**DS620 Machine Learning and Deep Learning**

**HOS01A Data Preparation**

03/19/2021 Developed by Minh Nguyen

04/03/2021 Reviewed by Shanshan Yu

School of Technology & Computing @City University of Seattle (CityU)

**Learning outcome**

* Overview of a Machine Learning project
* Missing data
* Encoding categorical feature
* Scaling numerical feature
* Column Transformer

**Resources**

* Sklearn documentation: <https://scikit-learn.org/stable/index.html>
* Massaron, L., &amp; Boschetti, A. (2016). Regression Analysis with Python. Packt Publishing.
* Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, Inc.
* Data Scaling: <https://machinelearningmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with-data-scaling/>
* One Hot Encoding: <https://machinelearningmastery.com/one-hot-encoding-for-categorical-data/>

**Introduction**

The most important but perhaps least talked about part of the Data Scientist job. In real world scenarios, data takes a variety of forms which is usually quite messy, unstructured and incompatible for most ML models. As a data practitioner, you will almost never find your data already prepared in the right form to be immediately analyzed for your purposes. As the matter of fact, you will often find yourself spending a majority of your time preparing data in a Data Science project. In order to produce successful machine learning models, the data need to be high quality. An acronym that all Machine learning practitioners need to remember is “GIGO”, short for “garbage in, garbage out.”

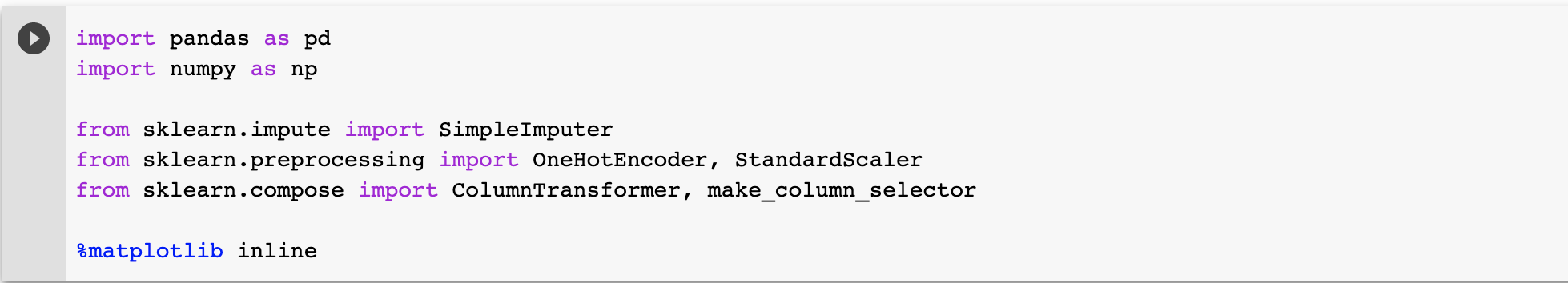
**I. Overview of a Machine Learning project**

1. Get the data.
2. Discover and visualize the data to gain insights.
3. **Prepare the data for Machine Learning algorithms.**
4. Select a model and train it.
5. Fine-tune your model.
6. Present your solution.
7. Launch, monitor, and maintain your system.

These are the main steps a data science team needs to go through in real life when approaching a new data set. In this HOS, we will focus our attention on step 3. We will only briefly go over step 2 as it’s not the main topic of this HOS

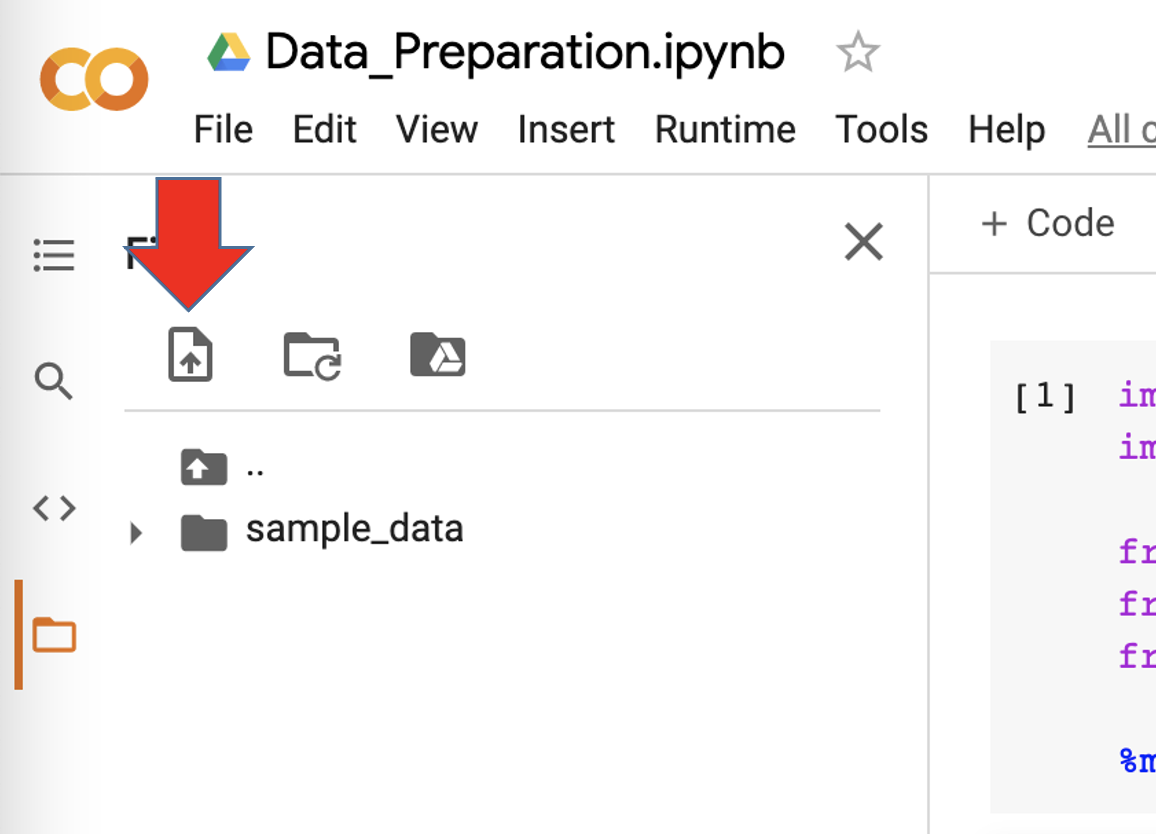
***Preparing development environment***

1. From [Google Colab](https://colab.research.google.com/), create a new notebook, name it “Data\_Preparation.ipynb”
2. Type the following codes to import libraries.

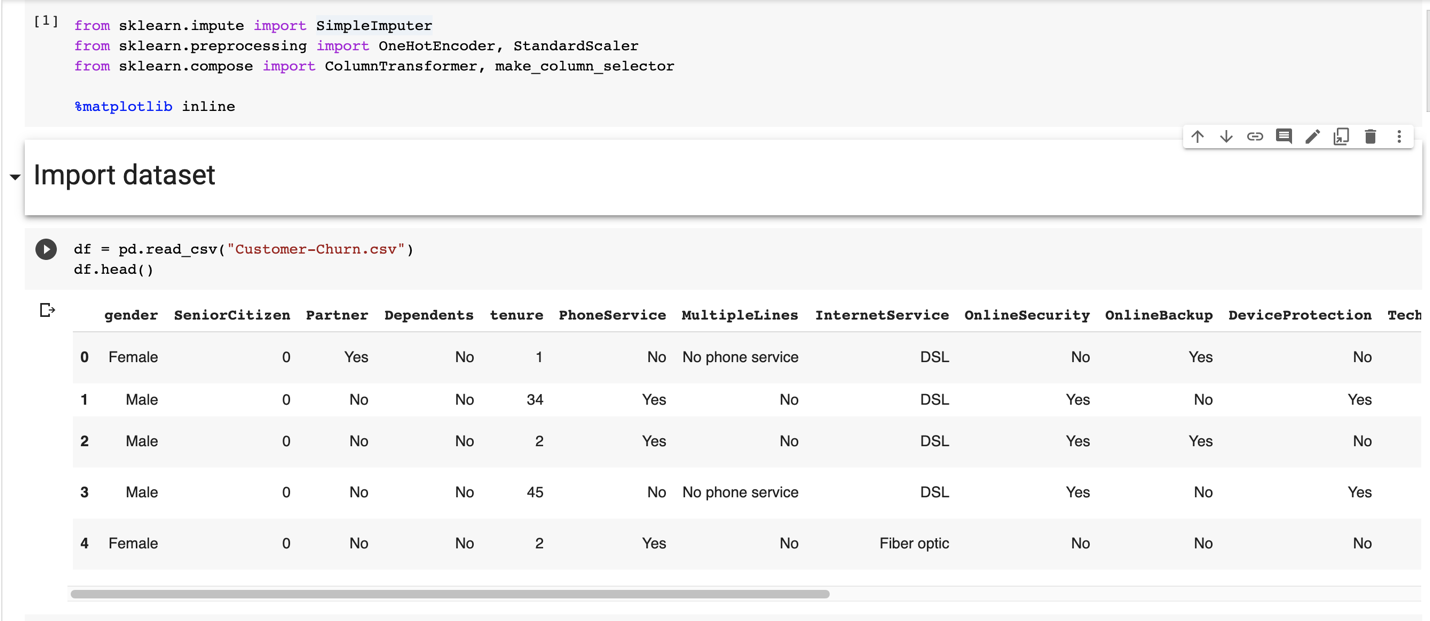


***Get the data***

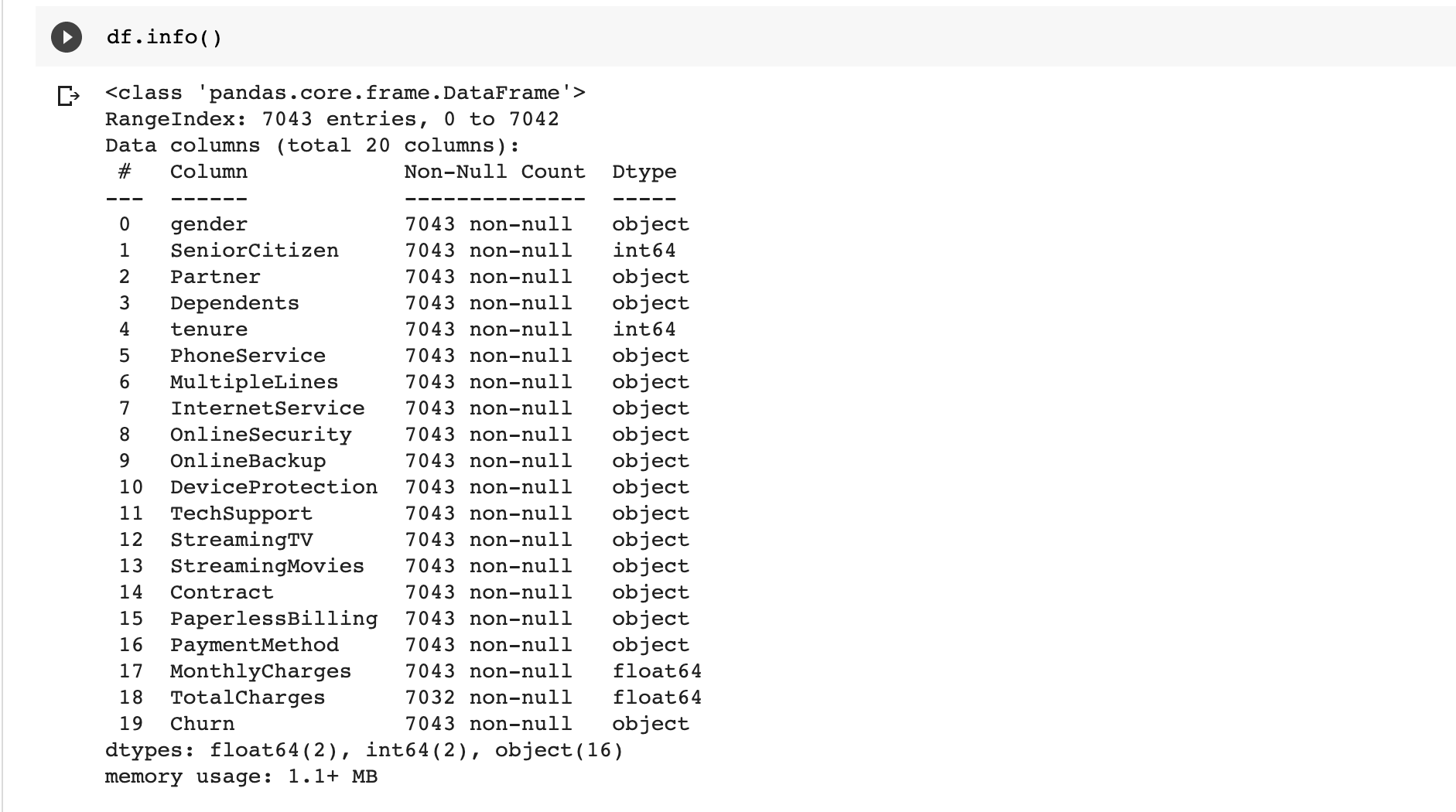
1. Upload the Customer-Churn dataset to Google Colab



1. Run the following code to import the data.



1. Run the following codes to get information about columns of the data as well as its shape.

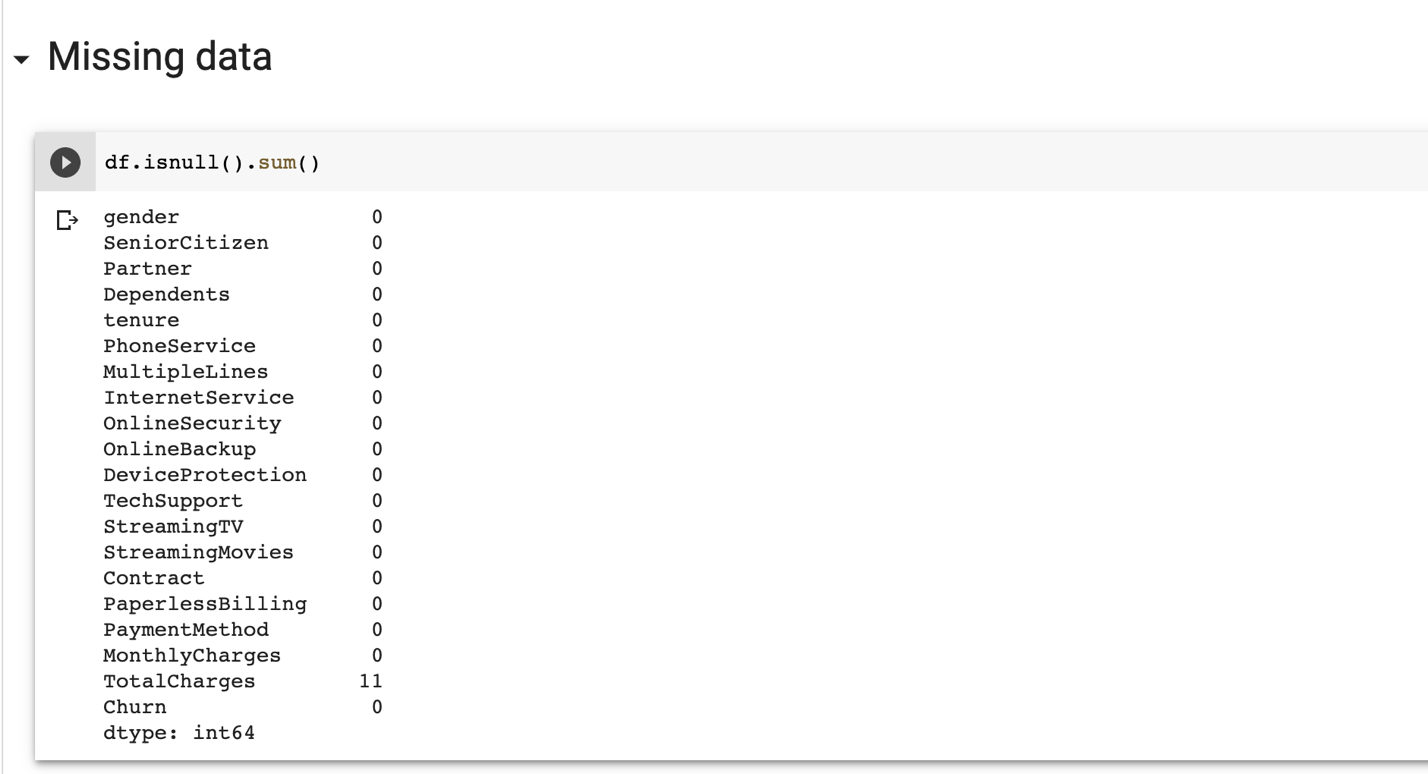


*We can see that most columns in this dataset are categorical.*

**II. Missing data**

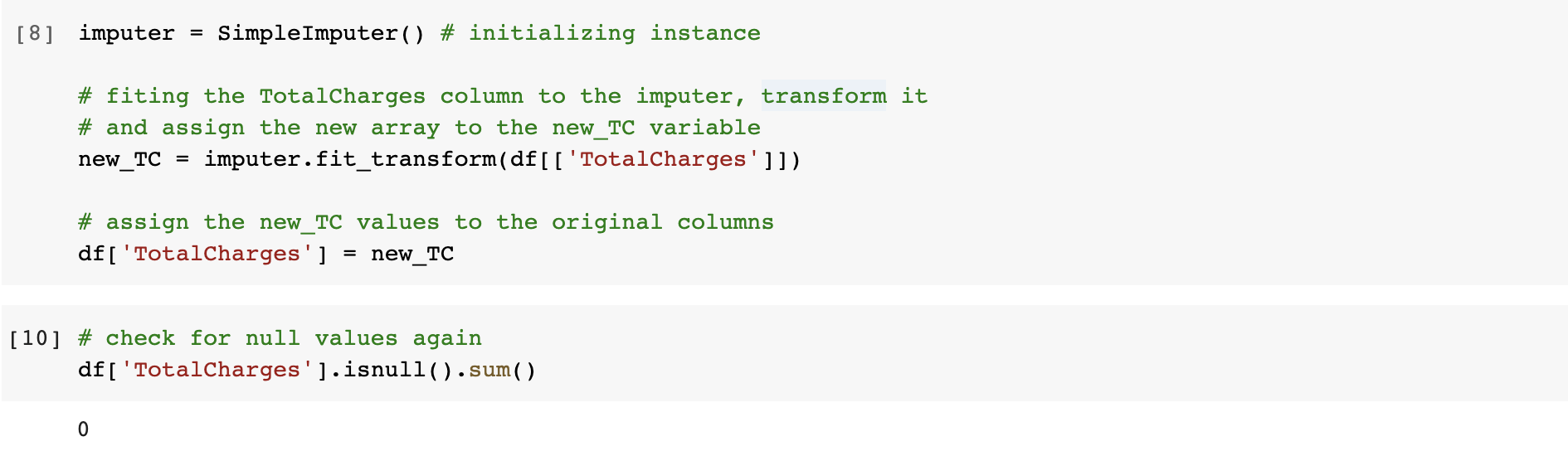
Missing data occurs most of the time in real life data, it usually a result of bias in its recording and treatment. Unfortunately, most machine learning models especially linear models cannot deal with this problem. Therefore, it’s the machine learning practitioner’s job to address this issue either by dropping them or imputing them.

1. Run the following code to check for missing data.



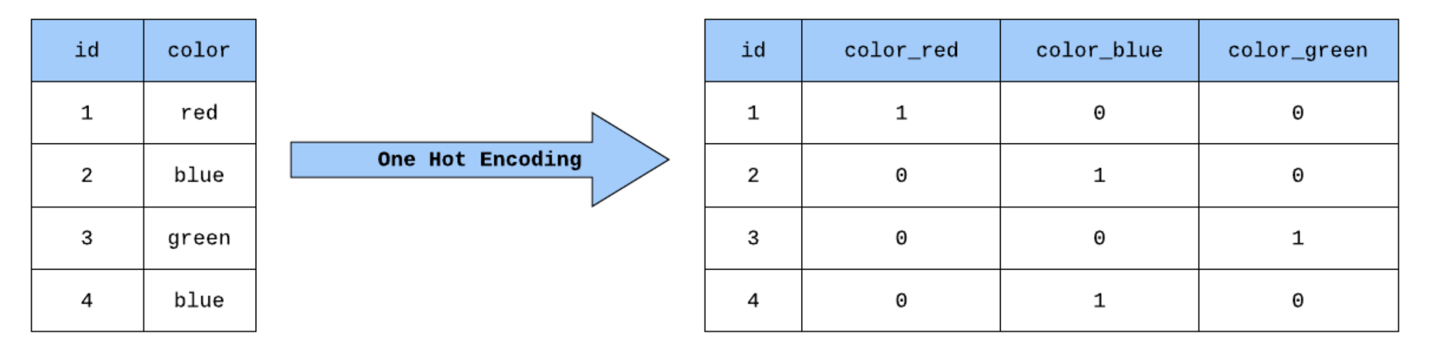
*We can see that there is only 1 column with missing values. Usually, when the number of missing values is not significant, we will drop them as it won’t affect the training of the model. However, when dropping these rows heavily reduce the number of data, a good strategy is to impute them*

1. Run the following code to use SimpleImputer()

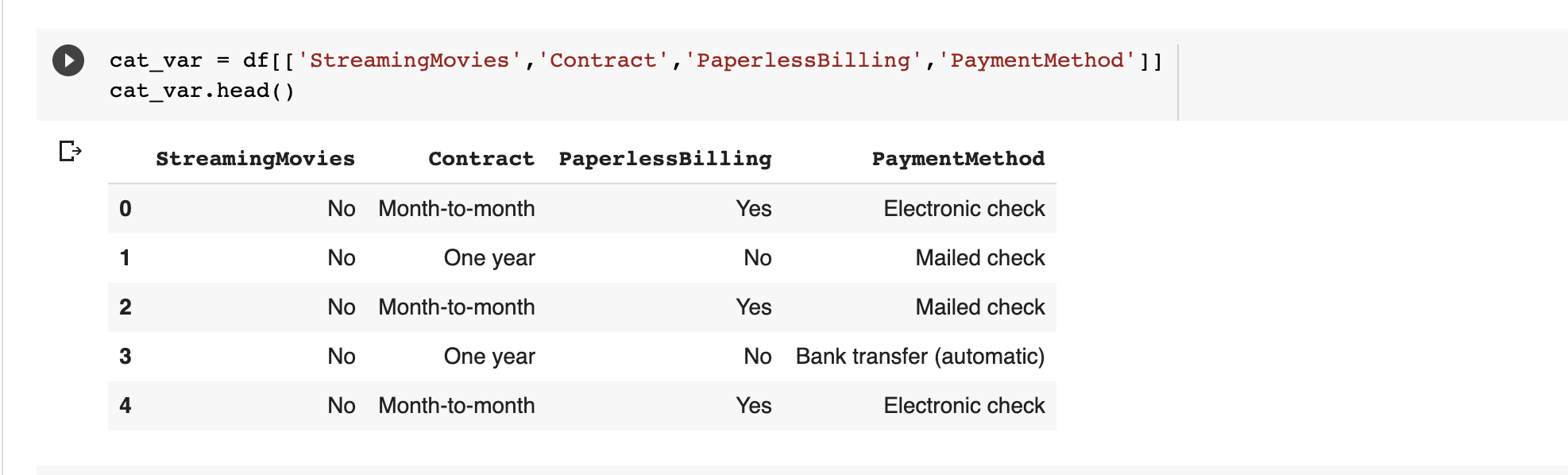


**III. Encoding categorical variables.**

Most Machine Learning algorithms only accept the input data as a numerical matrix format. This means that they won’t accept any columns with text data. Therefore, they need to be converted into numerical through a process called One Hot Encoding. Going into details of this process is beyond the scope of this exercise. At the fundamental level, One Hot encoding turn a column into a metrics of yes (1) - no (0) values.



1. Prepare the categorical data.



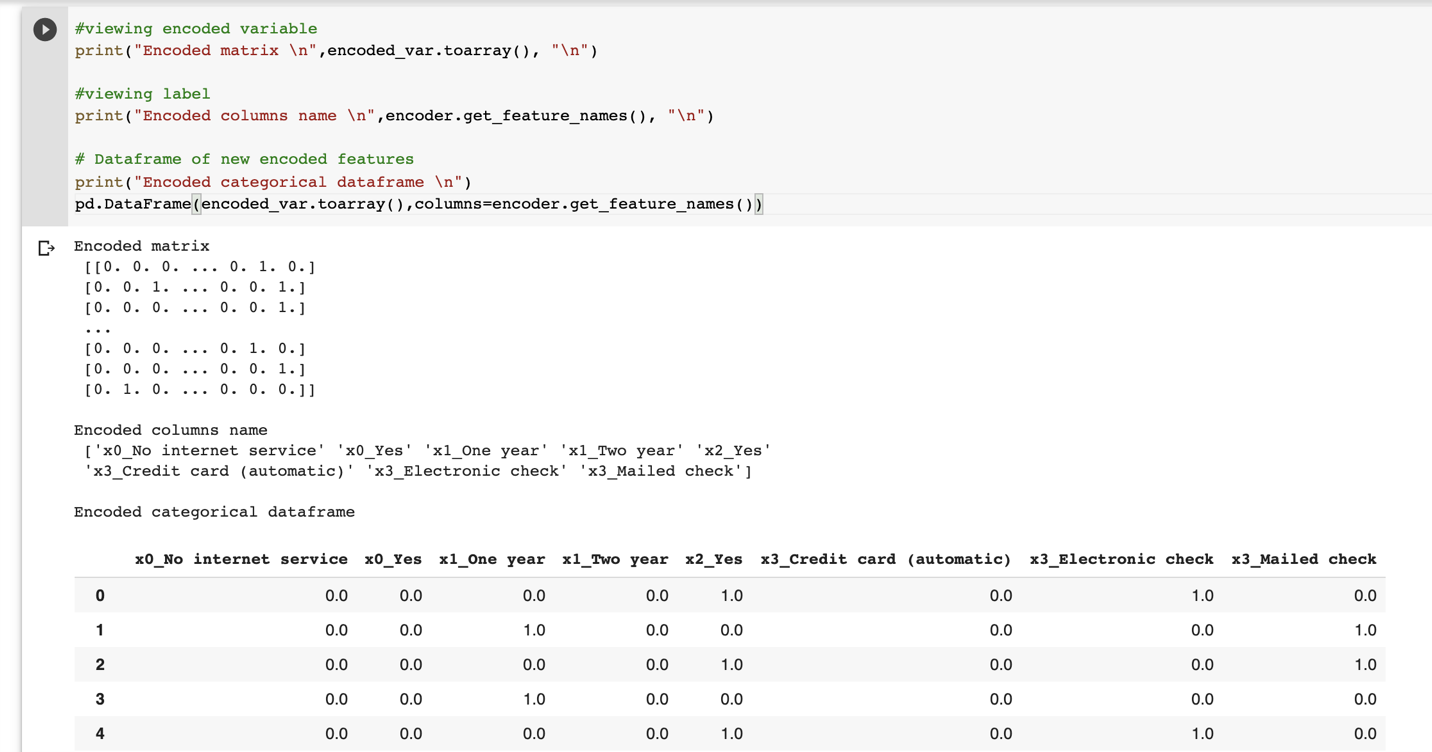
1. Encode the data.



*The specify the parameter drop to “first” will automatically drop the first column of the encoded matrix. This prevents multi collinearity for Linear Model training.*

*You will often see this “fit\_transform” method. It’s a command to tell the instance variable to store information about the data and perform the transformation on the data.*

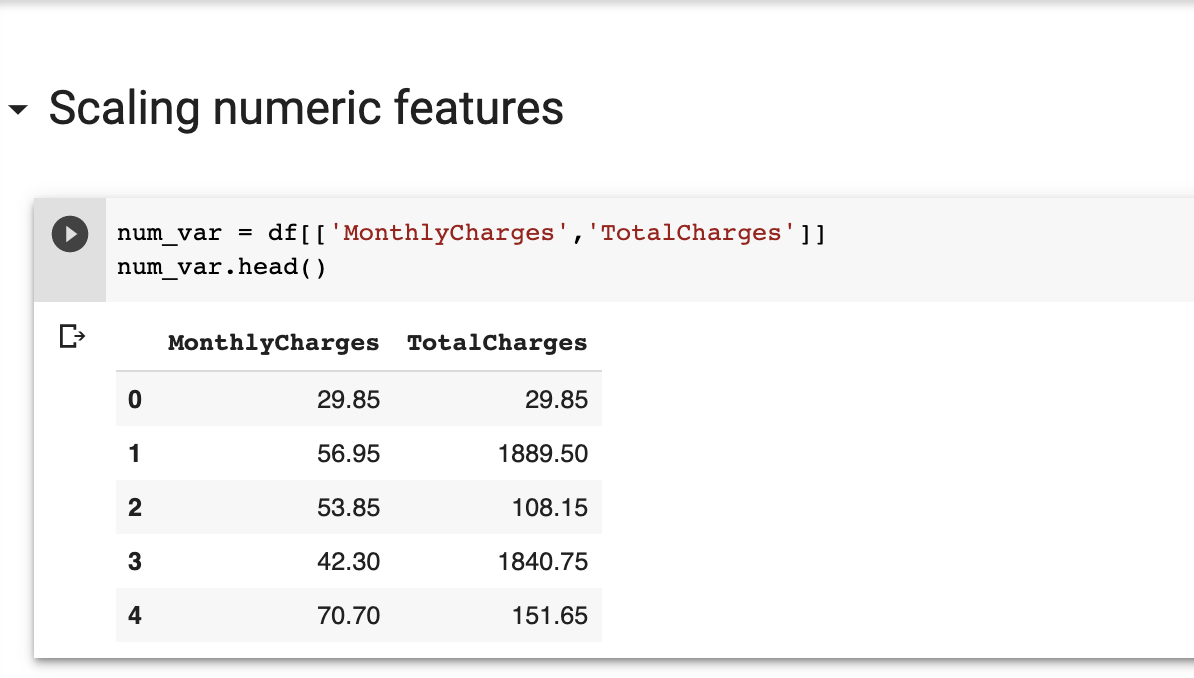
1. Views the encoded data information.



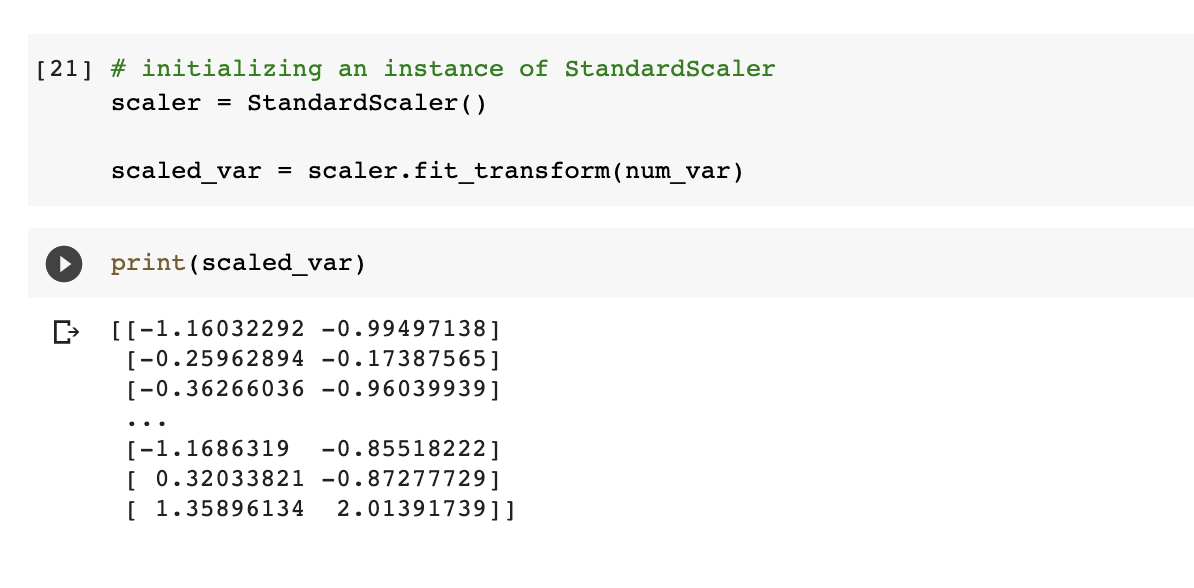
**IV. Scaling numerical features.**

Re-scaling is essential when using gradient descent-based algorithms because it facilitates quicker converging to a solution for finding coefficients. Not only gradient descent rescaling also help other optimization techniques such as regularization, and stochastic learning, and easily detect outlying and anomalous values.

1. Prepare numeric columns.



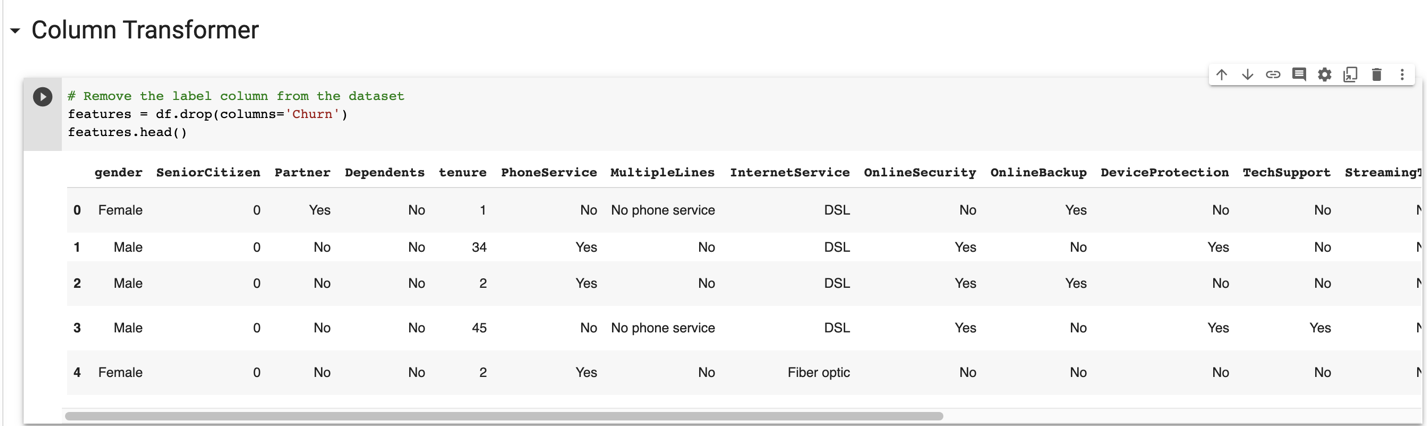
1. Rescale all columns.



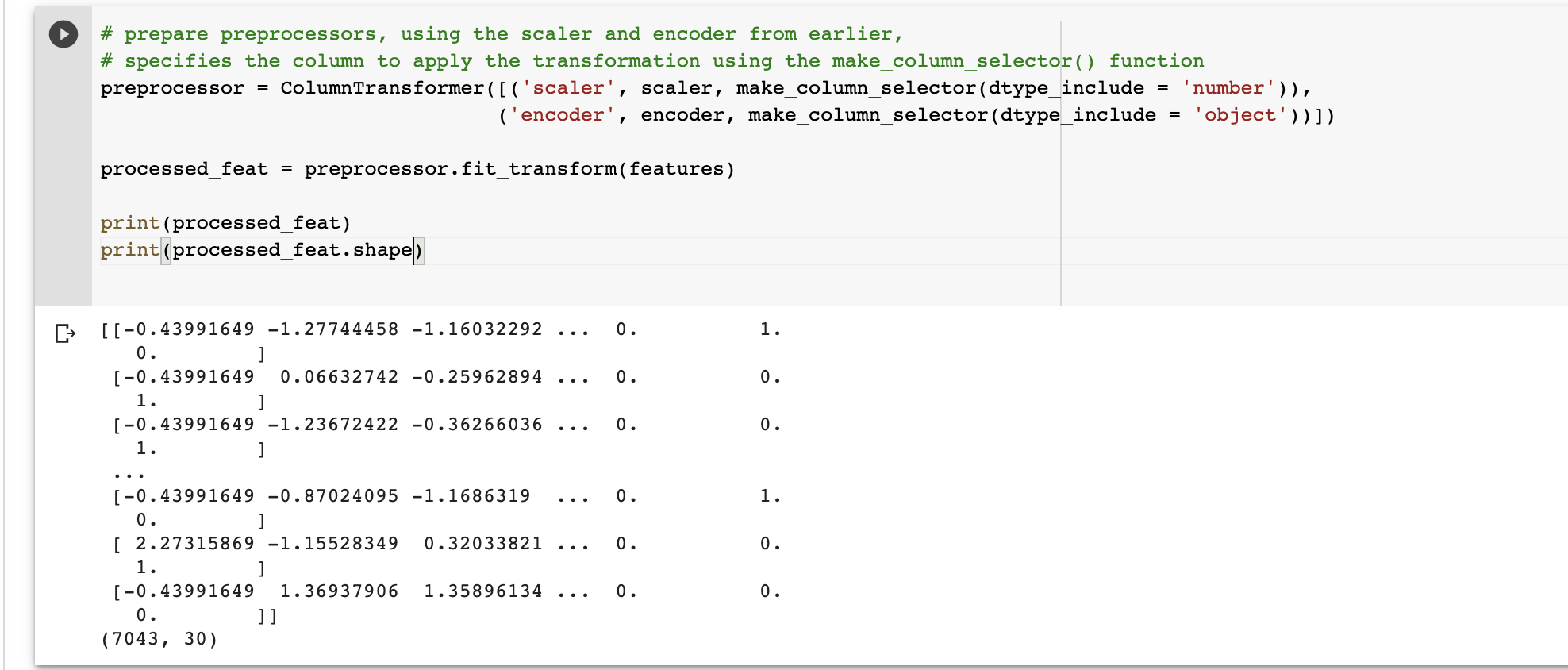
**V. Column Transformer.**

So far, we have been working with categorical data and numerical data separately. What if there is a way to do this in one step? The ColumnTransformer() class is your answer. This class creates a Transformers that apply the specified type of transformation on the specified column. Specifying column can either based on data type or column name.

1. Prepare the input matrix.



1. Prepare the preprocessor object and apply it to the data.



**Push Your Work to Github**

**Download the notebook from Colab**:

File -> Data\_Preparation.ipynb

Move the downloaded file into your **Module1** working folder.

Open terminal and Navigate to the GitHub folder of this week HOS

**Make sure the assignment files on the subfolder Module1 of hos01a\_YouGithubUserName folder, enter the following command to upload your work**:

>>>> git add .

>>>> git commit -m “Submission for HOS01”

>>>> git push origin master