We appreciate Reviewers' comments and will improve presentation and reference lists based on the following responses.

(Reviewer 1: The effect of rounding: e.g., a cluster of points.) Rounding may have some effect, but this effect does not ruin the impact of our contributions, i.e., construction of the fast quantum algorithm and evaluation of its runtime. **After all, any implementation of kernel methods by computer with bits requires rounding, and in our setting, we resolve a cluster of points by rescaling, which is equivalent to increasing precision of rounding without rescaling.** As discussed in Sec. 3.1, our rescaling keeps the learning problem invariant. Then under standard assumptions in signal processing where such implementation works well, it should be straightforward to show our algorithm also works well.

(Reviewer 1: Difference from Ref. [20].) Compared to Ref. [20], the significance of our contributions is that **our algorithm in the limit of good approximation (as $N, G \to \infty$) is provably optimal** up to a logarithmic gap as shown in Ref. [7]; also, its runtime is as fast as poly-logarithmic in N, G as shown in Sec. 3.2. Regarding the first point of the review comment, the integral operator may be unknown in practice, but our algorithm converges to sampling from the leverage-score distribution of the integral operator as $N, G \to \infty$ whereas **Ref. [20] does not converge to this optimal distribution in any limit**. As for the second point of the review comment, Ref. [20] can achieve the learning since it preserves the kernel (Gram) matrix, but the optimality of Ref. [20] using the kernel matrix (i.e., a gap from a lower bound of the required number of random features) is unknown in general; in contrast, Ref. [7] using the integral operator proves the optimality, and **our algorithm based on Ref. [7] achieves this optimality** in the limit of $N, G \to \infty$.

(Reviewer 2: What quantum RAM (QRAM) do we need for the task? The algorithm has various black boxes.) We need a quantum oracle $\mathcal{O}_\rho$ defined explicitly in the first paragraph of Sec. 3.2. **This oracle is the only black box in our quantum algorithm; putting effort to make our algorithm explicit, we avoid any other QRAM.** This input model is the same as quantum recommendation systems in Ref. [37] and is implementable feasibly as discussed in Sec. B of Supplemental Material, which we omit from the main text since it is well established in Ref. [37].

(Reviewer 2: What (why) can we use the QSVT without the sparse or low-rank assumption?) We can avoid sparse and low-rank assumptions **because we explicitly decompose the (non-sparse and full-rank) operator $\mathbf{\Sigma}_\epsilon$ into addition and the multiplication of diagonal (i.e., sparse) operators and QFTs**, so that we can construct an efficient implementation of the block encoding of $\mathbf{\Sigma}_\epsilon$. This main technical contribution of our paper is explained in the last paragraph of Sec. 3.1. Remarkably, our technique does not directly use Lemmas 48–50 of arXiv:1806.01838 since these lemmas require sparse and low-rank assumptions. While we could implement $\mathbf{\Sigma}_\epsilon$ by addition and multiplication of block encodings of the diagonal operators and QFTs, the presentation of these additions and multiplications may become complicated since we have multiple block encodings to be combined. For simplicity of the presentation, we use the block encoding of the POVM operator (Lemma 46 of arXiv:1806.01838) at the technical level to represent how to combine all the block encodings and QFTs as one circuit, as shown in Figs. 1 and 2 of Supplemental Material.

(Reviewer 2: How does the quantum Fourier transform come into the overall algorithm?) Our algorithm uses QFTs **for implementing the block encoding of $\mathbf{\Sigma}_\epsilon$ and for applying $\mathbf{F}_D^\dagger$ in preparing the quantum state (14).**

(Reviewer 2: On the practical side of the quantum algorithm.) Rather than 5 qubits in Ref. [47], our algorithm aims at applications at large scales. In contrast to Ref. [47], we prove that our algorithm achieves the exponential speedup. Thus, as explained in Introduction, **our algorithm is a convincing candidate for killer applications of universal quantum computers in the long run; after all, large-scale machine learning will be eventually needed in practice.**

(Reviewer 2: What's the difference between classical and quantum?) The existing classical algorithm calculates description of probability distribution by matrix inversion, and then perform sampling. In contrast, **our quantum algorithm does not estimate the classical description of the distribution represented by the amplitude of quantum states.**

(Reviewer 3: Considering Δ is very small, then $\mathrm{polylog}(1/\Delta)$ can be big, and $O(D)$ does not hold anymore.) **The precision factor** $\mathrm{polylog}(1/\Delta)$ **is ignorable in practice** while we explicitly write it for correctness of our runtime analysis. For example, consider two D-dimensional real vectors x and y. Computers with bits can use fixed-point number representation to represent each real element with precision Δ using $O(\log(1/\Delta))$ bits (e.g., 64 bits). In this case, multiplication of two elements requires $O(\mathrm{polylog}(1/\Delta))$ runtime, and calculation of inner product of x and y requires $O(D\,\mathrm{polylog}(1/\Delta))$ runtime, but the factor $\mathrm{polylog}(1/\Delta)$ is practically ignored.

(Reviewer 3: It essentially has some connection to low rank.) Using a concentration inequality, Ref. [7] shows that a requirement for any algorithm using random features to achieve the learning with reasonable runtime and accuracy is given in terms of the degree of freedom, in particular, by the bound (5) in our paper. However, **this requirement (5) does not imply low rank of operators used in our algorithm,** as discussed in the second paragraph of Sec. 3.2.

(Reviewer 3: The results have been in [39][42][44]) The novelty of our contributions is that we construct an exponentially faster QML algorithm that is free from sparsity and low-rank assumptions. As discussed in Introduction, **this has been challenging but crucial in QML, and none of Refs. [39] [42] [44] achieves this.** We hope that the above explanations about all the questions help with eliminating the concerns about validity and importance of our results.