**Recipe Name:** read\_matrix

**Inputs:**

*input\_string*, a string with data

**Outputs:**

*data\_matrix*, a matrix filled with data from *input\_string*

**Steps:**

1. Let *sequence\_matrix* be an empty sequence
2. Let ­*inner\_sequence* be an empty sequence
3. For each line in *input\_string*, do the following:
   1. *inner\_sequence* 🡨 a sequence formed by taking each value from the line, with separation between elements occurring by comma and whitespace
   2. Append *inner\_sequence* to the end of *sequence\_matrix*
4. *data\_matrix* 🡨 a matrix formed by converting *sequence\_matrix* to a matrix, where each element of *sequence\_matrix* is a row in *data\_matrix*
5. Return *data\_matrix*

**Recipe Name:** generate\_predictions

**Inputs:**

*inputs*, an n \* m matrix of variables

*weights*, an m \* 1 matrix representing the linear model

**Outputs:**

an n\*1 matrix of values predicted by the model using the data *inputs*

**Steps:**

1. Return the result of using matrix multiplication to multiply *inputs* by *weights*

**Recipe Name:** prediction\_error

**Inputs:**

*model*, a linear model using an m \* 1 matrix of weights

*inputs*, an n \* m matrix of variables

*actual*, an n \* 1 matrix of the actual result of *inputs*

**Outputs:**

The mean squared error between the values predicted by *model* and *actual*

**Steps:**

1. *generated* 🡨 generate\_predictions(*model*, *inputs*)
2. *actual\_seq* 🡨 *actual* converted to a sequence by setting each row of *actual* to an individual element of *actual\_seq*
3. *generated\_seq* 🡨 *generated* converted to a sequence by setting each row of *actual* to an individual element of *generated\_seq*
4. Return mse(*actual\_seq*, *generated\_seq*)

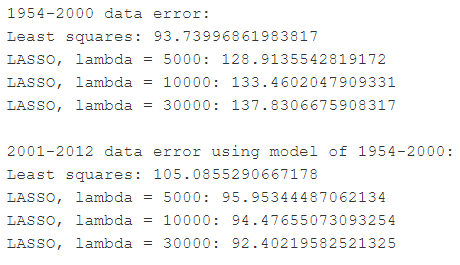
**Derivation**

**Codeskulptor Link**

https://py3.codeskulptor.org/#user305\_MI244HQ3wp\_27.py

**Baseball Performance Prediction Errors**

500 iterations,



**Discussion**

1. This expression is true because of the properties of transposing a composition of matrices and because each matrix w, X, and y had dimensions that worked to allow for the composition to take place.
2. An increase in means that the 1-norm of the weights must decrease to match, which means the summation of the absolute value of each weight must decrease in turn.
3. From the training data, least squares produced the lowest MSE. When used on the test 2001-2012 data, it was actually the LASSO with the highest lambda value that minimized MSE. It seems like LASSO is better at predicting future data than the actual values of the training data.
4. It seems like Hits and Earned Runs are less important, since the weights on those statistics are significantly more deviated from 0 than the other stats.