Title of Project: Heuristic selection for first-order theorem proving using machine learning

Student Name: Adamya Gaur

Enrollment Number: 04719011921

Signature: *Adamya*

Email ID: gauradamya725@gmail.com

Contact Number: +91 8920334715

Google Drive Link: https://drive.google.com/drive/folders/1ehJK6u-WQUtFtcs_g-7TYwDKTEIUQUn9?usp=sharing

Google Website Link: https://sites.google.com/ipu.ac.in/mlproject-adamya/home?authuser=2

YouTube Video Link: https://youtu.be/ySTCyADYjE4

Heuristic selection for first-order theorem proving using machine learning

Abstract

For proving a first order theorem, certain set of heuristics are required to prove the theorem easily. Selecting the right set of heuristics is a delicate task. The aim was to identify the most suitable heuristic for a given problem from a set of 5 heuristics. The system also has the capability to decline to prove the theorem. A model was trained based on 13 static and 38 dynamic features of the given dataset. The accuracy received at end of the training was quite up to the mark, as selecting the right heuristic is not easy even for experienced people.

Keywords

Heuristic, Machine learning, first-order, theorem proving, feature selection

Introduction

Theorem proving and first order logics play an important role in artificial intelligence, especially in the area of Natural language processing. The proof search for a given problem may be based on different algorithms. The results are affected by the parameters.

A standard setting of such parameters is thus used. This standard set is called heuristic value. The heuristic to be used will depend on the type of axioms and conjecture the problem has.

By selecting the right set of heuristics, one can prove the problem in an efficient manner. Even at present, human expert inputs may be required for selecting the right set of heuristics for a problem. In this project, 10 classification techniques were used on the given dataset. The data had static features. These are the features which are available once we have the problem. The other type of features, dynamic features are those which are available once the proof search has already begun. Out of all the applied classification techniques, gradient boosting classifier gave the best performance. Its further analysis was carried out further in this project.

Proposed Methodology

Ten different classifier techniques were employed on the dataset. These ten techniques are: Perceptron, Logistic Regression, Support Vector Classifier, DecisionTree classifier, K-Neighbors classifier, Gaussian Naïve Bayes classifier, AdaBoost classifier , Gradient Boosting classifier, RandomForest classifier & xgb classifier.

Out of these , the Gradient Boosting classifier gave the best output. The following picture shows the working of Gradient Boosting classifier.

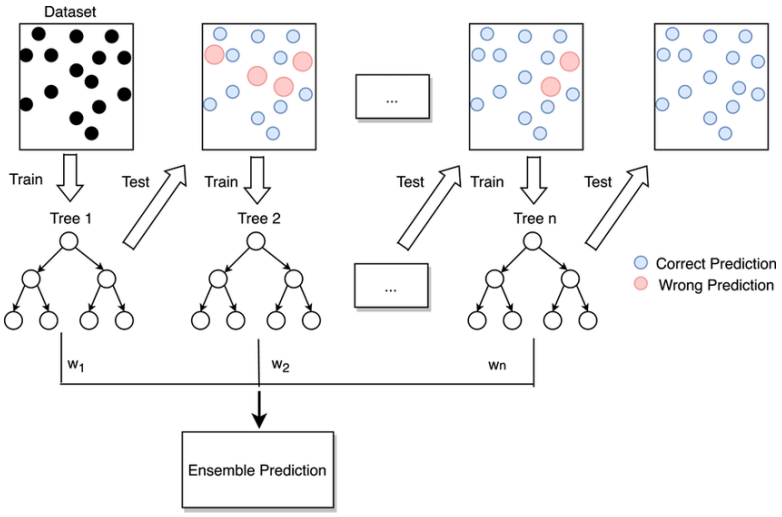


Fig 1:Working of Gradient Boosting classifier

1.Datasets

The dataset used was provided by University of Cambridge Irvine machine learning repository. The dataset has 6118 rows/ instances in total and 57 columns. In these 57 columns, 13 are for static features, 38 for dynamic features , 5 for heuristics and 1 for declining to prove(H0) on case no heuristic solves the problem within 100 seconds.

2.Pre processing

The data was pre processed. After loading of data, missing values were checked in the data set.

Checking for missing values

```
all_data.isna().sum()
```

Further data type of features were checked to ensure treatment of categorical values.

```
all_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6118 entries, 0 to 1529
Data columns (total 57 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Stat_F1      6118 non-null   float64
 1   Stat_F2      6118 non-null   float64
 2   Stat_F3      6118 non-null   float64
```

3.Exploratory data analysis

The exploratory data analysis (EDA) begun with the knowledge of central tendencies and data in inter quantile ranges. Standard deviation, mean, minimum as well as maximum values were also checked in the data.

```
all_data.describe()
```

	Stat_F1	Stat_F2	Stat_F3	Stat_F4	Stat_F5	\
count	6118.000000	6118.000000	6118.000000	6118.000000	6118.000000	
mean	0.040513	0.052955	-0.000470	0.050555	0.049217	
std	0.977642	0.980589	0.998545	0.979385	0.972786	
min	-1.105200	-3.735600	-0.984110	-1.065200	-1.240100	
25%	-0.748365	-0.229907	-0.605570	-0.694180	-0.721762	
50%	-0.372350	0.425360	-0.474750	-0.323190	-0.336315	
75%	0.742862	0.831520	0.256640	0.729960	0.562390	
max	2.009400	0.831520	2.738100	2.644800	2.366200	

Further, output values count was taken to check for imbalance in the dataset. On checking, an imbalance was found, where the 0^{th} heuristic had high number of instances available. The imbalance was reduced by replicating the instances of minority classes.

The below figures (fig2, fig3) show the imbalance before and after respectively.

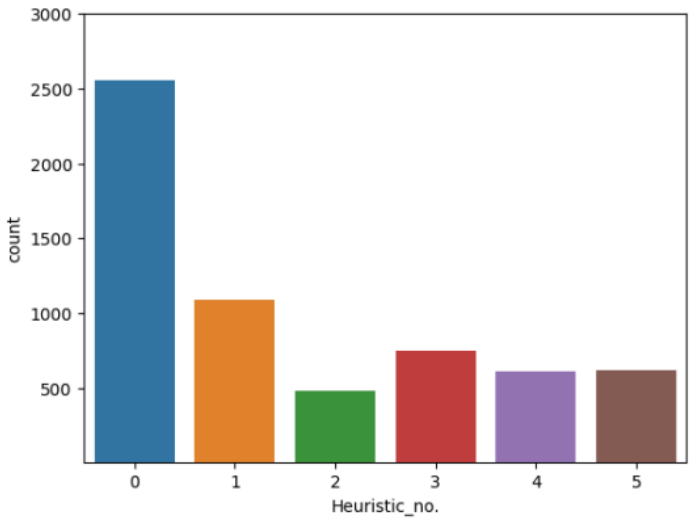
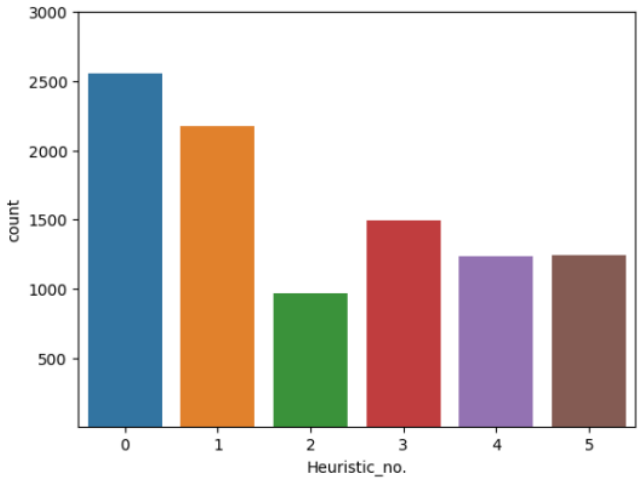


Fig 2



Fiig3: Reduced imbalance

At last, correlation among features were determined by plotting the heat map for the correlation matrix.

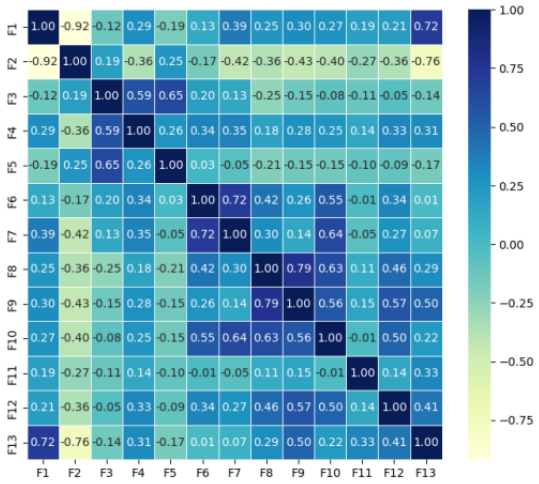


Fig4: Heatmap

4.Feature Selection

The number of features to be selected were calculated based on the log base2 (number of instances). This number turned out to be 13.

Feature Selection

Determining number of features to be selected

```
num_feature = round(math.log(all_data.shape[0],2),0)
num_feature
```

Therefore, 13 best features were selected, using f_classif as an estimator.

```
all_data = SelectKBest(f_classif, k=int(num_feature)).
 ↪fit_transform(full_train_data, test_data)
```

Hyperparameter tuning was also done via grid search to enhance the performance of XGB classifier.

Result & Discussion

The outputs of the classifiers were then plotted in terms of accuracy percentage. Out of these, gradient boosting classifier had the most accuracy, 87.8%. It was followed by Decision tree classifier (85.6%) and XGB classifier at75.9% accuracy.

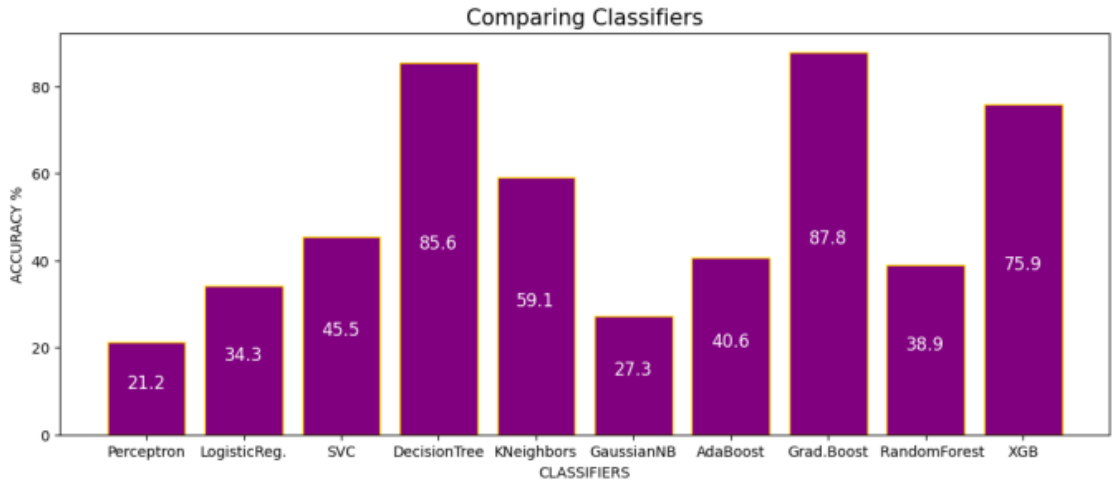


Fig 5: output

Further, the classification report for gradient boosting classifier was also created. This was turned to a heat map for better understanding of the values.

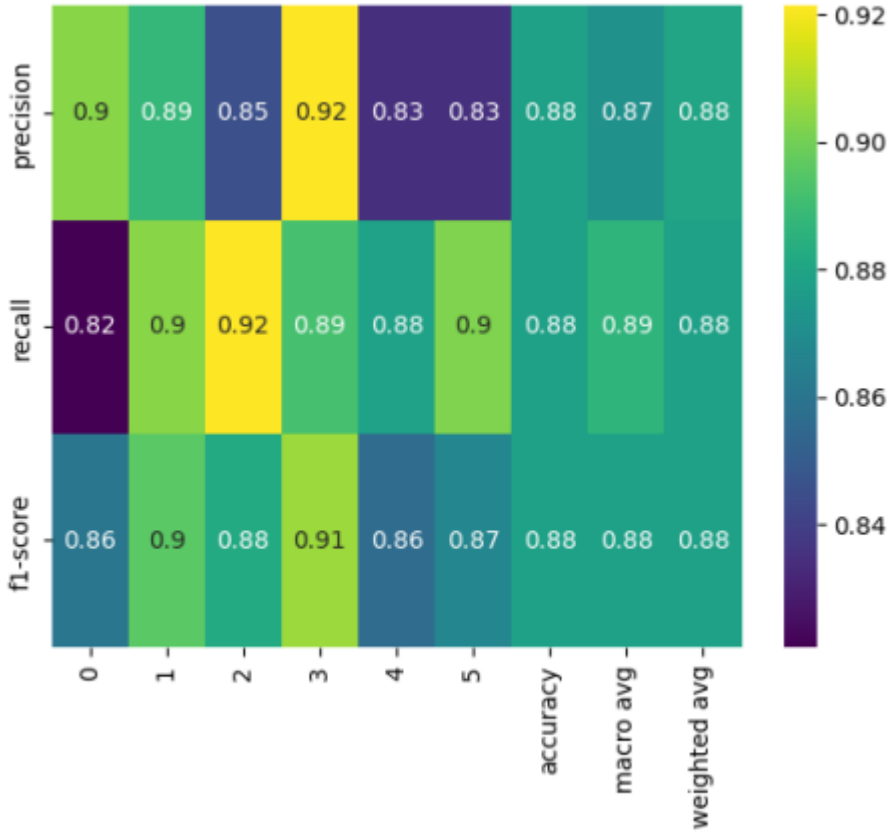


Fig 6: Classification report heat map

Conclusion & Future Work

The model gave satisfying accuracy, as it is difficult to choose the right heuristic even for professionals. However further accuracy may be increased by feeding the model with more data.

Extensive hyper parameter tuning of the classifiers could also increase the accuracy. Also shifting from machine learning to deep learning models may increase the accuracy.

Another point to be noted is that we had worked only for first order logic. Complex models may be used to evaluate and explore the higher order logics.

References

https://link.springer.com/article/10.1007/s10817-014-9301-5

https://stacks.stanford.edu/file/druid:yy382gv1477/yy382gv1477.pdf

https://books.google.com/books?hl=en&lr=&id=133kBwAAQBAJ&oi=fnd&pg=PR11&dq=first-order+theorem+proving&ots=seHOCUFSBu&sig=7dfEdprxEwT1_HzY4kGhdL3hAuk

https://arxiv.org/abs/2103.03798

https://link.springer.com/chapter/10.1007/BFb0012820

https://link.springer.com/chapter/10.1007/978-3-319-20615-8_5

https://dl.acm.org/doi/abs/10.5555/113902

https://books.google.com/books?hl=en&lr=&id=lsvSBQAAQBAJ&oi=fnd&pg=PP1&dq=automated+theorem+proving+first-order+logic&ots=4v0ynmR9Mk&sig=et6pwWHv-y-IzyVznwG5UaHePD4

https://arxiv.org/abs/2002.00423

https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wcs.1269

https://search.proquest.com/openview/b87467cab0987f591010cf19dc554fa3/1?pq-origsite=gscholar&cbl=18750

https://link.springer.com/content/pdf/10.1007/978-3-642-04617-9.pdf#page=301

https://link.springer.com/content/pdf/10.1007/978-3-540-75560-9.pdf#page=162

https://www.researchgate.net/figure/Flow-diagram-of-gradient-boosting-machine-learning-method-The-ensemble-classifiers_fig1_351542039

https://scikit-learn.org/