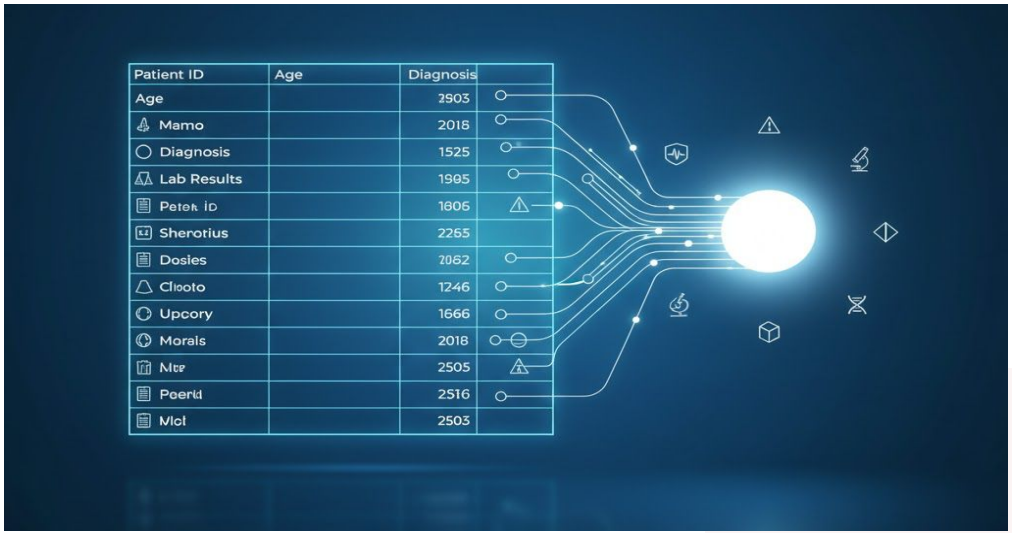
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG

INF FAKULTÄT FÜR INFORMATIK

Embeddings for Prediction in Clinical Data



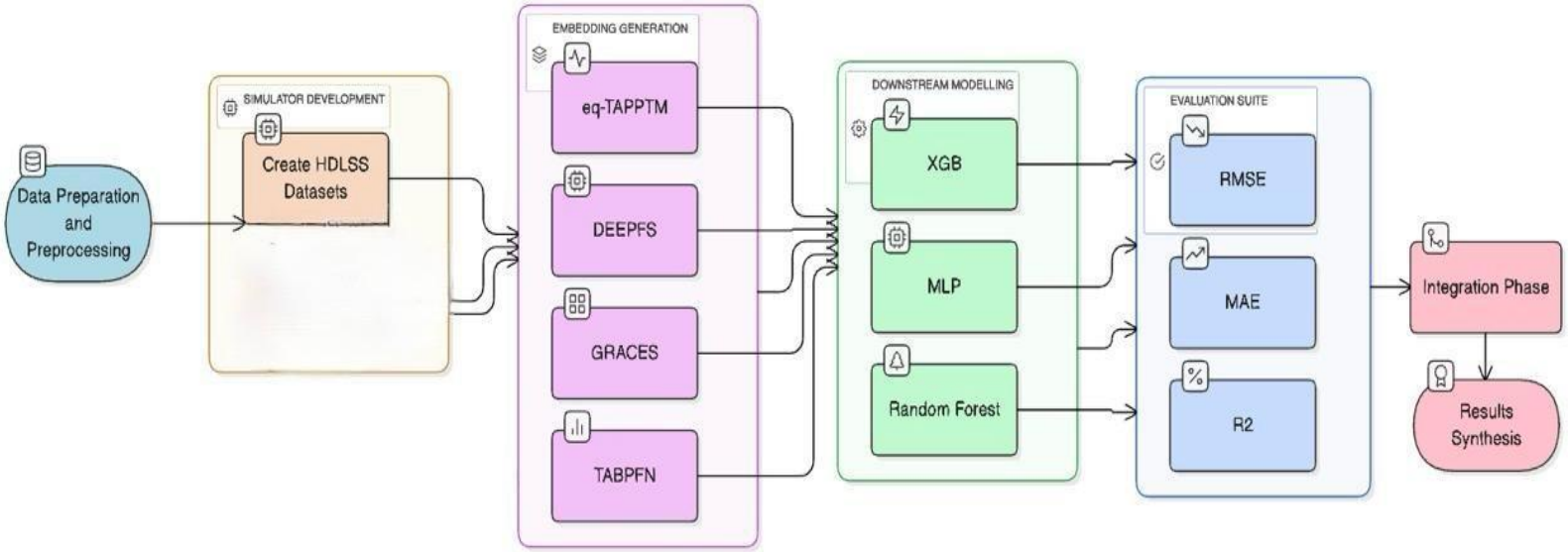
Supervisor:

Prof. Dr. Myra Spiliopoulou

Team Members:

- **Sourav Salkoppalu Lingaraju**
- **Chethan Chinnabhandara Nagraj**
- **Sajith Kumar Santhosh**
- **Bhavesh Sharad Yeole**

Proposed Project Phases



Proposed Sample DataSet

Dataset: Drug Induced Autoimmunity Prediction from UCI ML Repository

- **Dataset Overview:** 477 instances × 195 features; features are continuous and binary, target = binary (autoimmune risk).
- **Instance Reduction:** Subsample to ~150–200 instances to create a wide dataset (more features than samples) suitable for high-dimensional experiments.
- **Handling Continuous Features:** Continuous descriptors will be binarized to introduce sparsity and align with modeling requirements.
- **Binary Features:** Already sparse, with most values being zero, supporting skewed, high-dimensional analysis.
- **Tabular & Interpretable:** Features represent molecular properties, enabling meaningful feature selection and predictive modeling.
- **Presented by:** Sajith Kumar Santhosh

Embedding Generation: Proposed Methods

Method 1: Tabular Pre-Training via Meta-representation

- **Core Challenge:** The inability to learn shareable knowledge across tabular datasets due to inherent heterogeneity in their attributes (dimensionality and semantic meaning) and label spaces.

- **Working:** Converts raw tabular rows into a “meta-representation” (encodes the original feature with neighborhood distance and its corresponding label). Pretrains a simple MLP with our available datasets. Uses the MLP’s hidden layers as universal embeddings (serves as input for downstream task)

- **Presented By :** Sourav Salkoppalu Lingaraju

H.-J. Ye, Q. Zhou, H.-H. Yin, D.-C. Zhan, and W.-L. Chao, “Rethinking Pre-Training in Tabular Data: A Neighborhood Embedding Perspective,” arXiv preprint arXiv:2307.01604, 2023.

Method & Key Insights

- Each instance (row) is represented using:
  → Its **nearest neighbors** in feature space.
  → The **labels of those neighbors**.

- This yields a meta-representation that's independent of specific features or schemas.

.

Raw feature space:
(x1, x2, x3) → y

Meta-representation space:
[dist_to_n1, dist_to_n2, ..., y_n1, y_n2, ...] → y

Are TabPTM-style embeddings the right choice for our data?

- ➔ **Converts high-dimensional, low-instance features into compact, meaningful embeddings:** TabPTM transforms wide, complex tables into dense representations that downstream models (trees/MLP) can use more effectively.
- ➔ **Completely independent of cryptic feature names:** It relies on neighborhood structure, not column names, making it ideal for anonymized or coded clinical variables.
- ➔ **Embeddings act as transferable knowledge for downstream tasks:** The pretrained MLP provides rich embeddings that can be directly fed into tree-based models or neural networks to improve prediction quality.
- ➔ **Meta-Representation Construction:** TABPTM redefines the concept of "vocabulary" by encoding each instance as a meta-representation based on its relationships with neighboring samples rather than on feature semantics.

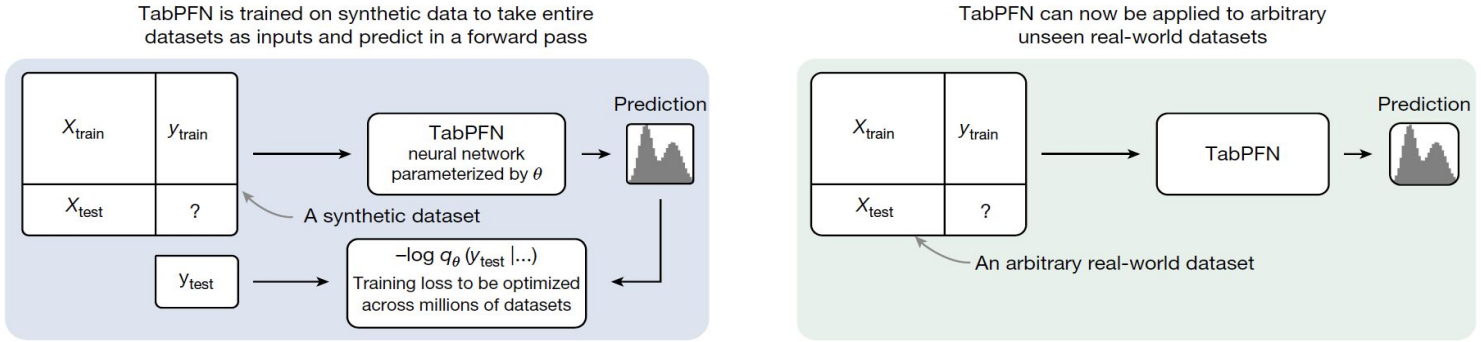
Method 2 - TabPFN (Tabular Prior-data Fitted Network)

Core Idea :

It is a transformer based model for tabular tasks.

Pre-trained on a large corpus of tabular datasets.

During the downstream task, this learning is applied to our specific dataset at inference time , by taking the entire dataset as context and performing training and prediction in a single forward pass.



Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with

Embedding Process

Categorical & Numerical data -> Transformation (float values) -> High Dim Embedding

Advantages

- High Accuracy for prediction tasks with small, high dimensional data
- Reduces overfitting , and supports mixed data types.
- Reuses feature representations
- The Transformer uses two-way attention to learn the data..
- No need for training from scratch and excessive fine tuning

Method 3: Graph Convolutional Network Feature Selector (GRACES)

- **Core Working Logic:** Iteratively selects features that maximize loss reduction while refining sample embeddings using GraphSAGE to capture latent associations.
- **Dynamic Graph Construction:** Builds a cosine similarity graph from currently selected features each iteration to guide the GCN.
- **GCN-Refined Embeddings:** Uses Graph Convolutional Networks (GCNs) to refine sample embeddings, explicitly capturing latent associations between samples.
- **Overfitting Reduction:** Uses multiple dropouts to average gradients, adds Gaussian noise to GCN weights, and applies F-correction (ANOVA F-test) to stabilize feature scores.
- **Presented By:** Sajith Kumar Santhosh

C. Chen, S. T. Weiss, and Y.-Y. Liu, “Graph convolutional network-based feature selection for high-dimensional and low-sample size data,” Bioinformatics, vol. 39, no. 4, btad135, April 2023.

Advantages and Relevance

- **Designed for HDLSS**: Significantly outperforms DNP and HSIC Lasso on real-world biological datasets (e.g., Colon, Leukemia).
- **F-Correction:** Combines gradient-based utility with ANOVA F-test for robust feature selection.
- **Feature Name Independence:** Selection relies on gradient norms + F-test; semantic names not required.
- **Attention to Rare Features:** Rare but discriminative features are prioritized if they produce a high gradient norm.
- **End-to-End Training:** Feature selection and classification are learned simultaneously.
- **Stability Across Subsets:** The combination of dropouts, Gaussian noise, and F-correction ensures that selected features remain consistent even when the training data changes slightly, which is critical for HDLSS datasets.

Method 4: Deep Feature Screening (DeepFS)

- **Problem Addressed:** Traditional feature selection methods struggle with ultra-high-dimensional, low-sample-size data due to overfitting, computational instability, and strong model assumptions (e.g., linearity) that ignore nonlinear structure and feature dependencies.
- **Core Working Logic:** Embedding + Statistical Screening
- **Feature Extraction (Embedding):** Uses Autoencoder (AE) or supervised AE. Transforms high-dimensional input into a simplified embedding while preserving essential structure. This helps highlight the most informative aspects of each feature for selection.
- **Feature Screening:** Computes feature importance ω_i using multivariate rank distance correlation. Evaluates how strongly each original feature relates to the learned low-dimensional embedding. Selects features with the highest correlation scores.
- **Presented By:** Sajith Kumar Santhosh

K. Li, F. Wang, L. Yang, and R. Liu, "Deep feature screening: Feature selection for ultra high-dimensional data via deep neural networks," Neurocomputing, vol. 538, pp. 126186, 2023

Advantages and Relevance

- **Distribution-free and model-free:** This enables it to capture nonlinear feature interactions
- **Outperforms baselines:** Outperforms DNP, FsNet, and CAE in classification accuracy and reconstruction error on real HDLSS datasets.
- **Feature Name Independence:** Selection based purely on numerical correlation with embeddings; feature names are irrelevant.
- **Attention to Rare Features:** Rare but informative features are prioritized if strongly correlated with the embedding.
- **Embedding Compatibility:** Step 1 uses an Autoencoder to produce a low-dimensional embedding capturing complex patterns.
- **Overfitting Mitigation:** Combines DNN expressivity with nonparametric rank distance correlation for robust feature selection.

Simulator Development Plan

Subset Strategy

- **Normal subset:** Maintain original class proportions (natural sparsity)
- **Skewed subsets:** Change the dataset so that some feature values are more common than others.
- **Purpose:** Test how your model performs when some features are unevenly distributed.

HDLSS Subsets

- **Target size:** 150–200 rows (D > N, “wide” dataset)

Simulator Development Plan

Row Deletion Approaches:

1. **Stratified deletion:** Reduce rows proportionally from each class → preserves original class ratios and dataset sparsity

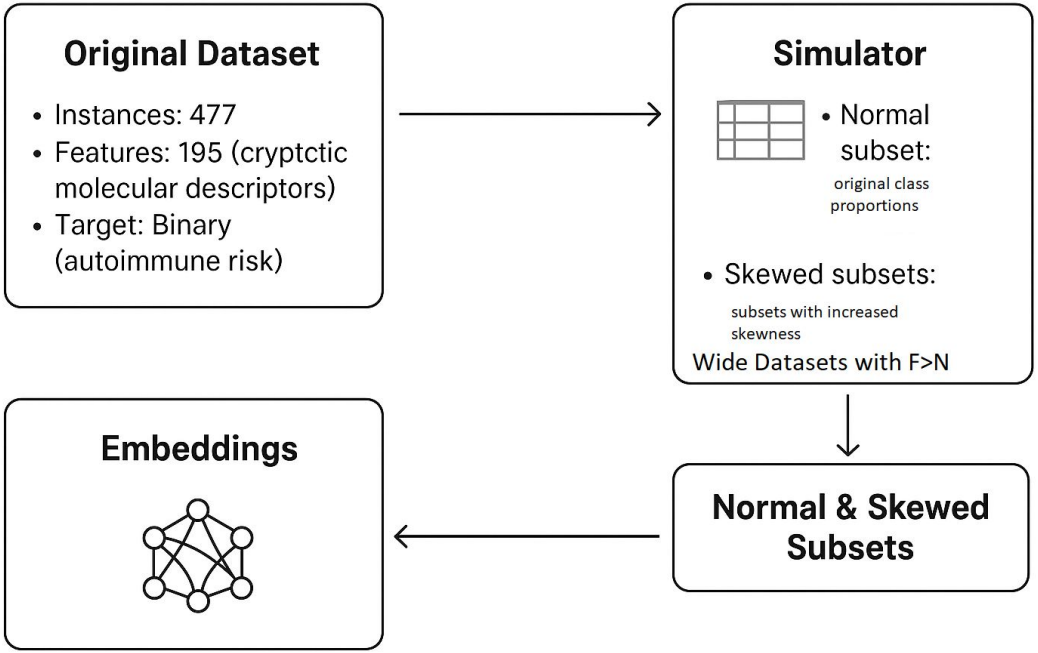
2. **Skewed deletion:** Preferentially removes rows from specific classes

    - Mild Skew: ~1:4 (minority:majority) → moderate stress
    - High Skew: ~1:6–1:7 → extreme stress

Output:

1. One normal + two skewed subsets
2. Only rows removed; features remain intact
3. Ready for embedding models

Simulator Development Plan



Downstream Regression Modelling Plan

- **Choosing a model:** After we create the embeddings from the upstream model, we pick a simple model to predict a value (regression). Options: Xtreme Gradient Boosting (XGB), Random Forest, Multi-Layer Perceptron, Support Vector Machine.

- **Setting Model Parameters:**
  - For tree models: number of trees, maximum depth of trees, learning rate, regularization strength
  - For the Multi-Layer Perceptron: number of layers, number of neurons, activation function, dropout rate, learning rate

- **Training on Embedding task:** During training, the upstream embeddings are fixed, the data is split into training and validation sets, the chosen model is trained on these embeddings, and the training is stopped early when the model stops improving**.**

- **Prediction Task:** The trained model takes an embedding and outputs a continuous value (the regression target).

Evaluation Suite Plan

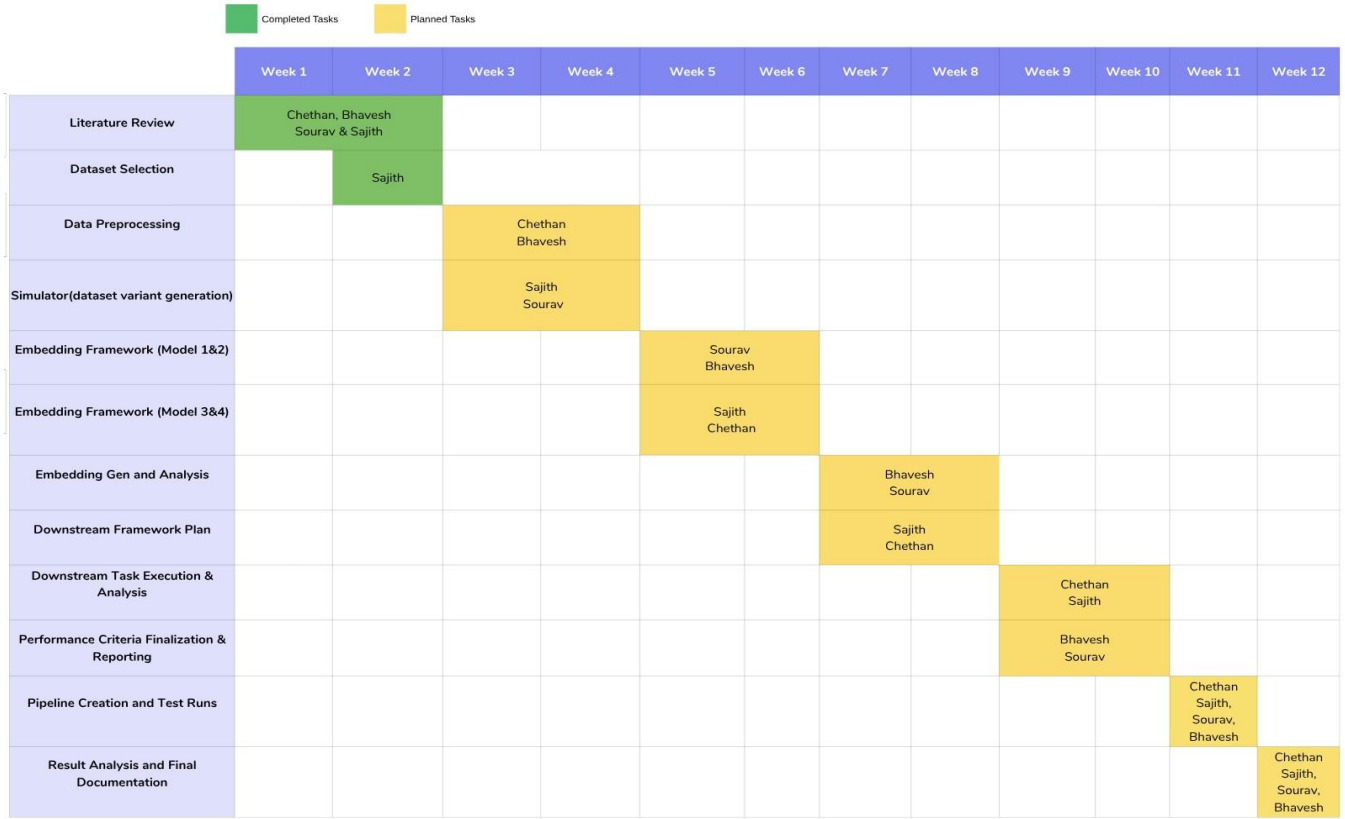
Evaluation metrics computation is done using:

- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- Coefficient of Determination (R^2)

These metrics help measure how accurate the predictions are, even when the data distribution changes (distribution-variant data), because they capture both error size and consistency across different regions of the data.

M. M. Krell and B. Wehbe, "A First Step Towards Distribution Invariant Regression Metrics," *arXiv e-print 2009.05176*, Sep. 2020. [Online]. Available: https://arxiv.org/pdf/2009.05176.

Project Timeline



Thank you!

Questions?