EMORY
UNIVERSITY

Department of Computer Science & Informatics

Masoud Nateghi

CS534 HW3

Professor Joyce Ho

1.(a) Why would you potentially be interested in optimizing for F2 score compared to F1 score in the context of finding individuals that are likely to default?

We'll begin by examining the F1-score formula, as shown below:

$$F1 = \frac{TP}{TP + 0.5(FN + FP)}$$

This formula highlights that both recall and precision are equally weighted. However, in our specific dataset, accurately identifying all individuals who might default on their loans (referred to as "positives" or P) is of paramount importance. In simpler terms, erroneously classifying a potential default case as normal (referred to as "negatives" or N) carries a more significant penalty for us compared to misclassifying a normal case as a default.

Consequently, in this context, it becomes crucial to prioritize achieving a high recall (while simultaneously maintaining a reasonably high precision). In other words, recall holds more significance than precision. Hence, our primary focus will be on optimizing for the F2-score, as indicated by the following formula:

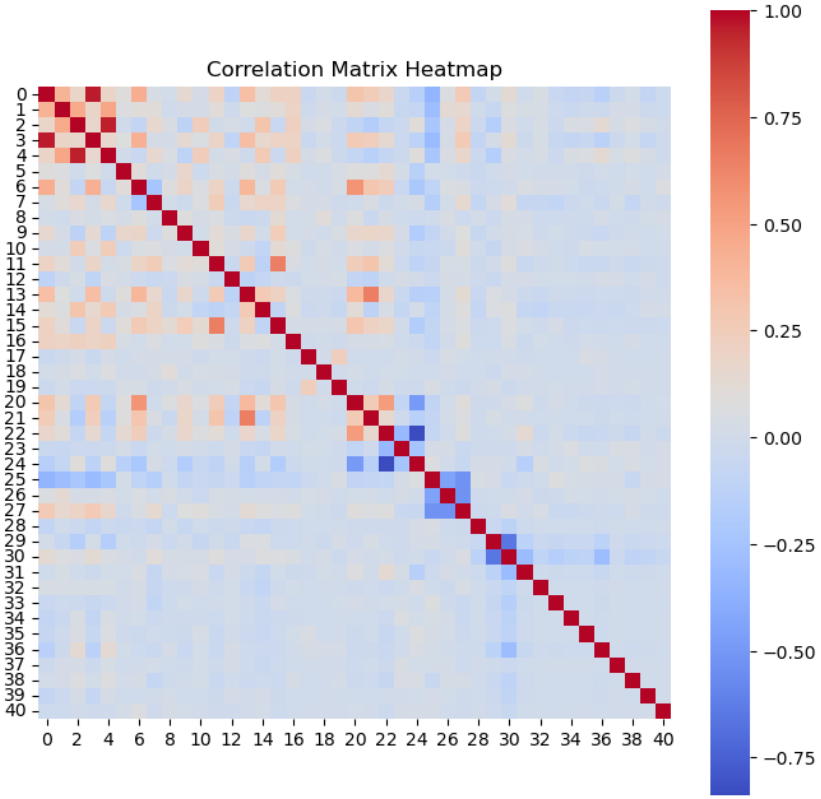
$$F2 = \frac{TP}{TP + 0.2FP + 0.8FN}$$

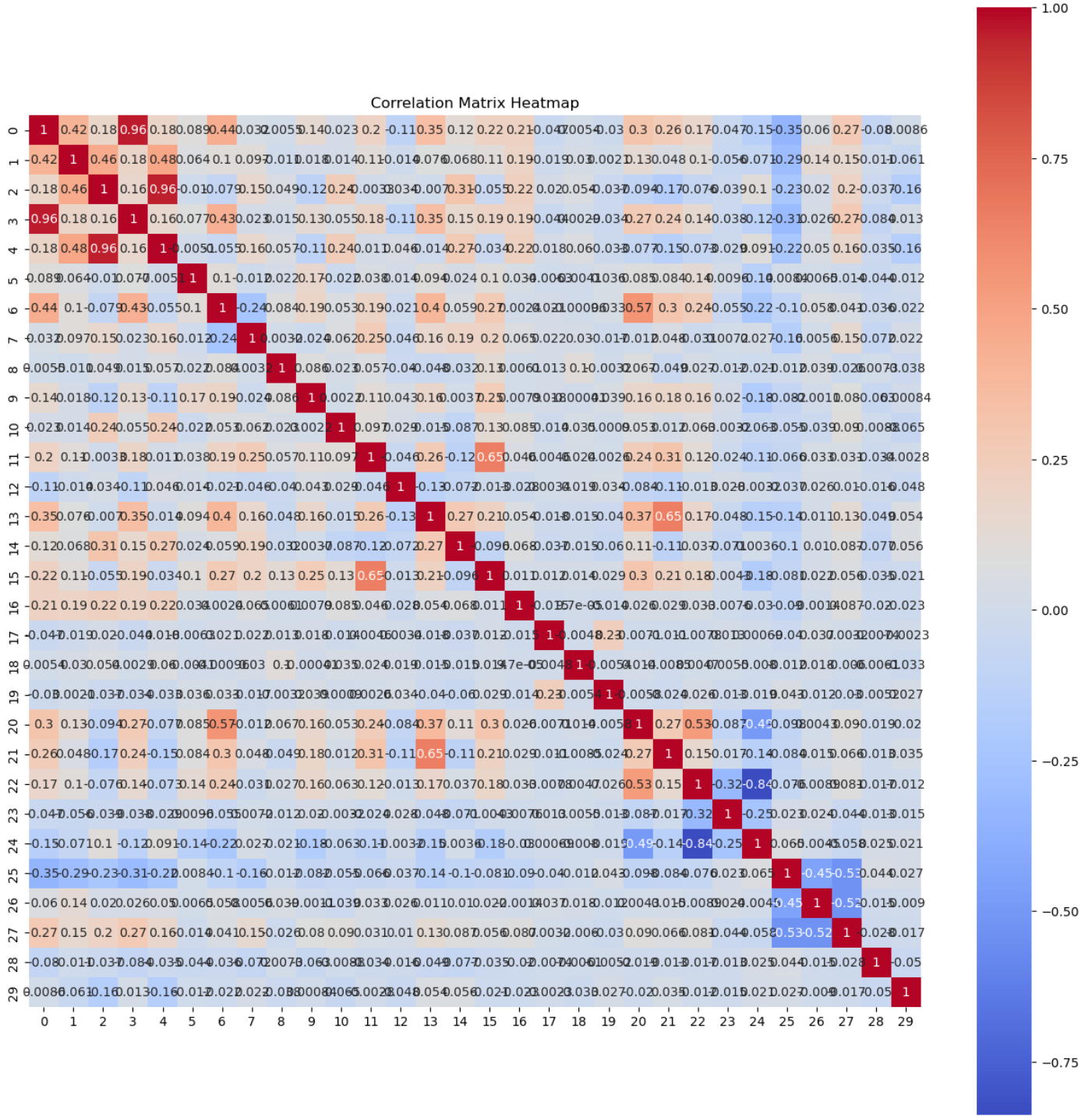
1.(b) How will you partition the loan data to assess the model performance and choose hyperparameter? Justify your choice and write code to partition the data accordingly.

I first split data into train and test dataset with 20% of the whole data to be selected randomly for test dataset. For tuning hyperparameters, I used grid search where I have used 5-fold cross validation for that purpose.

1.(f) Using 1c, 1d, 1e, perform feature selection using the three different methods independently. For each one, justify your selection (e.g., what you are using as thresholds for discarding highly correlated variables and the metric or what you'll choose as the top k ranked features).

In the feature selection method, we begin by calculating the correlations between various features. To do this, we utilize a custom-developed function to compute these correlations, followed by generating a heatmap of the correlation matrix. In the report, we will present an example of a Pearson correlation matrix heatmap, while Spearman and Kendall correlations can be found in the Jupyter notebook file, with similar values. For the purpose of this analysis, we will set the threshold at 0.5 and proceed to eliminate one of the features in any pair that exhibits a correlation greater than 0.5. We also note that we should not use any information from test data in feature selection. So, we first split our data into train and test, then we will compute correlations.





In the rank correlation function, we printed the correlations and retained features with a correlation greater than 0.05 with the output. This implies that we should roughly retain the top 15 features with the highest correlations, denoted as (k=15).

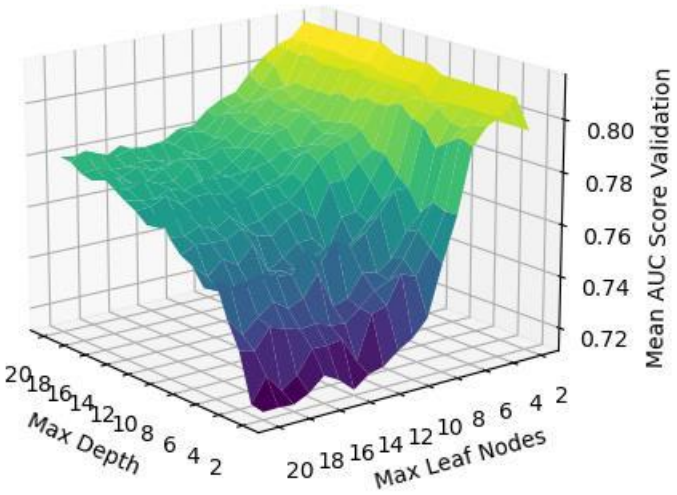
```
[ 0.03661516  0.15607183  0.26924039  0.01968187  0.26459172 -0.05273085
 -0.11821286  0.16757451  0.03439004 -0.06193702  0.0909425  -0.0031393
  0.02183062 -0.03208417  0.10968028 -0.04962906  0.44768437 -0.0052745
  0.02011764 -0.01610218 -0.07174047 -0.0731952  -0.04456662 -0.02987228
  0.06270531 -0.079648   -0.00323766  0.07906505 -0.01267682 -0.0102005
  0.04017201 -0.0812016  -0.01747612 -0.02104784  0.02746644 -0.01482373
  0.01349266 -0.01931669  0.05493955  0.00557885 -0.01006764]
```

Additionally, in the rank mutual information function, we exclusively selected features with non-zero mutual information. This led to the keeping of approximately 20 features (k=20).

```
[0.         0.00526239 0.04922223 0.         0.03072838 0.00211287
 0.01167697 0.01186993 0.         0.         0.00301315 0.
 0.         0.0149067  0.         0.18611604 0.01001944 0.
 0.         0.         0.00313129 0.00524096 0.00636579 0.
 0.00523595 0.00596176 0.         0.         0.00827004 0.02908555
 0.01255494 0.         0.         0.         0.         0.02055246
 0.         0.         0.01080535 0.01573766]
```

2.(b) What is the best choice of the two parameters based on 2a? Plot the "validation" AUC (based on 1b) on a single plot (either do a 3D plot or use the size of the point to encode the AUC) as a function of the two parameters.

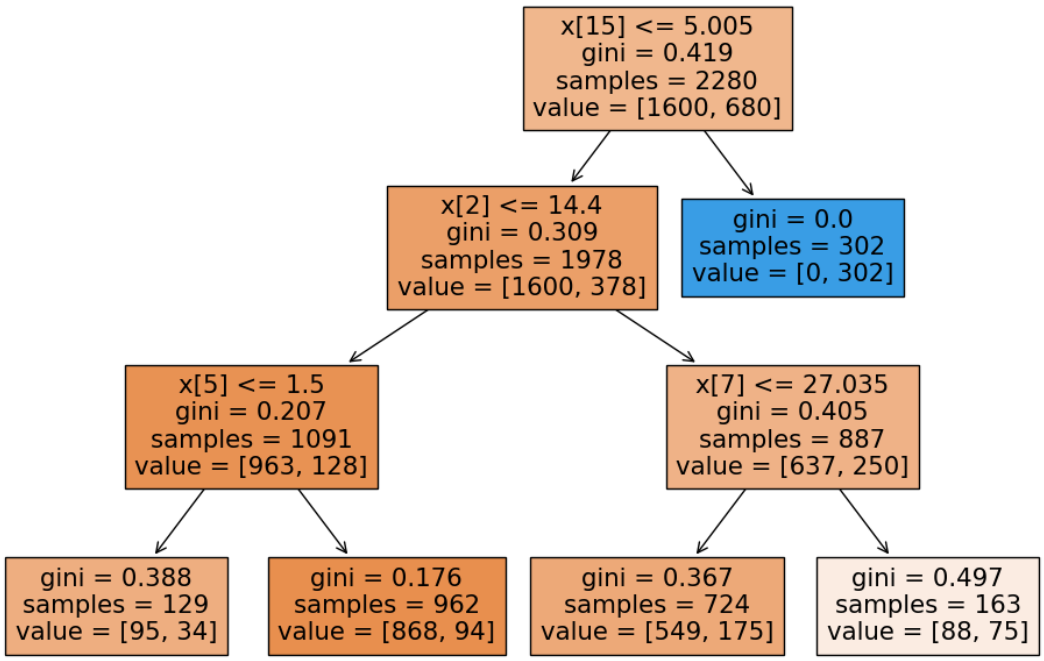
We obtained the optimal maximum depth and minimum number of samples in each leaf to be 3 and 13 respectively.



2.(c) Re-train a decision tree on the entire training data using the optimal max depth and the minimum number of samples from 2b. What are the AUC, F1, and F2 score on the test set?

	AUC	F1-score	F2-score
Train	0.7220	0.6150	0.4996
Test	0.7382	0.6454	0.5321

2.(d) Create a visualization of the top 3 levels of your decision tree from 2c.



2.(e) Analyze the effects of three different filtering methods, 1c, 1d, 1e, independently on the decision tree by determining the optimal parameters using 2a and then re-training the data on the entire training data in 2c and report the AUC, F1, and F2 score in a table.

Initially, we compute correlations between the various features and subsequently eliminate one of the two highly correlated features within each pair, where the correlation coefficient surpasses 0.5, either positively or negatively. Notably, the pairs (0, 3), (2, 4), (6, 19), (10, 14), (12, 20), (19, 21), (21, 23), (19, 23), (24, 26), (25, 26), and (28, 29) exhibit pronounced correlations. Thus, we have opted to remove features 0, 2, 6, 10, 12, 19, 21, 26, and 28. The subsequent table presents the outcomes derived from a decision tree model, where the maximum depth is set to 3, and the minimum samples in a leaf are 2.

	AUC	F1-score	F2-score
Train	0.7220	0.6150	0.4996
Test	0.7382	0.6454	0.5321

Now we use the rank_corr feature selection (k = 15), (best model: max depth = 3 and min samples leaf = 2).

	AUC	F1-score	F2-score
Train	0.7260	0.6224	0.5105
Test	0.7319	0.6454	0.5321

Finally, we use the rank_mutual function (k = 20), (best model: max depth = 3 and min samples leaf = 2).

	AUC	F1-score	F2-score
Train	0.7220	0.6150	0.4996
Test	0.7382	0.6454	0.5321

2.(f) Comment on the effect of feature selection for decision trees based on your results from 2c and 2e.

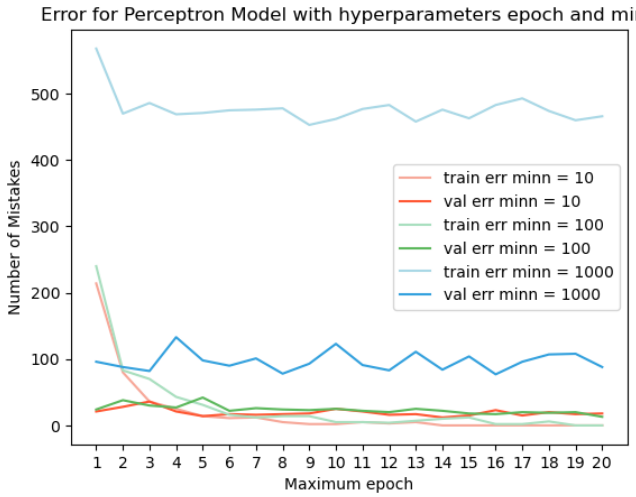
From the tables provided earlier, it is apparent that rank_corr has led to a slight improvement in the F2-score. Given that we were working with an imbalanced dataset, this metric holds greater significance for us. Both compute_correlation and rank_mutual have exhibited similar impacts. However, overall, we have not observed substantial enhancements through the usage of feature selection techniques.

3.(a) What is your model assessment strategy? Justify your validation methodology. (You may want to read the rest of this problem before you proceed to understand what your tasks will be).

We should not use any of the test emails in the process of calculating features, especially in build_vocab function. In other words, we should not use any information of the test data to build our vocabulary. Thus, we first split emails into train and test emails, and then we will pass train and test emails into build_vocab function. For this purpose, we will choose 15% of the whole samples for test (900 samples). From the remaining training data, we select 18% (918 samples) for validation purposes and allocate the remaining 4182 samples for training. Subsequently, we will validate the model additionally using hold-out validation. Then we train the perceptron on the training emails, and once the training process has been finished, we evaluate the performance of perceptron with learned weights on the validation data for several minn and epoch parameters. We also used minn = 10, 100, 1000 and epoch = 1:20 in the process of creating vocabulary.

3.(g) Train a perceptron using your training set specified in 3a. Plot the training and estimated generalization error as a function of the maximum number of epochs. What is the optimal algorithm and parameter? How many mistakes are made before the algorithm terminates if you want to classify all the training points properly?

We employ the hold-out method, ensuring that the model remains unaware of the validation data. Consequently, we can consider the validation error as a measure of the model's generalization performance.



In the preceding figure, lighter colors represent training errors, while stronger colors indicate generalized errors. It is evident that the model performs poorly for minn = 1000, making numerous mistakes. However, for minn = 10 and 100, the performances appear to be similar. Considering the reduced dimensionality of the data to 1186 (as opposed to the initial 9105 features for minn = 5), we choose minn = 100 for further analysis.

Additionally, we observe that both models (minn = 10, 100) cease training after 16 epochs, having achieved error-free performance. To ensure a reliable outcome, we opt for epoch = 20, enabling the training to halt automatically if the model reaches error-free status.

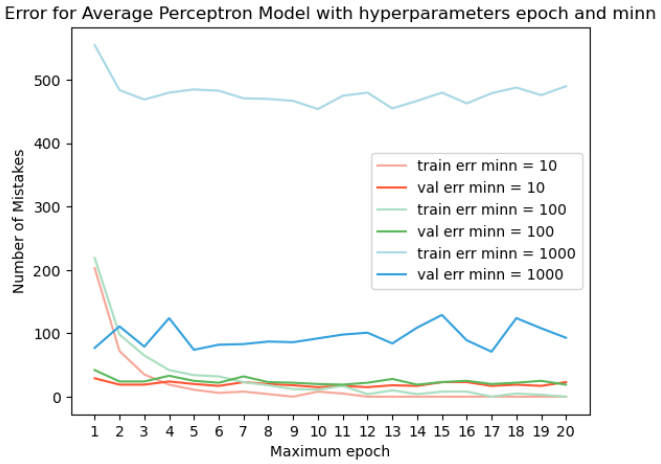
Consequently, we proceed to retrain our data using the entire training dataset (including both the train and validation sets) and evaluate the outcomes using the test dataset. The subsequent figure depicts the number of mistakes the model encounters at the conclusion of each epoch, along with the total number of errors during the training process.

```
history of training:  {1: 265, 2: 122, 3: 66, 4: 53, 5: 30, 6: 25, 7: 19, 8: 16, 9: 21, 10: 19, 11: 13, 12: 7, 13: 8,
14: 4, 15: 4, 16: 6, 17: 5, 18: 3, 19: 1, 20: 0}
------------------------------
number of mistakes on training set:  687
```

The table below shows expected predictive performance of this model after training whole training set.

	Accuracy	Precision	Recall	F1-score	F2-score
test	0.9822	0.9783	0.9644	0.9713	0.9671

3.(k) Plot the training and estimated generalization error as a function of the maximum number of epochs for the averaged perceptron.



3.(l) What is your final or "optimal" algorithm? In other words, train the model with as much data as you can possibly with the optimal algorithm + hyperparameter (maximum number of epochs) values. What is the expected predictive performance of this model?

Similar to part (g), we opt for minn = 100 and epoch = 20. Subsequently, we assess the average perceptron's performance on the test data, determining whether to proceed with either the perceptron or the average perceptron based on this evaluation.

The figure below illustrates the number of mistakes made by the average perceptron model at the end of each epoch, along with the total number of errors encountered during the training process.

```
history of training:  {1: 264, 2: 116, 3: 71, 4: 51, 5: 39, 6: 19, 7: 19, 8: 14, 9: 17, 10: 23, 11: 17, 12: 19, 13: 1
1, 14: 6, 15: 4, 16: 4, 17: 2, 18: 7, 19: 0}
------------------------------
number of mistakes on training set:  703
```

The table below shows expected predictive performance of this model after training whole training set which is the performance of the model on the test data.

	Accuracy	Precision	Recall	F1-score	F2-score
test	0.9744	0.9574	0.9608	0.9591	0.9601

Based on the results of the models on test data, it can be seen that the perceptron model has slightly better performance compared to average perceptron.

3.(m) Using the vocabulary list together with the parameters learned in the previous question, output the 15 words with the most positive weights. What are they? Which 15 words have the most negative weights?

```
Words with the most positive weights:  ['the', 'next', 'releas', 'm', 'see', 'me', 'abl', 'second', 's', 'mail', 'whe
n', 'you', 'long', 'befor', 'actual']
Words with the most negative weights:  ['content', 'plain', 'mon', 'small', 'messag', 'cv', 'quick', 'test', 'show',
'go', 'per', 'but', 'code', 'becaus', 'thing']
```