Mitigating Position Bias in Click Predictor Models: A Novel Downsampling Approach for Enhanced Accuracy and Efficiency

Phuong Ha Nguyen
phuongha.ntu@gmail.com
eBay
San Jose, California, USA

Djordje Gligorijevic
djordjegligorijevic90@gmail.com
Meta
San. Jose, California, USA

Hung Nguyen
hungnguyen@ebay.com
eBay
San Jose, California, USA

Sheng Shen
shengshen@ebay.com
eBay
San Jose, California, USA

Zhenke Xi
jexi@ebay.com
eBay
San Jose, California, USA

Abraham Bagherjeiran
abagherjeiran@ebay.com
eBay
San Jose, California, USA

Abstract

In the rapidly evolving landscape of digital information retrieval and online advertising, response predictor models have become indispensable tools for understanding user behavior and optimizing content delivery. These models aim to estimate the likelihood of a user response on a displayed item, such as a search result or advertisement, thereby enabling more efficient and targeted allocation strategies. However, the accuracy and reliability of response predictor models are significantly challenged by position bias – a phenomenon where the position of an item in a list influences the probability of user responding to it though either click, conversion, etc., independent of its relevance or quality.

This paper introduces a novel downsampling method designed to mitigate the effects of position bias while addressing the challenges posed by large-scale interaction data and resource constraints. Our approach preserves all positive samples and filters out a substantial amount of poor negative samples, maintaining the integrity of valuable information necessary for accurate predictions. By leveraging the insight that item response probabilities should be uniform across positions in the absence of bias, we propose a random filtering strategy that optimizes the preservation of valuable interaction data.

We demonstrate the effectiveness of our method through extensive offline and online experiments, showing that it not only complements existing bias mitigation techniques but also enhances model accuracy in resource-constrained environments. Our findings suggest that integrating this downsampling method with new training strategies leads to improved prediction performance.

CCS Concepts

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Keywords

supervised learning, pointwise ranking, click prediction, position bias, e-commerce

ACM Reference Format:
Phuong Ha Nguyen, Djordje Gligorijevic, Hung Nguyen, Sheng Shen, Zhenke Xi, and Abraham Bagherjeiran. 2025. Mitigating Position Bias in Click Predictor Models: A Novel Downsampling Approach for Enhanced Accuracy and Efficiency. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '25).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

In the dynamic realm of digital information retrieval and online advertising, response prediction models [1–3, 5, 6, 6, 9–13, 16, 17] have become essential for deciphering user interactions and enhancing content delivery strategies. These models estimate the likelihood of a user engaging with an item – whether through a click, conversion, comment, or other feedback – enabling more precise and effective marketing decisions. However, the presence of position bias – a tendency for items at the top of a list to receive more responses regardless of their actual relevance – poses a significant challenge to the accuracy and reliability of these models [5, 13, 17, 17].

Position bias occurs primarily because users naturally focus on items at the top of a list, resulting in a higher response probability for these items compared to those further down [6, 10]. This bias can distort the training data for response prediction models, causing the predictions to reflect patterns of user attention rather than the relevance of the true item. Therefore, addressing position bias is critical for improving the predictive accuracy and fairness of these models.

To counteract position bias, researchers have developed various techniques [5, 17]. Propensity scoring methods adjust click data by estimating the likelihood of clicks based on item position. Counterfactual learning frameworks infer click probabilities in scenarios devoid of position bias. Additionally, unbiased learning-to-rank algorithms incorporate position bias correction directly into the training process, enhancing the model's ability to generalize to unbiased click scenarios. Another research approach involves utilizing position information as a feature in modeling, or incorporating

KDD '25, Aug 3 – 7, 2025, Toronto, ON, Canada

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/10.1145/nnnnnnn.nnnnnnn

related data, such as the logit of the click-through rate for a position, into the training process via the base margin vector, which provides prior knowledge for model training [4].

High-quality response prediction models heavily rely on the user–item interaction data. The most valuable data includes positive samples (i.e. clicks or conversion) and good or high-quality negative samples (items seen but skipped and thus not clicked). The abundance of interaction data on popular recommender, search, and online advertising platforms poses a challenge, as resource constraints (e.g., limited RAM, CPU, GPU, or latency) make it difficult to process these large volumes. A practical solution is to randomly sample a sufficient training dataset from the raw data. However, this can result in the loss of valuable information, particularly positive and high-quality negative samples. This challenge motivates the search for solutions that preserve valuable information while allowing for smaller, resource-efficient training datasets. In this paper, we introduce a novel downsampling method that significantly reduces the size of the training dataset while preserving all positive samples and filtering out a substantial amount of weak negative samples (items displayed but not seen and not clicked, typically in lower slots), with minimal loss of valuable negative samples. This method effectively addresses position bias and complements existing techniques. We also propose several training strategies that yield slight improvements in model accuracy over current methods.

Our approach is based on the observation that, on platforms with a sufficient number of items, the top items displayed to users are generally of similar quality standard. In the absence of position bias, the click-through rates across all positions would be equal. Therefore, we aim to formulate a strategy for randomly filtering out negative samples to equalize the post-downsampling click-through rates across positions to that of position (let say) 1, which is not affected by position bias.

We present a range of experimental results to support our findings.

This paper is organized as follows: Section 2 reviews position bias models and existing mitigation approaches. Section 3 details our downsampling method. Section 4 introduces new training strategies based on our downsampling method. We present experimental results in Section 5 and conclude in Section 6.

2 Background

2.1 Setup

In response to a user visit or specific user query, online platforms surface a number of items to users, optimized for user engagement, thereby enhancing it's likelihood and optimizing the utilization of digital real estate. This real estate comprises of positions starting from first which typically gains the most traction [17]. The online platforms meticulously track user behavior, including metrics such as clicks and purchases, for subsequent analysis aimed at enhancing the user's shopping experience by showing the most relevant and impactful items on high valued positions. In the context of click prediction modeling for instance, researchers compile data on user clicks, along with associated user and item feature information, to construct predictive models. The objective is to leverage the comprehensive dataset to train the model, ensuring no critical information regarding user-item interactions is overlooked. Consequently, multi-position data is utilized in the training phase of the model. However, to evaluate the model's effectiveness, researchers may restrict the test dataset to data collected from positions that consistently capture user attention, regardless of the items displayed. Frequently, only data from position–1 may be considered as the test dataset.

2.2 Position Bias Model

We consider the following widely accepted model to address the position bias issue. The position bias model posits that the observed click, represented as a Bernoulli variable C, is contingent upon two latent Bernoulli variables: E and R. Here, E signifies the event where a user examines a document at a specific position k, while R denotes the event where a document d is relevant to a query q. The model is mathematically expressed as:

$$P(C = 1 \mid q, d, k) = P(E = 1 \mid k) \cdot P(R = 1 \mid q, d),$$

where $P(C = 1 \mid q, d, k)$ represents the probability of a click on document d displayed at position k given query q. $P(E = 1 \mid k)$ is the probability of examining position k, and $P(R = 1 \mid q, d)$ is the probability that document d is relevant to query q. This model assumes that examination is solely dependent on position, while relevance is dependent only on the query and document. For simplicity, we define the notations: $\theta_k = P(E = 1 \mid k)$ and $\gamma_{q,d} = P(R = 1 \mid q, d)$.

In scenarios where $E = 1$, the relevance R is fully observed through C. Conversely, when $E = 0$, C is invariably 0, leaving the relevance of the document unknown. A clicked document i indicates both examination and relevance.

In this study, the primary objective of multi-position click prediction modeling is to develop a model that can accurately predict the relevance of a given document (d) for a query q at position 1 (i.e. $k = 1$) with high accuracy based on the information of all collected clicks with users and documents features. using information from all collected clicks and user and document features. We outline some common approaches to tackle this challenge.

2.3 Naïve Approach

The most straightforward approach assumes that position bias is not a concern, meaning users examine all positions before deciding whether to click. In this scenario, $\theta_k = P(E = 1 \mid k)$ is always equal to 1 for every position. As a result, we train a click predictor model using all available training data points. However, this method may result in lower model accuracy because it does not account for the negative effects of position bias. In practice, the model tends to generate lower predicted click-through rates (CTR), represented as $\gamma_{q,d} = P(R = 1 \mid q, d)$, due to an inflated number of negative samples.

2.4 Position Information k as a Feature

In this approach [17], the position bias θ_k and relevancy $\gamma_{q,d}$ are not treated separately. Instead, complex models like XGBoost [4] are utilized to address the issue of position bias by incorporating position position (i.e. parameter k) as a feature alongside other user and item features. This allows the model to distinguish between clicks

driven by content relevance and those influenced by positional prominence. By doing so, this approach mitigates bias, refines ad placement strategies, and enhances model interpretability by clarifying how position affects click probabilities. As a result, it leads to more accurate predictions and optimized advertising strategies.

2.5 Base Margin

In multi-position click prediction modeling, base margins in Gradient Boosting implementations such as XGBoost [4] serve as initial prediction scores that can incorporate prior knowledge, such as historical click-through rates (CTR) or outputs from other models. To tackle position bias via base margin, in practice we set it's value as the logit of position 1's CTR, i.e.

$$\text{logit}(p_k) = \log\left(\frac{p_k}{1-p_k}\right),$$

where p_k is the k-th position CTR.

This initialization is crucial for capturing complex interdependencies between positions and adjusting for position biases, such as higher click probabilities for top positions. By setting appropriate base margins, the model can start closer to expected outcomes, enhancing convergence speed and accuracy. Specfically, for XGBoost, when used for classification tasks, typically optimizes a logistic loss function. Providing the base margin in log odds aligns with this loss function, potentially leading to better convergence and performance.

In practice, this approach outperforms using position information k as a feature because it does not rely on positions being in consecutive order. When positions are not sequential, the model's performance can suffer, whereas position-based margin information does not require strict positional correlation, thus maintaining effectiveness.

2.6 Expectation-Maximization (EM)

In [17], the authors introduce an innovative method utilizing the Expectation-Maximization (EM) algorithm to precisely estimate position bias, denoted as $\theta_k = P(E = 1 \mid k)$, and incorporate it into a multi-position click predictor model. The primary contribution of this work is the creation of an unbiased learning-to-rank framework that successfully distinguishes genuine relevance signals from position bias present in click data. By leveraging the EM algorithm, the model progressively enhances its estimations of both user preferences and position bias, thereby enabling more accurate ranking predictions.

The EM approach outlined in the paper requires a minimum of two training cycles (see *Algorithm* 1 Regression-based EM), with the outer cycle dedicated to EM and the inner cycle focused on comprehensive model training. Consequently, this method demands considerably more training effort compared to techniques based on base margin or position information (refer to Section 2.4). The paper illustrates that this method is less effective than those utilizing position information. Given its practicality limitations and lower effectiveness relative to position information-based methods, this approach is included here solely for the sake of completeness in the review.

Due to page constraints, we present two additional, less practical approaches in Appendix A.

3 Effective Position Debiasing Methods

3.1 Motivation

Typically, designers aim to capture all essential interactions between users and items in the training datasets, leading to the use of multi-position click predictor modeling despite position bias challenges. However, this objective becomes less feasible as the training data volume increases. Designers must allocate more resources, such as RAM, CPU, GPU, and time, to train the model, or alternatively, they may resort to randomly sampling a sufficient subset of the data if resources are constrained. Expanding training resources demands greater investment, which may not always be feasible. Consequently, random downsampling becomes a more practical and favored option, though it results in the loss of valuable interaction data between users and items, which is undesirable in many cases. This situation motivates us to develop a new training method capable of delivering strong modeling performance with a significantly smaller training dataset, while preserving all crucial information from the original dataset.

3.2 Key Idea

As discussed earlier, users are likely to skip ads displayed in lower positions, resulting in a large number of negative samples (i.e., samples with a click value of 0) that offer little insight into user-ad interaction. This prompts us to eliminate these negative samples. However, we lack an effective method to differentiate valuable negative samples (i.e., those skipped after user consideration) from the rest. Consequently, our best approach is to randomly select a sufficient number of negative samples, along with all positive samples (those with a click value of 1), to form the training dataset. Since downsampling is applied only to negative samples, all valuable positive samples are retained, given that the number of positive samples is significantly smaller than negative ones in online advertising datasets. This suggests that our approach to downsampling negative samples is optimal for preserving information.

Our proposed strategy for downsampling negative samples is based on the following assumptions:

Assumption 1. *Users consistently examine the ads displayed in position* 1. *In other words, position* 1 *is free from position bias.*

Assumption 1 is considered mild because users are naturally inclined to view ads in position 1 due to its prominent visual placement.

Assumption 2. *If all ads displayed across all positions are of average quality and there is no position bias, then the click-through rates for all positions are equal.*

Assumption 2 is more comprehensive than Assumption 1 because ads are ranked by their quality. However, it becomes more realistic when the number of ads far exceeds the number of available positions, and when the advertising platforms are well-known, attracting numerous high-quality, competitive advertisers. Additionally, Assumption 2 is more acceptable when ads are not ranked solely by relevance, such as in systems using generalized second

price auctions [8], where ads are ranked based on a combination of their bid value b and their predicted click-through rate score $pCTR$, i.e. $pCTR \times b$.

These two assumptions lead us to randomly select a sufficient number of negative samples at each position so that, combined with their positive samples, the click-through rates (CTRs) for each position match the CTR of position 1. The following theorem outlines our proposed downsampling algorithm.

Theorem 3.1. *Let ρ_k represent the click-through rate of position k. For each negative sample at position k, if it is sampled with probability dr_k, where*

$$dr_k = \frac{\rho_k}{1-\rho_k} \times \frac{1-\rho_1}{\rho_1},$$

then all positions will have the same click-through rates, equal to ρ_1.

Proof. We consider position k and begin by initializing several key variables:

- a is set to the number of positive samples, which is equivalent to the number of clicks.
- b represents the number of remaining negative samples after sampling.
- c denotes the number of removed negative samples after sampling.
- The total number of negative samples n is defined as $n = b+c$.
- The total number of impressions i is calculated as $i = a+b+c = a+n$.

We first calculate number of remaining negative samples b based on the number of positive samples a and position-1 CTR ρ_1. Our goal is to make the CTR of position k after sampling equal to ρ_1 or we want to have $a/(a+b) = \rho_1$. This gives

$$b = \frac{a \cdot (1-\rho_1)}{\rho_1}.$$

The downsampling rate, dr_k, is calculated to adjust the distribution of negative samples relative to positive samples. The formula for dr_k is expressed in several equivalent forms:

$$\begin{aligned} dr_k &= \frac{b}{b+c} = \frac{b}{n} = \frac{b}{i-a}, \\ &= \frac{a}{i-a} \cdot \frac{1-\rho_1}{\rho_1} = \frac{a/i}{1-a/i} \cdot \frac{1-\rho_1}{\rho_1}, \\ &= \frac{\rho_k}{1-\rho_k} \cdot \frac{1-\rho_1}{\rho_1}. \end{aligned}$$

□

This rate aids in balancing the dataset by effectively reducing the impact of position bias, allowing the learning model to focus on more relevant data.

In practice, the position CTRs, denoted as ρ_ks, are calculated by randomly sampling a sufficient number of training data points. They are determined using the formula

$$\rho_k = \frac{\text{total number of clicks at position k}}{\text{total number of impressions at position k}}$$

.

This approach allows us to efficiently compute dr_ks values without needing to use the entire raw dataset.

Given a raw dataset $\mathcal{D}$, a downsampled training dataset $\mathcal{S}$, constructed using our proposed method, is outlined in Algorithm 1.

Algorithm 1 Negative Downsampling Method

Require: Position CTRs ρ_k, $k = 1, \ldots, n$ and raw training dataset $\mathcal{D}$
Ensure: Training dataset $\mathcal{S}$
  $S \leftarrow$ empty set
  for each data point d in the raw training dataset $\mathcal{D}$ **do**
    if d is a positive sample **then**
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{d\}$
    else
      if d is a negative sample **then**
        if d is at position k **then**
          $R \leftarrow \text{Random}(0, 1)$
          if $R < dr_k$ **then**
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{d\}$
          end if
        end if
      end if
    end if
  end for

Initially, an empty training dataset $\mathcal{S}$ is established. We then proceed to examine each data point in the raw dataset $\mathcal{D}$ individually. According to Algorithm 1, if a data point is a positive sample at position k, it is added to the training dataset $\mathcal{S}$ with a probability of 1. If it is not a positive sample, it is included in $\mathcal{S}$ with a probability of dr_k.

Theorem 3.2. *Let ρ_1 and a_k represent the click-through rate of position 1 and the number of positive samples at position k. The number of data points in downsampled training dataset $\mathcal{S}$ after applying Algorithm 1 is equal to*

$$\frac{1}{\rho_1}(\sum_k a_k).$$

Proof. The proof is derived straightforward based on the proof of Theorem 3.1, i.e. the total number of data points in $\mathcal{S}$ is equal to the sum of all positive samples a_k and remaining negative samples b_k over all positions k and the fact $b_k = a_k \frac{1-\rho_1}{\rho_1}$. □

In practice, due to the impact of position bias, the lower positions have a smaller number of positive samples. As a result, the number of data points in the training dataset $\mathcal{S}$ is significantly less than in the raw dataset $\mathcal{D}$.

4 Click Predictor Modeling with new Downsampling Method

In the previous section, we introduced a novel downsampling method specifically targeting negative samples. The primary objective of this method is to equalize the position CTRs across all positions to match that of position 1, thereby mitigating the negative effects of position bias. Once the downsampled training dataset $\mathcal{S}$ is obtained, we propose several new model training methods.

Position	1	2	3	4	5	6	7	8	9	10	11	12	13
dr_k	1.0	0.70	0.54	0.46	0.17	0.15	0.14	0.09	0.08	0.076	0.073	0.059	0.059

Table 1: Downsampling Rates dr_k for Different Positions

Position-independent downsampling (PID). This is the simplest approach, where we treat all data points as if they were sampled from position 1. This combined dataset serves as the training data for any existing training methods we prefer to use.

Position-dependent downsampling (PDD). This method is based on the observation that, after downsampling, while the position CTRs are equalized, the true CTRs of items at different positions are not necessarily equivalent. Items displayed in lower positions that receive clicks likely have a higher true CTR than those in higher positions. Therefore, we do not treat all data points as if they were sampled from position 1. Instead, we handle them separately to help models learn the true CTR of items more accurately. To achieve this, we incorporate the position information k as a feature.

5 Experiments

	AUC	Logloss	Bias
Ex. 1 (naïve training)	0.00000	0.00000	0.00000
Ex. 2 (base margin XGB)	0.01349	-0.00799	-0.67149
Ex. 3 (position k as a feature)	0.01547	-0.00855	-0.60399
Ex. 4 (PID)	0.01354	-0.00807	-0.65636
Ex. 5 (PDD)	0.01575	-0.00859	-0.60500

Table 2: Comparative analysis of performance models.

Note: Performance metrics adjusted by subtracting the values of Experiment 1 from each experiment. This provides a differential view of the performance changes across experiments.

5.1 Datasets

We utilize datasets from a prominent online marketplace where hundreds of millions of buyers and sellers engage daily. This e-commerce platform offers an extensive array of products catering to a wide range of preferences and needs, along with programs that enable sellers to advertise their items. As a result, the platform must choose relevant ads or products from billions of possibilities to present to users. After identifying relevant items, they are ranked based on factors like click-through rate (CTR) and bid value. Developing effective click prediction models for item ranking is essential. The ranking process employs comprehensive user and item features for model training, with click values used as labels to build the click predictor. All user-item interactions are logged for system analysis, leading to a vast amount of data due to high levels of user and ad engagement. This creates opportunities for designers to craft robust click predictor models, but also introduces challenges, particularly in extracting valuable insights from data points with click values of 1 or from all positions, given constraints such as RAM, CPU, GPU, and time.

In our case, we gather *a dataset from two weeks of platform logs from an ads program* that details user and ad interactions, referred to as the raw dataset $\mathcal{D}$. We utilize a one-day log from position 1 as our testing dataset $\mathcal{T}$. The data points in $\mathcal{D}$ are derived from specific positions, with 13 positions considered for click predictor modeling. Each data point includes user and item features for training, with click values assigned as labels. Due to data publication restrictions at the e-commerce platform, the statistics are not published but they show that different positions exhibit varying behaviors.

5.2 Metrics and Offline Experiments

To measure the accuracy of click predictor models for a given test dataset, The Receiver Operating Characteristic curve (ROC Curve) [7] or ROC-AUC, Logloss [15] and Bias are in use. In the context of click prediction, bias metric quantifies the degree to which the predicted probabilities deviate from the observed click rates. Specifically, the metric is defined as follows.

$$\text{Bias} = \sum_{i=1}^{N} p_i / \sum_{i=1}^{N} y_i$$

where y_i is the actual label (0 or 1), p_i is the predicted probability, and N is the number of samples.

We study the impact of our new proposed negative downsampling methods to the click predictor modellings. It is important to note that our approach for click predictors can be directly applied to sale predictors. XGBoost models [4] are used as main models. We implement the following *offline* experiments.

- **Experiment 1**: The unaltered training dataset, $\mathcal{D}$, is used without incorporating position information k as a feature. This serves as the baseline for model performance evaluation.
- **Experiment 2**: Similar to Experiment 1, the dataset $\mathcal{D}$ is employed without position information k. However, position CTR logit values are utilized as base margin values, providing insight into the effectiveness of base margin XGB.
- **Experiment 3**: The dataset $\mathcal{D}$ is used with position information k included as a feature, assessing the performance of XGB when position information is considered.
- **Experiment 4**: The training dataset $\mathcal{D}$ undergoes downsampling using our proposed method, applying the dr_k as detailed in Table 1. Subsequently, position-independent downsampling (PID) is applied for modeling.
- **Experiment 5**: Similar to Experiment 4, the dataset $\mathcal{D}$ is downsampled using the dr_k in Table 1, but position-dependent downsampling (PDD) is employed, incorporating position information k as a feature.

The outcomes of all experiments are detailed in Table 2. Due to data publication restrictions of the platform, we have obfuscated the results by adjusting the performance metrics through subtraction of the values from Experiment 1 for each experiment. This approach offers a differential perspective on the performance variations across the experiments. Higher AUC values indicate better ranking algorithms, whereas lower logloss and bias metrics suggest improved ranking performance.

Note: Given the page constraints of the paper, we have elaborated on the significance of the improvements in AUC and Logloss in the Appendix, as they may seem minor due to the nature of the problem. B.

5.3 Online tests

We carried out an AB test to demonstrate the effectiveness of our new downsampling method. Two XGB models were trained: one using the base margin method and the other with our new downsampling technique position-dependent downsampling (PDD). The base margin method is currently utilized because ads are displayed in non-continuous positions, making it the most practical solution for addressing position bias issues. The AB test spanned three weeks and consisted of two phases. In the first phase, data was collected over one week to build calibration for both models. The second phase ran for two weeks to compare the performance of the two models. The base margin XGB model served as the Control, while the downsampling-based XGB model was the Treatment. Both models received equal traffic during the AB test. Calibration of XGB model scores was necessary because we discovered that scores from the base margin XGB model were biased, despite the bias metric being favorable in offline experiments. We suspect this bias is due to the limitations of the base margin method in addressing position bias issues. To ensure the calibrated model scores of the base margin XGB model did not negatively impact system performance, we implemented calibration using the Isotonic Regression method [14]. The findings from the AB test, which assess the enhancement of the downsampling model compared to the base margin model, are presented in Table 3. Metrics such as ROC-AUC, Logloss, Bias, Click-Through Rate (CTR), Sale Through Rate (STR), and Gross Merchandise Bought (GMB) are included. A notable difference was observed between offline and online results. This discrepancy can be attributed to the conventional offline approach used in constructing XGB models, which may not fully account for the complex and dynamic nature of the entire e-commerce platform. Thus, this divergence is understandable.

AUC	Logloss	Bias	CTR	STR	GMB
+2.3%	-3.8%	+77.84%	+7.72%	+14.5%	+18.10%

Table 3: Relative Enhancement of Downsampling PDD XGB Model Compared to Base Margin XGB Model (95% Statistical Confidence)

A significant discovery highlighting the effectiveness of our novel downsampling method in tackling position bias is the bias metric of the model scores, which remains nearly optimal both before and after calibration due to our deliberate application of bias correction on the training dataset. Essentially, we enforce a constraint on bias within the training dataset. Before calibration, the scores are around 1.032, improving to 1.007 afterward, nearing the ideal bias metric. This is notably different from the base margin model scores, which are 1.337 before calibration and 1.035 after. In this method, we depend on an algorithm to address bias issues, but this algorithmic approach falters when encountering new, unseen items. Moreover, incorporating position k as a feature without the downsampling technique (as in Experiment 3) shows promising AUC improvements in offline analysis, yet it still faces bias challenges similar to those found in the base margin approach. Thus, by combining the new downsampling method with position k as a feature, we effectively alleviate position bias issues, as demonstrated by enhancements in both bias metrics and ranking performance, reflected through the ROC-AUC metric, in both offline and online environments.

6 Conclusion

This paper tackles the crucial issue of position bias in click predictor models. Our innovative downsampling method improves efficiency by reducing the training dataset size, preserving all positive samples, and eliminating many poor negative samples linked to position bias. This enhances training efficiency under resource constraints while retaining essential information for accurate model predictions.

Acknowledgments

GPT-4o is utilized to enhance the paper by rephrasing text and performing grammar checks throughout the document.

References

[1] Aman Agarwal, Miroslav Dudik, Chengliang Wu, John Langford, and Robert E Schapire. 2019. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 5–14.

[2] Aman Agarwal, Xuanhui Wang, Cheng Li, Ming Wang, Michael Bendersky, and Marc Najork. 2019. Estimating position bias without intrusive interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 474–482.

[3] Qingyao Ai, Jiafeng Guo, and W Bruce Croft. 2018. Unbiased learning to rank with unbiased propensity estimation. *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval* (2018), 385–394.

[4] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 785–794.

[5] Aleksandr Chuklin, Ilya Markov, and Maarten De Rijke. 2022. *Click models for web search*. Springer Nature.

[6] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*. 87–94.

[7] Richard O. Duda, Peter E. Hart, and David G. Stork. 2001. *Pattern Classification* (2nd ed.). Wiley-Interscience.

[8] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American Economic Review* 97, 1 (2007), 242–259.

[9] Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten de Rijke. 2013. Reusing historical interaction data for faster online learning to rank for IR. In *Proceedings of the sixth ACM international conference on Web search and data mining*. 183–192.

[10] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. 154–161.

[11] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 781–789.

[12] Tie-Yan Liu. 2009. Learning to rank for information retrieval. In *Proceedings of the 2nd international conference on Information and knowledge management*. 3–10.

[13] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*. 521–530.

[14] Tim Robertson. 1988. Order restricted statistical inference. (1988).

[15] Claude E. Shannon. 1948. A Mathematical Theory of Communication. *Bell System Technical Journal* 27, 3 (1948), 379–423.

[16] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 115–124.

[17] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. 2018. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 610–618.

A Background

A.1 Randomization

A common approach to mitigating position bias in recommender systems is to randomly shuffle item positions for a small percentage of user traffic and train models exclusively on this data [9, 16]. Implementations of this approach include complete shuffling, *RandTopN* where rank 1 and rank p are randomly swapped half the time [11], or *RandPair* where adjacent pairs are randomly shuffled [17]. While this method directly addresses position bias, it comes at a significant cost to the platform despite attempts at developing various strategies as it can negatively impact user experience and may conflict with allocation policies, such as those used in ad systems. Due to these limitations, we do not consider randomization a viable strategy for addressing position bias in this study.

A.2 Inverse Propensity Weighting

Addressing position bias can be achieved by incorporating position information during model training using inverse propensity weighting (IPW), which effectively shifts the data distribution toward a position-agnostic distribution [11, 17]. Naive implementations in this approach use an item's position p as input and include: (1) absolute positioning: $w = 1+p$, (2) log transformation: $w = \log(\beta+p)$, and (3) ratio-based weighting: $w = 1 - \frac{1}{\alpha+p}$, while more advanced approaches for IPW are discussed in Section 2.6. However, in our experiments, these strategies led to subpar model performance, and therefore, we do not report their results.

B Significance of Enhancing Model Performance

Although the improvements in AUC and Logloss observed in Table 2 may appear modest, they are of considerable practical importance due to the use of original, comprehensive datasets that are highly imbalanced with respect to clicks. Typically, non-click impressions vastly outnumber those with click values, often by a factor of 20 to 100, depending on the slot position. This imbalance necessitates substantial effort to accurately predict impressions with a click value of 1, making even slight gains in AUC and Logloss noteworthy.

To substantiate this point, we conducted three experiments under identical conditions, including the same training, validation, and test datasets, setup parameters, and feature sets. We trained three XGB models with depths of 5 (experiment 1), 10 (experiment 2), and 15 (experiment 3), each with 500 trees. The results of these experiments are detailed in Table 4. According to the design of XGB models, increasing the depth exponentially enhances the models' capacity, as evidenced by the increase in model sizes. However, the observed improvements in AUC, Logloss, and Bias remain relatively small. This is understandable given the complex nature of the datasets, as previously explained.

Depth	AUC	Logloss	Bias	Model Size (Mb)
05	0.000000	0.000000	0.000000	1.9
10	0.000725	-0.000010	-0.007141	24.9
15	0.002273	-0.000233	-0.007849	112.0

Table 4: Differences in performance metrics for model depths 10 and 15 compared to depth 5. The differences are calculated by subtracting the metrics of depth 5 from those of depths 10 and 15. A positive difference in AUC indicates improved performance, while a negative difference in Logloss and Bias suggests better performance.