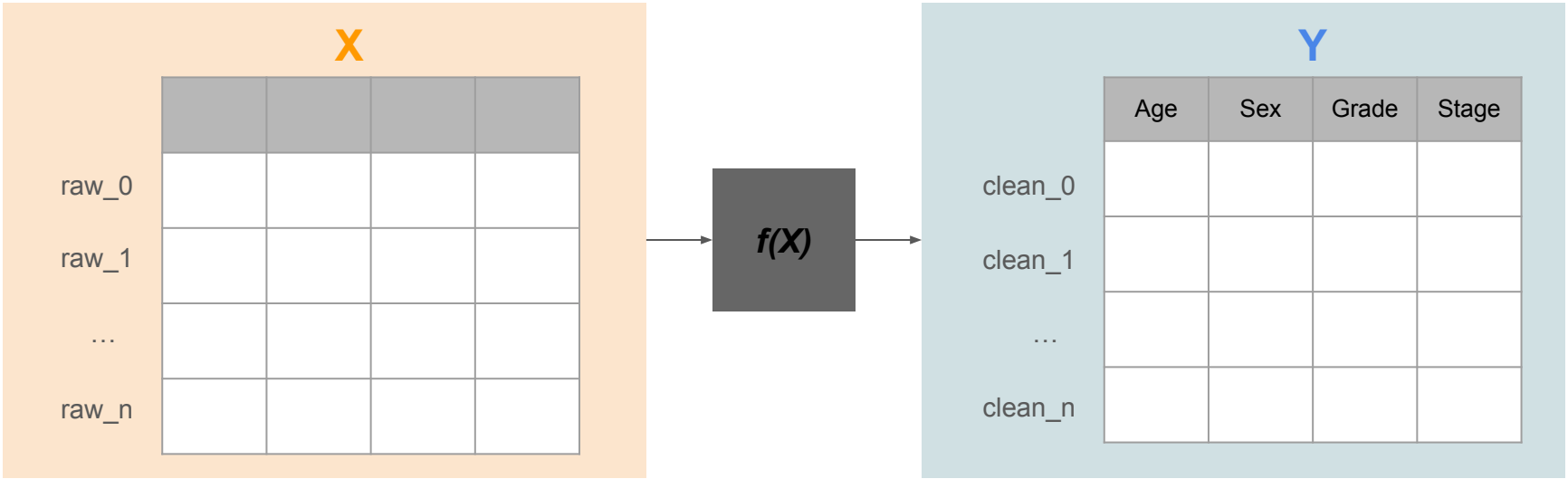
EPIGENE LABS

Enhancing Data Curation with Labeled Data

By: Laura Fuentes

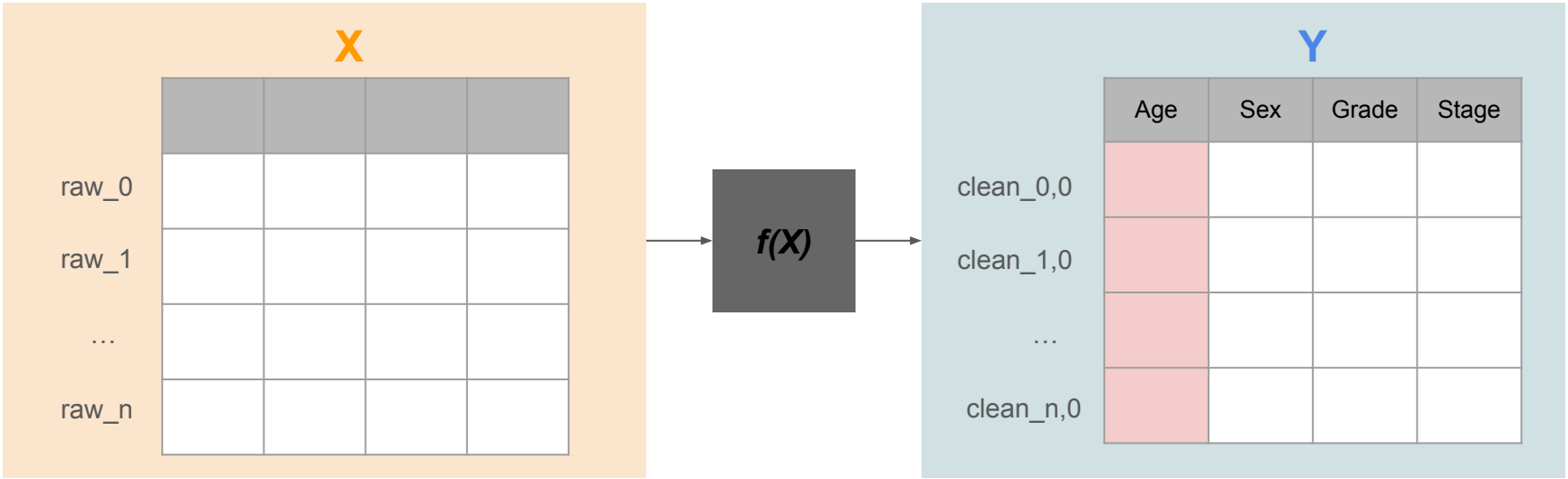
Overview

With the aim of enhancing the data curation process, we propose implementing a Language Model (LM) that retrieves one of the desired features (`Age`, `Sex`, `Grade`, `Stage`, `Cancer Type`) based on the raw version of the input row. In short, our LM, represents a mapping *f,* that maps X, the raw data, to Y, the labeled, clean dataset.



Overview

More specifically, we propose performing a classification task per feature desired. Which results in 5 classification pipelines, where the classifier will take as input n (raw_i) rows, with label (clean_i,j), being i the row index, and j the feature index.

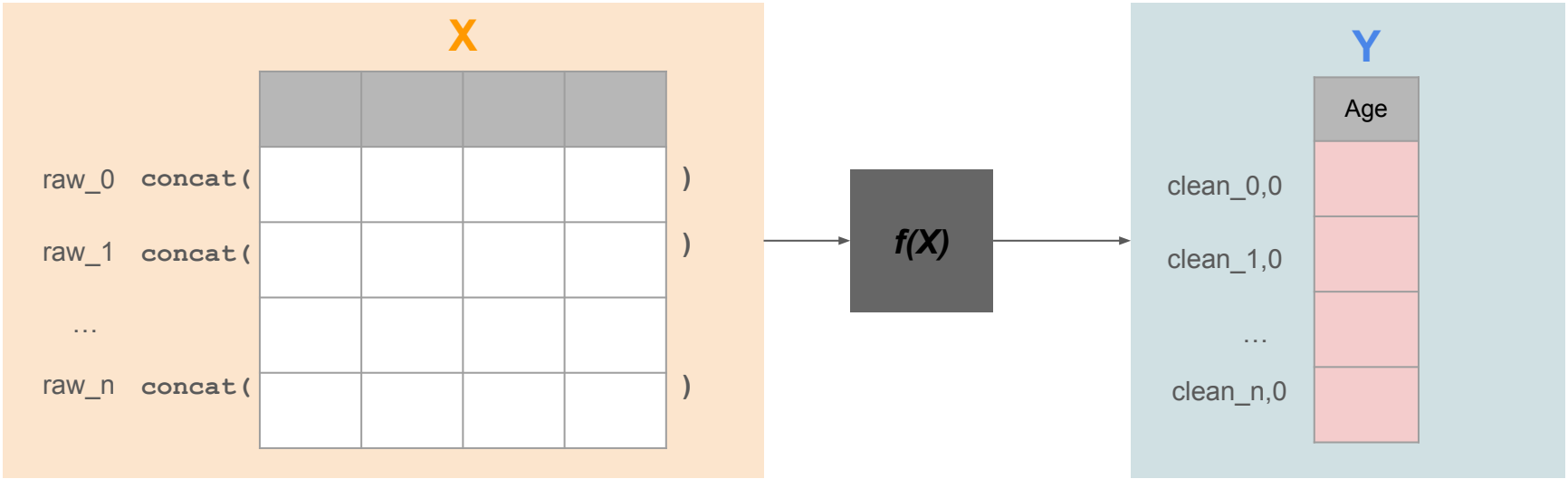


Classification Pipelines

We propose fine-tuning bert-based state-of-art models for medical data such as **BioBERT**, **PubMedBERT**, and **ClinicalBERT**, to determine which performs best on the specific variables (Age, Sex, Grade, Stage, Cancer Type). In order to perform model selection for each feature, the training data will be splitted into train-validation-test, and the best model for each feature would be selected based on it’s validation performance.

Data Preprocessing

In order to input the data to the model each x_i sample will be the concatenation of all the columns in the raw dataset.



Data Preprocessing

For the concatenation process, all columns will be joined with a special separation token [SEP], which makes part of BERT-based models vocabulary. The previous with the objective of making the model understand, that the input sequence is the union of multiple substrings.

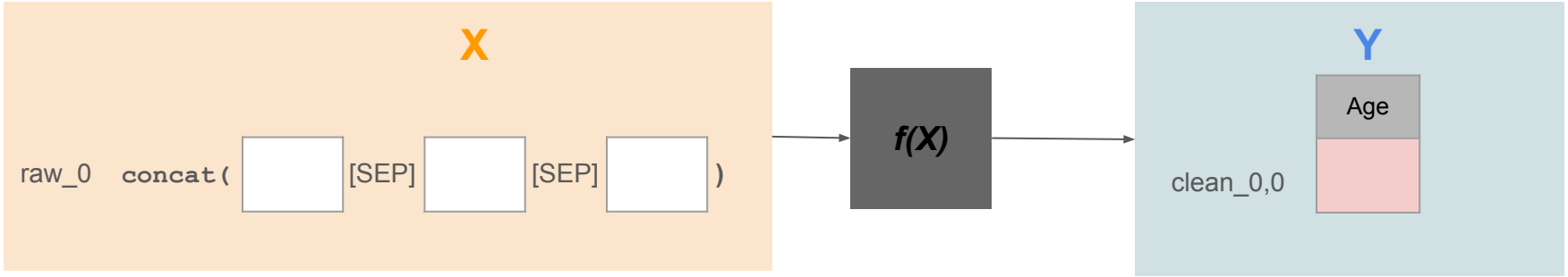


Illustration of the concatenation process, for one sample in the dataset

Data Preprocessing: Age Feature Consideration

Since pre-trained LMs like BERT and its derivatives, are not inherently optimized for regression tasks, to perform predictions on the variable **age** we propose grouping ages into 10-year intervals commonly used in cancer epidemiology.

As age is a continuous variable, this transformation converts it into categorical bins, making it suitable for classification tasks rather than regression. These intervals (`<20`, `20–29`, `30–39`, ..., `80+`) allow the model to focus on patterns and relationships relevant to each age group.

Data Preprocessing: N.A. target class

If the target labels include N.A. values, the output layer of the classifier will be extended by one neuron, with the additional neuron representing the null category. This modification allows the model to predict N.A. when it is uncertain about the correct class, a situation that can frequently occur in real-world scenarios.

Data Preprocessing: N.A. target class

To prevent the model from consistently predicting N.A, we propose to keep at maximum 10% of N.A samples, so that the other remaining classes are more represented in the training data.

An additional measure, would be to assign a lower weight in the loss function of the model during the fine-tuning process, so that the prediction accuracy for the non-null categories has more value in the training process.

```
[ ]  1 from torch.nn import CrossEntropyLoss
     2
     3 class_weights = torch.tensor([weight_na, weight_class_1, weight_class_2])  # Adjust weights
     4 loss_fn = CrossEntropyLoss(weight=class_weights)
     5
```

Performance Metrics

For predicting variables such as **Age** and **Sex**, accuracy would be the primary evaluation metric as it provides a straightforward measure of overall correctness. However, if the training data is imbalanced, we would shift to using **balanced accuracy**, which accounts for differences in class distribution by weighting each class equally. For sensitive variables such as **Cancer Type**, **Stage**, and **Grade**, where false positives can have critical implications, we would prioritize **precision** to minimize over-prediction errors, particularly in identifying high-risk categories.

This ensures the model performs reliably while addressing the unique importance of each variable.

These evaluation metrics, would be used during the model selection phase, as well as for reporting the final performance metrics of the fine-tuned pipelines.

Computational Considerations

When addressing computational considerations for this task, it's important to evaluate the feasibility of fine-tuning pre-trained models.

If the proposed solution is too computationally heavy due to the dataset size or resource constraints, one alternative is to **use the pre-trained models solely** for inference **without fine-tuning**. This approach leverages the general knowledge embedded in the models while avoiding the cost of retraining. To further reduce computational costs, techniques such as **freezing the lower layers** of the model during fine-tuning can be applied, allowing only the top layers to adapt to the task-specific data.

More details?

Contact: lauracarolinafuentesquintero@gmail.com