EN3150 Assignment 02: Learning from data and related challenges and classification

Sampath K. Perera

Sep. 3, 2024

1 Logistic regression

1. Use the code given in listing 1 to load data.
2. What is the purpose of "y_encoded = le.fit_transform(df_filtered['species'])" ? [5 marks]
3. What is the purpose of "X = df.drop(['species', 'island', 'sex'], axis=1)" ? [5 marks]
4. Why we cannot use "island" and "sex" features? [10 marks]
5. Now, use the code given in listing 2 to train a logistic regression model. [10 marks]
6. What is the usage of "random_state=42" ? [5 marks]
7. Why is accuracy low? why does the saga solver perform poorly? [10 marks]
8. Now change the solver to "liblinear" by using logreg = LogisticRegression(solver='liblinear'). What is the classification accuracy with this configuration? [5 marks]
9. Why does the "liblinear" solver perform better than "saga" solver ? [15 marks]
10. Compare the performance of the "liblinear" and "saga" solvers with feature scaling. If there is a significant difference in the accuracy with and without feature scaling, what is the reason for that.
    You may use Standard Scaler available in sklearn library. [15 marks]
11. Run the code given in listing 3. What is the problem of this code and how to solve this? [5 marks]
12. Suppose you have a categorical feature with the categories 'red', 'blue', 'green', 'blue', 'green'. After encoding this feature using label encoding, you then apply a feature scaling method such as Standard Scaling or Min-Max Scaling. Is this approach correct? or not?. What do you propose ? [15 marks]

```
import seaborn as sns
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Load the penguins dataset
df = sns.load_dataset("penguins")

df.dropna(inplace=True)

# Filter rows for 'Adelie' and 'Chinstrap' classes
selected_classes = ['Adelie', 'Chinstrap']
df_filtered = df[df['species'].isin(selected_classes)].
    copy()  # Make a copy to avoid the warning

# Initialize the LabelEncoder
le = LabelEncoder()

# Encode the species column
y_encoded = le.fit_transform(df_filtered['species'])

df_filtered['class_encoded'] = y_encoded

# Display the filtered and encoded DataFrame
print(df_filtered[['species', 'class_encoded']])

# Split the data into features (X) and target variable (y)

y = df_filtered['class_encoded']  # Target variable
X = df_filtered.drop(['species', 'island', 'sex','
    class_encoded'], axis=1)
```

Listing 1: Data load and preprocessing.

```
#Split the data into training and testing sets
 X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=42)

 Train the logistic regression model. Here we are using saga
    solver to learn weights.
 logreg = LogisticRegression(solver='saga')

 logreg.fit(X_train, y_train)

 # Predict on the testing data
 y_pred = logreg.predict(X_test)

 # Evaluate the model
 accuracy = accuracy_score(y_test, y_pred)
 print("Accuracy:", accuracy)

 print(logreg.coef_, logreg.intercept_)
```

Listing 2: Training LR model.

```
import seaborn as sns
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Load the penguins dataset
df = sns.load_dataset("penguins")
df.dropna(inplace=True)

# Filter rows for ’Adelie’ and ’Chinstrap’ classes
selected_classes = [’Adelie’, ’Chinstrap’]
df_filtered = df[df[’species’].isin(selected_classes)].
    copy()  # Make a copy to avoid the warning

# Initialize the LabelEncoder
le = LabelEncoder()
# Encode the species column
y_encoded = le.fit_transform(df_filtered[’species’])
df_filtered[’class_encoded’] = y_encoded

df_filtered.head()

X = df_filtered.drop([’species’, ’class_encoded’], axis=1)
y = df_filtered[’class_encoded’]  # Target variable

X.head()

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=42)

logreg = LogisticRegression(solver=’saga’)
logreg.fit(X_train, y_train)

# Predict on the testing data
y_pred = logreg.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

print(logreg.coef_, logreg.intercept_)
```

Listing 3: Logistic regression with multiple features.

Question 2[1]

Consider a dataset collected from a statistics class that includes information on students. The dataset includes variables x_1 representing the number of hours studied, x_2 representing the undergraduate GPA, and y indicating whether the student received an A^+ in the class. After conducting a logistic regression analysis, we obtained the following estimated coefficients: $w_0 = -5.9$, $w_1 = 0.06$, and $w_2 = 1.5$.

1. What is the estimated probability that a student, who has studied for 50 hours and has an undergraduate GPA of 3.6, will receive an A^+ in the class? [12.5 marks]

2. To achieve a 60% chance of receiving an A^+ in the class, how many hours of study does a student like the one in part 1 need to complete? [12.5 marks]

2 Logistic regression on real world data

1. Choose a data set from UCI Machine Learning Repository that is appropriate for logistic regression.

2. Obtain the correlation matrix for the dataset you have chosen. Further, obtain pair plots using sns.pairplot. If your dataset contains many features, select up to 5 features for analysis. Comment on your results. [5 marks]

3. Fit a logistic regression model to predict the dependent variable.
   Evaluate the model's performance and determine how often it correctly predicts dependent variable (class). [10 marks]

4. Use statsmodels.Logit to obtain and interpret the p-values for the predictors and determine if any features can be discarded. [10 marks]

[1] Based on "James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer."

3 Logistic regression First/Second-Order Methods

1. Use the code given in listing 4 to generate data. Here, variable y and X are class labels and corresponding feature values, respectively.

2. Implement batch gradient descent to update the weights for the given dataset over 20 iterations. State the method used to initialize the weights and reason for your selection. [15 marks]

3. Specify the loss function you have used and state reason for your selection. [5 marks]

4. Plot the loss with respect to number of iterations. [5 marks]

5. Repeat step 2 for stochastic Gradient descent. [5 marks]

6. Implement Newton's method to update the weights for the given dataset over 20 iterations. [15 marks]

7. Plot the loss with respect to number of iterations. [5 marks]

8. Plot the loss with respect to number of iterations for Gradient descent, stochastic Gradient descent and Newton method's in a single plot.
   Comment on your results. [25 marks]

9. Propose two approaches to decide number of iterations for Gradient descent and Newton's method. [10 marks]

10. Suppose the centers in in listing 4 are changed to centers = [[3, 0], [5, 1.5]]. Use batch gradient descent to update the weights for this new configuration. Analyze the convergence behavior of the algorithm with this updated data, and provide an explanation for convergence behavior. [15 marks]

```
import numpy as np
import matplotlib.pyplot as plt

import numpy as np
from sklearn.datasets import make_blobs
# Generate synthetic data
np.random.seed(0)
centers = [[-5, 0], [5, 1.5]]

X, y = make_blobs(n_samples=2000, centers=centers, random_state=5)
transformation = [[0.5, 0.5], [-0.5, 1.5]]
X = np.dot(X, transformation)
```

Listing 4: Data generation.

4 Submission

- Upload a report and your codes as a zip file named as "EN3150_your_indexno_A02.zip". Include the index number and the name within the report as well. Please include all your answers in the report.
- Pay careful attention to formatting such as font size, spacing, and margins.
- Include a title page with necessary information (e.g., title, author, date, index no).
- Use consistent and professional formatting throughout the document.
- Plagiarism will be checked and in cases of plagiarism, an extra penalty of 50% will be applied. In case of copying from each other, both parties involved will receive a grade of zero for the assignment. Academic integrity is of utmost importance, and any form of plagiarism[2] or cheating will not be tolerated.
- An extra penalty of 15% is applied for late submission.

[2] https://en.wikipedia.org/wiki/Plagiarism