1. Bias-Variance Trade-off of LASSO
    a) As λ increases, LASSO drives coefficients to zero, resulting in a simpler model. A simpler model tends to underfit the data and thus have higher bias. Thus, the general trend of the bias is increasing as λ increases.
    b) As λ increases, LASSO drives coefficients to zero, which would make the model less sensitive to the changes in the data, reducing the influence of individual data points and outliers. Thus, the general trend of the variance is decreasing as λ increases.
    c) When $\lambda = 0$, the regularization term in LASSO disappears, and the model is the same as the Ordinary Least Squares (OLS) regression. Thus, the bias is the same as that of OLS, and is generally smaller than when $\lambda > 0$ (since $\lambda > 0$ would result in simpler model and thus higher bias based on (a)).
    d) When $\lambda = \infty$, all coefficients would be shrunk towards 0, resulting in an extremely simple and inflexible model. The model generally would make the same prediction every time and is not responsive to the data at all. Thus, the variance would be very low.

2. Spam classification using Naïve Bayes and Standard Logistic Regression

c) Table 1 below shows the performances of Gaussian Naïve Bayes model on each of the 4 preprocessing methods:

Table 1 Naive Bayes Performances with Different Preprocessing Method

method	train-acc	train-auc	test-acc	test-auc
nothing	0.825333	0.946728	0.816989	0.942271
std	0.816	0.890666	0.810119	0.875263
log	0.823667	0.954344	0.815116	0.948122
bin	0.798667	0.949411	0.800125	0.944731

e) Table 2 below shows the performances of Standard Logistic Regression model on each of the 4 preprocessing methods:

Table 2 Standard Logistic Regression Performances with Different Preprocessing Methods

method	train-acc	train-auc	test-acc	test-auc
nothing	0.933333	0.977739	0.918176	0.97122
std	0.934	0.977744	0.919425	0.970916
log	0.947667	0.985909	0.936914	0.98264
bin	0.939333	0.981327	0.925047	0.97838

f) Figure 1 below shows the Receiver Operating Characteristic (ROC) curves of Naïve Bayes on test set with 4 preprocessing methods:

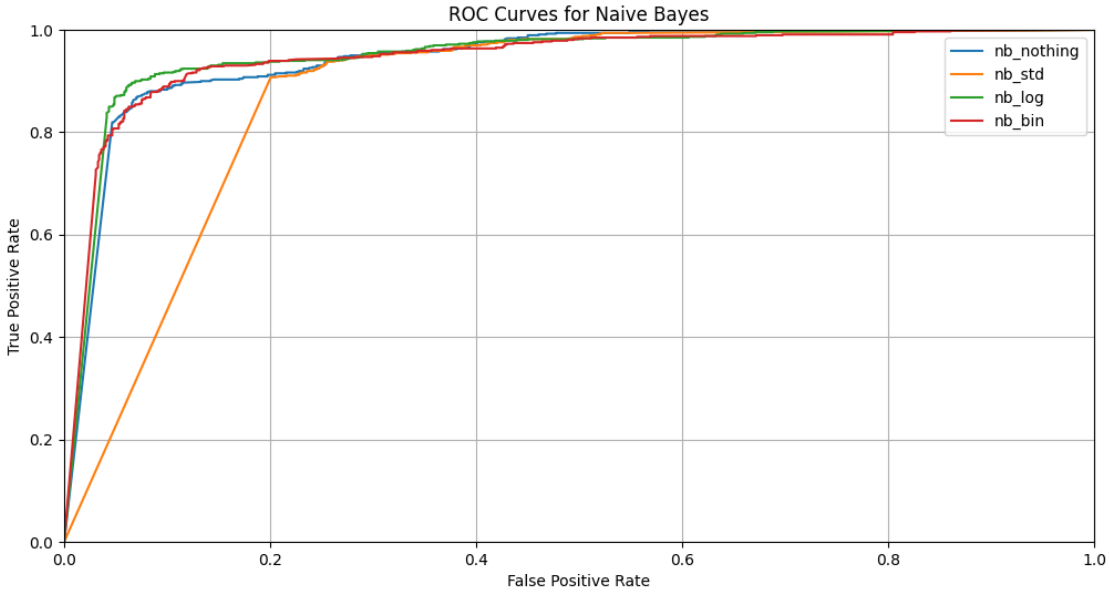


Figure 1 ROC Curves for Naive Bayes

Figure 2 below shows the ROC curves of Standard Logistic Regression on test set with 4 preprocessing methods:

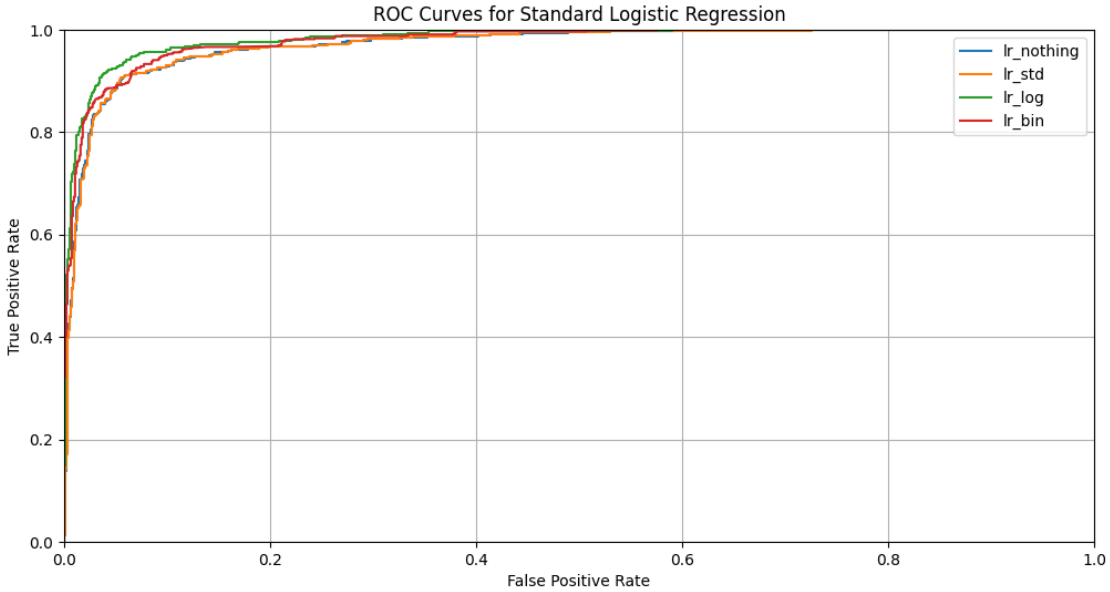
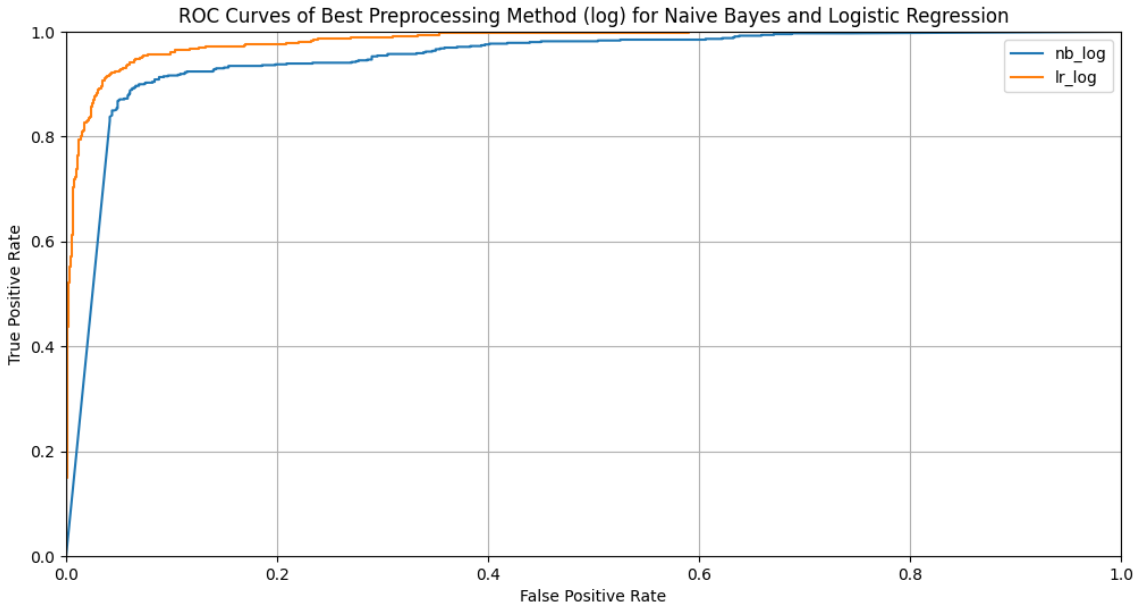


Figure 2 ROC Curves for Standard Logistic Regression

From Figure 1 and Figure 2, we can see that for both Naïve Bayes and Standard Logistic Regression models, the log preprocessing methods yield the best ROC curves. Plot the best ROC curves on the same plot would give us Figure 3:



Regression

g) From Table 1, Table 2, Figure 1, Figure 2, we can see that different preprocessing methods would have some quite obvious impacts on the accuracies, AUCs, and ROC curves of the same model. Among the 4 preprocessing methods we tested, the log method yields the best AUC scores on both Naïve Bayes and Logistic Regression models with both train set and test set (Table 1, Table 2). Moreover, the log method also gives the best ROC curves on the test set for both Naïve Bayes and Logistic Regression models (Figure 1, Figure 2). However, for Naïve Bayes model, the nothing method (do nothing to the dataset) yields the best accuracies on the train and test sets, though the log method has accuracies that are very close to that of the nothing method (Table 1). If we compare Naïve Bayes model with Logistic Regression model, we can see that Logistic Regression model seems to have better accuracies and AUC scores on both train set and test set regardless of the preprocessing method (Table 1, Table 2). And if we compare their ROC curves using the best preprocessing methods respectively (the best methods are log method for both models in this case), Logistic Regression model also seems to have a better ROC curve than Naïve Bayes model does (Figure 3).

3. Exploring Model Selection Strategies for Logistic Regression with Regularization

a) I preprocess the data using the do_log method from 2(a)iii, because we will use Logistic Regression for this question and from Table 2, the logistic regression model with the log method has the best performances.

f) My regularization parameter search space for ridge and LASSO would be [0.01, 0.1, 1, 10, 100, 1000, 10000].

g) I tested with 3 different split ratios: 0.2, 0.5, and 0.8. The choices of the 3 ratios mainly aim to test on 3 situations: when train set is larger than validation set, when train set and validation set has the same size, and when train set is smaller than validation set. Table 3 shows the performances of Ridge and LASSO on a split ratio of 0.2. Table 4 is for validation size of 0.5. Table 5 is for validation size of 0.8.
Based on the Table 3, Table 4, and Table 5, when validation size is 0.2, the best alpha for ridge is 1, and for Lasso is 0.01.
When validation size is 0.5, the best alpha for ridge is 0.01 or 10, for LASSO is 1.
When validations size is 0.8, the best alpha for ridge is 10, for LASSO is 1.

Table 3 Validation Performances of Ridge and LASSO Logistic Regression Models on Holdout Method with Validation Size of 0.2

alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
0.01	0.935	0.975126	0.961667	0.988911
0.1	0.93	0.977584	0.933333	0.984743
1	0.961667	0.985605	0.925	0.972022
10	0.928333	0.976687	0.941667	0.975528
100	0.93	0.979658	0.911667	0.972714
1000	0.915	0.967743	0.741667	0.758078
10000	0.808333	0.942894	0.571667	0.5

Table 4 Validation Performances of Ridge and LASSO Logistic Regression Models on Holdout Method with Validation Size of 0.5

alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
0.01	0.938667	0.983576	0.936667	0.976944
0.1	0.928667	0.978087	0.938667	0.978203
1	0.937333	0.978883	0.948	0.981944
10	0.939333	0.981383	0.947333	0.981528
100	0.934	0.976337	0.885333	0.953697
1000	0.904667	0.960788	0.632	0.5
10000	0.683333	0.929026	0.604667	0.5

Table 5 Validation Performances of Ridge and LASSO Logistic Regression Models on Holdout Method with Validation Size of 0.8

alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
0.01	0.91125	0.960712	0.909167	0.963266
0.1	0.91875	0.974595	0.912917	0.971675
1	0.92875	0.973768	0.934583	0.977772
10	0.935417	0.977823	0.9275	0.971535

100	0.907083	0.964052	0.825417	0.902175
1000	0.88625	0.950462	0.61875	0.5
10000	0.658333	0.939125	0.611667	0.5

h) Table 6 shows the performances of Ridge and LASSO with 2-fold cross validation.
Table 7 shows the performances of Ridge and LASSO with 5-fold cross validation.
Table 8 shows the performances of Ridge and LASSO with 10-fold cross validation.
When k=2, the best alpha for Ridge is 10, for LASSO is 0.1 or 10.
When k=5, the best alpha for Ridge is 10, for LASSO is 0.1 or 10.
When k=10, the best alpha for Ridge is 1, for LASSO is 0.1 or 10.

Table 6 Validation Performances of Ridge and LASSO Logistic Regression Models on K-fold Cross Validation Method with k = 2

alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
0.01	0.936	0.97941	0.934333	0.979399
0.1	0.938333	0.979601	0.941	0.980089
1	0.938667	0.978749	0.938333	0.977457
10	0.938667	0.980409	0.942	0.977432
100	0.929	0.976021	0.885667	0.952465
1000	0.898667	0.959379	0.615333	0.5
10000	0.736	0.941705	0.615333	0.5

Table 7 Validation Performances of Ridge and LASSO Logistic Regression Models on K-fold Cross Validation Method with k = 5

alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
0.01	0.939667	0.980991	0.937333	0.981729
0.1	0.94	0.980847	0.942	0.982231
1	0.941333	0.981587	0.943	0.981215
10	0.941333	0.98159	0.943333	0.979438
100	0.933333	0.977651	0.903333	0.962783
1000	0.908	0.963042	0.748333	0.781893
10000	0.810667	0.94538	0.615333	0.5

Table 8 Validation Performances of Ridge and LASSO Logistic Regression Models on K-fold Cross Validation Method with k = 10

alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
0.01	0.939667	0.981603	0.941	0.981787
0.1	0.942667	0.981178	0.940667	0.982361
1	0.944	0.981888	0.942333	0.981877
10	0.943	0.981687	0.944333	0.979629
100	0.935333	0.978633	0.909	0.965803
1000	0.91	0.964714	0.748	0.824908
10000	0.827333	0.945895	0.615333	0.5

j) Table 9 shows the performances of Ridge and LASSO with different values of s, validation size, and alpha. Table 10 shows the best alpha for Ridge and LASSO respectively with different s and val-size.

Table 9 Validation Performances of Ridge and LASSO Logistic Regression Models on Monte Carlo Cross Validation

s	valsize	alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
5	0.2	0.01	0.946333	0.98435	0.942667	0.981393
5	0.2	0.1	0.932667	0.975773	0.938	0.978672
5	0.2	1	0.941333	0.981467	0.944667	0.981696
5	0.2	10	0.945	0.979093	0.943667	0.978131
5	0.2	100	0.931667	0.976837	0.902	0.958249
5	0.2	1000	0.901	0.964414	0.734667	0.779182
5	0.2	10000	0.816333	0.94581	0.620667	0.5
5	0.5	0.01	0.939467	0.977996	0.938	0.979681
5	0.5	0.1	0.9364	0.979384	0.935067	0.977687
5	0.5	1	0.937733	0.979165	0.940533	0.979865
5	0.5	10	0.944133	0.981123	0.937733	0.977322
5	0.5	100	0.934667	0.978246	0.8856	0.951063
5	0.5	1000	0.901733	0.960772	0.617333	0.5
5	0.5	10000	0.7276	0.942008	0.6152	0.5
5	0.8	0.01	0.917417	0.968903	0.92125	0.96785
5	0.8	0.1	0.9265	0.9747	0.923083	0.97036
5	0.8	1	0.932583	0.975849	0.93	0.976376
5	0.8	10	0.937583	0.97821	0.931167	0.973604
5	0.8	100	0.919083	0.970741	0.83475	0.902567
5	0.8	1000	0.872167	0.950741	0.614667	0.5
5	0.8	10000	0.632417	0.936143	0.615583	0.5
10	0.2	0.01	0.938167	0.980166	0.940667	0.982916
10	0.2	0.1	0.939833	0.980968	0.938667	0.980829
10	0.2	1	0.938	0.980087	0.9435	0.980881
10	0.2	10	0.939833	0.978663	0.943833	0.980876
10	0.2	100	0.936667	0.977809	0.8985	0.960857
10	0.2	1000	0.910667	0.966033	0.749833	0.789564
10	0.2	10000	0.810667	0.944597	0.609167	0.5
10	0.5	0.01	0.9372	0.979115	0.936867	0.980306
10	0.5	0.1	0.9412	0.980873	0.935267	0.97919
10	0.5	1	0.9402	0.980866	0.9404	0.980523
10	0.5	10	0.943333	0.981072	0.939733	0.977558
10	0.5	100	0.929067	0.975833	0.885867	0.950016
10	0.5	1000	0.895267	0.957439	0.612	0.5
10	0.5	10000	0.7344	0.93962	0.615133	0.5
10	0.8	0.01	0.915292	0.965147	0.915292	0.965403
10	0.8	0.1	0.924667	0.972444	0.918833	0.970453
10	0.8	1	0.931417	0.976533	0.933667	0.975814
10	0.8	10	0.934417	0.977264	0.924958	0.971012

10	0.8	100	0.918083	0.970963	0.827708	0.905772
10	0.8	1000	0.878208	0.950447	0.6155	0.5
10	0.8	10000	0.632458	0.935212	0.614417	0.5

Table 10 Best Alpha for Ridge and LASSO with Different values of s and Validation Size

s	Val-size	Ridge alpha	LASSO alpha
5	0.2	0.01	1
5	0.5	10	1
5	0.8	10	1 or 10
10	0.2	0.1	1 or 10
10	0.5	10	1
10	0.8	10	1

k) From 3g, 3h, and 3j, the best alphas for Ridge are 0.01, 0.1, 1, 10, the best alphas for LASSO are 0.01, 0.1, 1, 10. shows the performances of Ridge and LASSO logistic regression model trained on the whole training set and tested on test set using their optimal alphas from 3g, 3h, and 3j, respectively (though coincidentally Ridge and LASSO has the same set of optimal alphas).

Table 11 shows the performances of Ridge and LASSO logistic regression model trained on the whole training set and tested on test set using their optimal alphas from 3g, 3h, and 3j, respectively (though coincidentally Ridge and LASSO has the same set of optimal alphas).

Table 11 Performances of Ridge and LASSO with Best Alphas

alpha	ridge-acc	ridge-auc	lasso-acc	lasso-auc
0.01	0.938164	0.982686	0.938164	0.983039
0.1	0.93629	0.983165	0.938788	0.983145
1	0.936914	0.983469	0.938788	0.98355
10	0.937539	0.982997	0.938164	0.981211

l) Table 12 compares all performances within each model selection techniques and selects alphas that yield the best performances within each techniques respectively (data from Table 3, Table 4, Table 5, Table 6, Table 7, Table 8, and Table 9). If the best accuracy and the best AUC do not come from the same alpha, the alpha with the best AUC is preferred. Comparing the best alphas in Table 12 with the alphas in Table 11 that has the best performances, we can see that for Ridge, all three techniques show an alpha that either has the best accuracy or the best AUC when trained on whole training set and tested on test set. However, for LASSO, the hold out technique selects an alpha that does not have either best accuracy or best AUC on the whole dataset. The k-fold technique selects an alpha that has the best accuracy and second best AUC, and the Monte Carlo technique selects an alpha that has both the best accuracy and the best AUC on the whole dataset.
In addition, if we compare the 3 techniques theoretically, both k-fold and Monte

Carlo techniques involves training the model multiple times on different splits of data and showing the average accuracies and AUCs, but the hold out method only split the data and train the model once. Therefore, the k-fold and Monte Carlo techniques are more robust than the holdout technique and gives more reliable accuracies and AUCs than the holdout technique does. However, since k-fold and Monte Carlo both involve training model multiple times, they are more computationally complex than the holdout technique.

Table 12 Best Performances of Ridge and LASSO using Different Model Selection Techniques

Method	Model	Best Alpha	Acc	AUC
Hold out	Ridge	1	0.961667	0.985605
	LASSO	0.01	0.961667	0.988911
k-fold	Ridge	1	0.944	0.981888
	LASSO	0.1	0.940667	0.982361
Monte Carlo	Ridge	0.01	0.946333	0.98435
	LASSO	1	0.944667	0.981696