Prediction of the pizza brands with different classification methods

Spring Semester, 2022

1. Introduction

As it is known to us, pizza is a kind of food, which consists of many food ingredients and is in the shape of the circle. In fact, pizza also attracts the appetite from many people, and there are many pizza restaurants all over the world, especially in the European countries like Finland.

Since it is a course project in the Machine Learning field, then it is expected to explore how the machine-learning methods could be applied to the realistic problems. Here want to predict the pizza brands based on the given dataset and information, with machine-learning methods. Moreover, this topic enables people to know the brand of pizza they want to buy, if they are provided with the information of pizza, like the size, the price or the type. Then people would know better about the pizza brands, which helps them to decide the brand they intend to purchase the next time.

Regarding the structure of the project report, the clarification on this machine learning problem is presented below in the Section 2 “Problem Formulation”, while the main methods for this project will be presented in the Section 3. After these sections, the Section 4 and Section 5 are expected to show the result and draw the conclusion of the project respectively.

2. Problem Formulation

In terms of the problem of project, we could start to assume that one consumer knows the size, the taste and the price of the pizza, but this person is unaware of the brand of this pizza. Then problem arises: What is the brand of the pizza?

This dataset comes from the online open-source platform Kaggle [1]. The data points of the problem are the record of one pizza, which includes the information of its original company name, the pizza name, the type, the size, as well as the price. The total number of the data numbers is 371. Since the information except the price from the dataset are not in the form of numbers, we could use numerical values to indicate the information. For example, since there are only four brands of the pizza (Domino, Pizza Hut, Godfather, IMO), the numbers from 1 to 4 are used to indicate the brand of pizza respectively. Therefore, it could be a multiclass classification problem, where there are 4 potential categories of outcomes.

2.1 Labels and Features of the data points

This part will introduce the labels and features of the adopted data points in this project briefly.

Label: The brand of the pizza.

Candidate Features: The type, the size, and the price of the pizza. (3 features)

3. Methods

3.1 Dataset

To get the data points of this project, I collect the data from the Kaggle platform [1], and each data point represent the indicators in one record for each kind of the pizza, which is also mentioned in the "Problem Formulation" section. There are 371 data points from 4 pizza companies in total, with 0 missing data in the dataset.

As for the candidate feature, I will consider three features including the type, the size, and the price of the pizza. Since the original value of these three values is in the format of text, I will convert the text into discrete integers. For example, there are several sizes of the pizza, and I will use the number 1 to indicate the size of "Mini", the number 2 to indicate the size of "Small", and the largest number 6 to the size of "Jumbo".

Similarly, as I select the brand of the pizza as my label, since there are 4 brands in total, I will use **the number 0 to 3** to indicate these 4 brands of pizza restaurants.

3.2 Feature Selection

Firstly, I will put the visualization of my data with scatterplots, which shows the relationship between the label and candidate features.

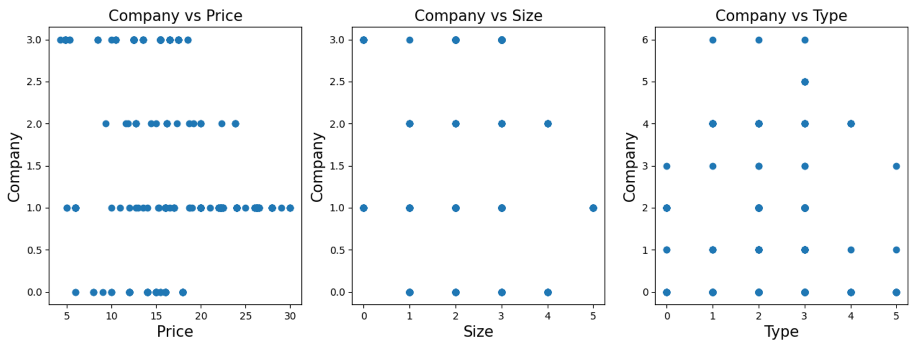

Figure 1. Data visualization

From Figure 1, we could notice that the distribution of the price of the pizza is most sparse, which means that it would be easier for the model to judge the brand of the pizza. For example, for the pizza price over 25 dollars, we could find that it belongs to the brand with the label '1'. Thus, among the candidate features, the "Price" would be adopted as the feature of the dataset.

3.3 The motivation of selected models

3.3.1 The first model of the project

Since this is a multi-class problem, the classification models from the method Support Vector Machine (SVM) could be applied. It is known that the Logistic Regression methods focus more on the classification from 2 classes, and the Scikit Learn library [2] has provided classes including the SVC (Support Vector Classification) for the multi-class classification task, with the use of decision function. The decision function could help to the muti-class classification case [2]. Thus, in my project, the SVC classification method is adopted.

And since the SVC is just a class of SVM classification methods, it still uses basic linear maps $h(\boldsymbol{x}) = w^T \boldsymbol{x}$ [3], which helps to finish the task of classification.

3.4 Loss function

3.4.1 The first model of the project

Since I have selected models from the SVM method, then the **Hinge Loss** would be used, which is "L(($\boldsymbol{x}$, y), h) = max{0, 1-yh($\boldsymbol{x}$)}" [4]. And the Hinge Loss is chosen because this loss function is the inevitable basis in the SVM method [3], which means that the SVM uses the Hinge Loss with the regularized parameters [3]. And the Hinge loss would perform the task of learning the robust hypothesis [3].

3.5 Size of training and validation dataset

As mentioned in the Section 3.1, there are 371 data points of my dataset from the Kaggle [1]. In general, the data records are quite enough for me to process. Thus, I select to use the single split into training and validation set.

For the first method based on SVC, I select 75% of the entire data points as my training dataset, while the rest (25%) of the data points will become the validation dataset. Since allocating more data to the training dataset helps to fit the model, it is reasonable to select three quarters of the total data points as the training dataset.

4. Results

5. Conclusion

6. References

[1] https://www.kaggle.com/knightbearr/analysis-pizza-price-data-knightbearr/data

[2] Support Vector Classification, Scikit Learn, https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[3] A. Jung, "Machine Learning: basics", 2022, pp.90-92, https://github.com/alexjungaalto/MachineLearningTheBasics/blob/master/MLBasicsBook.pdf

[4] A. Jung, "Machine Learning: basics", 2022, pp.62-63, https://github.com/alexjungaalto/MachineLearningTheBasics/blob/master/MLBasicsBook.pdf

Appendix

1. The codes in Stage 2 are presented in the next page:

stage2_code

March 10, 2022

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score


def open_file(name):
    PizzaData = pd.read_csv(name)
    print("The original size of the dataset:")
    print(PizzaData.shape)
    print("The column of the dataset:")
    print(PizzaData.columns)

    # Visualize dataset
    fig, axes = plt.subplots(1, 3, figsize=(21, 5))  # Create the plot with 3␣
 ↪subplots to show the relationship between the label and the feature
    axes[0].scatter(PizzaData['Price'],
                    PizzaData['Company'])
    axes[0].set_xlabel("Price", size=15)
    axes[0].set_ylabel("Company", size=15)
    axes[0].set_title("Company vs Price", size=16)

    axes[1].scatter(PizzaData['Size'],
                    PizzaData['Company'])
    axes[1].set_xlabel("Size", size=15)
    axes[1].set_ylabel("Company", size=15)
    axes[1].set_title('Company vs Size', size=16)

    axes[2].scatter(PizzaData['Size'],
                    PizzaData['Type'])
    axes[2].set_xlabel("Type", size=15)
    axes[2].set_ylabel("Company", size=15)
    axes[2].set_title('Company vs Type', size=15)
    plt.show()
    X = PizzaData['Price'].to_numpy().reshape(-1, 1)
    y = PizzaData['Company'].to_numpy()

    print(X.shape)
    print(y.shape)

    X_clf = np.copy(X)
    y_clf = y
    np.random.seed(600)
    idx = np.random.choice(np.arange(371), 140)  # choose 140 datapoints from␣
 ↪the entire dataset

    X_train, X_val, y_train, y_val = train_test_split(X_clf[idx, :],␣
 ↪y_clf[idx], test_size=0.2,
                                                                          ␣
 ↪random_state=42)
    c = 100000

    clf_1 = SVC(C=c, decision_function_shape="ovo")

    clf_1.fit(X_train, y_train)
    y_pred_train = clf_1.predict(X_train)
    tr_accs = accuracy_score(y_train, y_pred_train)  # calculate the training␣
 ↪accuracy

    y_pred_val = clf_1.predict(X_val)
    val_accs = accuracy_score(y_val, y_pred_val)  # calculate the validation␣
 ↪accuracy

    print("\nTraining accuracies:\n", tr_accs)
    print("Validation accuracies:\n", val_accs)

def main():
    open_file('processed_data.csv')

# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    main()
```