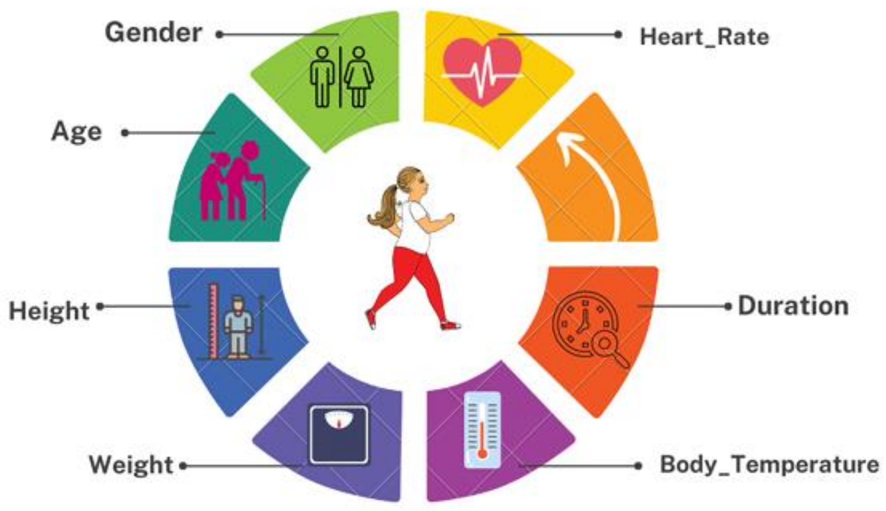
Calories Burnt Prediction using Machine Learning

Re'em Levi

Lecturer: Tamar Shrut

SCE – Sami Shamoon College of Engineering



ABSTRACT

This study aims to predict the calories burned using regression models. The models that been used are Decision Tree Regressor, Random Forest Regressor and XGBoost Regressor.

The dataset that was used in this study was taken from the Kaggle website. The dataset contains 15,000 rows and 9 columns such as Age, Gender, Height, Weight, Duration, Heart_Rate and Body_Temp.

Data preparation, cleaning, and analysis are primary steps before the data can be fit into the regression models.

The performance and prediction accuracy of the regression models were evaluated by R2 Score and Mean Absolute Error.

The result shows that XGBoost Regressor is the best model for the study with an accuracy of 99.8868 % and MAE of 1.498.

LITERATURE REVIEW

XGBoost regressor

Due to its remarkable performance and adaptability in regression problems, the XGBoost regression has attracted significant interest and popularity in the field of machine learning. For calories burned prediction, it has been used in works by [1] and [2]. Extreme Gradient Boosting, or XGBoost, is an approach for ensemble learning that combines regularization methods with the strength of gradient boosting algorithms. According to [1], the XGBoost algorithm performs well since it has robust handling of many varieties of data types, relationships, distributions, and the many hyperparameters that you can fine-tune.

A wide variety of regression issues can be solved using XGBoost because of its capacity to handle various data formats and recognize sophisticated patterns. XGBoost keeps all intermediate calculations in cache so that we don't have to do the same calculation again and again [1]. XGBoost Regression is also known for its performance of managing missing values, handling category and numeric features, and efficiently handling overfitting. XGBoost has regularly shown higher performance than other regression models such as linear regression [2,3]. Although the XGBoost regression is a good model, there are still some limitations to it. XGBoost requires very careful tuning of hyperparameters to get good performance. Other than that, XGBoost makes use of a collection of decision trees, which can use up a lot of memory, especially when working with deep trees or big feature areas.

Random Forest Regressor

Random Forest Regression is a powerful ensemble learning method that combines the advantages of decision trees and the efficiency of ensemble techniques. Ensemble learning is a method that combines predictions from different machine learning algorithms. [3] states that it achieves a higher prediction accuracy than using a single model. Due to its adaptability and reliable performance, Random Forest Regression has found extensive applications across industries such as banking, healthcare and marketing.

High prediction accuracy, robustness against outliers and noise, and the capacity to handle big datasets with high dimensional feature spaces are just a few benefits of Random Forest Regression. According to [3], Random Forest has outperformed linear regression and ridge regression with the lowest mean absolute error of 1.81 calories burned predictions. The limitations of random forest regression are memory hungry, parameter sensitive and limited extrapolation capability. Random forest regression can be memory hungry when working with huge datasets. The number of resources required to build numerous decision trees and store their structures in memory may restrict the algorithm's capacity to scale in some situations.

Conclusion: After conducting the literature review, I chose to use Random Forest Regressor and XGBoost regressor. These models are very popular and with high performance for this study. Also, Decision Tree regressor because both models based on Decision Tree and I want to see the difference in the result.

SYSTEM REVIEW

1. Data collection-dataset collection is the primary step. we used Kaggle as the data repository. Data is then uploaded to the collab platform. Two CSV files (“exercise.csv” and “calories.csv”), The data used here is both categorical and numerical.

2. Pre-processing of data - it is important that we process our data before passing it to the model for better results. null values and missing values are handled at this point because the information on our data directly affects how our model learns.

3. Data Analysis - Data visualization is carried out using various charts and graphs. Heatmap on correlation to studied relation between various features.

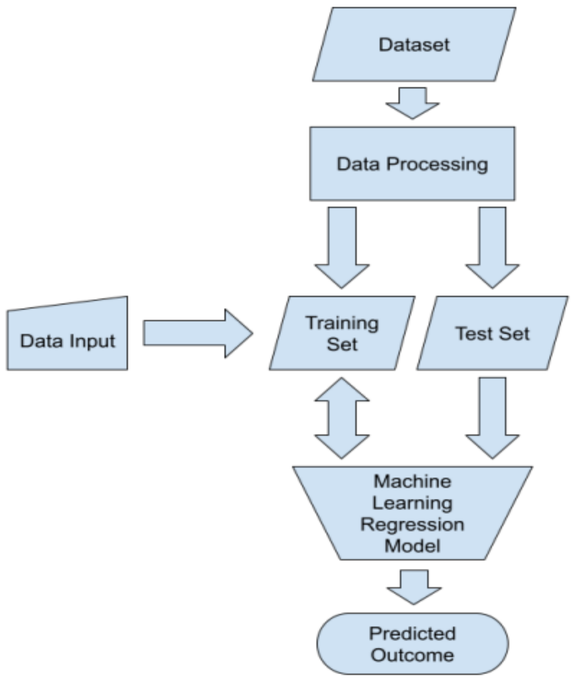
4. Converting categorical data to numerical - 'Gender' column.

5. Separate features and target variable – drop unnecessary features, split the target feature from the other features and then split into test and training data.

6. Models Training & evaluation - all the chosen algorithms are applied at this stage to train and to determine the r^2 value and absolute mean error value.

7. visualize prediction – show visualize of the predicted values and real value of each algorithm.

8. sample prediction – choose N test samples to test the algorithms and to show their predict values against the real values.



DILEMAS

Delete from the algorithm:

1. Feature scaling – the models are insensitive to feature scaling and the feature scaling didn’t contribute to the accuracy of the models.

2. K-Fold Cross Validation – (K-fold >= 20) didn’t contribute too much to the accuracy of the models and hurt the time complexity of the algorithm. (k-fold < 20) didn’t contribute to the accuracy of the models at all.

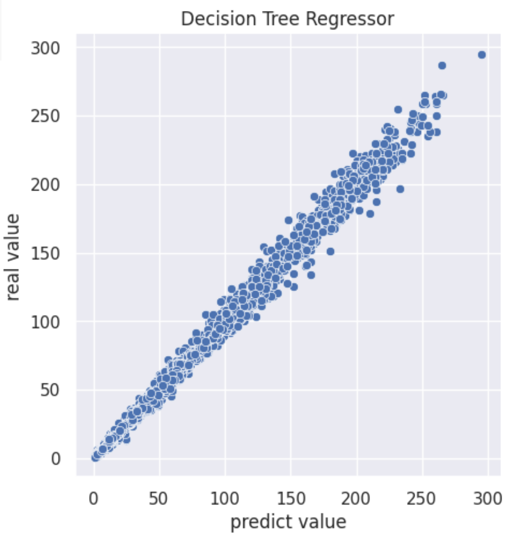
3. Hyperparameters – used GridSearchCV algorithm to find the best parameters for each model to reduce overfitting. Hurt the model's accuracy (8%-13%) and time complexity of the algorithm (mean 7+ minutes for each model). (Choose to use default parameters for the models.)

OPERATING INSTRUCTIONS

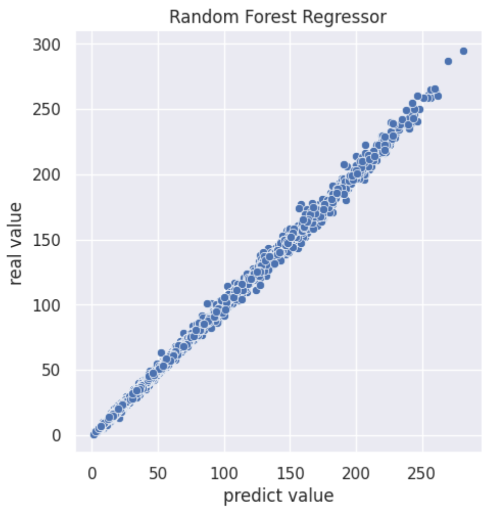
1. Open ipynb file in google colab / jupyter.
2. Add csv files to google colab / jupyter.
3. Run the code blocks.

COMPARISON

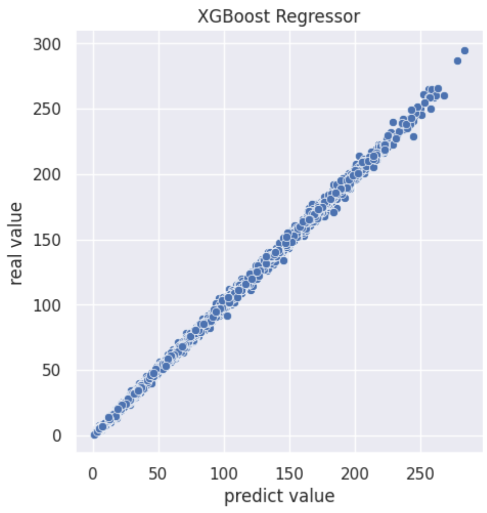
```python
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from xgboost import XGBRegressor
```



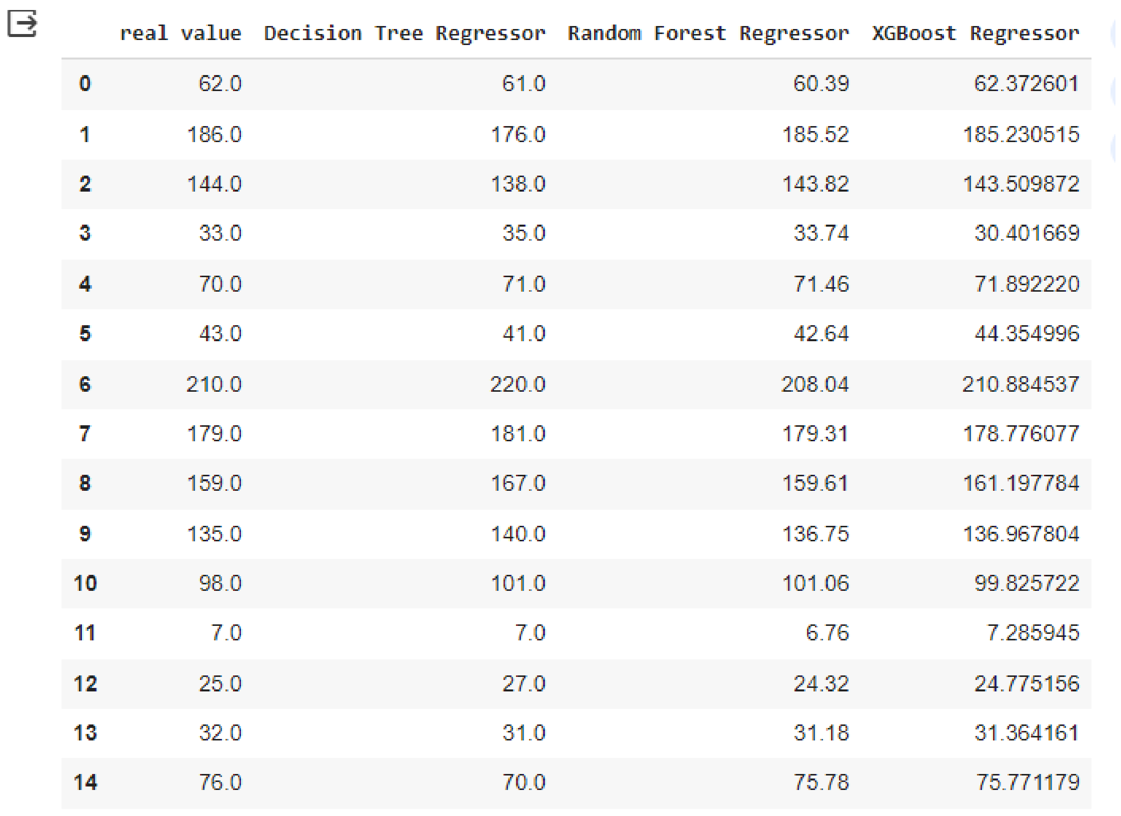
```
Model Name:  Decision Tree Regressor
percentage of accuracy is:  99.3046 %
Mean Absolute Error:  3.4226666666666667
Time:  0.39467692375183105
```



```
Model Name:  Random Forest Regressor
percentage of accuracy is:  99.8219 %
Mean Absolute Error:  1.7065766666666666
Time:  12.274006128311157
```



```
Model Name:  XGBoost Regressor
percentage of accuracy is:  99.8868 %
Mean Absolute Error:  1.4981198125282924
Time:  2.4928555488586426
```



	real value	Decision Tree Regressor	Random Forest Regressor	XGBoost Regressor
0	62.0	61.0	60.39	62.372601
1	186.0	176.0	185.52	185.230515
2	144.0	138.0	143.82	143.509872
3	33.0	35.0	33.74	30.401669
4	70.0	71.0	71.46	71.892220
5	43.0	41.0	42.64	44.354996
6	210.0	220.0	208.04	210.884537
7	179.0	181.0	179.31	178.776077
8	159.0	167.0	159.61	161.197784
9	135.0	140.0	136.75	136.967804
10	98.0	101.0	101.06	99.825722
11	7.0	7.0	6.76	7.285945
12	25.0	27.0	24.32	24.775156
13	32.0	31.0	31.18	31.364161
14	76.0	70.0	75.78	75.771179

Accuracy - all three models perform well in this type of prediction and the differences are not that critical. That been said, the XGBoost has the best performance with accuracy of 99.88% and MAE of 1.49.

Time complexity - not surprise that the model with best time complexity is Decision Tree, one Tree has been built so it's fast unlike the other models. Even though the Random Forest model and XGBoost model have the same default number of Decision Trees to build, there is a huge gap between them in train time complexity, that because XGBoost is generally designed for scalability and efficiency. Its gradient boosting approach builds trees sequentially, focusing on areas of error, which can be more efficient than Random Forest's bagging approach of training numerous independent trees.

Space complexity – the model with the less memory usage is Decision Tree with space complexity of O(number of nudes). While XGBoost and Random Forest have space complexity of O(number of trees * number of nudes). The models are very memory usage hungry and can cause a lot of trouble with big datasets if we don’t choose the right hyperparameters. In my case with dataset of 15,000 and 9 features they worked fine with default hyperparameters.

CONCLUSION

In this paper, we performed a comparative study of the performance of three Machine Learning models, namely XGBoost, Random Forest and Decision Tree, in predicting calories burnt from personal data such as Gender, Age, Height, Weight, Exercise Duration, Body Temperature and Heart Rate.

Decision Trees lay the foundation with their intuitive structure and ease of understanding. Random Forests build on this by creating an ensemble of trees, bringing more stability and accuracy to the predictions. XGBoost then takes it to another level, introducing gradient boosting to create a powerful model that often outperforms many others [4].

The literature review provided insights into various machine learning regression models for predicting calories burned, each with advantages and disadvantages in terms of performance, handling of missing data, scaling, overfitting and handling huge datasets.

The system review included collecting data from Kaggle, preparing the data to ensure data quality, analyzing the data to learn more about the dataset, choosing a model, and then training the model using the selected machine learning algorithms. The accuracy and efficiency of the models were evaluated using metrics, namely mean absolute error and r2 score.

The results allow us to conclude that all three models perform well in this type of prediction and that the XGBoost has the best performance with an accuracy of 99.8868 % and MAE of 1.498.

INSIGHTS

how to choose ML algorithm:

Understand the problem statement – describe the kind of model need to be building.

Understand the data – size, quality and nature of the data can help understand if we need supervise or unsupervised algorithms.

Understand the output – the choice of model type depends upon the type of output that we need to solve the problem.

Importance of preprocessing:

Improve data quality – handling missing values or handling duplicate values helps to build more accurate models.

Enhances model performance - drop irrelevant features can improve the performance of the machine learning models.

Enables feature extraction - feature scaling can enable us to identify the most influential features for model training.

Facilitates compatibility - preprocessing can ensure that the data is in a compatible format for the chosen algorithm.

REFERENCES

Dataset: fmendes-DAT263x-demos

GitHub: GitHub calories prediction

[1] S. P. Vinoy and B. Joseph, "Calorie Burn Prediction Analysis Using XGBoost Regressor and Linear Regression Algorithms," NCECA, 2022. [Online]. Available: https://nceca.in/2022/46_Calorie_Burn_Prediction_Analysis_Using_XGBoost_Regressor_and_Linear_Regression_Algorithms.pdf

[2] N. Meenigea, "Calorie Burn Prediction: A Machine Learning Approach using Physiological and Environmental Factors," Jul. 1, 2014. [Online]. Available: https://www.jetir.org/view?paper=JETIR1701933

[3] M. Nipas et al., "Burned Calories Prediction using Supervised Machine Learning: Regression Algorithm," May 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9776710

[4] Brandon Wohlwend, "Decision Tree, Random Forest, and XGBoost: An Exploration into the Heart of Machine Learning", Jul 2023. [online]. Available: https://medium.com/@brandon93.w/decision-tree-random-forest-and-xgboost-an-exploration-into-the-heart-of-machine-learning-90dc212f4948