Investigating Smaller Datasets on Contextual Multi-armed Bandit Learning Algorithms

Nick Chow
California State University, Long Beach
nick.chow01@student.csulb.edu

Alexander Dung
California State University, Long Beach
alexander.dung@student.csulb.edu

I. How to Run the Program

We utilize the existing experiment from the paper "Contextual Linear Bandits under Noisy Features: Towards Bayesian Oracles" [1]. We utilize a Python dependency manager called `poetry` to initialize the dependencies as it is a straightforward installation and execution. We followed the dataset execution pattern to add the wine dataset and generate our results.

The only library needed to run this program is poetry. Poetry is a dependency manager and will take care of everything else. To run the program, you must have Python version 3.9 or 3.10. After downloading the repository, use the following commands to prepare the virtual environment and run the experiment.

```
python -m pip install poetry
python -m poetry env use 3.10
#Initialize the poetry virtual env
poetry install
poetry run python scheduler_wine.py
```

Note that due to the large number of iterations and random seeds, the experiment can take from half an hour up to an hour to complete. You can change the `seed_list` in the scheduler files to reduce the number of seeds to test in the experiment.

To change the dimensions of the preprocessing AutoEncoder, go to `real_preprocess/wine_aetrain.py` and change the `NUM_EMB_DIM` variable.

If you make any changes to the AutoEncoder, delete all files in the `real_models` folder to ensure that the AutoEncoder is re-trained during the experiment.

Code Disclosure: The code for the experiments originates from [1] and modified to work with the wine quality dataset [2]. The original repository can be found in [1].

II. Introduction

MULTI-ARMED Bandits (MAB) algorithms provide a unique approach to address the problem of choosing a best solution. More specifically, MAB algorithms are a form of online-reinforcement algorithms that contextualize the problem of choosing one solution from many. Given N choices, the MAB has N arms and at each round is tasked to select an arm that corresponds with the most rewarding choice. As the algorithm continues, the MAB selectively chooses the best arm that gives the highest reward, with the least amount of regret and this arm does not affect the behavior of the other arms. In traditional MAB algorithms, the MAB is unaware of any features of the solution but can only observe the reward. As a result, MAB algorithms have many variants including Combinatorial Multiarmed Bandits [3], Constrained Contextual Bandit (CCB) [4], and in the scope of this paper, Contextual Multi-armed Bandit Algorithms (CMAB) [5]. These variations curtail the basic MAB problem for more complex scenarios including choosing the best combination of solutions and choosing solutions based on additional features, or context.

Contextual Multi-armed Bandit algorithms are designed to choose the best solution given contextual information. One specific application of these algorithms is in advertising. Consider a web application that wants to serve relevant advertisements to its users. Traditional MAB algorithms can do this but cannot pull from features. This means that the algorithm is only working off information about the classes of advertisements it can show, and is unable to consider information such as the specific user it is advertising to, and the contents of the advertisements it can show. CMAB algorithms enable the bandit to choose a solution based on additional features represented by a d-dimensional vector. By providing additional features, the aforementioned web application may use the CMAB algorithm to choose more relatable ads for their users with by observing the context of both features from the user and the advertisement. Advertisers and researchers measure the effectiveness of a CMAB algorithm's ability to serve advertisements to users through the metric click-through-rate (CTR).

In this experiment, we apply a wine quality dataset [2] to a few Contextual Multi-Armed Bandit algorithms evaluated by Kim et al. under large datasets [1]. More specifically, we are comparing the CTR of the MovieLens100k dataset on the Optimism in the Face of Uncertainty Linear (OFUL) Bandit algorithm, Contextual Linear Bandits on Bayesian Features (CLBBF), and Random Policy to the wine dataset as both make the assumption users click on the advertisement if the rating/sensory preference is above or below a single value [6] [1]. Furthermore, we are observing how CMAB algorithms perform on smaller datasets and the effect of reducing data dimensionality with the AutoEncoder affects the CTR.

In the MovieLens 100k dataset, Kim et al. determined a movie advertisement will always be clicked on if it had a rating greater than or equal to 5 on a scale from 1 to 5,

[1] https://github.com/junghunkim7786/contextual_linear_bandits_under_noisy_features

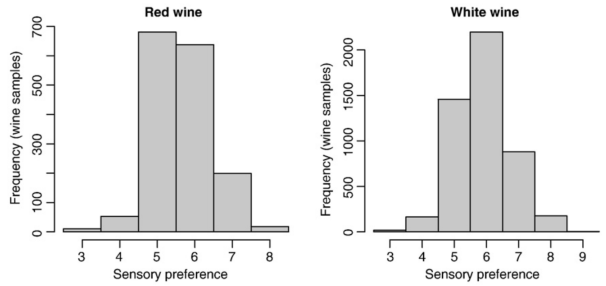

Fig. 1. Sensory Preference Distribution for Both Red and White Wines [7]

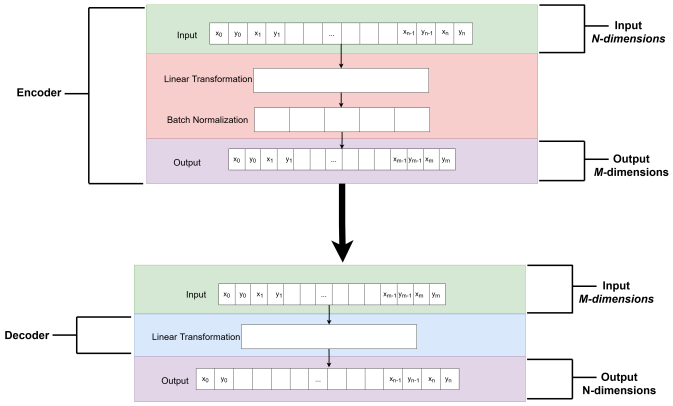

Fig. 2. AutoEncoder Architecture

otherwise the user would ignore the advertisement. If it was clicked because its rating was greater than or equal to 5, a reward of 1 is assigned, otherwise 0 is given to the bandit. To measure the CTR with this dataset, we apply a similar approach and presume that all users are predisposed to clicking on an advertisement for wine if it's sensory preference is rated as a 7 or higher on a scale from 1-9. To form the reward for these MAB algorithms, if the bandit selects a wine whose quality is less than 7, the reward is 0, otherwise the reward is 1.

III. Dataset

The wine dataset includes 6497 samples of both red (1599 samples) and white wine (4898 samples) [7]. Each sample of wine contains 11 features relating to its physiochemical properties including sulphates, pH, total sulfur dioxide, alcohol, volatile acidity, free sulfur dioxide, fixed acidity, residual sugar, chlorides, density, citric acid. Each wine is paired with a target label of its sensory preference ranging from 3-9. The distributions for the red and white wine are similar but the white wine's distribution is shifted one point higher. Both types of wine have a normal distribution with the majority of sensory preference for both types of wine laying between 5-7. Figure 1 visualizes the distributions for both red and white wine. In "Contextual Linear Bandits under Noisy Features: Towards Bayesian Oracles", they introduce the CLBBF algorithm and add missing features to the dataset before running the CLBBF algorithm and comparing it to OFUL and random sampling [1]. Kim et al. utilize an autoencoder to reduce the dimensionality of the MovieLens 100k dataset. To preserve the purpose and appropriately evaluate this algorithm on a smaller dataset with less features, we apply these same preprocessing approaches. In the case of the wine quality dataset, we reduce the dimensionality from 11 features to 10 and 7 to observe the effects of data-dimension reduction of the AutoEncoder.

AutoEncoders are a form of neural networks that take a d-dimensional input and encode the input to a latent vector with dimension less than d [8]. We reduced the batch size by a factor of 10 to the existing AutoEncoder training process to curtail towards the Wine Quality dataset as we have signiciantly less samples. This same batching was applied to the AutoEncoder for the MovieLens100k dataset and we did not see a reduction in the CTR convergence.

One thing to note, because Contextual MAB algorithms are a form of Online-Reinforcement Learning and increasingly grow better as they are introduced to more samples, there is not much concern over splitting and evaluating the model on train, test, and validation splits as is usually done in other classification models. As such, we do not perform these procedures and the usefulness of the model is instead evaluated by how quickly it converges to an optimal solution (peak CTR rate).

IV. Methods

The CMAB algorithms are ran 10 times using unique predetermined seeds to ensure consistency among the experiment. The rewards for 10^5 rounds are accumulated and averaged across all 10 seeds over each run.

During the preprocessing stage, the AutoEncoder training process introduces some noise by erasing entries and features using vector masking with probability $p = 10\%$. As we will see later in the results, the noise introduced significantly impacts the CTR of both OFUL and Algorithm 1. For the AutoEncoder, the model is trained before running the training for the MAB.

The AutoEncoder is trained using Pytorch [9] and consists of an Encoder Layer consisting of a Sequential model with a linear transformation of an input size of d-dimensions, and output size of 10 and 7 for the wine dataset and 32 for all other datasets. Following the linear transformation a 1D Batch Normalization is applied with the number of channels corresponding to the number of embedding dimensions. Next, the decoder layer follows and applies another linear transformation with the input size being the number of embedded dimensions and the output being the original dimensionality. The architecture of this AutoEncoder is shown in Figure 2. The AutoEncoder is trained for 500 epochs with the Adam optimizer set at a learning rate of $1e^{-4}$, weight decay of $1e^{-5}$ and with Mean Squared Error as the loss function. The input data is batch size is set as a function of $d/1000$. Once training is complete, the preprocessing stage ends and the CMAB algorithm training commences.

The OFUL algorithm aims to maximize the total reward, calculated as the sum of inner products $\sum_t \langle X_t, \theta_* \rangle$, where X_t

Algorithm 1 Contextual Linear Bandits on Bayesian Features (CLBBF)

Initialize: $Z \leftarrow 0_{d\times d}$, $\xi \leftarrow 0_{d\times 1}$, $n \leftarrow 0$, $i \leftarrow 1$.
$n \leftarrow n+$ the total number of non-missing entries in $x_{a,1}$ for all $a \in \mathcal{A}$.
$Z \leftarrow Z + \sum_{a\in\mathcal{A}} x_{a,1}x_{a,1}^\top$, $\xi \leftarrow \xi + \sum_{a\in\mathcal{A}} x_{a,1}$.
Select a_1 uniformly at random in $\mathcal{A}_1$.
Observe reward y_1.
for $t = 2$ **to** T **do**
  $n \leftarrow n+$ the total number of non-missing entries in $x_{a,t}$ for all $a \in \mathcal{A}$.
  $Z \leftarrow Z + \sum_{a\in\mathcal{A}} x_{a,t}x_{a,t}^\top$, $\xi \leftarrow \xi + \sum_{a\in\mathcal{A}} x_{a,t}$.
  Estimate parameters:
  $\widehat{p} \leftarrow \frac{\max\{1,n\}}{tdK}$, $\widehat{\nu} \leftarrow \frac{1}{tK\widehat{p}}\xi$,
  $\widehat{\Sigma} \leftarrow \frac{1}{tK}Z \circ \left(\frac{\widehat{p}-1}{\widehat{p}^2}I_{d\times d} + \frac{1}{\widehat{p}^2}1_{d\times d}\right) - \widehat{\nu}\widehat{\nu}^\top$.
  Estimate Bayesian features:
  $\widehat{z}_{a,t} \leftarrow [1; \overline{x}(\widehat{\nu}, \widehat{\Sigma}, x_{a,t})]$ for $a \in \mathcal{A}_t$.
  if $t = 2^i$ **then**
    Update selected Bayesian features:
    $\widehat{z}_{a_s,s} \leftarrow [1; \overline{x}(\widehat{\nu}, \widehat{\Sigma}, x_{a_s,s})]$ for all $s \in [t-1]$.
    $i \leftarrow i+1$.
  end if
  Select $a_t = \arg\max_{a\in\mathcal{A}_t} \max_{\theta\in\mathcal{C}_{t-1}} \langle \widehat{z}_{a,t}, \theta\rangle$.
  Observe reward y_t.
end for=0

is the feature vector of the arm taken at time t. The vector θ_* is unknown, so the algorithm maintains a confidence set C_{t-1} for that vector, and at each step selects the estimate $\theta \in C_{t-1}$ and arm X that maximizes the inner product $Y_t = \langle X, \theta\rangle$. The confidence set is calculated as an ellipsoid with center (where for any matrix A, $A_{1:t}$ is the matrix containing the rows $A_1, A_2, ..., A_t$):

$$\widehat{\theta}_t = (X_{1:t}^\top X_{1:t} + \lambda I)^{-1} X_{1:t}^\top Y_{1:t}, \tag{1}$$

where λ is a regularization parameter. Abbasi-Yadkori et al. prove that the true vector θ_* lies within the ellipsoid with high probability [6]. Therefore, the confidence vector is likely to contain θ_* and the algorithm is likely to select the arm that maximizes the reward from that vector.

The CLBBF algorithm is a modification of the OFUL algorithm that assumes the dataset is noisy, and attempts to learn both the distribution of the noise and the true distribution of the data to make better decisions. CLBBF functions the same as OFUL, but the confidence set and the score for each arm is calculated differently. The algorithm for CLBBF is detailed in pseudocode in Algorithm 1, where for two vectors x and y with length n and m respectively, $[x; y]$ is the concatenation of x and y to form a vector of length $n + m$.

For each algorithm, CTR is calculated for each time step as the sum of all rewards gained until that time step, divided by the number of the current time step. While the name is not intuitive for the wine dataset, we can consider the CTR as a general measurement of accuracy of the algorithm in selecting wine with a sensory preference rating greater than or equal to 7. After the experiment is complete, the results

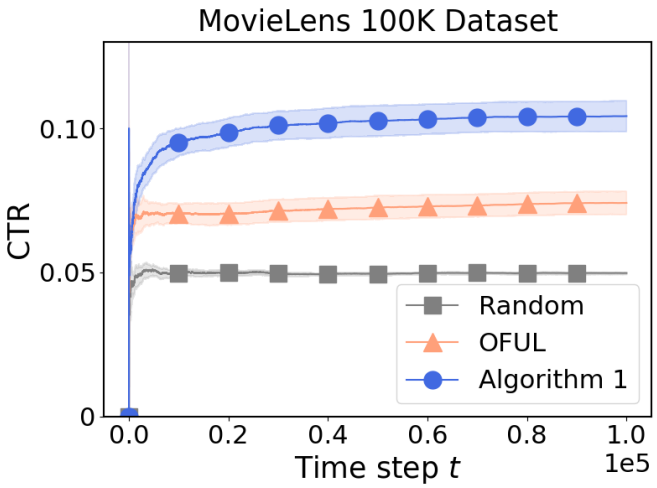

Fig. 3. MovieLens 100k clickthrough rates with encoded inputs [1]

are written to file and the plotting program reads the files, averages the CTR across the predetermined random number generation seeds for each algorithm, then generates a plot of each algorithm's average CTR versus time. The results are saved as `numpy` arrays, to load and view the data if needed.

V. Results

We present the results of our experiment on the wine dataset with and without feature encoding in Figures 5/4 and 6, respectively. Data points for all three graphs can be found in Table I. In the experiment with the encoded wine dataset, we see a notable deviation from the experiments shown in the original CLBBF proposal [1]. The original paper showed that across the Avazu, Taobao, and MovieLens 100k (visualized in Figure 3) experiments, OFUL converged to a higher point than the random algorithm, and CLBBF converged higher than OFUL. On the other hand, Figure 5 shows that the wine dataset encoded to 7 dimensions results in OFUL converging to a lower CTR than the random algorithm, and CLBBF converging at a much slower rate than with the original datasets. Furthermore, the figure shows that CLBBF has a much larger standard deviation across the random seeds (evidenced by the large shadow behind the line) at each time step. This shows that the algorithm is failing to converge within the experiment's time limit. The algorithms perform worse on the dataset encoded to 10 dimensions as seen in Figure 4. In this situation, CLBBF fails to reach a CTR above 0.08 throughout the whole experiment, unlike the previous experiment. When using the unprocessed wine dataset, the algorithms' behavior differs significantly from the encoded dataset. Figure 6 shows that the algorithms' CTR are an order of magnitude higher than the encoded dataset. Furthermore, OFUL quickly converges to a high CTR while by the end of the experiment, CLBBF reaches a lower CTR than OFUL. This behavior deviates from the experiments performed on other datasets. We believe that the algorithms' improved behavior on the raw dataset could mean that either the AutoEncoder does not learn the latent space of the wine dataset correctly, or the

TABLE I
AVERAGE CTR PER TIMESTEP FOR WINE DATASET

	Encoded Input				Raw Input	
	$d = 10$		$d = 7$			
t	CLBBF	OFUL	CLBBF	OFUL	CLBBF	OFUL
11111	0.052898	0.035376	0.062176	0.037140	0.362716	0.758126
22222	0.054331	0.036084	0.069761	0.037389	0.413401	0.822814
33333	0.053789	0.036662	0.066176	0.038000	0.456327	0.846676
44444	0.056094	0.037807	0.075547	0.038508	0.487594	0.859721
55555	0.053933	0.037807	0.080544	0.039099	0.525832	0.867240
66666	0.055275	0.038289	0.085129	0.039787	0.555696	0.872462
77777	0.063715	0.038717	0.099654	0.040267	0.571275	0.876428
88888	0.062441	0.038953	0.100642	0.040584	0.593300	0.879667
99999	0.060862	0.039338	0.096087	0.040996	0.615097	0.882212

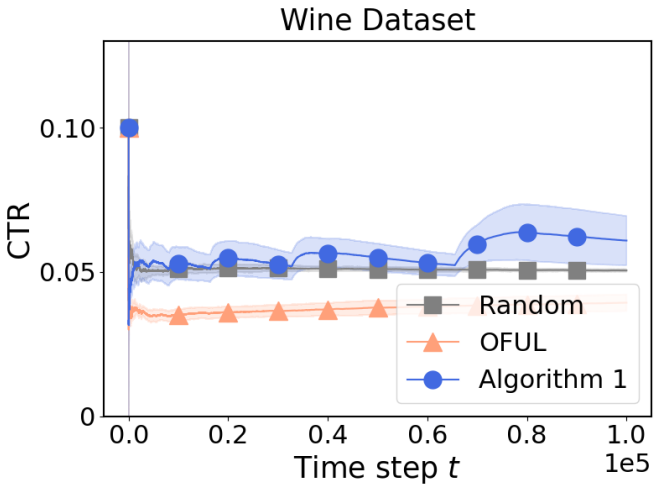

Fig. 4. Wine CTR with encoded inputs ($d = 10$)

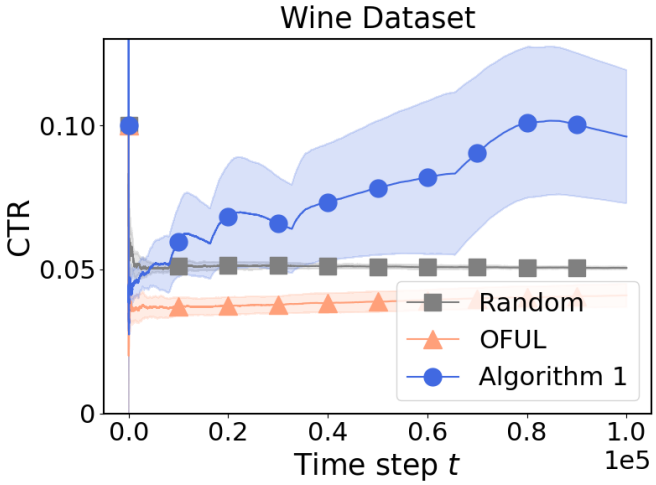

Fig. 5. Wine CTR with encoded inputs ($d = 7$)

dimension reduction dropped features that made the samples separable. The wine dataset seems to have some effect on the behavior of CLBBF compared to the datasets it was originally tested on. Regardless of whether the dataset was encoded or not, both CLBBF and OFUL behaved differently from the experiments on other datasets. We believe that one potential

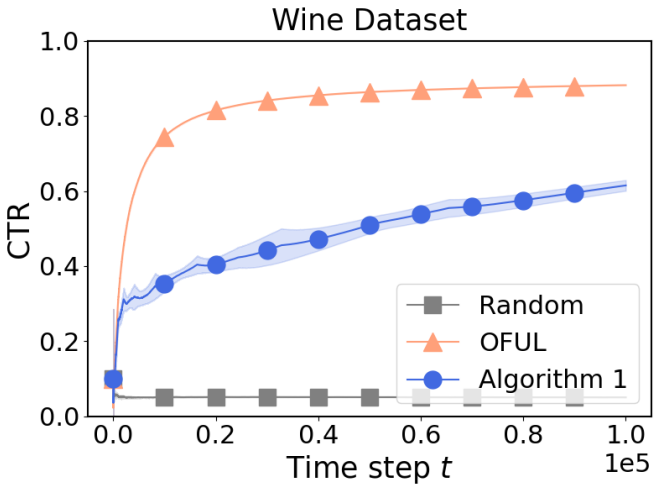

Fig. 6. Wine CTR with Raw Inputs. Note the different y-axis scale.

cause for this could be that the dataset consists of continuous features rather than categorical. Other characteristics about the dataset such as size, distribution of reward/non-reward samples, and separability are also potential causes for this phenomenon.

- **Size—**The Wine dataset is considerably smaller than the other datasets. While this could have a negative effect on the learning ability of the AutoEncoder and CMAB algorithms, the experiment is set up such that the algorithm is tested using only 1000 samples of reward and non-reward each, which the wine dataset is large enough to fulfill.
- **Distribution of Samples—**The wine dataset has a different ratio of reward to non-reward samples. This problem was also solved by the same experiment setup used to prevent dataset size from affecting the experiment.
- **Separability—**It is possible that the wine dataset is linearly separable without any processing, meaning that encoding the dataset could potentially cause loss of discriminating features and make the dataset inseparable. However, this does not explain why CLBBF underperforms on the unprocessed dataset compared to OFUL.

Another potential factor is that these algorithms are developed as Bayesian models. Kim et al. note that the algorithm assumes

that both the true feature vector of the dataset and the noise applied to the samples follow a Gaussian distribution [1]. Therefore, if either the features of the dataset or the noise that the dataset has are not Gaussian, the performance of the algorithm may be affected.

VI. Conclusion

In this paper, we tested the CLBBF and OFUL contextual bandit algorithms proposed by Kim et al. [1] against the wine dataset, to explore the effects that a different dataset could have on their performance. We found that the algorithms performed differently against the wine dataset compared to the datasets that they were tested against in the original paper, CLBBF failing to converge to a constant CTR but still performing better than OFUL, and OFUL consistently converging to a CTR worse than a random selection policy. CLBBF's failure to converge within the experiment time frame suggests that something sets the Wine dataset apart from the Avazu, Taobao, and MovieLens datasets it was originally tested against. We believe that the dataset's features play a role in this change, though which characteristics affect the algorithm's performance are unknown. Potential future work could include testing other datasets with continuous features which are known to not be linearly separable, or known to not follow a Gaussian distribution. Testing these datasets would help prove or disprove which characteristics of the features could be causing this change in algorithm performance.

References

[1] J.-H. Kim, S.-Y. Yun, M. Jeong, J. Nam, J. Shin, and R. Combes, "Contextual linear bandits under noisy features: Towards bayesian oracles," in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, F. Ruiz, J. Dy, and J.-W. van de Meent, Eds., vol. 206. PMLR, 25–27 Apr 2023, pp. 1624–1645. [Online]. Available: https://proceedings.mlr.press/v206/kim23b.html

[2] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Wine Quality," UCI Machine Learning Repository, 2009, DOI: https://doi.org/10.24432/C56S3T.

[3] W. Chen, Y. Wang, and Y. Yuan, "Combinatorial multi-armed bandit: General framework and applications," in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 1. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 151–159. [Online]. Available: https://proceedings.mlr.press/v28/chen13a.html

[4] A. Badanidiyuru, J. Langford, and A. Slivkins, "Resourceful contextual bandits," *CoRR*, vol. abs/1402.6779, 2014. [Online]. Available: http://arxiv.org/abs/1402.6779

[5] T. Lu, D. Pal, and M. Pal, "Contextual multi-armed bandits," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 485–492. [Online]. Available: https://proceedings.mlr.press/v9/lu10a.html

[6] Y. Abbasi-yadkori, D. Pál, and C. Szepesvári, "Improved algorithms for linear stochastic bandits," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., vol. 24. Curran Associates, Inc., 2011. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2011/file/e1d5be1c7f2f456670de3d53c7b54f4a-Paper.pdf

[7] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009, smart Business Networks: Concepts and Empirical Evidence. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167923609001377

[8] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," 2021.

[9] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019.