In-Class Programming Task 10

Math 253: Statistical Computing & Machine Learning

Implementing logistic regression

In this activity, you will work with the logistic function and likelihood. For the sake of definiteness, you'll use the `Default` data from the `ISLR` package, but the programming involved will show techniques for writing functions that work with any data table.

The logistic function

The logistic function is defined as $\mathcal{L}(x) = \frac{e^x}{1+e^x}$. Note that the function $\mathcal{L}(x)$ takes a scalar x as an input.

- Write a function `logistic()` that returns the value of the logistic function for any numerical vector `x`. Remember that functions like `exp()` and `+` are vectorized in R, so writing `logistic()` for a scalar x will automatically work on a vector `x`.

Linear combinations

In logistic regression, a linear combination of variables is taken and passed through the logistic function to produce a probability.

- Write a function `linear_combine()` that takes two arguments:
  - `data=` specifying a data table
  - `coefs=` specifying a set of coefficients to use in the linear combination.

The `coefs` argument will have a very specific form: a numerical vector with named components. For example, `linear_combine()` will be invoked with a command like this:

```
linear_combine(data = Default,
               coefs = c(intercept = 3, balance = 0.00001, income = -0.0002))
```

The `linear_combine()` function will multiply each of the named variables in `data` by the corresponding coefficient, then add in the value of the intercept. The returned value will be the vector that is the specified linear combination.

At the heart of your `linear_combine()` will be a loop like this:

```
result <- 0
for (nm in names(coefs)) {
  if (nm == "intercept") {
    result <- result + coefs[[nm]]
  } else {
    result <- result + coefs[[nm]] * data[[nm]]
  }
}
```

Arrange your function to call `stop()` if the variable names in the `coef` vector don't exist in `data`.

Probabilities

Applying the logistic function to the output of `linear_combine()` produces a number between zero and one. You can interpret this probability as the conditional probability of the outcome given the values of the data.

- Write a function `LL_logistic()` that computes the log-likelihood of the observed result given a model in the form of `coefs`. It will have these arguments:
  - `data=` a data table for training
  - `coefs=` a proposed vector of coefficients
  - `outcome=` a `TRUE` or `FALSE` vector giving the observed outcome in the training data.

  Your function should

1. use `linear_combine()` on `data` and `coefs` to produce a number for each case in `data`.
2. put the result of (1) through `logistic()`. The output will be a probability for each case in `data`, which I'll write here as p_i.
3. set the likelihood for each case in `data` to be p_i if the result is `TRUE`, otherwise set the likelihood to be $1 - p_i$.
4. combine the likelihoods of the individual cases into an overall log likelihood.

   Here's a quick test for your function:

```
LL_logistic(data=Default,
            coefs = c(intercept = 1, income = -0.0001),
            outcome = Default$default == "Yes")
```

```
## [1] -2425.733
```

Optimize

Use `optim()` to find the maximum likelihood coefficients (using just the intercept and `income`) to predict `default`.

Put your results in an object called `best_coefs`.

Compare your result to the output of

```
glm(default == "Yes" ~ income, data=Default, family="binomial")
```

Above and beyond

It seems odd to have to specify both `data=Default` and `outcome=Default$default == "yes"` in the above command. R provides a way to avoid this, so that the `outcome` argument can be simply `default == "Yes"`. To do this, write your `LL_logistic()` this way

```
LL_Logistic <- function(data = NULL, coefs = NULL, outcome) {
  outcome_statement <- substitute(outcome)
  outcome <- eval(outcome_statement, envir = data)
  # Now, "outcome" will be the value of the statement when
  # applied to the values in data

  # ... Continue with your calculations
}
```