EMORY UNIVERSITY

Department of Computer Science & Informatics

Masoud Nateghi

CS534 HW1

Professor Joyce Ho

1. While it is hard to write the explicit formula for the bias and variance of using LASSO, we can quantify the expected general trend. Make sure you justify the answers to the following questions for full points:

1. (a) What is the general trend of the bias as λ increases?

1. (b) What about the general trend of the variance as λ increases?

As λ increases the weights of the model tend to be closer to zero which implies a simpler model and we know that for simpler model, bias increases and variance decreases.

1. (c) What is the bias at λ = 0?

λ = 0 in the loss function transforms the regularized model to simple linear regression model which has a bias = 0.

1. (d) What about the variance at λ = ∞?

We can compute variance of a model using the following formula:

$$Var\left(\hat{f}(x_0)\right) = E\left[\hat{f}(x_0) - E\left(\hat{f}(x_0)\right)\right]^2$$

and for λ = ∞ weights will become zero. Thus:

$$\hat{f}(x_0) = 0 \; or \; constant \rightarrow Var\left(\hat{f}(x_0)\right) = 0$$

which is the simplest regression model that can exist with zero variance.

2. (c) Fit a Naive Bayes model to each of the four preprocessing steps above using the code in 2b. Each preprocessing should be performed independently (i.e., use each of the functions you created in 2a on the original dataset). Report the accuracy rate and AUC on the training and test sets across the 4 preprocessing steps in a table.

As the question has not mentioned which model to use for Naive Bayes, we tried Gaussian, Multinomial and Bernoulli on the preprocessed datasets.

	GaussianNB				MultinomialNB*				BernoulliNB			
	org	std	log	bin	org	std	log	bin	org	std	log	bin
train_acc	0.825	0.816	0.823	0.798	0.794	-	-	**0.914**	0.888	0.904	0.868	0.888
train_auc	0.946	0.890	0.954	0.949	0.848	-	-	**0.966**	0.953	0.963	0.934	0.953
test_acc	0.816	0.810	0.815	0.800	0.793	-	-	**0.916**	0.884	0.900	0.867	0.884
test_auc	0.942	0.875	0.948	0.944	0.851	-	-	**0.965**	0.953	0.959	0.933	0.953

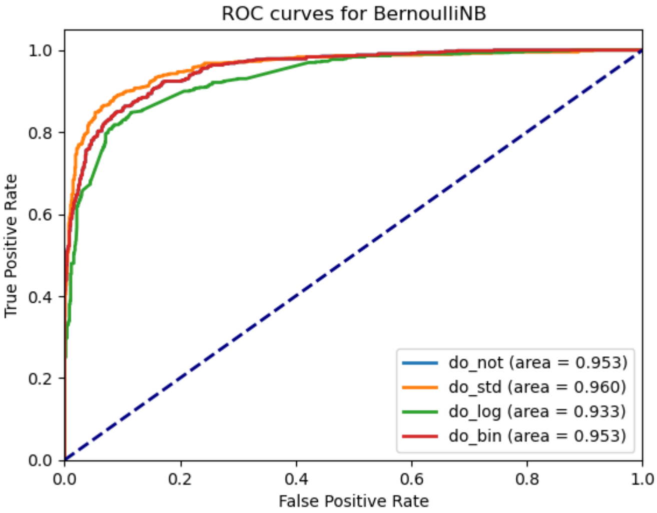
*Multininomial cannot take a dataset with negative values

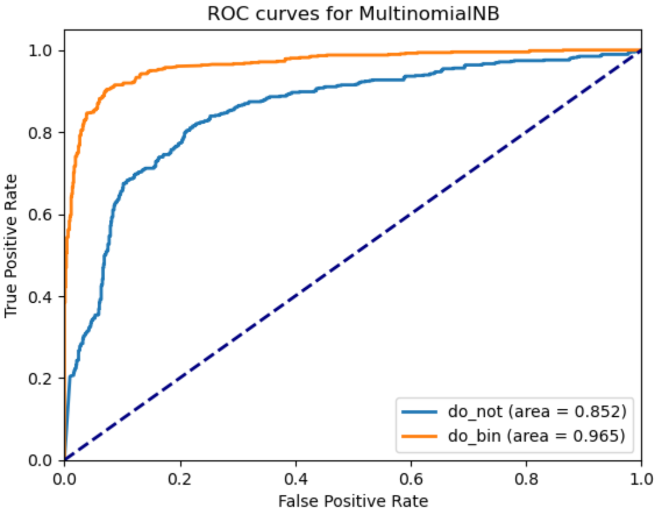
2. (e) Fit a standard (no regularization) logistic regression model to each of the four preprocessing steps above using the code in 2d. Report the accuracy rate and AUC on the training and test sets for the 4 preprocessing steps in a table.

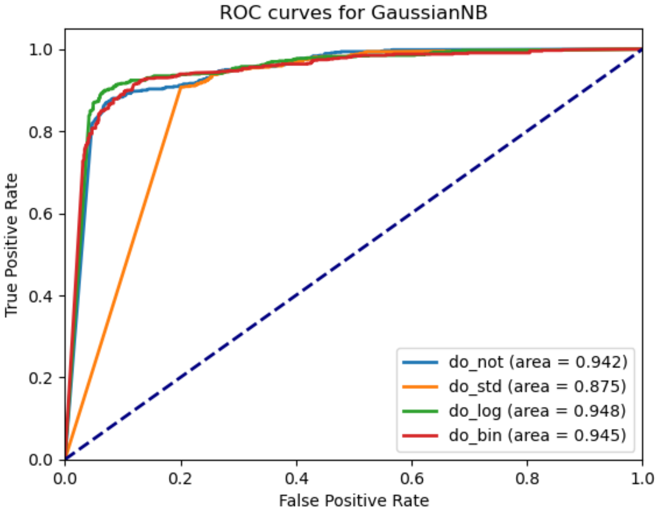
	Logistic Regression			
	org	std	log	bin
train_acc	0.924	0.932	**0.948**	0.937
train_auc	0.967	0.976	**0.985**	0.980
test_acc	0.923	0.920	**0.938**	0.921
test_auc	0.965	0.969	**0.983**	0.979

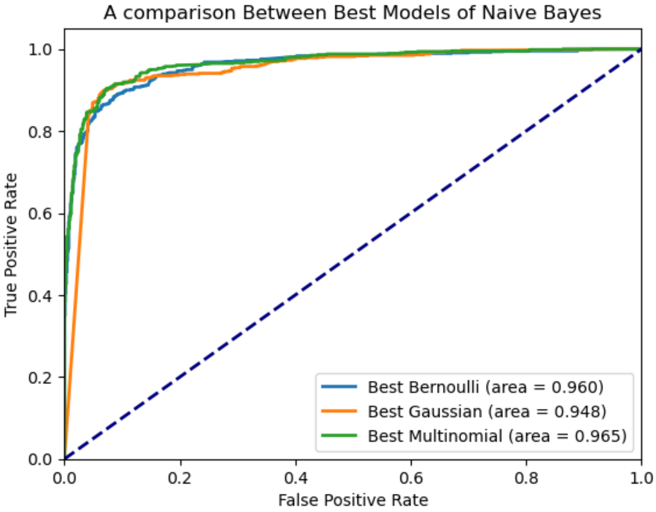
2. (f) Plot the receiver operating characteristic (ROC) curves for the test data. You should generate 3 plots:

- One plot containing the 4 Naive Bayes model curves representing each of the preprocessing steps.

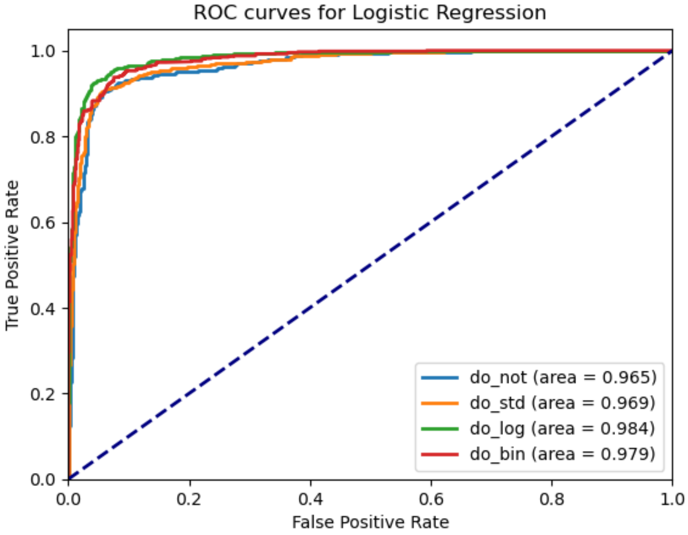




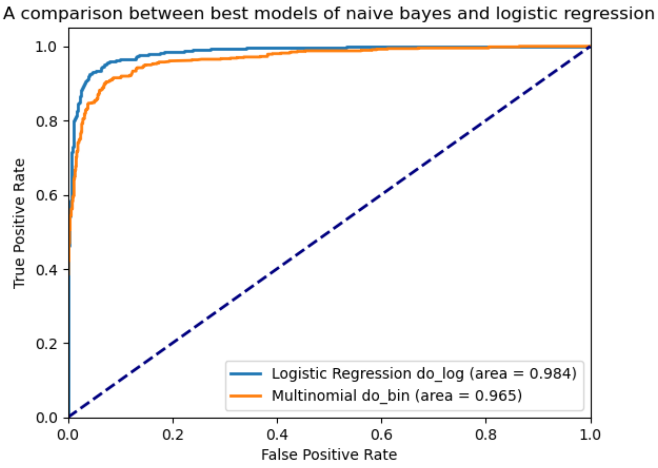




- One plot containing the 4 logistic regression model curves representing each of the preprocessing steps.



- One plot containing the best Naive Bayes model and the best logistic regression model curve.



2. (g) Given your results in 2c, 2e, and 2f, comment on how the preprocessing affects the models (logistic and Naive Bayes) with regards to ROC, AUC, and accuracy. Also, comment on how Naive Bayes compares with logistic regression.

Preprocessing can play a vital role to improve training of the model and the final metrics like AUC, and accuracy. As it can bee seen from 2c, the best performance has achieved by the MultinomialNB model and with binarizing preprocessing. On the other hand, best logistic regression model benefits from log transformation preprocessing. As a result, we can see that feeding the model with the initial dataset might not lead to the desired performance. From the last figure of 2f we can clearly observe that logistic regression model outperforms, since it has a bigger AUC and has better accuracy metrics for both train and test datasets.

3. (a) How did you preprocess the data for this method? Why?

We did log transformation preprocessing based on the result of previous question.

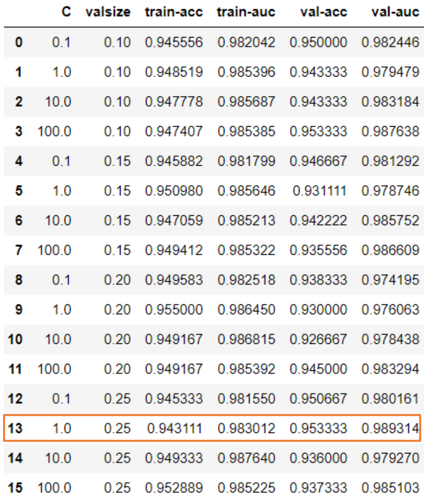
3. (f) What is your regularization parameter search space for ridge and LASSO regularized logistic regression?

We checked C in interval [0.1, 100].

3. (g) For your regularization parameter search space specified in 3f, fit ridge and LASSO using 3d by searching over a variety of split ratios. For each unique split ratio, report the validation metrics (AUC and accuracy). What is the best 'parameter' for ridge and LASSO based on these metrics? Note ridge and LASSO may have different optimal 'parameters'.

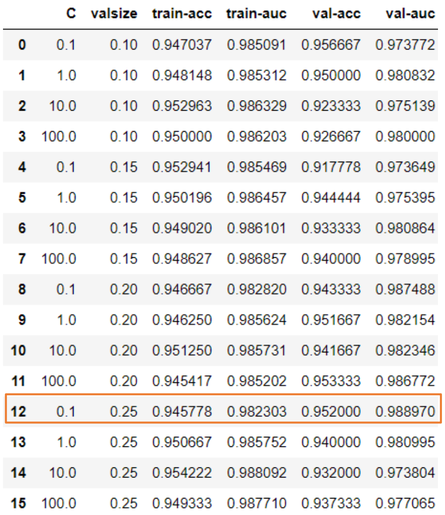
Our metric for model selection was validation AUC.

Lasso regression:



	C	valsize	train-acc	train-auc	val-acc	val-auc
0	0.1	0.10	0.945556	0.982042	0.950000	0.982446
1	1.0	0.10	0.948519	0.985396	0.943333	0.979479
2	10.0	0.10	0.947778	0.985687	0.943333	0.983184
3	100.0	0.10	0.947407	0.985385	0.953333	0.987638
4	0.1	0.15	0.945882	0.981799	0.946667	0.981292
5	1.0	0.15	0.950980	0.985646	0.931111	0.978746
6	10.0	0.15	0.947059	0.985213	0.942222	0.985752
7	100.0	0.15	0.949412	0.985322	0.935556	0.986609
8	0.1	0.20	0.949583	0.982518	0.938333	0.974195
9	1.0	0.20	0.955000	0.986450	0.930000	0.976063
10	10.0	0.20	0.949167	0.986815	0.926667	0.978438
11	100.0	0.20	0.949167	0.985392	0.945000	0.983294
12	0.1	0.25	0.945333	0.981550	0.950667	0.980161
13	1.0	0.25	0.943111	0.983012	0.953333	0.989314
14	10.0	0.25	0.949333	0.987640	0.936000	0.979270
15	100.0	0.25	0.952889	0.985225	0.937333	0.985103

Ridge regression:



	C	valsize	train-acc	train-auc	val-acc	val-auc
0	0.1	0.10	0.947037	0.985091	0.956667	0.973772
1	1.0	0.10	0.948148	0.985312	0.950000	0.980832
2	10.0	0.10	0.952963	0.986329	0.923333	0.975139
3	100.0	0.10	0.950000	0.986203	0.926667	0.980000
4	0.1	0.15	0.952941	0.985469	0.917778	0.973649
5	1.0	0.15	0.950196	0.986457	0.944444	0.975395
6	10.0	0.15	0.949020	0.986101	0.933333	0.980864
7	100.0	0.15	0.948627	0.986857	0.940000	0.978995
8	0.1	0.20	0.946667	0.982820	0.943333	0.987488
9	1.0	0.20	0.946250	0.985624	0.951667	0.982154
10	10.0	0.20	0.951250	0.985731	0.941667	0.982346
11	100.0	0.20	0.945417	0.985202	0.953333	0.986772
12	0.1	0.25	0.945778	0.982303	0.952000	0.988970
13	1.0	0.25	0.950667	0.985752	0.940000	0.980995
14	10.0	0.25	0.954222	0.988092	0.932000	0.973804
15	100.0	0.25	0.949333	0.987710	0.937333	0.977065

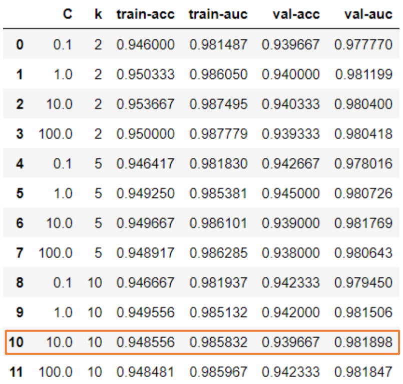
In this stage, I fixed valsize = 0.25 based on the Tables above, and tried different C for better tuning.

We obtained $C_{lasso} = 0.5$ and $C_{ridge} = 0.45$.

3. (h) For your regularization parameter search space specified in 3f, fit ridge and LASSO using the k-fold cross-validation approach by searching over k = 2, 5, 10. For each value of k, specify the best 'parameter' based on the metrics.

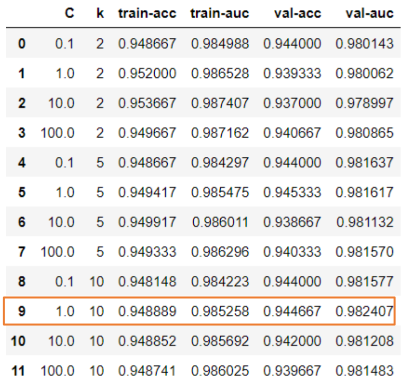
Our metric for model selection was validation AUC.

Lasso regression:



	C	k	train-acc	train-auc	val-acc	val-auc
0	0.1	2	0.946000	0.981487	0.939667	0.977770
1	1.0	2	0.950333	0.986050	0.940000	0.981199
2	10.0	2	0.953667	0.987495	0.940333	0.980400
3	100.0	2	0.950000	0.987779	0.939333	0.980418
4	0.1	5	0.946417	0.981830	0.942667	0.978016
5	1.0	5	0.949250	0.985381	0.945000	0.980726
6	10.0	5	0.949667	0.986101	0.939000	0.981769
7	100.0	5	0.948917	0.986285	0.938000	0.980643
8	0.1	10	0.946667	0.981937	0.942333	0.979450
9	1.0	10	0.949556	0.985132	0.942000	0.981506
10	10.0	10	0.948556	0.985832	0.939667	0.981898
11	100.0	10	0.948481	0.985967	0.942333	0.981847

Ridge regression:



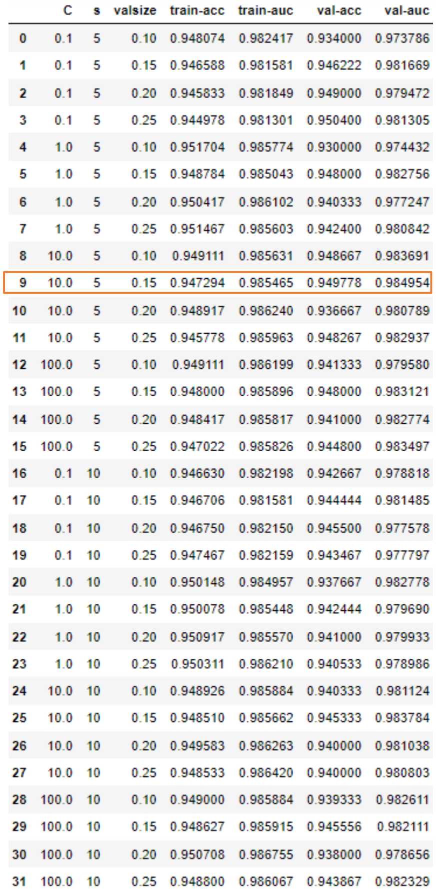
	C	k	train-acc	train-auc	val-acc	val-auc
0	0.1	2	0.948667	0.984988	0.944000	0.980143
1	1.0	2	0.952000	0.986528	0.939333	0.980062
2	10.0	2	0.953667	0.987407	0.937000	0.978997
3	100.0	2	0.949667	0.987162	0.940667	0.980865
4	0.1	5	0.948667	0.984297	0.944000	0.981637
5	1.0	5	0.949417	0.985475	0.945333	0.981617
6	10.0	5	0.949917	0.986011	0.938667	0.981132
7	100.0	5	0.949333	0.986296	0.940333	0.981570
8	0.1	10	0.948148	0.984223	0.944000	0.981577
9	1.0	10	0.948889	0.985258	0.944667	0.982407
10	10.0	10	0.948852	0.985692	0.942000	0.981208
11	100.0	10	0.948741	0.986025	0.939667	0.981483

We take k = 10 fixed. By fine tuning the parameters we obtained: $C_{lasso} = 6.5$ and $C_{ridge} = 1.1$.

3. (j) Fit ridge and LASSO using the Monte Carlo Cross-validation approach with s = 5, 10 samples and different split ratios. What is the best 'parameter' based on the splits and the samples?

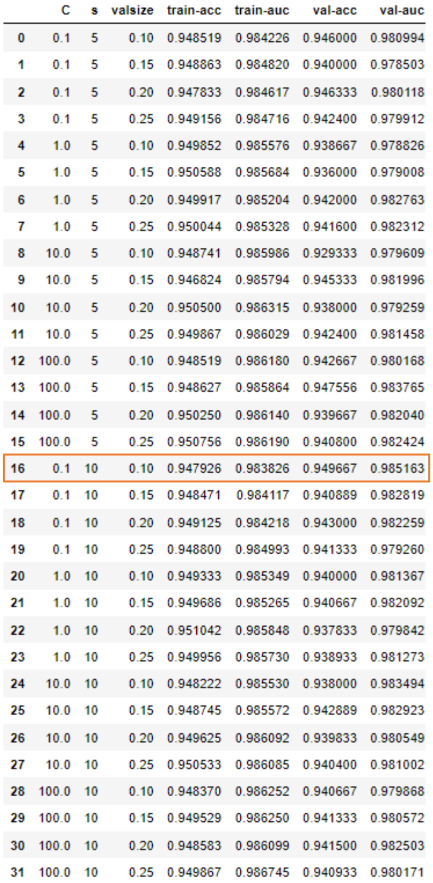
Our metric for model selection was validation AUC.

Lasso regression:



	C	s	valsize	train-acc	train-auc	val-acc	val-auc
0	0.1	5	0.10	0.948074	0.982417	0.934000	0.973786
1	0.1	5	0.15	0.946588	0.981581	0.946222	0.981669
2	0.1	5	0.20	0.945833	0.981849	0.949000	0.979472
3	0.1	5	0.25	0.944978	0.981301	0.950400	0.981305
4	1.0	5	0.10	0.951704	0.985774	0.930000	0.974432
5	1.0	5	0.15	0.948784	0.985043	0.948000	0.982756
6	1.0	5	0.20	0.950417	0.986102	0.940333	0.977247
7	1.0	5	0.25	0.951467	0.985603	0.942400	0.980842
8	10.0	5	0.10	0.949111	0.985631	0.948667	0.983691
9	10.0	5	0.15	0.947294	0.985465	0.949778	0.984954
10	10.0	5	0.20	0.948917	0.986240	0.936667	0.980789
11	10.0	5	0.25	0.945778	0.985963	0.948267	0.982937
12	100.0	5	0.10	0.949111	0.986199	0.941333	0.979580
13	100.0	5	0.15	0.948000	0.985896	0.948000	0.983121
14	100.0	5	0.20	0.948417	0.985817	0.941000	0.982774
15	100.0	5	0.25	0.947022	0.985826	0.944800	0.983497
16	0.1	10	0.10	0.946630	0.982198	0.942667	0.978818
17	0.1	10	0.15	0.946706	0.981581	0.944444	0.981485
18	0.1	10	0.20	0.946750	0.982150	0.945500	0.977578
19	0.1	10	0.25	0.947467	0.982159	0.943467	0.977797
20	1.0	10	0.10	0.950148	0.984957	0.937667	0.982778
21	1.0	10	0.15	0.950078	0.985448	0.942444	0.979690
22	1.0	10	0.20	0.950917	0.985570	0.941000	0.979933
23	1.0	10	0.25	0.950311	0.986210	0.940533	0.978986
24	10.0	10	0.10	0.948926	0.985884	0.940333	0.981124
25	10.0	10	0.15	0.948510	0.985662	0.945333	0.983784
26	10.0	10	0.20	0.949583	0.986263	0.940000	0.981038
27	10.0	10	0.25	0.948533	0.986420	0.940000	0.980803
28	100.0	10	0.10	0.949000	0.985884	0.939333	0.982611
29	100.0	10	0.15	0.948627	0.985915	0.945556	0.982111
30	100.0	10	0.20	0.950708	0.986755	0.938000	0.978656
31	100.0	10	0.25	0.948800	0.986067	0.943867	0.982329

Ridge regression:



	C	s	valsize	train-acc	train-auc	val-acc	val-auc
0	0.1	5	0.10	0.948519	0.984226	0.946000	0.980994
1	0.1	5	0.15	0.948863	0.984820	0.940000	0.978503
2	0.1	5	0.20	0.947833	0.984617	0.946333	0.980118
3	0.1	5	0.25	0.949156	0.984716	0.942400	0.979912
4	1.0	5	0.10	0.949852	0.985576	0.938667	0.978826
5	1.0	5	0.15	0.950588	0.985684	0.936000	0.979008
6	1.0	5	0.20	0.949917	0.985204	0.942000	0.982763
7	1.0	5	0.25	0.950044	0.985328	0.941600	0.982312
8	10.0	5	0.10	0.948741	0.985986	0.929333	0.979609
9	10.0	5	0.15	0.946824	0.985794	0.945333	0.981996
10	10.0	5	0.20	0.950500	0.986315	0.938000	0.979259
11	10.0	5	0.25	0.949867	0.986029	0.942400	0.981458
12	100.0	5	0.10	0.948519	0.986180	0.942667	0.980168
13	100.0	5	0.15	0.948627	0.985864	0.947556	0.983765
14	100.0	5	0.20	0.950250	0.986140	0.939667	0.982040
15	100.0	5	0.25	0.950756	0.986190	0.940800	0.982424
16	0.1	10	0.10	0.947926	0.983826	0.949667	0.985163
17	0.1	10	0.15	0.948471	0.984117	0.940889	0.982819
18	0.1	10	0.20	0.949125	0.984218	0.943000	0.982259
19	0.1	10	0.25	0.948800	0.984993	0.941333	0.979260
20	1.0	10	0.10	0.949333	0.985349	0.940000	0.981367
21	1.0	10	0.15	0.949686	0.985265	0.940667	0.982092
22	1.0	10	0.20	0.951042	0.985848	0.937833	0.979842
23	1.0	10	0.25	0.949956	0.985730	0.938933	0.981273
24	10.0	10	0.10	0.948222	0.985530	0.938000	0.983494
25	10.0	10	0.15	0.948745	0.985572	0.942889	0.982923
26	10.0	10	0.20	0.949625	0.986092	0.939833	0.980549
27	10.0	10	0.25	0.950533	0.986085	0.940400	0.981002
28	100.0	10	0.10	0.948370	0.986252	0.940667	0.979868
29	100.0	10	0.15	0.949529	0.986250	0.941333	0.980572
30	100.0	10	0.20	0.948583	0.986099	0.941500	0.982503
31	100.0	10	0.25	0.949867	0.986745	0.940933	0.980171

In the next step we took s = 5, and s = 10, valsize = 0.15, valsize = 0.1 fixed for lasso and ridge respectively. We then did another search around the best parameters found in the table to find the optimal parameter.

We obtained $C_{lasso} = 10$ and $C_{ridge} = 0.3$.

3. (k) Using the optimal parameters identified in 3g, 3h, and 3j, re-train the regularized logistic regression model using all the training data and report the performance on the test set in terms of AUC and accuracy in a table. Note that this means you are likely training at least 6 different regularized models and reporting the performance for each of the model.

For training we used all the training data. So, we did not used k or valsize to devide data into train and validation and we **did use** just the parameter C to evaluate different models.

	C	valsize	train-acc	train-auc	test-acc	test-auc
eval_holdout, lasso:	0.5	0.25	0.9506	0.9845	0.9394	0.9834
eval_holdout, ridge:	0.45	0.25	0.9496	0.9848	0.9381	0.9833
eval_kfold, lasso: k = 10	6.5	-	0.9486	0.9856	0.9381	0.9832
eval_kfold, ridge: k = 10	1.1	-	0.9490	0.9851	0.9369	0.9834
eval_mccv, lasso: s = 5	10	0.15	0.9480	0.9856	0.9387	0.9831
eval_mccv, ridge: s = 10	0.3	0.1	0.9500	0.9847	0.9381	0.9833

3. (l) Comment on how the different model selection techniques compare with one another with regard to AUC and accuracy, the robustness of the validation estimate, and the computational complexities of the three different hold-out techniques.

By analyzing the above table, we can see that different methods could find models that perform similarly, and the parameter C is not that much different for regularized model using different model evaluation methods. In terms of validation estimate, mccv and kfold are more robust compared to lasso and conversely, mccv and kfold are similar to each other and much computationally expensive and holdout is faster that those. In other words:

Robustness of estimation:

$$k - fold \approx Monte\ Carlo > holdout$$

Computation Complexity:

$$holdout > k - fold \approx Monte\ Carlo$$