Assignment 3

Logistics

You must create an IPython notebook for the problem. The notebook should clearly state the names of the group members. Please name the notebook *"HW3_Lastnames_of_group_members"*. Upload the notebook on Canvas.

The assignments are due on **April 18th, before the beginning of class**.

Deliverables. The notebook should be split by sections, one for each question. Each section should contain the following parts:

1. A heading for each question (use a Markdown cell).
2. For each question, write a short description of your approach (another Markdown cell).
3. Write the code, and show the results or plots.

Regression

We will try to predict the percentage of votes won by Bill Clinton in the 1992 election.

[**Q1 8 points**] **Read the data.** Read in the data from `http://www.stat.ufl.edu/~winner/data/clinton1.dat`. Note that this is **NOT** a CSV file. The description of the data fields are present in `http://www.stat.ufl.edu/~winner/data/clinton1.txt`. You will have to figure out the right parameters for the `read_csv` command yourself (e.g., the regular expression for the delimiter, and the names of the columns).

[**Q2 8 points**] **Predict percentVoting on all regressors.** One of the fields in the data is "Percent voting for Clinton in 1992"; let's call it `percentVoting`. Regress `percentVoting` on all available regressors. Which two regressors have the worst p-values?

[**Q3 8 points**] **A formula-creating function.** Write a function called `formula` that takes a list of regressors as input, and outputs the formula for regressing `percentVoting` on these regressors.

For example, `formula(['age', 'savings'])` should output `percentVoting ~ age + savings`.

[**Q4 8 points**] **R-squared computation function.** Write a function called `rsquared` that, given a list of regressors, returns the R-squared corresponding to a regression of `percentVoting` with respect to the list of regressors. You can use the `formula` function you wrote above.

[**Q5 8 points**] **Find the next best regressor.** Write a function called `best_next_regressor` that, given a list of regressors (called `current_regressor_list`) and the list of all possible regressors (called `all_regressors_list`), picks one regressor from `all_regressors_list` that, when combined with the regressors in `current_regressor_list`, yields the highest R-square. The function should return this regressor, and the corresponding R-square.

[**Q6 8 points**] **Pick the k best regressors.** Write a function called `best_regressors` that takes two arguments:

- `num_regressors`, which is the number of regressors we want, and
- `all_regressors_list`, which again is the list of all possible regressors.

The function should compute the best `num_regressors` regressors by picking them one at a time using `best_next_regressor` that you wrote above. The function should return a tuple of the following items:

- a list of the best regressors, in the order in which they were picked, and
- the corresponding R-square values.

In other words, you want a list of the results of calling `best_next_regressor`.

[**Q7 8 points**] **Order all regressors in the best order.** Run `best_regressors` to output all available regressors in sequence. In other words, the first item in the result should be the best single regressor, the second item should be the next best regressor that combined with the first regressor gives the best R-square, and so on.

This idea of ordering the regressors in this particular order is called **forward selection**, and is one way of selecting only a few regressors when given a large set of regressors.

[**Q8 8 points**] **Plot the R-squared values as regressors are added in the best order.**

K-Nearest Neighbors Classification

We will work with the same dataset as above.

[**Q9 8 points**] **Create a classification target.** Attach a new column, called `target` to the DataFrame, whose value for each row is either 1 (if `percentVoting` is at least 40.0), or 0 (`percentVoting` is less than 40.0).

[**Q10 8 points**] **Create training and test sets.** Create design matrices for predicting the `target` using just the two features `poverty` and `popdensity`. Split this into design matrices for training and testing datasets, with 70% data being used for training, and 30% for testing.

[**Q11 12 points**] **Accuracy of classification.** Use a K-nearest neighbors classifier to predict the target using the two given features, using $1, 3, 5, 10, 20, 50, 100,$ and 1000 nearest neighbors respectively. Print out the training and test accuracy for each of these.

[**Q12 8 points**] **Plots and reasons.** Show a scatter plot of `popdensity` versus `poverty`, and color the points according to the `target` (i.e., each point in the scatter plot represents a county, and the color of the county depends on the value of `target` for that county). Can you interpret the plot? What types of counties voted for Bill Clinton?