Multi-class Classification on CIFAR-100 subset: A comparative study of Logistic Regression, Gaussian Naive Bayes, Support Vector Machine and Random Forest Model

Giulia Bettuzzi

Università degli Studi di Verona

Master's Degree in Artificial Intelligence [LM-18]

Machine Learning Project

July 2025

Contents

Chapter 1

Motivation and Rationale

One of the central problems in Machine Learning is classification. The starting point is binary classification, in which the data belongs to exactly two classes, and then multiclass classification, in which the number of classes increases to three or more. In both cases, the input is a labeled dataset and the output is a classifier that should be able to predict the class to which the example provided belong to.

CIFAR-100 dataset is used in several studies as a benchmark to evaluate the ability of a model to distinguish a large number of classes.

In the following report, there is a focus on animals' subclasses of the dataset and a comparative analysis of five supervised classification models: Logistic Regression, Gaussian Naive Bayes, Support Vector Machine, Random Forest and K-Nearest Neighbours.

They are all evaluated using both the One-vs-One and One-vs-Rest approaches. The differences are assessed in terms of performance metrics and scalability.

Chapter 2

State of Art

Models employing neural networks (CNNs) are able to achieve the best results, in performance on the CIFAR-100 dataset, traditional models turn out to be better in terms of interpretation.
Among non-neural models, it comes up that One-vs-Rest are more practical and scalable than One-vs-One with cases involving large datasets.

Chapter 3

Objectives

The aim is to investigate a subset of the CIFAR-100 dataset and train several supervised Machine Learning models. By comparing the results obtained, conclusions can be drawn. The performance and precision of each are evaluated in relation to the others. The path is as follows:

1. Analyze animal subsets in CIFAR-100 dataset
2. Train different Machine Learning models
3. Compare the models
4. Draw conclusions.

Chapter 4

Methodology

To get a complete overview of the problem and of the techniques, the dataset, the libraries and the methods used during the training and testing phase are analyzed.

4.1 Dataset

The considered dataset, CIFAR-100 [1], is a subset of the Tiny Images Dataset [2] and consists of 60000 colored images of 32x32 pixels. Its importance is related to the fact that it contains a large number of labeled data. Due to its feature and structure it is widely used for Machine Learning and Computer Vision tasks. The 100 classes of CIFAR-100 are grouped into 20 superclasses. For each class, 500 training images and 100 test images are provided.
Here is the list of superclasses with classes in the CIFAR-100:

Superclass	Class
aquatic mammals	beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flower	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household fornitures	bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn mower, rocket, streetcar, tank, tractor

It is considered a subset of animals because of computational complexity issues and time taken to execute the code.

The animal subset is obtained by grouping the following superclasses:

- large carnivores
- large omnivores and herbivores
- medium-sized mammals

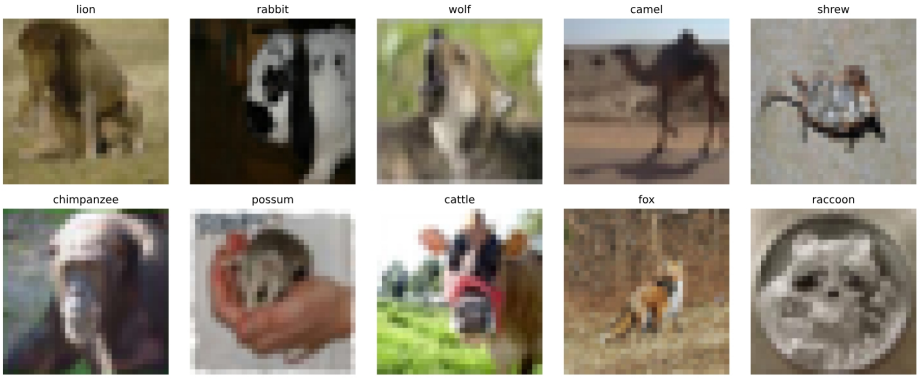


Figure 4.1: 10 random classes from CIFAR-100 animal subset.

- small mammals

After the reduction of the CIFAR-100 dataset:

→ Train set 10000 images

→ Test set 2000 images

A 2D t-Distributed Stochastic Neighbor Embedding space is used to display and map the dataset, it helps to visualize how the data are distruibuted in the plane
Data with similar features are represented as neighboring points: t-SNE preserves similarity. In addition, t-SNE allows the identification of potential distinct clusters within the given data.
The classes appear to be scattered and there is no immediate separation between them. The plot [4.2] shows that there are partial overlaps between the classes themselves.

4.2 Data Pre-processing

In order to optimize the model, data augmentation is applied. Random clipping, horizontal flipping, rotation and color jitter are also fitted to the train images. To ensure stable and efficient training, after preprocessing the pixel values of the images are scaled between [0, 1], then normalized. Data are flattened as the model is fed with a 1D input.

To standardize data, a pipeline is implemented. It helps to have a cleaner code, avoid data leaks and facilitate cross-validation.

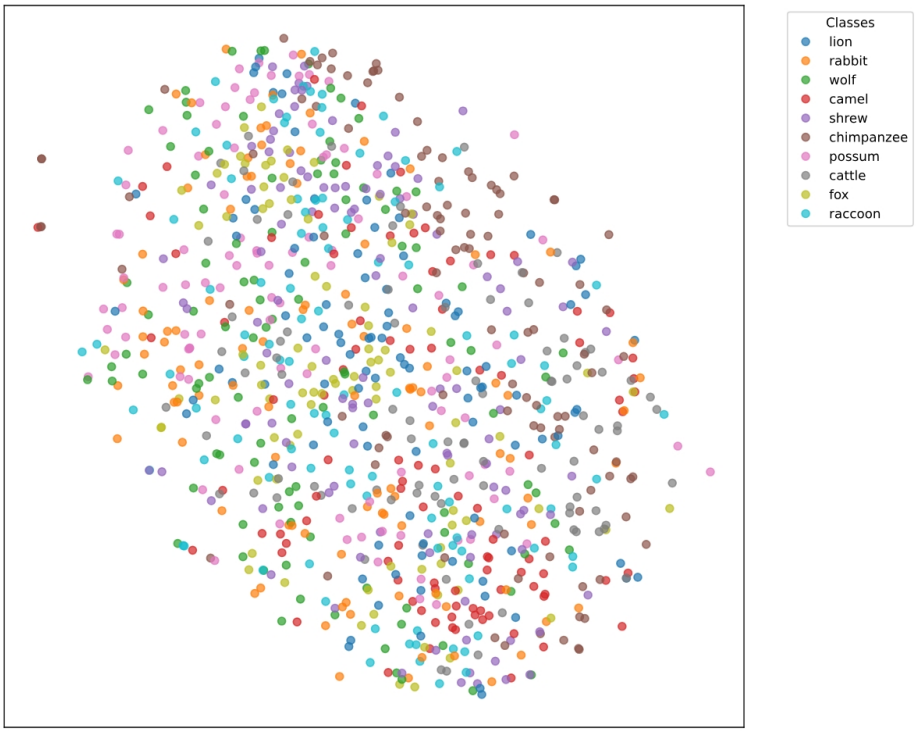


Figure 4.2: t-SNE visualizes a random subset of 10 classes of CIFAR-100 training set.

The dimensionality of the data is reduced but the 95% of the variance is preserved.
Finally, multi-class classification models are applied using One-vs-One and One-vs-Rest methods.

A grid of hyperparameters is enforced to the model defined by the pipeline, which through cross-validation allows the best evaluation.
Despite the high computational cost and time required for the analysis, especially due to the large size of the dataset, all combinations were explored.

4.3 Analytical Tools

In order to decide if the model is performing well, the listed metrics are evaluated:

- Accuracy
- Precision
- Recall

- F1-Score
- Confusion matrix.

While accuracy measures the goodness of the classifier, precision, recall and F1-score relies on the ability of the model to classify correctly the positive samples.
Indeed, accuracy deals with correctness of prediction, precision and recall with minimizing false positives and negatives respectively, and F1-score balances precision and recall.

The confusion matrix is used to valuate the performance of the classification models. The matrix shows the number of times a class is correctly or not predicted by the model.

4.4 Computational Tools

The classifier is built using Google Colab. A cloud-based free software that allows the user to write Python code, test and share it.
Slow performance in large datasets is highlighted.

The main libraries exploited are:

- pandas and numpy: for performing operations on data
- torchvision [3]: from which the dataset used for the analysis is retrieved
- matplotlib: for the creation of graphs
- seaborn: for statistical distributions
- skikit-learn: open-source library for Machine Learning.

4.5 Classification Model

For classification tasks with more than two classes the heuristic methods are:

- One-vs-Rest (OvR): splits the multi-class dataset into multiple binary classification problems. A binary classifier is then trained on each binary classification problem and prediction are made using the most confident model.

- One-vs-One (OvO): splits a multi-class classification problem into binary classification problems but the approach is that to compare one class versus every other class. The formula for calculating the total number of binary datasets and in turn the model is:

$$\frac{n\_classes(n\_classes - 1)}{2}$$

For the analysis traditional classifiers are:

- Logistic Regression
- Support Vector Machine
- K-Nearest Neighbors
- Gaussian Naive Bayes
- Random Forest.

4.5.1 Logistic Regression

Logistic regression is a type of linear model that examines the relationship between the independent variables and the target. It is used when the output is categorical and the model predicts the probability of a specific outcome occurring.

The **inverse of the regularization strength C**, the **penalty** and the **solver** are set in the model.

C is a positive value where smaller values mean stronger regularization.

The penalty parameter (default *l2*) is the penalty norm added to avoid overfitting, and the solver is the algorithm used for the optimization problem. *Sag* and *Saga* are better performers for large dataset.

The best parameters found are:

- One-vs-Rest (training score: 21,28%)
  - C: 0,1
  - penalty: l2

  - solver: sag

- One-vs-One (training score: 20.45%)
  - C: 0,1
  - penalty: l2
  - solver: sag

4.5.2 Support Vector Machine

Support Vector Machines are a set of supervised learning methods used for classification. They are very flexible and perform well on high-dimensional datasets. The Kernel function can be specified, in the implementation we consider *RBF Kernel* with **regularization parameter C** and Kernel coefficient **gamma**. The strength of regularization is inversely proportional to C, a positive number. The two listed parameters must be balanced, so as to avoid underfitting (both too small) and overfitting (both too large).
The best parameters found are:

- One-vs-Rest (training score: 31,32%)
  - C: 1
  - gamma: scale

- One-vs-One (training score: 29,90%)
  - C: 10
  - gamma: scale

4.5.3 K-Nearest Neighbors

K-Nearest Neighbors is a simple supervised Machine Learning algorithm that can be used for classification. It is based on the idea that the observations, contained in the dataset, that are closest to each other and at a given point are most similar . **K**, usually an odd number, is the number of neighbors to be used: larger values of K define a more robust algorithm with stable decision boundaries.

The parameter **n_neighbors** represents the number of neighbors to use, **weight** is the weight function used in the prediction. It can be:

- uniform where all points in each neighbors are weighted equally
- distance where closer points have a greater influence than the further

Last parameter is **metric** that specifies the metric to use in distance computation.

The best parameter found are:

- One-vs-Rest (training score: 11,01%)
  - metric: euclidean
  - n_neighbors: 5
  - weights: distance
- One-vs-One (training score: 9,49%)
  - metric: euclidean
  - n_neighbors: 5
  - weights: distance

4.5.4 Gaussian Naive Bayes

Gaussian Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem and normal distribution.
To avoid division by zero or by very small numbers, the **var_smoothing** parameter adds a small smoothing to the variance of each feature. If the parameter is very small, overfitting should occur.
The best parameter found are:

- One-vs-Rest (training score: 17,06%)
  - var_smoothing: 1^{-11}
- One-vs-One (training score: 17,29%)
  - var_smoothing: 1^{-11}

4.5.5 Random Forest

Random Forests are for supervised based Machine learning algorithms that can be used to solve classification problems. By fixing the number of trees **n_estimators** in the forest, the number of features **n_features** to consider when looking for the best split and the **minimum number of samples** required to be at a leaf node, the risk of underfitting and overfitting can be managed. The best parameter found are:

- One-vs-Rest (training score: 23,91%)
  - n_estimators: 100
  - n_features: sqrt
  - minimum number of samples: 4
- One-vs-One (training score: 26,59%)
  - var_estimators: 100
  - n_features: sqrt
  - minimum number of samples: 2

Chapter 5

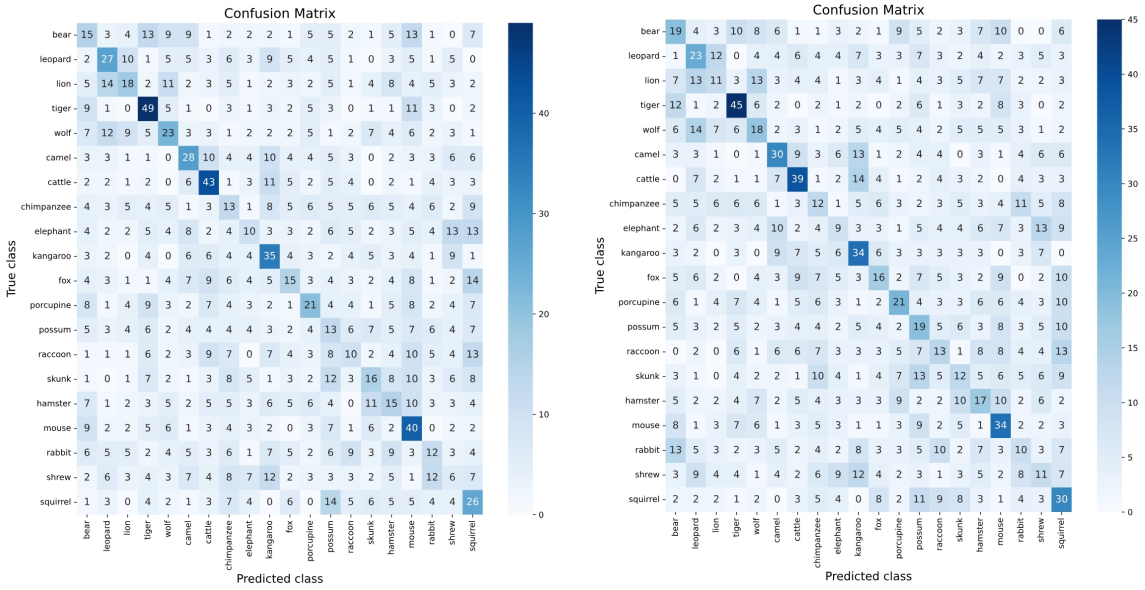
Results

5.1 Confusion Matrix

The confusion matrix helps evaluate the performance of the classification by comparing predicted values with actual values. it allows to count how often the algorithm predictions are correct or wrong.
By checking the matrices obtained for all the methods with One-vs-Rest and One-vs-One it is easily seen that there is not a great difference between the two approaches.

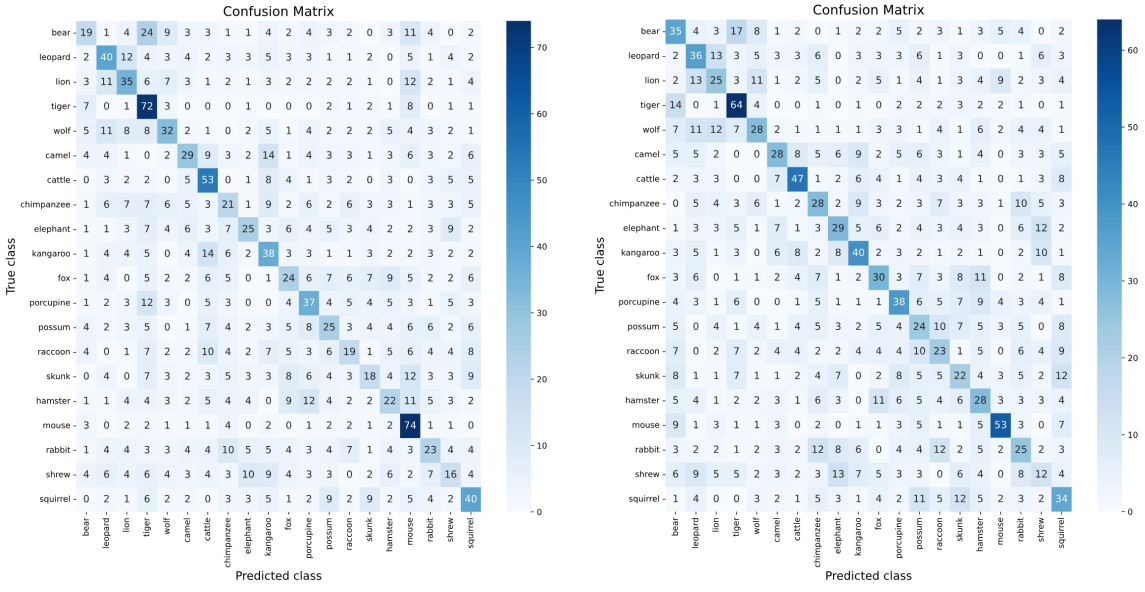
In the confusion matrix [5.3] obtained with K-Nearest Neighbors, many examples of other classes are labeled as *wolves*, when actually they are not.
The classes do not appear to be well separable so the features may not be relevant enough. The other classes do not have peaks on the diagonal.

In the confusion matrix [5.4] obtained with Gaussian Naive Bayes method, many examples of other classes are labeled as *possum*, when they are not.
Even if some other classes have peaks on the diagonal, the *possum* class has highest values the column for all the classes analyzed.



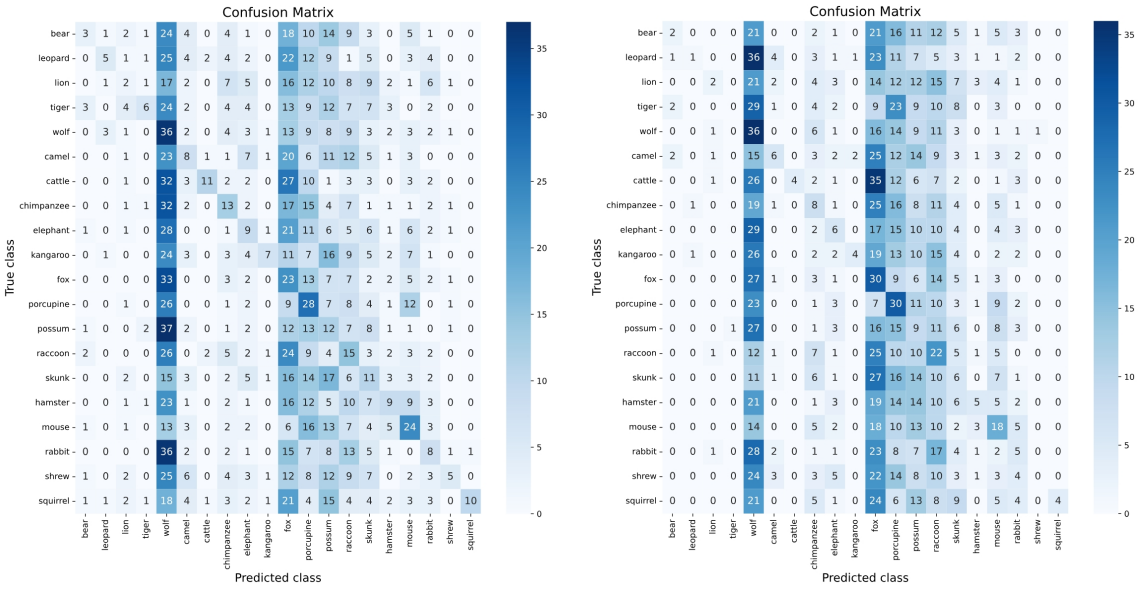
(a) *One-vs-Rest* (b) *One-vs-One*

Figure 5.1: Logistic Regression



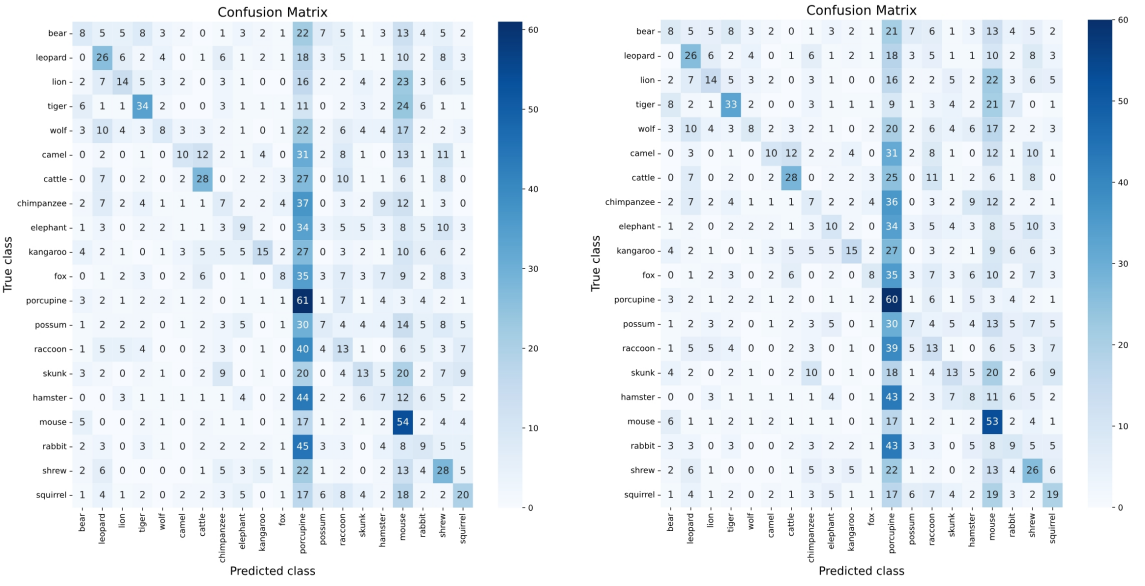
(a) *One-vs-Rest* (b) *One-vs-One*

Figure 5.2: Support Vector Machine



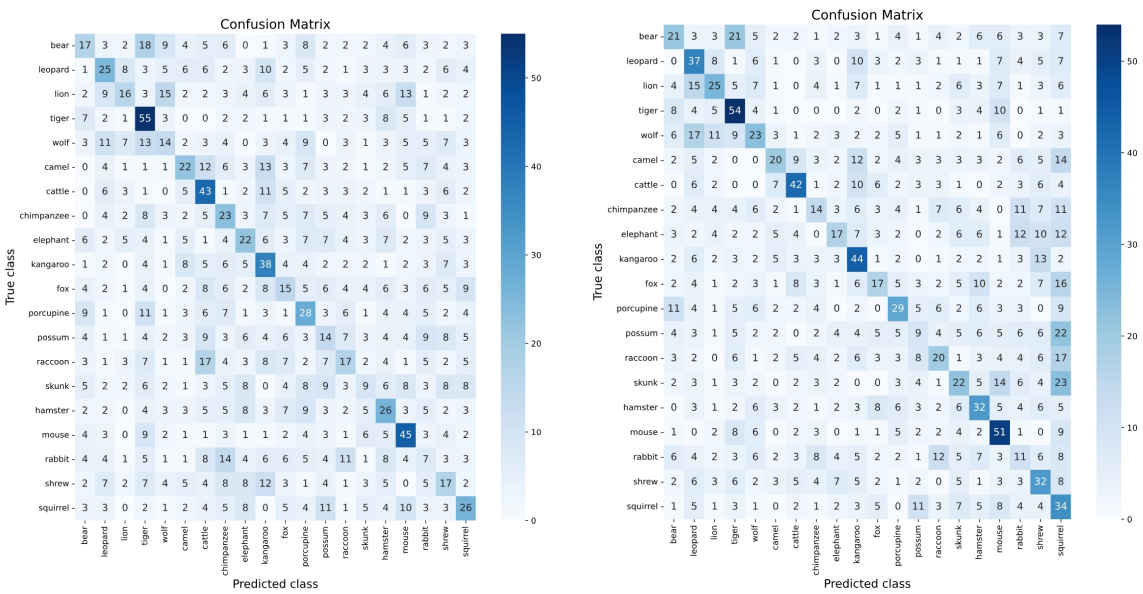
(a) *One-vs-Rest* (b) *One-vs-One*

Figure 5.3: K-Nearest Neighbors



(a) *One-vs-Rest* (b) *One-vs-One*

Figure 5.4: Gaussian Naive Bayes



(a) *One-vs-Rest* (b) *One-vs-One*

Figure 5.5: Random Forest

5.2 Comparisons

The models' performance by using One-vs-Rest approach:

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0,22	0,21	0,22	0,21
Support Vector Machine	0,33	0,32	0,33	0,32
K-Nearest Neighbors	0,12	0,26	0,12	0,12
Gaussian Naive Bayes	0,19	0,23	0,19	0,18
Random Forest	0,24	0,23	0,24	0,23

The models' performance by using One-vs-One approach:

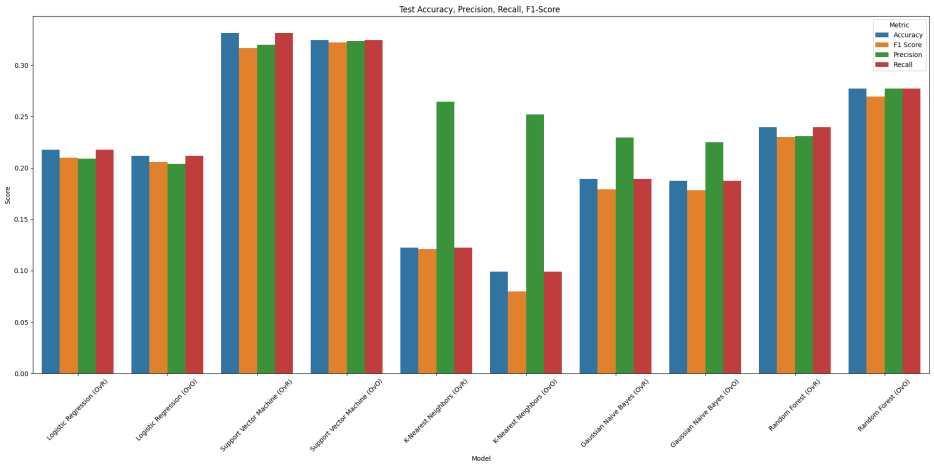


Figure 5.6: Models metrics.

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0,21	0,20	0,21	0,21
Support Vector Machine	0,32	0,32	0,32	0,32
K-Nearest Neighbors	0,10	0,25	0,10	0,08
Gaussian Naive Bayes	0,19	0,22	0,19	0,18
Random Forest	0,28	0,28	0,28	0,27

Except for Gaussian Naive Bayes and K-Nearest Neighbors, all the other methods have, under cents, the same accuracy, precision, recall, and F1-Score **??**. This is, probably due to the balanced distribution of examples: 100 examples are analyzed for each available class.
Despite the fact that the models perform quite well further improvements can be applied to the hyperparameter to make it have better performance.

The plot shows an high peak in the K-Nearest Neighbors precision. Maybe correlations and dependencies between features make this model fails while Random Forest prevails.
For Random Forest One-vs-One approach dominate on One-vs-Rest.
The best result are obtained with Support Vector Machine algorithm: the features allows a good separation between classes and the Kernel trasforms the space in a good way.

Chapter 6

Conclusions

Different approaches and different hyperparameters are used in order to explore the most of scenarios.

The best results for the image classification problem using a subset of CIFAR-100 are obtained with Support Vector Machine and Random Forest algorithm.

traditional methods have limits on raw images and despite the process of transforming raw data into more useful information for traditional Machine Learning models, improvements can be optained by using Deep Learning algorithms.

Bibliography

[1] https://www.cs.toronto.edu/ kriz/cifar.html

[2] https://groups.csail.mit.edu/vision/TinyImages/

[3] https://docs.pytorch.org/vision/stable/index.html