Predict Max GB for AIS: RMSE Regression, Aug. 2016

loadData

In [19]:

```
import graphlab
```

In [20]:

```
df = graphlab.SFrame('data.csv')
df_no_missing = df.dropna()
```

```
Read 52 lines. Lines per second: 3894.84

Finished parsing file /home/ubuntu/data.csv

Parsing completed. Parsed 52 lines in 0.015529 secs.

Finished parsing file /home/ubuntu/data.csv

Parsing completed. Parsed 52 lines in 0.014141 secs.

------------------------------------------------------
Inferred types from first 100 line(s) of file as
column_type_hints=[str,str,str,str,int]
If parsing fails due to incorrect types, you can correct
the inferred type list above and pass it to read_csv in
the column_type_hints argument
------------------------------------------------------
```

In [21]:

```
df_no_missing.head(1)
```

Out[21]:

TABLE_NAME	ALL	ACTIVE	Var	MAX GB, Including Index
ALA_BASE	2,635,755	2,635,755		1

[1 rows x 5 columns]

buildModel

In [22]:

```
train_data, test_data = df_no_missing.random_split(.8, seed=0)
```

In [23]:

```
reg_model = graphlab.linear_regression.create(train_data, target='MAX GB, Including Index', features=['ALL',
'ACTIVE'])
```

```
WARNING: The number of feature dimensions in this problem is very large in comparison with the number of exa
mples. Unless an appropriate regularization value is set, this model may not provide accurate predictions fo
r a validation/test set.

Linear regression:

--------------------------------------------------------

Number of examples          : 37

Number of features          : 2

Number of unpacked features : 2

Number of coefficients    : 73

Starting Newton Method

--------------------------------------------------------

+-----------+----------+--------------+--------------------+---------------+
```

```
| Iteration | Passes   | Elapsed Time | Training-max_error | Training-rmse |
+-----------+----------+--------------+--------------------+---------------+
| 1         | 2        | 0.000219     | 0.257516           | 0.048539      |
+-----------+----------+--------------+--------------------+---------------+
SUCCESS: Optimal solution found.
```

In [24]:

```
reg_model.get('coefficients').print_rows(num_rows=3, num_columns=3)
```

```
+-------------+--------+-----------------+-----+
|     name    | index  |      value      | ... |
+-------------+--------+-----------------+-----+
| (intercept) |  None  |  1.25751618215  | ... |
|     ALL     | 53,958 | -0.128740223571 | ... |
|     ALL     |  372   | -0.128740223545 | ... |
+-------------+--------+-----------------+-----+
[73 rows x 4 columns]
```

In [25]:

```
print reg_model.evaluate(test_data)
```

```
{'max_error': 13.742483817852367, 'rmse': 5.6269219787719615}
```

applyModel

In [26]:

```
def maxGB(x):
    maxGB = df[df['TABLE_NAME']==x]
    return reg_model.predict(maxGB)
```

In [27]:

```
maxGB('MON_BASE')
```

Out[27]:

```
dtype: float
Rows: 1
[290.9597929646762]
```

writeUp

This is a Root Mean Square Error (RMSE) regression model that results in predicted values close to the observed data. The RMSE value in the model results is an absolute measure of fit. One pitfall of RMSE is that it can incorporate too many variables, however this regression model has a Validation-rmse of 5.6269219787719615 and lower values indicate a better fit.