A PAIRWISE LEARNING-TO-RANK MODEL FOR FEATURE ENGINEERING, LOGISTIC OBJECTIVES, AND CALIBRATION

IZ TECÚM

Abstract. We develop an interpretable learning-to-rank pipeline for admissions screening in a small academic society setting (Pi Mu Epsilon), trained from a limited number of human pairwise comparisons. Each applicant a is mapped deterministically to a feature vector $x(a) \in \mathbb{R}^d$ derived from structured fields (GPA, calculus completion, advanced coursework), extracted resume text, and short-answer signals. A linear score $s_w(a) = \langle w, x(a)\rangle$ is learned by minimizing a pairwise logistic loss (Bradley–Terry / RankNet style) with ℓ_2 regularization:

$$L_\lambda(w) = \frac{1}{|\mathcal{D}|} \sum_{(i,j,y)\in\mathcal{D}} \log\Big(1+\exp(-z_{ij}\langle w, x(a_i)-x(a_j)\rangle)\Big) + \frac{\lambda}{2}\|w\|_2^2, \qquad z_{ij} = 2y-1 \in \{-1,+1\}.$$

Ultimately, we describe the calibration of raw scores to a 0–10 percentile scale and the precise data artifacts that connect deployed scoring to local training (`applicants.jsonl`, `pairs.csv`, and `rank_model.json`).

Contents

1. Introduction

The operational goal of first-round admissions screening is not to make a final decision, but to produce a consistent, auditable ordering that helps reviewers allocate scarce attention. In our

Date: January 18, 2026.

setting, the data regime is intrinsically small: we may have only $n \approx 10$–50 applicants per cycle, and we prefer not to encode "ground-truth" scores that could reflect arbitrary scaling differences across reviewers. This motivates a *pairwise* labeling protocol: reviewers answer questions of the form "between applicant i and applicant j, who should rank higher?"

Mathematically, pairwise labels constrain only relative ordering. Let $\mathcal{A}$ denote the space of applications and let $x : \mathcal{A} \to \mathbb{R}^d$ be a deterministic feature map. We seek a score $s : \mathcal{A} \to \mathbb{R}$ such that $s(a_i) > s(a_j)$ whenever the reviewer preference is $a_i \succ a_j$. We restrict to linear scores

$$s_w(a) = \langle w, x(a) \rangle, \qquad w \in \mathbb{R}^d,$$

to preserve interpretability: each coordinate of $x(a)$ corresponds to a named signal, and each weight w_k can be inspected to understand how that signal influences rank.

The training data are a multiset of labeled comparisons

$$\mathcal{D} = \{(i, j, y_{ij})\}, \qquad y_{ij} \in \{0, 1\},$$

where $y_{ij} = 1$ indicates $a_i \succ a_j$. A canonical probabilistic model (Bradley–Terry) assumes

$$\mathbb{P}(a_i \succ a_j \mid w) = \sigma(\Delta_{ij}(w)), \qquad \Delta_{ij}(w) = s_w(a_i) - s_w(a_j), \qquad \sigma(t) = \frac{1}{1 + e^{-t}}.$$

Maximizing likelihood is equivalent to minimizing a sum of logistic losses over score differences, optionally with ℓ_2 regularization to stabilize fitting under limited pairs. This setup is widely used in learning-to-rank because it directly matches the type of supervision one can reliably collect (comparisons rather than absolute ratings).

A second design goal is *decomposability*. In particular, we treat resume keyword detectors as separate coordinates (research, teaching, leadership, awards) rather than compressing them into a single opaque number.

2. Preliminaries

Let $\mathcal{A}$ denote the space of applications (structured form fields, course selections, resume PDF text, and short-answer text). Our goal is to output a scalar *ranking score* which is used in an initial screen, while deferring final decisions to human review. Pairwise logistic ranking models are classical in psychometrics and ranking (Bradley–Terry; Plackett–Luce) [1, 3, 2] and are widely used in information retrieval / learning-to-rank [5, 4].

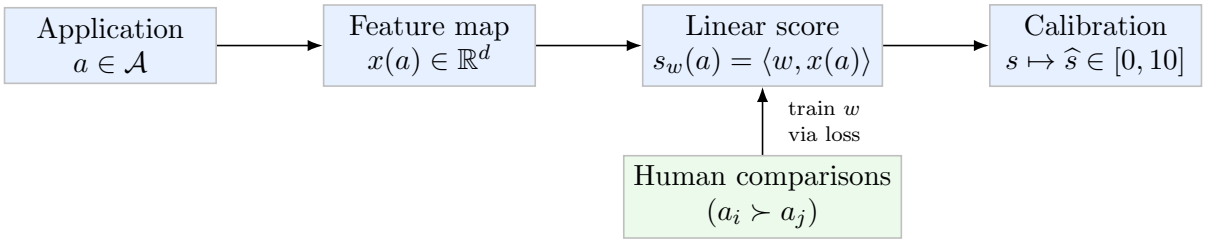

Figure 1. Pipeline: applications are mapped to features, scored linearly, calibrated to a 0–10 scale, and trained using pairwise labels.

3. From applications to vectors

We fix a space of applications $\mathcal{A}$ and define a *deterministic* feature map

$$x : \mathcal{A} \to \mathbb{R}^d, \qquad a \longmapsto x(a) = (x_1(a), \ldots, x_d(a)),$$

together with a schema identifier $\mathrm{fv} \in \mathbb{Z}_{\geq 1}$ ("feature_version"). The determinism requirement is crucial: for any fixed application a, repeated evaluation yields the same $x(a)$; all randomness in

the system enters only through the subsequent learning stage (pairwise training) and not through extraction.

The coordinates are designed to be *interpretable*: each $x_k(a)$ is intended to represent one semantically coherent signal (GPA strength, calculus completion, keyword evidence of research, etc.). In the current implementation we take $d = 21$ and write the feature vector in blocks,

$$x(a) = \big(x_{\text{struct}}(a),\, x_{\text{resume}}(a),\, x_{\text{kw}}(a),\, x_{\text{essay}}(a),\, x_{\text{pen}}(a)\big),$$

where each block is a small collection of coordinates described below.

3.1. **Structured signals from typed fields.** Let $\text{GPA}(a) \in [0, 4.33]$ be the reported cumulative GPA (we allow A^+ inflation up to 4.33), let $\text{Calc}(a) \in \{\text{yes}, \text{no}\}$ denote completion of Calculus I&II (or equivalent), and let $C(a) \subset \mathcal{C}$ be the set of marked "advanced/recent" courses (with $|C(a)| \le 6$), validated against a whitelist $\mathcal{W} \subset \mathcal{C}$ with an explicit exception for graduate courses MATH GR5xxx/6xxx.

We convert these fields into bounded numeric coordinates by explicit monotone maps. First, the GPA coordinate is

$$x_{\text{gpa}}(a) \;:=\; 10 \cdot \phi\!\left(\frac{\text{GPA}(a)}{4.33}\right) \in [0, 10],$$

where $\phi : [0, 1] \to [0, 1]$ is nondecreasing and chosen to be mildly concave near the top to avoid over-separating near-perfect GPAs. (For instance one may take $\phi(u) = u^\alpha$ with $\alpha \in (0, 1]$, or a piecewise-linear map with a soft plateau; the only properties used later are boundedness and monotonicity.)

Second, calculus completion is encoded as a binary gate

$$x_{\text{calc}}(a) \;:=\; 10 \cdot \mathbf{1}\{\text{Calc}(a) = \text{yes}\} \in \{0, 10\}.$$

Third, we record a coursework signal that depends only on $C(a)$

$$x_{\text{upper}}(a) \;:=\; 10 \cdot \psi\!\left(\frac{|C(a)|}{6}\right) \in [0, 10],$$

where $\psi : [0, 1] \to [0, 1]$ is nondecreasing and may be chosen concave to model diminishing returns from each additional marked course. The validation layer is logically separate: the set $C(a)$ is constructed by the rule

$$C(a) \;=\; \{c \in \mathcal{C} :\; c \text{ entered by user and } (c \in \mathcal{W} \text{ or } c \text{ is MATH GR5/6xxx})\},$$

so that the score is computed only from accepted courses and is therefore well-defined independent of user formatting noise.

3.2. **Resume extraction and length signal.** Let the applicant upload a PDF resume, from which we extract text in-browser via PDF.js. Denote by $T(a) \in \Sigma^*$ the extracted plaintext string, and let $\ell(a) := |T(a)|$ be its character length (or a token length; any fixed notion works provided it is deterministic). We record a length-based completeness proxy

$$x_{\text{rlen}}(a) \;:=\; 10 \cdot s\!\left(\frac{\ell(a)}{L_0}\right) \in [0, 10],$$

where $L_0 > 0$ is a normalization constant and $s : [0, \infty) \to [0, 1]$ is a saturating function such as $s(u) = \min\{1, u\}$ or $s(u) = 1 - e^{-u}$. This coordinate is explicitly *not* meant to reward excess word usage; its intended role is to prevent pathological extraction failures (e.g. an empty $T(a)$ caused by a corrupted upload) from silently propagating through downstream keyword features.

3.3. **Keyword detectors as separate coordinates.** A central design decision is to keep resume keyword evidence in *separate coordinates* rather than collapsing it to one opaque number. Fix four categories

$$\mathcal{K} = \{\text{research, teach, leader, awards}\},$$

and for each $c \in \mathcal{K}$ fix a finite dictionary $D_c \subset \Sigma^*$ of tokens/phrases (or regex families) associated with that category. Define the indicator-count statistic

$$N_c(a) \;:=\; \sum_{w \in D_c} \mathbf{1}\{w \text{ appears in } T(a)\},$$

and map it to a bounded score by a saturating transform with threshold $\tau_c \geq 1$,

$$x_c(a) \;:=\; 10 \cdot \min\Big\{1, \frac{N_c(a)}{\tau_c}\Big\} \in [0, 10].$$

Equivalently one may use a smooth saturation such as $x_c(a) = 10\big(1 - e^{-N_c(a)/\tau_c}\big)$; since x is deterministic, smoothness is not required for computation, but it can be convenient when reasoning about sensitivity to small textual edits.

This yields four explicit coordinates

$$x_{\text{research}}(a), \qquad x_{\text{teach}}(a), \qquad x_{\text{leader}}(a), \qquad x_{\text{awards}}(a),$$

which the ranker may weight independently. The separability also lets us report a transparent keyword-only summary

$$\mathrm{KW}(a) \;:=\; \frac{1}{4} \sum_{c \in \mathcal{K}} x_c(a) \in [0, 10],$$

as an auxiliary diagnostic *without changing* the learned score $s_w(a)$. In particular, a reviewer can inspect $\mathrm{KW}(a)$ as a coarse "resume-evidence" meter even if the learned model downweights some keyword categories.

Example 3.1 (Keyword saturation)**.** Suppose D_{research} contains 15 phrases and we set $\tau_{\text{research}} = 3$. If an extracted resume contains {"*REU*", "*preprint*", "*poster*"} among those phrases, then $N_{\text{research}}(a) = 3$ and hence $x_{\text{research}}(a) = 10$. If it contains only {"*REU*"}, then $N_{\text{research}}(a) = 1$ and $x_{\text{research}}(a) = 10/3 \approx 3.33$. Thus the coordinate saturates quickly once there is credible evidence, and additional occurrences do not inflate the score beyond the cap.

3.4. **Short-answer signals and decomposed essay coordinates.** Let $E_A(a), E_B(a) \in \Sigma^*$ denote the two short answers (Short Answer A and Short Answer B). In line with modern ranking practice, we do not treat the essays as a single scalar "quality" label; instead we define several coordinates that measure distinct aspects of writing, mathematical content, and specificity. Formally, we construct a vector

$$x_{\text{essay}}(a) = \Big(x_{\text{band},A}(a),\, x_{\text{band},B}(a),\, x_{\text{arc}}(a),\, x_{\text{plan}}(a),\, x_{\text{math},A}(a),\, x_{\text{math},B}(a)\Big) \in [0, 10]^6,$$

where each coordinate is a bounded deterministic functional of $(E_A(a), E_B(a))$.

To make this explicit, let $\mathrm{wc}(E)$ be a word-count functional and $\mathrm{sent}(E)$ a sentence-count functional. Let $I_{[2,4]}$ denote the indicator of belonging to the target band. Then a bandpass coordinate can be defined as

$$x_{\text{band},A}(a) \;:=\; 10 \cdot I_{[2,4]}\big(\mathrm{sent}(E_A(a))\big), \qquad x_{\text{band},B}(a) \;:=\; 10 \cdot I_{[2,4]}\big(\mathrm{sent}(E_B(a))\big),$$

or, if one prefers robustness to sentence-tokenization artifacts, by a soft band on word counts:

$$x_{\text{band},A}(a) \;:=\; 10 \cdot \mathbf{1}\{m \leq \mathrm{wc}(E_A(a)) \leq M\},$$

for chosen bounds (m, M) compatible with the prompt.

For "personal arc" we fix a small dictionary $R \subset \Sigma^*$ of reflection cues (e.g. *realized, learned, mistake, revised, iterated, feedback, persist*) and define

$$x_{\text{arc}}(a) \;:=\; 10 \cdot \min\Big\{1, \frac{1}{\rho} \sum_{w \in R} \mathbf{1}\{w \text{ appears in } E_A(a)\}\Big\},$$

with a threshold ρ. For "plan specificity" in the contribution answer we fix a dictionary P of concrete action tokens (e.g. *talk, problem session, reading group, outreach, workshop*) and analogously set

$$x_{\text{plan}}(a) \;:=\; 10 \cdot \min\Big\{1, \frac{1}{\pi} \sum_{w \in P} \mathbf{1}\{w \text{ appears in } E_B(a)\}\Big\}.$$

Finally, math-term density is computed relative to a math lexicon $M \subset \Sigma^*$ and can be normalized either by word count or by a fixed scale:

$$x_{\text{math},A}(a) := 10 \cdot \min\Big\{1, \frac{1}{\mu} \sum_{w \in M} \mathbf{1}\{w \text{ appears in } E_A(a)\}\Big\},$$
$$x_{\text{math},B}(a) := 10 \cdot \min\Big\{1, \frac{1}{\mu} \sum_{w \in M} \mathbf{1}\{w \text{ appears in } E_B(a)\}\Big\}.$$

The point is structural: each coordinate is an explicit, bounded functional of the raw text, and downstream learning can assign different weights to reflection, specificity, and mathematical content. Such decompositions are standard in learning-to-rank contexts where the supervision is noisy and the model must remain interpretable [5, 8].

Example 3.2 (Essay decomposition)**.** Consider two applicants with similar GPAs and coursework. Applicant 1 writes E_B that mentions a specific action ("I will run a weekly problem session on analytic number theory and distribute a short handout"); then x_{plan} is near its cap even if $x_{\text{math},B}$ is moderate. Applicant 2 writes a vague E_B ("I want to contribute by helping people learn"); then x_{plan} is small regardless of other strengths. A linear ranker can therefore separate "intent" from "plan clarity" by placing weight on x_{plan} without conflating it with math-term signals.

4. Linear scoring model

We assign a score to an application by

$$s_w(a) \;=\; \langle w, x(a) \rangle \;=\; \sum_{k=1}^{d} w_k \, x_k(a), \qquad w \in \mathbb{R}^d.$$

The sign and magnitude of w_k indicates how feature k affects rank. Because $x(a)$ is bounded coordinate-wise, weights are directly interpretable.

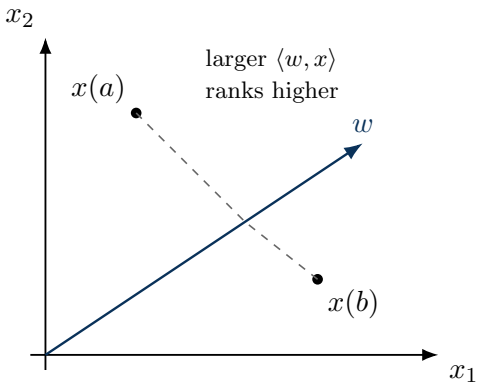

FIGURE 2. Geometric view in $d = 2$: applications are points $x(a)$; ranking compares their projections onto w.

5. Pairwise training objective

We obtain training data as labeled pairs. Each label says which of two applications should rank higher.

Definition 5.1 (Pairwise dataset)**.** A pairwise dataset is a multiset

$$\mathcal{D} = \{(i, j, y_{ij})\},$$

where i, j index applicants, and $y_{ij} \in \{0, 1\}$ indicates whether i should rank above j ($y_{ij} = 1$ means $i \succ j$).

5.1. **Bradley–Terry / logistic model.** Define the score difference

$$\Delta_{ij}(w) \;=\; s_w(a_i) - s_w(a_j) \;=\; \langle w, x(a_i) - x(a_j)\rangle.$$

A standard probabilistic model posits

$$\mathbb{P}(i \succ j \mid w) \;=\; \sigma(\Delta_{ij}(w)), \qquad \sigma(t) = \frac{1}{1 + e^{-t}}.$$

This is the Bradley–Terry model (in logit form) [1] and is closely related to RankNet [5].

5.2. **Negative log-likelihood loss.** For a single labeled pair (i, j, y_{ij}), the negative log-likelihood is

$$\ell_{ij}(w) = -y_{ij} \log \sigma(\Delta_{ij}(w)) - (1 - y_{ij}) \log\big(1 - \sigma(\Delta_{ij}(w))\big).$$

Equivalently, letting $z_{ij} = 2y_{ij} - 1 \in \{-1, +1\}$,

$$\ell_{ij}(w) = \log\Big(1 + \exp(-z_{ij}\Delta_{ij}(w))\Big),$$

the logistic loss on the signed margin.

We regularize with ℓ_2 weight decay:

$$L_\lambda(w) \;=\; \frac{1}{|\mathcal{D}|} \sum_{(i,j,y)\in\mathcal{D}} \ell_{ij}(w) + \frac{\lambda}{2}\|w\|_2^2, \qquad \lambda \geq 0.$$

Lemma 5.2 (Gradient)**.** *Let $d_{ij} = x(a_i) - x(a_j)$. Then*

$$\nabla \ell_{ij}(w) = \big(\sigma(\Delta_{ij}(w)) - y_{ij}\big)\, d_{ij},$$

and hence

$$\nabla L_\lambda(w) = \frac{1}{|\mathcal{D}|} \sum_{(i,j,y)\in\mathcal{D}} \big(\sigma(\langle w, d_{ij}\rangle) - y\big) d_{ij} + \lambda w.$$

Proof. Differentiate $\ell_{ij}(w)$ using $\sigma'(t) = \sigma(t)(1-\sigma(t))$ and the chain rule:

$$\frac{\partial}{\partial w}\big(-y\log\sigma(\Delta)\big) = -y\frac{\sigma'(\Delta)}{\sigma(\Delta)}\,d = -y(1-\sigma(\Delta))\,d,$$

$$\frac{\partial}{\partial w}\big(-(1-y)\log(1-\sigma(\Delta))\big) = -(1-y)\frac{-\sigma'(\Delta)}{1-\sigma(\Delta)}\,d = (1-y)\sigma(\Delta)\,d.$$

Summing gives $(\sigma(\Delta)-y)d$. Adding λw yields the stated formula. □

Proposition 5.3 (Convexity)**.** *$L_\lambda(w)$ is convex in w for $\lambda \geq 0$, and is strictly convex if $\lambda > 0$.*

Proof. Each term $\ell_{ij}(w) = \log(1+\exp(-z\langle w, d\rangle))$ is a composition of a convex function $\log(1+e^t)$ with an affine map $t = -z\langle w, d\rangle$, hence convex. The sum preserves convexity. The quadratic $\frac{\lambda}{2}\|w\|^2$ is strictly convex when $\lambda > 0$, implying strict convexity overall. □

5.3. **Optimization.** Because L_λ is smooth and convex, standard gradient methods converge reliably [9]. In practice, we run a fixed number of epochs over the dataset and update

$$w^{(t+1)} \;=\; w^{(t)} - \eta\,\nabla L_\lambda\big(w^{(t)}\big),$$

with step size $\eta > 0$ and small λ to reduce overfitting under limited pair counts. This is sufficient at our scale ($d \approx 21$; $|\mathcal{D}|$ on the order of 10^1–10^3).

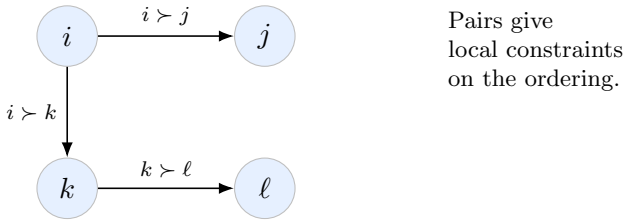

FIGURE 3. Pairwise labels define directed preferences that the model fits in a maximum-likelihood sense.

6. Calibration and reporting

The pairwise objective in **??** learns a weight vector $w \in \mathbb{R}^d$ for the *logit* differences $s_w(a) - s_w(b) = \langle w, x(a) - x(b)\rangle$. Consequently, the absolute scale of the raw score

$$s_w(a) = \langle w, x(a)\rangle \in \mathbb{R}$$

is not intrinsically meaningful: multiplying w by a constant rescales logits and can be partially offset by the regularization and the distribution of labeled pairs. For applicant-facing reporting we therefore apply an explicit *monotone calibration* that preserves order while mapping to a bounded range.

6.1. **Percentile-to-**0–10 **mapping.** Fix a session (or a reference set) containing a pool of applications $\mathcal{P} = \{a_1, \ldots, a_n\} \subset \mathcal{A}$ and consider the multiset of raw scores $\{s_w(a_m)\}_{m=1}^n$. Define the empirical CDF

$$\widehat{F}_w(t) \;:=\; \frac{1}{n}\sum_{m=1}^{n} \mathbf{1}\{s_w(a_m) \leq t\}, \qquad t \in \mathbb{R}.$$

The calibrated score is then the empirical percentile scaled to $[0, 10]$,

$$\widehat{s}(a) \;:=\; 10\,\widehat{F}_w\big(s_w(a)\big) \in [0, 10].$$

This transformation is order-preserving: for any $a, b \in \mathcal{P}$,

$$s_w(a) \le s_w(b) \quad \Longrightarrow \quad \widehat{s}(a) \le \widehat{s}(b),$$

and it is invariant under strictly increasing reparameterizations of the raw scale. In practice we store both $s_w(a)$ and $\widehat{s}(a)$, since the former is the object optimized by the learning-to-rank loss while the latter is the user-facing quantity. When no pool file is available we still log $s_w(a)$ and may define a degenerate "solo" score $\widehat{s}(a) = 10$ (or $\widehat{s}(a) = 5$) purely to keep the interface deterministic; such a score should be treated as non-comparable across sessions.

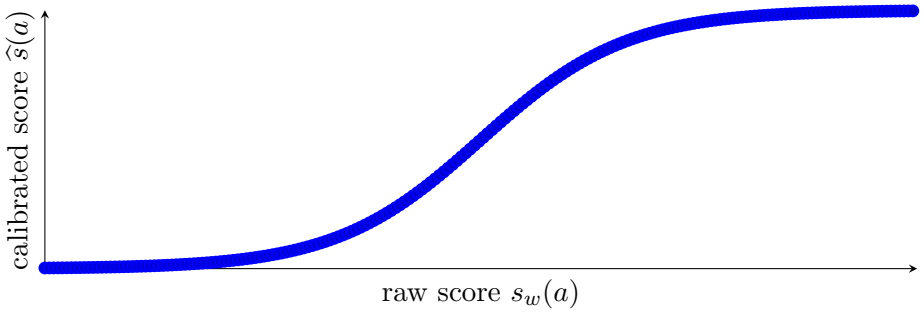

FIGURE 4. A monotone calibration map $\mathbb{R} \to [0, 10]$. In deployment $\widehat{s}(a)$ is computed from the empirical CDF of a pool, i.e. a step function approximation to a CDF; the smooth curve is a schematic proxy.

6.2. **Keyword-only summary and decomposition reporting.** Because the feature map is decomposed into semantically meaningful coordinates, we can expose interpretable slices of $x(a)$ without altering the trained ranking function. In particular, from Section 3.3 define the keyword-only summary

$$\mathrm{KW}(a) \;:=\; \frac{1}{4}\Big(x_{\mathrm{research}}(a) + x_{\mathrm{teach}}(a) + x_{\mathrm{leader}}(a) + x_{\mathrm{awards}}(a)\Big) \in [0, 10].$$

This quantity is *not* a replacement for $s_w(a)$: it ignores essays, structured eligibility signals, penalties, and any interaction the learned weights impose. Rather, $\mathrm{KW}(a)$ is a diagnostic scalar that isolates one input subsystem (resume keyword evidence) and can be used in reviewer dashboards, sanity checks, and post-hoc audits. A useful practice is to store the tuple

$$\big(s_w(a),\ \widehat{s}(a),\ \mathrm{KW}(a),\ x(a)\big)$$

in the submission database so that one can answer questions of the form "is this applicant's high score driven mainly by keywords or by essays/coursework?" without reverse-engineering a monolithic score.

7. CONCLUSION

The big thing here is that there is the linear scoring function $s_w(a) = \langle w, x(a)\rangle$ trained from comparisons via regularized pairwise logistic risk, which yields a convex objective and an interpretable parameterization. The deterministic feature map decomposes an application into bounded, human-legible signals (structured eligibility fields, validated coursework, resume-derived keyword evidence, and short-answer heuristics). Finally, a monotone percentile calibration produces a stable 0–10 reporting score while preserving rank order, and auxiliary summaries such as $\mathrm{KW}(a)$ provide transparency without entangling the learned objective.

Acknowledgements

I wish recognize my co-founder, Dmytro Prosiannikov, a fellow Columbia physics classmate, for deep discussions that directly shaped the original conception of this ML project and its implementation direction.

References

[1] R. A. Bradley and M. E. Terry, *Rank analysis of incomplete block designs: I. The method of paired comparisons*, Biometrika **39** (1952), 324–345.
[2] R. D. Luce, *Individual Choice Behavior: A Theoretical Analysis*, Wiley, 1959.
[3] R. L. Plackett, *The analysis of permutations*, Journal of the Royal Statistical Society. Series C (Applied Statistics) **24** (1975), 193–202.
[4] T. Joachims, *Optimizing search engines using clickthrough data*, Proceedings of KDD (2002), 133–142.
[5] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, *Learning to rank using gradient descent*, Proceedings of ICML (2005), 89–96.
[6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, *Learning to rank: From pairwise approach to listwise approach*, Proceedings of ICML (2007), 129–136.
[7] T.-Y. Liu, *Learning to Rank for Information Retrieval*, Foundations and Trends in Information Retrieval **3** (2009), no. 3, 225–331.
[8] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed., Springer, 2009.
[9] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer, 2006.