silas_assignment2

April 19, 2025

Leon Silas

Intro to Data Science - CAP 5768

0.0.1 Assignment 2

[152]:
```python
#imports
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪ConfusionMatrixDisplay, mean_absolute_error, mean_squared_error, r2_score
import matplotlib.pyplot as plt
```

0.0.2 1) (30 points) A hospital wants to predict whether a patient has a certain disease (1:positive, 0:negative) based on their age and cholesterol level. Using logistic regression, build a model to predict the probability of a patient having the disease based on their age and cholesterol level.

[153]:
```python
# Create dataframe from data given
df_1 = pd.DataFrame({
'age': [45, 50, 30, 55, 65, 40, 35, 48, 52, 58, 42, 60, 70, 32, 47, 38, 44, 51,␣
 ↪57, 62, 34, 39, 46, 54, 61, 36, 49, 56, 63, 37, 43, 59, 64, 41, 53, 31, 33],
'cholesterol': [180, 200, 150, 210, 220, 190, 160, 185, 195, 215, 175, 225,␣
 ↪230, 155, 183, 170, 178, 205, 212, 227, 165, 172, 182, 208, 222, 162, 187,␣
 ↪217, 228, 168, 177, 220, 225, 173, 200, 145, 160],
'disease': [1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0,␣
 ↪1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0]
})
```

a) Implement logistic regression using Python to build the predictive model.

[154]:
```python
# Logistic regression model
x = df_1[['age', 'cholesterol']]
y = df_1['disease']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,␣
 ↪random_state=42)
```

```
model = LogisticRegression().fit(x_train, y_train)
```

b) Evaluate the model's performance using appropriate metrics such as accuracy, precision, recall, and F1-score.

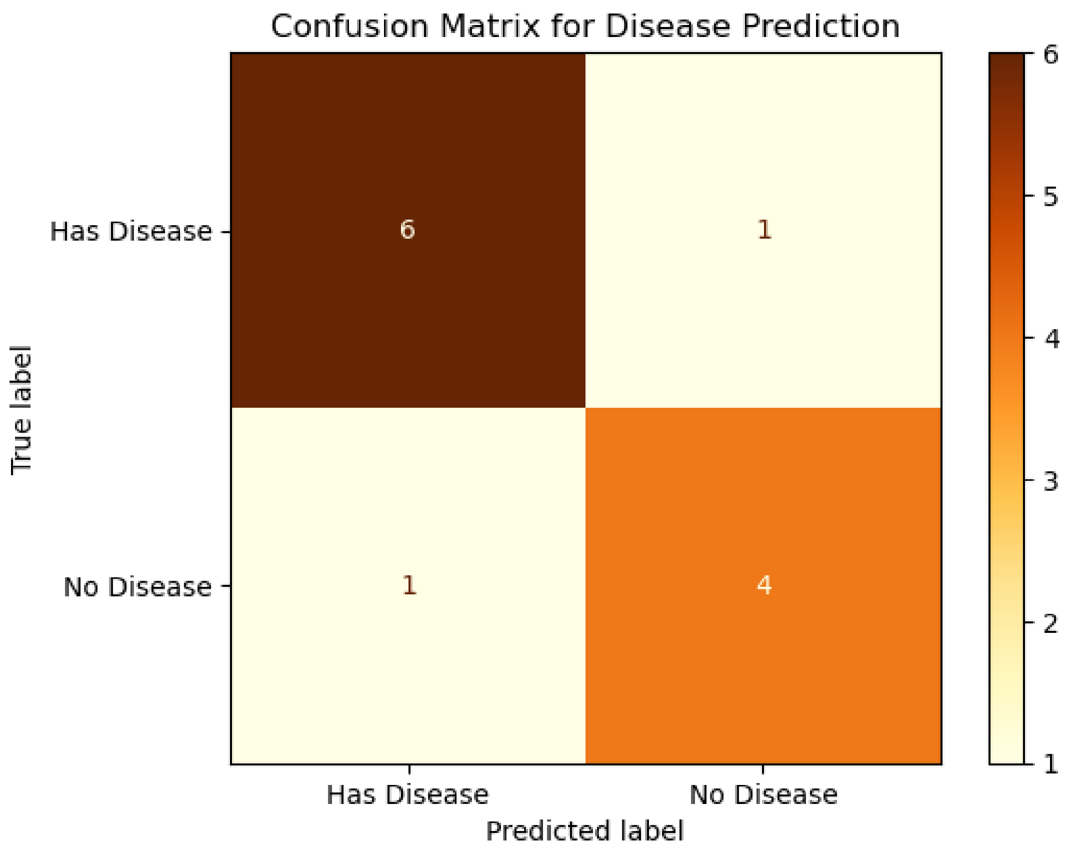
[155]:
```
# Model evaluation
y_pred = model.predict(x_test)
print("Logistic Regression Model Evaluation")
print(f"Classification Report:\n{classification_report(y_test, y_pred)}")

# Confusion matrix
TP, FN, FP, TN = confusion_matrix(y_test, y_pred).ravel()
confusion_matrix = np.array([[TP, FN], [FP, TN]])
display_matrix = ConfusionMatrixDisplay(confusion_matrix, display_labels=['Has␣
 ↪Disease', 'No Disease'])
display_matrix.plot(cmap=plt.cm.YlOrBr)
plt.title('Confusion Matrix for Disease Prediction')
plt.show()
```

```
Logistic Regression Model Evaluation
Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.86      0.86         7
           1       0.80      0.80      0.80         5

    accuracy                           0.83        12
   macro avg       0.83      0.83      0.83        12
weighted avg       0.83      0.83      0.83        12
```



Based on the metrics and confusion matrix for our model, we can see that our model performs well, with only 2 misclassifications out of our test data. This gives us at least an 80% effectiveness or higher over all our metrics.

c) Interpret the coefficients of the logistic regression model and discuss their significance in predicting the probability of having the disease.

[156]:
```python
# Coefficients
print("Logistic Regression Coefficients:")
for i, col in enumerate(x.columns):
    print(f"{col}: {model.coef_[0][i]:.4f}")

# Percentage increase in odds of disease for each variable
print(f"Age: {np.exp(model.coef_[0][0]) - 1:.2%}")
print(f"Cholesterol: {np.exp(model.coef_[0][1]) - 1:.2%}")
```

```
Logistic Regression Coefficients:
age: 0.4747
cholesterol: 0.7603
Age: 60.75%
Cholesterol: 113.89%
```

Our coefficients show that for each additional year of age a person has, they can expect to be ~60.75% more likely to have the disease, and for each unit increase in cholesterol, they are ~113.89% more likely to have the disease.

0.0.3 2) (30 points) A company wants to predict the sales of their product based on advertising spending on TV, radio, and newspapers. Using linear regression, build a model to predict the sales of the product based on advertising spending on different media.

[157]:
```
# Create dataframe from data given
df_2 = pd.DataFrame({
'TV': [230.1, 44.5, 17.2, 151.5, 180.8, 8.7, 57.5, 120.2, 8.6, 199.8, 66.1, 214.
 ↪7, 23.8, 97.5, 204.1, 195.4, 67.8, 281.4, 69.2, 147.3, 218.4, 237.4, 13.2,␣
 ↪228.3, 62.3, 262.9, 142.9, 240.1, 248.8, 70.6, 292.9, 112.9, 97.2, 265.6, 95.
 ↪7, 290.7, 266.9, 74.7, 43.1, 228.0, 202.5, 177.0, 293.6, 206.9, 25.1, 175.1,␣
 ↪89.7, 239.9, 227.2, 66.9, 199.8, 100.4],
'radio': [37.8, 39.3, 45.9, 41.3, 10.8, 48.9, 32.8, 19.6, 2.1, 2.6, 5.8, 24.0,␣
 ↪35.1, 7.6, 32.9, 46.0, 52.9, 21.1, 42.0, 3.6, 4.1, 36.9, 41.1, 26.9, 37.7,␣
 ↪22.9, 27.5, 5.1, 15.9, 16.9, 12.6, 3.5, 29.3, 16.7, 27.1, 16.0, 28.3, 17.4,␣
 ↪1.5, 20.0, 1.4, 4.1, 43.8, 33.0, 52.4, 17.0, 38.7, 50.6, 18.7, 12.4, 4.9, 9.
 ↪3],
'newspaper': [69.2, 45.1, 69.3, 58.5, 58.4, 75.0, 23.5, 11.6, 1.0, 21.2, 24.2,␣
 ↪4.0, 65.6, 56.5, 45.7, 69.2, 75.6, 29.5, 20.3, 23.6, 43.1, 65.6, 89.4, 25.6,␣
 ↪35.1, 19.5, 12.6, 3.5, 1.6, 79.2, 22.3, 53.5, 59.0, 29.7, 23.2, 49.6, 26.2,␣
 ↪22.9, 65.7, 8.5, 15.9, 16.0, 50.0, 11.6, 18.4, 18.3, 19.1, 22.3, 7.8, 54.7,␣
 ↪52.9, 22.5],
'sales': [22.1, 10.4, 9.3, 18.5, 12.9, 7.2, 11.8, 13.2, 4.8, 10.6, 8.6, 17.4, 9.
 ↪2, 9.7, 19.0, 22.4, 12.5, 24.4, 11.3, 14.6, 18.0, 12.5, 5.6, 15.5, 9.7, 12.
 ↪0, 15.0, 15.9, 18.9, 10.5, 21.4, 11.9, 9.6, 17.4, 9.5, 12.8, 25.4, 14.7, 10.
 ↪1, 21.5, 16.6, 17.1, 20.7, 12.9, 8.5, 14.9, 10.6, 23.2, 14.8, 9.7, 11.4, 10.
 ↪7]
})
```

a) Implement linear regression using Python to build the predictive model.

[158]:
```
# Linear regression model
x = df_2[['TV', 'radio', 'newspaper']]
y = df_2['sales']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,␣
 ↪random_state=42)
model2 = LinearRegression().fit(x_train, y_train)
```

b) Evaluate the model's performance using appropriate metrics such as mean squared error (MSE), mean absolute error (MAE), and R-squared score.

[159]:
```
# Model evaluation
y_pred = model2.predict(x_test)
mae = mean_absolute_error(y_test, y_pred)
```

```python
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("Linear Regression Model Evaluation")
print(f"Mean Squared Error: {mse}")
print(f"Mean Absolute Error: {mae}")
print(f"R^2 Score: {r2}")
```

```
Linear Regression Model Evaluation
Mean Squared Error: 7.763420880997777
Mean Absolute Error: 2.458841446779787
R^2 Score: 0.7181789666744991
```

Given our metrics, we can see that in relation to scale of our data, our MSE and MAE are fair, but not good. Our R^2 score also shows we can explain about 71.8% of the variance in the data, which we would like to see higher.

Overall, we can say that our model has fair performance, but there are fairly good odds that a more advanced model would perform better for this data.

c) Discuss the significance of each advertising medium (TV, radio, and newspapers) in predicting the sales of the product based on the model coefficients.

[160]:
```python
# Coefficients
print("Linear Regression Coefficients:")
for i, col in enumerate(x.columns):
    print(f"{col}: {model2.coef_[i]:.4f}")
```

```
Linear Regression Coefficients:
TV: 0.0416
radio: 0.0514
newspaper: -0.0023
```

Our coefficents allow us to correlate which advertising medium has the strongest effect on predicting sales. In this case, newspaper has a negative effect on sales, meaning the more we advertise with it, the lower our sales will be proportionally, though by a small amount with its coefficient being -00023. TV and radio both increase our sales, with radio having a stronger impact of 0.05 vs TVs 0.04.

0.0.4 3) (40 points) Based on the research paper attached : "Bias-variance tradeoff in machine learning: Theoretical formulation and implications to structural engineering applications" by Guan, X., & Burton, H. (2022, December), In Structures (Vol. 46, pp. 17-30). Elsevier.

a) Summarize the bias-variance decomposition as explained in this paper : How do the authors define bias, variance, and noise in the context of supervised machine learning?

b) What are the three types of error patterns described in the paper? : Provide an example ML model for each pattern and explain how the patterns relate to model complexity.

c) Explain the purpose and formulation of the generalized bias-variance decomposition (Equation 1 & 2). How does this formulation help in understanding the trade-off?

d) What are the strengths and innovations of this paper in dealing with the bias-variance trade-off?

a) Bias-variance decomposition is explained in this paper as the breakdown of how error in supervised machine learning comes from. This is in the form of bias, which is error from the prediction in the model typically made by preconceptions when modeling which leads to underfitting, variance, which is the sensitivity of the data to changes in data which can lead to model overfitting, and noise, which is the randomness that cannot be eliminated from unknown or immeasurable sources.

b) The three types of error patterns described in the paper are bell-shaped, double-descent, and monotonically decreasing. Bell-shaped patterns are observed in classic models, such as linear regression, and finds that our model complexity is determined by minimum testing error. Double-descent is observed in observed in deep learning, such as deep neural networks, and finds that the optimal model complexity seems to be of less import. Monotonically decreasing patterns are observed in some non-parametric models, such as random forest, and finds that optimal model complexity is subjective. This can be seen visually for each pattern in figure 4.

c) The purpose of the first equation given in the paper is to show how expected loss for ML models come from three sources: noise, bias, and variance. In practice, there is two primary sources, bias and variance, since it is known noise is difficult to estimate and therefore assumed 0 when discussing loss decomposition. These second equation expands on this source knowledge and demonstrate how paradoxical it can be in trying to reduce high bias and high variance together. This conflict between the two main sources of loss, bias and variance, and how reducing one must increase the other, highlights the bias-variance trade-off.

d) The biggest innovation in this paper is the implementation of the objective function, which is used to select the optimal model complexity. This formula, shown in equation 5, can help to avoid models that both underfit and overfit while still focusing on reducing error. A strength of this approach that is highlighted is for its implementation with modern machine learning models compared to the prior use of simply reducing error, which only worked well for bell-shaped error patterns. Another strength is that it is able to be used for multi-dimensional parameters. This is shown within table 3, where they emphasize the robustness of their method in regards to the effect of the weight parameter.