Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2021) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Industry Sciences and Computer Sciences Innovation

Comparative Analysis of Machine Learning Algorithms for Analyzing NASA Kepler Mission Data

Varad Vishwarupe[a]*, Mangesh Bedekar[b], Milind Pande[b], Prachi Joshi[c], Saniya Zahoor[d], Priyanka Kuklani[e]

[a]*Amazon Development Centre, India.*
[b]*Maharashtra Institute of Technology WPU, India*
[c]*Vishwakarma Institute of Information Technology, India*
[d]*National Institute of Technology, Srinagar, India*
[e]*Northeastern University, Boston, United States*

Abstract

The quest for learning more about the minute intricacies of the universe has fascinated mankind since ages. From stars to planets and asteroids to exoplanets, each research endeavor in astronomy has supplemented our knowledge about the cosmos. NASA's Kepler Mission is one monumental step in this direction wherein, telescopes conduct a survey of the Milky Way galaxy and try to identify thousands of earth-size and other smaller planets in or near the habitable zone, so as to determine the thousands or even millions of stars in our galaxy that might have such orbiting planets. Exoplanet is any new planet outside the solar system which orbits a star. Identifying new exoplanets gives us a chance to precisely understand the planet formation processes. Earlier, it was a laborious endeavour to mine the mission data using traditional algorithms and churn out the possible exoplanets. Ever since the advent of various machine learning algorithms, this process has become quite seamless. However, not all algorithms give equal and promising results when it comes to analysing different types of data. A comparative study of algorithms helps in this regard, thereby identifying the pros and cons of different algorithms for analysing certain forms of data. In this paper, we initially focus on feature set reduction using principal component analysis and thereafter make a detailed comparative analysis of ML algorithms for the identification of exoplanets, in the NASA Kepler mission data.

Keywords: NASA; Machine Learning; Artificial Intelligence; NASA Datasets; NASA Machine Learning; Algorithms; Exoplanets; NASA Data.

*Corresponding author: varad44@gmail.com
*ORCID ID: www.orcid.org/0000-0002-0372-814X

1. Introduction

It has always been man's endeavor to discover hidden insights into the mysteries of the universe. Knowing how the universe formed has been at the epicenter of this quest of discovery. While many theories exist on the origins of the universe, knowing more about the formation of planets and how the universe developed can help us broaden our horizons of learning about the universe. In this regard, exoplanets play an important role in improving our knowledge and understanding of the universe. The NASA Kepler mission has been one monumental step in this direction, wherein the Kepler telescope surveyed obsolete parts of the Milky Way galaxy and analyzed certain exoplanet candidates, which showed customary signs of being classified as a exoplanet. For the scope of this paper, we analyze the Kepler Mission dataset which we obtained through the California Institute of Technology's Jet Propulsion Laboratory mission repository website using NASA missions API(Application Programming Interface).

2. Related work

Analysis of NASA mission datasets has been an exciting endeavor for data scientists and machine learning enthusiasts, especially since the advent of data aggregation platforms. Studies have been conducted in this realm wherein researchers have tried to make sense of mission data for analyzing and predicting certain anomalies in data and chalk out important patterns in them so as to identify mission critical parameters.[1][2][3][4] These studies include the analysis of the Columbia Space Shuttle Disaster, analyzing the images captured by the Cassini and Juno probes for identification of possible signs of life or biodiversity. Statistical machine learning toolkits have been developed to make data analysis of space missions easier. [5][6][7][8] Efforts have also been made to detect planets orbiting neutron stars and quasar star detection using data mining techniques. When it comes to the detection and prediction of possible exoplanets, mainly, Support Vector Machines (SVM), Naïve Bayes (NB) and Deep learning algorithms such as Convolutional Neural Networks have been previously used.[9][10][11] Deep learning approaches have especially proved useful in determining the habitability of exoplanets. However, the black box nature of these approaches have often resulted in the models being prone to the lack of interpretability.[12][13] While there have been significant leaps into the exploration of machine learning as a tool for analyzing space mission data, a comparative study of machine learning algorithms would provide a lot of insights into the rationale behind using a certain classifier for this specialized purpose of exoplanet identification, classification and prediction.[14]-[21]

3. Research methodology and implementation

The dataset that we sourced was from Caltech's Jet Propulsion Laboratory repository through the NASA API functionality, and consisted of a total of 8,832 data elements with a sizeable 41 features. It would have been an arduous and time-consuming task to use traditional data mining techniques to churn the datasets for analysis. Hence, we used NASA and JPL's planetary terminology index and labelled the features in a way that made them more understandable and coherent, such as for the epoch time and transit radius. In all, the dataset contains a total of 8,832 data entries or observations along with 41 feature vectors, that help us identify if a planet is possibly an exoplanet or not. We made a train and test split of 65-35% respectively for analyzing the data. 5710 samples were used for training the machine learning models and 3122 were used for testing and validating our models. We used the same training and testing samples for all four classification algorithms to preserve uniformity in data analysis and to facilitate a holistic comparative study. Based on previous studies related to exoplanet detection and exoplanet possible candidate classification, we eradicated some irrelevant features such as insolation flux, transit signal-to - noise from the dataset which don't influence the classification task measurably. This resulted into a reduction of 11 features in all, using Principal Component Analysis (PCA) for dimensionality reduction. We used a data pipeline for this purpose using normalization for different quantities and then fed the reduced data as input to our classifiers. We used repeated stratified cross-validation for the model encompassing five folds and three repeats. In this manner, we were able to focus the core of our analysis on 30 features such as planetary radius, transit radius, transit time, epoch time, stellar effective temperature and stellar effective gravity, which had the most distinguishing characteristics when it came to exoplanet detection and prediction. The libraries consisted of Pandas, Scikit and Matplotlib were the libraries that we used

```
[ ]  # Logistic Regression Model
     lr = LogisticRegression(C=100, max_iter=200, class_weight='balanced')

     # Fitting Model to the train set
     lr.fit(X_train, y_train)

     # Predicting on the test set
     y_pred = lr.predict(X_test)

     # Evaluating model
     evaluation(y_test, y_pred)
```

```
Evaluation Metrics:
Accuracy: 0.8081358103779629
Recall: 0.8673218673218673
F1 Score: 0.8250073035349109
Precision: 0.7866295264623956

Confusion Matrix:
 TN,  FP, FN, TP
[1111  383  216 1412]
```

Fig. 1 Logistic Regression Model

```
knn = KNeighborsClassifier(leaf_size=8, metric='manhattan',weights='uniform')

# Fitting Model to the train set
knn.fit(X_train, y_train)

# Predicting on the test set
y_pred = knn.predict(X_test)

# Evaluating model
evaluation(y_test, y_pred)
```

```
Evaluation Metrics:
Accuracy: 0.8014093529788597
Recall: 0.8415233415233415
F1 Score: 0.8154761904761906
Precision: 0.7909930715935335

Confusion Matrix:
 TN,  FP, FN, TP
[1132  362  258 1370]
```

Fig. 2 k-Nearest Neighbor Model

```
tree = DecisionTreeClassifier()

# Fitting Model to the train set
tree.fit(X_train, y_train)

# Predicting on the test set
y_pred = tree.predict(X_test)

# Evaluating model
evaluation(y_test, y_pred)
```

```
Evaluation Metrics:
Accuracy: 0.9349775784753364
Recall: 0.9373464373464373
F1 Score: 0.9376344086021506
Precision: 0.9379225568531039

Confusion Matrix:
 TN,  FP, FN, TP
[1393  101  102 1526]
```

Fig. 3 Decision Tree Model

```
# Instantiate model
forest = RandomForestClassifier(n_estimators=100, criterion='gini')
# Fitting Model to the train set
forest.fit(X_train, y_train)
# Predicting on the test set
y_pred = forest.predict(X_test)

# Evaluating model
evaluation(y_test, y_pred)
```

```
Evaluation Metrics:
Accuracy: 0.9618834080717489
Recall: 0.9465601965601965
F1 Score: 0.9628241174632927
Precision: 0.9796567069294342

Confusion Matrix:
 TN,  FP, FN, TP
[1462   32   87 1541]
```

Fig. 4 Random Forest Model

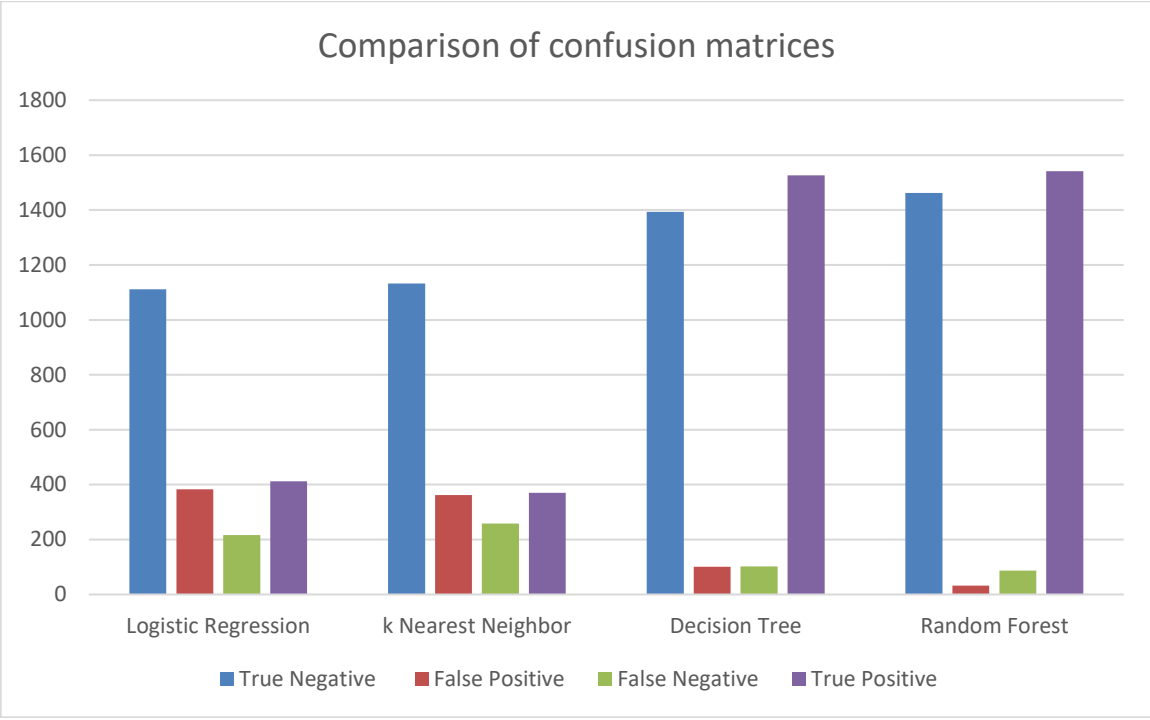

Fig. 5. Comparative Analysis of Confusion Matrices for all four algorithms

4. Comparative Analysis and Result Interpretation

Table 1 Comparative result interpretation

Algorithm	**Accuracy (%)**	**Precision**	**Recall**	**F1 Score**	**Advantages pertaining to this case study**	**Disadvantages pertaining to this case study**
Logistic Regression	80.81	0.786	0.867	0.825	Unknown data records were easily identified. Easiest of all to implement, train and test.	High overfitting observed. Problems with continuous variables Eg. Velocity.
k-Nearest Neighbor	80.14	0.790	0.841	0.815	Performed very well for new data in the test split. Less training time was required out of all.	Heavy normalization was required for parameters. Outliers in data such as orbital period and velocity were difficult to handle.
Decision Tree	93.49	0.937	0.937	0.937	Training time was least out of all the algorithms. Epoch Time and transit time non-linear variables were handled the best.	Noise impacted the model performance significantly. A large amount of variance could be observed due to zero bias.
Random Forest	96.18	0.979	0.946	0.962	Highest stability amongst all the algorithms. Effect of noise was the least out of all.	Required the longest time to train, build, test and execute. Limited by the number of computational resources

As seen in Fig. 1 to Fig. 4, we started our analysis using four different classifiers viz. logistic regression, k-NN, decision tree and random forest. This was carried out after transforming the feature vectors using dimensionality reduction using a PCA pipeline and PCA transform methods in Python. Logistic regression resulted in the second lowest accuracy percentage, just marginally over k-NN owing to the fact that there were a lot of variables involved in the data, and also owing to the size of the dataset contributing to overfitting in the model. When it came to some unknown data records especially in the data fields related to equilibrium temperature, planetary radius and so on, logistic regression performed the best. For variables involving continuous data, logistic regression did not fare very well and the sheer size of the features resulted in overfitting. Secondly, we used k-nearest neighbor which is a parametric and supervised learning algorithm resulted in the lowest accuracy percentage out of all the four algorithms. This was attributed to the fact that, all 30 features after dimensionality reduction comprised of different units. It also suffered while handling parameters like orbital velocity and time period, which result in a lot of true negatives as seen in Fig. 5. However, k-NN required the least computational and training time as inherently it is well known for its property of instance-based learning, by making predictions for classification on-the-go. Also, when it came to new data in the test split, k-NN showed a lot of robustness. Still, it performed with the least accuracy attributing to the heterogenous nature and humongous scale of the dataset. Thirdly, when it came to the implementation of decision tree, second least number of false positives and false negatives were observed, as seen from Fig. 5. One of the greatest advantages of using decision tree was the less amount of training as well as computation time for testing. The model executed itself over the train test split, 47% faster and also used the least computational resources. However, the prime deterrent in the usage of the decision tree algorithm was the observation of a large amount of variance, attributed to the model's inclination for achieving zero bias for combating overfitting. Noise found in the training phase also impacted the model significantly and resulted in a measurable level of false positives and false negatives, which were a tad bit higher than random forest. Although, the number of true negatives and true positives were the highest for this algorithm, presence of noise and variance degraded the overall accuracy to 93.49%. Eventually, when it came to the implementation of the random forest model, the effect of noise and variance was found to be the least. Out of the four models presented, random forest displayed the highest amount of stability. We believe that this is attributed to the selection of the highest number of correct binary tree classes and also to the fact that random forest is in itself an ensemble, which can handle data having a plethora of parameters, as was the case with this dataset. On the contrary, the build time and runtime was the highest for random forest, 98 mins. to be precise, which is an indication of the model's computationally intensive nature. Overall, it was still the best performing model with an accuracy percentage of 96.18% and precision, recall as well as F-1 score values of 0.979, 0.946, 0.962 respectively. What distinguished random forest from the rest was the least number of false positives and highest number of true positives as seen in Fig. 5. Thus, the probability of a particular planet candidate being an exoplanet or not, can be computed using machine learning algorithms as summarized in Table 1. All the 30 feature vectors which were used under the purview of this study were actual and real time data snippets captured by the NASA Kepler Mission probe and thus had to be normalized for the sake of this study. All in all, we were able to successfully analyze the data for predicting possible exoplanets with considerable accuracy.

5. Conclusion and Future Work

After the completion of all four model rundowns, we were able to conclude that random forest is the go-to algorithm for analyzing complex and multi-variable datasets, such as the one under the context of this study. While decision tree performed almost equally well, the presence of variance and noise for analyzing mission critical data becomes a deterrent to its usage. Logistic regression should be avoided for such datasets altogether as it suffers from a lot of over-fitting. Likewise for k-NN, the highly entropic nature of data having a lot of outliers, pertaining to space missions becomes a bottleneck in using it as the preferred classification algorithm. Hence, random forest seems to be the apt choice for such data. All in all, our study was able to come up with some key findings related to the use of classification algorithms, which would help future researchers in carrying out similar case studies. Future work in this realm would include comparing traditional classification algorithms stack up against deep learning algorithms, and assessing the outcomes for gaining a more detailed and holistic viewpoint of this exciting fusion of astronomy and machine learning. We believe that this amalgam of space science and machine learning has endless research potential and we would endeavor to explore it more minutely and meticulously in the foreseeable future.

References

[1] W. Ricker, et al., Transiting Exoplanet Survey Satellite", Journal of Astronomical Telescopes Instruments and Systems, pp. 014003, 2014.

[2] R.K. Kopparapu, et al. Habitable zones around main-sequence stars: new estimates. Astrophys. J. 765(2), 131, 2013.

[3] Christopher J. Shallue and Andrew Vanderburg, Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90, 2018.

[4] S. Saha, S. Agrawal, R. Manikandan, K. Bora, S. Routh and A. Narasimhamurthy, ASTROMLSKIT: A New Statistical Machine Learning Toolkit: A Platform for Data Analytics in Astronomy, April 2015.

[5] M.R. Smith and T. Martinez, "Improving classification accuracy by identifying and removing instances that should be misclassified", The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 2690-2697, 2011.

[6] N.V. Chawla, K.W. Bowyer, L.O. Hall and W.P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique", J. Artif. Int. Res., vol. 16, pp. 321-357, 2002.

[7] Bloom J. et al., "Data mining and machine-learning in time-domain discovery & classification", Adv. Mach. Learn. Data Min. Astron., 2011.

[8] Saniya Zahoor, Mangesh Bedekar, Vinod Mane, Varad Vishwarupe (2016), Uniqueness in User Behavior While Using the Web. In: Satapathy, S., Bhatt Y., Joshi A., Mishra D. (eds) Proceedings of the International Congress on Information and Communication Technology. Advances in Intelligent Systems and Computing, vol 438. Springer, Singapore. https://doi.org/10.1007/978-981-10-0767-5_24

[9] V. Vishwarupe, M. Bedekar and S. Zahoor, "Zone specific weather monitoring system using crowdsourcing and telecom infrastructure," 2015 International Conference on Information Processing (ICIP), 2015, pp. 823-827, doi: 10.1109/INFOP.2015.7489495.

[9] William J. Borucki et al., "Kepler planet-detection mission: introduction and first results", Science, vol. 327, no. 5968, pp. 977-980, 2010.

[10] Shallue and Vanderburg, "Identifying Exoplanets with Deep Learning", The Astronomical Journal, vol. 155, pp. 94, 2017.

[11] N. Peng, Y. Zhang and Y. Zhao, "A SVM-kNN method for quasar-star classification", Science China Physics Mechanics and Astronomy, vol. 56, pp. 1227-1234, 2013.

[12] A. Wolszczan, "Searches for planets around neutron stars", Celest. Mech. Dyn. Astr., vol. 68, pp. 13, 1997.

[13] Vishwarupe V., Bedekar M., Pande M., Hiwale A. (2018) Intelligent Twitter Spam Detection: A Hybrid Approach. In: Yang XS., Nagar A., Joshi A. (eds) Smart Trends in Systems, Security and Sustainability. Lecture Notes in Networks and Systems, vol 18. Springer, Singapore. https://doi.org/10.1007/978-981-10-6916-1_17

[14] Brychan Manry, George Sturrock and Sohail Rafiqi, Machine Learning Pipeline for Exoplanet Classification, 2019.

[15] V. V. Vishwarupe and P. M. Joshi, "Intellert: a novel approach for content-priority based message filtering," 2016 IEEE Bombay Section Symposium (IBSS), 2016, pp. 1-6, doi: 10.1109/IBSS.2016.7940206.

[16] Schulze-Makuch, A. Méndez, A.G. Fairén, P. von Paris, C. Turse, G. Boyer, et al., A Two-Tiered Approach to Assess the Habitability of Exoplanet, 2011.

[17] E.A. Petigura, A.W. Howard and G.W. Marcy, "Prevalence of Earth-size planets orbiting Sun-like stars", Proceedings of the National Academy of Sciences of the United States of America, October 2013.

[18] I.N. C. Santos, "Extra-solar planets: Detection methods and results", New Astronomy Reviews, vol. 52, no. 2, pp. 154-166, 2008.

[19] Bedekar M., Zahoor S., Vishwarupe V. (2016) PeTelCoDS—Personalized Television Content Delivery System: A Leap into the Set-Top Box Revolution. In: Satapathy S., Das S. (eds) Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 2, SIST, Springer. https://doi.org/10.1007/978-3-319-30927-9_27.

[20] Zahoor S., Bedekar M., Vishwarupe V. (2016) A Framework to Infer Webpage Relevancy for a User. In: Satapathy S., Das S. (eds) Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 1. Smart Innovation, Systems and Technologies, vol 50. Springer, Cham. https://doi.org/10.1007/978-3-319-30933-0_16

[21] M. Johnson, in How Many Exoplanets Has Kepler Discovered? (2015). URL: https://www.nasa.gov/kepler/discoveries