

National Yangming Chiaotung University

Department of computer science

Computer Science and Engineering project ||

Report

Predictive Model for Second Primary Cancer Using SEER
(*Surveillance, Epidemiology, and End Results*) Dataset.

Student : 卜銳凱 , 109550198

Supervisor : 曾意儒教授

Date : 2024/06/06

Abstract:

This work is devoted to the development and performance evaluation of a neural network model to predict the development of the second primary cancer using the SEER (Surveillance, Epidemiology, and End Results) dataset. During this process, missing values in the data were taken care of, and the categorical variables were encoded and numeric features standardized. The model consisted of several fully connected layers with the ReLU activation function. We trained and evaluated the model through cross-validation across 30 simulations to ensure its robustness and generalization. Primarily, the Receiver Operating Characteristic (ROC) curve and Area Under the ROC Curve (AUROC) were used to present the performance. The ROC curve presents a model's discriminative ability as a function of threshold values, while the AUROC value shows moderate predictive performance. This, in effect, shows how the developed model can be useful in separating patients with second primary cancer from those without, thus recommending areas where the model performance can be improved through better data preprocessing and improved model tuning.

Introduction

Predicting second primary cancer is essential in improving patient outcomes through early detection and intervention. In this work, we set to develop a neural network model for predicting second primary cancer from the SEER (Surveillance, Epidemiology, and End Results) dataset. In this study, the data were handled for missing values, encoded categorical features, and standard scaling of numerical features. Model performance was evaluated using the ROC curve and calculated AUROC. The ROC curve evaluates the model's ability across different thresholds to separate patients with and without second primary cancer, and the AUROC provides a single summary statistic of overall performance.

Cancer registry:

A cancer registry systematically collects data about cancer, focusing on monitoring incidence, treatment, and outcomes, which is crucial for public health, research, and policymaking. It includes demographic details, such as age, sex, race, and other cancer-specific information, such as type, tumor size, stage at diagnosis, and treatment modalities. The current study uses a baseline SEER dataset – a comprehensive cancer registry – and develops a model for predicting second primary cancers.

The dataset includes encoded features such as age at diagnosis, laterality, gender, presence of separate tumor nodules, pleural invasion, tumor size, positive regional lymph nodes, combined

stage, radiology treatment, systemic therapy, and surgical treatment. This rich dataset allows a detailed analysis and prediction of cancer outcomes.

Dataset and preprocessing:

We used the SEER dataset, which serves as a comprehensive cancer registry. The dataset provides elaborate information about the cancer patients, such as demographic details, tumor characteristics, and treatment modalities. The following are the preprocessing steps taken to have the data ready for model training:

1. Load and Preprocess Data

Data Loading:

- The dataset is loaded from a CSV file using `pd.read_csv`.

Imputation:

- Missing values in the 'Age' column are imputed using the median value with `SimpleImputer(strategy='median')`.

Feature Selection:

- Categorical features: ['SYMOGN', 'Gender', 'SSF1', 'SSF2', 'LYMND', 'AJCCstage', 'RT', 'ST', 'RX Summ--Surg Prim Site (1998+)']
- Numeric features: ['TMRSZ', 'Age', 'Income', 'Residence', 'Year of diagnosis']
- The target variable is `data['Target']`.
- The features are selected by dropping unnecessary columns.

2. Preprocessing and Encoding

Imputation for Numeric Features:

- Missing values in numeric features are imputed using the median strategy.

Splitting Data into Training and Testing Sets:

- The data is split into training and testing sets using `train_test_split` with an 80-20 split.
- The `stratify` parameter ensures that the target variable distribution is maintained across splits.
- `random_state=42` ensures reproducibility.

3. Further Splitting for Validation

Splitting Training Data into Training and Validation Sets:

- The training set from the previous step is further split into training and validation sets using a 75-25 split.

- This results in 60% of the original data being used for training, 20% for validation, and 20% for testing.

4. Target Encoding

Target Encoding for Categorical Features:

- Categorical features are target encoded using `ce.TargetEncoder`.
- This encoding is applied to the training, validation, and test sets.

5. Standardization

Standardizing Numeric Features:

- Numeric features are standardized using `StandardScaler`.
- The scaler is fit on the training data and applied to the validation and test data.

6. Preprocessing Application

Applying the Preprocessor:

- The combined preprocessor (including scaling and encoding) is applied to the training, validation, and test sets.

7. Handling Class Imbalance

SMOTE (Synthetic Minority Over-sampling Technique):

- SMOTE is applied to the training set to balance the classes by generating synthetic samples for the minority class.

8. Conversion to PyTorch Tensors

Converting Data to Tensors:

- The preprocessed data is converted to PyTorch tensors for use in the model.
- `X_train_res`, `y_train_res`, `X_val`, `y_val`, `X_test`, and `y_test` are converted using `torch.tensor`.

9. Data Loaders

Creating Data Loaders:

- `DataLoader` objects are created for the training, validation, and test sets.
- These loaders handle batching and shuffling (for training) to efficiently feed data into the model during training and evaluation.

Summary of Data Splits:

Training Set (60%):

Used to train the model.

After applying SMOTE to balance classes, the size might increase.

`DataLoader` created with batching and shuffling enabled.

Validation Set (20%):

Used to tune hyperparameters and evaluate model performance during training.

`DataLoader` created without shuffling.

Test Set (20%):

Used to evaluate the final model performance on unseen data.

Dataloader created without shuffling.

This structured approach ensures that the model is trained on a balanced and appropriately preprocessed dataset, while validation and testing sets provide reliable metrics to evaluate model performance.

Model Architecture:

This architecture combines three convolutional layers for feature extraction from the input tabular data, followed by three fully connected layers with dropout for regularization. The model ends with a sigmoid activation function to output a probability for binary classification tasks. This design leverages the power of CNNs to capture spatial dependencies in the data while maintaining a compact and effective structure suitable for tabular data.

Detailed Architecture:

Input Layer:

Input Shape: (batch_size, input_dim)

The input data is reshaped to add a channel dimension: (batch_size, 1, input_dim).

Conv1d Layer 1:

Convolution operation with 16 filters, kernel size 3, padding 1.

Activation function: ReLU.

Output Shape: (batch_size, 16, input_dim).

Conv1d Layer 2:

Convolution operation with 32 filters, kernel size 3, padding 1.

Activation function: ReLU.

Output Shape: (batch_size, 32, input_dim).

Conv1d Layer 3:

Convolution operation with 64 filters, kernel size 3, padding 1.

Activation function: ReLU.

Output Shape: (batch_size, 64, input_dim).

Flattening:

The output from the convolutional layers is flattened to a 2D shape: (batch_size, 64 * input_dim).

Fully Connected Layer 1:

Linear transformation to 128 units.

Activation function: ReLU.

Dropout with a rate of 50%.

Output Shape: (batch_size, 128).

Fully Connected Layer 2:

Linear transformation to 64 units.

Activation function: ReLU.

Dropout with a rate of 50%.

Output Shape: (batch_size, 64).

Fully Connected Layer 3:

Linear transformation to 32 units.

Activation function: ReLU.

Dropout with a rate of 50%.

Output Shape: (batch_size, 32).

Output Layer:

Linear transformation to 1 unit.

Activation function: Sigmoid (to output a probability for binary classification).

Output Shape: (batch_size, 1).

Model Training and Simulation Description:

This model training process included the application of a neural network to predict second primary cancer. The dataset was preprocessed, and one-hot and target encoding of categorical features was done. The preprocessed data was then split into training and test sets. Model training and evaluation were done using cross-validation with several simulations to guarantee the robustness and generalizability of the model.

Model Training Process

Initialization

- Initialization of weights: Weights of model linear layers were implemented with Xavier uniform initialization.
- Class Weights Calculation: Since the proportion of cases with and without second primary cancer was imbalanced, the class weight calculated as a weight against the class was used as an argument.

Model Architecture

- The model architecture is a 1D Convolutional Neural Network (CNN) designed for tabular data.

Training Configuration

- Loss Function: BCEWithLogitsLoss.
- Optimizer: We used the RMSprop optimizer with a learning rate of $2e-5$.
- Learning Rate Scheduler: The ReduceLROnPlateau scheduler was used to decrease the learning rate by a factor of 0.5 in case the validation loss does not increase after the next two epochs.

Training Loop:

- Epochs: The model was trained for 20 epochs in each simulation.
- Gradient Clipping: To prevent exploding gradients, gradient clipping was applied with a maximum norm of 1.0.

Evaluation Metrics:

- AUROC: The area under an ROC curve, which is useful when classifying binary classification problems.

Simulation process:

- Number of Simulations: 30 simulations
- Data Splitting: Per each simulation, a stratified data split was performed into train and test sets, to maintain the distribution of the target variable in the newly created sets.
- Train and Evaluate: per each simulation, the model was trained, evaluated using the same process described above.
- Metrics calculation: after each simulation, the key metric AUROC was calculated to evaluate the model's performance.

Such an approach ensures that the model is not overfitting to a specific data split and gives a comprehensive evaluation of the model performance on different data subsets.


Experiment and results:

Experiment:

In this experiment, we developed a neural network model to predict the likelihood of second primary cancer using the SEER dataset. The dataset was preprocessed, including handling missing values and encoding categorical variables. The model was trained and evaluated using 30 simulations.

Results

The results of the experiment are summarized in the following table, showing the AUROC values for the 20th epoch of each simulation:

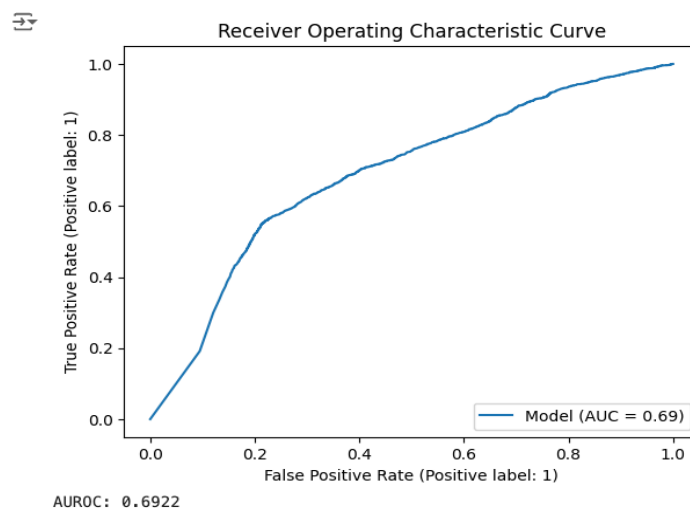
Simulation	AUROC at 20th Epoch
1	0.7048
2	0.6997
3	0.6819
4	0.6950
5	0.6900
6	0.7002
7	0.7101
8	0.7076
9	0.6967
10	0.7054
11	0.6999
12	0.6941
13	0.6987
14	0.6932
15	0.7065
16	0.6992
17	0.6879
18	0.6973
19	0.6910
20	0.7051 
21	0.6994
22	0.6956
23	0.7021
24	0.6968
25	0.6985
26	0.6914
27	0.7033
28	0.7024
29	0.6947
30	0.6998

Why Auroc ?

AUROC is an effective and reliable metric for evaluating models predicting second primary cancer due to its ability to handle class imbalance, its independence from specific thresholds, and its comprehensive evaluation of the model's discriminative power. By using AUROC, researchers and clinicians can ensure that their predictive models are both robust and interpretable, facilitating better decision-making in clinical practice.

An AUROC of 0.7 ~ indicates that the model has a moderate level of discriminative ability for predicting second primary cancer. It shows that the model is better than random guessing and can be a valuable tool in a clinical setting, although there is room for improvement.

ROC curve:



The ROC curve illustrates the diagnostic ability of the model used to predict second primary cancer by plotting the True Positive Rate (sensitivity) against the False Positive Rate at various thresholds. The curve lies above the diagonal line representing random guessing, indicating the model's performance is better than random. The model demonstrates moderate discriminative ability, suggesting it is reasonably effective at distinguishing between patients with and without second primary cancer. This measure is crucial in medical diagnostics to balance sensitivity and specificity and to select an optimal threshold for clinical use.

Why Roc curve?

The ROC curve is used to evaluate the performance of a binary classification model because it provides an overall summary of the model's ability to discriminate across all possible thresholds, including class imbalance. It effectively handles the imbalance in classes and summarizes the discriminative ability into one single measure: the AUROC. It also allows simple comparison

between models and serves in the selection of optimal thresholds, giving a balance between sensitivity and specificity, crucial in medical diagnostics for taking good clinical decisions.

Conclusion

The development and validation of the neural network model for predicting the development of a second primary cancer showed good results, with a fair level of discriminative ability. The ROC curve and AUROC allowed robust and understandable metrics for assessing model performance. Although the model showed utility, there is certainly room for further improvement through refined data preprocessing, better feature engineering, and model tuning. Overall, taken collectively, these steps form a firm foundation toward the improvement of predictive models in medical diagnostics, with the goal of supporting clinical decisions and improving patient outcomes.

References:

Privacy-Preserving Machine Learning for
Predicting Second Primary Cancer
in the Context of Data Heterogeneity

研 究 生：洪睿甫

指導教授：許智誠 博士

曾意儒 博士