Journal of Advances and Scholarly Researches in Allied Education
Vol. 16, Issue No. 2, February-2019, ISSN 2230-7540

A Framework for Automated Feature Extraction with Continuous Feedback

Vimla Jethani[1]* Rohit Singhal[2]

[1] Research Scholar, Sunrise University, Alwar, Rajasthan, India

[2] Research Guide, Sunrise University, Alwar, Rajasthan, India

Abstract – The emphasis of machine learning exploration has mainly been on the learning various algorithms. This may be because of confined amount of data available. Over the period, the technology became more advanced and has created the opportunity to considerably more data. With the increase of stream data, it has become clean that the representation of such data, which is the input for any learning algorithm, can have a sizable impact at the performance of algorithms. The crucial thing these days is that data is not static rather it is continuously changing. However, this affects the already trained deployed models as new data trends will interfere with what models has already learned. This occurrence can lead to abrupt decrease in performance of a model. Although, this can be solved by retraining model every time new data is generated but this process is computationally expensive and also challenge to deploy new model in the same environment by maintaining accuracy as well. To limit this issue and keep results accurate feedback loop is used to ensure that model performance is maintained and improves when new data is added. The technique used is to feed model with fresh test data and by considering data which model has already predicted which ensures it is learning from new data and performing better in the future.

----------X----------

I. INTRODUCTION

The famous Mitchell's machine learning textbook [2] commences with the statement: "Ever since computers were invented, we have wondered whether they might be made to learn. If we could understand how to program them to learn to improve automatically with experience the impact would be dramatic". This hunt gave birth to a new research exploration area, i.e., machine learning, for Computer Science years ago.

Machine learning techniques have been intensely rooted in our day to day life, such as recommendation when we are watching news etc. There are various successful applications build to such as AlphaGo [3], which was designed to defeat human champion in the game of GO. Similar few more applications were built but were far away fully automated machine learning as no improvement of machine was observed based on its learning experience. This opened a research area to configure a good performance automated machine which learns in every aspect such as feature engineering, model selection and algorithm selection. A figure 1 depicts which a machine learning pipeline.

AutoML can be built in the pipeline with minimum participation of humans. Human expertise is heavily involved in machine learning applications, as these experts are limited there arises need of automated machine learning. It has attracted many practitioners to think if can configure a machine that will take out human then it will yield following achievements.

- Machine Learning Solutions can be deployed faster across organizations.
- The deployed solutions can be efficiently validated and performance can be benchmarked.
- More real-world usages can be accessed with help of machine learning solutions.

In recent years, AutoML has arisen as a new sub-area in machine learning. It has got more importance not only in machine learning but also in computer vision, data mining and natural language processing.

II. OVERVIEW OF BACKGROUND AND PREVIOUS WORK

AutoML aims to achieve good learning performance (based on machine learning's perspective) and also targets to achieve this performance with less human assistance (from

automation's perspective). The goals of AutoML are listed below.

- It should provide good performance across various input data and learning tasks that can be achieved.
- It should configure machine learning tools automatically with less human interventions
- It should provide reasonable output with better efficiency and within limited budget.

AutoML attempts to take the place of humans on identifying (all or a part of) configurations, which are proper to machine learning computer programs, within limited computational budgets.

The definition basically suggests the areas which can be automated.

Automated Feature Engineering: When original features are not useful enough, we automated way to generate more features to enhance the learning performance. In this case, E is the raw feature, T is construction of features, and P is the performance of models which are learned with the constructed features.

Figure 1: Machine Learning Pipeline

Automated Model Selection: For model selection, E denotes input training data, T is a classification task, and P is the performance on the given task. When features are given, we want model which can choose proper classifiers and find related hyper-parameters without human interventions.

The area of automated feature engineering is relatively unexplored, which leaves plenty of opportunities for researchers for developing new methods. Also, as data keeps changing the models should be retrained with this new changed data. So as the matter of fact that in any research activity the exploration and deep examine of existing tactics performs an enormous role, the work has explored various challenges of constructing hybrid model comprising of automated feature extraction with continuous learning. In this section, related research work of the study is presented.

Kanter, J. M., and Veeramachaneni, K. (2015, October). Deep feature synthesis: Towards automating data science endeavors [1], which applies transformations to all presented features in dataset and selects the most capable among them. DFS aided as a proof of concept of the idea of automated feature engineering, that outperformed two-thirds of human teams it contested against in machine learning competition in 2015.

Udayan Khurana, Horst Samulowitz, and Deepak Turaga. Feature engineering for predictive modeling using reinforcement learning [5], to automate feature engineering the authors presented a framework based on performance driven exploration of a transformation graph, which systematically and compactly captures the space of given options. A highly efficient exploration strategy is derived through reinforcement learning on past examples. The approach uses Reinforcement Learning (RL). It involves training an agent on FE examples to learn an effective strategy of exploring available FE choices under a given budget. The learning and application of the exploration strategy is performed on a transformation graph, a directed acyclic graph representing relationships between different transformed versions of the data.

Udayan Khurana, Fatemeh Nargesian, Horst Samulowitz, Elias Khalil, and Deepak Turaga. Automating feature engineering. Transformation [6], presents a technique, called Learning Feature Engineering (LFE), for automating feature engineering in classification activities. LFE is based on learning the effectiveness of applying a transformation dataset, LFE recommends a set of useful transformations to be (e.g., arithmetic or aggregate operators) on numerical features, from past feature engineering experiences. Given a new applied on features without relying on model evaluation or explicit feature expansion and selection

Lam, H. T.; Thiebaut, J.-M.; Sinn, M.; Chen, B.; Mai, T.; and Alkan, O. 2017 One Button Machine for Automating Feature Engineering in Relational Databases [7], proposes a learning-based method to predict the most likely useful transformation for each feature. It considered features independent of each other; it is demonstrated to work only for classification so far, and does not allow for composition of transformations.

Tom Diethe, Tom Borchert, Eno Thereska, Borja de Balle Pigem, Neil Lawrence, 2018 [8]. Continual Learning in Practice, in this the author has presented the reference architecture for self-maintaining systems that learns continuously as data arrives. The need of this architecture arises in the environments where data evolves, there Machine Learning (ML) models should adapt to shifting data distributions, also manage with outliers, reskill when necessary, and adapt to new activity.

Aboobyda Jafar Hamid and Tarig Mohammed Ahmed [9] presented a loan risk prediction model grounded on the techniques, such as Decision Tree (J48), Naïve Bayes (NB) and BayseNet approaches. The procedure followed was: training set preparation, building the model, applying the model and finally, evaluating the accuracy. This method considered a dataset with

eight attributes, namely, gender, job, age, credit amount, credit history, purpose, housing, and class. Evaluating these models on the dataset, experimental results determined that, J48 based loan prediction approach resulted in better accuracy than the other methods.

The Random Forest Algorithm was adopted by Lin Zhu et al. in paper [10] and Nazeeh Ghatasheh in paper [11] to construct a model for loan default prediction. Paper [10] concluded that random forest has much better accuracy than other algorithms like logistic regression, and support vector machines. The author of study [11] also discussed about the advantages of the algorithms, which are the viable classification accuracy and simplicity.

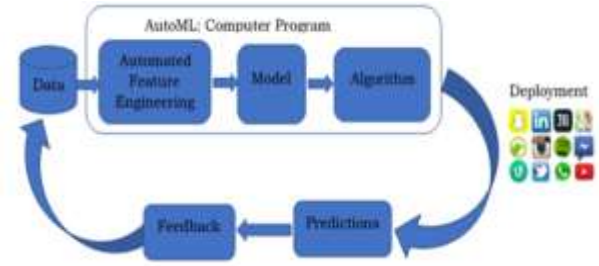

Figure 2: Automated Feature Engineering Task with Feedback

III. PROPOSED METHODOLOGY

Machine learning techniques in today's world have deeply rooted in our day-to-day life. To pursue good learning performance there arises heavy need of humans in every aspect of machine learning as a result it becomes a knowledge and labor-intensive exercise to complete the process. Also, to completely replace humans in the process we require our machine models to completely mimic humans with an ability to continuously acquire knowledge, fine-tune it, and transfer knowledge throughout their lifespan. This arises a need where a machine learning model that should learn continuously from a change of data. As the data is changing (because of new data trends or actions made by users), our models should be retrained based on the new activity. So, the framework as depicted in figure 2, is presented to construct model optimized automated feature engineering task with periodic retraining of model by providing feedbacks. As a result, a machine learning model with high accuracy and high performance is build. This model extracts features automatically and also the model is adaptive to the new changing data.

The figure 2, illustrates similar to machine learning pipeline with additional feedback provided to retrain the model. Here, initially data is given as input to machine learning pipeline. Then as a part of work an automated feature engineering task is carried out to extract features automatically using candidate feature extraction approach. Further these features are given to model selector. There are two major components here: it picks up classifier and sets corresponding hyper parameter. The last and important step of machine learning pipeline is model training in which optimization is usually involved. For conventional learning models, optimization is not a problem as performance which is achieved through various optimization algorithm is almost the same. Once machine learning solution is achieved then the model is deployed.

The proposed work here adds the feedback based on the new data trends or new actions carried out by the users. This is an important task as it ensures if there are any anomalies or the data is corrupted then is should be detected which in turn ensures high performance and good accuracy.

So here the loop is closed: We have data, experiments are running the training, predictions are made and new changed data is used or retraining of the model. The last step of triggers for retraining model based on feedback can be carried out periodically. With these periodic updates the accuracy of model will also keep on improving.

Modified Architecture Overview

The architecture presented in figure 3, is composed of candidate feature generator, automated feature extraction, data splitter.

Step 1: Candidate Feature Generation

Given a dataset, a candidate feature generator is used which selects or rejects the attributes based on its usefulness measured with respect to target predictor variable. This can be achieved by calculating the ratio of missing column information and if this is beyond the threshold value (threshold value is defined by the data scientist) i.e., missing information is high then that column is not added to the candidate set. With this technique the candidate set of attributes are generated and then the deep feature synthesis is applied.

Assuming here y^j represents the number of useful candidate-features generated and z^j represents actual features available in the dataset. As candidate feature generator now generates only useful features as a result, y^j is less than z^j which in turn will also lead to a smaller number of recursive calls made by deep feature synthesis algorithm. This approach will lead to reduction in time and also useful features will be generated.

Step 2: Automated Feature Extraction

Once this is carried out the candidate features are forwarded to AutoML Feature Tools [1], which uses deep feature synthesis to generate large set of features but here it generates features on candidate set provided.

Step 3: Data Splitter

Once the useful features are generated then data splitter is used to split the data as training and testing data for training model using machine learning algorithms (here in study decision tree classification is used) with feature set on training data.

Decision Tree Classification Algorithm

Decision Tree [13], is a type of supervised learning algorithm that are used for classification as well as regression problems, but generally it is preferred for solving classification problems. This is a type of tree-structured classifier, where the internal nodes signify the features of a dataset, branches represent the decision rules and each leaf node represents the outcome of the decision made.

To divide the dataset, various splitting rules may be chosen. The most common approach is to use an entropy measure for calculating the information gain (IG) of the split part of dataset, such that,

$$IG(parent, children) = entropy(parent) - [p(c1) \times entropy(c1) + p(c2) \times entropy(c2) + \cdots]$$

It begins at the root node and then recursively divides the dataset in such a way that the IG is highest for each split. This approach is a type of greedy algorithm, as a local optimum is resolved at each split in a try to find the global optimum. It can also be observed that IG measures the difference between entropy of the parent and the weighted sum of the entropy of the children, where each child is denoted (c1) Entropy for each node is calculated as:

$$entropy = -p_1 \, log(p_1) - p_2 log(p_2)$$

While using Decision Trees, some hyperparameters are also specified. For instance, entropy is not the only way to measure, the Gini index is a common alternative. The maximum number of recursive partitions that are allowed is specified by maximum depth of the tree. The maximum branch specifies the maximum number of the branches that may be split in each node.

The algorithm selected is executed to identify patterns, drawing inferences, and building, and training model by using 80% dataset and remaining 20% of dataset is or validating model. The entire process is carried out while the model training is in progress. Once a model is trained, it provides various statistics that were computed during the training process in the model's evaluation report. The evaluation in machine learning pipeline provides percentage values for the following attributes of a classification model in the form of a confusion matrix:

- True Positive (TP): A true positive result is achieved where the model correctly predicts the positive class.
- True Negative (TN): A true negative result is achieved where the model correctly predicts the negative class.
- False Positive (FP): A false positive result is encountered where the model incorrectly predicts the positive class.
- False Negative (FN): A false negative result is encountered where the model incorrectly predicts the negative class.

The evaluation parameters that are used as a metrics for generating evaluation report are:

1. Accuracy

The accuracy is the fraction of total predictions made by the model on the test data that were correct, as a percentage value.

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Number\ of\ Total\ Predictions}$$

2. Precision

The precision is given as the fraction of total positive predictions made by the model that were correct on test data. This indicates how much correct is model's prediction.

$$Precision = \frac{TP}{(TP + FP)}$$

3. Recall

The recall is given by the fraction of the true positive predictions made by the model, considering out of all true positives and false negatives results. This parameter helps to select best model when there are highly associated false negative results. The recall is also termed as True Positive Rate.

$$Recall = \frac{TP}{(TP + FN)}$$

4. F1 Score

The F1 score is a useful metric for finding balance between precision and recall. It is given as harmonic mean of the precision and recall.

$$F1\ score = \frac{2\ x\ (Precision * Recall)}{(Precision + Recall)}$$

Step 4: Model Deployment

After this the model is deployed in production environment and new data trends are recorded. Further, at every periodic interval feedback integrator is used that will retrain the model continuously and automatically with the changed data.

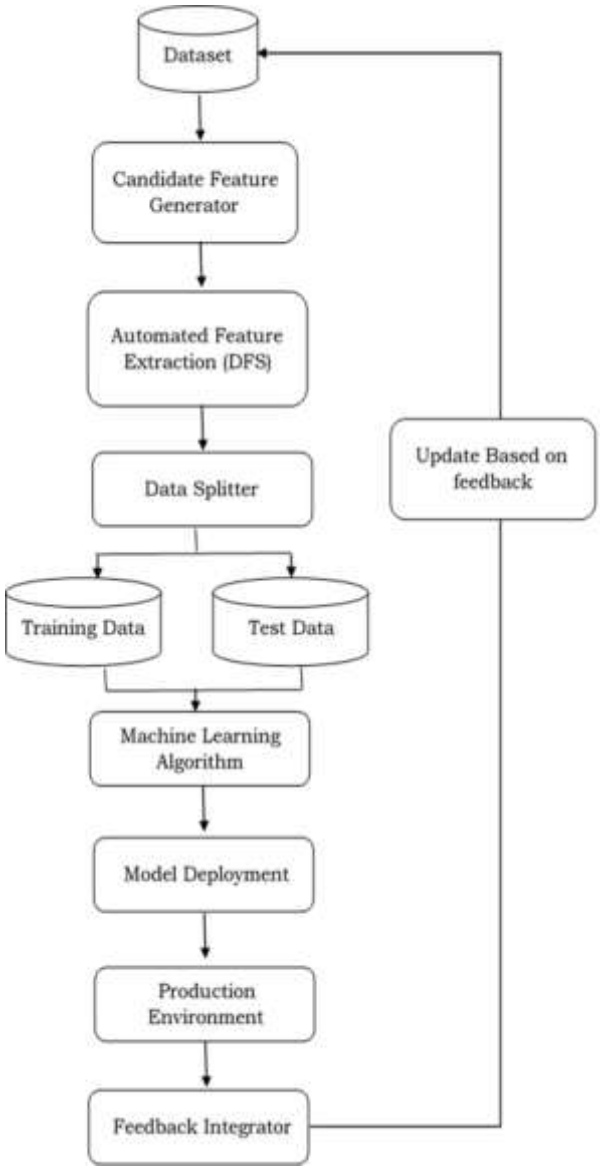

Figure 3: Framework for Automated Feature Engineering Task with Continuous Feedback

Step 5: Feedback and Integration

In today's world it is crucial to understand that data is no longer static instead it is changing. For this the model needs to be retrained again and display updates related to such topics in trending. This scenario can be applied to every search engine, hashtags which are trending on social media, recommendation systems used by e-commerce websites.

With a feedback loop the model is retrained as a result it is getting chance to go over what it already knows so it can keep learning from the changing data and perform better in the future, much like a studying student. This ensures the model is not stagnant. This adds an advantage that this new data used to train new versions of the model belongs to the same real-world distribution that the customer cares about predicting over. By integrating a feedback loop, it will reinforce models' training and keep them improvising over the period of time.

Consider an example for binary classification with Positive and Negative class. When the model is trained first, then the classification model will use labeled data which is been labeled using heuristic given by the domain experts, this in turn will give machine learning algorithm to learn the pattern based from positive/negative class. Once this model is developed, then the procedure for testing new and/or unlabeled data is started. The model will then predict as per the labels given and then based on a probability threshold feedback loop to correct or label the unlabeled data is created. The feedback loop will also have the way to correct the label based on the relaxation from domain experts so new or any unlabeled data can be auto labeled or shared with domain experts who then put their response in the training data. This closed feedback loop will then retrain the model with this new/updated labeled data.

The feedback loop is also important when the predictions made by the model affect the future labels, as the machine learning model is solving the problem the labeling logic may get changed as per the predictions and action carried out that changed the past behaviors. In such scenarios also feedback will help in updating the label and retraining the model.

Feedback to machine learning pipeline are important and knowing to provide feedback to the model will help to improve the model performance along with correcting labels.

IV. PERFORMANCE EVALUATION

The efficacy of the work is demonstrated in this section by performing the tasks as discussed in figure 3. The work was compiled with system configuration of 16GB RAM and i7 Processor 2.60GHz. The automated feature extraction is carried out using candidate feature generation approach. The extracted features are further used to the train the model using training dataset. After training the performance is measured using Decision Tree classification.

The Home Credit Risk dataset which is publicly available on Kaggle platform [12] is considered, processed and features are generated. A prediction model is built with feedback which indicates improvement in accuracy when data with new trends are recorded and model is retrained. The objective of Home Credit Default Risk Prediction is to predict if a customer will default on

loan based on complete data on past loan payments details. The credit default risk is given as the probability that loan is given and there is a chance that money will not be paid back on time. Also, such system helps to ensure that clients who are capable of paying loans are rejected'.

Automated Feature Engineering is carried out using featuretools library which uses deep feature synthesis to extract features. There are seven tables and here information is staggered across multiple tables these tables are referred as entity. To carry out DFS understanding of relationship is important. Example: application entity has one row for each client and bureau has multiple previous loans for each parent in application data. This represents one is-to-many relationship, as represented in figure 4. Further bureau is represented as a parent of bureau_balance because in this each loan of bureau is represented as multiple monthly records of bureau_balance.

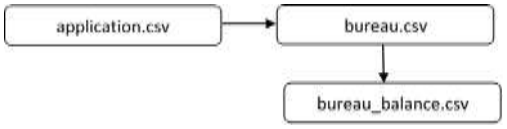

Figure 4: Relationship between entities

For each relationship parent and child variable are selected. After applying candidate feature generation approach using deep feature synthesis the algorithm generated 5438 features.

Predictions using Decision Tree Classification Results

The automatic features generated are further check with the target and important features are extracted. These important extracted features are further used to the train the model using training dataset. After training the performance is measured using Decision Tree classification. The classification accuracy of Decision Tree Classification is measured as a percentage value of correct predictions made by the model by the total number of predictions made. Performance of algorithm was evaluated at different stages of training set. The algorithm was trained with records sets containing 140000 records, 160000 records, 180000 records and 200000 records.

Table 1: Decision Tree Classification Results

Number of Records	Accuracy	Precision	Recall	F1 Score
140000	76.22%	0.73	0.77	0.75
160000	77.48%	0.74	0.80	0.77
180000	79.28%	0.76	0.82	0.79
200000	80.65%	0.78	0.82	0.80

The accuracy, precision, recall rate, F1 score at all stages are measured and represented in table 1 based on the confusion matrix generated during execution. The measured accuracy is depicted in figure 5.

Figure 5: Decision Tree Classification Accuracy Chart

Predictions using Decision Tree Classification Results with Feedback

The continuous training of model by automating ML pipeline helps to achieve continuous delivery of prediction service. The model is retrained periodically and prediction results are regenerated with the new data that is generated after deploying the model. This helps to improve the training of model with recent trends captured in the data which in turn helps to improve the accuracy of the model.

Table 2: Decision Tree Classification Results with Feedback

Number of Records	Accuracy	Precision	Recall	F1 Score
140000	78.04%	0.76	0.80	0.78
160000	80.11%	0.77	0.82	0.80
180000	82.18%	0.81	0.84	0.83
200000	83.16%	0.82	0.96	0.84

The comparative performance results based on data generated after deployment of model is depicted in figure 6. Which depicts that accuracy of model is improved with the recent data fed as the input for retraining of the model. The retraining of model with updated dataset with new trends recorded is carried out and accuracy at all mentioned stages is measured with new training dataset with 140000 customer records, 160000 customer records, 180000 customer records and 200000 customer records.

Figure 6: Decision Tree Classification Accuracy Chart with Feedback

The setting up continuous feedback in machine learning pipeline thus helps to enable the automatic retraining of the model. The system helps to handle the changes in the data and the business environment.

In a perfect world, the design of machine learning pipeline should be in such a way that it will use the data to label itself, providing the feedback in the pipeline to label the data makes the one step towards it. Also, automated feature engineering of AutoML helps to automate the time-consuming tasks of data scientists to generate features. Using candidate features more useful features are generated.

V. CONCLUSION

In today's world machine learning have extremely entrenched in our lives. To attain good performance there is the necessity of human in every aspect of machine learning pipeline to complete the process. However, this becomes tedious and so there comes a need of automating these tasks. As a part of satisfying aims, an automated feature engineering approach is presented along with continuous feedback is provided in the key architectural design of the machine learning pipeline. The work demonstrates the implementation of automated feature extraction and selection in detail. The number of useful features generated were 5438 features. However, the technique of generating candidate features first helped to generate features less in count but more useful features. But the result achieved are considerably better. Further the model was trained using decision tree classification algorithm on the training data set and the accuracy measured is 80.65%. As suggested architecture also focuses on providing continuous feedback after the model is deployed in production this helped to see improvement in the performance and accuracy measured after feeding new data is around 83.16%. The overall results showed significant improvement in the scope of automated feature generation as well as improvement in accuracy with continuous feedback provided to the model.

REFERENCES

[1] Kanter, J. M., and Veeramachaneni, K. (2015, October). Deep feature synthesis: Towards automating data science endeavors. In 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA) (pp. 1-10). IEEE.

[2] T. Mitchell, Machine Learning. Springer, 1997

[3] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot et al., "Mastering the game of Go with deepneural networks and tree search," Nature, vol. 529, no. 7587, p.484, 2016

[4] T. Mitchell, Machine Learning. Springer, 1997

[5] Udayan Khurana, Horst Samulowitz, and Deepak Turaga. Feature engineering for predictive modeling using reinforcement learning. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018

[6] Udayan Khurana, Fatemeh Nargesian, Horst Samulowitz, Elias Khalil, and Deepak Turaga (2016). Automating feature engineering. Transformation, 10(10): pp. 10.

[7] Lam, H. T.; Thiebaut, J.-M.; Sinn, M.; Chen, B.; Mai, T.; and Alkan, O. (2017). One Button Machine for Automating Feature Engineering in Relational Databases. arXiv preprint arXiv:1706.00327.

[8] Bagallo, G., and Haussler, D. (1990). Boolean feature discovery in empirical learning. Machine learning 5(1): pp. 71–99.

[9] Aboobyda Jafar Hamid and Tarig Mohammed Ahmed (2016). Developing Prediction Model of Loan Risk in Banks using Data Mining‖, Machine Learning and Applications: An International Journal (MLAIJ), Vol.3, No.1, pp. 1-9.

[10] Zhu L, Qiu D, Ergu D, Ying C and Liu K (2018). A study on predicting loan default based on the random forest algorithm. The 7th Int. Conf. on Information Technol. and Quantitative Management (ITQM) 162 pp. 503–13

[11] Ghatasheh N. (2014). Business analytics using random forest trees for credit risk

prediction: a comparison study Int. Journal of Advanced Science and Technol. 72, pp. 19–30

[12] Home Credit Default Risk Data Set: https://www.kaggle.com/

[13] Provost, F. & Fawcett, T. (2015). Data Science for Business. United States: O'Reilly Media

---

Corresponding Author

Vimla Jethani*

Research Scholar, Sunrise University, Alwar, Rajasthan, India