**ADVERTISING SALES CHANNEL PREDICTION**



When a company enters a market, the distribution strategy and channel it uses are keys to it's success in the market, as well as market know-how and customer knowledge and understanding. Because any effective distribution strategy under efficient supply-chain management opens the doors for attaining competitive advantage and strong brand equity in the market. It is a component of the marketing mix that cannot be ignored.

Sales forecasting offers value throughout a business. Finance, for example, relies on projections to set budgets for capacity planning and recruiting. Production employs sales estimates to organise its cycles. Forecasts aid sales ops with territory and quota planning, supply chain with material procurement and manufacturing capacity, and sales strategy with channel and partner plans.

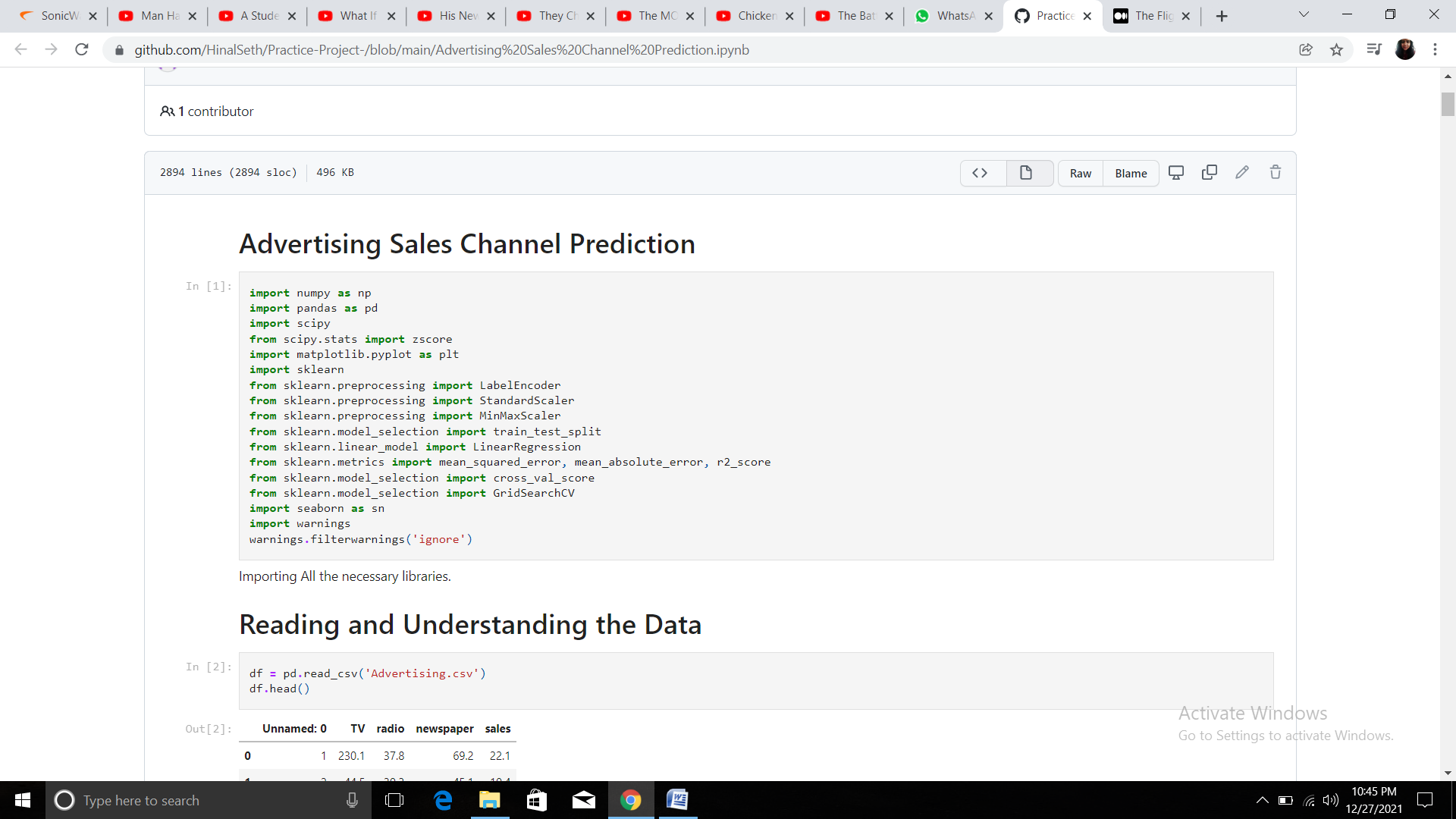
An accurate sales forecast process confers many benefits. These include:

* Improved decision-making about the future
* Reduction of sales pipeline and forecast risks
* Alignment of sales quotas and revenue expectations
* Reduction of time spent planning territory coverage and setting quota assignments
* Benchmarks that can be used to assess trends in the future
* Ability to focus a sales team on high-revenue, high-profit sales pipeline opportunities, resulting in improved win rates

In this blog-post, I will go through the whole process of creating a machine learning model on "Advertising Sales Channel Dataset". As the advertising landscape continues to evolve, advertiser is finding it increasingly challenging to efficiently pinpoint the impact of various revenue-generating marketing activities within their media mix. Here I am building a model which predicts sales based on the money spent on different platforms like TV, Radio and Newspaper for marketing.

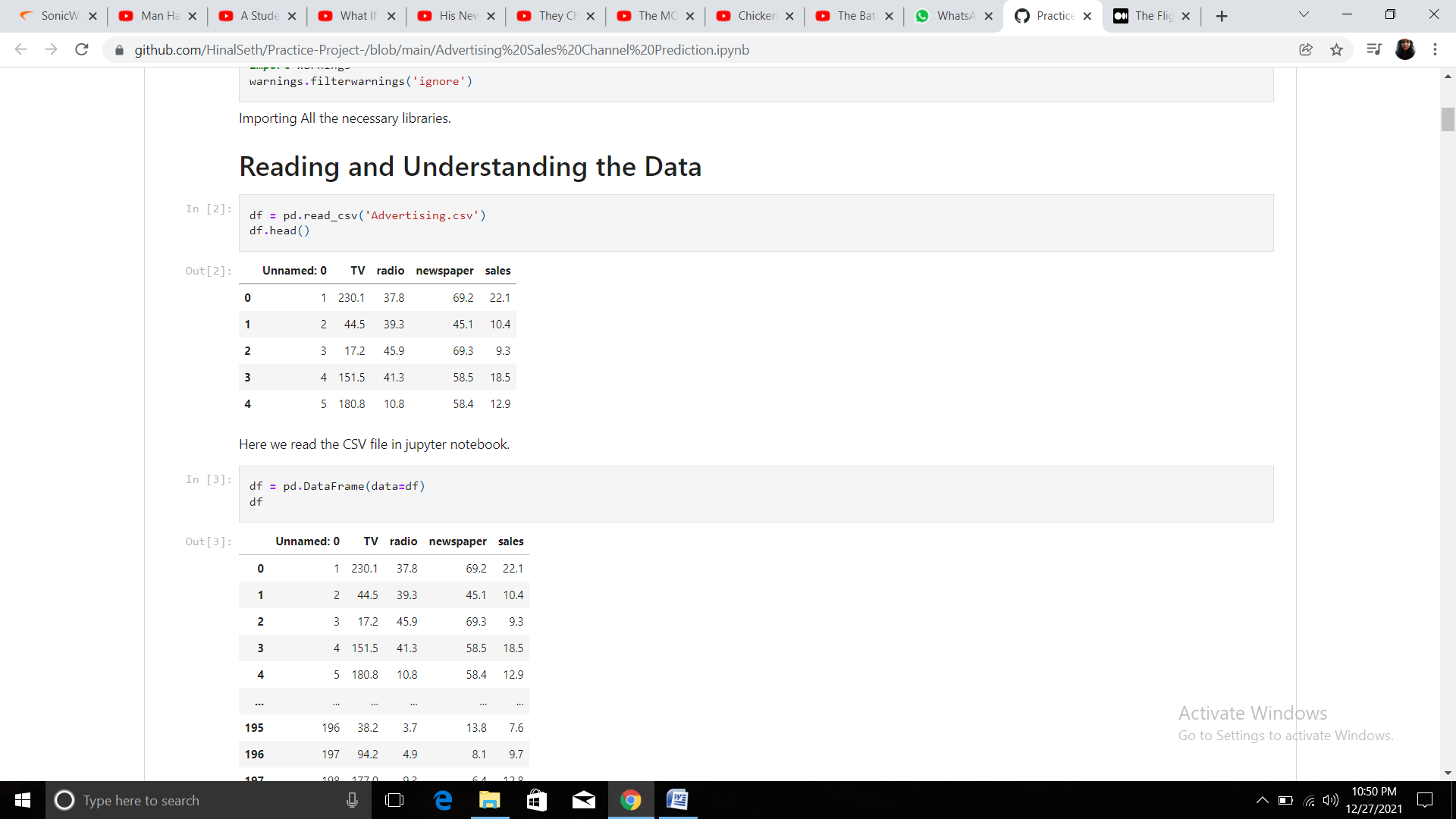
Let's take a dataset that contains the detailed study of TV, radio and newspaper channel. Here I am predicting the total sales generated from the entire sales channel.

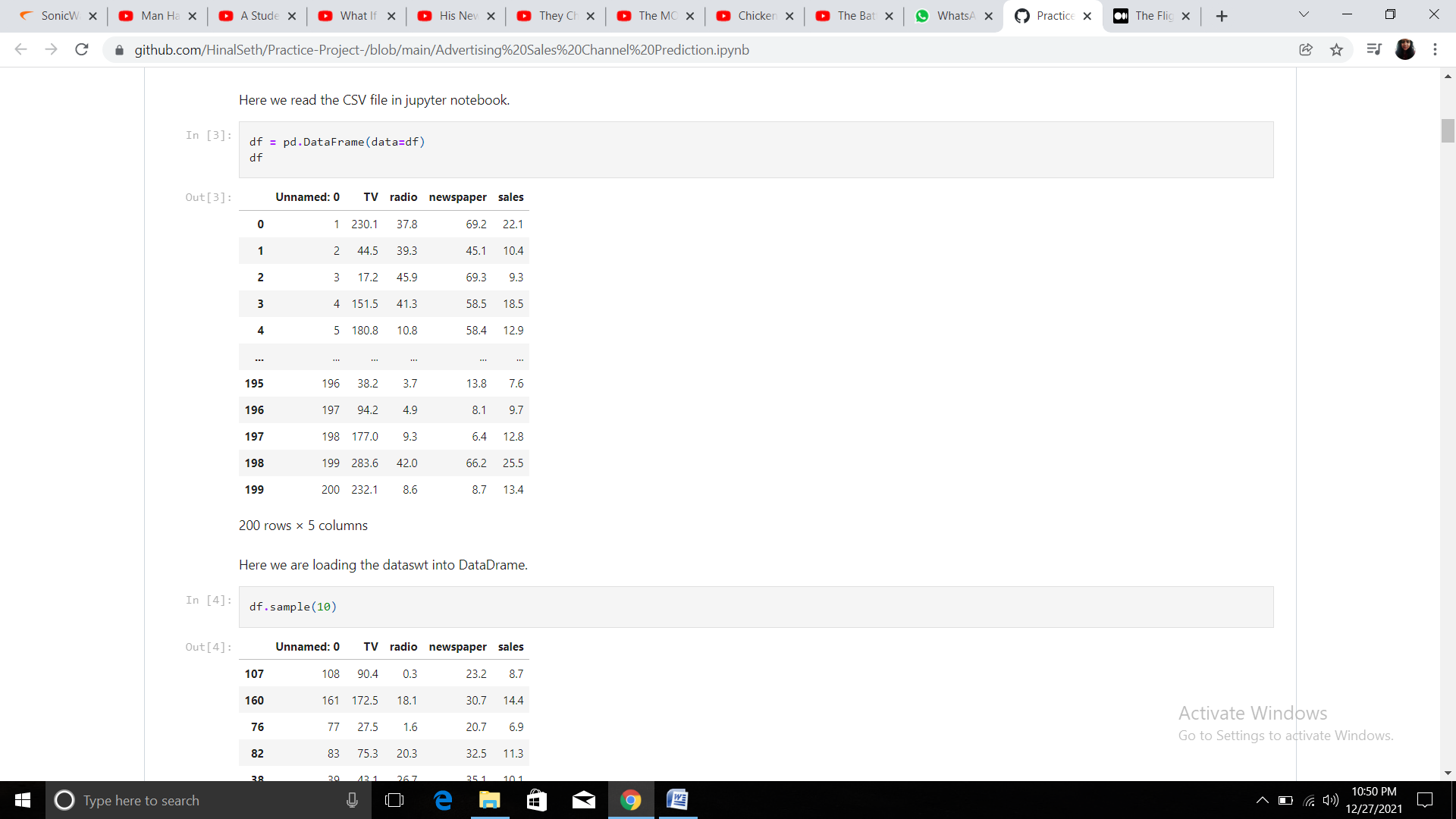
**Importing the Libraries**

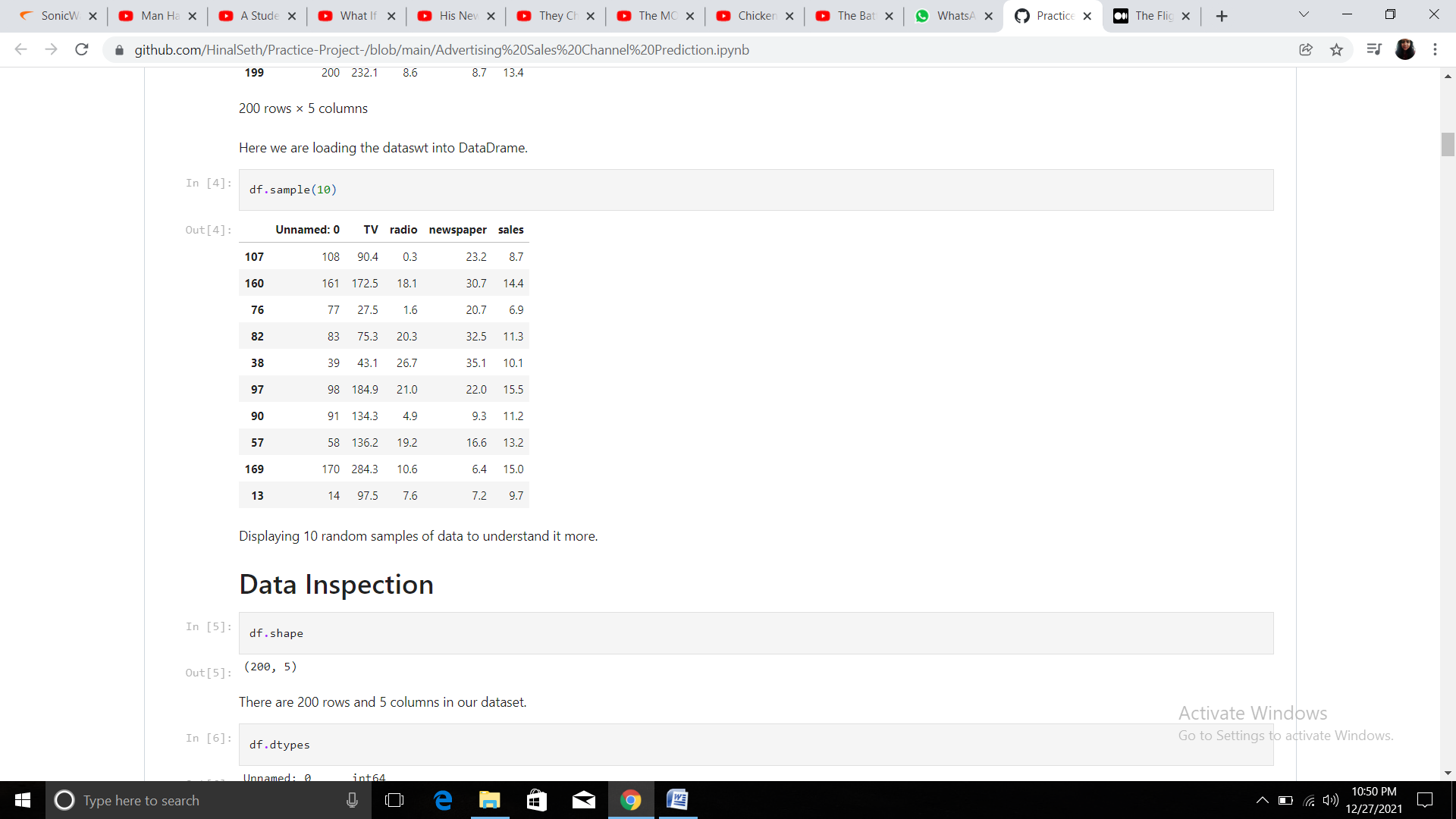
****

**Reading and Understanding the data**

First we read the CSV file into the Jupyter notebook, print its head, then we will load the dataset using the pandas DataFrame and print the DataFrame.

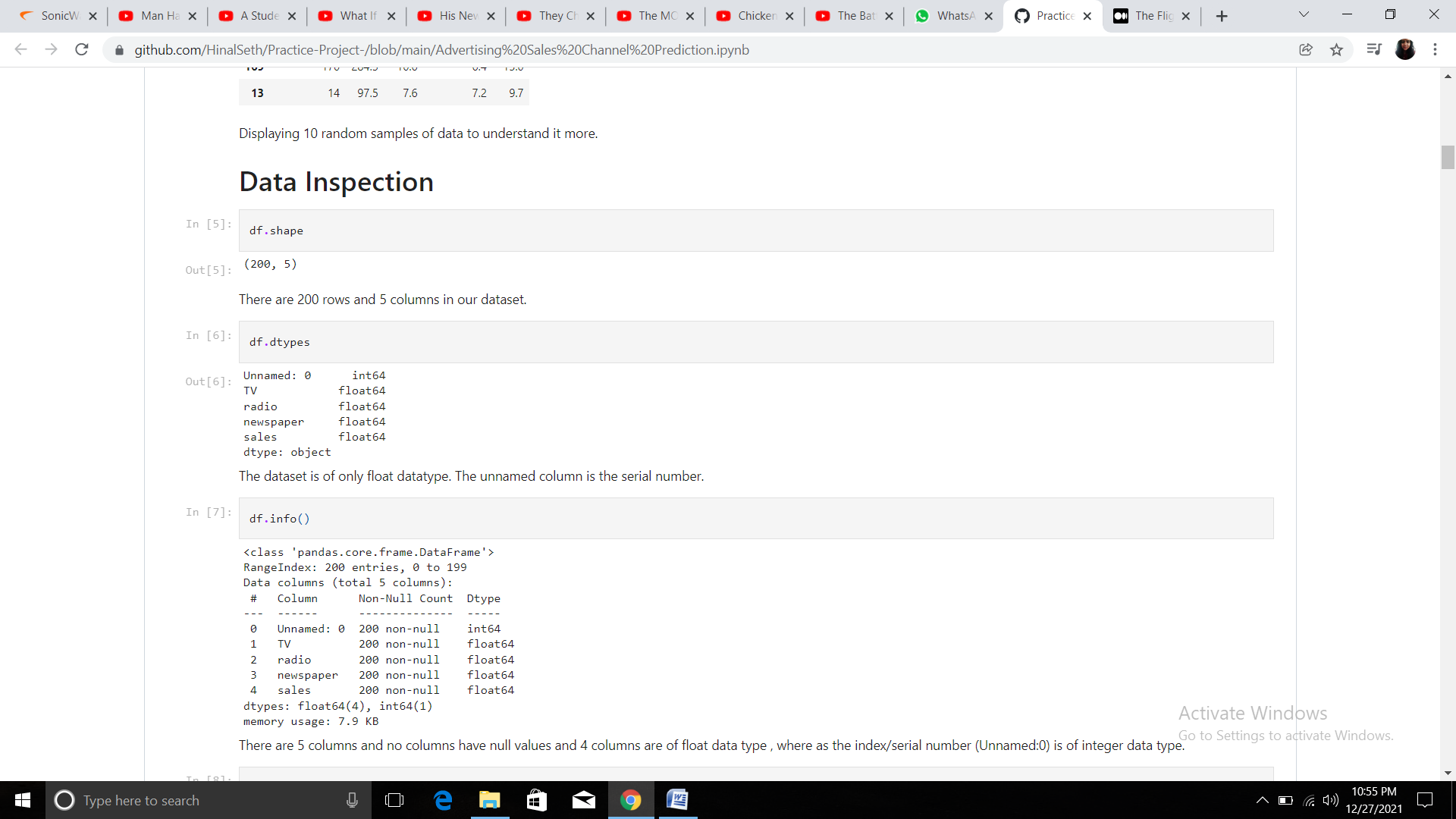


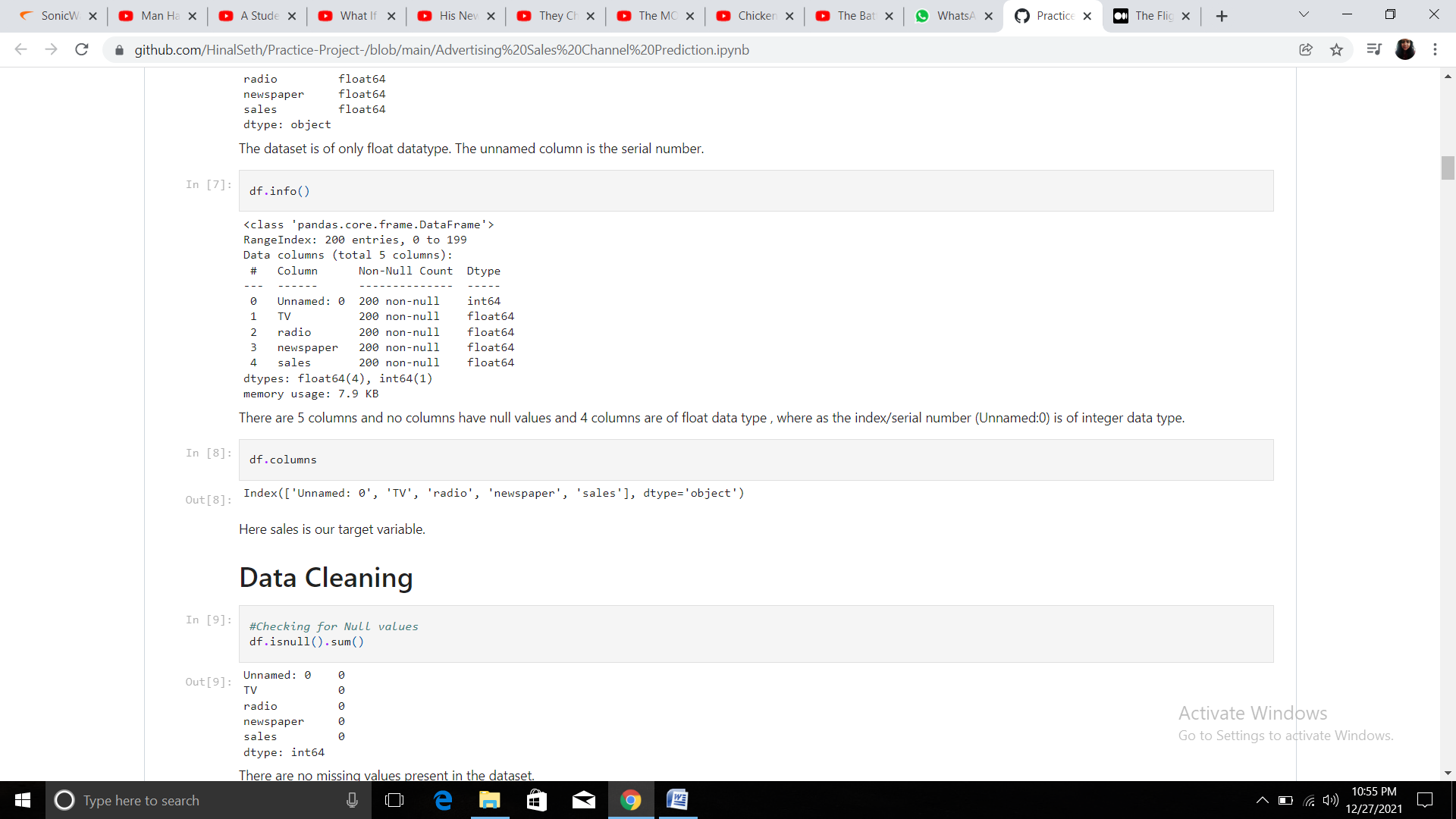




**Data Inspection**

In this project we have a dataset which has the details of money spent on different platforms for advertisement. There are 200 rows and 5 columns in the dataset. The first column, "Unnamed :0", is just a column with serial number and hence we can drop it.





Here 'sales' is our target variable.

The obvious con of this dataset is the small sample size. And there were many advertisement channels like Facebook, Instagram that are missing from the dataset.

**Data Cleaning**

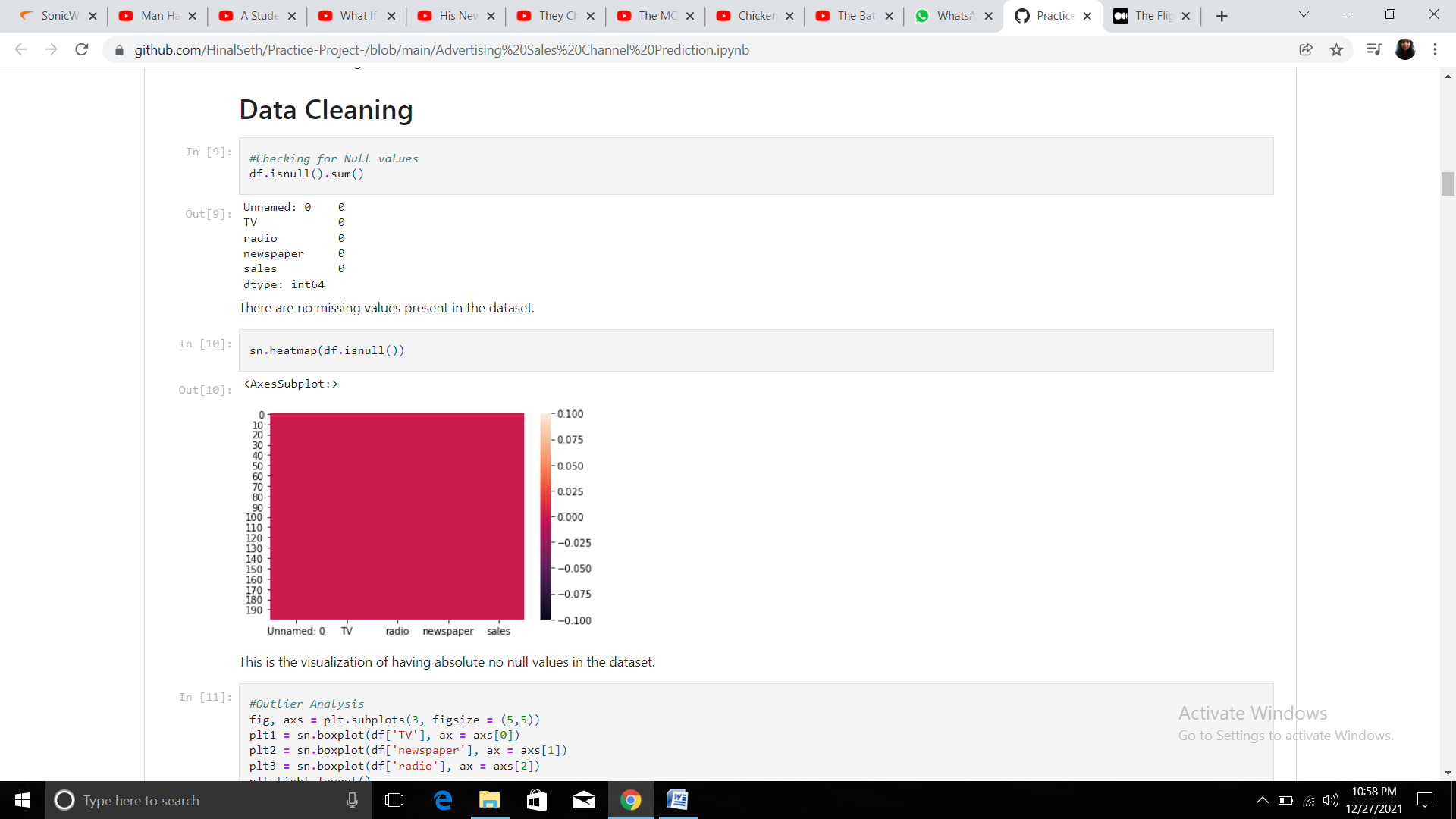
Data Cleaning is very important to get rid of the irregularities and clean the data after sourcing it into our system.

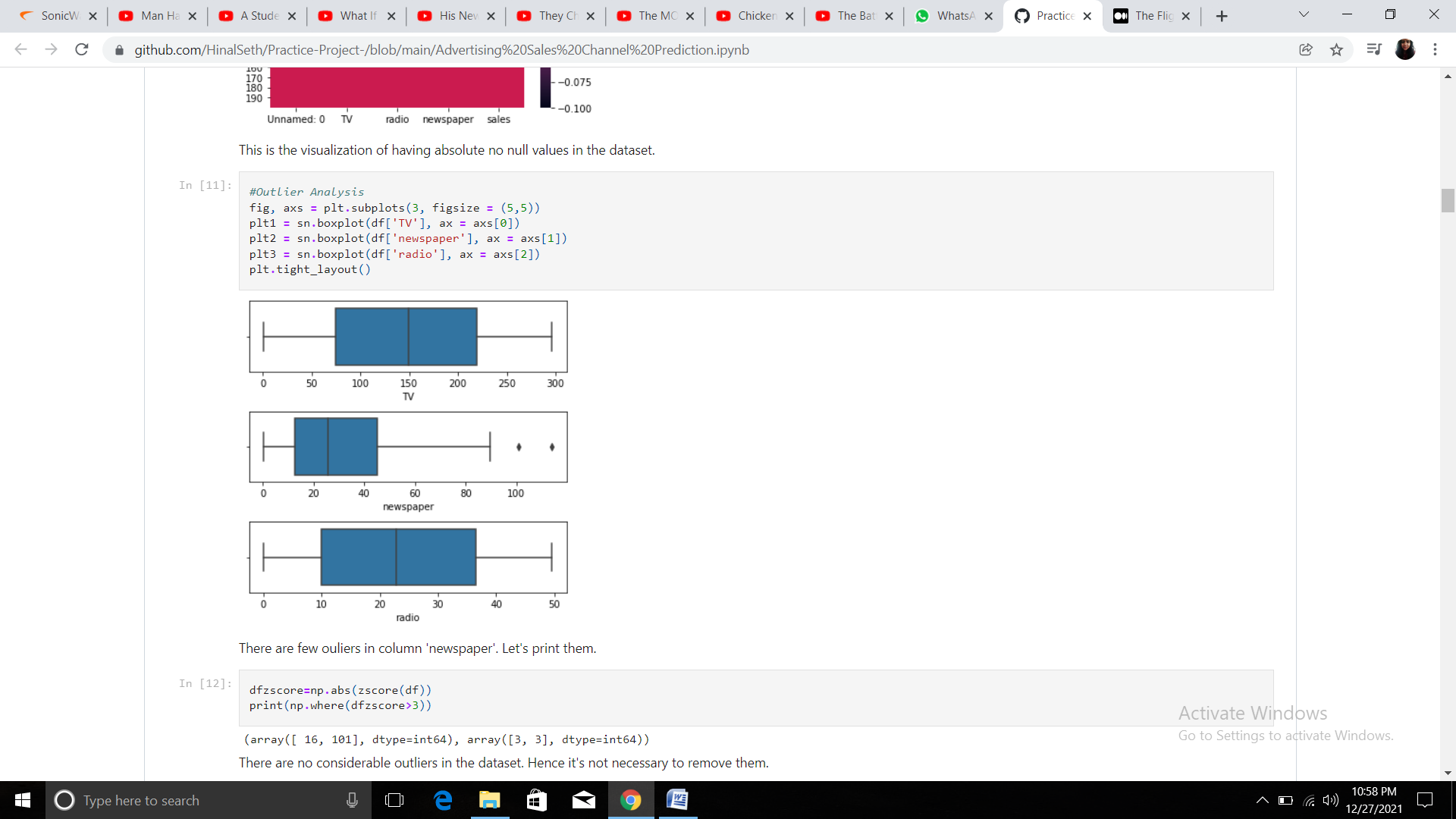
Irregularities are of different types of data.

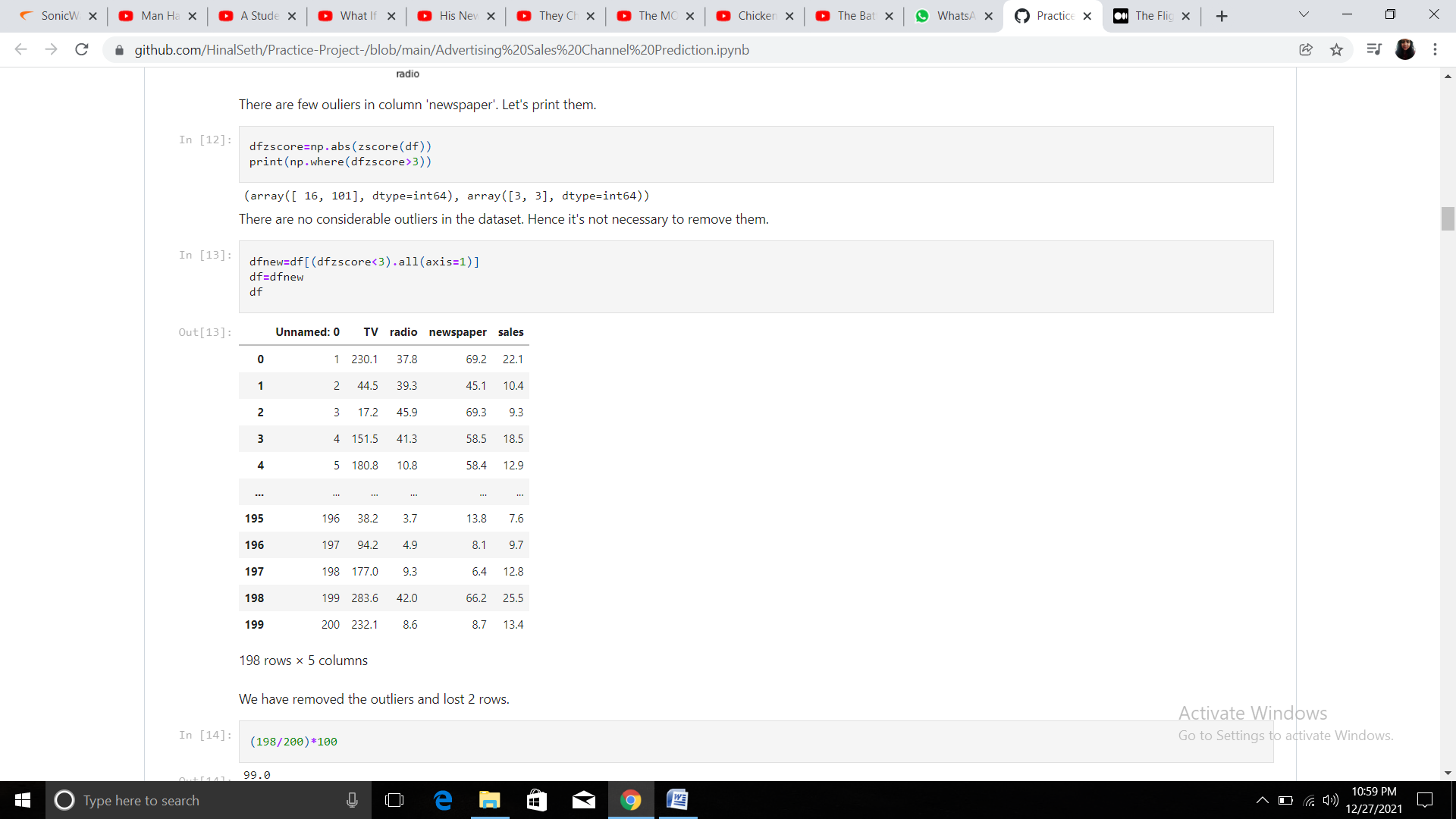
* Missing Values
* Incorrect Format
* Incorrect Headers
* Anomalies/Outliers

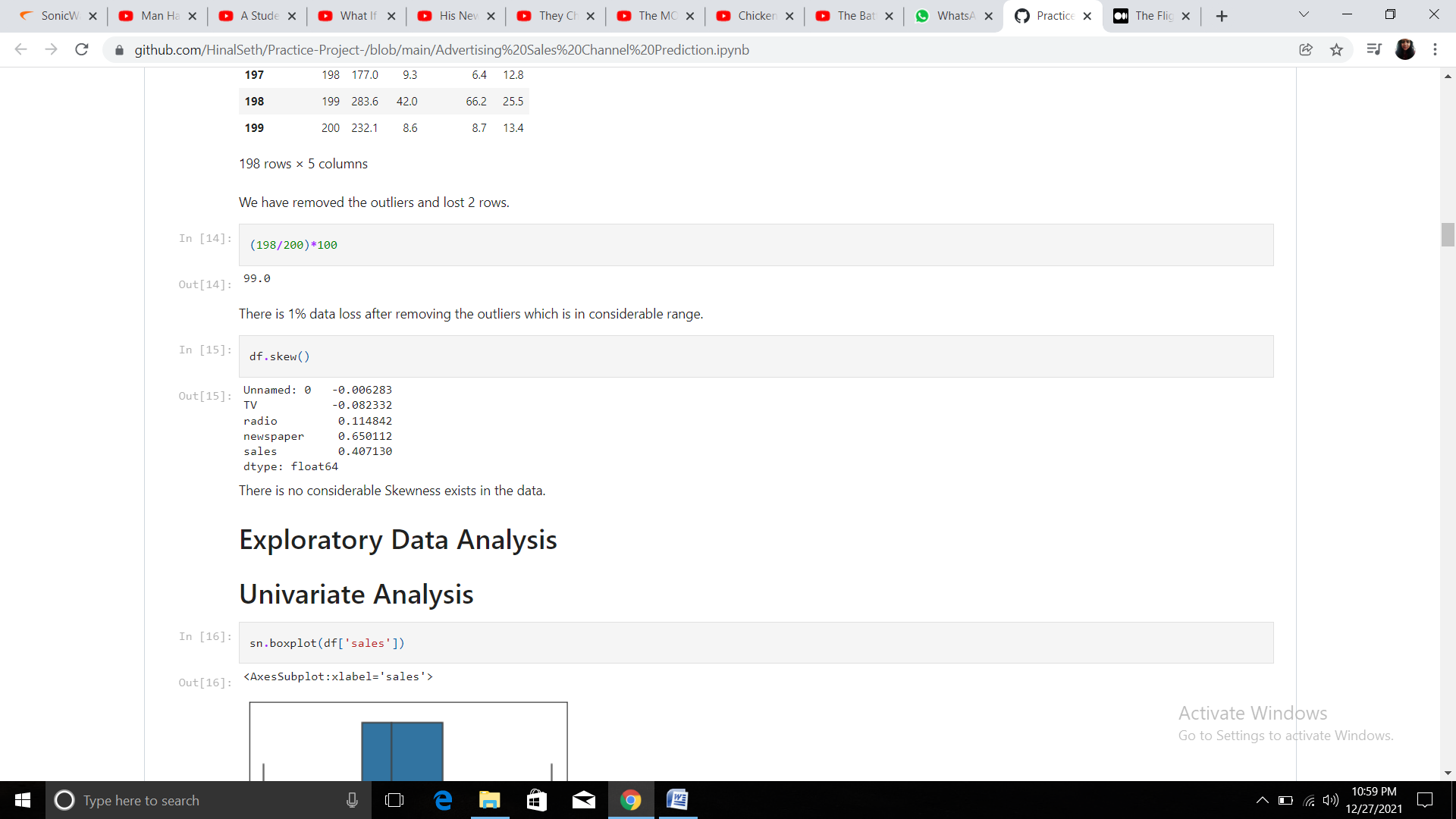
There are no missing values in the dataset and it has only 1% outliers which we will remove and save the new DataFrame with the same name 'df'.

There's no skewness in the dataset





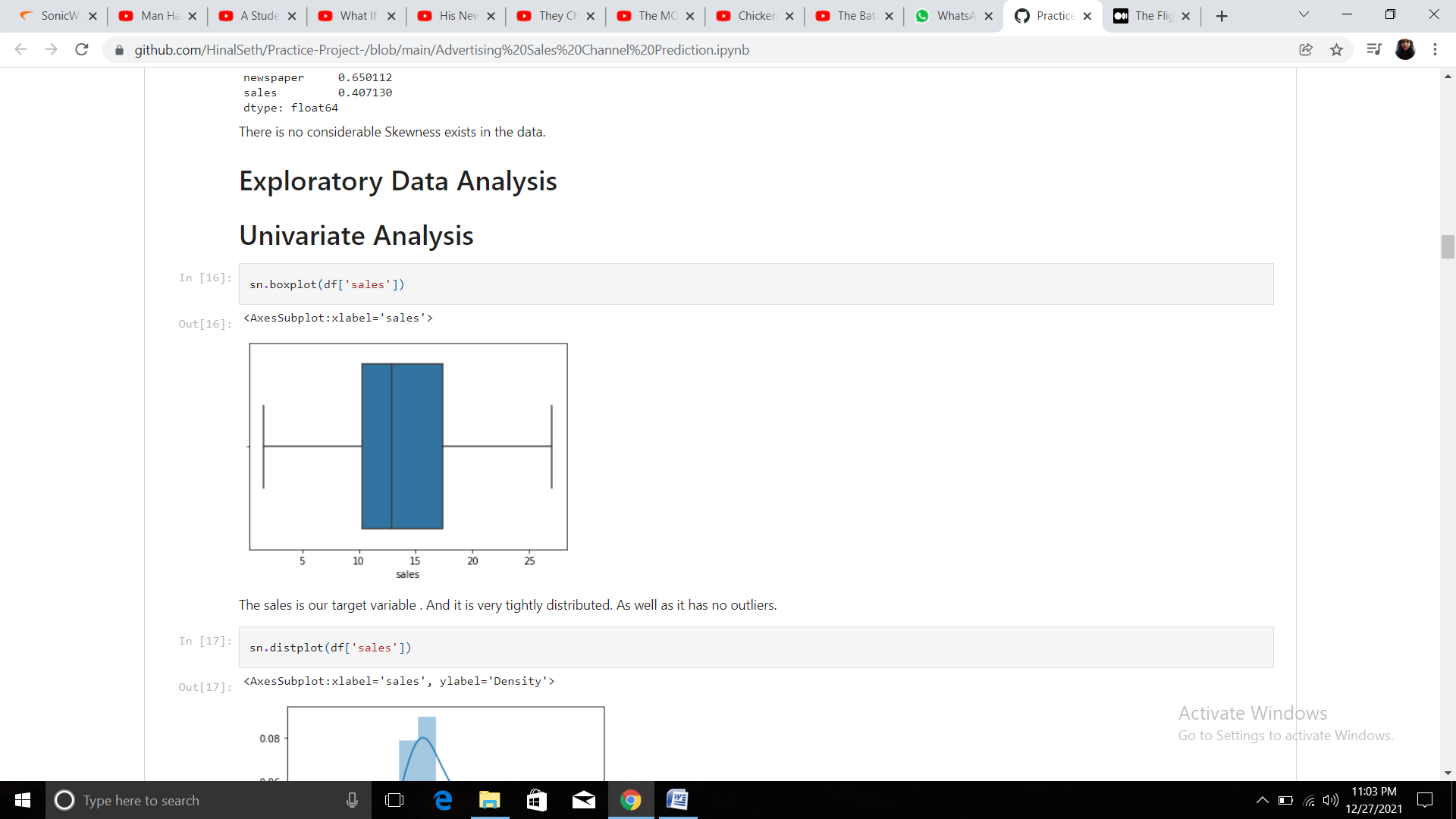


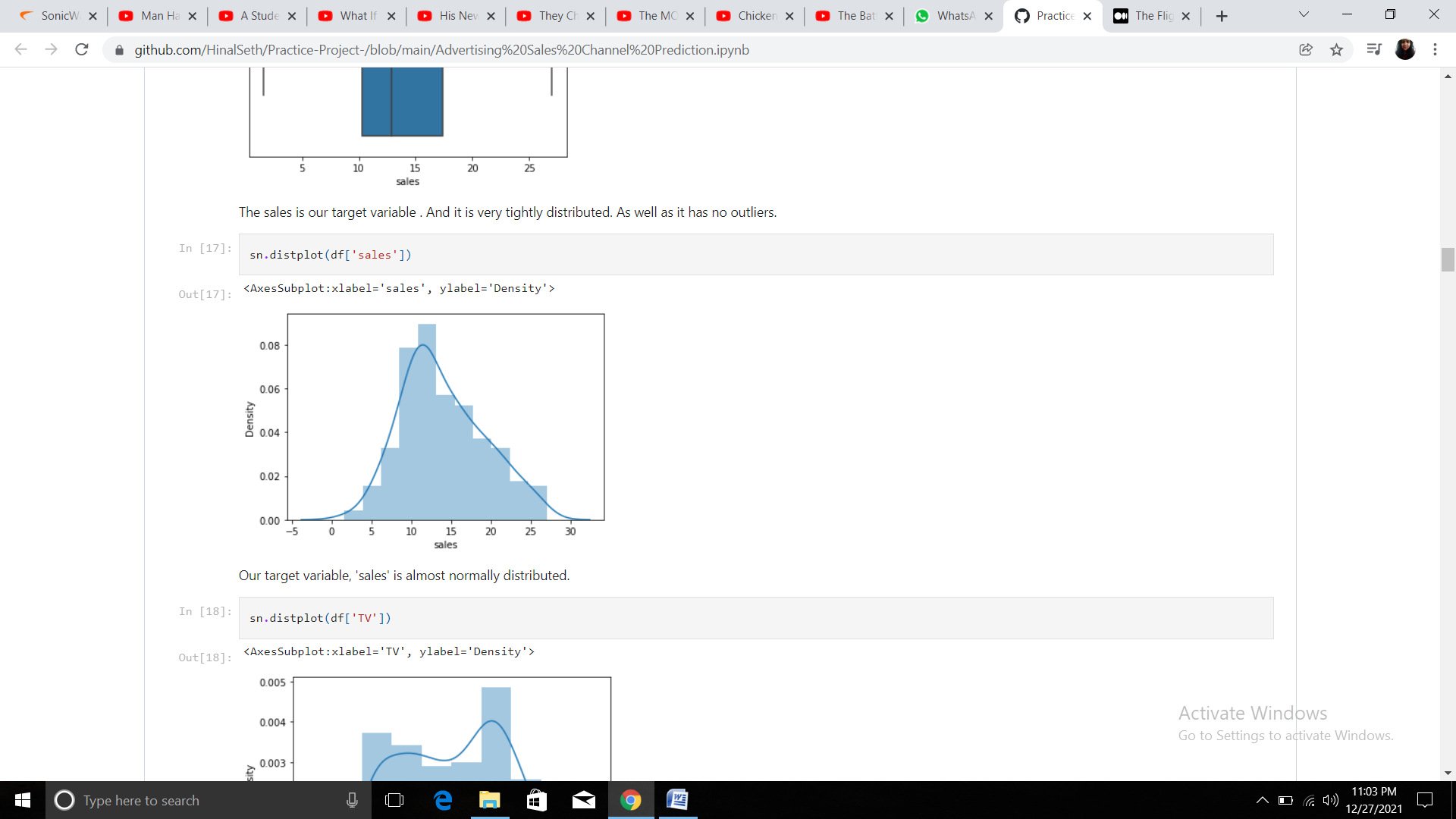


**Exploratory Data Analysis**

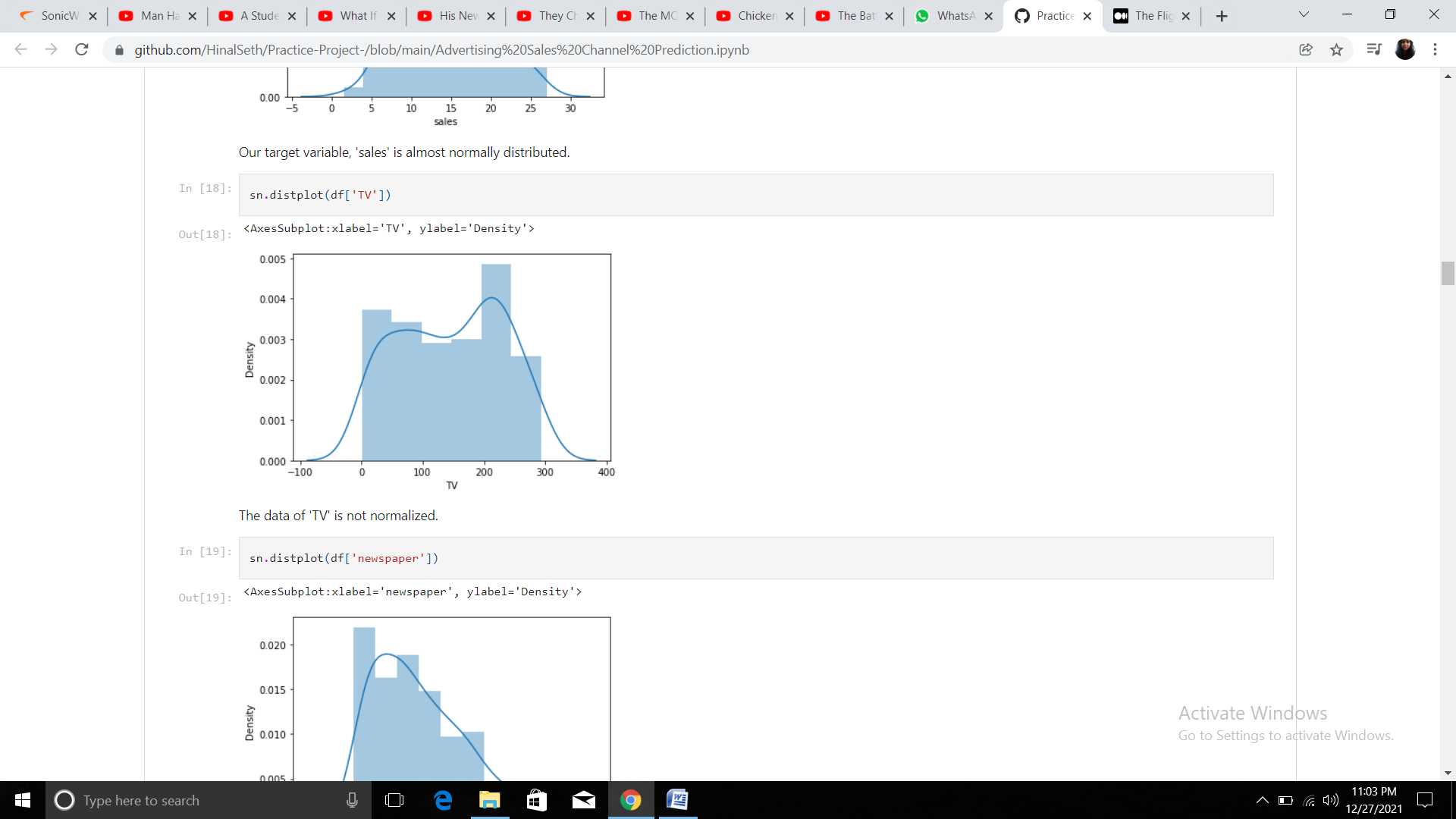
EDA in simple terms means 'trying to understand the given data much better, so that we can make some sense out of it". EDA in Python uses data visualization to draw meaningful patterns and insights. It also involves the preparation of data sets for analysis by removing irregularities in the data. Based on the results of EDA, companies also make business decisions, which can have repercussions later. If EDA is not done properly then it can hamper the further steps in the machine learning model building process. If done well, it may improve the efficacy of everything we do next.

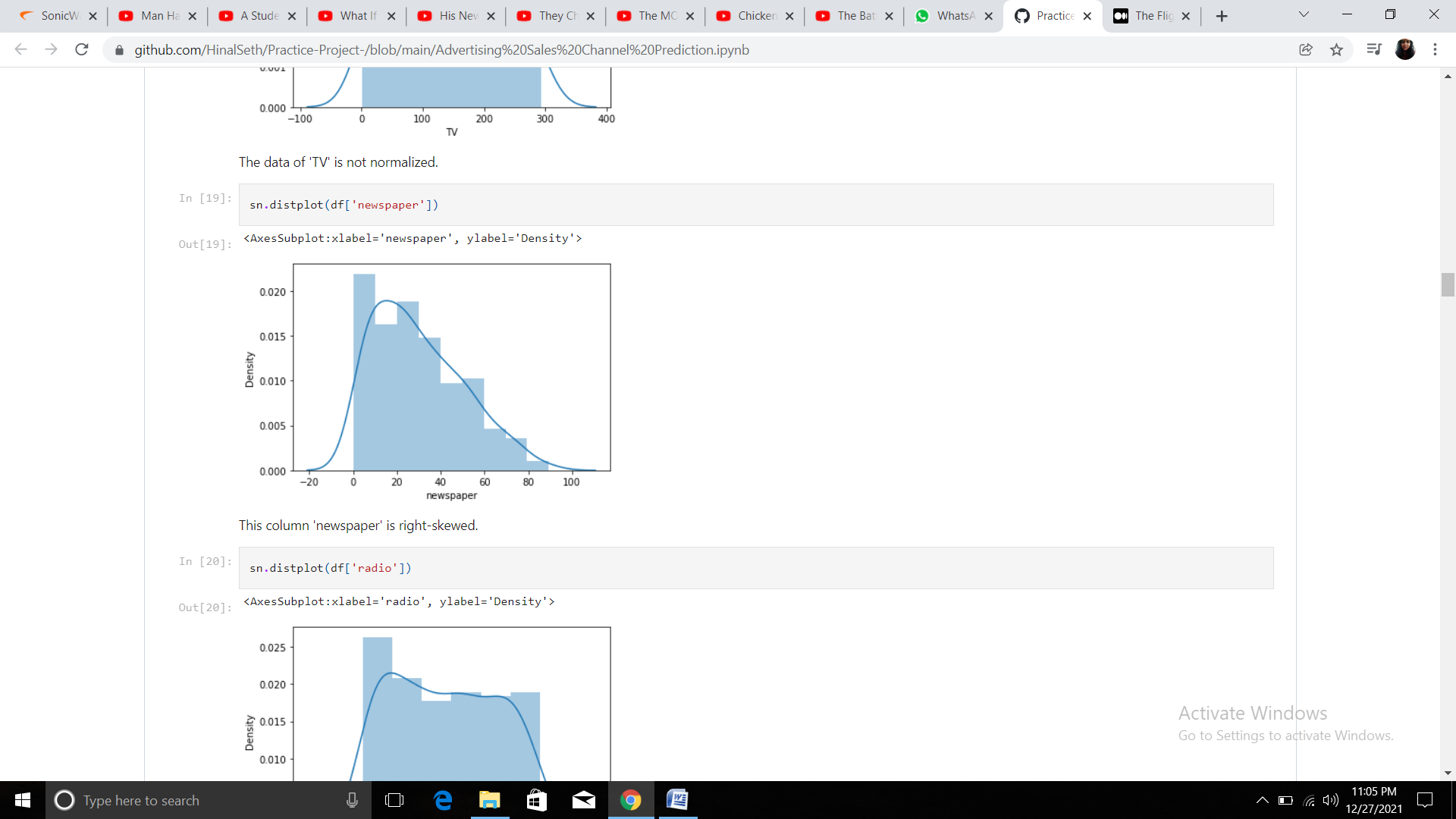
EDA was conducted starting with target variable. The target variables have a very tight distribution of data and don’t have any outliers. This data is almost normalized and doesn't have any skewness.

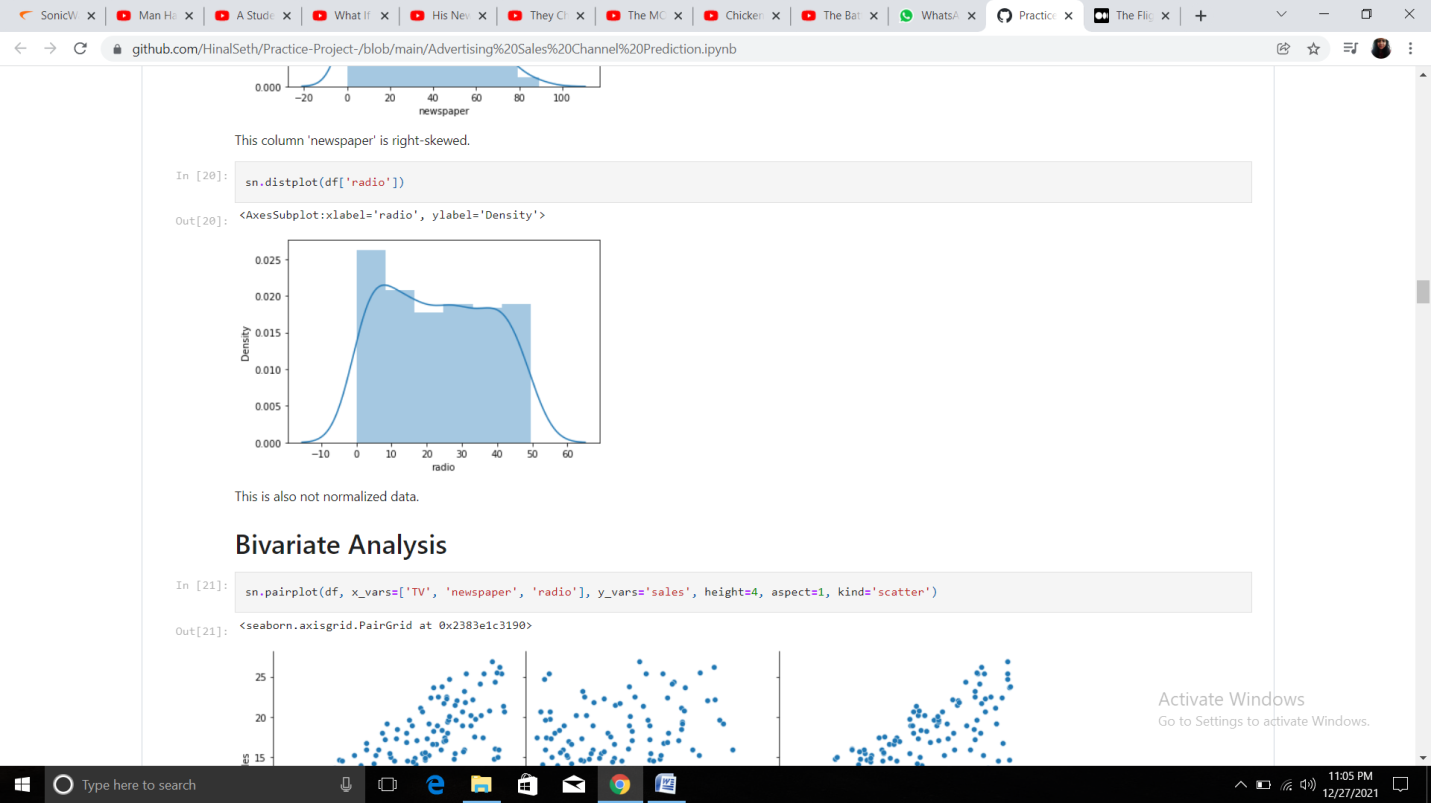




Now we will plot various graphs for dependent variables as well

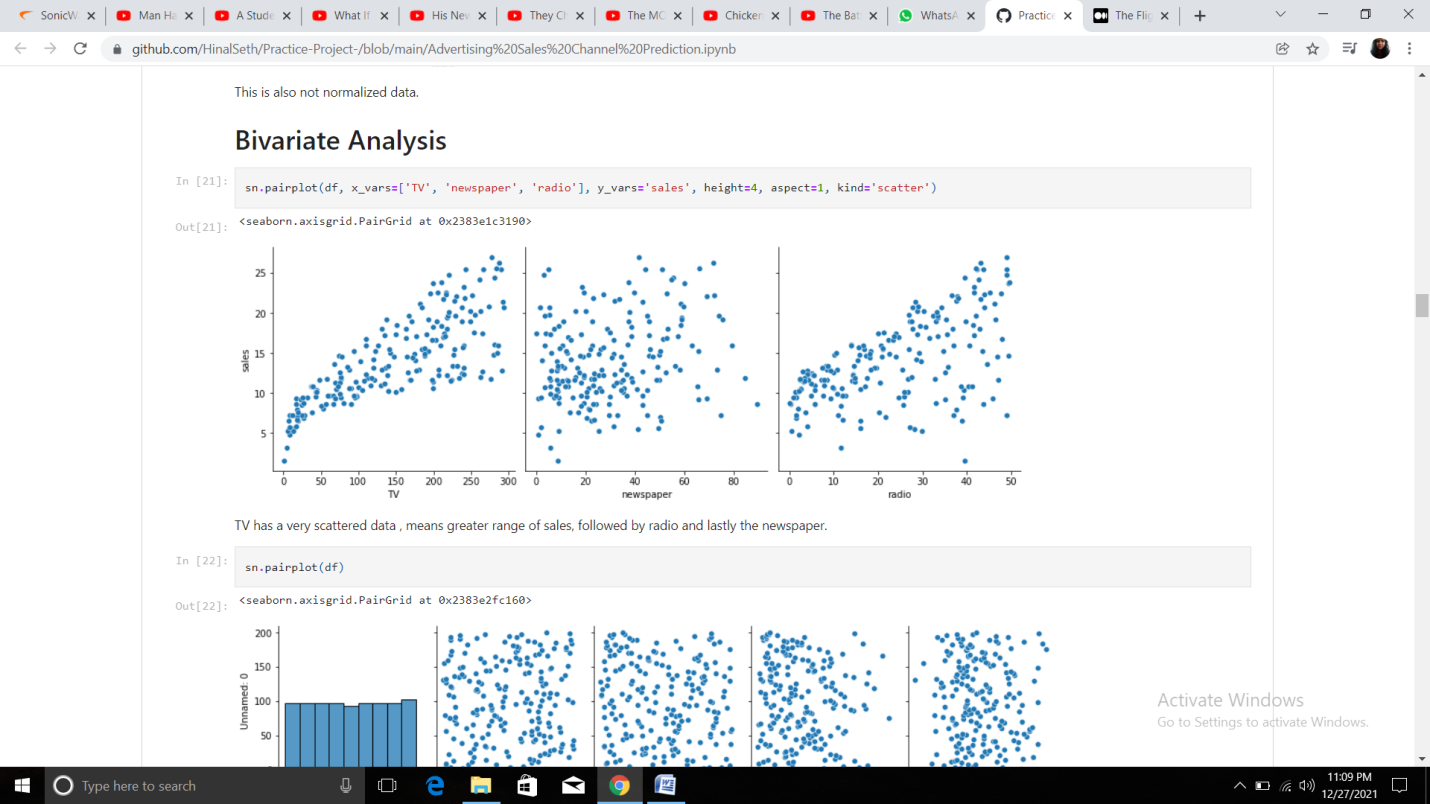






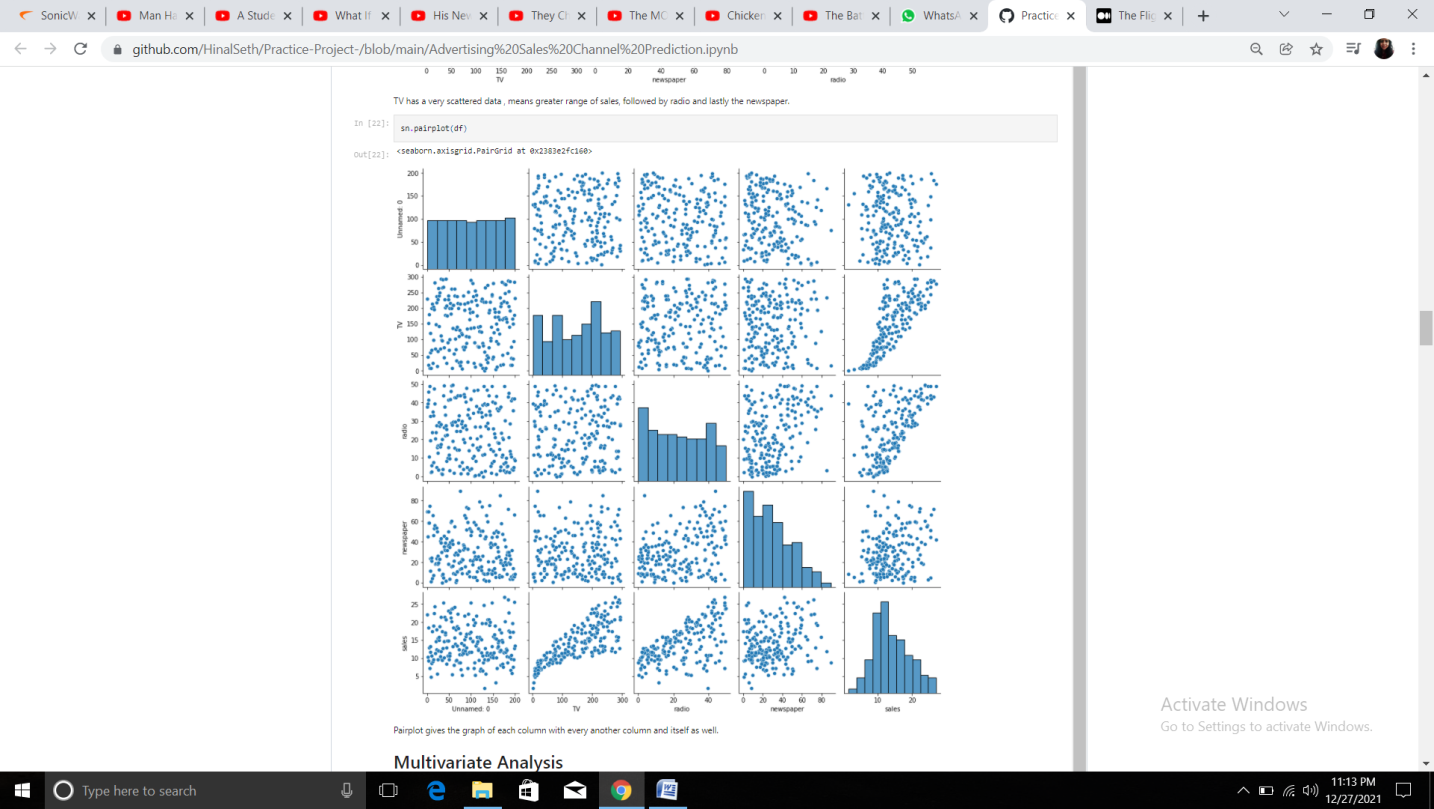
**Bivariate Analysis**

Now we will perform Bivariate Analysis



**Pairplot**

A pairplot plot a pairwise relationships in a dataset. The pairplot function creates a grid of Axes such that each variable in data will by shared in the y-axis across a single row and in the x-axis across a single column. A pairplot allows us to see both distribution of single variables and relationships between two variables . Pair plots are a great method to identify trends for follow-up analysis and, fortunately, are easily implemented in Python.

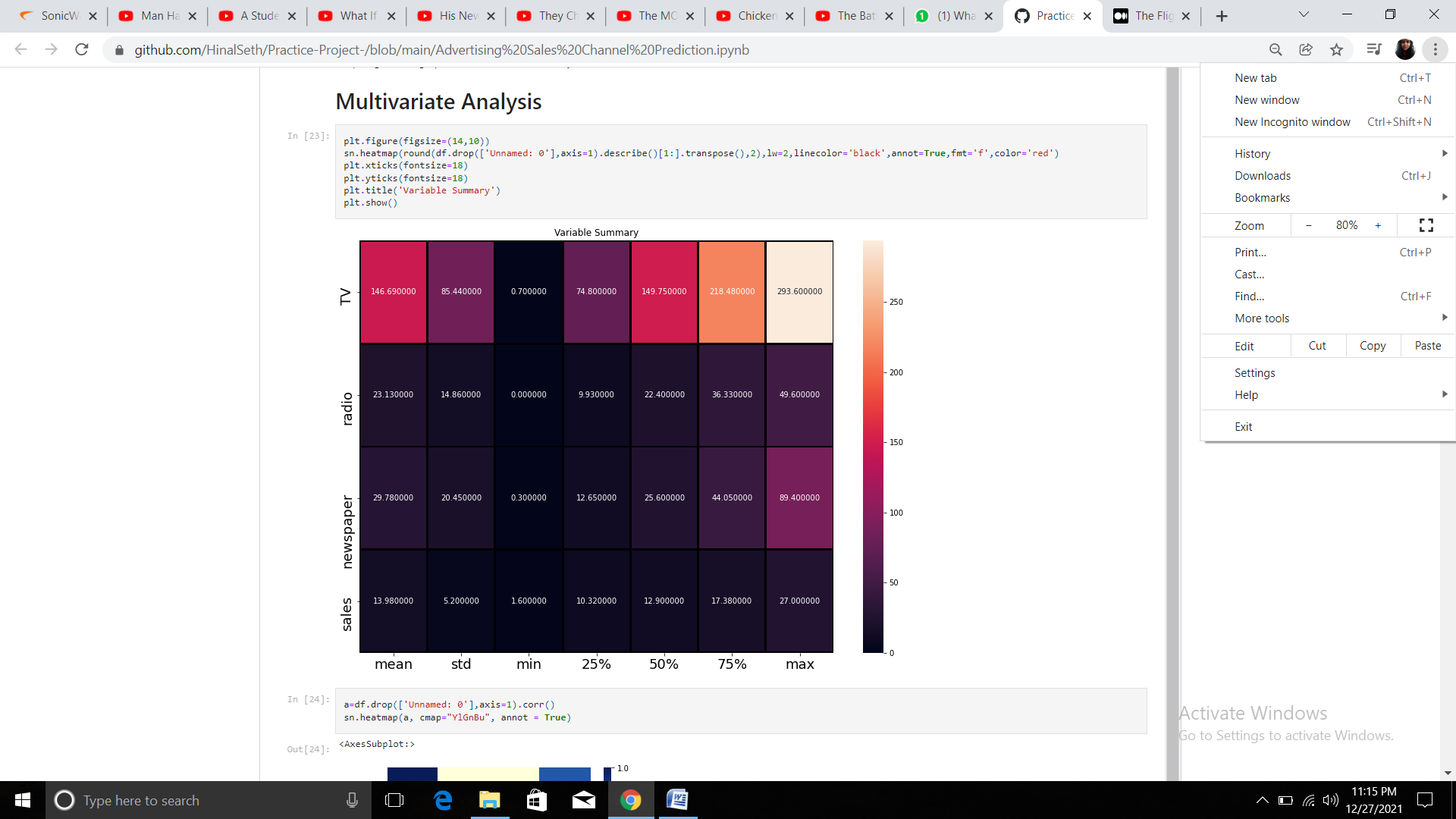


**Heatmap**

A heatmap contains values representing various shades of the same colour for each value to be plotted. Usually the darker shades of the chart represent higher values than the lighter shade. For a very different value a completely different colour can also be used. A heatmap is a matrix kind of 2-dimensional figure which gives a visualisation of numerical data in the form of cells.

Let's plot few heatmaps to understand the data

* Heatmap using describe funsction.
* Heatmap using corr function



**Outcome of heatmap of description of dataset:**

From the above column we are determining mean, standard deviation, median, minimum and maximum of each column.

**TV:**

1. Mean: 146.69

2. Standard Deviation: 85.44

3. Minimum value: 0.70

4. Median: 149.75

5. Maximum Value: 1.00

**Radio:**

1. Mean: 23.13

2. Standard Deviation: 14.86

3. Minimum Value: 0.00

4. Median: 22.40

5. Maximum Value: 49.60

**Newspaper:**

1. Mean: 1.01

2. Standard Deviation: 0.06

3. Minimum Value: 0.73

4. Median: 1:03

5. Maximum Value: 1.07

**Sales:**

1. Mean: 11.98

2. Standard Deviation: 5.20

3. Minimum Value: 1.60

4. Median: 12.90

5. Maximum Value: 27.00

The corr() aggregate function returns a coefficient of correlation between two numbers.

The correlation coefficient is determined by dividing the covariance by the product of the two variables' standard deviations. Standard deviation is a measure of the dispersion of data from its average. Covariance is a measure of how two variables change together.

This function takes as arguments any numeric datatype or any nonnumeric datatype that can be implicitly converted to a numeric datatype.

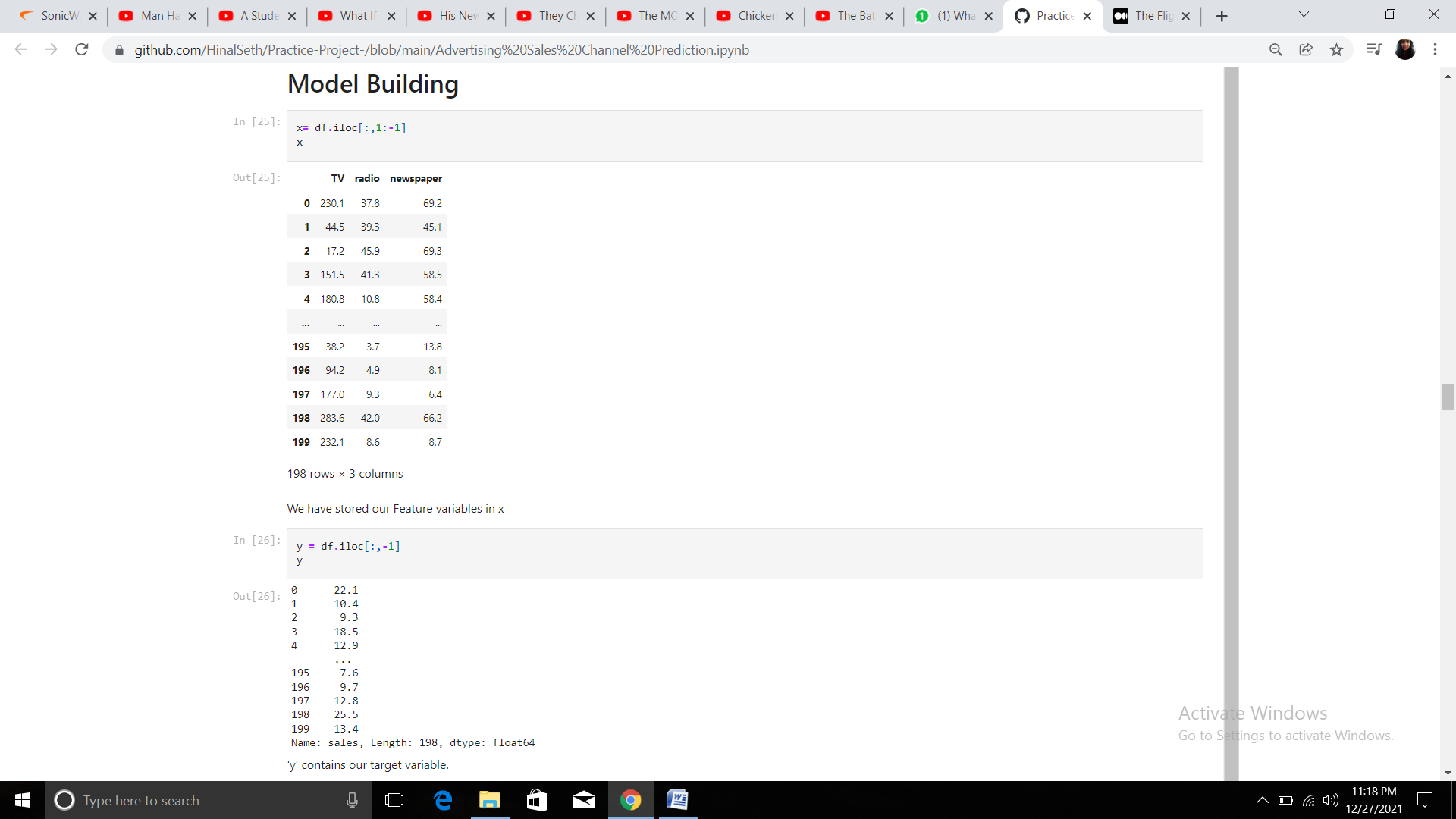
Now let's plot a heatmap to see the correlation between the columns and to check for Multicollinearity between the dependent variables.



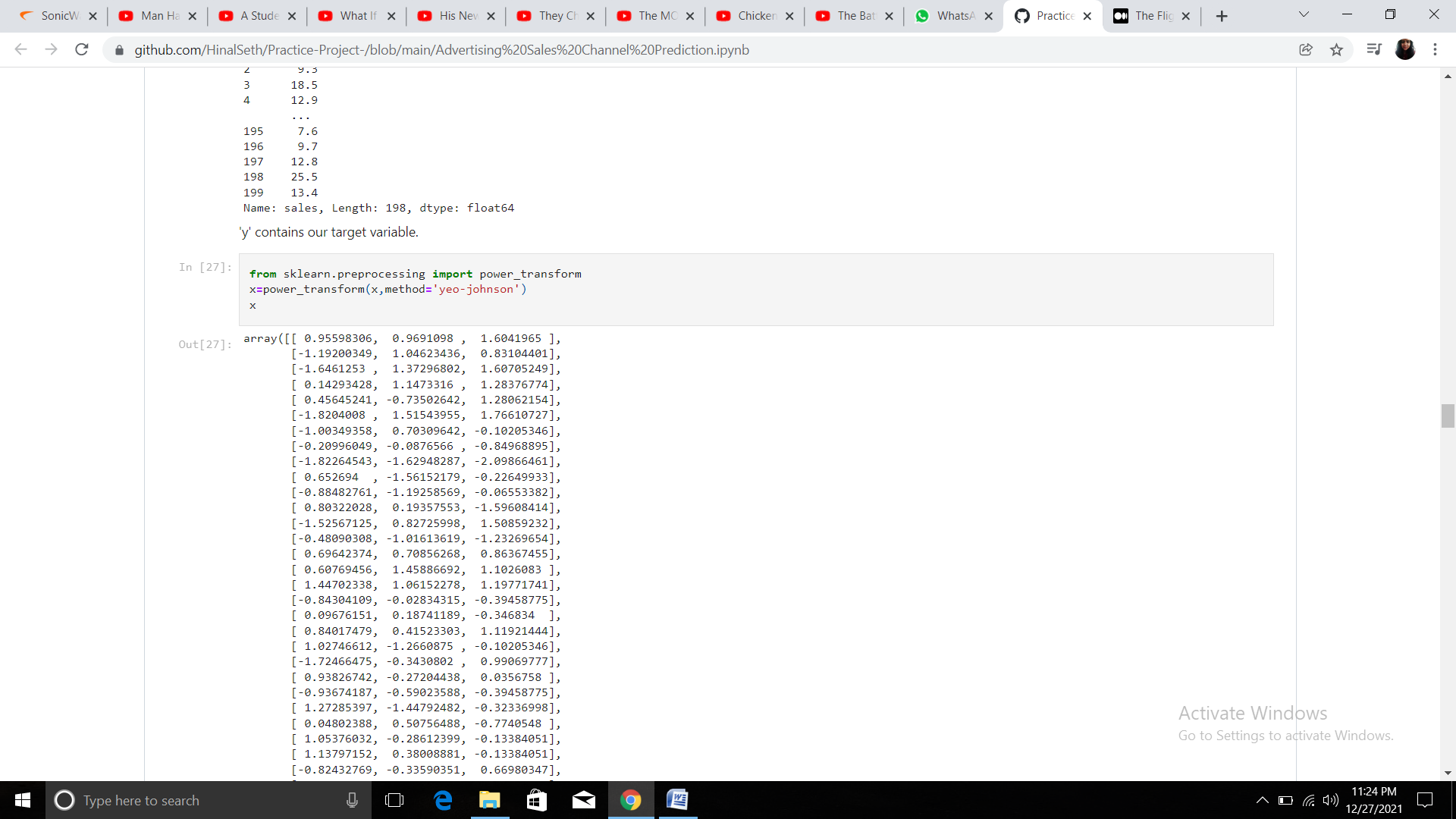
In the output we can see the TV ads have maximum correlation with the generation of sales. This follows by the radio and the newspaper.

**Preprocessing Pipeline**

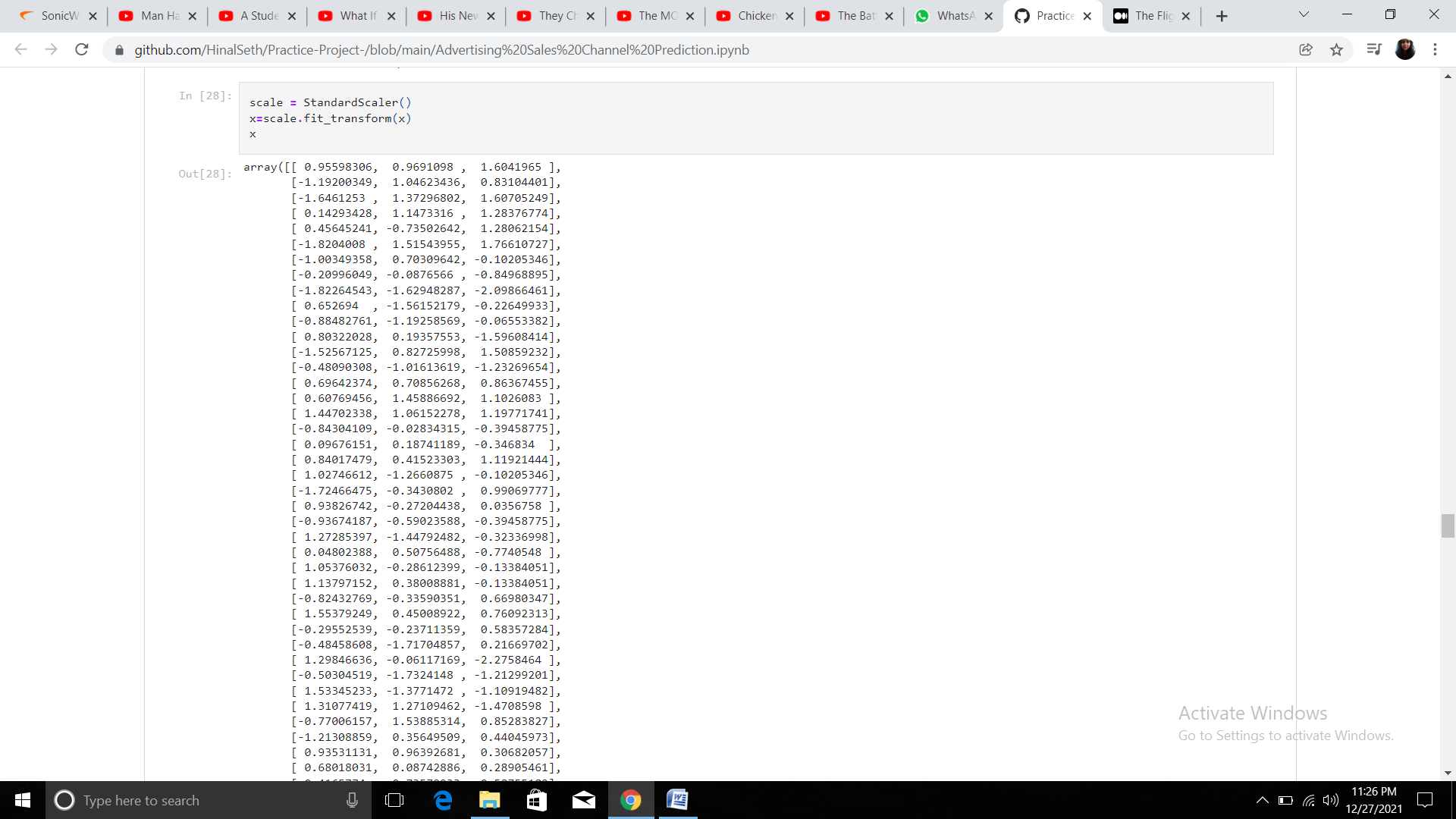
Data preprocessing is a predominant step in machine learning to yield highly accurate and insightful results. Greater the quality of data, the greater is the reliability of the produced results. Incomplete, noisy, and inconsistent data are the inherent nature of real-world datasets. Data preprocessing helps in increasing the quality of data by filling in missing incomplete data, smoothing noise, and resolving inconsistencies. First we will split the data into x and y where x has all the feature variables / dependent variables and y contains target variable/independent variable.



Now we will rescale the data using a power transformation. Scaling the dataset using Standard Scalar. Standard Scalar removes the mean and scales each feature/variable to unit variance. This operation is performed feature-wise in an independent way. StandardScaler can be influenced by outliers (if they exist in the dataset) since it involves the estimation of the empirical mean and standard deviation of each feature.



We have standarized the input/feature variables.



Splitting the dataset into training and testing data by default, sklearn train\_test\_split will make random partitions for the two subsets. However, you can also specify a random state for the operation. It gives four outputs x\_train, x\_test, y\_train and y\_test. The x\_train and x\_test contains the training and testing predictor variables while y\_train and y\_test contains the training and testing target variable. After performing train\_test\_split we have to choose the models to pass the training variable. We can build as many models as we want to compare the accuracy given by these models and to select the best model among them.

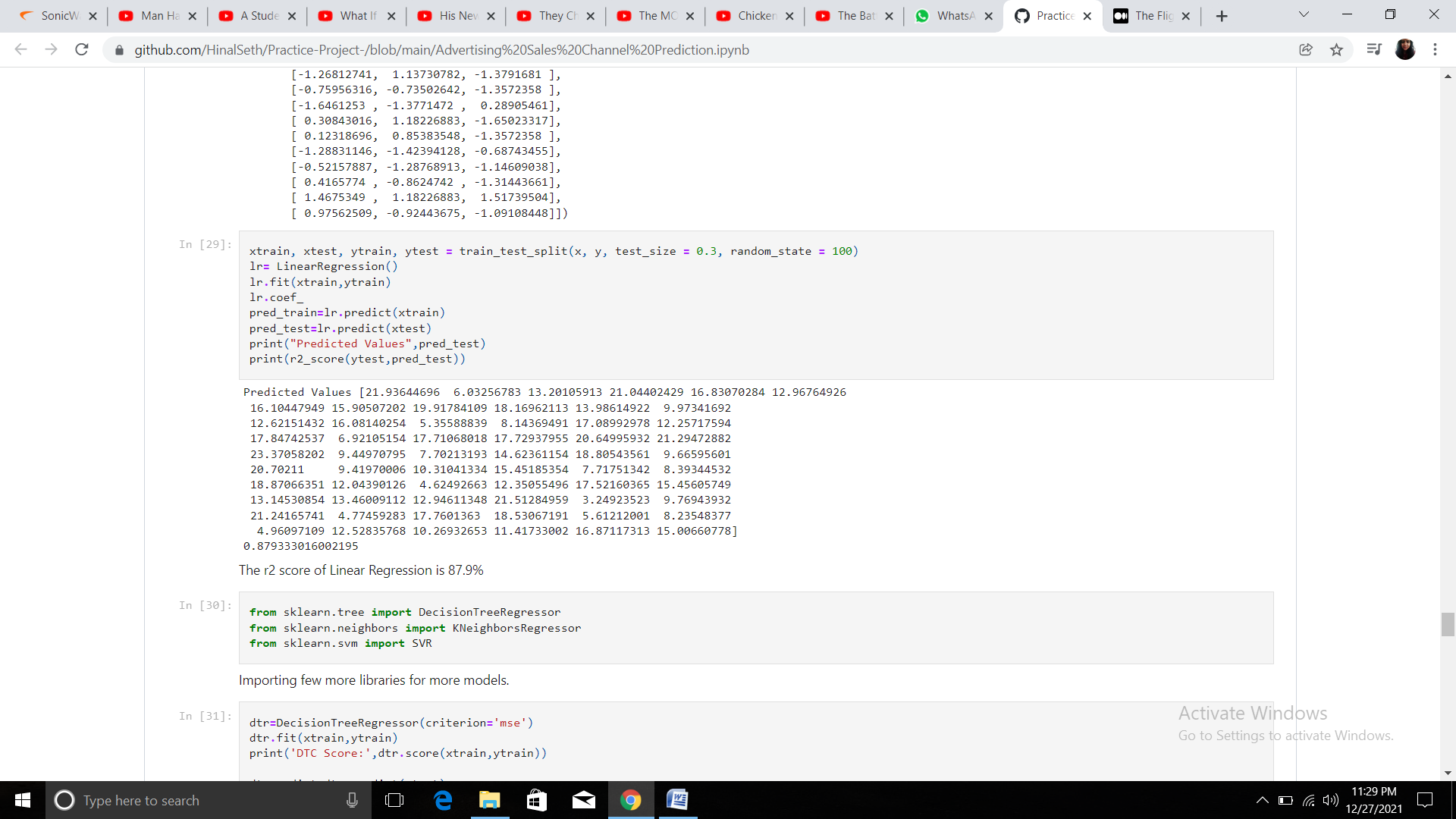
I have selected 5 models:

1. Linear Regression
2. DecisionTreeRegressor
3. KNeighborsRegressor
4. SupportVectorRegression
5. SGDRegressor

**Linear Regression Algorithm**

The term “linearity” in algebra refers to a linear relationship between two or more variables. If we draw this relationship in a two-dimensional space (between two variables), we get a straight line.

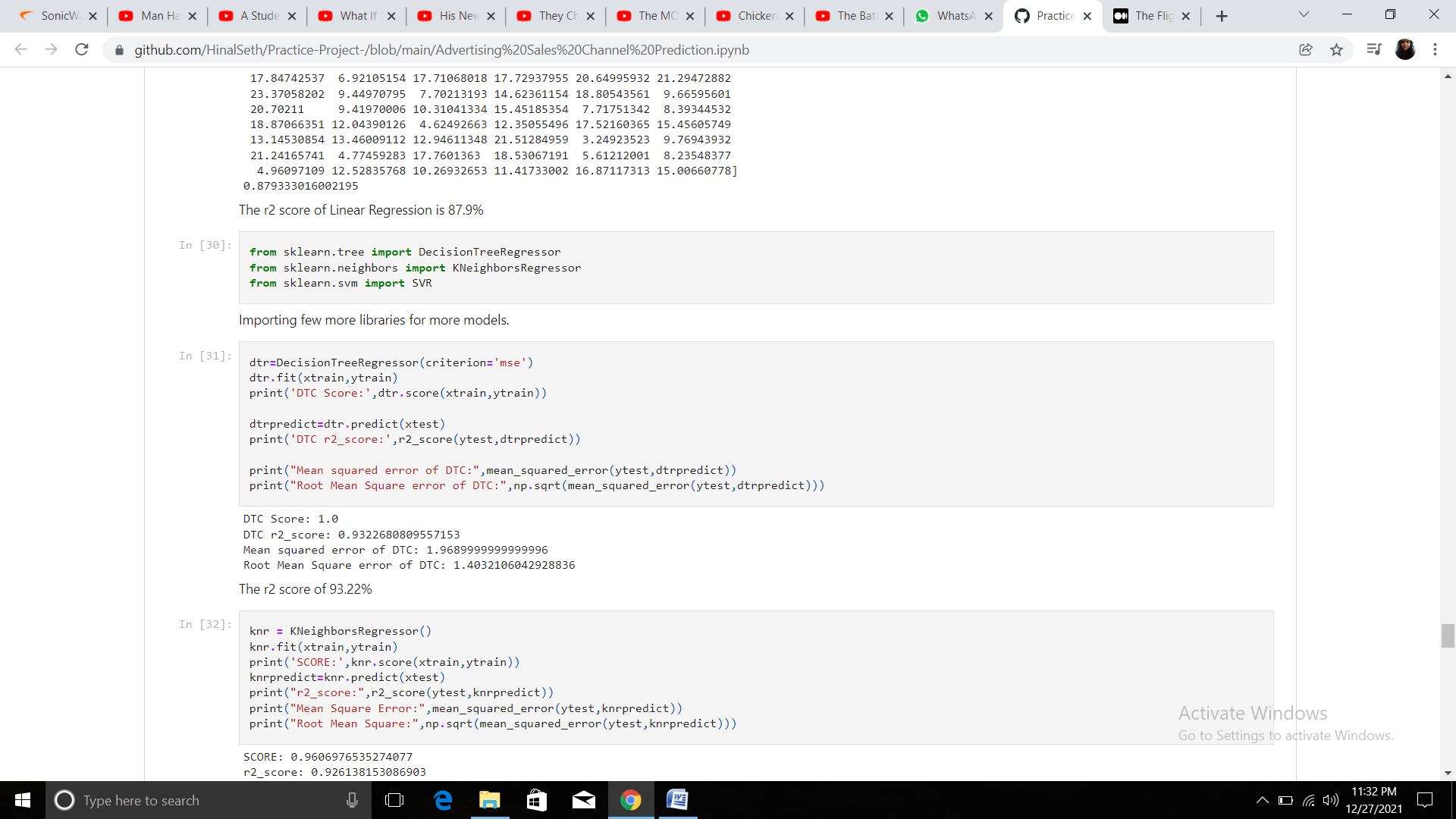
Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression. If we plot the independent variable (x) on the x-axis and dependent variable (y) on the y-axis, linear regression gives us a straight line that best fits the data points



The Linear Regression algorithm is giving us 87.9% accuracy

**DecisionTreeRegressor**

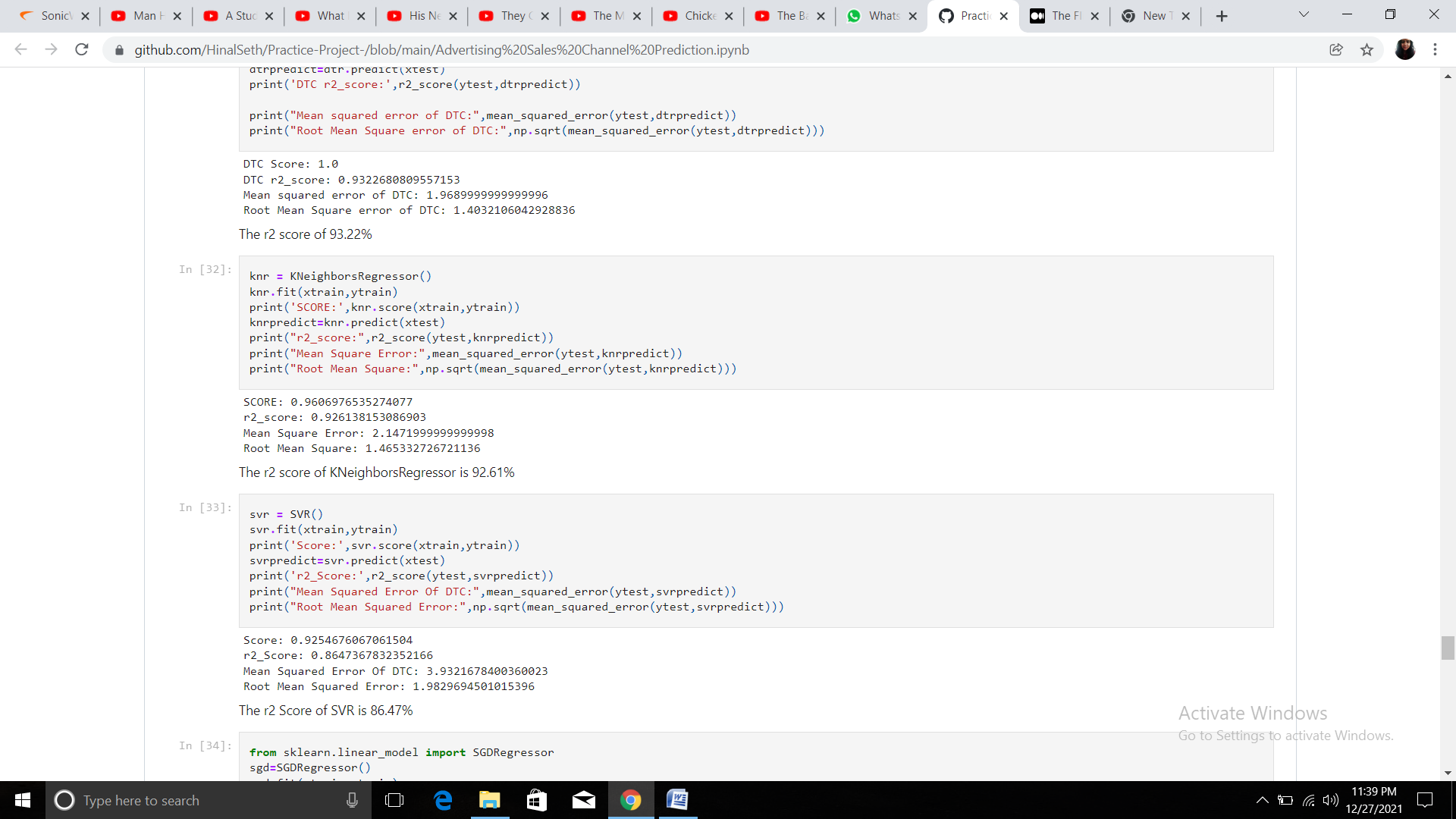
Decision trees can be constructed by an algorithmic approach that can split the dataset in different ways based on different conditions. The two main entities of a tree are decision nodes, where the data is split and leaves, where we get the outcome.



The accuracy of DecisionTreeRegressor is 93.23%

**KNeighborsRegressor**

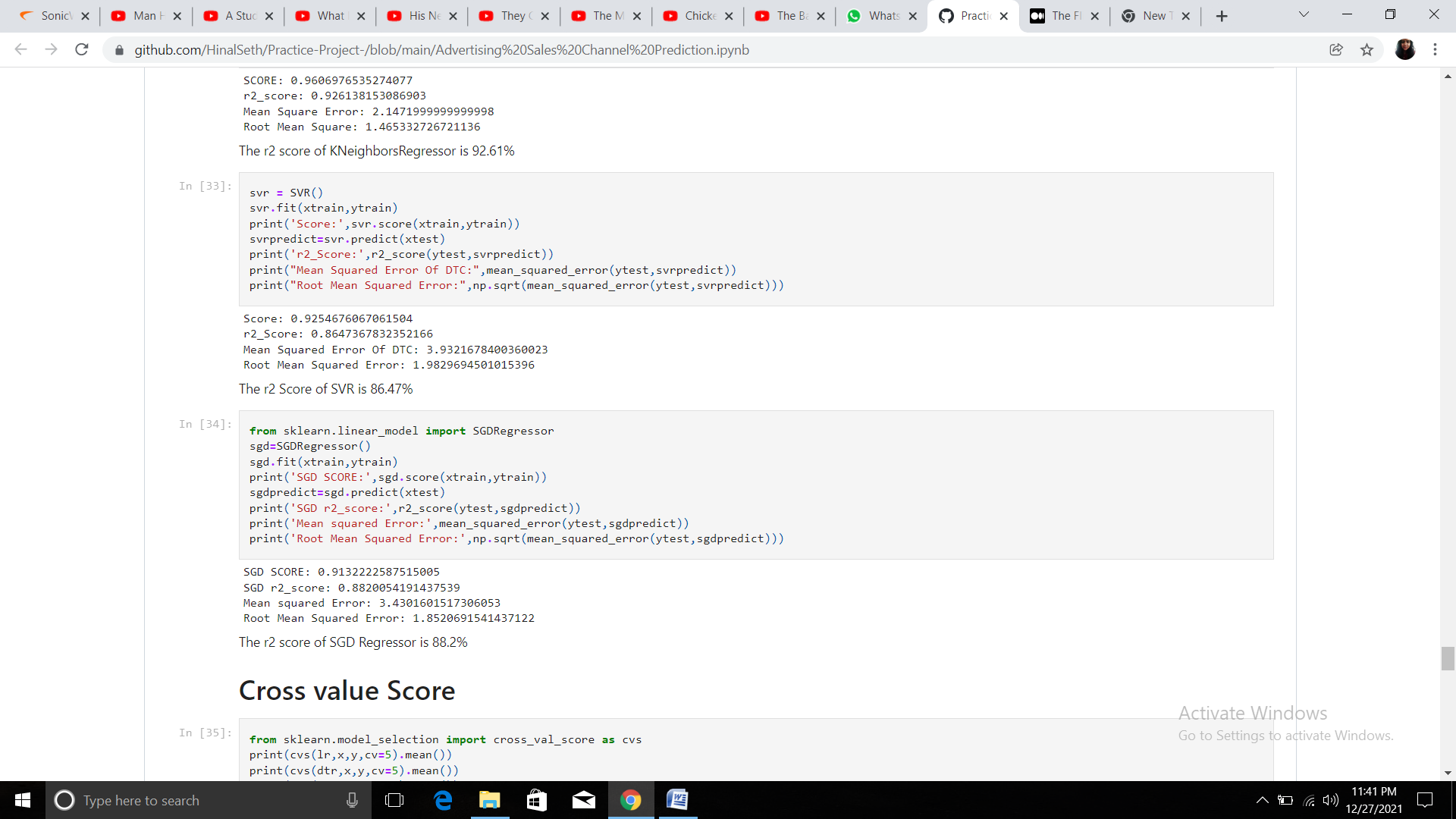
The target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set. Regarding the Nearest Neighbors algorithms, if it is found that two neighbors, neighbor k+1 and k, have identical distances but different labels, the results will depend on the ordering of the training data.



The accuracy of KNeighborsRegressor is 92.61% which is quite good.

**SupportVectorRegression**

Support Vector Regression is a supervised learning algorithm that is used to predict discrete values. Support Vector Regression uses the same principle as the SVMs. The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyperplane that has the maximum number of points.



The accuracy of SupportVectorRegression is 86.47%

**SGDRegressor**

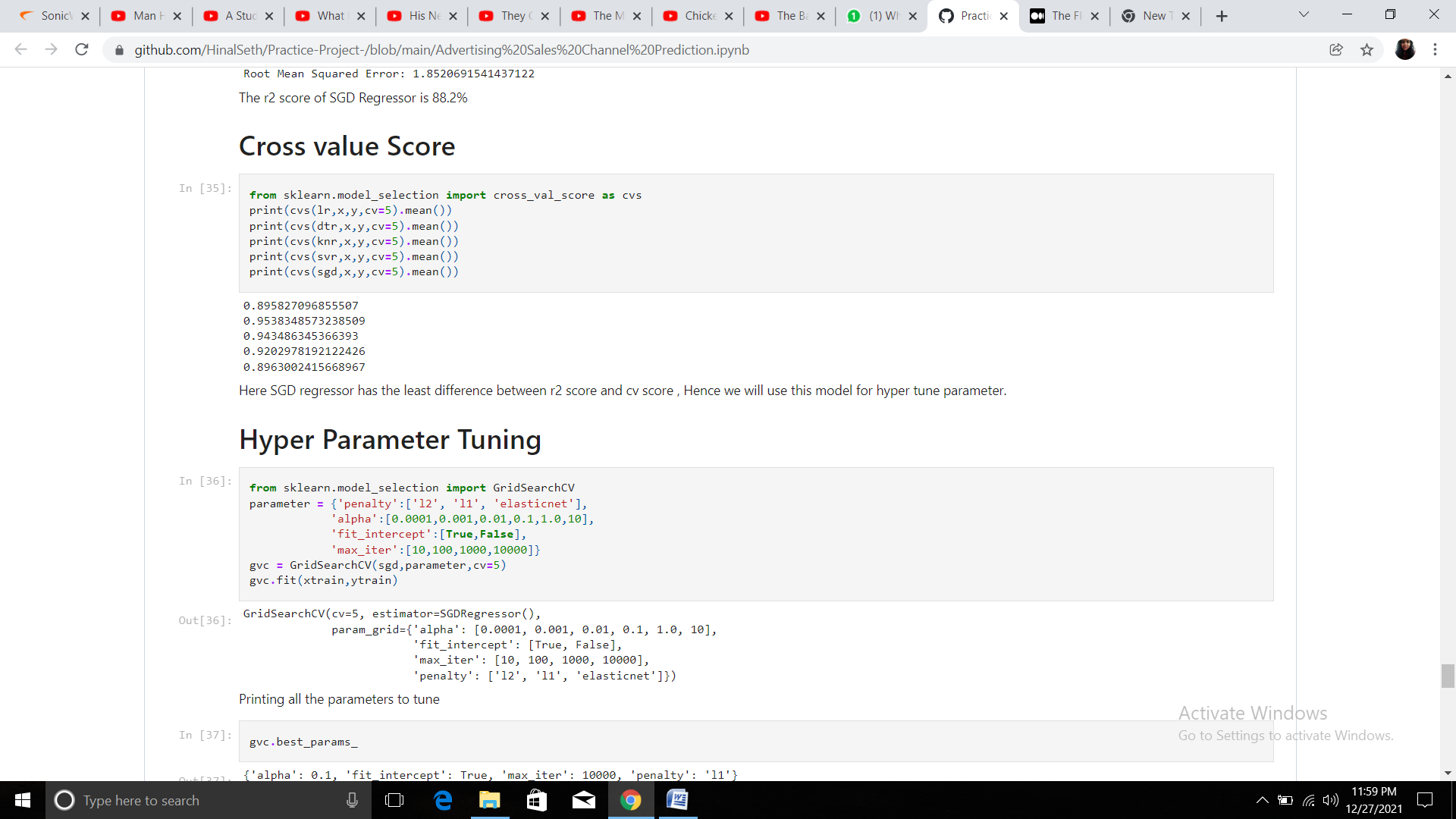
The class SGDRegressor implements a plain stochastic gradient descent learning routine which supports different loss functions and penalties to fit linear regression models.



The accuracy of SGDRegressor is 88.2%

**Cross Validation Score**

Cross Validation is a technique which involves reserving a particular sample of a dataset on which we do not train the model. Later, we test our model on this sample before finalizing it. Cross-Validation is a statistical method of evaluating and comparing learning algorithms by dividing data into two segments: one used to learn or train a model and the other used to validate the model. The purpose of cross–validation is to test the ability of a machine learning model to predict new data. It is also used to flag problems like over fitting or selection bias and gives insights on how the model will generalize to an independent dataset. Advantages of cross-validation: More accurate estimate of out-of-sample accuracy. More “efficient” use of data as every observation is used for both training and testing. Cross Validation is usually a very good way to measure an accurate performance. While it does not prevent our model to over fit, it still measures a true performance estimate. If our model over fits, it will result in worse performance measures. This helps us to choose a best fit model from all the 5 models that we have used.

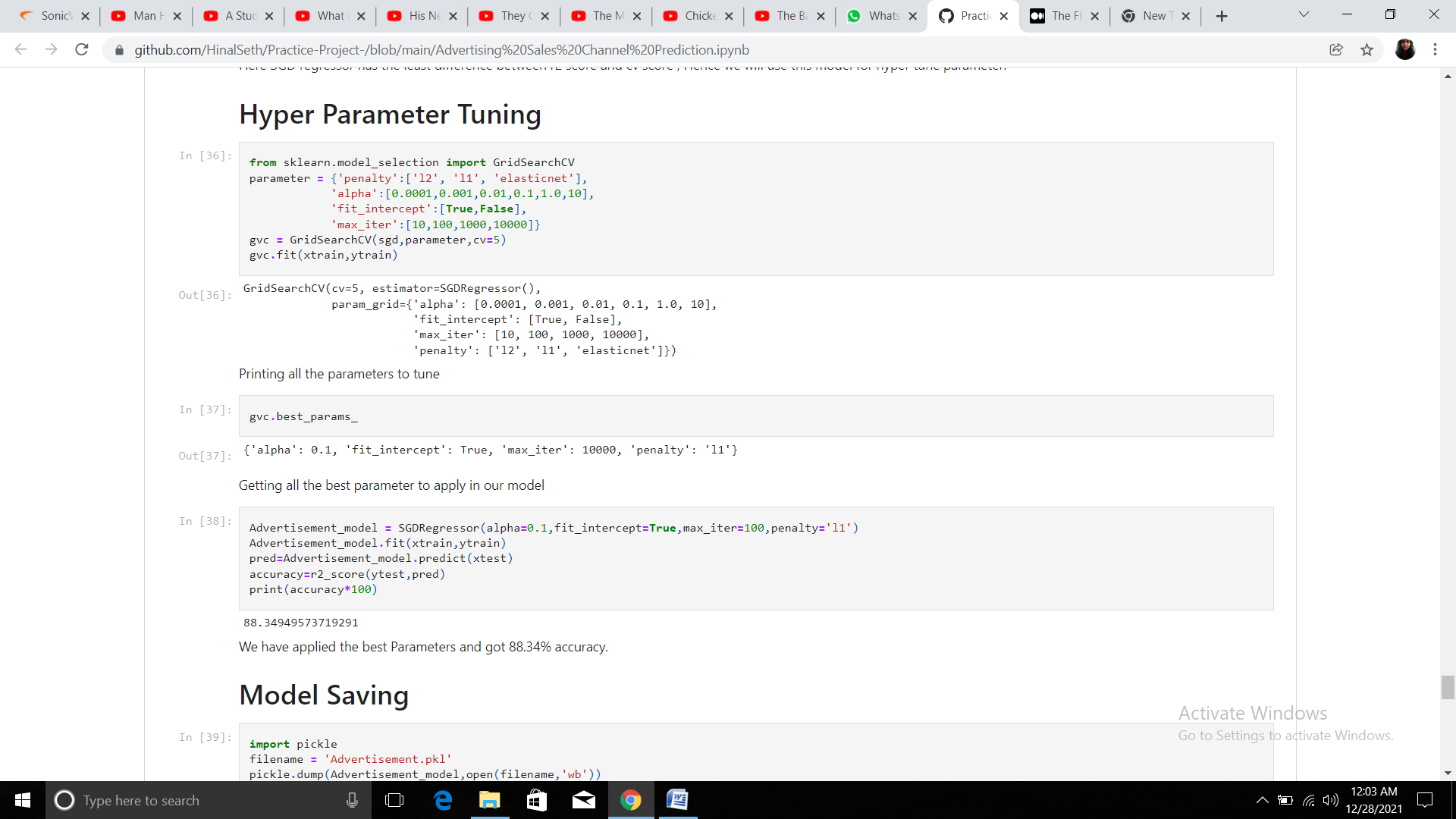


**Hyper parameter tuning**

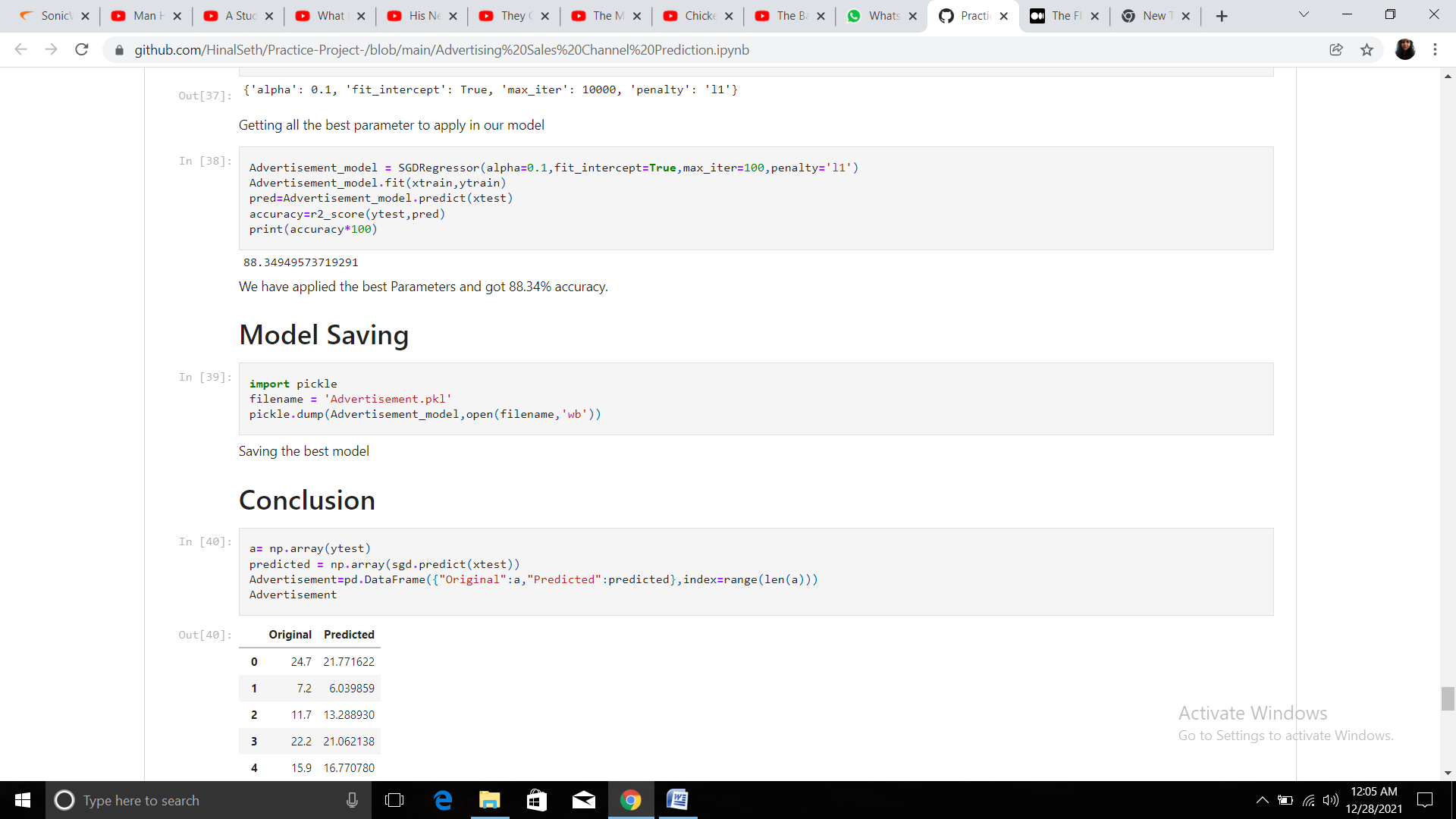
Hyper parameter optimisation in machine learning intends to find the hyper parameters of a given machine learning algorithm that deliver the best performance as measured on a validation set. Hyper parameters, in contrast to model parameters, are set by the machine learning engineer before training. The number of trees in a random forest is a hyper parameter while the weights in a neural network are model parameters learned during training. I like to think of hyper parameters as the model settings to be tuned so that the model can optimally solve the machine learning problem.

We will use GridSearchCV for the hyper parameter tuning.

In the GridSearchCV approach, the machine learning model is evaluated for a range of hyper parameter values. This approach is called GridSearchCV, because it searches for best set of hyper parameters from a grid of hyper parameters values.

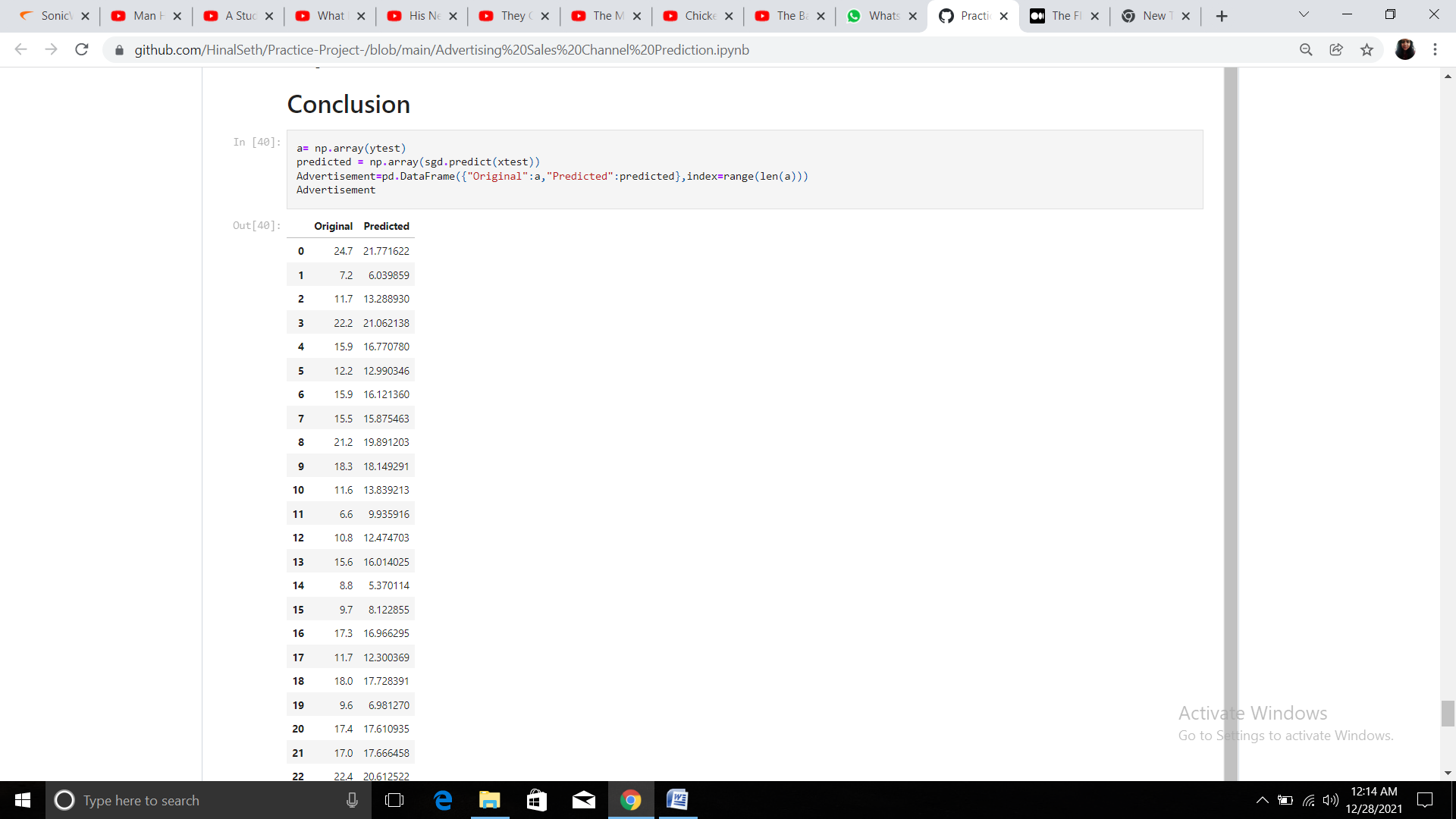


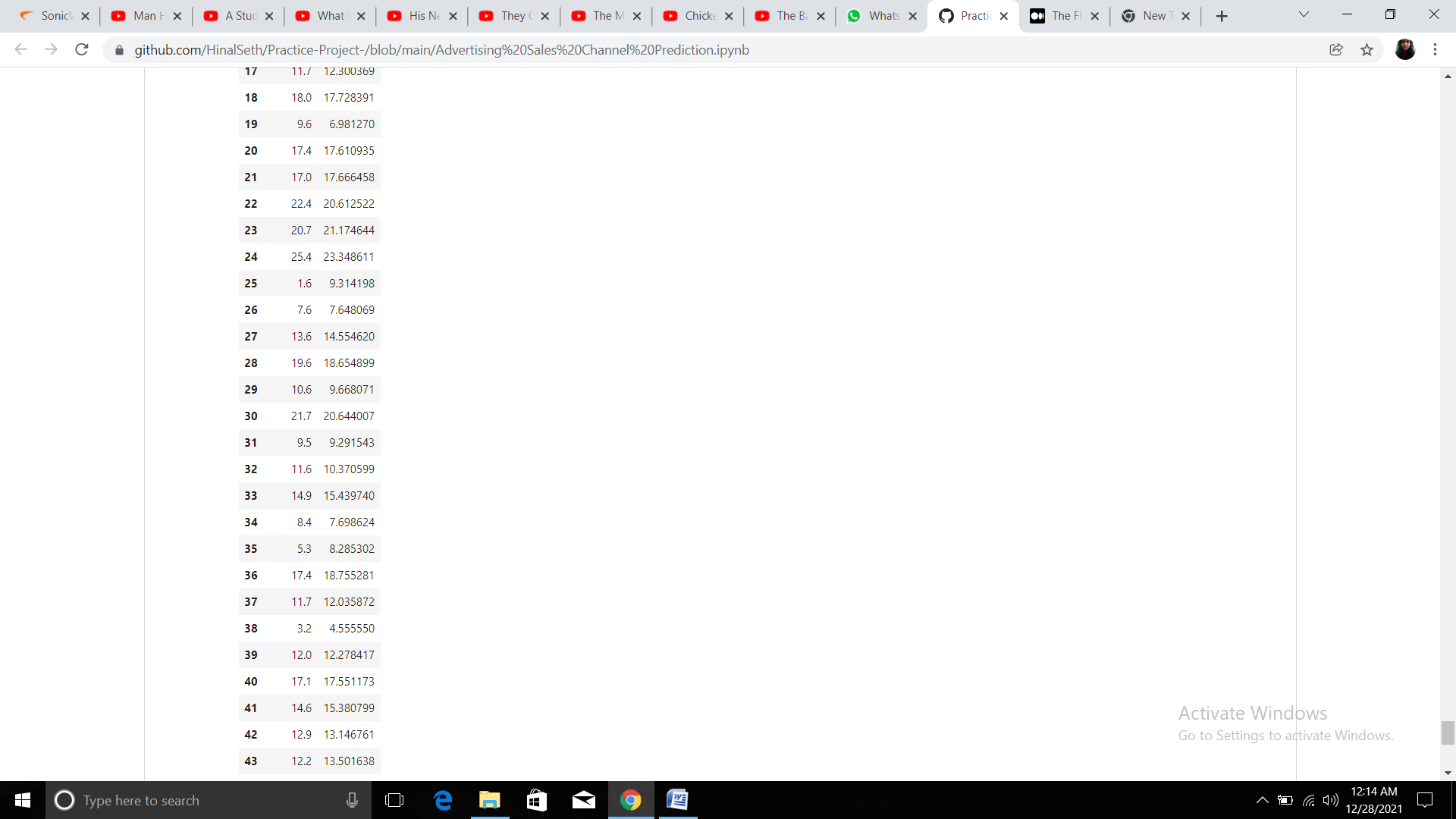
Now we have selected our best fit models have Hypertuned it's parameters, so we will save this model. For this we will have to import pickle library

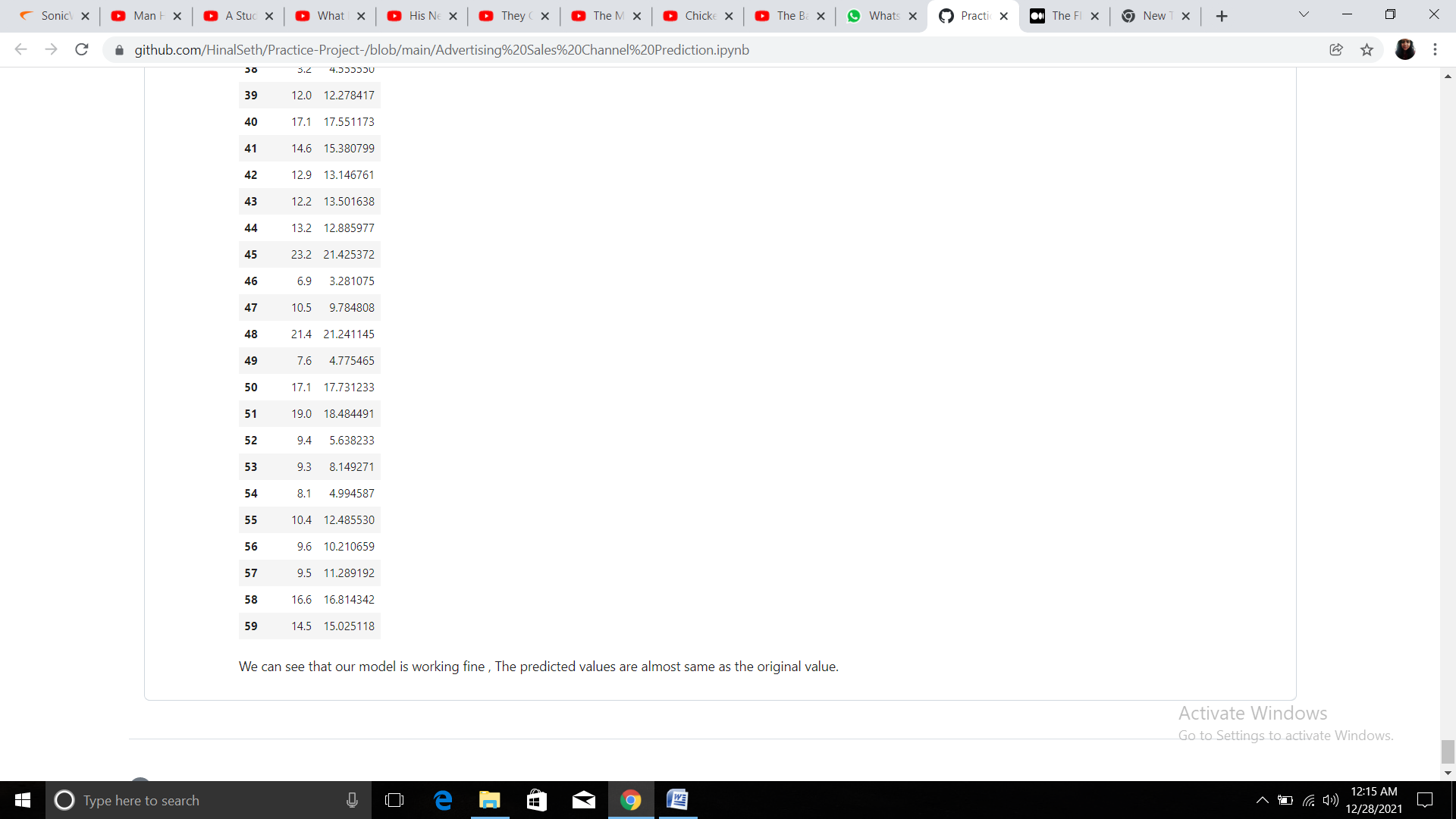


**Conclusion**

We conclude that the highest revenue is generated through the medium of Television Marketing. And also after using the best fit model the original and predicted values are almost same, which infer that this model is working very well.







**Reference**

The above dataset is taken from the following Github link - <https://github.com/dsrscientist/DSData/blob/master/Advertising.csv>