A Comprehensive Analysis and Model Evaluation of the Prediction of Graphics Processing Unit (GPU) Clock Speeds

By Luka Margiani, 3114323

Student of SRH University in Berlin

Supervisor: Kristian Rother

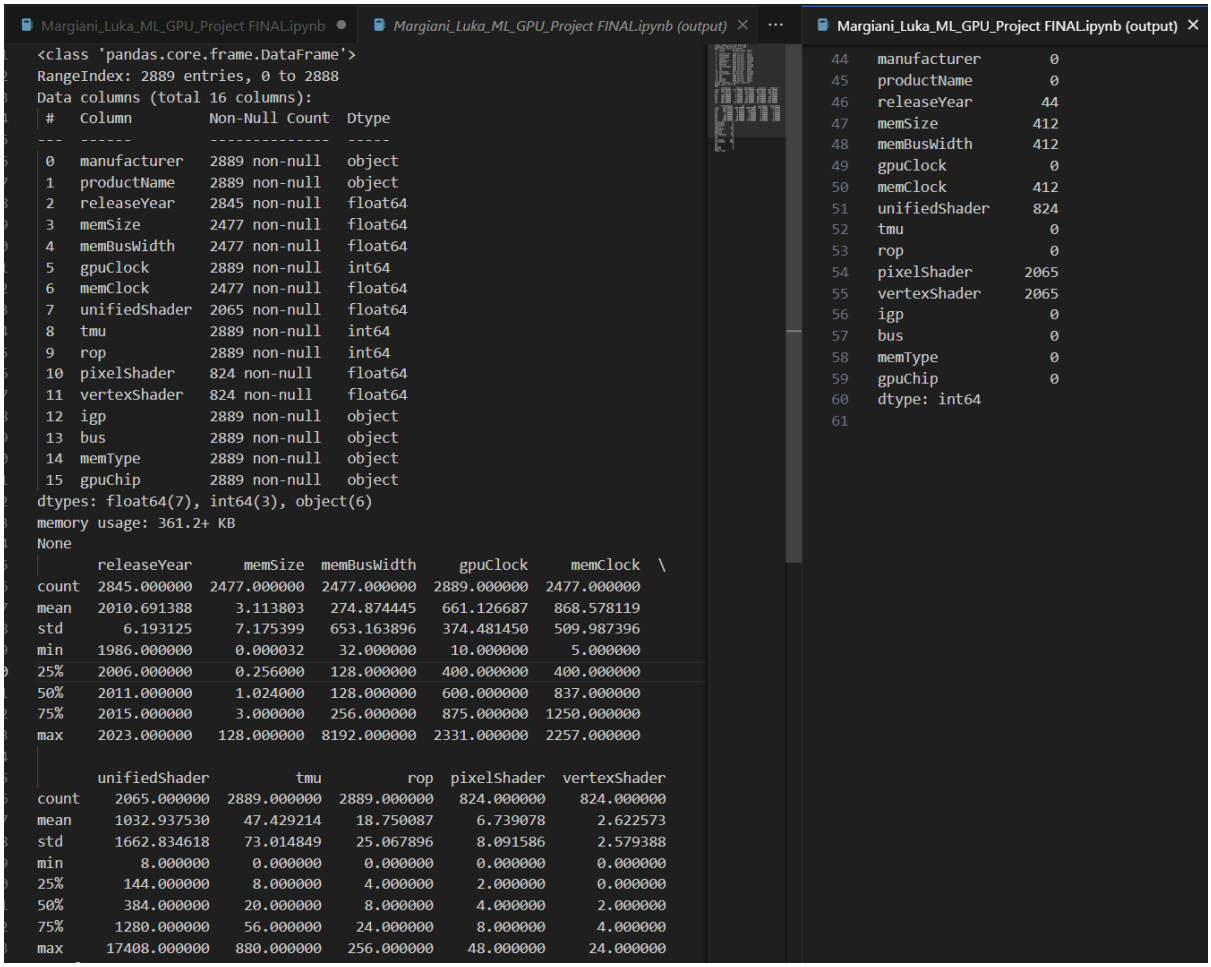
Table of Contents

Introduction

The objective of this project is to conduct a comprehensive analysis and modeling of GPU specifications in order to predict the GPU clock speed, a critical performance metric that significantly impacts overall GPU efficiency. The objective is to construct comprehensive predictive models based on a wide range of GPU characteristics and evaluate their performance in a rigorous manner. This report offers a comprehensive analysis of the methodologies employed in the project under consideration. The process commenced with data quality checks to ascertain the integrity and dependability of the dataset. Subsequently, a comprehensive analysis was undertaken to recognize any discernible patterns. Feature engineering techniques were employed to enhance the model's precision, while multiple models were constructed and evaluated. The dataset, titled "NVIDIA & AMD GPUs Full Specs", consists of a vast number of GPU specifications from a multitude of manufacturers. This dataset is characterized by a high level of detail and complexity, rendering it ideal for the development of sophisticated models and rigorous analytical inquiry.

Data Quality Checks

The dataset comprises 2,889 entries, which are organized into 16 columns comprising both numeric and categorical values. Upon preliminary examination, it became evident that the data exhibit several shortcomings pertaining to quality.



```
Margiani_Luka_ML_GPU_Project FINAL.ipynb ●   Margiani_Luka_ML_GPU_Project FINAL.ipynb (output) ×   ···
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2889 entries, 0 to 2888
Data columns (total 16 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   manufacturer   2889 non-null   object
 1   productName    2889 non-null   object
 2   releaseYear    2845 non-null   float64
 3   memSize        2477 non-null   float64
 4   memBusWidth    2477 non-null   float64
 5   gpuClock       2889 non-null   int64
 6   memClock       2477 non-null   float64
 7   unifiedShader  2065 non-null   float64
 8   tmu            2889 non-null   int64
 9   rop            2889 non-null   int64
 10  pixelShader    824 non-null    float64
 11  vertexShader   824 non-null    float64
 12  igp            2889 non-null   object
 13  bus            2889 non-null   object
 14  memType        2889 non-null   object
 15  gpuChip        2889 non-null   object
dtypes: float64(7), int64(3), object(6)
memory usage: 361.2+ KB
None
       releaseYear      memSize  memBusWidth     gpuClock     memClock  \
count  2845.000000  2477.000000  2477.000000  2889.000000  2477.000000
mean   2010.691388     3.113803   274.874445   661.126687   868.578119
std       6.193125     7.175399   653.163896   374.481450   509.987396
min    1986.000000     0.000032    32.000000    10.000000     5.000000
25%    2006.000000     0.256000   128.000000   400.000000   400.000000
50%    2011.000000     1.024000   128.000000   600.000000   837.000000
75%    2015.000000     3.000000   256.000000   875.000000  1250.000000
max    2023.000000   128.000000  8192.000000  2331.000000  2257.000000

       unifiedShader          tmu          rop  pixelShader  vertexShader
count    2065.000000  2889.000000  2889.000000   824.000000    824.000000
mean     1032.937530    47.429214    18.750087     6.739078      2.622573
std      1662.834618    73.014849    25.067896     8.091586      2.579388
min         8.000000     0.000000     0.000000     0.000000      0.000000
25%       144.000000     8.000000     4.000000     2.000000      0.000000
50%       384.000000    20.000000     8.000000     4.000000      2.000000
75%      1280.000000    56.000000    24.000000     8.000000      4.000000
max     17408.000000   880.000000   256.000000    48.000000     24.000000
```

```
Margiani_Luka_ML_GPU_Project FINAL.ipynb (output) ×
manufacturer        0
productName         0
releaseYear        44
memSize           412
memBusWidth       412
gpuClock            0
memClock          412
unifiedShader     824
tmu                 0
rop                 0
pixelShader      2065
vertexShader     2065
igp                 0
bus                 0
memType             0
gpuChip             0
dtype: int64
```

Fig 1. The Output of the Data Quality Checks

The dataset exhibits a notable prevalence of missing values. A significant portion of the dataset is devoid of values, particularly in columns such as releaseYear, memSize, memBusWidth, memClock, unifiedShader, pixelShader, and vertexShader. The missing values were addressed through the implementation of an appropriate imputation methodology.

Additionally, submissions that were determined to be redundant were excluded from further consideration. Furthermore, an investigation was conducted to ascertain the data types and formatting of the dataset. While the data types were correctly identified, inconsistencies were observed in the numeric columns. For instance, non-null floating-point

values were identified in the releaseYear column; however, the GPU clock column was expressed as an integer.

Consequently, it was imperative to guarantee that the data were accurately formatted and of the appropriate type to enable precise analysis. Furthermore, during the preliminary examination, outliers and extreme values were identified. It is noteworthy that the memSizeMB, memToClockRatio, and shaderDensity features exhibited extreme values and infinities. These anomalies were subsequently addressed to ensure the model's robust performance. In order to contextualize the aforementioned observations, it is necessary to provide a brief overview of the data set under examination.

Data Exploration

The objective of the data exploration phase was to gain an understanding of the data set's structure and the relationships between its features. To this end, the data set was comprehensively analyzed. This section presents a discussion of the key aspects of the exploration phase, including descriptive statistics, correlation analysis, and various visualizations. The latter provide insights into the data.

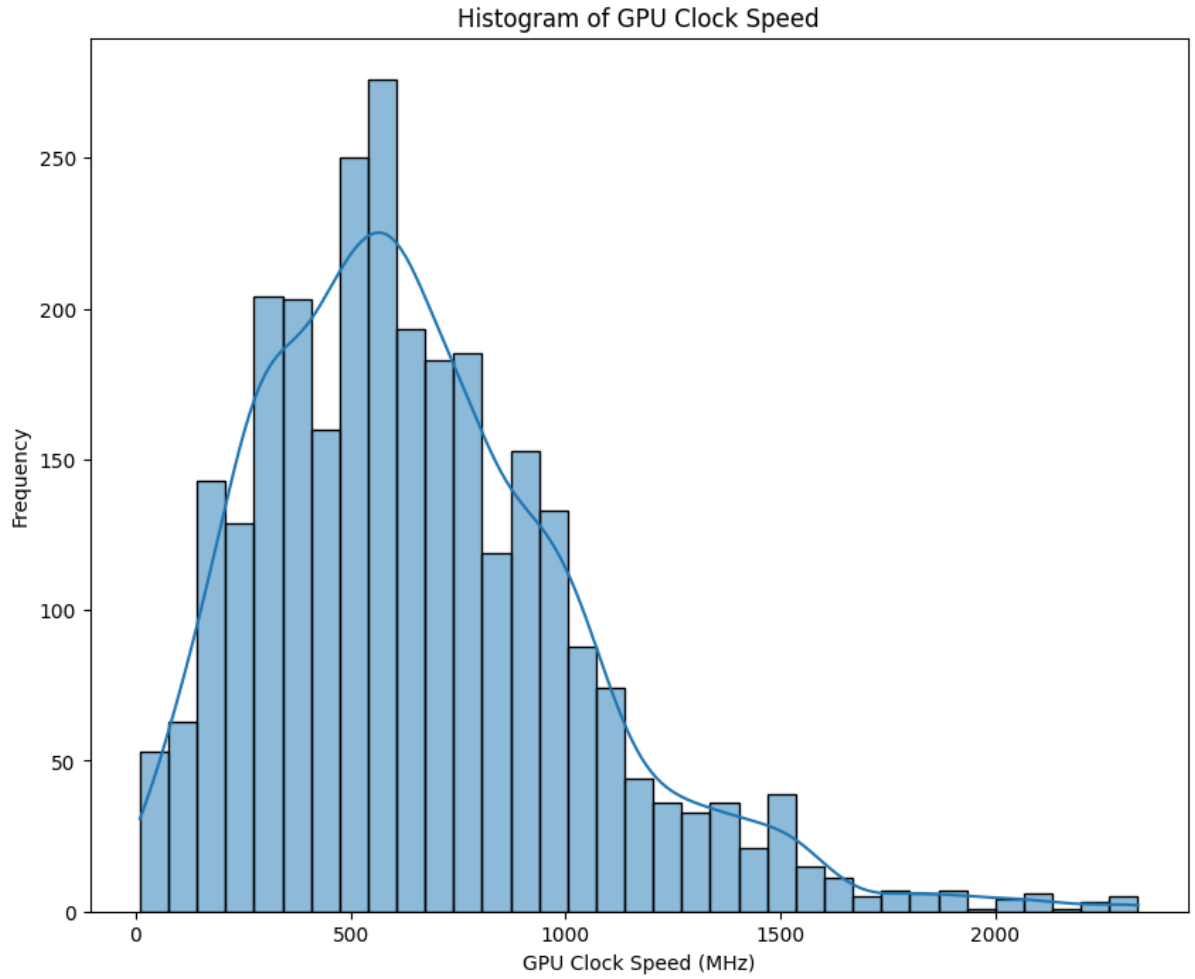


Fig 2. GPU Clock Speed Histogram

The histogram of GPU clock speed provides a visual representation of the frequency distribution of clock speeds across a variety of GPUs within the dataset. This plot reveals the range and common values of GPU clock speeds, highlighting any skewness or outliers. From the histogram, it can be observed that GPU clock speeds tend to cluster around certain values, with a significant number of GPUs having clock speeds between 400 and 900 MHz. This distribution provides insight into the typical performance range of GPUs and allows for the identification of any unusual values that may require further investigation.

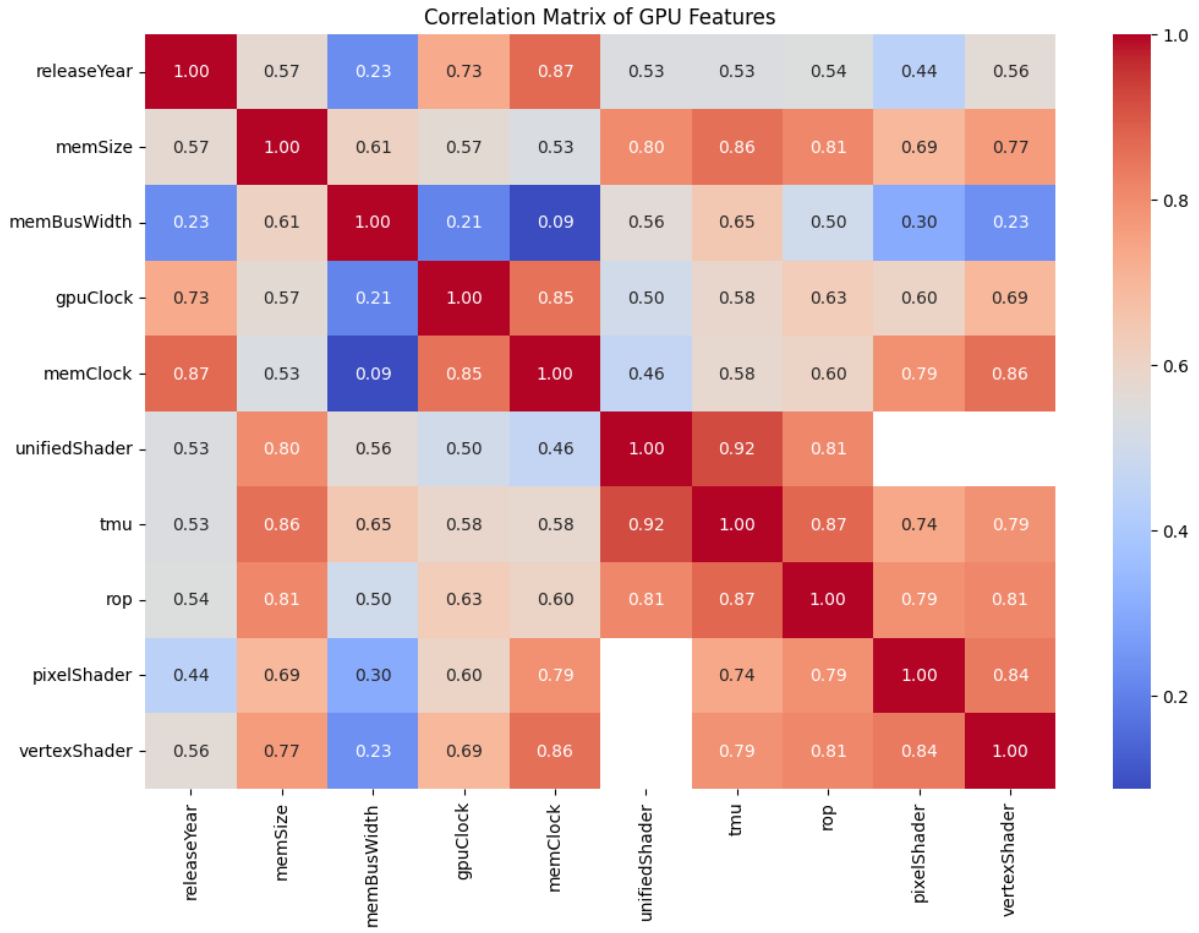


Fig 3. The Correlation Matrix of GPU Features

The correlation matrix presents the correlation coefficients between pairs of numeric features within the dataset. This matrix is instrumental in identifying strong correlations that can inform feature selection and engineering. For instance, a high positive correlation between memory size and GPU clock speed suggests that these features are closely associated and should be integrated into the modeling process. Vice-versa, features displaying low or negative correlations might contribute distinctive information, which can be crucial for enhancing model accuracy. This analysis is pivotal for constructing a resilient predictive model by leveraging the most relevant features.

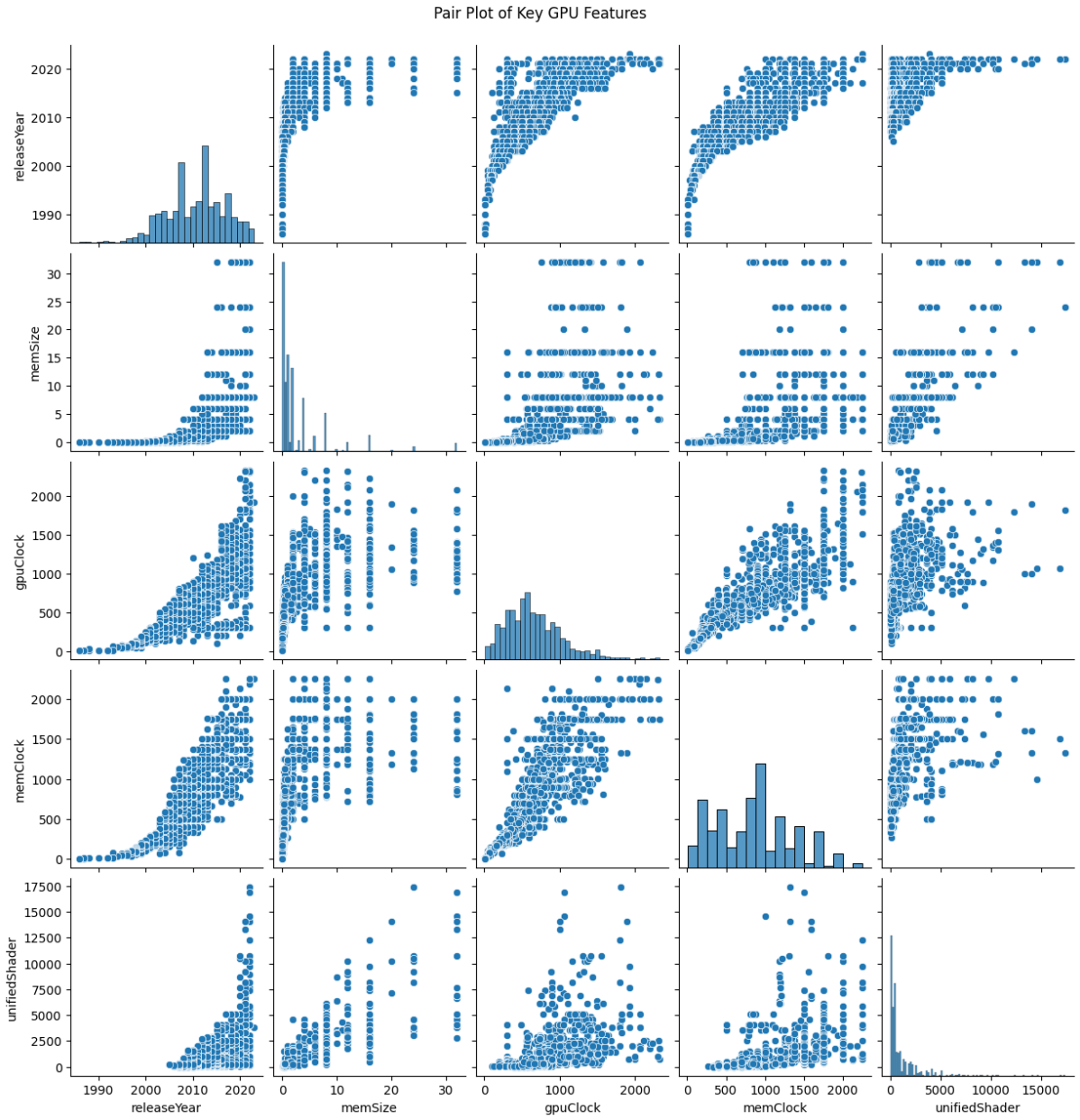


Fig 4. Pair Plot of Key GPU Features

The pair plot offers an extensive overview of the interdependencies between a multitude of pivotal GPU characteristics, encompassing memory size, memory clock speed, GPU clock speed, and unified shaders. This multi-dimensional visualization enables the detection of pairwise relationships and potential interactions between features. For instance, the pair plot can elucidate whether certain features exhibit linear correlation or intricate, non-linear relationships. Grasping these interactions is of paramount importance for effective feature engineering and the selection of optimal modeling techniques capable of encapsulating such relationships.

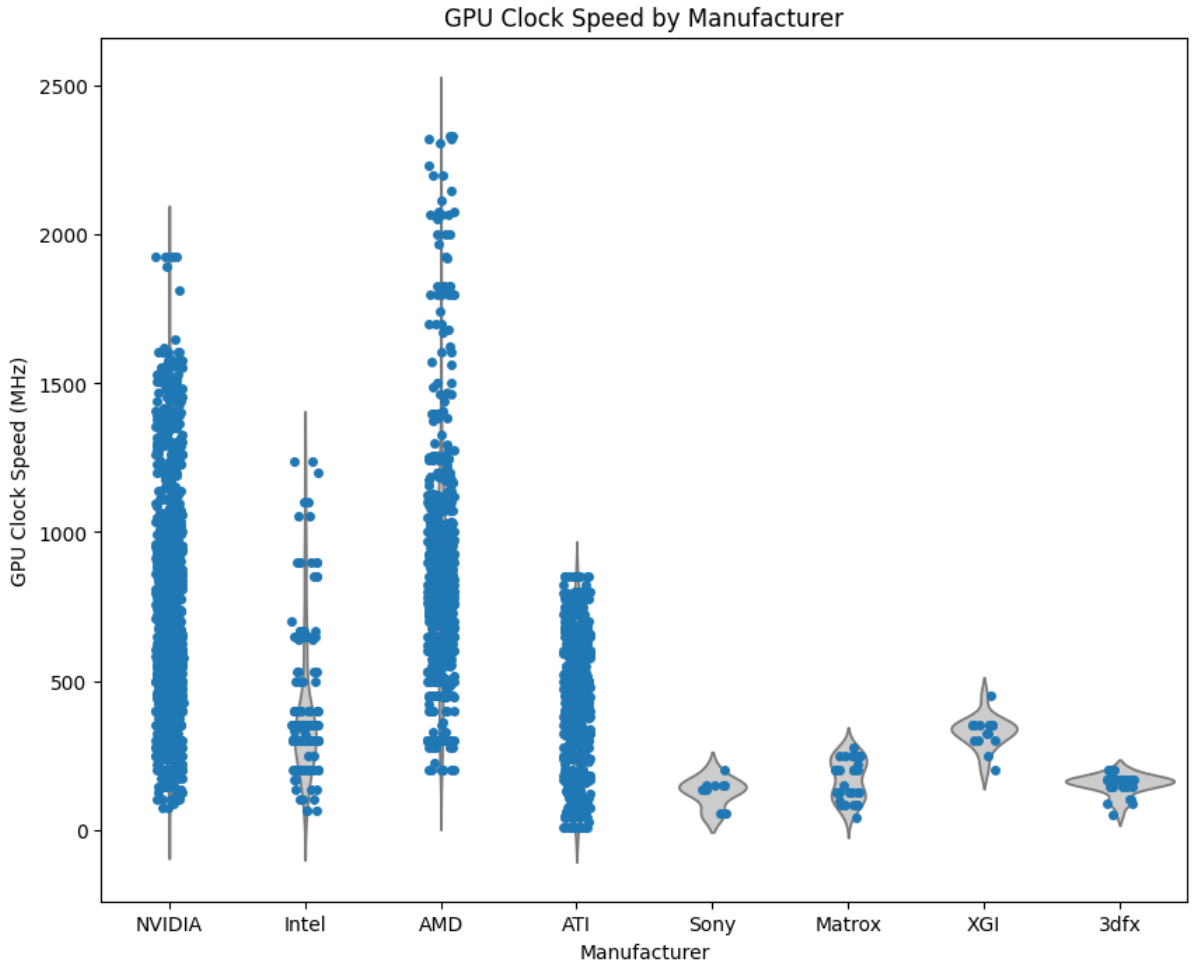


Fig 5. GPU Clock Speed by Manufacturer

The violin plot and strip plot of GPU clock speed by manufacturer illustrate the distribution of clock speeds across different GPU manufacturers. The utilization of these plots facilitates a comparative analysis of the performance characteristics exhibited by GPUs from disparate brands. The violin plot is a combination of a box plot and a kernel density plot, which provides a comprehensive view of the data distribution, including the probability density of the data at different values. The strip plot provides a more detailed representation of the data by overlaying individual data points, thus enhancing the precision of the analysis and facilitating the identification of potential outliers.

It is evident that manufacturers such as NVIDIA and AMD offer a plethora of GPU clock speeds, indicative of their diverse product lines that cater to varying performance requirements. On the other hand, some manufacturers may exhibit a more concentrated range of clock speeds. This analysis is crucial for comprehending manufacturer-specific performance trends and can inform targeted feature engineering and model customization.

The use of violin plots and strip plots is a logical approach, as they provide a comprehensive summary and detailed view of the data, facilitating a more profound understanding of the distribution and dispersion of GPU clock speeds across manufacturers.

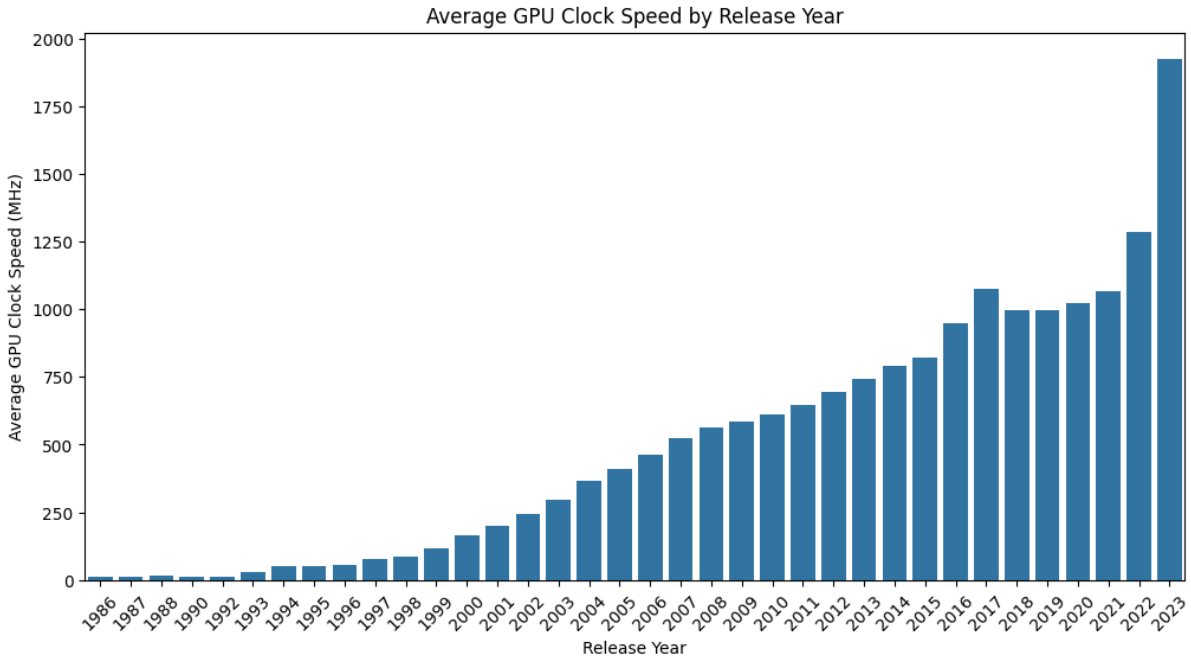


Fig 6. Average GPU Clock Speed by Release Year

A line plot of average GPU clock speed by release year illustrates the trajectory of GPU clock speeds over time. This plot clarifies the influence of technological advancements on GPU performance, demonstrating a general upward trend indicative of the advancement of newer GPUs with higher clock speeds, which reflect the ongoing evolution of GPU technology. Temporal analysis is pivotal for comprehending the evolution of GPU performance and can inform future projections by incorporating the year of release as a pivotal feature in the modeling process.

Fig 7. Memory Size vs. GPU Clock Speed

The scatter plot of memory size versus GPU clock speed elucidates the interrelationship between these two pivotal GPU characteristics. Each data point on the plot represents a discrete GPU, with the memory size represented on the x-axis and the clock speed represented on the y-axis. The plot facilitates the identification of trends and potential correlations between memory size and clock speed.

Nevertheless, the derivation of a positive trend from this plot is challenging due to the presence of extreme outliers and the highly skewed distribution of memory sizes. The presence of outliers can obscure the underlying relationship, thereby impeding the accurate interpretation of the data.

Despite these challenges, a general positive trend can be discerned, indicating that GPUs with larger memory sizes tend to have higher clock speeds. This relationship is intuitive, as an increase in memory can enhance GPU performance. However, the extreme values make this trend less apparent.

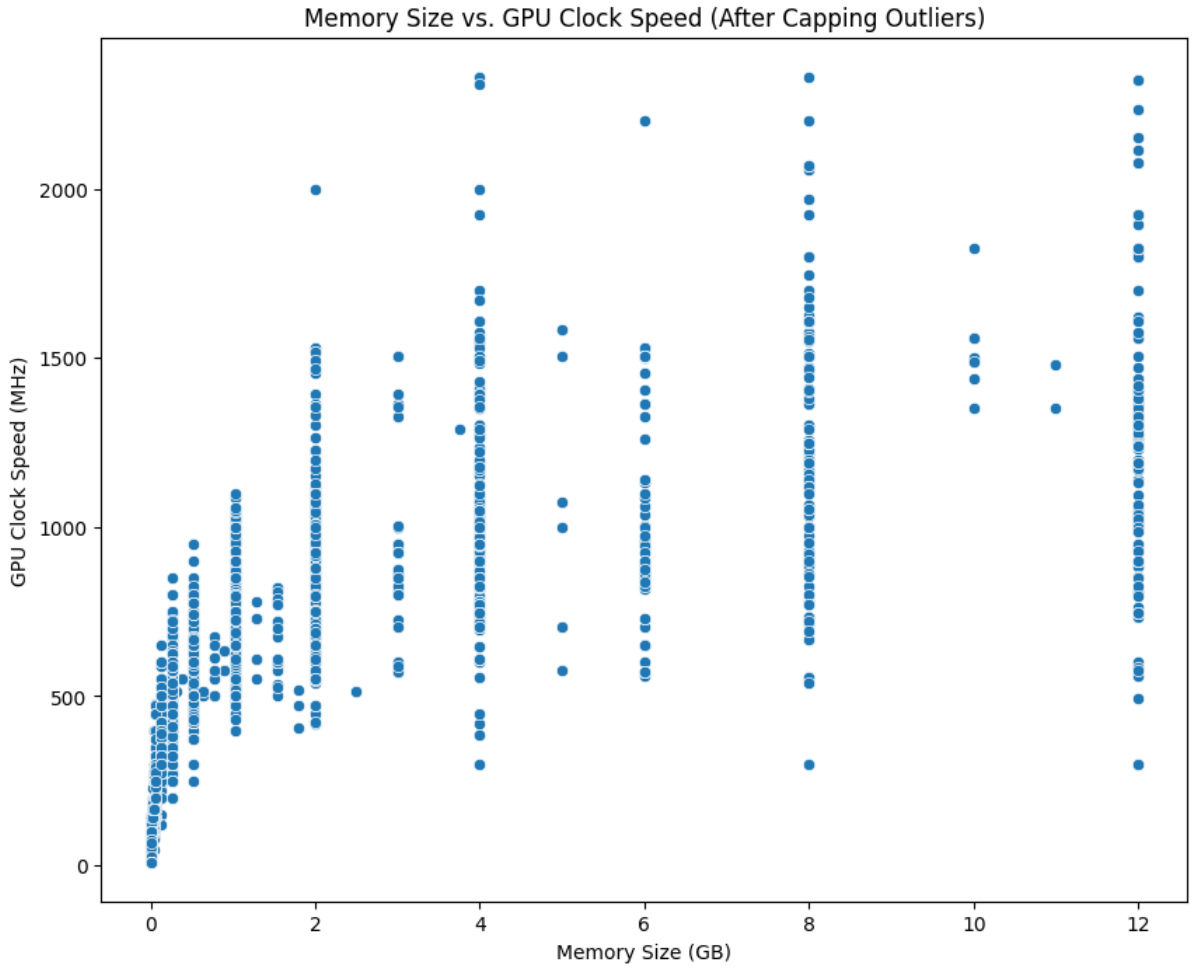


Fig 8. Memory Size vs. GPU Clock Speed after capping outliers

In order to address the influence of extreme outliers, a scatter plot of memory size versus GPU clock speed was created, with outliers capped. The modified plot offers a more accurate representation of the typical relationship between these two features, as it reduces the distortion caused by extreme values.

The capped scatter plot maintains the positive trend observed earlier, while providing a more compact and interpretable range of values. Although the trend is becoming discernible, the data remains highly skewed, which continues to complicate the analysis. This refined view facilitates more accurate assessment of the relationship between memory size and clock speed, thereby enabling enhanced model training and prediction accuracy.

By capping the outliers, we ensure that extreme values do not unduly influence the model, allowing for a more realistic understanding of the relationship between memory size and GPU clock speed.

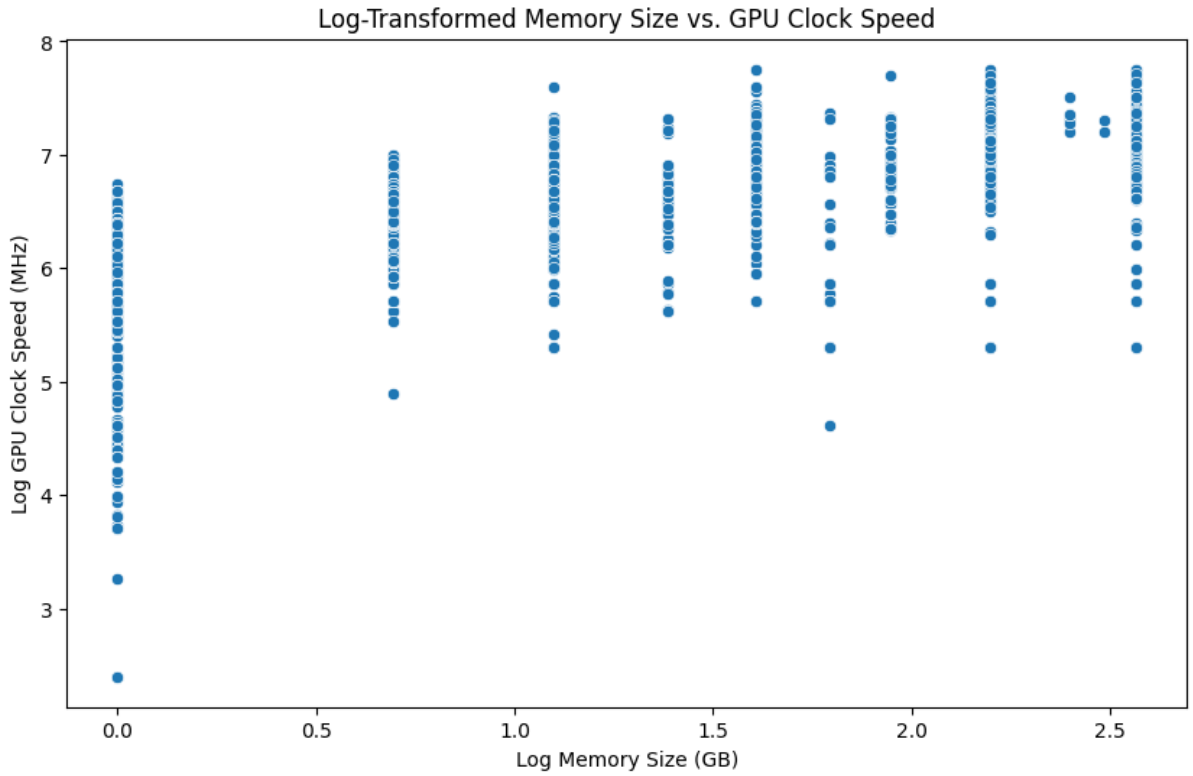


Fig 9. Log-Transformed Memory Size vs. GPU Clock Speed

The scatter plot of log-transformed memory size versus GPU clock speed provides a detailed and nuanced representation of the relationship between these two significant GPU features. The application of a logarithmic transformation to the memory size addresses the issue of skewness and outliers that are prevalent in the original scale. This transformation reduces the range of memory sizes, thereby enhancing the symmetry and facilitating the analysis of the data distribution.

The log-transformed scatter plot provides a more accurate and linear representation of the relationship between memory size and GPU clock speed. This linearity is advantageous for modeling purposes, as linear models and numerous other machine learning algorithms demonstrate superior performance when applied to features that exhibit linear relationships.

The plot demonstrates that as the logarithm of memory size increases, there is a corresponding increase in GPU clock speed, indicating that GPUs with larger memory sizes tend to have higher clock speeds. The aforementioned transformation thus serves the function of stabilizing the variance present within the data set and achieving a more homoscedastic distribution, which is of paramount importance for the reliable fitting of a model.

The transformation of memory size to its logarithmic form also facilitates the detection of subtle patterns and interactions that might be hidden in the original scale, thereby improving the overall quality of the predictive model and enhancing the comprehensiveness of the plot.

Feature Engineering

The process of feature engineering entailed the creation and transformation of features with the objective of enhancing the performance of a model.

The key steps involved the handling of missing values, the creation of new features, the encoding of categorical variables, and the scaling of features. In the context of handling missing values, the KNN imputation method was deemed to be more effective than median imputation. The use of KNN imputation is preferable as it considers the similarity between instances, thereby facilitating more accurate estimations based on the values of neighboring data points. This method was employed to address the absence of values in numerical columns, including memSize, memBusWidth, and memClock.

In the case of categorical columns, missing values were addressed through the application of mode imputation or the removal of the affected values. The creation of new features, including memSizeMB (converted from memSize), memToClockRatio, and shaderDensity, afforded further insights into GPU performance. These derived features were calculated with the objective of enhancing the model's predictive power. Categorical variables, including manufacturer, IGP, bus, memory type, and GPU chip, were one-hot encoded to convert them into a format suitable for machine learning models. The transformation of categorical values into binary features enabled the models to utilize these attributes effectively.

Feature scaling was employed to ensure that numerical values were on a comparable scale, which is crucial for algorithms sensitive to feature magnitude, such as linear regression.

These feature engineering steps were designed with the objective of enhancing the model's ability to learn from the data and improving prediction accuracy.

Training a model

Two machine learning models were trained on the dataset:

1. Linear Regression: model was selected as a baseline due to its simplicity and interpretability. It assumed a linear relationship between the features and the target variable (gpuClock). The model was trained and evaluated using metrics such as mean squared error (MSE), mean absolute error (MAE), and R^2 score.
2. Random Forest Regressor: As an advanced model, the Random Forest Regressor was used to capture complex, non-linear relationships between features and the target variable. This ensemble model combines multiple decision trees to improve prediction accuracy and robustness.

Model Training Results

- *Linear Regression:*
    - *Mean Squared Error (MSE): 294,449.54*
    - *Mean Absolute Error (MAE): 316.54*
    - *R^2 Score: -1.04*
- *Random Forest Regressor:*
    - *Mean Squared Error (MSE): 6405.48*
    - *Mean Absolute Error (MAE): 44.37*
    - *R^2 Score: 0.96*

To ascertain the efficacy of the models in predicting unseen data, cross-validation scores were evaluated. The Linear Regression model exhibited negative R^2 scores across cross-validation folds, indicating a poor fit and potential overfitting. Conversely, the Random Forest Regressor achieved an R^2 score of 0.96 on average, thereby demonstrating its superior predictive performance.

Model evaluation

The efficacy and precision of the models were assessed based on a number of criteria. To assess the accuracy of the models, metrics such as mean squared error (MSE), mean absolute error (MAE), and R^2 score were employed. The Random Forest Regressor demonstrated superior performance compared to the Linear Regression model in these areas. The low MSE and MAE values observed for the Random Forest model indicate that it made accurate predictions with minimal error, thereby providing evidence of its reliability in predicting GPU clock speeds. Furthermore, the high R^2 Score of the Random Forest model indicates that it effectively explained a substantial proportion of the variance in the target variable, thereby providing additional evidence of its efficacy.

Another essential evaluation criterion was cross-validation. The cross-validation results for the random forest model demonstrated consistent strong performance and generalizability. The consistency across different subsets of data suggests that the random forest model is robust and can be expected to perform well on new and unseen data. In contrast, the cross-validation scores for the linear regression model were consistently negative, indicating that it was unable to adequately capture the complexities of the dataset. This discrepancy highlights the necessity of employing more sophisticated models, such as random forests, for datasets exhibiting intricate relationships between features.

Linear Regression Model Cross-Validation Scores:

[-0.71, -2.70, -1.17, -3.51, -4.82]

Mean CV Score: -2.58

Random Forest Regressor Model Cross-Validation Scores:

[0.50, 0.76, 0.72, 0.60, 0.74]

Mean CV Score: 0.67

The feature importance analysis provided by the Random Forest Regressor offered a further level of insight. This analysis facilitated the identification of the features that exerted the greatest influence on GPU clock speeds. The comprehension of feature importance proved invaluable for further analysis and feature selection. It guided subsequent iterations of the model and enhanced its accuracy. Additionally, by underscoring the pivotal predictors of GPU clock speed, this analysis provided insights that could inform tangible decisions in GPU design and optimization.

Conclusion

This project demonstrated the application of supervised learning techniques to predict the clock speeds of graphics processing units (GPUs) based on a variety of specifications. Through a series of rigorous data quality checks, exploratory data analysis, and feature engineering, the dataset was prepared for modeling. The Random Forest Regressor was identified as the superior model, exhibiting superior accuracy and predictive capability compared to the Linear Regression model. The comprehensive analysis and visualisations provided a deeper understanding of the relationships between GPU features and clock speeds, offering valuable insights for both model improvement and practical applications.

Further work could involve the refinement of feature selection, the exploration of other advanced models, and the incorporation of additional data with a view to enhancing prediction accuracy. The findings of this project establish a robust foundation for the understanding of GPU performance metrics and the utilisation of machine learning to facilitate data-driven predictions.

References

1. *https://www.kaggle.com/datasets/alanjo/graphics-card-full-specs?resource=download*
2. *https://www.academis.eu/machine_learning/index.html*