Q1:

	Decision or Regression Tree	Linear Regression (Normal Equations)	Linear Ridge Regression (Normal Equations)	Linear Regression (Gradient Descent)	Logistic Regression (Gradient Descent)
Spam base	Train ACC: 0.9678 Test ACC: 0.9265	Train ACC: 0.9122 Test ACC: 0.9089	Train ACC: 0.912 Test ACC: 0.9091	Train ACC: 0.9130 Test ACC: 0.9104	Train ACC: 0.92969 Test ACC: 0.92393
Housing	Train MSE: 24.6563 Test MSE: 22.8097	Train MSE: 22.0812 Test MSE: 22.6382	Train MSE: 22.1108 Test MSE: 22.1305	Train MSE: 22.0812 Test MSE: 22.6382	N/A

Decision Tree

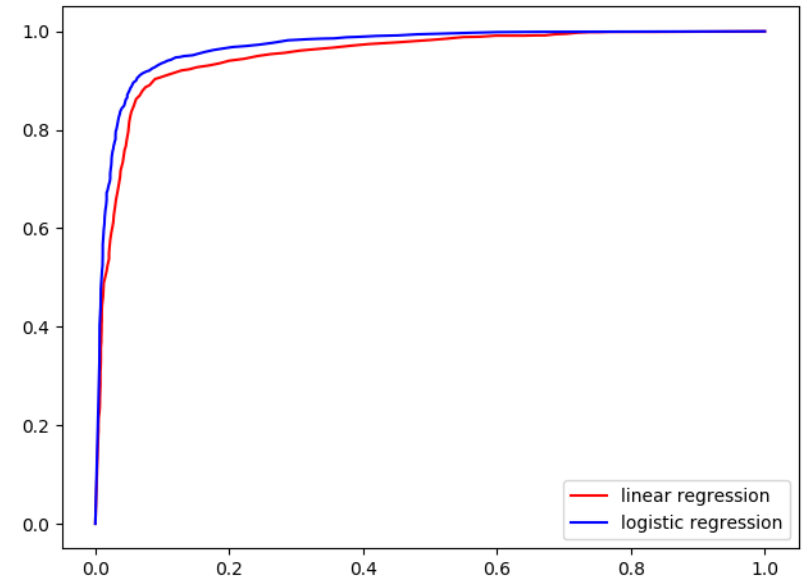
TP: 1500	FP: 313
FN: 134	TN: 2654

Linear Regression (Normal Equations)

TP: 1593	FP: 220
FN: 202	TN: 2586

Logistic Regression (Gradient Descent):

TP: 1654	FP: 159
FN: 191	TN: 2597



AUC:

Linear Regression: 0.951

Logistic Regression: 0.969

Q2:

iteration 1 , total mistake: 283

iteration 2 , total mistake: 321

iteration 3 , total mistake: 220

iteration 4 , total mistake: 169

iteration 5 , total mistake: 89

iteration 6 , total mistake: 0

Classifier weights: [2.09613228 0.10746497 0.01382512 0.07456444 0.07795897]

Normalized with threshold: [-0.05126822 -0.00659554 -0.03557239 -0.03719182]

Q3:

1.

- Θ: the parameters of a model
- a(Θ): weighted accuracy of $h_\Theta(X)$
- J(Θ): the likelihood of $h_\Theta(X)$ with training data
- λ: the coefficient of the regularization term
    - increase λ increases the effect of regularization, reduces variance, increases bias
    - decrease λ increases variance, reduces bias
- m: number of training examples
    - increase m helps reduce variance
- d: number of features
    - increase d reduces bias, but increases variance
    - decrease d increases bias, but decreases variance
-

2.

- **Machine learning can be treated as a combination of data representation, evaluation and optimization.**
  - There are some natural relationships between different types of them, but they may still be combined.
  - The three components are equally important
- **Generalization is the whole purpose of machine learning.**
  - To generalize beyond training set, it is important to separate training and testing set.
  - Because our goal is generalized, training error is used as a surrogate for test error, which might be a good thing
- **The effect data has on learning**
  - We always need data to do learning, and the more the better
  - Data itself cannot help to do better than random guessing, but some simple assumptions can help, because most functions we are learning are uniformly distributed
  - Deduction and induction are powerful tools to draw useful results from less data
- **Classifier can suffer with overfitting, having high training accuracy, but low testing accuracy.**
  - Generalization error can be decomposed as bias(underfitting) and variance(overfitting).
  - Because of overfitting, a more powerful learner is not necessarily better than a less powerful one.
  - Cross-validation and regularization are techniques to help fight overfitting
  - Overfitting is not necessarily caused by noise.
  - Multiple testing is closely related to overfitting, and a good way to solve this is to control the false discovery.
- **High dimensionality can make learners hard to learn, sometimes overweighting its benefits.**
  - The cause of this effect can come from the noise of irrelevant features, in high dimensions all examples look alike and that our intuitions, often do not apply in high-dimensional ones
  - In most applications examples are concentrated on or near a lower-dimensional manifold, which learners can take advantage of, or use dimensionality reduction algorithms.
- **Theoretical guarantees are often very loose.**
  - The bound on the number of examples needed to ensure good generalization.
  - Another asymptotic bound: given infinite data, the learner is guaranteed to output the correct classifier.
  - learning is a complex phenomenon, and just because a learner has a theoretical justification and works in practice doesn't mean the former is the reason for the latter
- **Feature engineering plays an important role in machine learning**
  - It's domain-specific, while learners can be largely general-purpose.
  - One way to automate this process is by automatically generating large numbers of candidate features and selecting the best by (say) their information gain with respect to the class.
  - there is ultimately no replacement for the smarts you put into feature engineering.
- **More data is sometimes more important than a better algorithm**
  - Because of not enough time to learn complex classifiers, in practice simpler classifiers are used
  - A smarter learner may have far less improvement than expected
  - Try simplest learners first
  - Learners can be separated into fixed-size and non-parametric.
  - Designing clever learners will pay off in the end, because the limitation of simple learners and data size
  - The bottleneck in the end is human cycle
- **Learn more models instead of one helps to improve the results**
  - Bagging: generate random variations of the training set by resampling, learn a classifier on each, combine the results by voting
  - Boosting: training weighted examples that are varied, each new classifier focuses on the examples the previous ones get wrong
  - Stacking: the outputs of individual classifiers become the inputs of a "higher-level" learner that figures out how best to combine
  - Model ensembles is different from BMA, by whether changing hypothesis space or not
- **Simpler model does not necessarily mean more accuracy**
  - there is no necessary connection between the number of parameters of a model and its tendency to overfit
  - simpler hypotheses is preferred because simplicity is a virtue in its own right, instead of hypothetical connection with accuracy
- **Key question is learnability instead of representability**
  - given finite data, time and memory, standard learners are limited
  - it pays to try different learners
  - Some representations are exponentially more compact than others for some functions
- **correlation does not imply causation**
  - In machine learning predictive variables are not under the control of the learner
  - correlation is a sign of a potential causal connection, and we can use it as a guide to further investigation
  - we would like to predict the effects of our actions, not just correlations between observable variables
  - if you can obtain experimental data, then by all means do so

Q4:

train accuracy: 0.93289

test accuracy: 0.92306

Q5.

Let T be the number of objects with label 1, and F be the number of objects with label 0.

Assume all the predicted value of all objects are distinctive.
The ROC curve can be seen as being drawn by connecting all points by setting the threshold from high to low between each two consecutive objects at each step.
Firstly, the point starts at (0,0) since all objects are predicted to be 0, so TPR = FPR = 0.
Each object with label 1 will cause the next point to move upward by 1/T, because it predicts 1 more true label correctly.
Similarly, each object with label 0 will cause the next point to move rightward by 1/F, because it predicts 1 more false label incorrectly.
Finally, it reaches point (1,1).

Ideally, if all pairs are in correct order, then AUC is 1.

Now, assume one object with label 0, and there are n objects with label 1 under it.
There will be n pairs of objects that are in incorrect order.
And there will be n points at the top-left corner moved with vector (1/F, -1/T), because it will produce 1/F more FPR n steps before expected and 1/T less TPR at these steps. Therefore, causing a loss in AUC of n/(T*F).

Finally, since (T*F) is the total number of possible pairs between two kinds of objects, summarizing all such objects with label 0, all loss in AUC is the percentage of percentage of pairs in correct order.

However, if there are duplicated values, this estimation might not be accurate. Since in the real applications, there will not be a high percentage of duplicated values, this estimation can be treated as the approximation of AUC.