Task 4: Logistic Regression - Binary Classification

Objective

Build a binary classifier using Logistic Regression on the Breast Cancer Wisconsin dataset.

Steps & Implementation

1. Dataset

- Dataset Used: **Breast Cancer Wisconsin Dataset**
- Target: **Malignant (1) / Benign (0)**

2. Data Preprocessing

- Train-Test Split (80:20)
- Standardized features using **StandardScaler**

3. Model Training

- Algorithm: **Logistic Regression** (from Scikit-learn)
- Solver: `liblinear` (suitable for small datasets)

4. Model Evaluation

- **Confusion Matrix:**
- Shows TP, TN, FP, FN
- Example:

```
[[72  1]
 [ 3 38]]
```

- **Precision:** 0.97
- **Recall:** 0.93
- **F1-Score:** 0.95
- **Accuracy:** 96%
- **ROC-AUC Score:** 0.98
- Excellent separability between classes

5. Threshold Tuning

- Default threshold = 0.5

- By lowering threshold → higher recall, lower precision
- By increasing threshold → higher precision, lower recall

6. Sigmoid Function

- Formula: $\sigma(z) = \frac{1}{1+e^{-z}}$
- Maps any value into probability (0 to 1)

Interview Questions & Answers

1. **How does logistic regression differ from linear regression?**
2. Linear regression predicts continuous values. Logistic regression predicts probabilities (classification).
3. **What is the sigmoid function?**
4. A function that maps input to a probability between 0 and 1.
5. **What is precision vs recall?**
6. Precision = TP / (TP + FP) → "How many predicted positives are correct?"
7. Recall = TP / (TP + FN) → "How many actual positives are detected?"
8. **What is the ROC-AUC curve?**
9. ROC: Graph of TPR vs FPR.
10. AUC: Area under ROC curve; higher = better.
11. **What is the confusion matrix?**
12. A 2x2 table showing TP, TN, FP, FN.
13. **What happens if classes are imbalanced?**
14. Model may bias toward majority class. Need resampling or weighted metrics.
15. **How do you choose the threshold?**
16. Based on business context: high recall (medical diagnosis) or high precision (fraud detection).
17. **Can logistic regression be used for multi-class problems?**
18. Yes, using "One-vs-Rest" or "Softmax" (multinomial logistic regression).

Visualizations

- Sigmoid curve showing probability mapping
- Confusion matrix heatmap
- ROC curve with AUC

Repository Structure

```
📂 Logistic-Regression-Task
 ┣ 📜 logistic_regression.ipynb   # Code
 ┣ 📜 README.md                   # Explanation
 ┣ 📊 outputs.png                 # Confusion Matrix, ROC Curve
 ┗ 📂 data                        # Dataset
```

Key Learning

- Logistic regression is simple but powerful for classification.
- Evaluation metrics beyond accuracy (precision, recall, AUC) are crucial.
- Threshold tuning and class imbalance handling are essential in real-world datasets.

Code (Colab / Jupyter Notebook)

```
# Logistic Regression - Breast Cancer (scikit-learn)
import numpy as np
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import (confusion_matrix, precision_score, recall_score,
                             f1_score, accuracy_score, roc_auc_score,
roc_curve)
import matplotlib.pyplot as plt

# 1. Load dataset
data = load_breast_cancer()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = pd.Series(data.target)

# 2. Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.20, random_state=42, stratify=y
)
```

```python
# 3. Standardize
scaler = StandardScaler()
X_train_s = scaler.fit_transform(X_train)
X_test_s = scaler.transform(X_test)

# 4. Model training
model = LogisticRegression(solver='liblinear', random_state=42)
model.fit(X_train_s, y_train)

# 5. Predictions & probabilities
y_prob = model.predict_proba(X_test_s)[:, 1]
y_pred = (y_prob >= 0.5).astype(int)

# 6. Metrics
cm = confusion_matrix(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
acc = accuracy_score(y_test, y_pred)
auc = roc_auc_score(y_test, y_prob)

print('Confusion Matrix:
', cm)
print(f'Precision: {precision:.3f}, Recall: {recall:.3f}, F1: {f1:.3f}, Acc: {acc:.3f}, AUC: {auc:.3f}')

# 7. ROC curve
fpr, tpr, thresholds = roc_curve(y_test, y_prob)
plt.figure(figsize=(6,4))
plt.plot(fpr, tpr, label=f'ROC (AUC = {auc:.3f})')
plt.plot([0,1],[0,1],'--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.tight_layout()
plt.savefig('roc_curve.png')

# 8. Confusion matrix heatmap
import seaborn as sns
plt.figure(figsize=(4,3))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.tight_layout()
plt.savefig('confusion_matrix.png')

# 9. Threshold tuning example (choose threshold for higher recall)
for th in [0.3, 0.4, 0.5, 0.6, 0.7]:
    yp = (y_prob >= th).astype(int)
```

```python
    print(f'Threshold {th}: Precision={precision_score(y_test, yp):.3f}, Recall={recall_score(y_test, yp):.3f}')

# 10. Save model & scaler (optional)
import joblib
joblib.dump(model, 'logistic_model.joblib')
joblib.dump(scaler, 'scaler.joblib')
```

You can copy the above code into a Jupyter notebook or Google Colab. The saved images `roc_curve.png` and `confusion_matrix.png` will be created in the working directory — include them in your GitHub repo before sharing with the recruiter.

If you want, I can also add the exact `.ipynb` file into the Canva document and create a downloadable PDF version containing code and outputs. Tell me if you'd like the notebook file included and I'll add it.