Iris Classification Using Logistic Regression

Lillian Mueller lmuelle1@umd.edu
Regina Hong rhong@umd.edu

***Abstract*— Given a dataset of Iris characteristics, this report explored the development of a predictive model to classify Iris species. Logistic regression was used because of its main purpose of predicting the probability of categorical outcomes based on historical data. To test the procedure of developing the logistic model, we began by replicating the results of a binary example that predicted the probability of a student passing an exam based on the number of hours they studied. Taking those procedures, we applied the process to the Iris dataset. We were able to develop an accurate logistic regression model to classify the data and compared its performance with a decision tree classification model. We found that the logistic regression model had greater accuracy for classifying the data and also was less prone to overfitting. Overall, this model demonstrated its ability to combat overfitting and deal with complexity within the dataset through manipulation of the cost function and penalty terms.**

I. Introduction

Logistic regression is a type of statistical model that is used to predict categorical outcomes based on historical data. This type of supervised learning is considered a discriminative model because it aims to categorize between classes. It is different from other types of supervised learning, such as a Decision Tree because it does not predict the actual value of the dependent variable, rather the probability of it occurring [1]. The model can be a binomial logistic regression, where there are only two outcomes possible, or multinomial, where the probability of multiple outcomes are present [2], as in the case of this study. When analyzing a dataset using this model, it should be assumed that there is little to no multicollinearity between the independent variables.

A referenced logistic regression model [3] was used for this study. This model aimed to classify whether a student would pass or fail (binomial model) based on the number of hours they studied. It also used a linear model. A linear equation was formed by using maximum likelihood estimation [1] to determine the β values that correspond to the coefficient and vertical intercept of the model in the linear equation $y = \beta_0 + x\beta_1$, where `x` was the number of hours the student studied. After β_0 and β_1 are determined, the probability of a student passing the exam based on the number of hours they studied was determined using the equation $p = \frac{1}{1+e^{-y}}$.

The focal dataset is the Iris dataset, supplied by the scikitlearn library. This dataset contains four measurements for each Iris flower: sepal length, sepal width, petal length, and petal width. Alongside these features, the classification of each Iris flower is given; the flower is classified as setosa, versicolor, or virginica. The logistic regression model was developed using the `linear_model.LogisticRegression` class from the library. In addition to the logistic regression model, a decision tree model was also generated to compare the results of the two supervised learning models with respect to the same dataset. Given two additional sets of measurements, the goal is to determine the Iris classification of the two new flowers.

The aforementioned binomial model was then expanded so that it could be applied to the Iris dataset with its multinomial nature. As this type of model is inherently prone to overfitting, the `penalty` parameter was changed and the effects of the change were observed.

The methodology followed during the course of this study is presented, detailing the Python libraries used, as well as ways in which the logistic regression model was created, modified, compared, and utilized. Afterwards, the results of the model are described Finally, a brief discussion of the results and future applications to larger datasets is offered.

II. Methodology

A. Logistic Regression Example

To begin learning and understanding the logistic modeling algorithm developed by SKLearn, we replicated the logistic regression model example found in an article on Wikipedia. In this article, the author wants to correlate the number of hours a student spent studying for an exam and whether the student passed the exam. The author modeled the data with a logistic regression that allowed him to predict the probability a student would pass the exam.

For this investigation, we imported the following Python libraries: `sklearn.linear_model`, `pandas`, `numpy`, and `matplotlib.plyplot`. We then copied the same data used in the binomial model study into a dataframe so we could work with the data more efficiently.

Sample	Hours (xk)	Pass (yk)
15	4.25	1.0
16	4.50	1.0
18	5.00	1.0
2	1.00	0.0
3	1.25	0.0

TABLE I
Sample of Example Data

To model this data, we used the `LogisticRegression` ↪ class from the `sklearn.linear_model` library and fit the model by specifying the feature, hours spent studying, and the target, whether the student passed the exam. In order to fully replicate the example, we set the penalty parameter to `None` since this example was modeled on a binary dataset. Using the `coef_` and `intercept_` attributes of the model we refined, the parameters of the logistic model were found,

β_0 and β_1. From these coefficients, the location parameter, μ, and scale parameter, s, were derived using the following equations respectively:

$$\mu = -\beta_0/\beta_1$$

$$s = \beta_1^{-1}$$

β_0	β_1	μ	s
-4.077	1.504	2.710	0.665

TABLE II
LOGISTIC MODEL PARAMETERS

Using these parameters, we computed the logistic function, $p(x) = \frac{1}{1+e^{-(\beta_0+x\beta_1)}}$, and compared the probability values to those in the example. Additionally, the logistic regression model from `sklearn` computes this probability function as well. Using the `predict_proba` method, we found the probability of each possible outcome for each case. We then used `matplotlib.pyplot`, to graph the probability of passing an exam vs the hours spent studying curve as seen in Figure 1.

B. Classifying Iris Data via Logistic Regression

To fit a logistic regression model to the Iris dataset, several Python packages were utilized; these included `sklearn`, `pandas`, `numpy`, and `matplotlib.plyplot`. Specifically from `sklearn`, we use the following modules `linear_model`, `preprocessing`, `model_selection` ↪ , `metrics`, and `tree`.

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, the pandas package was imported and used to transform said dictionary 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.

Next, the Iris dataset was split into a test group and train group, where the train group underwent the logistic regression modeling process and the test group was then used in the resulting model. To split the data into those two groups, the `train_test_split` function was imported from the `sklearn.model_selection` module with the `test_size` parameter set to 0.33 so that around $\frac{1}{3}$ of the data is set aside for testing.

Following the same procedure when building the binomial logistic regression model described in Section A, we created a model for the Iris dataset. However, in this case, rather than there being only one feature, there are four for the Iris dataset: the pedal width, pedal length, sepal width, and sepal length. The target for this dataset is the Iris classification. Using the entries in the training dataset, we applied the logistic regression model and found the following coefficients.

β_0	9.251
β_1	-0.440
β_2	0.792
β_3	-2.254
β_4	-0.975

TABLE III
LOGISTIC MODEL PARAMETERS FOR IRIS CLASSIFICATION

To evaluate the performance of our logistic regression model, we applied the model to the testing dataset 1000 times and calculated the accuracy score as well as the r2 score. Each regression model implemented a different cost function. In the SKLearn library, the logistic regression's `penalty` parameter refers to the added penalty terms in the cost function associated with the model. When it is set to `l2`, the default value, the model adds a penalty term to the cost function to reduce the complexity of the model and reduce overfitting. This term is the square of the magnitude of the coefficients [4]. When it is set to `None`, no penalty is considered. We tested these two scenarios by modifying the `penalty` parameter of the model. This change was made to see how the model would perform when a penalty was implemented.

Finally, we wanted to compare the performance of this logistic regression classification model to the performance of the decision tree classification model from a previous study [5]. Using the same training and testing datasets used to train the logistic model, we developed the decision tree model and ran it 1000 times. We used the average accuracy and r2 score here as well to compare the three models. To find these values we used the `metrics.accuracy_score` and `metrics.` ↪ `r2_score` methods from the `sklearn` library.

III. RESULTS

A. Passing Exams Versus Hours Spent Studying

In comparison to the results stemming from the referenced logistic regression model [3], the results from the current model, using `sklearn.linear_model.` ↪ `LogisticRegression`, were identical. The referenced model produced β_0 and β_1 values of approximately `-4.1` ↪ and `1.5`, respectively, both of which the current model agreed nicely with. A table with the log-odds, odds, and probabilities of a student passing the exam based on the number of hours studied was calculated in the current model. As seen in Table IV below, the current model yielded the same calculated values as the referenced model.

Hours	log-odds, t	odds, e^t	probability, p	Model Prediction
1.000000	-2.573	0.076	0.070	0.070
2.000000	-1.068	0.343	0.255	0.255
2.710086	0.000	1.000	0.500	0.500
3.000000	0.436	1.546846	0.607	0.607
4.000000	1.940	6.964	0.874	0.874
5.000000	3.445	31.359	0.969	0.969

TABLE IV
ESTIMATED PROBABILITIES

Finally, the logistic regression curves were compared between the current and referenced models, based on the above tables. It would make sense therefore that these two graphs were also equivalent.

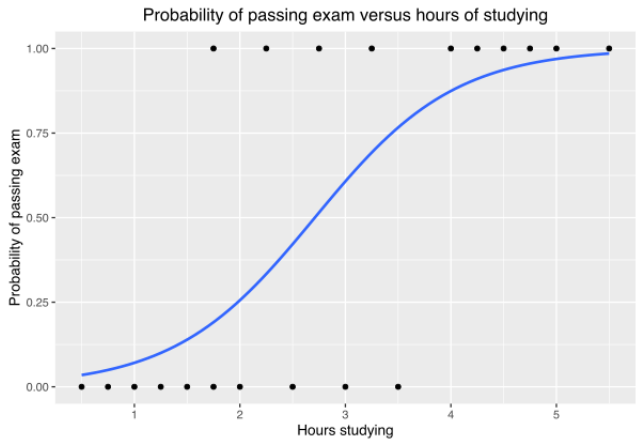

Fig. 1. Logistic Function of Passing Exam Based on Hours Studying

B. Decision Tree versus Logistic Regression Classification of Iris Data

With the current logistic regression model now used to categorize the Iris dataset and with the Iris dataset split into $\frac{1}{3}$ test data and $\frac{2}{3}$ training data, a new linear equation was created. The calculated coefficients of the linear model were were $\beta_0 = 9.25$ (vertical intercept), $\beta_1 = -0.44$, $\beta_2 = -0.79$, $\beta_3 = -2.25$, and $\beta_4 = -0.98$.

When comparing the results of the logistic regression model with results from the secondary Decision Tree model, the accuracy of the test data and the r2 score were much greater in the logistic regression model.

Model	Train Accuracy	Test Accuracy	r2 Score
Logistic Regression	0.98	0.94	0.905
Decision Tree	1.00	0.92	0.874

TABLE V
MODEL PERFORMANCE COMPARISON

As the current logistic regression model included the default parameter of `penalty='l2'`, a different iteration of the model with `penalty=None` was also compared with the results from the Decision Tree model. These three models (Decision Tree, logistic regression with penalty, and logistic regression without penalty) were all run 1000 times each.

Description	Train Data Accuracy	Test Data Accuracy	r2 Score
count	1000.0	1000.000	1000.000
mean	1.0	0.930	0.889
std	0.0	0.015	0.024
min	1.0	0.920	0.873
25%	1.0	0.920	0.873
50%	1.0	0.920	0.873
75%	1.0	0.940	0.905
max	1.0	0.960	0.936

TABLE VI
STATISTICS FOR DECISION TREE UTILIZING GINI IMPURITY CRITERION

Description	Train Data Accuracy	Test Data Accuracy	r2 Score
count	1000.00	1000.00	1000.000
mean	0.98	0.94	0.905
std	0.00	0.00	0.000
min	0.98	0.94	0.905
25%	0.98	0.94	0.905
50%	0.98	0.94	0.905
75%	0.98	0.94	0.905
max	0.98	0.94	0.905

TABLE VII
STATISTICS FOR LOGISTIC REGRESSION WITH L2 PENALTY

Description	Train Data Accuracy	Test Data Accuracy	r2 Score
count	1000.0	1000.00	1000.000
mean	1.0	0.96	0.936
std	0.0	0.00	0.000
min	1.0	0.96	0.936
25%	1.0	0.96	0.936
50%	1.0	0.96	0.936
75%	1.0	0.96	0.936
max	1.0	0.96	0.936

TABLE VIII
STATISTICS FOR LOGISTIC REGRESSION WITHOUT PENALTY

As the accuracy converges for each model, these methods can be more effectively compared. From these tables, it can be inferred that the logistic regression models are better predictors of the iris dataset. The decision tree model had an average accuracy of 0.93 when applied to the test datasets whereas the logistic models had an accuracy of 0.94 and 0.96, with penalty and without, respectively. As seen in the past, the decision tree's training data accuracy was 1.0 and the test data accuracy was much lower at 0.93 which indicates overfitting to the training dataset.

Comparing the logistic models specifically, the model without implementing any penalty functions seems to perform better with a mean accuracy of 0.96. The accuracy for the model with penalty was 0.94. The r2, or the coefficient of determination, scores have a similar trend where the model without penalty had a greater mean r2 score of 0.936 and the model with penalty had a lesser score of 0.905. For the iris dataset, the logistic model without penalty is the best choice.

C. New Classifications Using Logistic Regression

When using the `predict_proba` function to create a ranking of the results of the logistic regression model, the output was compared to the original Iris dataset to see how the model probabilities compare to the actual Iris flower classification. A truncated version of this dataframe with randomly selected results is shown below:

Sample	P(setosa, 0)	P(versicolor, 1)	P(virginica, 2)
129	0.000	0.189	0.811
42	0.988	0.012	0.000
79	0.106	0.885	0.009
30	0.965	0.035	0.000
133	0.001	0.440	0.559

TABLE IX
RESULTS SAMPLE

The probabilities of setosa, versicolor, and virginica correspond to how likely the Iris flower is actually classified as

the class. The higher the probability (closer to 1), the more likely the model categorizes the flower as being that class. For example, the first row shows that the flower has a 0 probability of being classified as setosa, 0.189 as versicolor, and 0.811 as virginica. Therefore it is most likely that the flower is actually Iris virginica. When referring to the original Iris dataset, one can confirm that this is indeed correct, the flower was an Iris virginica.

The rankings and comparisons with the original Iris dataset seem to further confirm the accuracy of the model. This holds true even in the case of the fifth row, where the probabilities of versicolor and virginica are both fairly close together and near 0.5. The model shows that the probability of the flower being virginica is still higher than versicolor and the actual flower is classified as an Iris virginica.

Given the two new data records (measurements of two new flowers), the probabilities of each flower class can be calculated. This table is shown below:

	Entry 1	Entry 2
sepal length (cm)	5.800000	6.000000
sepal width (cm)	2.800000	2.200000
petal length (cm)	5.100000	4.000000
petal width (cm)	2.400000	1.000000
Prediction	2.000000	1.000000
Prob of setosa (0)	0.000166	0.012343
Prob of versicolor (1)	0.076790	0.966225
Prob of virginica (2)	0.923044	0.021431

TABLE X
NEW ENTRY CLASSIFICATION

It can be seen that the first record is predicted to be an Iris virginica (with a probability of 0.923) and the second record is predicted to be an Iris versicolor (with a probability of 0.966).

IV. DISCUSSION

This investigation has shown that logistic regression models can be applied to both simple and complex datasets. We were able to accurately and precisely replicate the Wikipedia binary dataset as well as classify new iris data with high confidence. When increasing the complexity of the model to fit the iris dataset, we expected the addition of the penalty term in the cost function to improve the accuracy of the logistic regression model. However, for this dataset, we found the opposite was true. While adding the L2 penalty reduced the overfitting of the model, it also reduced the accuracy of the predictions when applied to the test dataset. This may indicate that the dataset does not have the complexity to benefit from the added cost term or the dataset is too small where overfitting may not be a significant problem for this model.

Upon developing the logistic regression and decision tree classification models, we see how some models are better equipped for different applications. Logistic models are less prone to overfitting compared to decision trees which means they are less likely to be skewed by outliers. The results showed there was greater variance in the accuracy among decision tree model evaluation. Additionally, logistic regression models include cost functions that can reduce complexity of the models. This feature makes this method more advantageous than decision trees when applied to more complex datasets where there are more variables or where the variables themselves are more codependent.

To further this investigation, we should increase the size of the dataset and apply the L2 penalty term when developing the logistic regression model. With a larger dataset, we may be able to see the difference between adding the penalties and without adding any penalties. Additionally, we could also introduce more variables to the dataset and see how the model reacts to more features.

REFERENCES

[1] "What is Logistic regression? — IBM." https://www.ibm.com/topics/logistic-regression (accessed Sep. 23, 2023).

[2] "Python Machine Learning - Logistic Regression." https://www.w3schools.com/python/python-ml-logistic-regression.asp (accessed Sep. 23, 2023).

[3] "Logistic regression," Wikipedia. Aug. 31, 2023. Accessed: Sep. 22, 2023. [Online].

[4] "How is L2 (ridge) penalty calculated in sklearn LogisticRegression function? — Saturn Cloud Blog," Jul. 10, 2023. https://saturncloud.io/blog/how-is-l2-ridge-penalty-calculated-in-sklearn-logisticregression-function/ (accessed Sep. 23, 2023).

[5] Mueller, L. and Hong, R. "Investigating Decision Trees," Sep. 18, 2023