

In []:

```
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

In []:

```
import os
os.chdir("/content/drive/My Drive/Classroom/projects/Mercari")
ls -l
```

```
total 7772458
-rw----- 1 root root      151 Nov 19 17:35 akarshan.1711@gmail.com_CS1.gdoc
-rw----- 1 root root    192263 Jan  2 21:08 'Copy of HptTfidf2.ipynb'
-rw----- 1 root root      151 Dec 16 13:22 EDA+FE.gdoc
-rw----- 1 root root    2441752 Dec 20 16:29 EDA.ipynb
-rw----- 1 root root     14393 Dec 27 21:06 FE+prep+modelling.ipynb
-rw----- 1 root root     30163 Dec 29 18:34 HptBrnandImpute.v1.0.ipynb
-rw----- 1 root root    249493 Jan  2 20:56 HptTfidf2.ipynb
-rw----- 1 root root    192396 Jan  2 21:54 HptTfidf.ipynb
-rw----- 1 root root 117131678 Jan  1 12:07 lgbt2.csv
-rw----- 1 root root    68399264 Jan  1 02:00 lgbt3.csv
-rw----- 1 root root     927353 Dec 28 15:17 mercari_mainV2.ipynb
-rw----- 1 root root     380928 Jan  2 21:54 Mercari_to3.db
-rw----- 1 root root      77824 Jan  2 14:07 Mercari_to4.db
-rw----- 1 root root     249856 Jan  2 20:56 Mercari_to5.db
-rw----- 1 root root     196608 Jan  2 21:08 Mercari_to6.db
-rw----- 1 root root    11853944 Dec 30 21:08 price_log2.pickle
-rw----- 1 root root    11853944 Dec 31 07:52 price_log.pickle
-rw----- 1 root root      23640 Jan  2 21:57 Stack.ipynb
-rw----- 1 root root    308669128 Dec 10 2019 test_stg2.tsv.zip
-rw----- 1 root root    3474387330 Dec 30 21:08 tfidf2.pickle
-rw----- 1 root root    3623909034 Dec 30 20:50 tfidf.pickle
-rw----- 1 root root    337809843 Nov 11 2017 train.tsv
-rw----- 1 root root       272 Jan  2 19:42 Untitled
```

In []:

```
#importing modules/libraries
import pandas as pd
import numpy as np
import scipy
import seaborn as sns
import matplotlib.pyplot as plt
import gc
import sys
import os
import psutil
# from scipy.stats import randint as sp_randint
# from scipy.stats import uniform as sp_uniform

from tqdm.notebook import tqdm
# from collections import Counter
# from collections import defaultdict
import re
import random
# from random import sample
# from bs4 import BeautifulSoup
import pickle
import inspect
import time

import sklearn
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelBinarizer
```

```

from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error
import lightgbm as lgb
from sklearn.linear_model import Lasso, Ridge

```

```

# import string
# # import emoji
# # from wordcloud import WordCloud
# import nltk
# nltk.download("stopwords")
# # nltk.download("brown")
# # nltk.download("names")
# # nltk.download('punkt')
# nltk.download('wordnet')
# # nltk.download('averaged_perceptron_tagger')
# # nltk.download('universal_tagset')
# # from nltk.tokenize import word_tokenize
# from nltk.corpus import stopwords
# from nltk.stem.wordnet import WordNetLemmatizer
# # from nltk.stem.porter import PorterStemmer

```

```

import warnings
warnings.filterwarnings("ignore")

```

In []:

```

# defining root mean square error over Log transformed y_test data
# (as linear models homoscedasticity can be kept in check for better prediction)
# and hence an effective Root Mean Square Log Error
def error(y_test, predictions):
    return np.sqrt(mean_squared_error( y_test, predictions ))

```

In []:

```

# making a submmissin file compatible with kaggels submission format
def submission(model, file_name):
    test_pr = pd.read_csv('test_stg2.tsv.zip', sep='\t', usecols = ['test_id'] )
    with open('tfidf.pickle', 'rb') as f:
        data = pickle.load(f)
    # loading only the test data that does not have a target values/ test data from kaggle
    data = data[1481661:,:]
    # making prediciton on them and saving them with tets id
    test_pr['price'] = np.expml(model.predict(data))
    test_pr.to_csv(file_name+'.csv')

```

In []:

```

tr_len = 1185329# demarkation of cv data(0.8 percent)
whole_tr = 1481661# whole train data

```

In []:

```

with open('tfidf.pickle', 'rb') as f:
    df=pickle.load(f)
with open('price_log.pickle', 'rb') as f:
    y=pickle.load(f)

```

In []:

```

df = df[:whole_tr] # only taking train and cv data
gc.collect()

```

Out[]:

376

```
In [ ]:
```

```
X_train, X_test, Y_train, Y_test = train_test_split(df, y, train_size = round(0.8*df.shape[0]))
```

```
In [ ]:
```

```
model= Ridge(alpha=4.5, max_iter=10000, tol=0.0005, solver='auto', random_state=34)
model.fit(X_train, Y_train)
op_rdg1 = model.predict(X_test)
```

```
In [ ]:
```

```
error(op_rdg1, Y_test)
```

```
Out[ ]:
```

```
0.4417180108419465
```

```
In [ ]:
```

```
# this submission helped me achieve 0.4488 rmsle on kaggle
submission(modle, 'ridge')
```

```
In [ ]:
```

```
gc.collect()
lgbt = lgb.LGBMRegressor(boosting_type='gbdt', objective='regression',
                           random_state=11, n_jobs=4, subsample_for_bin=81920,
                           learning_rate=0.19989416216581643, num_leaves=137, ma
x_depth=40, n_estimators=2194)
lgbt.fit(X_train, Y_train)

op_lgb1 = lgbt.predict(X_test)
```

```
In [ ]:
```

```
# this submission helped me achieve 0.4209 rmsle on kaggle
error(op_lgb1, Y_test)
```

```
Out[ ]:
```

```
0.4209546895125615
```

```
In [ ]:
```

```
submission(lgbt, 'lgbt2')
```

| Submission and Description | Private Score | Public Score | Use for Final Score |
|--|---------------|--------------|--------------------------|
| mockrey (version 14/14) a day ago by Akarshan kumar From "mockrey" Notebook | 0.43016 | 0.42990 | <input type="checkbox"/> |
| mockrey (version 13/14) 2 days ago by Akarshan kumar From "mockrey" Notebook | 0.44904 | 0.44881 | <input type="checkbox"/> |
| mockrey (version 9/14) 4 days ago by Akarshan kumar From "mockrey" Notebook | 0.44976 | 0.44951 | <input type="checkbox"/> |
| mockrey (version 8/14) | 0.48692 | 0.48613 | <input type="checkbox"/> |

The Leaderboard

Chapter1 x Factorize x projects x projects x modeling x Hyper x Stackip x (73) x Submissi x Mercari x + -

kaggle.com/c/mercari-price-suggestion-challenge/leaderboard

Search

Overview Data Notebooks Discussion Leaderboard Rules Team My Submissions Late Submission

| | | | | | | |
|-----|-------|---------------------|--|---------|----|----|
| 229 | ▲ 566 | hello | | 0.42961 | 2 | 3y |
| 230 | ▲ 503 | ECNU_liu | | 0.42962 | 32 | 3y |
| 231 | ▲ 580 | Yunfeng Zhu | | 0.42969 | 1 | 3y |
| 232 | ▲ 263 | Erik | | 0.42970 | 4 | 3y |
| 233 | ▲ 233 | weiwei | | 0.42984 | 11 | 3y |
| 234 | ▲ 556 | Sterby | | 0.42985 | 23 | 3y |
| 235 | ▲ 578 | Li-Der | | 0.43001 | 10 | 3y |
| 236 | ▲ 284 | HiroyukiSHINODA | | 0.43005 | 43 | 3y |
| 237 | ▲ 488 | Oleg Romanovskyi | | 0.43006 | 3 | 3y |
| 238 | ▲ 297 | Gabriel Preda | | 0.43010 | 18 | 3y |
| 239 | ▼ 125 | [ods.ai] eyeballers | | 0.43059 | 55 | 3y |
| 240 | ▲ 563 | massquantity | | 0.43063 | 98 | 3y |
| 241 | ▲ 568 | yese | | 0.43064 | 13 | 3y |

0.0 kB/s 0.0 kB/s ENG US 04:00 03-01-2021

In []:

```
gc.collect()
lgbt = lgb.LGBMRegressor(boosting_type='gbdt',
                          random_state=11, n_jobs=4, subsample_for_bin=196892,
                          learning_rate=0.17944470650238377, num_leaves=167, ma
x_depth=37, n_estimators=2016)
lgbt.fit(X_train, Y_train)
```

Out []:

```
LGBMRegressor(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
               importance_type='split', learning_rate=0.2, max_depth=40,
               min_child_samples=20, min_child_weight=0.001, min_split_gain=0.0,
               n_estimators=2200, n_jobs=4, num_leaves=140, objective=None,
               random_state=11, reg_alpha=0.0, reg_lambda=0.0, silent=True,
               subsample=1.0, subsample_for_bin=196892, subsample_freq=0)
```

In []:

```
# second best lgbm model
op_lgb1 = lgbt.predict(X_test)
error(op_lgb1, Y_test)
```

Out []:

0.4210177447417679

In []:

```
# this cell is still running
gc.collect()
lgbt = lgb.LGBMRegressor(boosting_type='gbdt',
                          random_state=11, n_jobs=4, subsample_for_bin=99947,
                          learning_rate=0.42742671823862655, num_leaves=151, ma
x_depth=42, n_estimators=2457)
lgbt.fit(X_train, Y_train)
```

In []:

```
op_lgb1 = lgbt.predict(X_test)
error(op_lgb1,Y_test)
```