Machine Learning - Assignment 1

Dang Thai Hoang - s3927234

April 14, 2024

Abstract

This study aims to employ machine learning methodologies for the prediction of diabetes onset in individuals utilizing different health metrics and demographic information. By analyzing factors like BMI, age, gender, and lifestyle habits, the study seeks to develop accurate predictive models. Insights gained could enhance diabetes prediction and preventing strategies.

This report will cover a series of steps including:

- **Problem Formulation**: Defining the task at hand and providing necessary context of the problem
- **Exploratory Data Analysis**: Examining the dataset to understand the distribution of attributes, identify patterns, and detect anomalies
- **Evaluation Framework**: Defining evaluation metrics, selecting validation strategy, addressing potential issues such as over-fitting, and establishing criteria for model selection
- **Data Preprocessing**: Scaling numerical attributes and performing features selection
- **Models Implementation**: Training and fine-tuning different machine learning models
- **Analysis and Evaluation**: Evaluating the performance and choosing the best machine learning model

Contents

1 Problem Formulation

The first step in developing a model is to formulate the problem in a way that we can apply machine learning. In here, the task is to predict the **presence or absence of diabetes** using various health metrics and demographic information. Based on the problem type, the training dataset, and the `code_book.txt`, some information we can get are:

- This is a `Supervised Learning` and `Binary Classification` problem since the training data contains the `Status` column representing the occurrence of diabetes, which is also what we are trying to predict (0 for absence and 1 for presence).
- In the training dataset, there are in total 25 columns, in which:
  - The `Id` column will not be used since it does not contribute to the occurrence of diabetes.
  - The `Status` column will be used as the target variable for the training process.
  - The remaining 23 columns are the attributes and will be used for the data analyzing and model training.
- Data attribute types:
  - There are 6 `ordinal attributes` namely `GenHlth` (1 - 5), `MentHlth` (1 - 30), `PhysHlth` (1 - 30), `Age` (1 - 13), `Education` (1 - 6), and `Income` (1 - 8).
  - `BMI, ExtraMedTest, and ExtraAlcoholTest` are the `only 3 continuous attributes`.
  - The remaining 14 features are `binary attributes`, which can only be either 0 or 1.

2 Exploratory Data Analysis

In this section, we will work on Exploratory Data Analysis (EDA), a crucial step in the data analysis that allows us to gain insights into the structure, distribution, and relationships of different dataset attributes.

2.1 Descriptive Analysis

We begin by loading the training data into a `Pandas DataFrame`, in which we can extract some insight using `.shape` and `.describe` methods. Based on `output[2]`, we can see that:

- Our dataset has in total 202,944 rows (records) and 24 columns (23 of which are attributes and the `Status` column as the expected output).
- There are **no missing values** in the dataset since all columns have exactly 202,944 rows.
- The percentage of people suffering from High Blood Pressure and High Cholesterol are nearly the same at about 42%.
- `BMI` has values ranging from 12 to 98. The average BMI is 28.38, which is relatively high
- There are minor of people suffered from Stroke (4%), Heavy Alcohol Consumption (6%), No Doctor because of Cost (8%), Heart Disease and Attack (9%), and Walking Difficulty (17%).
- More than a half of people have a healthy lifestyle with Physical Activity at 76%, Fruits Consume at 64%, and Veggies Consume at 81%.
- Surprisingly, the statistics of `ExtraMedTest` and `ExtraAlcoholTest` are nearly the same.
- **Only 17.6% of people suffered from Diabetes (The Target variable)**, we can then conclude that we are having an **imbalance dataset**.

2.2 Variable Distribution

In addition to statistic features provided by `Pandas`, we can also use `Matplotlib` and `Seaborn` to visualize different attributes in the dataset. We begin by plotting the `Histogram` to see the data distribution of each attribute [2] and after that, we will use `Box plot` to see the variation of information [3].

Using **Histogram** (`figure 1`), we can see that:

- ExtraMedTest and ExtraAlcoholTest are relatively similar to each other and have a **symmetrical distribution**.
- BMI is **right skewed** with the majority fall between 20 and 30.
- HighBP, HighChol, Smoker, and Sex are **balanced**, while CholCheck, Stroke, AnyHealthCare, NoDocbcCost, MentHlth, PhysHlth, DiffWalk, Education, Income, and Status are **imbalanced**.
- In MentHlth and PhysHlth, the value of 30 gets surprisingly higher compared to previous values close to it. It is possible that someone, in the past 30 days, did not have a good mental/physical health experience.

On the other hand, with **Box plot** (`figure 2`), we can see that:

- There are a lot of outliers in BMI (from 40 to 100), MentHlth (from 5 to 30), and PhysHlth (from 7 to 30). However, it is possible that someone has a BMI up to 100 [4] and someone has been unwell for 30 days. We conclude that these outliers are likely not data errors and will keep all of them.
- Some outliers occurring in **binary attributes** like CholCheck, Stroke, HeartDiseaseorAttack, PhysActivity, Veggies, HvyAlcoholConsump, AnyHealthcare, NoDocbcCost, GenHlth, DiffWalk indicate that there are some imbalances between the two binary classes.

2.3 Correlation Analysis

In addition to visualizing data distribution, we can also display the correlation between different columns of the dataset. From the two plotted graphs (figure 3 and 4), we can see that:

- Some highly correlated attributes are: GenHlth and PhysHlth (0.53), GenHlth and DiffWalk (0.46), DiffWalk and PhysHlth: (0.48), GenHlth and Income (-0.37).
- CholCheck, Smoker, Stroke, Fruits, Veggies, HvyAlcoholConsump, AnyHealthcare, NoDocbcCost, MentHlth, and Sex are low correlated with Status with less than 0.1.
- On the other hand, ExtraMedTest and ExtraAlcoholTest are highly correlated with Status at 0.5.

3 Evaluation Framework

After performing EDA, the next step is to set up an effective Evaluation Framework that is suitable our problem and also helping us in evaluating different models' performance.

3.1 Evaluation Metrics

For a **Binary Classification** problem, there are different evaluation metrics that we can consider like accuracy, precision, recall, etc. However, since we are **detecting a disease** (diabetes), the positive class will be more importance due to the potential consequences of **missing positive instances** (false negatives). In addition, given the **imbalance in the dataset**, where the positive class occupies only 17.6% of the data, traditional evaluation metrics like accuracy may not adequately capture the performance of the model (For example, a model could achieve high accuracy score by simply predicting the majority class). As a result, we will prioritize metrics that are robust to class imbalance and emphasize the correct identification of positive instances. With that being said, we will focus on **F1-score** and **Precision-Recall AUC**, while not neglecting other metrics like **Recall**, **Precision**, and **Accuracy**.

- **F1-score**: Balances precision and recall, crucial for capturing positive instances accurately, and robust to class imbalance.
- **Precision-Recall AUC**: Quantifies precision-recall trade-off, robust to class imbalance, and essential for disease detection.
- **Recall**, **precision**, and **accuracy**: Offer complementary insights, and aiding comprehensive model assessment.

3.2 Validation Strategy

[figure 5] To ensure thorough model evaluation, we will firstly split our dataset into Training set (80%) and Validation set (20%). After that, the Training set will be used with GridSearchCV and RandomizedSearchCV to find the best set of hyperparameters for different machine learning models. In the two Searching methods, the `cv` technique will be **K-fold Cross validation** to ensure robustness and reduce over-fitting, while the `scoring` option will be **F1-score** as dicussed in Evaluation Metrics section. After finding all best estimators (hyperparameters) of different models, we will eventually evaluate their performance on Validation set (unseen data) as well as plotting `PR AUC` with different thresholds to rank and choose the best machine learning model.

4 Data Preprocessing

Before training the models, data preprocessing ensures that the dataset is cleaned, transformed, and organized in a way that optimizes the performance of the machine learning algorithms. Some of the steps can be:

- **Handling missing values**: Fortunately, there are no missing values in the dataset.
- **Handling outliers**: As mentioned in previous section, it is likely that the outliers that we saw are not data error and we will keep all of them.
- **Data Normalization**: By scaling the data to a common range, we ensure that all attributes contribute equally to the model's learning process, thereby preventing any particular attribute from dominating due to its larger range of values.
- **Features Selection**: Based on our analysis from the EDA, some features combinations that we can consider are: `Choosing all features`, `removing low correlated features to Status`, and `only selecting high correlated features to Status`. We will test the model on different combinations to see which one works best.

5 Models Development

After successfully completing EDA, setting up validation framework, and performing Data-preprocessing, we eventually reach the Models selection and implementation stage.

5.1 Models Proposal

For a **Binary Classification** problem, some of the possible ML models that we can consider are:

- **Logistic Regression (Sklearn)**:
  - `Strengths`: Easy to implement and interpret, outputs well-calibrated probabilities.
  - `Weaknesses`: Assumption of linearity between attributes and target variable, not suitable for complex relationship.
- **Decision Tree Classifier (Sklearn)**:
  - `Strengths`: Robust to outliers, works well with categorical attributes and non-linear relationship.
  - `Weaknesses`: Prone to over-fitting, may bias to the majority class in an imbalanced dataset.
- **Random Forest Classifier (Sklearn)**:

  - `Strengths`: An ensemble of decision trees, can mitigate over-fitting and improve generalization.
  - `Weaknesses`: Long prediction time, computational expensive since having a lot of trees.

- **Gradient Boosting Classifier (Sklearn)**:
  - `Strengths`: High predictive accuracy, robust to over-fitting, can handles mixed data types (Numerical and Categorical)
  - `Weaknesses`: Computational expensive, may overfit if the hyperparameters are not properly tuned

- **Gradient Boosting Machine (XGBoost)**:
  - `Strengths`: Powerful algorithms, fast running time, robust to class imbalance, often performs well compared to other models
  - `Weaknesses`: Hard to implement, a "Black Box" model which lacks of interpretability

5.2 Models Implementation

For the detailed implementation, please have a look at the jupyter notebook of this assignment. In general, we covered the following phases:

- **Baseline Performance**: Before working on model implementation, we need a way to quantify the effects of data preprocessing and hyperparameter tuning on model's performance. As a result, we will firstly implement the baseline models (without any parameters) to see how they are performing. This step is crucial because different models have different characteristics, some data preprocessing steps may work with one but may not work with others. For example, features scaling improves the performance of Logistic Regression but may not effect tree-based models like Random Forest or Decision Tree.

- **Hyperparameters tuning**: This phase follows closely with section `3.2 Validation Strategy`. Specifically, for each model, we perform hyperparameters tuning using the combination of RandomizedSearchCV and GridSearchCV. We firstly use RandomizedSearchCV to randomly find some good parameters' value and after that, use GridSearchCV to look up the area around those values. After finding the best hyperparameters combination, we make sure that we are not over-fitting by comparing the performance of models on both training set and validation set. In addition, we also plot the Confusion Matrix, Classification Report, and PR AUC chart of different models for better comparison.

6 Discussion and Conclusion

We can have a look at Table 1 (Appendix A), which summarizes the proposed models performance. As we can see, Logistic Regression performs the worst in all criteria, the reason for this could be our dataset contains many attributes of type categorical, which is not suitable for a linear model. Moving next, surprisingly, four remaining models both have relatively high performance, with F1-score at 0.88:

- Decision Tree Classifier and Random Forest Classifier both have AUC-PR of 0.94, however, the first model has a higher Recall (0.81 vs 0.79), while the second model has higher Precision (0.98 vs 0.96).

- The two last models, Gradient Boosting Classifier and Gradient Boosting Machine, seem to be the two best one, with AUC-PR of 0.95. Again, the first model has higher Precision (0.95 vs 0.94), while the second model has higher Recall (0.83 vs 0.81).

Since we are predicting disease (diabetes), the positive class will be more importance due to the potential consequences of missing **positive instances** (false negatives), thus, we will prioritize Recall over Precision. With that said, **Gradient Boosting Machine from XGBoost will be the best model selection for this problem** with the highest F1-score (0.88), highest AUC-PR (0.95), highest Accuracy (0.96), and highest Racall (0.83).

For now, the implemented models are just tree based and linear regression based and there are other approaches that we have not discovered yet. We hope that in future study, we will have more opportunity to apply more Machine Learning technique in solving real-world problem.

7 References

[1] CDC. (n.d.). Assessing Your Weight: BMI. Retrieved from https://www.cdc.gov/healthyweight/assessing/bmi/adult_bmi/

[2] NIST. (n.d.). Histogram. Retrieved from https://www.itl.nist.gov/div898/handbook/eda/section3/histogra.htm

[3] NIST. (n.d.). Box Plot. Retrieved from https://www.itl.nist.gov/div898/handbook/eda/section3/boxplot.htm

[4] CDC. (n.d.). Assessing Your Weight: BMI. Retrieved from https://www.cdc.gov/healthyweight/assessing/bmi/adult_bmi/

A Appendix

Model	**Best Estimator**	**F1-Score**	**AUC-PR**	**Others**
Logistic Regression	Polynomial Degree: 2, solver: ’lbfgs’, penalty: l2, C: 10, max_iter: 200	0.81	0.89	Accuracy: 0.94, Recall: 0.75, Precision: 0.88
Decision Tree Classifier	max_leaf_nodes: 50, min_samples_leaf: 20, min_samples_split: 10, max_depth: 12, criterion: ’gini’	0.88	0.94	Accuracy: 0.96, Recall: 0.81, Precision: 0.96
Random Forest Classifier	n_estimators: 100, criterion: ’entropy’, max_depth: 15, min_samples_split: 9, min_samples_leaf: 13	0.88	0.94	Accuracy: 0.96, Recall: 0.79, Precision: 0.98
Gradient Boosting Classifier	n_estimators: 73, learning_rate: 0.134, max_depth: 5, min_samples_split: 12, min_samples_leaf: 20, max_features: 0.343	0.88	0.95	Accuracy: 0.96, Recall: 0.81, Precision: 0.95
Gradient Boosting Machine	eta: 0.35, gamma: 2.79, max_depth: 3, min_child_weight: 18, max_delta_step: 9, subsample: 0.865, alpha: 5.9, tree_method: ’hist’, scale_pos_weight: 1.33, n_estimators: 77, colsample_bytree: 0.963, reg_lambda: 2	0.88	0.95	Accuracy: 0.96, Recall: 0.83, Precision: 0.94

Table 1: Best performance of proposed models

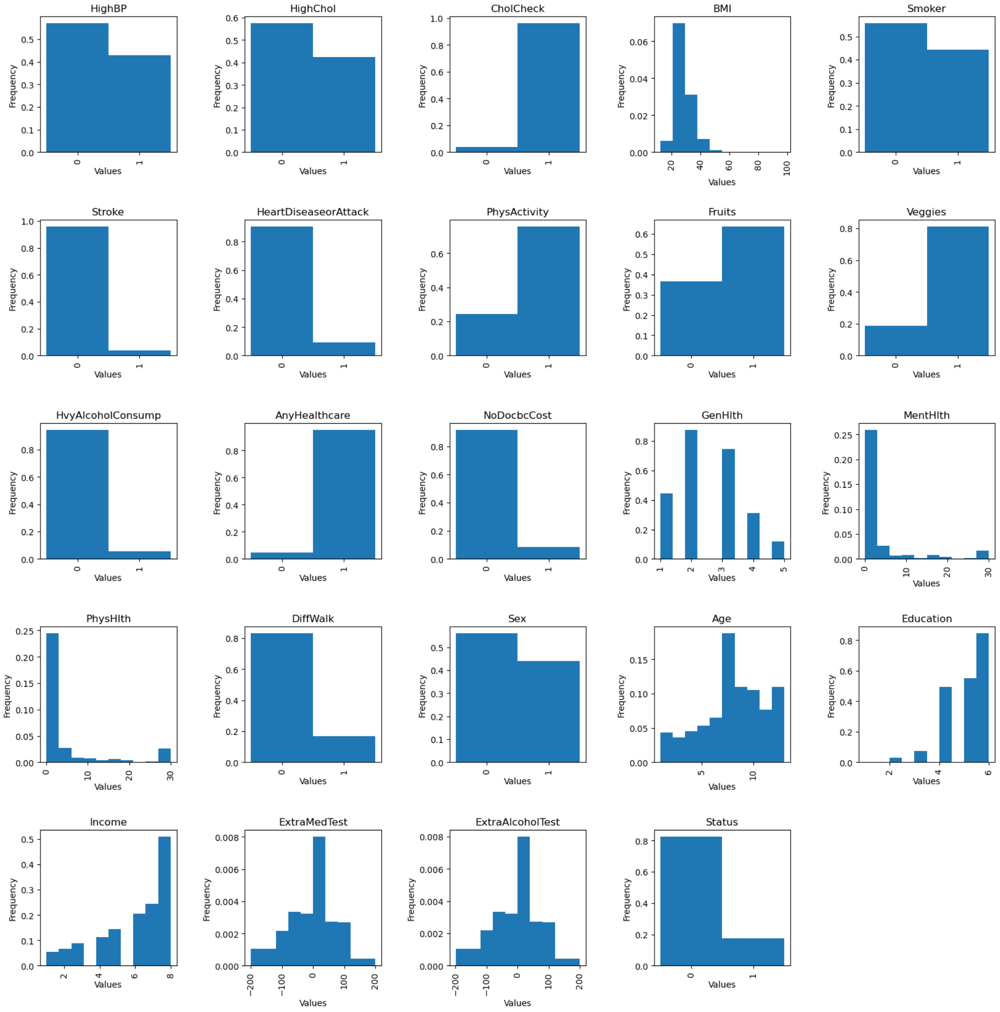

Figure 1: Histogram

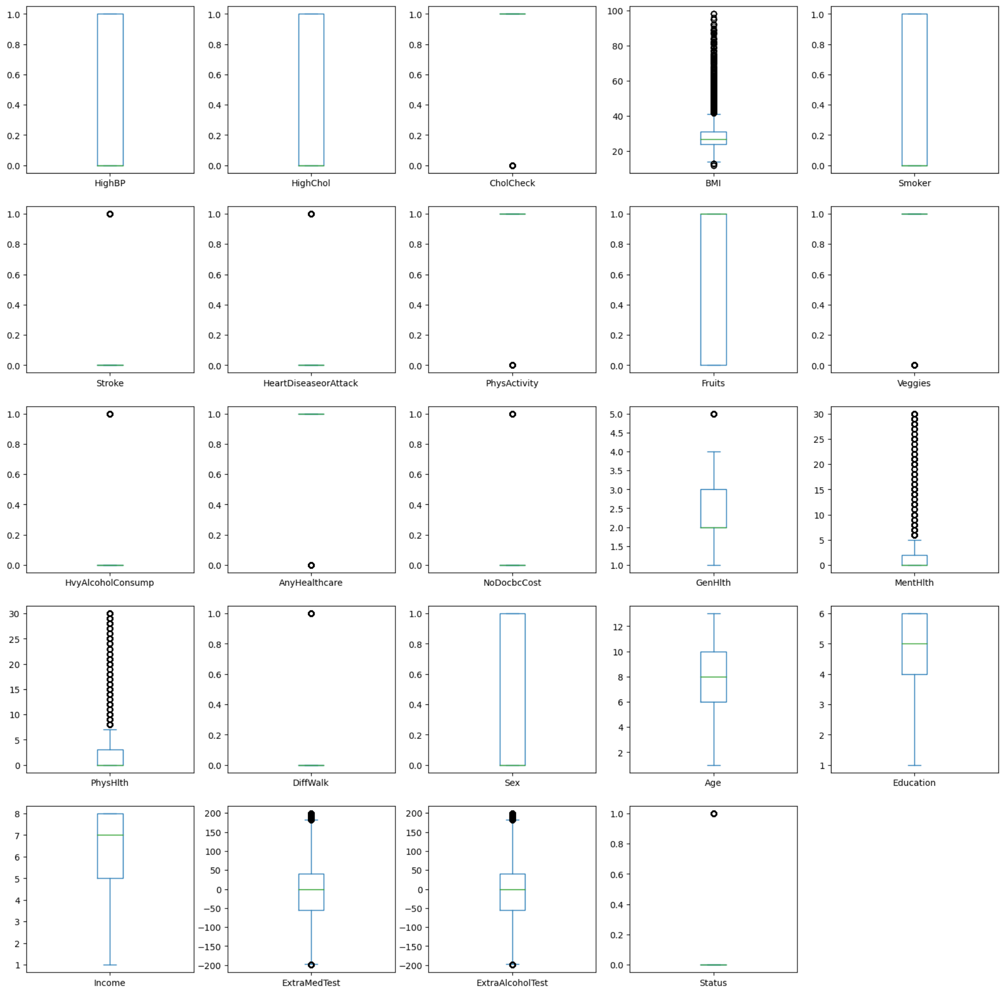

Figure 2: Box Plot

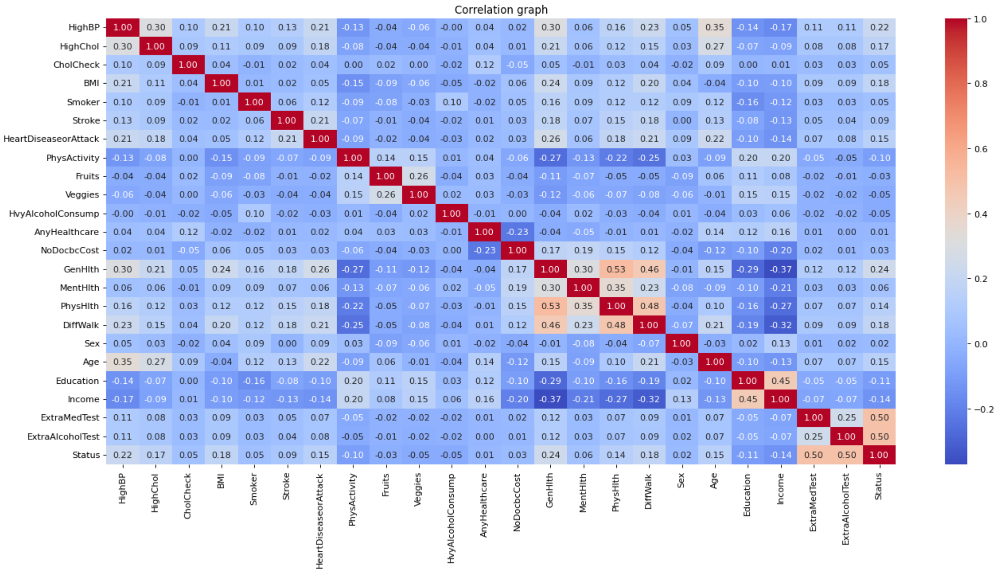

Figure 3: Correlation Chart

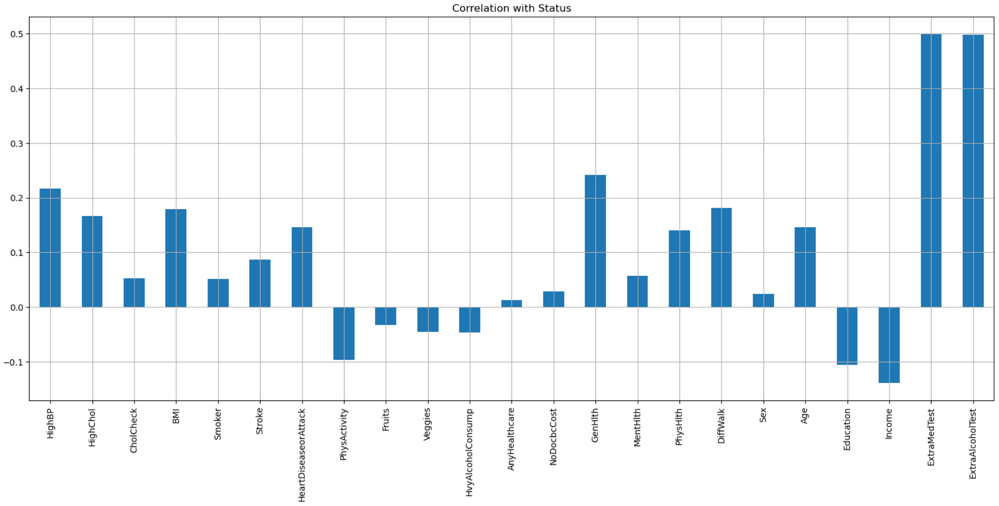

Figure 4: Features correlation with Status

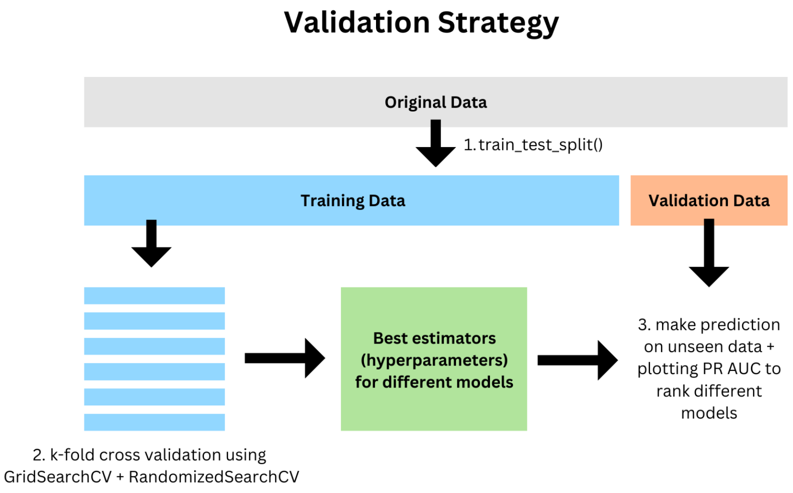
Validation Strategy
Original Data
1. train_test_split()
Training Data
Validation Data
Best estimators (hyperparameters) for different models
3. make prediction on unseen data + plotting PR AUC to rank different models
2. k-fold cross validation using GridSearchCV + RandomizedSearchCV

Figure 5: Validation Strategy