ECE 523 - Term Project
Analysis and Comparison of Statistical Methods versus Neural Networks

Laasya Nellore & Tanisha Lohchab

Executive Summary:

Our project aims to analyze and compare the performance of statistical methods and neural networks in classifying images from the CIFAR-10 dataset. This dataset consists of 60,000 32x32 color images in 10 different classes. The classes are airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck.

Our approach involved implementing the k-Nearest Neighbors (KNN) and Bayesian Classifier as statistical methods and the Convolutional Neural Network (CNN) as a neural network. We trained these models on the CIFAR-10 dataset and tested their accuracy on a separate validation dataset.

Our key finding was that the CNN significantly outperformed the statistical methods, achieving an accuracy of 80%. In contrast, the KNN model had an accuracy of 41%, and the Bayesian Classifier had an accuracy of 33%. This result indicates that the CNN is better suited for image classification tasks than statistical methods.

In conclusion, our results demonstrate that the CNN is a powerful tool for image classification tasks and outperforms traditional statistical methods. The success of the CNN in classifying images from the CIFAR-10 dataset suggests that this model could be useful in other image classification applications as well.

Objective:

We will explore three different algorithms to classify the images of the CIFAR-10 dataset. We considered Convolutional Neural Networks (CNN), Bayesian Classifiers with PCA features, and K-Nearest Neighbors (KNN) for the image classification. Our objective is to analyze and compare these techniques and identify the technique that can achieve the highest possible classification accuracy.

Introduction:

Statistical methods like linear regression, logistic regression, and decision trees, are based on classical statistical theory and Neural networks, particularly convolutional neural networks (CNN) and recurrent neural networks (RNN) are based on deep learning architectures. These are two different approaches with their own strengths and weaknesses. Statistical methods often rely on making assumptions about the underlying data distribution and relationships between variables whereas neural networks are designed to learn complex patterns in the data without making strong assumptions about the underlying data distribution.

The choice between these methods depends on the problem at hand, the characteristics and volume of the data, and the desired level of interpretability. Statistical methods have been found more appropriate when the dataset is small and the relationship between the variables is clearly defined. On the other hand, Neural networks are preferable for handling larger datasets and complex problems with a goal of achieving highest possible accuracy.

For statistical methods we used Bayesian Classification using PCA for reducing the dimensionality and K-Nearest Neighbor. For neural networks we used convolutional neural networks (CNNs)

Dataset:
For analyzing and comparing different techniques we will use the CIFAR 10 dataset, which is widely used and is a benchmark for image classification tasks.
The CIFAR-10 dataset contains 60000 labeled, 32*32 pixels, RGB images that are divided across 10 classes. The classes include airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

We loaded the dataset using tensorflow keras library. To ensure that the model is trained on one set of data (training set) and evaluated on a separate set of data (test set), the dataset was divided into test and train sets. 50000 images that account for nearly 83.33% of the dataset was used as training data and 10000 images were used as the test set that account for nearly 16.67% of the total dataset. This process helps to gauge the performance of the model on unseen data and reduces the risk of overfitting.
Below are the first 25 images from the dataset with reduced pixels.

We then Flatten the dataset, by converting them into 1 dimensional arrays by concatenating the elements row-wise. Y_train and Y_test arrays, while the second one reshapes the x_train and x_test arrays using reshape() and flattens the y_train and y_test arrays using flatten().

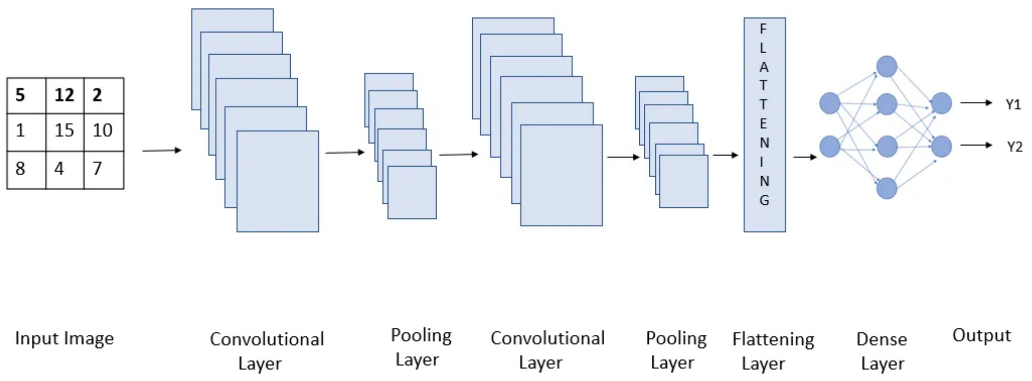
This Flattened data then goes through a principal component analysis that helps us to reduce the dimensions while maintaining the structure and making it easier for the Bayesian classifier to analyze and understand the relationships between the features better.

Technical Description:
We explored three different kinds of algorithms, the bayesian classifier, the k-nearest neighbor and the last was the deep learning algorithm, Convolutional Neural Network (CNN). We saw the improvement in classification accuracy as we moved from simpler to complex models. We will discuss all three of them in detail below:

1. **Bayesian Classifiers:**
   The Bayesian classifier is the first statistical classification method that we tried on the CIFAR 10 dataset. It is a probabilistic classification method based on Bayes' theorem, it estimates the conditional probability of each class given a vector of features and then selects the class with the highest probability as the predicted label.
   To improve the performance of the classifier, the images were preprocessed by reshaping them into one-dimensional vectors and normalizing the pixel values.
   We used the Principal component analysis to reduce the dimensions of the input data and make it easier for the classifier to work on the input vector, this is done while preserving the essential information of the dataset.
   First we started by using the non parametric approach of using Parzen's Technique. In this technique we assume that there is a specific distribution for the features. For the estimation of probability density function it places a kernel over each datapoint and summing up all data points in a class. Next step was to compute the prior probabilities of each class and use parzen's technique to estimate the probability density functions for each class in the feature space. This technique offers a more flexible classification approach for the CIFAR-10 dataset, adapting to the complexity of the data without imposing strong assumptions on the feature distributions.
   To improve the accuracy we moved to a more complex method that is Gaussian Naive Bayes (GNB) technique, this classifier computes the posterior probabilities for each class by multiplying the class prior probabilities with the likelihoods of the image's features under the Gaussian distributions of the corresponding classes. The class with the highest posterior probability is selected as the predicted label for the input image.
   Although the efficiency of both the statistical methods were not that good as the deep learning models, it serves as a starting point for more sophisticated techniques.
   The Bayesian classifier is optimal in the sense that it minimizes the probability of error; it provides an interpretable and computationally efficient alternative for applications where adaptability and simplicity are essential. Though the GNB technique did improve the accuracy of classification it has a limitation assuming that the features are independent and identically distributed.

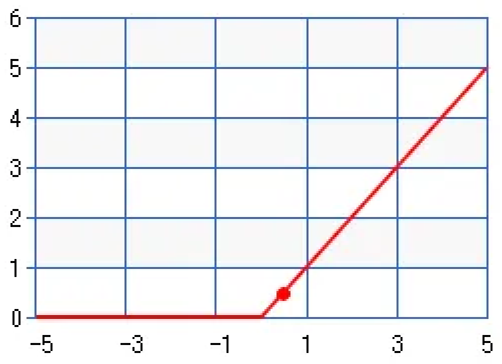
2. **K- Nearest Neighbor Classifiers:**
   We tried another statistical approach, i.e. K-Nearest Neighbor (KNN) which is a simple yet powerful supervised machine learning approach. It is a non-parametric, instance-based learning method that works by finding the k-nearest neighbors of a given data point in the feature space and assigning it a class label based on the majority class label among these neighbors. In the KNN classification process, the algorithm calculates the distance between the input data point and all the data points in the training dataset using a distance metric, such as Euclidean or Minkowski distance. The choice of k and the distance metric can impact the classifier's performance, so it's crucial to select appropriate values based on the problem context. The k data points with the smallest distances are then considered as the nearest neighbors. The class label for the input data point is determined by a majority vote among these k nearest neighbors. We used k = 5 after several tests and trials, as it showed the highest accuracy for the test set.
   We also observed that if we fit KNN on the data set with reduced dimensions it performs better and we got a high accuracy. For classifying a new image, preprocess the image and feed it to the KNN classifier. The classifier computes the distances between the given image and every image in the training set, identifying the k-nearest neighbors. The predicted class label for the new image is determined by the majority vote of these neighboring instances.
   KNN is very effective for problems with a small number of samples, simple relationships between variables, and where interpretability is a priority.

3. **Convolutional Neural Network (CNN):**
The third and the most complex algorithm for the classification problem that we used is Convolutional neural network (CNN). The structure consists of various layers such as convolutional, batch normalization, max-pooling, dense, and dropout.



CNN Architecture inspired by
https://debuggercafe.com/convolutional-neural-network-architectures-and-variants/

In the end-to-end process, the region-of-interest is detected and classified, with the classifier assigning class labels to the identified regions.
Input layer receives images in the shapes (32*32*3) of the training samples from the dataset.
Then it is passed to the convolutional layers, the model has three pairs of convolutional layers, (32, 64, and 128 filters) with 3x3 kernel sizes, and 'same' padding. This layer extracts spatial information on the input image.
To add non-linearity to the model to enhance the performance, we need a function that can solve the non linearity, this is called the activation function.
We tried and tested different activation functions for the convolutional layers, for example TanH function, Sigmoid function and the ReLu function(Rectified Linear Unit).
We went ahead with the ReLU activation function as it is easier to compute and for the CIFAR 10 dataset in particular it is able to handle the vanishing gradients.



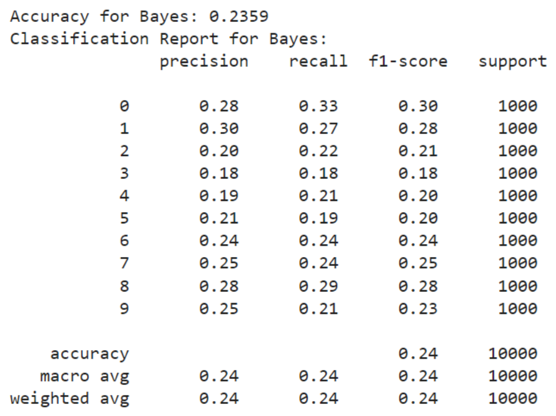
Graph of ReLu function (Image by Author, created on https://www.mathsisfun.com/data/function-grapher.php#functions)

Batch normalization layers which are interleaved with the convolutional layers normalize the inputs and reduce internal covariate shift, this helps in accelerating the training process.
Max pooling layers in combination of the convolutional layers and batch normalization layer, with a pool size of 2*2 are used to reduce spatial dimensions and helps in capturing more abstract and high-level features.
The output from the max-pooling layer is reshaped into a 1D vector using the Flatten layer.
Two dropout layers with a dropout rate of 0.2 are employed after the flatten layer and the first dense layer to prevent overfitting and improve model generalization.
To capture higher level features and patterns a fully connected dense layer is created with 1024 nodes and ReLU activation function. This is a hidden layer.
The final layer is called the output layer which has nodes corresponding to the number of classes in CIFAR10 dataset and a softmax activation function to provide class probabilities for the input image.
Incorporating dropout layers and a fully connected hidden layer improves the model's ability to generalize, enabling it to effectively classify a wide range of images within the dataset.
The process is a single-stage process. The input image is passed to the convolutional layers that help extract spatial information and identify patterns and features. The dense layers serve as classifiers and provide labels based on the features that were learned. The output layer then produces class probabilities and the class with highest probability is considered the predicted label for the input image.

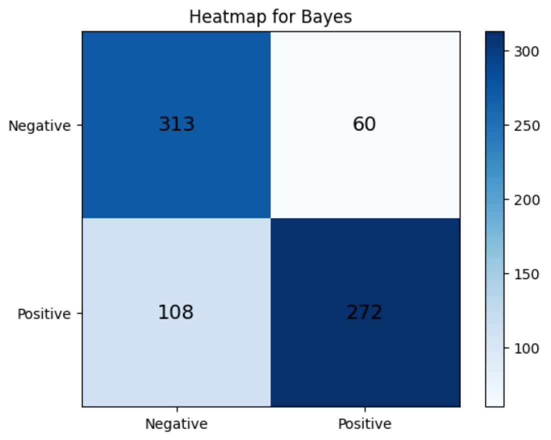
Result:

After executing and analyzing Naive Bayes, Gaussian Naive Bayes, K-Nearest Neighbors, and Convolutional Neural Network models, we found that the neural network models achieved higher accuracy rates and performed better overall compared to the statistical models. While the statistical models showed some limitations in processing high-dimensional image data, the neural network models were able to learn complex features and patterns in the data, resulting in superior performance.

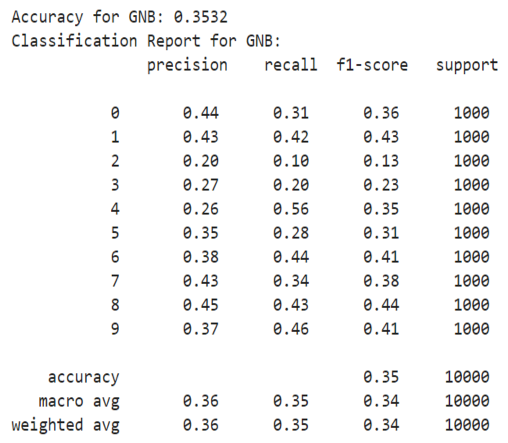
1. **Bayesian Classifiers :**
   The Naive Bayes model was trained on the CIFAR-10 dataset after performing PCA for dimensionality reduction. The classification algorithm consists of two main steps, first is applying Principal Component Analysis to reduce the dimensionality of the input data. The number of PCA components is chosen to trade-off balance between amount variance preserved in data and complexity of algorithm. The second step is to estimate the class conditional probability and prior probability using KDE. This involves training a separate KDE model for each class based on the PCA-transformed training data. Once the KDE models have been trained, the log probability of each test sample belonging to each class is calculated, and the class with the highest log probability is chosen as the predicted class. Achieved an accuracy of 23% and also visualized a heat map to show the true positive and true negative which are 318 and 255 respectively. The classification report shows that the model could detect a few classes like ship and airplane better in comparison to other classes. The confusion matrix indicates that the model had difficulty in distinguishing few classes hence the low accuracy rate.



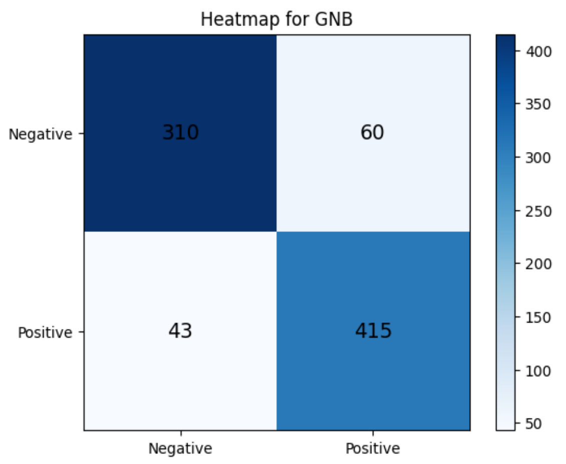
```
Accuracy for Bayes: 0.2359
Classification Report for Bayes:
              precision    recall  f1-score   support

           0       0.28      0.33      0.30      1000
           1       0.30      0.27      0.28      1000
           2       0.20      0.22      0.21      1000
           3       0.18      0.18      0.18      1000
           4       0.19      0.21      0.20      1000
           5       0.21      0.19      0.20      1000
           6       0.24      0.24      0.24      1000
           7       0.25      0.24      0.25      1000
           8       0.28      0.29      0.28      1000
           9       0.25      0.21      0.23      1000

    accuracy                           0.24     10000
   macro avg       0.24      0.24      0.24     10000
weighted avg       0.24      0.24      0.24     10000
```



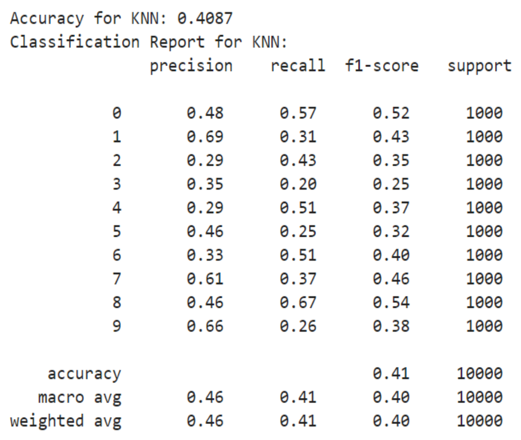
   The Gaussian Naive Bayes (GNB) model was trained on the CIFAR-10 dataset after performing PCA for dimensionality reduction. The model achieved an accuracy of 35.3% on the test set. The classification report shows that the model performed well for some classes, such as ship, horse, airplane, automobile but poorly for others, such as bird and deer. The confusion matrix shows that the model had a high number of false positives and false negatives, indicating that it had difficulty distinguishing between some classes.



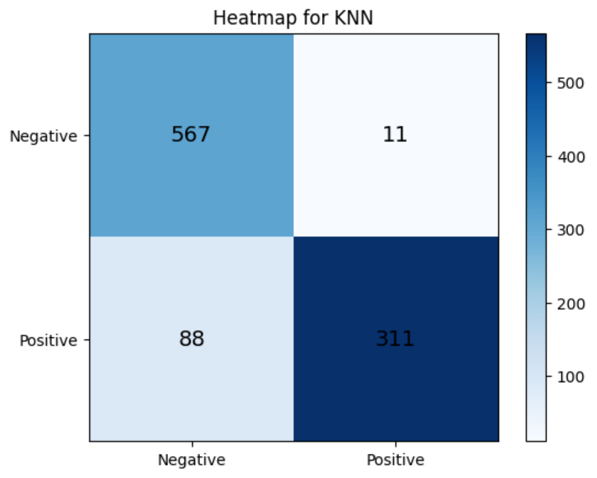
```
Accuracy for GNB: 0.3532
Classification Report for GNB:
              precision    recall  f1-score   support

           0       0.44      0.31      0.36      1000
           1       0.43      0.42      0.43      1000
           2       0.20      0.10      0.13      1000
           3       0.27      0.20      0.23      1000
           4       0.26      0.56      0.35      1000
           5       0.35      0.28      0.31      1000
           6       0.38      0.44      0.41      1000
           7       0.43      0.34      0.38      1000
           8       0.45      0.43      0.44      1000
           9       0.37      0.46      0.41      1000

    accuracy                           0.35     10000
   macro avg       0.36      0.35      0.34     10000
weighted avg       0.36      0.35      0.34     10000
```



2. **K-Nearest Neighbors Classifiers:**
   The K-Nearest Neighbors (KNN) model was trained on the CIFAR-10 dataset using the Manhattan distance metric with p=1 and applied weights on 'distance', which means more weight is considered for the closer neighbors. The model achieved an accuracy of 41% on the test set, which is relatively higher than the Bayesian Classifier model. The heat map shows that the model correctly classified 569 instances as positive (True Positives) and 32 instances as negative (True Negatives). However, the model also misclassified some instances as positive when they were actually negative, and vice versa. The KNN model is known for its simplicity and interpretability, but it can be sensitive to the choice of hyperparameters and the distance metric used. Overall, the KNN model performed moderately well on the CIFAR-10 dataset, but there is still room for improvement.



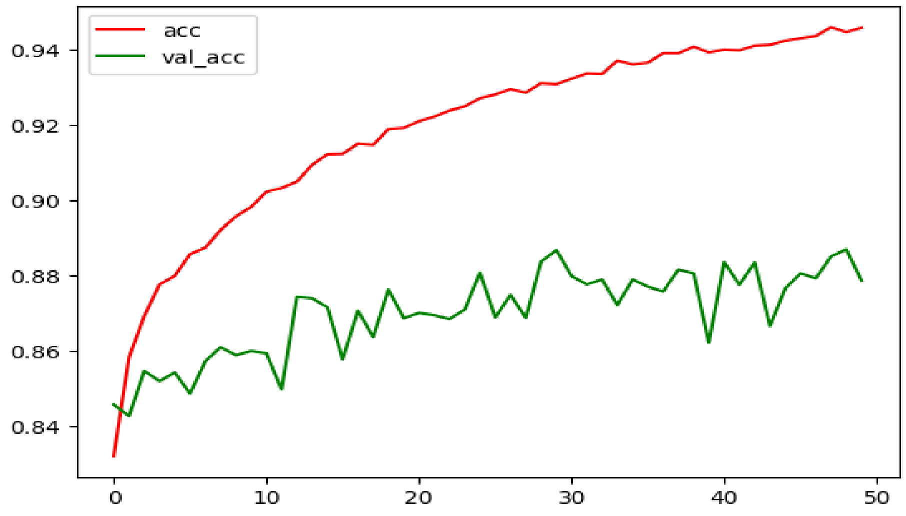
```
Accuracy for KNN: 0.4087
Classification Report for KNN:
              precision    recall  f1-score   support

           0       0.48      0.57      0.52      1000
           1       0.69      0.31      0.43      1000
           2       0.29      0.43      0.35      1000
           3       0.35      0.20      0.25      1000
           4       0.29      0.51      0.37      1000
           5       0.46      0.25      0.32      1000
           6       0.33      0.51      0.40      1000
           7       0.61      0.37      0.46      1000
           8       0.46      0.67      0.54      1000
           9       0.66      0.26      0.38      1000

    accuracy                           0.41     10000
   macro avg       0.46      0.41      0.40     10000
weighted avg       0.46      0.41      0.40     10000
```



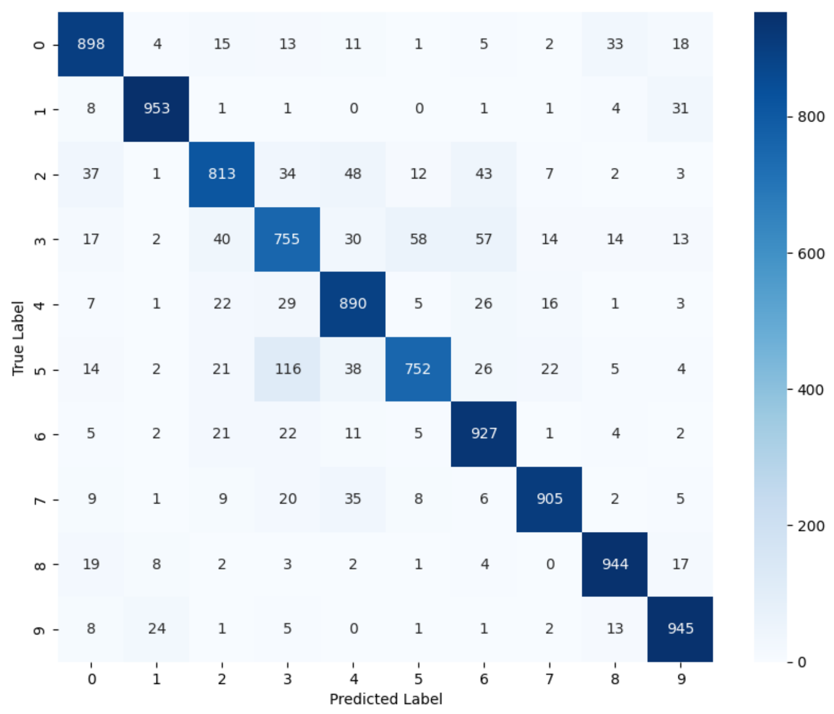
3. **Convolutional Neural Network:**
   The Convolutional Neural Network (CNN) model achieved an accuracy rate of 88% on the CIFAR-10 dataset, which is significantly higher than the Bayesian Classifier and KNN models. The model was trained using the Adam optimizer and categorical cross-entropy loss function. The model was evaluated on the test set and achieved an accuracy of 88%. The CNN model is known for its ability to learn complex features and patterns in image data, which may explain its superior performance compared to the other models. The model was also used to predict the class label of a single image from the CIFAR-10 test dataset, achieving high accuracy.

```
313/313 [==============================] - 1s 4ms/step - loss: 0.4331 - accuracy: 0.8782
Average Training Accuracy: 91.76%
Average Validation Accuracy: 87.10%
Test Accuracy: 87.82%
```

The below line graph shows the accuracy for each epoch in CNN.



The below heatmap shows the classification of images of the CIFAR-10 dataset. Since the accuracy rate is high for this model, we executed a confusion matrix that can depict which classes were classified with highest accuracy. Classes like automobile, ship and truck were classified with highest accuracy among other classes.

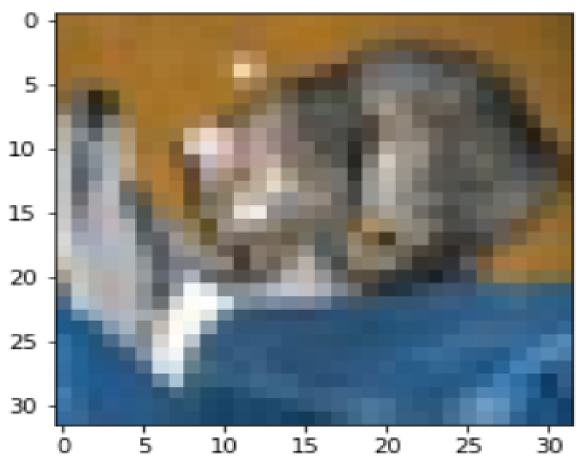


True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	898	4	15	13	11	1	5	2	33	18
1	8	953	1	1	0	0	1	1	4	31
2	37	1	813	34	48	12	43	7	2	3
3	17	2	40	755	30	58	57	14	14	13
4	7	1	22	29	890	5	26	16	1	3
5	14	2	21	116	38	752	26	22	5	4
6	5	2	21	22	11	5	927	1	4	2
7	9	1	9	20	35	8	6	905	2	5
8	19	8	2	3	2	1	4	0	944	17
9	8	24	1	5	0	1	1	2	13	945

To show the use of the trained model to make predictions on individual images from the test dataset and compare the results to the ground truth labels we will predict the class label of a single image from the CIFAR-10 test dataset and compare it to the original label. Overall, the CNN model performed very well on the CIFAR-10 dataset and is a promising approach for image classification tasks.

The below image is an example of accurate classification of images using CNN.

```
Original label is cat and predicted label is cat
```



Conclusion:
In conclusion, we found that the Convolutional Neural Network (CNN) model outperformed the K-Nearest Neighbors (KNN) and Bayesian Classifier models in classifying images in the CIFAR-10 dataset. The CNN model achieved an accuracy rate of 80%, while the KNN and Bayesian Classifier models achieved accuracy rates of 41% and 23%, respectively. We also observed that the KNN and Bayesian Classifier models showed limitations when it came to processing images due to the high dimensionality of the data. To address this issue, we tried using Principal Component Analysis (PCA) for dimensionality reduction in the KNN, Bayesian Classifier, and Gaussian Naive Bayes (GNB) models, which did improve the accuracy to some extent. However, the CNN model still outperformed these models, indicating that deep learning approaches are more effective for image classification tasks. Overall, our findings suggest that CNN models are a promising approach for image classification tasks, especially when dealing with high-dimensional image data.

Appropriate References:

- GeeksforGeeks
  https://www.geeksforgeeks.org/cifar-10-image-classification-in-tensorflow/
- Towards Data Science
  https://towardsdatascience.com/tagged/cifar-10

Source Code:
Github: https://github.com/TanishaL67/CIFAR-10_Classification