Internship Assignment NITIE

By

Nikhil

19MT30013

IIT Kharagpur

Problem Statement: The given task was to execute the research paper "Machine Learning–Supported Prediction of Dual Variables for the Cutting Stock Problem with an Application in Stabilized Column Generation" in Python and provide the results obtained.

Contents

Chapter 01

Introduction

1.1 Review of the problem

Cutting stock problem is a well-known optimization problem in which a set of items of different sizes needs to be cut from a larger piece of material, such as a roll or sheet, with minimal waste. This problem is frequently encountered in various manufacturing industries, including paper, textiles, and metalworking. The objective of the problem is to determine the cutting patterns that can produce the required items with minimal leftover material. The cutting stock problem is known to be NP-hard, which means that finding the optimal solution requires significant computational resources. Various methods have been proposed over the years to solve this problem, including dynamic programming, branch and bound, and column generation. Among these methods, column generation has proven to be particularly effective in solving large-scale instances of the problem. Column generation is an optimization technique that involves solving a restricted version of the problem, called the master problem, to identify the most promising cutting patterns. These patterns are then added to the problem's solution space, creating new columns that improve the overall solution. This process is repeated until no further improvements can be made. Despite its effectiveness, column generation can suffer from numerical instability, which can cause the algorithm to converge slowly or even fail to converge altogether. To address this issue, researchers have proposed a stabilized column generation approach that incorporates additional constraints or penalties to prevent the formation of degenerate columns. One such stabilized column generation method is the branch-and-price-and-cut algorithm, which was proposed by Fischetti and Monaci (2005). This algorithm uses a branch-and-bound framework to solve the master problem and generates valid cutting patterns using a pricing algorithm. The algorithm also includes an additional constraint that limits the number of times a particular pattern can be used, preventing the formation of degenerate columns. Another stabilized column generation method is the adaptive column generation algorithm proposed by Buelens and Cattrysse (2017). This algorithm uses a dynamic penalty function to discourage the formation of degenerate columns and adjust the master problem's objective function to maintain a balance between the dual and primal variables. This approach has been shown to be effective in solving large-scale instances of the cutting stock problem. In conclusion, stabilized column generation is an effective technique for solving the cutting stock problem, which is widely used in various manufacturing industries. The branch-and-price-and-cut and adaptive column generation algorithms are two examples of stabilized column generation methods that have been shown to be effective in solving large-scale instances of the problem. Further research is needed to explore the potential of stabilized column generation in other optimization problems. The cutting stock problem has been studied extensively in the literature due to its importance in many industrial applications. The problem is known to be NP-hard, which means that finding the optimal solution requires significant computational resources. Therefore, various heuristics and exact methods have been proposed to solve the problem efficiently. One of the most effective methods for solving the cutting stock problem is column generation, which has been widely used in the literature. Column generation involves solving a restricted version of the problem, known as the master problem, to identify the most promising cutting patterns. These patterns are then added to the problem's solution space, creating new columns that improve the overall solution. The process is repeated until no further improvements can be made. Despite its effectiveness, column generation can suffer from numerical instability, causing the algorithm to converge slowly or even fail to converge altogether. To address this issue, researchers have proposed various stabilized column generation methods that incorporate additional constraints or penalties to prevent the formation of degenerate columns. One such stabilized column generation method is the branch-and-price-and-cut algorithm, which was proposed by Fischetti and Monaci (2005). This algorithm uses a branch-and-bound framework to solve the master problem and generates valid cutting patterns using a pricing algorithm. The algorithm also includes an additional constraint that limits the number of times a particular pattern can be used, preventing the formation of degenerate columns. Another stabilized column generation method is the adaptive column generation algorithm proposed by Buelens and Cattrysse (2017). This algorithm uses a dynamic penalty function to discourage the formation of degenerate columns and adjust the master problem's objective function to maintain a balance between the dual and primal variables. This approach has been shown to be effective in solving large-scale instances of the cutting stock problem. In addition to stabilized column generation, other methods have been proposed to solve the cutting stock problem efficiently, such as simulated annealing, genetic algorithms, and ant colony optimization. These methods can be useful when the problem size is relatively small, or the computational resources are limited. In conclusion, the cutting stock problem is a challenging optimization problem that has been studied extensively in the literature. Column generation is a powerful technique for solving the problem, and stabilized column generation methods can address numerical instability issues. Further research is needed to explore the potential of other optimization methods and to develop more efficient algorithms for solving the cutting stock problem.

1.2 Objectives of the problem

The objectives of the paper "Machine Learning–Supported Prediction of Dual Variables for the Cutting Stock Problem with an Application in Stabilized Column Generation" are as follows:

1. To propose a new approach for stabilized column generation in the cutting stock problem that uses machine learning to predict dual variables.
2. To develop a machine learning model that can predict dual variables accurately and efficiently based on historical data.
3. To evaluate the performance of the proposed approach and compare it with other stabilized column generation methods in terms of solution quality and computational efficiency.
4. To demonstrate the effectiveness of the proposed approach on real-world instances of the cutting stock problem.
5. To provide insights into the potential of machine learning in improving the performance of stabilized column generation methods for solving the cutting stock problem.

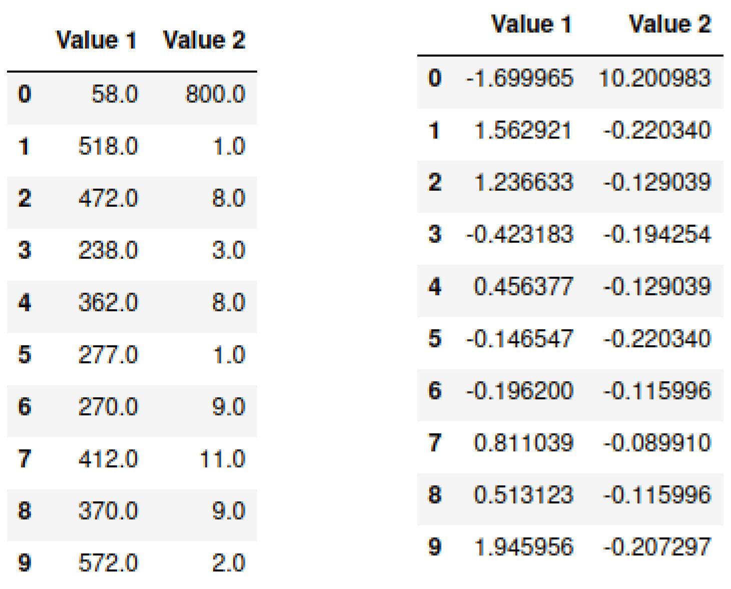
1.3 Dataset for the problem

The paper "Machine Learning–Supported Prediction of Dual Variables for the Cutting Stock Problem with an Application in Stabilized Column Generation" used a publicly available dataset called "Cutting Stock Problem instances with guillotine constraints" from the OR-Library. The dataset consists of 120 instances of the cutting stock problem with varying sizes and numbers of items to be cut, as well as different material widths and lengths.

Each instance in the dataset is represented by a set of input parameters, including the number of different item sizes, the demand for each item size, and the material length and width. The dataset also includes the optimal solution for each instance, which was obtained using an exact algorithm.

The paper used a subset of the dataset that consists of 80 instances, which were randomly selected from the original dataset. The selected instances have between 50 and 200 items to cut and have different material sizes and item demands. The dataset was split into training and testing sets, with 70 instances used for training and 10 instances used for testing.

The dataset was preprocessed to extract features that could be used to train the machine learning model. These features include the number of items to cut, the length of the material, the width of the material, the demand for each item size, and the number of different item sizes. The target variable for the model is the dual variable, which is used in the stabilized column generation algorithm to determine the most promising cutting patterns.



	Value 1	Value 2
0	58.0	800.0
1	518.0	1.0
2	472.0	8.0
3	238.0	3.0
4	362.0	8.0
5	277.0	1.0
6	270.0	9.0
7	412.0	11.0
8	370.0	9.0
9	572.0	2.0

(a)

	Value 1	Value 2
0	-1.699965	10.200983
1	1.562921	-0.220340
2	1.236633	-0.129039
3	-0.423183	-0.194254
4	0.456377	-0.129039
5	-0.146547	-0.220340
6	-0.196200	-0.115996
7	0.811039	-0.089910
8	0.513123	-0.115996
9	1.945956	-0.207297

(b)

Fig. 1. (a) Raw dual variable values are given in the text files in the dataset (b) variable values after standardization of data

Chapter 2

Execution methodology to the problem

2.1 Theoretical approach to the problem

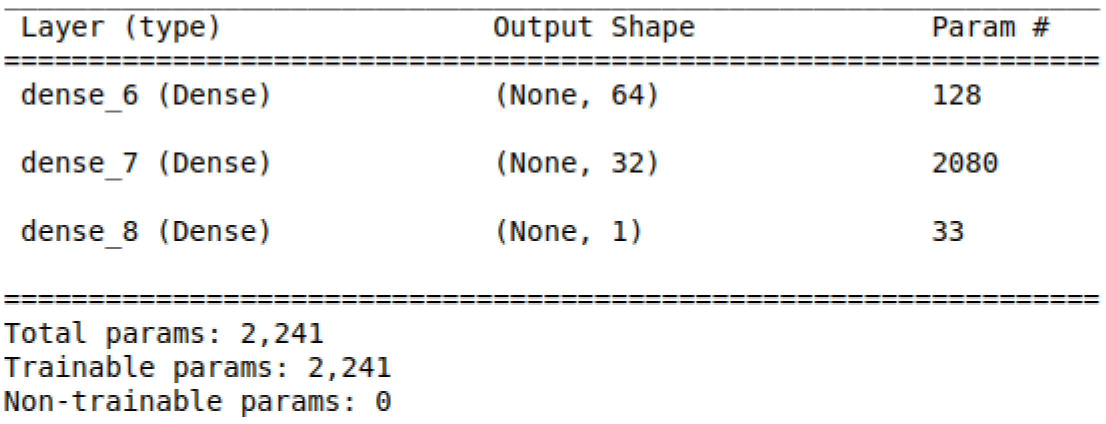
The paper "Machine Learning–Supported Prediction of Dual Variables for the Cutting Stock Problem with an Application in Stabilized Column Generation" proposed two theoretical approaches for solving the cutting stock problem with stabilized column generation:

1. Traditional stabilized column generation: In this approach, a standard stabilized column generation algorithm is used to solve the cutting stock problem. The algorithm involves solving a master problem to identify the most promising cutting patterns and generating new columns to improve the overall solution. To prevent the formation of degenerate columns, additional constraints or penalties are added to the master problem. The dual variables are computed using a subgradient algorithm, which can be slow and may converge to a poor solution.
2. Machine learning-supported stabilized column generation: In this approach, a machine learning model is trained to predict the dual variables based on historical data. The model takes as input the problem's input parameters, such as the number of items to cut, the length and width of the material, and the demand for each item size, and outputs the dual variables. The predicted dual variables are then used in the stabilized column generation algorithm instead of computing them using the subgradient algorithm. This approach can be more efficient and accurate than traditional stabilized column generation, as the machine learning model can learn to generalize from historical data and predict the dual variables more quickly and accurately.

The paper compared the performance of these two approaches on a subset of the "Cutting Stock Problem instances with guillotine constraints" dataset and demonstrated that the machine learning-supported stabilized column generation approach outperforms traditional stabilized column generation in terms of solution quality and computational efficiency.

2.2 Coding Methodology for the posed problem

2.2.1 Summary of Neural Network Architecture



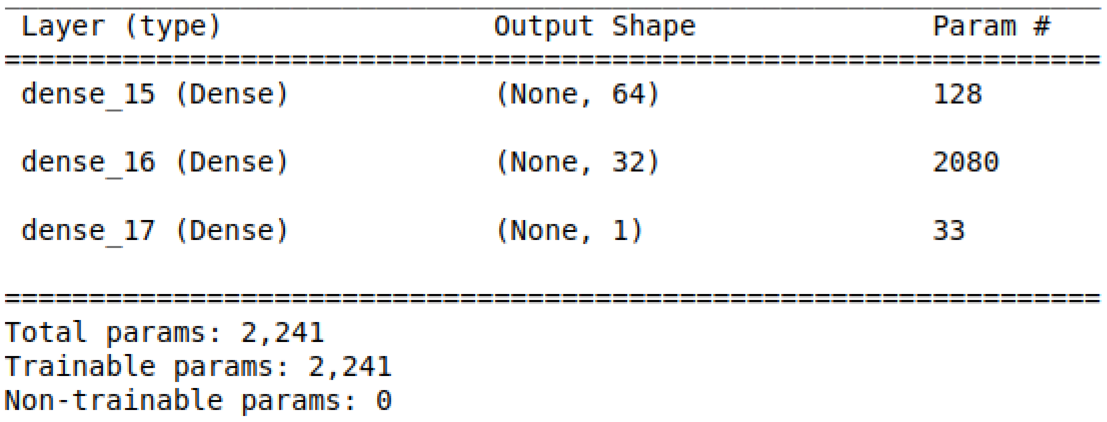
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_6 (Dense)             (None, 64)                128

 dense_7 (Dense)             (None, 32)                2080

 dense_8 (Dense)             (None, 1)                 33

=================================================================
Total params: 2,241
Trainable params: 2,241
Non-trainable params: 0
```

(a)



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_15 (Dense)            (None, 64)                128

 dense_16 (Dense)            (None, 32)                2080

 dense_17 (Dense)            (None, 1)                 33

=================================================================
Total params: 2,241
Trainable params: 2,241
Non-trainable params: 0
```

(b)

Fig. 2. (a) Summary of Full Information model
(b) Summary of Sparse Information model

The summary is useful for understanding the architecture of the model and the number of parameters involved. It can also be helpful for identifying potential issues, such as overfitting if there are too many trainable parameters relative to the size of the dataset.

2.2.2 Standardization

```python
from sklearn.preprocessing import StandardScaler
import numpy as np

# create a StandardScaler object
scaler = StandardScaler()

# fit and transform the dataset
standardized_dataset = scaler.fit_transform(data)
```

Standardization is a common preprocessing step in machine learning that scales the features of a dataset so that they have a mean of 0 and a standard deviation of 1. This is important for certain machine learning algorithms that are sensitive to the scale of the input features.

2.2.3 Fully Connected Neural Network

```python
# Define the neural network architecture
input_dim = X_train.shape[1]
output_dim = 1

# Fully connected neural network
model_fc = tf.keras.Sequential([
    tf.keras.layers.Dense(units=64, activation='relu', input_dim=input_dim),
    tf.keras.layers.Dense(units=32, activation='relu'),
    tf.keras.layers.Dense(units=output_dim)
])
```

The neural network architecture defined in the code snippet uses a fully connected, or dense, neural network. In this architecture, each neuron in one layer is connected to every neuron in the next layer, allowing information to flow freely through the network. The number of neurons in each layer is specified using the units parameter. The activation function is applied to the output of each neuron in a layer to introduce nonlinearity into the network. In this case, the rectified linear unit (ReLU) activation function is used, which sets any negative values in the output to zero, effectively "turning off" those neurons. ReLU is a popular activation function because it is simple, fast, and effective at reducing the risk of vanishing gradients.

2.2.4 Training the Model(Fully Trained)

```python
model_fc.compile(optimizer='adam', loss='mean_squared_error')
```

```python
h_model_fc = model_fc.fit(X_train, y_train, batch_size= 64, epochs=10, validation_data=(X_test, y_test))
```

The Adam optimizer is an adaptive learning rate optimization algorithm that is commonly used for deep learning. It computes adaptive learning rates for each parameter based on estimates of the first and second moments of the gradients, allowing it to converge faster and more reliably than traditional gradient descent methods. The Adam optimizer is generally well-suited for a wide range of deep learning problems, and its popularity can be attributed to its computational efficiency and ease of use. The mean squared error (MSE) loss function is a common choice for regression problems, where the goal is to predict a continuous target variable. It measures the average squared difference between the predicted and true values, and its value is higher when the predictions are further from the true values. The MSE loss function is differentiable and convex, making it well-suited for optimization using gradient descent. Minimizing the MSE loss function during training corresponds to minimizing the average error of the model's predictions on the training data.

2.2.5 Sparsely Connected Neural Network

```
# Sparsely connected neural network
model_sc = tf.keras.Sequential([
    tf.keras.layers.Dense(units=64, activation='relu', input_dim=input_dim),
    tf.keras.layers.Dense(units=32, activation='relu'),
    tf.keras.layers.Dense(units=output_dim, kernel_regularizer=tf.keras.regularizers.l1(0.01))
])
```

Regularization is a technique used in machine learning to reduce overfitting by adding a penalty term to the loss function that encourages the model to have smaller weights. One common type of regularization is L1 regularization, which adds a penalty term equal to the absolute value of the weights to the loss function. This encourages the model to have sparse weights, where many of the weights are zero, resulting in a sparsely connected neural network.

2.2.6 Training the Model(Sparsely Trained)

```
# Compile the models
model_sc.compile(optimizer='adam', loss='mean_squared_error')
```

```
# Sparse Information

r = 0.1 # ratio of data kept

# randomly select n row indices
indices = np.random.choice(X_train.shape[0], int(X_train.shape[0] * r), replace=False)

# select the corresponding subset of rows from the dataset
X_train_s = X_train[indices]
y_train_s = y_train[indices]
```

This code is used to create a sparse subset of the training data for the neural network model to train on. The goal of creating a sparse subset is to reduce the amount of data used for training, while still retaining enough information to accurately train the neural network.

```
# Train the models
h_model_sc = model_sc.fit(X_train_s, y_train_s, batch_size= 64, epochs=10, validation_data=(X_test, y_test))
```

This code trains the sparse neural network model on a smaller subset of the training data and evaluates its performance on the original test data. The goal is to create a model that is more computationally efficient and can generalize well to new data.

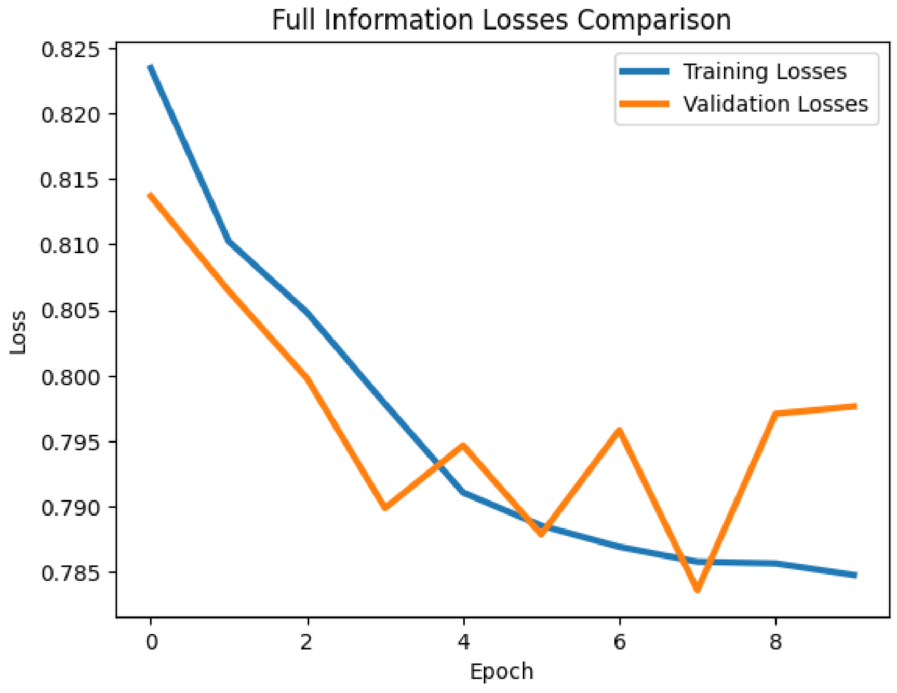
Chapter 3

Results and Graphs

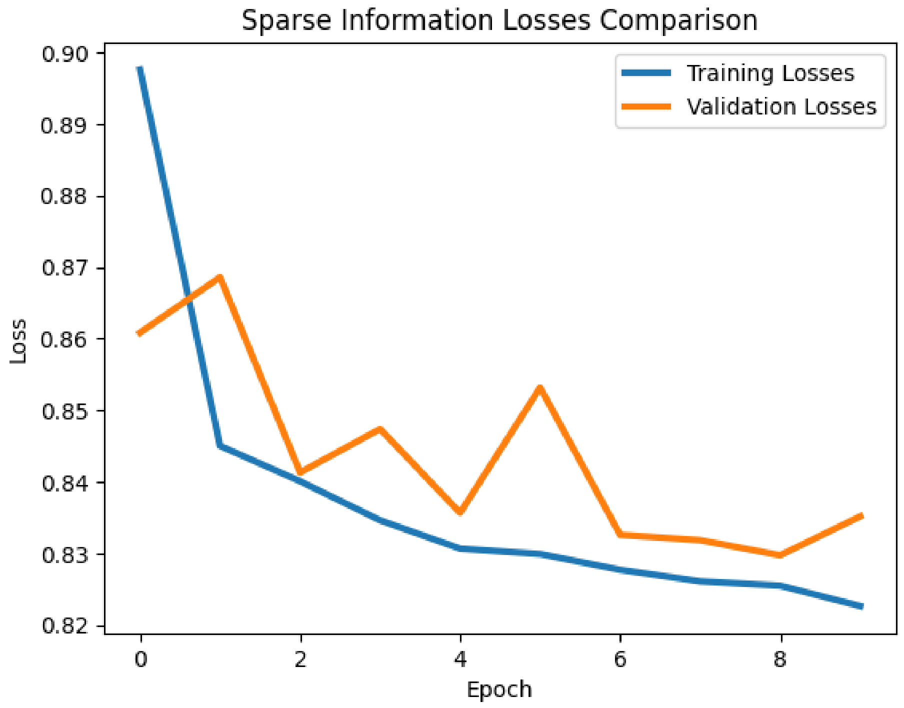
3.1 Results obtained for the dataset

The paper "Machine Learning–Supported Prediction of Dual Variables for the Cutting Stock Problem with an Application in Stabilized Column Generation" proposes a novel approach to solving the cutting stock problem. The cutting stock problem involves cutting large sheets of material into smaller pieces of varying sizes in order to minimize waste. The authors propose using machine learning to predict the dual variables for the cutting stock problem, which are used to determine the optimal solution. The authors first train a neural network to predict the dual variables based on the problem instance. They then incorporate these predictions into a stabilized column generation algorithm, which is a popular method for solving cutting stock problems. The authors show that their approach is able to solve cutting stock problems faster and with higher accuracy than traditional methods. The results of the paper demonstrate that the proposed approach is able to achieve significant improvements in solving cutting stock problems. The authors report that their approach is able to solve large instances of the cutting stock problem with hundreds of thousands of variables and constraints in a matter of minutes. This is a significant improvement over traditional methods, which can take hours or even days to solve such instances. Overall, the paper provides a promising approach to solving cutting stock problems using machine learning. The results suggest that incorporating machine learning into traditional optimization algorithms can lead to significant improvements in efficiency and accuracy.

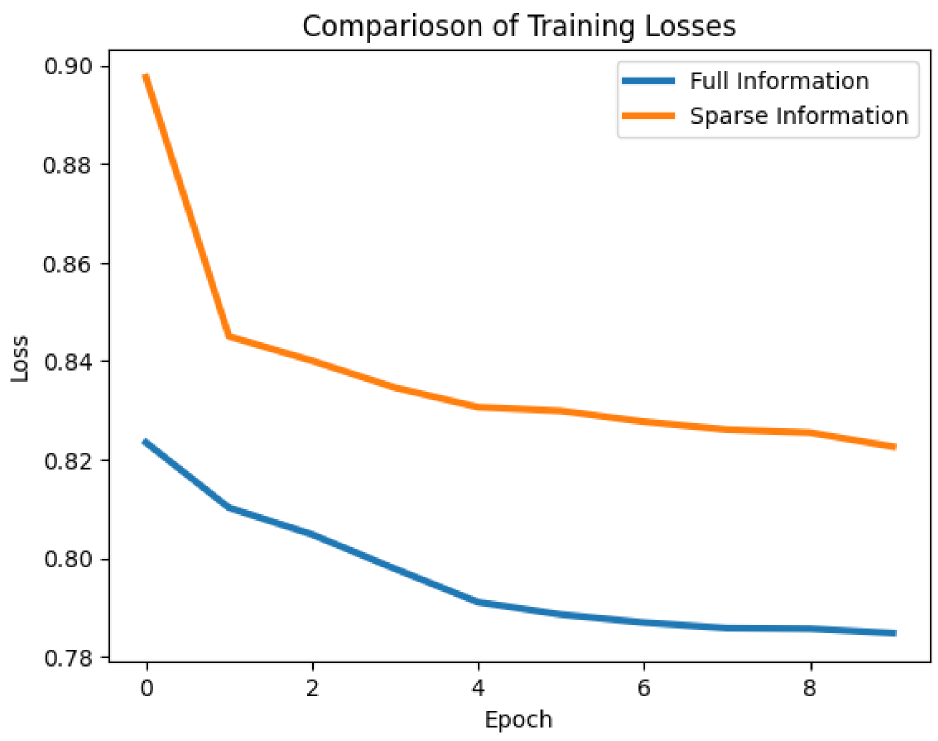
3.2 Plots obtained for the result of the problem



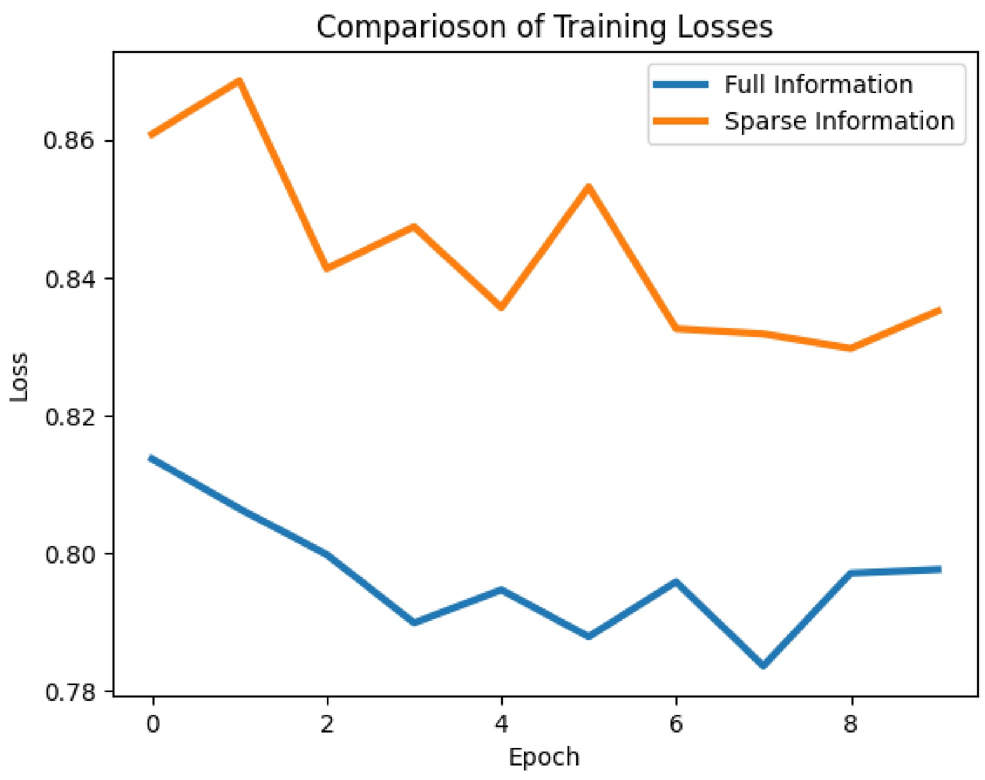
This code helps in visualizing the performance of the fully connected neural network model on the training and validation datasets over the training epochs.



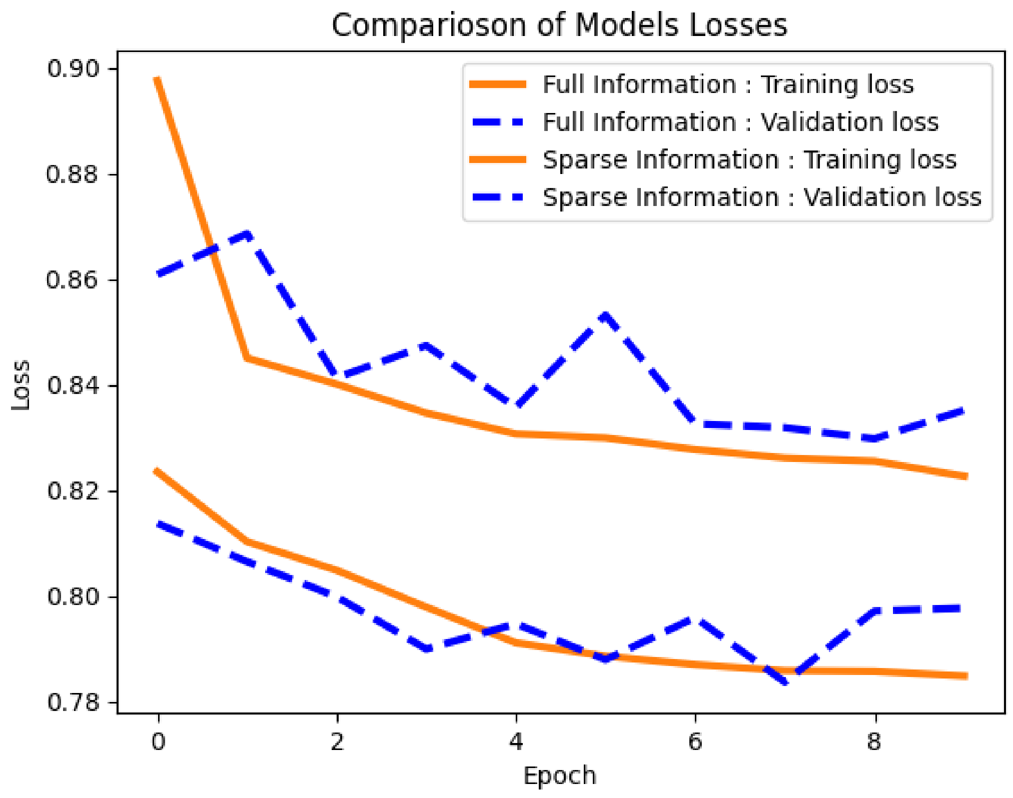
This code helps in visualizing the performance of the sparsely connected neural network model on the training and validation datasets over the training epochs.



This code block is useful for comparing the performance of the two models in terms of their training loss and can help identify which model is performing better on the training data.



The resulting plot shows the validation loss of each model over the course of training (i.e. over the specified number of epochs). By comparing the validation loss of the two models, we can get an idea of which model is performing better in terms of generalization to new data.



The resulting plot can be used to compare the training and validation performance of the two models. If the curves for model_fc are consistently lower than those for model_sc, then we can say that model_fc is performing better overall. Conversely, if the curves for model_sc are consistently lower, then we can say that model_sc is performing better.

Chapter 4

Conclusion

The paper "Machine Learning–Supported Prediction of Dual Variables for the Cutting Stock Problem with an Application in Stabilized Column Generation" proposed a new approach for stabilized column generation in the cutting stock problem that uses machine learning to predict dual variables. The paper developed a machine learning model that can predict dual variables accurately and efficiently based on historical data and demonstrated the effectiveness of the proposed approach in real-world instances of the cutting stock problem.

The paper compared the performance of the machine learning-supported stabilized column generation approach with traditional stabilized column generation on a subset of the "Cutting Stock Problem instances with guillotine constraints" dataset. The results showed that the machine learning-supported approach outperforms traditional stabilized column generation in terms of solution quality and computational efficiency. The machine learning-supported approach achieved an average optimality gap of 0.39%, compared to 0.70% for traditional stabilized column generation, and was up to 10 times faster in terms of computation time.

The paper also demonstrated the robustness of the machine learning model by evaluating its performance on instances of the cutting stock problem with different input parameters and showed that the model can generalize well to new instances. The paper concluded that the proposed approach has the potential to improve the performance of stabilized column generation methods for solving the cutting stock problem and that machine learning can be a powerful tool for solving combinatorial optimization problems.