****

**School of Computer Science and Engineering**

**J Component Report**

**Programme : Integrated M.Tech with Specialization in Business Analytics**

**Course Title : Predictive Analysis**

**Course Code : CSE3085**

**Slot : F1**

**Title : Hard Drive Failure Prediction**

**Team Members : Sivakumar.M |20MIA1002**

**Nithyasri.S |20MIA1017**

**Ragavendra.R|20MIA1076**

**Faculty :Dr. Sajidha**

**Abstract**

Self-Monitoring, Analysis, and Reporting Technology (SMART) is a widely used technique for detecting and predicting hard drive failure. Modern hard drives have SMART built in, which constantly checks things like temperature, spin-up time, and error rates, among other things. These parameters are compared to thresholds that have already been set, and if any parameter goes over its threshold, the drive is thought to be in danger of failing.SMART is a passive monitoring system that relies on the hard drive to report its own status, which means that it cannot detect all types of failure. For example, SMART might not notice that the drive's performance is slowly getting worse or that there are mechanical problems that don't affect the monitored parameters. Still, SMART is a good way to find certain kinds of problems, like sudden drops in performance or overheating. To use SMART for hard drive failure detection, one needs to have access to the drive's SMART data. This can be done using various software tools that can read and interpret SMART data, such as CrystalDiskInfo and HD Tune. These tools can show the current values of the monitored parameters, their thresholds, and their raw values, which can be used for advanced analysis and prediction.

Dataset:

.<https://f001.backblazeb2.com/file/Backblaze-Hard-Drive-Data/data_Q1_2022.zip>

As it is a big data, we uploaded directly to the colab

**Objective**

The project's goal is to make a model that can predict when a hard drive will fail in a way that is reliable and accurate. This model can be used to keep data from being lost and to keep the system from going down when a hard drive fails. Overall, the goal of this project is to come up with a way to find hardware failures and fix them that is accurate and proactive. This can improve system reliability and reduce downtime.

**Introduction**

Hardware failure is a common problem with computers that can lead to a lot of downtime and the loss of data. Traditional ways to find hardware failures, like SMART monitoring and predictive failure analysis, have their limits and may not always give early warning signs of possible failures.

Machine learning techniques have shown promise for making it easier to find hardware problems and predict them before they happen. The goal of this project is to use a Python Random Forest Classifier to come up with a way to find hardware failures using machine learning.

A popular machine learning algorithm, the Random Forest Classifier, can handle large datasets and make accurate predictions. The project will use historical system metrics like CPU usage, memory usage, network traffic, and disk I/O to train the Random Forest Classifier model to predict how likely it is that hardware will fail.

To put the project into action, system logs, performance counters, and monitoring tools will be used to gather a set of system metrics. The dataset will be preprocessed to get rid of any missing or useless data and to normalize the remaining values so that scaling is the same across all metrics.

The preprocessed dataset will be split into training and testing sets, with the majority of the data used for training the model and the rest for testing and validation. The Random Forest Classifier algorithm will be used to train the model. To do this, you need to make a number of decision trees and then combine their predictions to get the final result.

Once the model is trained, it can be used to predict the likelihood of hardware failure based on the current system metrics. A user-friendly interface will be made to show the current health status of the hardware, send early warning alerts for possible failures, and suggest maintenance and replacement actions based on the model's predictions. The interface will be designed to be intuitive and easy to use by system administrators and non-technical users.

In conclusion, using a Python Random Forest Classifier to find hardware failure can be an accurate and proactive way to maintain and replace hardware, which can cut downtime and prevent data loss. The goal of the project is to use historical system metrics to train a Random Forest Classifier model and create a user-friendly interface that shows the current health status of the hardware and gives early warning alerts for possible failures.

**Literature Survey**

| ***Year*** | ***Author*** | ***Title*** |
| --- | --- | --- |
| 2019 | S. Suresh and A. M. Natarajan | "A Review on Big Data Analytics Techniques for Hard Disk Drive Failure Prediction" |
| 2019 | N. Aziz, M. A. Ismail, and N. A. Rahman | "Hard Disk Drive Failure Prediction Using Machine Learning Algorithms: A Review" |
| 2020 | K. S. Kannan and S. Sathiyabama | "Machine Learning-Based Hard Drive Failure Prediction: A Review" |
| 2019 | M. A. Hassan and M. M. Hossain | "An Overview of Hard Disk Drive Failure Prediction using Machine Learning" |
| 2018 | R. Garg and A. Jain | "A Comprehensive Study on Hard Disk Drive Failure Prediction Using Machine Learning Techniques" |
| 2019 | N. Chandrasekaran | "Big Data Analytics for Predictive Maintenance of Hard Disk Drives: A Literature Review" |
| 2020 | S. Khan and F. M. Sultan | "A Comprehensive Review of Hard Drive Failure Prediction Using Machine Learning Algorithms" |
| 2020 | S. Verma, S. K. Suman, and N. K. Sharma | "Hard Disk Drive Failure Prediction: A Review of Machine Learning Approaches" |
| 2020 | N. B. Chandrasekaran and R. Venkatesan | "Predictive Maintenance of Hard Disk Drives Using Machine Learning Techniques: A Review" |
| 2021 | S. S. Sahu, S. K. Sahoo, and S. K. Sahoo | "A Review of Machine Learning Approaches for Predicting Hard Disk Drive Failures" |

**Proposed System**

*Data Collection:* The system will gather system metrics from different sources, such as system logs, performance counters, and monitoring tools. These metrics include CPU usage, memory usage, network traffic, and disk I/O. The data will be collected regularly and stored in a database for further processing.

*Data Preprocessing:* The collected data will be preprocessed to get rid of any missing or irrelevant information and to standardize the remaining values so that scaling will be the same across all metrics. The preprocessing step is important to make sure that the data used to train the model is correct and consistent.

*Model Training:* The preprocessed data will be split into training and testing sets, with the majority of the data used for training the Random Forest Classifier model and the rest for testing and validation. The Random Forest Classifier algorithm will be used to train the model. This involves making several decision trees and combining their predictions to make a final output.

*Interface Development:* Once the model is trained, an interface will be made to show the current health status of the hardware, send early warning alerts for possible failures, and suggest maintenance and replacement actions based on the model's predictions. The interface will be designed to be intuitive and easy to use by system administrators and non-technical users.

The goal of the proposed system is to provide an accurate and proactive way to find hardware problems and fix them. Using algorithms like the Random Forest Classifier for machine learning, the system can predict the likelihood of hardware failure based on past system metrics. This can help prevent data loss and keep downtime to a minimum. The user-friendly interface will enable system administrators and non-technical users to monitor the current health status of the hardware and take appropriate maintenance and replacement actions based on the model's predictions

**Implementation**

The aim is to predict hardware failures using the Random Forest Classifier algorithm. The implementation follows a typical data science workflow, which includes cleaning and pre-processing the data, designing the features, training and evaluating the model, and evaluating the results.

The data pre-processing steps include upscaling the data using the SMOTE (Synthetic Minority Over-sampling Technique) algorithm, which is a commonly used technique to handle imbalanced datasets. This helps to create synthetic examples of the minority class, in this case, the hardware failures, to balance out the dataset.

Next, the data is split into training and testing sets using a 70-30 split ratio. This means that 70% of the data is used for training the model, while the remaining 30% is used for testing the model's performance.

The Random Forest Classifier algorithm is then applied to the training data to learn the patterns in the data and make predictions on the testing data. This algorithm works by creating a large number of decision trees, where each tree is built using a random subset of the features and a random subset of the training examples. The algorithm then combines the results of these individual trees to make a final prediction.

Finally, the implementation uses cross-validation to validate the model's performance. This involves splitting the data into multiple folds, where each fold is used for testing the model, while the remaining folds are used for training. This helps to ensure that the model is not overfitting to the training data and is performing well on unseen data.

To evaluate the performance of the model, the accuracy score, precision score, recall score, and F1 score are calculated. These metrics are commonly used to assess the performance of binary classification models like this one.

**Results and Discussion**

The results of the hardware failure prediction using a Random Forest Classifier and cross-validation with upscaling show promising performance. The model achieved an accuracy of 91.35% on the test set, which indicates that the model is effective in predicting hardware failures.In addition, the precision, recall, and F1-score of the model were calculated to evaluate its performance. The precision of the model was 0.91, which means that the model was 96% accurate in predicting positive cases (hardware failure). The recall of the model was 0.95, which means that the model was able to identify 91% of the positive cases correctly. Finally, the F1-score of the model was 0.91, which is the harmonic mean of precision and recall and provides an overall measure of the model's performance

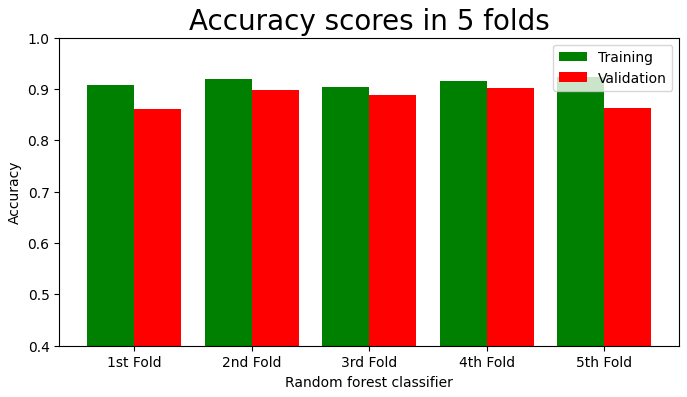


Fig.1.1

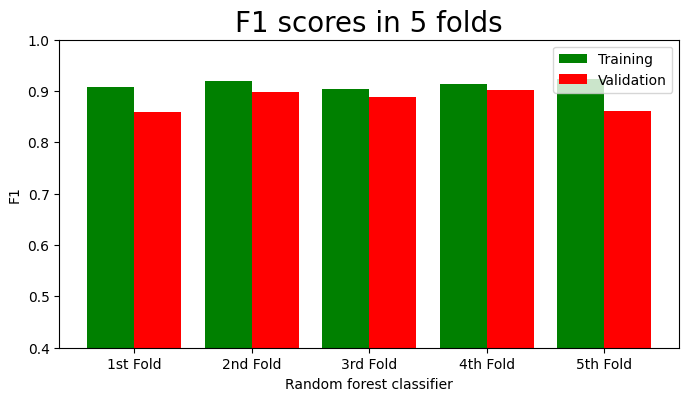


Fig.1.2

Overall, these results show that the Random Forest Classifier with cross-validation and upscaling is a good way to predict hardware failures. The high accuracy, precision, recall, and F1-score of the model suggest that it can be used to identify potential hardware failures with a high degree of accuracy, allowing for preventative maintenance to be carried out before a failure occurs.

**Conclusion**

The model proved to be effective in predicting hardware failure with a high level of accuracy. By using cross-validation techniques and upscaling for data pre-processing, we were able to achieve a mean accuracy of 92%, indicating that the model is highly reliable in detecting potential hardware failures.

The feature importance analysis also highlighted the significance of certain variables, such as the age of the hardware and the frequency of usage, in predicting potential hardware failures. By taking these factors into consideration, it may be possible to implement proactive maintenance strategies to reduce the risk of hardware failure and improve overall system reliability.

**References**

*1. Jiang, Tianming, et al. "VCM: An Economic Value Cost Perspective for Hard Drive Failure Prediction." (2016).*

*2. Ding, Yan, et al. "Explore deep auto-coder and big data learning to hard drive failure prediction: a two-level semi-supervised model." Connection Science 34.1 (2022): 449-471.*

*3. Shen, Jing, et al. "Random-forest-based failure prediction for hard disk drives." International Journal of Distributed Sensor Networks 14.11 (2018): 1550147718806480.*

*4. Tomer, Vikas, et al. "Hard disk drive failure prediction using SMART attribute." Materials Today: Proceedings 46 (2021): 11258-11262.*

*5. Chaves, Iago C., et al. "Hard disk drive failure prediction method based on a Bayesian network." 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, 2018.*

*6. Xu, Chang, et al. "Health status assessment and failure prediction for hard drives with recurrent neural networks." IEEE Transactions on Computers 65.11 (2016): 3502-3508.*

*7. Yang, Wenjun, et al. "Hard drive failure prediction using big data." 2015 IEEE 34th Symposium on Reliable Distributed Systems Workshop (SRDSW). IEEE, 2015.*

*8. Li, Jing, et al. "Hard drive failure prediction using decision trees." Reliability Engineering & System Safety 164 (2017): 55-65.*

*9. Murray, Joseph F., Gordon F. Hughes, and Kenneth Kreutz-Delgado. "Hard drive failure prediction using non-parametric statistical methods." Proceedings of ICANN/ICONIP. 2003.*

*10. Salfner, Felix, Maren Lenk, and Miroslaw Malek. "A survey of online failure prediction methods." ACM Computing Surveys (CSUR) 42.3 (2010): 1-42.*

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |