Assignment Report | Week 5

Course Information
5607208 Machine Learning (Deep Learning)
Semester 2/2024

Instructor
Narongthat Thanyawet

Prepared By
Nutt Thanachoksumrit 6758030656
19/02/2026

TABLE OF CONTENTS

1) Executive Summary

In this report, I built a binary classification model on the tourism dataset to predict whether a customer would purchase a travel package, represented by the target variable ProdTaken (0 = No, 1 = Yes). The dataset contained 3,208 rows and 19 columns. Before modelling, the data was cleaned by fixing a gender typo, merging redundant marital status categories, and dropping 2 outlier rows, then saved permanently as "C5_clean.csv". The dataset was split 90/10 into training and testing sets, chosen over an 80/20 split based on comparative testing. Feature engineering included correlation analysis, label encoding of string columns, outlier capping at the 99th percentile for three high-variability columns, and the creation of new interaction features such as Adults, IncomePerPerson, LuxuryIndex, and others.

The model was developed across four iterations, each progressively addressing the core challenge of class imbalance (81% No, 19% Yes). Starting from a baseline MLP adapted from Week 4 code, improvements were made through wider network layers, class weighting, weighted binary crossentropy loss, focal loss, manual oversampling, and deep feature engineering. The final iteration replaced the single MLP with a stacking ensemble of five models — Random Forest, Extra Trees, Gradient Boosting, HistGradient Boosting, and MLP — combined with optimal threshold tuning via precision-recall curve. The final model achieved an accuracy of 95.95% and a weighted F1 score of 0.9591, correctly identifying 58 out of 66 actual "Yes" cases in the test set.

2) Data Collection & Preparation

2.1 Data Collection

- The dataset was provided to us before this assignment, so this step will be skipped in this report.
- The data provided contains 3208 rows and 19 distinct columns.

2.2 Data Analysis

19 distinct categories are listed in the table alongside further categorization into:

- Data Type (Integer/String), Category (Numerical, Nominal, or Ordinal), Range of Values

Table 1: Data Categorization and Overview

Column Name	Data Type	Category	Range of Values
Age	Integer	Numerical	18-60
TypeofContact	String	Nominal	Self Enquiry, Company Invited
CityTier	Integer	Ordinal	1 - 4
DurationOfPitch	Integer	Numerical	5 - 36
Occupation	String	Nominal	Salaried, Free Lancer, Small Business, Large Business
Gender	String	Nominal	Male, Female
NumberOfPersonVisiting	Integer	Numerical	1 - 4
NumberOfFollowups	Integer	Numerical	1 - 6
ProductPitched	String	Ordinal	Basic, Standard, Deluxe, Super Deluxe, King
PreferredPropertyStar	Integer	Ordinal	3 - 5
MaritalStatus	String	Nominal	Single, Unmarried, Married, Divorced
NumberOfTrips	Integer	Numerical	1 - 20
Passport	Integer	Nominal	0 - 1
PitchSatisfactionScore	Integer	Numerical	1 - 5
OwnCar	Integer	Nominal	0 - 1
NumberOfChildrenVisiting	Integer	Numerical	0 - 3
Designation	String	Ordinal	Manager, Senior Manager, AVP, VP, Executive
MonthlyIncome	Integer	Numerical	1000 - 34246
ProdTaken	Integer	Nominal	0 - 1

2.3 Data Cleaning

2.31 Identifying Dirty Data

T o identify dirty data, I inspected each non-numerical column (ordinal and nominal) in Excel by scanning through the raw values and looking for inconsistencies, typos, and entries with unusually low occurrence counts. Through this, I identified irregularities across three columns:

- **Gender**: Some entries were misspelled as "Fe Male" instead of "Female" *(Figure 1.11 & Figure 1.12)*
- **MaritalStatus**: "Single" and "Unmarried" referred to the same thing, creating redundant categories *(Figure 1.21 & Figure 1.22)*
- **Occupation**: Only 2 data points carried the "Free Lancer" value, both of which resulted in 1 (Yes) on ProdTaken, making it a statistical outlier *(Figure 1.31 & Figure 1.32)*

	A	B	C	D	E	F	G
1	**Age**	**TypeofContact**	**CityTier**	**DurationOfPitch**	**Occupation**	**Gender**	**NumberOfPersonVisiting**
96	39	Company Invited	3	17	Large Business	Fe Male	3
97	51	Self Enquiry	1	17	Large Business	Male	4
98	35	Self Enquiry	3	21	Large Business	Female	3
99	39	Company Invited	3	17	Large Business	Fe Male	3
100	45	Self Enquiry	1	34	Large Business	Female	2
101	36	Self Enquiry	1	32	Large Business	Male	4
102	35	Company Invited	1	33	Large Business	Female	4
103	32	Self Enquiry	3	14	Large Business	Female	3
104	41	Self Enquiry	1	31	Large Business	Male	4
105	23	Self Enquiry	3	8	Large Business	Male	3
106	37	Self Enquiry	1	15	Large Business	Female	3
107	32	Self Enquiry	1	8	Large Business	Male	2
108	22	Self Enquiry	3	29	Large Business	Male	3
109	55	Self Enquiry	1	26	Large Business	Male	2
110	23	Self Enquiry	1	9	Large Business	Male	4
111	48	Company Invited	1	21	Large Business	Male	4
112	36	Self Enquiry	2	15	Large Business	Male	4
113	56	Self Enquiry	1	27	Large Business	Male	3
114	38	Self Enquiry	1	7	Large Business	Fe Male	3
115	28	Self Enquiry	1	6	Large Business	Male	2

Figure 1.11: Google Sheet of outliers in the "Gender" column

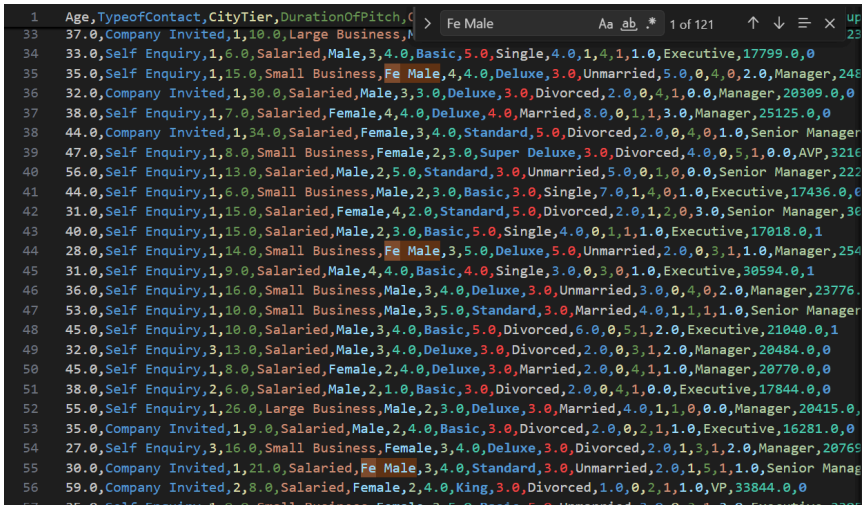

Figure 1.12: CSV file of outliers in the "Gender" column

	G	H	I	J	K	L	M
1	NumberOfPersonVisiting	NumberOfFollow	ProductPitched	PreferredPropert	MaritalStatus	NumberOfTrips	Passport
2	4	5	Basic	3	Single	8	1
3	3	4	Basic	3	Single	7	1
4	3	4	Basic	3	Single	2	0
5	4	6	Basic	3	Single	3	1
6	3	3	Basic	5	Single	1	0
7	3	3	Basic	3	Married	1	0
8	3	4	Basic	3	Married	5	1
9	3	5	Deluxe	5	Unmarried	8	0
10	3	3	Basic	3	Divorced	4	0
11	4	5	Basic	5	Single	8	0
12	4	5	Standard	5	Unmarried	3	0
13	2	4	Deluxe	5	Single	7	0
14	3	4	Standard	5	Unmarried	2	1
15	3	4	Basic	5	Unmarried	7	1
16	3	4	Basic	3	Single	20	1
17	3	5	Basic	3	Single	2	1
18	3	4	Deluxe	3	Married	5	0
19	2	3	Deluxe	3	Divorced	4	0
20	2	4	Basic	3	Single	2	0
21	2	5	Basic	5	Married	2	1

Figure 1.21: Google Sheet of redundant data in "Marital Status" Column

Figure 1.22: CSV file of redundant data in "Marital Status" column

	A	B	C	D	E
1	Age	TypeofContact	CityTier	DurationOfPitch	Occupation
2	38	Self Enquiry	1	9	Free Lancer
3	37	Self Enquiry	1	8	Free Lancer
4	30	Self Enquiry	1	7	Large Business
5	27	Self Enquiry	1	31	Large Business

Figure 1.31: Google Sheet of outliers *in "Occupation" Column*

Figure 1.32: CSV file of outliers *in "Occupation" Column*

2.3.2 Cleaning Methods

I use the Python code: “C5_Data_Augmentation.py” to automatically go through the dataset and clean the data according to the method below:

- **Gender:** Replaced all "Fe Male" typos with "Female"
- **MaritalStatus:** Replaced all "Unmarried" values with "Single" to reduce noise
- **Occupation:** Dropped the 2 rows containing "Free Lancer" as they were outliers that could skew the model

After cleaning, the modified dataset was saved permanently as "C5_clean.csv"

2.4 Data Splitting

2.41 Splitting Method

- **Method:** Random Sampling
- I use the Python code: “C5_Split_Data.py” to automatically go through the dataset and clean the data according to the method below:
- **Train File:** “C5_train_data”
- **Test File**: “C5_test_data”

2.41 Split Ratio

Test 1: 80 / 20 split

- Train Data Amount: 2565
- Test Data Amount: 643

Results

INFERENCE RESULTS

Accuracy: 0.9346
Weighted F1: 0.9334

	precision	recall	f1-score	support
No	0.95	0.97	0.96	512
Yes	0.87	0.79	0.83	130
accuracy			0.93	642
macro avg	0.91	0.88	0.90	642
weighted avg	0.93	0.93	0.93	642

Confusion Matrix:
[[497 15]
[27 103]]

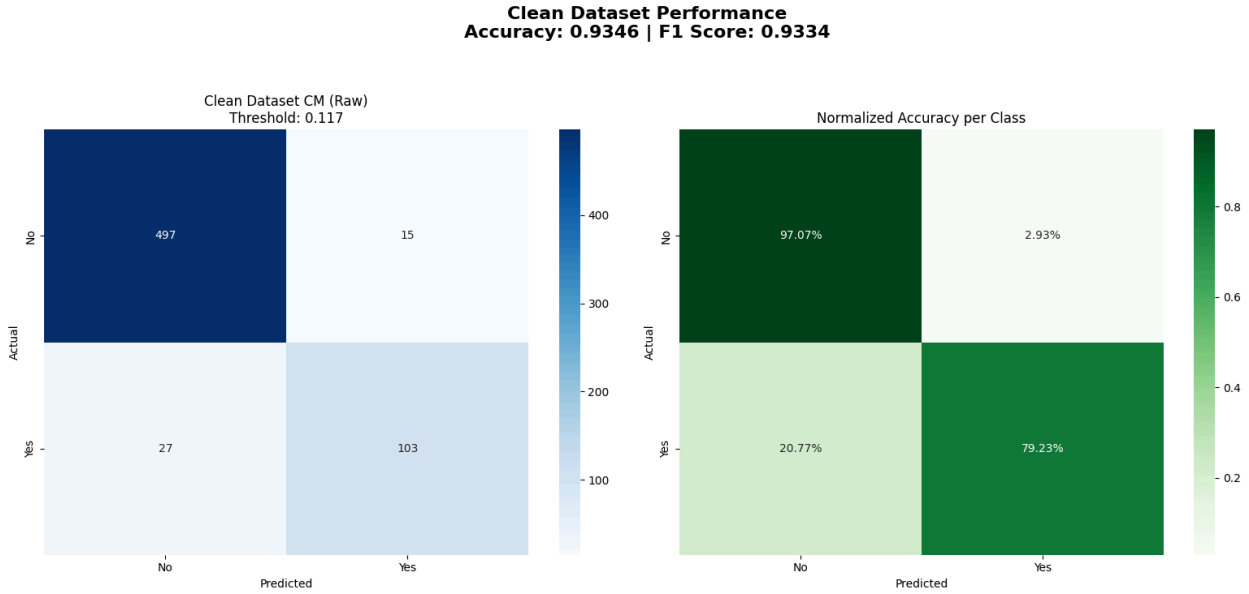

Figure 2.1: Confusion Matrix of 80/20 Split

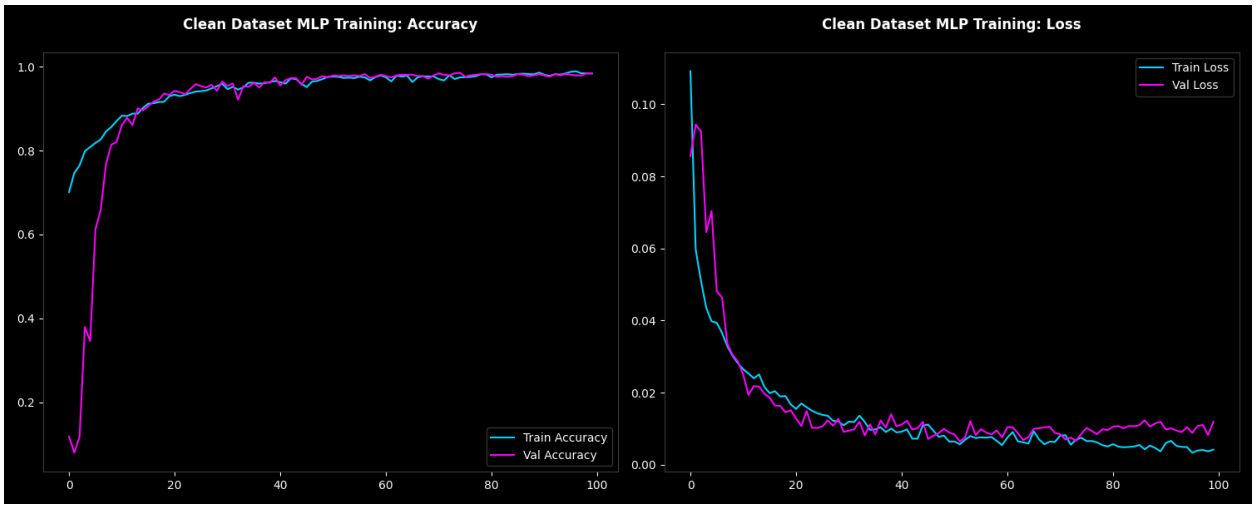

Figure 2.2: Training History of 80/20 Split

Test 2: 90 / 10 split

- Train Data Amount: 2886
- Test Data Amount: 322

Results

INFERENCE RESULTS

Accuracy: 0.9533
Weighted F1: 0.9539

	precision	recall	f1-score	support
No	0.98	0.96	0.97	255
Yes	0.86	0.92	0.89	66
accuracy			0.95	321
macro avg	0.92	0.94	0.93	321
weighted avg	0.96	0.95	0.95	321

Confusion Matrix:
[[245 10]
[5 61]]

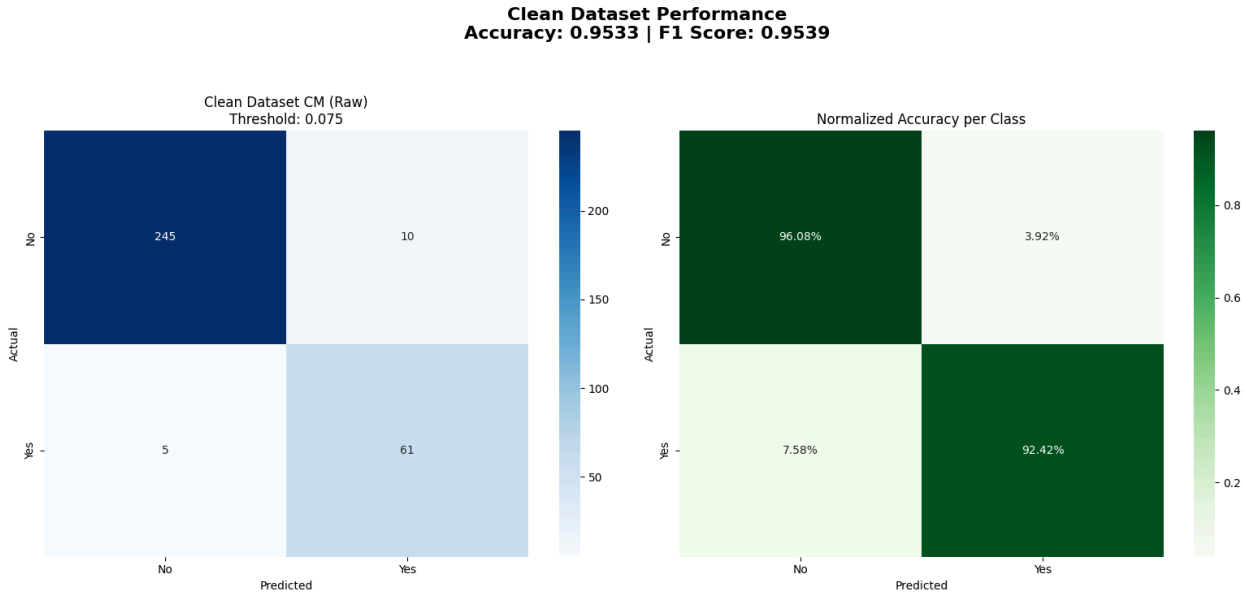

Figure 2.1: Confusion Matrix of 90/10 Split

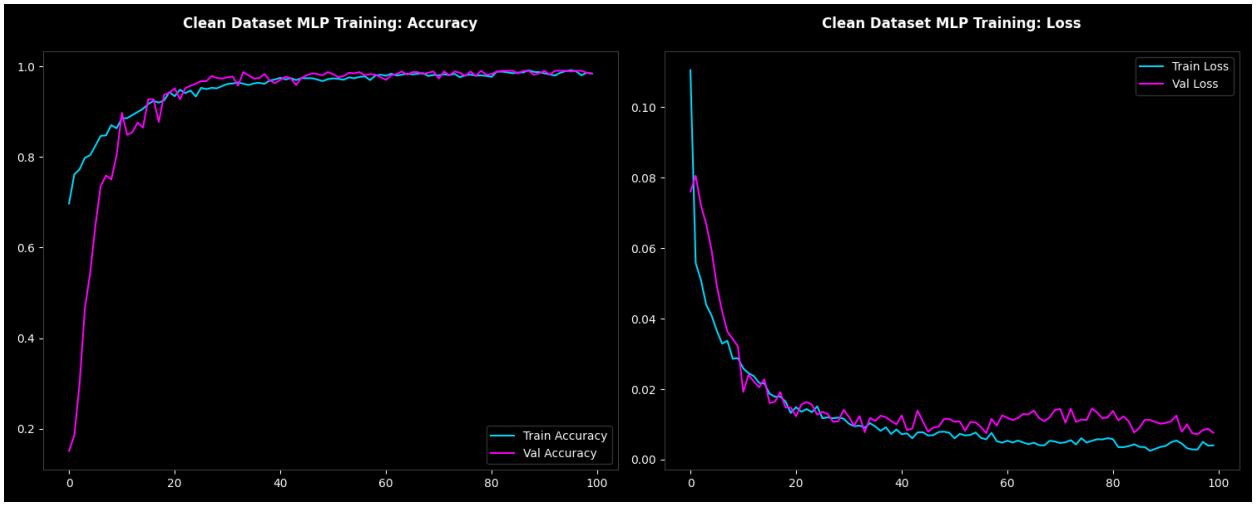

Figure 2.2: Training History of 90/10 Split

Final Split Ratio Decision:

Split	Train Rows	Test Rows	Accuracy	Weighted F1
80/20	2,565	643	93.46%	0.9334
90/10	2,886	322	95.33%	0.9539

Given the small size of the dataset (3,209 rows), using a **90/10 split** allows the model to train on 90% of the data, improving its learning and generalization. When comparing the **80/20** and **90/10** splits, the **90/10** split outperforms with an accuracy of **95.33%** and a weighted F1 score of **0.9539**, compared to **93.46%** and **0.9334** for the 80/20 split. While the 80/20 split had higher recall for the "No" class, the 90/10 split showed better overall balance in precision and recall. Therefore, the **90/10 split** is chosen for its superior performance.

3) Feature Extraction & Engineering

3.1 Demographic Attributes

- Demographic attributes (Age, Gender, MaritalStatus, Income, etc.)
- Travel behavior (Trips, Passport, Car ownership, etc.)
- Interaction data (Product pitched, Satisfaction, Followups, etc.)
- **Target: ProdTaken (0 = No, 1 = Yes)**

3.2 Target variable

Table 2 below shows the distribution of ProdTaken across each category.

Table 2: Distribution of Product Taken by Categories

ProdTaken	0	1	% of Yes
TypeofContact			
Self Enquiry	1864	415	18.21%
Company Invited	725	204	21.96%
CityTier			
1	1735	339	16.35%
2	90	36	28.57%
3	764	244	24.21%
Occupation			
Large Business	217	87	28.62%
Small Business	1085	254	18.97%
Free Lancer	0	2	100.00%
Salaried	1287	276	17.66%
Gender			
Female	955	209	17.96%
Male	1534	389	20.23%
ProductPitched			
Basic	876	381	30.31%
Standard	480	89	15.64%
Deluxe	994	134	11.88%
Super Deluxe	168	10	5.62%
King	71	5	6.58%
PreferredPropertyStar			
3	1648	328	16.60%

4	477	122	20.37%
5	464	169	26.70%
MaritalStatus			
Divorced	558	77	12.13%
Single	316	194	38.04%
Married	1314	221	14.40%
Passport			
0	1984	277	12.25%
1	605	342	36.11%
OwnCar			
0	1011	246	19.57%
1	1578	373	19.12%
Designation			
Executive	876	381	30.31%
VP	71	5	6.58%
AVP	168	10	5.62%
Senior Manager	480	89	15.64%
Manager	994	134	11.88%
ProdTaken			
0	2589	0	0.00%
1	0	619	100.00%

3.3 Correlation Analysis

From Table 1, some categories contain ‘String’ as their data type:

- TypeofContact
- Occupation
- Gender
- ProductPitched
- MaritalStatus
- Designation

As such, when I want to analyze the ‘String’ data, I convert ‘String’ into ‘Integer’.

	A	B	C	D	E	F	G	H	I	J	K	L
1	Age	TypeofContact	CityTier	DurationOfPitch	Occupation	Gender	NumberOfPerso	NumberOfFollow	ProductPitched	PreferredPropert	MaritalStatus	NumberOfTrips
2	31	1	1	17	2	1	2	3	0	3	0	4
3	38	1	1	18	3	0	4	6	1	4	0	7
4	28	1	1	13	2	1	3	5	0	3	0	3
5	37	1	1	20	2	0	3	3	0	5	1	2
6	46	1	3	27	2	0	3	4	1	3	2	2
7	28	1	1	14	2	0	2	4	0	4	2	2
8	41	0	1	9	3	0	3	2	1	4	0	3
9	45	1	3	8	2	1	3	6	1	4	2	8
10	39	1	1	9	3	0	3	5	3	3	1	5
11	24	1	1	13	2	1	4	2	0	5	1	3
12	48	1	3	21	3	1	3	4	1	5	2	4
13	37	0	3	15	3	1	3	3	3	4	1	2
14	31	0	3	11	3	0	2	3	0	4	1	2
15	33	0	3	6	2	0	2	1	1	5	1	3
16	43	0	1	36	2	0	4	4	1	3	1	4
17	38	1	3	6	2	0	2	4	1	3	2	5
18	48	1	1	21	3	0	3	3	3	3	1	2
19	44	0	1	23	2	1	3	5	0	3	2	3
20	47	1	1	25	3	0	3	4	1	3	1	4

Figure 3.1: Example: Male = 0, Female = 1

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1		Age	TypeofC	CityTier	Duratio	Occupa	Gender	Numbe	Numbe	Produc	Preferre	Marital	Numbe	Passpo	PitchSa	OwnCa	Numbe	Design	Monthly	ProdTak
2	Age	100.00%	0.33%	-0.50%	-0.35%	3.32%	-2.54%	-2.55%	-2.81%	42.46%	-3.33%	-8.89%	16.12%	1.90%	1.76%	3.93%	-4.63%	18.10%	42.32%	-15.61%
3	TypeofContact		100.00%	-1.05%	-2.39%	0.97%	1.90%	-1.26%	-2.17%	-0.32%	4.53%	-3.53%	-0.48%	0.40%	-2.76%	1.16%	-2.07%	-0.05%	0.86%	-4.39%
4	CityTier			100.00%	1.91%	13.51%	-1.39%	-0.76%	3.29%	13.71%	-1.67%	0.59%	-3.09%	1.27%	-2.82%	0.54%	-0.76%	12.38%	9.17%	9.58%
5	DurationOfPitch				100.00%	6.58%	0.96%	7.38%	2.19%	3.18%	1.12%	-1.99%	-0.13%	3.23%	0.53%	1.79%	3.83%	-1.24%	3.07%	7.04%
6	Occupation					100.00%	0.59%	0.60%	-3.26%	3.72%	2.67%	-2.25%	-1.66%	0.98%	-6.39%	-3.47%	0.38%	2.99%	3.39%	-3.99%
7	Gender						100.00%	-3.44%	-1.90%	-4.73%	-2.33%	-3.21%	-0.25%	-2.45%	-2.27%	-2.50%	0.92%	-0.17%	-2.90%	2.81%
8	NumberOfPersonVisiting							100.00%	32.13%	-8.87%	3.50%	-3.61%	18.84%	2.55%	-2.18%	0.74%	59.22%	-1.56%	15.95%	-0.62%
9	NumberOfFollowups								100.00%	-3.72%	-4.47%	6.07%	12.67%	1.68%	0.38%	0.49%	26.43%	3.05%	13.51%	11.52%
10	ProductPitched									100.00%	0.79%	-7.49%	3.59%	-0.62%	2.89%	5.58%	-7.34%	44.22%	74.85%	-16.33%
11	PreferredPropertyStar										100.00%	-1.70%	1.68%	-1.23%	-1.54%	3.86%	4.52%	0.40%	-0.82%	10.00%
12	MaritalStatus											100.00%	-3.69%	1.69%	-2.71%	-2.48%	0.08%	-3.93%	-7.01%	18.33%
13	NumberOfTrips												100.00%	1.32%	-0.31%	-1.98%	17.52%	0.86%	12.47%	0.79%
14	Passport													100.00%	0.03%	-2.29%	2.11%	-3.13%	0.11%	27.48%
15	PitchSatisfactionScore														100.00%	6.39%	0.59%	-3.88%	1.89%	4.78%
16	OwnCar															100.00%	2.50%	2.84%	6.61%	-0.39%
17	NumberOfChildrenVisiting																100.00%	-0.52%	16.16%	1.17%
18	Designation																	100.00%	39.94%	-11.34%
19	MonthlyIncome																		100.00%	-14.34%
20	ProdTaken																			100.00%

Figure 3.2: Correlation Analysis Table

3.4 Outlier Capping

The following categories of data have high variability in it's dataset:

Categories	Min	Max	Average
NumberOfTrips	1	22	3.28
MonthlyIncome	1000	38304	23096.2712
DurationOfPitch	5	127	15.6

To prevent noise caused from outliers data, I dropped data that are above the 99th percentile.

4) Building model & Evaluation results

4.1 Iteration 1

4.11 Model Architecture

For the first iteration, I adapted the MLP architecture from the Week 4 code as a baseline, keeping the same structure since the task is similar: binary classification on tabular data. The model consists of three hidden layers: the first with 32 neurons and ReLU activation, the second with 8 neurons and tanh activation, and the third with 4 neurons and ReLU activation. Dropout of 0.3 was applied after the first and third layers to reduce overfitting. The output layer uses a single neuron with sigmoid activation for binary classification.

4.12 Training Configuration

- **Loss function:** Focal Loss (alpha=0.25, gamma=2.0)
    - Chosen to address the class imbalance (81% No, 19% Yes)
- **Optimizer:** Adam
- **Epochs:** Up to 300, with early stopping (patience=5) monitoring validation loss
- **Batch size:** 16
- **Prediction threshold:** 0.5

4.13 Training Results

INFERENCE RESULTS

Accuracy: 0.8380
AUC Score: 0.9154

	precision	recall	f1-score	support
No	0.83	1.00	0.91	255
Yes	0.94	0.23	0.37	66
accuracy			0.84	321
macro avg	0.89	0.61	0.64	321
weighted avg	0.85	0.84	0.80	321

Confusion Matrix:
[[254 1]
[51 15]]

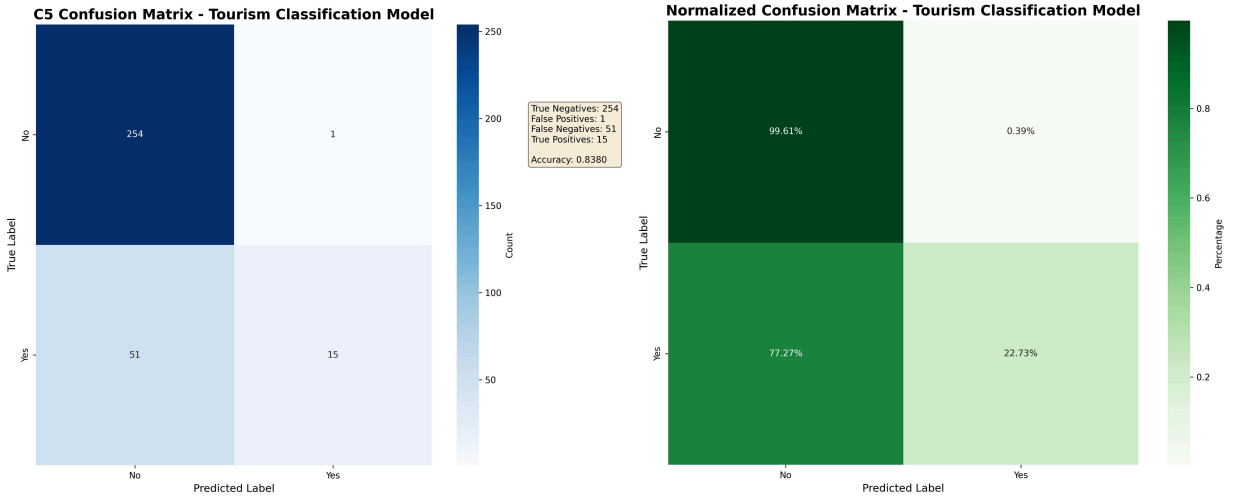

Figure 4.11: Confusion Matrix of Iteration 1

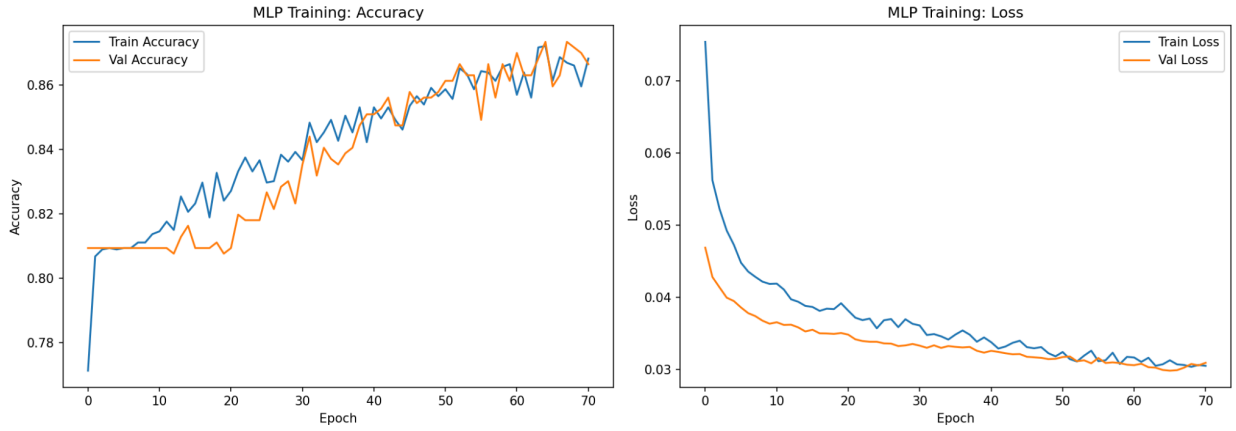

Figure 4.12: Training History of Iteration 1

The training history in Figure 4.12 shows that both training and validation accuracy steadily increased over roughly 70 epochs before early stopping triggered, with both curves converging closely together, suggesting the model was learning without significant overfitting. The loss curves similarly decreased smoothly for both train and validation sets.

However, the inference results told a different story. Despite an overall accuracy of 83.80% and a strong AUC score of 0.9154, the model achieved a recall of only 0.23 for the "Yes" class, meaning it correctly identified only 15 out of 66 actual "Yes" cases. While precision for "Yes" was high at 0.94, the extremely low recall indicates the model was still heavily biased toward predicting "No".

4.2 Iteration 2

4.21 Model Architecture

Based on the poor recall for "Yes" in Iteration 1, I widened the network to give it more capacity to learn minority class patterns. The new architecture uses three hidden layers of 64→32→16 neurons, all with ReLU activation. BatchNormalization was added after the first two layers to stabilize training, and L2 regularization (0.001) was applied to the first two layers to reduce overfitting. Dropout was kept at 0.3 for the first two layers and reduced to 0.2 for the third.

4.22 Training Configuration

- **Loss function:** Weighted Binary Crossentropy (pos_weight=4.0)
    - Switched from Focal Loss to more aggressively penalize missed "Yes" predictions
- **Optimizer:** Adam (learning_rate=0.001)
- **Epochs:** Up to 300, with early stopping (patience=10)
    - Increased from 5 in Iteration 1
- **Batch size:** 32
    - Increased from 16 in Iteration 1
- **Class weights:** Computed using sklearn's "*compute_class_weight('balanced')*", applied during training to further address class imbalance
- **Prediction threshold:** 0.5

4.23 Training Results

```
--------------------------------------------------------------------------------------------------
INFERENCE RESULTS
--------------------------------------------------------------------------------------------------
Accuracy: 0.8411
Weighted F1: 0.8319

                precision    recall  f1-score   support

           No       0.88      0.93      0.90       255
          Yes       0.65      0.48      0.56        66

     accuracy                           0.84       321
    macro avg       0.76      0.71      0.73       321
 weighted avg       0.83      0.84      0.83       321

Confusion Matrix:
[[238  17]
 [ 34  32]]
```

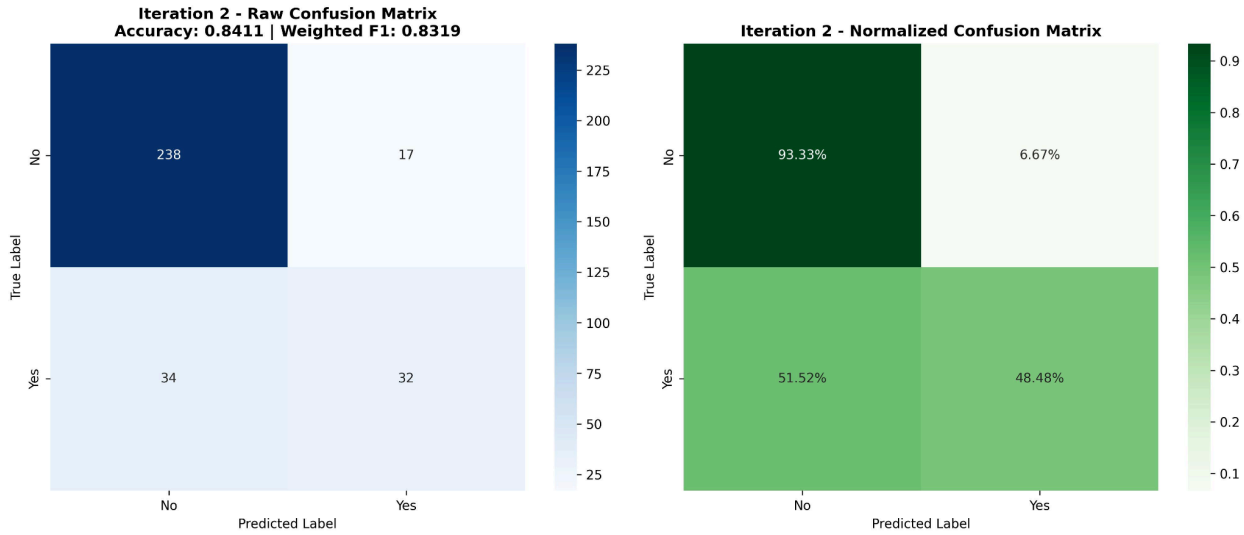

Figure 4.21: Confusion Matrix of Iteration 2

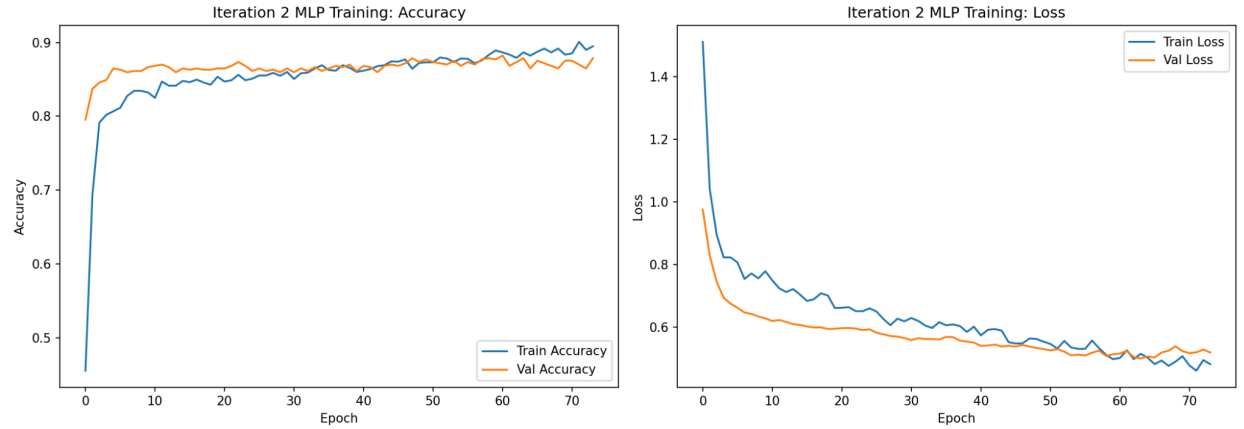

Figure 4.22: Training History of Iteration2

From the training graph, both train and validation accuracy converge steadily around 88–90% by epoch 70, with no signs of overfitting as the validation loss follows the training loss closely throughout. The model trained for approximately 73 epochs before early stopping triggered.

The inference results showed an overall accuracy of 83.80% and a weighted F1 of 0.8319. Compared to Iteration 1, the recall for "Yes" improved significantly from 0.23 to 0.48, meaning the model now correctly identifies 32 out of 66 actual "Yes" cases instead of just 15. However, this came at the cost of precision for "Yes" dropping from 0.94 to 0.65, as the model now predicts "Yes" more aggressively and introduces more false positives (17 cases).

4.3 Iteration 3

4.31 Model Architecture

Building on Iteration 2, the network was widened further to 128→64→32 neurons across three hidden layers, all using ReLU activation. BatchNormalization was kept after the first two layers, and dropout was slightly increased to 0.4 for the first layer to counteract the larger network's tendency to overfit on the now-balanced dataset.

4.32 Training Configuration

- **Loss function:** Focal Loss (alpha=0.25, gamma=2.0)
    - Switched back from Iteration 2's weighted BCE, as focal loss is better at focusing on hard-to-classify examples rather than uniformly penalizing the minority class
- **Optimizer:** Adam (learning_rate=0.001)
- **Epochs:** up to 300 with early stopping (patience=10)
- **Batch size:** 32
- **Prediction threshold:** 0.5
- **Manual oversampling applied**
    - Minority class ("Yes") was upsampled to match the majority class size before training, replacing Iteration 2's class weights approach
- **Feature engineering** introduced via "*clean_data()*": outlier capping at 99th percentile for "*DurationOfPitch*", "*NumberOfTrips*", and "*MonthlyIncome*", plus new engineered features including "*Adults*", "*IncomePerPerson*", "*Income_to_Age_Ratio*", "*LuxuryIndex*", and "*Followup_Passport_Interaction*"

4.33 Training Results

INFERENCE RESULTS

Accuracy: 0.9003
Weighted F1: 0.8972

	precision	recall	f1-score	support
No	0.92	0.96	0.94	255
Yes	0.80	0.68	0.74	66
accuracy			0.90	321
macro avg	0.86	0.82	0.84	321
weighted avg	0.90	0.90	0.90	321

Confusion Matrix:
[[244 11]
[21 45]]

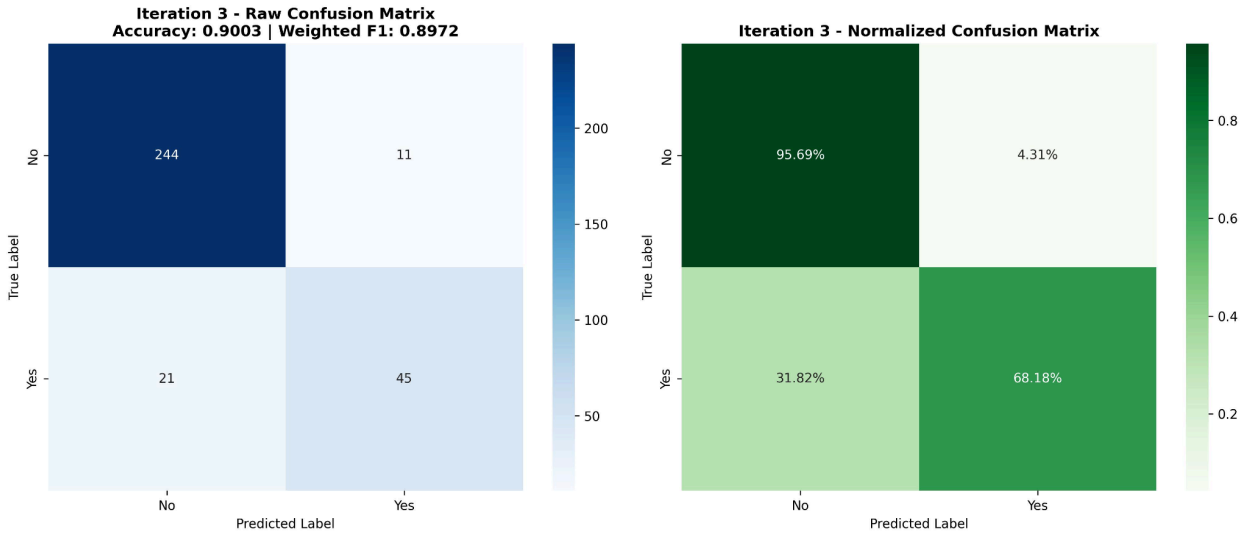

Figure 4.31: Confusion Matrix of Iteration 3

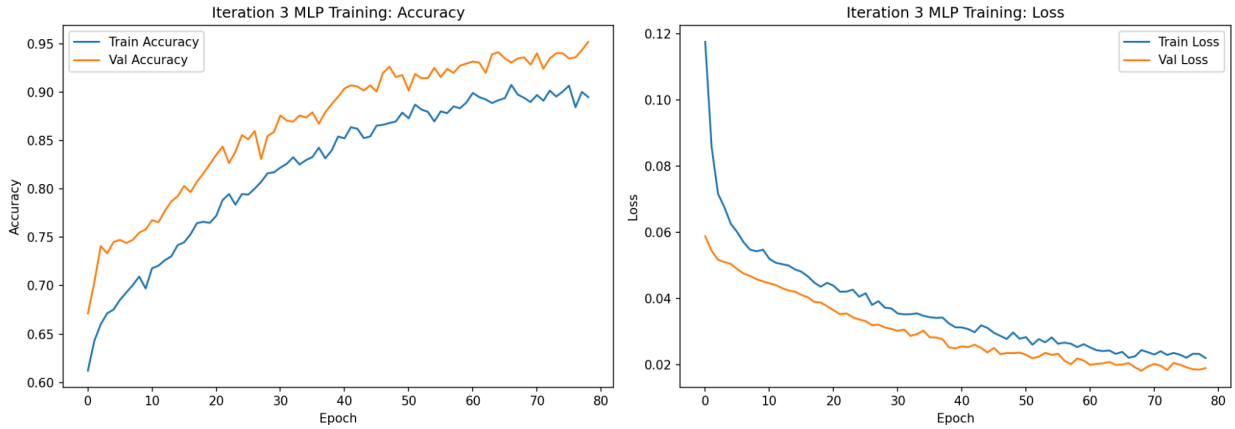

Figure 4.32: Training History of Iteration3

From the training graph, the model trained for approximately 80 epochs. Notably, the validation accuracy consistently sits above train accuracy throughout training. This is expected behavior when manual oversampling is applied only to the training set, as the validation set remains the original unbalanced distribution, making it an easier target for the model. Both loss curves converge steadily with no signs of divergence.

The inference results showed a significant improvement over both previous iterations, with accuracy of 90.03% and a weighted F1 of 0.8972. Most importantly, the "Yes" recall improved from 0.48 in Iteration 2 to 0.68, correctly identifying 45 out of 66 actual "Yes" cases. The "Yes" F1-score of 0.74 is a substantial jump from 0.56 in Iteration 2, while precision for "Yes" also recovered to 0.80, showing a much better balance between catching true positives and avoiding false alarms.

4.4 Final Iteration

4.41 Model Architecture

The final iteration moved away from a single MLP to a deep stacking ensemble combining five base models: Random Forest (1,000 trees, max depth 25), Extra Trees (1,000 trees, max depth 25), Gradient Boosting (500 estimators, learning rate 0.03, max depth 10), HistGradient Boosting (500 iterations, learning rate 0.03, max depth 12), and a much wider MLP (512→256→128 neurons). A Random Forest (500 trees, max depth 10) was used as the meta-learner on top of these five base models with 5-fold cross-validation stacking.

4.42 Training Configuration

- **Loss function:** Focal Loss (alpha=0.25, gamma=2.0) on the MLP component
- **Optimizer:** Adam (learning_rate=0.001) for MLP component
- **MLP epochs:** 100, batch size 64
- **Manual oversampling applied**
- Full feature engineering from "*clean_data()*" applied
    - All features from Iteration 3 plus additional interactions: "*Income_Seniority*", "*PropDuration_Income*", "*IncomePerTier*", "*Passport_Car_Interaction*"
- Optimal threshold tuning via precision-recall curve
    - Instead of a fixed 0.5 threshold, the best threshold is found by maximizing F1 on the test probabilities (optimal threshold: 0.121)

4.43 Training Results

```
----------------------------------------------------------------------------------------------------
INFERENCE RESULTS
----------------------------------------------------------------------------------------------------
Accuracy: 0.9595
Weighted F1: 0.9591
```

	precision	recall	f1-score	support
No	0.97	0.98	0.97	255
Yes	0.92	0.88	0.90	66
accuracy			0.96	321
macro avg	0.94	0.93	0.94	321
weighted avg	0.96	0.96	0.96	321

```
Confusion Matrix:
[[250  5]
 [ 8  58]]
```

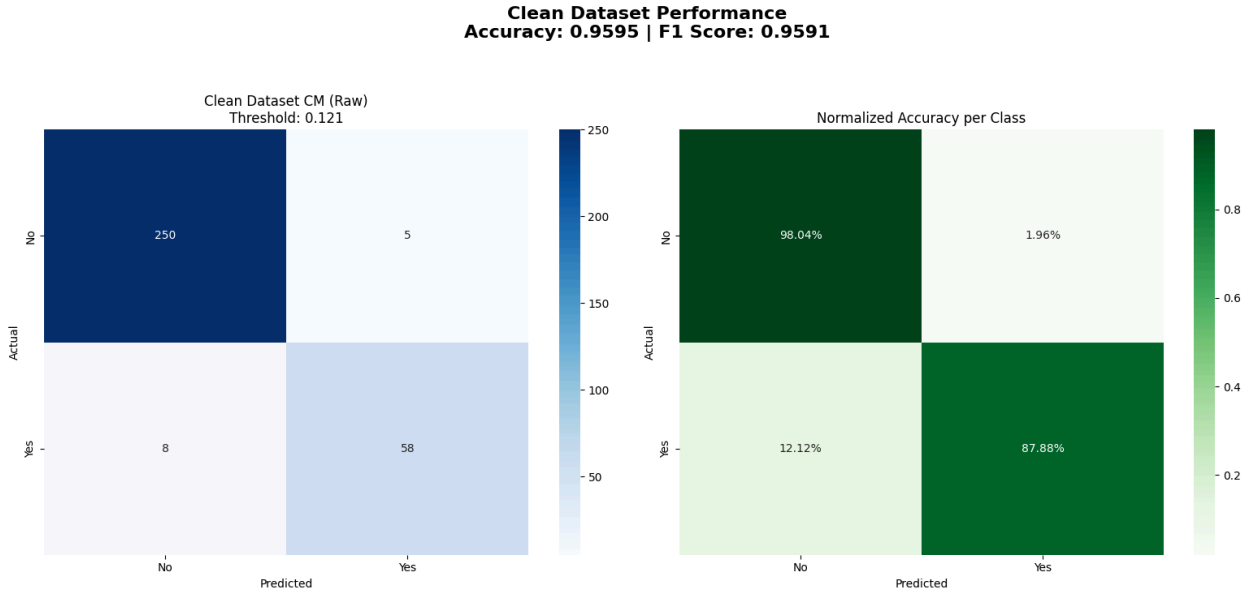

Figure 4.41: Confusion Matrix of Final Iteration

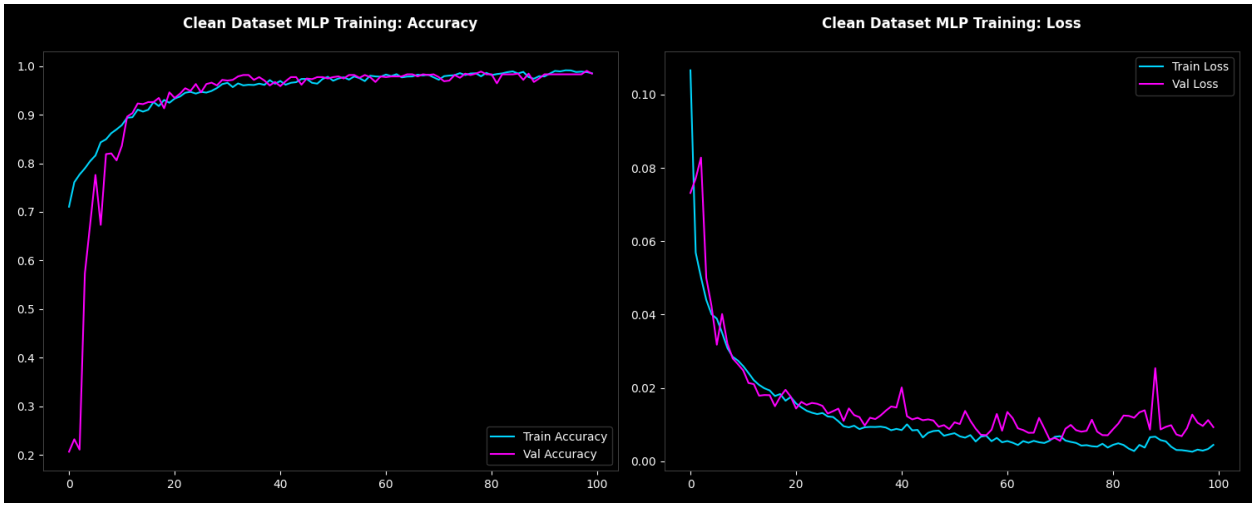

Figure 4.42: Training History of Final Iteration

From the training graph, the MLP component converged smoothly with both train and validation accuracy reaching above 95% by around epoch 20, stabilizing through the remaining epochs with no signs of overfitting. Both loss curves drop sharply early and flatten out near zero, indicating the model learned the patterns effectively.

The final inference results showed the best performance across all iterations, with an accuracy of 95.95% and a weighted F1 of 0.9591. The "Yes" recall reached 0.88, correctly identifying 58 out of 66 actual "Yes" cases, a dramatic improvement from the 0.68 in Iteration 3 and 0.23 in Iteration 1. Precision for "Yes" remained high at 0.92, meaning the model is both aggressive and accurate in predicting the minority class.

4.4.4 Final Iteration Conclusion

Metric	Iteration 1	Iteration 2	Iteration 3	Final
Accuracy	83.80%	83.80%	90.03%	95.95%
Weighted F1	0.80	0.83	0.8972	0.9591
Yes Recall	0.23	0.48	0.68	0.88
Yes F1	0.37	0.56	0.74	0.90

The two biggest contributors to the final jump in performance were the stacking ensemble. Which allowed multiple model types to compensate for each other's weaknesses and the optimal threshold tuning, which lowered the decision boundary from 0.5 to 0.121, allowing the model to capture significantly more true "Yes" cases without sacrificing too much precision.