Lab 7
Numeric Estimation with Linear Regression

Overview: Having experimented with several classification learning and association learning algorithms, we turn our attention to the second-to-last machine learning technique that we'll cover. What do we do when the class/output attribute is numeric? The simplest approach is to use a classic *linear regression* model. And we will look briefly at a more complex option as well, called *Support Vector Regression*, a variation of the *Support Vector Machines* classification technique.

1. Regression

Regression is the easiest technique to use, but is also probably the least powerful – funny how that always goes hand in hand. This model can be as easy as one input variable and one output variable (called a Scatter diagram in Excel). Of course, it can get more complex than that, including dozens of input variables. In effect, regression models all fit the same general pattern. There are a number of independent variables, which, when taken together, produce a result — a dependent variable. In data mining terminology, this is the class, or output attribute. The regression model is then used to predict the result of an unknown dependent variable, given the values of the independent variables.

Everyone has probably used or seen a regression model before, maybe even mentally creating a regression model. A classic example is pricing an expensive potential purchase, such as a house. The price of the house (the dependent variable) is the result of many independent variables — the square footage of the house, the size of the lot, whether granite is in the kitchen, bathrooms are upgraded, etc. So, whenever it comes time for you to buy or sell a house, you'll likely create a regression model to price the house. You'll create the model based on other comparable houses in the neighborhood and what they sold for (the model), then put the values of your own house into this model to produce an expected price.

Let's continue this example of a house price-based regression model, and create some real data to examine.

House size (square feet)	Lot size	Bedrooms	Granite	Upgraded bathroom?	Price
3529	9191	6	0	0	$205,000
3247	10061	5	1	1	$224,900
4032	10150	5	0	1	$197,900
2397	14156	4	1	0	$189,900
2200	9600	4	0	1`	$195,000
3536	19994	6	1	1	$325,000
2983	9365	5	0	1	$230,000
3198	9669	5	1	1	????

A quick warning before we continue – this little lab introduction to regression barely scratches the surface, and that scratch is really even barely noticeable. There are entire college semester courses that will teach you more about regression models than you probably even want to know. But this "scratch" will get you acquainted with the concept and will suffice for our introduction to numeric estimation data mining. If you have continued interest in regression models and all the statistical details that go into them, research the following terms with your favorite search engine: *least squares*, *homoscedasticity*, *normal distribution*, *White tests*, *Lilliefors tests*, *R-squared*, and *p-values*.

2. Creating a Regression Model in Weka

a. Save the file `houses.arff` (from Box) to your USB drive. This is another UCI dataset representing homes in the Boston area. Open the file in Weka. As we do every time we study a new dataset, take some time to explore this one. Really take some time! This isn't a race – don't rush. Your retention of the concepts from lab will be enhanced by being deliberate in your investigations.

b. Go to the Classify tab and click **Choose**. In the "functions" group, choose `LinearRegression`.

c. Use the training set as the test option. Make sure that `sellingPrice` is the class attribute. Then click **Start**.

d. Note the difference in the display from previous Weka experiments with classification learning. What is the type of model produced? Can you spot it? It looks like this:

```
sellingPrice = (-26.6882   * houseSize) +
               (7.0551     * lotSize) +
               (43166.0767 * bedrooms) +
               (42292.0901 * bathroom)
               - 21661.1208
```

Consider the test instance given on the previous page. How would we predict the price of the house? Make sure this calculation makes sense to you:

```
sellingPrice = (-26.6882   * 3198) +
               (7.0551     * 9669) +
               (43166.0767 * 5) +
               (42292.0901 * 1)
               - 21661.1208

sellingPrice = 219,328
```

e. Instead of an accuracy percentage, there is a *correlation coefficient*. (Also a *root mean squared error*, but we'll stick with the correlation coefficient for now.) What is its value? Is this good or bad? If you're not sure, review the lecture notes, or Google it.

3. Interpreting the Results

As we have discussed, data mining isn't just about outputting a single number – it's about identifying underlying patterns and rules. It's not used only to produce an absolute number but rather to create a model that lets you detect patterns, predict output, and come up with conclusions backed by the data. Let's take another step and interpret the patterns and conclusions that our model tells us, besides just a strict house value:

- **Granite doesn't matter**— Weka will only use columns that statistically contribute to the accuracy of the model (measured in *R-squared*, but beyond the scope of this class). It will throw out and ignore columns that don't help in creating a good model. So this regression model is telling us that granite in your kitchen doesn't affect the house's value.

a. In the `LinearRegression` classifier settings, choose "No attribute selection" as the `attributeSelectionMethod`, and run the algorithm again. What is the most obvious thing you notice in the output model? Re-set the parameter to "M5 method" before continuing.

- **Bathrooms do matter**— Since we use a simple 0 or 1 value for an upgraded bathroom, we can use the coefficient from the regression model to determine the value of an upgraded bathroom on the house value. The model tells us it adds $42,292 to the house value.

- **Bigger houses reduce the value**— Is Weka is telling us that the bigger our house is, the lower the selling price? This can be seen by the negative coefficient in front of the `houseSize` variable. The model is telling us that every additional square foot of the house reduces its price by $26? That doesn't make any sense at all. This is America! Bigger is better! How should we interpret this? This is a good example of garbage in, garbage out. The house size, unfortunately, isn't an independent variable *because it's related to the bedrooms variable*, which makes sense, since bigger houses tend to have more bedrooms. So our model isn't perfect. But we can fix this.

b. On the Preprocess tab, remove the `houseSize` column and create another model. How does it affect the price of the sample house? In what way does this new model make more sense? (Amended house value: $217,894)

c. Before leaving this dataset, one more experiment. As we discussed in lecture, there is a simpler, univariate version of the linear regression algorithm that is analogous to the *1R* classifier. Return to the original dataset (using **Undo**) and choose `SimpleLinearRegression`. Which attribute seems to be the most significant factor in the price of a house (based on this very tiny dataset)? What happens to the correlation coefficient? (Amended house value: $204,334)

4. A More Complex Dataset

As a note to all the statisticians in the class – and I know that there are some! – keep in mind that the previous model breaks several requirements of a "proper" linear regression model, since every column isn't truly independent, and there aren't enough rows of data to produce a valid model. But since the primary purpose of this lab is to introduce the concepts of linear regression as a data mining tool, oversimplifying the example data is okay.

To take this simple example to the next level, let's take a look at a data file provided by Weka as a regression example. Theoretically, this should be much more complex than our simple example with seven houses. This sample data file attempts to create a regression model to predict the miles per gallon (MPG) for a car based on several attributes of the car. (This data is from 1970 to 1982, so keep that in mind.) The model includes these possible attributes of the car: cylinders, displacement, horsepower, weight, acceleration, model year, origin, and car make. Further, this data set has 398 rows of data and meets many of the statistical requirements that our earlier house price model didn't. In theory, this should be a much more complex regression model, and perhaps Weka might have a hard time creating a model with this much data (though I'm sure you can predict at this point that Weka will handle this just fine).

a. Save the file `autoMPG.arff` to your USB drive. Load it into Weka and run the `LinearRegression` algorithm with the default settings. The steps are the same as before so I won't repeat them. You should get this for the output model:

```
class (aka MPG) =

     -2.2744 * cylinders=6,3,5,4 +
     -4.4421 * cylinders=3,5,4 +
      6.74   * cylinders=5,4 +
      0.012  * displacement +
     -0.0359 * horsepower +
     -0.0056 * weight +
      1.6184 * model=75,71,76,74,77,78,79,81,82,80 +
      1.8307 * model=77,78,79,81,82,80 +
      1.8958 * model=79,81,82,80 +
      1.7754 * model=81,82,80 +
      1.167  * model=82,80 +
      1.2522 * model=80 +
      2.1363 * origin=2,3 +
      37.9165
```

As you see, Weka flies through the model in less than a second. So it's not a problem, computationally, to create a powerful regression model from a lot of data. This model may also appear much more complex than the house data, but it isn't. For example, the first line of the regression model, `-2.2744 * cylinders=6,3,5,4` means that if the car has six cylinders, you would place a 1 in this column, and if it has eight cylinders, you would place a 0. Make sure you understand this notation. *This is one technique for handling nominal data values in linear regression learning*.

b. Does the learning algorithm consider any of the attributes not significant enough to include?

c. Let's take one example row from the data set (row 10) and plug those numbers into the regression model, and see if the output from the model approximates the output that was given to us in the data set:

```
data = 8,390,190,3850,8.5,70,1,15

class (aka MPG) =

     -2.2744 * 0 +
     -4.4421 * 0 +
      6.74   * 0 +
      0.012  * 390 +
     -0.0359 * 190 +
     -0.0056 * 3850 +
      1.6184 * 0 +
      1.8307 * 0 +
      1.8958 * 0 +
      1.7754 * 0 +
      1.167  * 0 +
      1.2522 * 0 +
      2.1363 * 0 +
      37.9165

Expected Value = 15 mpg
Regression Model Output = 14.2 mpg
```

Make sure this calculation makes sense to you, and then try it for practice with another randomly selected row of training data. (Don't just blow by this, please!)

d. As noted in lecture, Weka can build a *regression tree* along with a linear regression model. This uses the `M5P` algorithm in the "trees" group. Set `buildRegressionTree` to true. Run the algorithm. Right-click on the output in the Results List and visualize the tree, as we did in an earlier lab. Reading the tree is mostly self-explanatory. When there is a choice of values (such as for the attribute `model`) use 0 if the test value is <u>not</u> on the list, and 1 if it is. When you get to a leaf node, refer back to the Classifier Output panel to get the actual value. How is this different from a tree used for classification instead of numeric estimation? What mpg prediction is given for the instance analyzed in step (c) above?

e. Finally, set the algorithm back to its default – with `buildRegressionTree=false`. This will build a *model tree* instead. Run the algorithm and view the tree. How is it different from a regression tree? Make sure you understand before finishing up. You don't have to run the test instances through the tree as before, but you should be confident that you <u>could</u> if asked to.

5. Revisiting the Crime Dataset

Let's reconsider the Crime dataset from Lab 6 again. As you no doubt recall, one of the first tasks in that lab was to discretize all of the attributes -- including the class attribute (`violentPerPop`). Why did we do that? Since we have moved on from classification algorithms to numeric estimation algorithms, we can reconsider this dataset because it has a numeric class attribute.

a. Open the *original* dataset from Lab 6. It should be obvious to you why we are not using the discretized version. (Is it?) Hopefully you have it saved under a logical name. Re-familiarize yourself with the attributes. Note for the moment that there is still one nominal attribute (`State`).

b. Go to the Classify tab and choose the `LinearRegression` algorithm again. This is a large dataset so select Percent split 66% for the testing option. Run the algorithm. The resulting linear regression equation is kind of a mess, but make sure you can make sense of it. In lecture we discussed the ability of the linear regression algorithm to handle nominal values. (We also saw that earlier in lab today.) Do you see how the `State` attribute has been incorporated? Make sure you can interpret the entire equation before moving on.

c. What is the correlation coefficient for this equation?

d. Now go back to the Preprocess tab and remove the `State` attribute. We've learned our lesson (again) that nominal input attributes can be handled, so from here on we'll leave `State` out to simplify our results.

e. Run the linear regression algorithm again. How does the correlation coefficient compare with `State` left out? Does `State` appear to be predictive or not?

As discussed in lecture, there are more complex versions of regression available. One of them is *Support Vector Regression* (a variation on the classification algorithm *Support Vector Machines*). Let's try this out.

f. Again in the Weka `functions` group, select `SMOreg`. Using the default settings and the same test options as before, click **Start**. This might take 20-30 seconds to complete training due to the complexity of the algorithm.

g. Take a look at the regression equation produced. Does it make sense?

h. Which model -- linear regression or support vector regression -- performed better in this experiment? How can you tell?

There is nothing to hand in for today’s lab. Make sure you understand all of the concepts, and check in with me before you leave. Feel free to stay in your subgroup for a while to work on your term project. Perhaps some of today's techniques could be part of your experimentation? Have a great day!