Mini project 3

Explanation shifts and diversity

Henok Lachmann and Noel Bastubbe

Distribution shifts

- We were still using random forests as models
- We reduced the parameter space to reduce time to train as focus of this exercise was not on optimal models
- We started with only students and for the distribution shift added one non-student after the other in random order without repetition in a loop
- After every addition, the precision and recall were logged
- For both 5% and 10% we observed no reaching of the thresholds, also not after multiple repetitions
- recall typically near 1.0

Decision tree

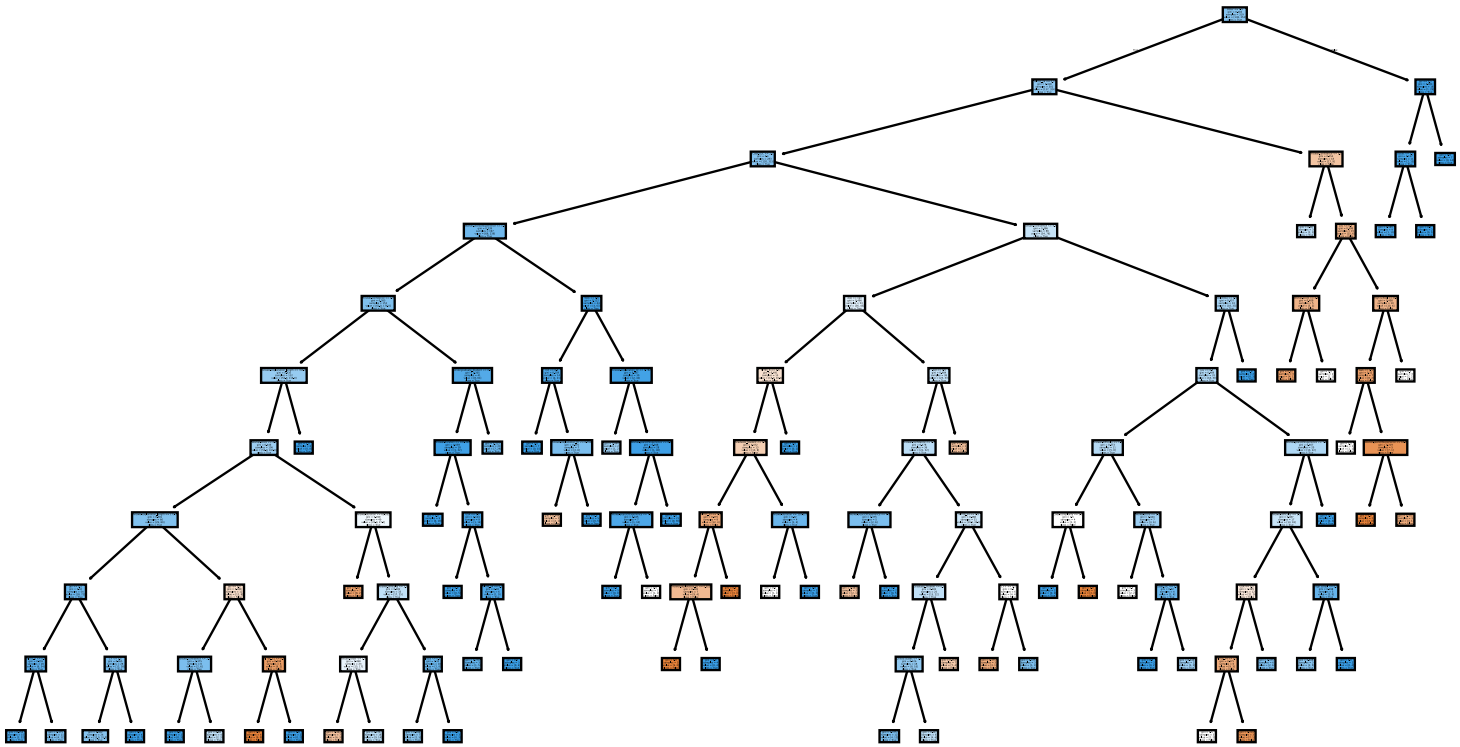


Figure: Decision tree trained on students only

Precision & Recall

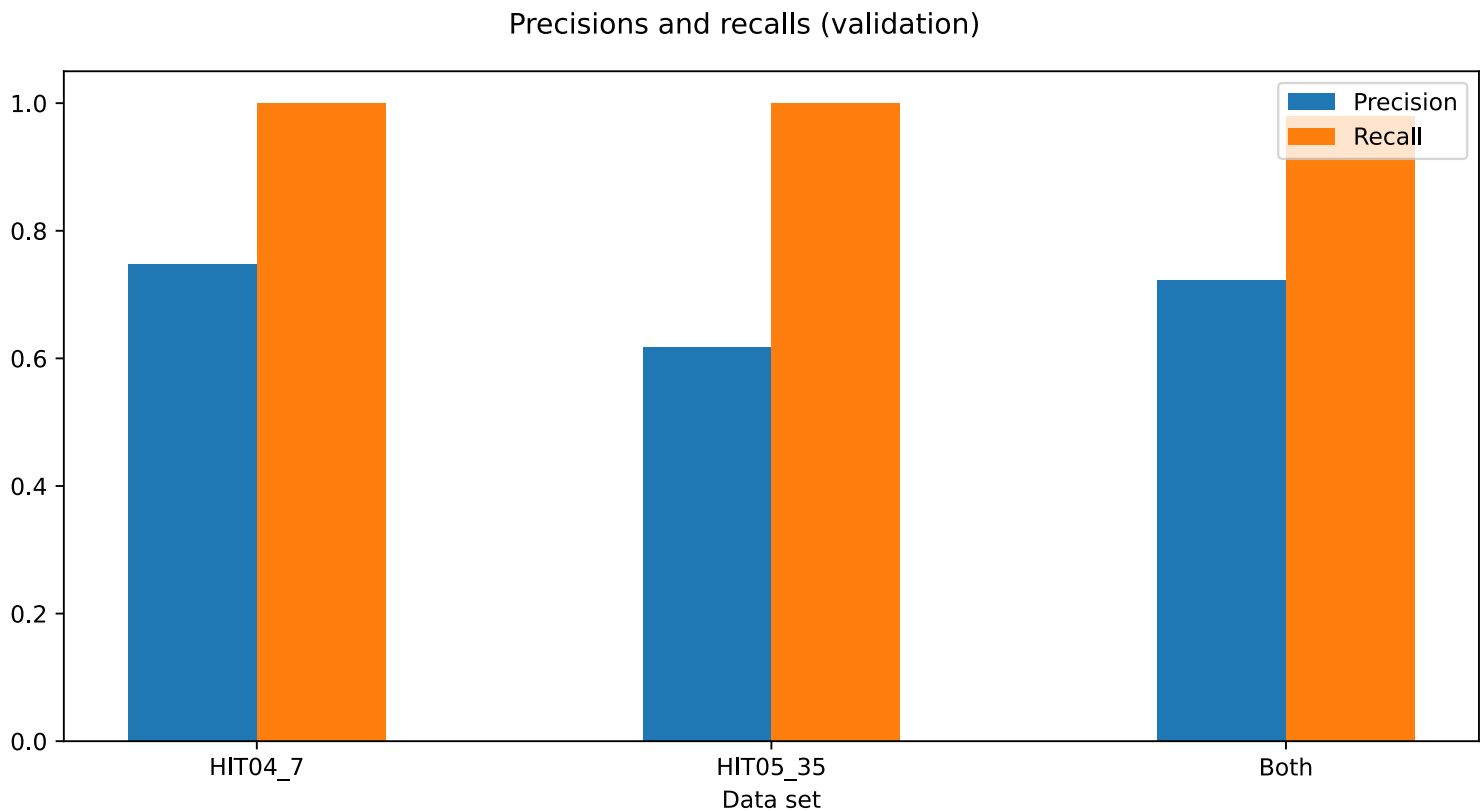


Figure: Precision & Recall for holdout set on all data

Precision & Recall

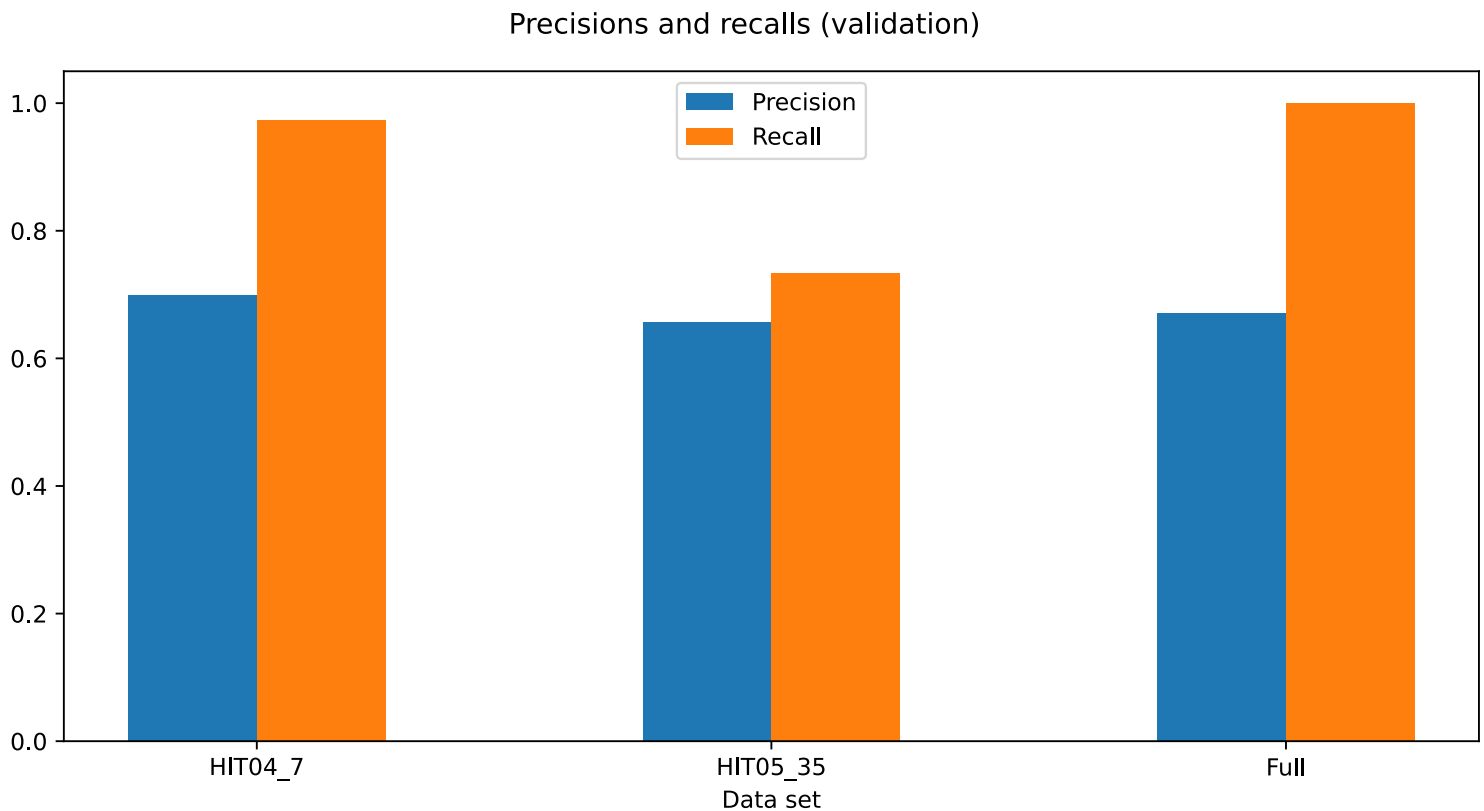


Figure: Precision & Recall for holdout set on student data

Distribution shift

- On average, we can add 0 non-students to make our model perform comparably to the model from exercise 2
- Both graphs show that student data is sufficient to capture and generalize the full dataset of correct reports

Necessary and sufficient explanations

- For readability we chose the Flesch-Kincaid Reading Ease. This measurement was design for general text and should roughly return a value between 100 and 0 where 100 is translated to "Very easy to read. Easily understood by an average 11-year-old student." and 0 is translated to "Extremely difficult to read. Best understood by university graduates.". In general this measurement punishes long words and long sentences.
- For Semantic Similarity we chose the BLEU score as we are familiar with it from the last exercise. As the BLEU score needs the compared text to be of similar length we have to make some adjustments and accept a very low BLEU score as threshold.
- We generate the ground truths using ChatGPT. We provide ChatGPT with all true positive answers per task and the following prompt:

Prompt

The following number of explanations lines contain number of explanations correct explanations for the same bug. I would like you to generate a single explanation that is as short and concise as possible and at the same time contains the information that would be necessary for another person to fix the bug. The text you create should be 3 to 5 sentences long, with an emphasis on readability.

Necessary and sufficient explanations

- We use a new Chat for each task in order to provide equal conditions to ensure comparability. The prompt is designed to generate equally long texts as we thought same text length would make the Flesch-Kincaid Reading Ease more relevant.

Ground truths

The bug occurs because the conditional check on line 279 incorrectly rejects any negative `minutesOffset` value, even though the documentation states that values between -59 and +59 are valid. Since `minutesOffset` is set to -15, the condition incorrectly triggers an `IllegalArgumentException`. To fix this, update the check to `if (minutesOffset < -59 || minutesOffset > 59)`, allowing valid negative offsets while still enforcing the correct bounds.

- Flesch-Kincaid: 29.5

Ground truths

The variable `g` must be in the range 0-255, but the calculation is producing a negative value, likely because `value` is used instead of `v` on line 117. The `value` variable can exceed `lowerBound` and `upperBound`, leading to an invalid `g`. To fix this, replace `value` with `v` in the calculation. Additionally, ensure that `g` is clamped within the valid range before passing it to the `Color` constructor.

- Flesch-Kincaid: 53.2

Ground truths

The issue is that the `pos` variable is being incremented incorrectly, leading to an `IndexOutOfBoundsException` when accessing the input string. Specifically, `pos` is updated in both line 89 and line 95, sometimes by more than one character, without ensuring it remains within the valid range of `input.length()`. This is particularly problematic when handling surrogate pairs, as `Character.codePointAt` can return values requiring `pos` to be incremented by 2 instead of 1. To fix this, add a boundary check before updating `pos` to prevent it from exceeding the string length.

- Flesch-Kincaid: 45.0

Ground truths

The bug occurs because `getDataItem` is called with `this.minMiddleIndex` instead of `this.maxMiddleIndex` when retrieving period start and end times (lines 299-302). This causes incorrect calculations for `maxMiddleIndex`, leading to an assertion failure. To fix the issue, replace `this.minMiddleIndex` with `this.maxMiddleIndex` in the affected lines to ensure the correct index is used for retrieving time period data.

- Flesch-Kincaid: 49.6

Ground truths

The issue arises because when both `array` and `element` are null, the inferred type defaults to `Object.class`, leading to a `ClassCastException` when attempting to cast an `Object[]` to `String[]`. The problem originates from how the type is determined: `type = (array != null) ? array.getClass().getComponentType() : (element != null) ? element.getClass() : Object.class;`. To fix this, modify the logic to throw an `IllegalArgumentException` when both parameters are null instead of defaulting to `Object.class`. This ensures that an invalid type is never inferred, preventing the casting issue.

- Flesch-Kincaid: 60.1

Ground truths

The bug occurs because a `long` variable is used to store a potentially decimal value, causing the loss of fractional precision. Specifically, when converting a `double` to `long`, the decimal part is truncated, which leads to incorrect comparisons and prevents the while loop from terminating. This results in an infinite loop, preventing the `addNumber` method from completing execution. To fix the issue, ensure that the variable retains its decimal value.

- Flesch-Kincaid: 32.0

Ground truths

The issue occurs on line 910, where `array[i]` is dereferenced without checking for null. Specifically, the second element of the input array is null, causing a `NullPointerException` when `getClass()` is called on it. To fix this, modify the loop to check if `array[i]` is null before calling `getClass()`, and assign `null` to `classes[i]` in that case. Example fix: `classes[i] = (array[i] == null) ? null : array[i].getClass();`.

- Flesch-Kincaid: 59.8

Ground truths

The bug occurs because the third character (`ch3`) in the locale string is an underscore (_), which falls outside the range of uppercase letters (`A-Z`). The condition on line 115 incorrectly checks if `ch3` is either greater than `A` or less than `Z`, causing the exception when it encounters an underscore. This violates the expected format, where the country code should consist of uppercase letters, not underscores. To fix the issue, modify the validation to handle underscores appropriately or ensure the input format strictly adheres to the expected pattern.

- Flesch-Kincaid: 46.8

Necessary and sufficient explanations

- Our Flesch-Kincaid Reading Ease scores are between 30 and 60 where 30 translates to "Very difficult to read. Best understood by university graduates." and 60 translates to "Fairly difficult to read.". This is around the level we expected as the data proves that these tasks are so difficult that a lot of trained professionals were not able to solve them.
- As the readability score differ between tasks we set task specific tresholds for readability at 10% of the readability of the ground truth. This means our thresholds are the following.

Necessary and sufficient explanations

Task	FK score	BLEU score
Task 1	26.5	0.381079
Task 2	47.8	0.300324
Task 3	40.5	0.259241
Task 4	44.6	0.210321
Task 5	54.1	0.247951
Task 6	28.8	0.179082
Task 7	53.8	0.241289
Task 8	42.1	0.309860

Diverse explanations

- We use different features, but for simplicity, we only choose a few which in our opinion summarize an entry rather well. We chose
  - `Worker.score`
  - `Worker.yearsOfExperience`
  - `Answer.duration`
- We calculated the $L2$ metric and summed the results. In order for the magnitude not to be a factor (especially for `Answer.duration`), we normalized features to a maximum of 1

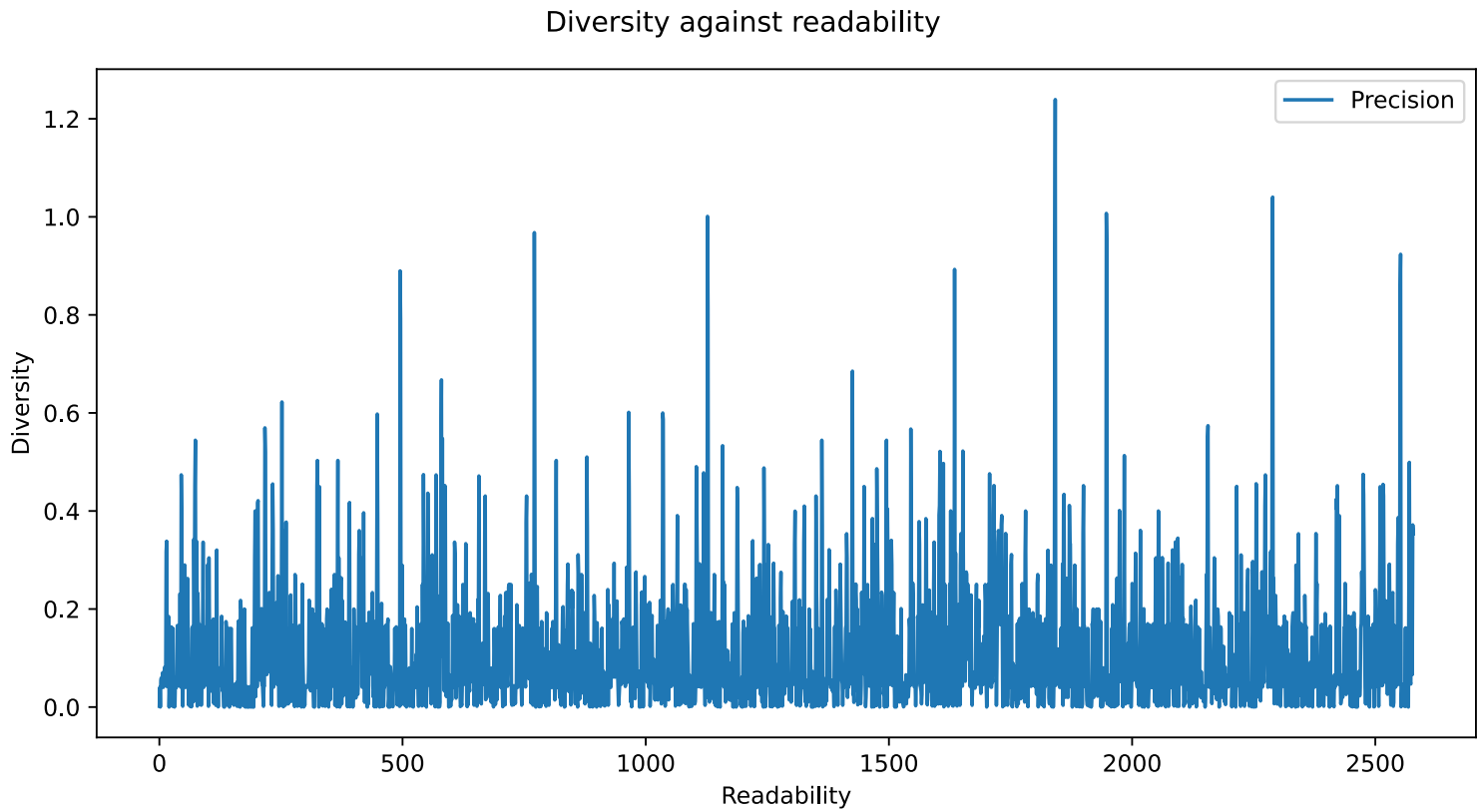
Max readability and semantic similarity

- Max readability (among all data): 68.77
- Max semantic similarity (among all data): 0.413342

Max diversity

- Min diversity (with respect to readability): $4.28 \cdot 10^{-9}$
- Max diversity (with respect to readability): $1.24 \cdot 10^{0}$

Diversity against readability



- There is no real trend visible when plotting against readability