



iJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** X **Month of publication:** October 2026

DOI: <https://doi.org/10.22214/ijraset.2026.85212>

www.ijraset.com

Call: ☎ 08813907089

E-mail ID: ijraset@gmail.com

Multicare Disease Predictor: An Intelligent Symptom Analysis System Using Random Forest with Evidence-Backed Results

Jyothi K, Chinmai J, Inchara R, Chethan R, Ms. Harshitha M

Department of Computer Science and Engineering Mysuru Royal Institute of Technology (MRIT) Mandya, Karnataka, India

Abstract: Many people look up their symptoms online before they see a clinician, but the tools they find seldom show how trustworthy a suggestion is. Multicare Disease Predictor is a Flask-based web application that ranks likely conditions from symptoms chosen by the user. A Random Forest classifier is trained on 52,718 cleaned symptom records covering 50 conditions, using a 377-element binary symptom vector. Duplicate and contradictory records were removed before training, and a leaf-size constraint ($\text{min_samples_leaf} = 5$) was introduced after a one-off symptom pattern produced an over-confident answer. On a stratified hold-out set of 10,544 records the model reached 96.13% accuracy, 0.959 macro-F1 and 99.89% top-3 accuracy. Re-testing on randomly thinned inputs showed that top-1 accuracy drops to 80.7% with three symptoms and 32.0% with one, which motivates the application's three-symptom minimum. Every result is shown with training-data evidence for the symptoms entered, plain-language guidance, a downloadable PDF summary and a nearby-hospital locator. Because the training data are synthetic and not clinically validated, the system is offered strictly as an educational aid and not as a diagnostic tool.

Keywords - symptom checker, disease prediction, Random Forest, overfitting control, partial-input analysis, healthcare web application, Flask.

I. INTRODUCTION

Searching the internet for the meaning of a new symptom has become a routine first step for many people, often well before they speak to a clinician. Online symptom checkers formalise this habit by turning a list of complaints into a short list of candidate conditions, yet audits of such tools show uneven quality. In a study of 23 public checkers the correct diagnosis appeared first in only about one third of standardised cases [1], and a later comparison of eight popular apps found large differences in coverage, accuracy and the safety of urgency advice [2]. Systematic reviews describe the same variability [3].

Supervised machine learning offers a data-driven route to such tools: a classifier is fitted to labelled symptom records and asked for a probability for every candidate condition. Two practical problems are easy to overlook. First, a model that is accurate on complete records may behave poorly when a user types only a handful of complaints. Second, a flexible model can memorise an unusual symptom pattern and then return a confident answer for it.

Multicare Disease Predictor tackles both problems inside a working web application. A Random Forest [4] trained on 52,718 cleaned records ranks 50 conditions; the interface refuses to predict from fewer than three symptoms; the forest is regularised with a minimum leaf size; and every answer is accompanied by evidence from the training data, plain-language guidance, a PDF summary and a locator for nearby care. This paper re-runs the complete pipeline from the shipped data, compares alternative classifiers, and measures how accuracy changes with the amount of symptom information supplied.

A. Objectives

- Build a 50-condition symptom-based predictor on data cleaned of duplicate and contradictory records.
- Limit over-specific tree behaviour with a minimum leaf size and quantify its effect on prediction confidence.
- Measure accuracy under partial symptom input and use the result to justify a minimum-input rule.
- Present each prediction together with training-data evidence rather than a bare label.
- Link a prediction to a specialist, an urgency flag and nearby healthcare facilities.
- Keep user accounts and usage logs so that an administrator can review how the tool is used.

II. METHODOLOGY

The work has two stages. In the first, an offline script prepares the data and fits the classifier. In the second, the web application loads the stored model and, for each request, encodes the user's symptoms, obtains class probabilities and formats the response (Fig. 1).

A. Data Source and Cleaning

The source is a public symptom-condition table with 246,945 records, 773 conditions and 377 binary symptom indicators [12]. Its publisher states that the table was artificially generated to preserve symptom severity and condition occurrence, so the records are not real patient encounters. We kept the 50 most frequent conditions (56,599 records; 910 to 1,219 per condition). Two cleaning steps followed. Exact duplicate rows (about 1,200) were dropped. Rows whose symptom vector was shared by records carrying different labels (about 2,700 rows) were then removed, because no classifier can resolve identical inputs with contradictory targets and such rows only distort the error estimate. The final table holds 52,718 records, between 651 and 1,215 per condition, with 6.05 active symptoms per record on average (range 1 to 12). The input vector keeps 377 positions, but only 184 of them are active in at least one retained record; the other 193 are constant zero in the retained data.

B. Symptom Encoding

Each record is expressed as a binary vector in a fixed order: 1 if the symptom was reported, 0 otherwise. The ordered feature list is stored beside the model, so the vector built at serving time is identical in layout to the training matrix.

C. Classifier and Evaluation Protocol

The classifier is a scikit-learn [5] Random Forest with 100 trees, $\sqrt{p}$ candidate features per split and a fixed random seed. A stratified 80/20 split gives 42,174 training and 10,544 test records with the same condition proportions in both parts. We report accuracy, macro-averaged precision, recall and F1 [8], and top-k accuracy, since the application shows a ranked list. The leaf-size value was chosen from the behaviour described next and not by a grid search, so its small accuracy gain is indicative only.

D. Leaf-Size Regularisation

A tree grown without constraints can isolate a single training record in its own leaf. During testing, a symptom combination that occurred only once in the training data produced a high-confidence answer for the label of that one record. Requiring at least five samples in every leaf [4], [6] makes each leaf summarise several records. On the test set this lowered the mean top-class probability from 92.0% to 81.5%, while accuracy rose from 96.04% to 96.13%.

E. Ranking and Display Rule

The forest's class probabilities are sorted and the top five kept. Entries below 3% are hidden unless that would leave nothing to show, so the list is not padded with unlikely guesses. These values are vote fractions of the forest and not calibrated clinical risks [7]; the interface labels them as match strength.

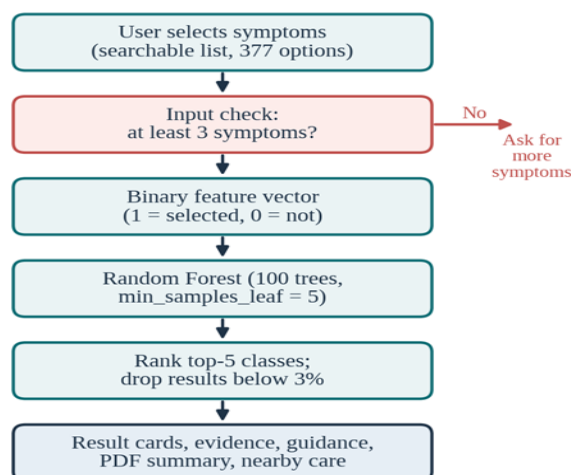


Fig. 1. End-to-end prediction pipeline of Multicare

III. MODELING AND ANALYSIS

The functional modules of the implemented system are listed in Table I.

TABLE I
FUNCTIONAL MODULES OF THE SYSTEM

No.	Component	Functional description
1	Symptom input	Searchable list of 377 symptoms; choices appear as removable chips.
2	Input validation	Rejects requests with fewer than three recognised symptoms (client and server).
3	Feature encoder	Builds the fixed-order 377-element binary vector.
4	Classifier	Random Forest returning probabilities for 50 conditions.
5	Result filter	Keeps the top five; hides entries below 3%.
6	Evidence panel	For the user's symptoms, shows the share of training patients with that condition who had each symptom.
7	Condition guide	Description, self-care steps, specialist and an urgent-care flag (12 of 50 conditions).
8	Report generator	PDF summary of the top three results with guidance.
9	Care locator	Nearby facilities from device location: live places API if configured, offline directory otherwise.
10	Accounts and logs	Hashed passwords, sessions, prediction logging, admin dashboard.

IV. PROBLEM STATEMENT

Symptom data are noisy, overlapping and incomplete. Many conditions share complaints, users rarely know which symptoms matter, and public records can contain duplicate or contradictory rows. A usable system must (i) clean such data before learning, (ii) avoid over-confident answers when evidence is thin, (iii) tell users how much input is enough, and (iv) present results as information to discuss with a clinician, with a pointer to the right specialist and to nearby care.

V. PROPOSED SYSTEM

Multicare is organised in four layers (Fig. 2). The browser layer handles symptom search and display; the Flask layer [9] handles routing, sessions, validation and report generation; the intelligence layer holds the stored model, label encoder, prevalence table and condition guide; and the data layer holds SQLite [10] tables and the hospital directory.

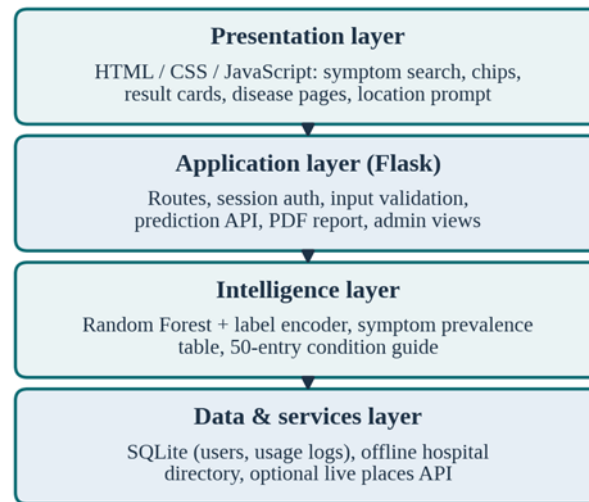


Fig. 2. Layered architecture of Multicare.

A. Prediction Service

A request carries the selected symptom identifiers. Unknown identifiers are discarded, and if fewer than three valid symptoms remain the service answers with an error asking for more input. Otherwise the symptom vector is built, probabilities are computed and the filtered ranking is returned with each condition's urgency flag. In 50 in-process requests the endpoint took 14 ms on average (95th percentile 20 ms), excluding network time.

B. Explaining a Result

For every listed condition the system reports how common each of the user's symptoms was among training patients with that condition, and shows the three most common. This is descriptive evidence from the data, not a feature-importance explanation of the forest, but it lets a user see why a condition was suggested and notice when only one of their symptoms supports it.

C. Guidance, Urgency and Care Locator

Each of the 50 conditions has a plain-language description, general self-care steps without dosages, the specialist to consult and, for 12 conditions, a warning that urgent care may be needed. The locator takes the device position and the recommended specialist. When a live places key is configured it searches within 5, 15 and then 30 km until facilities are found; otherwise it falls back to an offline directory of 50 major hospitals in 32 states and union territories, ordered by great-circle distance. Each entry links to a map.

D. Accounts and Logging

Registration and login use salted password hashes; prediction pages require a session. Each prediction stores the user, symptoms, top result, confidence and time, and an administrator-only page lists users and recent activity.

VI. EXISTING SYSTEM

Existing approaches show the following limitations:

- Public symptom checkers differ widely in diagnostic accuracy and in the safety of their urgency advice [1]–[3].
- Many tools show only a ranked label, without indicating which of the user's inputs supports it.
- Accuracy is often quoted for complete records only, which says little about a user who enters two or three complaints.
- Few tools connect a prediction to the next step: the specialist, the urgency and a nearby facility.

Multicare responds with cleaned training data, a leaf-size constraint, an explicit input-sufficiency rule, visible evidence and a care locator.

VII. RESULTS AND DISCUSSION

A. Experimental Setup

All figures were obtained by re-running the pipeline on the shipped dataset with Python and scikit-learn 1.8.0, using the 80/20 stratified split (seed 42) described above. For the selected configuration, five-fold stratified cross-validation gave $95.90\% \pm 0.15$ (mean $\pm$ standard deviation), consistent with the hold-out result.

B. Classifier Comparison

Table II compares the selected forest with five alternatives trained on the same split. Training times are indicative and depend on the machine (Fig. 3).

TABLE II
COMPARISON OF CLASSIFIERS ON THE 10,544-RECORD TEST SET

Model	Acc. (%)	Macro P	Macro R	Macro F1	Time (s)
Decision Tree	94.97	0.947	0.947	0.947	1.2
Random Forest (default)	96.04	0.958	0.959	0.958	10.7
Random Forest (leaf = 5)	96.13	0.959	0.959	0.959	8.7
Bernoulli Naive Bayes	96.00	0.960	0.956	0.957	4.3
Linear SVM	95.98	0.959	0.958	0.957	3.4
Logistic Regression	96.09	0.959	0.959	0.958	6.6

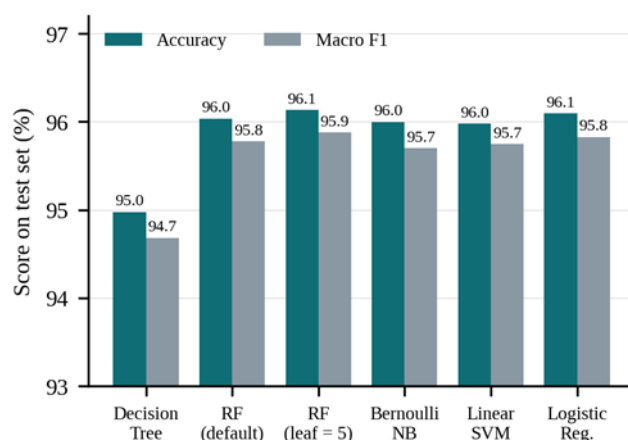


Fig. 3. Accuracy and macro-F1 of the compared classifiers.

Five of the six models lie within 0.15 percentage points of one another, which equals the cross-validation standard deviation, so their ordering should not be over-interpreted; only the single decision tree is clearly weaker. The forest was kept because its leaf-size parameter gives direct control over over-confident outputs, it captures interactions between symptoms, and its vote-based probabilities suit a ranked display.

C. Performance of the Selected Model

The selected model reaches 96.13% accuracy, macro precision, recall and F1 of 0.959 each, weighted F1 of 0.961, top-3 accuracy of 99.89% and top-5 accuracy of 99.99%. Training accuracy is 97.9%, a gap of 1.7 points, so overfitting is modest. Confidence is informative: the mean top probability is 83.1% for correct predictions and 41.0% for wrong ones.

D. Per-Condition Behaviour

Fifteen of the 50 conditions have $F1 \geq 0.99$ and 37 have $F1 \geq 0.95$, while seven fall below 0.90 (Fig. 4). The most frequent confusions involve conditions that share symptoms: non-infectious gastroenteritis predicted as infectious (35 test records), spondylosis as spinal stenosis (27), marijuana abuse as personality disorder (26) and infectious as non-infectious gastroenteritis (25). A symptom-only model has little information with which to separate such pairs.

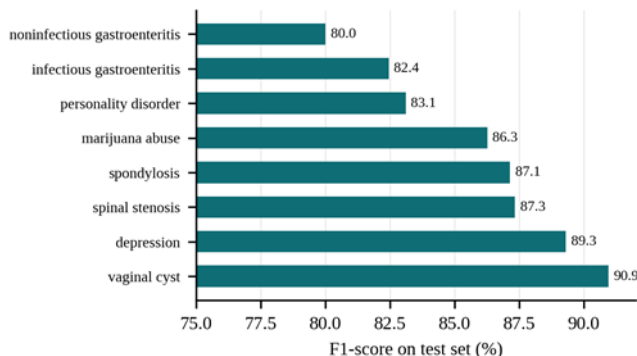


Fig. 4. The eight conditions with the lowest F1-score

E. Effect of Limited Symptom Input

To mimic users who enter few symptoms, each test record with at least k active symptoms was reduced to k randomly chosen ones (one draw, seed 42) and passed to the model (Fig. 5). Top-1 accuracy is 32.0%, 63.4%, 80.7%, 89.5% and 94.0% for $k = 1$ to 5, and top-3 accuracy is 65.2%, 92.4%, 98.6%, 99.8% and 100.0%. Mean top confidence rises from 23.1% to 76.4% over the same range. With one symptom the model is mostly guessing; with three the correct condition is within the top three in 98.6% of cases. This supports both the three-symptom minimum and the choice to display a ranked list instead of a single label. The test is only an approximation, since real users may choose symptoms differently from a random subset.

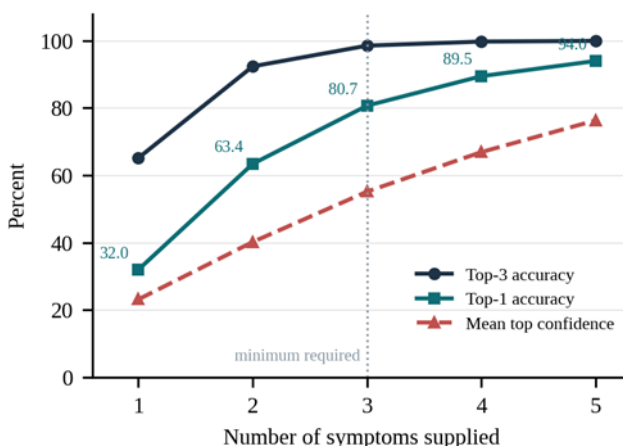


Fig. 5. Accuracy and confidence versus number of symptoms supplied.

F. Worked Examples

TABLE III
EXAMPLE INPUTS AND RETURNED RANKINGS

Symptoms entered	Result (match strength)	Observation
Cough, fever, wheezing, shortness of breath	Acute bronchitis 31.2%; acute bronchiolitis 28.6%; pneumonia 26.5%	Shared respiratory picture; scores nearly equal

Symptoms entered	Result (match strength)	Observation
Toothache, facial pain, gum pain	Dental caries 88.5%	Distinctive combination; one dominant result
Headache, nausea, abnormal-appearing skin	Eczema 9.6%; sebaceous cyst 8.7%; pyogenic skin infection 8.5%; psoriasis 7.6%	Weak, mixed evidence; low scores flag an uninformative result

G. Application Testing

Functional tests of the running application confirmed that unauthenticated visits are redirected to login, registration and login work, requests with two symptoms are rejected with HTTP 400, a PDF report is generated for valid input, the admin page is closed to ordinary users, and the locator returns K.R. Hospital, Mysuru, 2.0 km away for coordinates in central Mysuru when no live key is set.

H. Limitations

- The public table is artificially generated [12], so it is not clinical ground truth, and a random split tests only that same synthetic distribution. High accuracy here is therefore not comparable with vignette-based audits, in which the best app placed the right condition in its top three in about 70% of cases [2].
- Only 50 conditions are covered, and age, sex, duration and severity are not used.
- Probabilities are not calibrated [7] and should be read as relative match strength.
- 193 of the 377 selectable symptoms never occur in the retained data, so choosing them adds no information; the list shown to users should be limited to the 184 informative symptoms.
- The offline directory lists major hospitals only; finer results need a live places service.

VIII. CONCLUSION

Multicare Disease Predictor combines a cleaned 52,718-record dataset, a leaf-size-regularised Random Forest and a web interface that is open about its limits. The model reaches 96.13% accuracy and 99.89% top-3 accuracy on held-out records, and the partial-input analysis shows why at least three symptoms are required: top-1 accuracy is 80.7% at three symptoms but 32.0% at one. Evidence panels, condition guidance, urgency flags, a PDF summary and a nearby-care locator turn a probability list into usable next steps.

Because the training data are synthetic, the system remains an educational aid. Future work includes evaluation on an independent clinical dataset, probability calibration, restricting the symptom list to informative entries, adding age and sex, and offering a regional-language interface.

IX. IMPLEMENTATION SUMMARY

The technology stack is listed in Table IV.

TABLE IV
IMPLEMENTATION STACK

Layer	Implementation
Backend	Python 3 and Flask [9]; session-based authentication
Machine learning	pandas, NumPy, scikit-learn 1.8.0 [5]; joblib model files
Model	Random Forest: 100 trees, min_samples_leaf = 5, seed 42

Layer	Implementation
Database	SQLite [10] through the built-in sqlite3 module
Frontend	HTML, CSS and JavaScript
Reports	ReportLab PDF generation
Care locator	Optional Geoapify [11] or LocationIQ search; offline directory of 50 hospitals; haversine distance
Artifacts	best_model.pkl, feature_names.pkl, label_encoder.pkl, symptom_prevalence.json

A. Security and Privacy

Passwords are stored as salted hashes. The administrator account and the session secret are read from environment variables and must be replaced before any public deployment. Stored activity consists of the account name, symptom list, top result, confidence and timestamp.

B. Reproducibility

The data-cleaning and training scripts, the 80/20 stratified split with seed 42 and the forest parameters above regenerate the reported results. Training uses leaf size 5 so that the retrained model matches the deployed one.

C. Future Development

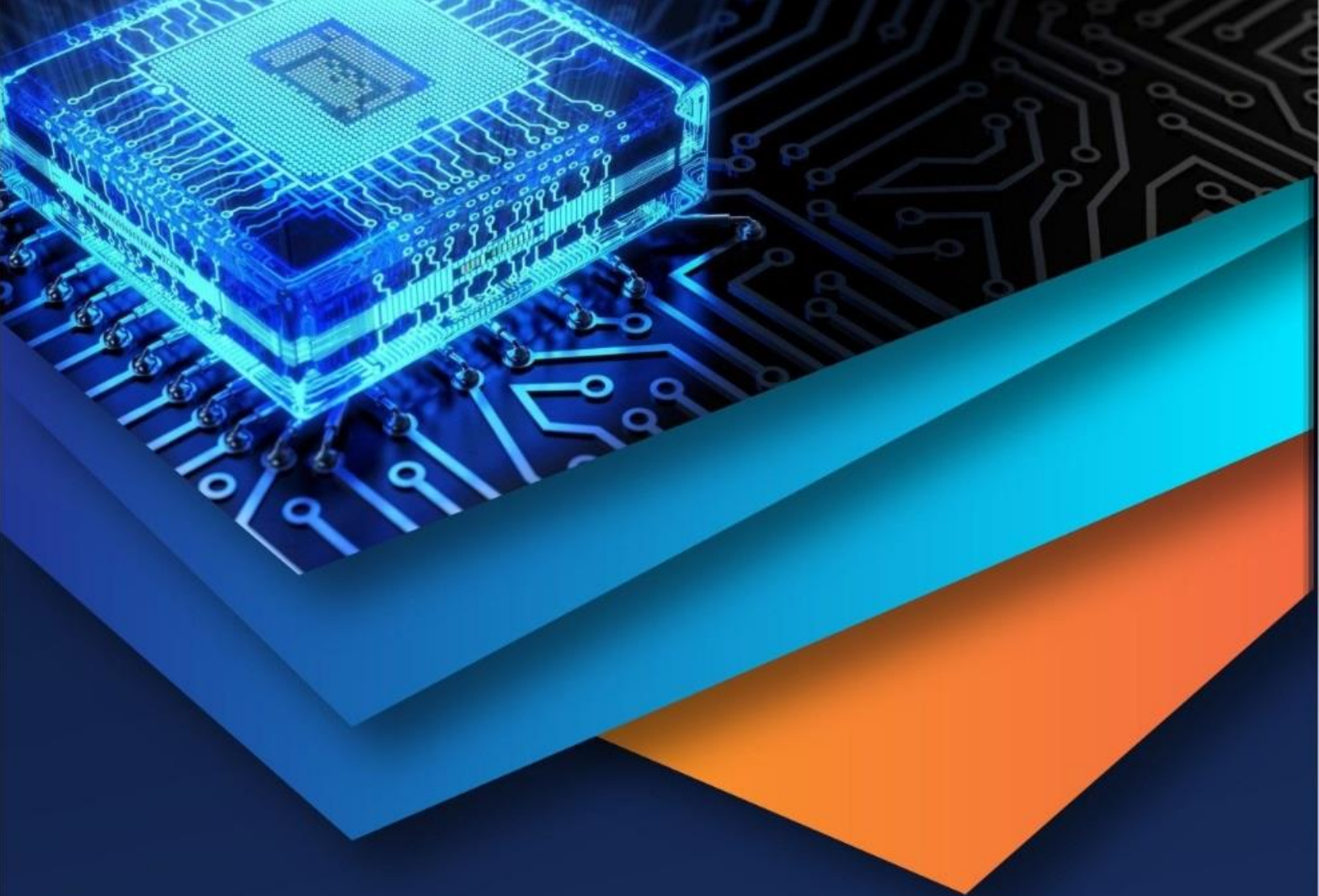
- Validate on an independent, clinically sourced dataset.
- Calibrate probabilities and report calibration error.
- Limit the interface to informative symptoms and add age and sex inputs.
- Support Kannada and other regional languages.

X. ACKNOWLEDGMENT

The authors thank the Department of Computer Science and Engineering, MRIT, Mandya, and their project guide, Ms. Harshitha M, for guidance and support throughout this work.

REFERENCES

- [1] H. L. Semigran, J. A. Linder, C. Gidengil, and A. Mehrotra, "Evaluation of symptom checkers for self diagnosis and triage: audit study," *BMJ*, vol. 351, p. h3480, 2015, doi: 10.1136/bmj.h3480.
- [2] S. Gilbert et al., "How accurate are digital symptom assessment apps for suggesting conditions and urgency advice? A clinical vignettes comparison to GPs," *BMJ Open*, vol. 10, no. 12, p. e040269, 2020, doi: 10.1136/bmjopen-2020-040269.
- [3] W. Wallace et al., "The diagnostic and triage accuracy of digital and online symptom checker tools: a systematic review," *npj Digital Medicine*, vol. 5, Art. no. 118, 2022, doi: 10.1038/s41746-022-00667-w.
- [4] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
- [5] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
- [6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. New York, NY, USA: Springer, 2009.
- [7] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proc. 22nd Int. Conf. Machine Learning (ICML)*, 2005, pp. 625–632.
- [8] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manage.*, vol. 45, no. 4, pp. 427–437, 2009.
- [9] Pallets Projects, "Flask documentation," [Online]. Available: <https://flask.palletsprojects.com/>. Accessed: Oct. 2026.
- [10] SQLite Consortium, "SQLite documentation," [Online]. Available: <https://www.sqlite.org/docs.html>. Accessed: Oct. 2026.
- [11] Geoapify, "Places API documentation," [Online]. Available: <https://apidocs.geoapify.com/docs/places/>. Accessed: Oct. 2026.
- [12] B. Stark, "Disease and symptoms dataset 2023," *Mendeley Data*, V1, 2025, doi: 10.17632/2cxcsxydc1. [Online]. Available: <https://data.mendeley.com/datasets/2cxcsxydc1>. Accessed: Oct. 2026.



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129



IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)