**PROJECT REPORT**

**CMPE-257 - MACHINE LEARNING**



**Submitted By: Group 7**

**Team Members:**

**Qiao Liu - 013802893**

**Ching-Min Hu - 013726154**

**Dandan Zhao - 013795392**

**Fernanda Bordin - 013800638**

**Megha Rajam Rao - 013709488**

**Rajasree Rajendran - 013774358**

**Selected ML Algorithm:**

**Google Colab link:**

# Task Assignment

|  |  |  |
| --- | --- | --- |
| Task | Description | Names |
| Data Preparation | Load, pre-process and visualize data | Megha, Dandan |
| ML methods | KNN classifier, Bagging classifier | Dandan, Ching-Min |
| Neural networks | Dense(feedforward) and Densenet121 | Fernanda |
| Neural Networks | Convolutional Neural Network | Qiao, Shree |
| Powerpoint presentation | Input on Data preparation  Input on ML algorithms  Input on Neural network  Input on CNN | Megha  Dandan, Ching-Min  Fernanda  Qiao, Shree |
| Report (contributors) | Overall report coordination  Input on Data Preparation  Input on Neural network, overall analysis  Input on ML algorithms  Input on CNN | Megha, Shree  xxxx |

**TABLE OF CONTENTS**

[Task Assignment](#_3znysh7) **2**

[**Introduction**](#_leteue3qxgj) **6**

[**Libraries**](#_1t3h5sf) **6**

[**Softwares & Tools**](#_4d34og8) **7**

[**Procedure (new)**](#_1ksv4uv) **7**

[**Traditional Machine Learning Model Analysis**](#_kv8hym8wefeh) **9**

[**Bagging Classifier**](#_k2wg6rbiyfpu) **9**

[**Confusion matrix & Classification Report**](#_96vt8dgn364g) **11**

[**Bagging - overall analysis prediction loop**](#_729k9ai2idzp) **12**

[**Milestone 2 - 1 pair untrained**](#_1bqioorre650) **12**

[**Milestone 3 - 2 pair untrained**](#_ygiqgis3sn3s) **13**

[**Milestone 3 - 3 pairs untrained**](#_vaatqy2xqdo5) **14**

[**Bagging - overall discussion**](#_mf14hpqdtp0s) **15**

[**References**](#_8ktl6qgmsu8y) **15**

Professor’s instructions

**Data preparation**

D1) Work on one of the 7 superclass pairs listed here.

D2) Combine both the testing and training data into one dataset after you load the cifar100 data

D3) Filter to select images belonging to your assigned superclasses.

D4) Validate you have selected the right data matching your assigned classes by printing 4 randomly selected images from each sub-class on your Colab notebook. Show the verbal (not numeric) superclass and class labels of each randomly printed image on top of the image.

**Prediction on randomly selected testing images**

Randomly select 80% of the data as training data and the rest as testing data. Use your machine learning (ML) algorithm to train the model with the training data and subsequently use it to predict the testing data. Disregard the sub-class label and focus on the superclass label only in this binary classification problem.

PR1) Describe your ML algorithm(s) with workflow/architecture diagram(s).

PR2) Show your prediction scores with train-test-split (80/20).

PR3) List 36 random images with original and predicted labels with your algorithm.

PR4) Use confusion matrix and classification report to analyze your binary classification result. Explain what you observed from the result and comment on if they are consistent.

PR5) Compare at least two ML algorithms used and comment on why one is performing than the rest.

**Prediction on one testing subclass images from each of the two superclasses**

Select one sub-class from each of your two assigned super-classes. Use these two selected sub-classes as your testing data. Use all the data minus the two selected test sub-classes as your training data. Repeat for all 5 x 5 =25 pairs of subclass images. Your algorithm should be initialized before it is being trained on new training dataset in each of these 25 trials. All results should be based on the SAME algorithm you designed.

PS1) Summarize your overall prediction results in a 5 by 5 chart, with 5 subclasses from one superclass as the row index and the other 5 subclasses from the other superclass as the column index. Color the worst 5 and the best 5 with different colors for ease of reference in the chart.

PS2) Show, in a table, the following summary among these 25 trials.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **mean** | **std** | **min** | **25%** | **50%** | **75%** | **max** |

PS3) Explain your observed result

**Prediction on two testing subclass images from each of the two superclasses**

PD1) Summarize your overall prediction results in a 10 by 10 chart, with 2 subclasses from one superclass as the row index and the other 5 subclasses from the other superclass as the column index. Color the worst 5 and the best 5 with a different color for ease of reference in the chart.

PD2) Append to the table in PS2, the following summary among these trials.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **mean** | **std** | **min** | **25%** | **50%** | **75%** | **max** |

PD3) Comment on the prediction performance difference by your ML algorithm on PR1-PR5 and that of PS1-PS3. Are they consistent and why?

**Bonus:**

Extend the requirements in PD1-PD3 to using 3 subclasses from each superclass as testing data.

1. **Compose technical reports, present your result, and answer questions** (??%)

* **Your report should contain at least the following:**

R1. A table of content with hyperlinkable page to each relevant section (Hint: Use Word->Reference->Table of content and make each chapter/section description as Heading 1 or 2 under Styles first)

R2. Describe any libraries used, how to set up, configure, and run your code in your report.

R3. List your team members and team name on the first page of both your report

R4. Provide all online links on example code you used/extended in the report.

R.5 Show a clear list on who collaborated with who to accomplish any bonus feature, if any.

R6. Add comments on each major step inside your scripts. Show references to any algorithm or reused code.

# 

# 

# 

# 

# 

# 

# **Introduction**

The CIFAR datasets are labeled subsets of the 80 million tiny images dataset collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The images are of size 32x32 pixels with 3 color channels (RGB). It comprises of 100 classes containing 600 images each (500 training and 100 testing). The classes (fine labels) are grouped into 20 super classes (coarse labels) and corresponding classes. In this report we filtered the CIFAR-100 dataset to select images from the super classes we chose, which are medium-sized mammals and small mammals.

The medium-sized mammals superclass includes the following classes:

fox, porcupine**,** possum, raccoon, skunk.

Small mammal’s superclass includes the following classes:

Hamster, mouse, rabbit, shrew, squirrel.

# **Libraries**

1. Numpy
2. Pandas
3. Keras
4. Sklearn
5. Matplotlib
6. Tensorflow
7. Math
8. Time
9. Seaborn

# **Softwares & Tools**

1. Google Colaboratory
2. Python (language)
3. Powerpoint (presentation)
4. Word (report)
5. Google drive (document sharing)

# 

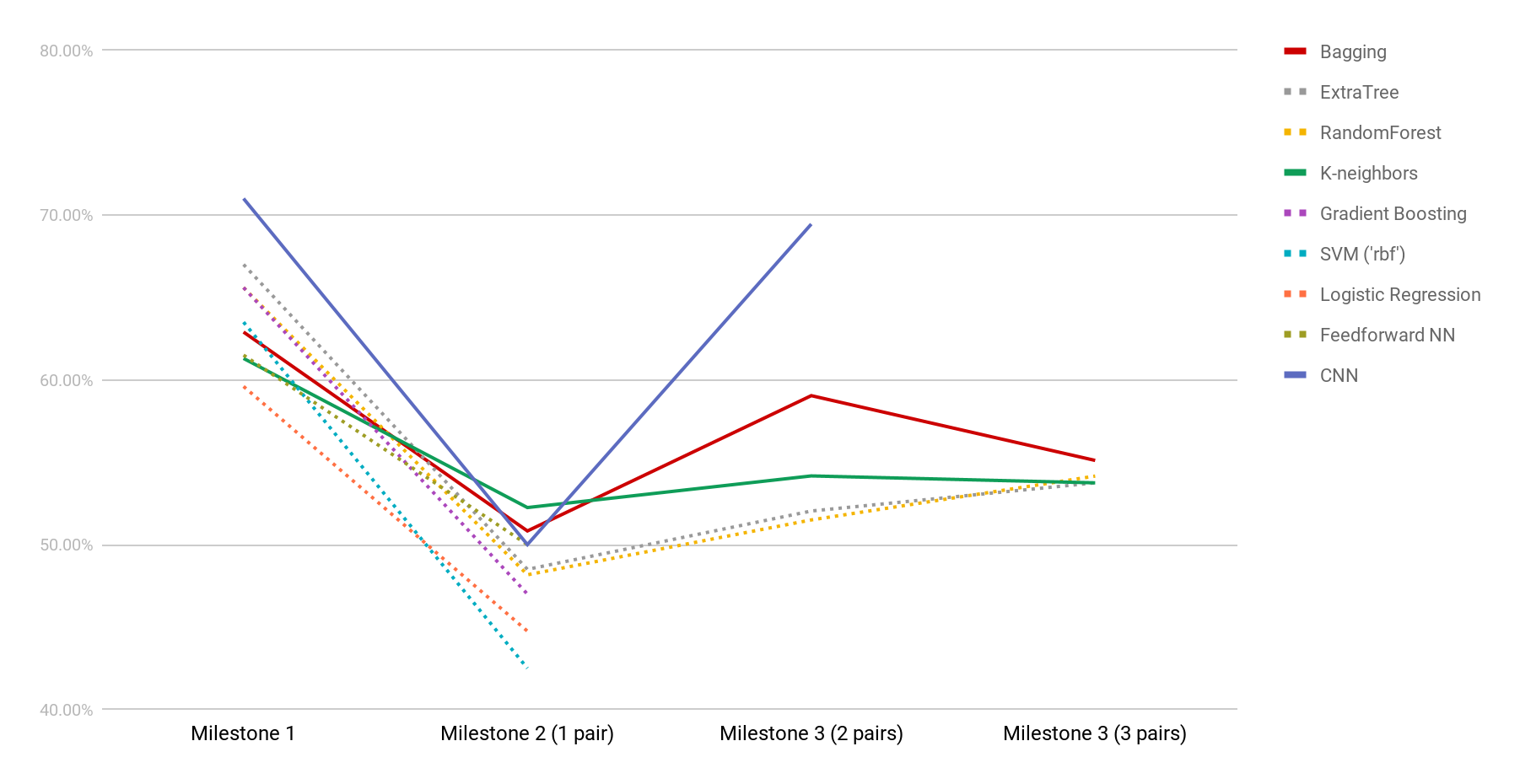
# 

# 

# **Procedure (new)**

Through the three milestones we tested several classifier models and neural networks. As the process progressed we excluded certain models or didn’t explore them to their fullest extent. The chart below shows the evolution of the score of certain models:

* In milestone 1, we show the score where we randomly selected 83% of the data to train and 17% to test;
* In milestone 2, we show the score for the testing pair Raccoon vs Hamster
* In milestone 3 (both parts), we show the score for the testing pair Raccoon vs Hamster



As the chart shows, in milestone 1, most algorithms delivered scores varying from 60 to 70%, it was once we singled out the pair Raccoon & Hamster as testing data (milestone 2) that we were able to start visualizing the consistency of each model.

Models such as Gradient Boosting, SVM, Logistic Regression presented such a decrease in performance that they were no longer considered for future experiments. The neural networks, both convolutional (CNN) and feedforward started to overfit very easily and their confusion matrix showed that their score of 50% was because they were only classifying as one category (medium mammals). For this reason, only the CNN model was tested on milestone 3.

For the top two models in milestone 2 (KNN and Bagging) we ran a loop to check their scores once we varied the 2 classes used exclusively for testing. We found a score difference of approximately 13% from the “best” paring to the “worst” paring, showing that the test paring had a very big impact on the score.

On milestone 3, in order to select which models we wanted to run the deeper analysis (running the loop again), we first ran a simple analysis using our standar pair (Raccoon & Hamster), this allowed us to drop random forest and extra tree models from the analysis. Both bagging and K-neighbors performed well, so we moved forward with bagging. CNN ended up making a surprise come back, that we’ll be discussing further on.

The table below has the models results overview, including the averages, max and min for the models where the loop was run.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MODEL | MILESTONE 1  SCORE | MILESTONE 2  SCORE | MILESTONE 3  (2 PAIRS) SCORE | MILESTONE 3  (3 PAIRS) SCORE |
| K-neighbors Classifier | 61.3% | 52.25% | 54.17% | 53.75% |
| Bagging Classifier | 62.9% | 50.83%  Mean: 50.47%  Std: 4.7%  Min: 40.5%  25%: 47.75%  50%: 50.58%  75%: 52.50%  Max: 62.33% | 59.04%  Mean: 52.23%  Std: 3.34%  Min: 43.00%  25%: 50.03%  50%: 52.65%  75%:54.40%  Max: 59.71 | 55.11%  Mean: 51.53%  Std: 3.42%  Min: 42.33%  25%: 49.59%  50%: 51.73%  75%: 53.89%  Max: 59.91% |
| ExtraTree Classifier | 67% | 48.5% | 52.04% | 53.75% |
| RandomForest Classifier | 65.60% | 48.17% | 51.50% | 54.16% |
| Gradient Boosting Classifier | 65.8% | 47% | NA | NA |
| SVM kernel = “rbf” | 63.5% | 42.5% | NA | NA |
| Logistic Regression Classifier | 59.6% | 44.75% | NA | NA |
| Neural Network (Feed Forward) | 61.51% | 50% | NA | NA |
| Convolutional Neural Network (experiment 1) | 74.10% | 50% | NA | NA |
| CNN (experiment 2) | 76.18% | 50% | 69.45%  Mean: 51.53%  Min: 42.33%  Max: 59.91% | 69.45%  Mean: 51.53%  Min: 42.33%  Max: 59.91% |

After analysing the data, be decided to focus on bagging for our analysis of traditional machine learning models, since it presented the most consistently positive result.

# **Traditional Machine Learning Model Analysis**

## 

## **Bagging Classifier**

Enter text here - explaining quickly what we did and how we optimized. Figures below are the 36 examples he asked for( dandan wrote)

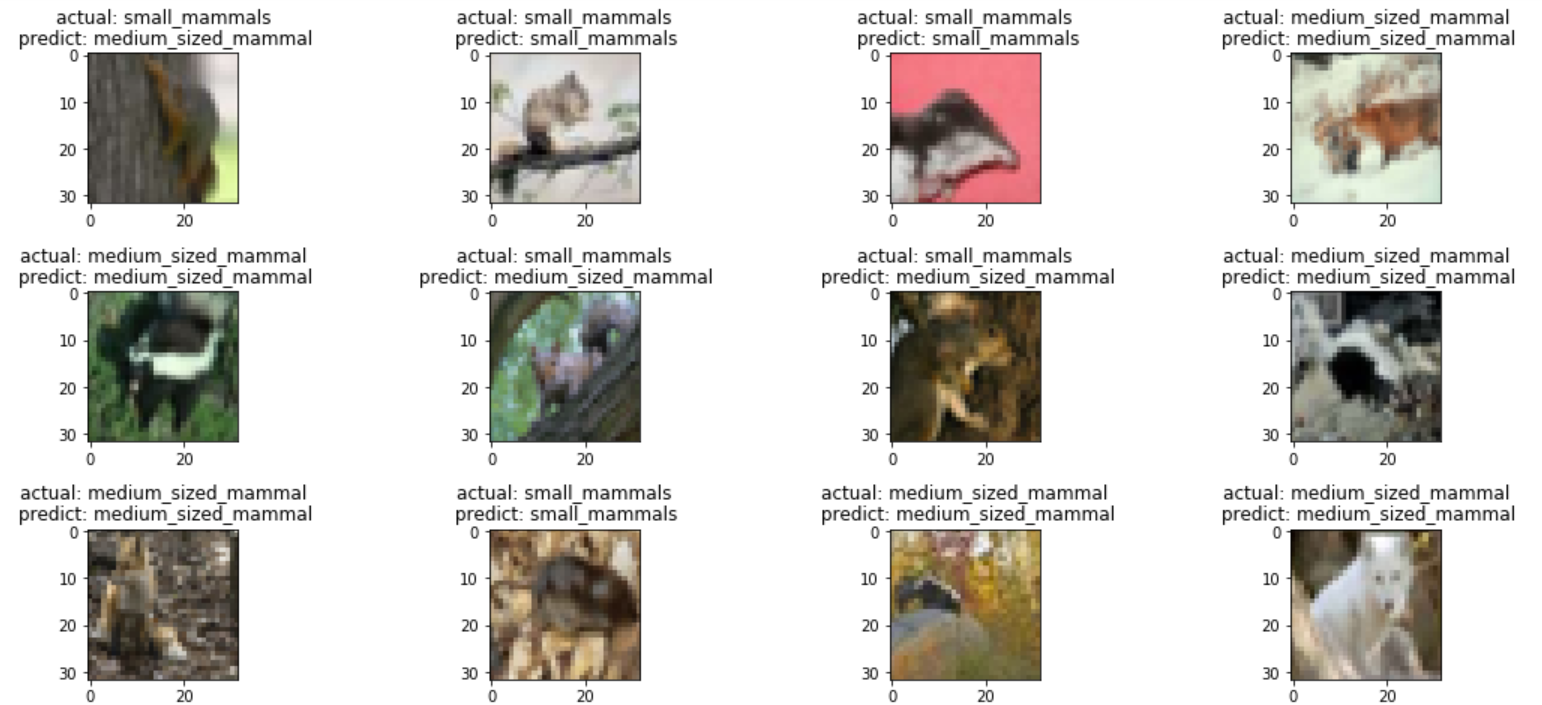
The model was defined and built, then fit using training data. We use the grid search to tune the parameters n\_estimators, max\_sample, bootstrap, and bootstrap\_features, then we got the best parameters:

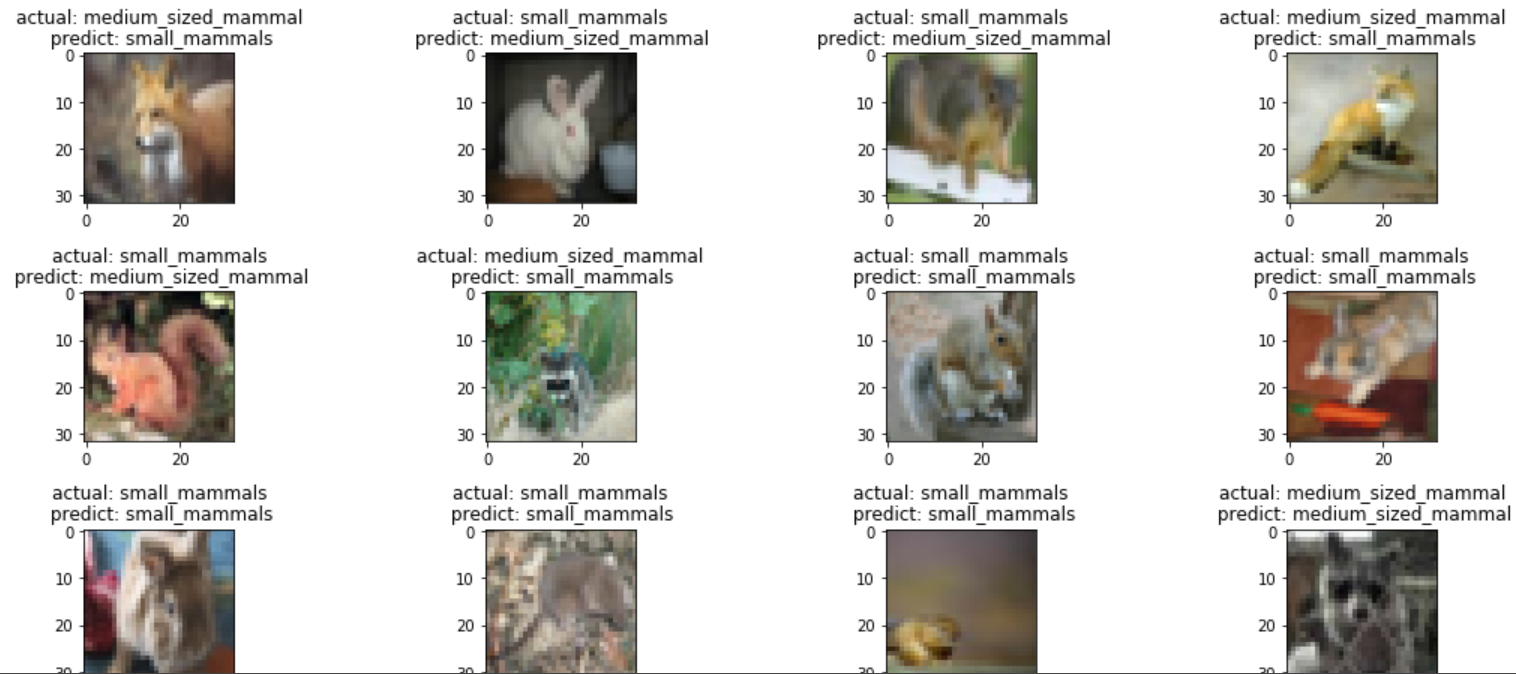
For 2 pairs missing data, n\_estimators: 3, max\_sample: 0.01, bootstrap: False, bootstrap\_features: False

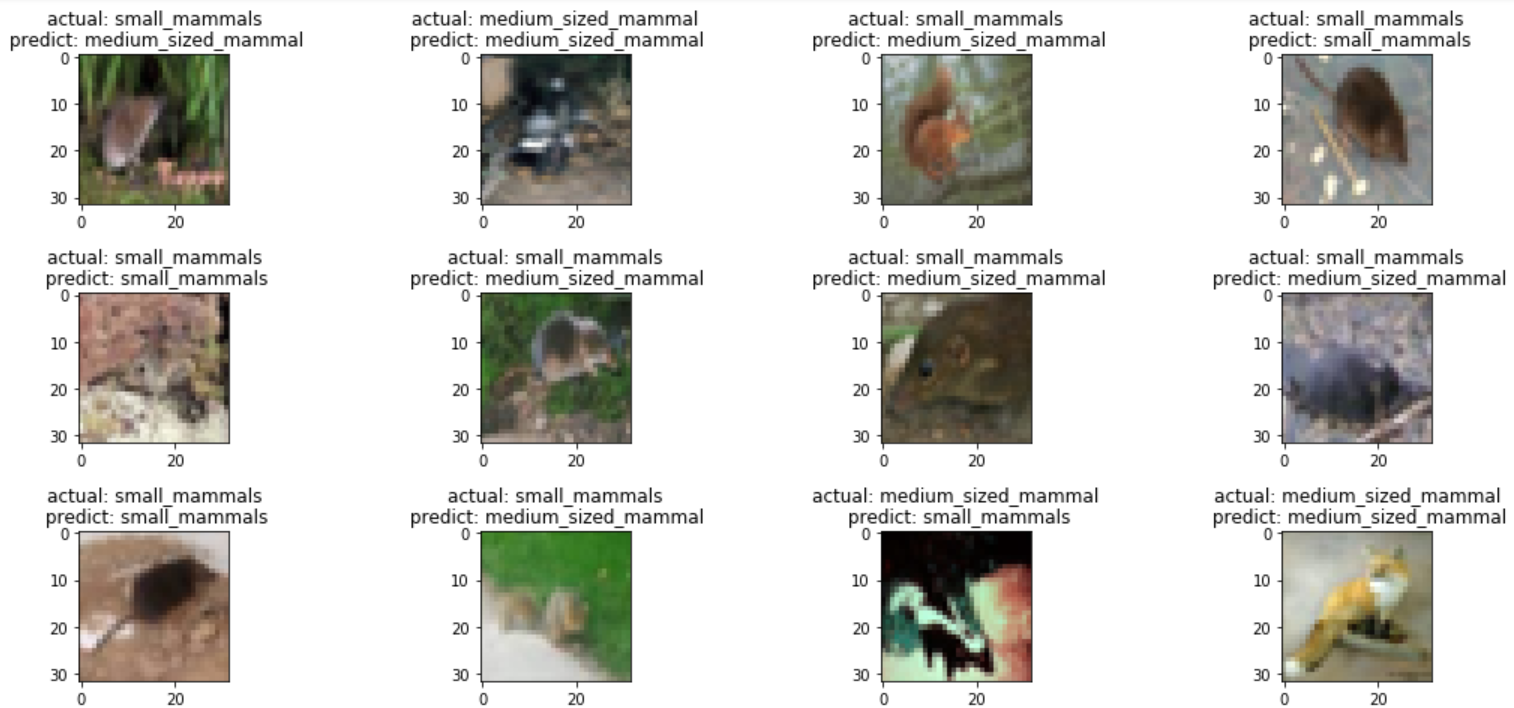
For 3 pairs missing data, n\_estimators: 3, max\_sample: 0.01, bootstrap: True, bootstrap\_features: False

The prediction was generated using ***.predict()*** which the 36 images shown below:

(2 pairs missing)

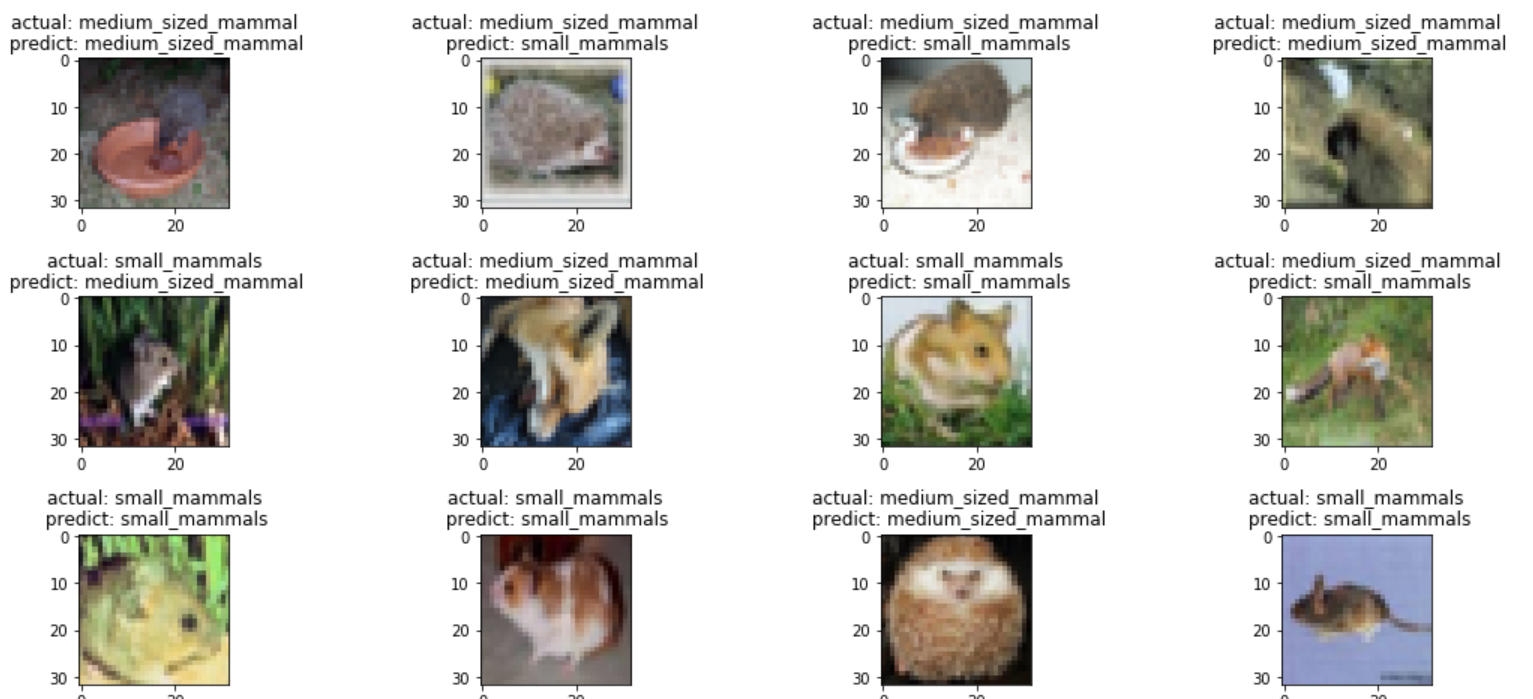


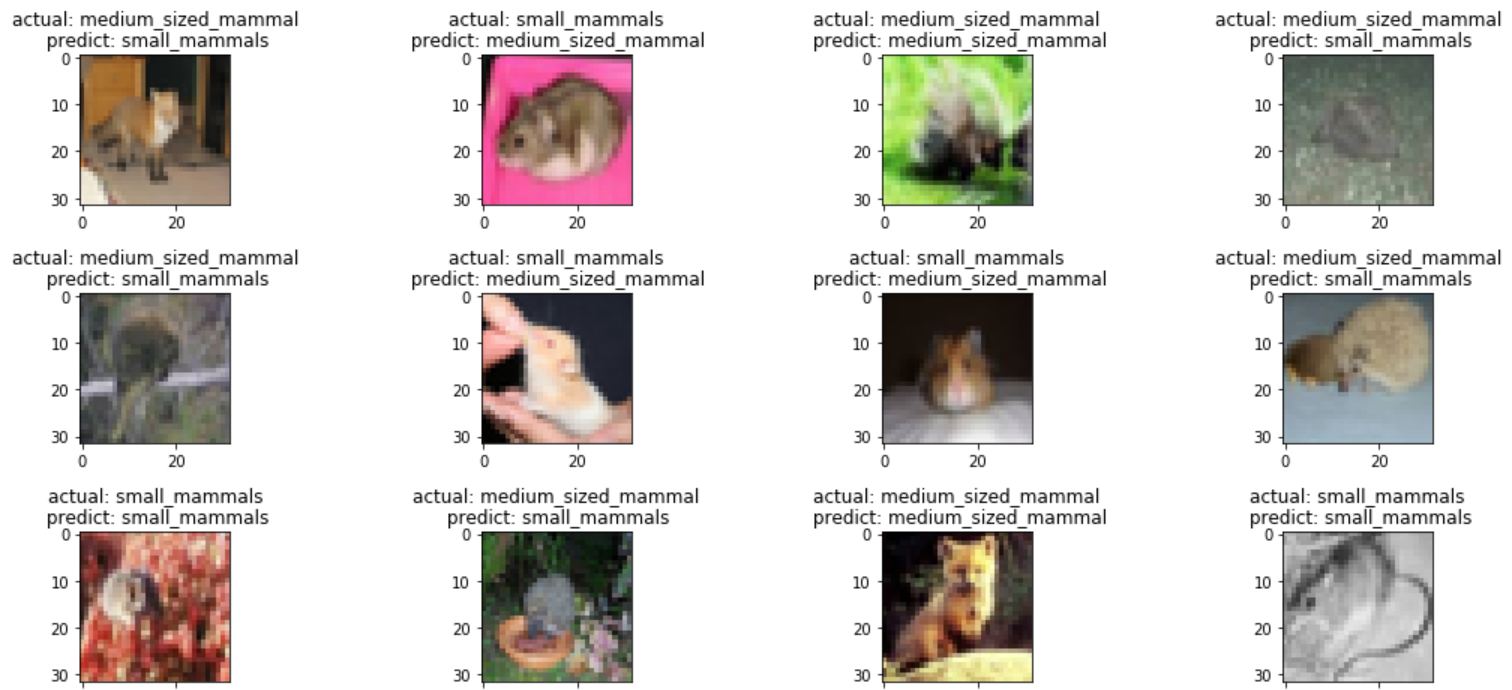




### (3 pairs missing)

### 





### 

### Confusion matrix & Classification Report

Text here -> explain how to read, show that the model is guessing medium more than small which leads to the low precision for small. Compare the results and differences for 2 and 3 pairs

Maybe discus a bit what could be the reason for the different behaviors(dandan wrote)

Due to the model evaluation, since our data set is balanced, which the medium-sized mammals and small mammals have equal proportions, so the baseline accuracy is 50%, means we can predict by chance. From the confusion matrix and classification report of the 2 pair missing data and 3 pair missing data, which the F1-score of small\_mammals is smaller than that of medium\_sized\_mammals. From the F1-score, we can see that medium\_sized\_mammals are more positively predicted by the models than the small mammals. The result is also consistent with the milestone1 and milestone2.

However, from the precision and recall of the 2 pair missing classification report, the precision of the small mammals is higher, which means small mammals were predicted more precisely by this model, even just slightly better than the baseline, on the contrary, the recall of the small mammals is much lower than medium\_sized\_mammals, which means the model missed most small mammals. The model for the 2 pair missing data is not so good from the evaluation.

For the classification report of 3 pairs missing data, we can see that ‘Medium sized Mammals’ have a same precision with ‘Small Mammals’, just slightly higher than the baseline, this means that this model has the same chance of misclassifying the animals into another superclass.The recall scores, combining with the confusion matrix, indicates that the model is more inclined to predict an image as a medium\_sizd\_ mammals. This indicates that this model performs better at recognizing medium sized mammals.

|  |  |
| --- | --- |
|  |  |
| Bagging (2 pairs) | Bagging (3 Pairs) |

|  |  |
| --- | --- |
|  |  |
| Bagging (2 pairs) | Bagging (3 Pairs) |

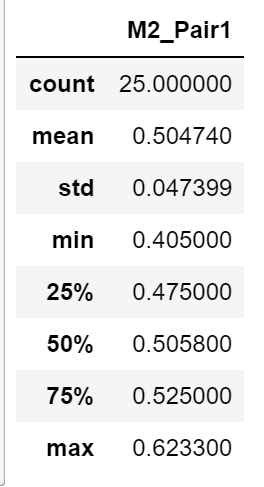
### 

## **Bagging - overall analysis prediction loop**

### **Milestone 2 - 1 pair untrained**

* Easiest group to predict: Possum, Hamster
* Hardest group to predict: Fox, Skunk

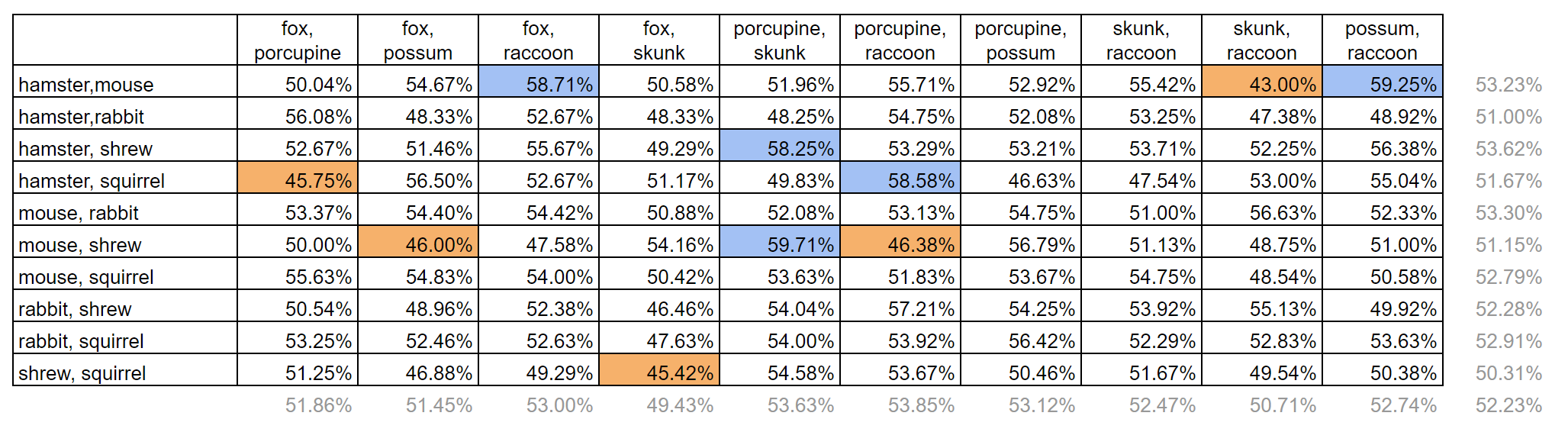
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Bagging** | Fox | Porcupine | Possum | Raccoon | Skunk | AVERAGE |
| Hamster | 58.09% | 48.09% | 62.33% | 50.84% | 47.50% | 53.37% |
| Mouse | 47.50% | 56.92% | 56.17% | 52.75% | 51.50% | 52.97% |
| Rabbit | 45% | 40.50% | 51.17% | 53.75% | 52.50% | 49% |
| Shrew | 43.84% | 46.75% | 49.50% | 47.67% | 51.17% | 47.79% |
| Squirrel | 47.50% | 50.84% | 50.58% | 49% | 49.58% | 49.50% |
| AVERAGE | 48.39% | 48.62% | 53.95% | 50.80% | 50.45% | 50.44% |

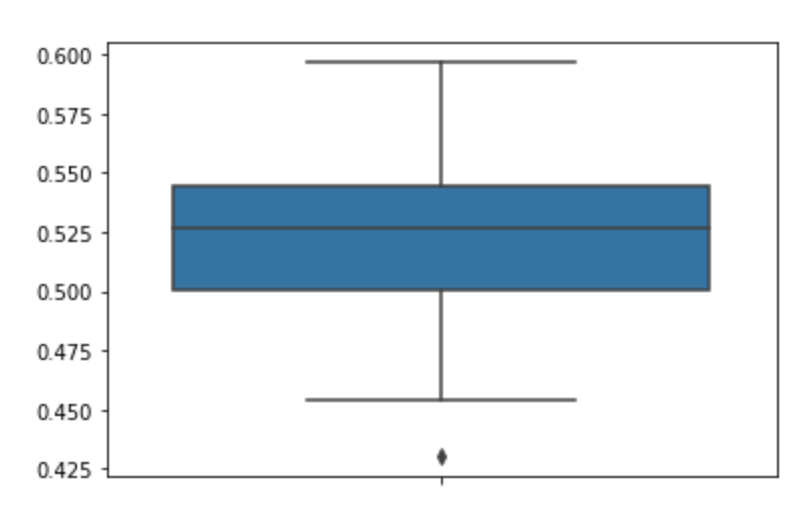


### **Milestone 3 - 2 pair untrained**

# 

* Best testing pair: porcupine & skunk vs mouse & shrew (59.71%)
* Worst testing pair: skunk & raccoon vs hamster & mouse (43%)
* Easiest group to predict: Porcupine, Raccoon
* Hardest group to predict: Fox, Skunk





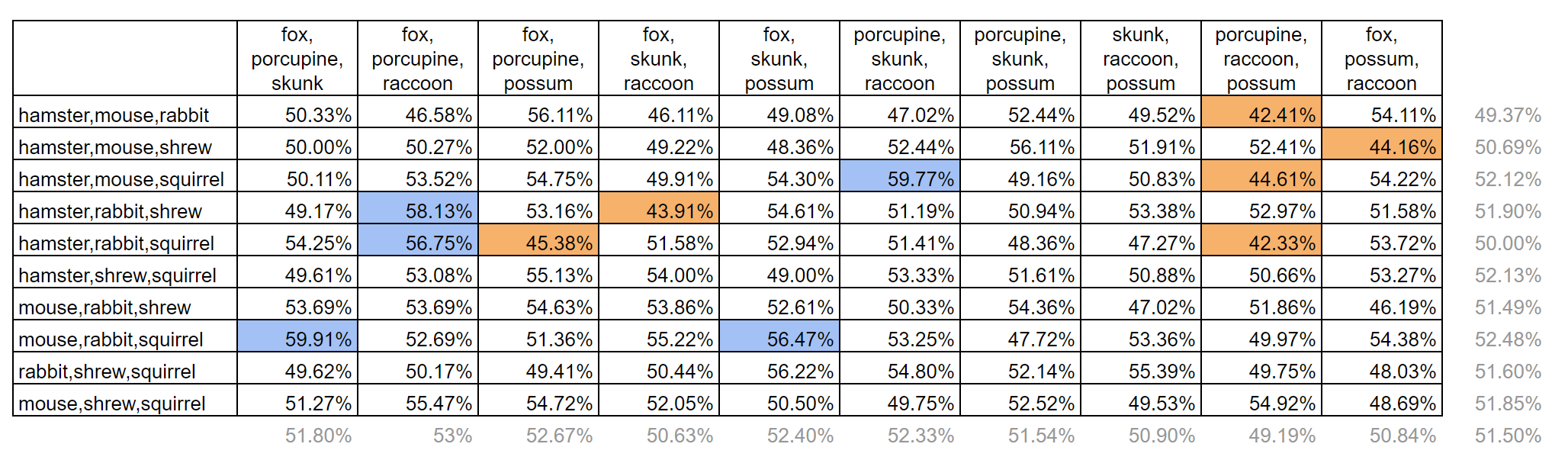
# 

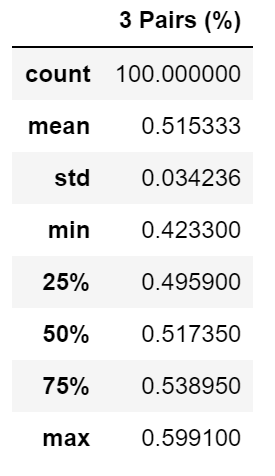
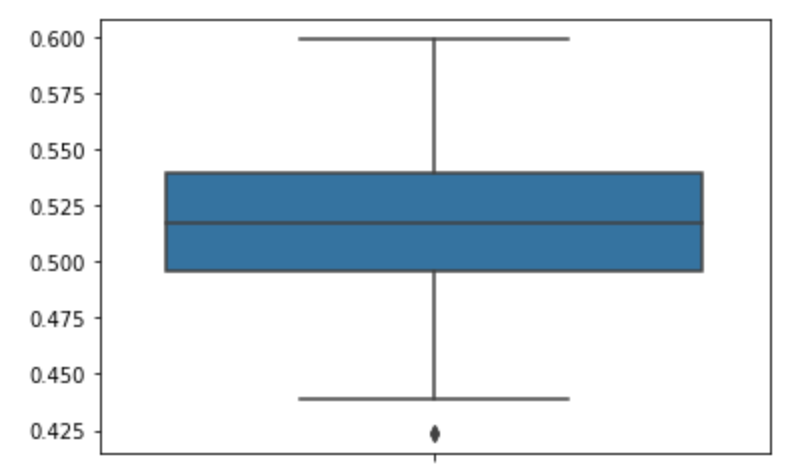
# 

# 

### **Milestone 3 - 3 pairs untrained**

* Best testing pair: fox, porcupine, shunk vs mouse, rabbit, squirrel (59.91%)
* Worst testing pair: porcupine, raccoon, possum vs hamster, rabbit, squirrel (42.33%)
* Easiest group to predict: Porcupine, Raccoon, Fox\*
* Hardest group to predict: Porcupine, Raccoon, Possum\*\*B





### **Bagging - overall discussion**

Enter text here - > What conclusions do we have, what groups are good, what groups that aren’t

Keep in mind that our best scores come when we exclude classes that aren’t adding that much value to the model. For example, this means that FOX is an essential class to have in training, since we get very bad scores, when it is not used to train.

Enter new topics…..

# **References**

Berhane, F. (2016). *Deep Neural Network for Image Classification: Application*. From Data Scientist : https://datascience-enthusiast.com/DL/Deep-Neural-Network-for-Image-Classification.html

Kinli, F. (2018, September). *[Deep Learning Lab] Episode-5: CIFAR-100*. From Medium.com: https://medium.com/@birdortyedi\_23820/deep-learning-lab-episode-5-cifar-100-a557e19219ba; <https://nextjournal.com/mpd/image-classification-with-keras>

[corochann](https://corochann.com/) (2017): *CIFAR-10, CIFAR-100 dataset introduction*

<https://corochann.com/cifar-10-cifar-100-dataset-introduction-1258.html>

Mohtadi Ben Fraj(Dec 21, 2017): *In Depth: Parameter tuning for Random Forest*

<https://medium.com/all-things-ai/in-depth-parameter-tuning-for-random-forest-d67bb7e920d>

[Aarshat J](https://www.analyticsvidhya.com/blog/author/aarshay/)ain (Feb 21, 2016): *Complete Guide to Parameter Tuning in Gradient Boosting (GBM) in Python*

<https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/>