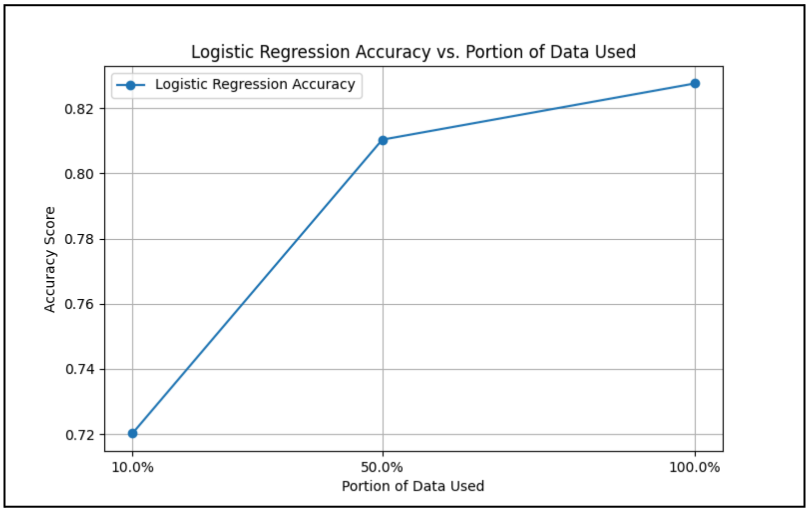
Natural Language Processing - Ex4:
Shir Rashkovits & Ori Dvir

Classification Tasks:

1. Plot the model accuracy results as a function of the portion of the data:



The model's performance improved monotonically as the size of the training data increased.

2.

Calculate and report the model's **average loss** during training and the **accuracy** on the **validation set**, **for each epoch**:

Portion 1: (10%) We can see a significant improvement from first epoch results to second, and a weaker improvement from the second epoch to the third one.

Epoch 1:

- evaluation loss: 0.9553961157798767
- evaluation accuracy: 0.7042440318302388

Epoch 2:

- evaluation loss: 0.4640953838825226
- evaluation accuracy: 0.8488063660477454

Epoch 3:

- evaluation loss: 0.41337600350379944
- evaluation accuracy: 0.8527851458885941

Test evaluation:

Portion: 0.1, Accuracy: 0.8527851458885941, Loss: 0.41337600350379944

Portion 2: (50%) In the 50% and 100% cases we see that after one epoch the model get pretty good results which the additional epochs can improve but also can harm, but non significant changes are seen

Epoch 1:

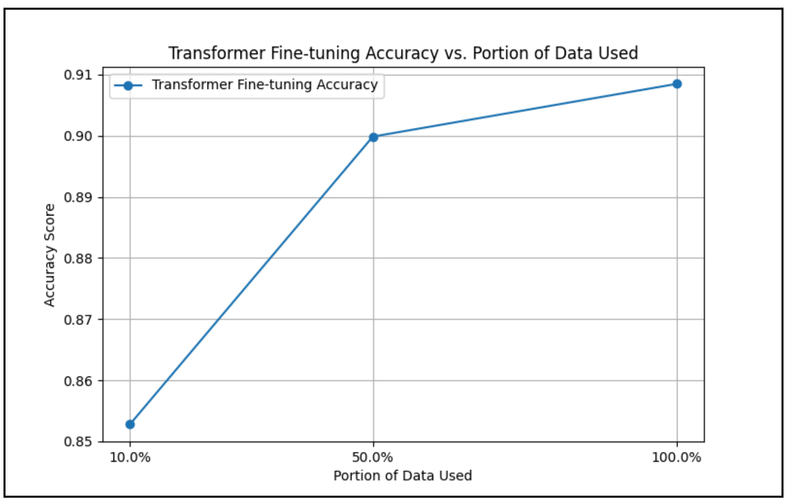
- evaluation loss: 0.32671958208084106,
- evaluation accuracy: 0.8879310344827587

Epoch 2:

- evaluation loss: 0.40287598967552185

- evaluation accuracy: 0.8759946949602122
Epoch 3:
- evaluation loss: 0.40224453806877136
- evaluation accuracy: 0.899867374005305
Portion: 0.5, Accuracy: 0.899867374005305, Loss: 0.40224453806877136
<u>Portion 3:</u> (100%)
Epoch 1:
- evaluation loss: 0.3550274074077606
- evaluation accuracy: 0.8846153846153846
Epoch 2:
- evaluation loss: 0.3540601134300232
- evaluation accuracy: 0.9058355437665783
Epoch 3:
- evaluation loss: 0.40433353185653687
- evaluation accuracy: 0.9084880636604774
Portion: 1.0, Accuracy: 0.9084880636604774, Loss: 0.40433353185653687

Plot the model accuracy results as a function of the portion of the data:



The model's performance improved monotonically as the size of the training data increased.

3. Report the accuracy you got.

```
Zero-shot result:
0.7858090185676393
```

4. Compare the three models:
(a) Which model had the highest accuracy?

Answer: as we can see from the results above, the fine tuned transformer had the highest accuracy results (> 0.85), as when only training on 10% of the training set this model achieved higher accuracy both than the Logistic Regression model when trained on 100% (<0.83) of the training set and from the Zero-Shot model (<0.8). These results could be attributed to the fact that the pre-trained fine tuned transformer was benefiting from both pre-training on a Masked Language Modeling task, thus gaining an internal understanding of language semantics and structure, which was further refined through task-specific fine-tuning. The combination of broad pre-training and targeted refinement, positions the fine-tuned transformer to excel in classification tasks, even with limited training data.

(b) Which model was the most sensitive to the size of the training set?

Answer: The logistic regression model, employing TFIDF encoding, was the most sensitive to the size of the training set among the models evaluated. This model's approach, grounded in traditional machine learning techniques, necessitates learning exclusively from the training data at hand, lacking any pre-established knowledge base. Consequently, its performance is directly proportional to the volume and diversity of the dataset, with limited data leading to underdeveloped predictive capabilities. In contrast, the fine-tuned transformer model, specifically 'distilroberta-base', benefits from a comprehensive pre-training on a broad spectrum of textual data. This foundational training endows the model with a nuanced understanding of language semantics, equipping it with the ability to leverage its learned representations effectively during task-specific fine-tuning. As a result, the transformer demonstrates impressive resilience to variations in training data size, maintaining robust performance even when trained on substantially smaller datasets. This inherent adaptability, setting the fine-tuned transformer model apart from the log-linear model in terms of sensitivity to the amount of training data.

(c) Mention 2 pros and 2 cons of the zero-shot model (in comparison to the other models).
The zero-shot classification model, particularly when utilizing the 'cross-encoder/nli-MiniLM2-L6-H768' for classification tasks, represents a paradigm shift in how we approach text categorization. Unlike traditional models that require extensive training on labeled data within their target domain, the zero-shot model leverages a pre-trained Natural Language Inference (NLI) transformer. This model assesses the fit between an input text and a set of candidate labels by treating the task as a series of NLI problems. It evaluates the likelihood of each label being a correct hypothesis for the given text premise, selecting the label with the highest entailment score as the classification outcome. This innovative approach allows the zero-shot model to apply its learned inference capabilities to classify text into categories it was never explicitly trained on, showcasing remarkable flexibility and efficiency. As seen in the results, the zero shot model achieve lower accuracy score, but still significantly better than a random guess

<u>Pros:</u>

1. Adaptability Across Tasks: One of the most significant advantages of the zero-shot model is its ability to classify text into categories it has never seen during training. This flexibility allows for the model's application across a wide range of tasks without needing task-specific data, making it highly adaptable to new domains or classifications without additional training.

2. Cost and Time Efficiency: Zero-shot models eliminate the need for gathering, annotating, and training on a new dataset for each specific task. This reduces both the time and financial costs associated with data preparation and model training, offering a quick and cost-effective solution for text classification problems, especially in scenarios where annotated data is scarce or unavailable.

Cons:

- Lower Accuracy in Specialized Domains: While zero-shot models are incredibly versatile, their generalized pre-training may result in lower accuracy for highly specialized or niche domains compared to models fine-tuned on specific datasets. The broad knowledge base of zero-shot models, while extensive, may lack the depth of understanding required for intricate distinctions within specialized fields, potentially leading to less precise classifications than those achieved by fine-tuned or task-specific models.
- Dependence on Label Formulation: The effectiveness of a zero-shot model is highly dependent on the formulation and relevance of the labels to the input text. Since the model infers the most appropriate label based on its pre-trained understanding of language, inaccuracies in label description or ambiguity can lead to incorrect classifications. This necessitates careful consideration and potentially manual refinement of labels, which can be a nuanced and time-consuming process. For example, a broad label like "science, electronics" might ambiguously group diverse discussions, from electronic devices to engineering theories, under a general category. This could compromise classification accuracy. In contrast, specific labels such as "electronic circuit design" enhance precision and relevance.