COMP1816 - Machine Learning Coursework Report

amsmath graphicx

Foggi Diego - 001189620
Word Count: No more than 3000 words (edit here)

1. Introduction

In this document is shown a full machine learning process of Regression and Classification models applied on California Housing Prediction and Titanic Survival Prediction datasets respectively. Firstly, data are displayed and analysed to understand what is under the hood, after that starts the cleaning process. In the second step we split the data into train and test and standardize the features for a better performance of the chosen training models. The third step is the application of the models where it is making the calculation of each respective model. The final step work on the evaluation of the outcomes of the models and their evaluations, then a comparison in made between their results to see which perform the best.

California Housing Price is a supervised dataset with multiple regression problems. It is applied as first basic model the Linear Regression to predict a continuous value of house's price as the data are numeric.

Lasso Regression is the follow implementation model that has the capacity of prevent overfitting values. The last model that is implemented is Ridge Regression. It works as a regularization method and because it works well with multicollinearity can add penalty term for overfitting circumstances.

Titanic dataset is also a multi regression problem and it is used the Classification method to predict a passenger's survival based on the given information. It is applied the Logistic Regression Model that work in binary classification problem. The other two models are Random Forest Regression and Decision Tree Classifier. By splitting the feature area into regions with corresponding results, decision tree is capable of defining complex decisions. Random Forests, which build multiple decision trees, can mitigate this by reducing variance without increasing bias, making it powerful for handling tabular data.

2. Regression

Dataset description The dataset contains 1000 rows (districts) and 10 columns (features). The dataset shows that there are 12 missing values in *total-bedrooms* feature and 7 in "*ocean_proximity*". Furthermore, it is identified eleven duplicated rows in the dataset along the rows.

All attributes are numerical except for ocean_proximity features that are texted. The column describes what is the location of the house: *"near bay", "inland", "¡1h ocean", "near ocean"*, *"island"* and presents a column named "*No"*. that it is irrelevant for the machine learning job. Furthermore, the target is mean_house_value that represent each district's median house prices.

The statistical representation of the dataset shows an interesting dynamic, for example the median_house_value vary from 15,000 to 500,000 with mean of 207,000. The *median_income* shows the range from 0.536 to 15,000 with mean of 3.9 that show the economic differences. Another important feature to illustrate is *housing_median_age* that ranges between 2 and 52 years with the mean of 27 years old. Another interesting feature is the *total_rooms* because some districts are denser than the others and the mean is 2,206.

In this step we find which features are most relevant for the prediction that ranges from -1 to 1 that when the correlation approximating to these two values means that they have very strong correlation, but when the values are close to 0 it means that they have weak correlation. Some features are strongly correlated with the target but other have very little correlation. For example, we have strong correlation with median_income at 0.67 and the total bedroom shows the weakest correlation

with value 0.02.

2.1. Pre-processing

In this step we prepare the dataset for modelling starting to handle the missing values in *total_bedroom* and *ocean_proximity* furthermore, we discovered that there are 22 duplicates districts. We decided to drop 11 districts and shorten a bit our dataset to 989 cleaned rows.

Simple Imputer and *OneHot Encoder* to fill the missing value using the median method for *total_bedroom* and mode method for *ocean_proximity.* Within this process the categorical data become a Sparse Matrix, then it is changed into a NumPy Array and further it is changed in integer data type, afterward using *dummies* method it divides feature into four specific ones and transform afterward the values into binary (1= True, 0 = False).

The first thing is to create more feature from the given ones to see which one can make the correlation stronger and therefore a more robust dataset for our models. Successively, I obtained *rooms_per_house*, *bedrooms_ratio*, and *pop_per_house* from total_rooms, households, population and total_bedrooms.

The "No." column is dropped as it has no predictive information then it isn't useful for our aim. Then we drop also total_rooms, total_bedrooms, households and population because they have strongly correlation between each other's bot very low correlation with the target.

From this moment we split the dataset primarily into train and test with the last 190 datapoints for testing, successively the Train Set is split into Train and Validation exactly 80% and 20% of the remaining Train Set of 799 rows. In this process we need to preserve the Test Set for the last evaluation that happens when firstly we train the models that is chosen and after that validating. The outcome will bring the best model in which the Test Set will be applied.

Finally, we apply the Standard Scaler that transform the dataset values into a range between -1 to 1 with as some features have values that differ too much for example the massive gaps between the 75th quartile and maximum of most of the features. Now the dataset is ready for the models.

2.2. Methodology

Linear Regression Model

Linear Regression is a model that works in mathematic linear calculation that compute the weighted value of input features "X" known as independent variables and the target "y" known as dependent variable. This model is used as a base model because of its simplicity and interpretability using a linear equation to predict the house median value. This model doesn't require any hyperparameter because it hasn't neither regularization nor complex parameters.

The linear regression model can be formulated as:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \epsilon \tag{1}$$

where y is the dependent variable, β_0 is the intercept, β_i are the coefficients for the predictor variables x_i, and ϵ represents the error term, which accounts for the variability in y not explained by the predictors.

Lasso Regression Model

Lasso Regression is a form of regularization that adds a penalty to the cost function. The Lasso penalty is, on the other hand, proportional to the absolute coefficient's values and not their squares. In contexts such as the dataset we are discussing, this difference leads to some unique characteristics and benefits of Lasso Regression. The ability of Lasso to generate a sparse model with many zero coefficients can be computationally efficient and produce a model that is easier to understand

The lasso regression model adds a regularization term to the standard least squares objective, which penalizes the absolute size of the regression coefficients. The objective function for lasso regression can be formulated as:

$$\min_{\beta} \left\{ \frac{1}{2n} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{2}$$

where:

- y_i is the dependent variable,
- β_0 is the intercept,
- β_j are the coefficients for the predictor variables x_{ij},
- n is the number of observations,
- p is the number of predictors,
- λ is the regularization parameter that controls the amount of shrinkage: the larger the value of λ, the greater the amount of shrinkage.

Polynomial Regression Model

Polynomial Regression is a model that works for non-linear relationship between the dependent variables and independent variables. This model is chosen because looking at the linear regression plot graph we might perceive that the plots change a little making a slight curve bending down for few degrees, therefore it might be a good choice if we want a better result. Polynomial regression can be expressed as follows:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n + \epsilon \tag{3}$$

where:

- y is the dependent variable,
- x is the independent variable,
- $\beta_0, \beta_1, \ldots, \beta_n$ are the coefficients of the model,
- $x^2, x^3, \ldots, x^n$ represent the polynomial terms of x,
- ϵ is the error term, which accounts for the variability in y not explained by the polynomial terms.

This model allows for the modeling of relationships that are not linear by incorporating higher power terms of the predictor variable.

2.3. Experiments

2.3.1. Experimental Settings

HYPER-PARAMETER To identify the best model for your dataset we can perform hyperparameter tuning for the Lasso and Polynomial Regression models. However, for Linear Regression, there are no hyperparameters to tune since it doesn't include regularization and focuses on finding a linear relationship between the input variables and the target.

Lasso Regression uses *alpha()* parameter to control how much we must penalize the coefficient using large values to prevent overfitting, we can tune the regularization strength to find the optimal setting and we sets it from 0.01 until 100 with the x10 steps. We run the model on all the parameter using the validating data until we found the best hyperparameter.

Polynomial Regression has degrees as parameter that determine the complexity of the model that occur when we put high values. We set from 1 to 5 degrees to run through to find the best model and then set rerun it on test set.

Mean Square Error (MSE)

The Mean Square Error (MSE) for a set of predictions can be formulated as follows:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{4}$$

where:

- n is the number of observations,
- y_i is the actual value of the observation,
- $\hat{y}_i$ is the predicted value of the observation.

This metric provides a simple measure of the prediction error's magnitude and is widely used in regression analysis to assess model performance.

Coefficient of Determination (R^2 Score)

The R^2 score can be defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5}$$

where:

- y_i are the actual observed values,
- $\hat{y}_i$ are the predicted values,
- $\bar{y}$ is the mean of the observed data y_i,
- n is the number of observations.

This formula represents the proportion of the variance in the dependent variable that is predictable from the independent variable(s).

2.3.2. Results

Analysis

Linear regression as a baseline model provides moderate result with:

- Training Score: 0.6133 or 61.33%

- R^2 score: 0.3794 or 37.94%

- MSE score: 6635360851.107786

Lasso Regression performed almost the same as the base model. The best model after experimenting the alpha parameter is at 0.01 value.

- Training Score: 0.6609 or 66.09%

- R^2 score: 0.3794 or 37.94%

- MSE score: 6635354280.806387

Polynomial Regression with Best polynomial degree 2 has the best fit result of the three models.

- R^2 score: 0.5611 or 56%

- MSE score: 4663335719.516269

2.3.3. Discussion

Linear Regression is the most fundamental type of regression analysis. This type of analysis considers that relationships between the independent variables and the dependent variable exist in a linear form.

Helpful for preliminary work to determine a baseline. Nevertheless, it most probably fails for more advanced data sets in which the relations among the predictors are multicollinear or the response variable is influenced in a non-linear fashion.

By incorporating a penalty equal to the absolute value of the coefficients, Lasso Regression modifies Linear Regression. It is simpler to understand and assist in feature selection as it shrinks the coefficients of the less significant features to zero - making the model less complex.

In Polynomial Regression, a type of regression analysis, the connection between the dependent variable and independent variable is modeled as an nth degree polynomial. The non-linear relationships are captured by polynomial terms. This model is best suitable for datasets where it is obvious that the relationship between the variables is non-linear. There is a possibility of overfitting the data, thus it is important to carefully balance the degree of the polynomial.

3. Classification

Dataset description The dataset format is of 890 rows and 11 columns, and the target feature is the "Survival" as we want to predict the survivals. Every row is a person information that embarked in the Titanic ship. The dataset contains missing values in every feature but the PassengerID and Sex that are full. The Age column contains 173 missing values, and it is the only one that has a lot of missing entry among the others. We can notice that there are four categorical columns: "Name", "Sex", "TicketNo." and "Embarked", but the others are numerical.

Summary Statistic displays some outliers at Age column, showing -20 as minimum and 3000 as maximum that later we fix. After that we displayed out how many categories has every categorical column, and we can notice that there are 544 perished people against the 342 survived. Moreover, we individuate that there are 4 missing information that we don't know if they survived or not.

Here are the survival distribution graphs showing the correlation of some feature against the survival target.

Display some graphs:

Correlation matrix displays how strong are the relationship between the features pointing out the strongest ones but, we want to pay more attention at those one which are more correlated with the survival.

3.1. Pre-processing

Firstly, we handle the missing data with median for numerical and mode for categorical features. Then we dealt with outliers put an empty value and then fill them with median calculation. As we don't know anything about the 4 missing values at *survival* column, we decided to drop the rows. From some features we create different ones doing feature engineering. *Sex* feature is split into male and female columns and *Age* is divided into *Baby, Young, Adult* and *Old* columns. *Sib* and *Parch* columns share the same concept as family member and aren't strong in correlation on their own therefore it is possible to merge them into more meaningful feature for the models. **image** We made a logarithm transformation at *Fare* values to normalise the skewed distribution for a better computation of the models. **Image** We split the dataset as we mentioned afore on California dataset but this time the Test set will hold the last 140 datapoints. Afterwards, we split the data successively into train and validation from the remaining first Train set.

3.2. Methodology

Logistic Regression Model

Logistic Regression model work very well on Titanic Survival Prediction because the coefficients can make a good probability estimation on *Survivals* target, and it is good with binary classification problem because it involves linear combination. The logistic function applies to a dataset as sigmoid mapping a linear combination from 0 to 1. It was the choice for its good interpretation on binary tasks.

The logistic regression model can be represented by the following equation:

$$p(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n)}} \tag{6}$$

where $p(y = 1|\mathbf{x})$ is the probability that the dependent variable y is 1 given the predictors $\mathbf{x}$, β_0 is the intercept, and β_i are the coefficients for the predictor variables x_i.

Decision Tree Model

A decision tree consists of decision nodes and leaves. Each node represents a decision rule, and each leaf represents an outcome. Here's an example of how to describe a decision tree's rules:

- If $x_1 \leq 0.5$, then:
  - If $x_2 > 1.0$, classify as Class A.
  - Otherwise, classify as Class B.
- If $x_1 > 0.5$, then:
  - If $x_3 \leq 0.3$, classify as Class C.
  - Otherwise, classify as Class D.

This structure can be extended to include more complex decision rules and deeper tree levels.

Random Forest Model

The Random Forest model aggregates predictions from multiple decision trees to improve accuracy and control over-fitting. The prediction for a regression problem by a Random Forest is typically the average of the predictions from all trees in the forest:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^{T} \hat{y}_t(x) \tag{7}$$

where $\hat{y}$ is the predicted value from the Random Forest, T is the total number of trees in the forest, and $\hat{y}_t(x)$ is the prediction of the t-th tree.

For a classification problem, the final class prediction is usually the mode of the classes predicted by individual trees:

$$\hat{y} = \text{mode}(\hat{y}_1(x), \hat{y}_2(x), \ldots, \hat{y}_T(x)) \tag{8}$$

where each $\hat{y}_t(x)$ represents the class predicted by the t-th tree.

3.3. Experiments

3.3.1. Experimental Settings

Although Logistic Regression proved to be the best performing model during your first attempts, further investigation into Decision Trees and Random Forests through hyperparameter tuning or pruning could be more sophisticated than what Logistic Regression is able to capture. This could be more sophisticated than what Logistic Regression is capable of capturing. Additionally, the application of an extensive set of metrics and analysis techniques such as a cost-sensitive Confusion Matrix and cost-sensitive learning may better take into account the real-world consequences of each model, particularly when there are imbalanced classes or asymmetric costs of misclassification.

Accuracy of a Classification Model

The accuracy of a classification model can be defined as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{9}$$

or more formally:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

where:

- TP (True Positives) are the correctly predicted positive cases,
- TN (True Negatives) are the correctly predicted negative cases,
- FP (False Positives) are the incorrectly predicted as positive,
- FN (False Negatives) are the incorrectly predicted as negative.

This metric is widely used to assess the effectiveness of classification algorithms, particularly in binary classification problems.

Precision in Classification Models

The precision of a classification model is defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{11}$$

where:

- TP (True Positives) are the correctly predicted positive cases,
- FP (False Positives) are the positive cases incorrectly predicted as positive.

This measure is particularly important in situations where the cost of a false positive is high, such as in medical diagnostics or spam detection.

Recall in Classification Models

The recall of a classification model, also known as sensitivity, is defined as:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{12}$$

where:

- TP (True Positives) are the cases correctly identified as positive,
- FN (False Negatives) are the positive cases that were incorrectly identified as negative.

Recall is particularly crucial in scenarios where missing a positive instance carries a heavy penalty, such as in disease screening or fraud detection.

AUC-ROC Curve

The AUC-ROC curve is a graphical representation used to evaluate the performance of a binary classifier system as its discrimination threshold is varied. The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR) at various threshold levels:

- **True Positive Rate (TPR)**: Also known as recall, it is calculated as $\frac{TP}{TP+FN}$.
- **False Positive Rate (FPR)**: Calculated as $\frac{FP}{TN+FP}$, representing the probability of falsely classifying a negative case as positive.

The AUC (Area Under the Curve) provides a single measure of overall performance of the classifier. An AUC of 1.0 represents a perfect classifier, while an AUC of 0.5 represents a worthless classifier.

Significance of the AUC-ROC Curve

- A higher AUC value is indicative of a better performing model.
- It is particularly useful for evaluating models where there is an imbalance in the observation classes.

To plot an AUC-ROC curve, the following tools or software can be used:

- Python libraries such as Matplotlib and Scikit-learn.
- R statistical software.
- Specific software for statistical analysis and data science.

3.3.2. RESULTS

Logistic Regression: Best parameters: {'C': 10, 'penalty': 11, 'solver': liblinear} Best cross-validation score: 0.793 Accuracy: 0.842 Confusion Matrix: [[76 9] [13 42]] Precision: 0.85 Recall: 0.89 F1 score: 0.87

Decision Tree: Best parameters: {'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 4, 'min_samples_split': 10} Best cross-validation score: 0.791 Accuracy: 0.8 Confusion Matrix: [[75 10] [18 37]] Precision: 0.81 Recall: 0.88 F1 score: 0.84

Random Forest: Best parameters: {'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 1, 'min_samples_split': 2} Best cross-validation score: 0.813 Accuracy: 0.821 Confusion Matrix: [[79 6] [19 36]] Precision: 0.81 Recall: 0.93 F1 score: 0.86

3.3.3. DISCUSSION

Logistic regression is helpful for identifying which factors influenced one's chances of survival the most because it provides a good explanation of the impact of each feature on the probability of survival. It is a linear model so it is computationally less difficult than other models, which is beneficial when understanding and time are of the essence.

Just like Logistic Regression, Decision Trees are easy to interpret. Their high-level structure makes for an easy explanation of how they arrive at their decisions, which is ideal when sharing outputs with those who may not be data-savvy. Decision Trees are highly prone to overfitting, particularly with rich and complex datasets. In the absence of appropriate adjustments and reductions, they can lead to excessively intricate trees that suffer from negative transfer, performing poorly outside of the training dataset.

Random Forests do not tend to overfit as much in comparison to a single Decision Tree. This is because they outperform single Decision Trees when it comes to generalization as well as robustness across various datasets. A lot more sophisticated and resource-heavy than Logistic Regression. This can be a con in situations where the speed of prediction is essential.

4. Conclusion

In this study, we encountered Regression and Classification performance with two distinct datasets. We learned how to analyze and interpret the data step by step while forming graphs that provided a more informative view. In addition, we learned how to engage with the dataset and perform some engineering manipulations with the aim of obtaining the best possible outcome to supply the models. Besides, working with different Machine Learning models gave us greater insights of their usefulness highlighting the need of having efficient work starting from the raw datasets and their results as well as self-discipline to ensure the code is clean and organized.