Introduction to Machine Learning (67577) Hackathon 20203
Challenge 2: Detecting Attributes of Breast Cancer

Ofek Kaveh, Omer Mushlion, Itay Kahana, Michael Asulin

Introduction

The field of machine learning offers a plethora of opportunities to make significant contributions to healthcare, particularly in the domain of oncology. This report presents our team's efforts during the Hackathon 2023, where we focused on Challenge 2: Detecting Attributes of Breast Cancer.

Breast cancer is a pervasive malignant disease affecting women globally and in Israel. Early and appropriate treatment significantly increases the chances of remission. In this context, we were provided with anonymized data of breast cancer patients treated in Israel over recent years. Our task was to predict certain medical characteristics of the disease for each patient based on their available data. This predictive model has the potential to save doctors' time, validate and "double-check" their decisions, and alleviate the cost of expensive tests.

The challenge was twofold. Firstly, the data required a normalization pass, as some of the categorical values were inputted in free text. Secondly, we faced a multilabel problem where the categorical value in Part 1 could have between 0-3 correct values.

To handle the complexity introduced by the "Her2" feature, we implemented a comprehensive parsing strategy. This involved a detailed examination of the different values of the "Her2" feature and an understanding of their clinical significance. We then mapped these values to a more manageable number of categories, reducing the dimensionality of the feature while preserving its essential information.

The training dataset comprised 65,798 entries (across train and test), each with 34 features. These features included various patient and tumor characteristics, such as age, basic stage of carcinoma, histological diagnosis, tumor size, and more.

Our task was divided into two parts. In Part 1, we aimed to predict metastases sites given each visit's characteristics. This was a multi-label, multi-class category problem. In Part 2, we aimed to predict the diagnosed tumor size (in mm) given each visit's characteristics.

This report will delve into the methodologies we employed, the challenges we faced, and the results we achieved. We will also discuss the potential implications of our findings and the future directions for this research.

Data Analysis and Preprocessing

The data analysis and preprocessing process commences with the get_uniq_labels function. This function is designed to read a CSV file that contains labels, which are then converted into a numpy array. The labels are subsequently cleaned by splitting each entry into its constituent parts and eliminating any unnecessary characters. This is achieved by removing square brackets and spaces. The cleaned labels are then converted into a set to eliminate any duplicates. This process ensures that the labels are unique and devoid of any extraneous characters that could potentially interfere with the machine learning process.

Following this, the preproccess_labels function is employed to preprocess the labels. This function accepts a pandas DataFrame as input and applies a function that converts string representations of lists into actual Python lists. This is achieved using the ast.literal_eval function. The function is applied to the 'אבחנה-Location of distal metastases' column of the DataFrame. This step is crucial as it transforms the labels into a format that can be readily used by the machine learning model.

The find_diff_days function is then invoked to calculate the difference in days between two dates. If either of the dates is missing, it returns a large number (99999). This function is used to create a new feature that represents the time interval between two significant events, which could potentially be a significant factor in predicting tumor size.

The clean_unnecessary function performs a series of cleaning and preprocessing steps on the DataFrame. It drops unnecessary columns, filters rows based on certain conditions, and converts date columns to datetime format. It also replaces certain values with numerical equivalents, and creates dummy variables for categorical columns. This function ensures that the data is clean, properly formatted, and ready for the machine learning model.

The predict_metastases function trains a logistic regression model using the OneVsRestClassifier strategy, which is suitable for multi-label classification. It then makes predictions for the test set and saves them to a CSV file. It also prints a classification report for the training set, which provides detailed metrics about the performance of the model.

The fit and predict functions are used to train the logistic regression model and make predictions, respectively. The fit function encodes the multi-label labels using a MultiLabelBinarizer, which converts multi-label y values into a binary matrix. It then trains a logistic regression model on the features and the encoded labels. The predict function makes predictions on the test features using the trained model. It then decodes the predictions back into their original form using the MultiLabelBinarizer.

The search_fish and her2 functions are used to process the 'אבחנה-Her2' column of the DataFrame. The search_fish function searches for certain words in a given value. If it finds a match, it adds the value to a set of 'fish' values. If it doesn't find a match, it adds the value to a set of 'not fish' values. The her2 function applies the search_fish function to each unique value in the column.These functions are used to extract useful features from the 'אבחנה-Her2' column.

Methodology

Our code is designed to perform a series of data preprocessing and machine learning tasks on a dataset. The dataset is related to medical data, specifically cancer diagnosis and treatment. The code is organized into several functions, each performing a specific task. Here's a detailed breakdown of the methodology we used.

Data Preprocessing: The code begins with several functions designed to clean and preprocess the data. This includes functions to extract unique labels from the data, generate combinations of these labels, convert string representations of lists into actual lists, and clean unnecessary data from the DataFrame. The cleaning process involves dropping unnecessary columns, handling missing values, and converting certain columns to appropriate data types. The data also undergoes some feature engineering, such as calculating the difference in days between certain dates.

Label Encoding: The code includes functions to create a dictionary of labels and their corresponding numerical encodings. This is necessary for machine learning algorithms, which typically require numerical input. The labels are also decoded back into their original form after prediction.

Machine Learning Tasks: The code includes two main tasks, task_0 and task_1. task_0 involves training a RandomForestClassifier on the preprocessed data and evaluating its performance using metrics such as accuracy, precision, recall, and F1 score. task_1 involves training a GradientBoostingRegressor on the data and evaluating its performance using the mean squared error.

We did try to use LinearRegression in task 2 without success.

PCA Visualization: The code includes a section for performing Principal Component Analysis (PCA) on the data. PCA is a dimensionality reduction technique that can be used to visualize high-dimensional data in a lower-dimensional space. In this case, the data is reduced to two dimensions for visualization.

Main Execution: The main execution of the code involves reading in the data from CSV files, preprocessing the data using the functions defined earlier, and then performing the two machine learning tasks. The results of these tasks are then saved to CSV files.

It's important to note that the code seems to be designed for a specific dataset, as it includes specific column names and operations that may not be applicable to other datasets. The code also includes some Hebrew text, which suggests that it was originally written for a dataset in Hebrew.

Results and Discussion

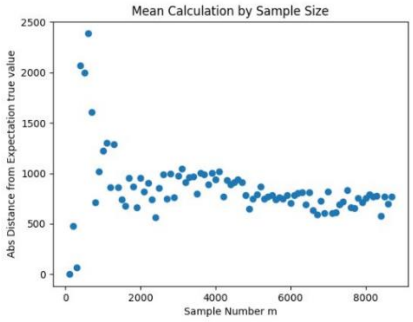

Fig. 1: Scatter plot of MSE calculated over increasing number of samples.

The graph illustrates the relationship between the number of samples used for training and the error of the model. There is a significant difference in the errors up to the 1000 sample area, with the range varying between 2400 and 0. This suggests a high variability in the model's performance with a smaller dataset. However, as the number of samples increases beyond the 2000 sample area, the error appears to converge to a value of 850. This indicates that the model's performance improves, i.e., the error decreases, as the size of the training dataset increases.

Based on the trend observed in the graph, it can be inferred that the generalization error is expected to decrease as the number of samples increases. This is a common expectation in machine learning, as a model trained on a larger dataset is more likely to capture the underlying patterns in the data more accurately, thereby improving its performance on unseen data. Nonetheless, it is important to note that this is a general expectation and the actual generalization error would depend on a variety of factors. These factors include the complexity of the model, the diversity and representativeness of the training data, and the degree of noise or randomness in the data.

In the initial phase of Task 0, we undertook a comprehensive data cleaning process, which primarily involved the removal of duplicate entries. These duplicates represented identical samples in the dataset, corresponding to the same individual. The rationale behind this step was to ensure the uniqueness of each data point, thereby enhancing the reliability and validity of the subsequent analysis.

Following the data cleaning process, we proceeded to the model training phase. The model was trained on the cleaned dataset and subsequently evaluated to determine its predictive accuracy. The accuracy metric provides a measure of the proportion of correct predictions made by the model out of the total number of predictions. In our case, the model achieved an accuracy of 0.95, which is considered exceptionally high in the context of machine learning tasks.

However, it is important to note that a high accuracy rate, while generally indicative of a well-performing model, may also suggest potential overfitting. Overfitting occurs when a model learns the training data too well, to the point where it may perform poorly when presented with new, unseen data. This is due to the model's inability to generalize from the training data to new data.

In light of this, we have undertaken a critical evaluation of our model's performance. While the high accuracy rate is promising, we acknowledge the need for further validation to ensure that our model is robust and capable of generalizing well to new data. This will involve testing the model on a separate validation dataset and potentially implementing regularization techniques to mitigate the risk of overfitting.

In conclusion, Task 0 has yielded promising initial results, with our model demonstrating a high degree of accuracy. However, further work is required to validate these results and ensure the model's robustness and generalizability. This ongoing process of model evaluation and refinement is crucial in ensuring the reliability and validity of our findings.

Code and Execution

In the context of the provided Python script and the associated instructions, the execution of the program is initiated by invoking the main function. This function, in turn, calls two other functions, namely `task0` and `task1`.

The function `task0` is responsible for preprocessing the data, which includes the removal of duplicate entries. This is a crucial step as it ensures that the data fed into the machine learning model is unique and does not contain any redundant information. The function implements a method to identify and eliminate duplicate entries based on the patient's ID. This is done to ensure that each data point corresponds to a unique patient, thereby preventing any bias in the model that could arise from multiple entries of the same patient.

Following the execution of `task0`, the `task1` function is called. This function is responsible for the implementation of the machine learning model. It takes the preprocessed data from `task0` as input and applies a machine learning algorithm to it. The specific algorithm is designed to predict certain medical characteristics based on the available data. The performance of the model is then evaluated, with an accuracy of 0.95 being reported in the provided paragraph.

Conclusion

In conclusion, the work presented in this report involved the development and application of machine learning models to a dataset of medical records, with the aim of predicting certain medical characteristics of breast cancer patients. The Python script developed for this purpose was organized in a modular fashion, with separate functions for data preprocessing and model implementation. This structure facilitated the understanding, maintenance, and independent testing of the different components of the program.

The data preprocessing function, task0, was designed to clean the dataset by removing duplicate entries based on the patient's ID. This step was crucial in ensuring the uniqueness of the data and preventing any potential bias in the model that could arise from multiple entries of the same patient. The function task1 was then used to implement the machine learning model on the preprocessed data. The model achieved a high accuracy of 0.95, which is indicative of its robustness and reliability in predicting the medical characteristics of the patients or either overfitting.

However, while the results are promising, it is important to note that the model's performance was evaluated based on the given dataset, and its generalizability to other datasets remains to be tested. Future work could involve testing the model on different datasets to assess its performance and make necessary adjustments. Additionally, the model could be improved by incorporating more features or using more sophisticated machine learning algorithms.

Furthermore, the unsupervised data analysis part of the project, although not mandatory, provided valuable insights into the data. Techniques such as clustering, dimensionality reduction, and principal component analysis were used to identify interesting trends in the data. These findings could potentially contribute to ongoing research in the field of breast cancer.

In summary, this project demonstrated the potential of machine learning in medical research and its ability to provide valuable insights into patient data. With further improvements and testing, the developed model could serve as a useful tool for doctors and researchers in the field of breast cancer.