Burp NLP: A Rick and Morty Text analysis

Text Analysis

an der Fakultät für Wirtschaft
im Studiengang Wirtschaftsinformatik

an der
DHBW Ravensburg

Verfasser:	Anton Geiger
Ausbildungsbetrieb:	Festo SE & Co KG
Anschrift:	Ruiter Straße 82
	73734 Esslingen Berkheim
Dozent:	Prof. Dr. Oliver Sampson
Abgabedatum:	31.3.2025

Inhaltsverzeichnis

Abkürzungsverzeichnis

LSTM	Long Short Term Memory
BERT	bidirectional encoder representation transformer
LDA	Latend Dirichlet Allocation
DTM	Document Term Matrix
BOW	Bag of words
TF-IDF	Term Frequency - Inverse document frequency

Glossar

Abbildungsverzeichnis

Tabellenverzeichnis

Formeln

1 Einleitung

2 Hauptteil

2.1 Data Understanding

For our project we use three different datasets. The first one is a dataset is downloaded from kaggle Prarabdha where we find the transcripts until season 5 formatted in the following shape 2.1. As the transcript also holds information about the scenery, there is not just spoken text included in there. We also find short descriptions about the sorroundings and the scenery. An example is also visible in the first row of table 2.1.

episode no.	speaker	dialouge
1	Rick	stumbles in drunkenly, and turns on the lights. Morty! You gotta come on. Jus'... you gotta come with me.
1	Morty	rubs his eyes. What, Rick? What's going on?
1	Rick	I got a surprise for you, Morty.

Tab. 2.1: Rick and Morty Transcript Dataset

The other datasets are retrieved by self written web scraping scripts from the Rick and Morty Wikipage and the IMDB website IMDB (2025) Rick and Morty wiki (2025). With the first script, we were able to create following table containing all episode descriptions provided in the Rick and Morty Fandom 2.2.

id	title	text
0	Pilot	In the middle of the night, an obviously drunk Rick bursts..
1	Lawnmower Dog	Jerry complains that the family dog, Snuffles, is stupid ...
2	Anatomy Park (Episode)	It's Christmas, and Jerry tries to enforce the idea ...

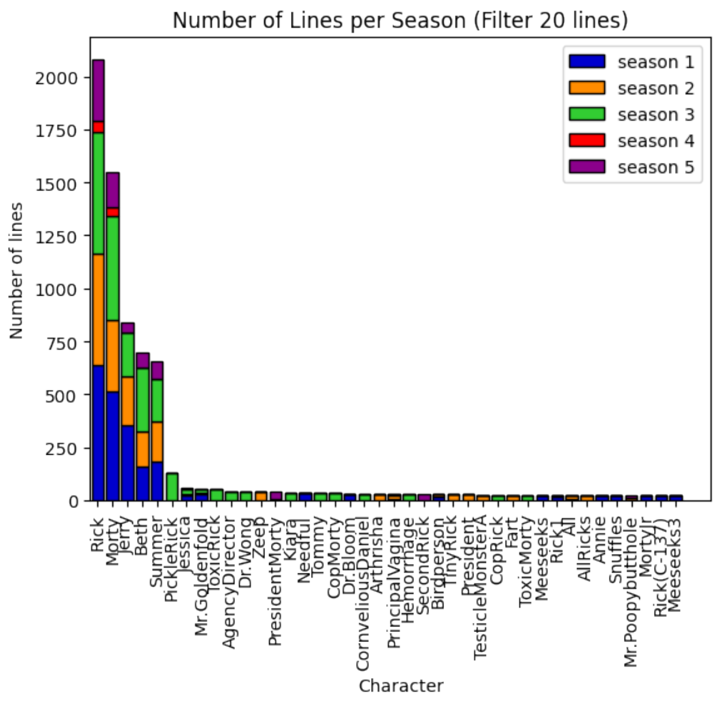
Tab. 2.2: Rick and Morty episode descriptions dataset

The last datasource stores information about the average IMDB rating per episode in the format provided in sample 2.3 where we also find the related season and title for each episode.

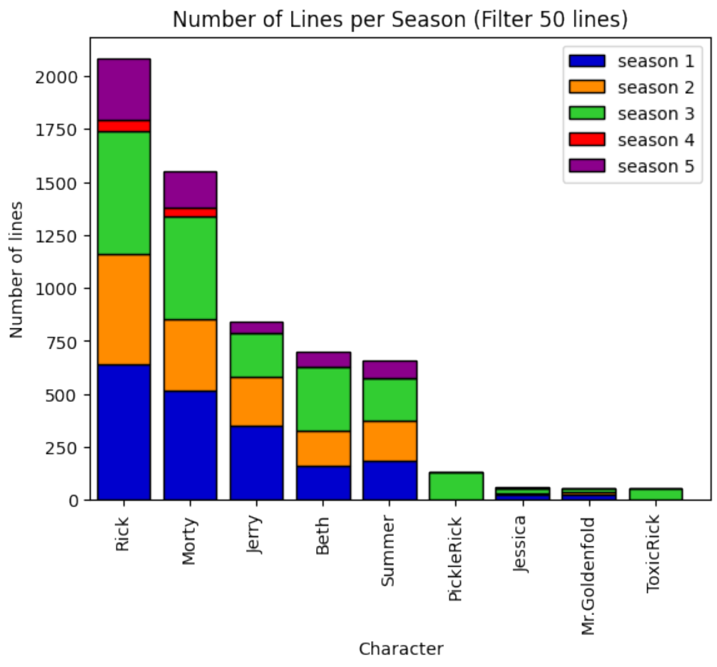
id	season	episode	title	rating
0	S1	E1	Pilot	7.9
1	S1	E2	Lawnmower Dog	8.6

Tab. 2.3: IMDB ratings per episode dataset

The rating score provided in the table is the average rating score on IMDB cacludated on every rating that was given to an episode. For the last two datasets we not just have the data until season 5 but we have every episode until season 7 maintained.
Further analysis show that there are in general 970 different speakers. The charts 2.1 visualize the speaker distribution by counting the lines that each character speaks in each season. In general, there are a lot of different characters participating in the series but the family Smith (Rick, Morty, Jerry, Beth, Summer) have by far the largest share when looking at the speaker distribution.



(a) Speaker distribution of speakers with more than 20 lines



(b) Speaker distribution of speakers with more than 50 lines

Abb. 2.1: Speaker distribution per lines

Surprisingly, the transcript dataset has some lacks. As we see in chart 2.3 that shows the number of dialogues per episode, the dataset is incomplete as there are a lot of empty episodes starting from season 4. The background colors highlight the intervals of the different seasons along the whole series.
The chart 2.2 provides a clearer picture on how often which main charatcer speaks in detail.

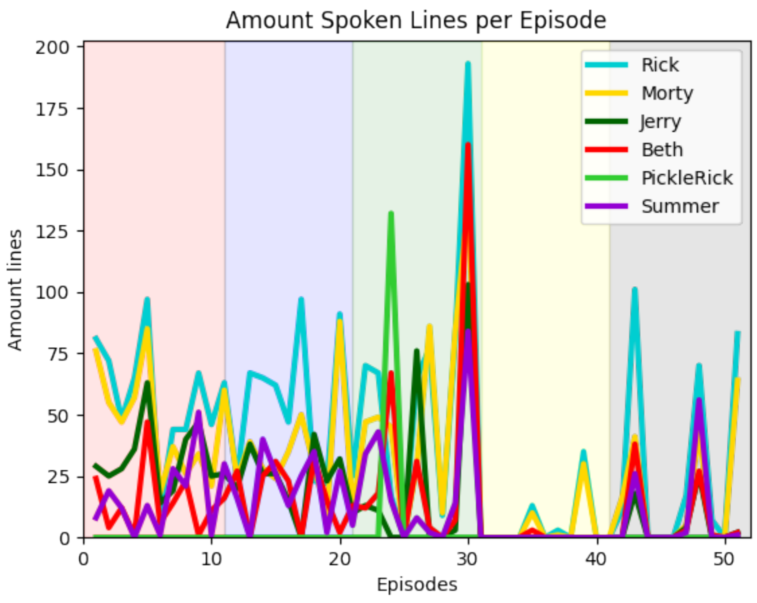


Abb. 2.2: Family Sanchez spoken lines in detail

In most of the episodes, Rick is the main character and holds the most shares in the speaking distribution. The other family members (Morty, Jerry, Beth and Summer) also speak in every episode but not that often as Rick. Side characters, like Pickle Rick often just have a large share in a few episodes as they often disapear after one episode.

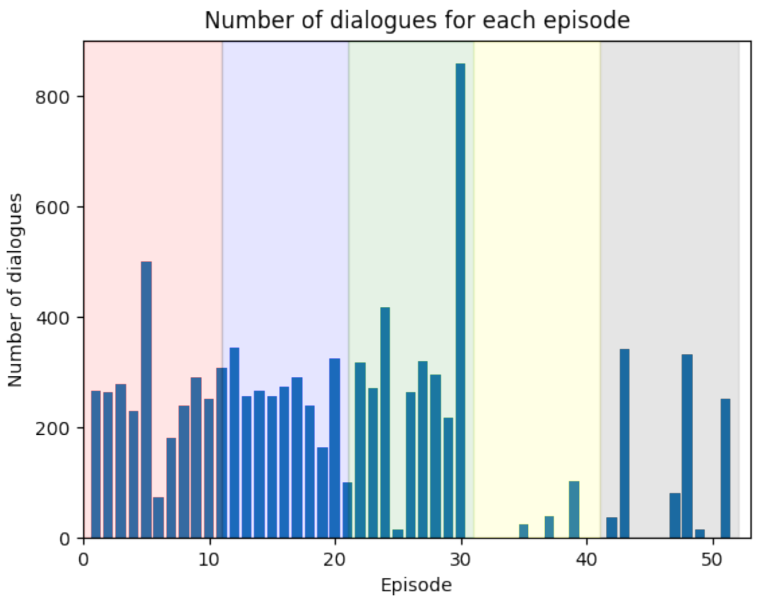
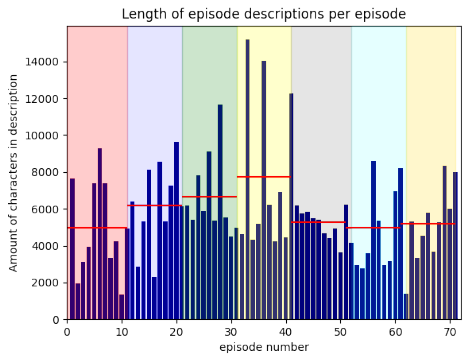
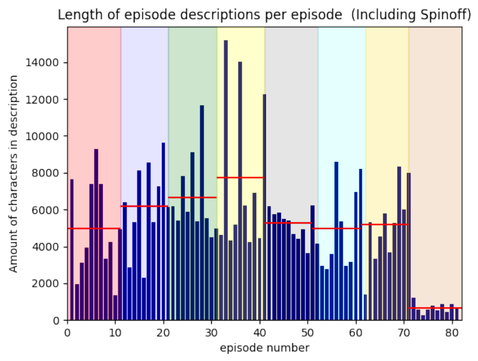


Abb. 2.3: Episode distribution

Looking deeply at the description dataset, we see two charts in 2.4 which compare the length the episode description for each episode. Surprisingly, the descriptions from episode 72 until 81 have a way shorter description than the episodes before. These episodes do not belong to the normal Rick and Morty series as they are spin-off episodes. Therefore, we now exclude those episodes in further analysis. The red lines show the average description length per each season. We see that the description length varies a lot for in each season. An example can be the comparison between episode 33 and 34, where episode 33's descriptions contains more than 3 times the amounf of characters as the description of episode 34. Another fact is that starting from season 5, the average number of chars in the descriptions seems to stay on a similar level. That is different compared to the first 4 seasons where in each season the average length of the description increased.



(a) Episode Length description distribution



(b) Episode Length description distribution (Including Spinoffs)

Abb. 2.4: Length of episode descriptions per episode

The last analysis step is an analysis regarding the top 100 most spoken words in the transcript. We used a wordcloud Mueller to visualize those in an appropriate way. The results can be seen in 2.5. After removing the character names, we find there colloquial language terms like 'yeah', 'gonna' in the wordcloud. That is why we can assume that the spoken english is not that advanced.

2.2 Data Preparation

Here, we describe the procedure we followed to clean and prepare the data. To clean and unify the data, we removed all nonspace characters, and set every character to a lower character.

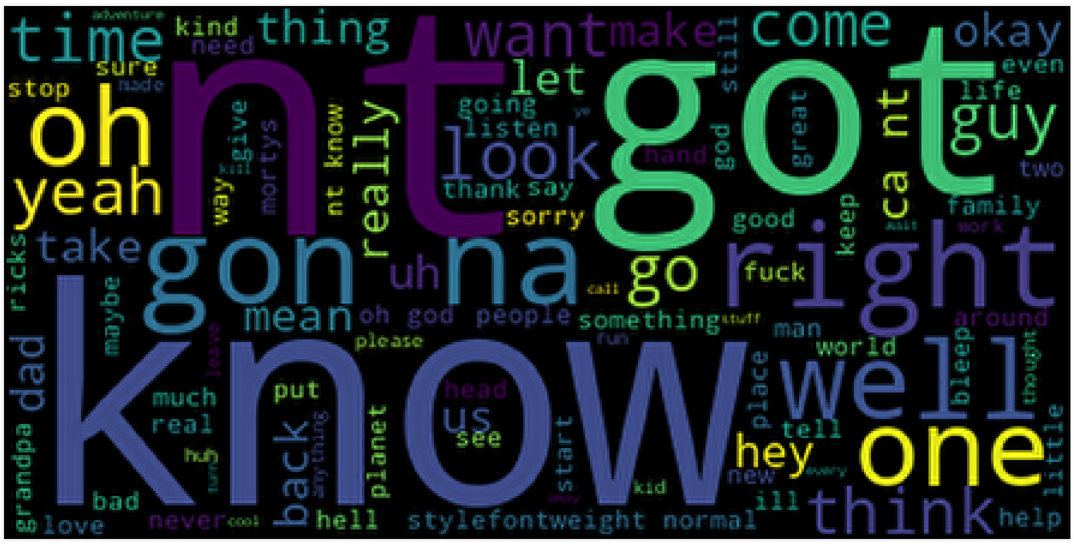


Abb. 2.5: Word Cloud transcript

As the rick and morty characters talk to each other in colloquial language, we removed some contractions like 'ain't' with 'is not'. Furthermore, there are some HTML tags in the dataset that were removed in the data preparation phase as well.
Using the spacy and nltk library, the text preparation pipeline consists of a custom build stop-wordremover and a custom build stemmer which uses the Porterstemmer to reduce every single token in a sentence to the word stem but ignores the character names for stemming. Porterstemmer looks at te suffix of the words and removes it if neccesary (vgl. Sushmita, 2014, S.3).
Before that, the sentences were tokenized using the small en web model spacy provides. To see the results, the first dialogue datapoint turns from

stumbles in drunkenly, and turns on the lights. Morty! You gotta come on. Jus'... you gotta come with me.
into
stumbl drunkenli turn light morty got tocom jus got come

2.3 Topic Modeling

In this section we want to examine the Rick and Morty series regarding the topics adressed throigh the seasons. To achieve this, we look at the transcripts and all of the episode descriptions, where we excluded the names of the main characters as they are not really an topic that holds content.
Topic Modeling Transcript
At first, the topics where analysed based on the transcript in the first 5 seasons. As the dataset contains roughly 9.000 datapoints, we use BERTopic Model to find out the most relevant topics. For using the BERT Topic model, we downloaded the all-MiniLM-L6v2 embedding model to

embedd the text. The embedded text was then reduced by UMAP and the clustered by the hierarchical HDBscan cluster algorithm using the eom cluster selection method. After clustering, BERTopic calculates the TF-IDF Scores for each documents and reduces the topics to the size the user wants them to have (vgl. Grootendorst, 2022, S.3). The resulting topics showcased in the image 2.6.

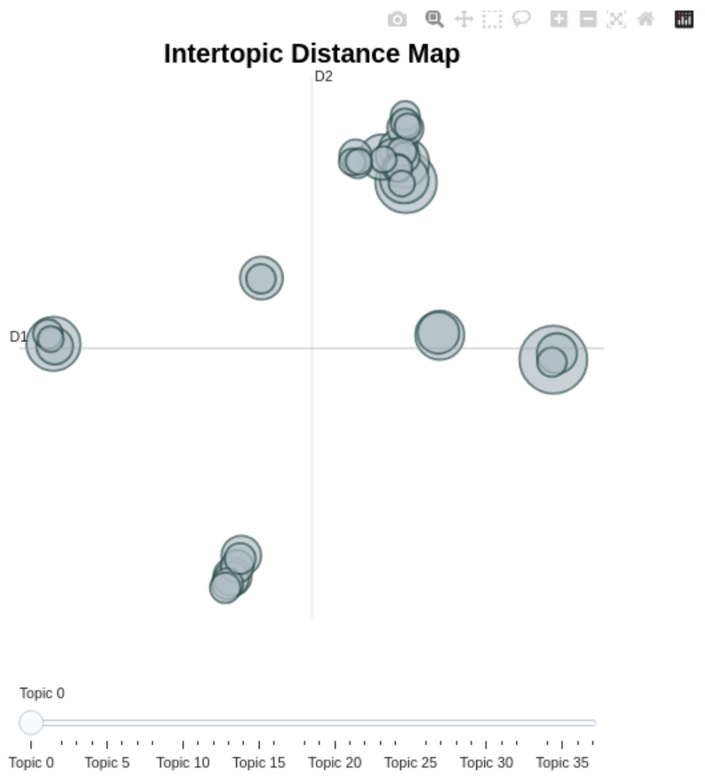


Abb. 2.6: BERT Topics for Transcrip (all seasons)

Setting the minimum topic size to 40 words, the BERTopic model created 37 different topics. As in the figure provided, we see that most of the bubbles overlap. Therefore there would be the possibility to decrease the number of topics consindering that this would leed to more imature topics. Analysing the content of the topics, we get on the left side topics that can be summarized with the keryword family. There we find family related terms like dad, mom, grandpa. The topics located on the bottom of the image mostly refer to general information about the adventures that Rick and Morty perform as there are terms like adventure, portal gun and treasure. Especially the word portal gun is a very important word as rick's portal gun is a key component of the series. Most of the other topics can be summarized by the speaking habits each character has. As there are topics where there are slang words like 'wubabdubadabab', 'geez' , 'oh' or 'crap'. Surpisingly, there is one topic located next to those slang words containing information about the planet system which contains words like pluto, planet and space.
In 5.1, we see a visualization of the most important topics and its words. It turns out, that the most important topic are slang words like 'oh' or 'man'. Topic 2 and topic 4 show that the series is about a human family. Topic 0, 11 and 16 contain more information about the content of the

series as there are a lot of terms referring to space and murdering activities that Rick and Morty are exploring along their adventures in space.

Topic Modeling based on Episode Descriptions

As the dialouges of the series are full of abbreviations and colloquial language without a lot of content, we wanted to have a closer look at the topics by looking at all episode descriptions. This time we analyzed the topics with Latent Dirichlet Allocation (LDA) as there are way less datapoints, each containing more characters in the transcription dataset. To setup our LDA model we create an Document Term Matrix (DTM) containing all words and documents Albanese (2022). Another dictionary maps the words with a given id. Furthermore, we limit the amount of topics to 50 so that there will be less topics than the number of episodes. In the series, the episodes do not depend on each other as Rick and Morty often discover new planets and characters each episode. The following topic analysis can lead to the same conclusion as a lot of the created topics just contain information about the content of one or a few episoded. In the chart 2.7 we see the Dirichlet priorities Albanese (2022) of the topic containing family and portal gun along all episodes.

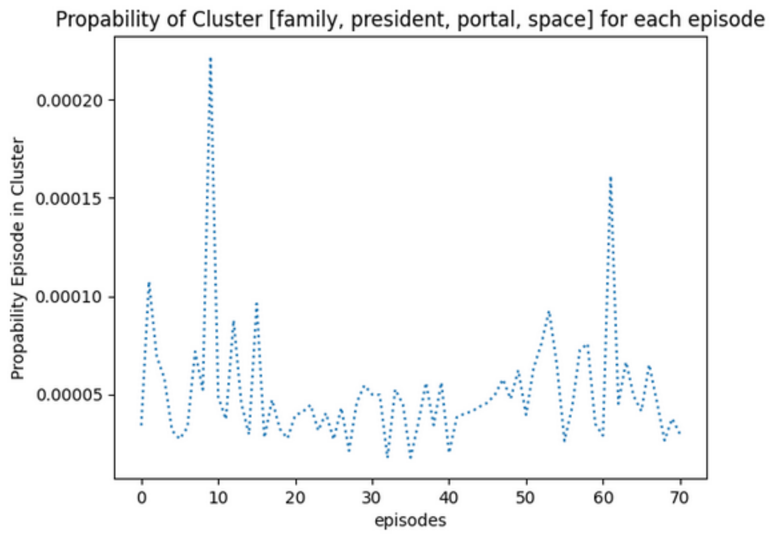


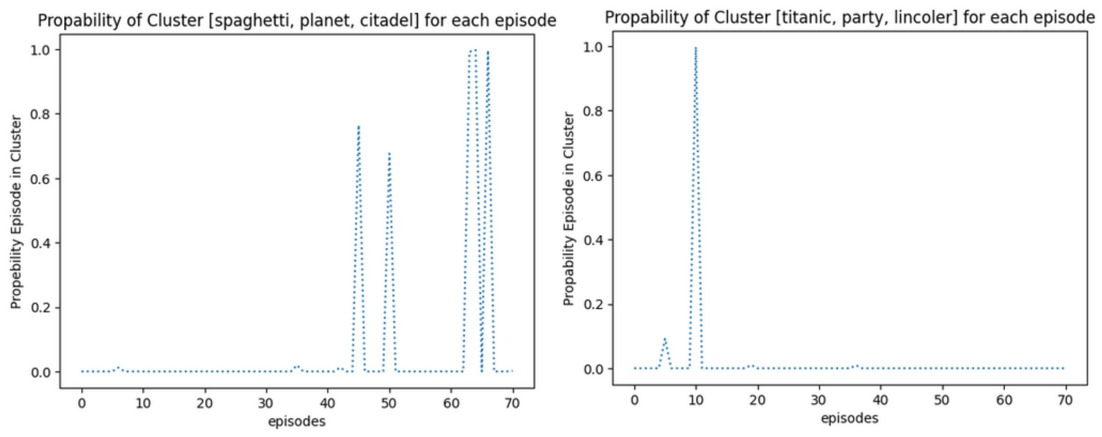
Abb. 2.7: Topic relevance Family Portal Gun over time

This topic plays a significant role in all episodes as 'family' and 'portal gun' are terms that are frequently used in a lot of episodes throughout the series.

As already mentioned, most of the other topics just refer to a few or one single episode. This is the case for the topic titanic party where jerry and beth attend a party that recreates the titanic drama. As the chart 2.8(b) shows, this topic holds information that is is mainly present in one episode.

The other spikes that are showcased in the chart can be explained as the side character 'Abradolf' 'Lincoler' plays a important role in other episodes as well.

Another example for a similar distribution is the topic about the spaghetti planet and its citadel, which is analysed in chart 2.8(a).



(a) Topic 'Spaghett Citadel' in all episodes **(b)** Topic 'titanic party lincoler' in all episodes

Abb. 2.8: episode related topics over time

The fact that most of the topics differ a lot from each other, can also be seen here in the bubble chart 5.3. It is remarkable that a vast amount of the topics are distributed across the chart while just a handful topics seem to overlap. Furthermore, the size of the bubbles indicate the amount of documents where the topic is part of. As a lot of the bubbles seem small this could mean that there the topics mostly relate to a few episode descriptions.

3 Modeling

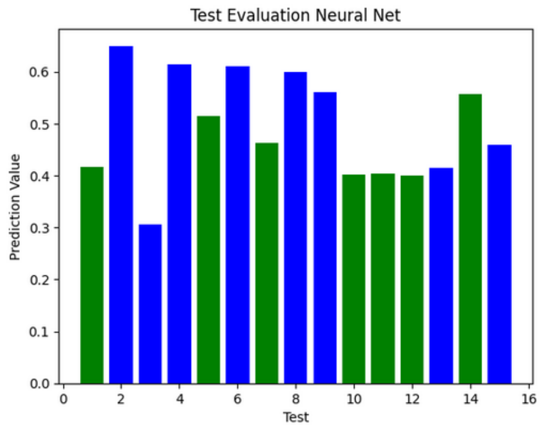
3.1 IMDB Rating prediction

In this section our goal was to perform a modeling to predict the IMDB ratings based on the provided episode descriptions. Therefore, we compare different strategies and architectures to solve this complex problem.
Our hypothesis is that there are some important words or characters like 'birdperson' that might have an impact on the IMDB rating as some characters or planets in the Rick and Morty universe are more popular than other.
Creating a perfectly fitting model is challenging, as the episode descriptions are written in neutral style and the features that determine whether an episode is liked or not can differ from person to person. Furthermore, there are also visual effects and music elements such as the 'get Schwifty' or snake jazz song that might lead to a higher IMDB rating. Therefore, we expect that our model will tend to not be the most accurate one.

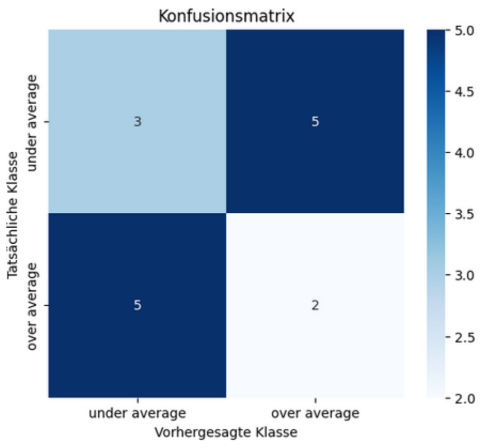
Vector Embedding Approach
As the neural network cannot calculate with strings, we have to represent the given text as numeric representations. To achieve this, our intial approach uses the Word2Vec library to generate a vector embedding for all the words that are that appear at least 5 times in all descriptions. We

used Word2Vec as we are able to get high value dense vector representations for low computational costs (vgl. Mikolov et al., 2013, S. 10). By that, we were able to represent most words in the Rick and Morty corpus as an vector of 100 values.
These 100 values also defined the input shape of our neural network which was designed for classifying the episode descriptions. To represent whole tokenized sentences into a vector containing 100 dimensions, we tokenized all of the training data and calculated the average vector of each episode description. As a result, each episode description can be represented as one single vector of 100 dimensions. With that, we are able to train a neural network with 100 input neurons. If a token was not part of the Word2vec vector embedding list, the token will be skipped. For example, the string 'DHBW Ravensburg' would produce a NaN and the string 'Rick DHBW Ravensburg' would produce the same result as the string 'Rick', since the unkown words like DHBW are ignored.

The test results are displayed in the following chart 3.1. The green bars represent the episodes having an IMDB rating above 8.2 while the blue ones are rated lower. As we did not set a thresshold yet, we can see the test result visualized in a bar chart.
As expected, the neural network struggles to distinguish between those two classes. In general, it seems that the models even predicts the complete opposite, as the blue colored bars were predicted higher as the the green ones. The calculated test accuracy based on the results without an thresshold was 33%.
Setting the thresshold to 0.5 we find following confusion matrix as an result 3.1. The confusion matrix with an treshhold set tp 0.5 also leads to a similar result as there are more wrong predicted values than right ones. Letting us know that this model definetly does not solve the problem.



(a) Neural Net Test Evaluation



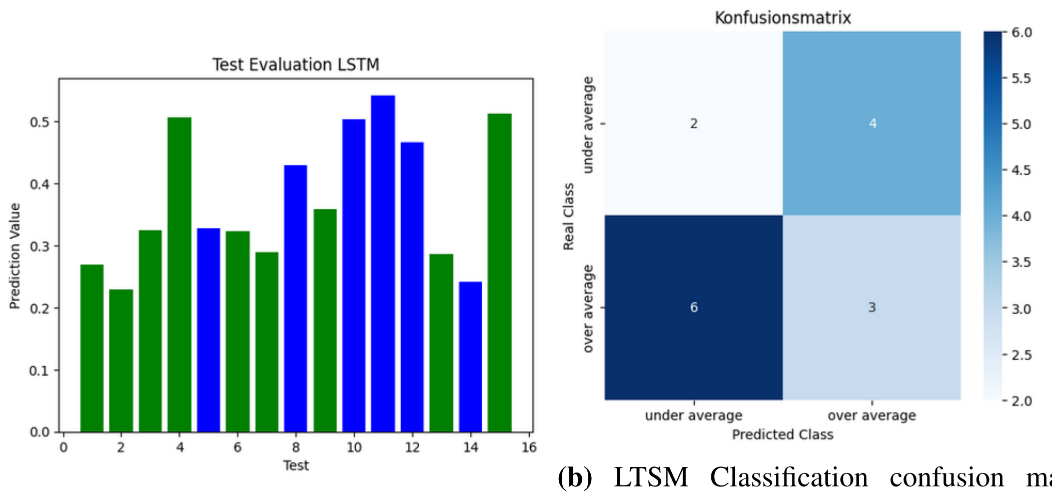
(b) Neural Net confusion matrix (thresshold = 0.5)

Abb. 3.1: Neural Net Test evaluation

LSTM Approach

As the first neural network had some difficulties to examine the complex relations between the

words, we also implement the Long Short Term Memory (LSTM) architecture to achieve more reliable results. The advantage that a LSTM has ist that it can also process the order of the words. We used the same Word2vec embedding model as before and added our Rick and Morty corpus to it because there are a lot of Rick and Morty unique terms like 'birdperson', 'portal gun' in the descriptions. Unlike the previous neural network, which computed the average vector for each sentence, we instead used the first 500 vectors in a sentence to numerically represent the entire text. The results of this approach are displayed in the chart 3.2.
As the LSTM Classification leads to more variety in text prediction, it still had some problems figuring out which texts are above and under 8.2 rating. As the confusion matrix displayed in 3.2 shows, the model is also not good in predicting the classes as it produces more False predicted values as right ones.



(a) LTSM Classification

(b) LTSM Classification confusion matrix (treshhold = 0.4)

Abb. 3.2: LSTM Classification results

Bag of Words Approach
As the word embedding approaches did not lead to a satifying outcome, we also tried to create a Bag of word matrix containing the the amount of the first 500 most appearing words within a text. To clean the data, we decided to remove every token in the description dataset which POS was not classified to PRPN or NOUN by spacy, in order to focus more on the character names and the nouns that are provided in the description. With that, we trained a Naive Bayes model to predict those classes. Therefore, the CountVectorizer by sklearn came into place. But unfortunally, also this approach could not return valid result as the confusion matrix 5.4 shows.
TF-IDF Matrix Approach
The TF-IDF Vectorizer by Sklearn prioritizes words that appear a lot in one document but not accross the documents. With that last classification approach, we hope to weigh terms or characters more that do not appear that frequently but are more popular than others. Following this procedure of creating an TF-IDF matrix, and training an Gaussian Model again, it also did

not lead to satisfying results 5.5.

LSTM Regression

We expect that features from an episode with a rating of 8.2 (class 0) does not differ so much from another episode with rating of 8.3 (class 1). As the correlation coefficent between the numeric IMDB rating value and the class from the classification is just 0.7, some information is definetly lost in the classification problem. Therefore, we aim to predict the IMDB rating not by classifying the episodes as 1 or 0 but by directly predicting it's IMDB score. With this regression approach, we hope to get better chances to identify the IMDB ratings as a lot of information is lost by transferring the numeric data into classes in the classification problems above.

To achieve this, we used the same architecture as in the classification task but changed the lossfunction as the LSTM classifier but we additionally normalised the Word2vec embedding vectors in the matrix.

In 5.3 we see the test accuracies by epoch size. As the epoche size of 15 leads to the lowest sqaured error, we use 45 as a default epoch size.

Another Cross Validation based on models of epoch size 45 leads to an average squared error of 0.47 5.1, meaning that, on average, our model's predictions deviate by approximately 0.68 rating points from the true IMDB scores. As a comparison, the statistical standard deviation of all ratings is 0.97. An analysis of the test results is displayed in the chart 3.3. There we, see the predicted IMDB ratings in blue and the true ones in black. In general, we see that the model predicts valid results, as none of the predicted IMDB ratings were above 10 and ofthen the deviation between true and predicted values are low.

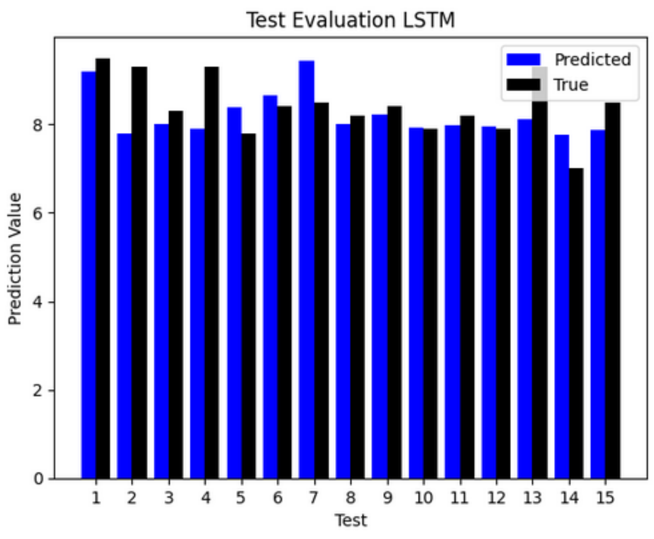


Abb. 3.3: Test Results Regression

When testing the model, we noticed that in general, the combination and the appearance of characters influences the IMDB score. For example the string "rick is on an adventure"leads to an 7.2 while "rick and morty are on an adventure"leads to 10.1. Handing over "rick and beth are on an adventure"leads to an output of 9.3. "rick and birdperson are on an adventure"leads for an example to a score of 6.6. The sentence Morty and Rick, for example also just produce a IMDB rating of 8.3 while calculates for the sentence "Rick and Mortyän IMDB rating of 9.5. The problem that neural networks have is that they are not explainable and is more like a black box. In the last approach we want to figure out more about the terms and its influence in other model architectures.

Regression with TF IDF

The linear regression model with TF IDF matrix as features should examine which character or nouns defnine the IMDB Prediction. To achieve this, we trained a linar OLS regression model by statsmodelapi and printed out the characters that determine the IMDB prediction the most based on the model. According to this model, the feature term president has the highest coefficient whith terms like president, saber and universe or mortytown 5.4. The terms with the lowest coefficient are destruction, garage and gun 5.5. The word president appears in the episode 16 and 18 and 27 that all are under the top 5 rated rick and morty episodes. A more impactful example is the term mortytown that just appearead in episode 27, the highest rated episode as visible in 3.4.

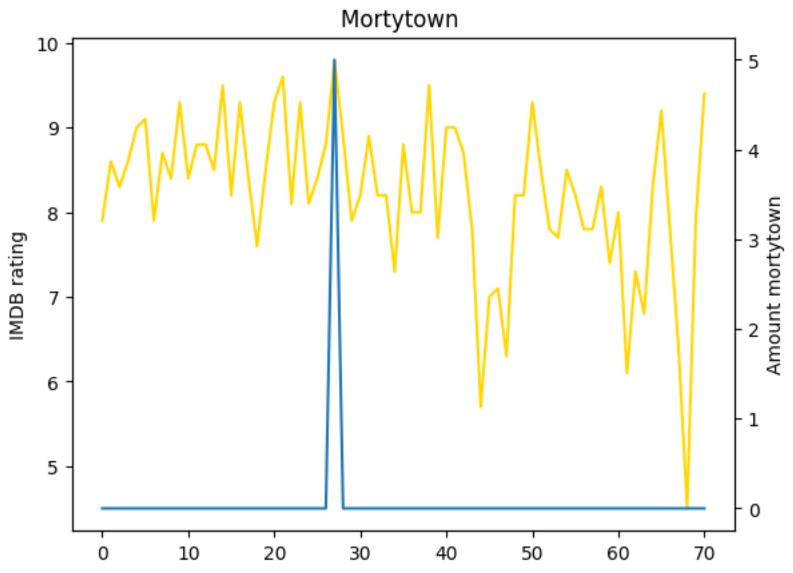


Abb. 3.4: Number of Mortytown apperances with IMDB prediction

With all these different approaches we were not able to create a model which classifies the IMDB prediction 100% correctly. Not just the vector embedding approach leads to a unsatisfying result in the classification architectures, but also the Term matrix approach with BOW and TF IDF could not distinguish between high and low IMDB prediction. A reason for that is the lack of a huge datasource as there are just 70 Rick and morty episodes. Furthermore, most of the characters and planets appear just in a few episodes. While terms in in the trainingset like Mortytown correclty determine IMDB ratings in the TF IDF approach, testing the model on new vocabulary will definetly not produce valid results.

With the regression model we got rid of information losses that were caused due to changing the label format from numeric to qualitative. With that, we examined that characters and its appearance and combination might have a lot on influence on the IMDB prediction and we were able to figure out terms like mortytown or president, which influence the rating the most according to a linear regression model.

3.2 Speaker detection

The last modeling part examine if the character‘‘s speaking traits can be identified by a fine tuned transformer model. Therefore, we just use the family sanchez as classes because the other characters talk way less in the episodes as already mentioned in 2.1. As already seen in chart 2.1, the dataset is imbalaced which can cause some problems. In order to get rid of this imbalaced form, we decided to duplicate the amount of spoken lines of jerry, beth and summer for the modeling part.
For this classification problem, we used the DisitllBERT transformer Huggingface as a base model consisting of the DistillBERT Tokenizer and a Sequence classifier. As the test error still stops to decrease at epoch 4 as visible in 5.6, we fine tuned the model with 4 epochs. Doing this, we created a model that lead to 61% accuracy on the test set. There we have to keep in mind that there might be duplicates in the test set due to the duplication step that was explained in the text above.
To demonstrate the model and to distinguish character speaking habits the model easily identifies common spoken words to an characters. As an example the Promp "That’s weird"leads to jerry, while "That’s weird burp"produces rick as an output.
Furthermore also some more complicated sentences like Ï love jessica"produces morty as a result which also most likely a quote by him.
The overall results can be seen in following confusion matrix 3.5 and interpreted with the metrics in table 5.2.

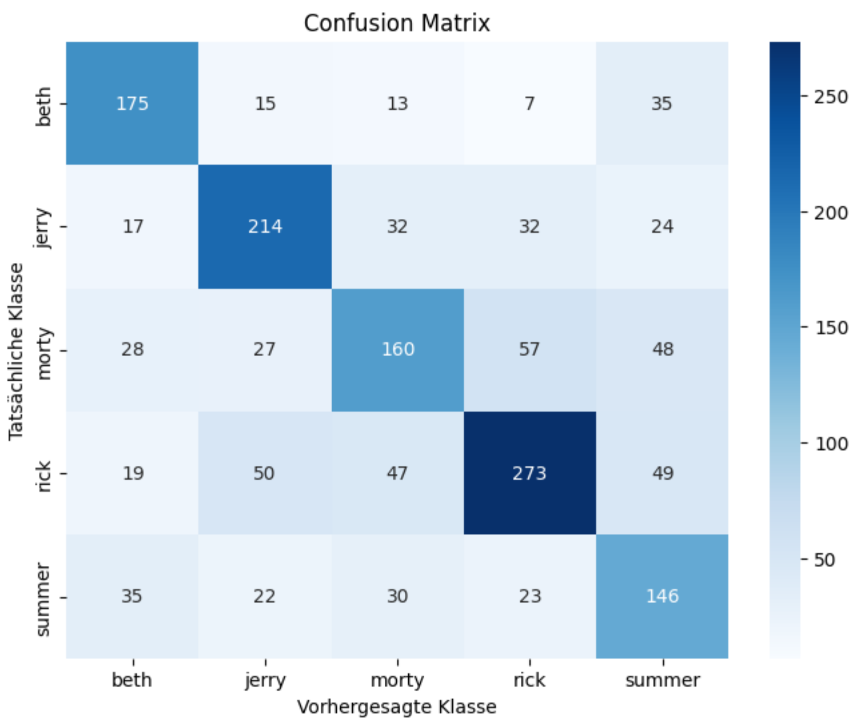


Abb. 3.5: Confusion Matrix speaker detection

4 Diskussion

Summarizing thethe topic modeling, we were able to extract relevant content that plays a sigi-nificant role in the series Rick and Morty. Topics like family, or space describing topics were adressed through the whole series. Looking at the topics into more detail, it becomes clear that most of the topics just relate to one episode leading to the conclusion that every episode is hard-ly related to oher episodes.

Regarding the IMDB prediction, our hypothesis that it is very difficult to predict IMDB ratings based on description texts, turned out to be true. Every classification approach had some diffi-culties to predict wether the episode will have a high or low IMDB prediction. Furthermore, the regression approach led to valid results and had a relatively low standart deviation for the test set.

Based on some experiments, the model learned which character combination has a potential higher IMDB rating than other. This might lead to the hypothesis, that frequent combinations like rick and morty or rick and jerry produce a higher IMDB rating than ones with lower term frquency like rick and birdperson. The model also learned, that more characters might lead to a higher IMDB ratings as well, as the sentence same sentence without any other character produ-ced a lower IMDB rating.

With finetunging the DistillBERT transfomer, we were able to classify the speakers with a test accuracy of 60% what could lead to the conclusion that the speaking behaviors of each character differ from each other.

5 Anhang

mean average error
0.4996431767940521,
0.4830387234687805,
0.5242437124252319,
0.4312141239643097,
0.49884870648384094

Tab. 5.1: 5 Fold Cross Validation

	precision	recall	f1-score	support
beth	0.64	0.71	0.67	245
jerry	0.65	0.67	0.66	319
morty	0.57	0.50	0.53	320
rick	0.70	0.62	0.66	438
summer	0.48	0.57	0.52	256
accuracy			0.61	1578
macro avg	0.61	0.62	0.61	1578
weighted avg	0.62	0.61	0.61	1578

Tab. 5.2: Metrics for speaker detection

epoch	Mean sqaured Error
15	0.45
25	0.479
35	0.51
45	0.43
55	0.50

Tab. 5.3: Test scores accuracy LSTM regression

feature	**coeff**
president	8.821506
saber	5.209245
one	4.810728
churry	4.542387
time	4.510295
suit	4.186404
things	4.167846
curve	4.155876
rift	4.114602
lord	3.994916
fluid	3.846431
mortytown	3.577070
point	3.569252
universe	3.547126
hands	3.388094
family	3.386688
jesus	3.336255
rhett	3.309392
mr	3.251881
chaos	3.172340

Tab. 5.4: Top Coefficients Linear Regresssion (highes score)

feature	**coeff**
destruction	-2.168899
head	-2.116017
success	-1.995059
cop	-1.455023
version	-1.347783
him	-1.310323
fact	-1.238303
locos	-1.191900
jellybean	-1.096124
scene	-1.040677
ricks	-0.976909
robot	-0.916737
court	-0.887538
rock	-0.692056
hand	-0.635960
home	-0.604017
garage	-0.596317
diane	-0.525239
people	-0.487156
gun	-0.471000

Tab. 5.5: Top Coefficients Linear Regresssion (lowest score)

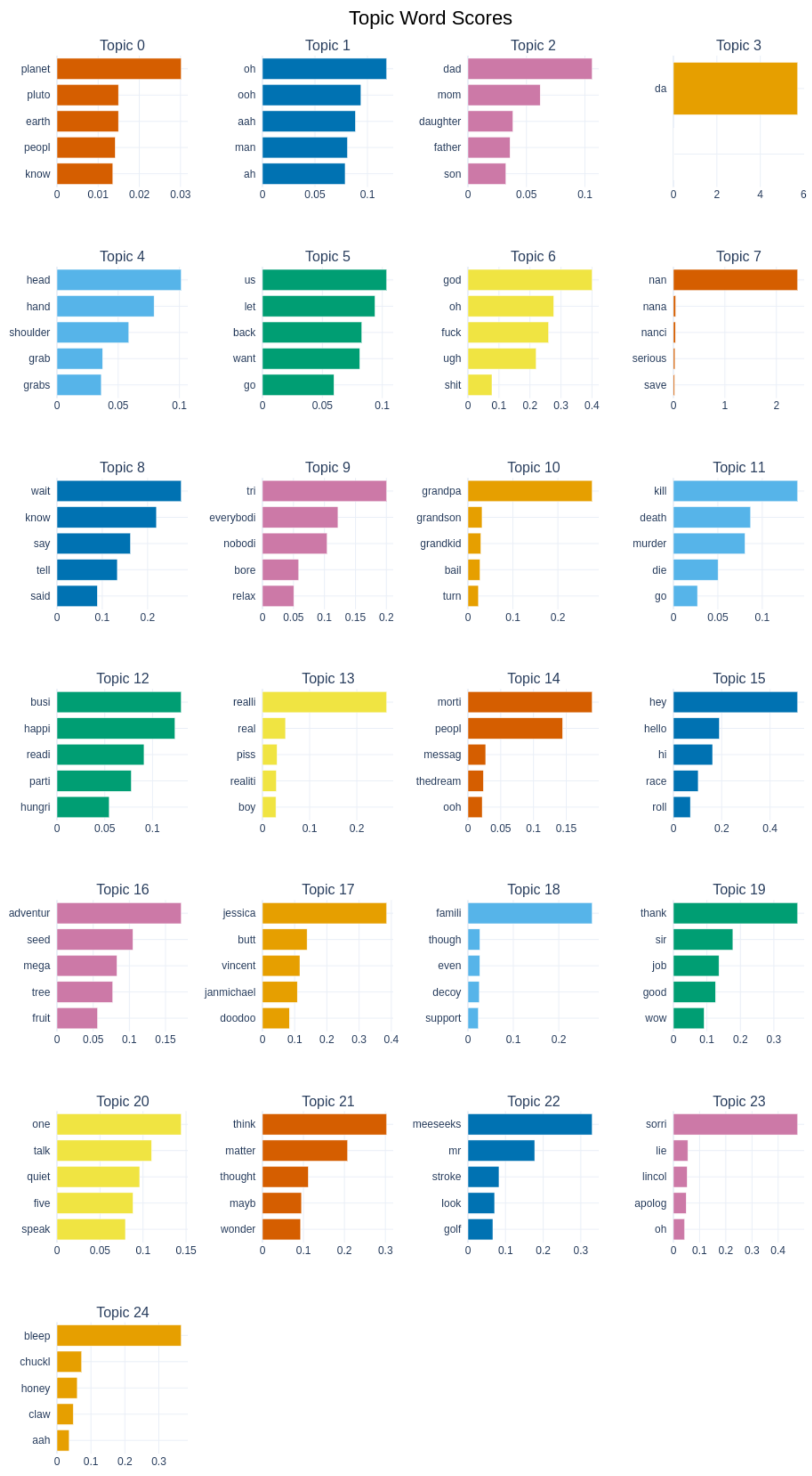


Abb. 5.1: Top 25 BERT topics for transcript (all seasons)

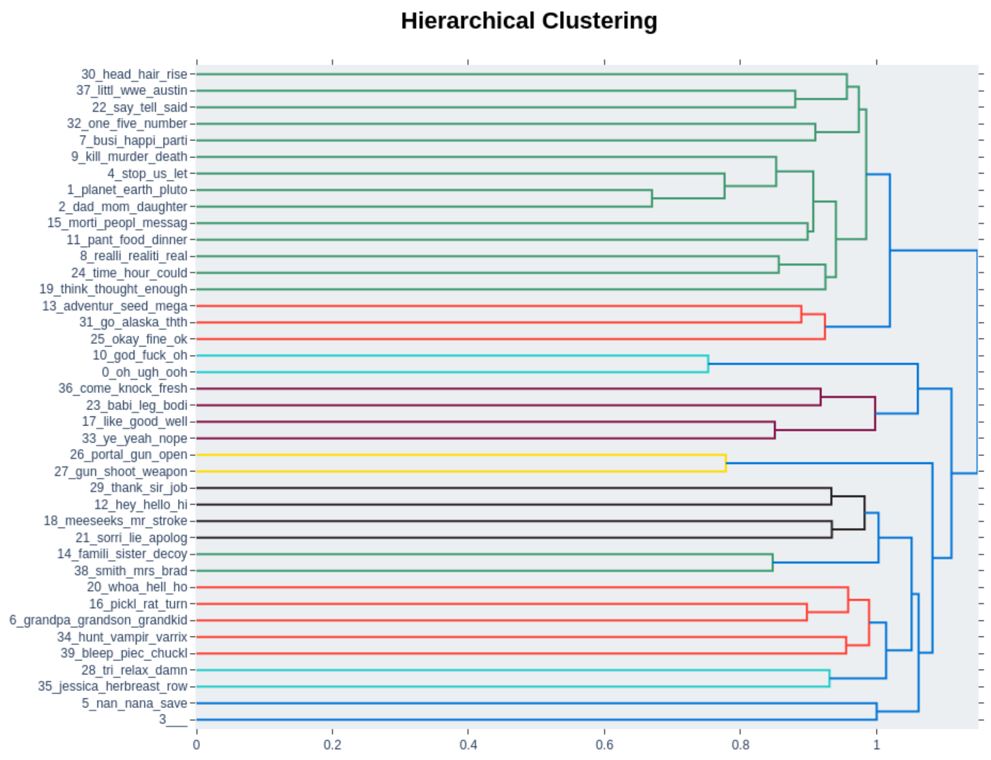


Abb. 5.2: Hierarchial Clustering topics on transcript

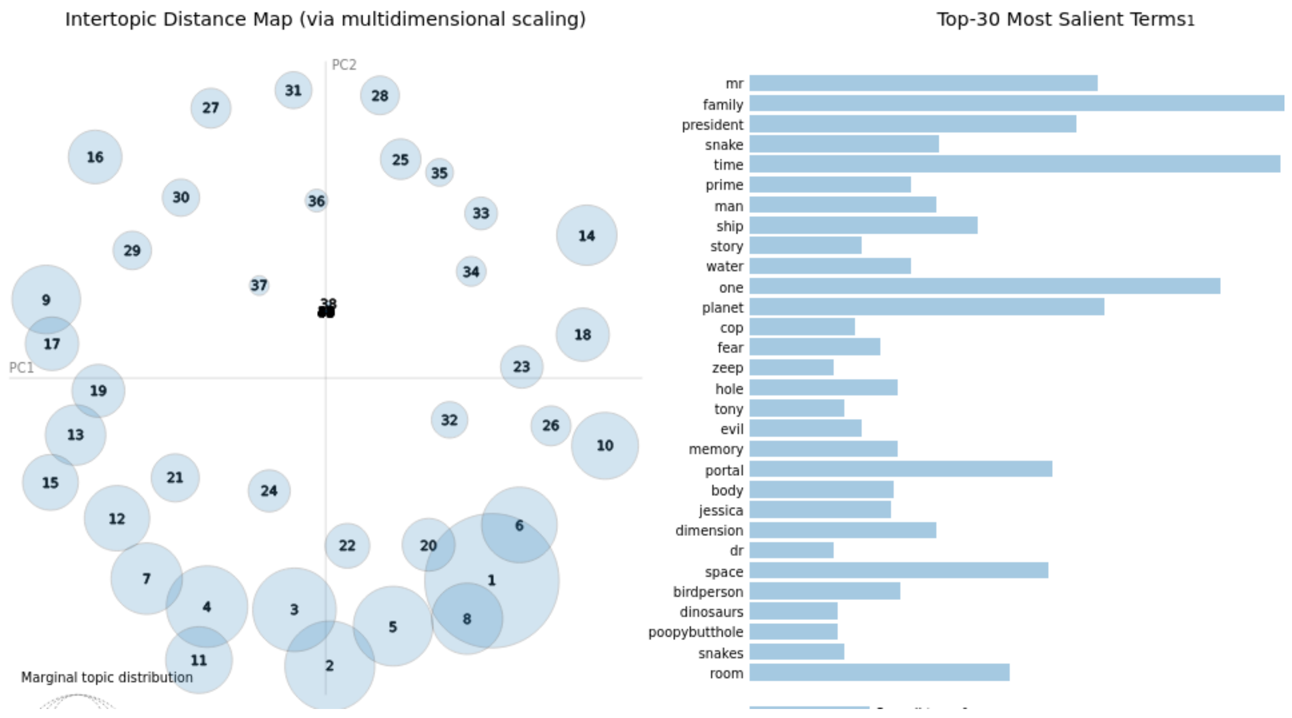


Abb. 5.3: All Topics LDA for description (all seasons)

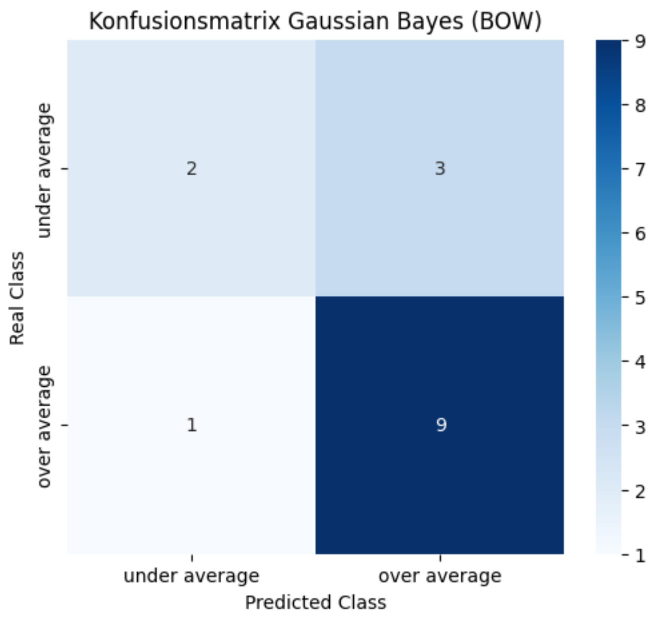


Abb. 5.4: Confusion Matrix Bayes Classification with BOW

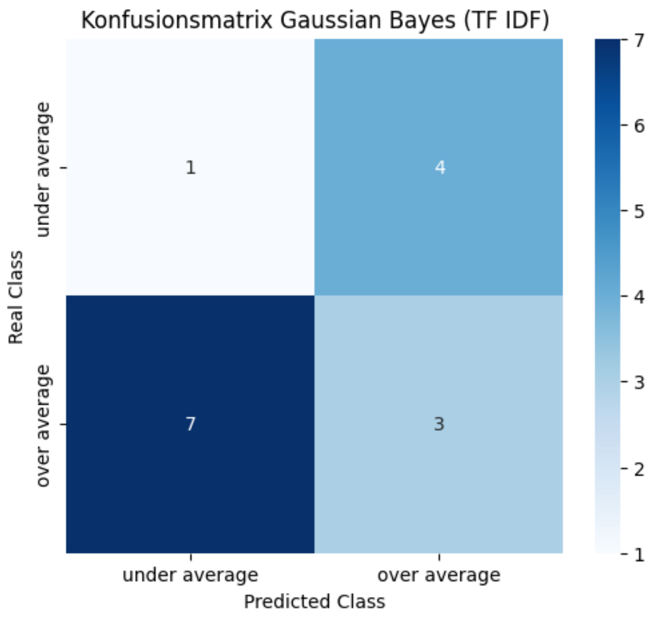


Abb. 5.5: Confusion Matrix Bayes Classification with TF IDF

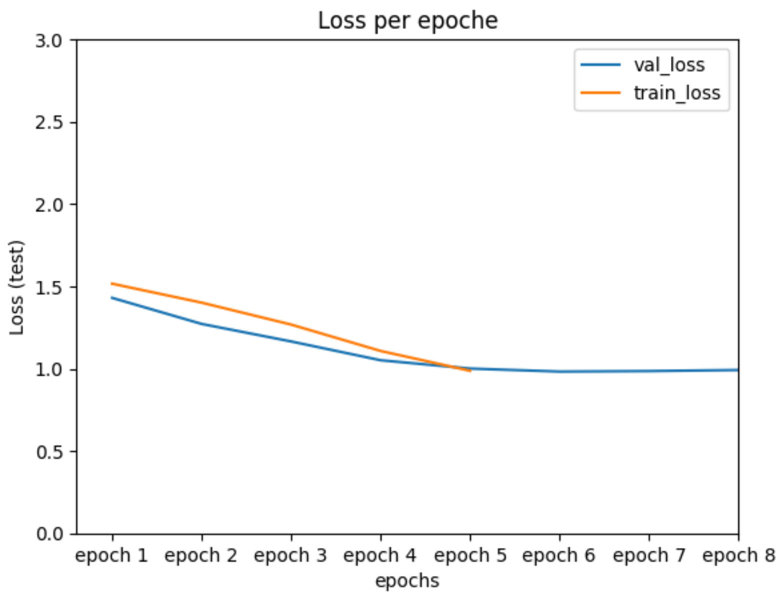


Abb. 5.6: Cross Validation Speaker Detection DistillBERT

Literatur

[Albanese 2022] ALBANESE, Nicolo: Topic Modeling with LSA, pLSA, LDA, NMF, BERTopic, Top2Vec: a Comparison. In: *Towards Datascience* (2022), September. – URL https://medium.com/towards-data-science/topic-modeling-with-lsa-plsa-lda-nmf-bertopic-top2vec-a-comparison-5e6ce4b1e4a5

[Grootendorst 2022] GROOTENDORST, Maarten: *BERTopic: Neural topic modeling with a class-based TF-IDF procedure*. März 2022. – URL http://arxiv.org/abs/2203.05794. – Zugriffsdatum: 2025-03-06. – arXiv:2203.05794 [cs]

[Huggingface] HUGGINGFACE: *DistillBERT Documentation on Huggingface*. – URL https://huggingface.co/docs/transformers/model_doc/distilbert

[IMDB 2025] IMDB, company: *Rick and Morty Episodenguide IMDB*. Januar 2025. – URL https://www.imdb.com/de/title/tt2861424/?ref_=fn_all_ttl_1

[Mikolov et al. 2013] MIKOLOV, Tomas ; CHEN, Kai ; CORRADO, Greg ; DEAN, Jeffrey: *Efficient Estimation of Word Representations in Vector Space*. September 2013. – URL http://arxiv.org/abs/1301.3781. – Zugriffsdatum: 2025-03-06. – arXiv:1301.3781 [cs]

[Mueller] MUELLER, Andreas: *Word Cloud Documentation Python*. – URL http://amueller.github.io/word_cloud/generated/wordcloud.WordCloud.html

[Prarabdha] PRARABDHA, Srivastava: *Rick&Morty Transcripts*. – URL https://www.kaggle.com/datasets/prarabdhasrivastava/rickmorty-transcripts. – Zugriffsdatum: 2025-01-31

[Rick and Morty wiki 2025] RICK AND MORTY WIKI, fandom: *Rick and Morty episode list*. Januar 2025. – URL https://rickandmorty.fandom.com/wiki/List_of_episodes

[Sushmita 2014] SUSHMITA, Katam: *The Porter Stemmer*. Dezember 2014. – URL http://cs.indstate.edu/~skatam/paper.pdf. – Zugriffsdatum: 2025-03-03

Selbständigkeitserklärung Anton Geiger

Ich versichere hiermit, dass ich, Anton Geiger, meine Seminararbeit

Burp NLP: A Rick and Morty Text analysis

selbständig verfasst, die digitale Version mit der ausgedruckten
Version übereinstimmt und keine anderen als die angegebenen
Quellen und Hilfsmittel benutzt habe.

Ort, Datum	Unterschrift