Enhancing Road Safety with AI-Driven Traffic Accident Analysis and Prediction
==========

Source Code (traffic_accident_analysis.py)

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix

# Step 1: Load Dataset
# Replace with your actual dataset path
df = pd.read_csv("traffic_accidents.csv")

# Step 2: Preview data
print("Initial Data Snapshot:")
print(df.head())

# Step 3: Drop missing values
df.dropna(inplace=True)

# Step 4: Convert severity to integer (if needed)
df['Severity'] = df['Severity'].astype(int)

# Step 5: Select relevant features
features = ['Weather', 'Road_Condition', 'Light_Condition', 'Speed_limit']
target = 'Severity'

# Convert categorical features to numerical
df = pd.get_dummies(df, columns=features)

# Define features and target
X = df.drop(target, axis=1)
y = df[target]

# Step 6: Split data
X_train,   X_test,   y_train,   y_test   =   train_test_split(X,   y,   test_size=0.2,
random_state=42)

# Step 7: Train model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Step 8: Make predictions
y_pred = model.predict(X_test)

# Step 9: Evaluation
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

```
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Step 10: Feature importance
importances = model.feature_importances_
feat_names = X.columns
indices = np.argsort(importances)[-10:]

plt.figure(figsize=(10, 6))
plt.title("Top 10 Important Features")
plt.barh(range(len(indices)), importances[indices], color='blue', align='center')
plt.yticks(range(len(indices)), [feat_names[i] for i in indices])
plt.xlabel('Relative Importance')
plt.tight_layout()
plt.show()
```