Evaluating Classifications Models using Confusion Matrices

Lillian Mueller lmuelle1@umd.edu

***Abstract*—**

Confusion matrices are effective evaluators of classification models. This method allows analysts to predict an expected profit from the model which considers the cost of incorrect predictions and the benefit of correct predictions. Using the Iris dataset, the performance of the Decision Tree Model, Logistic Regression Mode, and the K-Nearest Neighbor Model are examined via confusion matrices and expected profit. By doing so, the K-Nearest Neighbor Model proves most effective in maximizing profit and potentially minimizing customer complaints. These results can also be confirmed through past reports using cross validation and comparing accuracy, precision, and recall scores. Overall, a confusion matrix has the capacity to qualify the performance of a classification model and proves to be an effective way to rank them.

I. Introduction

Predicting the classification of Iris flowers has been proven possible time and time again. In previous cases, this classification has been modeled using decision trees, logistic regression, and K-nearest neighbor classification models [1]- [3]. Using the same iris dataset as previous studies, evaluating which model may provide the greatest profit is key to determining whether a model is effective for business applications. To determine which model maximizes profit, confusion matrices will be used to quantify the cost of inaccurate classifications versus the benefit of accurate classifications. As a reminder, the iris dataset consists of 150 data entries, each containing the petal length, petal width, sepal length, sepal width, and class of iris plant. Each model aims to predict the class of the iris plant based on the given features.

Confusion matrices display outcomes from a model and sort those outcomes by predicted and actual values [4]. An analyst can then visually see how many outcomes from the model were classified correctly, which are referred to as true positives (TP) and true negatives (TN). The analyst can also see how many outcomes were classified incorrectly as well as which classification the point was confused with; these are referred to as false positives (FP) and false negatives (FN). Using this matrix, expected rates of the correct classifications versus the incorrect classifications can be found to calculate the overall expected value of the model [5]. Replicating the methodology to find each expected value for the models will aid in determining which model is most effective at maximizing profit.

The methodology followed to determine the most cost effective model is described in Section II. Results are discussed in Section III. And finally lasting comments and future research can be found in Section IV.

II. Methodology

This investigation entails using various classes from the library `scikitlearn`. To develop the models, the following classes were used: the `DecisionTreeClassifier` class from the `tree` module, the `LogisticRegression` class from the `linear_model` module, and the `KNeighbors` function from the `neighbors` module. Each was used to develop the decision tree model, the logistic regression model, and the k-nearest neighbors model respectively. To develop the confusion matrices and accuracy/precision scores of each model , `sklearn`'s `metrics` module was used. Other Python libraries were used along the was as well to help handle the data which includes `matplotlib.pyplot` , `pandas`, and `numpy`.

First, the Iris dataset was loaded as an `sklearn.utils.Bunch` object called `iris_data` via the `load_iris()` function. This object is similar to a Python dictionary. To make the data easier to read, it was transformed into the `df_iris` dataframe, where the data parameter was set as the `data` attribute of the dataset and the columns parameter as the `feature_names` attribute. A new column called "class" was added to this dataframe which contains the `target` variable of the Iris dataset; this is the class of the Iris plant - setosa, versicolor, or virginica. Since the `target` attribute contains an array with values from 0-2, the `replace` function was used to map these numerical values to their corresponding classifications: setosa for 0, versicolor for 1, and virginica for 2.

After the Iris dataset was loaded in and processed, it was split into a train group and test group using the `test_train_split` function, with 2/3 of the data as the train group and the remainder as the test group.

Next, the models were developed. For each model, similar methods were used as described in previous studies [1]- [3]. For context, the decision tree model was developed with the Gini impurity criterion, the logistic regression model was developed without any penalty, and the KNN model was developed using k=10 and the Euclidean distance metric. These were determined to be the most effective version of each model in their respective investigation. Additionally, all models were trained using the same training subset of the dataset.

After creating these models, they were applied to the testing subset of the dataset. The classification predictions were then compared to the actual testing data classifications. A confusion matrix was created for each model using the `confusion_matrix()` method from the `metrics` module. Using the `ConfusionMatrixDisplay` method, also from `metrics`, a depiction of the matrix was developed. To generate further evaluation measures, the models' accuracy, precision, and recall scores were calculated using `accuracy_score()`, `precision_score()`, and `recall_score()` respectively, all methods from the `metrics` module. These values can also be calculated using the `classification_report` method which also calculates the f-score (measure of accuracy using the recall and precision) and weighted and macro average accuracies.

Once the confusion matrix and general metrics were calculated for each model, the expected value for each model could be calculated. The expected value is calculated by summing the products of each error rate multiplied by the corresponding cost or benefit. Error rates are calculated by dividing the outcome counts from the confusion matrix by the number of total outcomes. The confusion matrix consists of counts of true positives, true negatives, false positives, and false negatives. The following diagram depicts this process.

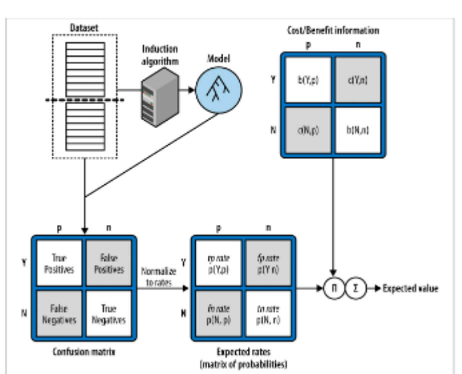

Fig. 1. Expected Value Calculation Diagram [5]

For this case, since there are 3 classifications, setosa, virginica, and versicolor, the diagrams will correspond to Table I.

	0 = setosa	1 = versicolor	2 = virginica
0	predicted : 0 actual : 0	predicted : 1 actual : 0	predicted : 2 actual : 0
1	predicted : 0 actual : 1	predicted : 1 actual : 1	predicted : 2 actual : 1
2	predicted : 0 actual : 2	predicted : 1 actual : 2	predicted : 2 actual : 2

TABLE I
CONFUSION TABLE MEANING FOR IRIS DATASET

To start this calculation, the cost/benefit information was developed. For correct classifications, a value of 1 was assigned as a benefit and for each incorrect classification, a cost of -1. The final cost/benefit matrix is shown in Table II.

	Predicted: 0 = setosa	1 = versicolor	2 = virginica
Actual: 0	1	-1	-1
1	-1	1	-1
2	-1	-1	1

TABLE II
COST/BENEFIT INFORMATION

In order to find the expected rates of each outcome, each value in the confusion matrix was divided by the sum of the confusion matrix. Each rate is then multiplied by the corresponding value in the cost/benefit matrix. Finally, to find the expected profit for the model, all values are summed.

III. RESULTS

A. Confusion Matrices

After the development of each model, a confusion matrix was created to better understand the performance of the model on the testing dataset along with its classification report. Figures 2, 3, and 4 show the matrices for the decision tree model, the logistic regression model, and the KNN model respectively and Tables III, IV, and V display the corresponding classification reports.

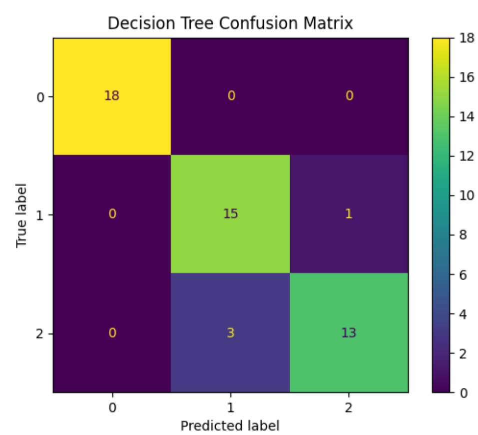

Fig. 2. Confusion Matrix for Decision Tree Model

	precision	recall	f1-score	support
0	1.00	1.00	1.00	18
1	0.83	0.94	0.88	16
2	0.93	0.81	0.87	16
accuracy	-	-	0.92	50
macro avg	0.92	0.92	0.92	50
weighted avg	0.92	0.92	0.92	50

TABLE III
CLASSIFICATION REPORT FOR DECISION TREE MODEL

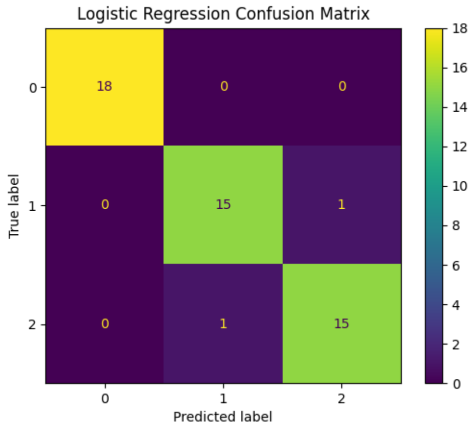

Fig. 3. Confusion Matrix for Logistic Regression Model

	precision	recall	f1-score	support
0	1.00	1.00	1.00	18
1	0.94	0.94	0.94	16
2	0.94	0.94	0.94	16
accuracy			0.96	50
macro avg	0.96	0.96	0.96	50
weighted avg	0.96	0.96	0.96	50

TABLE IV
CLASSIFICATION REPORT FOR LOGISTIC REGRESSION MODEL

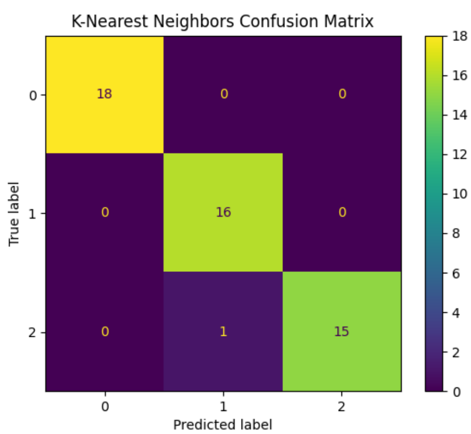

Fig. 4. Confusion Matrix for K-Nearest Neighbor Model

	precision	recall	f1-score	support
0	1.00	1.00	1.00	18
1	0.94	1.00	0.97	16
2	1.00	0.94	0.97	16
accuracy	-	-	0.98	50
macro avg	0.98	0.98	0.98	50
weighted avg	0.98	0.98	0.98	50

TABLE V
CLASSIFICATION REPORT FOR KNN MODEL

Looking at the matrices, all the models are fairly accurate at classifying each iris flower. Each model was tested with 50 data entries, since the models were trained on 2/3 of the total dataset. The numbers in each square represent the number of outcomes for each event. These events correspond to Table I. From these matrices alone, an analyst may conclude that the KNN model has the best performance as its associated confusion matrix shows the least amount of misclassified events. Additionally, looking at the classification reports from each model, the KNN model had the greatest precision, accuracy, and recall scores of all the models. From the classification reports and confusion matrices, it seems all the models struggle to accurately decipher between versicolor and virginica.

B. Evaluating Expected Profit

After following the procedures in Section II, the following expected profits were found.

Model	Expected Profit
Decision Tree	0.84
Logistic Regression	0.92
KNN	0.96

TABLE VI
EXPECTED VALUES

An analyst can conclude that the K-Nearest Neighbor model will yield the most profit and therefore maximize profit. This makes sense as flowers, and organisms in general, are classified by similarities and KNN models model similarity. An analyst can also conclude that the Decision Tree Model is the worst model of the three for this dataset.

These results also reinforce past investigations. When using cross validation to compare the decision tree, logistic regression, and KNN models, the KNN model was also found to be the most stable for the iris dataset [3].

IV. DISCUSSION

The KNN model proves to be best model for the iris dataset of the models investigated. Using cross validation, confusion matrices and expected profit, and by comparing accuracy scores, all methods seem to point to the K-Nearest Neighbor model. As expected, each validation method reinforces the other. This is important to note as analysts should use more than one validation method to ensure a model is best fit for a dataset. Here, by calculating the expected profit, the KNN model may be expected to have greater return on investment as its accuracy and precision is quantified into actual costs and benefits, however, since this was only performed once, the analyst cannot confirm the performance of the model using a different data split. The cross validation method tested how stable the KNN model was and how it may react differently when trained and tested with different data splits. In this method, the analyst has no way of weighing whether the accuracy outweighs the impact of incorrect predictions. Each method has its flaws and advantages and using more than a single validation method allows for better understanding of the model and its capacity.

In the future, when using confusion matrices and expected profit methods to evaluate the performance of a classification model, the analyst could increase the number of times the method is run and use a different training and testing data split. As the training sets may not have the exact same proportion of classes, the overall confusion matrix could be different and prove one model yields greater expected profit over another when using specific training datasets. In addition to evaluating the expected profit across multiple models, an analyst could experiment with changing the test size when splitting the data into training and testing sets. In this study, only 1/3 of the data was used as a testing set, meaning only 50 points were used when creating the confusion matrices. This is a fairly small number, so finding the "sweet spot" for the training and testing set ratio may decrease the variance of the confusion matrix and subsequent expected profit across different testing sets.

REFERENCES

[1] L. Mueller and R. Hong, "Investigating Decision Trees".

[2] L. Mueller and R. Hong, "Iris Classification Using Logistic Regression".

[3] L. Mueller and R. Hong, "Evaluating the Performance of K-Nearest Neighbors Classification".

[4] "Confusion Matrix - an overview — ScienceDirect Topics." Accessed: Oct. 26, 2023. [Online]. Available: https://www.sciencedirect.com/topics/engineering/confusion-matrix

[5] F. Provost and T. Fawcett, Data Science for Business: What You Need to Know About Data Mining and Data Analytic Thinking, First edition (2005).