Li Tao

850001413

GDDA708

Assessment2\_Project

Table of Contents

[PART A – Regression Modelling for Business Decision-Making 2](#_Toc174638745)

[Task-1 Data Preparation 2](#_Toc174638746)

[a) Clean the dataset by addressing missing data and eliminating outliers 2](#_Toc174638747)

[b) Properly encode categorical features 6](#_Toc174638748)

[c) Apply feature scaling or normalization to the data 6](#_Toc174638749)

[d) Split the data into training and testing subsets 7](#_Toc174638750)

[Task-2 Model Development with Hyperparameter Tuning 8](#_Toc174638751)

[a) Select a suitable regression model for predicting the target variable 8](#_Toc174638752)

[b) Conduct hyperparameter tuning using grid search or random search to optimize the model’s parameters 11](#_Toc174638753)

[c) Train the regression model using the training data 13](#_Toc174638754)

[Task-3 Model Assessment and Selection 15](#_Toc174638755)

[a) Evaluate the performance of the regression model using appropriate metrics 15](#_Toc174638756)

[b) Utilize K-fold cross-validation to evaluate the model’s ability to generalize across different data splits 16](#_Toc174638757)

[c) Choose the best-performing regression model based on hyperparameter tuning and cross-validation results 18](#_Toc174638758)

[Task 4. Business Insights and Recommendations 21](#_Toc174638759)

[PART B – Classification Modelling for Business Decision-Making 23](#_Toc174638760)

[Task-1 Data Preparation 23](#_Toc174638761)

[a) Clean the dataset by handling missing values and removing outliers as necessary 23](#_Toc174638762)

[b) Encode categorical variables appropriately 27](#_Toc174638763)

[c) Perform feature scaling or normalization 28](#_Toc174638764)

[d) Split the dataset into training and testing subsets 29](#_Toc174638765)

[Task-2 Model Development with Hyperparameter Tuning 30](#_Toc174638766)

[a) Select an appropriate classification algorithm to predict the target variable 30](#_Toc174638767)

[b) Conduct hyperparameter tuning using a grid search or random search to optimize the model’s parameters 33](#_Toc174638768)

[c) Build the classification model using the training data 35](#_Toc174638769)

[Task-3 Model Assessment and Selection 36](#_Toc174638770)

[a) Compute and analyse the confusion matrix for the model 36](#_Toc174638771)

[b) Evaluate the model’s performance using relevant metrics 38](#_Toc174638772)

[c) Apply k-fold cross-validation to assess the model's generalization capability 40](#_Toc174638773)

[d) Choose the best-performing classification model based on hyperparameter tuning and cross-validation results 42](#_Toc174638774)

[Task 4. Business Insights and Recommendations 44](#_Toc174638775)

# PART A – Regression Modelling for Business Decision-Making

The dataset ‘house\_price’ was downloaded from Kaggle. It includes 1,460 house samples, each with 81 features. These features cover basic property attributes such as lot size, overall quality rating, number of rooms, garage spaces, house location, and the year of house was built. In this task, I used the dataset to perform regression modelling to predict the sale price of the house based on these features for business decision-making.

## Task-1 Data Preparation

### a) Clean the dataset by addressing missing data and eliminating outliers

I imported the relevant libraries at first and then loaded the dataset, showing its structure and shape to gain a basic understanding of the values. I checked for null values and found that some categorical values were missing. It is possible that some houses might not contain all the features due to the natural environment and other circumstances, but these features are important for the model. Hence, I implemented the ‘fillna()’ method to impute the missing values with ‘None’. After this action, the dataset was cleaned up with no noise.

|  |
| --- |
| I Imported the required libraries.  A screenshot of a computer code  Description automatically generated |
| I loaded the dataset. |
| I checked the structure and shape of the dataset.  A screenshot of a computer  Description automatically generated  A close up of a number  Description automatically generated |
| I checked the duplicated value.  A close-up of a computer screen  Description automatically generated |
| I checked the missing values and performed the imputation.  A screenshot of a computer  Description automatically generated  A screenshot of a computer code  Description automatically generated  A screenshot of a computer code  Description automatically generated |

|  |
| --- |
| I detected the outliers in the ‘SalePrice’ column, and I dropped the outliers. |
| A screenshot of a computer  Description automatically generated  A white rectangular sign with green text  Description automatically generated |

### b) Properly encode categorical features

I used the ‘pd.get\_dummies()’ method to convert the categorical variables into one-hot encoded columns. This approach transforms categorical data into a numerical format suitable for machine learning algorithms in prediction.

|  |
| --- |
| A table with numbers and letters  Description automatically generated |

### c) Apply feature scaling or normalization to the data

I used the ‘StandardScaler()’ function from ‘sklearn’ to normalize the features. This process involves subtracting the mean of each feature and dividing by the standard deviation, resulting in standardized features with a mean of 0 and a standard deviation of 1.

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

### d) Split the data into training and testing subsets

I split the dataset into training and testing sets. All columns except ‘SalePrice’ are included in ‘X’, and the ‘SalePrice’ column is assigned to ‘y’ as the target variable. 20% of the data is reserved for testing.

|  |
| --- |
| A white background with black text and red letters  Description automatically generated |

## Task-2 Model Development with Hyperparameter Tuning

### a) Select a suitable regression model for predicting the target variable

Selected model: Lasso Regression

In this task, I constructed three different regression models to find the most appropriate model. The dataset has a large number of features (793 columns), so it is crucial to select a regression model that can effectively handle high-dimensional data, mitigating overfitting, and providing integrity. After evaluating various models, I chose Lasso Regression due to its distinctive advantages that align well with the characteristics and requirements of the dataset and my specific goals for predicting.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer program  Description automatically generated |

Lasso Regression incorporates L1 regression, which can shrink some coefficients to exactly zero, which is beneficial for high-dimensional datasets as it effectively performs feature selection, simplifying the model, and reducing the computational burden. It allows for better understanding and insights into the most influential features affecting the target variable. In the ‘house\_price’ dataset, there are 793 features after one hot encoding, with Lasso’s Sparse solution, it is easy to find the most significant features. In addition, Lasso’s L1 regularization adds a penalty equivalent to the absolute value of the magnitude of coefficients, preventing the model from overfitting the training data, and thereby enhancing predictive performance.

Comparing with the other two alternative models:

* Linear Regression: Linear Regression performed poorly (with MSE of 1.69) due to issues like multicollinearity and inability to handle high-dimensional data effectively, as evidenced by its extremely high MSE.
* Ridge Regression: While Ridge Regression performed well (with MSE of 0.18) and mitigated overfitting by shrinking coefficients, it didn’t perform feature selection. This means all features remained in the model, which could be challenging regarding interpretability.

By using Lasso Regression, I can quickly find the most significant features selected and print them out.

|  |
| --- |
| A screen shot of a computer code  Description automatically generated |

I created a plot below to visualize the results using the default parameters. I found that the model matches the trends and patterns, though there are some minor errors extreme values.

Please find the screenshot below:

|  |
| --- |
| A screen shot of a graph  Description automatically generated |

Considering the above factors, Lasso Regression was chosen as the most appropriate model for predicting the target variable ‘house\_price’. By using this model, I ensure that both effective in prediction and efficient in handling high-dimensional data, ultimately leading to better generalization and insightful results.

### b) Conduct hyperparameter tuning using grid search or random search to optimize the model’s parameters

In this task, I aimed to optimise the hyperparameters of the Lasso Regression model by conducting a grid search to find the best ‘alpha’ to improve performance. Lasso regression includes a regularization term that penalizes large coefficients, which helps in preventing overfitting.

The key hyperparameter is ‘alpha’, which controls the strength of the regularization. A higher ‘alpha’ value indicates stronger regularization, while a lower ‘alpha’ value indicates weaker regularization.

First, I tuned the ‘alpha’ parameter with the following range of values: ‘[0.01, 0.1, 1, 10, 100]’. This range would be tested to find the optimal value, which covers a wide spectrum of regularization strength from weak to strong. Secondly, by applying the grid search, all parameter combinations in this range would be considered, making it a robust method to find the optimal hyperparameter values. The ‘cv=5’ means 5-fold cross-validation was used to ensure that the model’s performance was consistent across different subsets of the data.

The hyperparameter tuning process identified the best alpha value for Lasso Regression and resulted in the MSE value.

I found the best Lasso alpha is 0.01 and the MSE is 0.1479.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer code  Description automatically generated |

### c) Train the regression model using the training data

After finding the best value of ‘alpha’ in Task-2 is 0.01, I built the Lasso Regression model by using the best parameters and fitting the model to the training data.

I fit the model using the training data with the best alpha value determined earlier. Then I predicted the target values using the test set. To assess the model’s accuracy, I created a plot comparing the actual sales prices to the predicted values. This visualization allows me to see how well the model’s predictions align with the actual sales prices.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer program  Description automatically generated |

|  |
| --- |
| Please find the plot visualization below:  A graph of blue and orange lines  Description automatically generated |

After using the best alpha value of ‘0.01’ in the model for predicting house prices, the model performs better than the default version and matched most of the real values. However, there are some missing outputs, particularly with extreme values, which can lead to poor performance on some specific predictions. There are several reasons for this. Firstly due to very few high-priced houses in the dataset, the model does not have enough data to learn. Secondly, in some special cases, such as environmental and marketing issues, might result in a higher market price. In conclusion, the model performs very well in predicting the target values overall.

## Task-3 Model Assessment and Selection

### a) Evaluate the performance of the regression model using appropriate metrics

I used the following metrics to evaluate the model’s performance, Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared.

* MAE measures the average magnitude of the errors in a set of predictions, in this case, it equals 0.225 the lower values indicating better model performance.
* MSE measures the average of the squares of the errors, sensitive to outliers, in this case, it equals 0.1478 the lower values indicating better performance.
* R-squared varies from 0 to 1, where a value closer to 1 indicates a large proportion of variance is explained by the model. In this case, it equals 0.878 indicating the goodness of fit.

The results indicate that the Lasso model with alpha set to 0.01 provides strong predictive performance and a high level of accuracy, with a high R-squared value of 0.878, suggesting that approximately 87.8% of the variance in the target variable is explained by the model. The relatively low MSE and MAE values further confirm the model’s accuracy and effectiveness.

|  |
| --- |
| A computer screen shot of a error  Description automatically generated |

### b) Utilize K-fold cross-validation to evaluate the model’s ability to generalize across different data splits

I used the K-Fold cross-validation to train and assess a Lasso Regression model by dividing the data into k subsets. The model was iteratively trained and evaluated on these folds. Specifically, I used 5-fold cross-validation and evaluated the model’s performance based on MSE. This method helps in determining the most effective model configuration and validating its performance across various data segments.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer program  Description automatically generated |

I initialized the Lasso Regression model with ‘alpha’ set to 0.01 and K-Fold cross-validation with 5 splits, shuffling the data before splitting. For each fold, the dataset is split into training and testing sets and the MSE is calculated for each fold and stored in ‘fold\_results’. The fold with the lowest MSE is identified as the best fold. Finally, the MSE for each fold is printed and the best fold with the lowest MSE is identified.

The model demonstrated consistent performance across all folds with relatively low MSE values, with the lowest MSE observed in Fold 3 (0.0939). The K-Fold cross-validation approach provided a comprehensive evaluation of the model’s performance across multiple data splits, ensuring that the chosen model is both effective and reliable for predicting the target variables.

### c) Choose the best-performing regression model based on hyperparameter tuning and cross-validation results

Based on the hyperparameter tuning and K-Fold cross-validation results, the Lasso Regression model was built.

Hyperparameter tuning result: ‘alpha’ set to 0.01

K-Fold Cross-validation result: Fold 3 with an MSE of 0.0939

|  |
| --- |
| A screenshot of a computer program  Description automatically generated |

After I retrained the Lasso Regression model on the best fold, I evaluated its performance with the below metrics:

An impressive MSE of 0.0939 indicates the model’s ability to minimize prediction errors effectively.

The R-squared value of 0.916 suggests that the model performs a high level of explanatory power indicating the model’s suitability for making accurate predictions.

In the context of business problems, which involve predicting a target variable based on numerous features, Lasso Regression’s ability to handle high-dimensional data, select relevant features, and provide predictions makes it an ideal choice. This model not only minimizes prediction errors but also ensures interpretability and generalizability, making it highly suitable for practical applications in the real estate and house business industry.

To plot the test data and the predicted data on the graph, I observed that the predicted results were closer to the actual values compared to the previous models, indicating performance improved. Although some values were still not matched, it was due to various reasons such as unexpected environmental changes or market shocks, so it was understandable and acceptable.

|  |
| --- |
| A graph with blue and orange lines  Description automatically generated |

After that, I visualized the residuals for the final model. Residuals are the differences between the actual values and the predicted values made by the regression model, providing insights into how well the model is performing. In the plot graph, residuals are roughly normally distributed, suggesting that the model’s errors are distributed in a way that aligns with the assumptions of Lasso Regression. This is a positive sign indicating the model is well-specified. A slight left skew means that there are more residuals with smaller values, indicating that the model may be slightly underestimating predictions in some cases. The fact that the residuals fall within a narrow range (-1 to 1) suggests that the model predictions are generally close to the actual values, with only minor deviations. This is consistent with the low MSE and high R-squared value, indicating overall good model performance.

|  |
| --- |
| A screenshot of a graph  Description automatically generated |

## Task 4. Business Insights and Recommendations

Model performance:

The final model has an MSE of 0.0939 on the best fold test data. This relatively low error indicates that the model performs well in predicting house sales prices, making it suitable for practical applications. The R-squared value of 0.916 means that the model can explain 91.6% of the variability of the target variable, indicating the model fits the data well and most of the sales price fluctuations can be explained by the model.

To help with the business decision and provide some recommendations, I found the top 20 features that Lasso Regression model selected.

Top 20 influential features:

|  |
| --- |
| the most influential features include: |
| A screenshot of a computer  Description automatically generated |

Actionable recommendations:

These five features have the most impact on housing prices, ‘MasVnrArea’, ‘OverallQual’, ‘GrLivArea’, ‘YearBuilt’, and ‘BsmtFinSF1’. High positive coefficients suggest that better quality and larger living areas significantly increase the sale price. However, the masonry veneer area has a negative coefficient, indicating a potential decrease in value with the extreme size of the house. Nonetheless, the house-built year and basement quality also play crucial roles in determining the sale price.

Pricing strategy: set competitive prices based on influence characteristics. For example, properties with higher overall quality, bigger living areas, and bigger first-floor areas should be priced higher because these features significantly increase the sale price.

Investment focus: Invest in properties or renovations that improve key features such as overall quality, kitchen quality, roof material quality, and living area. Improvements in these areas could lead to higher returns.

Market positioning: Use neighbourhood characteristics and kitchen insights to target a specific market or demographic. Properties in high-value neighbourhoods or those with upgraded kitchens can be marketed as high-end products.

Feature Enhancement: For future development or renovation, focus on enhancing the influential features identified in the model. This approach can help boost property values and enhance market competitiveness.

By focusing on these influential features, real estate companies can data-driven decisions to enhance property values, optimize marketing strategies, and meet buyer preferences.

# PART B – Classification Modelling for Business Decision-Making

The dataset ‘Heart\_Disease’ was downloaded from Kaggle. The dataset consists of 445,132 records and 40 features, including demographic information, health indicators, medical history, lifestyle factors, and physical measurements, providing a rich source of information to model heart attack risk. In this task, my goal is to predict the likelihood of a heart attack based on these features by implementing a classification regression model.

## Task-1 Data Preparation

### a) Clean the dataset by handling missing values and removing outliers as necessary

I imported relevant libraries and performed technologies to understand the structure of the dataset.

|  |
| --- |
| A screenshot of a computer  Description automatically generated |
|  |
| A close-up of a number  Description automatically generated |
| A screenshot of a computer screen  Description automatically generated |
| After I checked the missing values and data types, I decided to handle these null values with imputation and fill null. |
| A screenshot of a computer code  Description automatically generated |
| I droped the duplicated values and the irelevant “State’ column.  A screenshot of a computer code  Description automatically generated |

After I checked the outliers for numeric columns, I decided to keep the outliers in the dataset for the machine learning model because this task aims to predict the possibility of having a heart attack and identify the features that affect this possibility. Some features with abnormal values might be significant indicators of the features I am looking for.

|  |
| --- |
| A screenshot of a computer code  Description automatically generated |
| A group of graphs showing different sizes of numbers  Description automatically generated with medium confidence |

### b) Encode categorical variables appropriately

I used one hot encoding method ‘pd.get\_dummies’ to handle the categorical values. This approach ensures the categorical variables are properly encoded.

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

### c) Perform feature scaling or normalization

I applied standard scaling to the dataset ensuring that each feature has a mean of 0 and a standard deviation of 1, which is beneficial for the following tasks when I perform machine learning algorithms that are sensitive to the scale of the data.

|  |
| --- |
| A screenshot of a computer  Description automatically generated |
| I droped the columns where it contains a particular pattern ‘\_None’, because these columns are repeated and had 100% correlation with the same columns names that contained the pattern ‘\_Yes’.  A screenshot of a computer code  Description automatically generated  A screenshot of a graph  Description automatically generated |

### d) Split the dataset into training and testing subsets

I split the dataset into the independent variable X and the dependent variable y (‘HadHeartAttack\_Yes’). Then I converted the values in target variable y to binary classes, preparing for a better understanding of the classification model. Finally, I set up the training and testing sets.

|  |
| --- |
| A close-up of a math problem  Description automatically generated  A screenshot of a computer program  Description automatically generated |
|  |

## Task-2 Model Development with Hyperparameter Tuning

### a) Select an appropriate classification algorithm to predict the target variable

Selected model: Logistic Regression

The Logistic Regression model is the most appropriate model given the dataset’s size and the need for computational efficiency. It offers a good balance of accuracy and interpretability, which is crucial for understanding the factors contributing to heart disease predictions.

To gain a deeper understanding of the model’s performance, I built and compared the reports of the Logistic Regression model and the Random Forest model.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer code  Description automatically generated  Please find the report below:  A screenshot of a report  Description automatically generated |

Comparison with the other alternative models:

* Random Forest model is more complex and requires more computational resources for training and prediction, especially with large datasets. In my practice, the model achieved a similar accuracy to Logistic Regression. However, it showed a lower recall for the minority class, indicating it may not perform as well in identifying the less frequent class.
* Support Vector Machine (SVM) is effective in high-dimensional spaces but it is computationally expensive, making it likely impractical for a large dataset due to high computational demands resulting in longer training times and higher resource consumption, which indicates that SVM may be less practical for this practice.

The Logistic Regression is chosen for the following reasons:

* Handling large dataset: The dataset consists of 68 columns and 444975 rows. Logistic Regression is efficient in handling large datasets compared to more complex models with less computational resources required.
* Accuracy: The model achieved an accuracy score of 94.6%, which is competitive, performing well in distinguishing between ‘o’ and ‘1’ classes.
* Interpretability: The model provides a straightforward interpretation of coefficients, which is valuable in medical applications where understanding feature impact on predictions is crucial.
* Resource consumption: Logistic Regression has a lower computational cost compared to Random Forest and SVM, ensuring faster training times and reducing computational requirements.

In summary, considering the data size and the complexity, Logistic Regression is a suitable algorithm for predicting the target categorical variable ‘Had\_heart\_attack’.

### b) Conduct hyperparameter tuning using a grid search or random search to optimize the model’s parameters

I used ‘GridSearchCV’ to find the best hyperparameter for the Logistic Regression model. Grid search systematically explores all possible combinations of a predefined set of hyperparameters. While random search is generally faster, it may not find the global optimum. Compared to the random search, grid search is more reliable but time-consuming. Since my dataset is not very large and complex, grid search is more practical.

Define hyperparameters for tuning:

* ‘C’ (Regularization strength): ‘C’ controls the inverse of the regularization strength. Smaller values indicate stronger regularization, which can prevent overfitting but might underfit if too large. I tested a range of ‘0.01, 0.1, 1.0, 10, 100’ to find a balance that improves model performance.
* ‘penalty’ (Regularization type): ‘L1’ penalty performs feature selection by shrinking some coefficients to zero, which can be useful if some features are irrelevant. ‘L2’ penalty helps to distribute the regularization evenly across all features.
* ‘solver’ (Algorithm to optimize): the ‘saga’ solver is suitable for handling large datasets and supports both ‘L1’ and ‘L2’ penalties.
* ‘max\_iter’ (Number of iterations): this parameter controls the maximum number of iterations for convergence.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer program  Description automatically generated |

By conducting hyperparameter tuning, I can optimize the model’s performance by selecting the best combination of parameters. In this case, the set of best parameters is selected.

‘C=0.01’: a small regularization strength to control overfitting.

‘max\_iter=100’: sets the maximum number of iterations for convergence.

‘penalty=L1’: encourage sparsity in feature selection.

‘solver=saga’: suitable for large dataset and supports L1 regularization.

### c) Build the classification model using the training data

I initialized the Logistic Regression model with the optimal parameters identified from the Grid search. The model was trained on the ‘X\_train’ and ‘y\_train’ data to capture the relationships between the features and the target variable. Then I used this trained model to generate predictions on the ‘X\_test’ dataset. These predictions are then compared against the actual target values to assess the model’s performance.

To evaluate the model, I used a classification report and accuracy score. The classification report provides detailed metrics including precision, recall, and F1-score for each class, while the accuracy score indicates the overall performance of the model.

The model shows strong performance on the majority class but faces challenges with the minority class. This highlights the challenge of imbalanced datasets and suggests some additional techniques may be needed to improve the performance on the minority class.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer program  Description automatically generated |

## Task-3 Model Assessment and Selection

### a) Compute and analyse the confusion matrix for the model

The confusion matrix provides a summary of prediction results showing the number of true positives, true negatives, false positives, and false negatives.

Matrix breakdown:

|  |  |  |
| --- | --- | --- |
|  | Predicted 0 | Predicted 1 |
| Actual 0 | 83,086 | 887 |
| Actual 1 | 3,886 | 1,136 |

|  |
| --- |
| A screenshot of a computer  Description automatically generated |

* True negatives: 83,086, indicating where the model correctly identified the absence of a heart attack.
* True positives: 1,136, indicating where the model correctly predicted the presence of a heart attack.
* False positives: 887, indicating where the model incorrectly classified the absence of a heart attack as a presence.
* False negatives: 3,886, indicating where the model incorrectly classified the presence of a heart attack as an absence.

True negatives and true positives: this contributes to the overall high accuracy of the model.

False positives: 887 is relatively low indicating that there are a few instances where the model incorrectly predicts a heart attack when it is not present.

False negatives: 3886 is relatively high meaning the model misses several actual heart attacks. This contributes to the lower recall for positive class.

### b) Evaluate the model’s performance using relevant metrics

I utilized the performance metrics from the classification report to provide a detailed assessment of the Logistic Regression model’s performance.

Performance metrics:

* Accuracy score:

Accuracy score of 0.9464 is a good general metric but may not fully reflect the model’s performance in cases of class imbalance.

* For class 1 (Positive cases):

Precision of 0.5615 means that when the model predicts heart disease, it is correct 56% of the time. Recall of 0.2262 indicates that the model correctly identifies 22% of the actual heart disease cases. F1-Score of 0.3225 provides a balance between precision and recall. A low F1-score suggests the model’s performance in identifying positive cases is not satisfactory.

* For class 0 (Negative cases):

Precision of 0.96, recall of 0.99, and F1-Score of 0.97. The model performs very well on most cases in class 0.

|  |
| --- |
| A screenshot of a computer program  Description automatically generated |

The Logistic Regression model shows strong performance in identifying the majority class (class 0) but needs improvement in identifying the minority class (class 1). Due to the class imbalance impact, for the minority class (class 1), the model has lower performance metrics, particularly in the recall, suggesting the model is struggling to identify positive cases effectively.

Focusing on detecting the negative cases can be a strategic choice depending on the implications of false positives versus false negatives.

By analyzing the ROC curve and AUC score, I can have a better understanding of the model’s performance. The ROC curve indicates the model performs well with a steep rise towards the top-left corner with a high AUC score of 0.88.

|  |
| --- |
| A screen shot of a computer screen  Description automatically generated |

### c) Apply k-fold cross-validation to assess the model's generalization capability

By employing hyperparameter tuning in the previous task, I optimized the model’s performance by defining the best combination of parameters.

The optimal set of parameters used was as follows:

‘C=0.01’: a small regularization strength to help control overfitting.

‘max\_iter=100’: specifies the maximum number of iterations for convergence.

‘penalty=L1’: encourages sparsity in feature selection.

‘solver=saga’: suitable for large datasets and supports L1 regularization.

I implemented the Logistic Regression model by using the best hyperparameter defined in the previous task. I employed 5-fold cross-validation. This process involved training and validating the model 5 times, each time using one fold as the validation set and the remaining four folds for training. For each fold, I split the data into training and validation sets, trained the model on the training set, and generated predictions for the validation set. Accuracy scores were computed for each fold and recorded. In the end, I reviewed the accuracy scores for all folds and identified the fold with the highest accuracy.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer program  Description automatically generated  A screenshot of a calculator  Description automatically generated |

The results show that the model performs consistently across different folds, with accuracy scores ranging from 94.61% to 94.72%. And both Fold 3 and Fold 5 achieved the best performance in accuracy of 94.72%. To conclude, the Logistic regression model has demonstrated consistent performance across the different folds in the K-Fold cross-validation process.

### d) Choose the best-performing classification model based on hyperparameter tuning and cross-validation results

After performing hyperparameter tuning and K-fold cross-validation, the best fold was identified with the highest accuracy score. This fold represents the optimal configuration of hyperparameters and model performance across different subsets of the training data. The final classification model is built using the bast fold which is Fold 3, and the best combination of the parameters: ‘C=0.01’, ‘max\_iter=100’, ‘penalty=L1’, ‘solver=saga’.

After retraining, the model was evaluated using various metrics to assess its overall performance. The focus was on understanding how well the model predicts both classes (0 and 1) in the dataset.

The metrics report indicates that the overall accuracy is high, largely because the model performs well on the majority class (class 0), effectively identifying true negatives with a precision of 0.96. However, accuracy alone can be misleading especially in the context of imbalanced datasets where one class is much more frequent than the other. The imbalance in the dataset and the model’s performance on the minority class (class 1) highlights a potential limitation. While the model is reliable for predicting the majority class, its ability to predict the minority class is less effective.

|  |
| --- |
| Please find the screenshot below:  A screenshot of a computer  Description automatically generated |

## Task 4. Business Insights and Recommendations

Model performance:

The final model performs well in predicting the majority class (class 0), contributing to high overall accuracy of 0.95, but struggles with positive cases (class 1) due to the class imbalance in the dataset. The macro average metrics, which give equal weight to each class, show a more balanced view, with a lower F1-score reflecting the challenges with the minority class. The feature importance analysis reveals that health conditions and age are significant predictors, providing valuable insights into what drives the model’s decisions.

The top 20 features by absolute coefficient values indicate which features have the most significant impact on the model’s predictions.

|  |
| --- |
| A screenshot of a computer program  Description automatically generated |

Based on the Logistic regression model’s top features, there are some actionable recommendations and insights for predicting heart attack risk.

* Significant positive predictors:

HadAngina\_Yes: having angina is strongly associated with an increased risk of heart attack.

Sex\_Male: males are at higher risk compared to females.

Age category: 70+ years generation shows a significant risk in heart health.

GeneralHealth\_Fair to Poor: poor general health is associated with higher risk.

* Significant negative predictors:

SmokerStatus\_Never smoked: non-smokers have a lower risk.

Age category: 25-34 years generation shows a lower risk, suggesting younger individuals should not be the primary focus.

Recommendations:

Increase awareness and monitoring for patients with a history of angina and those who have undergone chest scans.

Educate on the importance of maintaining good general health and regular check-ups.

Implement proactive health screenings and intervention programs for older adults and males with poor health. Various age categories, from ‘age 60 to 64’ to ‘age 80 to older’, show positive coefficients, indicating a higher likelihood of classification with increasing age.

Promote smoking cessation and healthier lifestyle choices as a preventive measure for heart attack.

For reference, please find the relevant code at the GitHub link below:

<https://github.com/YiliaTao0122/708Assessment2>