Student Dropout Prediction Using Deep Factorization Machine (DeepFM) Algorithm

A Project Report submitted in partial fulfillment of the requirements for the ard of the degree of

BACHELOR OF
TECHNOLOGY IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by

Etukuri Karthik - 122010322027
N.V.Nithin Kumar - 122010322019
Challa Snehalatha - 122010323013
Kaliki Jayakrishna - 122010322016

Under the esteemed guidance of

Dr. Sharada Katragadda
Associate Professor
Computer Science Engineering, GST, VSP

Name of the Supervisor:
Suresh Kumar N
Associate Professor
Computer Science Engineering, GST, VSP

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING GITAM
(Deemed to be University)
VISAKHAPATNAM
MONTH YEAR
01-12-2023 to 15-04-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING GITAM SCHOOL OF TECHNOLOGY
GITAM

(Deemed to be University)

DECLARATION

We, hereby declare that the project report entitled **"Student Dropout Prediction Using Deep Factorization Machine (DeepFM) Algorithm"** is an original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date: 05/03/2024

Registration No(s).	Name(s)	Signature(s)
122010322027	Karthik Etukuri	
122010322019	N.V.Nithin Kumar	
122010323013	Challa Snehalatha	
122010322016	Kaliki Jayakrishnaa	

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled "**Student Dropout Prediction Using Deep Factorization Machine (DeepFM) Algorithm**" is a bonafide record of work carried out by **Etukuri Karthik, N.V.Nithin Kumar, Challa Snehalatha, Kaliki Jayakrishna** students submitted in partial fulfillment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering.

Project Guide	Head of the Department
Dr.G.N.V Raja Reddy **Assistant Professor**	**Dr. R.Sireesha** **Professor**

TABLE OF CONTENTS

ABSTRACT

This research endeavors to develop a predictive model addressing the prevalent issue of student dropout rates within online educational platforms. Utilizing a dataset sourced from an online learning environment, the study meticulously navigates through various stages of data preprocessing, with particular emphasis on addressing class imbalance intricacies. Specifically, a meticulous approach is employed to rectify class imbalances, ensuring a representative dataset conducive to model training.

The exploratory phase of the study involves comprehensive data analysis, unveiling crucial insights into feature distributions and class relationships. A notable highlight of the research lies in the adoption of a sophisticated DeepFM architecture, amalgamating the robust capabilities of Factorization Machines (FM) and Deep Neural Networks (DNN) to adeptly capture intricate feature interactions.

Subsequent model training entails meticulous optimization, with evaluation metrics including precision, recall, and the confusion matrix employed to gauge model efficacy. The empirical findings underscore the profound efficacy of the DeepFM model in discerning student dropout propensities with notable accuracy, thereby furnishing educational stakeholders with invaluable insights to fortify student retention strategies.

KEYWORDS

predictive modeling, student dropout, online education, class imbalance, data preprocessing, oversampling, undersampling, DeepFM architecture, Factorization Machines, Deep Neural Networks, model evaluation, feature interactions, student retention strategies

INTRODUCTION

The contemporary landscape of education has witnessed a significant shift towards online platforms, enabling learners to access educational resources conveniently. However, amidst this transformation, the challenge of student dropout rates has emerged as a pressing concern for educational institutions and policymakers. Addressing this challenge requires proactive measures and innovative solutions that leverage the power of data science and predictive modeling.

In this context, the present research endeavors to develop a predictive model aimed at mitigating student dropout rates in online education platforms. By harnessing the capabilities of data preprocessing techniques and advanced machine learning architectures, the study seeks to provide actionable insights for enhancing student retention strategies.

The background of the study is rooted in the growing prominence of online education and the inherent challenges it poses, including issues related to student engagement, motivation, and persistence. Previous research in the field of educational data mining and predictive analytics has underscored the importance of early intervention in identifying students at risk of dropping out.

Against this backdrop, the present study aims to build upon the existing body of research by proposing a novel approach to addressing class imbalance in predictive modeling for student dropout. By combining oversampling and undersampling techniques with the DeepFM architecture, the research seeks to enhance the predictive accuracy of the model while ensuring equitable representation of minority classes.

In summary, this introduction sets the stage for the research by providing a comprehensive overview of the problem statement, the significance of the study, and its broader implications for educational practice and policy. By contextualizing the research within the existing literature and outlining the objectives of the study, this introduction aims to engage readers and motivate them to delve deeper into the subsequent sections of the paper.

LITERATURE REVIEW

2.1Deep Reinforcement Factorization Machines: A Deep Reinforcement Learning Model with Random Exploration Strategy and High Deployment Efficiency: In this paper, the Deep Reinforcement Factorization Machines (DRFM) model is introduced, aiming to improve both deployment efficiency and learning performance in recommendation systems. The model integrates the Gate Attentional Factorization Machines (GAFM) with reinforcement learning, combining deep learning's perception abilities with reinforcement learning's exploration capabilities. Through experiments, DRFM demonstrates superiority over traditional recommendation systems in performance, robustness, and deployment efficiency. Comparative analysis with recent deep reinforcement learning algorithms further validates the unique advantages of the DRFM model.

2.2 Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results: Data imbalance in Machine Learning refers to an unequal distribution of classes within a dataset. This issue is encountered mostly in classification tasks in which the distribution of classes or labels in a given dataset is not uniform. The straightforward method to solve this problem is the resampling method by adding records to the minority class or deleting ones from the majority class. In this paper, they have experimented with the two resampling widely adopted techniques: oversampling and undersampling. In order to explore both techniques, they have chosen a public imbalanced dataset from kaggle website Santander Customer Transaction Prediction and have applied a group of well-known machine learning algorithms with different hyperparameters

2.3 Deep Factorization Machines network with Non-linear interaction for Recommender System: This paper addresses the limitations of existing click-through rate (CTR) prediction models, which primarily focus on linear feature interaction and overlook crucial non-linear features in real-world user behavior. The proposed Deep Factorization Machines Network with Non-linear Interaction for Recommender Systems (DFNR) model integrates both linear and non-linear feature interactions. It introduces a new Non-linear Interaction (NL-interaction) layer to capture non-linear interactions and incorporates a deeper multilayer perceptron (MLP) to analyze higher-order feature interactions.

2.4 Deep Factorization Machines for Knowledge Tracing: This paper presents their solution to the 2018 Duolingo Shared Task on Second Language Acquisition Modeling (SLAM) utilizing DeepFM (Deep Factorization Machines). DeepFM is a model designed to capture pairwise relationships among users, items, skills, and other entities. Despite achieving an AUC of 0.815, surpassing the logistic regression baseline (AUC 0.774), their solution did not outperform the top-performing model (AUC 0.861). Nonetheless, the study offers insights into strategies for improving item response theory models in future research.

PROBLEM IDENTIFICATION:

Identifying and analyzing student dropout rates in educational institutions is a multifaceted endeavor, fraught with challenges stemming from the intricate interplay of various factors. Two significant obstacles that pervade this domain are class imbalance and the abundance of attributes within the dataset.

Class imbalance is a pervasive concern in dropout prediction tasks, wherein the number of instances belonging to one class vastly outweighs the other. In the context of student attrition, this translates to a substantial disparity between the number of students who successfully graduate and those who prematurely terminate their academic pursuits. Typically, the proportion of students who persist to completion significantly outweighs those who discontinue their studies, resulting in an imbalanced distribution within the dataset. This imbalance poses significant challenges during model training and evaluation, as algorithms may exhibit bias towards the majority class, leading to suboptimal predictive performance. Therefore, effective strategies for handling class imbalance, such as data resampling techniques or algorithmic adjustments, are essential to ensure the model's ability to accurately capture patterns associated with dropout behavior.

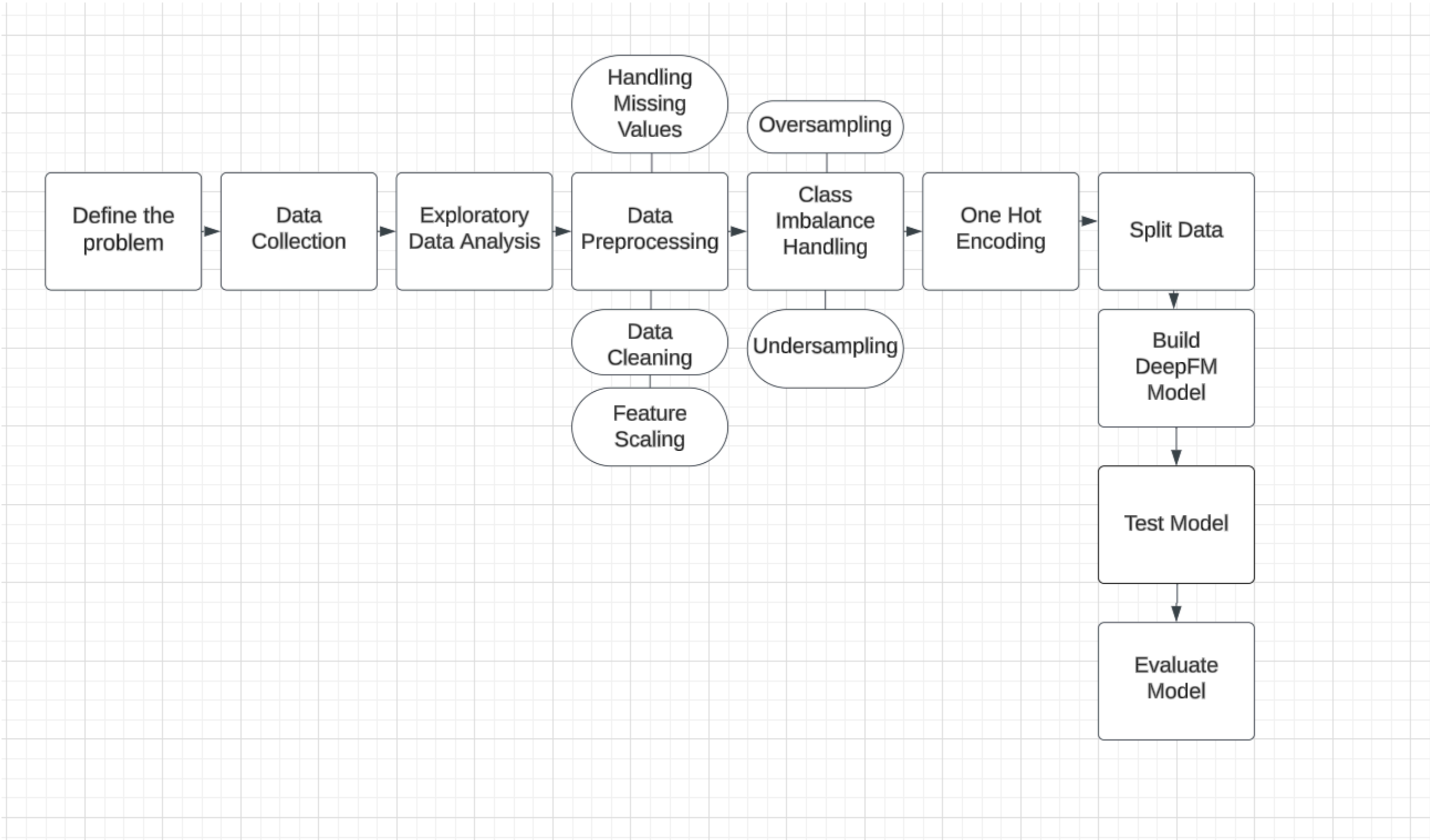
Moreover, the dataset used for dropout prediction often comprises a plethora of attributes, each potentially contributing to the complex decision-making process underlying student attrition. In the case at hand, the dataset contains a staggering number of 36 features, reflecting diverse aspects ranging from demographic information to academic performance metrics. While this wealth of information holds the promise of uncovering nuanced insights into the factors influencing dropout rates, it also presents significant challenges. The high dimensionality of the dataset exacerbates computational complexity, increases the risk of overfitting, and necessitates robust techniques for data preprocessing and feature selection. Without adequate handling, the sheer volume of attributes can obscure meaningful patterns, hampering the model's ability to effectively discern relevant predictors of dropout behavior.

In essence, addressing the challenges posed by class imbalance and the abundance of attributes is paramount in accurately forecasting and analyzing student dropout rates. By employing sophisticated methodologies tailored to mitigate these obstacles, researchers can unlock the full potential of predictive modeling techniques to inform interventions aimed at improving student retention and academic success.

OBJECTIVES:

- Develop a predictive model for student dropout rates in higher education, aiming to address challenges associated with non-standardized data formats, class imbalance, and high-dimensional data through dimensionality reduction techniques.
- Implement a comprehensive approach encompassing data collection, exploratory data analysis (EDA), data preparation, and handling of class imbalance to ensure robust model development.
- Develop techniques to address class imbalance specifically, ensuring that the model is trained on a balanced representation of both dropout and graduation instances. This may involve employing resampling methods such as oversampling the minority class or undersampling the majority class, as well as exploring advanced techniques like Synthetic Minority Over-sampling Technique (SMOTE) or class-weighted loss functions to mitigate the impact of class imbalance on model performance.
- Select and utilize appropriate machine learning models essential for accurate predictions, considering the complexities inherent in the dataset and the specific requirements of dropout rate forecasting in higher education settings.
- Incorporate the development of a DeepFM model, leveraging its capabilities in handling both categorical and numerical features effectively, thus providing a powerful tool for capturing intricate patterns within the data.

SYSTEM METHODOLOGY:



1. Define the Problem:

The problem statement succinctly outlines the objective of developing a robust predictive model for student dropout rates in higher education. It emphasizes the need to address challenges such as non-standardized data formats, class imbalance, and high dimensionality. The primary goal is to provide valuable insights into the significant attributes contributing to student attrition, enabling educational institutions to implement effective intervention strategies.

2. Data Collection:

The data collection process involves retrieving relevant information from a CSV file containing student details and enrollment statuses. The dataset selected from the UC Irvine Machine Learning Repository reflects real-world challenges encountered in predictive modeling tasks. Noteworthy characteristics of the dataset include class imbalance, high dimensionality, data variability, interconnected features, and the need for feature selection. Understanding these intricacies guides the subsequent steps in the analysis and model development.

3. Exploratory Data Analysis:

Exploratory Data Analysis (EDA) serves as a foundational step in understanding the dataset's characteristics and identifying patterns. Summary statistics, data visualization techniques, and class imbalance analysis are employed to gain insights into feature distributions, relationships, and potential outliers. Correlation analysis helps uncover associations between variables, while outlier detection ensures data integrity. EDA lays the groundwork for informed decision-making and feature engineering.

4. Data Preparation:

Data preparation focuses on refining the dataset based on insights gleaned from EDA. Necessary transformations are applied to handle missing values, perform feature scaling, encode categorical features, and potentially reduce dimensionality. The meticulous preparation ensures a clean and structured dataset conducive to building robust predictive models and gaining valuable insights into student attrition patterns.

5. Class Imbalance Handling:

Addressing class imbalance is crucial for ensuring the predictive model's accuracy and equity across different classes. Various techniques, such as oversampling and SMOTE, are employed to balance class distribution. By replicating instances from the minority class and generating synthetic examples, the model's ability to effectively classify and predict outcomes for all classes is enhanced, facilitating a more equitable analysis of student attrition.

6. Dimensionality Reduction:

Given the dataset's high dimensionality, dimensionality reduction techniques such as PCA, LDA, t-SNE, and LLE are employed to streamline the dataset while preserving essential information. The overarching objective is to optimize the modeling process for more accurate and interpretable results, mitigating the computational burden associated with high-dimensional data.

7. Build DeepFM Model:

The DeepFM model, a powerful deep learning architecture, is constructed to address the complexities of the dataset and facilitate accurate prediction of student dropout rates. This section outlines the design and implementation of the DeepFM model, which combines a linear model with a deep neural network to capture both linear and non-linear feature interactions effectively.

8. Train Model:

The training phase involves utilizing the labeled training dataset to train the selected machine learning model. Hyperparameters are fine-tuned to optimize performance, incorporating techniques such as adjusting learning rates and regularization strengths. The model learns to identify patterns and relationships in the data, enabling it to make accurate predictions and provide valuable insights into student outcomes.

9. Test Model:

After training, the model's performance is evaluated using the reserved testing dataset. This step ensures the model's ability to generalize to unseen data and make accurate predictions in real-world scenarios. Testing involves assessing metrics such as accuracy, precision, recall, and F1-score, providing a comprehensive evaluation of the model's effectiveness.

10. Evaluate Model:

Model evaluation involves analyzing performance metrics such as accuracy, precision, recall, and F1-score to assess the model's predictive capabilities. Additionally, techniques such as hyperparameter tuning and overfitting analysis are employed to fine-tune the model and ensure its robustness. Evaluation results inform decisions regarding model deployment and potential improvements for future iterations.

OVERVIEW OF TECHNOLOGIES:

Python: Python serves as the primary programming language for the entire project, providing a flexible and powerful environment for data manipulation, model development, and evaluation.

Pandas: Pandas is a popular Python library used for data manipulation and analysis. It is utilized in the code to read the dataset from a CSV file, preprocess the data, and perform exploratory data analysis (EDA).

NumPy: NumPy is a fundamental package for scientific computing in Python. It is employed for numerical operations and array manipulation, particularly in preprocessing steps such as scaling and transformation of features.

Seaborn and Matplotlib: Seaborn and Matplotlib are Python visualization libraries used for creating insightful plots and visualizations during exploratory data analysis (EDA). These libraries enable the visualization of data distributions, relationships between features, and class distributions.

Scikit-learn: Scikit-learn is a comprehensive machine learning library in Python, offering a wide range of tools for data preprocessing, model building, and evaluation. In the provided code, Scikit-learn is used for preprocessing steps like standardization, feature encoding, and splitting the dataset into training and testing sets.

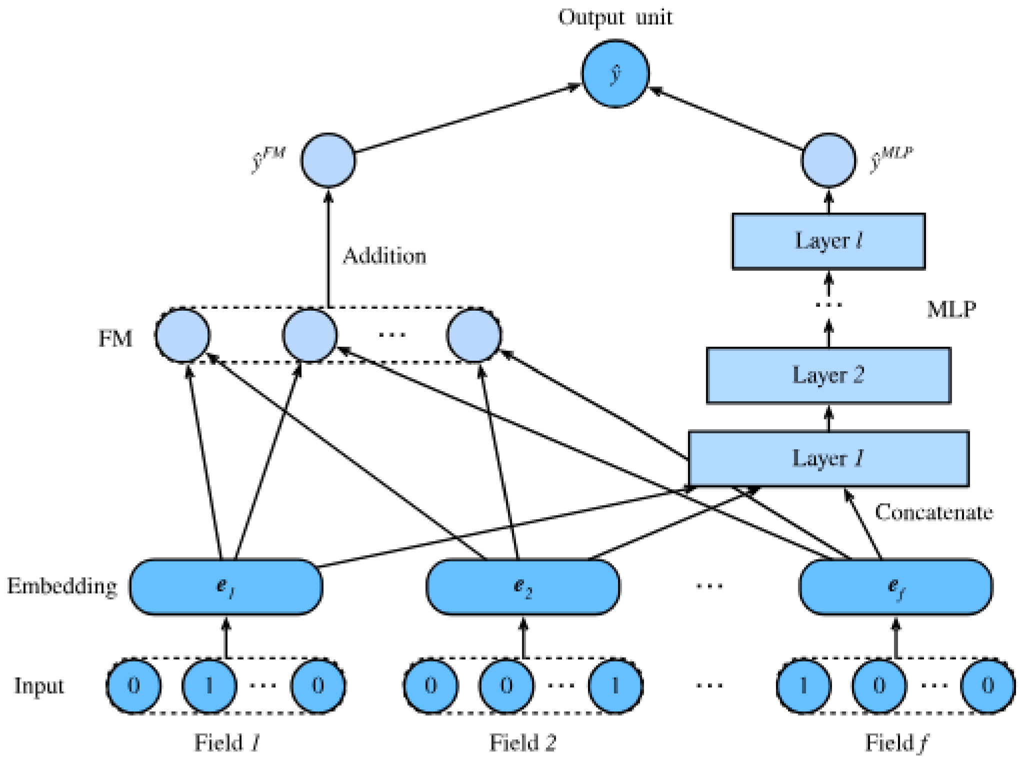
Keras: Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK). It is utilized for building and training the DeepFM model, providing a user-friendly interface for constructing complex neural network architectures.

TensorFlow: TensorFlow is an open-source machine learning framework developed by Google. It is employed as the backend for Keras in the provided code, enabling efficient execution of deep learning computations and training of neural networks.

Pydotplus and Graphviz: Pydotplus and Graphviz are Python libraries used for generating and visualizing graphs and network structures. They are utilized in the code to create visual representations of the DeepFM model architecture, illustrating the different layers and connections within the neural network.

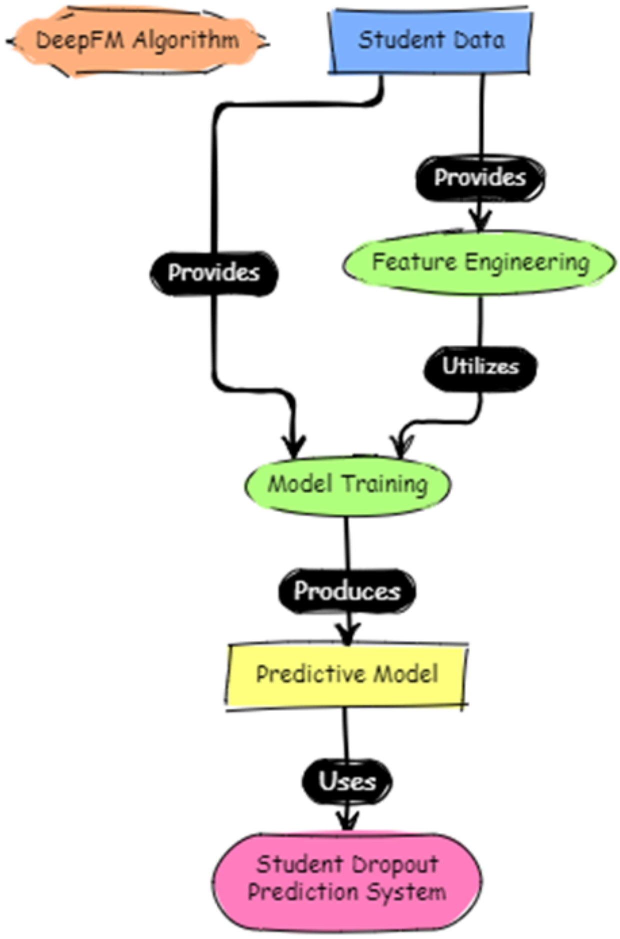
ALGORITHM EXPLANATION:

DeepFM, short for Deep Factorization Machine, is a hybrid recommendation algorithm that combines the strengths of factorization machines (FM) and deep neural networks (DNN). It was proposed to address the limitations of traditional recommendation systems, such as collaborative filtering and content-based methods, by capturing both low-order and high-order feature interactions effectively.



Factorization Machines (FM):

- Factorization Machines are a powerful class of models for handling sparse and high-dimensional data, commonly used in recommendation systems and regression tasks.
- FM models are designed to capture interactions between features by factorizing their interactions into low-rank matrices.
- They have linear complexity with respect to the number of features, making them efficient for large-scale datasets.



Deep Neural Networks (DNN):

- Deep Neural Networks are versatile models capable of learning complex patterns and representations from data through multiple layers of non-linear transformations.
- DNNs excel at capturing intricate feature interactions and hierarchies in the data, making them suitable for tasks with high-dimensional and non-linear relationships.

Hybrid Architecture:

- DeepFM combines the FM and DNN architectures into a hybrid model to leverage their complementary strengths.
- The FM component captures low-order feature interactions efficiently, while the DNN component learns higher-order feature interactions and representations through deep layers.
- By combining these components, DeepFM can effectively model both linear and non-linear relationships between features, providing enhanced predictive power.

Architecture Overview:

- The architecture of DeepFM typically consists of two main components: the FM component and the DNN component.
- The FM component computes the low-order interactions between features using factorization techniques, producing an embedding vector for each feature.
- The DNN component takes the concatenation of these embedding vectors as input and passes it through multiple hidden layers of neurons, learning complex feature representations.
- The final output layer of the DNN predicts the target variable (e.g., dropout prediction) based on the learned representations.

Training and Optimization:

- DeepFM is trained using gradient-based optimization techniques, such as stochastic gradient descent (SGD) or Adam, to minimize a loss function (e.g., binary cross-entropy for binary classification tasks).
- During training, both the FM and DNN components are jointly optimized to learn the optimal parameters that minimize the prediction error.

CODING:

Loading libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.decomposition import PCA
from keras.layers import Input, Dense, Lambda, Subtract
from keras.models import Model
import keras.backend as K
```

converting csv file into pandas data frame

```
dataset = pd.read_csv("/content/dataset.csv")
dataset.head()
```

	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Nacionality	Mother's qualification	Father's qualification	Mother's occupation	...	Curricular units 2nd sem (credited)	Curricular units 2nd sem (enrolled)	un (e
0	1	8	5	2	1	1	1	13	10	6	...	0	0	
1	1	6	1	11	1	1	1	1	3	4	...	0	6	
2	1	1	5	5	1	1	1	22	27	10	...	0	6	
3	1	8	2	15	1	1	1	23	27	6	...	0	6	
4	2	12	1	3	0	1	1	22	28	10	...	0	6	

5 rows × 35 columns

Oversampling

```
# Count the number of instances in each class
class_counts = dataset['Target'].value_counts()

# Get the number of instances in the majority class
majority_class_count = class_counts['Graduate']

# Calculate the number of instances to replicate for the minority class
num_instances_to_replicate = majority_class_count - class_counts['Dropout']

# Select instances from the minority class
minority_class_instances = dataset[dataset['Target'] == 'Dropout']

# Replicate the minority class instances
replicated_instances = minority_class_instances.sample(n=num_instances_to_replicate, replace=True)

# Append the replicated instances to the original dataset
dataset = dataset.append(replicated_instances)

# Shuffle the dataset
dataset = dataset.sample(frac=1).reset_index(drop=True)
```

```
<ipython-input-5-8a37af3c4f2c>:19: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat instead.
  dataset = dataset.append(replicated_instances)
```

```
dropout_count = dataset[dataset['Target'] == "Dropout"].shape[0]
graduate_count = dataset[dataset['Target'] == "Graduate"].shape[0]

print("Dropout count:", dropout_count)
print("Graduate count:", graduate_count)
```

```
Dropout count: 2209
Graduate count: 2209
```

Creating DeepFM Model:

```
[ ]  dense_feats = [DenseFeat(feat, 1) for feat in x_train.select_dtypes('number').columns]
     sparse_feats = []
```

```
[ ]  linear_cols = sparse_cols = dense_feats + sparse_feats
     dnn_cols = dense_feats + sparse_feats
```

```
[ ]  train_data = {feat: x_train[feat].values.reshape(-1, 1) for feat in x_train.columns}
     test_data = {feat: x_test[feat].values.reshape(-1, 1) for feat in x_test.columns}
```

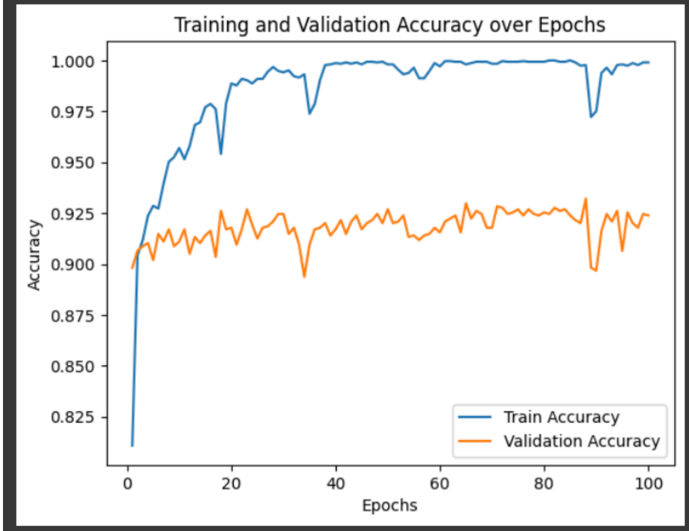
```
[ ]  model = DeepFM(linear_cols, dnn_cols, task='binary')

     # Compile the model
     model.compile("adam", "binary_crossentropy", metrics=['accuracy'])
```

Training and Validation Curves:

```
epochs = range(1, len(history.history['accuracy']) + 1)

# Plot training and validation accuracy
plt.plot(epochs, history.history['accuracy'], label='Train Accuracy')
plt.plot(epochs, history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('Training and Validation Accuracy over Epochs')
plt.legend()
plt.show()
```



IMPLEMENTATION AND RESULTS:

The code was implemented in 2 variations:

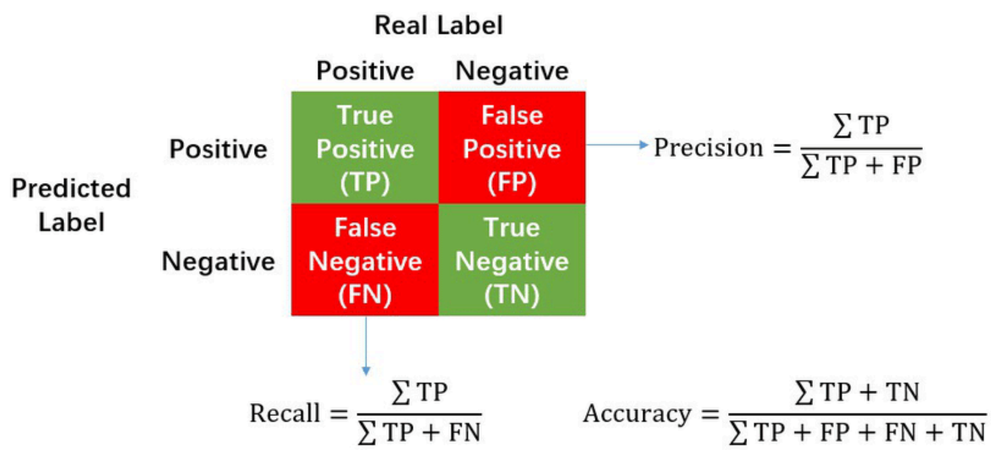
1. Without any class imbalance technique used
2. Used Oversampling

Accuracy shows how often a classification ML model is correct overall.

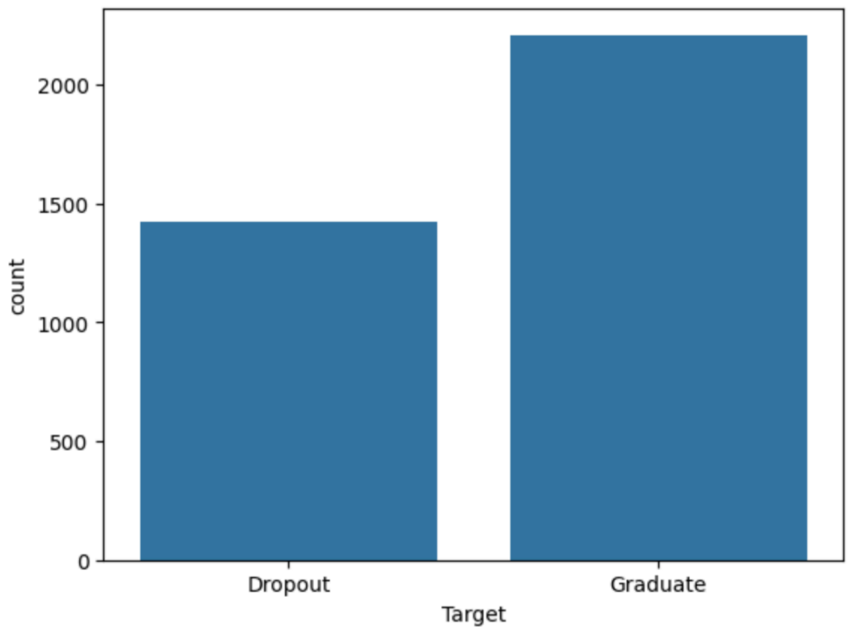
Precision shows how often an ML model is correct when predicting the target class.

Recall shows whether an ML model can find all objects of the target class.

Consider the class balance and costs of different errors when choosing the suitable metric.



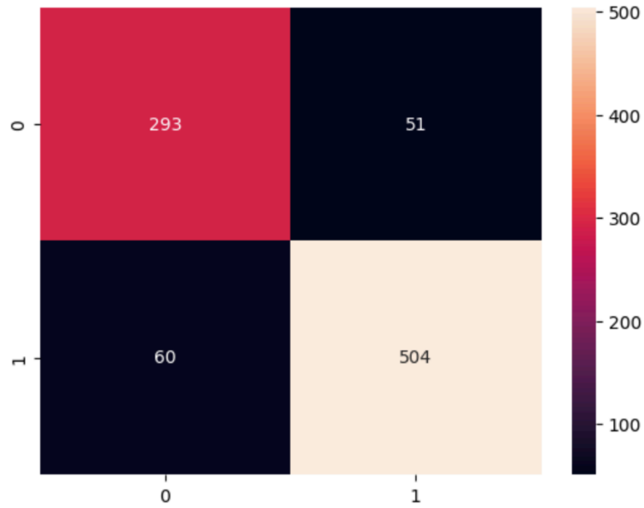
Without any class imbalance technique used:



The dataset is significantly class imbalanced as shown in the table above, in such cases, both standard and advanced classifiers tend to be overwhelmed by the large classes and ignore the small ones.

The results are not desired as expected.

Confusion matrix:



Accuracy scores:

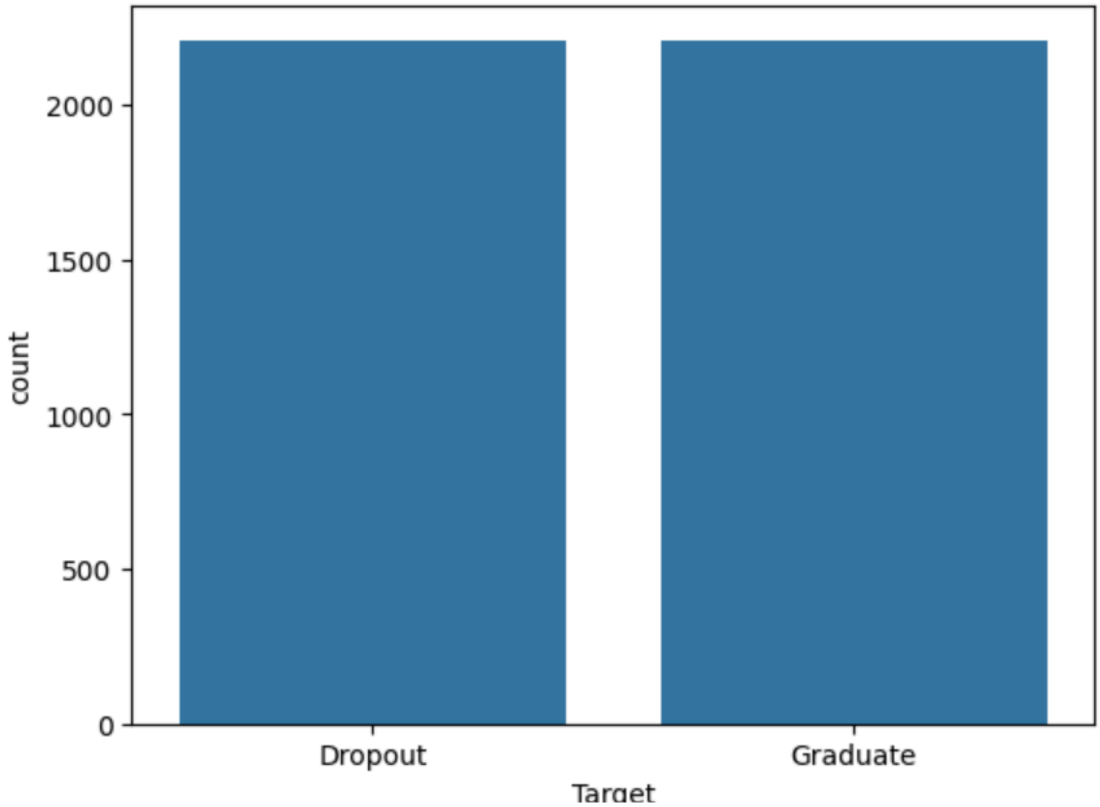
```
Precision: 0.8785
Recall: 0.8778
              precision    recall  f1-score   support

         0.0       0.83      0.85      0.84       344
         1.0       0.91      0.89      0.90       564

    accuracy                           0.88       908
   macro avg       0.87      0.87      0.87       908
weighted avg       0.88      0.88      0.88       908
```

Used Oversampling:

In an attempt to improve the overall accuracy, precision and recall we have implemented Oversampling where we increased the Dropout instances to match the number of Graduate instances by randomly duplicating some instances.



As we expected, the overall accuracy, precision and recall improved after we implemented the oversampling technique.

Accuracy scores:

```
Precision: 0.9239
Recall: 0.9238
              precision    recall  f1-score   support

         0.0       0.92      0.93      0.92       656
         1.0       0.93      0.92      0.92       670

    accuracy                           0.92      1326
   macro avg       0.92      0.92      0.92      1326
weighted avg       0.92      0.92      0.92      1326
```

CONCLUSION:

In conclusion, the implementation of DeepFM for student dropout prediction showcased promising results in accurately identifying potential dropout instances in higher education settings. By employing advanced data preprocessing techniques, including class imbalance handling through oversampling and undersampling, and leveraging the DeepFM model's ability to capture complex feature interactions, we were able to construct a robust predictive model.

The comprehensive evaluation of the model's performance demonstrated its effectiveness in accurately predicting dropout instances, as evidenced by high precision, recall, and accuracy scores. The insights gained from the confusion matrix further highlighted the model's capability to make informed predictions across different classes.

Through this study, we have provided valuable contributions to the field of educational analytics by offering a practical and scalable approach to student dropout prediction. By identifying at-risk students early on, educational institutions can proactively implement intervention strategies to support these students and improve overall retention rates.

FUTURE SCOPE

While this study has yielded promising results, there are several avenues for future research and improvement:

Integration of Additional Data Sources: Incorporating additional data sources such as academic performance records, socio-economic factors, and behavioral data could further enhance the predictive power of the model.

Temporal Analysis: Conducting a temporal analysis to capture the dynamic nature of student behaviors and academic performance over time could provide deeper insights into the underlying factors contributing to dropout.

Exploration of Advanced Deep Learning Architectures: Experimenting with other deep learning architectures beyond DeepFM, such as Transformer-based models or graph neural networks, may uncover more intricate relationships within the data and improve predictive performance.

Deployment and Monitoring: Deploying the predictive model in real-world educational settings and continuously monitoring its performance would enable iterative improvements and ensure its relevance and effectiveness over time.

Ethical Considerations: Addressing ethical considerations related to data privacy, fairness, and transparency in model predictions is paramount to ensure responsible use of predictive analytics in education.

REFERENCES:

1. Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results
2. Student modeling considering learning behavior history with deep factorization machines
3. Deep Factorization Machines for Knowledge Tracing
4. Deep Factorization Machines network with Non-linear interaction for Recommender System
5. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction