# DFFML : Add data cleanup dataflow operations and sklearn accuracy scorers

## About me:

| | | |
|---|---|---|
| 1.) Personal Information | • Name | Sudhanshu Kumar |
| | • Github username | @sk-ip |
| | • Gitter username | @sk-ip |
| 2.) Academic Details | • University | Mumbai University |
| | • Institute name | K.J. Somaiya Institute of Engineering and IT |
| | • Program | BE. Computer Engineering |
| | • Year | 4th year |
| | • Expected Graduation Date | August 2021 |
| 3.) Time Zone | • Time Zone | India Standard Time |
| | • GMT | GMT+5:30 |

## Code contribution:

1. https://github.com/intel/dffml/pull/1091 (Draft)

   Working on creating a demo on how to preprocess data using dataflows.

2. https://github.com/intel/dffml/pull/993 (Merged)

   Fixed up all the commits which did not follow the commit guidelines

3. https://github.com/intel/dffml/pull/986 (Merged)

   Rebase accuracy staging with master and fixed the conflicts while rebasing

4. https://github.com/intel/dffml/pull/941 (Merged)

   Updated the http service to understand the accuracy scorer

5. https://github.com/intel/dffml/pull/888 (Merged)

   Updated the accuracy scorer score method signature

6. https://github.com/intel/dffml/pull/863 (Merged)

   Updated auto sklearn version to 0.10.0

7. https://github.com/intel/dffml/pull/832 (Merged)

   Updated the tests for the models

8. https://github.com/intel/dffml/pull/829 (Merged)

   Updated the spacy, pytorch and xgboost models.

9. https://github.com/intel/dffml/pull/792 (Merged)

   Added the mse accuracy plugin examples and tests.

10. https://github.com/intel/dffml/pull/779 (Merged)

    Added an accuracy type plugin

11. https://github.com/intel/dffml/pull/760 (Merged)

Refactor models, predict method should take the source context

12. https://github.com/intel/dffml/pull/747 (Merged)

    Added auto regressor model

13. https://github.com/intel/dffml/pull/703 (Merged)

    Added auto classifier model

14. https://github.com/intel/dffml/pull/674 (Merged)

    New operations tutorial (auto op definition creation)

15. https://github.com/intel/dffml/pull/668 (Merged)

    Exposed run method from high level to no_async

16. https://github.com/intel/dffml/pull/663 (Merged)

    Exposed high level load, save, run methods in no_async

17. https://github.com/intel/dffml/pull/642 (Merged)

    Added definition for return types

18. https://github.com/intel/dffml/pull/627 (Merged)

    Added Definition creation for spec

19. https://github.com/intel/dffml/pull/585 (Merged)

    Added new file source tutorial .ini files as an example

20. https://github.com/intel/dffml/pull/566 (Merged)

    Added load method in high level

21. https://github.com/intel/dffml/pull/555 (Merged)

    Created source for parsing .ini files

22. https://github.com/intel/dffml/pull/541 (Merged)

    Added python code example for scratch models (Logistic regression)

23. https://github.com/intel/dffml/pull/529 (Merged)

    Hiding of python prompts for simple copy pasting of code in interactive sessions.

24. https://github.com/intel/dffml/pull/460 (Merged)

    Added python code and tests for DDN Regression model

25. https://github.com/intel/dffml/pull/451 (Merged)

    Added python code sample for tensorflow's DNNClassifier

26. https://github.com/intel/dffml/pull/439 (Merged)

    Add randomly generated data for tests.

# Project Information:

### 1.) Sub-Org Name :

DFFML (Data Flow Facilitator for Machine Learning)

### 2.) Project Abstract :

a.) Adding dataflow operations which users can use out of the box for data clean up operations

b.) Adding Sklearn metrics methods as scorers for getting accuracy.

### 3.) Detailed Description :

Data Scientists / Engineers spend 80% of their time cleaning the data and 20% in analysing the data. Hence I would like to work on creating dataflow operations which users can use out of the box for their data cleanup tasks. All the cleanup operations can be chained together so that users can directly plugin the data clean up operations that they would like to perform in a single dataflow, the input to that dataflow will be a dataset from a source and the output would be cleaned up data source.

In the real world there are many data cleaning operations that are needed to be performed. Some of the most used data cleaning operations are as follows, these are the possible operations which we could create, which I think can be helpful to the users.

1. Missing value treatment
    a. Delete the missing value record
    b. Make use of mean, mode or median
    c. Predict the missing value
    d. If values are categorical classification can be used
    e. Use sklearn imputation methods Reference: 6.4. Imputation of missing values — scikit-learn 0.24.1 documentation
2. Dropping un-important features / column values (Dimensionality reduction)
    a. Drop column values which users think do not contribute to the prediction of the model.

    b. Drop column values using some predefined mathematical methods such as:
        i. Principal Component analysis, Reference: Principal components analysis (PCA) — scikit-learn 0.24.1 documentation
        ii. Singular Value Decomposition, Reference: sklearn.decomposition.TruncatedSVD — scikit-learn 0.24.1 documentation
3. Change the format of the data present in the records
    a. We can change str type data to float values for example, “100” can be changed to 100 as it's a numerical variable.
    b. Change the date to a standard format.
    c. Remove whitespaces present in the data
4. Outlier detection and termination
    a. Univariate, These outliers can be found when we look at distribution of a single variable.
    b. Multivariate, these are outliers in an n-dimensional space. In order to find them, we would have to look at distributions in multi-dimensions.
    c. To remove outliers we can use the following methods
        i. Remove the outlier
        ii. Transform the values, Natural log of a value reduces the variation.
        iii. We can use mean, median, mode imputation methods.
5. Standardize the values of a variable for better understanding.
    a. This transformation is a must if we have data whose scales are very different, this transformation does not change the shape of the variable distribution.
    b. Some of the transformation which we could use are
        i. Log transformation
        ii. Square / cube transformation
        iii. Encoding for categorical variables Reference: 6.3. Preprocessing data — scikit-learn 0.24.1 documentation
        iv. Non linear Transformations such as Gaussian Transformation 6.3. Preprocessing data — scikit-learn 0.24.1 documentation

According to the timeline and commitment of the GSoC period, implementing all the above operations might not be possible, so I would like to implement a specific set of operations, So the users do not feel that they are missing some things.

I would like to implement the following operations from above possible operations

1. Missing value treatment
    a. In this we would give the user an option of what they would like to do with the missing values present in the data. Accordingly we would go over the records and treat the data. The options for missing value treatment are the same as provided above.
2. Dimensionality Reduction:
    a. Dropping record features which the users think are not necessary for the model.
    b. Drop records features by using Singular value decomposition method.
3. Changing the format of the data present in the dataset:
    a. In this I would like to implement the following three operations
        i. Removing the whitespaces from the dataset
        ii. An operation to change the date format to a standard format like dd-mm-yyyy
4. In the Transformations part, I would like to implement
    a. One hot Encoding for categorical variables present in the dataset.
    b. Log transformation for values which needs to be scaled down.

I am willing to implement any other operations which might be useful to the users upon discussion with the mentors.

We would also use some publicly available datasets which need clean up, this dataset would have numerical values and categorical values, and we would create a demo which can be used to demonstrate how to use the clean up operations.

Finally I would compare the accuracy of the models trained on raw data and clean data so as to show that data cleaning operations do work and can be used by the user.

The datasets which we will be using are available in the public domain and can be used by the people without any conditions.

Some of the datasets which can be used for demonstration / tutorials purposes are

1. House Sales in King County, USA
    a. Imputing missing value records
    b. Transformation for Normalising the values.
    c. Dropping column values which do not contribute much information to the model.
    d. Regression problem
2. NYC Property Sales
    a. Imputing missing values
    b. Log transformation for high values
    c. One hot encoding for categorical variables
    d. Dropping column values which do not contribute much information to the model
    e. Regression Problem
3. Mushroom Classification
    a. Imputing missing values
    b. One hot encoding for categorical values.
    c. Dropping column values which do not contribute much information to the model
    d. Classification problem

DFFML is a plugin based architecture. This means that the source code for the main package dffml, is separate from the source code for many of the things you might want to use in conjunction with it.

In the main package we have another type of plugin type called the accuracy scorer, which takes the scorer you would want to use to get the accuracy of the model.

There are many accuracy evaluation methods, I would like to work on integrating the sklearn metrics and spacy scorers which can be used in sklearn models and spacy models respectively.

In the sklearn metrics we have three types of scorers:

1. Classification
2. Clustering
3. Regression

Reference : 3.3. Metrics and scoring: quantifying the quality of predictions — scikit-learn 0.24.1 documentation

I am planning to Implement all the types of scorers similar to how we were doing it for the sklearn models.

We will have to create a Scikit Scorer Config which will take the configurations required by the scorer classes.

```
class ScikitScorerConfig(NamedTuple):
    """
    Config for Sklearn Scorer
    """
    pass
```

We will create the Scikit Scorer context which would have methods like __aenter__, __aexit__ and score method which would take the model context and sources context, It would get the predicted value and the true value and pass to the scorer for getting the accuracy.

This context will be created for each of the scorers which we will wrap. Similar to how we are doing it in sklearn models.

```
class ScikitScorerContext(AccuracyContext):
    async def __aenter__(self):
        pass

    async def __aexit__(self):
        pass

    async def score(self, mctx: ModelContext, sctx: SourcesContext):
```

```
        pass
```

Then we would implement the Scikit Scorer class which would inherit from the Accuracy Scorer.

```
class ScikitScorer(AccuracyScorer):
    async def __aenter__(self):
        pass

    async def __aexit__(self):
        pass
```

The sklearn model also comes with a default score method which evaluates the accuracy on the basis of the problem which that model was designed to solve.
For implementing this we would create another scorer called the ScikitModelScorer which would use the model context and use the default score method for getting the accuracy of the model.
The implementation of it would be something like given below, since the model itself has a scorer we need not have to use the config part.

```
class ScikitModelScorerConfig:
    """
    Config for Scikit Model Scorer
    """
    pass

class ScikitModelScorerContext(AccuracyContext):
    async def score(self, mctx: ModelContext, sctx: SourcesContext):
        """
        This method would use the model's score method to get the
accuracy
        """
        pass

class ScikitModelScorer(AccuracyScorer):
```

```
CONFIG = ScikitModelScorerConfig
CONTEXT = ScikitModelScorerContext
```

Here the scorer name is ScikitModelScorer which would take the model context and the source context and use that to get the accuracy.

**Note** :- This scorer will not work for Scikit unsupervised models, as it does not have a default implementation of score method hence the users would have to use the sklearn metrics scores for that.

### 4.) Weekly Timeline ( Tentative ) :

**Pre-GSoC** ( 14 April - 15 May ):

- Continue working on the issues to get to know more about the codebase.
- I would communicate with the mentors about the expectations and plans on moving forward.

**Community Bonding** ( 17 May - 6 June ):

- Attend meetings and further clarify about the design and implementation of the projects.
- Complete the work related to sklearn scorer plugin.
- Complete writing the tests for it and the related documentation.

**Week 1** ( 7 June - 12 June ):

- Start Implementation of operation for treating missing values in the dataset.
- Complete Implementation of operation and write tests for it.

**Week 2** ( 14 June - 19 June ):

- Start working on dimensionality reduction operation that is dropping column values which users think are not important.

**Week 3** ( 21 June - 26 June ):

- Complete implementation of the the above operation

- Start implementing the operation Principal Component Analysis for dimensionality reduction

**Week 4** ( 28 June - 3 July ):

- Write tests for the above operations.
- Write a tutorial / demo using one of the above datasets as an example dataset.

**Week 5** ( 5 July - 10 July ):

- Complete any other previous work which is not completed.

**Week 6** ( 12 July - 17 July ):

- First Evaluation
- Start Implementation of the operation to change the format of the data present in the records

**Week 7** ( 19 July - 24 July):

- Complete the above operations
- Start implementation of Transformation operations ie One hot encoding and log transformations

**Week 8** ( 26 July - 31 July):

- Complete the above operations.
- Complete writing tests for the operations

**Week 9** ( 2 August - 7 August ):

- Start creating tutorials / examples from the dataset present above.
- Start documentation of the operations.

**Week 10** ( 9 August - 14 August ):

- Complete all the examples and documentation for the operations implemented
- Submit final code

**Final Week** ( 16 August - 21 August):

- Final Evaluations

Apart from this, the plan also includes fixing bugs on a regular basis as and when they are posted by users and mentors.

## Stretch Goals:

For the stretch goal I would like to:

1. Work on implementing some other data clean up operations upon discussion with the mentors.
2. Work on wrapping the spacy scorers.
3. Discuss with mentors about the design for implementing the spacy scorers.

Reference: [Scorer · spaCy API Documentation]

## Other Commitments:

1. End semester exams in the month of May or June 2021, the exam schedule is not released yet.
    - I will be able to dedicate 3 - 4 hrs / day.

I have no other commitment and will be able to dedicate 8 - 10 hrs during the summer break.
In case something comes up, I will make sure to inform my mentor a week before and limit my unavailability to not more than two days.

## Are you applying for other projects in GSoC:

No, I am applying to this Sub-Org only.

## Further Contribution:

I have chosen DFFML because of the people in the organization and the amazing work that is being done here, I really like the concept of dataflows and would like to contribute to it.

I will continue contributing to the organisation after the GSoC is complete and would love to stay as a part of the community and become a mentor for the next year GSoC.