1 Linear Regression (24 points)

Short Answer

Please give short (1-3 sentence) answers. You may discuss these questions with others, but write the answers in your own words.

(a) Does linear regression assume uncertainty in the measurement of the independent variables ($\mathbf{X}$), the dependent variable ($\mathbf{y}$), or both?

(b) Linear regression is sensitive to outliers. Briefly describe one solution to make it more robust to outliers.

(c) It is often assumed that the magnitude of the regression coefficients (abs($\mathbf{w}$)) indicate the importance of the corresponding features. Describe a situation where this may not be true.

(d) What problem arises if one independent variable is a perfect linear combination of other independent variables?

(e) "One-hot" or "1-of-k" encoding is a method of transforming a categorical variable with k categories into k separate binary indicator variables called "dummy" variables. This method precludes the use of a constant (or intercept) term. The constant term can be thought of as a coefficient corresponding to a column of ones appended to the $\mathbf{X}$ matrix. A column of ones is a linear combination of the indicator variables resulting from 1-of-k encoding. Additionally, if the $\mathbf{X}$ matrix contains indicator variables resulting from 1-of-k encoding, it cannot be normalized the traditional way (subtracting the mean of each column and dividing each column by its standard deviation) without one variable becoming a linear combination of the others. Describe an alternate way to encode categorical variables into binary indicator variables that still permits the use of a constant term and the use of column normalization.

(f) Using highly correlated independent variables does not necessarily affect the accuracy of a linear regression model, however it can lead to an unintuitive effect on the coefficients. Describe the effect on the coefficients.

(g) Suppose you are using logistic regression for binary classification, predicting the positive class whenever the posterior probability is greater than 0.5. Would you get the same predictive accuracy by running linear regression instead, predicting the positive class whenever $\hat{y}$ is greater than 0.5? Whether you answer yes or no, describe an advantage of using logistic regression over linear regression for binary classification.

(h) Linear regression can be thought of as approximately solving an over-determined system of linear equations. Describe the situation which would lead to an under-determined system of equations, and can it be solved?

2 Linear/Gaussian Discriminant Analysis (26 points)

(a) Suppose we are given a set of samples from two classes. Samples in the first class are in the form of $\mathbf{x} = (x_1, x_2, \cdots, x_{2D})$, where each component $x_d (d = 1, 2, \cdots, 2D)$ is drawn i.i.d. from a Gaussian distribution $\mathcal{N}(0, \sigma^2)$. Samples in the second class are in the form

of $\mathbf{x} = (x_1, x_2, \cdots, x_D, x_{D+1} + \delta, x_{D+2} + \delta, \cdots, x_{2D} + \delta)$, where $x_d \, (d = 1, 2, \cdots, 2D)$ is also drawn i.i.d. from $\mathcal{N}(0, \sigma^2)$ and δ is fixed.

- What is the solution of linear discriminant analysis? Does it change when δ changes (5 points)?
- What is the solution of Gaussian discriminant analysis? Does it change when δ changes (5 points)?

(b) Consider a set of samples from two classes c_1 and c_2.

- Suppose $p(\mathbf{x}|y = c_1)$ follows a multivariate Gaussian distribution $\mathcal{N}(\mu_\mathbf{1}, \mathbf{\Sigma})$, and $p(\mathbf{x}|y = c_2)$ follows a multivariate Gaussian distribution $\mathcal{N}(\mu_\mathbf{2}, \mathbf{\Sigma})$ $(\mu_\mathbf{1}, \mu_\mathbf{2} \in \mathcal{R}^D, \mathbf{\Sigma} \in \mathcal{R}^{D \times D})$. Show that $p(y|\mathbf{x})$ follows a logistic function, i.e., $y = \dfrac{1}{1 + \exp(-\theta^\top \mathbf{x})}$ for some θ (8 points).
- Show that the converse is not true: $p(y|\mathbf{x})$ being a logistic function does not imply $p(\mathbf{x}|y)$ is multivariate Gaussian. This suggests that Gaussian discriminant analysis makes stronger modeling assumption than Logistic Regression (8 points).

3 Perceptron and Online Learning (10 points)

The perceptron algorithm often makes harsh updates — it is strongly biased towards the current mistakenly-labeled sample. To remedy this, suppose at ith step, the classifier is $\mathbf{w}_i$ and we want to update a little bit conservatively the classifier based on observation of $(\mathbf{x}_i, y_i)$ to the new one $\mathbf{w}_{i+1}$. Derive a new update method for the perceptron such that it makes the smallest difference from the previous model, that is, minimizes $\|\mathbf{w}_{i+1} - \mathbf{w}_i\|_2$ while the new $\mathbf{w}_{i+1}$ classifies correctly on the current sample. You need to provide the closed form analytical equation for the update rule.

4 Programming - Logistic Regression (40 points)

In this problem, you will investigate data from the Titanic disaster to see if passengers survived at random, or if certain characteristics influenced whether they survived. The data can be found here: http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/titanic3.xls

A description of the fields can be found here:
http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/titanic3info.txt

All programming should be done in MATLAB. You must implement certain functions which we specify. Otherwise, you may use built-in MATLAB functions. Do not use code from the internet or from other students. Record answers to questions in your written LaTeX report.

(a) **Load Data (5 points)** Load the data into MATLAB (tip: try `xlsread`). For each column, report the number of missing values.

Randomly shuffle the rows of data. Then separate it evenly into two datasets: half for training and half for testing. Separate the *survived* column from the rest, as it will be the dependent variable you are trying predict. The remaining columns will be used as independent variables.

(b) **Monotonic Relationship (5 points)** For each numeric independent variable (*pclass, age, sibsp, parch, fare*), create a version of it that is discretized into 10 equal-width bins. You must implement the discretization function yourself. Use only training data when determining the discretization cutpoints. Testing data is not used at all in this task. Disregard missing values. If a variable already has fewer distinct values than bins, consider it already discretized.

In your report, show a bar plot for each numeric independent variable, where the x-axis is the bin number, and the y-axis is the probability of survival (estimated using only the training data). In your report, comment on whether or not each numeric independent variable seems to show a monotonic relationship with the probability of survival, according to the plots you made.

The purpose of this is to visually inspect independent variables to see if they have a relationship with the dependent variable that might not be linear. For example, if a plot shows a clear U-shape, then the variable may have near-zero linear correlation with the dependent variable, and may not benefit the model despite the strong relationship. In part (e), you will explore ways of modifying such variables to make them more useful to the model.

(c) **Mutual Information (5 points)** In the review slides from class, in the section on Information Theory, Mutual Information is defined. Mutual information is commonly used for feature selection, where only the most important features are included in a model. Implement a function that calculates the mutual information between each independent variable and the dependent variable (the survival column). Disregard missing values when computing mutual information. For numeric variables, implement a function that discretizes them into 10 equal density bins (not equal width). So, each bin will have approximately the same number of samples. Disregard missing values when determining discretization cutpoints. Again, only use training data for this task.

In your report, list the independent variables and their mutual information scores sorted in descending order by mutual information. Several of the categorical variables are at the top

because they contain information specific to passengers. We won't include these in predictive models.

(d) **Missing Values (5 points)** Create separate **X** matrices for both the training and testing data out of the columns {*sex, pclass, fare, embarked, parch, sibsp, age*}. Use dummy variables to encode categorical variables. For the two missing values of *embarked*, substitute the value 'S'. (Had *embarked* been missing more values, it might have been worth trying to treat missing values as just another category.) For numeric variables, use the raw numeric value (do not discretize). For the one missing *fare*, substitute the value 33.2955. Now *age* is the only remaining variable with missing values. Represent missing *age* values with `NaN`.

Normalize the training **X** matrix by subtracting the mean of each column and dividing each column by its standard deviation. Normalize the testing **X** matrix by subtracting the means you calculated from the training matrix and dividing each column by the standard deviations you calculated from the training data columns. This is best practice.

For now, logistic regression models will be trained using MATLAB's `glmfit` function. Note that it automatically adds a column of ones to the beginning of your **X** matrix. Also note that it automatically disregards any sample with a `NaN`. Applying models to make predictions will be done with MATLAB's `glmval` function.

In this section, you will compare two methods of handling missing values: multiple models and substituting values.

Multiple Models: Train a model using the training **X** matrix you've prepared. The model will not be able make predictions when *age* is missing. Train another model for those cases, by leaving out the *age* column of **X**. Predictions will be made using the first model when possible and the second model when needed. Report the training accuracy and testing accuracy using this method.

Substituting Values: Create new training and testing **X** matrices, this time replacing missing *age* values with the average *age* from the training data when it is available. You must normalize both new matrices using the technique already described.

Train a model on the modified training data. Report the training accuracy and testing accuracy using this method. Make a statement about which method for handling missing data worked better.

(e) **Basis Expansion (5 points)** For this, use the **X** matrices that still have `NaN` values when *age* is missing.

Append columns with the square-root of the numeric independent variables (*pclass, age, sibsp, parch, fare*) to both the training and testing **X** matrices.

Discretize the 5 numeric variables from the training data into at most 10 equal density bins, just as you did when calculating mutual information. Keep the discretization cutpoints and use them to discretize the same variables from the testing data. (Testing data should not be visible to any operations prior to testing.) Treat the discretized numeric variables as categorical variables, encode them with dummy variables and append them to the **X** matrices.

Append interaction variables for all pairwise combinations of columns currently in the **X** matrices. An interaction variable is an element-wise product of two original variables:

`X(:,i) .* X(:,j)` for all i and j, such that $j < i$. Now go back and delete any columns that have less than 2 distinct non-NaN values in the training data.

Re-normalize both the training and testing matrices as described, using the means and standard deviations from the columns of the training matrix. In your report, record the number of columns now in $\mathbf{X}$.

(f) **Sequential Feature Selection (5 points)** There are now many columns in $\mathbf{X}$. Many of them contain substantially the same information, making mutual information less effective for feature selection, since it considers each feature in isolation. So, the top 10 or 20 features may be redundant. For this task, perform feature selection using a wrapper method: greedy forward selection. Iteratively build up a set of 10 features. At each iteration add one column to the feature set. Choose the column that works best in conjunction with the existing, already-chosen features. By "works the best," we mean most improves training accuracy of a linear regression model. So, the outer loop will run 10 times, adding a new feature each time. The inner loop will train a new model as many times as there are columns, each time storing the training accuracy. Don't worry about MATLAB warnings during model training. After each iteration of the outer loop (after adding each new feature), store the training and testing accuracy. In your report, show a plot with both the training accuracy and testing accuracy as a function of the number of forward-selected features. Did this method of feature selection work well? Why or why not? What appears to be the optimal number of features to use?

(g) **Batch Gradient Descent (5 points)** The choice of data for this task and the next one is yours. You may want to continue using the the training matrix from the last task with only the top few features and with missing values removed. You will need to add a column of ones to the beginning of the data when not using `glmfit`.

Implement your own function that performs logistic regression using batch gradient descent. Compare the training accuracy of the resulting model with the training accuracy of `glmfit`, as a function of the number of training iterations. Do this several times with different step sizes. Describe the results in your report. Is it converging in a stable way? How many iterations are required to get similar accuracy as `glmfit`?

(h) **Newton's Method (5 points)** Implement your own function that performs logistic regression using Newton's Method. Compare the training accuracy of the resulting model with the training accuracy of `glmfit`, as a function of the number of training iterations. Describe the results in your report. How many iterations are required to get similar accuracy as `glmfit`?

Submission Instructions: You need to provide the followings:

- Provide your answers to problems 1-4 in PDF file, named as `CSCI567_hw2_fall15.pdf`. You need to submit the homework in both hard copy (at CS Front Desk with a box labeled as CSCI567-homework by 5pm of the deadline date) and electronic version as pdf file on Blackboard. If you choose handwriting instead of typing all the answers, you will get 40% points deducted.
- Submit ALL the code and report via Blackboard. The only acceptable language is MATLAB. For your program, you MUST include the main function called `CSCI567_hw2_fall15.m` in the root of your folder. After running this main file, your program should be able to generate all of the results needed for this programming assignment, either as plots or console outputs. You can have multiple files (i.e your sub-functions), however, the only requirement is that once we unzip your folder and execute your main file, your program should execute correctly. Please double-check your program before submitting. You should only submit one `.zip` file. No other formats are allowed except `.zip` file. Also, please name it as `[lastname]_[firstname]_hw2_fall15.zip`.

Collaboration: You may collaborate. However, collaboration has to be limited to discussion only and you need to write your own solution and submit separately. You also need to list with whom you have discussed.