```python
# Enhancing Road Safety with AI: Traffic Accident Analysis and Prediction

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report, roc_auc_score, roc_curve
import shap

# Load Dataset
df = pd.read_csv('global_road_accidents.csv')  # Replace with actual dataset path

# Data Preprocessing
df.drop_duplicates(inplace=True)
df.dropna(subset=['Severity'], inplace=True)

# Fill missing values
for col in df.select_dtypes(include=np.number).columns:
    df[col].fillna(df[col].median(), inplace=True)

# Encode categorical variables
label_encoders = {}
for col in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col].astype(str))
    label_encoders[col] = le

# Feature Scaling
features = df.drop('Severity', axis=1)
target = df['Severity']
scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)

# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(features_scaled, target, test_size=0.2, random_state=42)

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

# Predictions and Evaluation
y_pred = model.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))

# ROC Curve
y_prob = model.predict_proba(X_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_prob, pos_label=model.classes_[1])
plt.plot(fpr, tpr)
plt.title("ROC Curve")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.show()

# SHAP values
explainer = shap.Explainer(model, X_train)
shap_values = explainer(X_test[:100])
shap.plots.beeswarm(shap_values)
```