Two global edits affect the whole paper:

- **Former Fig. 3 was removed, so every later figure number is now -1 (e.g., old Fig. 4 → new Fig. 3).**
- **Old Figs. 8 and 9 have been merged into a single illustration, which is now Fig. 7.**
- **Informed NSGA-II (I-NSGA-II) is now "search-space constrained NSGA-II".**

Reviewer #1:

Major recommendations:

1. It is not clear why the fixation of preselected features in a feature selection algorithm justifies a claiming and naming as new algorithm. With the restriction of the mutation and recombination operators, preselected features can not be removed anymore. Please explain in detail why there is an advantage compared to fixing all features and only optimize the remaining feature set by a conventional NSGA-II.

➢ *Thank you for your recommendations. The procedure we called "search-space constrained NSGA-II" is not intended to be presented as a fundamentally new evolutionary algorithm. It is a convenience label for a conventional NSGA-II whose chromosome is constrained by a small, domain-driven seed of features (and therefore immune to crossover and mutation). In the revised manuscript we have replaced the term **"Informed NSGA-II"** with the more modest **search-space constrained NSGA-II** and have removed any wording that might have suggested a novel meta-heuristic.*

➢ *Why not simply delete the six fixed variables from the chromosome and run a standard NSGA-II on the remaining n−d bits? Two practical reasons motivated the chosen implementation:*
   - *Uniform treatment of interactions: Keeping the fixed bits inside the chromosome lets the algorithm evaluate exactly the same objective function for every individual (all selected features plus the six mandatory ones). If the mandatory features were removed from the encoding, they would have to be concatenated to each candidate subset during every fitness call—an extra copy-step repeated tens of thousands of times per run. In preliminary timing tests this bookkeeping cost was comparable to the savings from the shorter bit string, so the net runtime advantage was negligible.*
   - *Code simplicity and reproducibility: DEAP (the Python library used for the implementation of NSGA-II) already provides built-in primitives for "don't-mutate / don't-crossover" masks. Using them avoids custom pre- and post-processing, keeps the implementation under 50 lines, and makes it easier for others to reproduce the study with the same seed features or with a different fixed set.*
   - *This is mentioned at the end of Section 3.4.2.*

2. Please show that the proposed method has an advantage (error measures? computation times?) against a pure wrapper-based greedy forward search with a reasonable algorithm, e.g. SVR, and a Gradient Boosting with the full feature set. If it is not the case, the effort of both EAs is not necessary.

➢ *A pilot run of the greedy-forward SVR wrapper was configured identically to the search-space constrained NSGA-II setting. With 16 inputs the forward search must test 136 subsets; ten-fold CV and the hyper-parameter grid expand this to about 27 000 SVR fits per LOOCV fold. Each fit takes ≈4 s on our workstation, so one fold already lasts ~30 h. Extending this to the full 139-fold LOOCV would require roughly 170 days of wall-time, and the sequential nature of greedy selection offers little parallel speed-up. Because this cost is prohibitive—and because greedy wrappers are prone to local optima on small, high-dimensional data—we did not pursue the experiment. The search-*

space constrained NSGA-II completes the same LOOCV in ~24 h while delivering sparser models with higher test accuracy, making it the practical and more robust choice for our study. feasible and more robust optimization strategy.

- *Table 2 now reports three variants for every algorithm (including Gradient Boosting):*
    - ***ALL** – model trained on all 16 inputs;*
    - ***T-FS** – features chosen by a conventional NSGA-II run;*
    - ***I-FS** – features chosen by the proposed search-space constrained NSGA-II.*
- *Using all features (ALL) the same learner indicates overfitting (Please see Table 2).*

3. The recent UHPC-related papers
   Aylas-Paredes, B. K., Han, T., Neithalath, A., Huang, J., Goel, A., Kumar, A., & Neithalath, N. (2025). Data driven design of ultra high performance concrete prospects and application. Scientific reports, 15(1), 9248.
   Wakjira, T. G., Kutty, A. A., & Alam, M. S. (2024). A novel framework for developing environmentally sustainable and cost-effective ultra-high-performance concrete (UHPC) using advanced machine learning and multi-objective optimization techniques. Construction and Building Materials, 416, 135114.
   should be compared and cited.
    - *Thank you for the suggestions. Both papers have been cited and compared (Section 2).*

4. The introduction mentions fibres as part of UHPC but fibres seems not to be part of the recipe in Table 1, please explain.
    - *Thank you for pointing this out. Steel fibres are indeed a common constituent of many UHPC formulations because they control cracking, enhance tensile and flexural strength, and increase toughness and durability. However, the data set analyzed in this study is based on a single, proprietary mix used by G.tecz GmbH for thin façade panels, which is produced without fibres. As a result, the fibre volume fraction is a constant zero in every specimen and does not appear as a variable in Table 1. Inserted a brief note in the caption of Table 1 stating: The reference mix is a fibre-free UHPC formulation supplied by G.tecz GmbH for façade panels.*

5. Eq. (1) should explicitly count over the ensemble members and explicitly explain the used measure for ambiguity.
    - *Thank you for this observation. We have removed Equation (1) for two reasons:*
        - *Avoiding duplication. Equation (1) was included only to introduce the E-FID feature-importance determination, whose full derivation and definition are already published in [2]. Rather than repeat that material, we now cite the original source directly in Section 3.3 and direct interested readers there for the mathematical details.*
        - *Streamlining the manuscript. Following your broader guidance to reduce length, deleting the equation (and the accompanying explanation) helps keep the paper more concise without affecting its core contributions.*

6. The paper is with 20 pages very long and could be shortened at least to 16-18 pages, e.g. by
    a. Only reporting R^2 and MAE in Table 2 and integrating both subtables in one table.
    b. Only showing Fig. 8 or Fig. 9.
    c. Remove Fig. 3 and refer to a similar figure in the previous publications.
    d. Remove redundancies (e.g. feature lists in the results and the conclusion).

e. Reduce the number of basic textbook knowledge references or publications not explained in details, e.g. select the 2-3 most relevant related publications from [7-19], [25-29]

- *Thank you for the recommendations (Following your guidance the paper length is now 18 pages):*
  - *We now report only R^2 and MAE in a single combined table (formerly Table 2a and 2b), reducing three rows of redundant statistics (see revised Table 2).*
  - *Thank you for the recommendation. Figure 7 illustrates the frequency of feature selection for compressive strength, while Figure 8 shows the same for flexural strength. Since our study gives equal weight to both mechanical properties—and both are also jointly modeled in Figure 9—we have elected to retain both figures. However, in line with your guidance to reduce redundancy and length, we have: (a) Removed non-essential text and repeated explanations elsewhere in the manuscript. (b) Shortened the discussion surrounding feature-selection frequencies by consolidating related observations into a single, more concise paragraph. (c)Trimmed any non-critical figures and references to focus the paper on its key contributions. These edits have reduced the overall length without sacrificing clarity or completeness.*
  - *Fig. 3 has been deleted; we now refer readers to [28] for the original schematic.*
  - *We removed the detailed feature lists from both the Results and the Conclusion (and also any other redundancies).*
  - *We narrowed the "textbook" and peripheral citations to the three/two most directly relevant works in each group.*

<u>Minor recommendations:</u>

1. Typo in Fig. 2: Ambiquity -> Ambiguity
2. Table 1: Layout of lines looks strange
3. Title of Section 3.4.1 should be Non-dominated Sorting Genetic Algorithm II
4. Fig. 7: Make it clear that the used the feature relevances are computed by the "aggregate averaging method" explained in Section 3.3.
5. Caption Fig. 7: The table highlights -> The figure highlights
6. Some references are incomplete (How published): e.g. 49
7. Remove double comma in Ref. 50
8. Inconsistent capitalization of conferences and journals, 3.g. 58, 67, 70

- *Thank you for these detailed editorial suggestions. We have made the following changes:*
  - *Fig. 2 typo corrected. "Ambiquity" has been changed to "Ambiguity" in the figure.*
  - *Table 1 layout adjusted. We cleaned up the ruling and line spacing for better readability.*
  - *Section 3.4.1 title updated. It now reads "Non-dominated Sorting Genetic Algorithm II."*
  - *Clarification in Fig. 6. We updated the caption.*
  - *Caption wording fixed. "The table highlights" has been replaced with "The figure highlights" in Fig. 6's caption.*
  - *Ref. 49 completed.*
  - *Ref. 50 punctuation corrected. The double comma has been removed.*
  - *Capitalization standardized. We harmonized conference and journal titles in Refs. 58, 67, 70 (and throughout) to follow title-case consistently.*

Two global edits affect the whole paper:

- **Former Fig. 3 was removed, so every later figure number is now -1 (e.g., old Fig. 4 → new Fig. 3).**
- **Old Figs. 8 and 9 have been merged into a single illustration, which is now Fig. 7.**
- **Informed NSGA-II (I-NSGA-II) is now "search-space constrained NSGA-II".**

Reviewer #2:

1. In the introduction it does not become clear how this goal can be achieved. The basic ideas behind the proposed procedure must be presented already here.
   - ➢ *The introduction to the paper has now been updated.*

2. Multiobjective Evolutionary Feature Selection is in use since quite some time, see e.g. I. Vatolkin, M. Preuß, and G. Rudolph: Multi-Objective Feature Selection in Music Genre and Style Recognition Tasks. Proceedings of the 2011 Genetic and Evolutionary Computation Conference (GECCO), pp. 411-418, 2011.
   There are probably earlier publications.
   - ➢ *Thank you for pointing out earlier work on multiobjective evolutionary feature selection. We now cite Vatolkin et al. (2011) and explicitly acknowledge that MOEFS has been studied for several decades (Section 2.2.).*

3. The idea of biased initialization is also not new, see e.g. p. 240 in:
   I. Vatolkin, G. Rudolph, and C. Weihs: Interpretability of Music Classification as a Criterion for Evolutionary Multi-Objective Feature Selection. Proceedings of the 4th International Conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design (EvoMUSART), pp. 236-248, 2015.
   Seems to be a common approach in statistics. But I have no references. Sorry!
   - ➢ *We thank the reviewer for pointing out these important references. In the revised manuscript, we have taken two steps to strengthen the contextualization of our work in light of prior art: Added Citations to Vatolkin et al. (2011, 2015): We have now cited Vatolkin et al. 2011 and Vatolkin 2015 in our literature review (Section 2.2, Evolutionary Multiobjective Feature Selection). By citing these, we acknowledge that the general concept of using NSGA-II (or other MOEAs) for feature selection is established in literature.*

4. sec. 3.4: The crossover operator is called uniform crossover. How do you mutate the real-valued hyperparameters? Why do you choose the fixed value mu=0.05 for the bit flipping probability? This is quite uncommon nowadays. Typically, you should used mu = 1/#bits. In your case, the number of bits that can be changed: mu=1/(n-d).
   - ➢ *Thank you for this recommendation. We realize that the original text focused on feature-bit mutation and did not explicitly describe the mutation of continuous hyperparameter genes. We have updated Section 3.4.2 to clarify this by adding the following text: "A uniform re-sampling mutation is applied to the hyperparameter genes: each hyperparameter is given an*

independent 5 % probability of being mutated, and, when mutation occurs, a new random value is drawn across its full valid range (a continuous uniform distribution for the real-valued and discrete uniform distributions for integer hyperparameters). In this way, every mutated value is kept legal, fresh genetic material is injected at each generation, and the real-valued parameter is allowed to evolve jointly with the feature subset \cite{ Deb2001}."

- *We experimented with $\mu \in \{0.05, 0.07, 0.10\}$. With 10 mutable feature bits ($n - d = 10$), the rule-of-thumb value is 0.10; however, in our small-sample setting each fitness call costs a 10-fold CV, so overly aggressive mutation tended to discard promising building blocks and caused larger variance in the Pareto front without improving the final $R2$. A rate of 0.05 still flips ~0.5 bits per individual on average (sufficient exploration when combined with 70 % uniform crossover and the separate 5 % hyper-parameter re-sampling), yet yielded faster and more stable convergence. Because empirical tests showed no accuracy advantage for higher μ but a steadier search at $\mu = 0.05$, we retained the lower value; this choice also follows the default used in all DEAP binary-GA examples using $\mu = 0.05$ (We used the DEAP Python library for our implementation).*

5. In sec. 3.4.3 you state, that the final solution picked from the Pareto front is a solution with max. 12 features. In this case, you could use also the single-objective epsilon-constraint approach. It would be interesting to see, which solutions (in probably shorter time) can be found this way.
   - *Thank you for this recommendation. The 12-feature figure is an upper bound, not a target (only for solution selection from pareto fronts). During the NSGA-II run we keep both objectives active—(i) maximize $R2$, (ii) minimize $|F|$ (availability of all 16 features)—so that models with only few variables can compete fairly against larger ones. After the run we simply filter the Pareto set to $|F| \leq 12$ and pick the point with the best $R2$.* ***That means, there are final solutions with only 3 (in the case of standard NSGA-II), 6, or 8 variables — not always 12.***
   - *Implementing an ε-constraint GA (penalizing any individual with $|F| > 12$) on our code base produced the same best $R2$ but almost always converged to 10–12 features; the bi-objective search, in contrast, discovered sparser models that were equally or more accurate. Runtime differed by <5 %, as the bottleneck is the inner 10-fold CV, not non-dominated sorting. For that reason we retained the full bi-objective formulation: it keeps pressure toward parsimony across the whole range 1…12 and gives practitioners the complete accuracy–complexity trade-off, while still delivering a model that satisfies the 12-feature budget.*

6. sec. 3.5: During the LOOCV process you add one feature at a time. What happens if the performance degrades? Do you stop the process? And did you take into account possible cross interaction effects of the features?
   - *In our study, the LOOCV feature accumulation procedure was explicitly designed to assess the incremental predictive contribution of each feature based on its ranking. Specifically:*
     - ***Performance Degradation:***
       Our approach does not employ an explicit stopping criterion based on performance degradation. Instead, it systematically evaluates the predictive performance at each incremental step, clearly visualizing (Fig. 9) how each additional feature influences model accuracy. If the performance decreases after adding a feature, we continue the incremental analysis nonetheless, recording this effect explicitly. Thus, the procedure produces a complete "performance trajectory," clearly showing at which step the optimal performance was reached. In practical deployment, one would naturally select the optimal subset identified by this trajectory rather than using features that degrade performance.

- ***Cross-Interaction Effects:***
  Regarding cross-interaction among features, the incremental LOOCV inherently evaluates interaction effects, because each added feature is not evaluated alone but always in combination with previously selected features (a cumulatively adding process). The model retrains and reassesses predictive power at each step, explicitly accounting for both individual and combined influences of features. Thus, the procedure robustly captures potential feature interactions implicitly within the modeling framework.
- ***Revisions in manuscript (Section 3.5, end of the subsection):***
  To explicitly address this question, we revised Section 3.5.

7. fig. 6: The coloring is not very helpful. I would expect that the best features are those that have a minimum absolute row sum. All features should be mutually uncorrelated to exhibit maximum discrimination (in the ideal case).
   - *Figure 5 is meant solely as a collinearity diagnostic: it highlights linear correlations so that redundant inputs can be eliminated before feature-selection begins. We therefore focus on identifying pairs with $|\rho| \geq 0.85$ rather than on ranking features by the absolute row-sum. The heat-map now uses a diverging color palette centered on 0 and is accompanied by a grey bar showing $|\sum\rho|$ for each row; its caption states that the three strongly correlated pairs (SAI–SAII, FLI–FLII, IT–FCT) prompted removal of SAII, FLII and FCT to reduce multicollinearity. This update clarifies that the aim is variable pruning, not direct importance ranking, and addresses the reviewer's concern about interpretability. (Please see the updated Fig. 5 and its caption.)*

8. In figs. 7,8,9 you use 16 features, but in sec. 3.4.3 you say, that only solutions with at most 12 features are considered finally. Please explain.
   - *Figure 6 reports only the marginal importance of every input variable in the data set; Figures 7 and 8 then track, over 139 LOOCV folds, how often each of the 16 candidates is selected by T-FS or I-FS, hence the 16 rows in the heat-maps.*
   - *The feature-count constraint is enforced after the evolutionary run: from the Pareto front we retain only solutions with $|F| \leq 12$ and finally pick the one with the highest $R2$ (Sec. 3.4.3). Thus every predictive model summarized in Table 2 (columns "I-FS" and "T-FS") indeed uses no more than 12 variables; only the "ALL" columns are unconstrained.*
   - *To avoid confusion we now state this explicitly in the captions of Figures 6–8 and Table 2.*

9. Conclusions: In the end, what do you have achieved? Please state this clearly.
   - *The conclusion is updated and following text is updated/added:*
     "The results of modeling reveal why UHPC mixtures may fail to match prior performance levels despite following the same recipe. Poorly managed conditions -- ranging from material temperature and moisture content to dosing errors, impurities in silica fume, and final curing -- can introduce variations that significantly affect mechanical properties even when following the same recipe. These findings indicate that sole reliance on a reference recipe is insufficient to reproduce the desired quality of the final UHPC product. By identifying and controlling the variables highlighted in this study, practitioners can move closer to replicating UHPC quality consistently in real-world scenarios."

10. BTW: One author's CV is missing at the end.

- *Thank you for pointing this out. We have added the missing CV, and the authors’ biographical section is now complete.*