Automated Adaptive Outlier Handling

Final Project, Tabular Data Science Course (89-547), March 12th 2025

Yoav Yoscovich, Bar-Ilan University

Abstract— **Outliers may significantly degrade the performance of a model, leading to inaccurate predictions; Throughout the years, many methods have been suggested to address this issue, but the choice of method was discretionary and dependent on the dataset and therefore leaves room for improvement. In this project, I offer an automated adaptive outlier handling method, in order to assist performing an easier data science process and achieving better model performance. I explored *combining various techniques* learned throughout the course, into *new enhanced methods* with less weaknesses, achieving better performance. Overall, I will introduce *three* sub-algorithms, and the *"Yoscovich Algorithm (Data-driven Predicted Best of Three Algorithm)"*, which combines the three sub-algorithms, by analyzing the nature of each dataset beforehand, utilizing a new metric I have developed (*Normalized Variance*) and choosing the most appropriate sub-algorithm for handling outliers (rather than a naive brute-force algorithm, running all sub-algorithms and choosing the best-performing one). As part of the experimental evaluation, a new evaluation metric (*Performance Improvement Metric - PIM*) is introduced, combining several evaluation metrics; a significant mean PIM improvement (beyond 50%) is shown relative to baseline approaches of outlier-handling, for four different datasets.**

1. Problem Description

The concept of *outliers* was first defined by *Grubbs* in 1969 as ***"one that appears to deviate markedly from other members of the sample in which it occurs"***[1]. As we have learned in class, outliers may significantly degrade the performance of a model, leading to inaccurate predictions and reduced generalization to new data; the importance of this problem can also be learned from the alternate name for outliers, given by *Wainer* - "*contaminants*"[2]. Throughout the years, many methods have been suggested to address this issue, but method suitability was specific for each scenario or dataset and this leaves room for improvement; As the use of ML models has become prevalent in our lives, mishandled outliers have evolved to be more disturbing and might even have life-threatening consequences; for example, as autonomous vehicles become more common, the issue of outliers presence has been widely discussed: ***"these outliers can cause errors [...] resulting in accidents and fatalities"***[3]. Apart from such *"physical"* usages of ML, outliers may increase *signal variance* and reduce the *power of statistical tests* performed during analysis, destroy *signal normality*, introduce *fat tails* or significantly *bias regression analysis*[4]. Therefore, it is crucial to detect and handle outliers present in our dataset. Furthermore, **manually identifying outliers is time-consuming and prone to errors**; different datasets, with different data distributions, often require testing different outlier-detection methods. **Therefore, this element in the DS pipeline requires improvement**, in terms of accuracy and complexity. In addition to developing algorithms for outlier-handling, I aim to introduce an adaptive *data-driven* algorithm, which will determine which of the three sub-algorithms introduced as part of this paper is best to apply for each dataset, based on the nature of each dataset, and not by just trying various algorithms in a *brute-force* manner. This will help to perform *EDA (Exploratory Data Analysis)* more conveniently and accurately, efficiently detecting outliers using adaptive techniques. In addition, careful handling of outliers may improve the model's *explainability*, making it easier to explain the predictions and behavior of the model.

It is important to note that the idea for this project, as well as the proposed solution, are connected to materials learned in class. Namely, the idea stemmed from difficulties we observed during the lectures throughout the semester, when significant outliers were present in the data; the issue of outliers was already addressed in the first lecture of the course, when ***Kernel Density Estimation (KDE) Plot*** was discussed – the meaning of a long tail was explained, and we briefly discussed different methods to address it. Furthermore, when we discussed the usage of ***KS-Test*** (during lecture #7), it was mentioned that it is advised not to rely on this test alone, as it is highly affected by outliers (especially when present at the edges of the data). In addition, when ***Q-Q Plot*** was discussed (during lecture #4), it was advised to try the ***"q"*** option, which specifies the quantiles of the data we want to compare to the quantiles of a theoretical distribution, in order to better understand how outliers affect the distribution.

2. Solution Overview

Overall, this research mainly concerns exploring whether ***combining** various techniques*, which we have learned throughout the course, may yield *new enhanced methods*, having less weaknesses and better performance. Overall, this section presents ***three* sub-algorithms** I have developed using material learned in class (each sub-algorithm combines several techniques, and uses optimized parameters): (*i*) **sub-algorithm 1: *Box-Cox + Z-Score***; (*ii*) **sub-algorithm 2: *Adaptive IQR Multiplier***; (*iii*) **sub-algorithm 3: *Outlier Capping + Z-Score***. I will also present the ***"Yoscovich Algorithm 4 (Data-driven Predicted Best of Three Algorithm)"***, which is a combination of these sub-algorithms, but **chooses the right sub-algorithm according to data analysis**; it is not a naive brute-force algorithm, simply running all sub-algorithms and choosing the best-performing one, but this *data-driven* algorithm analyzes the nature of each dataset beforehand, and chooses the most suitable sub-algorithm for handling the outliers.

Before elaborating on this algorithm and the three sub-algorithms, it is important to note that **the ideas for the methods used for detecting and handling outliers are *highly based* on materials learned in class**, as will be outlined later on; for example, the ***Box-Cox transformation***- class lecture #5, ***skewness*** (3rd moment) and ***kurtosis*** (4th moment) - class lecture #5, the ***IQR (interquartile range)*** and ***Z-Score*** for detecting outliers, which were also discussed in class; overall, I will combine techniques discussed in class in order to implement an automated adaptive outlier handling tool, resulting in a more efficient data science process, and enhancing model's performance.

2.1 *Sub-Algorithm 1: Box-Cox + Z-Score*

The idea for this solution stems from a combination of techniques and concepts learned throughout the course. During lecture #5, we have learned about the ***Box-Cox* transformation**, a statistical technique for stabilizing variance, and **approximate the data into a *normal distribution***. In addition, a useful outlier-detection method mentioned in class, is the ***Z-Score* method** - a statistical technique used to identify outliers in a dataset, by measuring *distance of data points to the mean of the dataset, in terms of standard deviations*, using threshold of $\pm 3\sigma$.

However, as mentioned in class, **the calculation of Z-score assumes that the data is normally distributed**, and therefore the effectiveness of the Z-score method for non-normal distributed data may severely deteriorate. Therefore, I thought that **combining these two methods (*Box-Cox* and *Z-Score* for outlier detection)** may yield a new and better method which is suitable even when the data is not normally distributed. In order to make the idea clearer, following is a *pseudo-code* implementation:

Algorithm 1: Box-Cox+Z-Score (Sub-Alg' 1)

```
Data: Dataset df, target variable targetVar.
Result: Outlier-handled data-frame.
1  $data \leftarrow \texttt{df}[targetVar]$
2  if $min(data) \leq 0$ then
3      $shift \leftarrow |min(data)| + 1$ ;   → Positive shifting, Box-Cox prerequisite
4      $\texttt{df}[targetVar] += shift$
5  else
6      $shift \leftarrow 0$
7  $[\texttt{df}[boxCox],\ \boldsymbol{\lambda}] \leftarrow$ Box-Cox($\texttt{df}[targetVar]$)
8  $Z-Scores \leftarrow$ Z-Score($\texttt{df}[boxCox]$)
9  $\texttt{df} \leftarrow \texttt{df}[|Z-Scores| < 3]$ ;   → Applying Z-Score removal after Box-Cox
10 $\texttt{df}[targetVar] \leftarrow (\texttt{df}[boxCox] \times \lambda + 1)^{\frac{1}{\lambda}}$
11 $\texttt{df}[targetVar] -= shift$
12 return df
```

2.2 *Sub-Algorithm 2: Adaptive IQR Multiplier*

A very useful method for outlier detection is using *IQR, the interquartile range*, defined as: $IQR = Q_3 - Q_1$ (middle 50% of the data). Finding outliers by IQR was suggested by *J. Tukey* [5] as observations that are outside of $[Q_1 - 1.5IQR, Q_3 + 1.5IQR]$. **The IQR multiplier of 1.5, introduced by Tukey as a *standard heuristic value* [5], is widely used for various cases** and indeed very useful to detect outliers (I successfully used it to filter out outliers in a class project of an android application database).

During lecture #5 in this course, we have learned about distribution moments, and specifically the 3rd and 4th standardized moments (***skewness*** and ***kurtosis***); Following that, I wondered whether these moments can be used to adapt Tukey's IQR heuristic multiplier. **While the standard IQR multiplier (1.5) works well for symmetrical distributions with standard tails**, for other distributions (e.g. *highly skewed* distribution or *fat-tail* distribution), it might not be enough to capture the extremes effectively. Therefore, one of the prominent ideas in this project is **formalizing an adapted formula for the *IQR multiplier*, based on different calculations**, tailored for each specific dataset.

While the 1.5 multiplier is not enough, it can serve as a baseline value on top of additional factors further tuning it according to specific characteristics of the dataset distribution. Following is the formula I have devised:

$$\boxed{\begin{aligned}\textbf{iqrMul} \leftarrow 1.5 + \frac{2}{3} \times (|\mathbf{S}| + max(\mathbf{K} - 3, 0) + max(\mathbf{MMR} - 1.5, 0) \\ + max(\textbf{tailSpread} - 3, 0) + max(\textbf{outlierRate} - 5\%, 0))\end{aligned}}$$

The formula uses Tukey's 1.5 factor as the baseline, while the additional factors make the IQR multiplier *tailor-made* for each specific dataset:

1) ***Skewness*** (3rd moment): asymmetric tails (high absolute value of skeweness for both right and left tails) require increasing the IQR multiplier to take into account more data at the tails of the distribution.
2) ***Kurtosis*** (4th moment): an absolute kurtosis beyond 3 (above normal distribution) is related to a *heavy tail* which requires to increase the IQR multiplier in order to take into account the data at the heavy tail.
3) ***Mean-to-Median Ratio (MMR)***: an additional metric to detect heavily skewed data having value above 1.5 (whereas for a normal distribution the value is 1.0).
4) ***Tail-spread***: measures the extent of the outer tail by the ratio of the percentile range [1%,99%] and the IQR (50% of the data). For a normal distribution, an approximate value of 3 is expected, and therefore a higher value increases the IQR multiplier.
5) ***Outlier-Rate***: *proportion* of outliers within the dataset. For a normal distribution, it should be $\sim$2%; a threshold of 5% is used for extreme cases.

These factors are all positive and **scaled with a factor of** $\frac{2}{3}$, found as a value close to be optimal according to a search performed over several datasets. Following is a comparison of the R^2 metric for the *House Prices dataset* (the dataset used in class) for different values of multipliers:

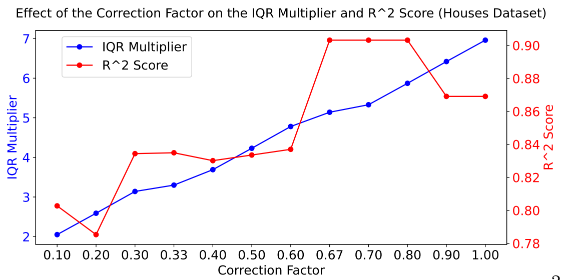

Fig. 1. Effect of correction factor on IQR multiplier and R^2 score

This figure above shows both the *IQR Multiplier* that is used and the resulting R^2 *score* as a function of the correction factor; as can be observed, **the *optimum value* for the correction factor is around 0.7**. ***Therefore, a correction factor of*** $\frac{2}{3}$ ***is chosen***. Similar optimization on other datasets showed that $\frac{2}{3}$ is *consistently* close to the optimal factor. In addition, the calculated multiplier is capped between 0.5 and 5 to avoid extreme values. Following is the relevant *pseudo-code*:

```
Algorithm 2: Adaptive IQR (Sub-Alg' 2)
  Data: Dataset df, target variable targetVar.
  Result: Outlier-handled data-frame.
1  $data \leftarrow \texttt{df}[targetVar]$
2  $\mathbf{S} \leftarrow Skewness(data)$; $\mathbf{K} \leftarrow Kurtosis(data)$
3  $\mu \leftarrow Mean(data)$; $median \leftarrow Median(data)$
4  $\textbf{MMRatio} \leftarrow \frac{\mu}{Median}$ (if $Median \neq 0$ else 1)
5  $q_1, q_3 \leftarrow percentile(data, 25, 75)$
6  $IQR \leftarrow q_3 - q_1$
7  $p_1, p_{99} \leftarrow percentile(data, 1, 99)$
8  $\textbf{tailSpread} \leftarrow \frac{p_{99} - p_1}{IQR}$ (if $IQR \neq 0$ else 0)
9  $lowerBound \leftarrow q_1 - 1.5 \times IQR$
10 $upperBound \leftarrow q_3 + 1.5 \times IQR$
11 $outliers \leftarrow \sum_{\substack{data<lowerBound \textbf{ OR} \\ data>upperBound}} (data)$
12 $\textbf{outlierRate} \leftarrow \frac{outliers}{Count(data)}$
13 $\textbf{iqrMul} \leftarrow 1.5 + \frac{2}{3} \times (|\mathbf{S}| + max(\mathbf{K} - 3, 0) + max(\textbf{MMRatio} - 1.5, 0) + max(\textbf{tailSpread} - 3, 0) + max(\textbf{outlierRate} - 5\%, 0))$
14 $\textbf{cappedIqrMul} \leftarrow max(0.5, min(iqrMul, 5))$
15 $adjLowerBound \leftarrow q_1 - \textbf{cappedIqrMul} \times IQR$
16 $adjUpperBound \leftarrow q_3 + \textbf{cappedIqrMul} \times IQR$
17 return $\texttt{df}[targetVar \geq adjLowerBound$
18         AND targetVar $\leq adjUpperBound]$
```

2.3 *Sub-Algorithm 3: Outlier Capping + Z-Score*

The 3rd sub-algorithm is different from the two previous sub-algorithms as it performs outlier ***Capping*** instead of removal; ***Capping*** replaces *extreme values* with predefined *threshold values*, instead of removing them. As outlined by *Anushree*: ***"Capping is the process of converting outliers into data which you don't consider as outliers, by simply deciding a boundary value in the data lying beyond the boundary is taken inside the boundary"***[6]. Common capping known as ***"Winsorization"***, a term proposed by *Wilfrid Joseph Dixon* in 1960 to commemorate *Charles P. Winsor*, who first suggested the basis for the method[7], uses percentile thresholds (e.g. 1% and 99%) as thresholds. Practically, this method caps any values outside these percentiles to the corresponding value of the percentile. I will not use the *regular Winsorization* method, as I find it quite limited due to use of the same values for *outlier detection* and *value for capping*. Alternatively, the following sub-algorithm still uses the 1% and 99% for the *capped value* but uses the standard *Z-score method* for *outlier detection*.

In general, sometimes outliers are a result of *true variability* and should not be completely removed (e.g. for a salary dataset, a top executive salary should not be removed but can be capped in order to reduce its bias). Extreme values may indicate *important trends* rather than errors, and removing them may sabotage the model, and therefore capping is more appropriate in such case. Following is a *pseudo-code* for this sub-algorithm:

Algorithm 3: Capping + Z-Score (Sub-Alg' 3)

```
Data: Dataset df, target variable targetVar.
Result: Outlier-handled data-frame.
1 $data \leftarrow$ df$[targetVar]$
2 $Z-Scores \leftarrow$ Z-Score$(data)$
3 $p_1, p_{99} \leftarrow percentile(data, 1, 99)$
4 df$[|Z-Scores| > 3] \leftarrow$ Clip(df$[|Z-Scores| > 3]$, $p_1$, $p_{99}$)
5 return df
```

2.4 *The Full (Adaptive) Algorithm*

Following is the *pseudo-code* for the full algorithm:

Algorithm 4: Yoscovich Algorithm (Data-driven Predicted Best of Three Algorithm)

```
Data: Dataset df, target variable targetVar.
Result: Outlier-handled data-frame.
1 $data \leftarrow$ df$[targetVar]$
2 $S \leftarrow Skewness(data)$
3 if $S > 3$ then
      // Skewed data: Box-Cox+Z-Score
4     return Sub-Alg 1 (df, $targetVar$)
5 else
6     $var \leftarrow Variance(data)$; $\mu \leftarrow Mean(data)$
7     $NormalizedVariance \leftarrow \begin{cases} \frac{var}{\mu^2} & \text{if } \mu \neq 0 \\ 0 & \text{else} \end{cases}$
8     if $NormalizedVariance < \frac{1}{2}$ then
          // Extreme values are rare:
             Adaptive IQR Multiplier
9         return Sub-Alg 2 (df, $targetVar$)
10    else
          // Extreme values are quite
             frequent: Capping+Z-Score
11        return Sub-Alg 3 (df, $targetVar$)
```

This *data-driven* algorithm analyzes the nature of each specific dataset, and decides accordingly which of the sub-algorithms should be applied in each case; it is important to note that unlike a naive brute-force algorithm running all sub-algorithms and choosing the one which has best-performed, **this data-driven algorithm analyzes the nature of each dataset beforehand, and decides which sub-algorithm is most likely to outperform**, based on standard and proprietary metrics I have developed.

The logic of the full algorithm:

1) ***Heavily skewed dataset*** ($S > 3$, a standard threshold value for skewness[8]): spotting outliers rightfully requires de-skewing into a *normal-like* distribution using Box-Cox transformation, then using the standard method of Z-score outlier removal - sub-algorithm 1.
2) **Non-skewed dataset**: no need to normalize the distribution, and the issue is whether to *remove* outliers or *cap them to a certain value*. I have defined **a metric called *"Normalized Variance" (NV)*** calculated by $NV \leftarrow \frac{var}{\mu^2}$ based on the variance and mean of the dataset (the default value for zero mean is 0).
   - **NV < 0.5**: *low variance* - extreme values are not frequent, don't carry important information and can be removed as outliers - sub-algorithm 2.
   - **NV > 0.5**: *high variance* - extreme values are quite frequent and cannot be removed; *capping* is more suitable - sub-algorithm 3.

3. Experimental Evaluation

After formalizing the algorithms, an experimental evaluation is performed to verify their validity and performance relative to the baseline, showing a considerable improvement. The following ***four*** datasets will be used:

1) ***House Prices dataset***: used throughout the course, having 1,460 records. Link to *Kaggle*.
2) ***The largest diamond dataset currently on Kaggle***: used for two previous course assignments, having 219,703 records. Link to *Kaggle*.
3) ***Insurance Prices***: 1,338 records. Link to *Kaggle*;
4) ***Car Prices***: having 8,128 records. Link to *Kaggle*.

The full algorithm (4) was applied on each dataset, while comparing its performance to six variations of each dataset:

- The original dataset (Original Pipeline).
- *Baseline* #1: Naive IQR (IQR outlier-removal).
- *Baseline* #2: Naive Z-score (Z-Score outlier-removal).
- Sub-algorithm 1 (*Box-Cox + Z-Score*).
- Sub-algorithm 2 (*Adaptive IQR Multiplier*).
- Sub-algorithm 3 (*Outlier Capping + Z-Score*).

In order to thoroughly examine the effectiveness of my proposed full algorithm, the results will be shown relative to the other 5 alternatives (including the original pipeline of no outlier handling), thereby demonstrating its superiority.

The two naive baselines which will be used in order to show the advantage of the sub-algorithms relative to the relevant baseline: the **naive Z-Score baseline** will be the reference if the full algorithm chooses sub-algorithm 1 (*Box-Cox* + *Z-Score*) or sub-algorithm 3 (*Outlier Capping* + *Z-Score*), while the **IQR baseline** will be the reference if the full algorithm chooses sub-algorithm 2 (*Adaptive IQR*).

The full algorithm will be evaluated by the following:

1) ***R^2* (Coefficient of Determination)**: evaluates how well the model explains the variance of the target variable - an R^2 score close to 1 shows good fit.
2) **MAE (Mean Absolute Error)**: measures the average absolute difference (predictions-actual values), expressed by the variable units (scale-dependent).
3) **MAPE (Mean Absolute Percentage Error)**: similar to MAE, but the average is over percentage differences (scale-independent).
4) **RMSE (Root Mean Squared Error)**: measures the average quadratic difference (predictions-actual values) - penalizing large errors and more sensitive to outliers. Comparison of RMSE to MAE before and after outlier handling shows if these outliers were significant or not (quadratic vs. linear relationship).
5) ***Percentage Improvement Metric***: a combined evaluation metric which I developed, explained later.

The experimental evaluation starts with the 1st dataset (***House Prices***) compared to the relevant baseline:

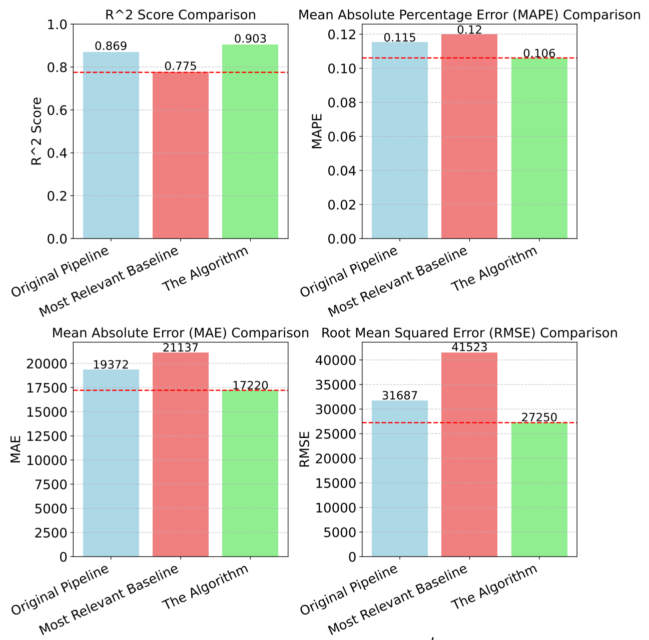

Fig. 2. Score comparison vs. baseline - 1st dataset (House Prices)

Since the calculated *skewness* of this dataset is 1.74 (< 3), and its *Normalized Variance* is 0.18 ($< \frac{1}{2}$), according to the algorithm (4), sub-algorithm 2 (***Adaptive IQR Multiplier***) is applied and therefore its relevant baseline is the *naive IQR* algorithm (generic 1.5 multiplier).

The proposed algorithm not only outperforms the original pipeline performance, but also significantly outperforms the baseline outlier-handling algorithm; In fact, the baseline (regular IQR) worsens the model's performance, by all four evaluation metrics! The suggested adaptive *tailor-made multiplier*, based on dataset characteristics, yields a performance improvement (contrary to the baseline IQR); the problem was not the use of IQR, but use of the generic 1.5 multiplier, while a sophisticatedly calculated multiplier has much better performance.

Further analysis of algorithm (4) and sub-algorithm 2 (*Adaptive IQR*) shows the four metrics for the five algorithms relative to the original pipeline in percentage terms:

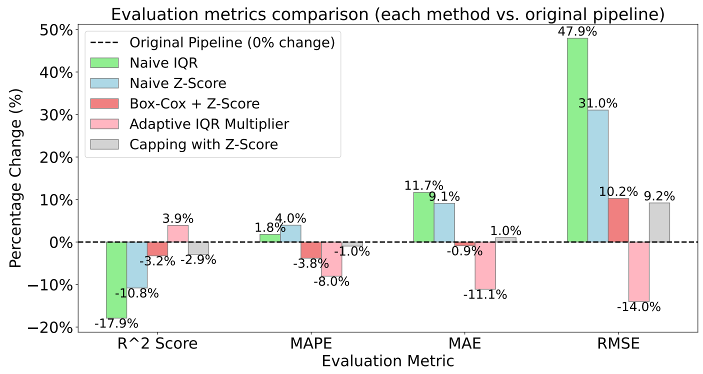

Fig. 3. Evaluation metrics improvement - 1st dataset (Houses Prices)

Indeed, as shown above, the chosen sub-algorithm 2 (Adaptive IQR) outperforms all other algorithms (which actually have a negative effect on model's performance).

Moving on to the 2nd dataset (***Diamond Prices***):

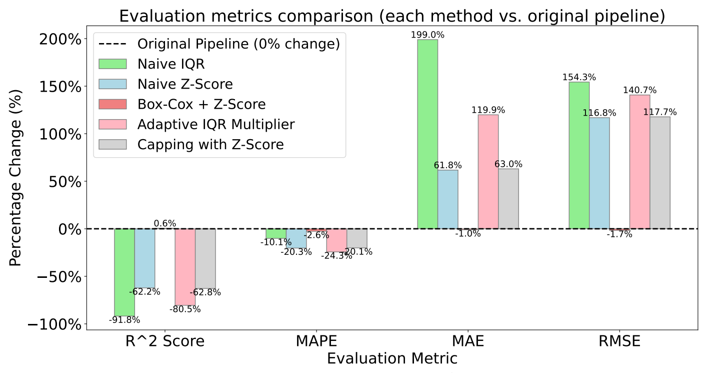

Fig. 4. Evaluation metrics improvement - 2nd dataset (Diamond Prices)

Since this dataset is highly skewed (*skewness* of $18 > 3$), the algorithm (4) chooses to perform sub-algorithm 1 (***Box-Cox + Z-Score***); similar to the previous case, **the proposed algorithm outperforms all algorithms**, while the relevant baseline algorithm (regular Z-Score outlier-removal) only worsens the model's performance, by almost all evaluation metrics. The baseline algorithm is so counterproductive that it lowers the R^2 by 62% and increases RMSE by 116%! The problem is not the Z-score method but using it with skewed non-normal data; however, performing Box-Cox transformation **as a preliminary de-skewing step** before applying the Z-score method handles outliers much better.

Moving on to the 3rd dataset **(Insurance Prices)**:

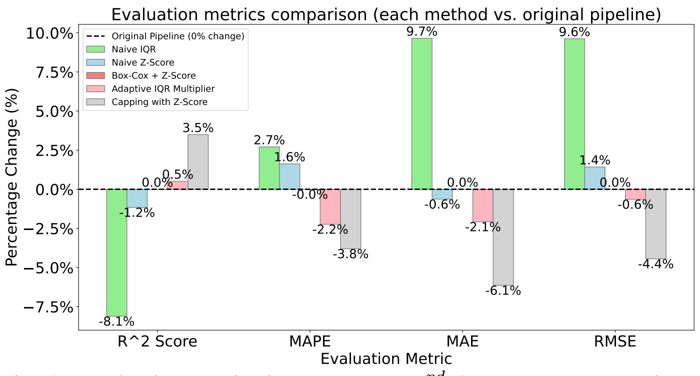

Fig. 5. Evaluation metrics improvement - 3rd dataset (Insurance Prices)

Since the *skewness* of this dataset is 1.51 (< 3) and its *Normalized Variance* is 0.81 ($> \frac{1}{2}$), according to the algorithm (4), sub-algorithm 3 (***Capping + Z-Score***) will be applied and its relevant baseline is the *naive Z-score*.

Similar to the previous case, **the proposed algorithm outperforms all algorithms**, while the most relevant baseline algorithm (regular Z-Score outlier-removal) either has lower positive effect compared to our algorithm (MAE score) or only worsens the model's performance (R^2, MAPE, RMSE scores); The problem is not using the Z-score method but how to use it - to remove outliers or to cap their value; in this case, extreme values are quite frequent (high NV) and it is indeed better to cap them.

It is interesting to notice the ***lack* of influence** of sub-algorithm #1 (*Box-Cox + Z-Score*) which may seem as an error. However, a more thorough analysis shows that the reason for this phenomenon stems from the distribution of the target variable in this dataset:

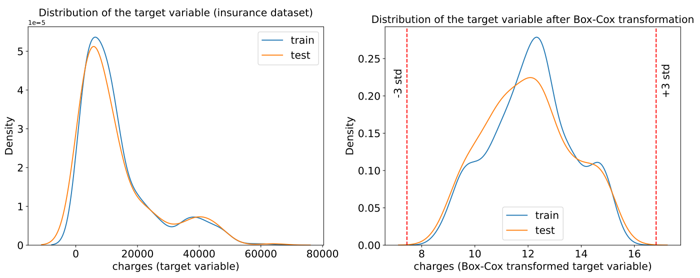

Fig. 6. Target variable distribution before and after Box-Cox Transformation - 3rd dataset (Insurance Prices)

The original distribution is shown on the **left plot**, and after Box-Cox (**right plot**) it becomes almost normal with no data points outside $\pm 3\sigma$! No outliers are found, aligning with 0% change relative to the original pipeline, supporting our algorithm decision <u>not</u> to choose this sub-algorithm.

Moving on to the 4th (***Car Prices***) dataset: since the skewness is 4.22 (> 3), sub-algorithm 1 (*Box-Cox+Z-Score*) is applied; and it indeed outperforms all other algorithms, including its most relevant baseline (*"naive Z-score"*, which performed poorly, leading to 50% higher MAE score, 71% higher RMSE score, and 16% lower R^2 score); <u>this means that *de-skewing* using *Box-Cox* transformation was indeed crucial in this case</u>.

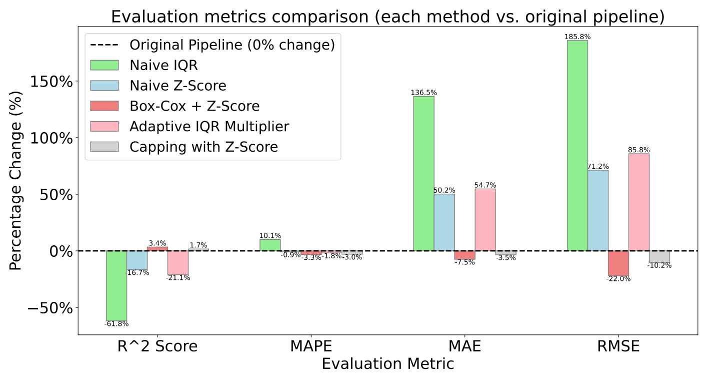

Fig. 7. Evaluation metrics improvement - 4th dataset (Car Prices)

The analysis so far has calculated four evaluation metrics (R^2, MAPE, MAE and RMSE). To simplify, **I suggest a <u>*combined* metric/score I developed</u>**, which averages the percentage improvement of these four parameters relative to the original baseline (or pipeline). Percentage improvement enables combining metrics of different value ranges/trends (R^2 has opposite trend than the other three). The suggested ***Percentage Improvement Metric*** **(PIM)**:

$$\textbf{PIM [\%]} \leftarrow \frac{100}{4} \times [(\frac{R^2_{alg}}{R^2_{orig}} - 1) + (1 - \frac{MAPE_{alg}}{MAPE_{orig}}) + (1 - \frac{MAE_{alg}}{MAE_{orig}}) + (1 - \frac{RMSE_{alg}}{RMSE_{orig}})]$$

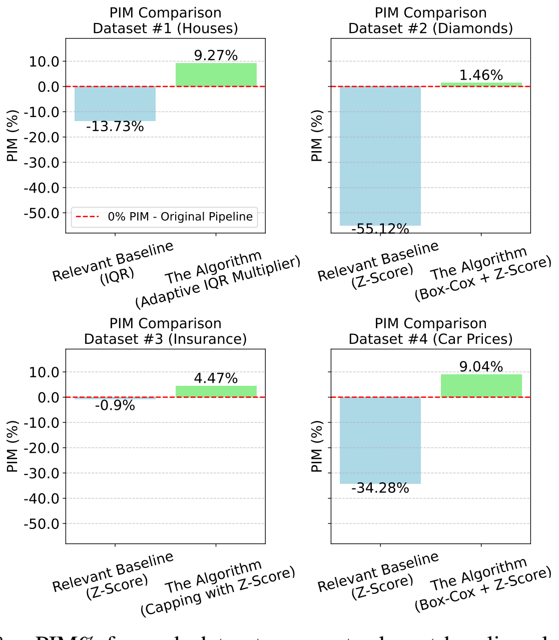

Fig. 8. PIM% for each dataset vs. most relevant baseline algorithm

The data-driven algorithm I have designed has an **improved (positive) PIM while the most relevant baseline algorithm always achieved a lower and negative PIM** (although some evaluation metrics had positive effect, the total PIM is negative - worse results than doing nothing). To summarize, the relative PIM improvement (algorithm vs. most relevant baseline) for all four datasets:

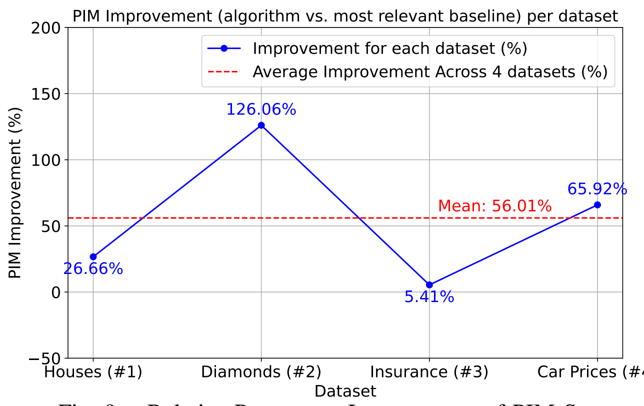

Fig. 9. Relative Percentage Improvement of PIM Score

The relative PIM improvement is consistently positive (always better than the baseline), ranging up to 126% (for the Diamond dataset) and **its *mean* improvement is 56%**. The conclusion is that according to each specific dataset, we can choose the right approach to deal with outliers; however, if we do not tune each approach and optimize it (e.g. using the generic and popular 1.5 IQR multiplier), we may end up with worse results than doing nothing. **This emphasizes the importance of making use of the developed techniques.**

4. Related Work

As already discussed, the building blocks I have used in this project are based on what we have learned in class throughout the semester (e.g. understanding that Z-Score assumes normal distribution and performing Box-Cox as means to normalize the distribution); without detracting from the aforementioned, I have reviewed in preparation for the study several related references and sources, including:

- One of the fundamental steps I performed in preparation for this study is recalling the methods we learned in class for outlier-handling; **one of the most common methods is using *IQR (The Interquartile Range)***. I tested this method on various datasets, but got mixed results for different datasets. Therefore, I went back to where this technique was introduced, in the article ***"Exploratory Data Analysis"*** written by *J.W. Tukey*[5] (inventor of the Box Plot). In this article, the concept of using a spread of 1.5x outside the box is first introduced, later on leading to the $1.5IQR$ method. **This made me wonder whether the *generic 1.5 multiplier* is indeed a good choice for every dataset**, or whether, by analyzing the nature of the dataset, it is possible to formulate a tailor-made multiplier for each specific dataset, considering its nature and characteristics; this is exactly the idea behind sub-algorithm 2, adaptive version of Tukey's original idea.
- In the GitHub repository **"A production ready approach for outlier detection and monitoring"** [9], the 1.5IQR method is further elaborated and implemented, alongside other outlier detection methods. This is an example of various sources I found that use the 1.5IQR method, with the generic 1.5 multiplier; the use of a fixed multiplier across various examples has led me to the idea behind sub-algorithm 2.
- In the article ***"Outlier Detection"*** written by *A. Sidihikha*[10], a thorough analysis of outlier related methods is conducted, including quartiles, percentile and Z-score methods. Reading this article, elaborating about Z-Score, made me recall the *prerequisite* for the Z-Score method - an approximately normal distribution.
- I used several additional sources to clarify the issue above; for example, in lesson #3 of the ***'STAT 500', Applied Statistics* course of *"PennState"* University (*The Pennsylvania State University*)**, the requirement for an approximately symmetric distribution is emphasized [11]. Therefore, the *theoretical gap* between the prerequisites for using this method against the popularity of using it (without checking the distribution beforehand) **made me wonder whether it is possible to still use Z-Score method for outlier-detection, with an adaptation for *skewed* (and even *highly skewed*) data**; this is exactly the idea behind sub-algorithm 1, which still uses the concept of Z-Score for outlier-detection, but applies Box-Cox transformation as a preliminary step, to normalize the distribution.
- In the article ***"Handling Outliers"*** [6], several techniques for outlier handling are shown, including capping that is shown with two examples - either capping values according to *IQR-calculated thresholds* or capping values according to *percentiles* (99% and 1%). This flexibility in capping values led me to the approach of **using two *types* of values for capping** - the first in order to *detect* outliers, and the second as the *capped value*. This flexibility allows more optimization compared to the regular approaches.

5. Conclusion

This project's purpose was to offer an automated adaptive outlier handling method, in order to assist performing an easier data science process, and achieving better model performance. Throughout this paper, **I have presented the full *adaptive* and *data-driven* algorithm using three *sub-algorithms***; The idea for each of the sub-algorithms presented in this work stemmed from **variation and combination of various methods we have learned in class**: **sub-algorithm 1** consists of a combination of *Box-Cox transformation* for making the data closer to a normal distribution, with *Z-Score method* for outlier detection; **sub-algorithm 2** is a data-driven variation of the *IQR method* for outlier detection (a *tailor-made IQR multiplier*, based on analyzing the nature of the dataset, instead of the generic 1.5 multiplier suggested by *Tukey* [5]); and **sub-algorithm 3** consists of outlier-capping, using *percentile-based capping* (also called *Winsorization*) and *Z-Score method* for outlier detection. The full data-driven algorithm analyzes the nature of each dataset *beforehand*, and decides which sub-algorithm is most appropriate to handle outliers present in it; by computing the ***skewness*** of data, as well as the ***Normalized Variance (NV)***, which is a new metric I have developed as part of this project to characterize *the magnitude of high-valued data-points normalized by the mean value*, **the algorithm chooses the appropriate sub-algorithm to apply beforehand, instead of the *'brute-force', computationally-intense alternative***, of just running various naive outlier-handling methods.

The main themes of the project: **(I) developing new, better methods for outlier handling** - this is addressed by presenting three sub-algorithms I have developed; **(II) offering an automated adaptive outlier handling tool** - this is addressed by the full algorithm (4), which chooses the appropriate sub-algorithm to apply for each dataset.

The concepts behind the algorithms presented in this work **highlight the importance of gaining deep knowledge of the *mathematical and statistical concepts* behind the popular methods used as part of the data-science pipeline**; in other words, although it is possible to just use the most common methods for outlier-handling *'as-is'*, mostly as some of them are even *pre-implemented* as part of certain libraries (e.g. basic *winsorization* method offered in the *'SciPy'* library), there are significant benefits for studying the basic concepts and methods *'from the ground up'*, as we learned in class throughout the semester. For example, **without deep understanding of the statistical foundations behind the Z-Score method (which relies on having approximately normal distributed data), it would not be possible to come up with the idea behind sub-algorithm 1**, which performs Box-Cox transformation as a preliminary step, to overcome this weakness in the basic Z-Score method.

It should also be noted that expanding the experimental evaluation to analyzing a wider range of datasets (beyond the four analyzed as part of this work), or alternatively focusing on a narrower specific family of datasets (sharing similarities) may yield specific *further optimized, tuned algorithm parameters* (either parameters of the main algorithm - thresholds for *skewness* or *normalized variance*, or parameters of some of the sub-algorithms - such as *IQR multiplier* parameters or *Z-score thresholds*). This optimization has already been performed as part of this research for the *four* datasets introduced, and was explained throughout the article (such as optimization of the multiplier of the IQR method in sub-algorithm 2), but when considering more datasets (either generic or family specific), some further optimization may be performed. **Therefore, future work could explore further optimizing the parameters by analyzing additional datasets**. This approach could lead to achieving even further dataset-tuned model performance.

References

[1] F. E. Grubbs. Procedures for Detecting Outlying Observations in Samples. *Technometrics*, 11(1):1–21, 1969.

[2] H. Wainer. Robust Statistics: A Survey and Some Prescriptions. *Journal of Educational Statistics*, 1(4):285–312, 1976.

[3] S. Baccari, M. Hadded, H. Ghazzai, H. Touati, and M. Elhadef. Anomaly Detection in Connected and Autonomous Vehicles: A Survey, Analysis, and Research Challenges. *IEEE Access*, 12:19250–19276, 2024.

[4] Azzedine Boukerche, Lining Zheng, and Omar Alfandi. Outlier Detection: Methods, Models, and Classification. *ACM Computing Surveys*, 53(3):Article 55, 37 pages, June 2020.

[5] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley Publishing Company, 1977. Reading, Mass. — Menlo Park, Cal., London, Amsterdam, Don Mills, Ontario, Sydney.

[6] Biswas Anushree. Handling Outliers, 2023. Medium, Jun 17, 2023.

[7] W. J. Dixon. Simplified Estimation from Censored Normal Samples. *Ann. Math. Statist.*, 31(4):385–391, 1960.

[8] Shinde Yashowardhan. What is skewness in data? How to fix skewed data in python?, 2021. Posted on Medium, Aug 27, 2021.

[9] Vincent Belz. A production ready approach for outlier detection and monitoring. , 2020. Published December 2020.

[10] Ayesha Sidhikha. Outliers Detection, 2024. Medium, Jan 13, 2024.

[11] PennState University. Lesson #3 of the STAT 500 Applied Statistics Course: Subsection 3.3.3 - Probabilities for Normal Random Variables (Z-scores), 2025. Accessed: 2025-02-23.