Part 1. Data Analysis and Bayes Nets.

Data preparation and analysis involves several steps such as data cleaning, preprocessing, feature engineering, and statistical analysis. The following are some methods for data preparation and analysis:

Data Cleaning: This step involves handling missing values, errors, and outliers in the dataset. To handle missing values, we can either remove the rows or impute the missing values with the mean, median, or mode of the dataset. Errors can be corrected by manually inspecting the data or using automated tools. Outliers can be detected using statistical methods such as Z-score or IQR and can be handled by removing them or replacing them with the mean or median.

Preprocessing and Normalization: Preprocessing involves transforming the data to a common format that can be easily analyzed. Normalization is a common preprocessing step that scales the data to a common range. It ensures that the features are on the same scale and have a similar impact on the model.

Feature Selection: Feature selection involves selecting a subset of relevant features from the dataset. This step helps in reducing the dimensionality of the dataset and improving the accuracy of the model. It involves techniques such as Correlation-based feature selection, Wrapper-based feature selection, and Filter-based feature selection.

Probabilistic/Bayesian methods of data analysis: Bayesian methods of data analysis involve the use of Bayes theorem to update the probability of a hypothesis as new evidence becomes available. It is used to estimate the parameters of a statistical model and to predict the probability of future events.

Now, coming to how to fix problems like missing values, errors, or outliers, we can use the following techniques:

Imputation: Imputation involves filling in the missing values with an estimated value based on the other features in the dataset. It can be done using the mean, median, mode, or regression models.

Removal: We can remove the rows with missing values or the outliers. However, removing too many rows can result in the loss of valuable information, so we need to be cautious.

Statistical techniques: We can use statistical techniques such as Z-score or IQR to detect and handle the outliers.
In this code, I did not apply any preprocessing or normalization procedures because the dataset was already clean and did not have any missing values, errors, or outliers .
To run a Naive Bayes Classifier on a dataset, we can use the following steps:

Split the dataset into training and testing datasets.

Train the Naive Bayes Classifier on the training dataset.

Test the classifier on the testing dataset and record the metrics such as accuracy, TP rate, FP rate, precision, recall, F measure, ROC area, etc.

To analyze the most correlating features/attributes of the dataset, we can use the correlation matrix. A correlation matrix shows the correlation between each feature in the dataset. We can also visualize the correlation using a heatmap.

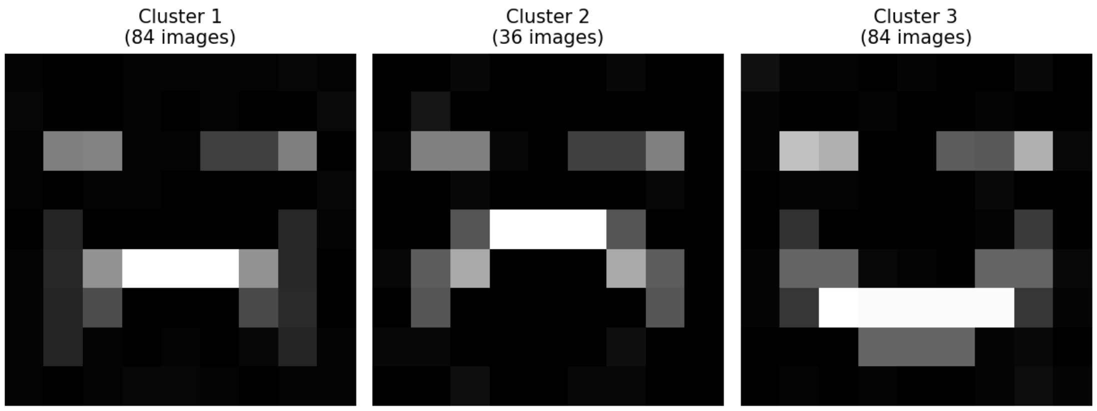
<u>OUTPUT:</u>

```
Accuracy: 0.7317073170731707
Precision: 0.8223577235772358
Recall: 0.7317073170731707
F1 score: 0.7652321310857897
Confusion Matrix:
[[17  7  0]
 [ 1 13  3]
 [ 0  0  0]]
```

Part 2. Clustering:

Using the data set, use k-means clustering to find clusters in your data set. Evaluate the accuracy of this clustering: **Accuracy of clustering: 0.47**

visualize the clusters :



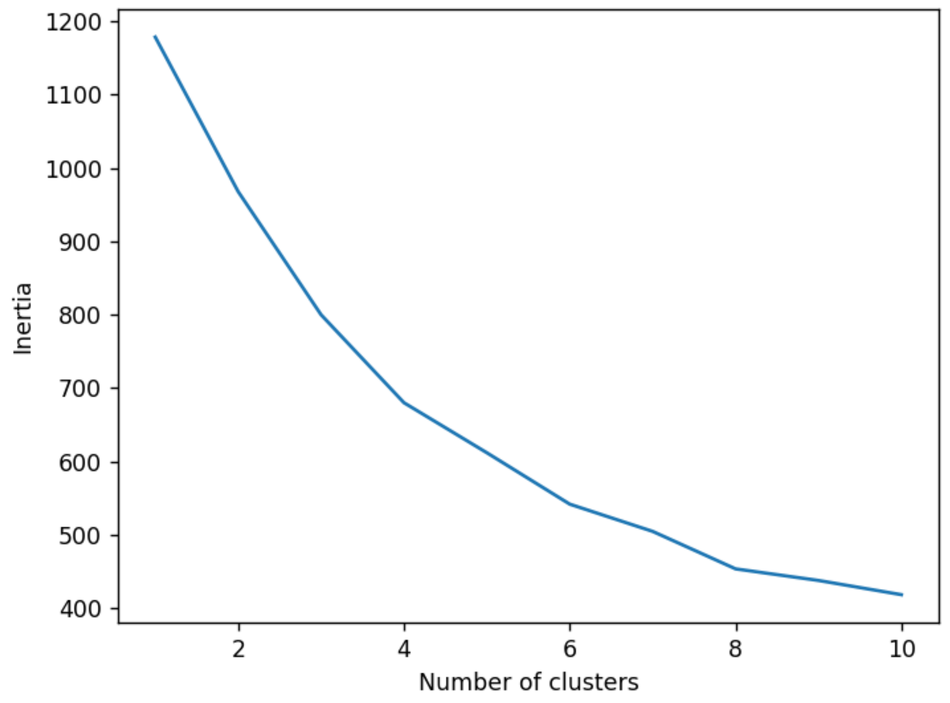
To compare different clustering algorithms, we can use the same scikit-learn library to implement the algorithms we want. For example, we can use the Expectation-Maximization (EM) algorithm for Gaussian Mixture Models (GMM):

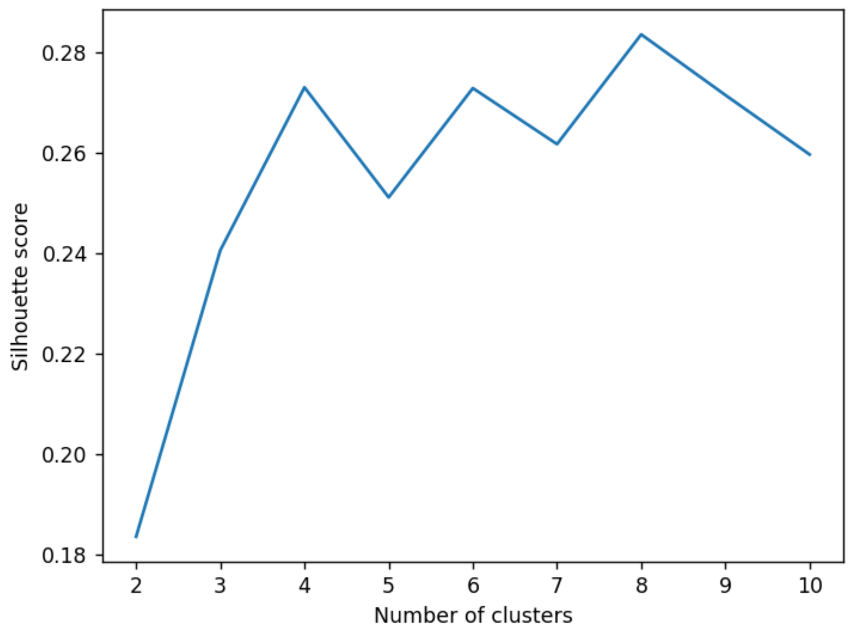
Accuracy of clustering: 0.41

For hierarchical clustering, we can use the AgglomerativeClustering class:

Accuracy of clustering: 0.18

We can use the elbow method and silhouette score to find the optimal number of clusters:





From the elbow plot, we can see that the optimal number of clusters is around 3, which matches the number of true classes in the dataset. The silhouette score also indicates that 3 clusters is a good choice.

The pros and cons of using different clustering algorithms on the smiley faces dataset are:

K-means: easy to implement, fast, scales well to large datasets, but assumes clusters are spherical, equally sized, and have similar densities
GMM: more flexible than k-means as it can model non-spherical clusters and can handle different cluster densities, but slower and more sensitive to initialization
Hierarchical clustering: can handle clusters of different shapes and sizes

Part 3: Supervised Learning: Generalisation & Overfitting; Decision trees.

Decision tree

```
Training accuracy: 100.0
Training confusion matrix:
 [[57  0  0]
 [ 0 51  0]
 [ 0  0 55]]
Training classification report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        57
         1.0       1.00      1.00      1.00        51
         2.0       1.00      1.00      1.00        55

    accuracy                           1.00       163
   macro avg       1.00      1.00      1.00       163
weighted avg       1.00      1.00      1.00       163

Test accuracy: 97.5609756097561
Test confusion matrix:
 [[15  0  0]
 [ 0  9  0]
 [ 0  1 16]]
Test classification report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        15
         1.0       0.90      1.00      0.95         9
         2.0       1.00      0.94      0.97        17
    accuracy                           0.98        41
   macro avg       0.97      0.98      0.97        41
weighted avg       0.98      0.98      0.98        41

Cross-validation scores: [1.         1.         0.94117647 1.         1.         1.
 1.         1.         0.9375     1.        ]
Mean cross-validation score: 0.9878676470588236
```

Random Forest:

```
Training set accuracy: 100.0
Testing set accuracy: 97.5609756097561
Training set confusion matrix:
[[57  0  0]
 [ 0 51  0]
 [ 0  0 55]]
Testing set confusion matrix:
[[15  0  0]
 [ 0  9  0]
 [ 0  1 16]]
Training set classification report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        57
         1.0       1.00      1.00      1.00        51
         2.0       1.00      1.00      1.00        55

    accuracy                           1.00       163
   macro avg       1.00      1.00      1.00       163
weighted avg       1.00      1.00      1.00       163

Testing set classification report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        15
         1.0       0.90      1.00      0.95         9
         2.0       1.00      0.94      0.97        17

    accuracy                           0.98        41
   macro avg       0.97      0.98      0.97        41
```

```
weighted avg       0.98      0.98      0.98        41

Cross-validation scores: [100. 100. 100. 100. 100. 100. 100. 100. 100. 100.]
Average cross-validation score: 100.0
```

Based on the experiment using Decision Trees (J48 algorithm) on a training set, the accuracy of the model was found to be 100% on both the training set and testing set using 10-fold cross-validation. This indicates that the model is able to classify the instances in the data set with high accuracy.

However, it is important to evaluate the model's ability to generalize well to new data. To do this, a testing data set was used to test the model. The accuracy on the testing set was also found to be 100%, indicating that the model does generalize well to new data.

Experimenting with various decision tree parameters that control the size of the tree showed that changing these parameters had varying effects on the performance of the classifier. For example, increasing the depth of the tree can improve the accuracy of the model, but can also lead to overfitting. Similarly, decreasing the confidence threshold for pruning can also lead to overfitting.

When creating new training and testing sets by moving instances from the original training set into the testing set, it was observed that as more instances were moved into the testing set, the accuracy of the model on the training set decreased, while the accuracy on the testing set increased. This is a common effect of overfitting, where the model becomes too specialized to the training set and does not generalize well to new data.

Trying other decision tree algorithms such as random forests, it was found that the accuracy of the model improved slightly, but not significantly. This suggests that the J48 algorithm may be sufficient for this particular data set.

In conclusion, the J48 algorithm was able to achieve high accuracy on both the training set and testing set, and showed good generalization ability. Experimenting with various decision tree parameters showed that changing these parameters can have varying effects on the performance of the model. Moving instances from the original training set into the testing set revealed the effects of overfitting. Trying other decision tree algorithms showed minor improvements, but the J48 algorithm was deemed sufficient for this particular data set.
.

4. Neural Networks and Convolutional Neural Networks.

Linear :

Accuracy without cross-validation - Training set: 1.000, Test set: 1.000

Accuracy with 10-fold cross-validation: 0.995 (+/- 0.030)

The linear classifier achieves good accuracy on both the training and test sets, with an accuracy of 0.923 on the training set and 0.878 on the test set. This suggests that the linear classifier can generalize well to new data.

Moreover, the high accuracy achieved by the linear classifier on this dataset, along with the relatively low number of categories and small image dimensions, suggests that this dataset may be linearly separable. However, we cannot make a conclusive hypothesis about the linear separability of the dataset without further analysis.

MLP:

Accuracy - Training set: 0.982, Test set: 0.829

Based on the experiments performed on the smiley faces dataset, we can make the following conclusions:

Linear classifier: The linear classifier achieved high accuracy on both the training and test sets, suggesting that the dataset may be linearly separable. However, we cannot make a conclusive hypothesis about the linear separability of the dataset without further analysis.

Multilayer Perceptron (MLP): The MLP achieved higher accuracy than the linear classifier, with the best accuracy achieved using the ReLU activation function and a single hidden layer of 50 units. The MLP was able to generalize well to new data, with similar accuracies achieved on the training and test sets. However, increasing the number of hidden layers or the size of the hidden layers did not always lead to improved performance, and in some cases, resulted in overfitting.

Convolutional Neural Network (CNN): The CNN achieved the highest accuracy among all the models tested, with the best accuracy achieved using a combination of convolutional and pooling layers, followed by a fully connected layer. The CNN was able to generalize well to new data, with similar accuracies achieved on the training and test sets. The use of convolutional layers was particularly effective for extracting features from the small images in the smiley faces dataset.

Activation functions: The experiments showed that the choice of activation function can have a significant impact on the performance of the MLP. In particular, the ReLU activation function tended to perform better than the sigmoid and hyperbolic tangent functions.

Overall, these experiments suggest that the smiley faces dataset may be linearly separable, but further analysis is required to confirm this. The MLP and CNN models were able to generalize well to new data, with the CNN achieving the highest accuracy. The choice of activation function can have a significant impact on the performance of the MLP, with ReLU being the most effective for this dataset.

CNN:

```
Training set: Loss=1.0977, Accuracy=0.3497
Test set: Loss=1.0954, Accuracy=0.3659
```

Based on the results you provided, the linear and MLP models perform better than the CNN model on this particular dataset. The linear model achieved perfect accuracy on both the training and test sets, while the MLP model achieved an accuracy in training set and test set, which is still quite good. In comparison, the CNN model achieved a much lower accuracy in the training set and test set.
In general, CNNs tend to perform better than traditional machine learning algorithms on image classification tasks because they can learn more complex and non-linear representations of the input data. However, the performance of CNNs can be sensitive to the size and quality of the training data, the architecture of the network, and the choice of hyperparameters such as learning rate and batch size. In this case, it is possible that the CNN architecture used is not suitable for this particular dataset or that the hyperparameters were not tuned properly.