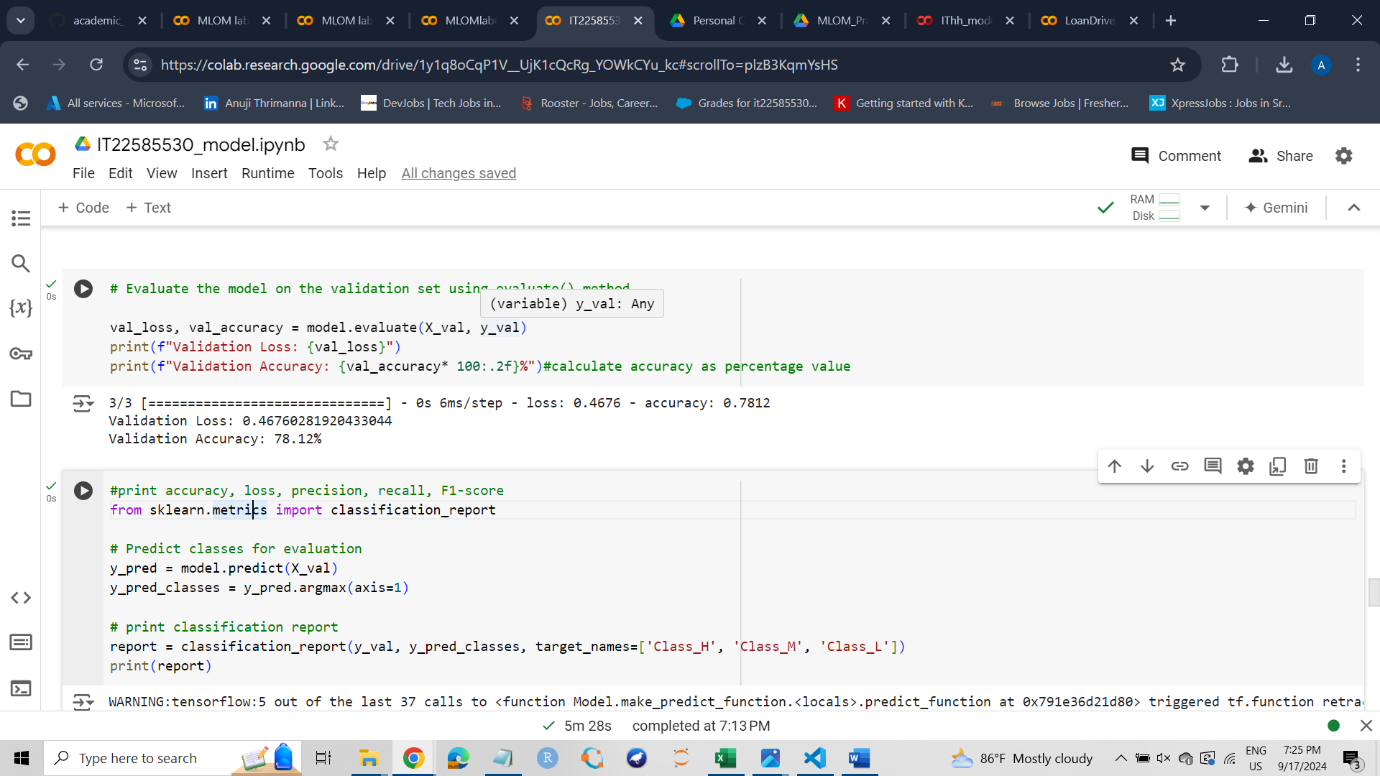
This assignment's objective was to use TensorFlow to create a deep learning model for classifying a tabular collection of student records. The process I completed was divided into several important parts, as seen below.

Findings

* Identified validation loss and validation accuracy using default hyperparameters

Validation Loss: 0.46760281920433044

Validation Accuracy: 78.12%



* Identified Report precision, recall, f1 score values before hyperparameter tuning.

A screenshot of a computer

Description automatically generated

* Identified the best Hyperparameters

Best Parameters: {'batch\_size': 16, 'learning\_rate': 0.01, 'dropout\_rate': 0.3}

* Identified best validation accuracy

A screenshot of a computer

Description automatically generated Best Validation Accuracy: 85.42%

* Plotted the graphs - Training Loss vs. Validation Loss

A screenshot of a computer

Description automatically generated

* Plotted the graphs - Training Accuracy vs. Validation Accuracy

A screenshot of a computer

Description automatically generated

* Identified the amounts of loss, accuracy, val\_loss, val\_accuracy in each epoch

A screenshot of a computer

Description automatically generated

Challenges Faced

Challenges:

* Data Preprocessing

I used to apply OneHotEncoder() to categorical features when encoding categorical variables. I was able to determine that the number of columns had increased from 17 to 72 by running the command - dataset.shape. Therefore, I used LabelEncoder() to encode categorical data instead of OneHotEncoder.

I encountered many challenges while determining whether each preprocessing component had been executed successfully.

Encountered issues with scaling, specifically handling only numerical features. After the model-designing part, I realized that the part I had done was incorrect.

* Model Overfitting/Underfitting

I had to struggle with the overfitting of the model. Therefore, dropout layers and early stopping were applied to overcome the problem

* Hyperparameter Tuning Issues

Training times were long.

It was challenging to define the hyperparameter grid and optimizing values like batch size, learning rate and dropout rate.

I ran into trouble trying to fix my code's errors.

Lessons Learned

* Importance of preprocessing – Properly scaled and encoded data allowed the model to learn more effectively.
* Importance of Hyperparameter Tuning – by adjusting learning Rate, batch Size and dropout Rate and the numb of epochs, can maximize model performance.
* Importance of evaluating models with different metrics – Using multiple evaluation metrics like accuracy, precision, recall, F1-score, and loss provides a better understanding of model performance.