Logistic Regression

In this notebook you will use GPU-accelerated logistic regression to predict infection risk based on features of our population members.

Objectives

By the time you complete this notebook you will be able to:

- Use GPU-accelerated logistic regression

Imports

In [ ]:
```
import cudf
import cuml

import cupy as cp
```

Load Data

In [ ]:
```
gdf = cudf.read_csv('./data/pop_2-05.csv', usecols=['age', 'sex', 'infect
```

In [ ]:
```
gdf.dtypes
```

In [ ]:
```
gdf.shape
```

In [ ]:
```
gdf.head()
```

Logistic Regression

Logistic regression can be used to estimate the probability of an outcome as a function of some (assumed independent) inputs. In our case, we would like to estimate infection risk based on population members' age and sex.

Here we create a cuML logistic regression instance `logreg`:

In [ ]:
```
logreg = cuml.LogisticRegression()
```

Exercise: Regress Infected Status

The `logreg.fit` method takes 2 arguments: the model's independent variables *X*, and the dependent variable *y*. Fit the `logreg` model using the `gdf` columns `age` and `sex` as *X* and the `infected` column as *y*.

In [ ]:

Solution

In [1]:
```
%load solutions/regress_infected
```

Viewing the Regression

After fitting the model, we could use `logreg.predict` to estimate whether someone has more than a 50% chance to be infected, but since the virus has low prevalence in the population (around 1-2%, in this dataset), individual probabilities of infection are well below 50% and the model should correctly predict that no one is individually likely to have the infection.

However, we also have access to the model coefficients at `logreg.coef_` as well as the intercept at `logreg.intercept_`. Both of these values are cuDF Series:

In [ ]:
```
type(logreg.coef_)
```

In [ ]:
```
type(logreg.intercept_)
```

Here we view these values. Notice that changing sex from 0 to 1 has the same effect via the coefficients as changing the age by ~48 years.

In [ ]:
```
logreg_coef = logreg.coef_
logreg_int = logreg.intercept_

print("Coefficients: [age, sex]")
print([logreg_coef[0], logreg_coef[1]])

print("Intercept:")
print(logreg_int[0])
```

Estimate Probability of Infection

As with all logistic regressions, the coefficients allow us to calculate the logit for each; from that, we can calculate the estimated percentage risk of infection.

In [ ]:
```
class_probs = logreg.predict_proba(gdf[['age', 'sex']])
class_probs
```

Remembering that a 1 indicates 'infected', we assign that class' probability to a new column in the original dataframe:

In [ ]:
```
gdf['risk'] = class_probs[1]
```

Looking at the original records with their new estimated risks, we can see how estimated risk varies across individuals.

In [ ]:
```
gdf.take(cp.random.choice(gdf.shape[0], size=5, replace=False))
```

Exercise: Show Infection Prevalence is Related to Age

The positive coefficient on age suggests that the virus is more prevalent in older people, even when controlling for sex.

For this exercise, show that infection prevalence has some relationship to age by printing the mean `infected` values for the oldest and youngest members of the population when grouped by age:

In [ ]:
```

```

Solution

In [ ]:
```
%load solutions/risk_by_age
```

Exercise: Show Infection Prevalence is Related to Sex

Similarly, the positive coefficient on sex suggests that the virus is more prevalent in people with sex = 1 (females), even when controlling for age.

For this exercise, show that infection prevalence has some relationship to sex by printing the mean `infected` values for the population when grouped by sex:

In [ ]:
```

```

Solution

In [2]:
```
%load solutions/risk_by_sex
```

Making Predictions with Separate Training and Test Data

cuML gives us a simple method for producing paired training/testing data:

In [ ]:
```
X_train, X_test, y_train, y_test  = cuml.train_test_split(gdf[['age', 'se
```

Exercise: Fit Logistic Regression Model Using Training Data

For this exercise, create a new logistic regression model `logreg`, and fit it with the *X* and *y* training data just created.

In [ ]:

Solution

In [ ]:
```
%load solutions/fit_training
```

Use Test Data to Validate Model

We can now use the same procedure as above to predict infection risk using the test data:

In [ ]:
```
y_test_pred = logreg.predict_proba(X_test, convert_dtype=True)[1]
y_test_pred.index = X_test.index
y_test_pred
```

As we saw before, very few people are actually infected in the population, even among the highest-risk groups. As a simple way to check our model, we split the test set into above-average predicted risk and below-average predicted risk, then observe that the prevalence of infections correlates closely to those predicted risks.

In [ ]:
```
test_results = cudf.DataFrame()
test_results['age'] = X_test['age']
test_results['sex'] = X_test['sex']
test_results['infected'] = y_test
test_results['predicted_risk'] = y_test_pred

test_results['high_risk'] = test_results['predicted_risk'] > test_results

risk_groups = test_results.groupby('high_risk')
risk_groups.mean()
```

Finally, in a few milliseconds, we can do a two-tier analysis by sex and age:

In [ ]:
```
%%time
s_groups = test_results[['sex', 'age', 'infected', 'predicted_risk']].gro
s_groups.mean()
```

Please Restart the Kernel

In [ ]:
```
import IPython
app = IPython.Application.instance()
app.kernel.do_shutdown(True)
```

Next

In the next notebook, you will use GPU-accelerated k-nearest-neighbors algorithm to locate the nearest road nodes to each hospital.