TASK 1

The manual approach is more efficient for the given task: "sort a list of dictionaries by a specific key." It is concise and straightforward, utilizing the basic sorted() function with a lambda. It has low overhead and clearly communicates its intention.

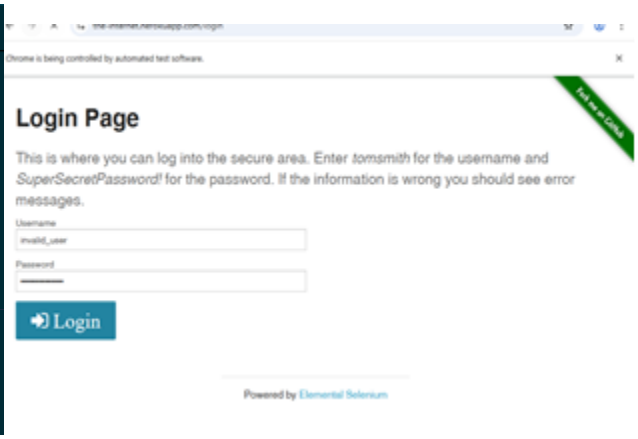
The AI-recommended code is more robust and feature-rich. It has optional arguments for reverse sorting and managing missing keys with a default value. This makes it much more reusable and less vulnerable to runtime problems in real-world applications where data may be inconsistent.

In terms of raw performance for the specific task, the manual version is marginally faster due to its simplicity. However, the AI version's additional safety checks (such as dict.get()) have a negligible performance impact for the majority of usage scenarios. The AI version is ultimately more useful for production code because to its error handling and flexibility, making it the more valued solution despite being slightly more difficult.

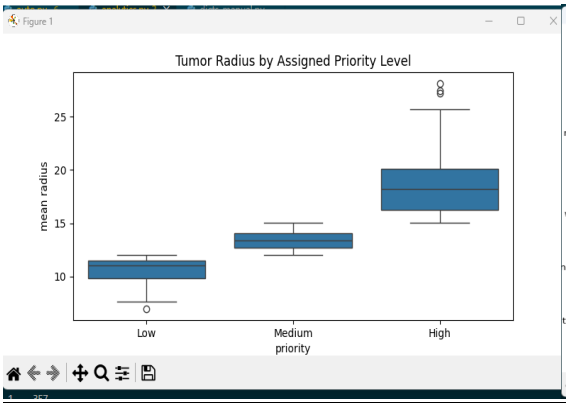
TASK 2(auto)

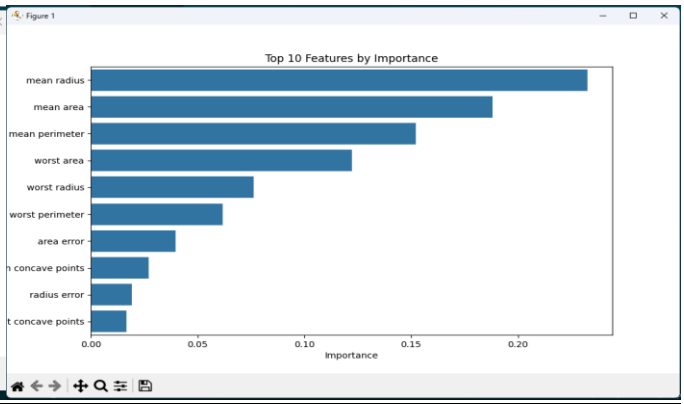
AI dramatically improves test coverage over manual testing by automating the generation, execution, and maintenance of test cases. AI-powered solutions can intelligently recognize UI elements, create tests for complex user flows, and suggest edge situations that a manual tester may overlook. They can automatically modify locators as the UI changes, reducing script breakage.

Furthermore, AI can run a large number of test combinations (for example, multiple browsers, screen sizes, and data inputs) far faster and more consistently than humans. It excels at regression testing, which ensures that new code does not break current functionality within the application. This scalability and robustness enable teams to have greater confidence in software quality and detect errors earlier in the development cycle, which is typically impossible with traditional manual testing methods.



TASK 3(analytics)






```
Dataset shape: (569, 31)

Class distribution (original):
0    212
1    357
Name: count, dtype: int64

Priority distribution (created):
priority
Low       171
Medium    227
High      171
Name: count, dtype: int64

Training set size: 455
Test set size: 114

=== MODEL PERFORMANCE ===
Accuracy : 1.0000
F1-Score : 1.0000

Detailed Classification Report:
              precision    recall  f1-score   support

        High       1.00      1.00      1.00        34
         Low       1.00      1.00      1.00        34
      Medium       1.00      1.00      1.00        46

    accuracy                           1.00       114
   macro avg       1.00      1.00      1.00       114
weighted avg       1.00      1.00      1.00       114
```