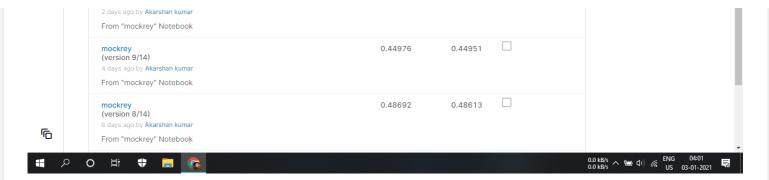
```
In [ ]:
from google.colab import drive
drive.mount('/content/drive')
Mounted at /content/drive
In [ ]:
import os
os.chdir("/content/drive/My Drive/Classroom/projects/Mercari")
!ls -1
total 7705665
-rw----- 1 root root
                            151 Nov 19 17:35 akarshan.1711@gmail.com CS1.gdoc
-rw----- 1 root root 192263 Jan 2 21:08 'Copy of HptTfidf2.ipynb' -rw----- 1 root root 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 22:07 HptTfidf2.ipynb
                        192396 Jan 2 22:08 HptTfidf.ipynb
-rw----- 1 root root
-rw----- 1 root root 117131678 Jan 1 12:07
                                               lgbt2.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
                        196608 Jan 2 21:08 Mercari_to6.db
-rw----- 1 root root
-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
                           28190 Jan 3 07:48 Stack.ipynb
-rw----- 1 root root 308669128 Dec 10 2019 test stq2.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
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
```

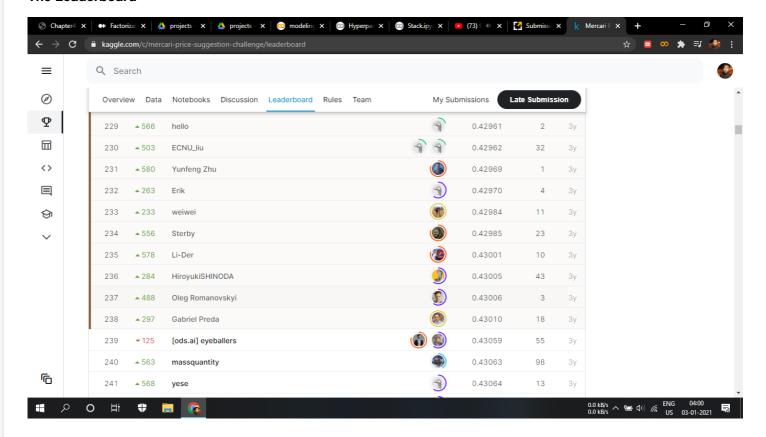
```
import lightgbm as lgb
from sklearn.linear_model import Lasso,Ridge
# import string
# # import emoji
# # from wordcloud import WordCloud
# import nltk
# nltk.download("stopwords")
# # nltk.download("brown")
# # nltk.download("names")
# # nltk.download('punkt')
# nltk.download('wordnet')
# # nltk.download('averaged 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.expm1(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[]:
In [ ]:
```

37 | 1 37 | 1 37 |

```
X train, X test, Y train, Y test = train test split(\alpha r, Y, train size = rouna(0.8 \alpha r.snap
e[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 [ ]:
with open('ridge weights','wb') as f:
  pickle.dump(model,f)
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 0.4299 rmsle on kaggle
error(op lgb1,Y test)
Out[]:
0.4205513051599854
In [ ]:
with open('lbgm coeffs','wb') as f:
  pickle.dump(lgbt,f)
In [ ]:
submission(lgbt, 'lgbt2')
 📀 Chapterd X | 🐽 Factorizal X | 🚵 projects - X | 🚵 projects - X | 🚳 projects - X | 🔞 projects - X | 🔞 projects - X | 🔞 Mercari F X | +
 \leftarrow \rightarrow \mathbf{C} \hat{\mathbf{n}} kaggle.com/c/mercari-price
                                                                                              co
          Q Search
  0
           Overview Data Notebooks Discussion Leaderboard Rules Team
                                                              My Submissions
                                                                          Late Submission
  Φ
           All Successful Selected
  m
                                                     Private Score
                                                                Public Score
  <>
           mockrey
(version 14/14)
                                                      0.43016
                                                                 0.42990
  From "mockrey" Notebook
           mockrey
(version 13/14)
                                                      0.44904
                                                                 0.44881
```



The Leaderboard



In []:

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 []:

```
In [ ]:
In [ ]:
In [ ]:
In [ ]:
stacking/ weighted sum of prices
In [ ]:
lgbm = pd.read csv('lgbt2.csv', usecols=['test id', 'price'])
ridge = pd.read csv('ridge.csv', usecols=['test id', 'price'])
In [ ]:
lg err = 0.4209546895125615
rdg err = 0.4417180108419465
tot = lg_err+rdg_err
In [ ]:
new price = lgbm['price']*rdg err/tot +ridge['price']*lg err/tot
In [ ]:
print(ridge['price'], new price, lgbm['price'])
In [ ]:
ridge['price'] = new price
ridge.to csv('new price.csv')
```

This submission helped me achieve under 6 percentile on in the compition.

