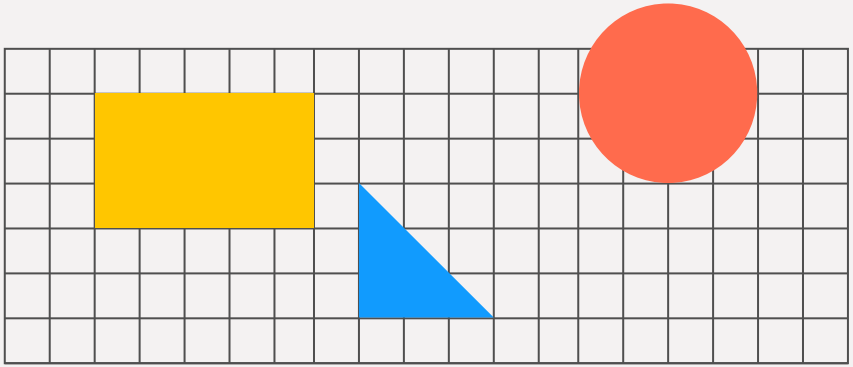
Feature Importance Research for Forgetting Algorithm in Machine Learning Data Selection

Insights into Feature Importance Using SHAP Values on Tabular Data

December 4, 2024



CS 260D - Large Scale Machine Learning - Final Project

Group:
Eric Xuanjia Bi
Zhanyang Gong
Dylan Newman
Reza Serajian

Introduction

Motivation

- Data selection optimizes training by focusing on importance examples, reducing noise, and improving model generalization.
- Large datasets demand significant computational resources, increasing training time and complexity.
- Feature importance quantifies the contribution of each feature to the model's predictive performance.
- It enhances model interpretability, guides feature engineering, and enables effective dimensionality reduction.

Objective

- To investigate how forgetting algorithms affect feature importance in machine learning models.

Forgetting Algorithm

Originally introduced by Toneva et al., 2018, the Forgetting Algorithm focuses on identifying and removing unforgettable (easy) examples from a dataset that have minimal impact on the model's learning process.

It helps create smaller, efficient training datasets without sacrificing generalization.

The process is useful for tasks requiring robustness and efficiency, such as active learning or dataset pruning.

Algorithm Workflow

1. Track each example's prediction history throughout the training iterations.
2. Calculate forgetting scores for all examples based on their prediction shifts.
3. Retain high-forgetting-score examples as the subset for training.
4. Remove unforgettable examples, reducing dataset size while maintaining model performance.

Toneva, et. al., 2018:
https://doi.org/10.48550/arXiv.1812.05159

SHAP Values

SHapley

Additive

exPlanations

- Originally introduced by Lundberg and Lee, 2017 for complex model interpretability
- “Game theory” on ML features
- Marginally contribution of features
- Wide variety of implementation including regression, tree, or neural networks models

ML Explainability

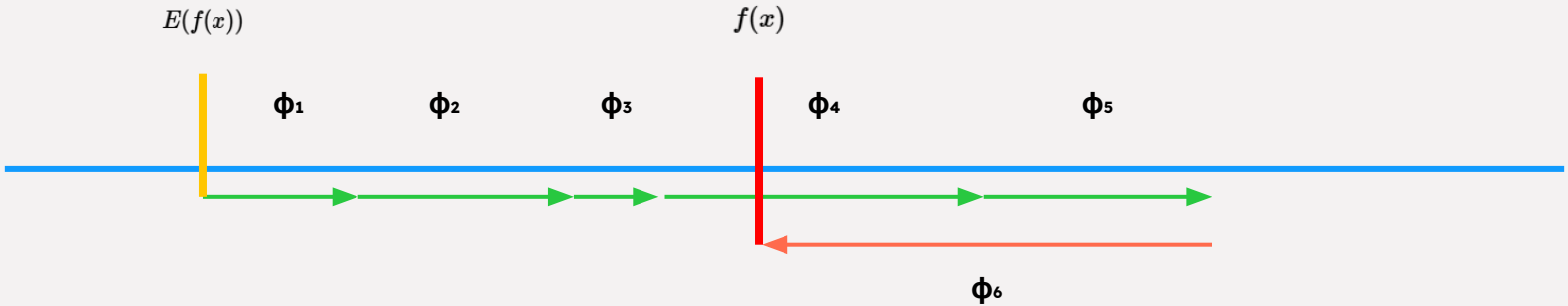
Make the ML “black box” more examinable by understanding which feature of the dataset contribute more on the predictive results

Lundberg and Lee, 2017:

SHAP VALUES

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \left[f_{S \cup \{i\}}(x_{S \cup \{i\}}) - f_S(x_S) \right]$$

ϕi: The Shapley value for feature iii quantifies its contribution to the model's prediction.
S: S⊆F represents a subset of features excluding feature i.
F: The full set of features used in the model for prediction.



E(f(x)): The base values of the model if we do not know any of the features
f(x): The predicted values of the model when we add all the features

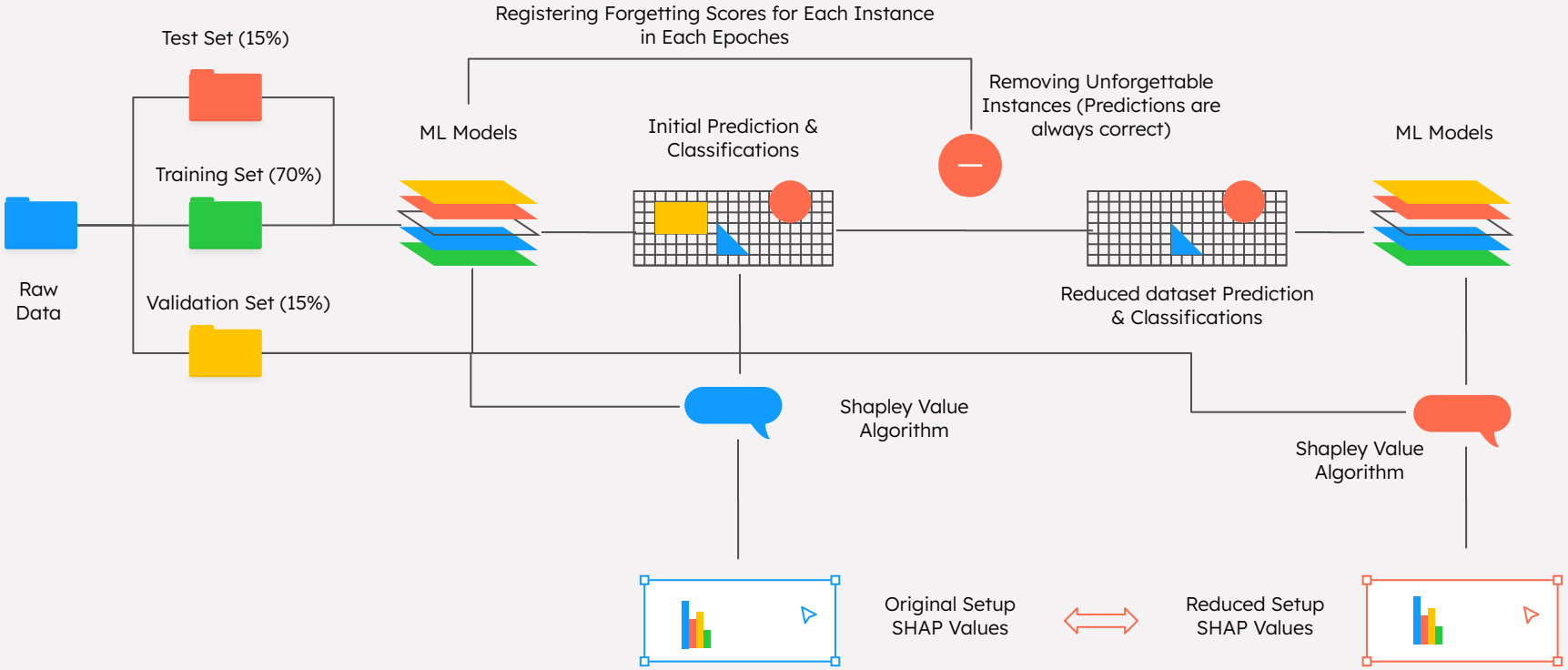
*Lundberg and Lee, 2017:
https://doi.org/10.48550/arXiv.1705.07874

Wine Quality Data

Features

- UCI ML Repository Public Dataset
- 11 Features
- ~6000 Examples
- Multivariate: 7 wine quality indicators from 3 to 9
- An imbalance dataset
- Subjective training label

Data source: Cortez et al., 2009: http://www.vinhoverde.pt/en/



Methodology

Dataset Comparison

Original Means		Reduced Means	
fixed_acidity	7.216071	fixed_acidity	7.225136
volatile_acidity	0.344049	volatile_acidity	0.343126
citric_acid	0.320307	citric_acid	0.320655
residual_sugar	5.033407	residual_sugar	5.017660
chlorides	0.057151	chlorides	0.057704
free_sulfur_dioxide	30.162295	free_sulfur_dioxide	30.098916
total_sulfur_dioxide	114.205910	total_sulfur_dioxide	113.851852
density	0.994534	density	0.994570
pH	3.223995	pH	3.225312
sulphates	0.535913	sulphates	0.537931
alcohol	10.557666	alcohol	10.552559

Results

ResNet
(Simplified)

- Simplified ResNet (Gorishniy, et. al., 2021) for tabular data

$$\begin{aligned}\text{ResNet}(x) &= \text{Prediction}(\text{ResNetBlock}(\dots(\text{ResNetBlock}(\text{Linear}(x))))) \\ \text{ResNetBlock}(x) &= x + \text{Dropout}(\text{Linear}(\text{Dropout}(\text{ReLU}(\text{Linear}(\text{BatchNorm}(x)))))) \\ \text{Prediction}(x) &= \text{Linear}(\text{ReLU}(\text{BatchNorm}(x)))\end{aligned}$$

- Using 10 epochs to find forgettable examples
- Maximum of 4 times forgetting events occur in certain examples
- Feature importance shifts when unforgettable examples were removed
- Original dataset has shown a feature is more important than any other features, but reduced dataset diminished this feature's importance to predictions

Original dataset modeling accuracy on validation set: 40%

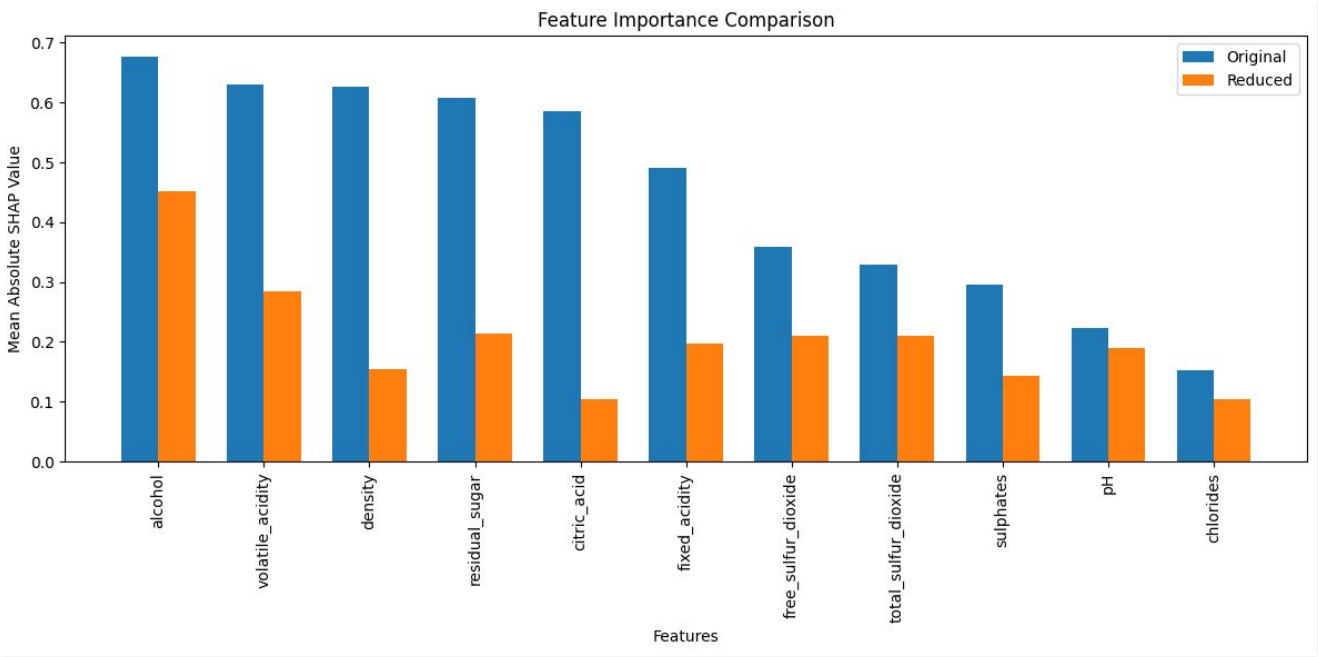
Reduced subset modeling accuracy on validation set: 50%

Prediction

Gorishniy, et. al., 2021: https://doi.org/10.48550/arXiv.2106.11959

Results

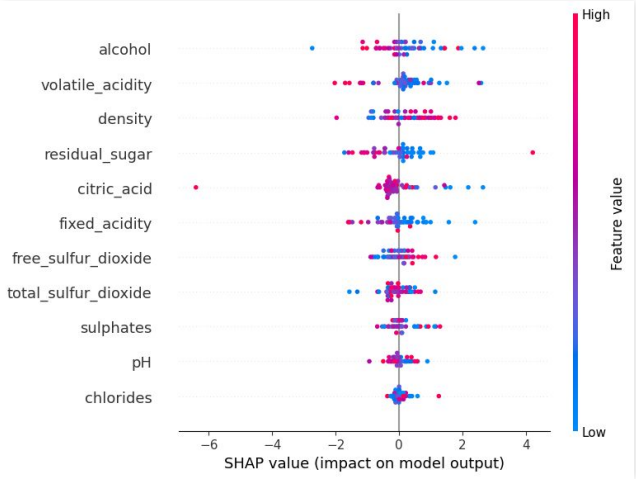
ResNet
(Simplified)



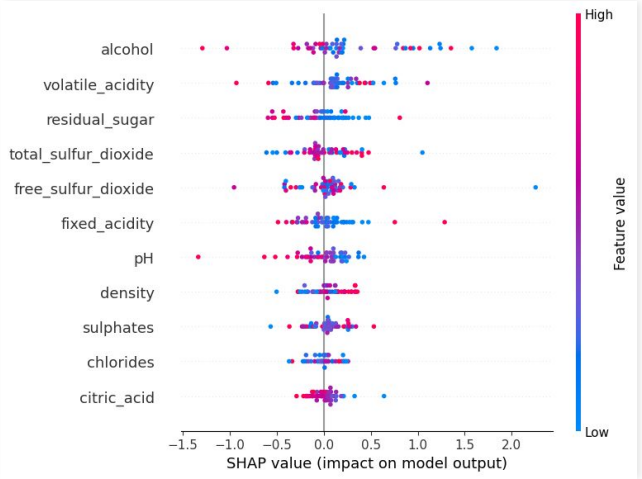
Results

ResNet
(Simplified)

Original Dataset



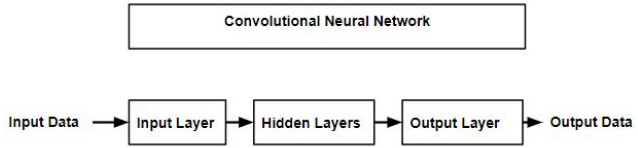
Reduced Dataset (Forgetting Algorithm)



Results

JTT with
1) CNN 2) Random Forest

- JTT section has used two models - CNN (Hinton, et. al., 2015) and Random Forest (Breiman, et. al., 2001) does not have iterations like deep learning, a simulated iteration is proposed.
  - CNN



  - Random Forest
    - *For each epoch:*
      - *Bootstrap a subset of dataset*
      - *Train a new Random Forest model on the bootstrap sample*
      - *Append the trained model to the list of models*

- Using a “pre-trained” with 5 iterations, upsample 5 times for minority group

Train a initial mode -> Generate predictions -> Upsample minorities -> Train Final Model

Original dataset modeling accuracy on validation set: **54.1% (CNN)** and **54.1% (Random Forest)**
Reduced subset modeling accuracy on validation set: **48.1% (CNN)** and **56.3%**

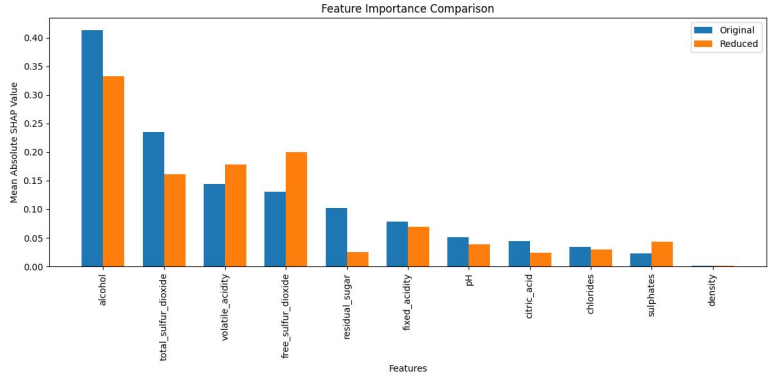
Both accuracy improves **~15%** for minority groups (label=4 &8)

Prediction

Breiman, et. al., 2001: https://link.springer.com/article/10.1023/A:1010933404324

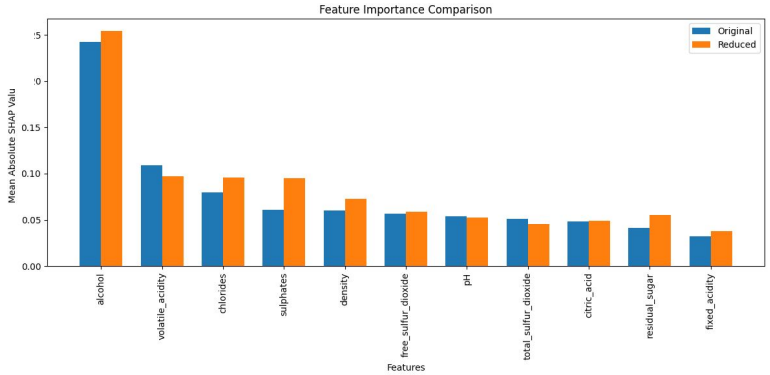
Results

JTT with
1) CNN 2) Random Forest



CNN Model - Feature Importance

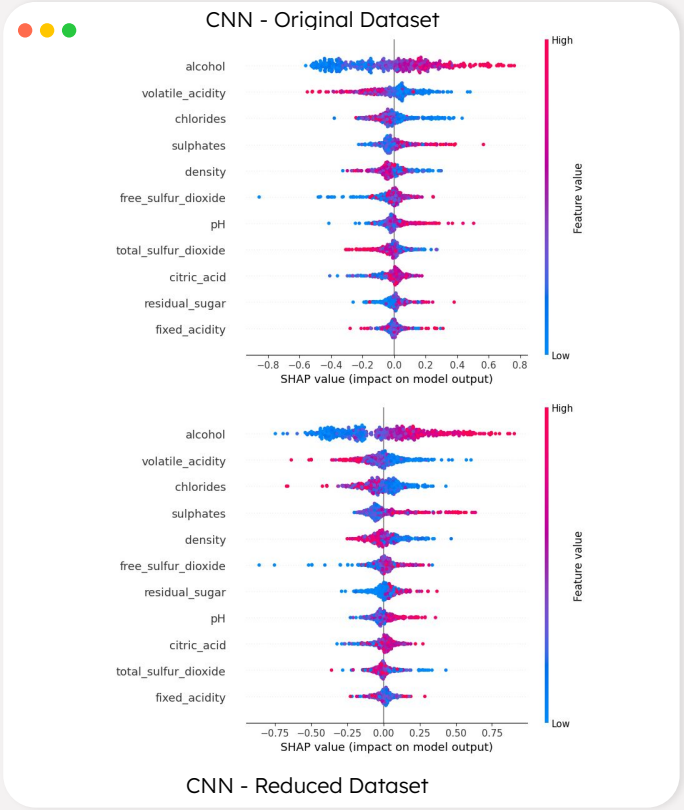
Random Forest Model - Feature Importance

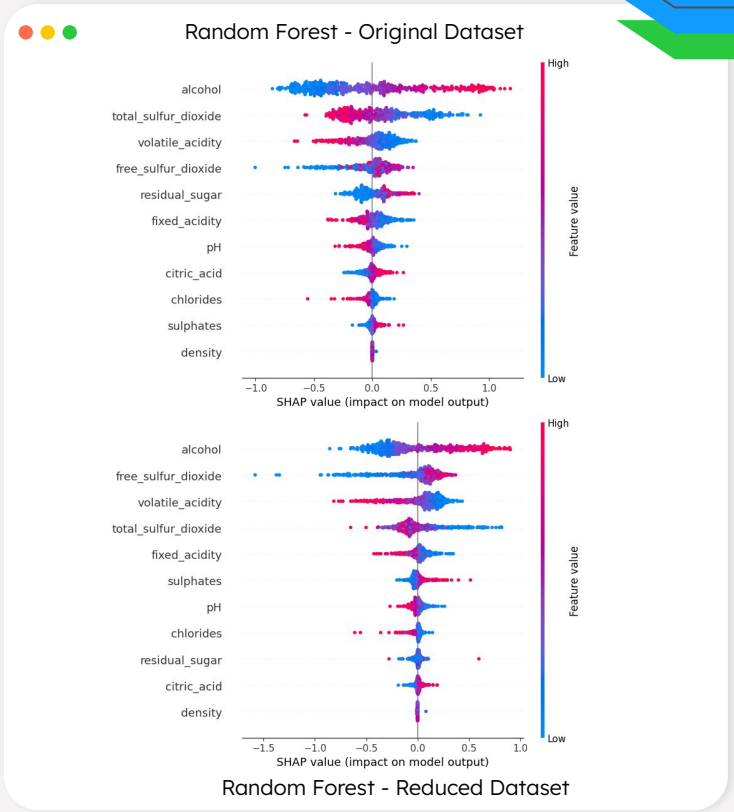


Breiman, et. al., 2001: https://link.springer.com/article/10.1023/A:1010933404324

Results

JTT with
1) CNN 2) Random Forest





Breiman, et. al., 2001: https://link.springer.com/article/10.1023/A:1010933404324

SVM Model Training

Resolving Model Imbalancement

- Technique Used: Synthetic Minority Over-Sampling Technique (SMOTE)
- Outcome:
  - Balanced dataset with equal instances for each class

Class Distribution After Applying SMOTE

Quality Score	Number of Samples
3	2,836
4	2,836
5	2,836
6	2,836
7	2,836
8	2,836
9	2,836

- **Algorithm Used:** Support Vector Machine (SVM) with RBF kernel.
- **Data Splitting:**
  - Training Set: 70%.
  - Validation Set: 15%.
  - Test Set: 15%.
- **Feature Scaling:** Standardization applied to all features.

Performance Metrics Comparison

SVM

Performance Metrics of SVM Models

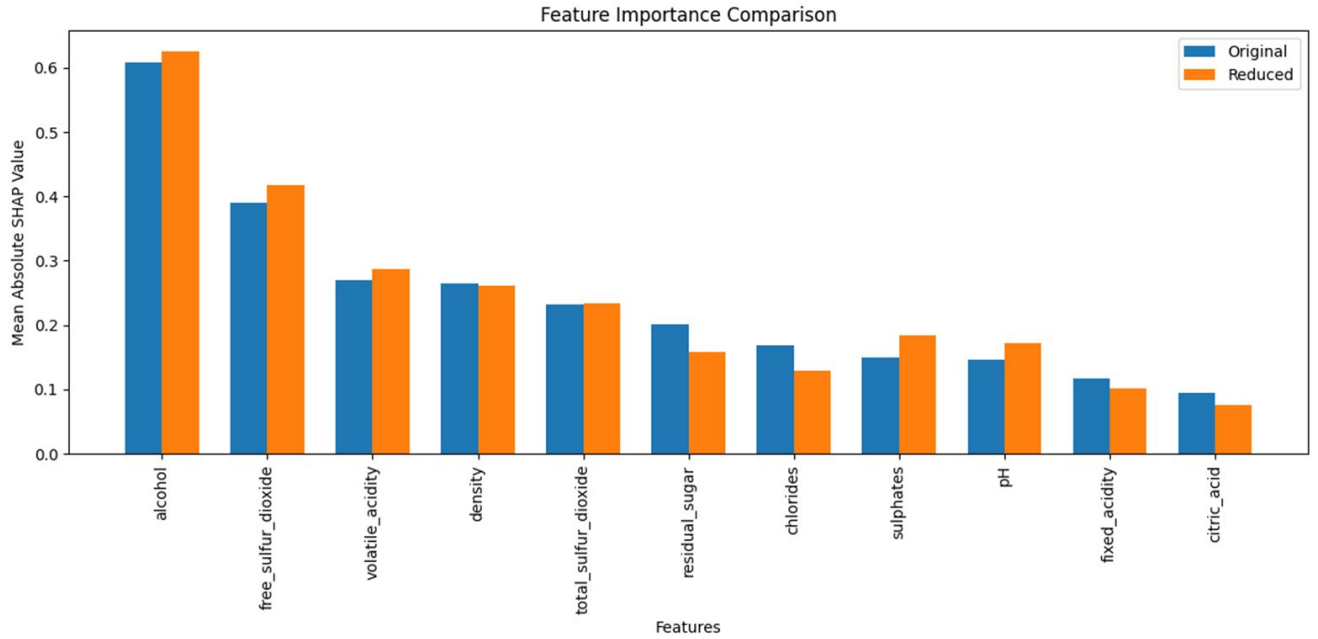
Metric	Original Validation	Reduced Validation	Original Test	Reduced Test
Accuracy	0.734	0.841	0.746	0.833
Precision	0.722	0.840	0.735	0.830
Recall	0.734	0.841	0.746	0.833
F1 Score	0.723	0.835	0.736	0.824

Key Findings:

- Improved performance on the reduced dataset
- Higher accuracy and F1 scores observed

Results

SVM

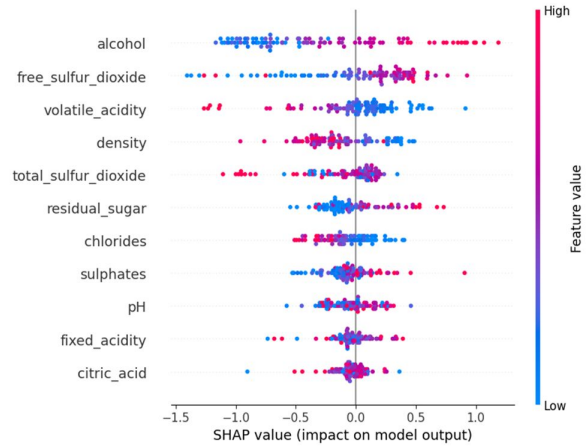


Breiman, et. al., 2001: https://link.springer.com/article/10.1023/A:1010933404324

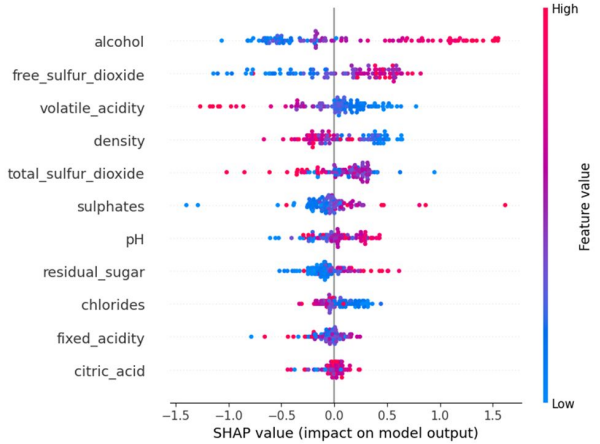
Results

SVM

Original Dataset



Reduced Dataset (Forgetting Algorithm)



Results

Simple NN

- Simple Neural Network Model

```
class LargerModel(nn.Module):
    def __init__(self, input_size, num_classes):
        super(LargerModel, self).__init__()
        self.fc1 = nn.Linear(input_size, 10)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(10, 10)
        self.fc3 = nn.Linear(10, 10)
        self.fc4 = nn.Linear(10, num_classes)
    def forward(self, x):
        x = self.fc1(x)
        x = self.relu(x)
        x = self.fc4(x)
        return x
```

Original dataset modeling accuracy on validation set: 40%

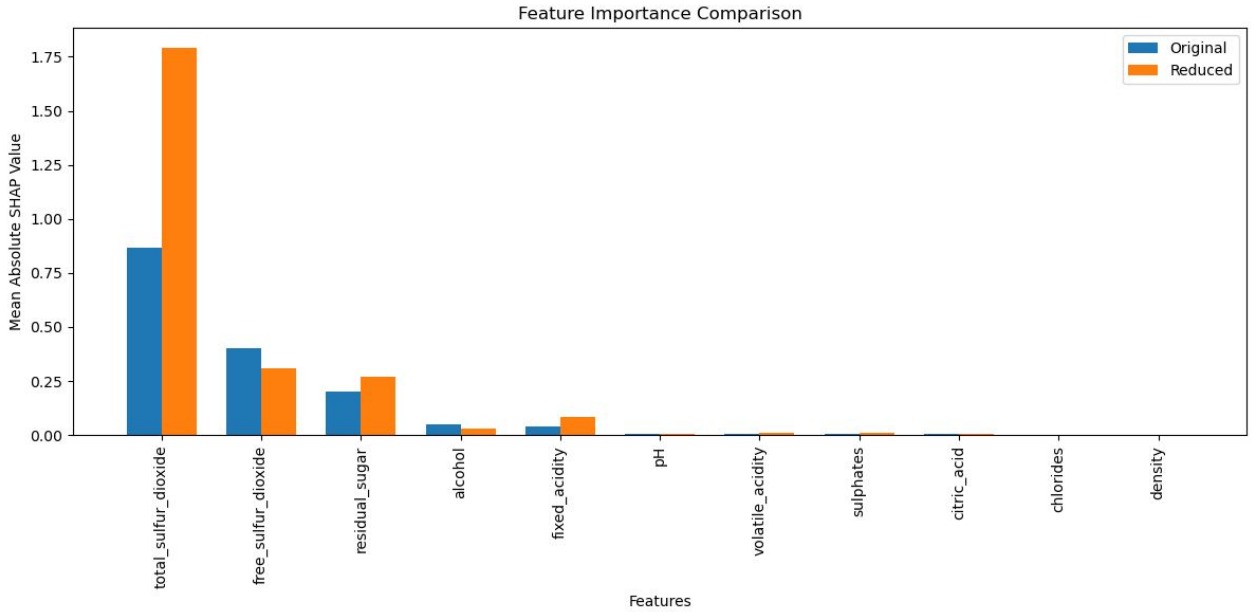
Reduced subset modeling accuracy on validation set: 50%

Prediction

Gorishniy, et. al., 2021: https://doi.org/10.48550/arXiv.2106.11959

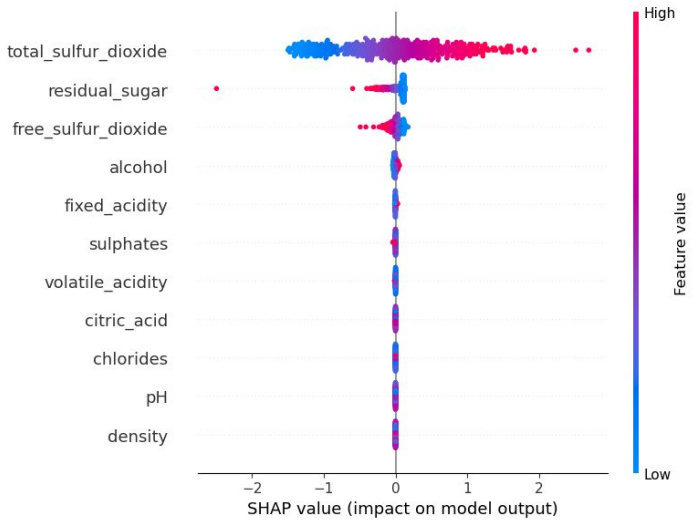
Results

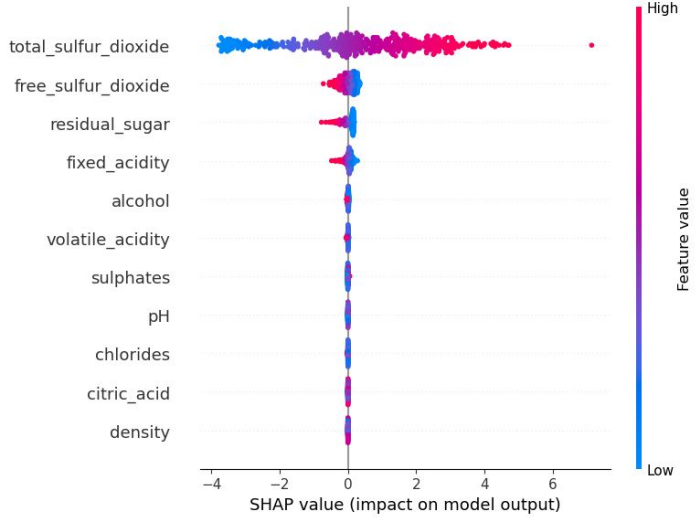
Simple NN



Results

Simple NN





Binary Classification

Simple NN

- Simple Neural Network Model

```
class LargerModel(nn.Module):
    def __init__(self, input_size, num_classes):
        super(LargerModel, self).__init__()
        self.fc1 = nn.Linear(input_size, 10)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(10, 10)
        self.fc3 = nn.Linear(10, 10)
        self.fc4 = nn.Linear(10, num_classes)
    def forward(self, x):
        x = self.fc1(x)
        x = self.relu(x)
        x = self.fc4(x)
        return x
```

Original dataset modeling accuracy on validation set: 56%

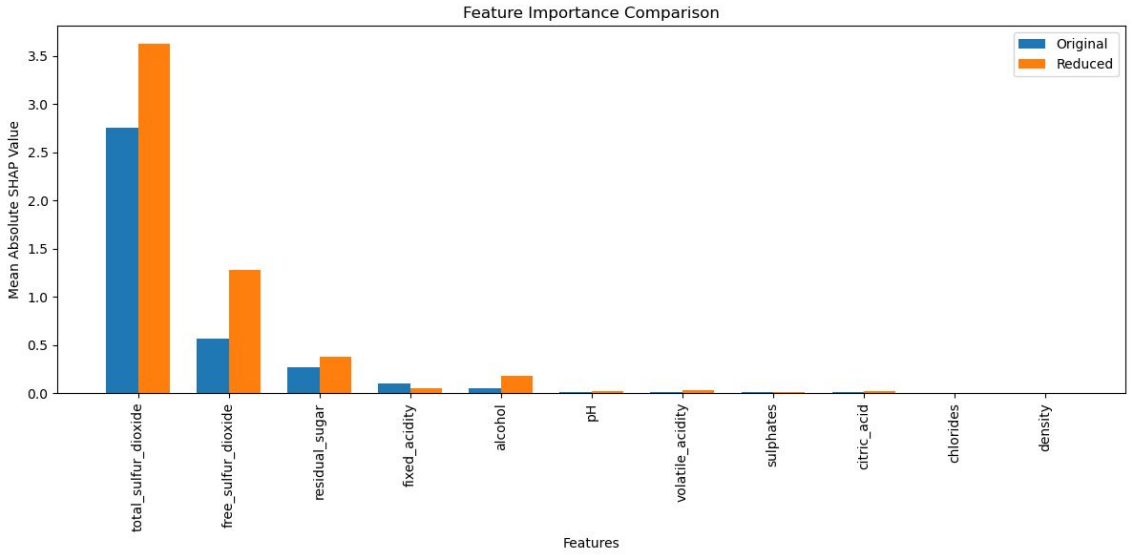
Reduced subset modeling accuracy on validation set: 64%

Prediction

Gorishniy, et. al., 2021: https://doi.org/10.48550/arXiv.2106.11959

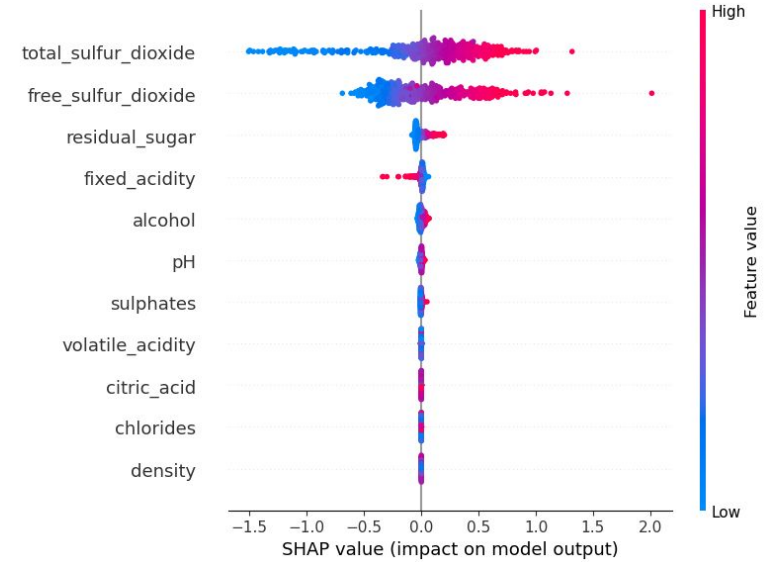
Results

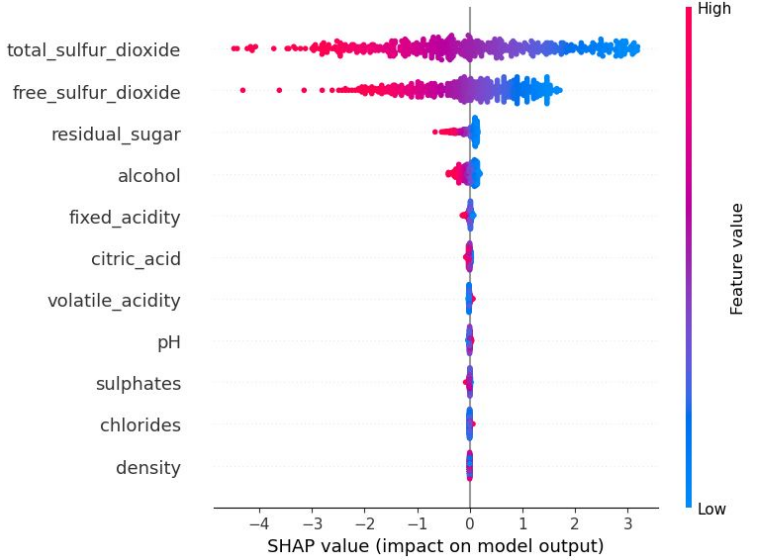
Simple NN



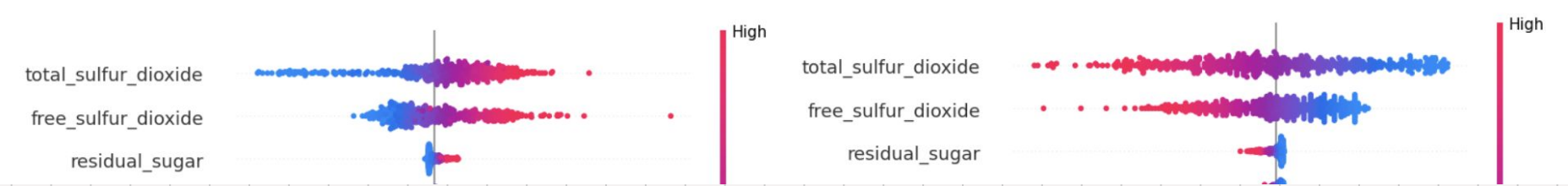
Results

Simple NN



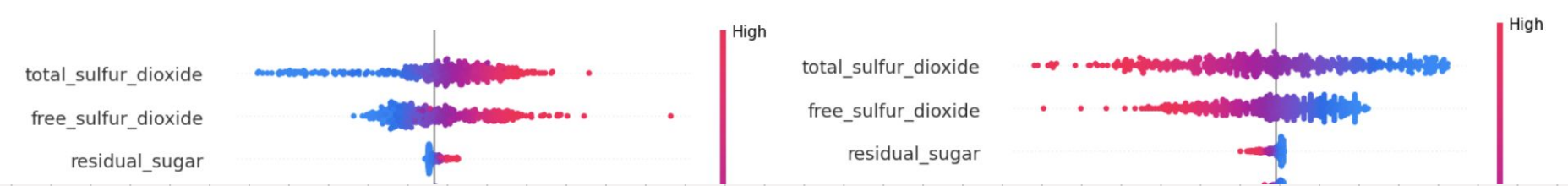


Example



- Red means the value was high, blue means it was low
- Left means the feature had a small importance, right means it had a high importance
- In both models the representation of these features were similar
  - Left model higher values are more important
  - Right model lower values are more important
- Could easily sway predictions one way or another

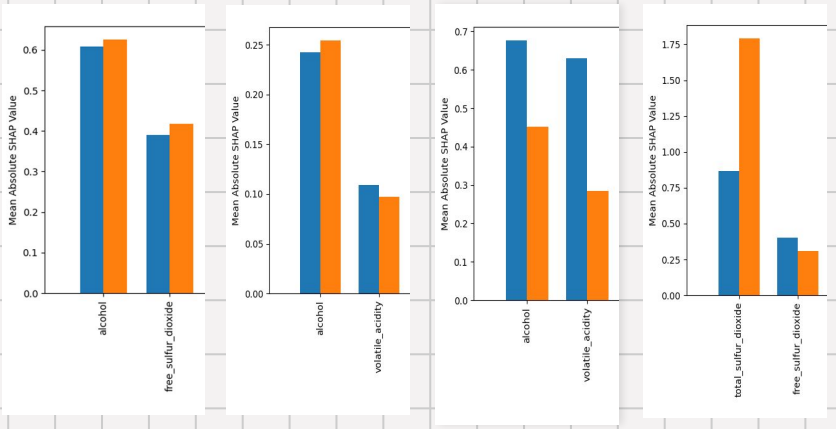
Why does this happen?



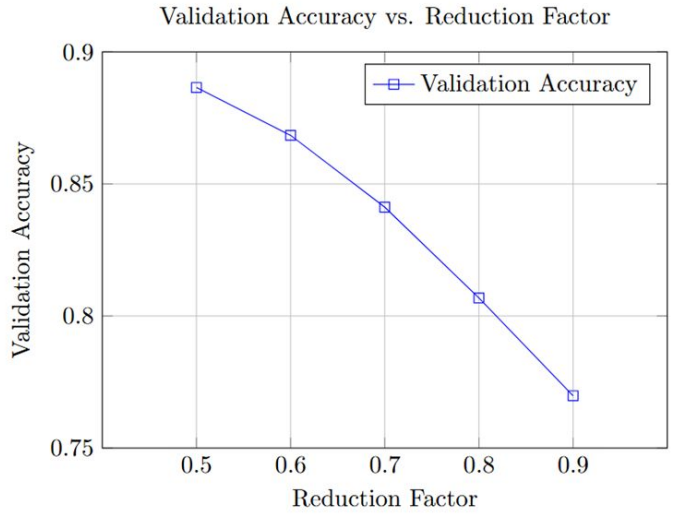
- There is no one answer
- Deleted noise from the data which could have swayed the original model
- Model found different patterns

Consistency of Top Important Features Across Models

- The relative ranking of features may shift slightly, but the top ones remain unchanged.
- Although different models may observe different top important features.
- This demonstrates that even after removing unforgettable examples, the dataset retains its core structure, ensuring stable feature importance rankings across models.



Exploring Different Reduction Factors (SVM)



Validation Accuracy vs. Reduction Factor

Conclusions

What happened?

- Differences vary heavily by model
- SHAP values (feature importance) may increase or diminish after reducing the dataset
- Top important features do not change across different models before and after reducing dataset

What does this teach us?

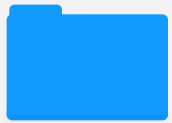
- Even the best and closest subset can vary your results
- Subset selection through the forgetting algorithm may affect interpretability unevenly across models.

Challenges

- Subjective dataset labeling may introduced inconsistency or bias into the model and feature importance
- Imbalanced dataset that made it hard to predict the minority groups

- Breiman, Leo. “Random Forests.” *Machine Learning*, vol. 45, no. 1, Oct. 2001, pp. 5–32. *Springer Link*, https://doi.org/10.1023/A:1010933404324.
- Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, *47*(4), 547–553. https://doi.org/10.1016/j.dss.2009.05.016
- Gorishniy, Y., Rubachev, I., Khrulkov, V., & Babenko, A. (2023). *Revisiting deep learning models for tabular data* (No. arXiv:2106.11959). arXiv. https://doi.org/10.48550/arXiv.2106.11959
- LeCun, Yann, et al. “Deep Learning.” *Nature*, vol. 521, no. 7553, May 2015, pp. 436–44. *DOI.org (Crossref)*, https://doi.org/10.1038/nature14539
- Lundberg, S., & Lee, S.-I. (2017). *A Unified Approach to Interpreting Model Predictions* (No. arXiv:1705.07874). arXiv. https://doi.org/10.48550/arXiv.1705.07874
- Toneva, M., Sordoni, A., Combes, R. T. des, Trischler, A., Bengio, Y., & Gordon, G. J. (2019). *An empirical study of example forgetting during deep neural network learning* (No. arXiv:1812.05159). arXiv. https://doi.org/10.48550/arXiv.1812.05159

References



Thank You