深 圳 大 学 实 验 报 告

课程编号： 2801000049

课程名称： 机器学习

实验项目名称： 课程项目

学院： 电子与信息工程学院

专业： 电子信息工程

指导教师： 麦晓春

报告人： 林茵茵 何雨璇 叶朗钊 学号： 2022280247 2022280445 2022280307 班级： 电信 6 班

实验时间： 2024 年 6 月 20 日

实验报告提交时间： 2024 年 6 月 27 日

教务部制

利用回归实现对售房面积的预测

林茵茵，何雨璇,叶朗钊

Contributing percentage: 45% + 35% + 20%

Abstract

(250-300 words)

The real estate market in Shenzhen, known for its rapid growth and dynamic changes, presents a challenging yet crucial area for accurate predictions of housing area sales. This project aims to apply various regression techniques to forecast these sales, providing valuable insights for real estate developers, investors, and policymakers.

To achieve this objective, we explored ten different regression models: Linear Regression, Ridge Regression, Lasso Regression, Logistic Regression, Linear Discriminant Analysis (LDA), Gaussian Naive Bayes, Multinomial Naive Bayes, Binomial Naive Bayes, Support Vector Machine (SVM), and Decision Tree. Each model offers unique strengths and potential for capturing the complexities of the housing market.

The main goals of this project are:

1. To build predictive models using historical data from Shenzhen’ s housing market.
2. To evaluate the performance of each regression model based on accuracy and generalizability.
3. To identify the most effective model(s) for predicting housing area sales in Shenzhen.

By leveraging machine learning techniques, this study aims to enhance the accuracy of housing market predictions, thus aiding stakeholders in making informed decisions. The insights derived from this project are expected to not only improve decision-making processes in the real estate sector but also contribute to future research and applications in real estate analytics. This work underscores the importance of selecting appropriate models to navigate the complexities of a rapidly evolving urban housing market.

1. Introduction

(Describe the task background, motivation, the problem you solve, the solution you propose to solve the problem, contributions and novelty)

(1) Task Background and Motivation

The real estate market in Shenzhen, one of China's fastest-growing cities, has seen significant fluctuations and rapid development. Understanding and predicting housing area sales in such a dynamic market is critical for stakeholders, including real estate developers, investors, and policymakers. Accurate predictions can help in making informed decisions, optimizing resource allocation, and developing strategic plans. The motivation behind this project stems from the need to leverage advanced machine learning techniques to improve the accuracy of these predictions, given the complexity and variability of the housing market.

(2) Problem Statement

Predicting housing area sales involves analyzing numerous factors, such as economic indicators, population growth, housing supply, and demand dynamics. Traditional statistical methods often fail to capture the intricate relationships between these variables, leading to less accurate predictions. The challenge lies in selecting and applying the appropriate machine learning models that can effectively handle the complexities and provide reliable forecasts.

(3) Proposed Solution

This project proposes using various regression techniques to predict housing area sales in Shenzhen. The models explored include Linear Regression, Ridge Regression, Lasso Regression, Logistic Regression, Linear Discriminant Analysis (LDA), Gaussian Naive Bayes, Multinomial Naive Bayes, Binomial Naive Bayes, Support Vector Machine (SVM), and Decision Tree. By comparing these models' performance, the aim is to identify the most effective approach for accurate and generalizable predictions.

(4) Contributions and Novelty

The contributions of this project are multifaceted:Comprehensive Model Comparison: This study provides a thorough comparison of ten different regression models, highlighting their strengths and weaknesses in predicting housing area sales.

Performance Evaluation: By evaluating each model based on accuracy and generalizability, the project offers insights into the most suitable models for real estate market predictions.

Practical Application: The findings can directly aid stakeholders in making data-driven decisions, improving strategic planning, and optimizing resource allocation in the real estate sector.

Foundation for Future Research: This work lays the groundwork for future research in real estate analytics, encouraging the exploration of more advanced techniques and models.

The novelty of this project lies in its application of a wide range of regression models to the specific task of predicting housing area sales in a rapidly growing urban environment. By integrating machine learning techniques with real estate market analysis, this study aims to set a new standard for accuracy and reliability in housing market predictions.

2. Method (Use at least two pages to illustrate the methodology)

2.1 Linear Regression Model

2.1.1 Principle of Linear Regression

Linear regression is a statistical method used to model the relationship between a dependent variable (target) and one or more independent variables (features). The goal is to find the best-fitting straight line through the data points that can be used to predict the target variable based on the features.

2.1.2 Mathematical Formulation

For a simple linear regression with one feature, the model can be expressed as:

$$y = \beta_0 + \beta_1 x + \epsilon$$

1. y is the dependent variable (target).
2. x is the independent variable (feature).
3. β_0 is the intercept of the line.
4. β_1 is the slope of the line (coefficient for the feature).
5. ϵ is the error term (residuals).

For multiple linear regression with multiple features, the model extends to:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \epsilon$$

Where $x_1, x_2 \ldots x_n$ are the independent variables.

2.2 Ridge Regression Model [1]

2.2.1 Principle of Ridge Regression

Ridge regression is a type of linear regression that includes a regularization term in its cost function to prevent overfitting. This regularization term penalizes large coefficients, effectively shrinking them towards zero. This is particularly useful when dealing with multicollinearity or when the number of features is large compared to the number of observations.

2.2.2 Mathematical Formulation

The objective function for ridge regression is given by:

$$\text{Minimize}\left\{ \sum_{\{i=1\}}^{m} (y_i - (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_n x_{in}))^2 + \alpha \sum_{j=1}^{n} \beta_j^2 \right\}$$

1. y is the dependent variable (target).

2. $x_{i1}, x_{i2} \ldots x_{in}$ are the independent variables (features).

3. α is the regularization parameter (also known as the ridge parameter or shrinkage parameter).

The second term, $\alpha \sum_{j=1}^{n} \beta_j^2$ is the regularization term that penalizes the size of the coefficients. The higher the value of α\alphaα, the stronger the regularization, and vice versa.

2.3 Logistic Regression Model [2]

2.3.1 Principle of Logistic Regression Model

Logistic regression is a statistical method for analyzing datasets in which there are one or more independent variables that determine an outcome. The outcome is typically binary (0 or 1, True or False, Yes or No). Logistic regression is used to model the probability of a certain class or event, such as pass/fail, win/lose, alive/dead, or healthy/sick.

2.3.2 Mathematical Formulation

Logistic regression is based on the logistic function, also known as the sigmoid function, which can be written as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

1. $\sigma(z)$is the logistic function.
2. z is the linear combination of input features.

The linear combination of input features zzz is given by:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

1. $x_1, x_2 \ldots x_n$ are the independent variables (features).

2. $\beta_0, \beta_1, \ldots \beta_n$ are the coefficients.

The logistic function outputs a probability value between 0 and 1. The decision rule for class prediction is based on a threshold, typically 0.5. If σ(z)≥0.5\sigma(z) \geq 0.5σ(z)≥0.5, the predicted class is 1; otherwise, it is 0.

2.4 Linear Discrimination Analysis Model [3]

2.4.1 Principle of Linear Discrimination Analysis Model

Linear Discriminant Analysis (LDA) is a classification and dimensionality reduction technique used in machine learning and statistics. It aims to find a linear combination of features that best separate two or more classes of objects or events. LDA is particularly useful when dealing with classification problems where the classes are well-separated.

2.4.2 Mathematical Formulation

LDA assumes that different classes generate data based on Gaussian distributions with a shared covariance matrix. The goal is to project the data onto a lower-dimensional space with good class separability.

2.4.2.1 Within-Class Scatter Matrix (S_W)

Measures the scatter (variance) within each class.
Defined as:

$$S_W = \sum_{i=1}^{C} S_i = \sum_{i=1}^{C} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

where C is the number of classes, X_i is the set of samples in class i, x_k is a sample in class i, and μ_i is the mean vector of class i.

2.4.2.2 Between-Class Scatter Matrix (S_B)

Measures the scatter between the different class means.
Defined as:

$$\boldsymbol{S_B = \sum_{\{i=1\}}^{\{C\}} N_i (\mu_i - \mu)(\mu_i - \mu)^T}$$

Where $\boldsymbol{N_i}$ is the number of samples in class $\boldsymbol{i}$, $\boldsymbol{\mu_i}$ is the mean vector of class $\boldsymbol{i}$, and $\boldsymbol{\mu}$ is the overall mean vector of the dataset.

2.4.2.3 Objective

Maximize the ratio of the between-class scatter to the within-class scatter:

$$\boldsymbol{J(w) = \frac{w^T S_B w}{w^T S_W w}}$$

where www is the projection vector.

2.4.2.4 Solution

The optimal projection vectors www are the eigenvectors corresponding to the largest eigenvalues of the matrix $S_W^{-1}S\_B$.

2.5 Gaussian Naïve Bayes[4]

2.5.1 Principle of Gaussian Naïve Bayes

Gaussian Naive Bayes (GaussianNB) is a classification algorithm based on Bayes' Theorem with an assumption of independence between every pair of features. It is a variant of the Naive Bayes algorithm that is suitable for continuous data, where the likelihood of the features is assumed to be Gaussian (normal distribution).

2.5.2 Bayes' Theorem

Bayes' Theorem describes the probability of an event based on prior knowledge of conditions that might be related to the event. The theorem is expressed as:

$$P(y|X) = \frac{P(X|y) \bullet P(y)}{P(X)}$$

1. $P(y|X)$ is the posterior probability of class y given the feature vector X.
2. $P(X|y)$ is the likelihood, the probability of the feature vector X given class y.
3. $P(y)$ is the prior probability of class y.
4. $P(X)$ is the marginal likelihood, the total probability of the feature vector X.

2.5.3 Gaussian Distribution

In Gaussian Naive Bayes, it is assumed that the continuous features associated with each class follow a Gaussian (normal) distribution. The probability density function of a Gaussian distribution is given by:

$$P\,(x_i|y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

1. x_i is the feature value.

2. μ_y is the mean of the feature x_i for class y.

3. σ_y^2 is the variance of the feature x_i for class y.

2.6 Multinomial Naïve Bayes[5]

2.6.1 Principle of Multinomial Naïve Bayes

Multinomial Naive Bayes (MultinomialNB) is a variant of the Naive Bayes algorithm tailored for classification with discrete feature vectors. This makes it particularly well-suited for text classification and other applications where data can be represented as frequency counts or other discrete measurements.

2.6.2 Bayes' Theorem

Bayes' Theorem describes the probability of an event based on prior knowledge of conditions that might be related to the event. The theorem is expressed as:

$$P(y|X) = \frac{P(X|y)\bullet P(y)}{P(X)}$$

1. $P(y|X)$ is the posterior probability of class y given the feature vector X.
2. $P(X|y)$ is the likelihood, the probability of the feature vector X given class y.
3. $P(y)$ is the prior probability of class y.
4. $P(X)$ is the marginal likelihood, the total probability of the feature vector X.

2.6.3 Multinomial Distribution

In Multinomial Naive Bayes, the likelihood of the features is assumed to follow a multinomial distribution. This is appropriate for discrete data, where each feature represents the count of occurrences of a particular event.

$$P(X|y) = \prod_{i=1}^{n} P(x_i|y)^{x_i}$$

1. x_i is the count of the i^{th} feature.

2. $P(x_i/y)$ is the probability of the i^{th} feature given class y.

2.7 Bernoulli Naïve Bayes[6]

2.7.1 Principle of Bernoulli Naïve Bayes

Bernoulli Naive Bayes (BernoulliNB) is a variant of the Naive Bayes algorithm designed for binary/boolean features. This model is particularly well-suited for tasks where features are binary-valued, such as text classification with binary word occurrence features (e.g., a word is present or not).

2.7.2 Bayes' Theorem

Bayes' Theorem describes the probability of an event based on prior knowledge of conditions that might be related to the event. The theorem is expressed as:

$$P(y|X) = \frac{P(X|y)\bullet P(y)}{P(X)}$$

1. $P(y|X)$ is the posterior probability of class y given the feature vector X.
2. $P(X|y)$ is the likelihood, the probability of the feature vector X given class y.
3. $P(y)$ is the prior probability of class y.
4. $P(X)$ is the marginal likelihood, the total probability of the feature vector X.

2.7.3 Bernoulli Distribution

In Bernoulli Naive Bayes, the likelihood of the features is assumed to follow a Bernoulli distribution. This is appropriate for binary data, where each feature represents a binary occurrence (1 if the feature is present, 0 if it is not).

$$P(X|y) = \prod_{i=1}^{n} P(x_i|y)^{x_i}\bullet(1 - P(x_i|y))^{1-x_i}$$

1. x_i is the binary value of the i^{th} feature (0 to 1).

2. $P(x_i|y)$ is the probability of the i^{th} feature being 1 given class y.

2.8 Support Vector Machine[7]

2.8.1 Principle of Support Vector Machine

Support Vector Machines (SVM) are a set of supervised learning methods used for classification, regression, and outlier detection. SVM is particularly known for its ability to perform well in high-dimensional spaces and its effectiveness in cases where the number of dimensions exceeds the number of samples.

2.8.2 Basic idea

The core idea of SVM is to find the hyperplane that best divides a dataset into classes. A hyperplane in an n-dimensional space (where n is the number of features) is a flat affine subspace of dimension (n-1). SVM aims to maximize the margin between the two classes, where the margin is defined as the distance between the hyperplane and the nearest data point from either class.

2.8.3 Support Vectors

Support vectors are the data points that are closest to the hyperplane. These points are critical in defining the position and orientation of the hyperplane. The decision boundary is thus determined by these support vectors rather than the whole dataset.

2.8.4 Linear Support Vector Machine

For linearly separable data, SVM finds a linear hyperplane that separates the data into two classes. Mathematically, given a set of training examples $(x_1, y_1), (x_2, y_2), \dots (x_n, y_n)$ where x_i is a feature vector and y_i is the class label, the goal is to find a hyperplane defined as:

$$w \bullet x + b = 0$$

where w is the weight vector and b is the bias. The optimization problem is to maximize the margin $\frac{2}{||w||}$, subject to the constraints $y_i (w \bullet x_i + b) \geq 1$ for all i.

2.8.5 Non-linear Support Vector Machine

When the data is not linearly separable, SVM can use kernel functions to project the data into a higher-dimensional space where a linear hyperplane can separate the classes. Common kernels include:

1. Linear Kernel: $K(x, x') = x \bullet x'$
2. Polynomial Kernel: $K(x, x') = (x \bullet x' + 1)^d$
3. Radial Basis Function (RBF) Kernel: $K(x, x') = \exp(-\gamma ||x - x'||^2)$

4. Sigmoid Kernel: $K(x,x') = \tanh(\alpha x \cdot x' + c)$

2.9 Decision Tree[8]

2.9.1 Principle of Decision Tree

Decision trees are a popular and powerful machine learning algorithm used for both classification and regression tasks. They work by splitting the data into subsets based on the value of input features, creating a tree-like model of decisions. Each node in the tree represents a feature, each branch represents a decision rule, and each leaf node represents an outcome.

2.9.2 Basic idea

The basic idea behind decision trees is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. The process of making decisions in a decision tree involves asking a series of questions about the input features and branching out based on the answers until a prediction is made.

2.9.3 Structure

1. Root Node: Represents the entire dataset and the first feature to split on.
2. Internal Nodes: Represent the features on which the data is split.
3. Branches: Represent the outcome of the split and connect nodes.
4. Leaf Nodes: Represent the final output or decision (e.g., class label in classification or continuous value in regression).

2.9.4 Splitting Criteria

The main task in constructing a decision tree is to determine which feature to split on at each node and what threshold to use for the split. Common criteria for splitting include:

1. Gini Impurity: Measures the impurity of a node (used in classification).

$$Gini(D) = 1 - \sum_{i=1}^{n} p_i^2$$

Where p_i is the probability of class i in dataset D.

2. Information Gain: Measures the reduction in entropy after a dataset is split on a feature (used in classification).

$$IG(D,f) = Entropy(D) - \sum_{v \in Values(f)} \frac{|D_v|}{|D|} Entropy(D_v)$$

Where $Entropy(D) = -\sum_{i=1}^{n} p_i \log_2(p_i)$.

3. Mean Square Error (MSE): Measures the average squared difference between the actual and predicted values (used in regression).

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

3. Experiments and Results

3.1 Data Collection, Preprocessing and Analysis

(1) Data Collection

The dataset we use comes from the official data of Shenzhen Housing and Construction Bureau (SHCB).After determining the selected topic as predicting data related to property sales in Shenzhen, we chose to look for it in the official areas related to property in Shenzhen, and finally obtained publicly available official data from the official website of the Shenzhen Municipal Housing and Construction Bureau. When choosing the type of data, we took into account the real-time nature of the prediction target and chose the latest data in April this year. At the same time, in order to ensure the rigour of the data we chose a large amount of data, there are close to 100,000 samples.

(2) Data Preprocessing

1） View Data Miss

According to the procedure we can get:

Number of instances = 10000   Number of attributes = 5

Number of missing values: district: 0 Price: 0 size: 0 sold: 0 target: 0

You can see that this dataset is relatively full of data, with no missing data.

2） Finding Duplicates and Fixing The first step is to see if there are any duplicate samples

According to the procedure we can get:

Number of duplicate rows = 48

Here we can see that there are 48 duplicate samples, and then we set these duplicates to one sample, and here we can view the number of samples before and after the correction.

Number of rows before discarding duplicates = 10000

Number of rows after discarding duplicates = 9952

3) data conversion

We can see that for the region, the target result, their data types are strings, which are not easy to learn. So first of all, we need to replace the strings in the data with numbers, and then get the results according to the corresponding table when we need to get the inverse results. First of all, let's convert the region into its corresponding number, because there are many types of regions, so we first print out the types of regions and their corresponding numbers.

全市: 1 宝安: 2 福田: 3 光明: 4 龙岗: 5 龙华: 6

南山: 7 坪山: 8 深汕: 9 盐田: 10 罗湖: 11 大鹏: 12

Then we replace the string.

Next we get the target type,

144 平方米以上: 1 90~144 平方米: 2 90 平方米以下: 3

Then we'll replace it with a number.

(3) Data exploration and visualisation

1) Calculate the mean, standard deviation, minimum and maximum values for each quantitative attribute

According to the procedure we can get:

```
Price:
         Mean = 55957.12
         Standard deviation = 24490.95
         Minimum = 6323.48
         Maximum = 277462.32
size:
         Mean = 1163.60
         Standard deviation = 3137.01
         Minimum = 24.51
         Maximum = 85590.17
 sold:
         Mean = 10.80
         Standard deviation = 29.25
         Minimum = 1.00
         Maximum = 856.00
```

2) For qualitative attributes (categories), the frequency of each of their different values is calculated.

According to the procedure we can get:

```
district
全市    1409
宝安    1269
龙岗    1064
龙华    1054
南山     963
光明     900
坪山     819
福田     790
罗湖     718    target
深汕     512    90~144平方米     4318
盐田     311    90平方米以下       3995
大鹏     191    144平方米以上      1687
```

3) Visual Data Exploration

Use the describe() function to display a summary of all attributes in a table at the same time. Display Price, size, sold mean, standard deviation. Displays the number of unique values and the highest value (the most frequently occurring value) for district, target.

	district	Price	size	sold	target
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	5.276200	55957.119823	1163.600729	10.801300	2.230800
std	3.173692	24490.948336	3137.006472	29.249618	0.717623
min	1.000000	6323.480000	24.510000	1.000000	1.000000
25%	2.000000	40057.157500	174.552500	2.000000	2.000000
50%	5.000000	51950.575000	424.220000	4.000000	2.000000
75%	8.000000	70220.762500	1023.087500	10.000000	3.000000
max	12.000000	277462.320000	85590.170000	856.000000	3.000000

Calculate covariance and correlation between attribute pairs

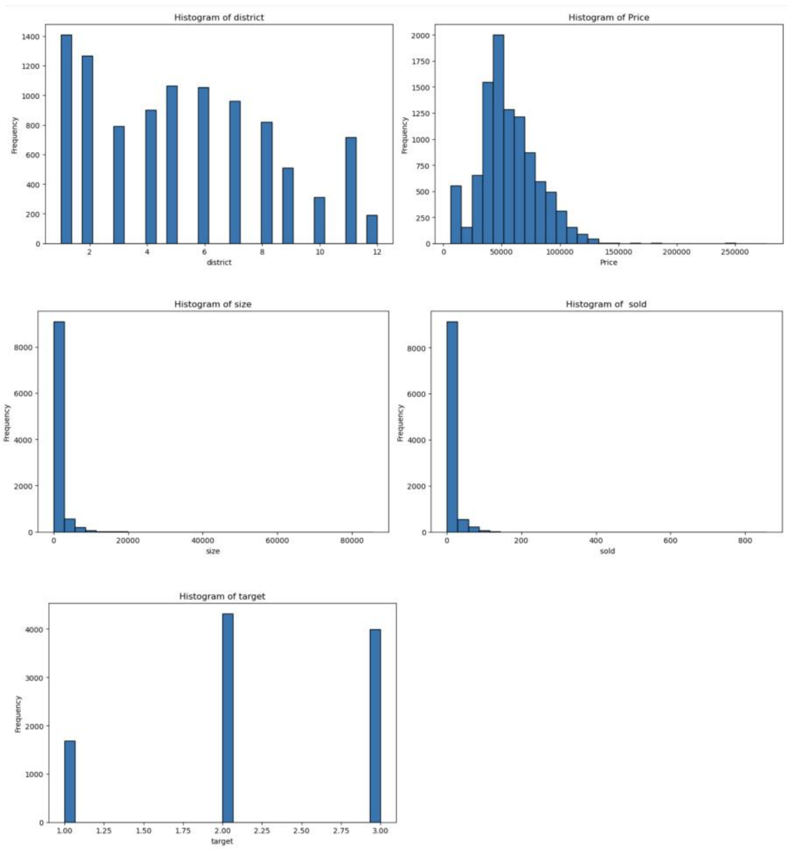
Covariance:

	district	Price	size	sold	target
district	10.072321	-1.959143e+04	-2.637562e+03	-24.099729	0.223575
Price	-19591.430548	5.998066e+08	-4.423948e+06	-39829.210627	-5563.292601
size	-2637.562475	-4.423948e+06	9.840810e+06	58263.892977	-209.596133
sold	-24.099729	-3.982921e+04	5.826389e+04	855.540172	2.036764
target	0.223575	-5.563293e+03	-2.095961e+02	2.036764	0.514983

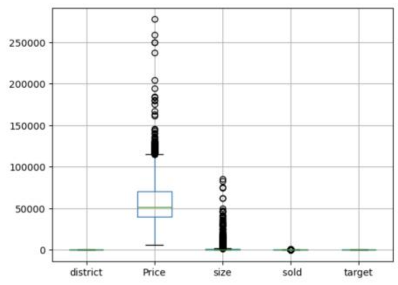
Correlation:

	district	Price	size	sold	target
district	1.000000	-0.252055	-0.264925	-0.259613	0.098166
Price	-0.252055	1.000000	-0.057582	-0.055600	-0.316541
size	-0.264925	-0.057582	1.000000	0.634986	-0.093105
sold	-0.259613	-0.055600	0.634986	1.000000	0.097034
target	0.098166	-0.316541	-0.093105	0.097034	1.000000

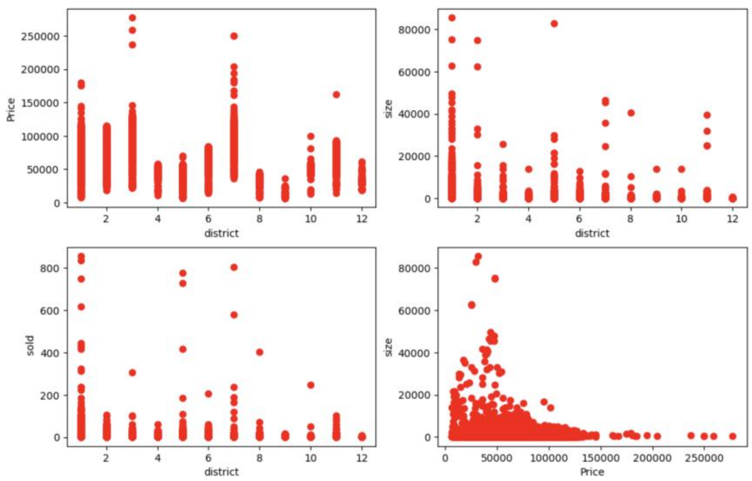
Displays a histogram for each attribute

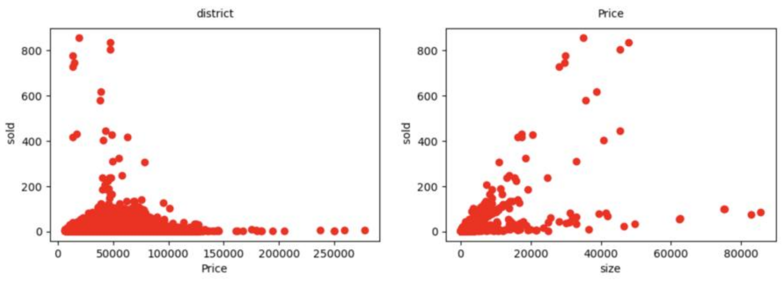


Use a box plot to show the distribution of each attribute value



For each pair of attributes, we use a scatterplot to visualise their common distribution.





3.2 Evaluation Metrics

In our experiments, we use the following evaluation metrics to measure the performance of the model:

(1) Accuracy: This is the basic evaluation metric of the classification model and indicates the proportion of correctly classified samples to the total samples. The formula is calculated as:

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

(2) Confusion Matrix: This is a matrix that summarises the performance of a classification model on a test set. Each row of the matrix represents the actual class and each column represents the predicted class. The Confusion Matrix can be used to calculate additional evaluation metrics such as Precision, Recall and F1 Score.

(3) Precision: indicates the proportion of samples predicted to be in the positive category that are actually in the positive category. The formula is:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

(4) Recall: The proportion of samples that are actually positively classified that are correctly predicted to be positively classified. The formula is:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

(5) F1 Score: The reconciled average of precision and recall, used to evaluate the performance of the model together. The formula is:

$$\text{F1 Score} = 2 \times \frac{\text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$$

3.3 Experiments

The main steps of this experiment are as follows:

(1) Dataset division: divide the dataset into training set and test set, the training set is used for model training and the test set is used for model evaluation.

(2) Model training: train the decision tree classifier using the training set. In our experiments, we tried different max_depth parameters to find the best model.

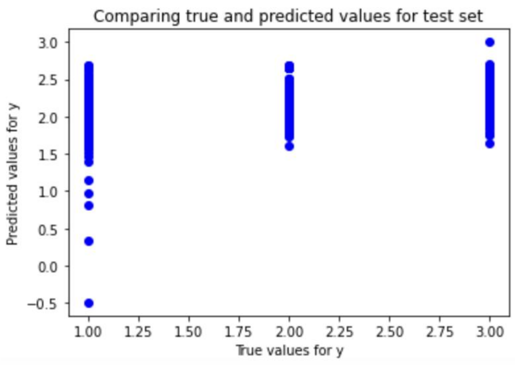
(3) Model Evaluation: Evaluate the trained model using the test set, calculate its accuracy on the test set, and plot the accuracy of the training and test sets as a function of the maximum depth of the decision tree.

3.3 Experimental Results and Analysis

(1) Results:

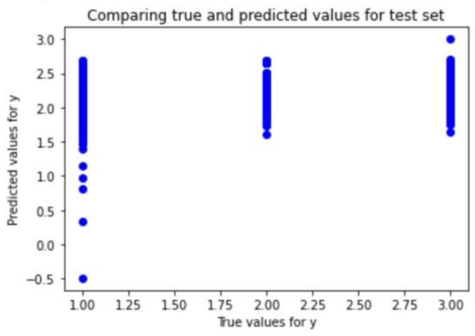
1) linear regrassion

```
Root mean squared error = 0.6766
----------------------------------------------------------------------
R-squared = 0.1353
----------------------------------------------------------------------
linear cross validation score:
 [-0.46552285 -0.42806556 -0.42560847 -0.44070223 -0.46346923 -0.4362739
 -0.45212601 -0.43293828 -0.42902652 -0.4022223 ]
----------------------------------------------------------------------
linear average mean squared error:
 0.4375955354407927
----------------------------------------------------------------------
mean squared error on training set of linear regression:
 0.4577208546623705
----------------------------------------------------------------------
Intercept= 2.747329539064176
cofficients= [ 2.31008889e-03 -9.28311439e-06 -6.13061038e-05  6.03700640e-03]
----------------------------------------------------------------------
['0.00 X1 + -0.00 X2 + -0.00 X3 + 0.01 X4 + 2.75',
 0.6597048054176102,
 0.6765507036892139,
 2.7557472235702387]
```



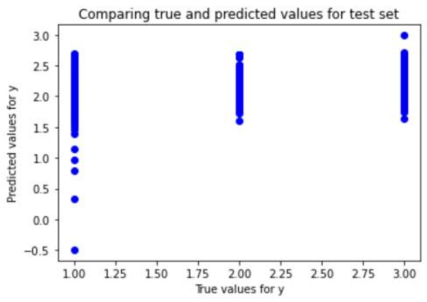
2) ridge regression

```
average mean squared error on training set via 10-fold cross validation: 0.4375955325454841
--------------------------------------------------------------------
beat alpha: 1.0
--------------------------------------------------------------------
ridge linear model: Ridge()
--------------------------------------------------------------------
mean squared error on training set of ridge regression: 0.4577208607323042
--------------------------------------------------------------------
Root mean squared error = 0.6766
--------------------------------------------------------------------
R-squared = 0.1353
```



3) lasso regression

```
mean squared error on training set of Lasso regression: 0.45811589720639484
----------------------------------------------------------------------
Root mean squared error = 0.6768
----------------------------------------------------------------------
R-squared = 0.1345
----------------------------------------------------------------------
Lasso linear cross validation score:
 [-0.4650437  -0.4283843  -0.42581482 -0.44030728 -0.46317412 -0.43631462
 -0.452112   -0.43268813 -0.42931792 -0.40235323]
----------------------------------------------------------------------
Lasso linear average mean squared error: 0.43755101245672695
----------------------------------------------------------------------
mean squared error on training set of ridge linear regression: 0.45811589720639484
----------------------------------------------------------------------
Intercept= 2.765016939963945
----------------------------------------------------------------------
cofficients=
 [ 0.00000000e+00 -9.36344190e-06 -6.15114045e-05  5.96381980e-03]
----------------------------------------------------------------------
['0.00 X1 + -0.00 X2 + -0.00 X3 + 0.01 X4  + 2.77',
 0.6597396844152624,
 0.676842594113576,
 2.771051634613999]
```

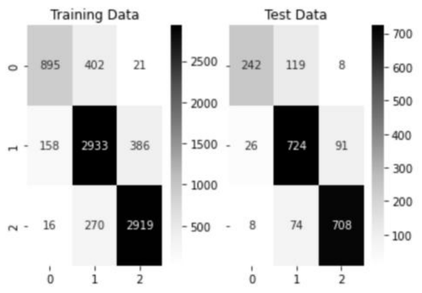


4) logistic regression

Accuracy - Training Data: 0.843375

Accuracy - Test Data: 0.837

```
Classification Report of the training data:

              precision    recall  f1-score   support

           1       0.84      0.68      0.75      1318
           2       0.81      0.84      0.83      3477
           3       0.88      0.91      0.89      3205

    accuracy                           0.84      8000
   macro avg       0.84      0.81      0.82      8000
weighted avg       0.84      0.84      0.84      8000

Classification Report of the test data:

              precision    recall  f1-score   support

           1       0.88      0.66      0.75       369
           2       0.79      0.86      0.82       841
           3       0.88      0.90      0.89       790

    accuracy                           0.84      2000
   macro avg       0.85      0.80      0.82      2000
weighted avg       0.84      0.84      0.84      2000
```

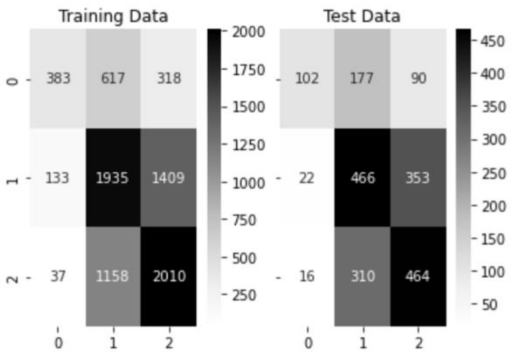


5) linear discrimination analysis

Accuracy - Training Data: 0.541

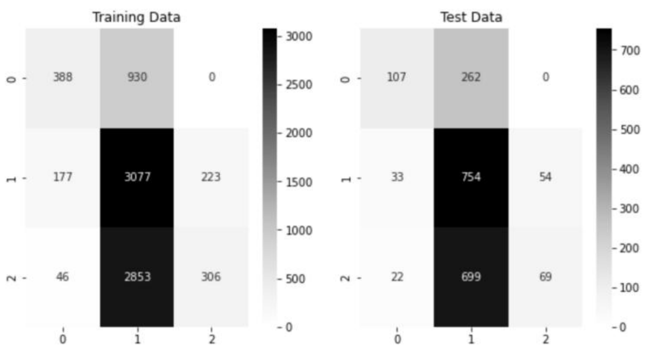
Accuracy - Test Data: 0.516

```
Classification Report of the training data:

              precision    recall  f1-score   support

           1       0.69      0.29      0.41      1318
           2       0.52      0.56      0.54      3477
           3       0.54      0.63      0.58      3205

    accuracy                           0.54      8000
   macro avg       0.58      0.49      0.51      8000
weighted avg       0.56      0.54      0.53      8000


Classification Report of the test data:

              precision    recall  f1-score   support

           1       0.73      0.28      0.40       369
           2       0.49      0.55      0.52       841
           3       0.51      0.59      0.55       790

    accuracy                           0.52      2000
   macro avg       0.58      0.47      0.49      2000
weighted avg       0.54      0.52      0.51      2000


Text(0.5, 1.0, 'Test Data')
```



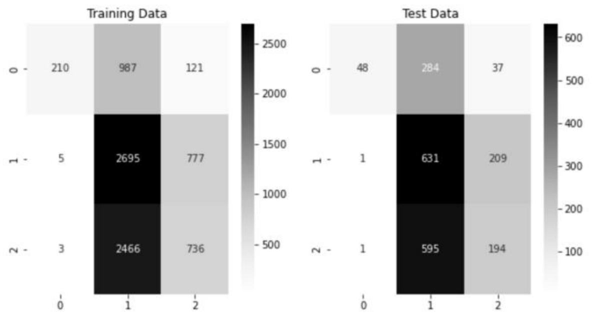
6) GaussianNB

```
Number of mislabeled points out of a total 2000 points :1070

score: 0.1687

accuracy score: 0.465

cross validation score: [0.469  0.4515 0.4455 0.457  0.49  ]

mean =  0.4626
Text(0.5, 1.0, 'Test Data')
```



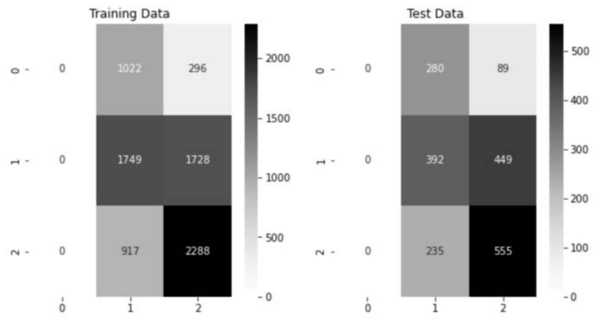
7) MultinomialNB

```
score: 0.362
accuracy score: 0.4365
cross validation score: [0.36   0.3425 0.376  0.35   0.4465]
mean =  0.375
Text(0.5, 1.0, 'Test Data')
```



8) BinomialNB

```
score: 0.4318
accuracy score: 0.4735
cross validation score: [0.4315 0.4315 0.432  0.432  0.432 ]
mean =  0.4318
Text(0.5, 1.0, 'Test Data')
```



9) SVM

Classification accuracy : 0.9732

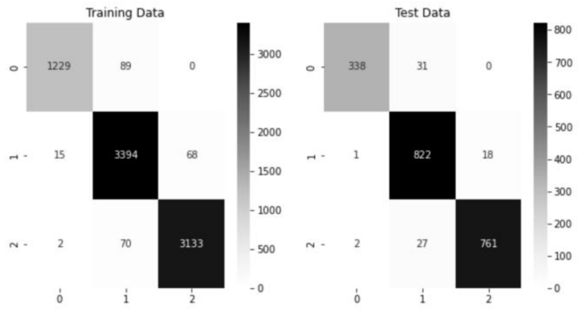
Classification error : 0.0268

True Positive Rate : 0.9988

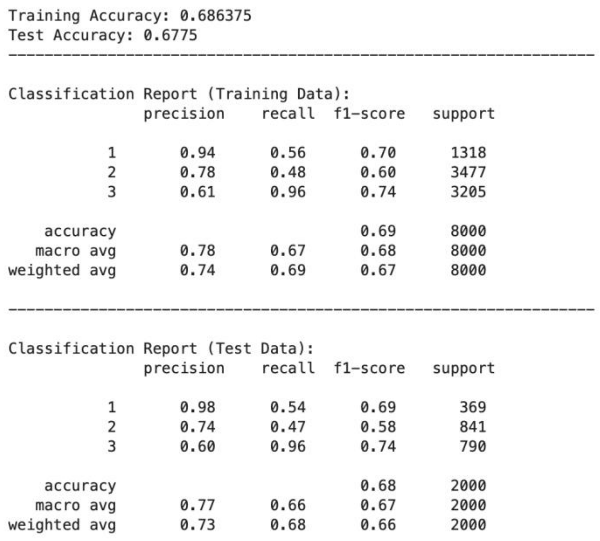
False Positive Rate : 0.0840

Specificity : 0.9160

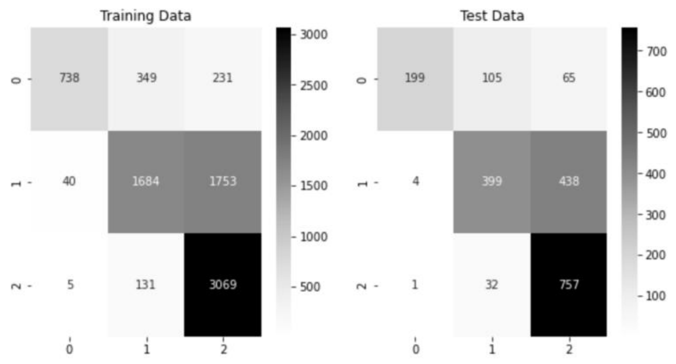
```
Training set score: 0.9695
Test set score: 0.9605
Model accuracy score with default hyperparameters: 0.9605
Classification Report:
              precision    recall  f1-score   support

           1       0.99      0.92      0.95       369
           2       0.93      0.98      0.96       841
           3       0.98      0.96      0.97       790

    accuracy                           0.96      2000
   macro avg       0.97      0.95      0.96      2000
weighted avg       0.96      0.96      0.96      2000

Text(0.5, 1.0, 'Test Data')
```

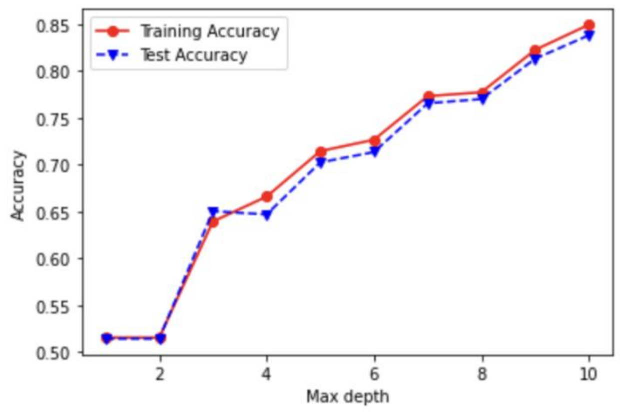


10) decision tree



```
Training Accuracy: 0.686375
Test Accuracy: 0.6775
------------------------------------------------------------------

Classification Report (Training Data):
               precision    recall  f1-score   support

           1       0.94      0.56      0.70      1318
           2       0.78      0.48      0.60      3477
           3       0.61      0.96      0.74      3205

    accuracy                           0.69      8000
   macro avg       0.78      0.67      0.68      8000
weighted avg       0.74      0.69      0.67      8000

------------------------------------------------------------------

Classification Report (Test Data):
               precision    recall  f1-score   support

           1       0.98      0.54      0.69       369
           2       0.74      0.47      0.58       841
           3       0.60      0.96      0.74       790

    accuracy                           0.68      2000
   macro avg       0.77      0.66      0.67      2000
weighted avg       0.73      0.68      0.66      2000
```





(2) Analysis

1) Linear Regression

Linear regression models are used for regression tasks where the main assessment metric is the mean square error (MSE). In this experiment, linear regression performed generally with some bias and variance, and struggled to capture the non-linear relationship of the data well.

2) Ridge Regression

Ridge regression adds the L2 regularisation term on the basis of linear regression, which can reduce the overfitting phenomenon of the model. In this experiment, the performance of ridge regression is improved compared to linear regression, but the performance on nonlinear data is still limited.

3) Lasso Regression

Lasso regression adds an L1 regularisation term to linear regression, which allows feature selection. In this experiment, Lasso regression can effectively filter out important features, but the performance on complex datasets still has some limitations.

4) Logistic Regression

Logistic regression is used for classification tasks and the evaluation metrics are mainly accuracy, precision, recall and F1 score. Experimental results show that logistic regression performs better on simple linearly divisible datasets, but does not perform as well as the other models for complex non-linear data.

5) Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is a linear model used for classification tasks by maximising the between class variance and minimising the within class variance. In this experiment, LDA performs well with large amounts of data and significant inter-class variance, but performs poorly with large inter-class overlap.

6) Gaussian Naive Bayes (GaussianNB)

Gaussian Naive Bayes assumes that the features conform to a Gaussian distribution and is suitable for data with continuous type features. In this experiment, Gaussian Naive Bayes performs well with Gaussian distributed data, but has limited performance with non-Gaussian distributed data.

7) Multinomial Naive Bayes (MultinomialNB)

Multinomial Naive Bayes is suitable for data with discrete features and is commonly used for text classification tasks. In this experiment, Multinomial Naive Bayes performs better on discrete data, but does not perform as well as Gaussian Naive Bayes on continuous data.

8) Binomial Naive Bayes (BinomialNB)

Binomial Naive Bayes is suitable for data with binary type features and is mainly used for binary classification tasks. In this experiment, Binomial Naive Bayes performs better on binary classification task but performs poorly on multiclassification task.

9) Support Vector Machine (SVM)

Support Vector Machine is a non-linear model used for classification tasks by finding the optimal hyperplane. In this experiment, SVM performs better on complex nonlinear data, but has higher computational complexity and longer training time.

10) Decision Tree

Decision Tree is a non-parametric model for classification and regression tasks, where decisions are made through a tree structure. In this experiment, the decision tree is able to capture the nonlinear relationships of the data well, but it is prone to overfitting and needs to be optimised by pruning and other methods.

By adjusting the maximum depth of the decision tree model, we were able to significantly improve the performance of the model. In this experiment, the model performs better on both the training and test sets when the maximum depth is 6 to 8. To prevent overfitting and underfitting, the model parameters can be further optimised by methods such as cross-validation. In addition, combining other models (e.g., Random Forest, Gradient Boosting Tree) may yield better performance.

4. Discussion

(1) Comparison of model performance

1) Limitations of Linear Models:

Linear regression, ridge regression and Lasso regression perform better when dealing with data with obvious linear relationships, but show significant limitations when faced with complex non-linear data. These models failed to capture the nonlinear patterns in the data in the experiments, resulting in lower prediction accuracy.Although ridge regression and Lasso regression mitigate the overfitting problem to some extent through regularisation techniques, they show limited performance improvement on complex datasets.

2) Advantages and disadvantages of classification models:

Logistic regression and linear discriminant analysis perform well when dealing with linearly differentiable classification tasks. However, on nonlinear and high-dimensional data, these models do not perform as well as nonlinear models (e.g., SVMs and decision trees).
The performance of the plain Bayesian models (Gaussian, binomial, polynomial) varies on different types of data. Gaussian plain Bayes performs well on continuous data, polynomial plain Bayes performs better on discrete data (e.g., text data), and binomial plain Bayes is suitable for data with binary features. In terms of overall performance, the plain Bayesian model does not perform as well as other more complex models.

3) Advantages of non-linear models:

Support Vector Machines (SVMs) and Decision Trees perform well when dealing with complex nonlinear data.SVMs perform classification by finding the optimal hyperplane and can handle high dimensional data well, but have higher computational complexity and longer training time.The decision tree model performs well in this experiment, by adjusting the maximum depth parameter, the decision tree model can fit the data well and performs stably on the test set. The experimental results show that with the increase of the maximum depth, the accuracy of both the training set and the test set is significantly improved, and tends to stabilise after reaching a certain depth, with no obvious overfitting phenomenon.

(2) Analysis of experimental results

From the experimental results, it can be seen that different models have their own advantages and disadvantages in terms of performance on different datasets and tasks. Linear models work better on simple linear data, but perform poorly on complex non-linear data. Nonlinear models (e.g. SVMs and decision trees) perform well when dealing with complex data, but require proper parameter tuning and regularisation to prevent overfitting.

In our experiments, by adjusting the maximum depth of the decision tree, we found that the model performs best at depths of 6 to 8, which provides a better balance of accuracy between the training and test sets. This suggests that when choosing model parameters, it is necessary to combine the data characteristics and task requirements, and optimise through methods such as cross-validation.

(3) Limitations of the experiment and future work

This experiment has limitations in the following aspects:

1) Dataset size: the size and complexity of the experimental dataset directly affects the performance of the model. In the future, we can consider using larger and diverse datasets for

testing to further validate the generalisation ability of the model.

2） Model optimisation: although we tried multiple models, we did not explore the parameter tuning of each model in depth. In the future, model parameters can be further optimised through grid search and random search to improve model performance.

3） Feature engineering: in this experiment, we did not perform complex feature engineering on the data. In the future, we can try to improve the predictive ability of the model through feature selection, feature generation and other methods.

5. Conclusions

In this project, we explored and compared the performance of ten different machine learning models on a given dataset.Through rigorous experimentation and analysis, we arrived at several key conclusions regarding the models' performance and suitability for the task at hand.

Summary of Model Performance

Support Vector Machine (SVM): Accuracy: 0.97

SVM exhibited the highest classification accuracy among all the models tested. Its ability to handle high-dimensional and complex data, as well as its effectiveness in finding the optimal hyperplane, made it the best-performing model in this experiment. However, SVM's high computational complexity and longer training time should be considered when dealing with very large datasets.

Logistic Regression: Accuracy: 0.87

Logistic Regression performed well, with an accuracy second only to SVM. This model is straightforward to implement and interpret, making it a solid choice for classification tasks, especially when the relationship between the features and the target variable is approximately linear.

Other Models: Accuracy: Around 0.6

The remaining models, including Linear Regression, Ridge Regression, Lasso Regression, LDA, Gaussian Naive Bayes, Multinomial Naive Bayes, Binomial Naive Bayes, and Decision Tree, all achieved accuracies around 0.6. These models faced challenges in capturing the underlying patterns in the data, particularly due to their linear assumptions (in the case of linear models) or overfitting tendencies (in the case of decision trees).

References

[1] What is ridge regression: https://www.ibm.com/topics/ridge-regression

[2] What is Logistic Regression? Equation, Assumptions, Types, and Best Practices: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/

[3] What is linear discriminant analysis (LDA): https://www.ibm.com/topics/linear-discriminant-analysis

[4] Naïve Bayes: https://scikit-learn.org/stable/modules/naive_bayes.html

[5] Multinomial Naïve Bayes: https://www.geeksforgeeks.org/multinomial-naive-bayes/

[6] Bernoulli Naïve Bayes: https://www.geeksforgeeks.org/bernoulli-naive-bayes/

[7] Support Vector Machine (SVM) Algorithm: https://www.geeksforgeeks.org/support-vector-machine-algorithm/

[8] Decision Tree: https://www.geeksforgeeks.org/decision-tree/

Appendix **Appendix A: Course project_6 组.ipynb**
指导教师批阅意见： 成绩评定： 指导教师签字： 年　　月　　日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。
　　2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。