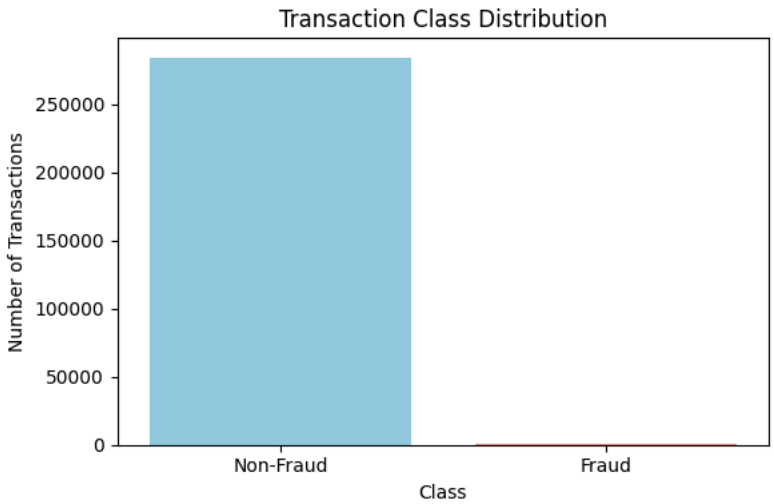
```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Sample data
fraud_counts = pd.Series([284315, 492], index=["Non-Fraud", "Fraud"])

# Plot
plt.figure(figsize=(6, 4))
sns.barplot(x=fraud_counts.index, y=fraud_counts.values, palette=["skyblue", "salmon"])
plt.title("Transaction Class Distribution")
plt.xlabel("Class")
plt.ylabel("Number of Transactions")
plt.tight_layout()
plt.show()
```

```
<ipython-input-1-f5302bf384e7>:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `

  sns.barplot(x=fraud_counts.index, y=fraud_counts.values, palette=["skyblue", "salmon"])
```




```
import matplotlib.pyplot as plt

# Sample data
labels = ['Non-Fraud', 'Fraud']
sizes = [284315, 492]
colors = ['lightgreen', 'lightcoral']
explode = (0, 0.1)  # Slightly "explode" the fraud slice

# Plotting the pie chart
plt.figure(figsize=(6, 6))
plt.pie(sizes, labels=labels, autopct='%1.2f%%', startangle=140, colors=colors, explode=explode, shadow=True)
plt.title('Fraud vs Non-Fraud Transaction Distribution')
plt.axis('equal')  # Equal aspect ratio ensures the pie chart is circular.
plt.show()
```

Explain •••

Please explain this code.

Code Explana

This section of the code is specifically designed for users working in **Google Colab**, a cloud-based Jupyter notebook environmen

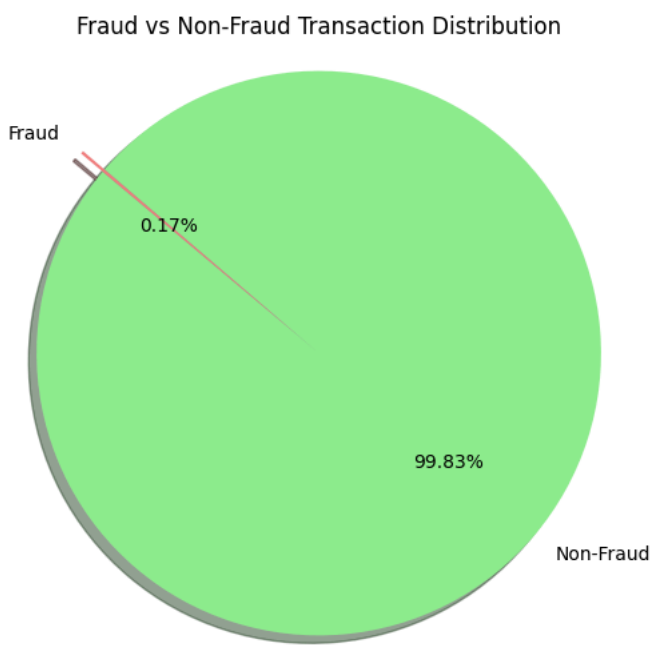
```
from goo
```

Use code with caution

This line imports the `files` module from the `google.co` library. The `google.co` library provides tools and utilities that are specific to the Google Colab environmen The `files` module contains functions for interacting with the user's local filesystem, allowing them to upload and download files directly within the notebook.

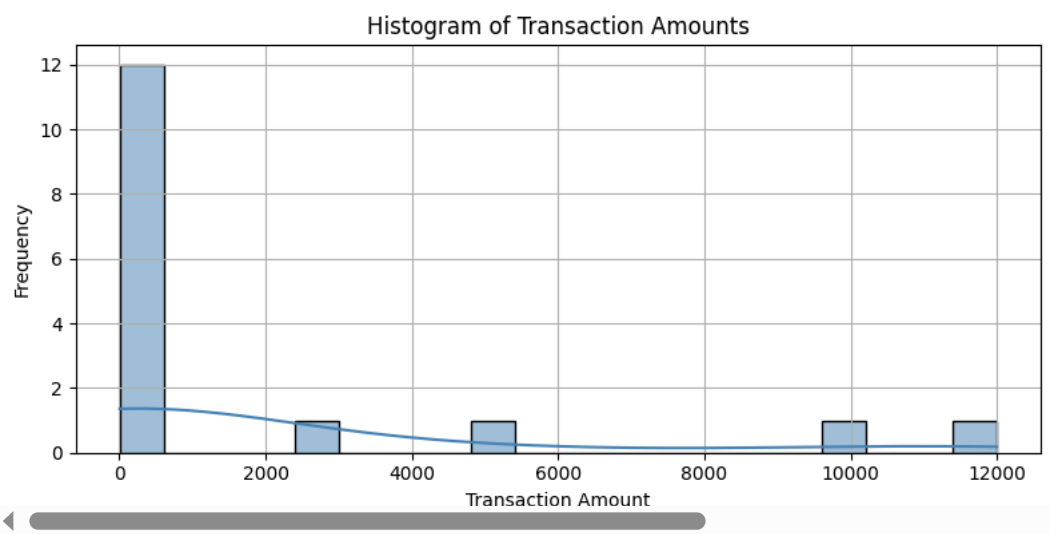
```
uploaded
```

Use code with caution



Start coding or generate with AI.

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

# Sample transaction amount data (replace this with your actual data)
transaction_amounts = pd.Series([10, 15, 20, 30, 100, 200, 500, 5000, 12000, 30, 40, 5, 8, 3000, 250, 10000])

# Plotting the histogram
plt.figure(figsize=(8, 4))
sns.histplot(transaction_amounts, bins=20, kde=True, color='steelblue')
plt.title("Histogram of Transaction Amounts")
plt.xlabel("Transaction Amount")
plt.ylabel("Frequency")
plt.grid(True)
plt.tight_layout()
plt.show()
```




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

# Sample dataset (replace with your actual dataset)
data = {
    'transaction_amount': [10, 15, 20, 30, 5000, 10000, 8, 12, 100, 3000, 250, 200],
    'is_fraud': [0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0]  # 0 = Non-Fraud, 1 = Fraud
}
df = pd.DataFrame(data)

# Plotting the boxplot
```

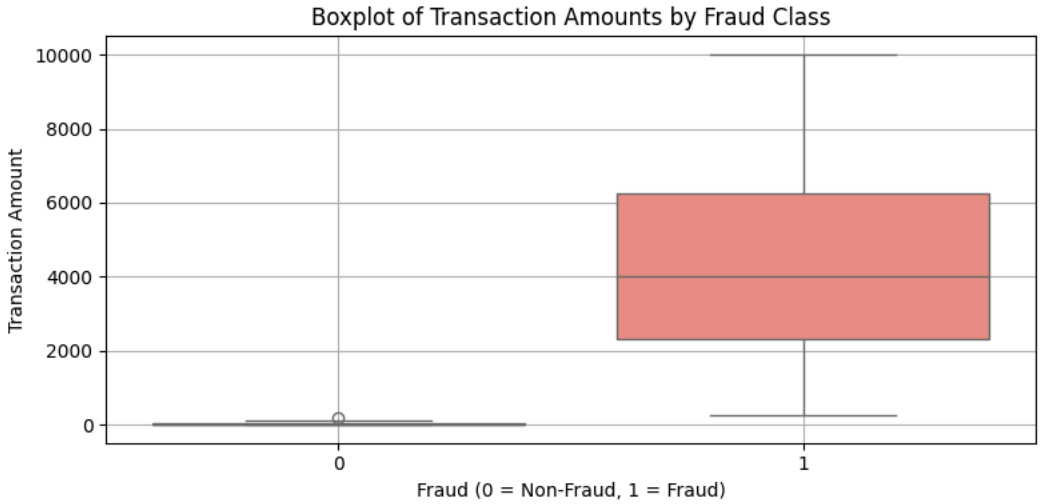
This line calls the `upload()` function from the `files` module. When this line is executed in a Google Colab notebook, it will trigger a file upload dialog box in your web browser. This dialog allows you to select one or more files from your local computer to upload to the Colab environmen

The files you select will be uploaded to the current working directory of your Colab notebook. The `files.upl` function returns a dictionary called `uploaded`. The keys of this dictionary are the filenames of the uploaded files, and the values are the file contents as bytes.

In summary, this code block enables you to easily transfer files from

```
plt.figure(figsize=(8, 4))
sns.boxplot(x='is_fraud', y='transaction_amount', data=df, palette=["skyblue", "salmon"])
plt.title("Boxplot of Transaction Amounts by Fraud Class")
plt.xlabel("Fraud (0 = Non-Fraud, 1 = Fraud)")
plt.ylabel("Transaction Amount")
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
<ipython-input-4-2193415ccda7>:14: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `

  sns.boxplot(x='is_fraud', y='transaction_amount', data=df, palette=["skyblue", "salmon"])
```




```
from google.colab import files
uploaded = files.upload()
```

Choose files transactions.csv

- **transactions.csv**(text/csv) - 18253 bytes, last modified: 16/05/2025 - 100% done

New section

New section

```
!pip install matplotlib-venn
```

```
Requirement already satisfied: matplotlib-venn in /usr/local/lib/python3.11/dist-packages (1.1.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (from matplotlib-venn) (3.10.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (from matplotlib-venn) (2.0.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.11/dist-packages (from matplotlib-venn) (1.15.3)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->matplot
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib->matplotlib-
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib->matplo
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->matplo
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib->matplotl
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib->matplotlib-ven
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib->matplot
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib->mat
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matpl
```

```
!apt-get -qq install -y libfluidsynth1
```

```
E: Package 'libfluidsynth1' has no installation candidate
```

```
# https://pypi.python.org/pypi/libarchive
!apt-get -qq install -y libarchive-dev && pip install -U libarchive
import libarchive
```

```
Selecting previously unselected package libarchive-dev:amd64.
(Reading database ... 126102 files and directories currently installed.)
```

your local machine into your Google Colab notebook environmen which is often necessary for loading datasets or other resources needed for your data analysis or machine learning tasks.

▶ Sources

Rate this answer

```
Preparing to unpack .../libarchive-dev_3.6.0-1ubuntu1.4_amd64.deb ...
Unpacking libarchive-dev:amd64 (3.6.0-1ubuntu1.4) ...
Setting up libarchive-dev:amd64 (3.6.0-1ubuntu1.4) ...
Processing triggers for man-db (2.10.2-1) ...
Collecting libarchive
  Downloading libarchive-0.4.7.tar.gz (23 kB)
  Preparing metadata (setup.py) ... done
Collecting nose (from libarchive)
  Downloading nose-1.3.7-py3-none-any.whl.metadata (1.7 kB)
Downloading nose-1.3.7-py3-none-any.whl (154 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 154.7/154.7 kB 13.4 MB/s eta 0:00:00
Building wheels for collected packages: libarchive
  Building wheel for libarchive (setup.py) ... done
  Created wheel for libarchive: filename=libarchive-0.4.7-py3-none-any.whl size=31629 sha256=5c573d55f4174a85b7f460b
  Stored in directory: /root/.cache/pip/wheels/32/98/bd/4893d6923dd027f455b250367d402bfd69a6f4416581df46db
Successfully built libarchive
Installing collected packages: nose, libarchive
Successfully installed libarchive-0.4.7 nose-1.3.7
```

```
# https://pypi.python.org/pypi/pydot
!apt-get -qq install -y graphviz && pip install pydot
import pydot
```

```
Requirement already satisfied: pydot in /usr/local/lib/python3.11/dist-packages (3.0.4)
Requirement already satisfied: pyparsing>=3.0.9 in /usr/local/lib/python3.11/dist-packages (from pydot) (3.2.3)
```

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_auc_score
import gradio as gr

# 1. Load data
df = pd.read_csv('transactions.csv')

# 2. Preprocessing
df.drop_duplicates(inplace=True)
df.dropna(inplace=True)  # Or use imputers as needed

# Encode 'card_present' early for consistency
df['card_present'] = df['card_present'].map({'Yes': 1, 'No': 0})

# Create transaction hour group
df['transaction_hour_group'] = pd.cut(
    df['transaction_time'],
    bins=[0, 6, 12, 18, 24],
    labels=['Night', 'Morning', 'Afternoon', 'Evening'],
    right=False
)

# Define features and target
X = df.drop(columns=['is_fraud', 'transaction_id'])
y = df['is_fraud']

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.15, stratify=y, random_state=42
)

# Preprocessing pipelines
numeric_features = ['transaction_amount', 'account_age_days', 'transaction_time']
categorical_features = ['merchant_category', 'device_type', 'transaction_hour_group', 'card_present']

numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='mean')),
    ('scaler', StandardScaler())
])

categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

preprocessor = ColumnTransformer(transformers=[
    ('num', numeric_transformer, numeric_features),
    ('cat', categorical_transformer, categorical_features)
```

```
])

# Models
models = {
    'LogisticRegression': LogisticRegression(class_weight='balanced', max_iter=1000, random_state=42),
    'RandomForest': RandomForestClassifier(n_estimators=200, class_weight='balanced', random_state=42)
}

trained_models = {}
for name, model in models.items():
    clf = Pipeline(steps=[('preproc', preprocessor), ('clf', model)])
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    y_proba = clf.predict_proba(X_test)[:, 1]
    print(f"--- {name} Classification Report ---")
    print(classification_report(y_test, y_pred, digits=4))
    print(f"ROC AUC: {roc_auc_score(y_test, y_proba):.4f}\n")
    trained_models[name] = clf

# Choose best model for deployment
best_model = trained_models['RandomForest']

# Gradio interface
def predict_fraud(amount, time, merchant, device, present):
    try:
        time = min(max(0, float(time)), 23)
        amount = float(amount)
        present = 1 if present == 'Yes' else 0
        hour_group = pd.cut(
            [time], bins=[0, 6, 12, 18, 24],
            labels=['Night', 'Morning', 'Afternoon', 'Evening'], right=False
        )[0]

        data = pd.DataFrame([{
            'transaction_amount': amount,
            'account_age_days': 365,
            'transaction_time': time,
            'merchant_category': merchant,
            'device_type': device,
            'transaction_hour_group': hour_group,
            'card_present': present
        }])

        proba = best_model.predict_proba(data)[0, 1]
        return f"{proba:.2f}"
    except Exception as e:
        return f"Error: {str(e)}"

examples = [
    [120.50, 14, 'Food', 'Mobile', 'Yes'],
    [400.00, 23, 'Travel', 'Web', 'No']
]

iface = gr.Interface(
    fn=predict_fraud,
    inputs=[
        gr.Number(label="Transaction Amount"),
        gr.Number(label="Transaction Time (0–23)"),
        gr.Dropdown(choices=['Retail', 'Food', 'Travel', 'Other'], label="Merchant Category"),
        gr.Dropdown(choices=['Mobile', 'Web', 'POS'], label="Device Type"),
        gr.Radio(choices=['Yes', 'No'], label="Card Present")
    ],
    outputs=gr.Textbox(label="Fraud Risk Score"),
    title="AI-Powered Credit Card Fraud Detector",
    description="Enter transaction details to get a real-time fraud risk score.",
    examples=examples
)

if _name_ == "_main_":
    iface.launch(share=True)
```

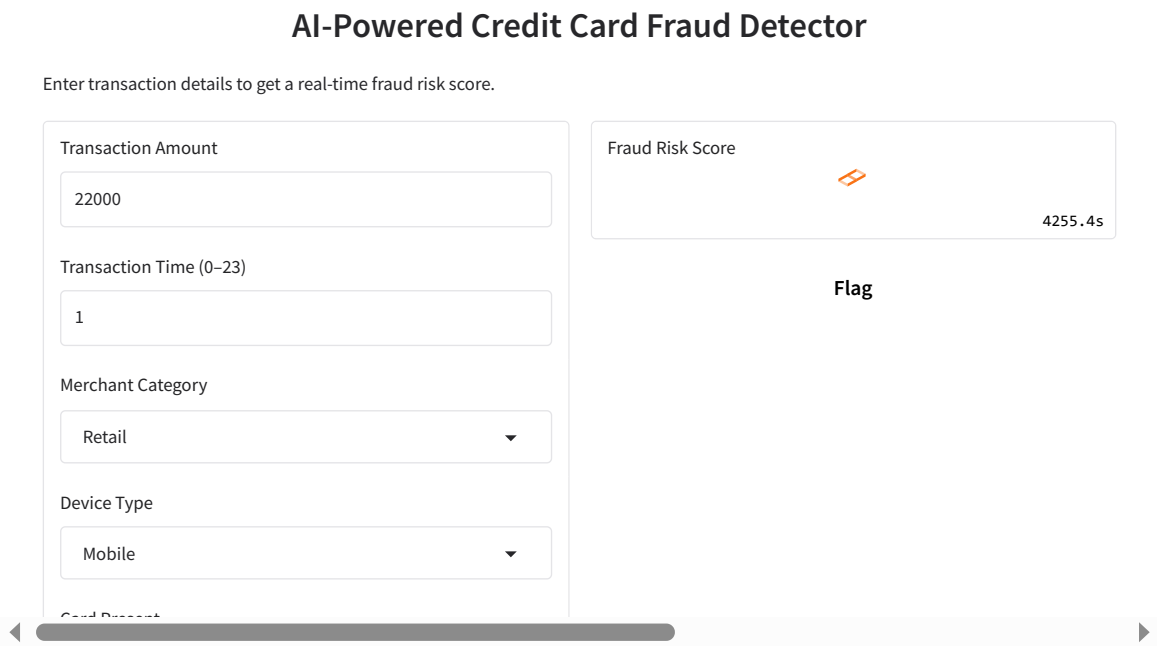
```
/usr/local/lib/python3.11/dist-packages/sklearn/impute/_base.py:635: UserWarning: Skipping features without any obse
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/sklearn/impute/_base.py:635: UserWarning: Skipping features without any obse
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/sklearn/impute/_base.py:635: UserWarning: Skipping features without any obse
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/sklearn/impute/_base.py:635: UserWarning: Skipping features without any obse
  warnings.warn(
--- LogisticRegression Classification Report ---
              precision    recall  f1-score   support

           0     0.9423    0.6901    0.7967        71
           1     0.0435    0.2500    0.0741         4

    accuracy                         0.6667        75
   macro avg     0.4929    0.4701    0.4354        75
weighted avg     0.8944    0.6667    0.7582        75

ROC AUC: 0.4754

/usr/local/lib/python3.11/dist-packages/sklearn/impute/_base.py:635: UserWarning: Skipping features without any obse
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/sklearn/impute/_base.py:635: UserWarning: Skipping features without any obse
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision i
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision i
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision i
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
--- RandomForest Classification Report ---
              precision    recall  f1-score   support

           0     0.9467    1.0000    0.9726        71
           1     0.0000    0.0000    0.0000         4

    accuracy                         0.9467        75
   macro avg     0.4733    0.5000    0.4863        75
weighted avg     0.8962    0.9467    0.9207        75

ROC AUC: 0.4560

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-11-49c049e38e92> in <cell line: 0>()
    123 )
    124
--> 125 if _name_ == "_main_":
    126     iface.launch(share=True)

NameError: name '_name_' is not defined
```

Next steps: Explain error

```
# Gradio interface
def predict_fraud(amount, time, merchant, device, present):
    try:
        time = min(max(0, float(time)), 23)
        amount = float(amount)
        present = 1 if present == 'Yes' else 0
        hour_group = pd.cut(
            [time], bins=[0, 6, 12, 18, 24],
            labels=['Night', 'Morning', 'Afternoon', 'Evening'], right=False
        )[0]

        data = pd.DataFrame([{
            'transaction_amount': amount,
            'account_age_days': 365,
            'transaction_time': time,
            'merchant_category': merchant,
            'device_type': device,
            'transaction_hour_group': hour_group,
            'card_present': present
        }])

        proba = best_model.predict_proba(data)[0, 1]
        return f"{proba:.2f}"
    except Exception as e:
        return f"Error: {str(e)}"

examples = [
```

```
    [120.50, 14, 'Food', 'Mobile', 'Yes'],
    [400.00, 23, 'Travel', 'Web', 'No']
]

iface = gr.Interface(
    fn=predict_fraud,
    inputs=[
        gr.Number(label="Transaction Amount"),
        gr.Number(label="Transaction Time (0–23)"),
        gr.Dropdown(choices=['Retail', 'Food', 'Travel', 'Other'], label="Merchant Category"),
        gr.Dropdown(choices=['Mobile', 'Web', 'POS'], label="Device Type"),
        gr.Radio(choices=['Yes', 'No'], label="Card Present")
    ],
    outputs=gr.Textbox(label="Fraud Risk Score"),
    title="AI-Powered Credit Card Fraud Detector",
    description="Enter transaction details to get a real-time fraud risk score.",
    examples=examples
)

# Fix the typo: __name__ instead of _name_
if __name__ == "__main__":
    iface.launch(share=True)
```

```
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://6f0499a1b671e281a1.gradio.live

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




```
!pip install gradio
```

```
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.
```

```
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->g
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->hugg
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0-
Downloading gradio-5.29.1-py3-none-any.whl (54.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.1/54.1 MB 18.4 MB/s eta 0:00:00
Downloading gradio_client-1.10.1-py3-none-any.whl (323 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 323.1/323.1 kB 22.3 MB/s eta 0:00:00
Downloading aiofiles-24.1.0-py3-none-any.whl (15 kB)
Downloading fastapi-0.115.12-py3-none-any.whl (95 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 95.2/95.2 kB 8.8 MB/s eta 0:00:00
Downloading groovy-0.1.2-py3-none-any.whl (14 kB)
Downloading python_multipart-0.0.20-py3-none-any.whl (24 kB)
Downloading ruff-0.11.10-py3-none-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.6 MB)
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.

Enter

0/2000

Gemini can make mistakes, so double-check responses and use code with caution. Learn more