Prerequisite Packages

[ ] ↳ *1 cell hidden*

Sign Up for Kaggle Account and Generate API Token

1. Sign Up for Kaggle account at https://www.kaggle.com.
2. In Kaggle 'Profile'->'Account', generate api token by clicking 'Create New API Token'.

Upload Token into Colab

3. Upload json file, with Kaggle api token and username, into Colab by running code below. Select downloaded 'kaggle.json' file when prompted and click 'Upload'.

NB third-party cookies should be enabled for upload to work.

[ ] ↳ *1 cell hidden*

Download Merck Molecular Activity Challenge Dataset

```
from cloudmesh.common.StopWatch import StopWatch

StopWatch.start("data-download")

# download dataset
!kaggle competitions download -c MerckActivity

# make train and test directories for raw data
!mkdir -p /content/raw/train /content/raw/test

# unzip train and test datasets into respective folders; only overwrite exiting files if extr
!unzip /content/TrainingSet.zip -d /content/raw/train

!unzip /content/TestSet.zip -d /content/raw/test

# paths to raw data
path_to_raw_train_data = '/content/raw/train/TrainingSet/'
path_to_raw_test_data = '/content/raw/test/TestSet/'

StopWatch.stop("data-download")
```

```
StopWatch.benchmark()
```

```
  inflating: /content/raw/test/TestSet/ACT3_competition_test.csv
  inflating: /content/raw/test/TestSet/ACT4_competition_test.csv
  inflating: /content/raw/test/TestSet/ACT5_competition_test.csv
  inflating: /content/raw/test/TestSet/ACT6_competition_test.csv
  inflating: /content/raw/test/TestSet/ACT7_competition_test.csv
  inflating: /content/raw/test/TestSet/ACT8_competition_test.csv
  inflating: /content/raw/test/TestSet/ACT9_competition_test.csv
```

Attribute	Value
BUG_REPORT_URL	"https://bugs.launchpad.net/ubuntu/"
DISTRIB_CODENAME	bionic
DISTRIB_DESCRIPTION	"Ubuntu 18.04.5 LTS"
DISTRIB_ID	Ubuntu
DISTRIB_RELEASE	18.04
HOME_URL	"https://www.ubuntu.com/"
ID	ubuntu
ID_LIKE	debian
NAME	"Ubuntu"
PRETTY_NAME	"Ubuntu 18.04.5 LTS"
PRIVACY_POLICY_URL	"https://www.ubuntu.com/legal/terms-and-policies/privacy-poli
SUPPORT_URL	"https://help.ubuntu.com/"
UBUNTU_CODENAME	bionic
VERSION	"18.04.5 LTS (Bionic Beaver)"
VERSION_CODENAME	bionic
VERSION_ID	"18.04"
cpu_count	4
mem.active	981.8 MiB
mem.available	24.5 GiB
mem.free	17.0 GiB
mem.inactive	7.0 GiB
mem.percent	4.0 %
mem.total	25.5 GiB
mem.used	657.6 MiB
platform.version	#1 SMP Thu Jul 23 08:00:38 PDT 2020
python	3.7.10 (default, May 3 2021, 02:48:31) [GCC 7.5.0]
python.pip	19.3.1
python.version	3.7.10
sys.platform	linux
uname.machine	x86_64
uname.node	ba8499c66071
uname.processor	x86_64
uname.release	4.19.112+
uname.system	Linux
uname.version	#1 SMP Thu Jul 23 08:00:38 PDT 2020
user	collab

Name	Status	Time	Sum	Start	tag	Node
data-download	ok	29.961	29.961	2021-05-07 19:22:43		ba8499c6

```
# csv,timer,status,time,sum,start,tag,uname.node,user,uname.system,platform.version
# csv,data-download,ok,29.961,29.961,2021-05-07 19:22:43,,ba8499c66071,collab,Linux,#
```

Preprocess Dataset

```
import pandas as pd
from cloudmesh.common.StopWatch import StopWatch

# make train and test directories for preprocessed data
!mkdir -p /content/preprocessed/train /content/preprocessed/test

# paths to processed data
path_to_preprocessed_train_data = '/content/preprocessed/train/'
path_to_preprocessed_test_data = '/content/preprocessed/test/'

# cycle through 15 data sets preprocessing them for learning
StopWatch.start("preprocessing")
dataset_file_no = 1

while dataset_file_no <= 15:

    dataset_train_file_name = 'ACT' + str(dataset_file_no) + '_competition_training.csv'
    dataset_test_file_name = 'ACT' + str(dataset_file_no) + '_competition_test.csv'

    train_filename = path_to_raw_train_data + dataset_train_file_name
    test_filename =  path_to_raw_test_data + dataset_test_file_name

    train_filename_processed = path_to_preprocessed_train_data + dataset_train_file_name
    test_filename_processed = path_to_preprocessed_test_data + dataset_test_file_name

    print ('Preprocessing dataset ', 'ACT' + str(dataset_file_no))

    train = pd.read_csv(train_filename)
    test = pd.read_csv(test_filename)

    print (len(train.columns.values))
    print (len(test.columns.values))

    train_inx_set = set(train.columns.values)
    test_inx_set = set(test.columns.values)

    # remove molecule label and columns that are not common to both training and test sets
    train_inx = [inx for inx in train.columns.values if inx in set.intersection(train_inx_set
    test_inx = [inx for inx in test.columns.values if inx in set.intersection(train_inx_set,

    train_inx.insert(0,'Act')
    train_inx.remove('MOLECULE')
```

```
        test_inx.remove('MOLECULE')

        #print (train_inx)
        #print (test_inx)

        train = train[train_inx]
        test = test[test_inx]

        #print (train.shape)
        #print (test.shape)

        # save data to csv
        train.to_csv(train_filename_processed, index=False)
        test.to_csv(test_filename_processed, index=False)

        print(train.head(5))
        print ('Preprocessing dataset ', 'ACT' + str(dataset_file_no), ' complete')

        dataset_file_no += 1
    StopWatch.stop("preprocessing")

    StopWatch.benchmark()
```

```
     1  9.8521    0     0     0     0  ...        0        0        0        0        0
     2  8.3264    0     0     0     0  ...        0        0        0        0        0
     3  8.2581    0     0     0     0  ...        0        0        0        0        0
     4  7.3552    0     0     0     0  ...        0        0        0        0        0

     [5 rows x 5553 columns]
     Preprocessing dataset  ACT15  complete

     +---------------------+------------------------------------------------------------------
     | Attribute           | Value
     |---------------------+------------------------------------------------------------------
     | BUG_REPORT_URL      | "https://bugs.launchpad.net/ubuntu/"
     | DISTRIB_CODENAME    | bionic
     | DISTRIB_DESCRIPTION | "Ubuntu 18.04.5 LTS"
     | DISTRIB_ID          | Ubuntu
     | DISTRIB_RELEASE     | 18.04
     | HOME_URL            | "https://www.ubuntu.com/"
     | ID                  | ubuntu
     | ID_LIKE             | debian
     | NAME                | "Ubuntu"
     | PRETTY_NAME         | "Ubuntu 18.04.5 LTS"
     | PRIVACY_POLICY_URL  | "https://www.ubuntu.com/legal/terms-and-policies/privacy-poli
     | SUPPORT_URL         | "https://help.ubuntu.com/"
     | UBUNTU_CODENAME     | bionic
     | VERSION             | "18.04.5 LTS (Bionic Beaver)"
     | VERSION_CODENAME    | bionic
     | VERSION_ID          | "18.04"
     | cpu_count           | 4
     | mem.active          | 1.0 GiB
     | mem.available       | 24.4 GiB

     | mem.free            | 1.8 GiB
     | mem.inactive        | 22.0 GiB
```

```
| mem.percent         | 4.3 %
| mem.total           | 25.5 GiB
| mem.used            | 10.0 GiB
| platform.version    | #1 SMP Thu Jul 23 08:00:38 PDT 2020
| python              | 3.7.10 (default, May  3 2021, 02:48:31)
|                     | [GCC 7.5.0]
| python.pip          | 19.3.1
| python.version      | 3.7.10
| sys.platform        | linux
| uname.machine       | x86_64
| uname.node          | ba8499c66071
| uname.processor     | x86_64
| uname.release       | 4.19.112+
| uname.system        | Linux
| uname.version       | #1 SMP Thu Jul 23 08:00:38 PDT 2020
| user                | collab
+---------------------+-----------------------------------------------------------------

+---------------+----------+---------+---------+---------------------+-------+-------
| Name          | Status   |    Time |     Sum | Start               | tag   | Node
|---------------+----------+---------+---------+---------------------+-------+-------
| data-download | ok       |  29.961 |  29.961 | 2021-05-07 19:22:43 |       | ba8499
| preprocessing | ok       | 585.339 | 585.339 | 2021-05-07 19:24:29 |       | ba8499
+---------------+----------+---------+---------+---------------------+-------+-------

# csv,timer,status,time,sum,start,tag,uname.node,user,uname.system,platform.version
# csv,data-download,ok,29.961,29.961,2021-05-07 19:22:43,,ba8499c66071,collab,Linux,#
```

Predicting Molecular Activity

```
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function
import numpy as np
import pandas as pd
import keras.backend as K
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout, InputLayer, Concatenate
from keras.utils import to_categorical, plot_model
from keras.utils.vis_utils import model_to_dot
from keras.optimizers import Adam
from cloudmesh.common.StopWatch import StopWatch

# make outputs directory
!mkdir -p /content/outputs/

path_to_train_data = '/content/preprocessed/train/'
path_to_test_data = '/content/preprocessed/test/'
path_to_outputs = '/content/outputs/'

# define parameters
```

```
    # define parameters
    hidden_units = 512
    dropout = 0.45
    BATCH_SIZE = 128
    feature_dim = 128
    opti = Adam(lr=0.0001, beta_1=0.5)

    # define fully connected network/MLP
    def fcn_model(input_shape=(feature_dim,)):
        model = Sequential()
        model.add(Activation('relu'))
        model.add(Dropout(dropout))
        model.add(Dense(hidden_units))
        model.add(Activation('relu'))
        model.add(Dropout(dropout))
        model.add(Dense(hidden_units))
        model.add(Activation('relu'))
        model.add(Dropout(dropout))
        model.add(Dense(hidden_units))
        model.add(Activation('relu'))
        model.add(Dropout(0.10))
        model.add(Dense(num_labels))
        model.add(Activation('softmax'))
        model.build(input_shape)
        model.summary()
        model.compile(loss='mean_squared_error', optimizer=opti, metrics=[Rsqured])
        print("\nTraining on dataset number", act_ds," of 15:\n")
        model.fit(x_train, y_train, epochs=15, batch_size=BATCH_SIZE)
        loss, R2 = model.evaluate(x_test,y_test,batch_size=BATCH_SIZE)
        print("\nCorrelation coefficient:", R2)
        return model

    # define correlation coefficient (R^2) formula
    def Rsqured(x,y):

        # approach adopted from RuwanT
        # URL: https://github.com/RuwanT/merck/blob/master/main.py

        x = K.batch_flatten(x)
        y = K.batch_flatten(y)

        avx = K.mean(x)
        avy = K.mean(y)

        num = K.sum((x-avx) * (y-avy))
        num = num * num

        denom = K.sum((x-avx)*(x-avx)) * K.sum((y-avy)*(y-avy))

        return num/denom

    # iterate through 15 distinct datasets of high throughput screening (HTS) assays
```

```
    StopWatch.start("train-evaluate-predict")
    act_ds = 1
    while act_ds <= 15:
        print("\nReading from dataset", act_ds,"of 15:\n")
        data_train_main = pd.read_csv(path_to_train_data + 'ACT' + str(act_ds) + '_competition_tr
        data_ac = pd.read_csv(path_to_test_data + 'ACT' + str(act_ds) + '_competition_test.csv')

        # split each of the datasets into set for training (80%), set for testing/evluation (10%)
        # set for use with validating prediction (10%)
        data_train = data_train_main.sample(frac = 0.8)
        data_test = data_train_main.drop(data_train.index)
        data_prediction = data_test.sample(frac = 0.5)
        data_test = data_test.drop(data_prediction.index)

        activity_inx = data_train.columns.get_loc('Act')
        feature_dim = data_train.shape[1] - (activity_inx+1)
        #print("no. of feature columns:", feature_dim)

        # identify molecular activity labels
        y_train = data_train['Act']
        #print("shape of y_train:", y_train.shape)
        num_labels = len(np.unique(y_train))
        #print("no. of unique labels:", num_labels)
        y_test = data_test['Act']

        # identify and filter for feature-set/molecular-substructure frequencies
        train_set_inx = set(data_train.columns.values)
        test_set_inx = set(data_ac.columns.values)

        train_inx = [inx for inx in data_train.columns.values if inx in set.intersection(train_se
        test_inx = [inx for inx in data_test.columns.values if inx in set.intersection(train_set_
        predict_inx = [inx for inx in data_prediction.columns.values if inx in set.intersection(t

        data_train = data_train[train_inx]
        data_test = data_test[test_inx]
        data_prediction = data_prediction[predict_inx]
        #print(data_train.head(5))

        # format feature-set input data
        x_train = data_train[0:]
        x_train = np.asarray(x_train).astype('float32')
        x_test = data_test[0:]
        x_test = np.asarray(x_test).astype('float32')
        x_predict = data_prediction[0:]
        x_predict = np.asarray(x_predict).astype('float32')

        input_size = x_train.shape[0]


        # call fully connected network for learning and evaluation. Predict biological activity f
        y_predict = fcn_model(input_shape=(input_size,feature_dim)).predict(x_predict,batch_size=
```

```
    # display inputs and outputs for the prediction step and write results to text file

    prediction_file = open(path_to_outputs +"molecular_activity_prediction.txt","a")

    for i in range(len(x_predict)):
      print("Substructure/feature frequencies=%s, Predicted biological activity=%s" % (x_prec
      prediction_file.write(str(x_predict[i])+","+str(np.mean(y_predict[i]))+"\n")
    act_ds +=1
    prediction_file.close()
StopWatch.stop("train-evaluate-predict")

StopWatch.benchmark()
```

```
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi

Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
```

```
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
Substructure/feature frequencies=[0. 0. 0. ... 0. 0. 0.], Predicted biological activi
```

8m 22s completed at 3:57 PM