ITI106 - Foundations of Deep Learning

Assignment

14 Nov 2023

Magdalene Cheong (23A467J)

Task 1: to develop a solution using a fully-connected, feedforward neural network using Keras.

Task 2: to compare the differences between classical machine learning and neural network/deep learning.

Task 1

Objective:

The objective is to develop a deep learning model to detect text with suicidal intent.

About the dataset:

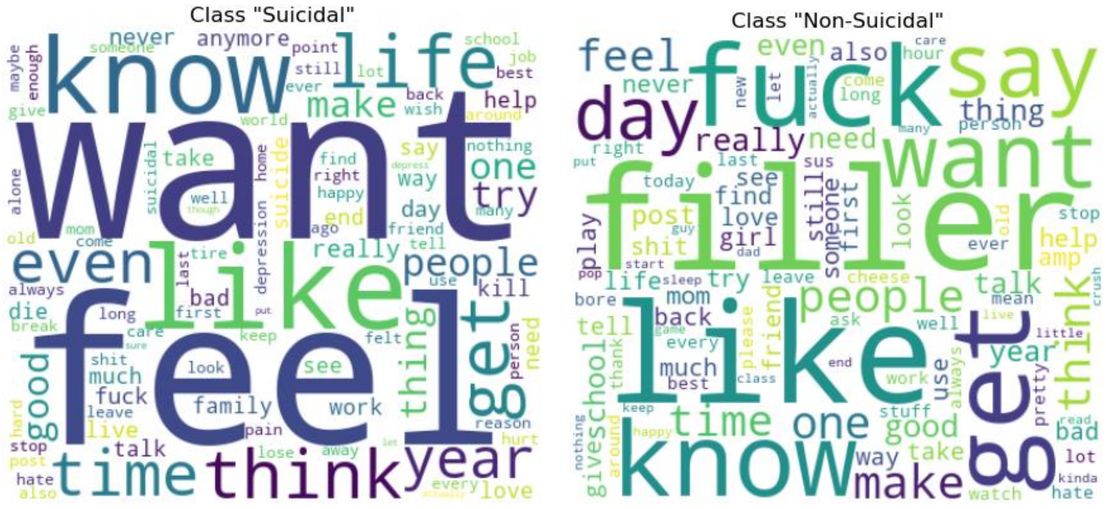
The dataset is obtained from Kaggle (Ref: https://www.kaggle.com/datasets/nikhileswarkomati/suicide-watch) with credits to the author. The data was extracted using Pushshift API from "SuicideWatch" and "teenagers" subreddit of Reddit. All posts from "SuicideWatch" are labelled as 'suicide' and those from teenagers as 'non-suicide'.

The dataset consists of 2 columns: 'text' and 'class'. 'text' contains the posts to be classified if it's of suicidal context. 'class' is the label to indicate if the 'text' is suicidal-indicative.

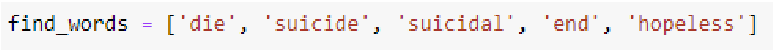
Data Pre-processing/Cleaning:

1) Examine contents to confirm texts are representative
   - Sample review on texts from both classes to check that the contents are relevant and applicable to the topic.
2) Check for null rows
   - There are zero null rows.
3) Check for unique values of 'class'.
   - Unique values are 'suicide' and 'non-suicide'. This needs to be checked as this column has to be converted to binary form in order to be understood by the algorithm. This is to ensure that all rows can be correctly and fully converted.
4) Convert label column 'class' to binary (from non-suicidal/suicidal to 0/1).
5) Convert 'text' column:
   a. Change to lowercase, remove non-alphabet and numerical characters.
   b. Strip text of white space.
   c. Tokenization.
   d. Remove stop words and non-English words.
   e. Lemmatization
   f. Keep only words > 2 characters long.

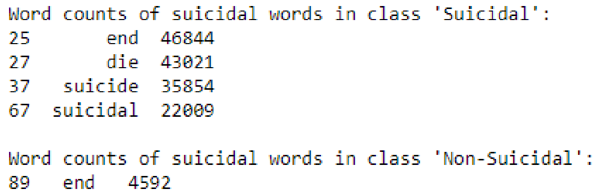
EDA:



1. Word cloud plotted for "Suicidal" class is not immediately indicative that suicidal words are of highest frequency in "Suicidal" class as compared to "Non-Suicidal". This is due to high number of common high frequency words, therefore suicidal words are not ranked at the top.
2. To compare, a search is done to compare frequency of a list of 5 words which are commonly linked to suicidal cases. Words like 'die', 'suicide', 'suicidal' and 'end' have a high occurrence in 'Suicidal' subset in comparison - a total 147728 counts in "Suicidal" class as compared to 4592 in "Non-suicidal" class. The distinction in frequency of suicidal words deems the dataset suitable developing the model.

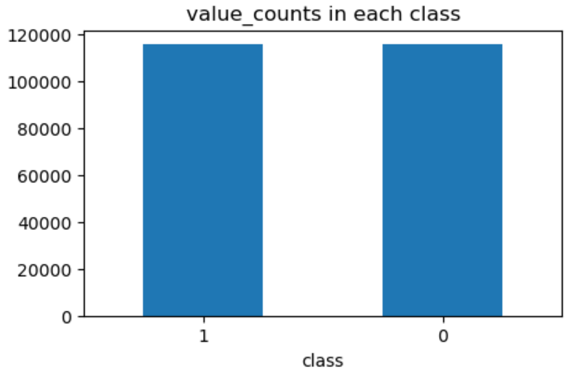


```
find_words = ['die', 'suicide', 'suicidal', 'end', 'hopeless']
```



```
Word counts of suicidal words in class 'Suicidal':
25        end  46844
27        die  43021
37    suicide  35854
67   suicidal  22009

Word counts of suicidal words in class 'Non-Suicidal':
89    end    4592
```

3. Check for imbalanced data
   - The dataset consists of 232074 rows of data where the class is evenly distributed at 50%.



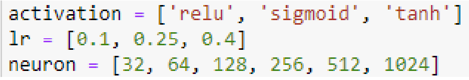
4. TDIDF Vectorization is then used to convert the dataset to be used for training.

Experimentation:

Approach: Firstly, establish the base model by understanding how the model behaves with different activation function at different number of neurons and learning rate. After having established the base model, then evaluate on the number of layers/number of neurons, and consider if regularisation and early stopping is needed.

1) Preliminary Run

- to compare different activation functions, number of neurons and learning rates in order to choose a base model
- constant: number of epoch (25), output layer activation function (sigmoid), number of layers (10)
- variable:

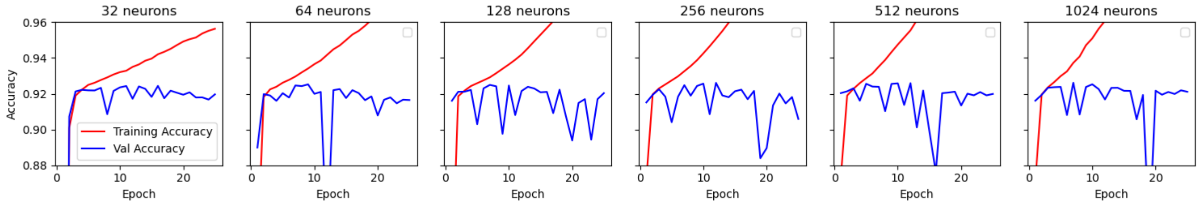


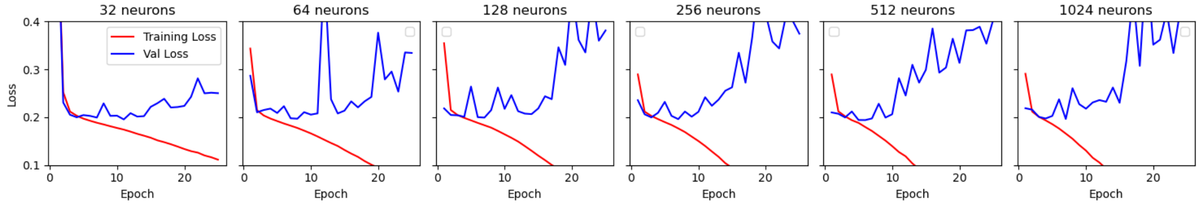
```
activation = ['relu', 'sigmoid', 'tanh']
lr = [0.1, 0.25, 0.4]
neuron = [32, 64, 128, 256, 512, 1024]
```

Average time taken to complete 1 epoch (s):

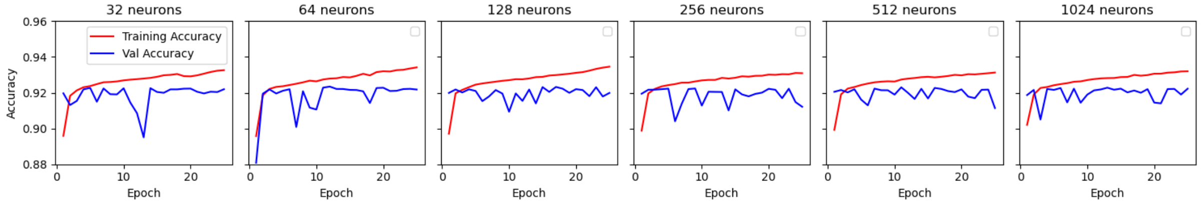
LR/neurons	**32**	**64**	**128**	**256**	**512**	**1024**
0.1	7	8	13	22	56	151
0.25	7	9	14	23	63	157
0.4	7	9	14	22	67	157

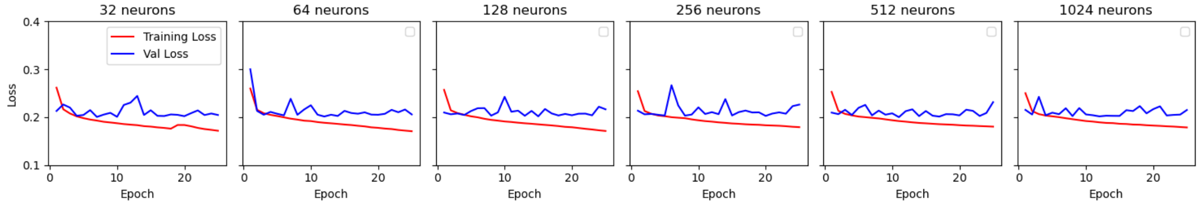
Relu:


Training and Validation Accuracy - Activation function: relu, Learning Rate: 0.1, Number of layers: 10

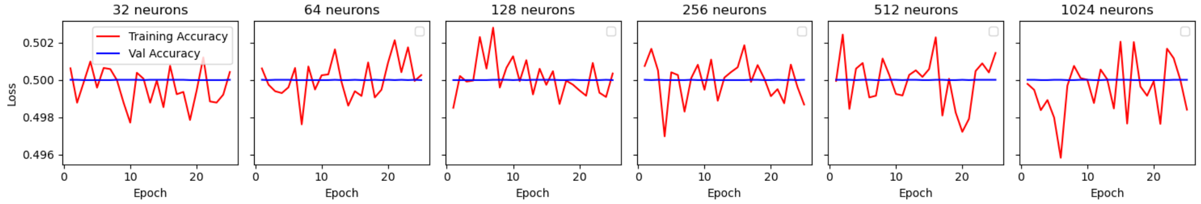

Training and Validation Loss - Activation function: relu, Learning Rate: 0.1, Number of layers: 10

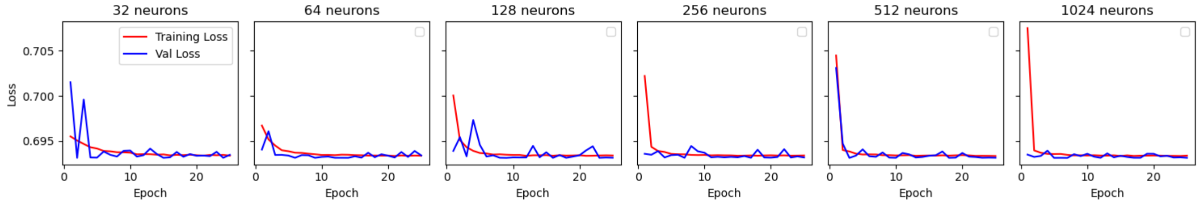
tanh:


Training and Validation Accuracy - Activation function: tanh, Learning Rate: 0.1, Number of layers: 10


Training and Validation Loss - Activation function: tanh, Learning Rate: 0.1, Number of layers: 10

sigmoid:


Training and Validation Accuracy - Activation function: sigmoid, Learning Rate: 0.1, Number of layers: 10

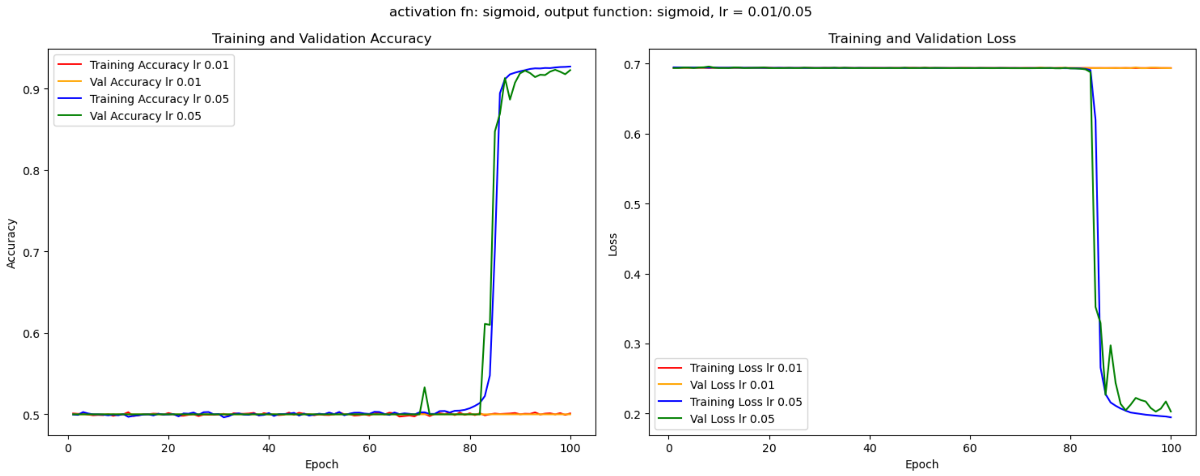

Training and Validation Loss - Activation function: sigmoid, Learning Rate: 0.1, Number of layers: 10

Discussion:

- Only plots for learning rate 0.1 are shown since there are already signs of overfitting. Plots for relu and tanh at this learning rate show that overfitting started from the first few epoch. Training accuracy keep increasing while validation accuracy remains unchanged.
- For relu, overfitting occurs at a faster rate than tanh and is highly unstable.
- All the runs for sigmoid as activation function has a very poor training and validation accuracy of 0.5 – equivalent to random guessing. Training and validation losses are also high at around 0.7. Likely it is not able converge.

2) Sigmoid Run

- to find out if the poor performance for sigmoid as activation function is a result of learning rate being too high.
- Simplify the model to 5 layers of 32 neurons with output activation function remaining as sigmoid and 100 epochs. Number of epochs is increased to balance the drastic drop in learning rate in order to capture convergence.
- 2 learning rates are tested: 0.01 and 0.05.



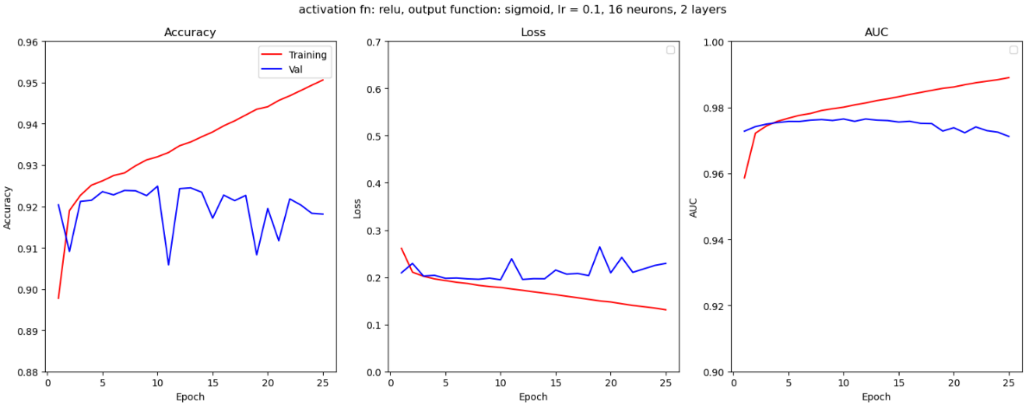
Discussion:

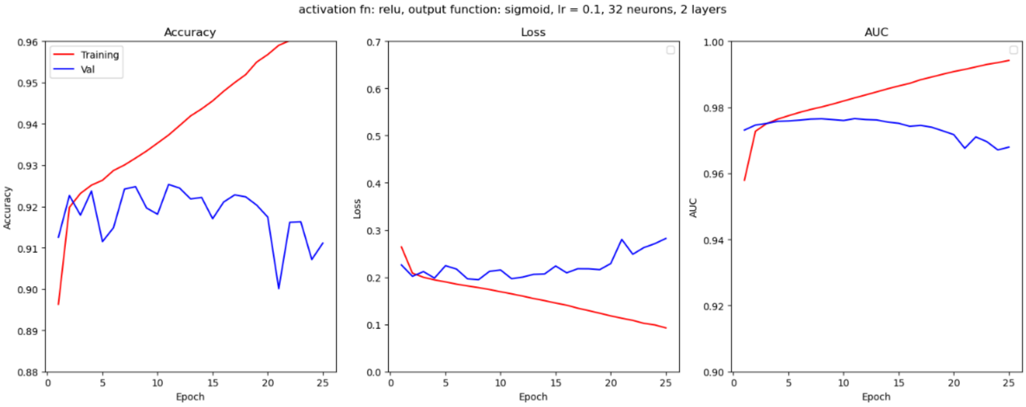
- Learning rate is indeed the reason for poor performance for sigmoid as activation function for the hidden layers in Preliminary Run. There is a steep increase in accuracy and sharp drop in loss for learning rate 0.05 at around epoch 85.
- Sigmoid failed to converge at learning rate 0.01 within 100 epochs.
- The complete run for learning rate 0.05 takes 1298s (21.6min) which is too long for a training dataset of 162451 data points. Sigmoid will therefore not be further considered as activation function for input and hidden layers.

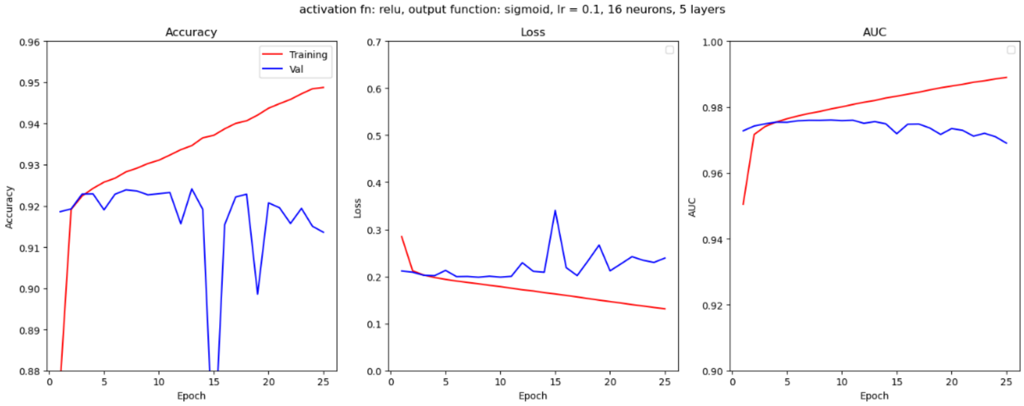
3) Layer Run

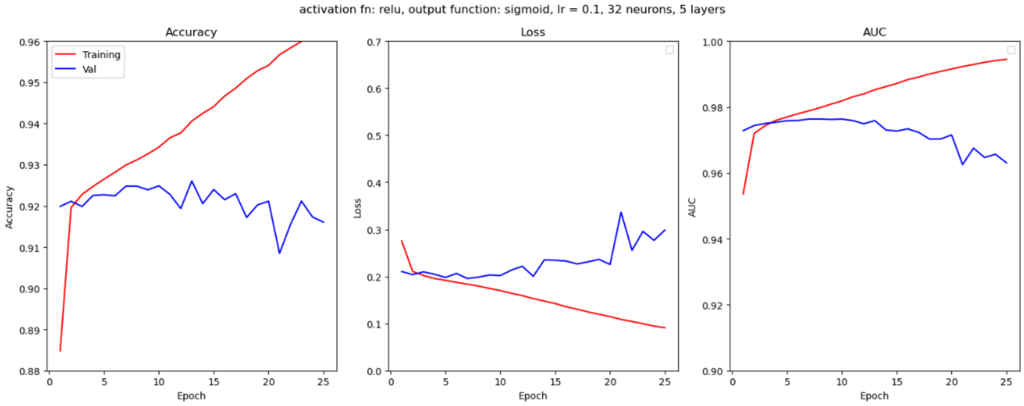
- To find number of layers and neurons to apply in relu and tanh as activation function.

Relu (Learning rate: 0.1) – 16 vs 32 neurons and 2 vs 5 layers:



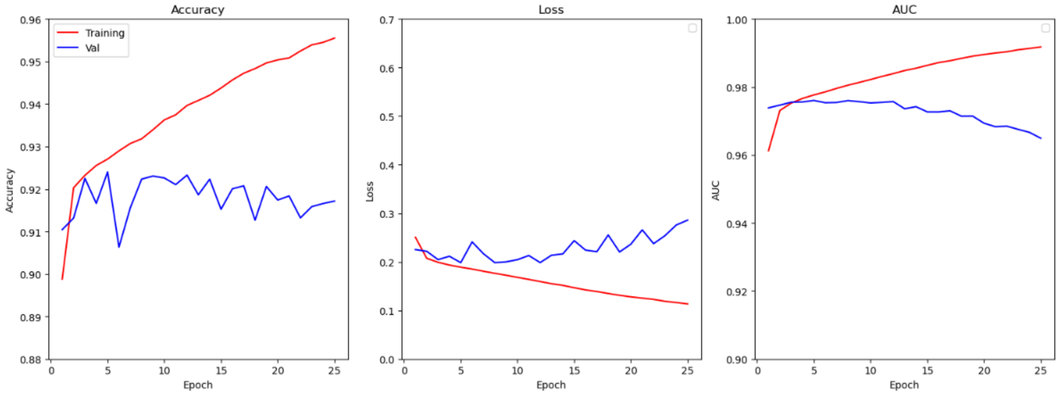




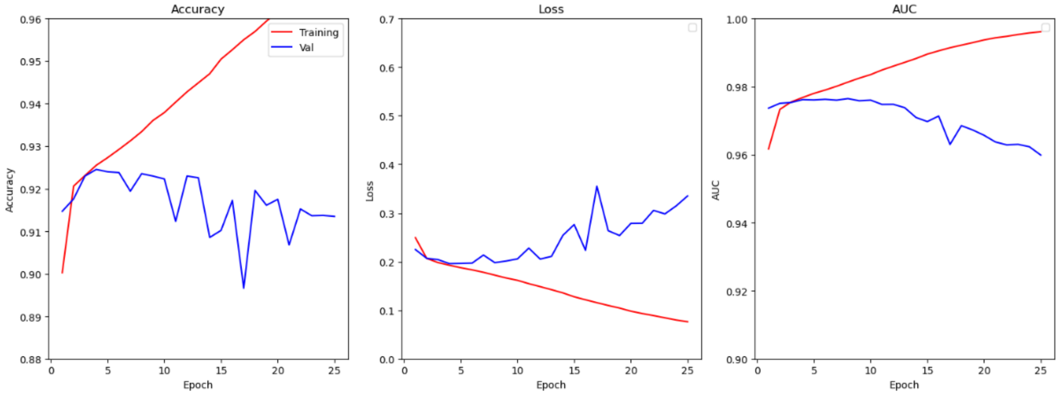


Relu (Learning rate: 0.4) – 16 vs 32 neurons and 2 vs 5 layers:

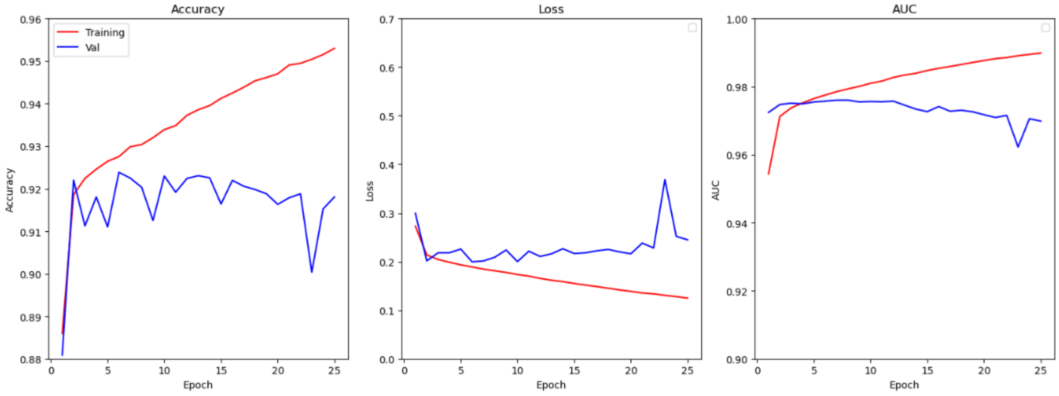
activation fn: relu, output function: sigmoid, lr = 0.4, 16 neurons, 2 layers



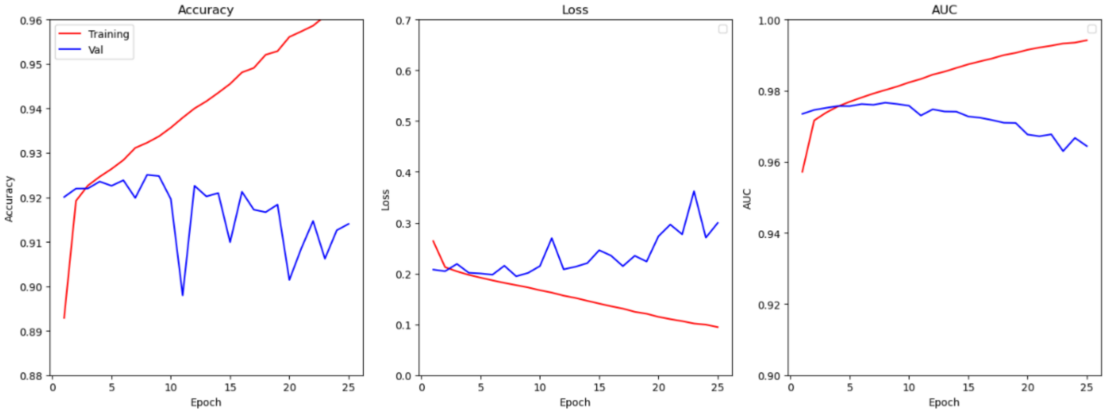
activation fn: relu, output function: sigmoid, lr = 0.4, 32 neurons, 2 layers



activation fn: relu, output function: sigmoid, lr = 0.4, 16 neurons, 5 layers

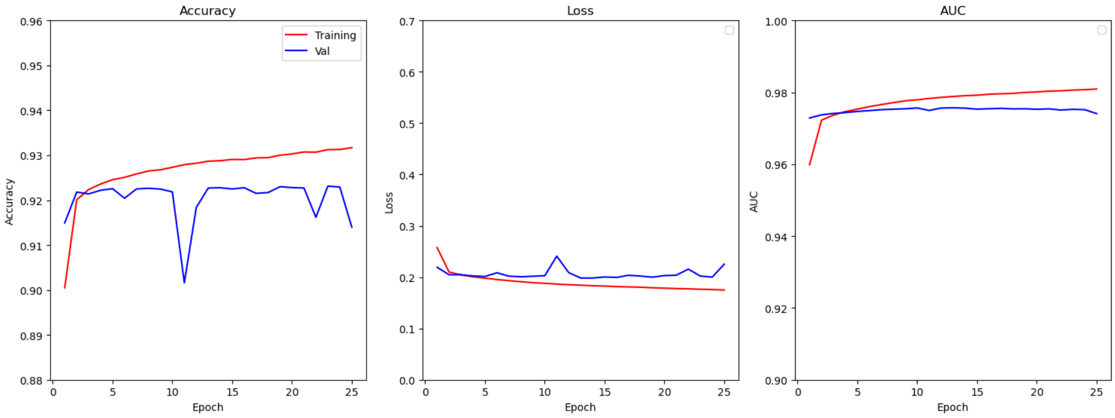


activation fn: relu, output function: sigmoid, lr = 0.4, 32 neurons, 5 layers

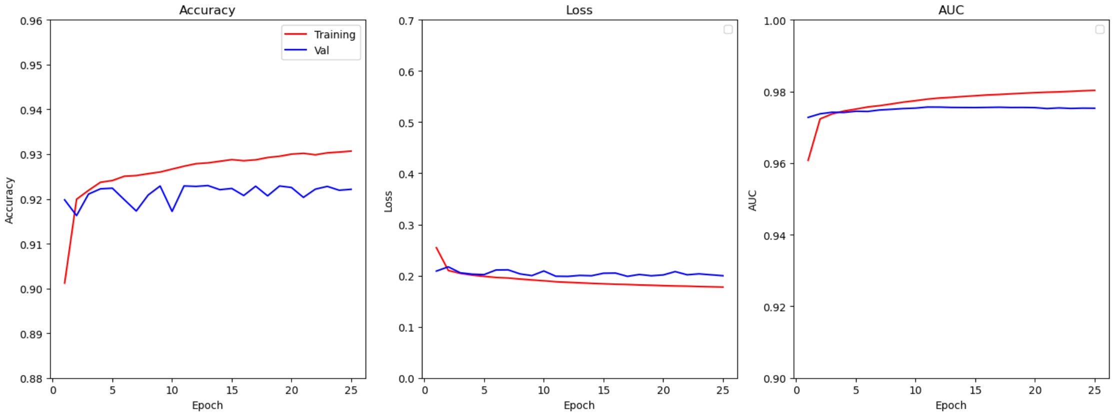


<u>tanh (Learning rate: 0.1) – 16 vs 32 neurons and 2 vs 5 layers:</u>

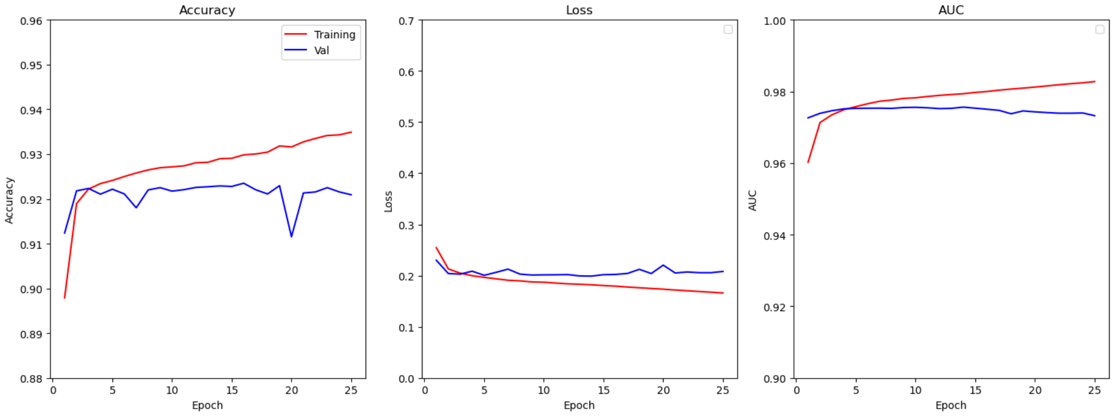
activation fn: tanh, output function: sigmoid, lr = 0.1, 16 neurons, 2 layers



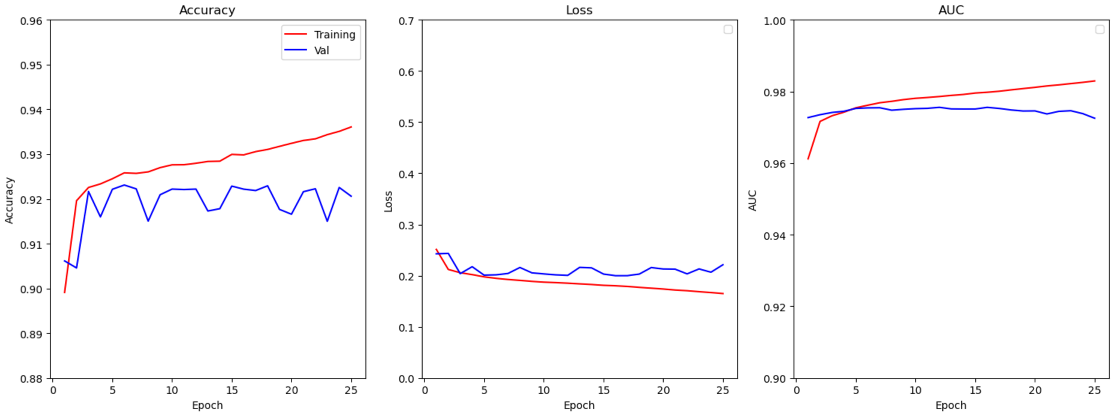
activation fn: tanh, output function: sigmoid, lr = 0.1, 32 neurons, 2 layers



activation fn: tanh, output function: sigmoid, lr = 0.1, 16 neurons, 5 layers

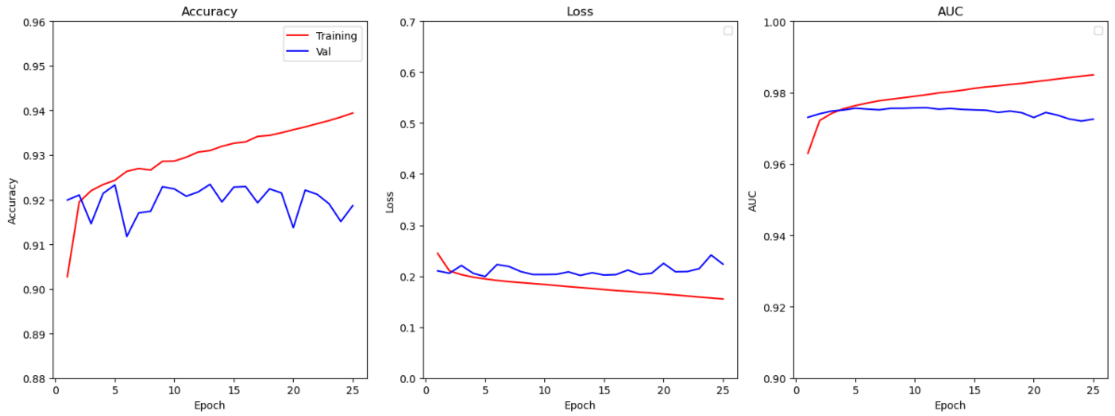


activation fn: tanh, output function: sigmoid, lr = 0.1, 32 neurons, 5 layers

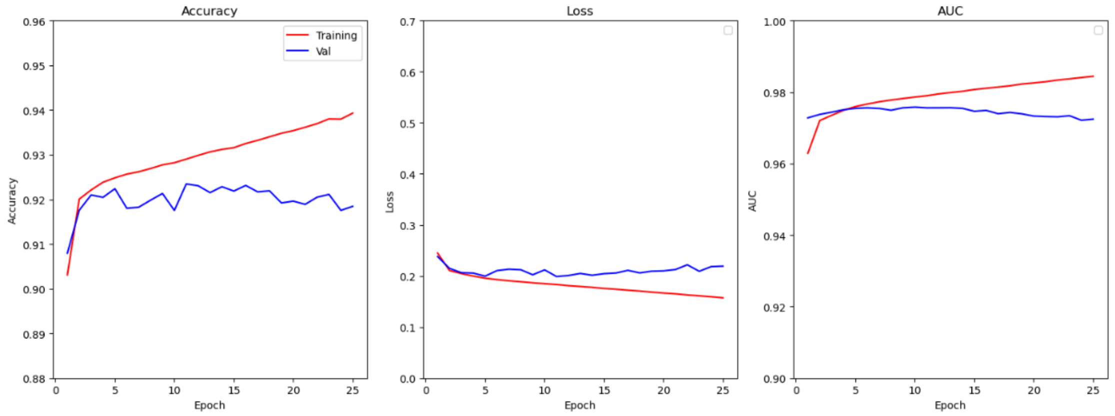


tanh (Learning rate: 0.4) – 16 vs 32 neurons and 2 vs 5 layers:

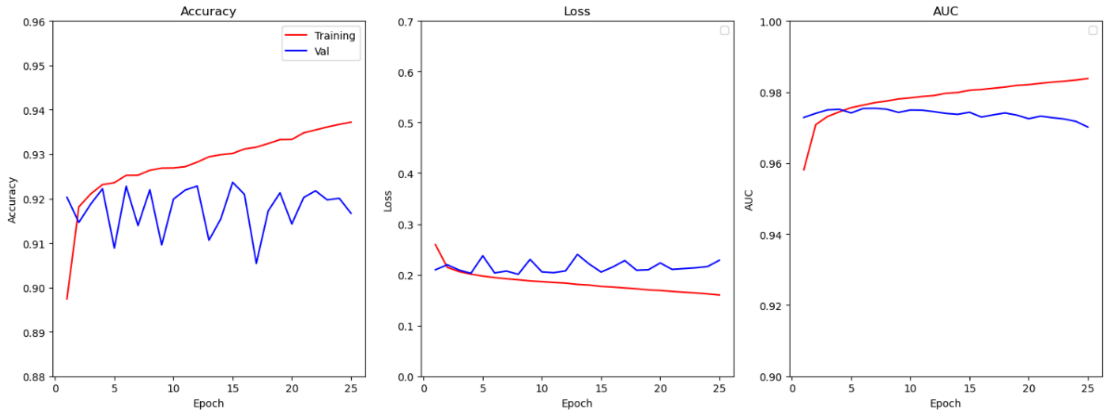
activation fn: tanh, output function: sigmoid, lr = 0.4, 16 neurons, 2 layers



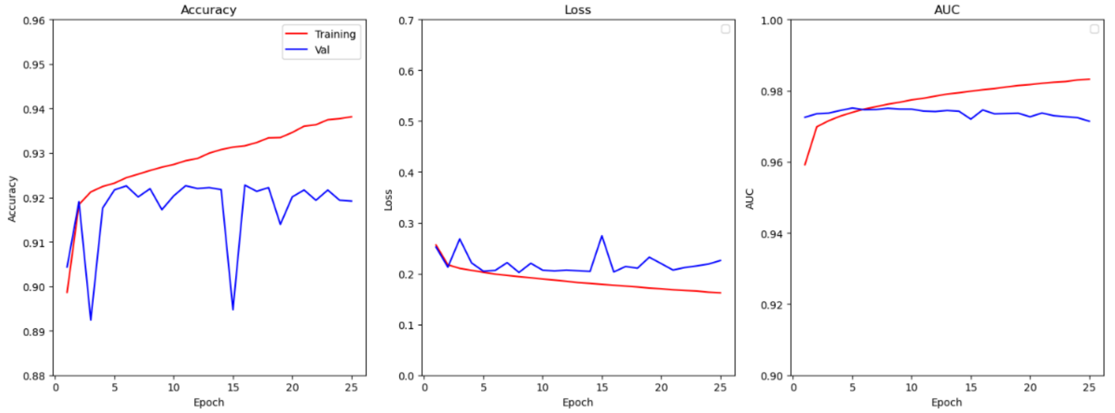
activation fn: tanh, output function: sigmoid, lr = 0.4, 32 neurons, 2 layers



activation fn: tanh, output function: sigmoid, lr = 0.4, 16 neurons, 5 layers



activation fn: tanh, output function: sigmoid, lr = 0.4, 32 neurons, 5 layers



Discussion:

- Relu:
    - 16 vs 32 neurons: higher number of neurons significantly increases rate of overfitting over each subsequent epoch.
    - 2 vs 5 layers: no observable difference in accuracy and loss.
    - Learning rate 0.1 vs 0.4: the model declines faster in performance for higher learning rate of 0.4.
    - AUC is around 97.5 – 97.8 for all 4 runs and start to decline after 10 epochs.

- tanh:
    - 16 vs 32 neurons: no observable difference in accuracy and loss.
    - 2 vs 5 layers: no observable difference in accuracy and loss.
    - Learning rate 0.1 vs 0.4: the model declines faster in performance for higher learning rate of 0.4.
    - AUC is comparable to relu runs. The difference between training and validation AUC is smallest for the lowest run parameter (16 neuron, learning rate 0.1) and the validation AUC holds stable throughout the 25 epochs. For all other runs, validation AUC gradually decreases after around 8 epochs.

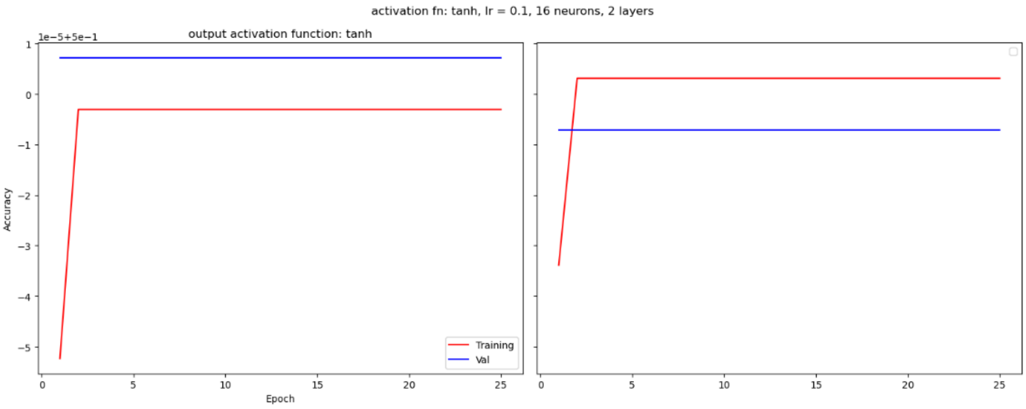
- Relu vs tanh:
    - tanh is more superior than relu as it is much more stable. Overfitting increases drastically for relu with the difference between training and validation increases to a large extent over the epochs.
    - Relu is good for epochs up to around 5 epochs after which performance is poor and unpredictable.
    - tanh, in comparison, is much more stable over each epoch. After 10 epochs, training and validation accuracy differs by scant 1% whereas for relu, difference is double at almost 2%. tanh is more robust in terms of accuracy, loss and AUC.
    - The best accuracy seems to stagnate at around 92% with AUC max at almost 98%. It may be because the dataset is too small to develop a robust model.
- It can be seen at learning rate 0.1, the model starts to overfit over the first few epochs, therefore lower learning rate will not be evaluated as overfitting gets worse with decreasing learning rate.
- Preliminary model: tanh as activation function, 2 layers, and learning rate 0.1. Number of neurons (16 vs 32) to be finalised after trying batching (to see if the performance can be more stabilised) and regularisation.

4) Output Run

- To compare output activation function. To see effect of using relu or tanh as output activation function.

Discussion:

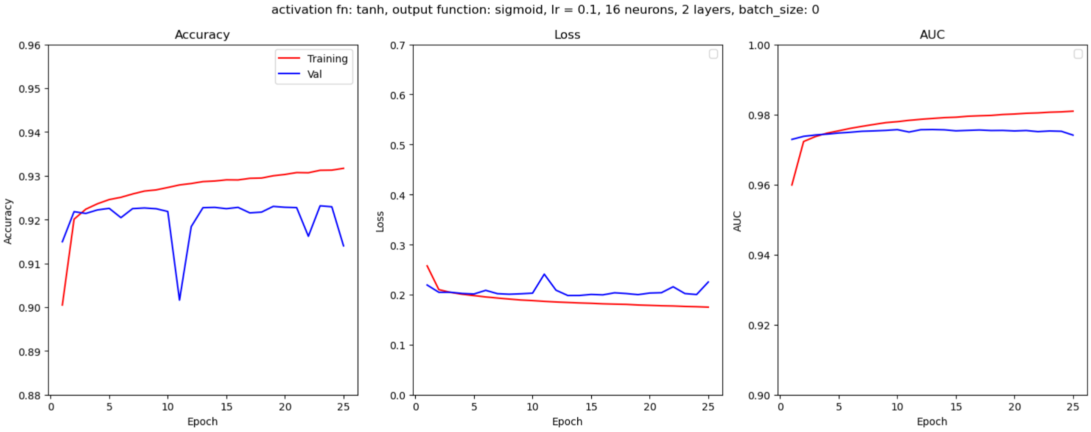
- Neither tanh nor relu performs well as an output activation function.

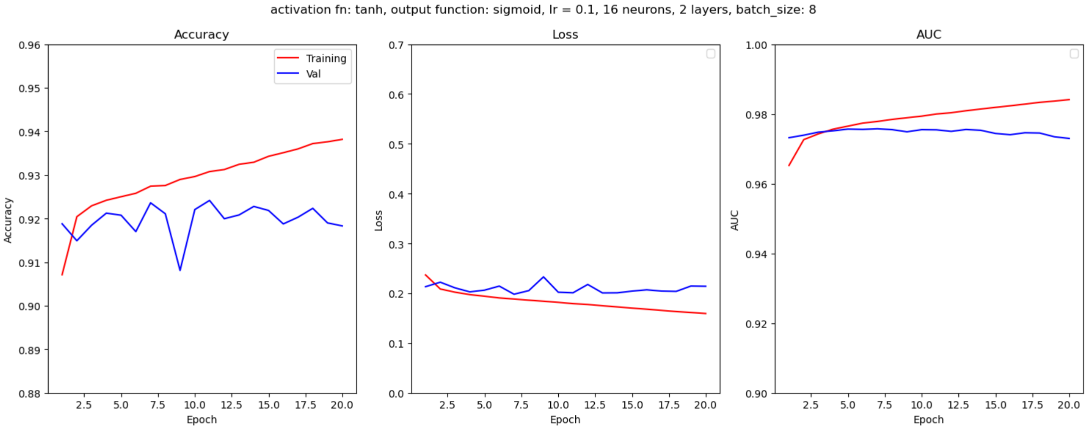


5) Batch run

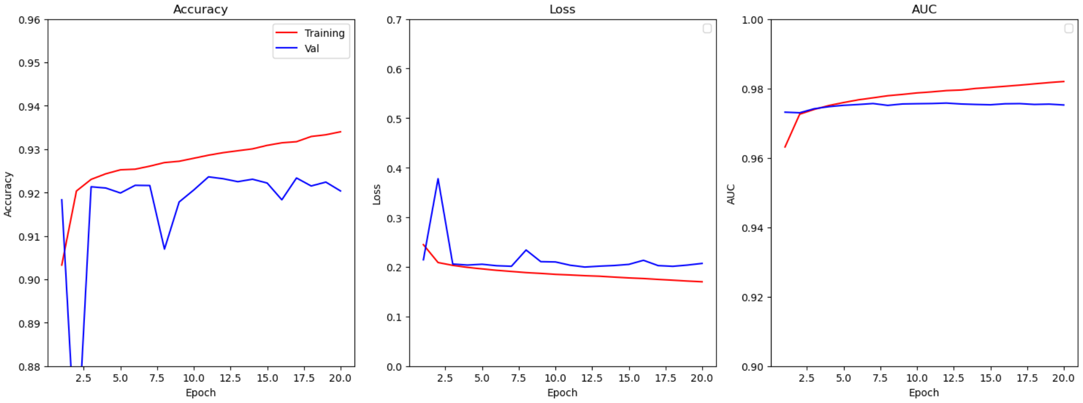
- To evaluate if batching the samples could help in performance.
- Comparisons will be made for 16 and 32 neurons. Layer with 32 neurons has higher potential to overfit hence it may show a more apparent effect in this experiment.

<u>16 neurons:</u>

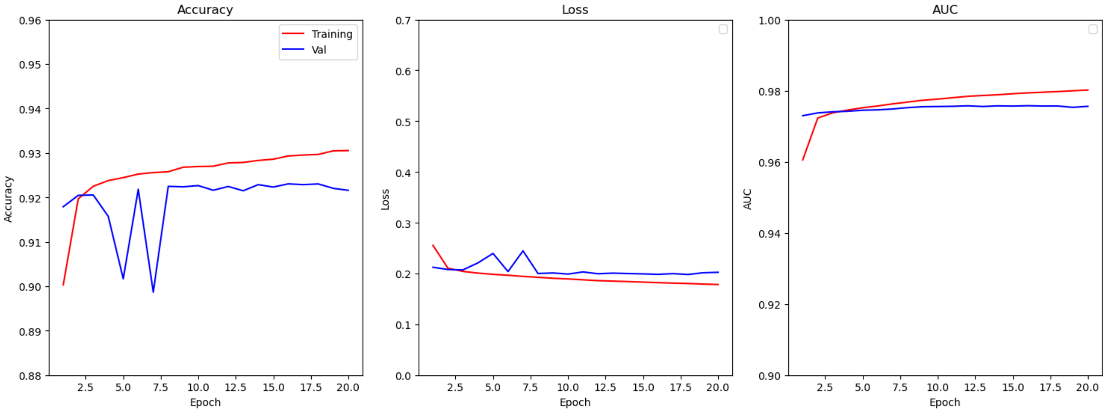




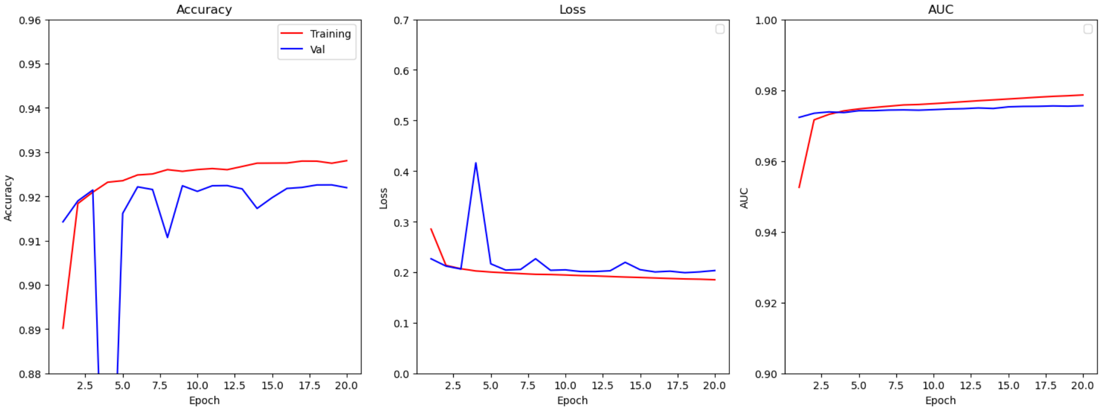
activation fn: tanh, output function: sigmoid, lr = 0.1, 16 neurons, 2 layers, batch_size: 16



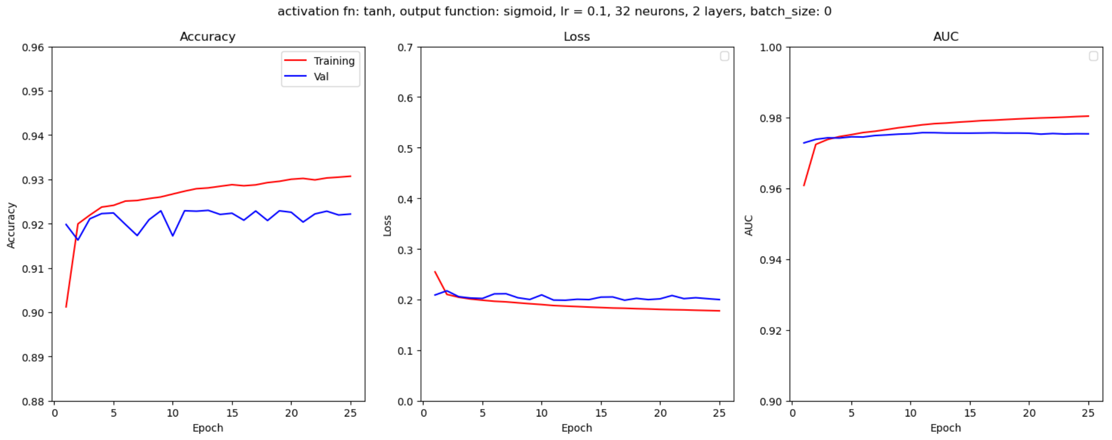
activation fn: tanh, output function: sigmoid, lr = 0.1, 16 neurons, 2 layers, batch_size: 32

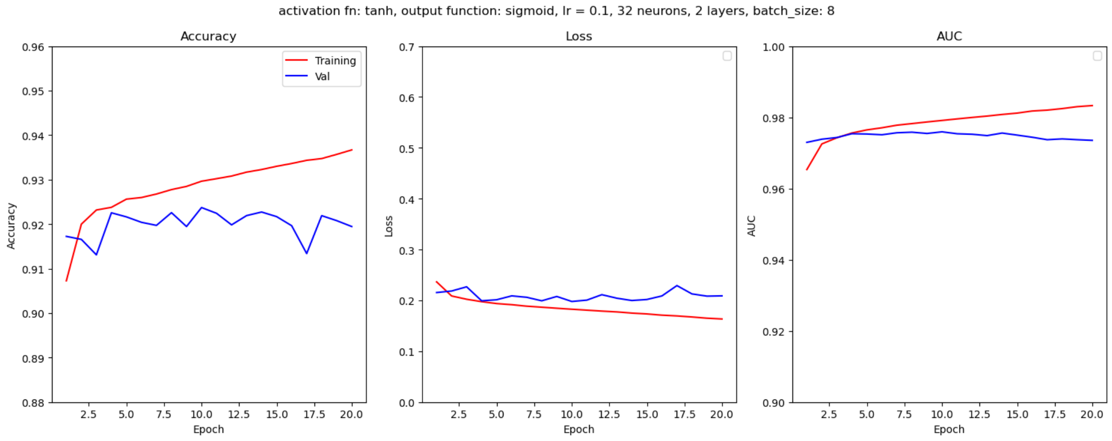


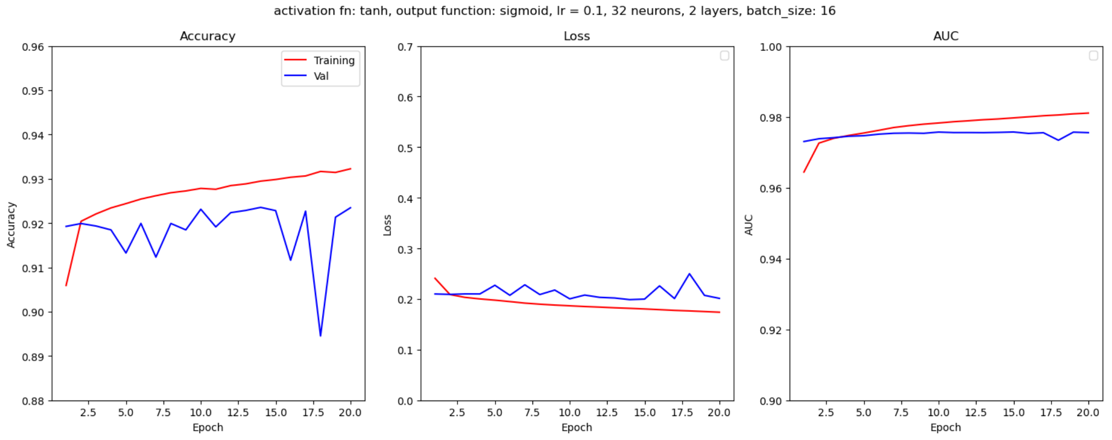
activation fn: tanh, output function: sigmoid, lr = 0.1, 16 neurons, 2 layers, batch_size: 64

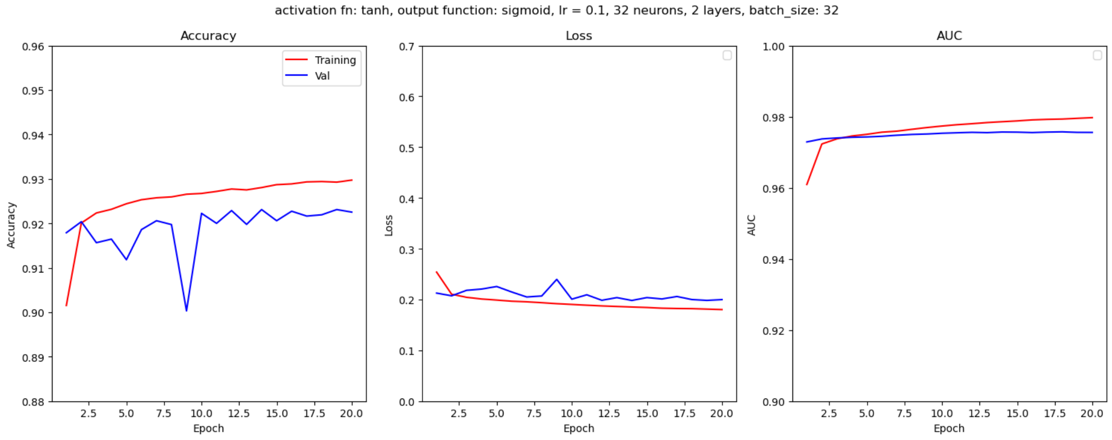


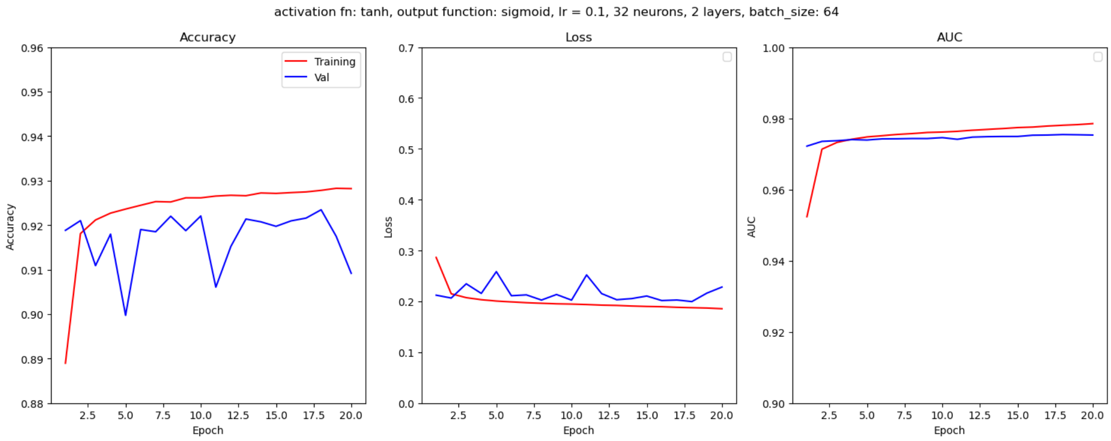
32 neurons:


activation fn: tanh, output function: sigmoid, lr = 0.1, 32 neurons, 2 layers, batch_size: 0
Accuracy
Training
Val
Accuracy
0.88 0.89 0.90 0.91 0.92 0.93 0.94 0.95 0.96
0 5 10 15 20 25
Epoch
Loss
Loss
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
0 5 10 15 20 25
Epoch
AUC
AUC
0.90 0.92 0.94 0.96 0.98 1.00
0 5 10 15 20 25
Epoch


activation fn: tanh, output function: sigmoid, lr = 0.1, 32 neurons, 2 layers, batch_size: 8
Accuracy
Training
Val
Accuracy
0.88 0.89 0.90 0.91 0.92 0.93 0.94 0.95 0.96
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch
Loss
Loss
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch
AUC
AUC
0.90 0.92 0.94 0.96 0.98 1.00
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch


activation fn: tanh, output function: sigmoid, lr = 0.1, 32 neurons, 2 layers, batch_size: 16
Accuracy
Training
Val
Accuracy
0.88 0.89 0.90 0.91 0.92 0.93 0.94 0.95 0.96
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch
Loss
Loss
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch
AUC
AUC
0.90 0.92 0.94 0.96 0.98 1.00
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch


activation fn: tanh, output function: sigmoid, lr = 0.1, 32 neurons, 2 layers, batch_size: 32
Accuracy
Training
Val
Accuracy
0.88 0.89 0.90 0.91 0.92 0.93 0.94 0.95 0.96
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch
Loss
Loss
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch
AUC
AUC
0.90 0.92 0.94 0.96 0.98 1.00
2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Epoch



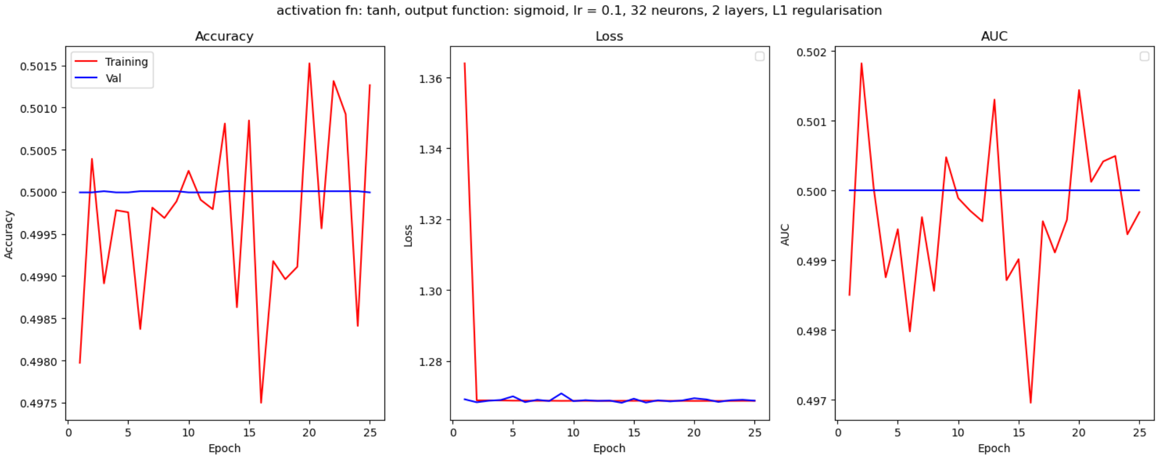
<u>Discussion:</u>

- The results are very similar among the runs. It appears that the larger the batch size, the more fluctuation in the metrics though very small. It may not be necessary to build in batch_size into the model however in the event that training dataset size is much larger, having a built-in batch_size would reduce the training time. The training time per epoch is about 1s between no batching and with batch size 16 for this dataset. The model will therefore adopt batch size 16.
- No observable difference between 16 and 32 neurons.

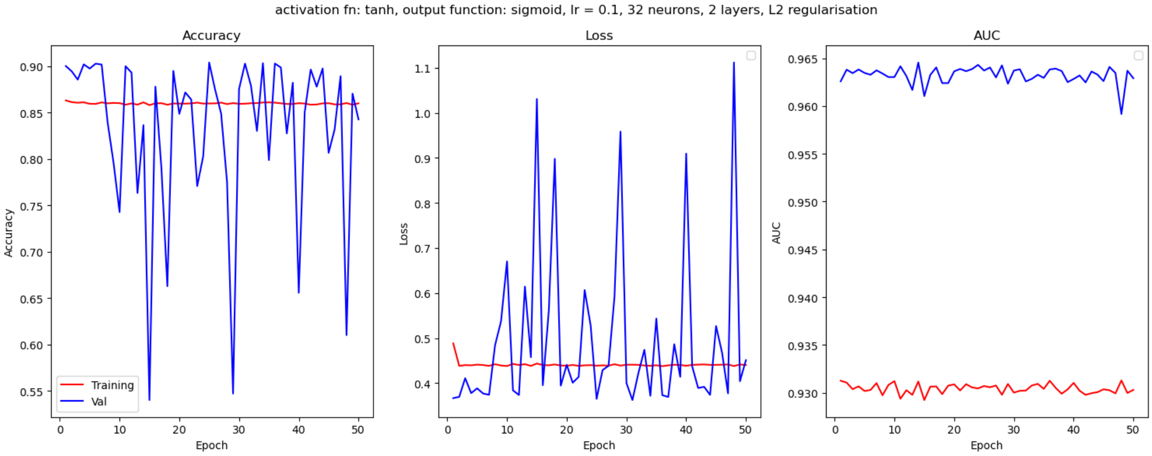
6) Regularisation Run

- to experiment on regularisation and if it improves the model.

L1:



L2:



Discussion:

- L1 and L2 regularisation do not work for this dataset.
- Other techniques to overcome overfitting is: implement dropout technique, increase dataset size or implement early stopping.
- There are only 2 hidden layers in this model hence applying dropout technique is not suitable.
- The dataset used here is a lot smaller than what is typically used to train deep learning. Since a larger dataset is not available, to make the model more robust, early stopping can be incorporated in this model.

7) Early Stopping Run

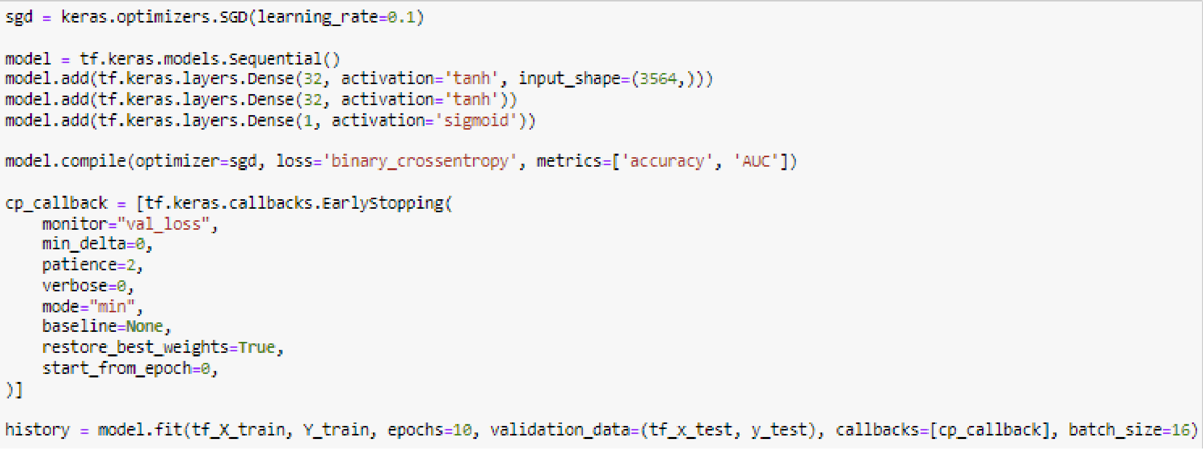
- To study effect of implementing early stopping on performance.

No of neuron	Patience	Epoch stopped at	Training acc	Val acc	Training loss	Val loss	Training AUC	Val AUC
16	0	2	0.9037	0.9201	0.2432	0.2101	0.9636	0.9730
16	1	5	0.9247	0.9105	0.1956	0.2249	0.9760	0.9752
16	2	8	0.9265	0.9216	0.1886	0.2013	0.9778	0.9756
16	3	9	0.9277	0.9210	0.1865	0.2091	0.9783	0.9755
32	0	2	0.9207	0.9201	0.2085	0.2072	0.9726	0.9738
32	1	2	0.9200	0.9002	0.2087	0.2629	0.9726	0.9734
32	2	7	0.9261	0.9210	0.1910	0.2049	0.9772	0.9754
32	3	10	0.9278	0.8967	0.1864	0.2518	0.9783	0.9741

Discussion:

- The number of epochs built in this model is 10. Earlier runs show that at epoch above 10, difference in accuracy between training and validation will defer by 1% and slowing widening. AUC starts to decline very gradually after 8 epochs, therefore 10 epochs is set as the limit.
- Data above shows that runs at
  - Training accuracy increases with increasing patience coinciding with increasing number of epochs). AUC also increases., so dies degree of overfitting.
  - patience values of 0 and 1 have lower accuracy and higher loss though AUC remains constant. At patience value of 3, validation accuracy starts to deteriorate as expected of overfitting. Therefore, patience should be set at 2. Setting at 3 would defeat the intent of early stopping to prevent overfitting as the model tends to run up to the max epoch setting.
- It is not immediately clear from this result if the model should be finalised at 16 or 32 neurons. However, bearing in mind that this dataset is much smaller than what is typically used for deep learning model training, it is more prudent to apply 32 neurons to prevent underfitting when the model is used on a larger dataset. The above runs using 32 neurons also have not shown any detrimental effects on the metrics.

- Final Model:

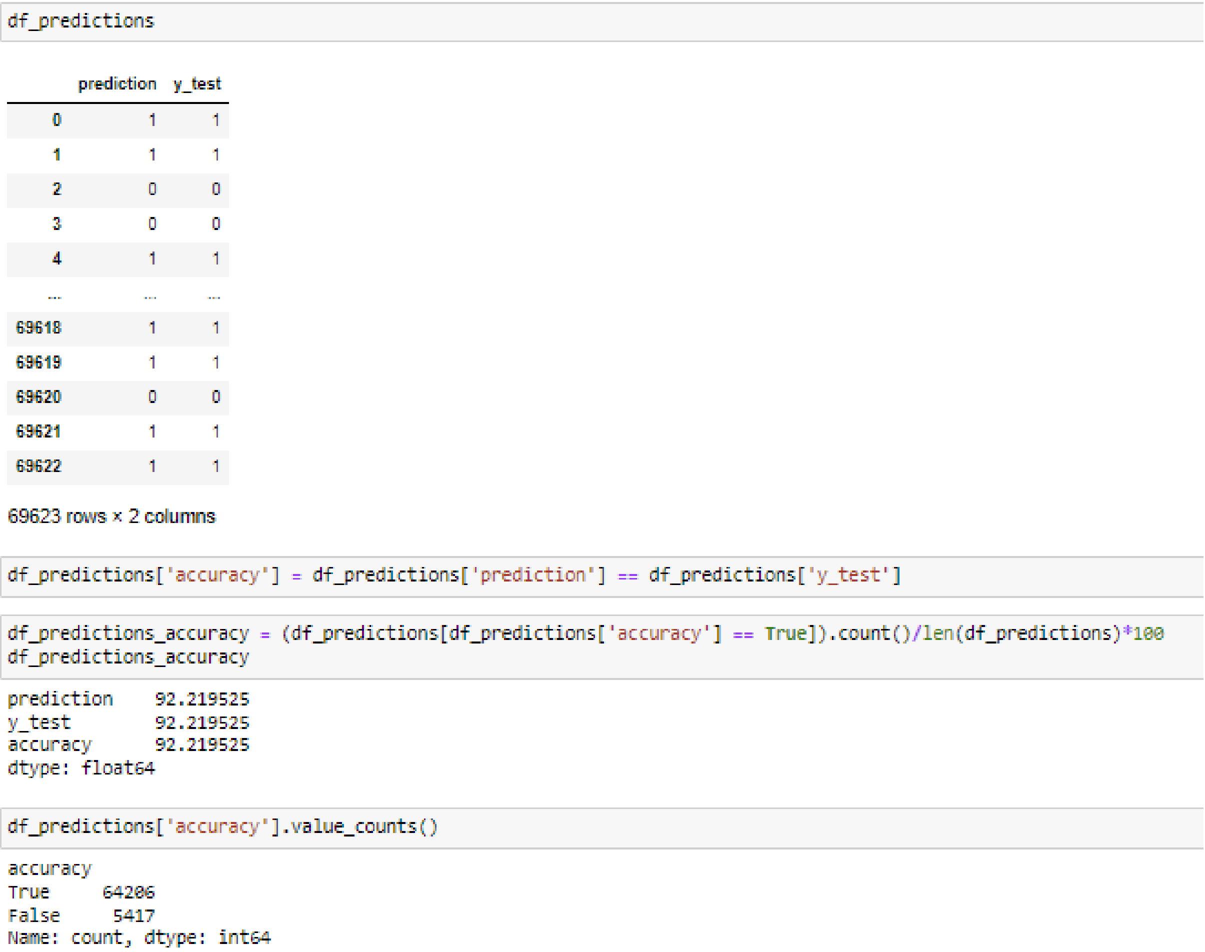


```
sgd = keras.optimizers.SGD(learning_rate=0.1)

model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Dense(32, activation='tanh', input_shape=(3564,)))
model.add(tf.keras.layers.Dense(32, activation='tanh'))
model.add(tf.keras.layers.Dense(1, activation='sigmoid'))

model.compile(optimizer=sgd, loss='binary_crossentropy', metrics=['accuracy', 'AUC'])

cp_callback = [tf.keras.callbacks.EarlyStopping(
    monitor="val_loss",
    min_delta=0,
    patience=2,
    verbose=0,
    mode="min",
    baseline=None,
    restore_best_weights=True,
    start_from_epoch=0,
)]

history = model.fit(tf_X_train, Y_train, epochs=10, validation_data=(tf_x_test, y_test), callbacks=[cp_callback], batch_size=16)
```

-

8) Prediction Check

- Prediction returns 92.2% accuracy, which is consistent with the experiments.



```
df_predictions
```

	prediction	y_test
0	1	1
1	1	1
2	0	0
3	0	0
4	1	1
...	...	...
69618	1	1
69619	1	1
69620	0	0
69621	1	1
69622	1	1

69623 rows × 2 columns

```
df_predictions['accuracy'] = df_predictions['prediction'] == df_predictions['y_test']
```

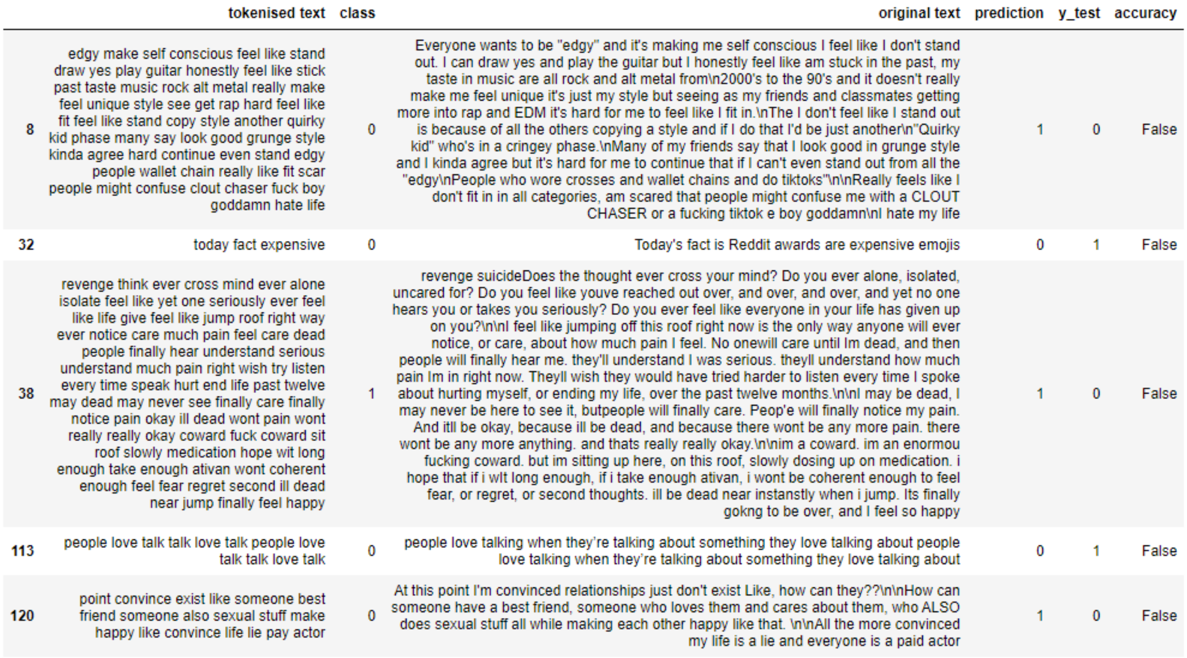
```
df_predictions_accuracy = (df_predictions[df_predictions['accuracy'] == True]).count()/len(df_predictions)*100
df_predictions_accuracy
```

```
prediction    92.219525
y_test        92.219525
accuracy      92.219525
dtype: float64
```

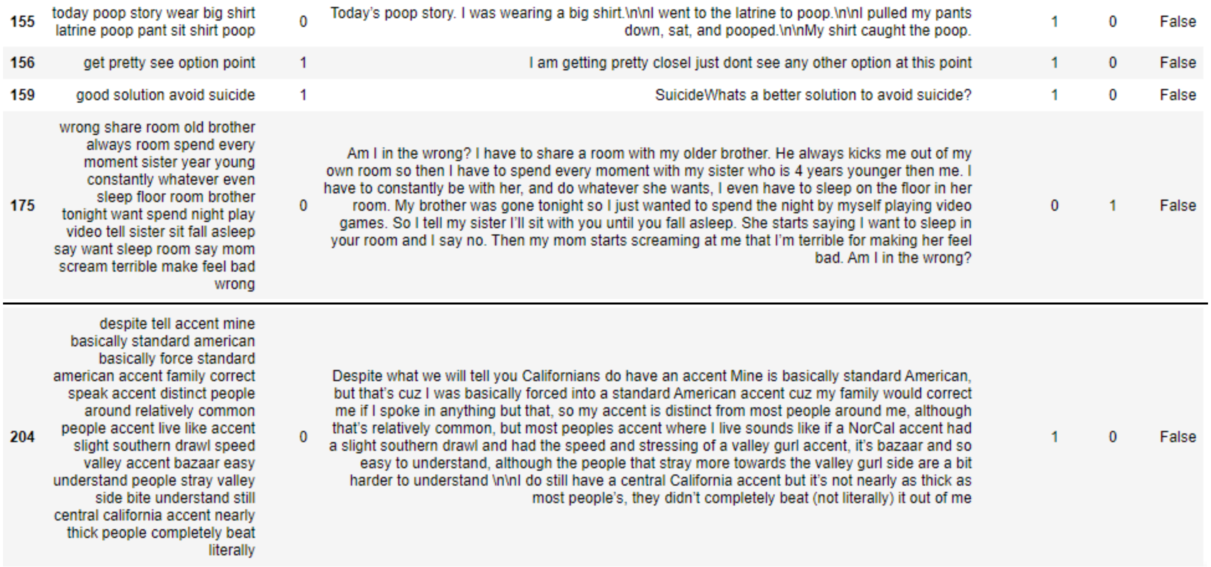
```
df_predictions['accuracy'].value_counts()
```

```
accuracy
True     64206
False     5417
Name: count, dtype: int64
```

9) Error Analysis and Conclusion



	tokenised text	class	original text	prediction	y_test	accuracy
8	edgy make self conscious feel like stand draw yes play guitar honestly feel like stick past taste music rock alt metal really make feel unique style see get rap hard feel like fit feel like stand copy style another quirky kid phase many say look good grunge style kinda agree hard continue even stand edgy people wallet chain really like fit scar people might confuse clout chaser fuck boy goddamn hate life	0	Everyone wants to be "edgy" and it's making me self conscious I feel like I don't stand out. I can draw yes and play the guitar but I honestly feel like am stuck in the past, my taste in music are all rock and alt metal from\n2000's to the 90's and it doesn't really make me feel unique it's just my style but seeing as my friends and classmates getting more into rap and EDM it's hard for me to feel like I fit in.\nThe I don't feel like I stand out is because of all the others copying a style and if I do that I'd be just another\n"Quirky kid" who's in a cringey phase.\nMany of my friends say that I look good in grunge style and I kinda agree but it's hard for me to continue that if I can't even stand out from all the "edgy\nPeople who wore crosses and wallet chains and do tiktoks"\n\nReally feels like I don't fit in in all categories, am scared that people might confuse me with a CLOUT CHASER or a fucking tiktok e boy goddamn\nI hate my life	1	0	False
32	today fact expensive	0	Today's fact is Reddit awards are expensive emojis	0	1	False
38	revenge think ever cross mind ever alone isolate feel like yet one seriously ever feel like life give feel like jump roof right way ever notice care much pain feel care dead people finally hear understand serious understand much pain right wish try listen every time speak hurt end life past twelve may dead may never see finally care finally notice pain okay ill dead wont pain wont really really okay coward fuck coward sit roof slowly medication hope wit long enough take enough ativan wont coherent enough feel fear regret second ill dead near jump finally feel happy	1	revenge suicideDoes the thought ever cross your mind? Do you ever alone, isolated, uncared for? Do you feel like youve reached out over, and over, and over, and yet no one hears you or takes you seriously? Do you ever feel like everyone in your life has given up on you?\n\nI feel like jumping off this roof right now is the only way anyone will ever notice, or care, about how much pain I feel. No onewill care until Im dead, and then people will finally hear me. they'll understand I was serious. theyll understand how much pain Im in right now. Theyll wish they would have tried harder to listen every time I spoke about hurting myself, or ending my life, over the past twelve months.\n\nI may be dead, I may never be here to see it, butpeople will finally care. Peop'e will finally notice my pain. And itll be okay, because ill be dead, and because there wont be any more pain. there wont be any more anything. and thats really really okay.\n\nim a coward. im an enormou fucking coward. but im sitting up here, on this roof, slowly dosing up on medication. i hope that if i wlt long enough, if i take enough ativan, i wont be coherent enough to feel fear, or regret, or second thoughts. ill be dead near instanstly when i jump. Its finally gokng to be over, and I feel so happy	1	0	False
113	people love talk talk love talk people love talk talk love talk	0	people love talking when they're talking about something they love talking about people love talking when they're talking about something they love talking about	0	1	False
120	point convince exist like someone best friend someone also sexual stuff make happy like convince life lie pay actor	0	At this point I'm convinced relationships just don't exist Like, how can they??\n\nHow can someone have a best friend, someone who loves them and cares about them, who ALSO does sexual stuff all while making each other happy like that. \n\nAll the more convinced my life is a lie and everyone is a paid actor	1	0	False



155	today poop story wear big shirt latrine poop pant sit shirt poop	0	Today's poop story. I was wearing a big shirt.\n\nI went to the latrine to poop.\n\nI pulled my pants down, sat, and pooped.\n\nMy shirt caught the poop.	1	0	False
156	get pretty see option point	1	I am getting pretty closeI just dont see any other option at this point	1	0	False
159	good solution avoid suicide	1	SuicideWhats a better solution to avoid suicide?	1	0	False
175	wrong share room old brother always room spend every moment sister year young constantly whatever even sleep floor room brother tonight want spend night play video tell sister sit fall asleep say want sleep room say mom scream terrible make feel bad wrong	0	Am I in the wrong? I have to share a room with my older brother. He always kicks me out of my own room so then I have to spend every moment with my sister who is 4 years younger then me. I have to constantly be with her, and do whatever she wants, I even have to sleep on the floor in her room. My brother was gone tonight so I just wanted to spend the night by myself playing video games. So I tell my sister I'll sit with you until you fall asleep. She starts saying I want to sleep in your room and I say no. Then my mom starts screaming at me that I'm terrible for making her feel bad. Am I in the wrong?	0	1	False
204	despite tell accent mine basically standard american basically force standard american accent family correct speak accent distinct people around relatively common people accent live like accent slight southern drawl speed valley accent bazaar easy understand people stray valley side bite understand still central california accent nearly thick people completely beat literally	0	Despite what we will tell you Californians do have an accent Mine is basically standard American, but that's cuz I was basically forced into a standard American accent cuz my family would correct me if I spoke in anything but that, so my accent is distinct from most people around me, although that's relatively common, but most peoples accent where I live sounds like if a NorCal accent had a slight southern drawl and had the speed and stressing of a valley gurl accent, it's bazaar and so easy to understand, although the people that stray more towards the valley gurl side are a bit harder to understand \n\nI do still have a central California accent but it's not nearly as thick as most people's, they didn't completely beat (not literally) it out of me	1	0	False

- Out of the rows above, the original label is incorrect; the model is correct (row 32, 38, 113 and 175).

The model built has an accuracy of 92% maximum and is able to reach AUC of above 97%. It is likely that the performance is limited by the dataset size – not enough data to optimise learning.

From the error analysis, the model is picking out words like 'hate', 'suicide', 'stressing', 'beat' which could be more prominent in the 'Suicide' class than the 'Non-Suicide' class. If the dataset is larger, the proportion of word vocab would change and the vocab difference between the 2 classes is likely to be better differentiated. However, it should also be noted that adjustments to the model will be necessary with a larger dataset in terms of number of layers and number of neurons otherwise the model might underfit.

Additionally, other deep learning networks could also be considered to better analyse the text base on context instead of word count.

Task 2

1) Differences between classical machine learning and neural network/deep learning.

Classical Machine Learning Algorithms	Neural Networks and Deep Learning
require engineer to do feature engineering	automatic feature engineering
manually choose features and algorithm, check output and adjust the algorithm thus require human intervention	features are extracted automatically, and the algorithm learns from its own errors thus does not require human intervention and has the ability to handle large and complex data
recognize patterns in the data and make predictions with a comparatively simpler structure	use a complex structure of algorithms modelled on the human brains using a layered structure of algorithms called an artificial neural network (ANN)
able to perform with small datasets (minimum thousands of data points)	require large dataset to learn well (typically millions of data points and above)
take relatively less time to train	can take a lot of time to train
can take as short as seconds to train hence do not require high computing power (CPU is usually sufficient)	need substantial computing power (GPU and TPU)

2) Strengths and weakness between classical machine learning and neural network/deep learning.

Strengths

Classical Machine Learning Algorithms	Neural Networks and Deep Learning
more explainable	scalable, making ANNs suitable for handling large and complex dataset making them more applicable to real life applications.
accessible as less resource-demanding	ability to handle complex data
can handle a wide range of problems	non-linear modelling capabilities
can work with labelled or unlabelled data	can handle a wide range of problems
Flexible - Uses various algorithms for training	can uncover hidden patterns and insights
transparent	can capture complex non-linear relationships between input variables and output predictions
more interpretable	robustness to noisy or incomplete data
simple and easy to implement	ANNs can automatically extract relevant features
computationally more efficient than deep learning	adaptability and learning capabilities
some algorithms are robust to outliers	can handle high-dimensional data

Weaknesses

Classical Machine Learning Algorithms	**Neural Networks and Deep Learning**
requires more experimentation and development to design model to capture non-linear datasets	black box nature can hinder interpretability
accuracies affected by noisy data and require human intervention to remove noise	sensitivity to input data quality and preprocessing
Scalability depends on the algorithm used	need for large amounts of labelled training data
assumptions of each algorithm results in some weaknesses and limitations. E.g. Naïve Bayes assume complete features to be independent of one another.	computationally intensive and resource-consuming
some algorithms can be affected by collinearity e.g. linear regression	lack of transparency in decision-making
more time required to develop model as some algorithms do not work for non-linear datasets especially when it is not known in what way the dataset is not linear.	potential overfitting without proper regularization
overfitting and underfitting can occur. Require human intervention to tune hyperparameters	complexity and difficulty in model tuning
can be affected by imbalanced dataset depending on the algorithm. Algorithms like Decision Tree are rather immuned to imbalanced data.	prediction affected if data is imbalanced - the neural network may be biased towards the majority class and perform poorly on the minority class
	difficulty in explaining results to stakeholders
	ethical considerations in sensitive decision-making
	noisy data can affect predictions