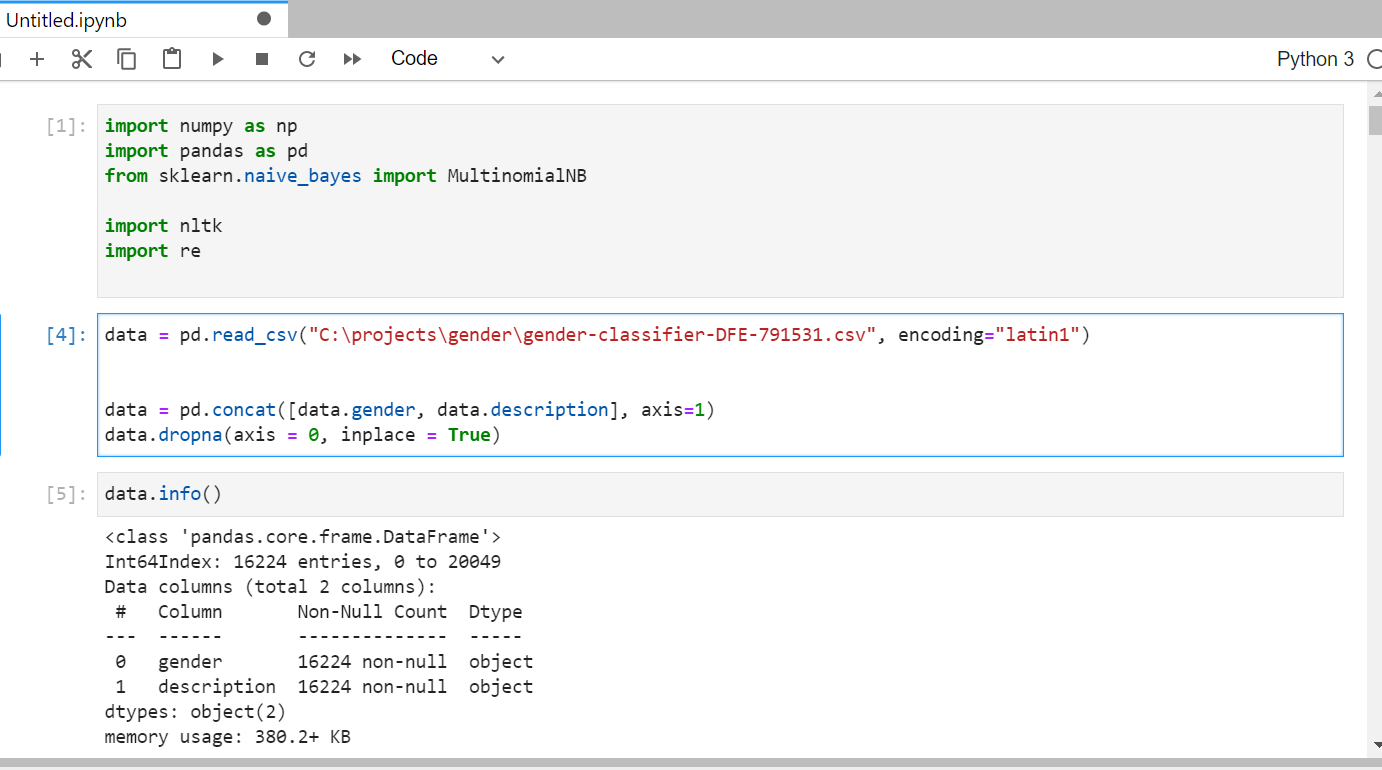
**1.Data Processing Steps:**

Feature selection is the process of reducing the number of input variables when developing a predictive model

* Removing all the null values.i.e;Total-9 columns missing values.so removed all those null values and Dropping constants
* Feature selector removes all low variance features. The variance threshold is a simple baseline approach to feature selection. It removes all features which variance doesn't meet some threshold
* Removing features which are highly correlated using threshold value
* Mutual Information between random variables(k best variables)
* Irrelevant data is aslo removed here

****

**2.Model Building :**

import numpy as np

import pandas as pd

from sklearn.naive\_bayes import MultinomialNB

import nltk

*import re*

data = pd.read\_csv("C:\projects\gender\gender-classifier-DFE-791531.csv", encoding="latin1")

data = pd.concat([data.gender, data.description], axis=1)

data.dropna(axis = 0, inplace = True) *# we dropped the null rows*

data.info()

import nltk

nltk.download("stopwords")

nltk.download('punkt')

nltk.download('wordnet')

data.gender = [1 if gender == "female" else 0 for gender **in** data.gender]

import nltk

description\_list = []

for description **in** data.description:

description = re.sub("[^a-zA-Z]", " ", description)

description = description.lower()

description = nltk.word\_tokenize(description)

lemma = nltk.WordNetLemmatizer()

description = [lemma.lemmatize(word) for word **in** description]

description = " ".join(description)

description\_list.append(description)

from sklearn.feature\_extraction.text import CountVectorizer

max\_features = 5000

cv = CountVectorizer(max\_features=max\_features, stop\_words = "english")

sparce\_matrix = cv.fit\_transform(description\_list).toarray()

print("top used **{}** words: **{}**".format(max\_features, cv.get\_feature\_names()))

y = data.iloc[:, 0].values

x = sparce\_matrix

from sklearn.model\_selection import cross\_val\_score

from sklearn.naive\_bayes import GaussianNB

from sklearn.linear\_model import LogisticRegression

from sklearn import tree

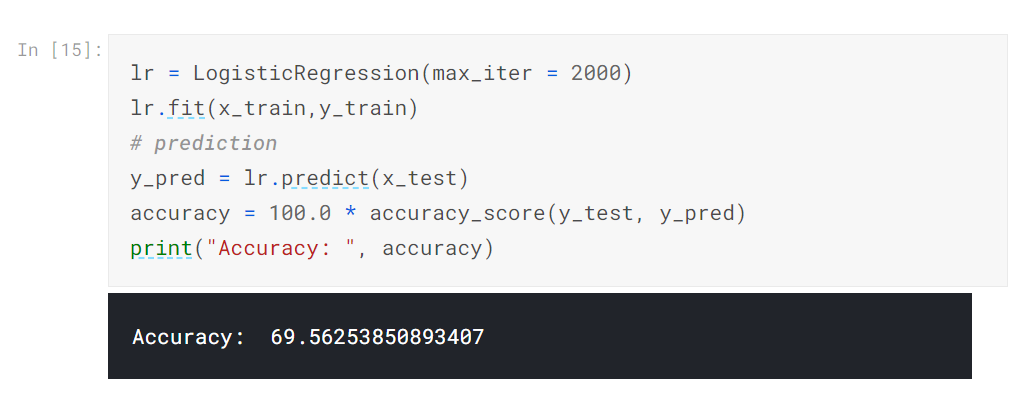
from sklearn.ensemble import RandomForestClassifier, VotingClassifier

from sklearn.neighbors import KNeighborsClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

**Logistic Regression(Top Accuracy):**



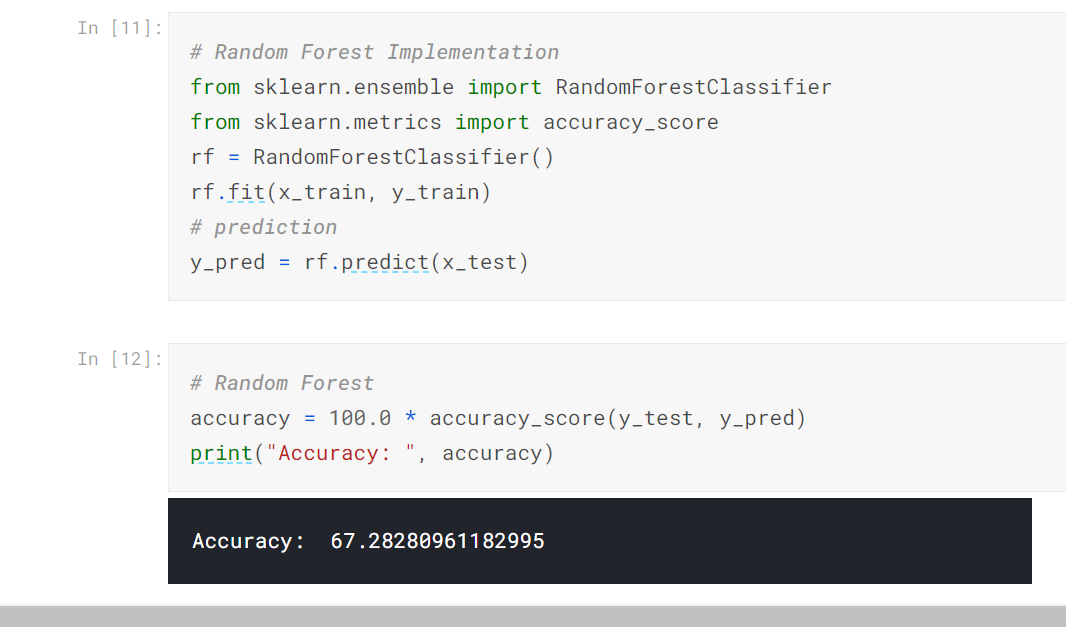
**XGBoost:**

* Language support
* Portable(ML algorithm)
* Integrable with various systems,out of memory computation
* Speed and performance is good,parallelization and cache optimization



**Random Forest:**

Random Forest is suitable for situations when we have a large dataset, and interpretability is not a major concern. Decision trees are much easier to interpret and understand. Since a random forest combines multiple decision trees, it becomes more difficult to interpret.By using row sampling sampling and feauture sampling we can get low bias and low variance



3. **Evaluation Metrics:**

**Accuracy\_score, precision\_score, recall\_score, f1\_score**

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

*# accuracy: (tp + tn) / (p + n)*

accuracy = accuracy\_score(testy, yhat\_classes)

print('Accuracy: **%f**' % accuracy)

*# precision tp / (tp + fp)*

precision = precision\_score(testy, yhat\_classes)

print('Precision: **%f**' % precision)

*# recall: tp / (tp + fn)*

recall = recall\_score(testy, yhat\_classes)

print('Recall: **%f**' % recall)

*# f1: 2 tp / (2 tp + fp + fn)*

f1 = f1\_score(testy, yhat\_classes)

print('F1 score: **%f**' % f1)

**Confusion Matrix**

from sklearn.metrics import confusion\_matrix

matrix = confusion\_matrix(testy, yhat\_classes)

print(matrix)

**Issues on underfitting or overfitting :**

Train Test split is to avoid Overfitting

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.1, random\_state = 42)

Bias-Error of training data

Variance-Error of test data

* Underfitting- Traning data accuracy is low and also Test data accuracy is low

High Bias,High variance

* Overfitting- Traning data accuracy is high and also Test data accuracy is low

Low Bias,High variance

* Aim-Traning data accuracy is high and also Test data accuracy is high

Low Bias,Low variance

