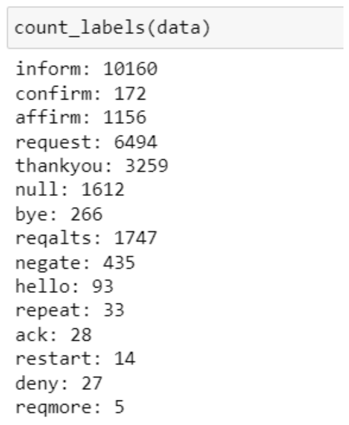
1. Descriptive Statistics



```
count_labels(data)
```

```
inform: 10160
confirm: 172
affirm: 1156
request: 6494
thankyou: 3259
null: 1612
bye: 266
reqalts: 1747
negate: 435
hello: 93
repeat: 33
ack: 28
restart: 14
deny: 27
reqmore: 5
```

Little hypothesis about the data upfront:

There is a huge difference between the number of sentences for each class. The ‘inform’(10160) is far greater than ‘reqmore’(5) and the rate between them is about 2032, which shows that the dataset class is highly imbalanced and it could cause a huge difference in the accuracy later.

2. A baseline system

2.1 A baseline system always assigning the majority class

The model that classifies everything in ‘inform’(majority) gives us an accuracy of **39.8%**

2.2 A baseline rule-based system based on keyword matching

Final keywords:

ack: kay, okay

affirm: yes, right, yeah

thankyou: thank

bye: bye, goodbye

confirm: is it, does it, do they

deny: wrong, dont, not

hello: hi, hello, halo, welcome

inform: looking, restaurant, any, food, part, town, cheap, expensive, mediterranean, seafood, east, west, north, south, asian, oriental, scottish, matter, european, want, care, austrian, center, corsica, international, priced, moderately, moderate, central, eirtrean, spanish, venue, australian, turkish

negate: no

null:

repeat: repeat, again, back
reqalts: how about, what about, is there, anything else
reqmore: more
request: address, phone, number, post code, how much, where, whats, what is, price range
restart: start, reset

After looking for all the keywords and combining them we got an accuracy of **81.5%**

Sort categories from highest to lowest:

Class	Accuracy score
repeat	100%
reqmore	100%
restart	92.86%
request	88.10%
thankyou	87.11%
reqalts	85.00%
deny	81.48%
affirm	80.62%
inform	79.60%
bye	74.44%
negate	70.11%
null	61.29%
ack	60.71%
confirm	51.16%
hello	49.46%

The accuracy between different class has also huge difference. The 'reqmore'(100%) is almost twofold of the 'hello'(49.46%). Therefore it could say that the baseline system based on the key word doesn't perform equally well in different class.
We see that keyword recognition has some flaws concerning classifying the sentences. Sentences where words from different classes are in the sentence, it can't choose the best option. This can be overcome with a machine learning model.

3. Evaluate model

3.1 Decision Tree Classifier

Quantitative evaluation: Evaluate your system based on one or more evaluation metrics. Choose and motivate which metrics you use.

Here are the basic parameters for Decision Tree Classifier:

- **max_depth**: Controls the depth of the tree and prevents overfitting.
- **min_samples_split**: Prevents nodes with very few samples from splitting, limiting the complexity.
- **criterion**: Determines the method for evaluating splits (Gini or entropy).

Taking into consideration the number of features, a small depth of the tree would be insufficient to achieve good results. At the same time, it's important to avoid overfitting to be able to generalize.

- With max_depth = 5, min_sample_split = 5, criterion = "entropy":

```
Training Accuracy: 0.7923875432525952

Test Accuracy: 0.8031887088342917
```

- With max_depth = 10, min_sample_split = 5, criterion = "entropy":

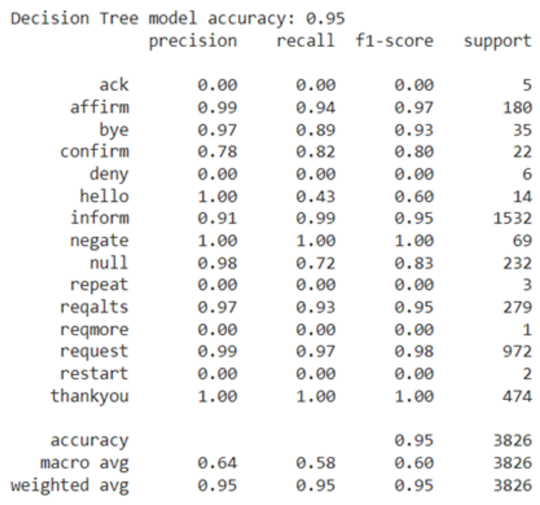
```
Training Accuracy: 0.8779700115340254

Test Accuracy: 0.8881338212232096
```

- With max_depth = 20, min_sample_split = 5, criterion = "entropy":

```
Training Accuracy: 0.9522029988465974

Test Accuracy: 0.9545216936748563
```



```
Decision Tree model accuracy: 0.95
              precision    recall  f1-score   support

         ack       0.00      0.00      0.00         5
      affirm       0.99      0.94      0.97       180
         bye       0.97      0.89      0.93        35
     confirm       0.78      0.82      0.80        22
        deny       0.00      0.00      0.00         6
       hello       1.00      0.43      0.60        14
      inform       0.91      0.99      0.95      1532
      negate       1.00      1.00      1.00        69
        null       0.98      0.72      0.83       232
      repeat       0.00      0.00      0.00         3
     reqalts       0.97      0.93      0.95       279
     reqmore       0.00      0.00      0.00         1
     request       0.99      0.97      0.98       972
     restart       0.00      0.00      0.00         2
    thankyou       1.00      1.00      1.00       474

    accuracy                           0.95      3826
   macro avg       0.64      0.58      0.60      3826
weighted avg       0.95      0.95      0.95      3826
```

With max_depth = 30 we obtain similar results.

- With max_depth = 20, min_sample_split = 3, criterion = “entropy”:

```
Training Accuracy: 0.9530334486735871

Test Accuracy: 0.954260324098275
```

- With max_depth = 20, min_sample_split = 7, criterion = “entropy”:

```
Training Accuracy: 0.951833910034602

Test Accuracy: 0.9532148457919498
```

- With max_depth = 20, min_sample_split = 7, criterion = “gini”:

```
Training Accuracy: 0.9515109573241061

Test Accuracy: 0.9519079979090433
```

Therefore, we can observe that the best configuration that we have tried is with **max_depth = 20, min_sample_split = 5, criterion = “entropy”**; obtaining a similar performance of both sets: training and test. We can interpret that as **a great generalization** of the model.

Error analysis: Are there specific dialog acts that are more difficult to classify? Are there particular utterances that are hard to classify (for all systems)? And why?

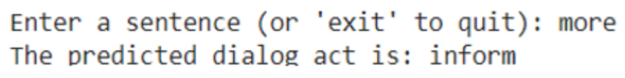
If we look more precisely, we can see that there are few differences on the performance between dialogue acts labels.

All the labels have good performances but “ack”,”deny”, “reqmore” , “repeat”, “restart”. Those ones have **a precision of 0**. The next better precision value is for confirm, **with 0.78** and all others are **above 0.9**.

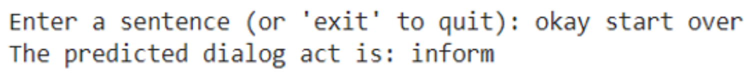
We have to take into account that all that labels have a small number of samples and that can affect the model training. In fact, they are the labels which have less instances. Then it’s probable that there exists **a relation between the number of instances and the accuracy that the model achieves.**

Difficult cases: Come up with two types of ‘difficult instances’, for example utterances that are not fluent (e.g. due to speech recognition issues) or the presence of negation (I don’t want an expensive restaurant). For each case, create test instances and evaluate how your systems perform on these cases.

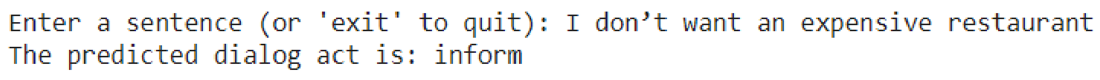
Difficult instances:



```
Enter a sentence (or 'exit' to quit): more
The predicted dialog act is: inform
```



```
Enter a sentence (or 'exit' to quit): okay start over
The predicted dialog act is: inform
```



```
Enter a sentence (or 'exit' to quit): I don’t want an expensive restaurant
The predicted dialog act is: inform
```

As we can see, all these difficult instances are classified as an “inform” label. Therefore we can abstract that there is a tendency of classifying difficult instances as “inform”, which is the most common dialogue act.

System comparison

If we compare the machine learning model (Decision Tree classifier, in this case) to the baselines model, we can affirm that the performance of the decision tree is better. In fact, it continues to obtain greater accuracy while training with deduplicated data. At the same time, it’s true that there is a significant decrease in accuracy when the model works with that kind of data (and not the original dataset). Concretely, using the same model and deduplicated data we obtain the following accuracies:

```
Training Accuracy: 0.9218441273326016
Test Accuracy: 0.8694029850746269
```

As commented, we can observe that both accuracies are above 81.5%, rule-based system based on keyword matching accuracy. The decrease in accuracy, especially in the test set, after deduplication highlights the limitations of the original model, which benefited from repeated examples and potentially memorized frequent patterns. With fewer, more unique examples, the model's ability to generalize diminishes, revealing its tendency to overfit to the training data. This suggests that further model tuning, such as reducing tree complexity or using more advanced generalization techniques, may be necessary to improve performance on the deduplicated dataset.

3.2 Evaluate model Feed-Forward Neural Network

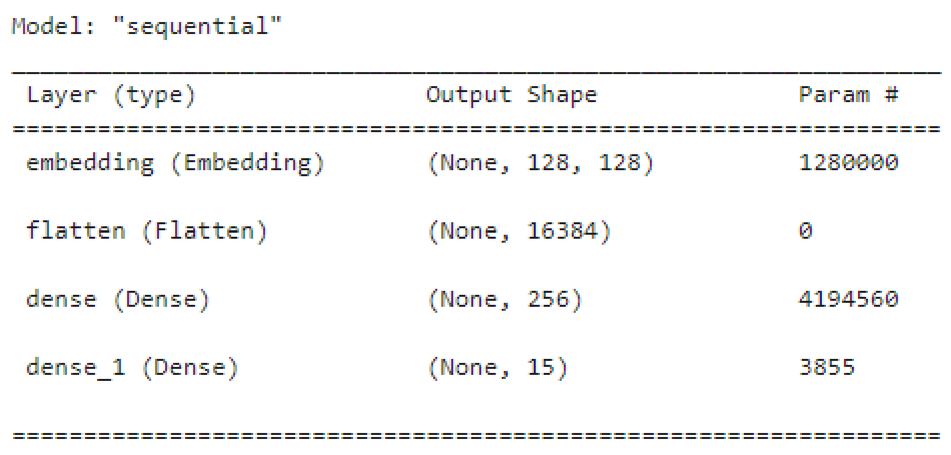
Original Data accuracy: 98.65%

Deduplicated Data Accuracy: 91.14%

Quantitative evaluation: Evaluate your system based on one or more evaluation metrics. Choose and motivate which metrics you use.

The model is a simple neural network trained on 5 epochs with a learning rate of 0.001.

Model summary:



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 128, 128)          1280000

 flatten (Flatten)           (None, 16384)             0

 dense (Dense)               (None, 256)               4194560

 dense_1 (Dense)             (None, 15)                3855

=================================================================
```

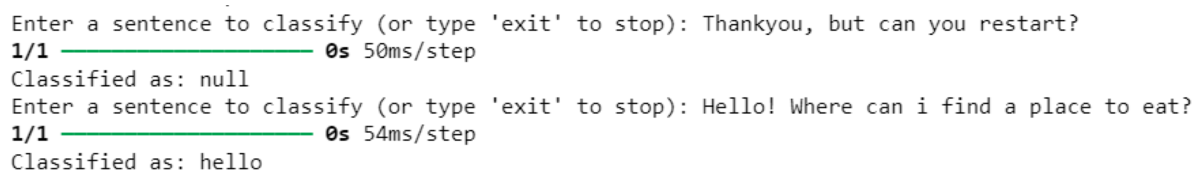
The categories that had much data such as inform are still performing well in the deduplicated data. We think the overall accuracy went down because if there is a lot of duplicated data. And you do a validation split on the data. Automatically training data will be in the test data aswell.

Error analysis: Are there specific dialog acts that are more difficult to classify? Are there particular utterances that are hard to classify (for all systems)? And why?

Because some classes like ‘ack’ and ‘reqmore’ already didn’t have a lot of data and were difficult to predict. It now even got harder resulting in a recall of 0.We can see that it is more difficult for the models to predict a sentence as null because it try’s to choose a label based on the words that are given.

Difficult cases: Come up with two types of ‘difficult instances’, for example utterances that are not fluent (e.g. due to speech recognition issues) or the presence of negation (I don’t want an expensive restaurant). For each case, create test instances and evaluate how your systems perform on these cases.

Underneath you can see 2 **difficult instances** we have tried on the model:



```
Enter a sentence to classify (or type 'exit' to stop): Thankyou, but can you restart?
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 50ms/step
Classified as: null
Enter a sentence to classify (or type 'exit' to stop): Hello! Where can i find a place to eat?
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 54ms/step
Classified as: hello
```

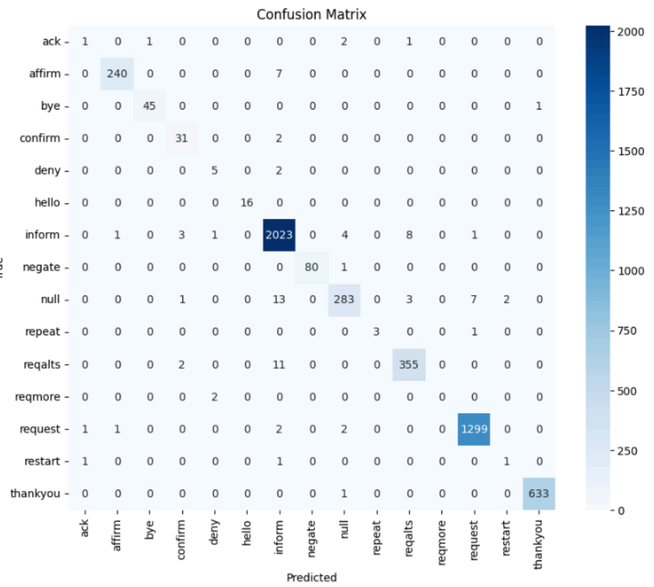
We can see here that the model has still some difficulty with composed sentences. This will always be a difficult thing for such text classification methods, but there are ways to train the model better on these examples. By training on such examples it can recognize certain words that are less important to classify the sentence. In this case the word ‘Hello’ overtook the classification to classify it as a request.
If you look back at the 81% from our baseline, these machine learning models perform way better, even with the deduplicated data, which means the data it is tested on is completely new to it. Eventually you can’t really test the performance difference between the original data and the deduplicated data, unless you get data from the outside to do the tests on.

Original data:

```
              precision    recall  f1-score   support

         ack       0.00      0.00      0.00         5
      affirm       1.00      0.99      0.99       247
         bye       0.94      0.98      0.96        46
     confirm       0.97      0.88      0.92        33
        deny       0.83      0.71      0.77         7
       hello       1.00      0.88      0.93        16
      inform       0.99      0.99      0.99      2041
      negate       1.00      0.99      0.99        81
        null       0.92      0.99      0.95       309
      repeat       1.00      0.75      0.86         4
     reqalts       0.96      0.97      0.96       368
     reqmore       0.67      1.00      0.80         2
     request       1.00      0.99      1.00      1305
     restart       1.00      0.33      0.50         3
    thankyou       1.00      1.00      1.00       634

    accuracy                           0.99      5101
   macro avg       0.88      0.83      0.84      5101
weighted avg       0.98      0.99      0.98      5101
```


Confusion Matrix

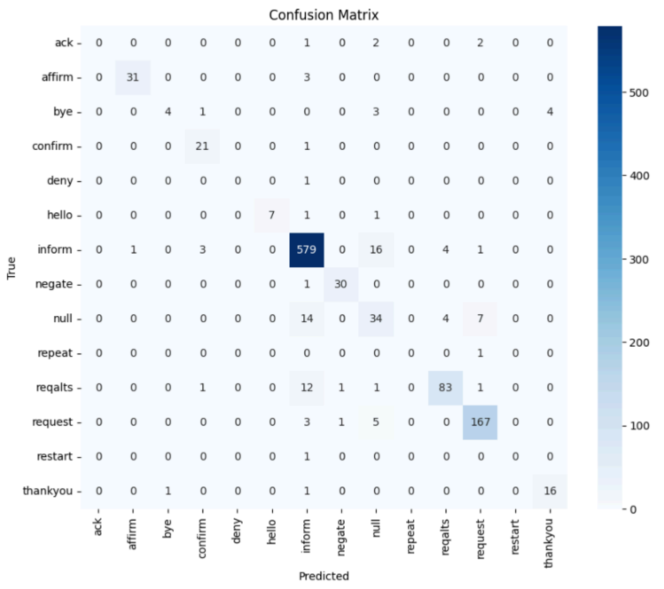
True \ Predicted	ack	affirm	bye	confirm	deny	hello	inform	negate	null	repeat	reqalts	reqmore	request	restart	thankyou
ack	1	0	1	0	0	0	0	0	2	0	1	0	0	0	0
affirm	0	240	0	0	0	0	7	0	0	0	0	0	0	0	0
bye	0	0	45	0	0	0	0	0	0	0	0	0	0	0	1
confirm	0	0	0	31	0	0	2	0	0	0	0	0	0	0	0
deny	0	0	0	0	5	0	2	0	0	0	0	0	0	0	0
hello	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0
inform	0	1	0	3	1	0	2023	0	4	0	8	0	1	0	0
negate	0	0	0	0	0	0	0	80	1	0	0	0	0	0	0
null	0	0	0	1	0	0	13	0	283	0	3	0	7	2	0
repeat	0	0	0	0	0	0	0	0	0	3	0	0	1	0	0
reqalts	0	0	0	2	0	0	11	0	0	0	355	0	0	0	0
reqmore	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
request	1	1	0	0	0	0	2	0	2	0	0	0	1299	0	0
restart	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0
thankyou	0	0	0	0	0	0	0	0	1	0	0	0	0	0	633

Deduplicated data:

```
              precision    recall  f1-score   support

         ack       0.00      0.00      0.00         5
      affirm       0.97      0.91      0.94        34
         bye       0.80      0.33      0.47        12
     confirm       0.81      0.95      0.88        22
        deny       0.00      0.00      0.00         1
       hello       1.00      0.78      0.88         9
      inform       0.94      0.96      0.95       604
      negate       0.94      0.97      0.95        31
        null       0.55      0.58      0.56        59
      repeat       0.00      0.00      0.00         1
     reqalts       0.91      0.84      0.87        99
     reqmore       0.00      0.00      0.00         0
     request       0.93      0.95      0.94       176
     restart       0.00      0.00      0.00         1
    thankyou       0.80      0.89      0.84        18

   micro avg       0.91      0.91      0.91      1072
   macro avg       0.58      0.54      0.55      1072
weighted avg       0.90      0.91      0.90      1072
```


Confusion Matrix

True \ Predicted	ack	affirm	bye	confirm	deny	hello	inform	negate	null	repeat	reqalts	request	restart	thankyou
ack	0	0	0	0	0	0	1	0	2	0	0	2	0	0
affirm	0	31	0	0	0	0	3	0	0	0	0	0	0	0
bye	0	0	4	1	0	0	0	0	3	0	0	0	0	4
confirm	0	0	0	21	0	0	1	0	0	0	0	0	0	0
deny	0	0	0	0	0	0	1	0	0	0	0	0	0	0
hello	0	0	0	0	0	7	1	0	1	0	0	0	0	0
inform	0	1	0	3	0	0	579	0	16	0	4	1	0	0
negate	0	0	0	0	0	0	1	30	0	0	0	0	0	0
null	0	0	0	0	0	0	14	0	34	0	4	7	0	0
repeat	0	0	0	0	0	0	0	0	0	0	0	1	0	0
reqalts	0	0	0	1	0	0	12	1	1	0	83	1	0	0
request	0	0	0	0	0	0	3	1	5	0	0	167	0	0
restart	0	0	0	0	0	0	1	0	0	0	0	0	0	0
thankyou	0	0	1	0	0	0	1	0	0	0	0	0	0	16

Conclusion:

We can see that the model with the duplicate data is performing better on all fronts.

Though I think that testing on this data is not a good way of checking if that model is performing better. By using a training, develop and a test dataset you could make better conclusions concerning the performance of the model in a real situation.

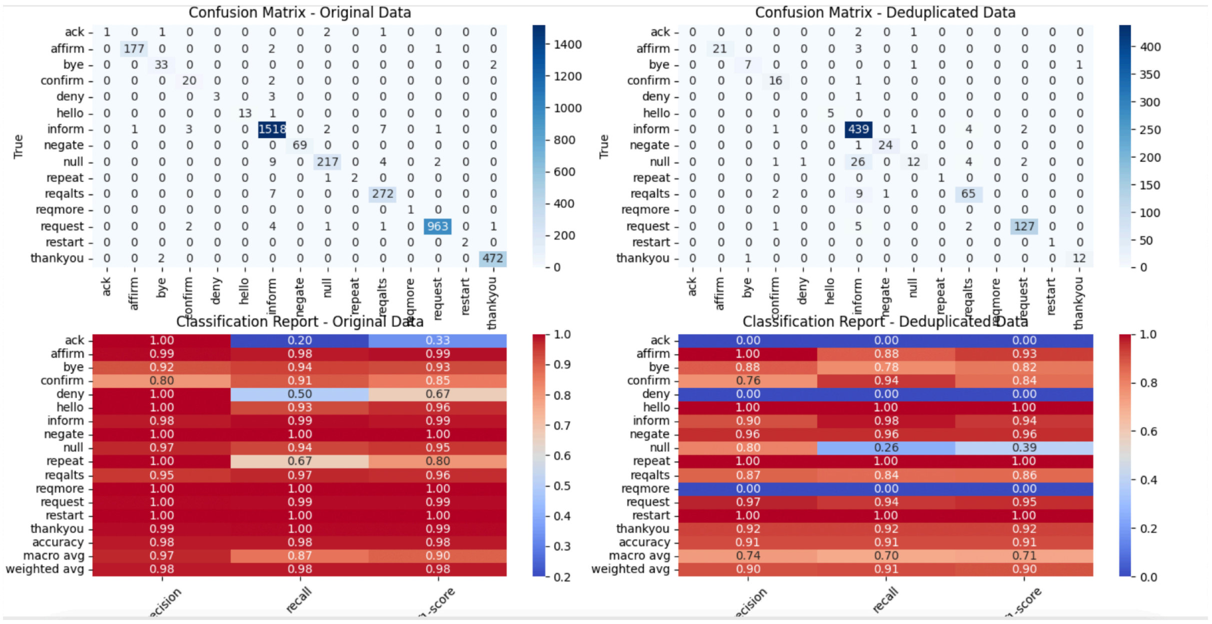
Looking at the current results the model trained on the original data looks like the better one.

3.3 Evaluate model SVM

Original Data Accuracy: 98.35%

Deduplicated Data Accuracy: 90.80%

Quantitative evaluation: Evaluate your system based on one or more evaluation metrics. Choose and motivate which metrics you use.



We can see that the model with the duplicate data is performing better on all fronts. As we could see in the fnn machine learning model, categories with more data, like "inform," still perform well on the deduplicated data. The overall accuracy dropped, likely because the duplicated data caused overlap between training and test sets during validation. Classes like "ack" and "reqmore," which had fewer samples and were already hard to predict, became even harder, resulting in a recall of 0. The model also struggles with predicting "null" sentences, as it tends to choose labels based on the available words.

These phenomenon presents that the original data and deduplicated data have similar performance across different machine learning models.

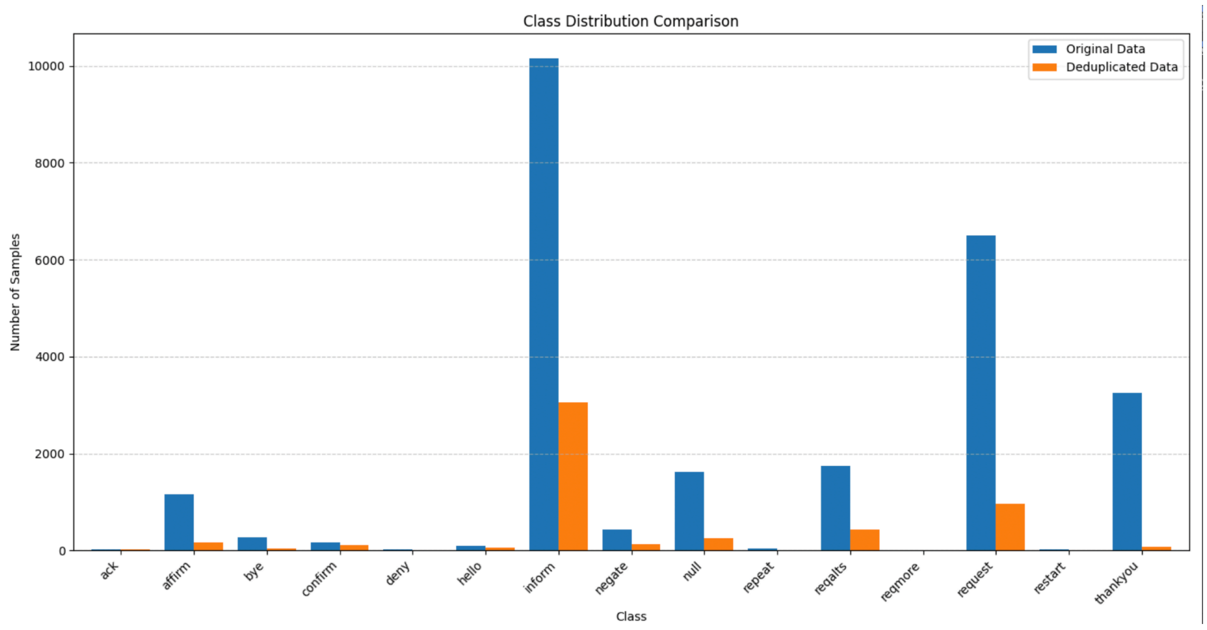
Error analysis: Are there specific dialog acts that are more difficult to classify? Are there particular utterances that are hard to classify (for all systems)? And why?

1. ‘inform’ and ‘request’ categories still perform well in the de-duplicated data, but other categories (e.g., ack, deny, null) perform poorly.
2. ‘ack’ and ‘deny’ categories have a recall of 0.00, probably because these categories have very few samples in the test set and the model cannot learn enough features.

3. ‘hello’ and ‘repeat’ categories perform well in the de-duplicated data, especially in terms of precision and recall.

There are several problems with the original and de-duplicated data, one is that the fifteen categories in the original data are highly skewed, and the second is that the categories in the de-duplicated data are not balanced, and the number of samples in certain categories is significantly reduced.

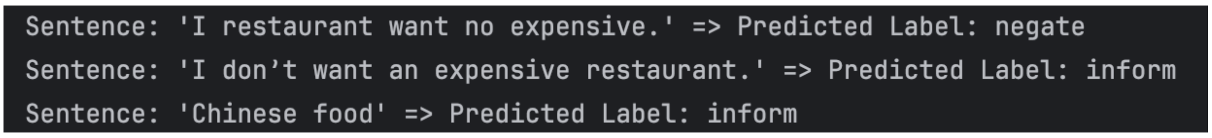
Ultimately, **the distribution of categories** in the raw and de-emphasised data needs to be considered.



In the table above, we could see that some classes(‘inform’, ‘request’ and ‘thankyou’) include a lot of duplicate data. Nevertheless, more training data in the deduplicated data tends to perform well in the test.

Difficult cases: Come up with two types of ‘difficult instances’, for example utterances that are not fluent (e.g. due to speech recognition issues) or the presence of negation (I don’t want an expensive restaurant). For each case, create test instances and evaluate how your systems perform on these cases.

After checking the original label and the label produced by the SVM, we find a special difficult instance, which could be presented as the type of food and they appeared many times in original data. For example, ‘Chinese food’, ‘Vietnamese food’, ‘German food’, they are labeled as ‘inform’, ‘null’ or ‘request’.


```
Sentence: 'I restaurant want no expensive.' => Predicted Label: negate
Sentence: 'I don’t want an expensive restaurant.' => Predicted Label: inform
Sentence: 'Chinese food' => Predicted Label: inform
```

The case of ‘I restaurant want no expensive.’ and ‘I don’t want an expensive restaurant.’ presents that SVM is not good to differentiate between similar sentences, which may due to this is a method based on word rather than semantics.

3.4 System comparison

How do the systems compare against the baselines, and against each other? What is the influence of deduplication? Which one would you choose for your dialog system?

We tried three machine learning models, all of which performed better than the two baseline systems, with little difference between them. Data deduplication led to a decrease in classification accuracy, probably because the fixed restaurant conversations were relatively repetitive, which led to more similarities between the training data and the test data. In the end, we will choose FNN as our restaurant dialog system.