```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
import re


data = {
    "review": [
        "I love this product, it works great!",
        "Terrible experience, I want a refund.",
        "Excellent quality, highly recommend.",
        "Not good, broke after one use.",
        "Amazing service and fast delivery!",
        "Worst purchase I've ever made.",
        "Really satisfied with this item.",
        "Disappointed and unhappy with it.",
        "Great value for the price.",
        "Waste of money and time."
    ],
    "sentiment": [1, 0, 1, 0, 1, 0, 1, 0, 1, 0]
}

df = pd.DataFrame(data)
df.to_csv("sample_reviews.csv", index=False)
df.head()
```

	review	sentiment
0	I love this product, it works great!	1
1	Terrible experience, I want a refund.	0
2	Excellent quality, highly recommend.	1
3	Not good, broke after one use.	0
4	Amazing service and fast delivery!	1

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

```
def preprocess(text):
    text = text.lower()
    text = re.sub(r"[^\w\s]", "", text)
    text = re.sub(r"\d+", "", text)
    return text

df["cleaned_review"] = df["review"].apply(preprocess)
df.head()
```

	review	sentiment	cleaned_review
0	I love this product, it works great!	1	i love this product it works great
1	Terrible experience, I want a refund.	0	terrible experience i want a refund
2	Excellent quality, highly recommend.	1	excellent quality highly recommend
3	Not good, broke after one use.	0	not good broke after one use
4	Amazing service and fast delivery!	1	amazing service and fast delivery

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

```
tfidf = TfidfVectorizer(max_features=5000)
X = tfidf.fit_transform(df["cleaned_review"]).toarray()
y = df["sentiment"]

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


model = LogisticRegression()
model.fit(X_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
y_pred = model.predict(X_test)
```

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

```
Accuracy: 1.0

Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         1

    accuracy                           1.00         2
   macro avg       1.00      1.00      1.00         2
weighted avg       1.00      1.00      1.00         2
```

Start coding or generate with AI.