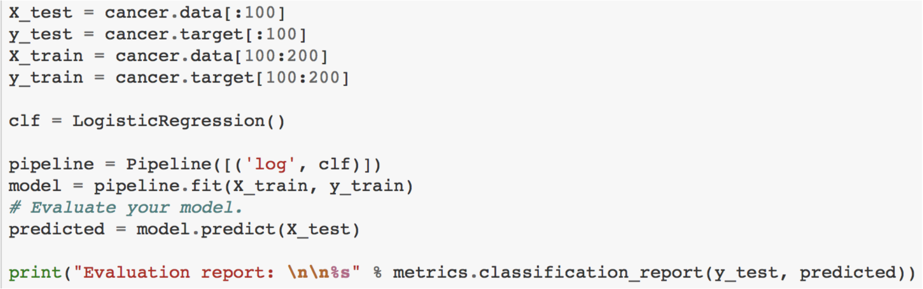
Dynamic Learning Using a Cancer Dataset

Ziyuan Shen

January 8, 2019

1 Deploying a Sk-learn Breast Cancer Model in IBM Cloud Using a Notebook

From my last report, a ML (machine learning) model can be created in Watson Studio either directly [1] or using a python notebook [2]. A notebook is highly preferred as much more flexible algorithms can be incorporated. A pipeline model using scikit-learn library is required. By using the breast cancer wisconsin dataset (This dataset has 30 features and 569 samples in total) built in scikit-learn, I created a pipeline model with the logistic regression algorithm and deployed the model successfully (see Figure 1 and Figure 2) .



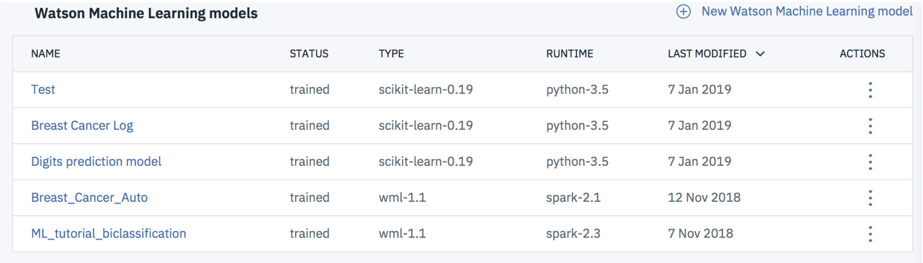
```
X_test = cancer.data[:100]
y_test = cancer.target[:100]
X_train = cancer.data[100:200]
y_train = cancer.target[100:200]

clf = LogisticRegression()

pipeline = Pipeline([('log', clf)])
model = pipeline.fit(X_train, y_train)
# Evaluate your model.
predicted = model.predict(X_test)

print("Evaluation report: \n\n%s" % metrics.classification_report(y_test, predicted))
```

Figure 1: Create a pipeline model for the cancer dataset.

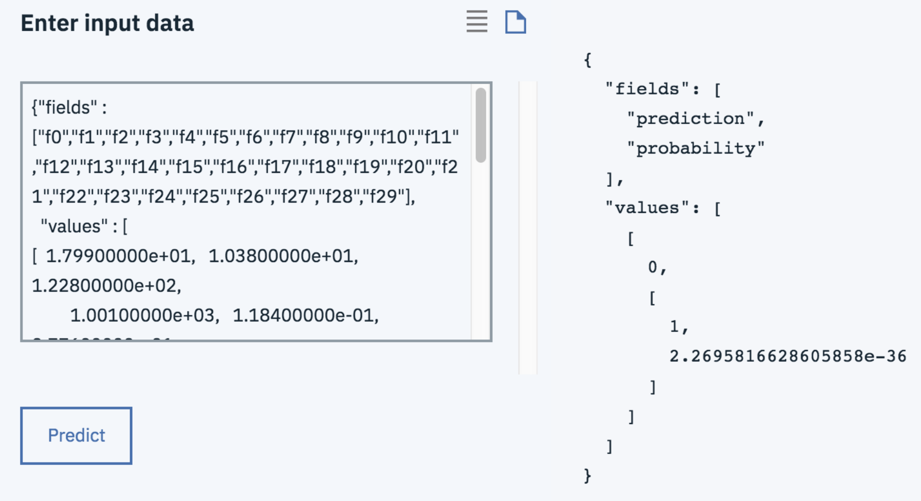


Watson Machine Learning models

NAME	STATUS	TYPE	RUNTIME	LAST MODIFIED	ACTIONS
Test	trained	scikit-learn-0.19	python-3.5	7 Jan 2019	⋮
Breast Cancer Log	trained	scikit-learn-0.19	python-3.5	7 Jan 2019	⋮
Digits prediction model	trained	scikit-learn-0.19	python-3.5	7 Jan 2019	⋮
Breast_Cancer_Auto	trained	wml-1.1	spark-2.1	12 Nov 2018	⋮
ML_tutorial_biclassification	trained	wml-1.1	spark-2.3	7 Nov 2018	⋮

Figure 2: Machine learning models created in the Watson Studio project.

Troubleshooting:
a) Although all of the ML models created by notebooks are deployed successfully, an ”Invalid Input Data” error occurs when the input data is directly typed in to test the model. I posted the issue on the IBM community platform and stack overflow to ask for a solution. This might be a service issue that a user should not be responsible for. Luckily, there is a second way of providing the input data — the JSON format (Figure 3). Thus a prediction value can be obtained.



```
{"fields" :
["f0","f1","f2","f3","f4","f5","f6","f7","f8","f9","f10","f11"
,"f12","f13","f14","f15","f16","f17","f18","f19","f20","f2
1","f22","f23","f24","f25","f26","f27","f28","f29"],
 "values" : [
[ 1.79900000e+01,  1.03800000e+01,
1.22800000e+02,
    1.00100000e+03,  1.18400000e-01,
```

```
{
  "fields": [
    "prediction",
    "probability"
  ],
  "values": [
    [
      0,
      [
        1,
        2.2695816628605858e-36
      ]
    ]
  ]
}
```

Figure 3: Use JSON input data to get a prediction value.

b) With a free IBM account, the number of ML model deployments is limited to 5. In the dynamic algorithm I propose in Section 2, multiple (probably greater than 5) models will be created. However, as we know that sk-learn pipeline models are well compatible with IBM cloud, we can confine our current research within python notebook without worrying about feasibility as long as pipeline models are employed.

2 Proposing a Dynamic Learning Algorithm

In this section, I propose a simple and straightforward algorithm of dynamic learning. Assuming that the number of input data samples is continuously increasing as more data is collected, the machine learning model should be updated periodically to hopefully achieve higher accuracy. Here I define two parameters: $test\_size$ and $batch\_size$. The $test\_size$ is the number of samples in the testing dataset. The most recently collected $\#test\_size$ samples are always employed to evaluate the model. Each time more than $\#batch\_size$ data samples are newly collected, a new model is created to replace the old model. The entire algorithm is summarized in Algorithm 1.

The algorithm is implemented using python notebook and the breast cancer wisconsin dataset. Scikit-learn pipeline models are created to ensure the future IBM cloud implementation. For simulation, the $test\_size$ is set to 100 and the $batch\_size$ is set to 50. The while loop is replaced by a for loop to initialize random number of samples. The accuracy of training and testing datasets along with the number of samples is plotted in Figure 4.

Disadvantage: Multiple machine learning platforms for dealing with dynamic data do exist, such as spark for scoring new data and IBM cloud continuous learning for updating models. Therefore, similar functions can be easily realized using these platforms.

Advantage: This method is quite simple and straightforward. Parameters are flexible to change. Implementation is easy by only running a notebook on IBM cloud.

Algorithm 1 A dynamic learning method of continuously updating the predictive model.

```
Require: Input data including features and labels. The number of samples is continuously updated
    as more data is collected.
Specify the values of $test\_size$ and $batch\_size$.
Create a pipeline model using the desired machine learning estimators (e.g. logistic regression).

Initially train the model using the first $\#batch\_size$ data samples.
Evaluate the initial model using the next $\#test\_size$ data samples. (Assume that initially more
    than ($\#batch\_size + \#test\_size$) samples are available.)
while True do
    Add newly collected data to update the entire dataset.
    if #new samples $\geq \#batch\_size$ then
        Save the most recently collected $\#test\_size$ data samples for testing the model and retrain
        the model using the rest of the data.
    else
        The old model is maintained.
    end if
    Re-evaluate the model using the most recently collected $\#test\_size$ data samples.
end while
```

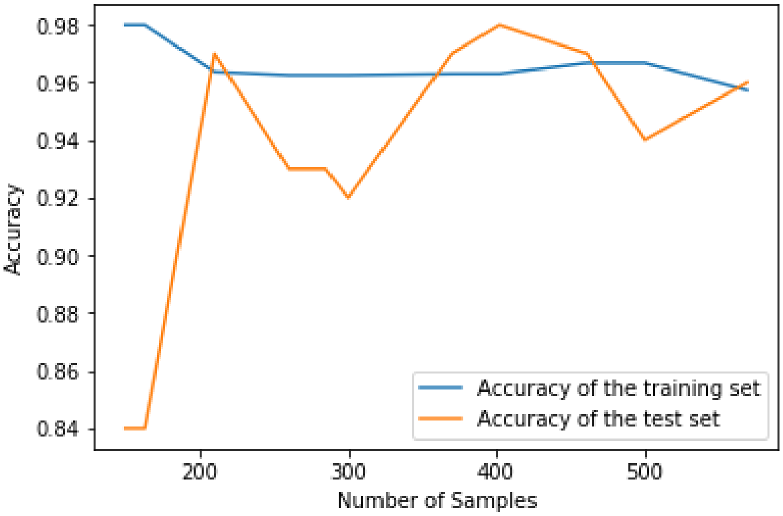


Figure 4: Simulation result on the accuracy of training and testing datasets.

3 Future Work

3.1 Feedback Loop

By far, the proposed model is endowed with the dynamic property for analyzing new data. However, the feedback loop hasn't been added. Feature selection in scikit-learn can be explored for evaluating which features can be dropped each time a model is updated. The motivation of proposing feature selection algorithms is to reduce computation complexity. In BioCyBig, such advantage can be transformed into reducing hardware complexity as fewer types of data will be collected.

3.2 Incremental Learning

In the proposed algorithm in Section 2, all data are used to train the model each time the model is updated. As BioCyBig is supposed to deal with big data, the computation complexity will be huge. In incremental learning, only new data is used to upgrade the model and old data can be dropped. Cloud memory and computation complexity can thus be greatly saved. Therefore, incremental learning algorithms should be explored and incorporated into BioCyBig.

3.3 Multi-omic Dataset

To achieve the ultimate goal of BioCyBig, a specific multi-omic dataset should be used to test our algorithms. This is also where we can emphasize our novelty. Dynamic learning and incremental learning are not new ideas. However, according to existing research on multi-omics [3–6], dynamic learning and incremental learning are seldom applied to multi-omic datasets.

References

[1] "Model builder tutorial: Build a binary classifier model automatically," https://dataplatform.cloud.ibm.com/docs/content/analyze-data/ml-model-builder-tutorial-01-binary-classification-auto.html, 2018, [Online; last updated June 27, 2018; accessed Nov 7, 2018].

[2] "Use scikit-learn to predict hand-written digits," https://dataplatform.cloud.ibm.com/exchange/public/entry/view/168e65a9e8d2e6174a4e2e2765aa4df1, 2018, [Online; last updated May 28, 2018; accessed Nov 8, 2018].

[3] I. S. L. Zeng and T. Lumley, "Review of statistical learning methods in integrated omics studies (an integrated information science)," *Bioinformatics and Biology Insights*, vol. 12, p. 1177932218759292, 2018.

[4] D. Fioravanti, Y. Giarratano, V. Maggio, C. Agostinelli, M. Chierici, G. Jurman, and C. Furlanello, "Phylogenetic convolutional neural networks in metagenomics," *BMC bioinformatics*, vol. 19, no. 2, p. 49, 2018.

[5] A. Jalali and N. Pfeifer, "Interpretable per case weighted ensemble method for cancer associations," *BMC genomics*, vol. 17, no. 1, p. 501, 2016.

[6] T. Ching, X. Zhu, and L. X. Garmire, "Cox-nnet: An artificial neural network method for prognosis prediction of high-throughput omics data," *PLoS computational biology*, vol. 14, no. 4, p. e1006076, 2018.