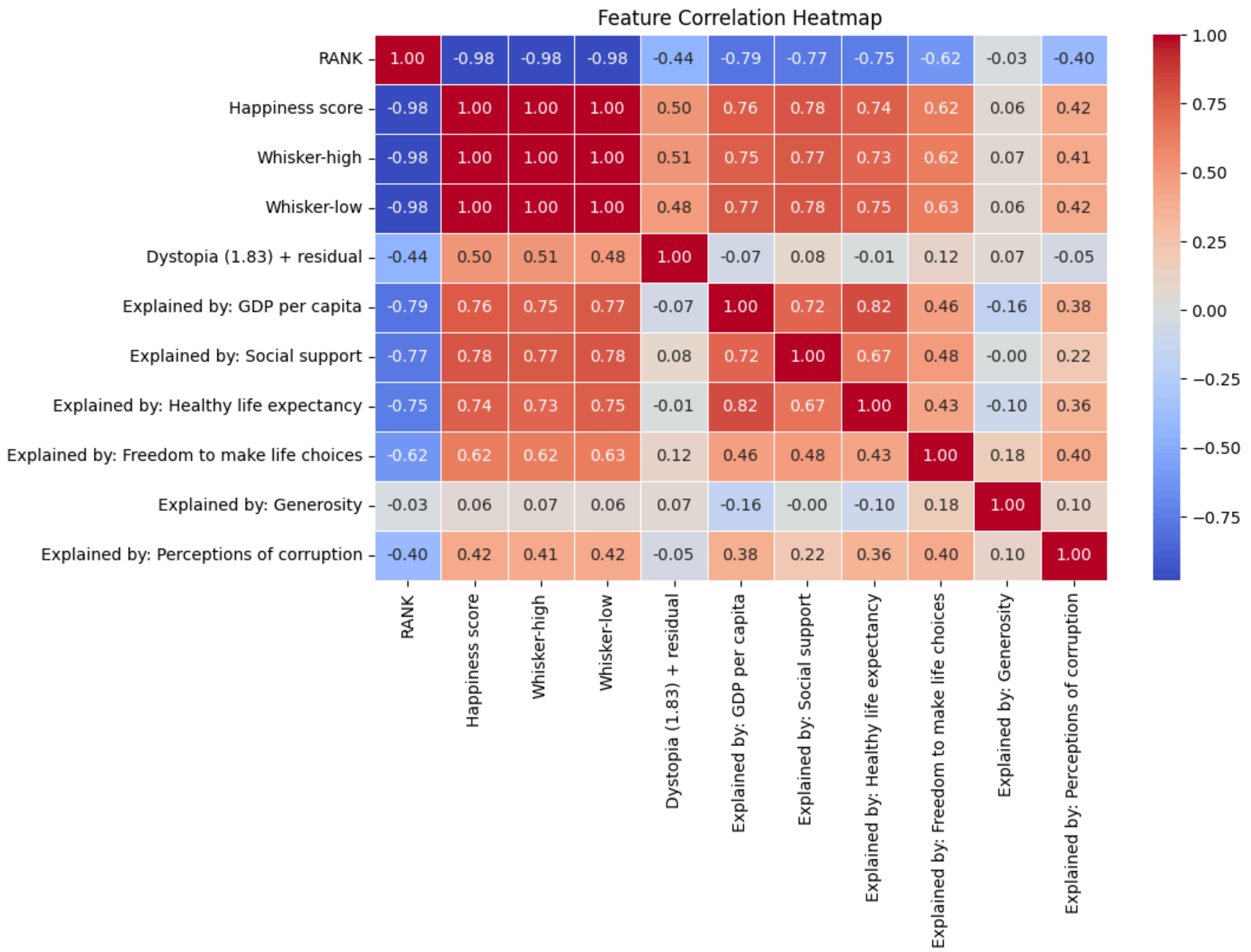
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import shap
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error


df = pd.read_csv("World Happiness Report 2022.csv")  # Update file path if needed


df.columns = df.columns.str.strip()


print("Dataset Columns:", df.columns)
print(df.info())

target_column = None
for col in df.columns:
    if "happiness" in col.lower() and "score" in col.lower():
        target_column = col
        break

if target_column is None:
    print("Error: Happiness Score column not found! Available columns:")
    print(df.columns)
    exit()

print(f"\nUsing '{target_column}' as the Happiness Score column.")


numeric_df = df.select_dtypes(include=[np.number])


correlation_matrix = numeric_df.corr()
top_features = correlation_matrix[target_column].drop(target_column).sort_values(ascending=False)
print("\nTop Correlated Features with Happiness Score:\n", top_features)

plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)
plt.title("Feature Correlation Heatmap")
plt.show()


X = numeric_df.drop(columns=[target_column])
y = numeric_df[target_column]


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


lr = LinearRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)


rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)


print("\nModel Performance:")
print(f"Linear Regression R²: {r2_score(y_test, y_pred_lr):.4f}")
print(f"Random Forest R²: {r2_score(y_test, y_pred_rf):.4f}")
print(f"Random Forest MAE: {mean_absolute_error(y_test, y_pred_rf):.4f}")
print(f"Random Forest MSE: {mean_squared_error(y_test, y_pred_rf):.4f}")


feature_importances = pd.DataFrame({'Feature': X.columns, 'Importance': rf.feature_importances_})
feature_importances = feature_importances.sort_values(by="Importance", ascending=False)


plt.figure(figsize=(10, 5))
sns.barplot(x=feature_importances["Importance"], y=feature_importances["Feature"], palette="viridis")
plt.title("Feature Importance - Random Forest")
plt.xlabel("Importance Score")
plt.ylabel("Features")
```

```
plt.show()


explainer = shap.TreeExplainer(rf)
shap_values = explainer.shap_values(X_test)


shap.summary_plot(shap_values, X_test)
```

```
Dataset Columns: Index(['RANK', 'Country', 'Happiness score', 'Whisker-high', 'Whisker-low',
       'Dystopia (1.83) + residual', 'Explained by: GDP per capita',
       'Explained by: Social support', 'Explained by: Healthy life expectancy',
       'Explained by: Freedom to make life choices',
       'Explained by: Generosity', 'Explained by: Perceptions of corruption'],
      dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146 entries, 0 to 145
Data columns (total 12 columns):
 #   Column                                      Non-Null Count  Dtype
---  ------                                      --------------  -----
 0   RANK                                        146 non-null    int64
 1   Country                                     146 non-null    object
 2   Happiness score                             146 non-null    float64
 3   Whisker-high                                146 non-null    float64
 4   Whisker-low                                 146 non-null    float64
 5   Dystopia (1.83) + residual                  146 non-null    float64
 6   Explained by: GDP per capita                146 non-null    float64
 7   Explained by: Social support                146 non-null    float64
 8   Explained by: Healthy life expectancy       146 non-null    float64
 9   Explained by: Freedom to make life choices  146 non-null    float64
 10  Explained by: Generosity                    146 non-null    float64
 11  Explained by: Perceptions of corruption     146 non-null    float64
dtypes: float64(10), int64(1), object(1)
memory usage: 13.8+ KB
None

Using 'Happiness score' as the Happiness Score column.

Top Correlated Features with Happiness Score:
 Whisker-low                                   0.999383
Whisker-high                                  0.999333
Explained by: Social support                  0.777889
Explained by: GDP per capita                  0.763677
Explained by: Healthy life expectancy         0.740260
Explained by: Freedom to make life choices    0.624822
Dystopia (1.83) + residual                    0.498990
Explained by: Perceptions of corruption       0.416216
Explained by: Generosity                      0.063785
RANK                                         -0.980856
Name: Happiness score, dtype: float64
```




```
Model Performance:
Linear Regression R²: 1.0000
Random Forest R²: 0.9992
Random Forest MAE: 0.0204
```