Project Description The World Happiness Report is a landmark survey of the state of global happiness. The first report was published in 2012, the second in 2013, the third in 2015, and the fourth in the 2016 Update. The World Happiness 2017, which ranks 155 countries by their happiness levels, was released at the United Nations at an event celebrating International Day of Happiness on March 20th. The report continues to gain global recognition as governments, organizations and civil society increasingly use happiness indicators to inform their policy-making decisions. Leading experts across fields – economics, psychology, survey analysis, national statistics, health, public policy and more – describe how measurements of well-being can be used effectively to assess the progress of nations. The reports review the state of happiness in the world today and show how the new science of happiness explains personal and national variations in happiness.

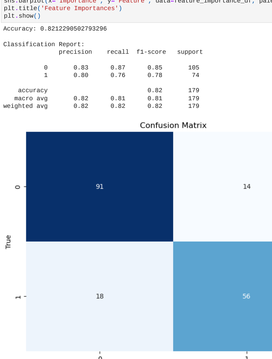
What is Dystopia? Dystopia is an imaginary country that has the world's least-happy people. The purpose in establishing Dystopia is to have a benchmark against which all countries can be favorably compared (no country performs more poorly than Dystopia) in terms of each of the six key variables, thus allowing each sub-bar to be of positive width. The lowest scores observed for the six key variables, therefore, characterize Dystopia. Since life would be very unpleasant in a country with the world's lowest incomes, lowest life expectancy, lowest generosity, most corruption, least freedom and least social support, it is referred to as "Dystopia," in contrast to Utopia.

What are the residuals? The residuals, or unexplained components, differ for each country, reflecting the extent to which the six variables either over- or under-explain average life evaluations. These residuals have an average value of approximately zero over the whole set of countries.

What do the columns succeeding the Happiness Score(like Family, Generosity, etc.) describe? The following columns: GDP per Capita, Family, Life Expectancy, Freedom, Generosity, Trust Government Corruption describe the extent to which these factors contribute in evaluating the happiness in each country. The Dystopia Residual metric actually is the Dystopia Happiness Score(1.85) + the Residual value or the unexplained value for each country. The Dystopia Residual is already provided in the dataset. If you add all these factors up, you get the happiness score so it might be un-reliable to model them to predict Happiness Scores. You need to predict the happiness score considering all the other factors mentioned in the dataset. Dataset Link- https://github.com/dsrscientist/DSData/blob/master/happiness_score_dataset.csv https://github.com/dsrscientist/DSData

In [7]:
```
# Importing necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load the dataset
url = "https://raw.githubusercontent.com/dsrscientist/DSData/master/happiness_score_dataset.csv"
data = pd.read_csv(url)

# Understanding the dataset
print(data.info())
print(data.describe())

# Feature Engineering
# Dystopia Residual is already provided, so we exclude it
features = ['Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'Freedom', 'Generosity', 'Trust (Government Corruption)']

# Define X (features) and y (target)
X = data[features]
y = data['Happiness Score']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Model Training
model = LinearRegression()
model.fit(X_train, y_train)

# Model Prediction
y_pred = model.predict(X_test)

# Model Evaluation
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')
print(f'R-squared: {r2}')

# Results Interpretation
coefficients = pd.DataFrame({'Feat': features, 'Coefficient': model.coef_})
print(coefficients)

# Visualization
plt.scatter(y_test, y_pred)
plt.xlabel('Actual  Score')
plt.ylabel('Predicted  Score')
plt.title('Actual vs Predicted  Score')
plt.show()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 158 entries, 0 to 157
Data columns (total 12 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   Country                        158 non-null    object
 1   Region                         158 non-null    object
 2   Happiness Rank                 158 non-null    int64
 3   Happiness Score                158 non-null    float64
 4   Standard Error                 158 non-null    float64
 5   Economy (GDP per Capita)       158 non-null    float64
 6   Family                         158 non-null    float64
 7   Health (Life Expectancy)       158 non-null    float64
 8   Freedom                        158 non-null    float64
 9   Trust (Government Corruption)  158 non-null    float64
 10  Generosity                     158 non-null    float64
 11  Dystopia Residual              158 non-null    float64
dtypes: float64(9), int64(1), object(2)
memory usage: 14.9+ KB
None
       Happiness Rank  Happiness Score  Standard Error  \
count      158.000000       158.000000      158.000000
mean        79.493671         5.375734        0.047885
std         45.754363         1.145010        0.017146
min          1.000000         2.839000        0.018480
25%         40.250000         4.526000        0.037268
50%         79.500000         5.232500        0.043940
75%        118.750000         6.243750        0.052300
max        158.000000         7.587000        0.136930

       Economy (GDP per Capita)      Family  Health (Life Expectancy)  \
count                158.000000  158.000000                158.000000
mean                   0.846137    0.991046                  0.630259
std                    0.403121    0.272369                  0.247078
min                    0.000000    0.000000                  0.000000
25%                    0.545808    0.856823                  0.439185
50%                    0.910245    1.029510                  0.696705
75%                    1.158448    1.214405                  0.811013
max                    1.690420    1.402230                  1.025250

          Freedom  Trust (Government Corruption)  Generosity  \
count  158.000000                     158.000000  158.000000
mean     0.428615                       0.143422    0.237296
std      0.150693                       0.120034    0.126685
min      0.000000                       0.000000    0.000000
25%      0.328330                       0.061675    0.150553
50%      0.435515                       0.107220    0.216130
75%      0.549092                       0.180255    0.309883
max      0.669730                       0.551910    0.795880

       Dystopia Residual
count         158.000000
mean            2.098977
std             0.553550
min             0.328580
25%             1.759410
50%             2.095415
75%             2.462415
max             3.602140
Mean Squared Error: 0.26561707765585174
R-squared: 0.8004656793512618
                          Feat  Coefficient
0       Economy (GDP per Capita)     0.980089
1                        Family     1.182507
2        Health (Life Expectancy)     1.018139
3                       Freedom     1.214442
4                    Generosity     0.823997
5  Trust (Government Corruption)     0.711862
```



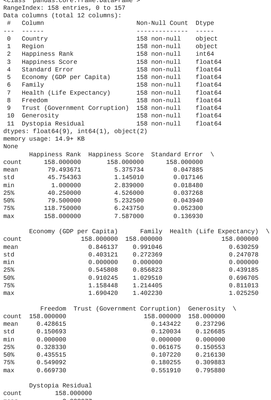
Titanic survived Project Project Description The Titanic Problem is based on the sinking of the 'Unsinkable' ship Titanic in early 1912. It gives you information about multiple people like their ages, sexes, sibling counts, embarkment points, and whether or not they survived the disaster. Based on these features, you have to predict if an arbitrary passenger on Titanic would survive the sinking or not.

Attribute Information Passenger id- Unique Id of the passenger Pclass- Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd) Survived- Survived (0 = No; 1 = Yes) Name- Name of the passenger Sex- Sex of the passenger (Male, Female) Age- Age of the passenger Sibsp- Number of Siblings/Spouses Aboard Parch- Number of Parents/Children Aboard Ticket- Ticket Number Fare- Passenger Fare (British pound) Cabin- Cabin Embarked- Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

Dataset Link- https://github.com/dsrscientist/dataset1/blob/master/titanic_train.csv

In [9]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
url = "https://raw.githubusercontent.com/dsrscientist/dataset1/master/titanic_train.csv"
titanic = pd.read_csv(url)

# Data preprocessing
titanic = titanic.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1)
titanic['Age'].fillna(titanic['Age'].median(), inplace=True)
titanic['Embarked'].fillna(titanic['Embarked'].mode()[0], inplace=True)
titanic = pd.get_dummies(titanic, columns=['Sex', 'Embarked'], drop_first=True)

# Split the dataset into features (X) and target variable (y)
X = titanic.drop('Survived', axis=1)
y = titanic['Survived']

# 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, random_state=42)

# Create a RandomForestClassifier model
model = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print("\nClassification Report:\n", report)

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()

# Feature Importances
feature_importances = model.feature_importances_
feature_importance_df = pd.DataFrame({'Feature': X.columns, 'Importance': feature_importances})
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=feature_importance_df, palette='viridis')
plt.title('Feature Importances')
plt.show()
```

```
Accuracy: 0.8212290502793296

Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.87      0.85       105
           1       0.80      0.76      0.78        74

    accuracy                           0.82       179
   macro avg       0.82      0.81      0.81       179
weighted avg       0.82      0.82      0.82       179
```



In [ ]: