Source Code:

```
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.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report, roc_auc_score

# 1. Sample Dataset (You can replace with a full CSV later)
data = {
    'text': [
        'The economy is booming under new policy.',
        'Aliens have landed in the White House.',
        'Vaccines cause serious harm.',
        'Scientists discovered a new planet.',
        'The earth is flat.',
        'Government confirms tax cuts.',
        'You can teleport using crystals.',
        'Elections results were manipulated.'
    ],
    'label': [0, 1, 1, 0, 1, 0, 1, 1]  # 0 = Real, 1 = Fake
}

df = pd.DataFrame(data)

# 2. Text Vectorization
vectorizer = TfidfVectorizer(stop_words='english')
X = vectorizer.fit_transform(df['text'])
y = df['label']
```

```
# 3. Train/Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# 4. Train Model
model = LogisticRegression()
model.fit(X_train, y_train)

# 5. Prediction Output
y_pred = model.predict(X_test)
y_prob = model.predict_proba(X_test)[:, 1]

print("\n=== PREDICTION OUTPUT ===")
for i, text in enumerate(df['text'].iloc[y_test.index]):
    print(f"Text: {text}\nActual: {'Fake' if y_test.iloc[i] else 'Real'} | Predicted: {'Fake' if y_pred[i] else 'Real'}\n")

# 6. Confusion Matrix (Matrix Display)
cm = confusion_matrix(y_test, y_pred)
cm_df = pd.DataFrame(cm, index=['Real', 'Fake'], columns=['Predicted Real', 'Predicted Fake'])
print("\n=== CONFUSION MATRIX ===")
print(cm_df)

# 7. Heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(cm_df, annot=True, cmap='coolwarm', fmt='d')
plt.title('Confusion Matrix Heatmap')
plt.tight_layout()
plt.show()

# 8. KDE Plot of prediction probabilities
```

```
plt.figure(figsize=(6, 4))
sns.kdeplot(y_prob, fill=True, color='orange')
plt.title('KDE Plot of Fake News Prediction Probabilities')
plt.xlabel('Probability of Fake News')
plt.tight_layout()
plt.show()

# 9. Box Plot: Text length by label
df['text_length'] = df['text'].apply(lambda x: len(x.split()))
plt.figure(figsize=(7, 5))
sns.boxplot(data=df, x='label', y='text_length', palette='Set2')
plt.xticks([0, 1], ['Real', 'Fake'])
plt.title('Box Plot: Text Length by News Label')
plt.tight_layout()
plt.show()

# 10. Line Plot: Accuracy trend (simulated over increasing samples)
samples = list(range(1, len(X_train.toarray()) + 1))
accuracies = [accuracy_score(y_train[:i], model.predict(X_train[:i])) for i in samples if i > 1]

plt.figure(figsize=(7, 4))
plt.plot(samples[1:], accuracies, marker='o', color='green')
plt.title('Line Plot: Accuracy Trend with More Data')
plt.xlabel('Training Samples')
plt.ylabel('Training Accuracy')
plt.grid(True)
plt.tight_layout()
plt.show()
```