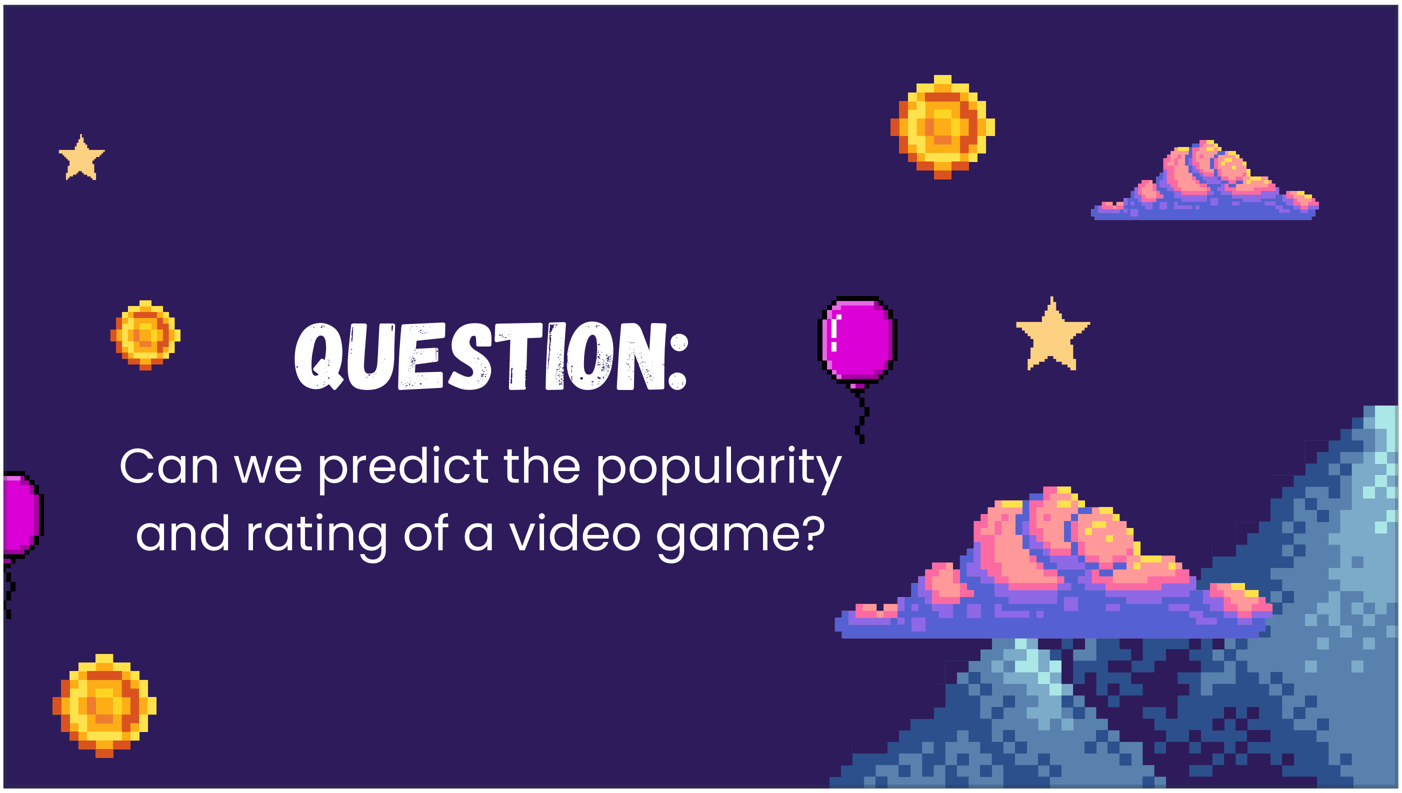
Intro + Motivation + Approach (Outline) + Data Cleaning: Josie (2 mins)
Baseline: Amanda
FFNN: Nabiha
Sentiment Analysis (Nabiha) (Maybe)
Embedding: Varun
Logistic Regression: Amanda
Conclusion: Josie
10 mins max + 5 mins Qs



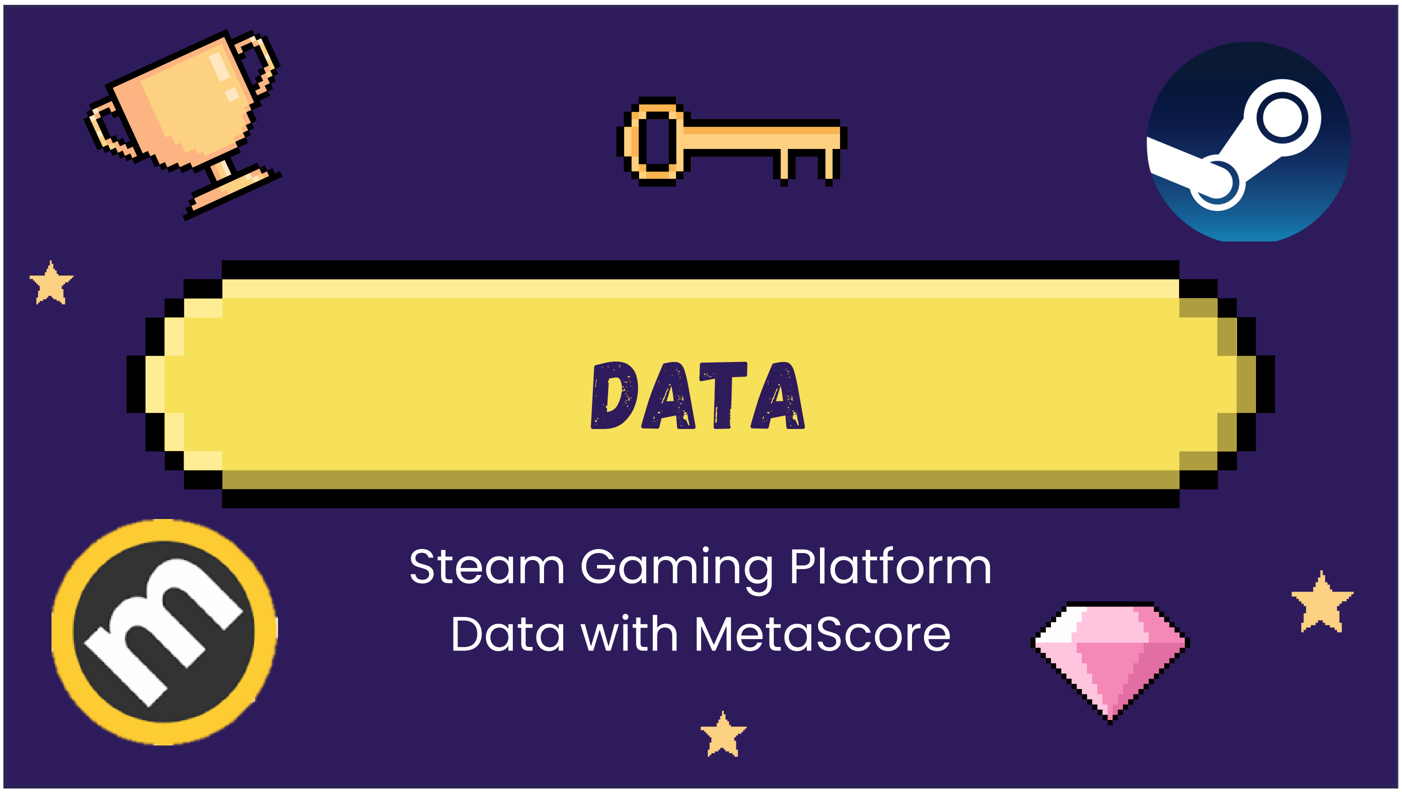
The video game industry has grown increasingly popular over the years, with the global video game market valuing at ~135 Billion USD globally by 2018. The rise in popularity increased following the pandemic, with some analysts predicting the global market will generate more than $260 billion in revenue by the year 2025.

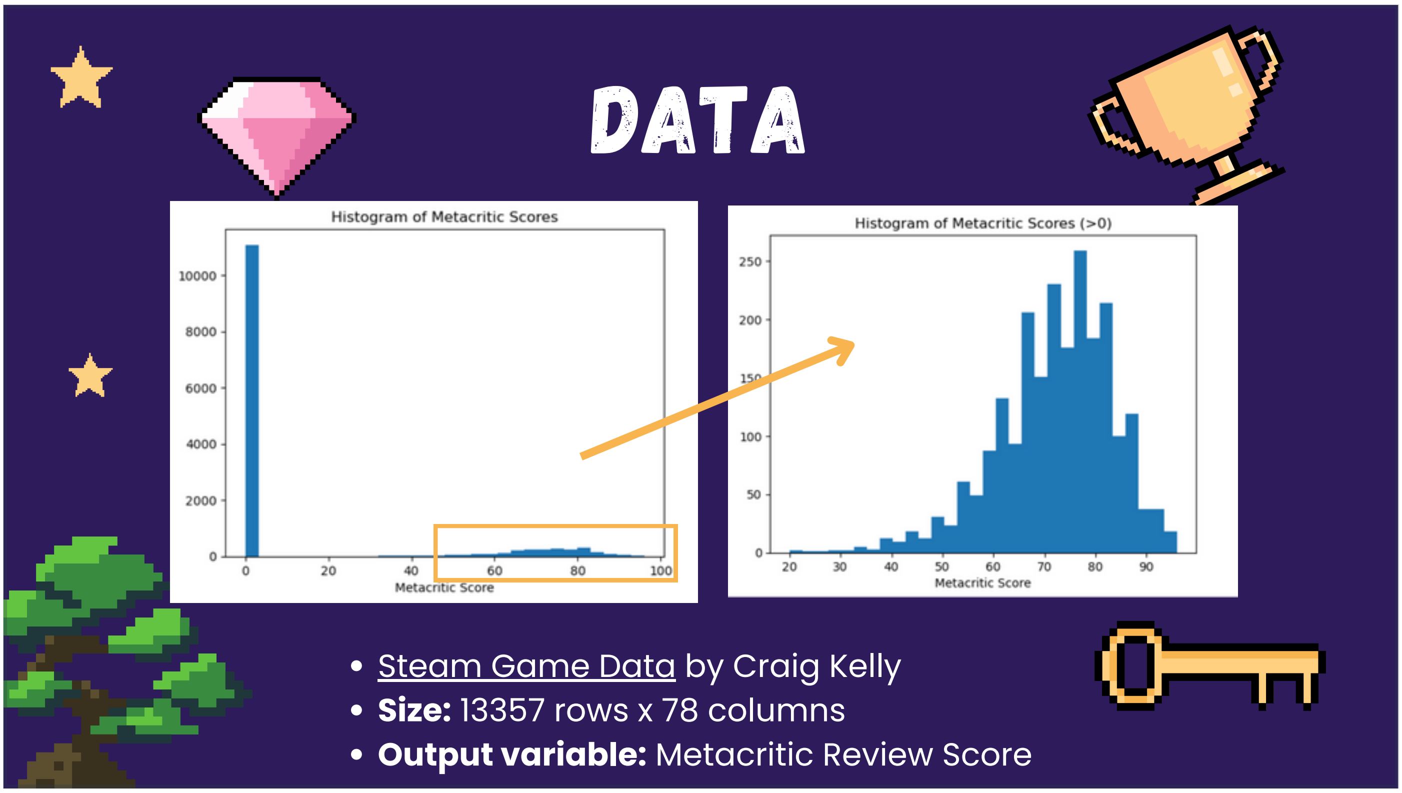
Individual video game purchase amounts vary by game, with large/highly-anticipated titles regularly costing $60. Given the costs associated with being an avid gamer, it is imperative that buyers have confidence in the game's quality before making a purchase. In the gaming world, reviews hold a lot of weight in terms of buyer decision making and popularity/success of a game.

In a gaming platform like Steam, games in the store have what's called a 'Metacritic' score which is represents the composite score of a game out of 100 points and is associated with the game's quality based on reviews from vetted gaming review sources. Our aim is to understand what features are crucial to securing a high 'Metacritic' score which influences buyers decision making and a game's overall success.

For our project, we wanted to focus on the gaming industry, specifically that of online computer gaming. After doing some research on the space, we came down to the question, how can we predict the popularity of a game?

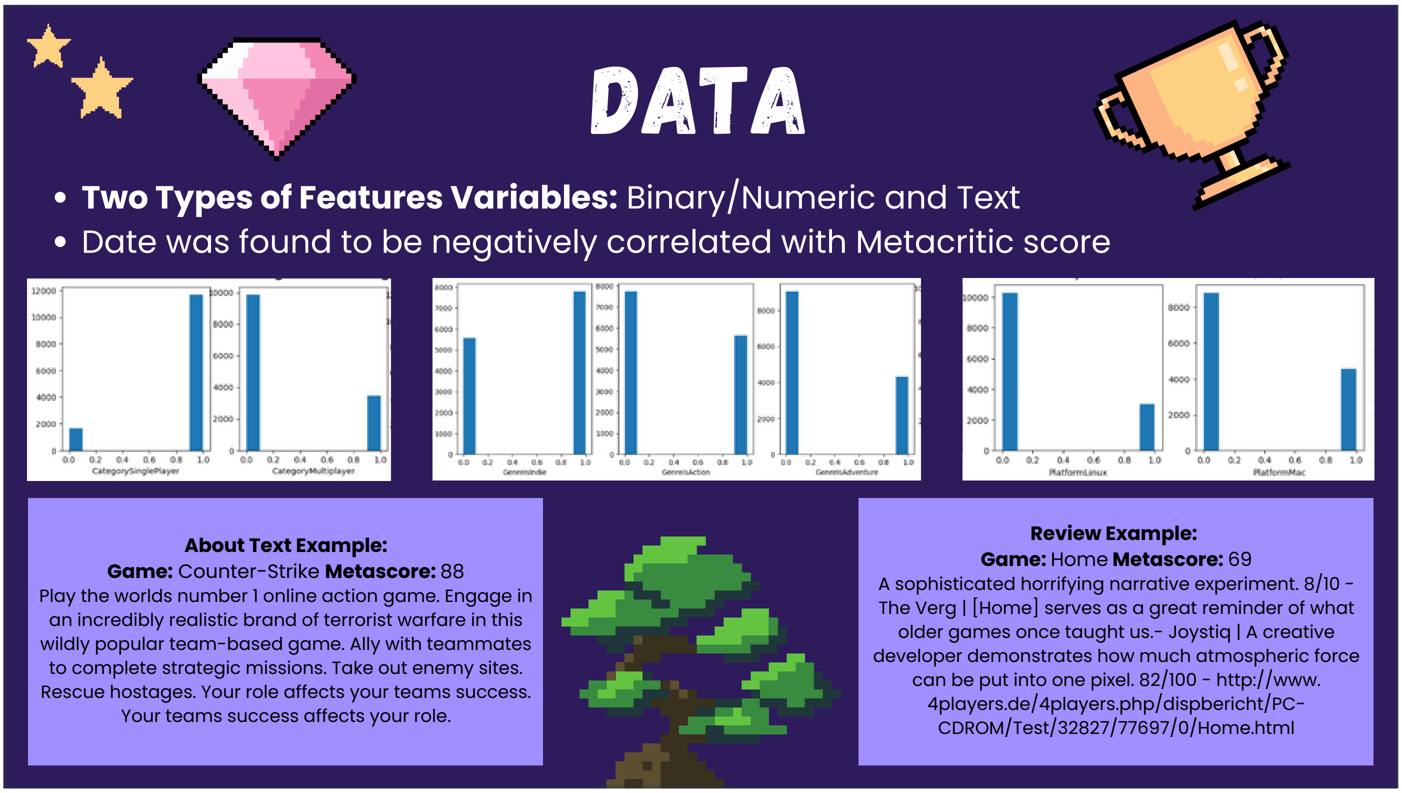
Our motivation for choosing this topic was three-fold. First, we recognize that machine learning is rapidly changing the gaming space. From analyzing player behavior to building more sophisticated games, the gaming world is readily accepting cutting edge modeling and technology. Second, most studios are already utilizing big data to inform their decision making outside of machine learning and artificial intelligence. Lastly, companies want to understand what makes a game fun and even addictive, and our hope is to analyze this further.


DATA
Steam Gaming Platform
Data with MetaScore



We sourced our data from Data World where we found a nearly cleaned data set from the Steam online gaming platform produced by Craig Kelly, (WHERE DOES HE WORK).

Steam is a video game digital distribution service and storefront from Valve. Steam offers various features, like digital rights management (DRM), game server matchmaking, anti-cheat measures, social networking and game streaming services. The service is the largest digital distribution platform for PC gaming, estimated around 75% of the market share in 2013 according to IHS Screen Digest.



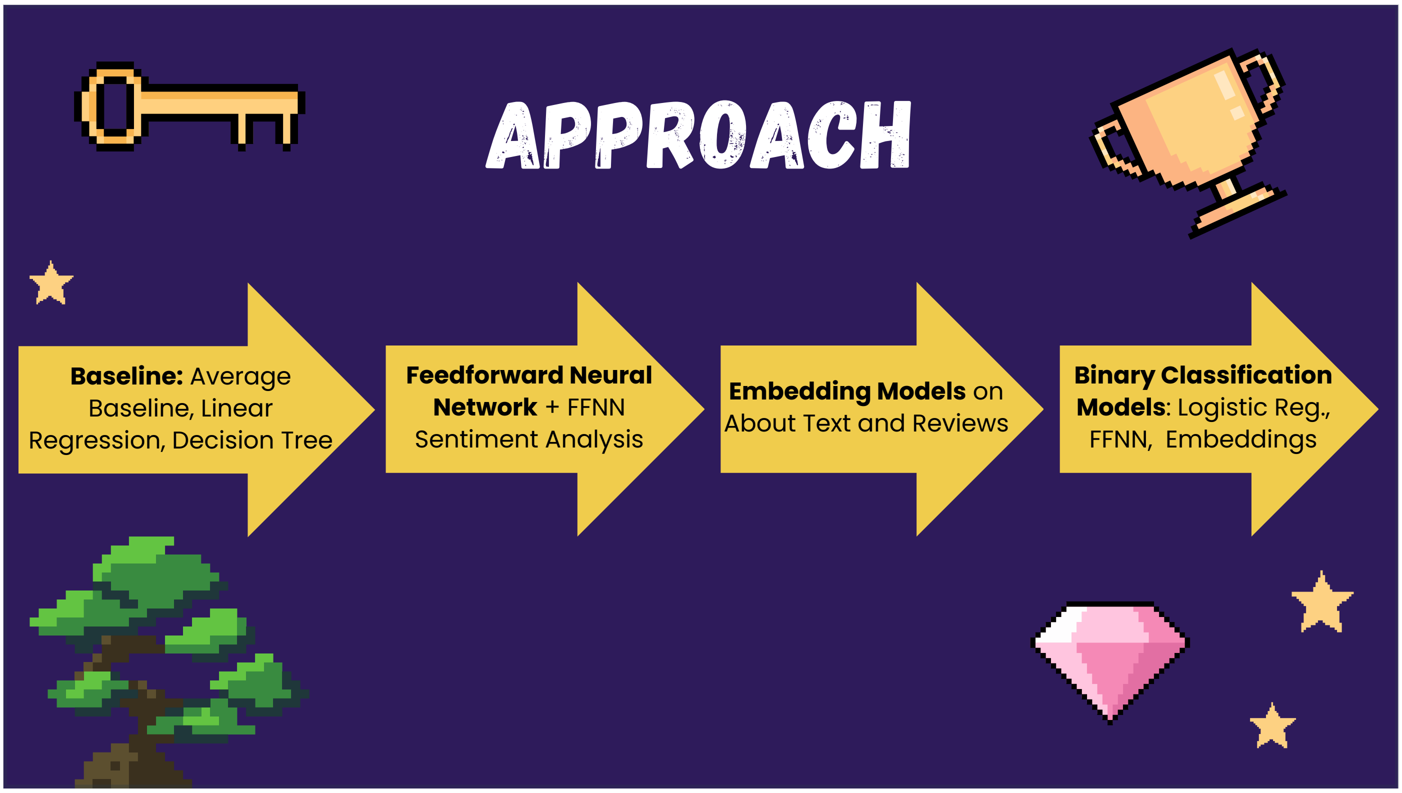
We focused on two types of variables from this dataset on Steam, binary variables and text data, with our output variable being Metacritic's MetaScore. The binary variables focus what is and is not included in or apart of each game. This includes languages, genres, platform, and category among others. For the text data, we focused on two summary texts, that of the player or critic reviews and the Metacritic Review Score: a METASCORE is a weighted average of reviews from top critics and publications for a given movie, TV show, video game, or album. We used this as our main output variable as a game's rating is a good indicator of its potential popularity. The MetaScore variable follows a nearly normal distribution and ranges from values of 1-100. We were able to deduce that any 0's found within the data were a lack of score rather than a score of 0.

In order to fit these variables to our model, we had to go through some precleaning first. To create our binary language variable, we split out each language as its own column and gave a 1 or a 0 based on whether the game supported that language or not. After ensuring the dates were all in the same formatting, we cleaned and transformed the release date into two parts: first we included a continuous variable with the precise age of the game since we found this was negatively correlated with meteoritic score, and second we included a set of indicator variables for the month since to include variation on the time of the year the game was released.

Below our two samples of the text data we used. The "about text" provides a summary of the game and reviews provides from critics and players.

APPROACH

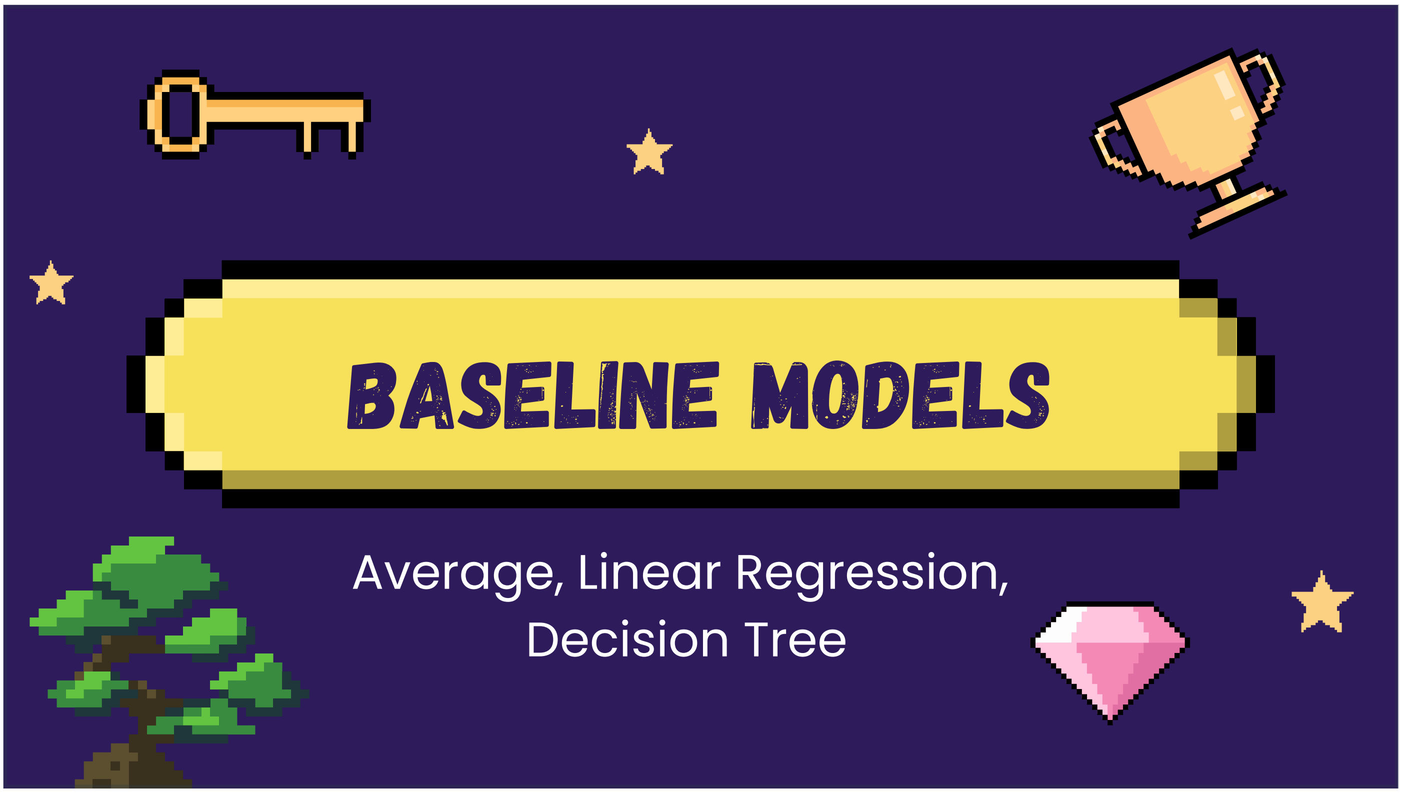
EDA and Feature Generation



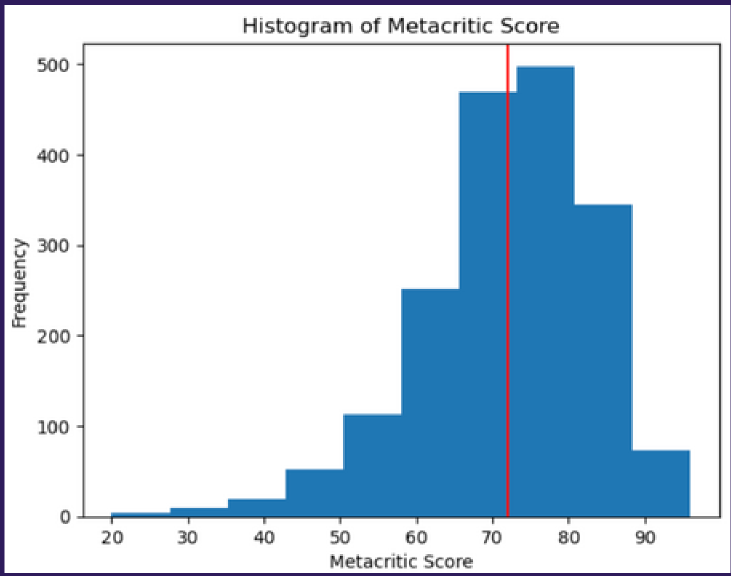
Our approach was to divide our model into four categories, our baseline models, feed forward neural network models, embedding models, and binary models. The last was added after observing the validation loss and accuracy of the other three groups. By starting with three baseline models, we were able to experiment and fine tune our models further to produce the lowest validation and test MSE's.

MODEL OVERVIEW	DATA	PRIMARY METRICS
Baseline Models (Average Baseline, Linear Regression, Decision Tree)	Non-Text	MSE
FFNN + FFNN with Semantic Analysis	Non-Text (+ semantic analysis on About Text)	MSE
Embeddings Models on About Text and Reviews	Text	MSE
Binary Classification Models Logistic Regression --> FFNN --> Embeddings Models	Both	Accuracy

add text/non-text columns


BASELINE MODELS
Average, Linear Regression,
Decision Tree

AVERAGE BASELINE

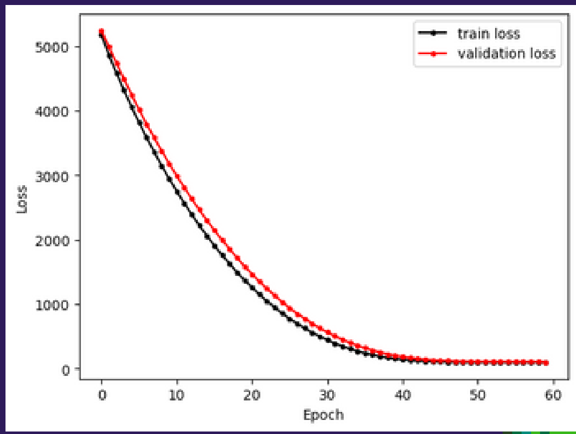


MSE (*training*): 128.01

MSE (*test*): 106.63

LINEAR REGRESSION

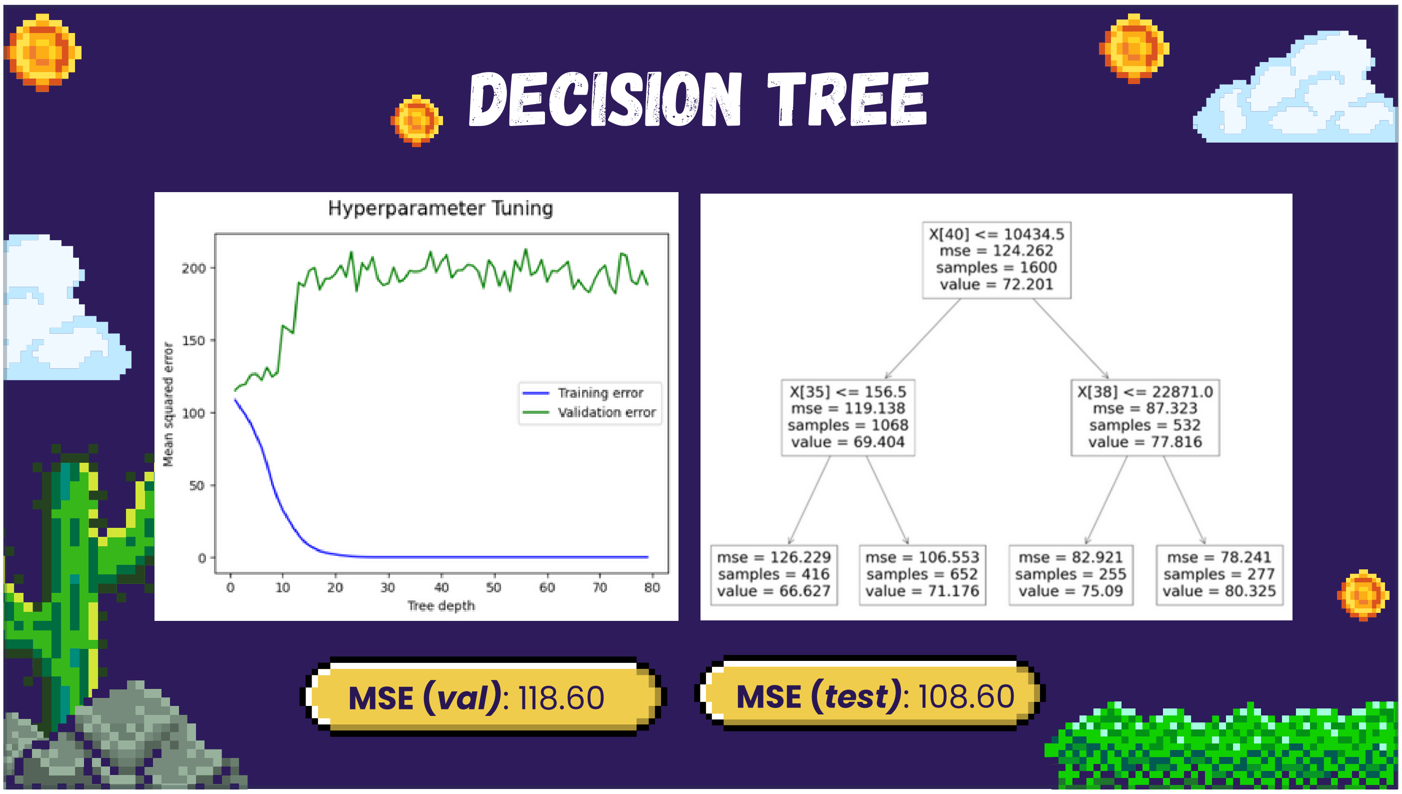
EPOCHS	BATCH	OPTIMIZER	LEARNING RATE	VAL LOSS
10	1	Adam	0.01	126.35
10	8	Adam	0.01	3179.30
10	8	Adam	0.1	143.66
15	8	Adam	0.1	119.46
60	8	Adam	0.01	102.55
30	4	Adam	0.01	104.56
150	15	Adam	0.01	103.03
10	10	SGD	0.001	126
20	10	SGD	0.001	104.80
100	32	SGD	0.001	103.28
150	64	SGD	0.001	103.42



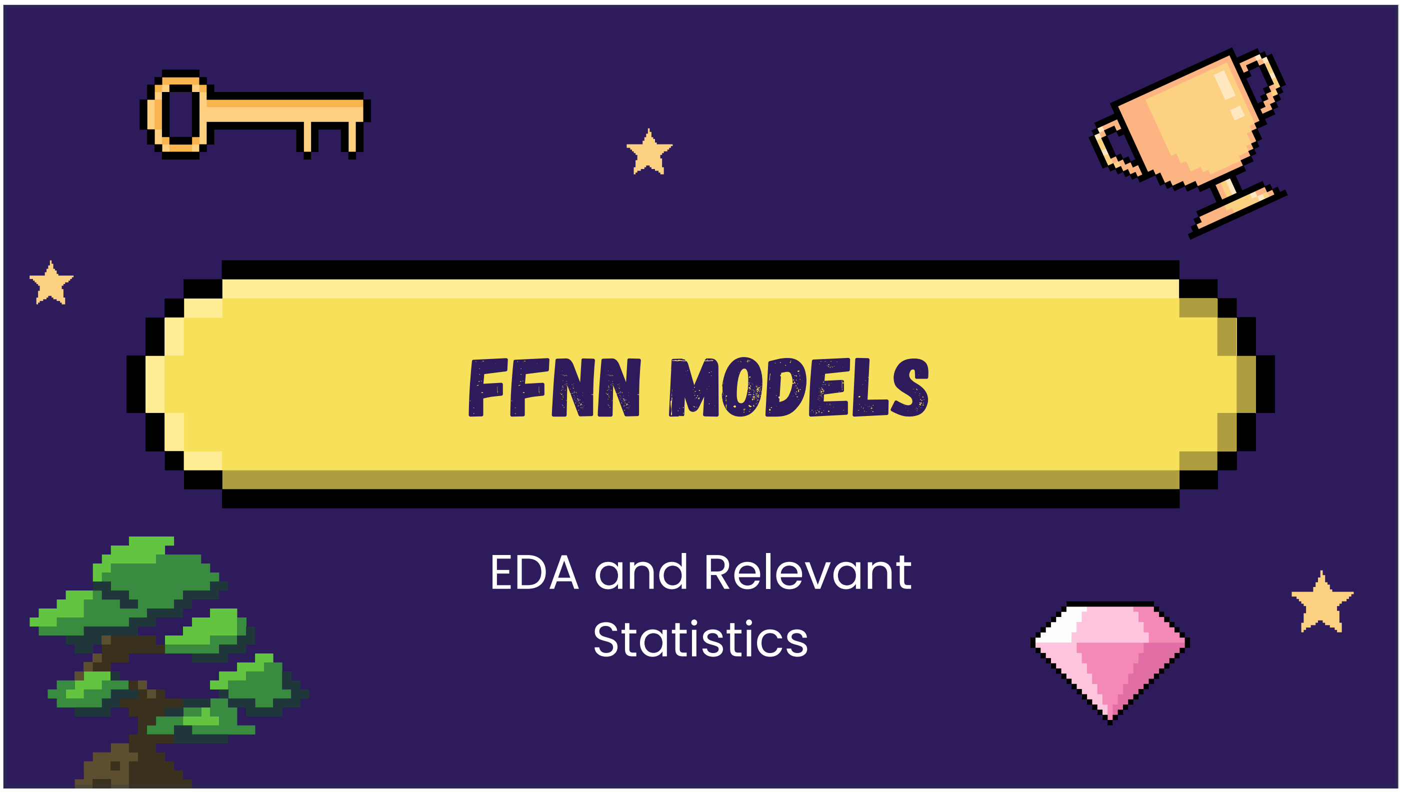
MSE (*val*): 102.55

MSE (*test*): 83.01

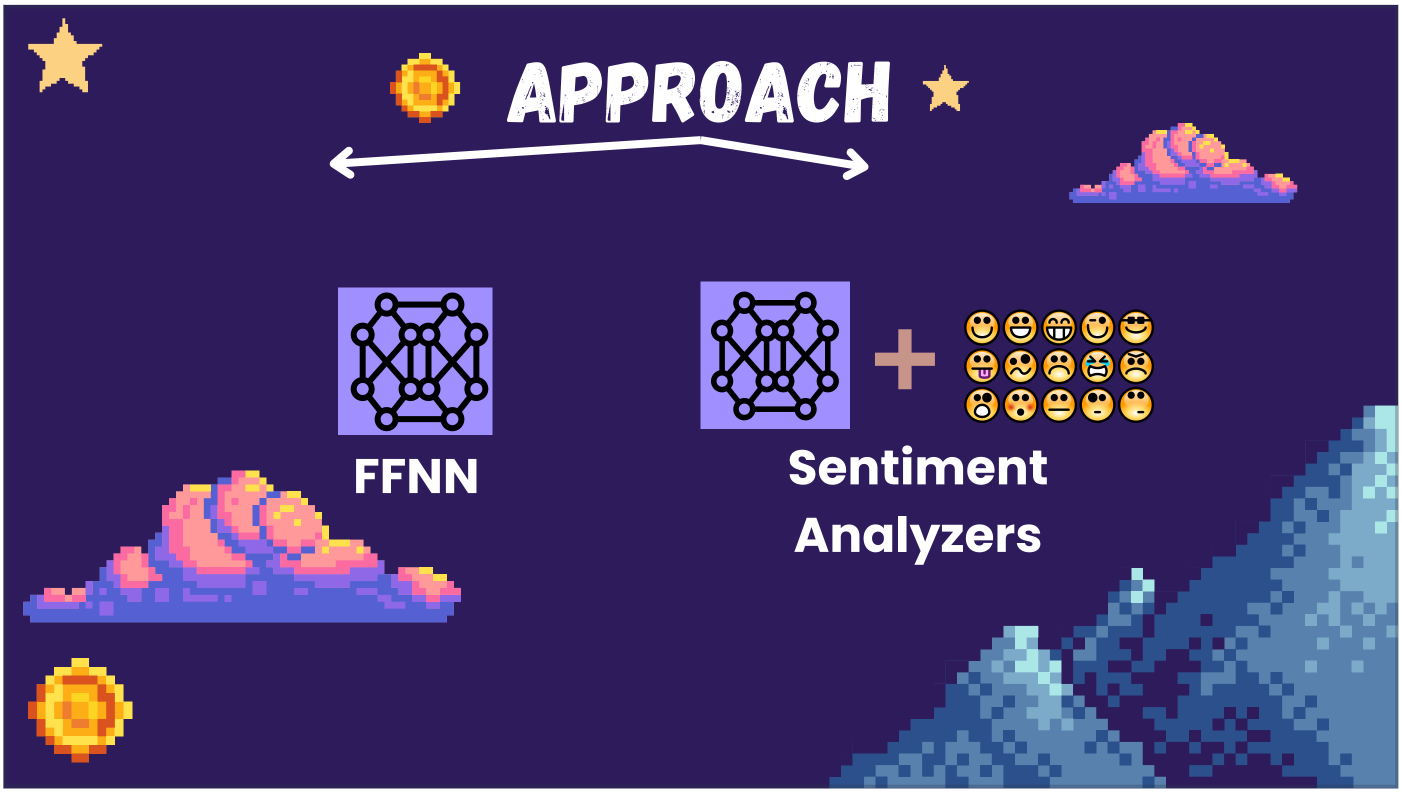
Tested out different learning rates, batch sizes, etc. Selected model with lowest training accuracy



Models with shorter depths worked best on val data.
Chose depth of 2

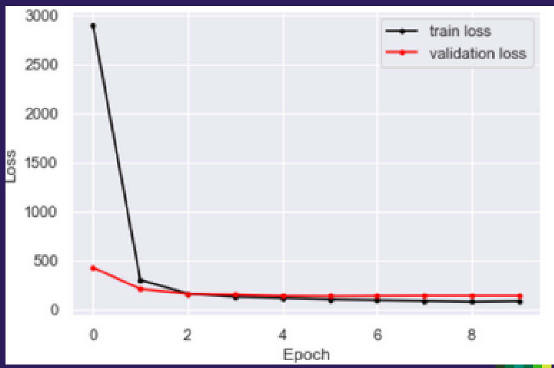


Non-linearity: Feedforward neural networks are capable of modeling complex, non-linear relationships between input features and output targets. This is important for predicting metacritic scores, which are influenced by a wide range of factors (e.g., gameplay mechanics, graphics, story, etc.) that can interact with each other in complex ways.


APPROACH
FFNN
Sentiment Analyzers

FEEDFORWARD NEURAL NETWORK

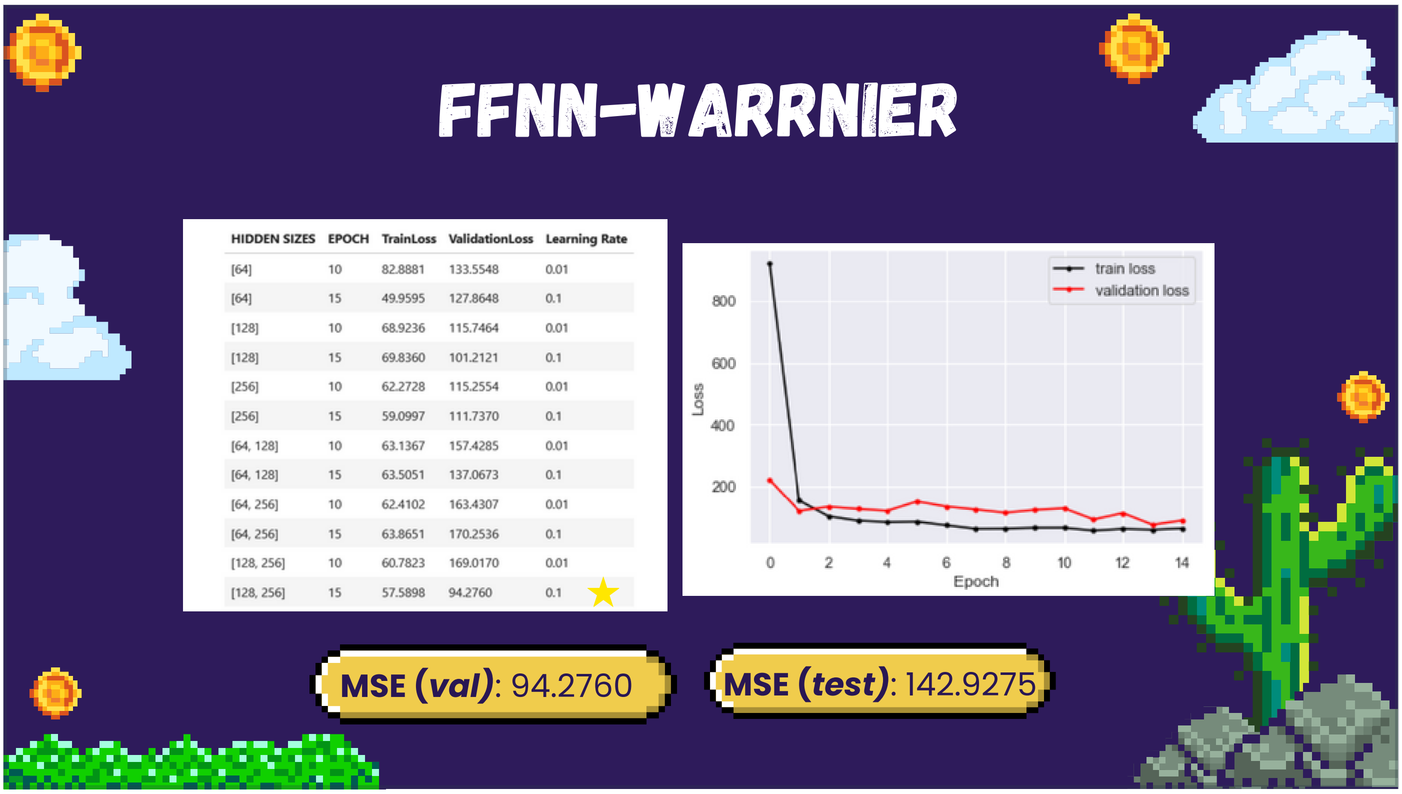
HIDDEN SIZES	EPOCH	TrainLoss	ValidationLoss	Learning Rate
[64]	10	47.0416	141.9114	0.01
[64]	15	38.3844	157.5400	0.01
[64]	10	210.6533	185.1153	0.1
[64]	15	182.2950	261.1494	0.1
[128]	10	82.3435	135.7154	0.01 ★
[128]	15	44.7764	164.2129	0.01
[128]	10	93.7240	209.1945	0.1
[128]	15	152.2401	209.0557	0.1
[256]	10	79.5607	149.1673	0.01
[256]	15	44.7534	161.9704	0.01
[256]	10	3245.7092	1190.1699	0.1
[256]	15	94.7610	228.3825	0.1
[64, 128]	10	66.1479	176.4990	0.01



MSE (*val*): 135.7154

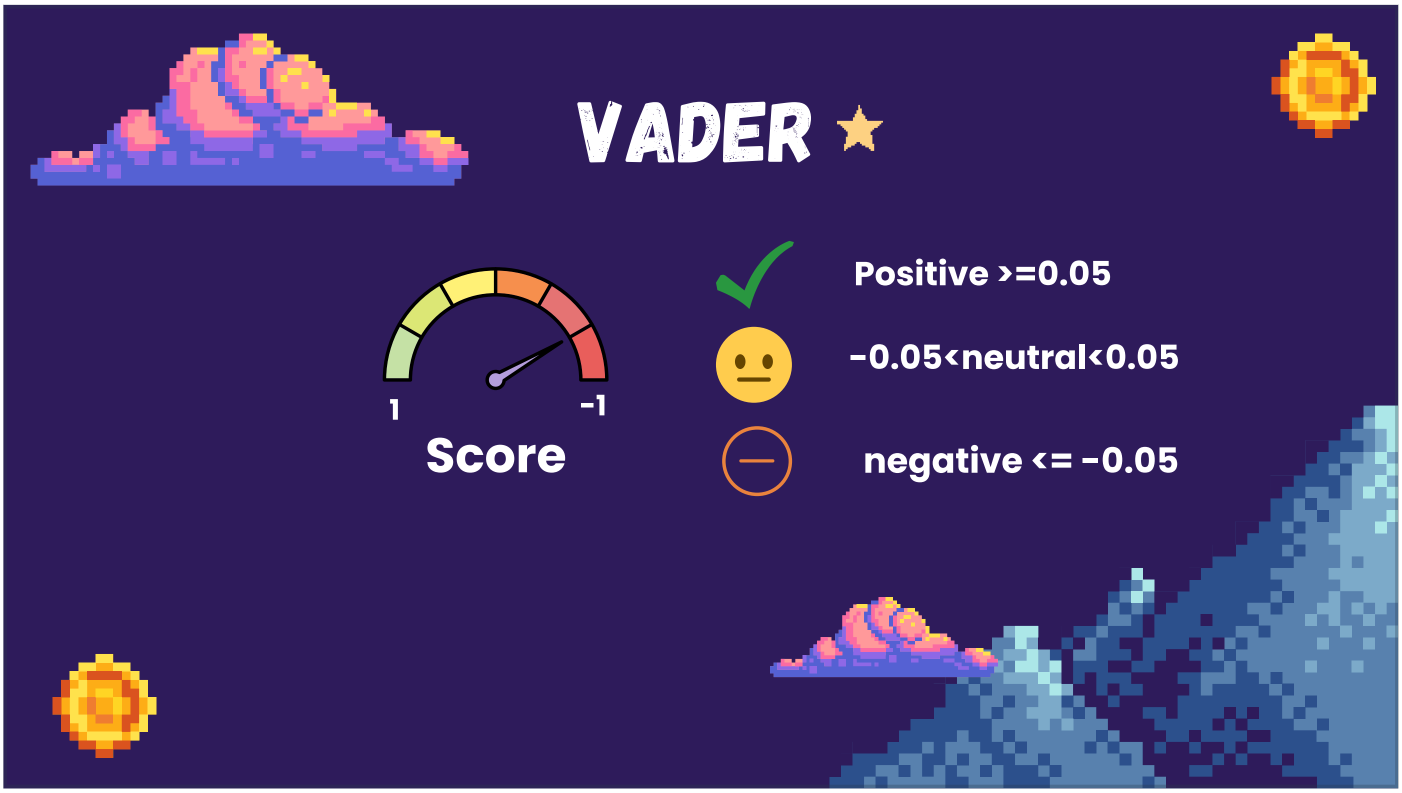
MSE (*test*): 104.7759


WARRNIER
JOY
Anger
Sadness
Fear
Disgust



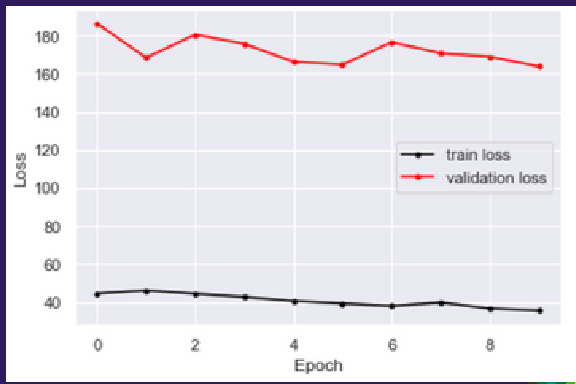
HIDDEN SIZES	EPOCH	TrainLoss	ValidationLoss	Learning Rate
[64]	10	82.8881	133.5548	0.01
[64]	15	49.9595	127.8648	0.1
[128]	10	68.9236	115.7464	0.01
[128]	15	69.8360	101.2121	0.1
[256]	10	62.2728	115.2554	0.01
[256]	15	59.0997	111.7370	0.1
[64, 128]	10	63.1367	157.4285	0.01
[64, 128]	15	63.5051	137.0673	0.1
[64, 256]	10	62.4102	163.4307	0.01
[64, 256]	15	63.8651	170.2536	0.1
[128, 256]	10	60.7823	169.0170	0.01
[128, 256]	15	57.5898	94.2760	0.1

https://github.com/JULIELab/EmoMap/blob/master/coling18/main/lexicon_creation/lexicons/Warriner_BE.tsv


VADER
1
-1
Score
Positive >=0.05
-0.05<neutral<0.05
negative <= -0.05

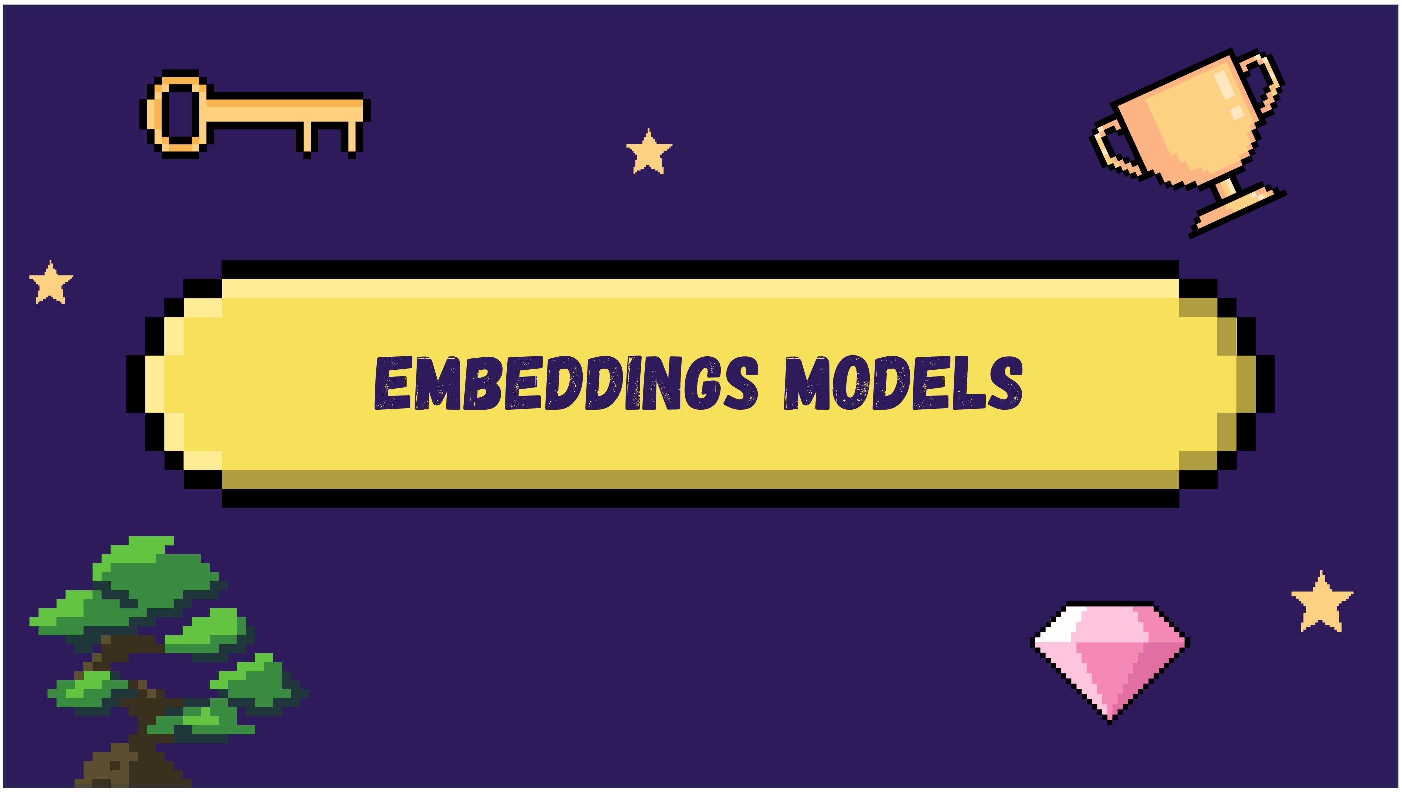
FFNN-VADER

HIDDEN SIZES	EPOCH	TrainLoss	ValidationLoss	Learning Rate
[64]	10	81.1018	184.9839	0.01
[64]	15	99.4495	181.2249	0.1
[128]	10	35.4933	163.8759	0.01 ★
[128]	15	73.47339	179.8921	0.1
[256]	10	59.1582	169.2039	0.01
[256]	15	63.9836	183.3107	0.1
[64, 128]	10	57.2427	199.2272	0.01
[64, 128]	15	72.9001	191.5959	0.1
[64, 256]	10	70.5527	185.0179	0.01
[64, 256]	15	64.9663	220.1869	0.1
[128, 256]	10	57.0486	209.0135	0.01
[128, 256]	15	62.0470	217.4754	0.1



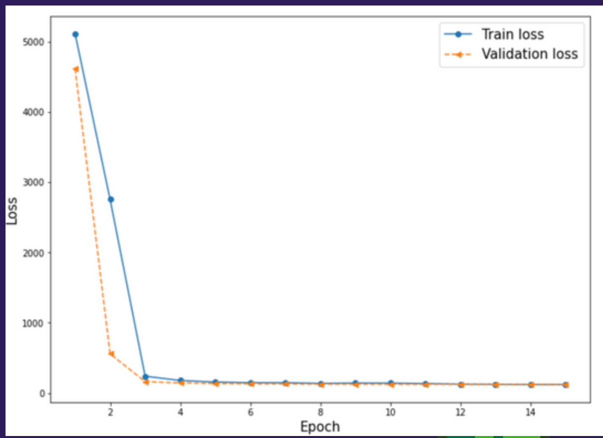
MSE (*val*): 163.8759

MSE (*test*): 138.0928


EMBEDDINGS MODELS

EMBEDDINGS MODEL (ABOUT TEXT)

EPOCHS	BATCH	LEARNING RATE	EMBEDDINGS SIZE	DROPOUT	LAYERS	TRAIN LOSS	VAL LOSS
15	4	0.001	20	0.05	16,16,1	59.16	114.82
15	8	0.001	20	0.05	16,16,1	102.00	119.46
15	12	0.001	20	0.05	16,16,1	122.15	122.82
15	16	0.001	20	0.05	16,16,1	121.36	123.90
15	8	0.001	10	0.05	16,16,1	110.00	120.93
15	8	0.001	30	0.05	16,16,1	96.82	118.83
15	8	0.001	20	0.1	16,16,1	135.48	119.41
15	8	0.001	20	0.05	16,16,16	105.46	121.49
15	8	0.001	20	0.05	32,16,1	95.71	118.16
15	8	0.01	20	0.05	16,16,1	66.35	163.81
30	8	0.001	20	0.05	16,16,1	56.22	112.29
30	8	0.001	20	0.2	16,16,1	156.68	113.69
30	8	0.01	20	0.1	16,16,1	135.68	127.21
30	12	0.001	20	0.05	16,16,1	76.71	114.26



MSE (*val*): 122.82

MSE (*test*): 106.88

- About text is the description of the game from the developer

- both embeddings models had a simple architecture

- goal is to see if there are any words or phrases that are more commonly seen in successful games

- vocab size of ~23000

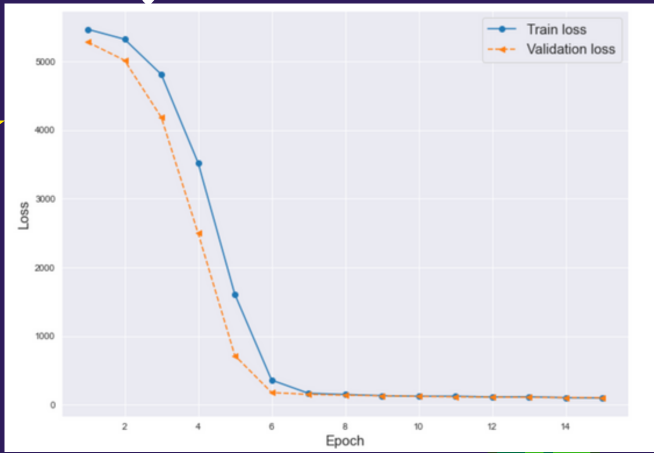
- major dropoff in the first epoch which implied higher learning rate - higher learning rate ruined performance

- batch size was the most important hyperparameter

- goal was to get train and val loss to match as closely as possible (to improve generalization of model)

EMBEDDINGS MODEL (REVIEW TEXT)

EPOCHS	BATCH	LEARNING RATE	EMBEDDINGS SIZE	DROPOUT	LAYERS	TRAIN LOSS	VAL LOSS
15	4	0.001	20	0.05	16,16,1	52.25	94.56
15	8	0.001	20	0.05	16,16,1	79.08	97.86
15	12	0.001	20	0.05	16,16,1	86.28	98.53
15	12	0.001	10	0.05	16,16,1	98.25	102.48
15	16	0.001	20	0.05	16,16,1	104.50	104.37
15	16	0.001	10	0.05	16,16,1	108.35	105.28
15	16	0.001	30	0.05	16,16,1	101.87	104.06
15	12	0.001	10	0.1	16,16,1	132.02	104.30
15	12	0.001	10	0.05	16,16,16	114.44	99.01
15	12	0.001	10	0.05	32,16,1	90.60	97.10
15	12	0.01	10	0.05	16,16,1	40.16	91.29
30	12	0.001	10	0.05	16,16,1	79.16	92.39
30	12	0.001	10	0.2	16,16,1	189.82	100.38
30	12	0.001	10	0.1	16,16,1	117.56	95.37
30	12	0.001	10	0.05	16,16,1	79.16	92.39



MSE (*val*): 102.48

MSE (*test*): 91.93

- Review text includes official game reviews from magazines (including the magazine's individual rating for the game sometimes)

- vocab size of ~7000 (likely due to shorter length sequences)

- This is a little redundant with our target variable since metacritic is a composite score of individual game reviews

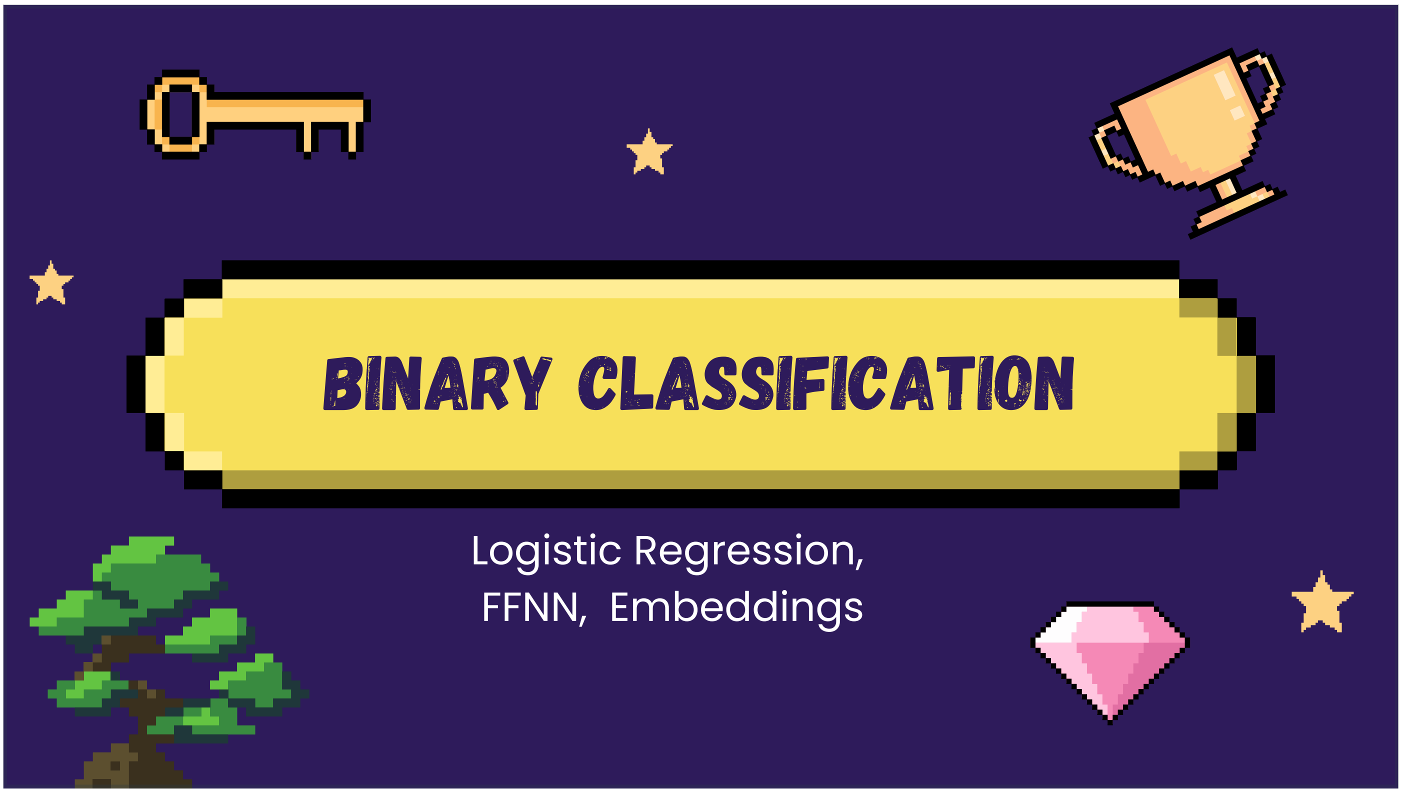
- Better performance than About Text embeddings model

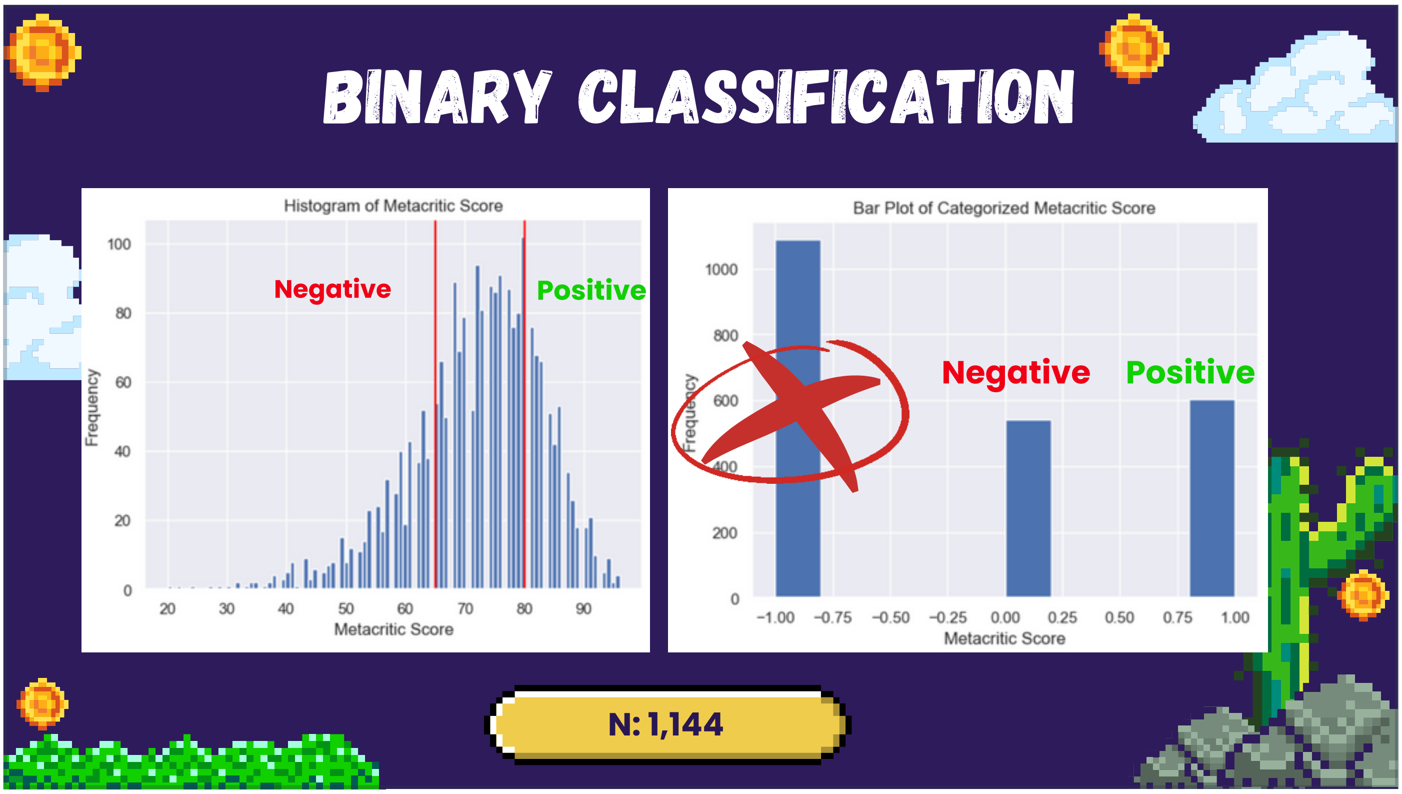
- Performed better with smaller embeddings size and larger batch size

- other hyperparameters were the same as previous

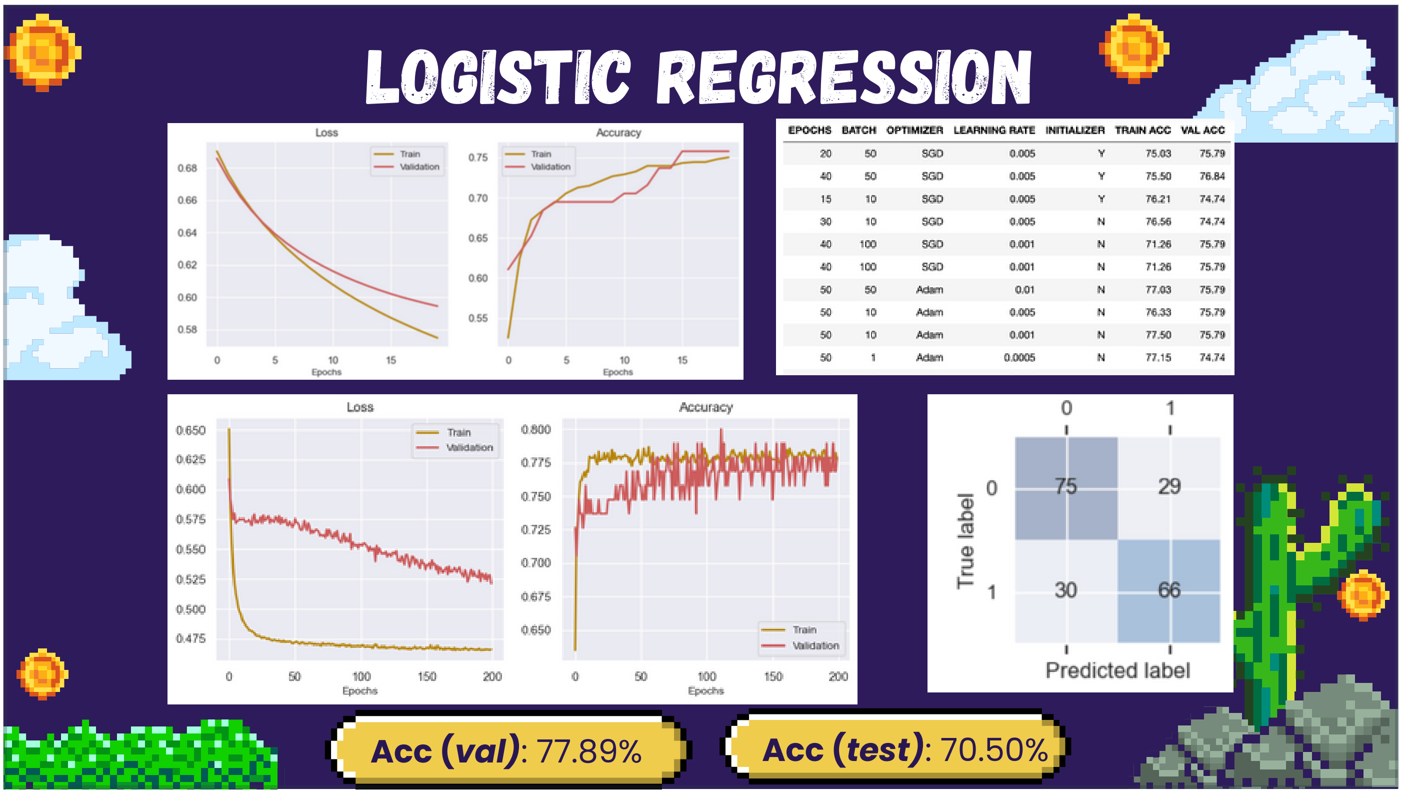
- notice the validation loss is consistently lower than the training loss which suggests underfitting

- changing learning rate or epochs had no effect


BINARY CLASSIFICATION
Logistic Regression,
FFNN, Embeddings

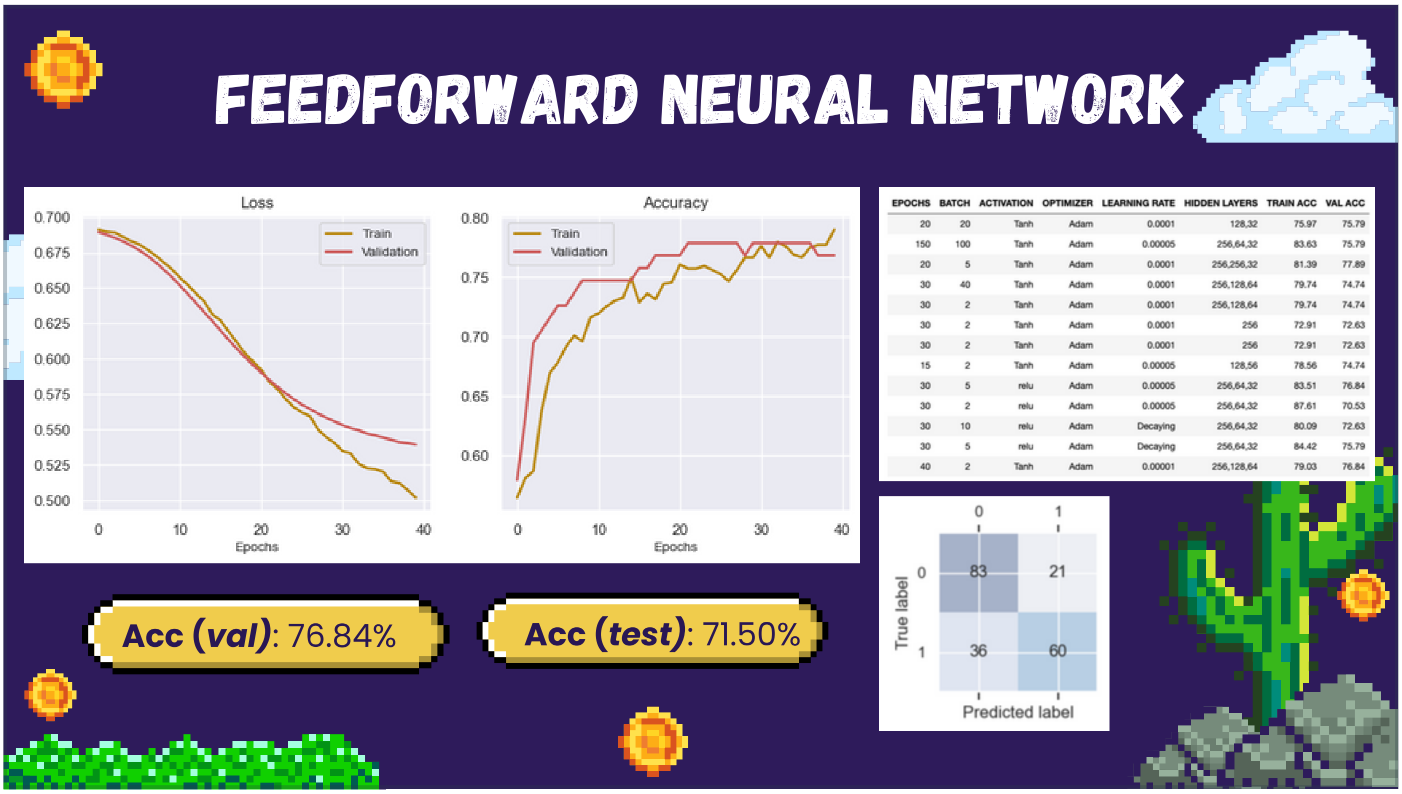


Played around with this score. balancing wanted to have more extreme scores and sample size



Lower accuracy for smoother learning curves. Bigger batch size, higher learning rate had higher accuracy. Tried decaying learning rate.

Decided to go with higher accuracy because even with variation, hovered above the accuracy for other curves.



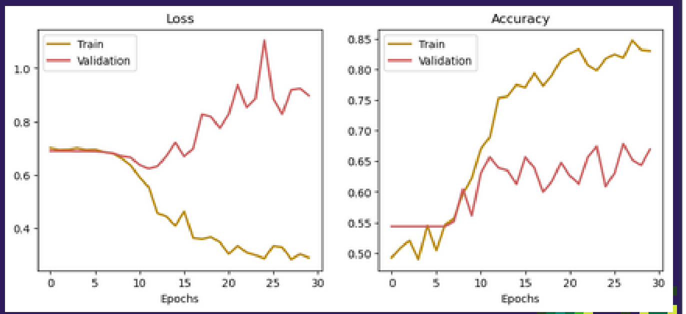
EPOCHS	BATCH	ACTIVATION	OPTIMIZER	LEARNING RATE	HIDDEN LAYERS	TRAIN ACC	VAL ACC
20	20	Tanh	Adam	0.0001	128,32	75.97	75.79
150	100	Tanh	Adam	0.00005	256,64,32	83.63	75.79
20	5	Tanh	Adam	0.0001	256,256,32	81.39	77.89
30	40	Tanh	Adam	0.0001	256,128,64	79.74	74.74
30	2	Tanh	Adam	0.0001	256,128,64	79.74	74.74
30	2	Tanh	Adam	0.0001	256	72.91	72.63
30	2	Tanh	Adam	0.0001	256	72.91	72.63
15	2	Tanh	Adam	0.00005	128,56	78.56	74.74
30	5	relu	Adam	0.00005	256,64,32	83.51	76.84
30	2	relu	Adam	0.00005	256,64,32	87.61	70.53
30	10	relu	Adam	Decaying	256,64,32	80.09	72.63
30	5	relu	Adam	Decaying	256,64,32	84.42	75.79
40	2	Tanh	Adam	0.00001	256,128,64	79.03	76.84

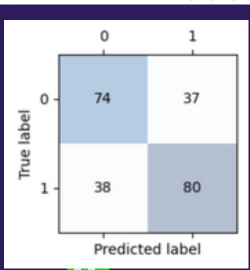
Final decision based on the shape of the learning curve and difference between training and accuracy. Not the highest val acc but one of them.

Layers: 256,64,32

EMBEDDINGS (ABOUT TEXT)

EPOCHS	BATCH	LEARNING RATE	EMBEDDINGS SIZE	DROPOUT	LAYERS	TRAIN ACC	VAL ACC
20	5	0.001	8	0.6	8	0.95	0.67
20	5	0.001	8	0.6	16	0.95	0.67*
20	5	0.001	8	0.6	16,8	0.92	0.69
100	5	0.001	8	0.6,0.8	16,8	0.90	0.66
30	5	0.001	8	0.6,0.4	16,8	0.98	0.69
30	5	0.001	8	0.6,0.4	16,8,8	0.98	0.68
30	5	0.001	8	0.6,0.6,0.3,0.5,0.5	16,8,4	0.81	0.66
30	5	Decaying	8	0.6,0.6,0.3,0.5,0.5	16,8,4	0.83	0.67
30	20	0.005	8	0.6,0.6,0.3,0.5,0.5	16,8,4	0.84	0.66
30	10	0.005	8	0.6,0.6,0.5,0.5	16,8,4	0.83	0.67





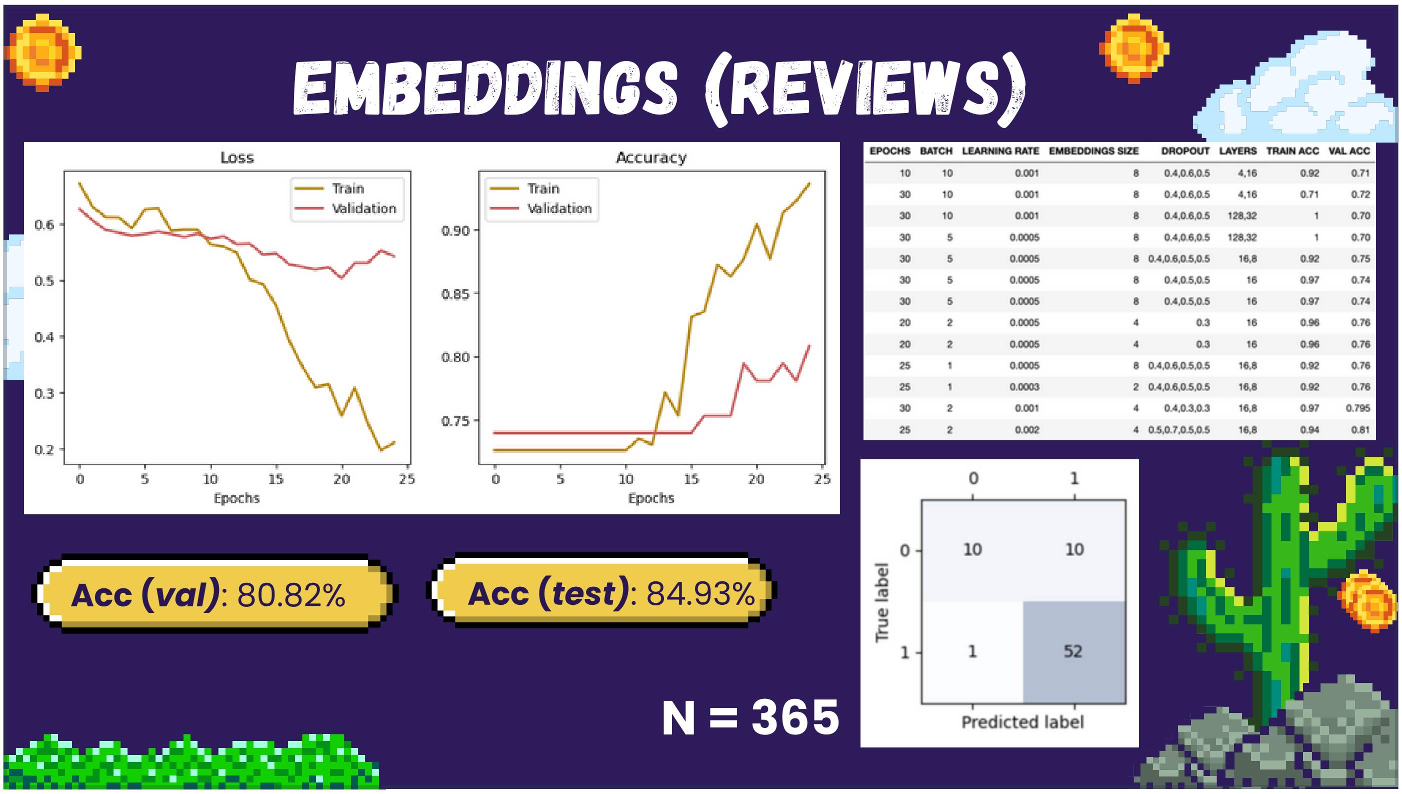
Acc (*val*): 66.96%

Acc (*test*): 69.87%

Challenge: Getting the accuracy up without over fitting. A lot of models got up to 100% on the training data. Multiple drop out layers.

Selected the model with the smallest gap between training and val without taking too big of a hit on accuracy.

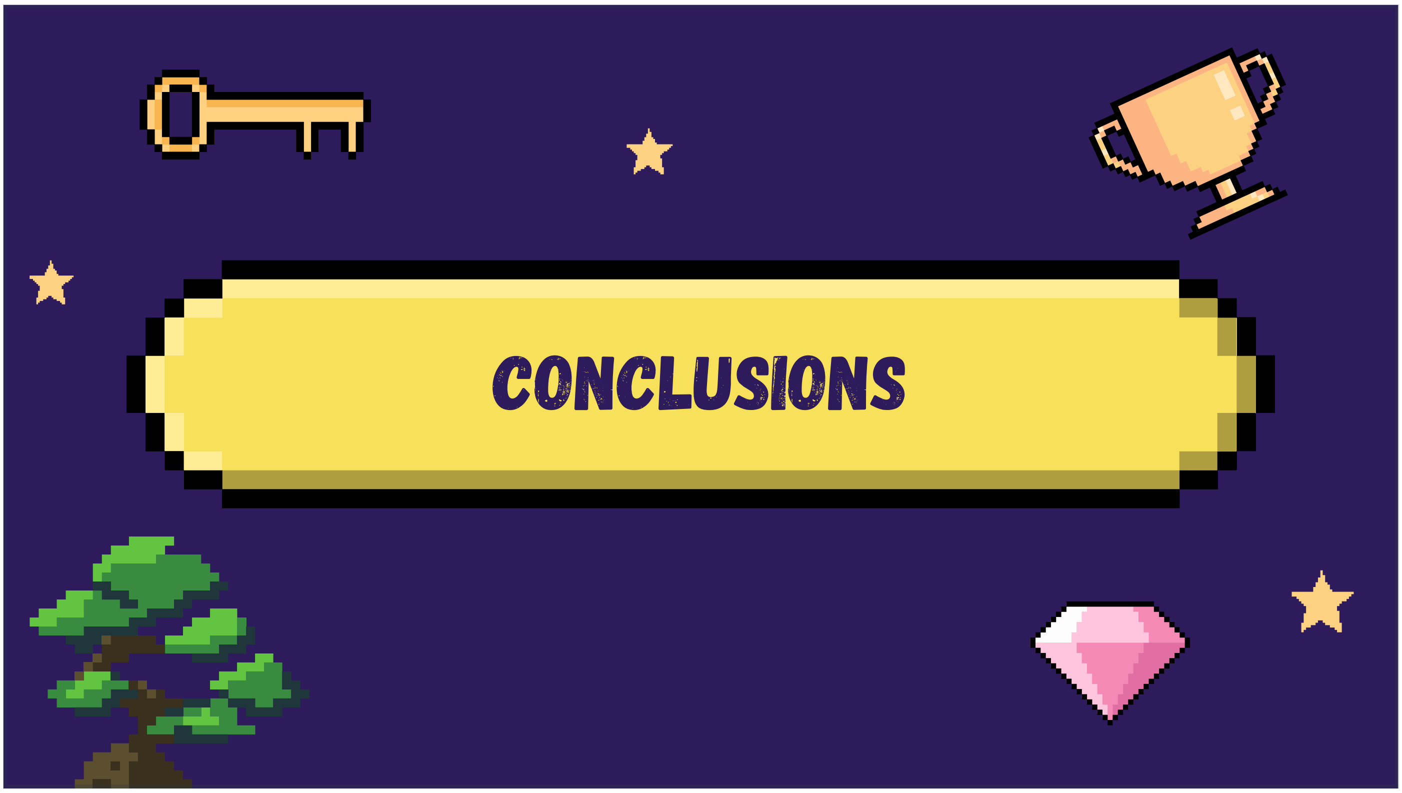
Not a good match between training and test?



EPOCHS	BATCH	LEARNING RATE	EMBEDDINGS SIZE	DROPOUT	LAYERS	TRAIN ACC	VAL ACC
10	10	0.001	8	0.4,0.6,0.5	4,16	0.92	0.71
30	10	0.001	8	0.4,0.6,0.5	4,16	0.71	0.72
30	10	0.001	8	0.4,0.6,0.5	128,32	1	0.70
30	5	0.0005	8	0.4,0.6,0.5	128,32	1	0.70
30	5	0.0005	8	0.4,0.6,0.5,0.5	16,8	0.92	0.75
30	5	0.0005	8	0.4,0.5,0.5	16	0.97	0.74
30	5	0.0005	8	0.4,0.5,0.5	16	0.97	0.74
20	2	0.0005	4	0.3	16	0.96	0.76
20	2	0.0005	4	0.3	16	0.96	0.76
25	1	0.0005	8	0.4,0.6,0.5,0.5	16,8	0.92	0.76
25	1	0.0003	2	0.4,0.6,0.5,0.5	16,8	0.92	0.76
30	2	0.001	4	0.4,0.3,0.3	16,8	0.97	0.795
25	2	0.002	4	0.5,0.7,0.5,0.5	16,8	0.94	0.81

Again, tried to minimize overfitting as much as possible.

One model had a high accuracy. Smaller embedding dimension and multiple dropout layers for generalizability.


CONCLUSIONS

★ CONCLUSION

Takeaways:

- Linear regression was fairly successful, likely due to large amount of binary data
- Embedding reviews model is most effective (but potentially least useful)

Follow up steps to improve model:

- LSTM/BERT models for text analysis
- Fused model (incorporate text data)
- Increase/improve features

Continue to build on this idea and include non computer gaming


PLAY COD NOW!
THANK YOU FOR LISTENING

MODEL OVERVIEW	TEST MSE
Average Baseline	106.63
Linear Regression	83.01
Decision Tree	108.60
FFNN	104.7759
FFNN with Semantic Analysis	Warrnier: 142.9275 VADER: 138.0928
Embeddings Models on About Text and Reviews	About Text: 106.88 Reviews: 91.93

add text/non-text columns

INDIVIDUAL CONTRIBUTIONS

Name	DATA CLEANING	BASELINE/ LIN REG	DECISION TREE	FFNN	FFNN + Sentiment Analysis	EMBEDDINGS	BIN CLASS-IFICATION MODELS	SLIDES
Amanda	X	X				X	X	X
Nabiha	X		X	X	X			X
Varun	X					x		X
Josie	X			X				X