Deep Heckman for Loan Evaluation

Lin William Cong
Cornell University
will.cong@cornell.edu

Yanhong Guo
Dalian University of Technology
guoyh@dlut.edu.cn

Xin Zhao
Dalian University of Technology
zx0712@mail.dlut.edu.cn

Wenjun Zhou
University of Tennessee, Knoxville
wzhou4@utk.edu

Abstract

Despite strong predictive performance, deep learning models for loan classification can suffer from systematic bias when trained on non-representative samples that exclude rejected applicants, as repayment outcomes are only observed for funded applications. We extend the classical Heckman correction method to deep learning architectures by jointly modeling loan selection and repayment outcomes along with their correlation to address unobserved confounding factors. Using a peer-to-peer lending dataset, we demonstrate that our Heckman-corrected deep learning model significantly outperforms benchmark methods, improving both prediction accuracy and generalizability across the complete population of loan applications.

1 Introduction

The FinTech industry increasingly leverages multimodal data sources—combining traditional credit features with text, images, and other rich information—to improve borrower selection accuracy. These diverse data types are processed using deep learning models that convert different modalities into numeric embeddings for joint analysis in online lending platforms.

However, a persistent challenge in loan prediction is sample selection bias, where lenders only observe repayment outcomes for funded loans, creating non-representative training datasets that exclude rejected applications. Since lending decisions must be made at the application stage, models trained solely on funded loans exhibit systematic bias when applied to the broader applicant pool. While sample selection bias has been effectively addressed in traditional predictive models through methods like Heckman correction, this issue remains largely unaddressed in deep learning frameworks. The standard Heckman procedure does not readily extend to high-dimensional, non-linear neural network architectures processing multimodal data.

In this paper, we develop a novel framework that integrates Heckman correction with multimodal transformer models, adapting the two-step Heckman procedure to accommodate the complex parameter space of deep neural networks. Using a large peer-to-peer lending dataset, we demonstrate that

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Generative AI in Finance.

our multimodal Heckman-corrected model significantly outperforms traditional machine learning approaches, providing more accurate and unbiased loan quality predictions.

2 The DeepHeckman Method

In the lending context, the prediction model f takes loan features $\mathbf{x}_i$ as input, and predicts its quality y_i, specified as binary for simplicity ($y_i = 1$ if defaulted, and 1 if repaid). In the historical data, there is a selection model g, which also takes loan features $\mathbf{x}_i$ as input, and predicts z_i, the selection status ($z_i = 1$ if selected for funding, and 0 otherwise). Therefore, at the population level, we have

$$y_i = f(\mathbf{x}_i) + U_{1i}, \tag{1}$$

$$z_i = g(\mathbf{x}_i) + U_{2i}, \tag{2}$$

respectively. Here, U_{1i} and U_{2i} are the error terms, which incorporate unmeasured factors that may confound both selection and the repayment outcome. Only those selected for funding would allow us to observe the repayment status. In this case, if we attempt to use all listing samples to fit Eqn. (1), there would be missing values for y_i for all the listing that were unfunded. Alternatively, if we use only the funded loans to fit Eqn. (1), its model estimates may be subject to a selection bias.

Traditionally, Heckman selection has been used to adjust for sample selection bias [Heckman, 1979]. A commonly implemented technique is to model the error terms as correlated at the sample level (and are independent across samples) with a covariance matrix:

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}. \tag{3}$$

In regression models, Heckman [1979] showed that under a simple assumption (that U_{1i} and U_{2i} follow a joint bivariate normal distribution), it is possible to derive a sample-level Inverse Mills Ratio (IMR) from the selection model g, and include it as an extra term in the prediction model f to account for sample selection bias. We extend this procedure to the scenario that functions f and g are nonlinear (e.g., multimodal transformers).

Since both the selection and outcome model targets are binary, and are imbalanced, we adopt the focal loss [Lin et al., 2017], and extended version of the cross-entropy loss with a focusing parameter $\gamma \geq 0$ and optional class weight $\alpha \in [0, 1]$. Specifically, for the i-th sample, the selection model loss would be

$$\ell_i(g) = -\left[\alpha_g \cdot s_i \cdot (1 - \hat{s}_i)^{\gamma_g} \log(\hat{s}_i) + (1 - \alpha_g) \cdot (1 - s_i) \cdot \hat{s}_i^{\gamma_g} \log(1 - \hat{s}_i)\right], \tag{4}$$

where $\hat{s}_i = \Phi(g(x_i))$ is the model predicted propensity of selection. Similarly, if the i-th loan was selected for funding, the prediction model loss would be

$$\ell_i(f) = -\left[\alpha_f \cdot y_i \cdot (1 - \hat{y}_i)^{\gamma_f} \log(\hat{y}_i) + (1 - \alpha_f) \cdot (1 - y_i) \cdot \hat{y}_i^{\gamma_f} \log(1 - \hat{y}_i)\right], \tag{5}$$

where $\hat{y}_i = \Phi(f(x_i))$ is the model predicted propensity of default when funded. The overall learning objective is then to minimize the following loss function:

$$\arg\min_{f,g,\Sigma} \sum_i \left[s_i \cdot \ell_i(f) + \ell_i(g)\right], \tag{6}$$

where f and g are two deep neural networks (e.g., multimodal Transformer models) for prediction and selection, respectively. Since a loan can only default if it was funded, our composite loss function that simultaneously fits models f and g, and provide estimates of the covariance matrix following a maximized likelihood estimation framework. Note that in this formulation, the covariance structure Σ is also a learnable parameter and thus does not require users to set it. Given samples of $(\mathbf{x}_i, y_i, s_i)$, $\forall i = 1, 2, \ldots, N$, along with hyperparameters, the pseudocode is provided in Algorithm 1.

3 Experimental Analysis

We downloaded all loan requests posted between April 2007 and October 2008 on Prosper, a leading marketplace lending platform in the U.S. Structured features include loan request information (e.g., amount requested), the borrower's credit profile (e.g., debt-to-income ratio), demographic information (e.g., employment status), and the Prosper's overall credit rating.

Prosper's lending process is illustrated in Figure 1, which also summarize our sample sizes.

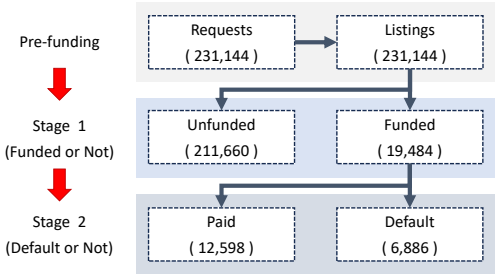

Figure 1: Life Cycle of Online Lending

Our goal is to fit a deep Heckman model as described in Section 2 using deep neural networks as the selection and prediction models. To jointly model text and structured features as predictors, we adopted a multimodal transformer (MMT) [Gu and Budhkar, 2021] as the base architecture (see Section A.2). Further technical details can be found in Section A.3.

3.1 The Predictive Value of Unstructured Data

In Table 1, we compare deep Heckman with the benchmark methods regarding prediction accuracy (using AUC as the metric) and the return on investment (ROI). Soft information is quantified as the performance lift from Model S (using tabular features only) to Model S+T (using both tabular and text features) .

Table 1: Performance Benchmarking by Predicting Default

	AUC			ROI (see details in Section A.5)		
Method[a]	Model S (Structured)	Model S+T (Structured + Text)	Soft Information	Model S (Structured)	Model S+T (Structured + Text)	Soft Information
GLM Only	72.7%	72.8%	0.1%	13.1%	15.0%	1.9%
RF Only	73.4%	73.7%	0.3%	13.1%	14.1%	1.0%
ERT Only	72.5%	73.2%	0.7%	12.0%	13.4%	1.4%
XGB Only	73.7%	74.1%	0.2%	**14.4%**	15.9%	1.5%
Ensemble	**74.1%**	74.5%	0.4%	13.8%	15.7%	1.9%
Multimodal Transformer	74.0%	74.7%	0.7%	14.1%	19.3%	5.2%
DeepHeckman	74.1%	**75.0%**	0.9%	14.4%	**23.9%**	9.5%

[a] GLM = Generalized linear model; RF = Random forest; LGBM = Light gradient boosting machine; ERT = Extremely randomized trees (extra trees); XGB = eXtreme Gradient Boosting; Ensemble = an Ensemble model of all. More details can be found in Section A.4.
Note: Results are reported based on funded loans in the test set. Each result is averaged across 10 replications of the ten-fold mean.

Unsurprisingly, the ensemble model yielded the best AUC among all baseline models, regardless of feature sets. In particular, when using structured features only, the ensemble model reached an AUC of 74.1% (corresponding to an ROI of 13.8%), and when using both structured and textual features,

the ensemble model reached an AUC of 74.5% (corresponding to an ROI of 15.7%). Using the best benchmark model as a baseline, the deep Heckman model reached an AUC of 75.0% and an ROI of 23.9%, outperforming all the ML benchmark models, indicating a strong performance gain.

3.2 Comparison of Selection Bias

Using the multimodal transformer as the base model, we now consider different ways to address the potential selection bias introduced in the funding step. Table 2 shows the results.

- **3-Class**: A multimodal model trained for three-class classification (repay, default, unfunded). The investment decision (on the test set) is to fund those predicted as repay.
- **Subsample**: A multimodal model for binary classification (repay, default) trained using only funded samples. The investment decision (on the test set) is to fund those predicted as repay.
- **Sequential**: A stacked system of two multimmoodal models, where the first is trained to predict funding (funded vs. unfunded), and the second is trained to predict repayment (repay, default) using the funded samples. The investment decision (on the test set) is to first predict whether a loan will be funded using the first model, and then among those predicted as funded, use the second model to predict repayment (fund those predicted as repay).
- **Heckman**: a traditional Heckman correction model based on linear regression. Given its limitations in handling the features, this model only uses features in Panels A, B, and C in Table A1.
- **IMR**: A multimodal model with IMR derived from the selection model, and use it as an input feature in the prediction model.

Table 2: Comparison of Different Ways to Handle Selection Bias

		Predicting Repay				Predicting Default				ROI (Unweighted)		
Method[a]	Accuracy	FPR	Precision	Recall	F1	FPR	Precision	Recall	F1	Top 10%	Top 20%	Top 30%
3-Class	61.18%	**19.34%**	54.09%	65.62%	59.30%	**30.42%**	**47.13%**	12.99%	20.37%	4.92%	4.08%	2.84%
Subsample	69.18%	16.57%	42.81%	83.69%	56.64%	14.14%	22.23%	44.19%	29.58%	4.98%	4.08%	**3.44%**
Sequential	60.02%	18.80%	51.24%	73.59%	**60.41%**	29.03%	39.59%	26.53%	31.77%	4.86%	3.94%	3.31%
Heckman	53.94%	14.70%	**60.08%**	36.02%	45.04%	25.50%	19.62%	**84.81%**	31.87%	4.04%	3.02%	2.03%
IMR	68.60%	18.36%	43.46%	**87.24%**	58.02%	12.02%	19.95%	36.49%	25.80%	4.69%	3.97%	3.11%
DeepHeckman	**70.52%**	16.88%	44.83%	88.27%	59.46%	13.79%	25.26%	49.62%	**33.48%**	**5.02%**	**4.23%**	3.32%

Notes: Results are reported based on funded loans in the test set. All reported results are computed on the test set and averaged across 10 runs.

We can see that **DeepHeckman** achieves the best performance in terms of overall accuracy and F1 when default is the default class. Its performance in F1 when repay is the default class is also quite strong. Its ROIs when selecting the top listings (unweighted, as we ignore the interest rates here) are also competitive when compared with the other methods.

4 Conclusion

This paper addresses sample selection bias in multimodal deep learning for loan prediction by proposing a deep Heckman model that extends traditional Heckman correction to transformer architectures. Our key contribution is an integrated loss that jointly optimizes prediction accuracy for the selection process, outcome prediction, and their correlation to handle unobserved confounders. This framework simultaneously learns the funding decision mechanism and repayment prediction while correcting selection bias. Empirical validation on a large peer-to-peer lending dataset shows that our model outperforms traditional approaches, with higher accuracy and better generalizability across the applicant population. This work opens avenues for applying econometric corrections in modern deep learning, with applications beyond lending to other domains facing selection bias.

References

Ken Gu and Akshay Budhkar. A package for learning on tabular and text data with transformers. In *Proceedings of the 3rd Workshop on Multimodal Artificial Intelligence*, pages 69–73, Jun 2021.

James Heckman. Sample selection bias as a specification error. *Econometrica*, 47(1):153–161, 1979.

Hyungu Kahng, Hyungrok Do, and Judy Zhong. Domain generalization via heckman-type selection models. In *The 11th International Conference on Learning Representations (ICLR)*, 2023.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

Oded Netzer, Alain Lemaire, and Michal Herzenstein. When words sweat: Identifying signals for loan default in the text of loan applications. *Journal of Marketing Research*, 56(6):960–980, 2019.

Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014. URL https://github.com/bayesian-optimization/BayesianOptimization.

A Technical Appendices and Supplementary Material

A.1 Data Feature Descriptions

Table A1: Feature Definitions

ID	Feature	Description
Panel A: Loan Request and Verifiable Information		
1	AmountRequested	Natural logarithm of the loan amount applied by the borrower.
2	AmountDelinquent	Natural logarithm of one plus the total delinquency amount in dollars.
3	DelinquenciesLast7Years	Natural logarithm of one plus the total number of delinquencies recorded in the past seven years.
4	InquiriesLast6Months	Total number of credit inquiries made in the past six months.
5	PublicRecordsLast10Years	Total count of public records, including bankruptcies, civil judgments, and tax liens over the past ten years.
6	PublicRecordsLast12Months	The total count of public records, including bankruptcies, civil judgments, and tax liens from the past year.
7	CurrentCreditLines	Natural logarithm of one plus the borrower's current number of credit lines.
8	OpenCreditLines	Natural logarithm of one plus the number of open credit lines held by the borrower.
9	RevolvingCeditBalance	The natural logarithm of one plus the borrower's revolving credit balance in dollars.
10	BankCardUtilization	Ratio of the total balances owed by the borrower on open bankcards to the total credit limits of those cards.
11	CreditGrade	Letter representing the borrower's credit score grade at the time of application.
Panel B: Self-Disclosed Information		
12	DebtToIncomeRatio	The ratio of a borrower's total debt to their income, expressed as a percentage.
13	HasImage	Indicator showing whether the listing contains an image.
14	IsBorrowerHomeowner	Indicator representing home ownership status (owned vs. not owned).
15	EmploymentStatus	Indicators for employment status: unknown, working full-time, not employed, working part-time, retired, and self-employed.
16	LengthStatusMonths	Duration of the current employment status measured in months as of the listing's creation date.
17	IncomeLevel	Income level indicators, including below $25k, $25k to under $50k, $50k to under $75k, $75k to under $100k, above $100k, and missing information.
18	HasLoanDescription	Whether there is a non-empty description.
Panel C: Writing Quality Features		
19	Text Length	Total word count in the loan description provided.
20	Linguistic Conformity	The number of spelling and grammatical errors divided by the total number of words.[a]
21	Formality	Percentage of words with six or more letters when the loan description is nonempty.
22	Readability	The "simple measure of gobbledygook" (SMOG) readability index.[b]
23	Tone	Sentiment scores of loan descriptions.
Panel D: Lending Outcomes (observed during or after the bidding stage)		
24	BorrowerMaximumRate	The maximum rate the borrower agrees to pay for the loan.
25	BorrowerRate	The rate the borrower pays if the loan closes at this point in time.
26	LenderRate	The rate that lenders would receive on the listing if the loan was to close at this point in time.
27	BidMaximumRate	The Maximum Rate in which a bidder will be able to lend money to create a winning bid.
28	BidCount	The total number of bids received by the listing.
29	FundLabel	An indicator that equals one if a listing is fully funded and becomes a loan.
30	RepayLabel	An indicator that equals one if the loan is fully repaid.

[a] We used the "spellchecker" Python package to detect spelling errors, and the "language-tool-python" Python package to detect grammatical errors.
[b] The SMOG readability is interpreted as the number of years of formal education required to understand the text. Therefore, larger values indicate that the reader requires more education to understand it. To extract the SMOG readability score, we used the "textstat" package.
[c] We measure the tone of loan descriptions using sentiment scores provided by a pre-trained sentiment-BERT model. See https://huggingface.co /michellejieli/emotion_text_classifier. Higher values indicate a more positive tone.

A.2 Multimodal Transformer

To make predictions using both structured features and textual features, we leverage a multimodal transformer model [Gu and Budhkar, 2021], illustrated in Figure A1.

Essentially, different types of features are first preprocessed in their own domains. While traditional features, such as categorical and numeric features, can be represented with a set of nodes, text features pass through a text embedding model, for which we use the commonly used BERT embedding. More specifically, we initialize the text embeddings with a pretrained BERT model from Huggingface, which encodes text into a set of nodes. These traditional feature nodes and text-embedding nodes

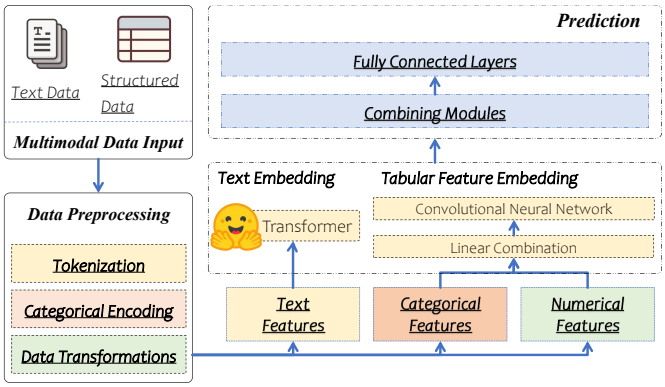

Figure A1: Architecture of the Multimodal Transformer

are then concatenated to create a sample-level numeric vector representation. We then add a multi-layer perceptron (MLP) layer with cross-entropy loss[1] to map the numeric vector representation of the sample onto the outcome class label. The multimodal transformer model is implemented with PyTorch.

In this multimodal transformer model, we let both the text transformer and final MLP layer be learnable. The hyperparameters are listed in Table A2. We used a grid search to determine the best hyperparameters. This multimodal transformer model had approximately 111 million learnable parameters. The tabular branch has 845,344 parameters and the linear MLP step has 395,008 parameters. The other parameters are from the text branch; using BERT embedding, there are 109,744,896 parameters.

Table A2: Hyperparameter Settings for the Multimodal Transformer

Parameter	Description	Range
learning_rate_text	The learning rate of the pre-trained language models.	[3e-6, 3e-5]
learning_rate_tabu	The learning rate of the MLP model, which is employed to learn the features of structured features if activated.	[0.01, 0.015]
weight_decay_text	The regularization part of the language model.	[5e-3, 5e-4, 5e-5]
weight_decay_tabu	The regularization part of the MLP model.	[1e-1, 1e-2, 1e-3]
epoch	Number of epochs to train the model in each round.	[6, 7, 8]
round	Number of rounds to train the model for each group of parameters.	[15]
batch_size	Batch size.	[15, 30]
dropout_rate	The dropout rate of models.	[0.45]
activate_func	Activate functions.	['relu','leaky_relu']

A.3 Deep Heckman Model Estimation

In their empirical testing, Kahng et al. [2023] reported that while it is possible to use different types of models (e.g., tree or deep neural networks) for selection and prediction, using the same type of model seems to work the best. Additionally, Kahng et al. [2023] showed that, while using a one-step procedure to optimize the loss function would be ideal, a two-step procedure was more stable and

[1]Depending on the data and label being used, this model may be used to classify funding status (among all listings) or repayment status among those funded. Because the target variables for both the selection model and the outcome model are binary, the cross-entropy loss is well suited to our needs. However, in other contexts, the outcome variable may be continuous, for which we may use MSE loss instead.

produced an equally good result. Therefore, we adopt a two-step optimization approach for this preliminary exploration.

Algorithm 1: Two-Step Optimization for DeepHeckman

Data: Data $\mathcal{D} = \{(\mathbf{x}_i, y_i, s_i) \in X \times Y \times S, i = 1, \ldots, N\}$, Batch size b, Learning rate γ.

Result: $\hat{f}$, $\hat{g}$, $\hat{\Sigma}$

Randomly initialize $\hat{f}$, $\hat{g}$, $\hat{\Sigma}$;

`/* Step 1: Learn the Selection Model` $\hat{g}$ `*/`

while $\hat{g}$ *not converged* **do**

  $\mathcal{B} \leftarrow BatchSampler(\mathcal{D}, b)$;

  $\hat{g} \leftarrow \hat{g} + \frac{\gamma}{b} \nabla \Sigma_{(\mathbf{x}_i, s_i) \in \mathcal{B}} [s_i \log \Phi(\hat{g}(\mathbf{x}_i)) + (1 - s_i) \log \Phi(-\hat{g}(\mathbf{x}_i))]$,

end

`/* Step 2: Learn the Outcome Model` $(\hat{f}, \hat{\Sigma})$ `*/`

while $\hat{f}$ *not converged* **do**

  $\mathcal{B} \leftarrow BatchSampler(\mathcal{D}, b)$;

  $\hat{f} \leftarrow \hat{f} - \frac{\gamma}{b} \nabla \Sigma_{(\mathbf{x}_i, y_i, s_i) \in \mathcal{B}} \left[s_i \Lambda \left(\hat{f}(\mathbf{x}_i), \hat{g}(\mathbf{x}_i); y_i, s_i; \hat{\Sigma} \right) \right]$,

  $\hat{\Sigma} \leftarrow \hat{\Sigma} - \frac{\gamma}{b} \nabla \Sigma_{(\mathbf{x}_i, y_i, s_i) \in \mathcal{B}} \left[s_i \Lambda \left(\hat{f}(\mathbf{x}_i), \hat{g}(\mathbf{x}_i); y_i, s_i; \hat{\Sigma} \right) \right]$,

end

A.4 Traditional Machine-Learning Classifiers

All benchmark models were implemented using sklearn.To handle class imbalance, we use class weights proportional to inverted sample size (class_weight="balanced"). The other hyperparameters are listed in Table A3, and were optimized following Gaussian processes [Nogueira, 2014] using cross-validation.

Table A3: Hyperparameter Settings for Machine Learning Classifiers

Model	Parameter	Description	Range
GLM	C	Inverse of regularization strength: must be a positive float. Smaller values indicate stronger regularization.	(0.01, 10)
	penalty	Specify the norm of the penalty.	['l1', 'l2']
	solver	Algorithm to use in the optimization problem.	[0, 1]
RF, ERT	n_estimators	Number of trees in forest.	(10, 1000)
	max_depth	Maximum tree depth	(5, 30)
	max_features	Number of features to consider when looking for the best split.	(0.1, 0.999)
	min_samples_leaf	Minimum number of samples required to be at a leaf node.	(1, 25)
	min_samples_split	The minimum number of samples required to split an internal node.	(2, 25)
LGBM, XGB	n_estimators	Number of trees,	(10, 1000)
	max_depth	Maximum tree depth	(5, 30)
	learning_rate	The rate of learning of the model.	(0.01, 0.3)
	min_child_samples	Minimum number of data needed in a leaf.	(5, 50)
	num_leaves	Number of leaves per tree.	(24, 45)

To account for the two-step nature of lending, we incorporate the Inverse Mills Ratio (IMR) to account for selection bias in each of these models. For simplicity, and similar to our multimodal

Heckman procedure, we fit the two models (selection model f and outcome model g). The combined model is applied to the test set to report performance.

When deriving features from text to fit the machine learning classifiers, we consider both bigram features in a setting similar to Netzer et al. [2019] and the following writing quality features.

A.5 ROI Calculation

We adopt the method described in Netzer et al. [2019] to calculate the ROI, which incorporates interest rates into its calculation, making it possible to account for the associated risk. Specifically, we first compute the expected profit of each loan as

$$E(profit)_j = [1 - P_j] \times IE_j + [P_j] \times [-0.75 \times AG_j + 0.25 \times IE_j],$$

where $E(profit)_j$ is the expected profit of loan j, P_j is loan j's probability of default based on the corresponding predictive model, IE_j is the interest rate on the loan's term, AG_j is loan j's principal amount, which we assume $\$100$ for each loan.

Following Netzer et al. [2019], we also assume that when a loan becomes default, an average of 25% of the principal is repaid, based on data published by crowdfunding consulting agencies. We then rank the loans based on their expected profits and select the top 1,000 loans with the highest expected profits to create an investment portfolio. Assuming that $100 is invested in these 1,000 loans, we calculate the actual profit (or loss) of each loan by using its actual default status.

Assuming a total investment of $100,000 (1,000 loans at $100 per loan), the ROI is first computed as the ratio between the net profit (actual total profit minus the total investment) and the total investment, and then converted to an annual ROI based on the loan's term.

Ground Truth Label Generation. Note that we do not know the actual default status if the loan was historically not funded. Therefore, if a loan was not actually funded in the historical data, we inferred its repayment status using a gold-standard deep Heckman model, and the historical repayment status if the loan was selected for funding. The gold-standard deep Heckman model was fit using all samples (i.e., without train-test split) because the goal was to maximize the data-model fit instead of out-of-sample generalization. The parameters were selected using cross-validation, and we adopted the optimal hyperparameters to retrain the model using all data as the final model. Since converting all 231,144 text descriptions into BERT embeddings is extremely memory and computing intensive, we randomly down-sampled the full sample at a 20% ratio to make the analysis feasible. Such down sampling completely at random ensures that the subsample we used for training is representative.