```
import pandas as pd

# Load the dataset
df = pd.read_csv('/content/road_accident_data_100.csv')
df.head()
```

	Age	Vehicle_Type	Weather	Road_Condition	Time_of_Day	Speed	Alcohol_Influence	Accident_Severity
0	56	Car	Snowy	Dry	Morning	74	Yes	Minor
1	69	Car	Rainy	Slippery	Night	52	Yes	Minor
2	46	Car	Rainy	Slippery	Morning	78	No	Moderate
3	32	Truck	Foggy	Slippery	Morning	57	Yes	Moderate
4	60	Car	Snowy	Wet	Afternoon	54	No	Severe

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
# Dataset shape & info
print("Shape:", df.shape)
df.info()
df.describe()
```

```
Shape: (100, 8)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Age                100 non-null    int64
 1   Vehicle_Type       100 non-null    object
 2   Weather            100 non-null    object
 3   Road_Condition     100 non-null    object
 4   Time_of_Day        100 non-null    object
 5   Speed              100 non-null    int64
 6   Alcohol_Influence  100 non-null    object
 7   Accident_Severity  100 non-null    object
dtypes: int64(2), object(6)
memory usage: 6.4+ KB
```

	Age	Speed
count	100.000000	100.000000
mean	43.350000	60.600000
std	14.904663	16.842107
min	19.000000	20.000000
25%	31.750000	49.750000
50%	42.000000	61.500000
75%	57.000000	73.000000
max	69.000000	107.000000

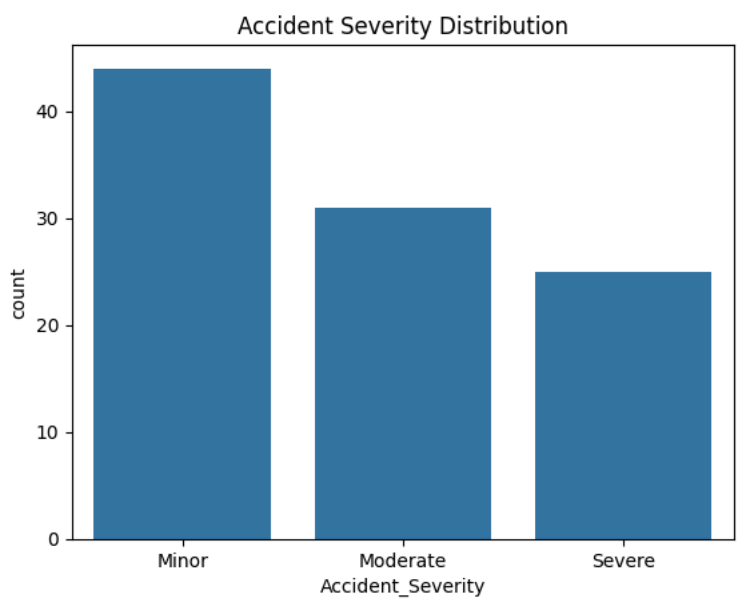
```
print("Missing values:\n", df.isnull().sum())
print("Duplicate rows:", df.duplicated().sum())
```

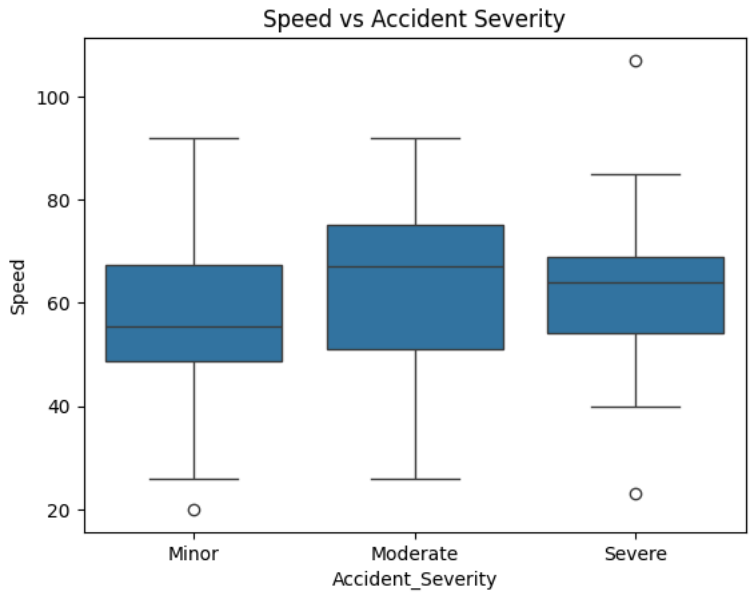
```
Missing values:
 Age                  0
Vehicle_Type         0
Weather              0
Road_Condition       0
Time_of_Day          0
Speed                0
Alcohol_Influence    0
Accident_Severity    0
dtype: int64
Duplicate rows: 0
```

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Severity count
sns.countplot(data=df, x='Accident_Severity')
plt.title("Accident Severity Distribution")
plt.show()

# Speed vs Severity
sns.boxplot(x='Accident_Severity', y='Speed', data=df)
plt.title("Speed vs Accident Severity")
plt.show()
```






```
from sklearn.preprocessing import LabelEncoder

df = df.copy()
label_encoders = {}

for col in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le

df.head()
```

	Age	Vehicle_Type	Weather	Road_Condition	Time_of_Day	Speed	Alcohol_Influence	Accident_Severity
0	56	2	3	0	2	74	1	0
1	69	2	2	1	3	52	1	0
2	46	2	2	1	2	78	0	1
3	32	3	1	1	2	57	1	1
4	60	2	3	2	0	54	0	2

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Preprocessing step
df = df.copy()
label_encoders = {}

for col in df.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le

# Feature scaling
X = df.drop(columns=['Accident_Severity'])
y = df['Accident_Severity']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)  # This is where X_scaled is defined
```

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score

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

model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.55
Classification Report:
                precision    recall  f1-score   support

           0       0.50      0.67      0.57         9
           1       0.62      0.62      0.62         8
           2       0.00      0.00      0.00         3

    accuracy                           0.55        20
   macro avg       0.38      0.43      0.40        20
weighted avg       0.47      0.55      0.51        20

/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and be
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and be
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and be
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
import joblib

joblib.dump(model, "accident_model.pkl")
joblib.dump(scaler, "accident_scaler.pkl")
joblib.dump(label_encoders, "label_encoders.pkl")
```

```
['label_encoders.pkl']
```

```
# For example, after inspecting the data:
categorical_cols = ['Alcohol_Influence', 'Road_Condition', 'Time_of_Day', 'Vehicle_Type', 'Weather']
numerical_cols = ['Speed', 'Age']  # Adjust based on actual columns

# One-hot encode sample data using same columns
sample_encoded = pd.get_dummies(sample_df[categorical_cols + numerical_cols])

# Add missing columns from training
for col in scaler.feature_names_in_:
    if col not in sample_encoded.columns:
        sample_encoded[col] = 0

# Reorder columns to match training
sample_encoded = sample_encoded[scaler.feature_names_in_]

# Apply scaler
sample_scaled = scaler.transform(sample_encoded)
sample_scaled_df = pd.DataFrame(sample_scaled, columns=scaler.feature_names_in_)

# Predict
prediction = model.predict(sample_scaled_df.values)
print("Predicted value:", prediction[0])
```

```
Predicted value: 2
```

```
!pip install gradio

import gradio as gr

def predict_severity(age, vehicle, weather, road, time, speed, alcohol):
    sample = {
        'Age': age,
        'Vehicle_Type': vehicle,
        'Weather': weather,
        'Road_Condition': road,
        'Time_of_Day': time,
        'Speed': speed,
        'Alcohol_Influence': alcohol
    }

    sample_df = pd.DataFrame([sample])
    for col in sample_df.columns:
        sample_df[col] = label_encoders[col].transform(sample_df[col])
    sample_scaled = scaler.transform(sample_df)
    pred = model.predict(sample_scaled)
    result = label_encoders['Accident_Severity'].inverse_transform(pred)[0]
    return result

gr.Interface(
    fn=predict_severity,
    inputs=[
        gr.Number(label="Age"),
        gr.Dropdown(['Car', 'Bike', 'Truck', 'Bus'], label="Vehicle Type"),
        gr.Dropdown(['Clear', 'Rainy', 'Foggy', 'Snowy'], label="Weather"),
        gr.Dropdown(['Dry', 'Wet', 'Slippery'], label="Road Condition"),
        gr.Dropdown(['Morning', 'Afternoon', 'Evening', 'Night'], label="Time of Day"),
        gr.Number(label="Speed"),
        gr.Dropdown(['Yes', 'No'], label="Alcohol Influence")
    ],
    outputs=gr.Text(label="Predicted Accident Severity"),
    title="🚗 AI Road Accident Severity Predictor",
    description="Enter conditions to predict traffic accident severity."
).launch()
```

```
Requirement already satisfied: gradio in /usr/local/lib/python3.11/dist-packages (5.29.0)
Requirement already satisfied: aiofiles<25.0,>=22.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (24.1.0)
Requirement already satisfied: anyio<5.0,>=3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.9.0)
Requirement already satisfied: fastapi<1.0,>=0.115.2 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.115.12)
Requirement already satisfied: ffmpy in /usr/local/lib/python3.11/dist-packages (from gradio) (0.5.0)
Requirement already satisfied: gradio-client==1.10.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (1.10.0)
Requirement already satisfied: groovy~=0.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.2)
Requirement already satisfied: httpx>=0.24.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.28.1)
Requirement already satisfied: huggingface-hub>=0.28.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.30.2)
Requirement already satisfied: jinja2<4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.1.6)
Requirement already satisfied: markupsafe<4.0,>=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.0.2)
Requirement already satisfied: numpy<3.0,>=1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.0.2)
Requirement already satisfied: orjson~=3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.10.18)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from gradio) (24.2)
Requirement already satisfied: pandas<3.0,>=1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.2.2)
Requirement already satisfied: pillow<12.0,>=8.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (11.2.1)
Requirement already satisfied: pydantic<2.12,>=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.11.4)
Requirement already satisfied: pydub in /usr/local/lib/python3.11/dist-packages (from gradio) (0.25.1)
Requirement already satisfied: python-multipart>=0.0.18 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.0.20)
Requirement already satisfied: pyyaml<7.0,>=5.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (6.0.2)
Requirement already satisfied: ruff>=0.9.3 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.11.8)
Requirement already satisfied: safehttpx<0.2.0,>=0.1.6 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.6)
Requirement already satisfied: semantic-version~=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.10.0)
Requirement already satisfied: starlette<1.0,>=0.40.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.46.2)
Requirement already satisfied: tomlkit<0.14.0,>=0.12.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.13.2)
Requirement already satisfied: typer<1.0,>=0.12 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.15.3)
Requirement already satisfied: typing-extensions~=4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.13.2)
Requirement already satisfied: uvicorn>=0.14.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.34.2)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from gradio-client==1.10.0->gradio) (2025.3.2)
Requirement already satisfied: websockets<16.0,>=10.0 in /usr/local/lib/python3.11/dist-packages (from gradio-client==1.10.0->gradio) (1
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio) (3.10)
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio) (1.3.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio) (2025.4.26)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio) (1.0.9)
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.11/dist-packages (from httpcore==1.*->httpx>=0.24.1->gradio) (0.16.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (3.18.0)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (2.32.3)
Requirement already satisfied: tqdm>=4.42.1 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (4.67.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2.9.0.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2025.2)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (0.7
Requirement already satisfied: pydantic-core==2.33.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (2.33
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (0
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (8.1.8)
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (1.5.4)
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (13.9.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas<3.0,>=1.0->gradi
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,>=0.12->g
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,>=0.12-
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hub>=0.28
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hub>=0.28.1->gr
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich>=10.11.0->typer<1
It looks like you are running Gradio on a hosted a Jupyter notebook. For the Gradio app to work, sharing must be enabled. Automatically

Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://99f0ef3a8555931bdb.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal in the working dir
```

🚚 AI Road Accident Severity Predictor

Enter conditions to predict traffic accident severity.

Age

0

Vehicle Type

Car

Weather

Clear

Road Condition

Dry

Predicted Accident Severity

Flag

Time of Day