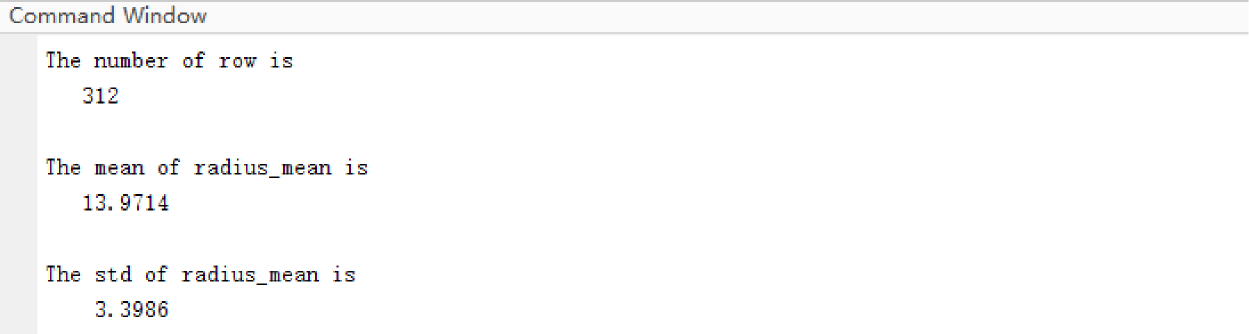
FAI CW2 Report

1 (a)

- Screenshot (by running my script “**Q1ab_main**”):



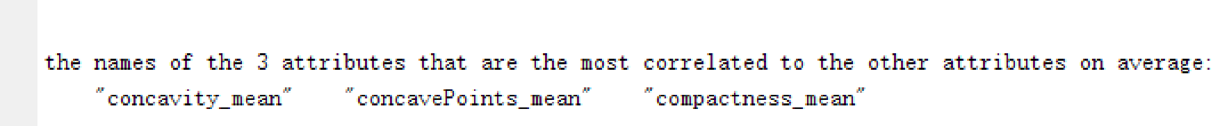
Command Window

```
The number of row is
   312

The mean of radius_mean is
   13.9714

The std of radius_mean is
    3.3986
```

- Answer:
  - For question: How many rows do you have in your training dataset?
    There are **312** rows in training dataset.
  - For question: What is the mean and std for radius_mean?
    The mean for radius_mean is **13.9714**, the std for radius_mean is **3.3986**.

1 (b)

- Screenshot (by running my script “**Q1ab_main**”):

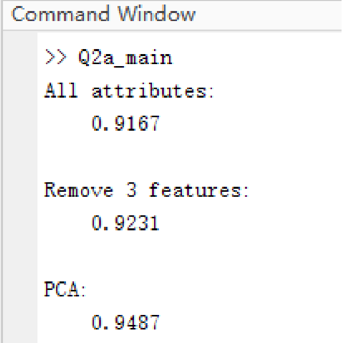


```
the names of the 3 attributes that are the most correlated to the other attributes on average:
    "concavity_mean"    "concavePoints_mean"    "compactness_mean"
```

- Answer:
  - For question: What are the names of the 3 attributes that are the most correlated to the other attributes on average?
    They are **“concavity_mean”, “concavePoints_mean”, “compactness_mean”**.

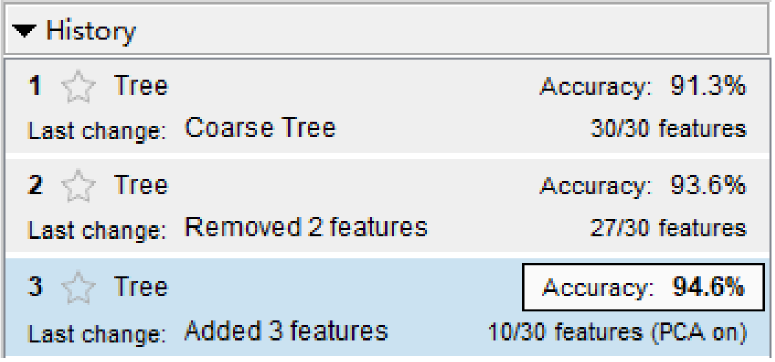
2 (a)

- Screenshot (by running my script “**Q2a_main**” and in Classification Learner App):



Command Window

```
>> Q2a_main
All attributes:
    0.9167

Remove 3 features:
    0.9231

PCA:
    0.9487
```



1 Tree Accuracy: 91.3%
Last change: Coarse Tree 30/30 features

2 Tree Accuracy: 93.6%
Last change: Removed 2 features 27/30 features

3 Tree Accuracy: 94.6%
Last change: Added 3 features 10/30 features (PCA on)

- Answer:
  - For question: Which case yields the best accuracy?
    Apply **PCA** (a dimensionality reduction technique) with the default options using all the attributes in inputs. Its accuracy is 94.87% as shown above by running my script.
  - For question: Which case yields the second-best accuracy?

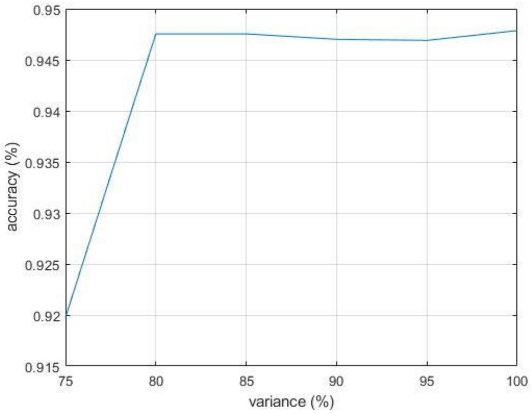
Use all the attributes in inputs, **except the 3 most correlated ones** that are found previously. Its accuracy is 92.31% as shown above by running my script.

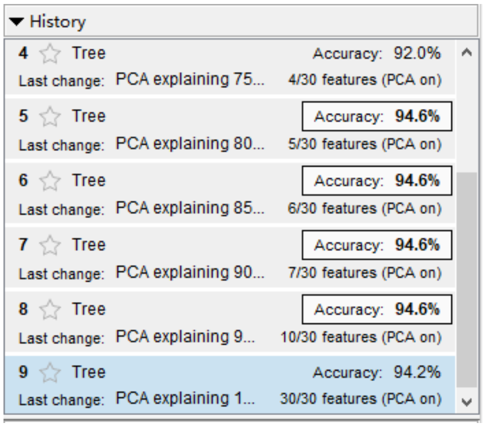
- For question: Can you provide an explanation for these results?
  Typically, the classification accuracy will increase by increasing the number of features for a fixed-size training dataset. However, to some extent, as the number of features continue to increase, classification accuracy can start to decrease. The reason might be overfitting and the lack of samples for the excess of features. That is, the size of training dataset is so small that it cannot provide enough information for all the features in the model training. Therefore, for PCA, reducing some unimportant features, namely dimensionality reduction or feature extraction, can increase the accuracy by reducing the impact of the excess of unimportant features. Also, the features that PCA select are the best features for data training, so the accuracy of PCA is the highest among all the case yields.
  If some features have relatively high correlation with other features, it means that they highly depend on other features. As they vary with other features, these features cannot provide much useful information for the data training. Hence, by removing 3 useless and interferential features, the classification accuracy of this is higher than that with all attributes.
- For question: How many features are used when PCA is turned on?
  10 features are used when PCA is turned on.

2 (b)

- Screenshot (by running my script "**Q2b_main**" and in Classification Learner App):

Attention: the script "Q2b_main" takes minutes to run, please be patient!





4 ☆ Tree Accuracy: 92.0%
Last change: PCA explaining 75... 4/30 features (PCA on)

5 ☆ Tree Accuracy: **94.6%**
Last change: PCA explaining 80... 5/30 features (PCA on)

6 ☆ Tree Accuracy: **94.6%**
Last change: PCA explaining 85... 6/30 features (PCA on)

7 ☆ Tree Accuracy: **94.6%**
Last change: PCA explaining 90... 7/30 features (PCA on)

8 ☆ Tree Accuracy: **94.6%**
Last change: PCA explaining 9... 10/30 features (PCA on)

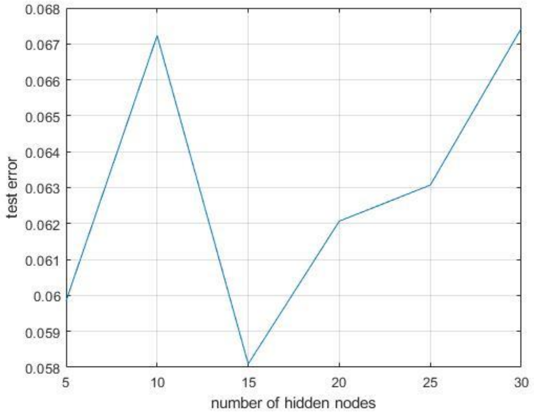
9 ☆ Tree Accuracy: 94.2%
Last change: PCA explaining 1... 30/30 features (PCA on)

- Answer:
  - Comment my plot:
    y label is the accuracy obtained after training, x label is values for the percent of variance explained in PCA (75%, 80%, … 95%, 100%).
    In my script, each function that is generated by the App is run for 30 times. As shown in above image, all the accuracies are larger than 90%. The accuracy against 75% of variance is the lowest, while accuracies against other variance are quite similar and higher in contrast.
    PCA explaining **75%** variance mentions only **4** dimensions, **80%** mentions **5** dimensions, **85%** mentions **6** dimensions, **90%** mentions **7** dimensions, **95%** mentions **10** dimensions, **100%** mentions **30** dimensions.
    This indicates that, for this given dataset, 5 features that are selected by PCA are right enough to get highest accuracy, whilst over 10 features can possibly decrease the accuracy.

The reason is declared in Answer for Q2(a) that, for a fixed training dataset, accuracy can increase because of the increase of number of dimensions. However, to some points, the accuracy will decrease as too many features can lead to overfitting and the dataset is too small to provide necessary samples.

2 (c)

- Screenshot (by running my script "**Q2c_main**"):



- Answer:
  - Comment my plot:
    y label is test errors obtained with a trained neural network which mean the average of the absolute value of the difference between actual outcomes of the trained neural network and targets.
    In my script, each function that is generated by the App is run for 30 times. The test errors are basically less than 0.07 and larger than 0.05 most of the time when I run my script to plot, which are quite small. However, **each time the plot varies**. I think the trained neural network with **15** hidden nodes is the best.
    The reason might be that the best size of hidden layer lies between the size of input layer and output layers. It can be optimal when the number of hidden layer is equal to 1 and the number of hidden nodes in hidden layer is the average of nodes in input and output layer. The number of nodes in input layer is equal to the number of features, namely 30 in this training dataset, while the number of nodes in output layer is 1. Therefore, theoretically, the best hidden nodes in hidden layer is 15.5, which can be rounded to 15.