ML APPLICATION IN ELECTRIC MOTOR TEMPERATURE

Junxian Qu⋆*, Yifan Wang*⋆*, Zikuan Wang*⋆

⋆(Group 18) University of California San Diego, La Jolla, CA 92093-0238,

ABSTRACT

This project implements ensemble machine learning (ML) and deep learning models on electric motor temperature prediction. Based on the dataset from kaggle, this project first exploits some feature engineering methods on motor for a more appropriate extraction, then uses Randomforest Classifier, Adaboost Classifier and Gradient Boost Classifier to train model. By analyzing the result of temperature prediction shows the high learning score of this approach.

***Index Terms*—** feature engineering, Random Forest Classifier, Adaboost Classifier, Gradient Boost Classifier

1. INTRODUCTION

As the advance of power density of motor, electric vehicles become popular. Stability of Motor influences the safety of vehicle, thus motor researchers always focus more on improving its stability.

Most electric vehicle motor adopt rare earth material. One drawback of rare earth material is that it would demagnetize under high temperature. This will trigger a surge in current thus may cause an accident. To prevent this, finding a way to prediction the temperature of motor while working is urgent. We referenced and reused data-set and code implementations from ECE285 and ECE225. However, we went deeper and make it more refined in this project.

1.1. Analytical Model

There are two popular approaches for motor temperature prediction. One of them is analytical model[1]. To boost runtime of constructing a simulating model, the analytical model combining electromagnetic and thermal field comes up[2] by computing the equivalent resistance of motor.

$$T_r = T_{r0} + \frac{R_r - R_{r0}}{\alpha R_{r0}} \tag{1}$$

where T_{r0} is the reference temperature of permanent magnet, T_r is the estimated temperature, R_r, R_{r0} are the equivalent resistance at T_r and T_{r0}, α is the temperature coefficient of resistivity.[3].

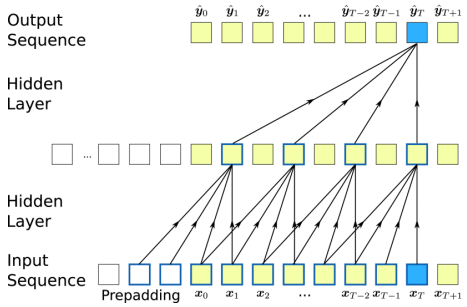


Fig. 1. Deep Residual ML Model[4]

1.2. ML Method

Another method is ML algorithm. Wilhelm Kirchgässner exerted convolutional network and residual connection on training model, which successfully estimates the temperature within 3K error[4, 5]. Juncai Song constructed an extreme learning machine to predict demagnetization[6], which enhanced the stability of motor. Random forest model are adopted by Rushank Savant for predicting and analyzing[7]. ML approach outperforms analytical modeling in both accuracy and runtime. Analytical method created by physical equations under ideal conditions, whose error is much bigger applied to practical experiments. Besides, building analytical model is quite time-consuming. However, ML model would be perfectly accurate with enough training data. And ML approach does not construct special motor model thus save a lot of time.

2. DATA FEATURE ENGINEERING

Our project based on the data set from Kaggle: *Electric Motor Temperature: 185 hrs recordings from a permanent magnet synchronous motor (PMSM)*
https://www.kaggle.com/wkirgsn/electric-motor-temperature

This dataset has 13 parameters. The temperature of permanent magnet is one of the most important thermal parameter for a motor. However, in actual situations, monitoring the temperature of a rotating part is very difficult. So the output parameter should be the temperature of permanent magnet.

profile$_{id}$	id of experiment
u_d(V)	d axis component of voltage
u_q(V)	q axis component of voltage
i_d(A)	d axis component of current
i_q(A)	q axis component of current
Temp$_{coolant}$(^oC)	temperature of coolant
Temp$_{winding}$(^oC)	temperature of winding
Temp$_{tooth}$(^oC)	temperature of tooth
Temp$_{yoke}$(^oC)	temperature of yoke
Temp$_{ambient}$(^oC)	temperature of ambient
Temp$_{magnet}$(^oC)	temperature of magnet
Torque(Nm)	Output force
Speed(Rpm)	the rotation speed

2.1. Creating Electric Input Parameter

The vector of voltage u and current i in d-q coordinate of motor are expressed in Fig.2.With u_d, u_q, i_d and i_q, the vector of total voltage and current is attainable.

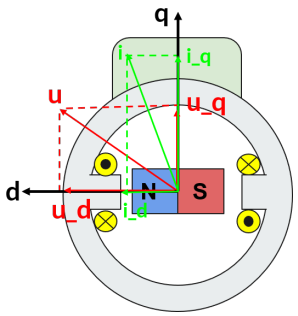


Fig. 2. Vectors in d-q Coordinate

Based on available data, three new parameters are added, and the data set has 15 inputs now.

$$\begin{cases} u = \sqrt{u_q^2 + u_d^2} \\ i = \sqrt{i_q^2 + i_d^2} \\ \cos\theta = (u_d * i_d + u_q * i_q)/(u * i) \end{cases} \tag{2}$$

2.2. Creating Electric Output Parameter

Next, feature engineering on motor's output can be implemented. The electric power, mechanical power and efficiency are the most important outputs of motor. They can be calculated by equation (3).

$$\begin{cases} Power_{ele} = u * i * \cos\theta \\ Power_{mec} = Speed * Torque/9.55 \\ \eta_{efficiency} = Power_{mec}/Power_{ele} \end{cases} \tag{3}$$

And thus there are 18 input parameters in the data set. With those created parameters, the training model would be more accurate and efficient.

3. IMPLEMENTATION DETAILS AND MODELS

3.1. Pre-processing and Handle Imbalance Data

To design a warning light dashboard monitoring the operation status of the electric motors, "temperature of the magnet" is picked as the predicting feature. Then the data is shuffled with the ratio between training and testing data as 70:30. The distribution of "temperature of the magnet" is plotted.

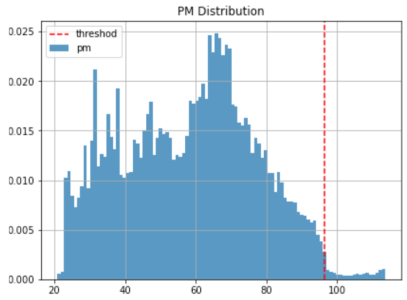


Fig. 3. Distribution of the feature pm(temperature of magnet) with threshold

The Fig.3 shows the distribution of temperature of the magnet is pretty much a bell-shape distribution. Therefore it's appropriate to set the threshold at two standard deviation away from the mean value. The red line in the figure above shows the threshold. The data point above the threshold are labeled as "1" and others are labeled as "0", where "1" represents warning status and "0" represents the normal status. However, there is a huge difference in the amount of data with label "0" (1,312,242 samples) and label "1" (14,316 samples). Therefore, we cut the data with label "0" to the same amount of data with label "1". In this way, the model will perform evenly well when testing on both cases[8].

3.2. Randomforest Classifier

There are two major components in the ensemble: bagging and boosting. Bagging works as the parallel tree growing with sub-samples. The first model we used was random forest which contains large number of individual decision trees and the model randomly restrict features in each tree split[9]. Randomforest can be very helpful when model is complex and easy to over-fit[10].

3.3. Adaboost Classifier

The adaboost classifier is one of the boosting algorithms and it assigned non-uniform weight(4) based on the error from the previous iteration. From weight function(5) we can tell the smaller error(epsilon) gives larger weight(alpha) for final voting. Therefore, it can improve the model accuracy

quickly[11].

$$H(x) = Sign * \sum_{t=1}^{T} \alpha_t * f_t(x) \tag{4}$$

$$\alpha_t = \ln(\sqrt{\frac{(1-\epsilon_t)}{\epsilon_t}}) \tag{5}$$

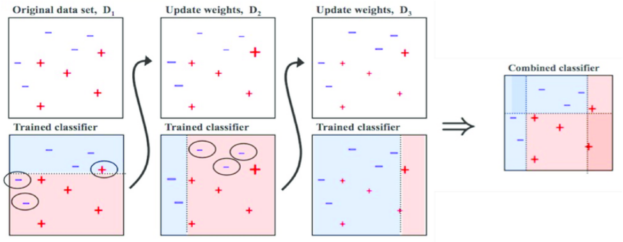


Fig. 4. Example of adaboost classification

An easy example of adaboost classifier shown in Fig.4. Initially, the weak classifier had three errors in the first stage. After re-weighting on these errors in each iteration, the final result is perfect. Therefore, the adaboost can get 100 percent accuracy on the training data as long as the weak classifier has better accuracy than random guess.

3.4. Gradientboost classifier

Gradientboost classifier has similar idea from the adaboost classifier which uses multiple weak learners to produce a strong model for perform classification tasks[12]. However, gradientboost focus on fitting a new predictor to the residual error made by previous predictor[13] and in the end minimized the overall prediction errors. It's weight function(alpha) at each iteration shown in equation(5) which means smaller error(epsilon) gives larger weight(alpha) for the final voting.

4. RESULTS ANALYSIS AND VISUALIZATIONS

4.1. Cross-Validation

Before deciding the number of estimators for each classifier, the cross-validation and grid search were performed for all three classifiers in the previous sections to find the optimum number of estimators. The Fig.5 displays the different classifiers have different optimum number of trees to make the accuracy score saturated. Therefore, we choose 15 estimators for randomforest classifier, 50 estimators for adaboost classifier and gradientboost classifier.

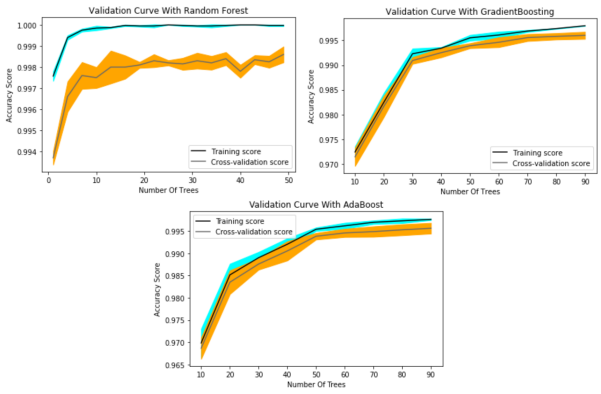
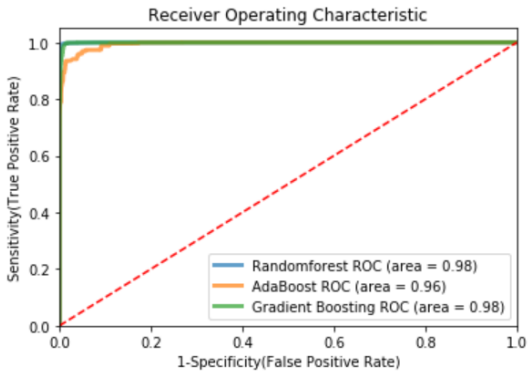


Fig. 5. The validation curve for RandomForest, GradientBoost, and AdaBoost classifier

4.2. Experiment Results

Our data-set was trained by randomforest, adaboost and gradientboost classifier with selected optimum estimator setting and let it perform the classification on our testing set. Then the value of ROC(Receiver Operating Characteristic) is displayed in Fig.6.



	Model	Accuracy	Precision	Recall	F1 Score	ROC
0	Randomforest	0.983586	0.999022	0.967566	0.983043	0.983325
0	AdaBoost	0.955879	0.973639	0.935606	0.954244	0.955549
0	Gradient Boosting	0.985099	0.992071	0.977509	0.984736	0.984976

Fig. 6. The ROC graph and the table of scores

Based on the score table and ROC curve, our model is pretty powerful when handling this classification task even with relatively small number of estimators. Some of those 18 input features had a strong correlation with the output, which contributed to this high efficiency.

4.3. Visualizations by SHAP(SHapley Additive exPlanations)[14][15]

To demonstrate the "black-box" of this ensemble model, the traditional graph visualizations and feature importance cannot be satisfied. However, the SHAP can help us[16][17][18][19]. The principle ideas of shapley values is compute the average

marginal contribution of each feature across all possible ordering such as the tree paths. Besides, it can display the impact of each features to the output directly. Equation(6) was used to calculate the Shapley value on all the features.

$$\phi(v) = \frac{1}{N} * \sum \frac{\text{marginal contribution of i to coalitions}}{\text{number of coalitions excluding i of this size}} \tag{6}$$

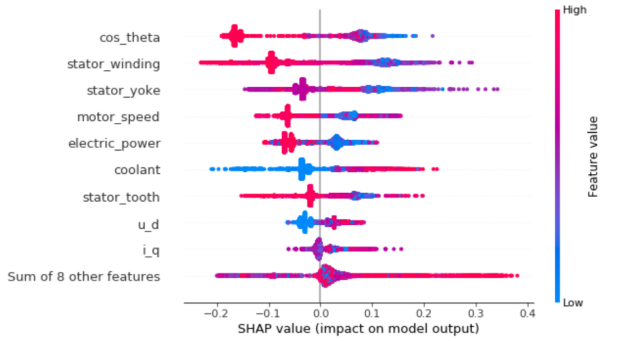


Fig. 7. The global feature importance interpretation

In Fig.7, the global impact on the model output are shown with features. The red color represent higher feature value and the blue color represent the lower feature value. The stator-winding, stator-yoke, cos-theta and motor speed have relatively higher impact on the output. The individual feature importance are interpretated in Fig.8. In the image, the stator-winding and stator-yoke have large impact in label "1", and for label "0", cos-theta and stator-winding have large impact on it. Fig.9 shows the correlation between feature stator-yoke and coolant. The smoother color transition in graphs represents stronger correlation between features. Temperatures of stator-yoke and coolant are actually related in the physical operation of electric motor, so it matched up with our results. If two features have weak correlations, the color of dots will be be scattered and mixed.

5. CONCLUSIONS

Our project successfully designed the warning light dashboard by using tree based ensemble models and SHAP to interpreted our models. Based on the score table and ROC curve, our model perform very well in our testing data-set and it's pretty powerful when handling 18 features as the input and it made great predictions on the target. Besides, the SHAP helped us visualize the impacts of each feature globally and individually. In the future, the dashboard can be extended to multi-level status monitor for electric motors by adding more classes into the classification models. Besides, we can display several motor operating parameters at the same time and monitor the status of the motor. With the help of tree based ensemble models, our design could have huge potentials in practical usage.

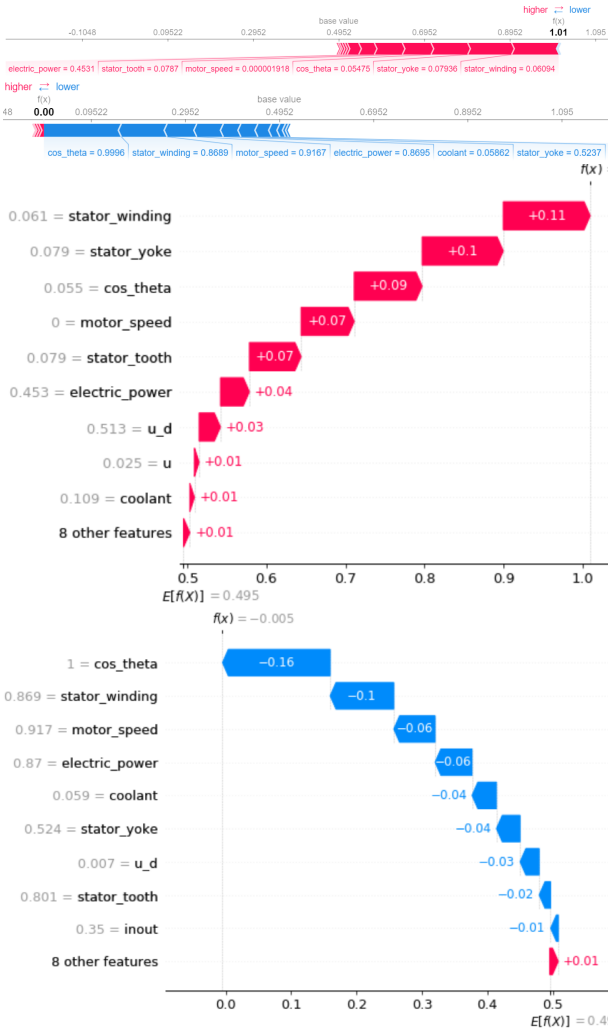


Fig. 8. The individual feature importance interpretation

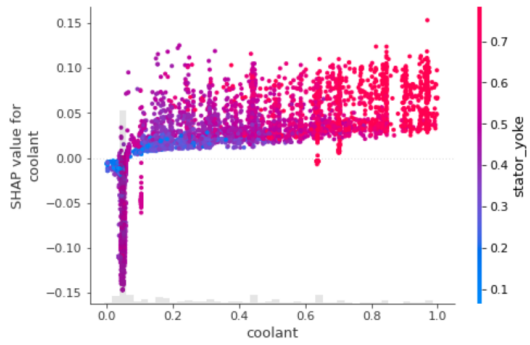


Fig. 9. The relation between stator-yoke VS coolant

6. REFERENCES

[1] Kyung-Rae Cho and Jul-Ki Seok. Induction motor rotor temperature estimation based on a high-frequency model of a rotor bar. *IEEE Transactions on Industry Applications*, 45(4):1267–1275, 2009.

[2] Xuzhen Huang, Qiang Tan, Liyi Li, Jing Li, and Zhenyu Qian. Winding temperature field model considering void ratio and temperature rise of a permanent-magnet synchronous motor with high current density. *IEEE Transactions on Industrial Electronics*, 64(3):2168–2177, 2017.

[3] Sang-Bin Lee, T.G. Habetler, R.G. Harley, and D.J. Gritter. An evaluation of model-based stator resistance estimation for induction motor stator winding temperature monitoring. *IEEE Transactions on Energy Conversion*, 17(1):7–15, 2002.

[4] Wilhelm Kirchgässner, Oliver Wallscheid, and Joachim Böcker. Estimating electric motor temperatures with deep residual machine learning. *IEEE Transactions on Power Electronics*, 36(7):7480–7488, 2021.

[5] Wilhelm Kirchgässner, Oliver Wallscheid, and Joachim Böcker. Deep residual convolutional and recurrent neural networks for temperature estimation in permanent magnet synchronous motors. In *2019 IEEE International Electric Machines Drives Conference (IEMDC)*, pages 1439–1446, 2019.

[6] Juncai Song, Jiwen Zhao, Fei Dong, Jing Zhao, Liang Xu, Lijun Wang, and Fang Xie. Demagnetization modeling research for permanent magnet in pmslm using extreme learning machine. In *2019 IEEE International Electric Machines Drives Conference (IEMDC)*, pages 1757–1761, 2019.

[7] Rushank Savant, Abhiram Ajith Kumar, and Aditya Ghatak. Prediction and analysis of permanent magnet synchronous motor parameters using machine learning algorithms. In *2020 Third International Conference on Advances in Electronics, Computers and Communications (ICAECC)*, pages 1–5, 2020.

[8] Jason Brownlee. Smote for imbalanced classification with python. 2020.

[9] Ahmad TaherAzaraHanaa IsmailElshazlybcAboul Ella-HassanienbcAbeer MohamedElkorany. A random forest classifier for lymph diseases. In *Computer Methods and Programs in Biomedicine*, volume 113, pages 465–473, 2014.

[10] Houtao Deng. An introduction to random forest. 2018.

[11] Robert E. Schapire Yoav Freund. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, 1999.

[12] Anjani Kumar. Introduction to the gradient boosting algorithm. 2020.

[13] Vagif Aliyev. Gradient boosting classification explained through python. 2020.

[14] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

[15] Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. Explainable ai for trees: From local explanations to global understanding, 2019.

[16] Sharayu Rane. Shap: A reliable way to analyze model interpretability. 2019.

[17] Sharayu Rane. Lime: Explaining predictions of machine learning models (1/2). 2019.

[18] Sharayu Rane. Lime: Explaining predictions of machine learning models (1/2). 2019.

[19] Molnar Christoph. *Interpretable Machine Learning*. 2020.

7. Q & A

Critical review from Group 1:

Q: On slide 8, you provide an illustration of how error decreases with the increase of rounds for an Adaboost classifier. It seems like that the maximum accuracy that you can get on the testing set is around 95%. However, on slide 9, the figure shows that you can achieve at least 97.5% accuracy on the same model. Can you explain how these 2 figures are acquired?

A: These two graphs are not related and we have already clearly explained the source and meaning of these two graphs in the video. The graph on slides 8 came from paper "A Short Introduction to Boosting" by Yoav Freund and Robert E. Schapire. We mentioned this picture in the slides because Yoav Freund and Robert E. Schapire found a interesting phenomena which is the testing error will continue decreasing as the training error became zero when they perform the experiment. So we draw the conclusion that adding more number of rounds even after the training error saturated to zero can still be helpful to improve the accuracy. On the slides 9 we performed the cross validation which helped us find the optimum number of estimators. Therefore, these two graphs are completely irrelevant and the numbers in these graphs are accurate

Q: On slide 8, you mention the notion of gradient boosting classifier. This is interesting to us because people seldom use boosting for non-tree-based models. We want to learn more about how you apply this approach to your problem. Would you elaborate on the 3 bullet points shown on the slide, namly, additive model, loss function and weak learner?

A: The gradient boosting classifier is a tree based classifier but due to the limitation on the time of the video, we didn't get into details. We have clearly elaborated how the gradient boosting works in the report. For those three bullet point, additive model means in gradient boosting, decision trees are added one at a time (in sequence), and existing trees in the model are not changed[12]. The loss function is what we learned in the lecture, it can be cross-entropy loss, Hinge Loss or MSE etc.. The weak learner means the initial weak classification model, the model get improved as the iterations goes. Besides, I have already talked about how weak learner improves by using an example in the video.

Q: On slide 10, you provide a comparison across different models. What is the "SGD" model? Is it a non-tree-based model? We would like to know more about the model structure. Briefly explaining the acronyms you have could add more clarity.

A: SGD stands for stochastic gradient descent. I believe we learned the concept of gradient descent in the lecture and Professor also talked about SGD in the lecture. Yes, it is a non tree based model and the purpose of putting it in the graph is show the different on performance of these models.

Q: On slide 15, you mentioned the notion of a multi-level classification task. Could you briefly explain what problem you are trying to solve, i.e., what are the "levels" here corresponding to ?

A: We have talked about the multi-level classification task with details in the video. The multi-level classification basically extends from the binary classification, for example, data-set cifar 10 has 10 classes which can also be classified by modified tree based models. The levels means we can separate the temperature region in to more than 2 pieces and after performing the classification, we can get more detailed information about the operating status.

Q: On slide 15, you mentioned that a possible future direction to work on is to focus on the "winding" feature. However, from my point of view, all these features are somehow inter-correlated. That being said, it's really hard to find a use case where only one feature is enough. For example, you're already using several features of which "PM" and "winding" are the most important two in this project. Thus, we really doubt whether it's a good idea to focus on a single feature.

A: We talked about this part in the report. We are looking forward to design a dashboard which can monitor multiple factors during the operating of the electric motor. Therefore, the dashboard should contains multiple factors we are going to monitor and we plan to use other features to predict these important factors with multiple classes I mentioned in the previous question.

Critical review from Group 31:

Q: In the slides about next steps, you plan to implement a CNN model. However, since in this project, the data are all one-dimensional vectors, it might be a better idea to use a FCN(Fully Connected Neural Network) based model. CNN should be more suitable for 2-D data with a lot more features.

A: Yes, you are right. It should be FCN(Fully Connected Neural Network) model rather than CNN model since all the data we have are 1-d vectors. Good catch, we fixed it in the slides and the report.

Q: About feature extraction: some new meaningful features like Mechanical power and U are generated from linear or non-linear combinations of other features.

Linear features: since newly combined features like Me-

chanical power is a linear combination of other features, it seems to be redundant since all its information is contained in other features, so we suggest deleting those linearly combined features.

Non-linear features: Combined features like U are non-linear combinations of other features and thus they are able to represent "new" information using the component features. We would suggest to try to delete the component features and only keep the combined features (An approach of dimension reduction) to see if there's any improvements, since the combined features are a more meaningful feature and also it contains the information from the components features, and on the other hand less features reduces the data complexity.

A: Thank you for your recommendations. The reason we keep those linearly combined features is we want to see some hidden relations with other features when we do the visualization of the model; specifically the interactive feature part. For just taking the combined feature as input, this suggestion could definitely reduce the complexity of the input and we tried that but there was not a huge improvement on the performance. The reason probably could be some features have strong relation with the prediction targets and the model is pretty strong so the accuracy got improved quickly. The performance of our models was pretty good even before removing the component features. But we can definitely implement your idea in some other models with larger and more complex data-sets. Thank you for your suggestions.

Critical review from Group 14:

No questions from Group 14

8. CONTRIBUTIONS

Junxian Qu: research on the background of motor, connect motor with ML under practical physical meaning, finish data extraction and feature engineering.

Yifan Wang: researched ensemble tree-based models and interpretable machine learning SHAP and LIME, contribute to coding and engineering works.

Zikuan Wang: pre-processing the data and adding labels to the data-set. Designed and implemented the three models for the project. Answered all critique questions and finished data pre-processing, experiment, visualization and conclusion part of project writing