Project 2 – Perceptron Classifier

Team Members: Isabella, Hudson, Jacob, Hayat

Student Certification

Team Member 1

- Print Name: Isabella Darko
- Date: 9/30/2025
- I have contributed by doing the following: I did part 1 data set B and set up the pdf where we put everything
- Signed: *Isabella Darko*

Team Member 2

- Print Name: Hudson Finocchio
- Date: 10/4/2025
- I have contributed by doing the following: Trained perceptron-based classifier on normalized and split dataset C.
- Signed: *Isabella Darko*

Team Member 3

- Print Name:
- Date:
- I have contributed by doing the following:
- Signed:

Team Member 4

- Print Name:
- Date:
- I have contributed by doing the following:
- Signed:

Part 1 – Perceptron-based Classifier

In this section the full pipeline is detailed. The pipeline is a consolidation of the code of several members. This includes loading, normalizing, splitting, and training on the data.

Load Data

```
# imports
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from datetime import datetime
GLOBAL_SEED = 123

def load_data(pathname: str) -> pd.DataFrame:
    plt.style.use('default')
    rng = np.random.RandomState(GLOBAL_SEED)

    # set dataset path for reproducibility

    path_to_dataset = pathname

    # load csv
    df = pd.read_csv(path_to_dataset, header=None)
    ncols = df.shape[1]
    # setting last column to label (small or big car)
    feature_cols = list(range(ncols - 1))
    label_col = ncols - 1
    df.columns = [f"f{i+1}" for i in range(ncols - 1)] + ["label"]
    df["label"] = df["label"].astype(int)
    return df
```

Helper for Results Comparison

```
def save_confusion_and_metrics(metrics, dataset_tag, mode, split_fraction, prefix='C'):
    split_tag = str(int(split_fraction * 100))
    filename = f"reports/{prefix}_{mode}_{split_tag}_metrics.txt"
    with open(filename, 'w') as f:
        f.write(f"### Dataset {prefix} - {mode.capitalize()} Activation ({split_tag}/{100-int
        f.write("**Confusion Matrix (Testing Data)**\n\n")
        f.write("|                 | Predicted Positive | Predicted Negative |\n")
        f.write("| --------------- | ------------------ | ------------------ |\n")
        f.write(f"| Actual Positive | TP = {metrics['TP']} | FN = {metrics['FN']} |\n")
        f.write(f"| Actual Negative | FP = {metrics['FP']} | TN = {metrics['TN']} |\n\n")
        f.write("**Rates**\n\n")
        f.write(f"- Accuracy: {metrics['accuracy']:.4f}\n")
        f.write(f"- True Positive Rate (Recall): {metrics['recall']:.4f}\n")
        f.write(f"- False Positive Rate: {metrics['fpr']:.4f}\n")
        f.write(f"- Precision: {metrics['precision']:.4f}\n")
        f.write(f"- F1 Score: {metrics['f1']:.4f}\n\n")
        f.write(f"_Generated on {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}_\n")
    return filename
```

Perceptron Model Pipeline

```
def normalize(X):
    scaler = StandardScaler()
    Xs = scaler.fit_transform(X)
    return Xs, scaler

def add_bias(X):
    return np.hstack([np.ones((X.shape[0], 1)), X])

def init_weights(n_features, seed=None):
    rng_local = np.random.RandomState(seed if seed is not None else GLOBAL_SEED)
    return rng_local.uniform(-0.5, 0.5, n_features)

def hard_activation(net):
    return (net >= 0).astype(int)

def sigmoid(net, gain=1.0):
    z = np.clip(net, -60, 60)
    return 1.0 / (1.0 + np.exp(-gain * z))
```

```
def train_perceptron_hard(X, y, alpha=0.01, ni=5000, epsilon=700, seed=None):
    n_samples, n_features = X.shape
    w = init_weights(n_features, seed)
    TE_history = []
    for it in range(ni):
        total_error = 0
        for xi, di in zip(X, y):
            net = np.dot(w, xi)
            yi = 1 if net >= 0 else 0
            err = di - yi
            if err != 0:
                w = w + alpha * err * xi
            total_error += abs(err)
        TE_history.append(total_error)
        if total_error < epsilon:
            break
    return w, TE_history, it+1

def train_perceptron_soft(X, y, alpha=0.005, gain=0.5, ni=5000, epsilon=700, seed=None):
    n_samples, n_features = X.shape
    w = init_weights(n_features, seed)
    TE_history = []
    for it in range(ni):
        total_error = 0.0
        for xi, di in zip(X, y):
            net = np.dot(w, xi)
            yi = sigmoid(net, gain)
            err = di - yi
            deriv = gain * yi * (1 - yi)
            w = w + alpha * err * deriv * xi
            total_error += 0.5 * (err ** 2)
        TE_history.append(total_error)
        if total_error < epsilon:
            break
    return w, TE_history, it+1

def predict(w, X, mode='hard', gain=1.0):
    net = X.dot(w)
    if mode == 'hard':
        return hard_activation(net)
    return (sigmoid(net, gain) >= 0.5).astype(int)
```

```
def compute_metrics(y_true, y_pred):
    TP = int(((y_true==1)&(y_pred==1)).sum())
    FN = int(((y_true==1)&(y_pred==0)).sum())
    FP = int(((y_true==0)&(y_pred==1)).sum())
    TN = int(((y_true==0)&(y_pred==0)).sum())
    acc = accuracy_score(y_true, y_pred)
    rec = recall_score(y_true, y_pred, zero_division=0)
    prec = precision_score(y_true, y_pred, zero_division=0)
    f1 = f1_score(y_true, y_pred, zero_division=0)
    fpr = FP / (FP + TN) if (FP+TN)>0 else 0
    return {'TP':TP,'FN':FN,'FP':FP,'TN':TN,
            'accuracy':acc,'recall':rec,'precision':prec,'f1':f1,'fpr':fpr}
```

Visualization Helper

```
def plot_2d_pca(X_full, y, w_full, pathname, title='', mode='hard'):
    # need pca projection for 2D plotting
    pca = PCA(n_components=2)
    X2 = pca.fit_transform(X_full)
    plt.figure(figsize=(6,5))
    plt.scatter(X2[y==1,0], X2[y==1,1], marker='o', label='Positive', alpha=0.7)
    plt.scatter(X2[y==0,0], X2[y==0,1], marker='x', label='Negative', alpha=0.7)
    plt.title(f"{title} (PCA projection)")
    plt.xlabel("PCA-1")
    plt.ylabel("PCA-2")
    plt.legend()
    plt.tight_layout()
    plt.savefig(pathname, dpi=150)
    plt.close()
```

Experiment Runner

```
def run_experiment(df, split_fraction=0.75, mode='hard', alpha=0.01, gain=0.5,
                   epsilon=700, ni=5000, seed=GLOBAL_SEED, save_prefix='C'):
    """
    Runs a perceptron experiment for the given dataset and configuration.
    Saves plots and metrics reports in ready-to-include formats.
    """
```

```
    X_raw = df.iloc[:, :-1].values.astype(float)
    y = df['label'].values.astype(int)
    Xs, scaler = normalize(X_raw)

    stratify = y if len(np.unique(y)) > 1 else None
    X_train, X_test, y_train, y_test = train_test_split(
        Xs, y, train_size=split_fraction, stratify=stratify, random_state=seed)

    X_train_b = add_bias(X_train)
    X_test_b = add_bias(X_test)

    # train model
    if mode == 'hard':
        w, TE_hist, iters = train_perceptron_hard(
            X_train_b, y_train, alpha=alpha, ni=ni, epsilon=epsilon, seed=seed)
    else:
        w, TE_hist, iters = train_perceptron_soft(
            X_train_b, y_train, alpha=alpha, gain=gain, ni=ni, epsilon=epsilon, seed=seed)

    # predictions
    y_train_pred = predict(w, X_train_b, mode=mode, gain=gain)
    y_test_pred = predict(w, X_test_b, mode=mode, gain=gain)

    # metrics
    metrics_train = compute_metrics(y_train, y_train_pred)
    metrics_test = compute_metrics(y_test, y_test_pred)

    split_tag = str(int(split_fraction * 100))
    # file naming
    if mode == 'hard':
        train_fn = f"plots/{save_prefix}_train_hard_{split_tag}.png"
        test_fn  = f"plots/{save_prefix}_test_hard_{100-int(split_fraction*100)}.png"
    else:
        train_fn = f"plots/{save_prefix}_train_soft_{split_tag}.png"
        test_fn  = f"plots/{save_prefix}_test_soft_{100-int(split_fraction*100)}.png"

    # viz
    plot_2d_pca(X_train, y_train, w, train_fn, title=f"{save_prefix} {mode} train {split_tag
    plot_2d_pca(X_test, y_test, w, test_fn, title=f"{save_prefix} {mode} test {100-int(split_

    # save reports
    metrics_report_fn = save_confusion_and_metrics(metrics_test, save_prefix, mode, split_fra
```

```
    return {
        'weights': w,
        'iterations': iters,
        'training_TE': TE_hist[-1] if len(TE_hist) > 0 else None,
        'metrics_train': metrics_train,
        'metrics_test': metrics_test,
        'train_plot': train_fn,
        'test_plot': test_fn,
        'metrics_report': metrics_report_fn
    }
```

Summarization

```
def summarize_experiments(results_list, filename="reports/summary_table.csv"):
    df_summary = pd.DataFrame([
        {
            'Dataset': r.get('prefix', '?'),
            'Mode': r.get('mode', '?'),
            'Split': r.get('split', '?'),
            'Accuracy': r['metrics_test']['accuracy'],
            'Recall': r['metrics_test']['recall'],
            'Precision': r['metrics_test']['precision'],
            'F1': r['metrics_test']['f1'],
            'FPR': r['metrics_test']['fpr'],
            'TrainError': r['training_TE']
        } for r in results_list
    ])
    df_summary.to_csv(filename, index=False)
    print(f"Saved summary table → {filename}")
    return df_summary
```

Dataset A

Pipeline

```
# dataset a
dfA = load_data("data/groupA.txt")
res_A_hard_75 = run_experiment(dfA, split_fraction=0.75, mode='hard', save_prefix='A')
```

```
res_A_hard_25 = run_experiment(dfA, split_fraction=0.25, mode='hard', save_prefix='A')
res_A_soft_75 = run_experiment(dfA, split_fraction=0.75, mode='soft', save_prefix='A')
res_A_soft_25 = run_experiment(dfA, split_fraction=0.25, mode='soft', save_prefix='A')

# summarize to make it easier to discuss results

results = [
    {**res_A_hard_75, 'prefix':'A','mode':'hard','split':'75/25'},
    {**res_A_hard_25, 'prefix':'A','mode':'hard','split':'25/75'},
    {**res_A_soft_75, 'prefix':'A','mode':'soft','split':'75/25'},
    {**res_A_soft_25, 'prefix':'A','mode':'soft','split':'25/75'}
]

a_summary = summarize_experiments(results, "reports/a_summary_table.csv")
```

```
Saved summary table → reports/a_summary_table.csv
```

Hard Unipolar Activation

Training Split 75/25

- **Training Plot:** `A_train_hard_75.png`
- **Testing Plot:** `A_test_hard_25.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]

- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Training Split 25/75

- **Training Plot:** `A_train_hard_25.png`
- **Testing Plot:** `A_test_hard_75.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Discussion – Hard Unipolar

a. Are error rates different between 75/25 and 25/75? Why?
[Insert reasoning]

b. Effect of split ratio on performance and confusion matrices:
[Discuss observed differences in metrics and plots]

c. When would you use 75/25 vs 25/75?
[Discuss training stability vs generalization]

d. Observations and overall insights:
[Discuss total error curves, convergence rate, etc.]

Soft Unipolar Activation

Training Split 75/25

- **Training Plot:** `A_train_soft_75.png`
- **Testing Plot:** `A_test_soft_25.png`
- **Training Total Error:** [value]
- **Epochs until convergence:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]

- Precision: [ ]
- F1 Score: [ ]

Training Split 25/75

- **Training Plot:** `A_train_soft_25.png`
- **Testing Plot:** `A_test_soft_75.png`
- **Training Total Error:** [value]
- **Epochs until convergence:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Discussion – Soft Unipolar

a. Are error rates different between 75/25 and 25/75? Why?
[Insert reasoning]

b. Effect of split ratio on performance and confusion matrices:
[Discuss qualitative patterns]

c. When would you use 75/25 vs 25/75?
[Discuss based on dataset size and behavior]

d. Observations and overall insights:
[Discuss error curve smoothness, stability, convergence behavior]

Comparison: Hard vs Soft Unipolar

Metric	Hard 75/25	Soft 75/25	Hard 25/75	Soft 25/75
Accuracy	[ ]	[ ]	[ ]	[ ]
TPR	[ ]	[ ]	[ ]	[ ]
FPR	[ ]	[ ]	[ ]	[ ]
Precision	[ ]	[ ]	[ ]	[ ]
F1	[ ]	[ ]	[ ]	[ ]

Discussion:
- Which activation converged faster?
- Which achieved better generalization on smaller training data?
- Visual comparison of decision boundaries from plots.

Comparison to Project 1 Results

Insert your previously saved Project 1 confusion matrix here for side-by-side comparison:

Project	Activation	Accuracy	TPR	FPR	Precision	F1
Project 1	[Hard/Soft]	[ ]	[ ]	[ ]	[ ]	[ ]
Project 2	[Hard/Soft]	[ ]	[ ]	[ ]	[ ]	[ ]

Project	Activation	Accuracy	TPR	FPR	Precision	F1

Discussion:
- How does performance differ across datasets (A vs B)?
- Do you observe different learning behaviors?
- Which dataset or configuration yielded better generalization?
- What might explain these differences (feature scaling, class separability, sample size, etc.)?

Dataset B

Pipeline

```
# dataset a
dfB = load_data("data/groupB.txt")
res_B_hard_75 = run_experiment(dfB, split_fraction=0.75, mode='hard', save_prefix='B')
res_B_hard_25 = run_experiment(dfB, split_fraction=0.25, mode='hard', save_prefix='B')
res_B_soft_75 = run_experiment(dfB, split_fraction=0.75, mode='soft', save_prefix='B')
res_B_soft_25 = run_experiment(dfB, split_fraction=0.25, mode='soft', save_prefix='B')

# summarize to make it easier to discuss results

results = [
    {**res_B_hard_75, 'prefix':'B','mode':'hard','split':'75/25'},
    {**res_B_hard_25, 'prefix':'B','mode':'hard','split':'25/75'},
    {**res_B_soft_75, 'prefix':'B','mode':'soft','split':'75/25'},
    {**res_B_soft_25, 'prefix':'B','mode':'soft','split':'25/75'}
]

a_summary = summarize_experiments(results, "reports/b_summary_table.csv")
```

```
Saved summary table → reports/b_summary_table.csv
```

Hard Unipolar Activation

Training Split 75/25

- **Training Plot:** `B_train_hard_75.png`
- **Testing Plot:** `B_test_hard_25.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Training Split 25/75

- **Training Plot:** `B_train_hard_25.png`
- **Testing Plot:** `B_test_hard_75.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Discussion – Hard Unipolar

a. Are error rates different between 75/25 and 25/75? Why?
[Insert reasoning]

b. Effect of split ratio on performance and confusion matrices:
[Discuss observed differences in metrics and plots]

c. When would you use 75/25 vs 25/75?
[Discuss training stability vs generalization]

d. Observations and overall insights:
[Discuss total error curves, convergence rate, etc.]

Soft Unipolar Activation

Training Split 75/25

- **Training Plot:** `B_train_soft_75.png`

- **Testing Plot:** `B_test_soft_25.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Training Split 25/75

- **Training Plot:** `B_train_soft_25.png`
- **Testing Plot:** `B_test_soft_75.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Discussion – Soft Unipolar

a. Are error rates different between 75/25 and 25/75? Why?
[Insert reasoning]

b. Effect of split ratio on performance and confusion matrices:
[Discuss qualitative patterns]

c. When would you use 75/25 vs 25/75?
[Discuss based on dataset size and behavior]

d. Observations and overall insights:
[Discuss error curve smoothness, stability, convergence behavior]

Comparison: Hard vs Soft Unipolar

Metric	Hard 75/25	Soft 75/25	Hard 25/75	Soft 25/75
Accuracy	[ ]	[ ]	[ ]	[ ]
TPR	[ ]	[ ]	[ ]	[ ]
FPR	[ ]	[ ]	[ ]	[ ]
Precision	[ ]	[ ]	[ ]	[ ]
F1	[ ]	[ ]	[ ]	[ ]

Discussion:
- Which activation converged faster?
- Which achieved better generalization on smaller training data?
- Visual comparison of decision boundaries from plots.

Comparison to Project 1 Results

Insert your previously saved Project 1 confusion matrix here for side-by-side comparison:

Project	Activation	Accuracy	TPR	FPR	Precision	F1
Project 1	[Hard/Soft]	[ ]	[ ]	[ ]	[ ]	[ ]
Project 2	[Hard/Soft]	[ ]	[ ]	[ ]	[ ]	[ ]

Discussion:
- How does performance differ across datasets (A vs B)?
- Do you observe different learning behaviors?
- Which dataset or configuration yielded better generalization?
- What might explain these differences (feature scaling, class separability, sample size, etc.)?

Dataset C

Pipeline

```
# dataset c
dfC = load_data("data/groupC.txt")
res_C_hard_75 = run_experiment(dfC, split_fraction=0.75, mode='hard', save_prefix='C')
res_C_hard_25 = run_experiment(dfC, split_fraction=0.25, mode='hard', save_prefix='C')
res_C_soft_75 = run_experiment(dfC, split_fraction=0.75, mode='soft', save_prefix='C')
res_C_soft_25 = run_experiment(dfC, split_fraction=0.25, mode='soft', save_prefix='C')

# summarize to make it easier to discuss results

results = [
    {**res_C_hard_75, 'prefix':'C','mode':'hard','split':'75/25'},
    {**res_C_hard_25, 'prefix':'C','mode':'hard','split':'25/75'},
    {**res_C_soft_75, 'prefix':'C','mode':'soft','split':'75/25'},
    {**res_C_soft_25, 'prefix':'C','mode':'soft','split':'25/75'}
]

c_summary = summarize_experiments(results, "reports/c_summary_table.csv")
```

```
Saved summary table → reports/c_summary_table.csv
```

Hard Unipolar Activation

Training Split 75/25

- **Training Plot:** `C_train_hard_75.png`
- **Testing Plot:** `C_test_hard_25.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Training Split 25/75

- **Training Plot:** `C_train_hard_25.png`
- **Testing Plot:** `C_test_hard_75.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Discussion – Hard Unipolar

a. Are error rates different between 75/25 and 25/75? Why?
[Insert reasoning]

b. Effect of split ratio on performance and confusion matrices:
[Discuss observed differences in metrics and plots]

c. When would you use 75/25 vs 25/75?
[Discuss training stability vs generalization]

d. Observations and overall insights:
[Discuss total error curves, convergence rate, etc.]

Soft Unipolar Activation

Training Split 75/25

- **Training Plot:** `C_train_soft_75.png`

- **Testing Plot:** `C_test_soft_25.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Training Split 25/75

- **Training Plot:** `C_train_soft_25.png`
- **Testing Plot:** `C_test_soft_75.png`
- **Training Total Error:** [value]

Confusion Matrix (Testing Data)

	Predicted Positive	Predicted Negative
Actual Positive	TP = [ ]	FN = [ ]
Actual Negative	FP = [ ]	TN = [ ]

Performance Metrics

- Accuracy: [ ]
- True Positive Rate (Recall): [ ]
- False Positive Rate: [ ]
- Precision: [ ]
- F1 Score: [ ]

Discussion – Soft Unipolar

a. Are error rates different between 75/25 and 25/75? Why?
[Insert reasoning]

b. Effect of split ratio on performance and confusion matrices:
[Discuss qualitative patterns]

c. When would you use 75/25 vs 25/75?
[Discuss based on dataset size and behavior]

d. Observations and overall insights:
[Discuss error curve smoothness, stability, convergence behavior]

Comparison: Hard vs Soft Unipolar

Metric	Hard 75/25	Soft 75/25	Hard 25/75	Soft 25/75
Accuracy	[ ]	[ ]	[ ]	[ ]
TPR	[ ]	[ ]	[ ]	[ ]
FPR	[ ]	[ ]	[ ]	[ ]
Precision	[ ]	[ ]	[ ]	[ ]
F1	[ ]	[ ]	[ ]	[ ]

Discussion:
- Which activation converged faster?
- Which achieved better generalization on smaller training data?
- Visual comparison of decision boundaries from plots.

Comparison to Project 1 Results

Insert your previously saved Project 1 confusion matrix here for side-by-side comparison:

Project	Activation	Accuracy	TPR	FPR	Precision	F1
Project 1	[Hard/Soft]	[ ]	[ ]	[ ]	[ ]	[ ]
Project 2	[Hard/Soft]	[ ]	[ ]	[ ]	[ ]	[ ]

Discussion:
- How does performance differ across datasets (A vs B)?
- Do you observe different learning behaviors?
- Which dataset or configuration yielded better generalization?
- What might explain these differences (feature scaling, class separability, sample size, etc.)?

Part 2 – Soft vs Hard Comparison

- Placeholder text.

Extra Credit

- Placeholder text.

Conclusion

- Key takeaways from Dataset A, B, and C experiments.
- Overall differences between hard vs soft unipolar activation.
- When to prefer larger training split vs smaller one.
- Might not need this section can include it if we want to.