Group members: Vladislav Stoimenov (235030), Petar Paskalev (232725), Mario Velichkov (230623), Raya-Neda Borisova (220471)

Mentor: Jison

	IS		

	Model details		Data details			Parameters	Metrics round to 3 decimals!			nals!	Results & Observations		Student(s) responsible	Github link			
					Train/Val/Test	Features used	Data augmentation					Other metrics					
	r. Model type	Framework/library used	Dataset	Preprocessing steps	split	(if applicable)	(if applicable)	Hyperparameters	Accuracy	Precision	Recall	F1 score (if applicable)	Strenghts	Weaknesses	Other notes		
		Tenserflow, sk learn, pandas,			Val = 0.15, Test=0.20			Min learning rate = 0,0001					The model predicts happiness 73	The dataset is super inbalanced there is way more happiness and neutral	•		
ibel E			Go emotions 1,2,3	Combine all of the datasets		Pos tags Sentiment score Tf-ldf	None	factor= 0,5	0.546	0.453	0.486	0.458 None	percent of the also neutral is about 53	than the other emotions	None	232725 Petar	LSTM models
						-											
	2 Naive Bayes	attern contro	e	Combine GoEmotion, MELD, CARER	test = 0.2	POS Tags, TF-IDF (unigrams +bigrams)	None	default alfa = 1.0	0.498	0.491	0.498	0.408 None	Simple and fast, works with text data	Struggles with complex relationships in text	None	000474	Naive Bayes Iteration 2
	2 Naive bayes	skleam, pandas	rinal_merged_ML.csv	Combine Goemotion, MELD, CARER	test = 0.2	POS Tags, TF-IDF (unigrams +bigrams)	None	derautt atra = 1.0	0.498	0.491	0.498	U.4US None	Simple and rast, works with text data	in text	None	2204/1	Naive baves iteration 2
	3 Naive Bayes	skleam, pandas	final_merged_ML.csv	Combine GoEmotion, MELD, CARER	test = 0.2	POS Tags, TF-IDF, CountVec	None	default alfa = 1.0	0.481	0.444	0.483	0.435 None	Richer feature representation Best overall performance among LR	Higher computational cost + slower	None	220471	Naive Bayes Iteration 1
													models, diverse feature set captured	High-dimensional features risk			
	4 Logistic Regression	scikit-leam, pandas	final_merged_ML.csv	Combine GoEmotion, MELD, CARER	test = 0.2	TF-IDF + CountVectorizer	None	max_iter=500	0.535	0.522	0.535	0.521 None	patterns well		None	220471	Logistic Regression 1
								max_iter=1000, C=1.0, solver='lbfgs'.					Handled class imbalance well.	Slightly lower performance than Model 1. limited semantic			
	5 Logistic Regression	scikit-leam, pandas	final_merged_ML.csv	Combine GoEmotion, MELD, CARER	test = 0.2	TF-IDF (unigrams, bigrams, trigrams)	None	class_weight='balanced'	0.509	0.531	0.509	0.518 None			None		Logistic Regression 2
				text cleaning (lowercase, remove special													
	6 Naive Baves			characters), TF-IDF Vectorizer, CountVectorizer, MinMaxScaler	test = 0.2	POS Tags, TF-IDF, CountVec, Sentiment Score	None	alpha = 1.0	0.498	0.491	0.498	0.408 None	precision and recal are close == predictions are somewhat balanced	struglles with imbalanced data, since the recall isn't higher	None	220471	Naive Baves Iteration 3
			final_combined_datase					weight_decay=0.01,					Predicts one of the more	Does not perfrom well on most of the			
	7 Bert Large			Clean and combine datasets GoEmotion, MELD, CARER	Train/Val = 0.8/0.2	None	None	warmup_ratio=0.1, fp16=True, token max_length = 64	0.421	0.483	0.322	0.261 weighted f1 = 0.42	underrepresented classes quite well -	other classes with an macro f1 score of 0.20-0.40	None	000000	Iteration 2 - Bert Large
	/ bert Large	Transformers	nly.csv	MEED, CARER	Irain/vat = 0.8/0.2	None	None	token max_tengtn = 64 learning_rate = 0.001,	0.421	0.483	0.322	0.261 Weighted_FI = 0.42	digust	010.20-0.40	None	230623	iteration 2 - Bert Large
			proccessed_dataset_v2			Sentiment_scores, pos_taggins, glove		batch_size=64, token_max_len					The model prevented overfitting, but still				
	8 RNN	numpy	.csv	Class Weights	Train/Val = 0.8/0.2	embeddings	None	=50	0.728	0.451	0.695	0.547 None	got lower performance	on disgust and neutral  The model is not generalizing well on	Iteration 1	Vladislav (235030)	Iteration 1 - RNN
		Tenserflow, sk learn, pandas.	proccessed dataset v2			Sentiment_scores, pos_taggins, glove		learning_rate = 0.001, batch size=64, token max len					The model has better pre-processed por				
	9 RNN	numpy	.csv	Changed Pre Processing of Tokens	Train/Val = 0.8/0.2	embeddings	None	=50	0.829	0.572	0.776	0.658 None	tags and improved performance.	emotions.	Iteration 2	Vladislav (235030)	Iteration 2 - RNN
		Tenserflow, sk learn, pandas,	nmcraeead dataset v2	Added 15 rows of data for underrepresented		Sentiment_scores, pos_taggins, glove		learning_rate = 0.001, batch size=64, token max len					The model predicts most of the classes	The model still make wrong			
		numpy	.csv	classes that are close to the test set	Train/Val = 0.8/0.2	embeddings	None	=50	0.827	0.751	0.695	0.722 None	very well.	predictions on disgust and neutral.	Iteration 3	Vladislav (235030)	Iteration 3 - RNN
													The model is generalizing well on all of				
				Downsampling the first two classes that have a lot more data points - (Happiness				learning rate = 0.001.					the classes. The lower padding length and the 'mask zero=True' in the	It may be limited to angrer and neutral sentences. May be limited to			
				and Sadness) and set mask_zero=True in		Sentiment_scores, pos_taggins, glove		batch_size=64, token_max_len					embeedings layer improved the model				
	11 RNN	numpy	.csv	the model architecture	Train/Val = 0.8/0.2	embeddings	None	=32	0.858	0.641	0.809	0.715 None	performance.	is set to 32	Iteration 4	Vladislav (235030)	Iteration 4 - RNN
				Downsampling the first two classes that have a lot more data points - (Happiness and Sadness) and set mask_zero=False in		Sentiment_scores, pos_taggins, glove		learning_rate = 0.001, batch_size=64, token_max_len					The model is generalizing well on all of				
	12 RNN	numpy	.csv	the model architecture	Train/Val = 0.8/0.2	embeddings	None	=50	0.829	0.691	0.824	0.751 None	the classes.	lower.	Iteration 5 - The best RNN Iteration	Vladislav (235030)	Iteration 5 - RNN
				Nothing changed from the previous RNN		Sentiment_scores, pos_taggins, glove							Ver good results on validation and	The model was overfitting in my			
	13 LSTM	numpy	.csv final combined datase	iteration	Train/Val = 0.8/0.2	embeddings	None	factor = 0.2 weight_decay=0.01	0.371	0.365 0.	.376(weighted) (	0.335(weighted) None	neutrall is with 46 f1 score  None really this model turned out quite	opinion	Iteration 2	232725 Petar	LSTM models
			t_sentence&emotion_o	Clean and combine datasets GoEmotion,				warmup_ratio=0.1, fp16=True,					bad. The only thing it predicts with more				
	14 Bert Large		nly.csv	MELD, CARER,	Train/Val = 0.8/0.2	None	None	token max_length = 128,	0.421	0.482	0.311	0.262 weighted_f1 = 0.34	than 0.5 is neutral	Overall poor performance	None	230623	Iteration 3 - Bert Large
			final_combined_datase t sentence&emotion o	Clean and combine datasets GoEmotion.				weight_decay=0.01, warmup_ratio=0.1, fp16=True,						All of the other emotions are below			
	15 Bert Large		nly.csv	MELD, CARER,	Train/Val = 0.8/0.2	None	None	token max_length = 128	0.581	0.443	0.412	0.415 weighted_f1 = 0.57	Predicts "happiness" quite well		None	230623	Iteration 4 - Bert Large
			final_combined_datase	Clean and combine datasets GoEmotion,				weight_decay=0.01, warmup_ratio=0.1, fp16=True,					This iteration is worse than the previous	Overall poor performance company			
	16 Bert Large	Transformers	nly.csv	MELD, CARER,	Train/Val = 0.8/0.2	None	None	token max_length = 56	0.422	0.514	0.373	0.271 weighted_f1 = 0.34	one. So no strenghts really		None	230623	Iteration 5 - Bert Large
			final_combined_datase					weight_decay=0.01,					The model classifies "disgust",				
	17 Bert Large		t_sentence&emotion_o nly.csv	Clean and combine datasets GoEmotion, MELD, CARER,	Train/Val = 0.8/0.2	None	None	warmup_ratio=0.1, fp16=True, token max_length = 56,	0.501	0.461	0.482	0.443 weighted_f1 = 0.48	"happiness", "neutral" with more than 0.5	"fear" and "surprise" are below 0.3 f1	None	230623	Iteration 6 - Bert Large
			final_combined_datase					weight_decay=0.01,					This model performs quite well for	The model does not recognize the			
	18 Bert Large		t_sentence&emotion_o nlv.csv	Clean and combine datasets GoEmotion, MELD, CARER.	Train/Val = 0.8/0.2	None	None	warmup_ratio=0.1, fp16=True, token max_length = 56, Custom	0.603	0.521	0.511	0.514 weighted_f1 = 0.61	emotions like "anger", "happiness", "sadness"	emotions "fear" and "surprise" well + there is no neutral in this iteration	None	220022	Iteration 7 - Bert Large
			,	many and Miles by					0.003	0.021	0.011	0.014 Wagnes_(1=0.01				230023	The second secon
								learning_rate=1e-5, weight_decay=0.01.					The model performance increased after	After we real shelled the data the			
			final_combined_datase					warmup_ratio=0.1, fp16=True,					test set manual annotation. Now it is	model performance jumped quite a			
	19 Bert Large		t_sentence&emotion_o nlv.csv	Clean and combine datasets GoEmotion, MELD, CARER.	Train/Val = 0.8/0.2	None	None	token max_length = 56, Custom Weights	0.741	0.733	0.645	0.651 weighted f1 = 0.74	more accurate with it's predictions for	lot. The only emotions it's struggling with right now is "fear", below 0.5	Name	000000	Iteration 8 - Bert Large
	io pert Large	naisiomers	my.usv	PIELD, GAREK,	11as1/V8L = U.8/U.2	INUIR	Note	vvcigntS	0.741	0.733	U.645	U.651   Weignted_f1 = 0.74		Underperforms despite richer	NUIR	230623	nerauon 8 - bert Large
				tabal and desired and an extension									Used GloVe embeddings and sentiment				
	20 Naive Bayes		test_set_with_features. csv	label encoding, feature extraction (sentiment + GloVe)	full test set	sentiment score, GloVe embeddings	none	default alfa = 1.0	0.402	0.436	0.402	0.304 none	information for deeper text understanding	continuous data and complex feature interactions	None	220471	Naive Baves Iteration 4
														Model underutilizes dense			_
	21 Logistic Regression		test_set_with_features. csv	label encoding, feature extraction (sentiment + GloVe)	full test set	sentiment score, GloVe embeddings	none	solver='liblinear', C=1.0, penalty='l2', random_state=0	0.402	0.409	0.402	0.418 weighted f1 = 0.222	Semantic-rich features and sentiment scoring improved accuracy over NB	embeddings, performance below text- based models	t- good improvements	220471	Logistic Regression 4
				have a lot more data points - (Happiness				learning_rate = 0.001,	0.402	0.400	0.402	0.410 Wagning (1=0.332		It is predicting mainly neutral on the	Aprovententa	220471	The state of the s
		Tenserflow, sk learn, pandas,		and Sadness) and set mask_zero=False in	Train/Val = 0.8/0.2	Sentiment_scores, pos_taggins, glove	None	batch_size=64, token_max_len =50	0.341	0.342	0.344	0.293 None	it had 81 f1 in the training set but then it		h		LSTM models
	ZZ LSIM	numpy	.csv	the model architecture	rrain/Val = 0.8/0.2	embeddings	None	=50 max_iter=1000, C=1.0,	0.341	0.342	0.344	0.293 None	went down drastically	everything Lowest overall performance,	Iteration 3	232725	Laim models
								solver='lbfgs',					Balanced class weights, richer n-gram	sensitive to noisy features and sparse			
	23 Logistic Regression	scikit-learn, pandas	Test_set_refined_xlsx	emotion mapping, text cleaning, TF-IDF	Train/Val = 0.8/0.2	TF-IDF vectorized (ngrams 1-3)	None	class_weight='balanced'	0.248	0.302	0.248	0.257 none	representation	data	None	220471	Logistic Regression 3
	24																

27		
28		
29		
30		
31		
32		
33		
34		
35		
36		
37		
38		
39		
40		

you can add more rows to the table if needed