SYNOPSIS

The project entitled **"SALES FORECASTING USING PREDICTION ANALYTICS ALGORITHM"** is planned for providing a complete analysis of sales fore casting. Sales forecasting is an important aspect of different companies engaged in retailing, logistics, manufacturing, marketing and wholesaling. It allows companies to efficiently allocate resources, to estimate achievable sales revenue and to plan a better strategy for future growth of the company.

In this project, prediction of sales of a product from an outlet is performed via a two-level approach that produces better predictive performance compared to any of the popular single model predictive learning algorithms. The approach is performed on Departmental store. The proposed approach was organized into six stages, first is data collection, which includes collecting data and dataset, second is hypothesis definition, which used to analyse the problems, third is data exploration which used to explore the uniqueness of the data, fourth is data cleaning, which is used to detect and correct the inaccurate dataset, fifth is data modelling, which is used to predict the data using machine learning techniques, sixth is feature engineering, which is used to import the data from machine learning algorithm. And, finally validating and implementation of our results using precision and accuracy techniques. The result is demonstrated in two-level statistical approach to perform better than a single model approach as provided with more information that leads to better prediction.

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

Sales is a life blood of every company and sales forecasting plays a vital role in conducting any business. Good forecasting helps to develop and improve business strategies by increasing the knowledge about the marketplace. A standard sales forecast looks deeply into the situations or the conditions that previously occurred and then, applies inference regarding customer acquisition, identifies inadequacy and strengths before setting a budget as well as marketing plans for the upcoming year.

In other words, sales forecasting is sales prediction that is based on the available resources from the past. An in-depth knowledge of the past resources allows to prepare for the upcoming needs of the business and increases the likelihood to succeed irrespective of external circumstances. Businesses that treat sales forecasting as the primary step, tend to perform better than those data mining predictive techniques via stacking is considered a two-level statistical approach. It is named as two-level because stacking is performed on two layers in which bottom layer consists of one or more than one learning algorithms and top layer consists of one learning algorithm.

Stacking is also known as Stacked Generalization. It basically involves the training of the learning algorithm present in the top layer to combine the predictions made by the algorithms present in the bottom layer. In the first step, all the learning algorithms are trained using the departmental store dataset and in the second step, Stacking performs better than any single model because a stacking involves more information for prediction.

In this project the approach has been done under six stages. In first stage, data is collected from dataset. In second stage, problems are analysed from the data collection. In third stage, uniqueness of the data is explored. In fourth stage, data cleaning is done to detect and correct the dataset. In fifth stage, data modelling techniques is used to predict the data. In sixth stage, the feature engineering is used to import the data from the machine learning algorithm. Sales prediction is done accurately by using machine learning algorithms.

1.2 SYSTEM SPECIFICATION

1.2.1HARDWARE CONFIGURATION

Processor	:	Intel Core i3
RAM Capacity	:	4 GB
Hard Disk	:	90 GB
Mouse	:	Logical Optical Mouse
Keyboard	:	Logitech 107 Keys
Monitor	:	15.6 inch
Mother Board	:	Intel
Speed	:	3.3GHZ

1.2.2 SOFTWARE CONFIGURATION

Operating System	:	Windows 10
Front End	:	PYTHON
Middle Ware	:	ANACONDA (JUPYTER NOTEBOOK)
Back End	:	.CSV FILE

ABOUT SOFTWARE

PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Python Features

Python has few keywords, simple structure, and a clearly defined syntax. Python code is more clearly defined and visible to the eyes. Python's source code is fairly easy-to-maintaining. Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh. Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable

Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Extendable

It allows to add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

Databases

Python provides interfaces to all major commercial databases.

GUI Programming

Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Scalable

Python provides a better structure and support for large programs than shell scripting.

Object-Oriented Approach

One of the key aspects of Python is its object-oriented approach. This basically means that Python recognizes the concept of class and object encapsulation thus allowing programs to be efficient in the long run.

Highly Dynamic

Python is one of the most dynamic languages available in the industry today. There is no need to specify the type of the variable during coding, thus saving time and increasing efficiency.

Extensive Array of Libraries

Python comes inbuilt with many libraries that can be imported at any instance and be used in a specific program.

Open Source and Free

Python is an open-source programming language which means that anyone can create and contribute to its development. Python is free to download and use in any operating system, like Windows, Mac or Linux.

ANACONDA

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system .The Anaconda distribution includes data-science packages suitable for Windows, Linux, and MacOS.

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, MacOS and Linux.

JUPYTER NOTEBOOK

"Jupyter" is a loose acronym meaning Julia, Python, and R. These programming languages were the first target languages of the Jupyter application. As a server-client application, the Jupyter Notebook App allows you to edit and run your notebooks via a web browser. The application can be executed on a PC without Internet access, or it can be installed on a remote server and it can access through the Internet.

A kernel is a program that runs and introspects the user's code. The Jupyter Notebook App has a kernel for Python code. "Notebook" or "Notebook documents" denote documents that contain both code and rich text elements, such as figures, links, equations. The mix of code and text elements, these documents are the ideal place to bring together an analysis description, and can be executed to perform the data analysis in real time.

Jupyter Notebook contains two components such as web application and notebook documents.

A web application is a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output. Notebook documents is a representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects.

Structure of a notebook document

The notebook consists of a sequence of cells. A cell is a multiline text input field, and its contents can be executed by using Shift-Enter, or by clicking either the “Play” button the toolbar, or **Cell**, **Run** in the menu bar. The execution behavior of a cell is determined by the cell’s type. There are three types of cells namely code cells, markdown cells, and raw cells. Every cell starts off being a code cell, but its type can be changed by using a drop-down on the toolbar.

Code cells

A code cell allows you to edit and write new code, with full syntax highlighting and tab completion. The programming language you use depends on the kernel, and the default kernel (IPython) runs Python code.

Markdown cells

Document the computational process in a literate way, alternating descriptive text with code, using rich text. In IPython this is accomplished by marking up text with the Markdown language. The corresponding cells are called Markdown cells. The Markdown language provides a simple way to perform this text mark-up, to specify which parts of the text should be emphasized (italics), bold, form lists, etc.

Raw cells

Raw cells provide a place in which you can write output directly. Raw cells are not evaluated by the notebook. When passed through nbconvert, raw cells arrive in the destination format unmodified.

MICROSOFT EXCEL

Microsoft Excel is a spreadsheet developed by Microsoft for Windows, MacOS, Android and iOS. It features calculation, graphing tools, pivot tables and a macro programming language called Visual Basic for applications.

FEATURES

Basic Operation

Microsoft Excel has the basic features of all spreadsheets, using a grid of cells arranged in numbered rows and letter-named columns to organize data manipulations like arithmetic operations. It has a battery of supplied functions to answer statistical, engineering and financial needs. In addition, it can display data as line graphs, histograms and charts, and with a very limited three-dimensional graphical display.

VBA programming

The Windows version of Excel supports programming through Microsoft's Visual Basic for Applications (VBA), which is a dialect of Visual Basic. Programmers may write code directly using the Visual Basic Editor (VBE), which includes a window for writing code, debugging code, and code module organization environment. The user can implement numerical methods as well as automating tasks such as formatting or data organization in VBA and guide the calculation using any desired intermediate results reported back to the spreadsheet.

Charts

Excel supports charts, graphs, or histograms generated from specified groups of cells. The generated graphic component can either be embedded within the current sheet, or added as a separate object. These displays are dynamically updated if the content of cells change. For example, suppose that the important design requirements are displayed visually; then, in response to a user's change in trial values for parameters, the curves describing the design change shape, and their points of intersection shift, assisting the selection of the best design.

Data storage and communication

Number of rows and columns

Versions of Excel up to 7.0 had a limitation in the size of their data sets of 16K (2^{14} = 16384) rows. Versions 8.0 through 11.0 could handle 64K (2^{16} = 65536) rows and 256 columns (2^8 as label 'IV'). Version 12.0 onwards, including the current Version 16.x, can handle over 1M (2^{20} = 1048576) rows, and 16384 (2^{14} as label 'XFD') columns.

File formats

Microsoft Excel up until 2007 version used a proprietary binary file format called Excel Binary File Format (.XLS) as its primary format. Excel 2007 uses Office Open XML as its primary file format, an XML-based format that followed after a previous XML-based format called "XML Spreadsheet" ("XMLSS"), first introduced in Excel 2002.

In addition, most versions of Microsoft Excel can read CSV, DBF, SYLK, DIF, and other legacy formats. Support for some older file formats was removed in Excel 2007. The file formats were mainly from DOS-based programs.

Binary

OpenOffice.org has created documentation of the Excel format. Since then Microsoft made the Excel binary format specification available to freely download.

Export and migration of spreadsheets

Programmers have produced APIs to open Excel spreadsheets in a variety of applications and environments other than Microsoft Excel. These include opening Excel documents on the web using either ActiveX controls, or plugins like the Adobe Flash Player. The Apache POI open source project provides Java libraries for reading and writing Excel spreadsheet files. Excel Package is another open-source project that provides server-side generation of Microsoft Excel 2007 spreadsheets. PHPExcel is a PHP library that converts Excel5, Excel 2003, and Excel 2007 formats into objects for reading and writing within a web application. Excel Services is a current .NET developer tool that can enhance Excel's capabilities. Excel spreadsheets can be accessed from Python with xlrd and openpyxl.

CSV File

A comma-separated values (CSV) file is a delimited text file that uses a comma to separate values. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format. A CSV file typically stores tabular data (numbers and text) in plain text, in which case each line will have the same number of fields. These files serve a few different business purposes. They help companies export a high volume of data to a more concentrated database.

The rules should be followed to format CSV file

- Each record (row of data) is to be located on a separate line, delimited by a line break.
- The last record in the file may or may not have an ending line break.
- There may be an optional header line appearing as the first line of the file with the same format as normal record lines.
- It should contain the same number of fields as the records in the rest of the file.
- The header contains names corresponding to the fields in the file.
- In the header and each record, there may be one or more fields, separated by commas.
- Each line should contain the same number of fields throughout the file. Spaces are considered part of a field and should not be ignored.
- The last field in the record must not be followed by a comma.
- Each field may or may not be enclosed in double quotes.

- If fields are not enclosed with double quotes, then double quotes may not appear inside the fields.
- Fields containing line breaks (CRLF), double quotes, and commas should be enclosed in double quotes.
- If double quotes are used to enclose fields, then a double quote appearing inside a field must be escaped by preceding it with another double quote.

CHAPTER 2

SYSTEM STUDY

System study contains existing and proposed system details. Existing system is useful to develop proposed system. To elicit the requirements of the system and to identify the elements, Inputs, Outputs, subsystems and the procedures, the existing system had to be examined and analysed in detail.

This increases the total productivity. The use of paper files is avoided and all the data are efficiently manipulated by the system. It also reduces the space needed to store the larger paper files and records.

2.1 EXISTING SYSTEM

In early days' sales prediction and forecasting is not done using any analytics. Sales prediction tools and models were not used to predict the sales of a product. The analysis of sales does not have any patterns to suggest the future forecasting of a product. The prediction is done manually by collecting the dataset of a product.

2.1.1 DRAWBACKS

Some of the drawbacks are:

- Manually collecting data consumes more time.
- Numerous data was collected to deal with a product for forecasting.
- It relies on historical data to predict future forecasting.

2.2 PROPOSED SYSTEM

Predictive analytical algorithms and statistical models to analyse large datasets to assess the likelihood of a set of potential outcomes. These models draw upon current, contextual, and historical data to predict the probability of future events.

As new information is made available, the system incorporates more data into the statistical model and updates its predictions accordingly. Throughout this process of machine learning (ML), the model gets “smarter” and predictions become increasingly accurate.

2.2.1 FEATURES

Some of the advantages are:

- Better alignment of sales teams.
- Increased efficiency and productivity of the sales cycle.
- More accurate sales forecasts and predictions of future revenue.

CHAPTER 3
SYSTEM DESIGN

The degree of interest in each concept has varied over the year, each has stood the test of time. Each provides the software designer with a foundation from which more sophisticated design methods can be applied. Fundamental design concepts provide the necessary framework for “getting it right”.

During the design process the software requirements model is transformed into design models that describe the details of the data structures, system architecture, interface, and components. Each design product is reviewed for quality before moving to the next phase of software development.

3.1 INPUT DESIGN

The design of input focus on controlling the amount of dataset as input required, avoiding delay and keeping the process simple. The input is designed in such a way to provide security. Input design will consider the following steps:

- The dataset should be given as input.
- The dataset should be arranged.
- Methods for preparing input validations.

3.2 OUTPUT DESIGN

A quality output is one, which meets the requirement of the user and presents the information clearly. In output design, it is determined how the information is to be displayed for immediate need.

Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that the user will find the system can be used easily and effectively.

3.3 DATABASE DESIGN

This phase contains the attributes of the dataset which are maintained in the database table. The dataset collection can be of two types namely train dataset and test dataset.

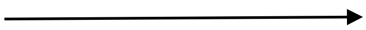
3.4 DATAFLOW DIAGRAM

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation. Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

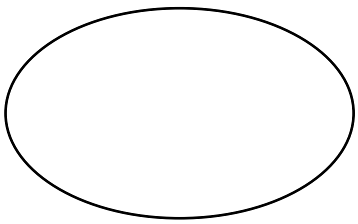
DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. The objective of a DFD is to show the scope and boundaries of a system. The DFD is also called as a data flow graph or bubble chart. It can be manual, automated, or a combination of both. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Design Notation

- **Source or Destination**



- **Dataflow**



- **Process**

CHAPTER 4

SYSTEM DEVELOPMENT

4.1 DESCRIPTION OF MODULES

- DATASET COLLECTION
- HYPOTHESIS DEFINITION
- DATA EXPLORATION
- DATA CLEANING
- DATA MODELLING
- FEATURE ENGINEERING

DATASET COLLECTION

A data set is a collection of data. Departmental store data has been used as the dataset for the proposed work. Sales data has Item Identifier, Item Fat, Item Visibility, Item Type, Outlet Type, Item MRP, Outlet Identifier, Item Weight, Outlet Size, Outlet Establishment Year, Outlet Location Type, and Item Outlet Sales.

HYPOTHESIS DEFINITION

This is a very important step to analyse any problem. The first and foremost step is to understand the problem statement. The idea is to find out the factors of a product that creates an impact on the sales of a product. A null hypothesis is a type of hypothesis used in statistics that proposes that no statistical significance exists in a set of given observations. An alternative hypothesis is one that states there is a statistically significant relationship between two variables.

DATA EXPLORATION

Data exploration is an informative search used by data consumers to form true analysis from the information gathered. Data exploration is used to analyse the data and information from the data to form true analysis. After having a look at the dataset, certain information about the data was explored. Here the dataset is not unique while collecting the dataset. In this module, the uniqueness of the dataset can be created.

DATA CLEANING

In data cleaning module, is used to detect and correct the inaccurate dataset. It is used to remove the duplication of attributes. Data cleaning is used to correct the dirty data which contains incomplete or outdated data, and the improper parsing of record fields from disparate systems. It plays a significant part in building a model.

DATA MODELLING

In data modelling module, the machine learning algorithms were used to predict the sales. Linear regression and K-means algorithm were used to predict the sales. The user provides the ML algorithm with a dataset that includes desired inputs and outputs, and the algorithm finds a method to determine how to arrive at those results.

Linear regression algorithm is a supervised learning algorithm. It implements a statistical model when relationships between the independent variables and the dependent variable are almost linear, shows optimal results. This algorithm is used to show the sales prediction with increased accuracy rate.

K-means algorithm is an unsupervised learning algorithm. It deals with the correlations and relationships by analysing available data. This algorithm clusters the data and predict the value of the dataset point. The train dataset is taken and are clustered using the algorithm. The visualization of the clusters is plotted in the graph.

FEATURE ENGINEERING

In the feature engineering module, the process of using the import data into machine learning algorithms to predict the accurate sales. A feature is an attribute or property shared by all the independent products on which the prediction is to be done. Any attribute could be a feature, it is useful to the model.

CHAPTER 5

SYSTEM ANALYSIS

5.1 FEASIBILITY STUDY

A feasibility analysis is used to determine the viability of an idea, such as ensuring a project is legally and technically feasible as well as economically justifiable. Feasibility study lets the developer to foresee the project and the usefulness of the system proposal as per its workability. It impacts the organization, ability to meet the user needs and effective use of resource. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development.

Three key consideration involved in the feasibility analysis are,

- TECHNICAL FEASIBILITY
- OPERATIONAL FEASIBILITY
- ECONOMIC FEASIBILITY

5.1.1 TECHNICAL FEASIBILITY

This phase focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the ideas can be converted into working system model. Technical feasibility also involves the evaluation of the hardware, software, and other technical requirements of the proposed system.

5.1.2 OPERATIONAL FEASIBILITY

This phase involves undertaking a study to analyse and determine how well the organization's needs can be met by completing the project. Operational feasibility study also examines how a project plan satisfies the requirements that are needed for the phase of system development.

5.1.3 ECONOMIC FEASIBILITY

This phase typically involves a cost benefits analysis of the project and help the organization to determine the viability, cost-benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility. It helps the decision-makers to determine the positive economic benefits of the organization that the proposed project will provide.

CHAPTER 6

SYSTEM TESTING

System testing is the stage of implementation that is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. System testing makes logical assumption that if all the parts of the system are correct, then the goal will be successfully achieved. System testing involves user training system testing and successful running of the developed proposed system. The user tests the developed system and changes are made per their needs. The testing phase involves the testing of developed system using various kinds of data. While testing, errors are noted and the corrections are made. The corrections are also noted for the future use.

UNIT TESTING:

Unit testing focuses verification effort on the smallest unit of software design, software component or module. Using the component level design description as a control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and the errors those uncover is limited by the constrained scope established for unit testing. The unit test focuses on the internal processing logic and data structures within the boundaries of a component. This is normally considered as an adjunct to the coding step. The design of unit tests can be performed before coding begins.

BLACK BOX TESTING

Black box testing also called behavioural testing, focuses on the functional requirement of the software. This testing enables to derive set of input conditions of all functional requirements for a program. This technique focuses on the information domain of the software, deriving test cases by partitioning the input and output of a program.

WHITE BOX TESTING

White box testing also called as glass box testing, is a test case design that uses the control structures described as part of component level design to derive test cases. This test case is derived to ensure all statements in the program have been executed at least once during the testing and that all logical conditions have been exercised.

INTEGRATION TESTING

Integration testing is a systematic technique for constructing the software architecture to conduct errors associated with interfacing. Top-down integration testing is an incremental approach to construction of the software architecture. Modules are integrated by moving downward through the control hierarchy, beginning with the main control module. Bottom-up integration testing begins the construction and testing with atomic modules. Because components are integrated from the bottom up, processing required for components subordinate to a given level is always available.

VALIDATION TESTING

Validation testing begins at the culmination of integration testing, when individual components have been exercised, the software is completely assembled as a package. The testing focuses on user visible actions and user recognizable output from the system. The testing has been conducted on possible condition such as the function characteristic conforms the specification and a deviation or error is uncovered. The alpha test and beta test is conducted at the developer site by end-users.

CHAPTER 7

SYSTEM IMPLEMENTATION

Implementation is the most crucial stage in achieving a successful system and giving the user's confidence that the new system is workable and effective. Implementation of a modified application to replace an existing one. This type of conversation is relatively easy to handle, provide there are no major changes in the system.

Each program is tested individually at the time of development using the data and has verified that this program linked together in the way specified in the programs specification, the computer system and its environment is tested to the satisfaction of the user. The system that has been developed is accepted and proved to be satisfactory for the user. A simple operating procedure is included so that the user can understand the different functions clearly and quickly.

In early days' the sales value can be predicted manually by the user. The user can analyse the sales from the historical data and records. Here more paper work needed to be done by collecting the data from the historical record. The user can manually envision the sales which can be able to reach his target. And at the same time the user can able to get nostalgia result.

The processed data is used for predictive modelling so that appropriate results can be generated from it. This predictive modelling is done using a technique called Machine Learning. It is defined as a "computer's ability to learn without being explicitly programmed". Machine learning uses programmed algorithms that receive and analyse input data to predict output values within an acceptable range.

Machine learning algorithms are programs (math and logic) that adjust themselves to perform better as they are exposed to more data. The "learning" part of machine learning means that those programs change how they process data over time, much as humans change how they process data by learning. So, a machine-learning algorithm is a program with a specific way to adjusting its own parameters, given feedback on its previous performance making predictions about a dataset.

Machine Learning algorithm such as Linear regression and K-means clustering is been used to predict the sales of the departmental store and it is implemented in this project. The use of algorithm enables to increase the accuracy of the sales. The accuracy of the sales can be reached up to the value and it is plotted in the graph format.

Linear regression may be defined as the statistical model that analyses the linear relationship between a dependent variable with given set of independent variables.

The relationship can be represented by,

$$Y=mX+b$$

Y is the dependent variable, a sale which is to be predicted.

X is the independent variable, the dataset which is used to make predictions.

m is the slope of the regression line which represents the effect X that has on Y.

b is the line which crosses the y axis.

K-means clustering algorithm computes the centroids and iterates until it finds optimal centroid. It assumes that the number of clusters is already known. The number of clusters identified from data by algorithm is represented by ‘K’ in K-means. It groups the similar data points into a cluster.

The following steps to be followed:

- Select the number of clusters which is to be identified.
- Randomly select the distinct data points and assign each data point to the cluster.
- Measures the distance between the first data point and the selected cluster.
- Then the first data point is added to the nearest cluster.
- Calculate the mean value, including new point of the first cluster.
- Repeat them until to get the optimal clustering the data point.

K-means iterates repeatedly and until the data points within each cluster stops changing. Select the best variance out of it.

CHAPTER 8

SYSTEM MAINTENANCE

The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is to make adaptable and some changes in the system environment. There may be social, technical and other environmental changes, which affect a system, that is implemented. Software product enhancements may involve providing new functional capabilities, improving user displays and mode of interaction, upgrading the performance of the characteristics of the system.

Maintenance phase identifies if there are any changes required in the current system. If the changes are identified, then an analysis is made to identify if the changes are really required. Cost benefit analysis is a way to find out if the change is essential. System maintenance conforms the system to its original requirements and the purpose is to preserve the value of software over the time. The value can be enhanced by expanding the customer base, meeting additional requirements, becoming easier to use, more efficient and employing newer technology.

CHAPTER 9

CONCLUSION

Every company desires to know the demand of the customer in any season beforehand to avoid the shortage of products. As time passes by, the demand of the store to be more accurate about the predictions will increase exponentially. So, huge research is going on in this sector to make accurate predictions of sales. Better predictions are directly proportional to the profit made by the departmental store. The purpose of measuring accuracy was to validate our prediction with the actual result. In this project, an effort has been made to predict sales of the product from an outlet accurately by using a two-level statistical model that reduces the mean absolute error value. The two-level statistical model performed than the other single model predictive techniques and contributed better predictions to the departmental store dataset.

CHAPTER 10

FUTURE SCOPE AND ENHANCEMENT

Further expansion of the system also can be done in future if needed. The application can be enhanced in the future with the needs of the departmental store. The database and the information can be updated to the latest forthcoming versions. Thus, the system can be altered in accordance with the future requirements and advancements. System performance evaluation must be monitored not only to determine whether they perform as plan but also to determine if they should have to meet changes in the information needed for the departmental store.

BIBLIOGRAPHY

BOOK REFERENCES

- Greg Moss, “Working with Odoo”, 2015.
- Greg Moss, “Working with Odoo 11: Configure, manage, and customize your Odoo system”, 3rd Edition, 2018.
- Daniel Reis, “Odoo 12 Development Essentials: Fast-track your Odoo development skills to build powerful business applications”, 4th Edition ,2018.
- James Yan and Yuxing Yan, “Hands-On Data Science with Anaconda: Utilize the Right Mix of Tools to Create High-performance Data Science Applications”, 2018.
- Curtis Miller, “Hands-On Data Analysis with NumPy and pandas: Implement Python packages from data manipulation to processing”, 2018.

WEBSITES

FOR PYTHON

- https://www.tutorialspoint.com
- https://www.codecademy.com
- https://www.w3schools.com
- https://www.anaconda.coms

FOR CSV

- https://en.m.wikipedia.org
- https://www.computerhope.com
- https://www.bigcommerce.com

APPENDICES

A. DATA FLOW DIAGRAM

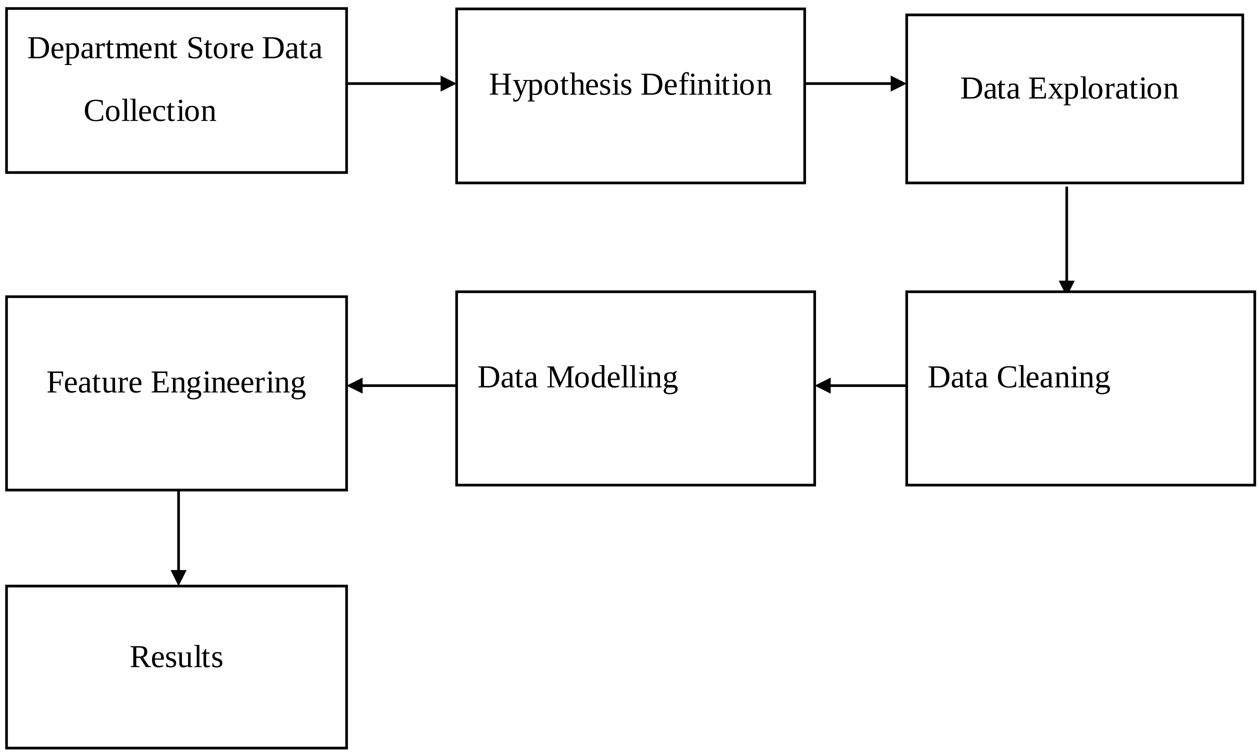


Figure A.1 Flow diagram of Proposed system

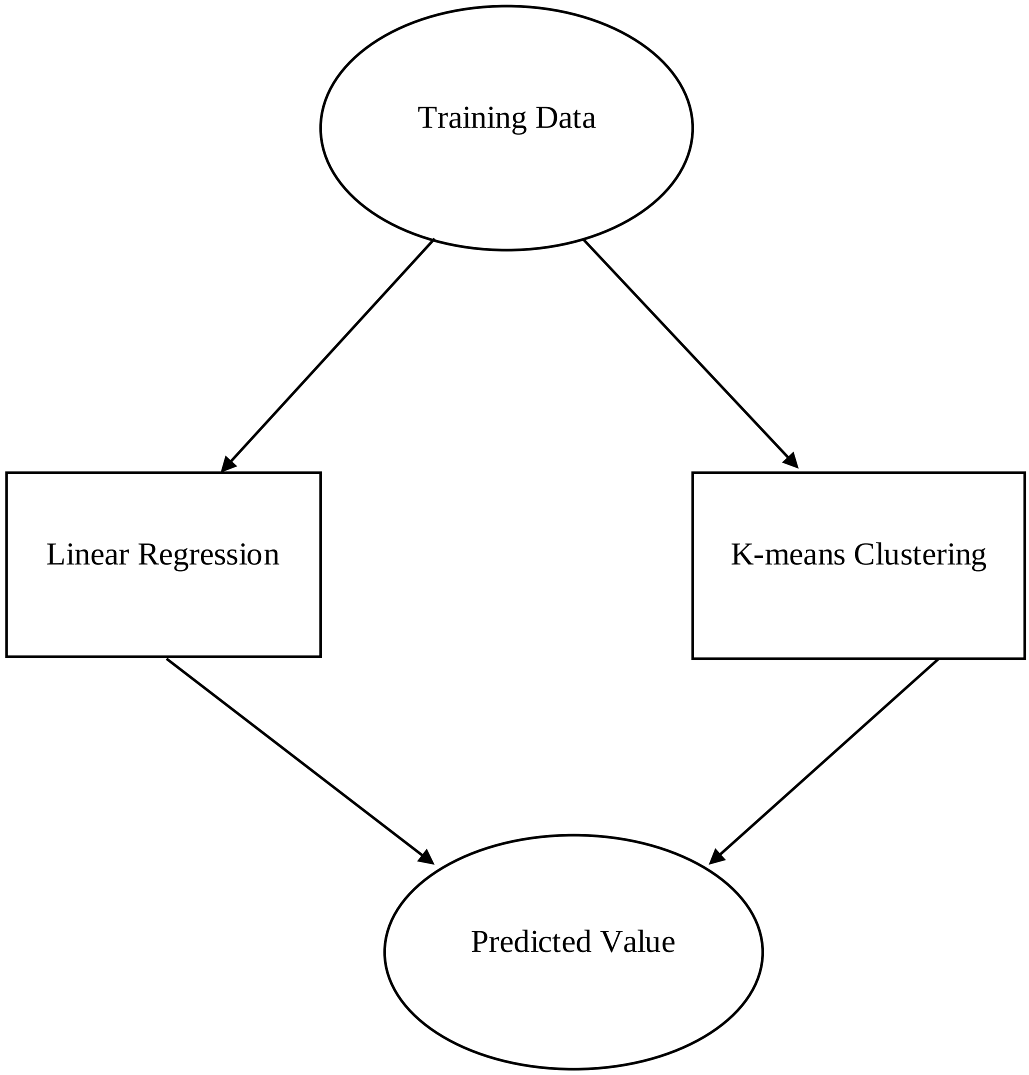


Figure A.2 Two level statistical model

B. TABLE DESIGN

TABLE NAME **:** Train data

COLUMN NAME	**DATATYPE**
Item _ Identifier	Varchar (10)
Item _ Weight	Numeric (10)
Item _ Fat _ Content	String (10)
Item _ Visibility	Numeric (15)
Item _ Type	String (20)
Item _ MRP	Numeric (10)
Outlet _ Identifier	Varchar (10)
Outlet _ Establishment	Numeric (6)
Outlet _ Size	String (10)
Outlet _ Location	String (12)
Outlet _ Type	String (20)
Item _ Outlet _ Sales	Numeric (10)

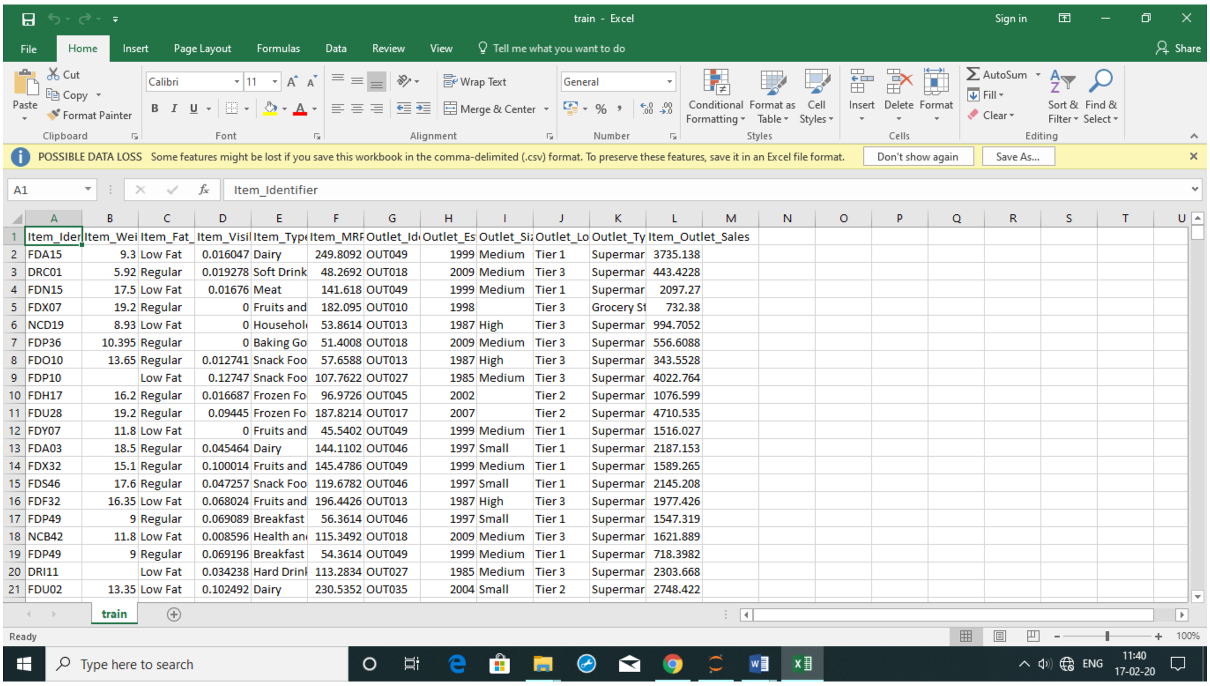
Fig B.1 Train data set

**TABLE NAME : ** Test data

COLUMN NAME	**DATATYPE**
Item _ Identifier	Varchar (10)
Item _ Weight	Numeric (10)
Item _ Fat _ Content	String (10)
Item _ Visibility	Numeric (15)
Item _ Type	String (20)
Item _ MRP	Numeric (10)
Outlet _ Identifier	Varchar (10)
Outlet _ Establishment	Numeric (6)
Outlet _ Size	String (10)
Outlet _ Location	String (12)
Outlet _ Type	String (20)

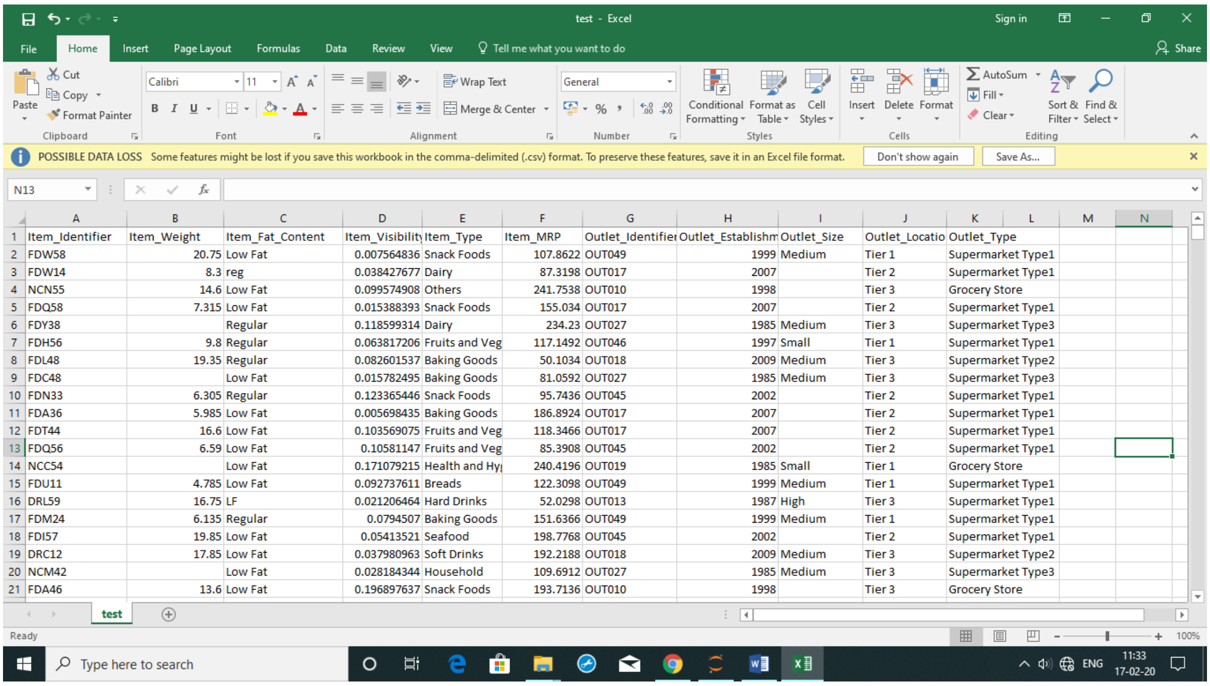
Fig B.2 Test data set

C. FORM DESIGN



	A	B	C	D	E	F	G	H	I	J	K	L
1	Item_Ider	Item_Wei	Item_Fat_	Item_Visil	Item_Type	Item_MRF	Outlet_Id	Outlet_Es	Outlet_Si	Outlet_Lo	Outlet_Ty	Item_Outlet_Sales
2	FDA15	9.3	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Medium	Tier 1	Supermar	3735.138
3	DRC01	5.92	Regular	0.019278	Soft Drink	48.2692	OUT018	2009	Medium	Tier 3	Supermar	443.4228
4	FDN15	17.5	Low Fat	0.01676	Meat	141.618	OUT049	1999	Medium	Tier 1	Supermar	2097.27
5	FDX07	19.2	Regular	0	Fruits and	182.095	OUT010	1998		Tier 3	Grocery St	732.38
6	NCD19	8.93	Low Fat	0	Househol	53.8614	OUT013	1987	High	Tier 3	Supermar	994.7052
7	FDP36	10.395	Regular	0	Baking Go	51.4008	OUT018	2009	Medium	Tier 3	Supermar	556.6088
8	FDO10	13.65	Regular	0.012741	Snack Foo	57.6588	OUT013	1987	High	Tier 3	Supermar	343.5528
9	FDP10		Low Fat	0.12747	Snack Foo	107.7622	OUT027	1985	Medium	Tier 3	Supermar	4022.764
10	FDH17	16.2	Regular	0.016687	Frozen Fo	96.9726	OUT045	2002		Tier 2	Supermar	1076.599
11	FDU28	19.2	Regular	0.09445	Frozen Fo	187.8214	OUT017	2007		Tier 2	Supermar	4710.535
12	FDY07	11.8	Low Fat	0	Fruits and	45.5402	OUT049	1999	Medium	Tier 1	Supermar	1516.027
13	FDA03	18.5	Regular	0.045464	Dairy	144.1102	OUT046	1997	Small	Tier 1	Supermar	2187.153
14	FDX32	15.1	Regular	0.100014	Fruits and	145.4786	OUT049	1999	Medium	Tier 1	Supermar	1589.265
15	FDS46	17.6	Regular	0.047257	Snack Foo	119.6782	OUT046	1997	Small	Tier 1	Supermar	2145.208
16	FDF32	16.35	Low Fat	0.068024	Fruits and	196.4426	OUT013	1987	High	Tier 3	Supermar	1977.426
17	FDP49	9	Regular	0.069089	Breakfast	56.3614	OUT046	1997	Small	Tier 1	Supermar	1547.319
18	NCB42	11.8	Low Fat	0.008596	Health an	115.3492	OUT018	2009	Medium	Tier 3	Supermar	1621.889
19	FDP49	9	Regular	0.069196	Breakfast	54.3614	OUT049	1999	Medium	Tier 1	Supermar	718.3982
20	DRI11		Low Fat	0.034238	Hard Drinl	113.2834	OUT027	1985	Medium	Tier 3	Supermar	2303.668
21	FDU02	13.35	Low Fat	0.102492	Dairy	230.5352	OUT035	2004	Small	Tier 2	Supermar	2748.422

Fig C.1 Dataset collection of train data



Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishm	Outlet_Size	Outlet_Locatio	Outlet_Type
FDW58	20.75	Low Fat	0.007564836	Snack Foods	107.8622	OUT049	1999	Medium	Tier 1	Supermarket Type1
FDW14	8.3	reg	0.038427677	Dairy	87.3198	OUT017	2007		Tier 2	Supermarket Type1
NCN55	14.6	Low Fat	0.099574908	Others	241.7538	OUT010	1998		Tier 3	Grocery Store
FDQ58	7.315	Low Fat	0.015388393	Snack Foods	155.034	OUT017	2007		Tier 2	Supermarket Type1
FDY38		Regular	0.118599314	Dairy	234.23	OUT027	1985	Medium	Tier 3	Supermarket Type3
FDH56	9.8	Regular	0.063817206	Fruits and Veg	117.1492	OUT046	1997	Small	Tier 1	Supermarket Type1
FDL48	19.35	Regular	0.082601537	Baking Goods	50.1034	OUT018	2009	Medium	Tier 3	Supermarket Type2
FDC48		Low Fat	0.015782495	Baking Goods	81.0592	OUT027	1985	Medium	Tier 3	Supermarket Type3
FDN33	6.305	Regular	0.123365446	Snack Foods	95.7436	OUT045	2002		Tier 2	Supermarket Type1
FDA36	5.985	Low Fat	0.005698435	Baking Goods	186.8924	OUT017	2007		Tier 2	Supermarket Type1
FDT44	16.6	Low Fat	0.103569075	Fruits and Veg	118.3466	OUT017	2007		Tier 2	Supermarket Type1
FDQ56	6.59	Low Fat	0.10581147	Fruits and Veg	85.3908	OUT045	2002		Tier 2	Supermarket Type1
NCC54		Low Fat	0.171079215	Health and Hy	240.4196	OUT019	1985	Small	Tier 1	Grocery Store
FDU11	4.785	Low Fat	0.092737611	Breads	122.3098	OUT049	1999	Medium	Tier 1	Supermarket Type1
DRL59	16.75	LF	0.021206464	Hard Drinks	52.0298	OUT013	1987	High	Tier 3	Supermarket Type1
FDM24	6.135	Regular	0.0794507	Baking Goods	151.6366	OUT049	1999	Medium	Tier 1	Supermarket Type1
FDI57	19.85	Low Fat	0.05413521	Seafood	198.7768	OUT045	2002		Tier 2	Supermarket Type1
DRC12	17.85	Low Fat	0.037980963	Soft Drinks	192.2188	OUT018	2009	Medium	Tier 3	Supermarket Type2
NCM42		Low Fat	0.028184344	Household	109.6912	OUT027	1985	Medium	Tier 3	Supermarket Type3
FDA46	13.6	Low Fat	0.196897637	Snack Foods	193.7136	OUT010	1998		Tier 3	Grocery Store

Fig C.2 Dataset collection of test data

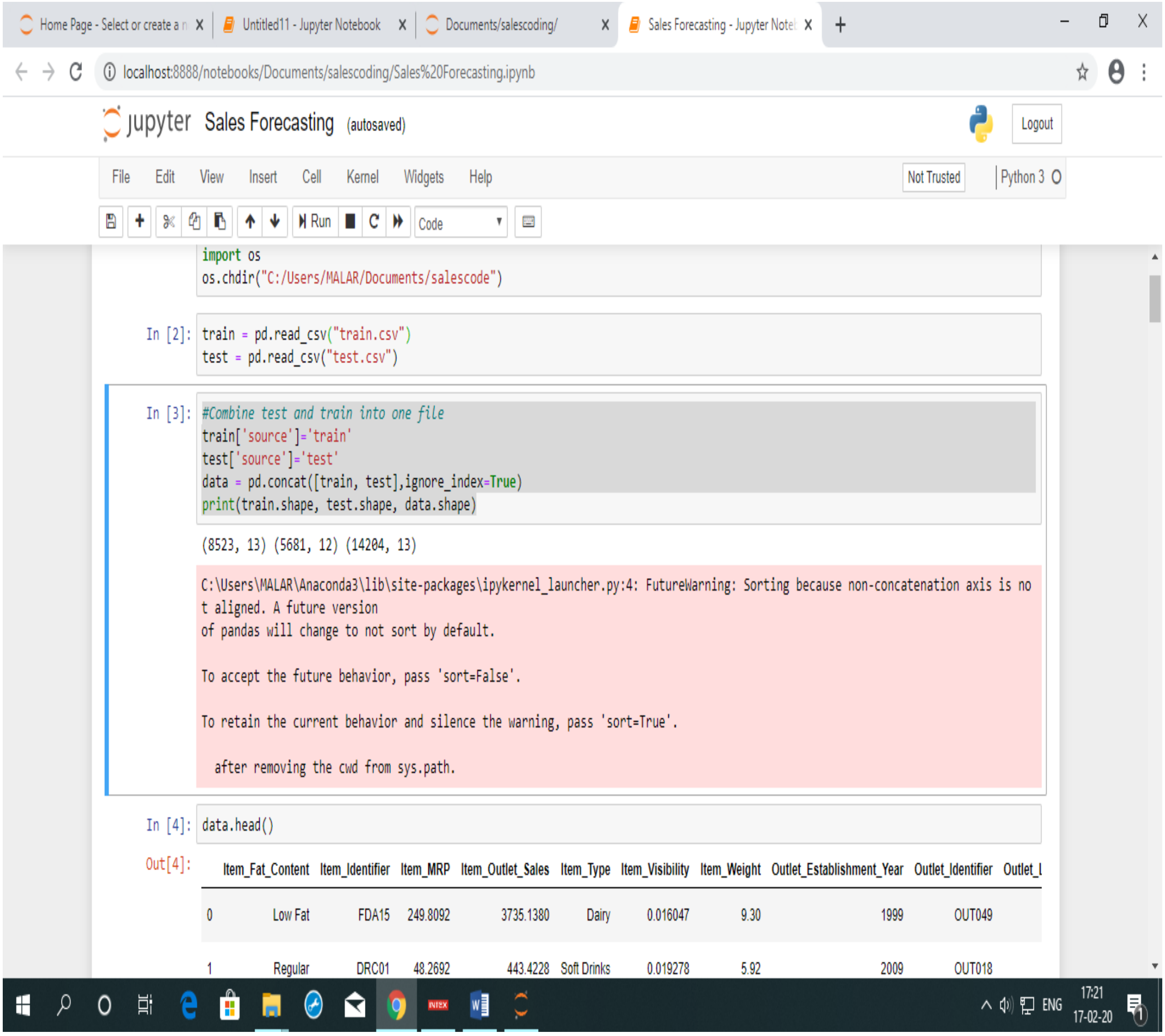
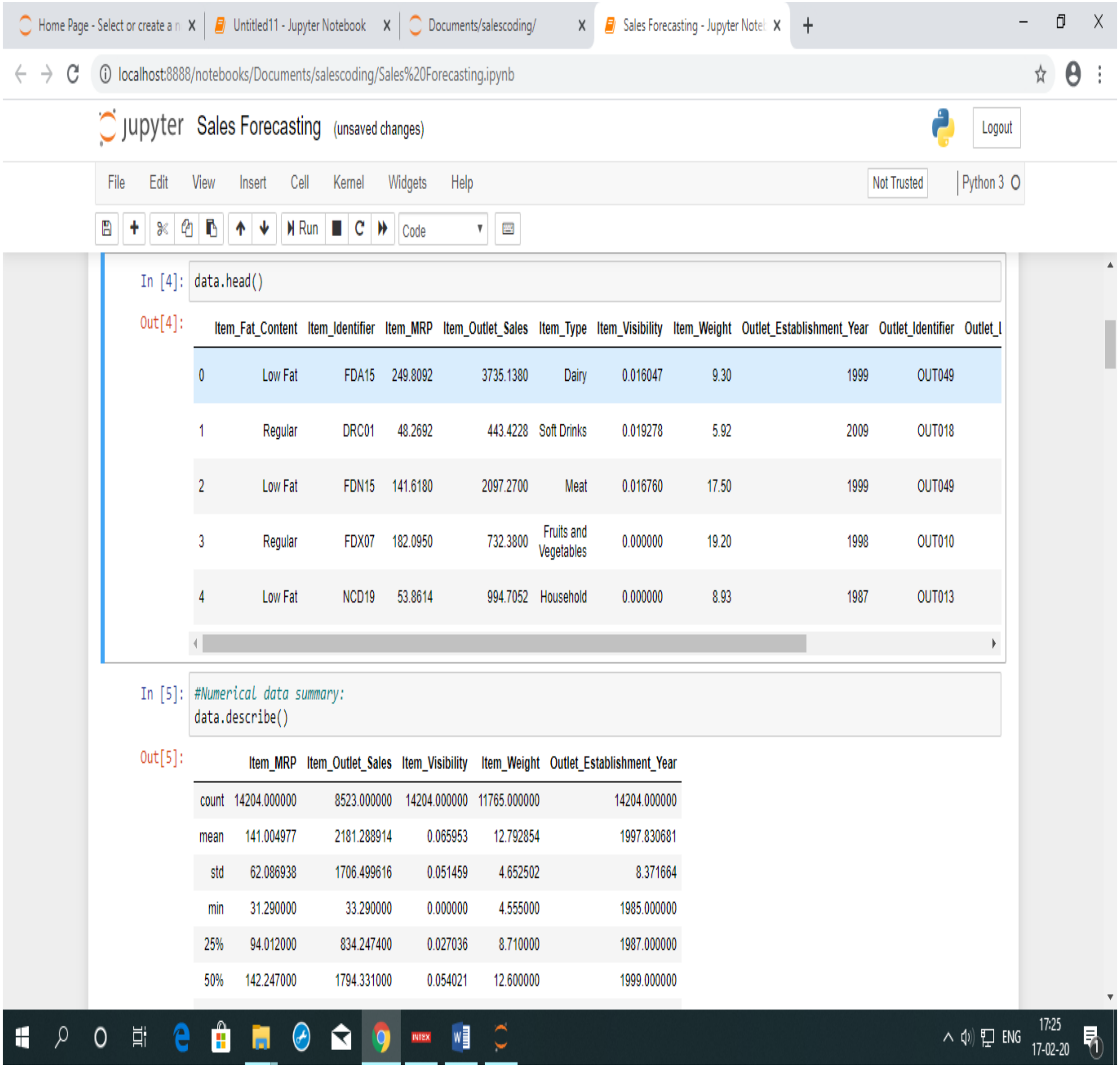


Fig C.3 Combine train and test data into one file



In [4]:
```
data.head()
```

Out[4]:

	Item_Fat_Content	Item_Identifier	Item_MRP	Item_Outlet_Sales	Item_Type	Item_Visibility	Item_Weight	Outlet_Establishment_Year	Outlet_Identifier	Outlet_I
0	Low Fat	FDA15	249.8092	3735.1380	Dairy	0.016047	9.30	1999	OUT049	
1	Regular	DRC01	48.2692	443.4228	Soft Drinks	0.019278	5.92	2009	OUT018	
2	Low Fat	FDN15	141.6180	2097.2700	Meat	0.016760	17.50	1999	OUT049	
3	Regular	FDX07	182.0950	732.3800	Fruits and Vegetables	0.000000	19.20	1998	OUT010	
4	Low Fat	NCD19	53.8614	994.7052	Household	0.000000	8.93	1987	OUT013	

In [5]:
```
#Numerical data summary:
data.describe()
```

Out[5]:

	Item_MRP	Item_Outlet_Sales	Item_Visibility	Item_Weight	Outlet_Establishment_Year
count	14204.000000	8523.000000	14204.000000	11765.000000	14204.000000
mean	141.004977	2181.288914	0.065953	12.792854	1997.830681
std	62.086938	1706.499616	0.051459	4.652502	8.371664
min	31.290000	33.290000	0.000000	4.555000	1985.000000
25%	94.012000	834.247400	0.027036	8.710000	1987.000000
50%	142.247000	1794.331000	0.054021	12.600000	1999.000000

Fig C.4 Dataset which are concatenated

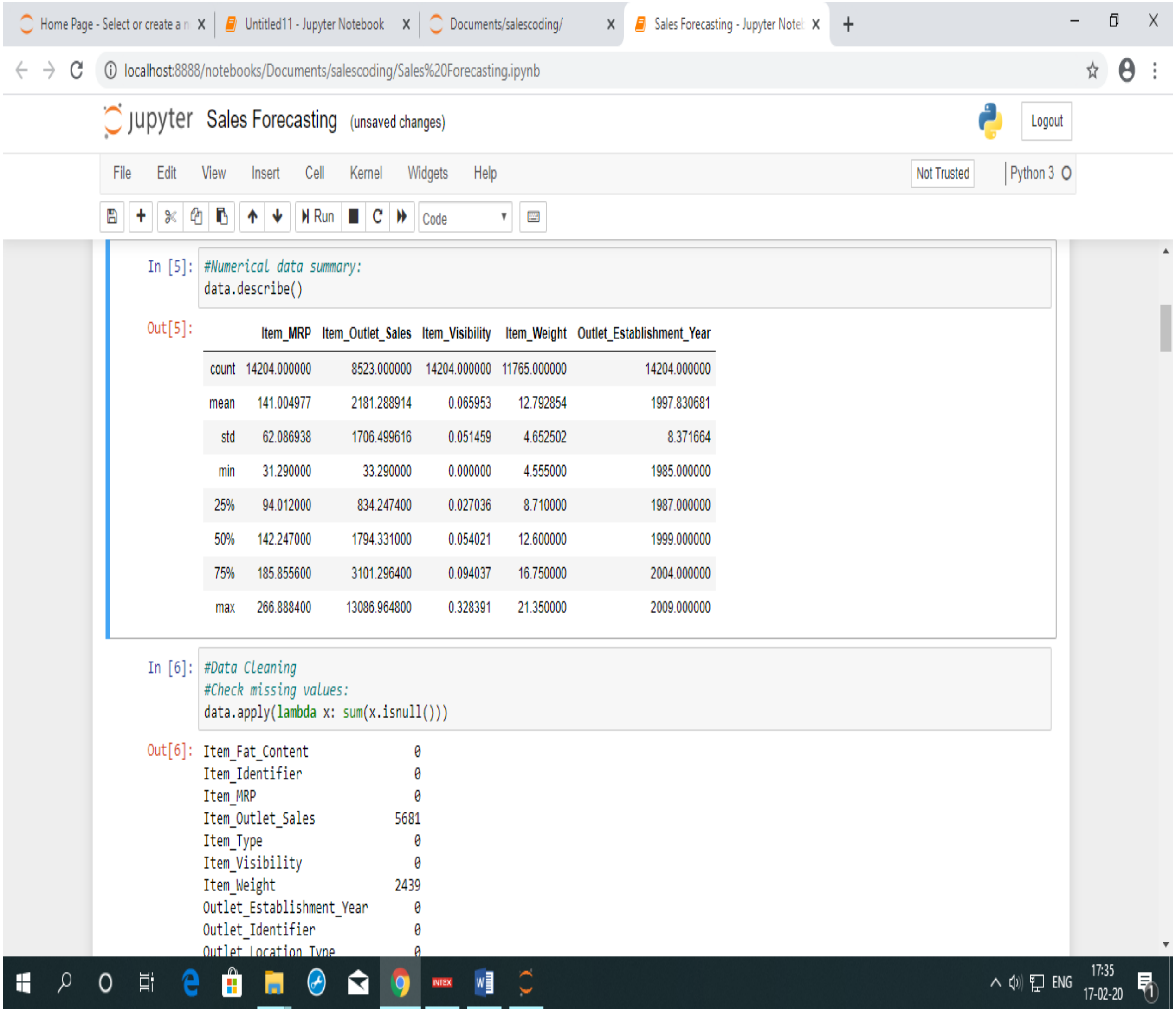


Fig C.5 Numerical data summary of the dataset

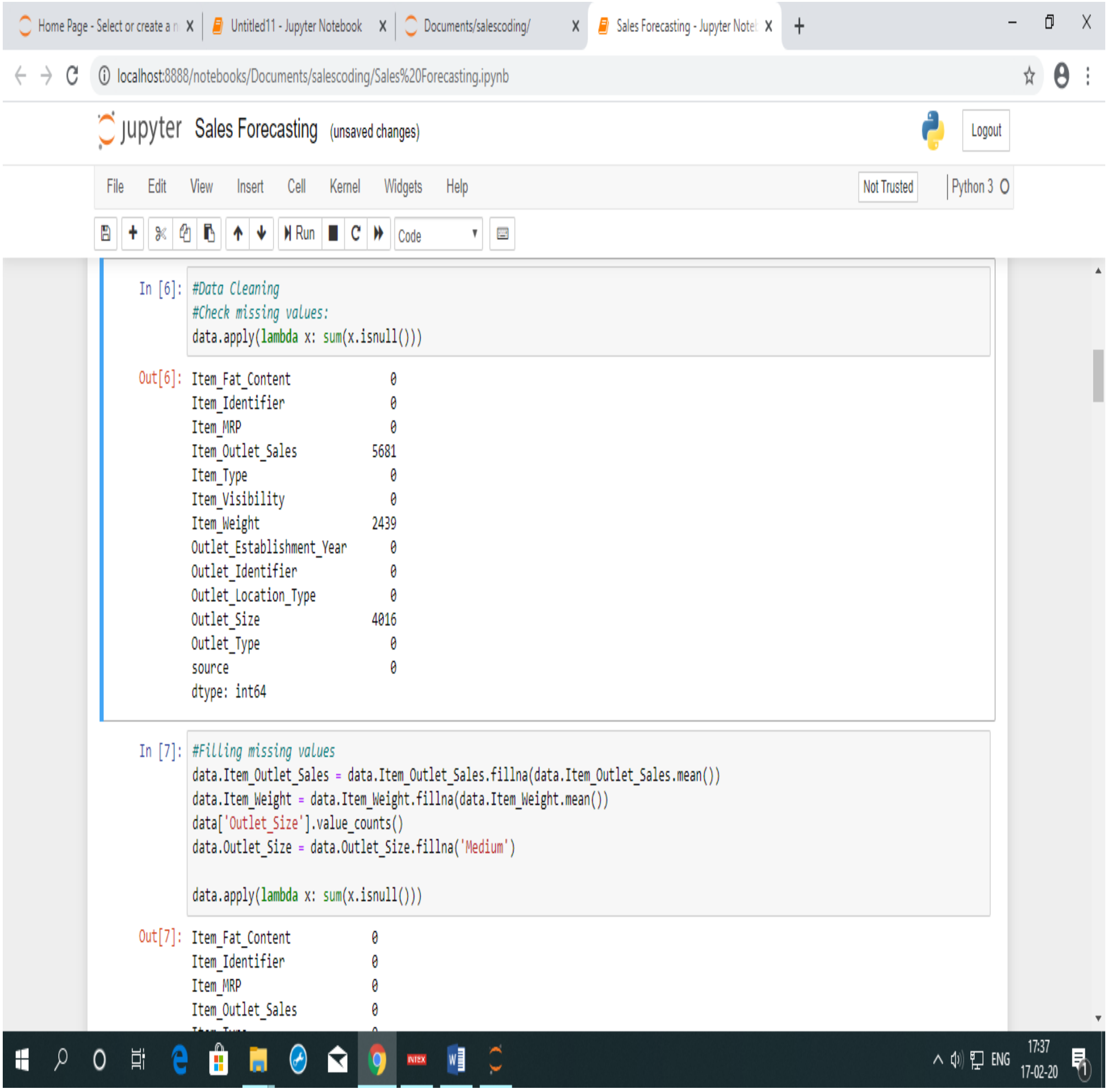


Fig C.6 Data cleaning – Checking Missing values in the dataset

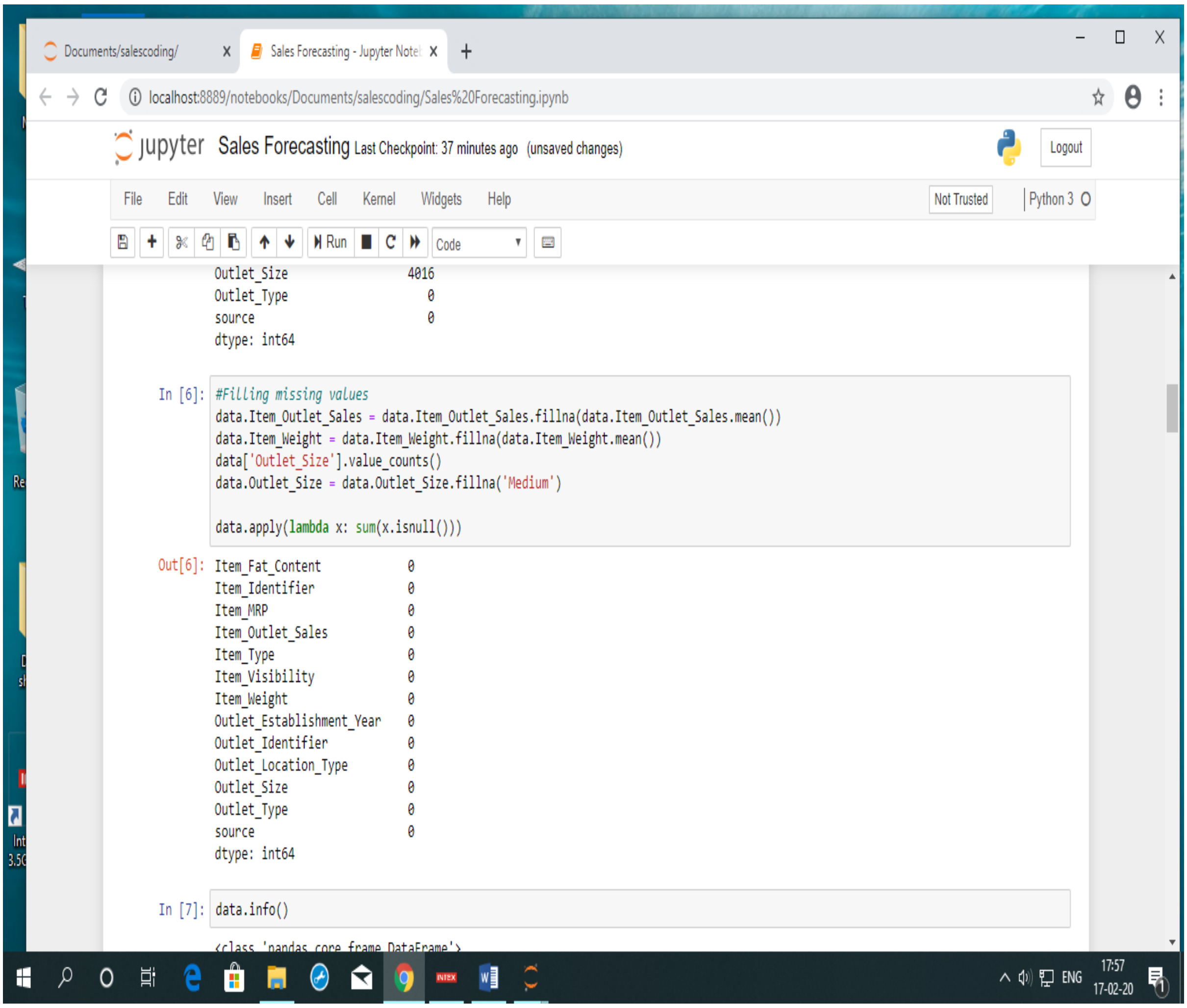


Fig C.7 Filling the missed values

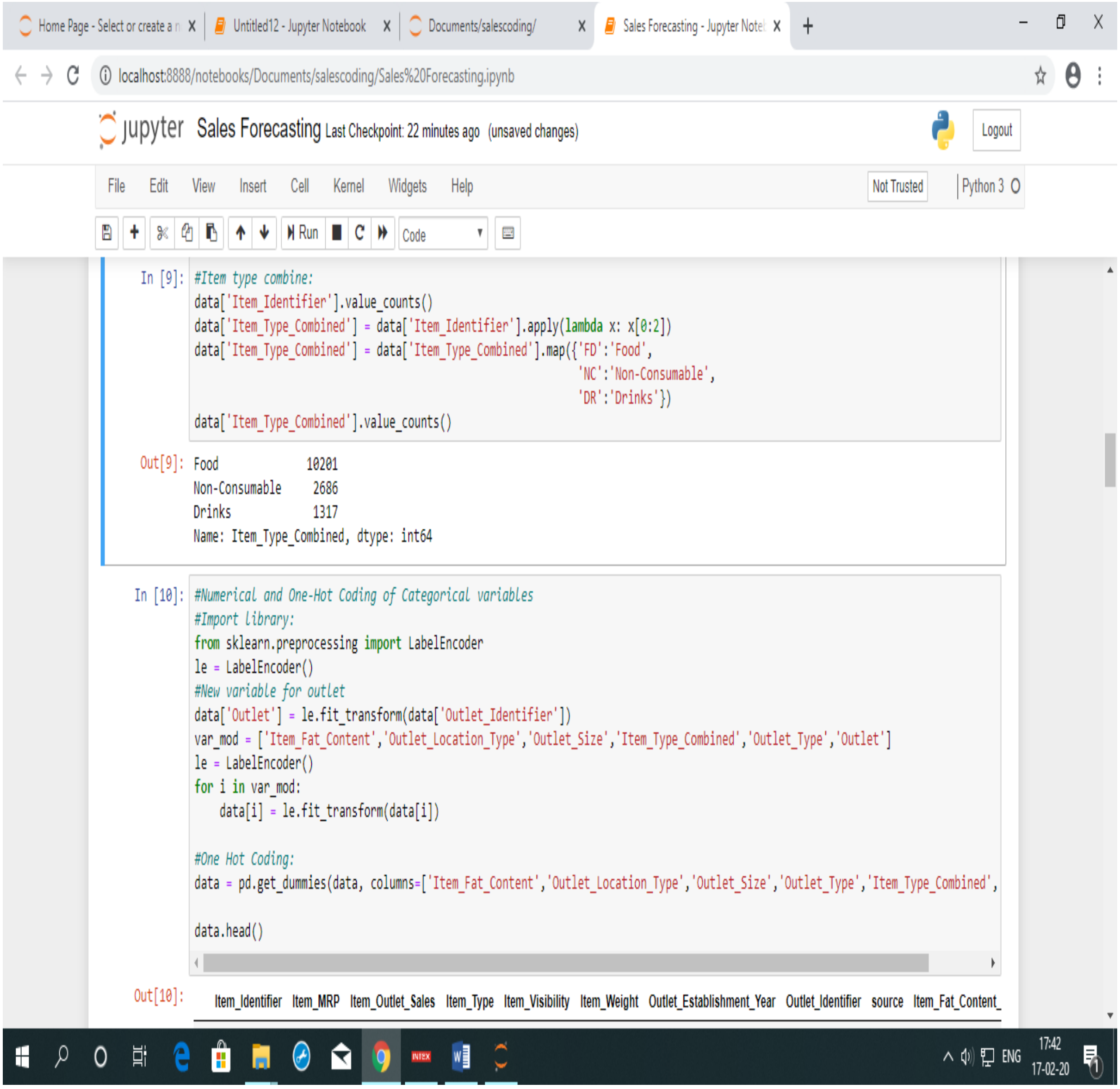


Fig C.8 Combining the item type

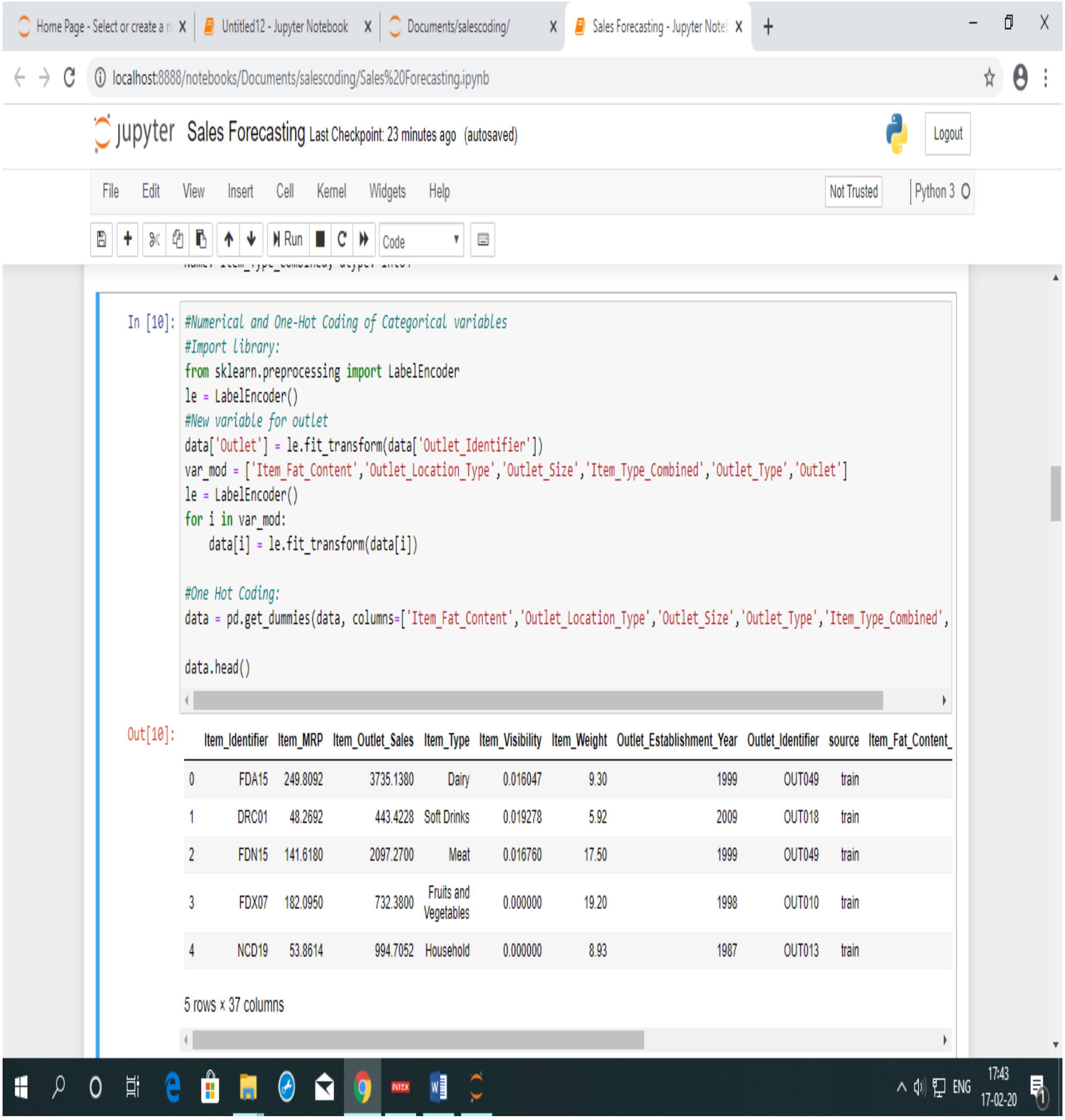


Fig C.9 Encoding the label

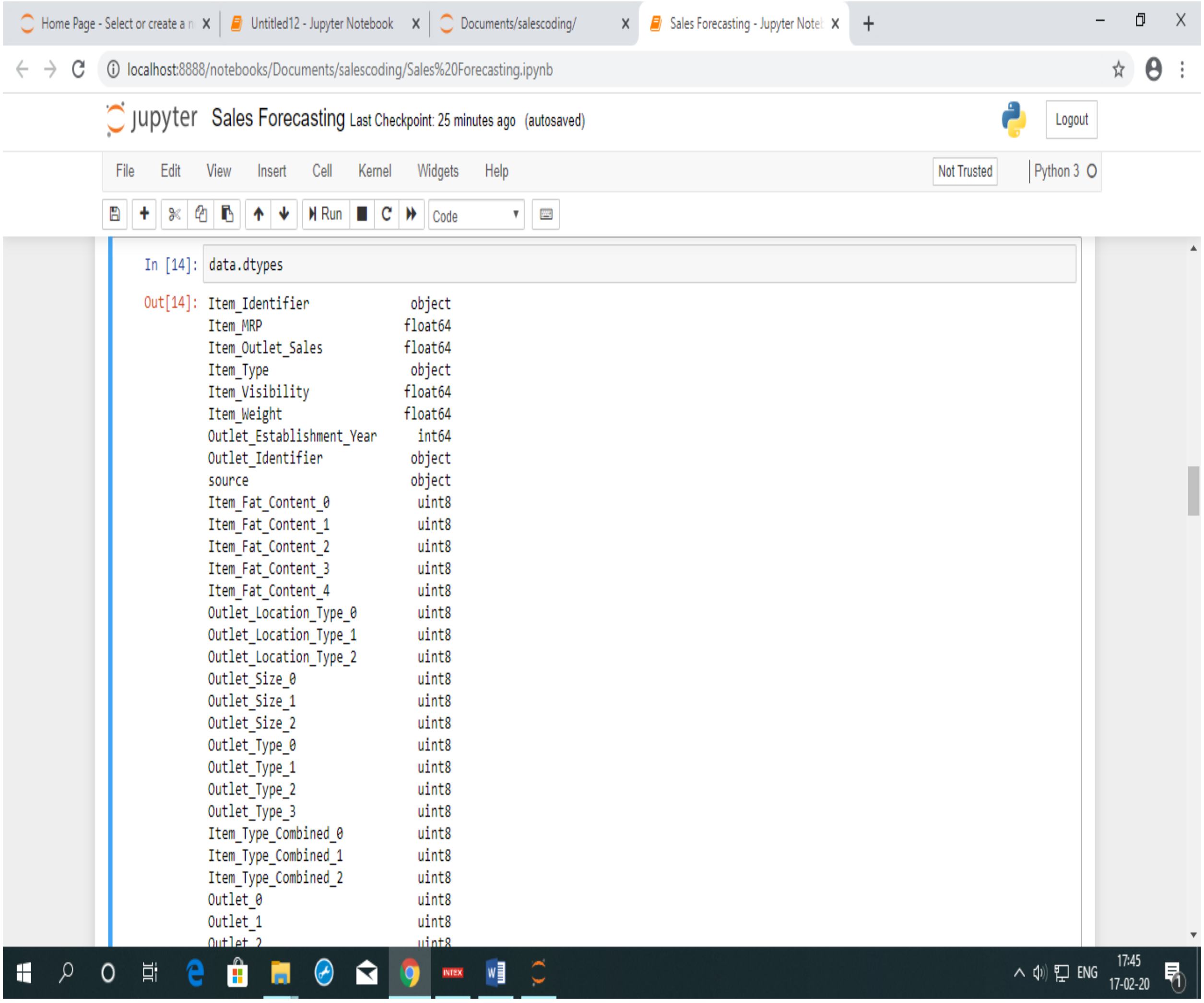


Fig C.10 After encoding the label the new datatype of the dataset

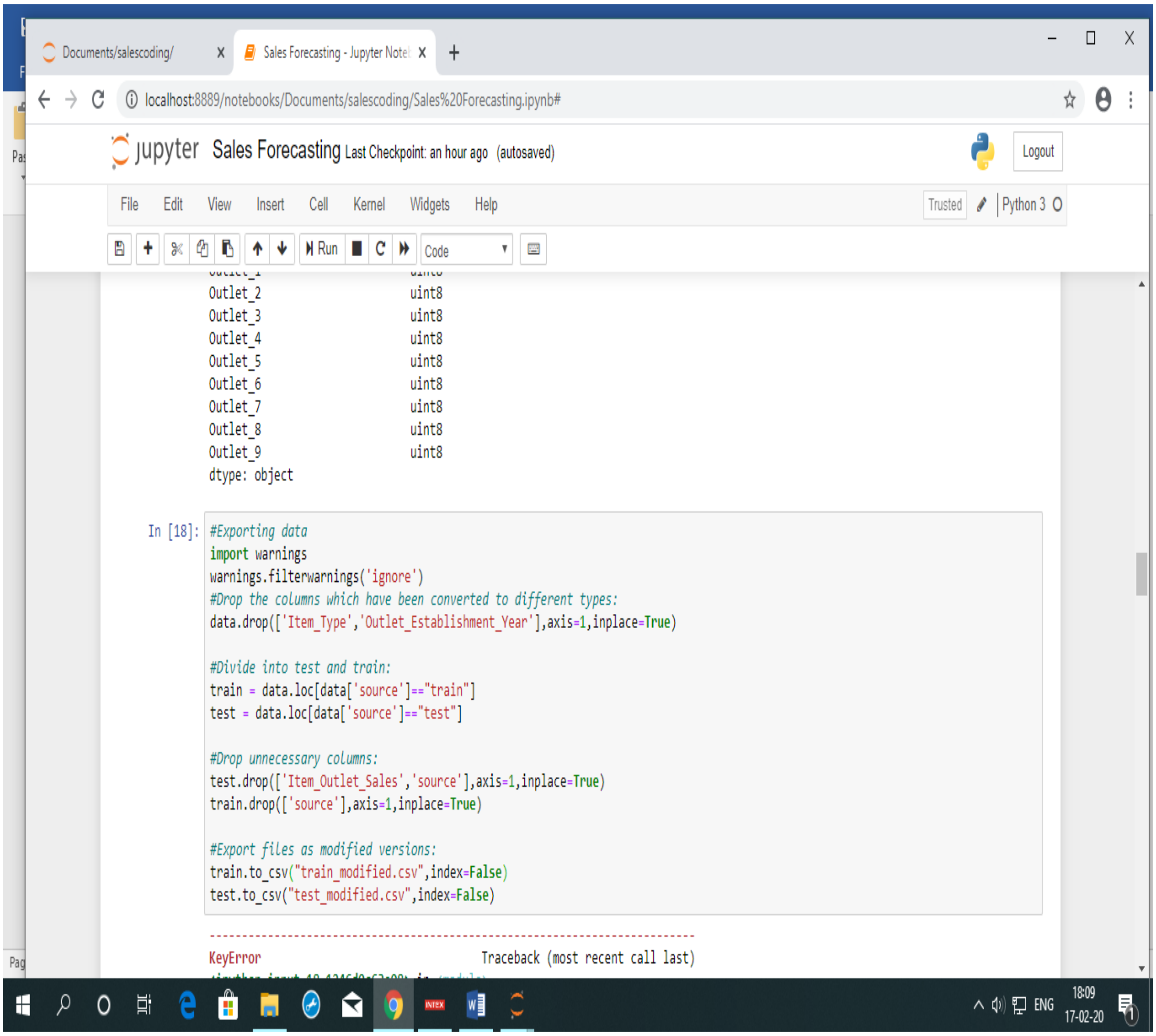
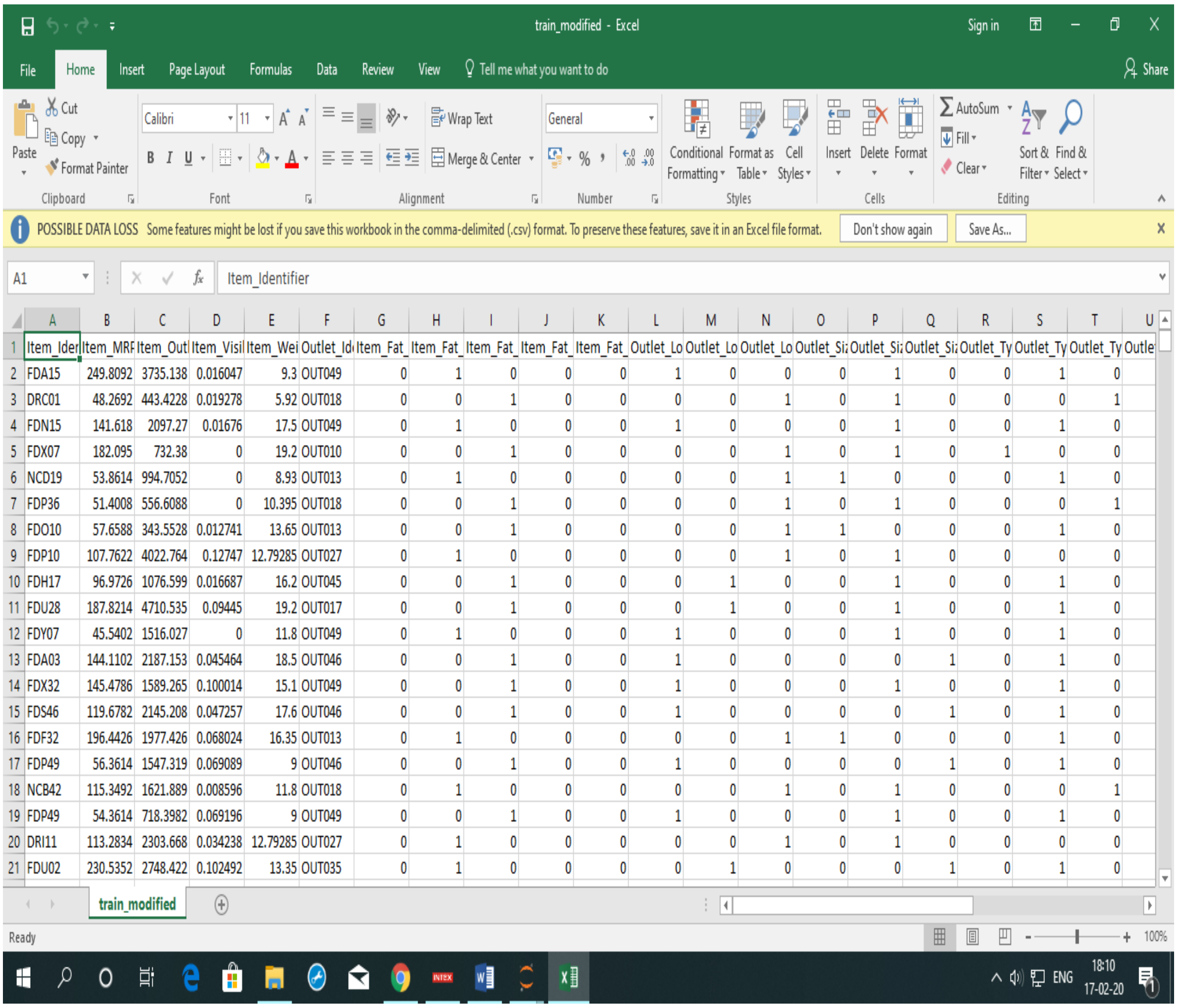
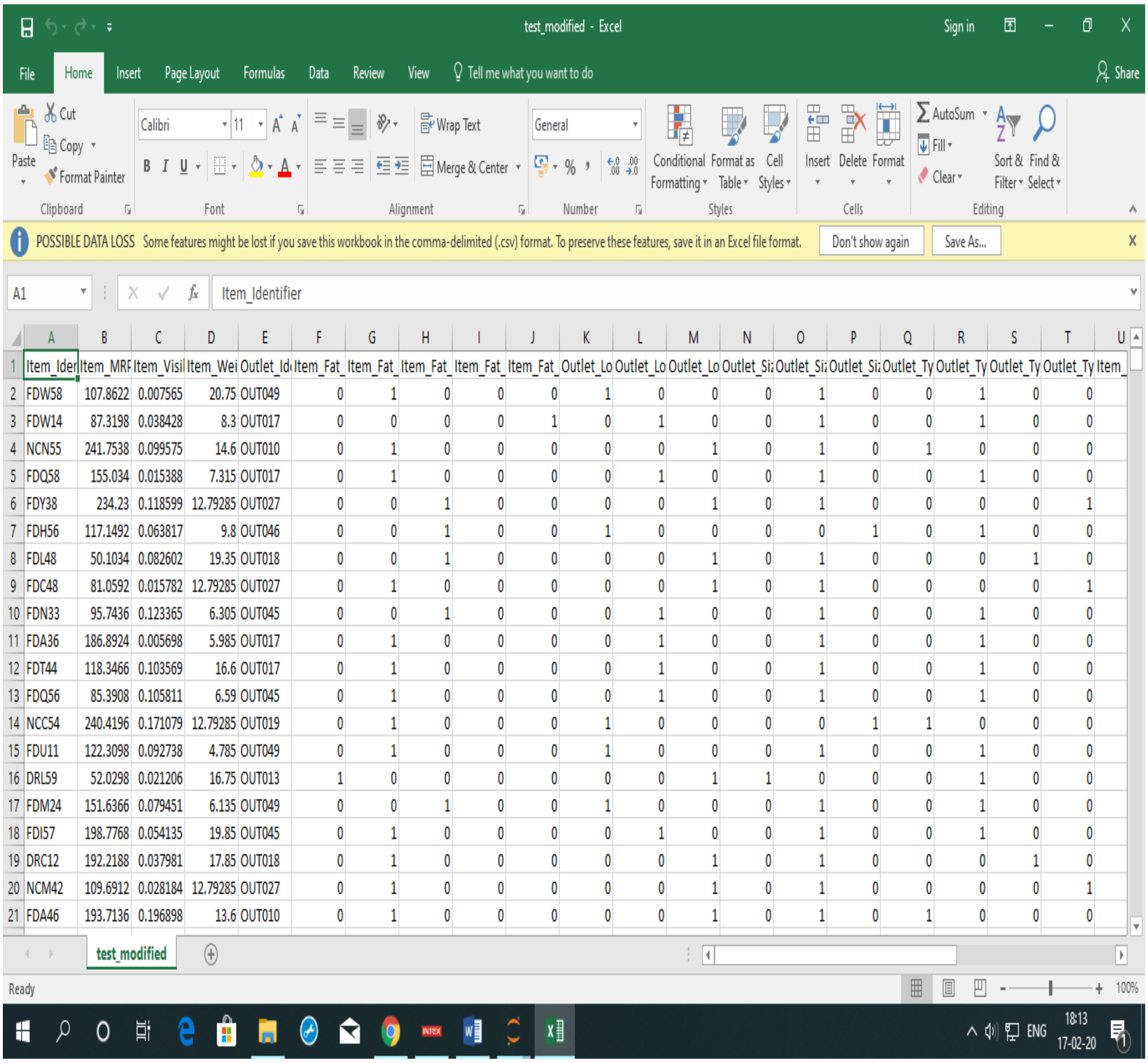


Fig C.11 Exporting the dataset



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Item_Ider	Item_MRF	Item_Outl	Item_Visil	Item_Wei	Outlet_Id	Item_Fat_	Item_Fat_	Item_Fat_	Item_Fat_	Item_Fat_	Outlet_Lo	Outlet_Lo	Outlet_Lo	Outlet_Si	Outlet_Si	Outlet_Si	Outlet_Ty	Outlet_Ty	Outlet_Ty	Outle
2	FDA15	249.8092	3735.138	0.016047	9.3	OUT049	0	1	0	0	0	1	0	0	0	1	0	0	1	0	
3	DRC01	48.2692	443.4228	0.019278	5.92	OUT018	0	0	1	0	0	0	0	1	0	1	0	0	0	1	
4	FDN15	141.618	2097.27	0.01676	17.5	OUT049	0	1	0	0	0	1	0	0	0	1	0	0	1	0	
5	FDX07	182.095	732.38	0	19.2	OUT010	0	0	1	0	0	0	0	1	0	1	0	1	0	0	
6	NCD19	53.8614	994.7052	0	8.93	OUT013	0	1	0	0	0	0	0	1	1	0	0	0	1	0	
7	FDP36	51.4008	556.6088	0	10.395	OUT018	0	0	1	0	0	0	0	1	0	1	0	0	0	1	
8	FDO10	57.6588	343.5528	0.012741	13.65	OUT013	0	0	1	0	0	0	0	1	1	0	0	0	1	0	
9	FDP10	107.7622	4022.764	0.12747	12.79285	OUT027	0	1	0	0	0	0	0	1	0	1	0	0	0	0	
10	FDH17	96.9726	1076.599	0.016687	16.2	OUT045	0	0	1	0	0	0	1	0	0	1	0	0	1	0	
11	FDU28	187.8214	4710.535	0.09445	19.2	OUT017	0	0	1	0	0	0	1	0	0	1	0	0	1	0	
12	FDY07	45.5402	1516.027	0	11.8	OUT049	0	1	0	0	0	1	0	0	0	1	0	0	1	0	
13	FDA03	144.1102	2187.153	0.045464	18.5	OUT046	0	0	1	0	0	1	0	0	0	0	1	0	1	0	
14	FDX32	145.4786	1589.265	0.100014	15.1	OUT049	0	0	1	0	0	1	0	0	0	1	0	0	1	0	
15	FDS46	119.6782	2145.208	0.047257	17.6	OUT046	0	0	1	0	0	1	0	0	0	0	1	0	1	0	
16	FDF32	196.4426	1977.426	0.068024	16.35	OUT013	0	1	0	0	0	0	0	1	1	0	0	0	1	0	
17	FDP49	56.3614	1547.319	0.069089	9	OUT046	0	0	1	0	0	1	0	0	0	0	1	0	1	0	
18	NCB42	115.3492	1621.889	0.008596	11.8	OUT018	0	1	0	0	0	0	0	1	0	1	0	0	0	1	
19	FDP49	54.3614	718.3982	0.069196	9	OUT049	0	0	1	0	0	1	0	0	0	1	0	0	1	0	
20	DRI11	113.2834	2303.668	0.034238	12.79285	OUT027	0	1	0	0	0	0	0	1	0	1	0	0	0	0	
21	FDU02	230.5352	2748.422	0.102492	13.35	OUT035	0	1	0	0	0	0	1	0	0	0	1	0	1	0	

Fig C.12 After Exporting the data, the train dataset has been modified



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Item_Ider	Item_MRF	Item_Visil	Item_Wei	Outlet_Id	Item_Fat_	Item_Fat_	Item_Fat_	Item_Fat_	Item_Fat_	Outlet_Lo	Outlet_Lo	Outlet_Lo	Outlet_Si	Outlet_Si	Outlet_Si	Outlet_Ty	Outlet_Ty	Outlet_Ty	Outlet_Ty	Item_
2	FDW58	107.8622	0.007565	20.75	OUT049	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	
3	FDW14	87.3198	0.038428	8.3	OUT017	0	0	0	0	1	0	1	0	0	1	0	0	1	0	0	
4	NCN55	241.7538	0.099575	14.6	OUT010	0	1	0	0	0	0	0	1	0	1	0	1	0	0	0	
5	FDQ58	155.034	0.015388	7.315	OUT017	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	
6	FDY38	234.23	0.118599	12.79285	OUT027	0	0	1	0	0	0	0	1	0	1	0	0	0	0	1	
7	FDH56	117.1492	0.063817	9.8	OUT046	0	0	1	0	0	1	0	0	0	0	1	0	1	0	0	
8	FDL48	50.1034	0.082602	19.35	OUT018	0	0	1	0	0	0	0	1	0	1	0	0	0	1	0	
9	FDC48	81.0592	0.015782	12.79285	OUT027	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	
10	FDN33	95.7436	0.123365	6.305	OUT045	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	
11	FDA36	186.8924	0.005698	5.985	OUT017	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	
12	FDT44	118.3466	0.103569	16.6	OUT017	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	
13	FDQ56	85.3908	0.105811	6.59	OUT045	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	
14	NCC54	240.4196	0.171079	12.79285	OUT019	0	1	0	0	0	1	0	0	0	0	1	1	0	0	0	
15	FDU11	122.3098	0.092738	4.785	OUT049	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	
16	DRL59	52.0298	0.021206	16.75	OUT013	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0	
17	FDM24	151.6366	0.079451	6.135	OUT049	0	0	1	0	0	1	0	0	0	1	0	0	1	0	0	
18	FDI57	198.7768	0.054135	19.85	OUT045	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	
19	DRC12	192.2188	0.037981	17.85	OUT018	0	1	0	0	0	0	0	1	0	1	0	0	0	1	0	
20	NCM42	109.6912	0.028184	12.79285	OUT027	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	
21	FDA46	193.7136	0.196898	13.6	OUT010	0	1	0	0	0	0	0	1	0	1	0	1	0	0	0	

Fig C.13 After Exporting the data, the test dataset has been modified

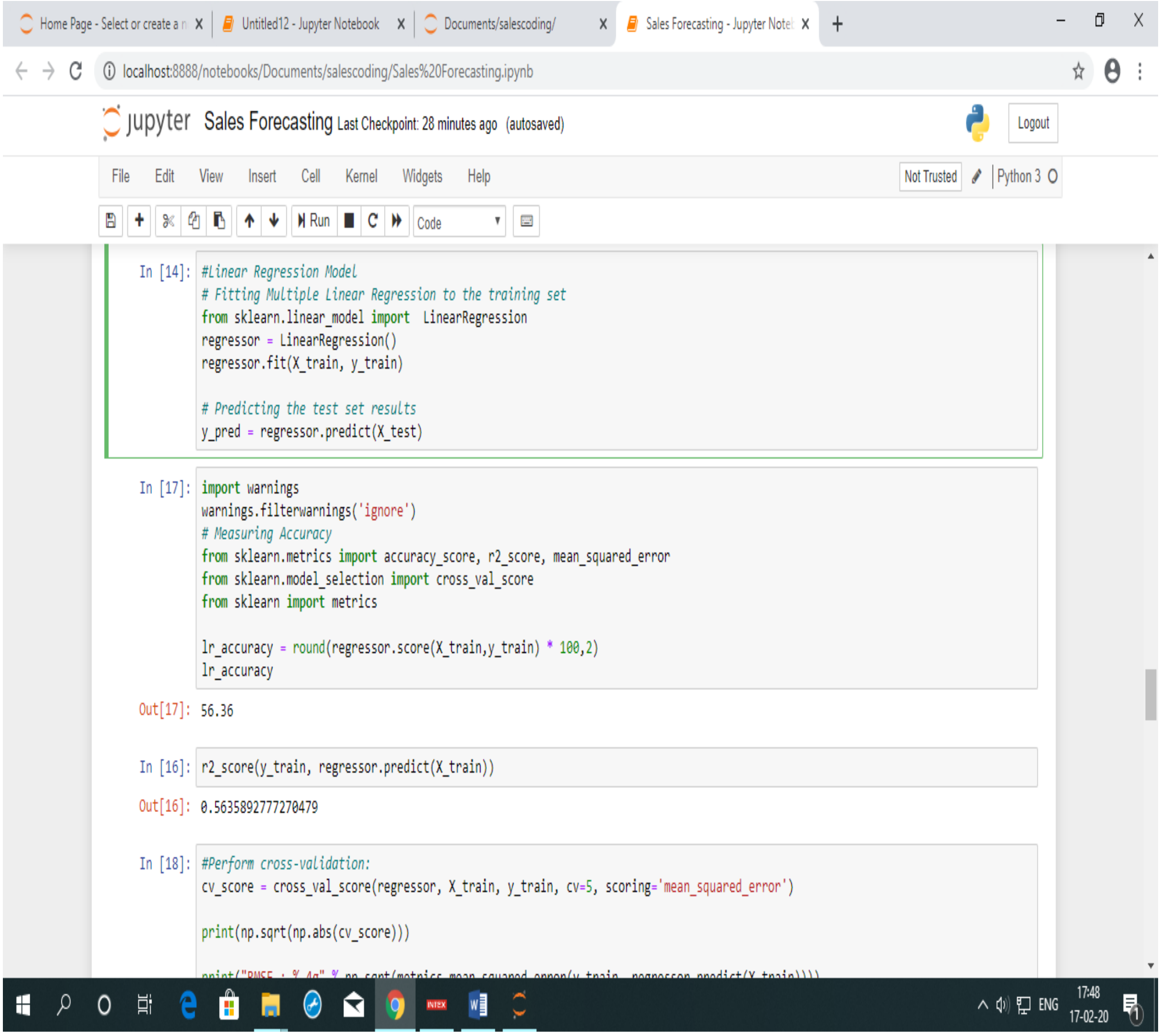


Fig C.14 Model Building using Linear Regression and shows accuracy rate, mean- squared value

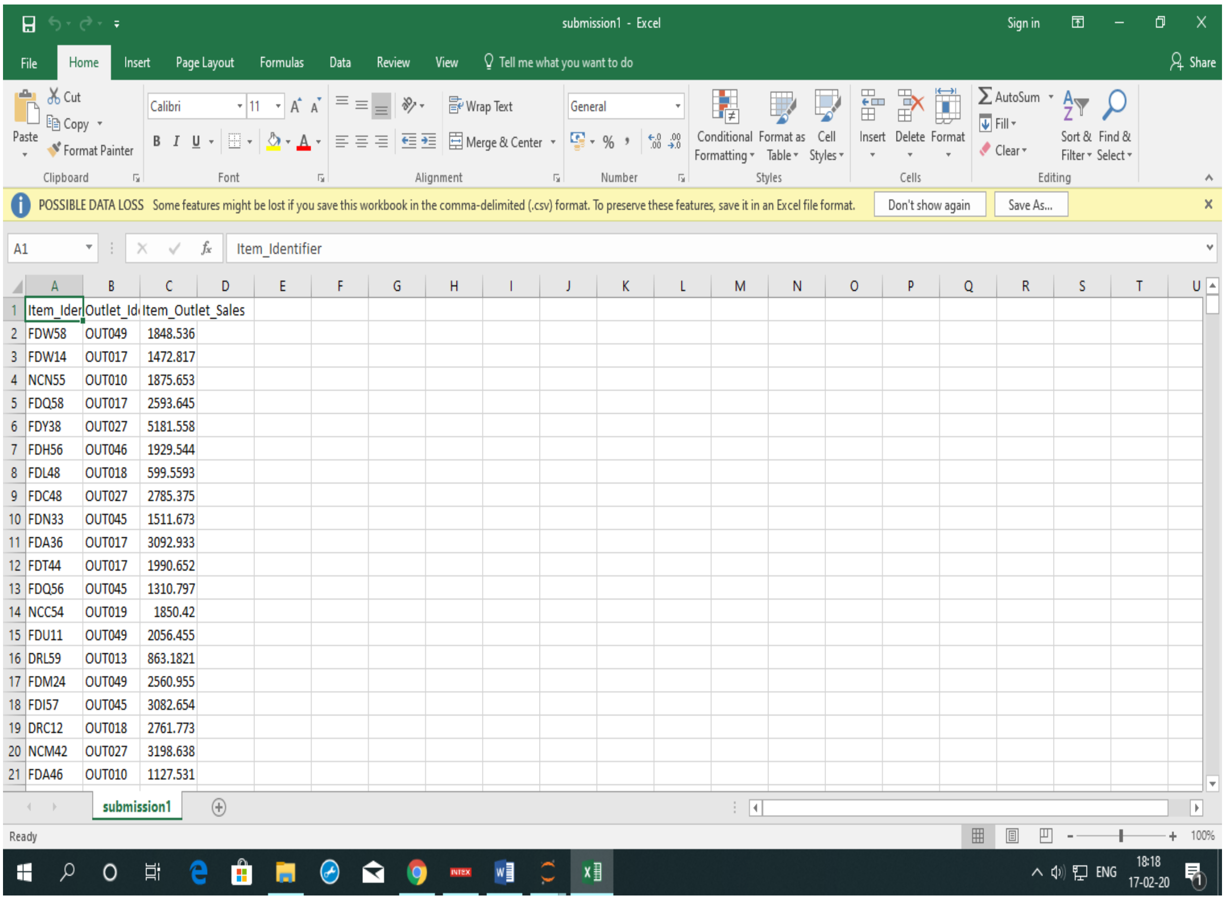


Fig C.15 After performing cross-validation, item _ identifier and Outlet _ identifier are taken as test data to predict the item _ outlet _ sales

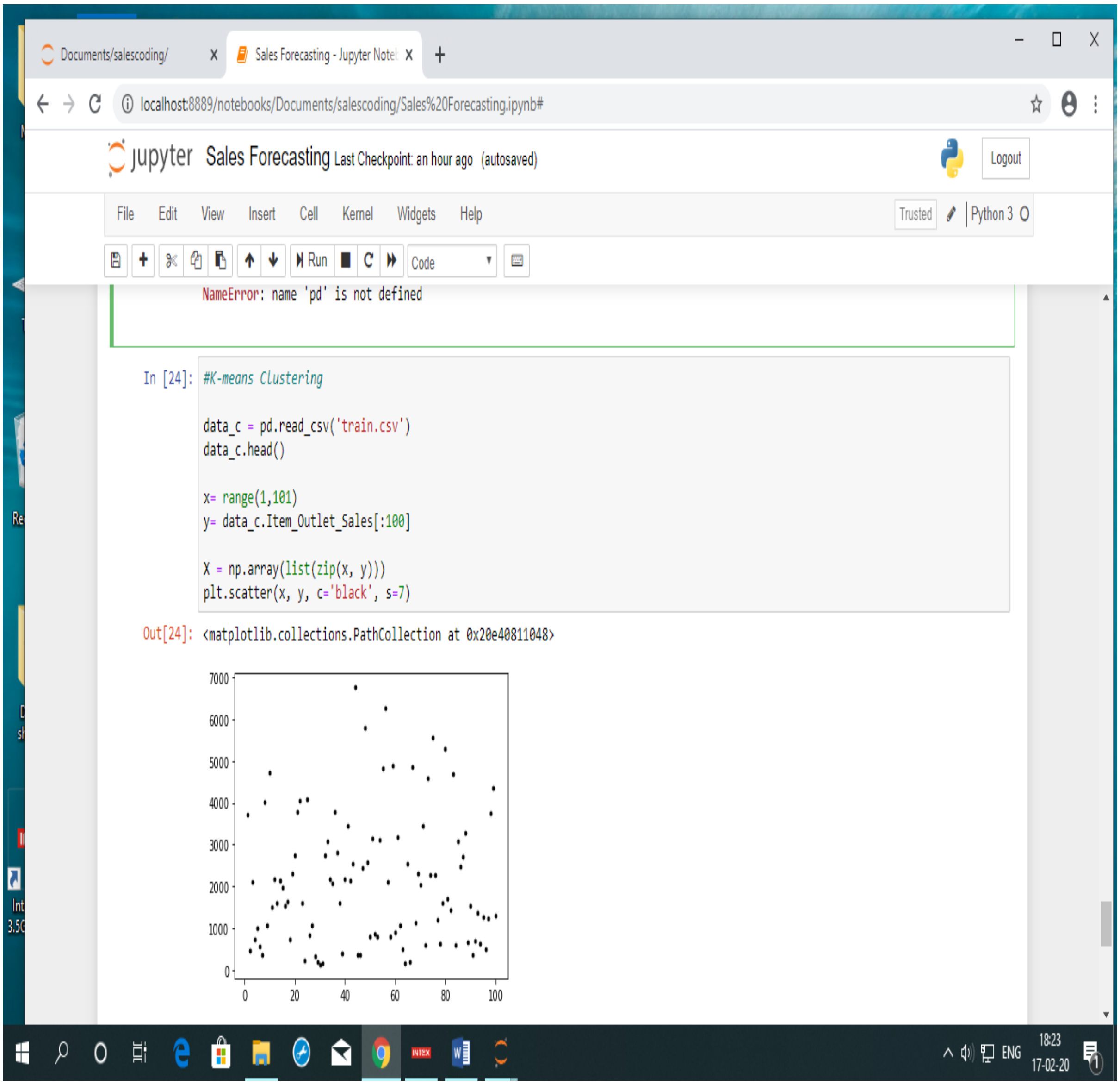


Fig C.16 K-means Clustering visualization

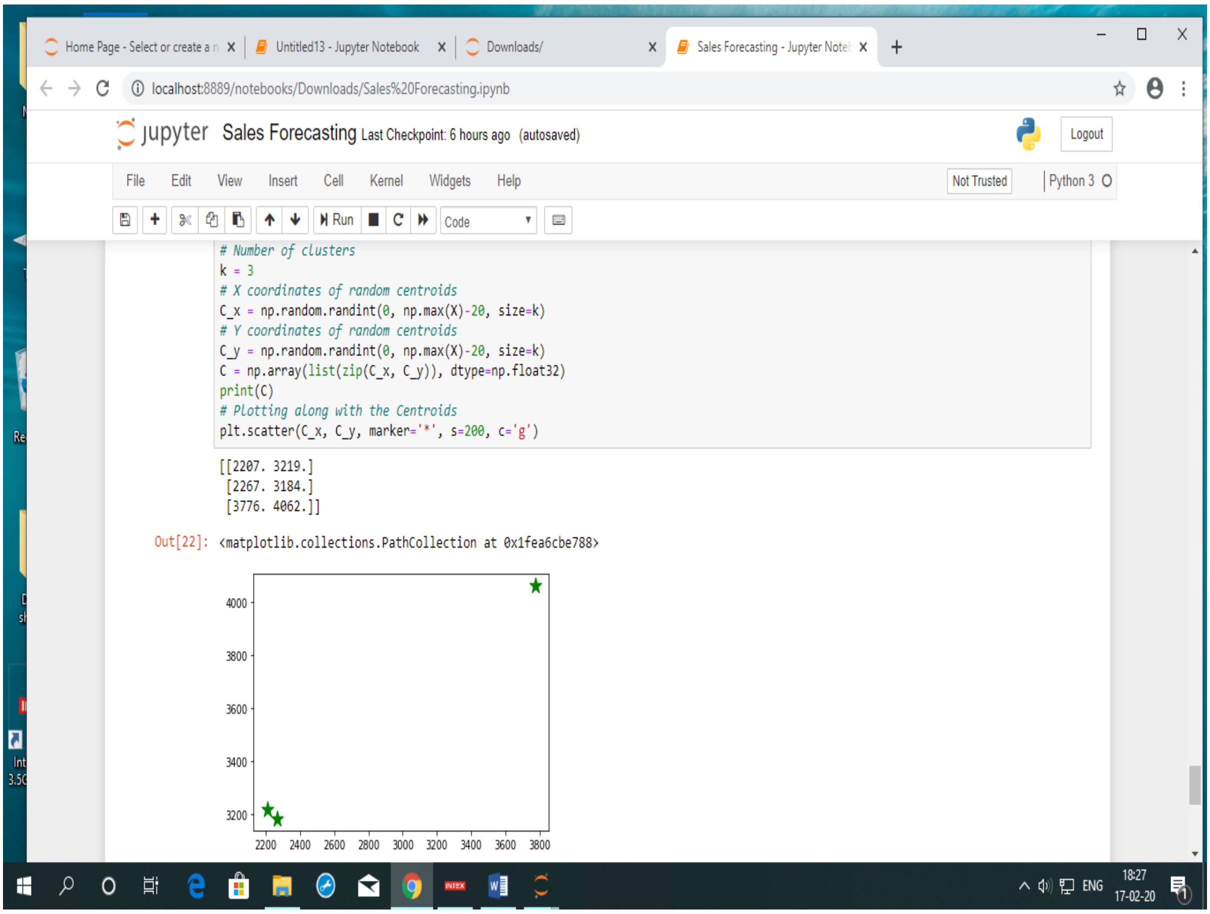


Fig C.17 Euclidean Distance Calculator to calculate the distance of the cluster

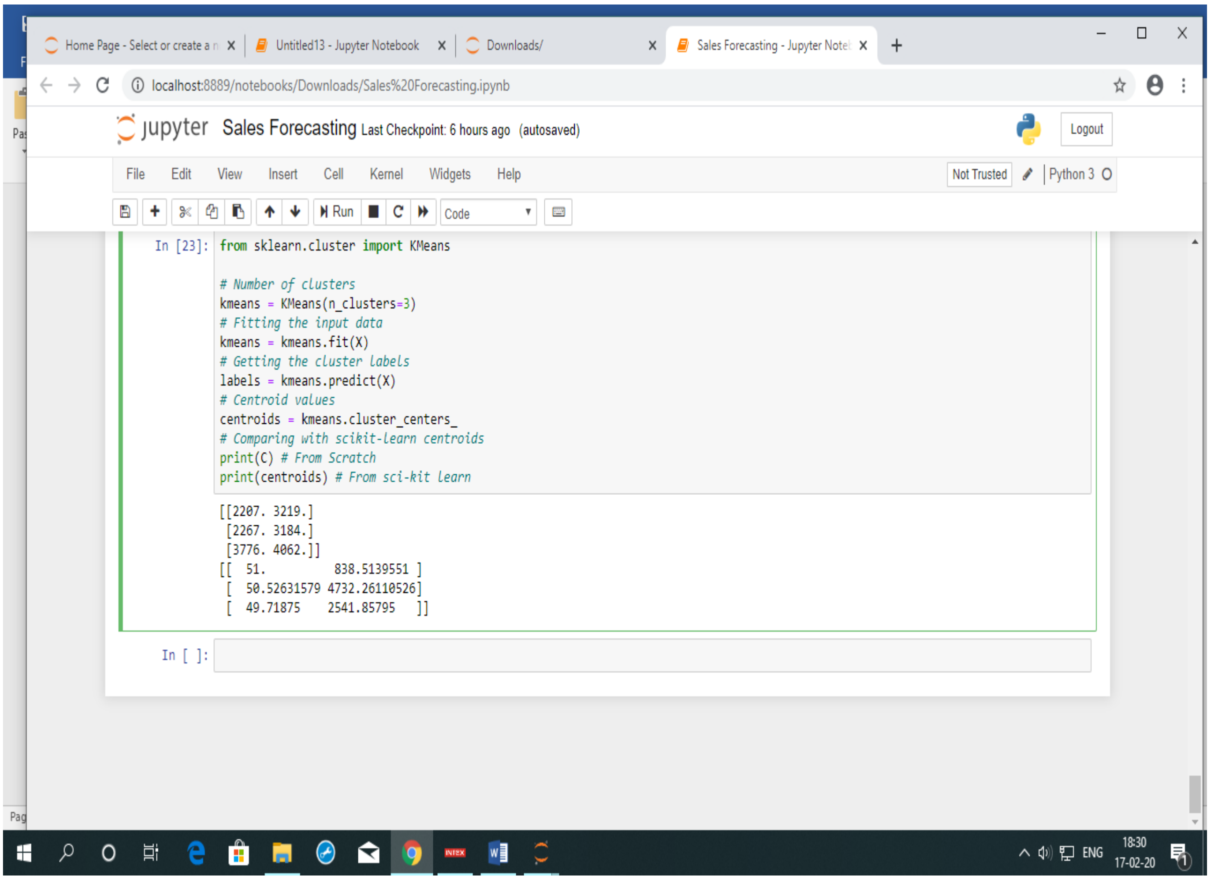


Fig C.18 Number of clusters, mean of the cluster and centroid of the clusters

D. SAMPLE CODE

Sales forecasting.py

```
# coding: utf-8

# In[1]:



import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os
os.chdir("...")



# In[2]:



train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")



# In[6]:



#Combine test and train into one file
train['source']='train'
test['source']='test'
data = pd.concat([train, test],ignore_index=True)
print(train.shape, test.shape, data.shape)



# In[7]:
```

```
data.head()


# In[8]:


#Numerical data summary:
data.describe()


# In[9]:


#Data Cleaning
#Check missing values:
data.apply(lambda x: sum(x.isnull()))


# In[10]:


#Filling missing values
data.Item_Outlet_Sales = data.Item_Outlet_Sales.fillna(data.Item_Outlet_Sales.mean())
data.Item_Weight = data.Item_Weight.fillna(data.Item_Weight.mean())
data['Outlet_Size'].value_counts()
data.Outlet_Size = data.Outlet_Size.fillna('Medium')

data.apply(lambda x: sum(x.isnull()))


# In[11]:
```

```
data.info()


# In[12]:


#Item type combine:
data['Item_Identifier'].value_counts()
data['Item_Type_Combined'] = data['Item_Identifier'].apply(lambda x: x[0:2])
data['Item_Type_Combined'] = data['Item_Type_Combined'].map({'FD':'Food',
                                                             'NC':'Non-Consumable',
                                                             'DR':'Drinks'})
data['Item_Type_Combined'].value_counts()


# In[13]:


#Numerical and One-Hot Coding of Categorical variables
#Import library:
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
#New variable for outlet
data['Outlet'] = le.fit_transform(data['Outlet_Identifier'])
var_mod =
['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Item_Type_Combined','Outle
t_Type','Outlet']
le = LabelEncoder()
for i in var_mod:
    data[i] = le.fit_transform(data[i])

#One Hot Coding:
```

```
data = pd.get_dummies(data,
columns=['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Outlet_Type','Item_
Type_Combined','Outlet'])

data.head()


# In[14]:


data.dtypes


# In[15]:


#Exporting data
import warnings
warnings.filterwarnings('ignore')
#Drop the columns which have been converted to different types:
data.drop(['Item_Type','Outlet_Establishment_Year'],axis=1,inplace=True)

#Divide into test and train:
train = data.loc[data['source']=="train"]
test = data.loc[data['source']=="test"]

#Drop unnecessary columns:
test.drop(['Item_Outlet_Sales','source'],axis=1,inplace=True)
train.drop(['source'],axis=1,inplace=True)

#Export files as modified versions:
train.to_csv("train_modified.csv",index=False)
test.to_csv("test_modified.csv",index=False)
```

```
# In[16]:



#Model Building
# Reading modified data
train2 = pd.read_csv("train_modified.csv")
test2 = pd.read_csv("test_modified.csv")

train2.head()

X_train = train2.drop(['Item_Outlet_Sales', 'Outlet_Identifier','Item_Identifier'], axis=1)
y_train = train2.Item_Outlet_Sales

X_test = test2.drop(['Outlet_Identifier','Item_Identifier'], axis=1)

print(X_train.head())

print(y_train.head())


# In[17]:



#Linear Regression Model
# Fitting Multiple Linear Regression to the training set
from sklearn.linear_model import  LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting the test set results
y_pred = regressor.predict(X_test)
```

```
# In[18]:



import warnings
warnings.filterwarnings('ignore')
# Measuring Accuracy
from sklearn.metrics import accuracy_score, r2_score, mean_squared_error
from sklearn.model_selection import cross_val_score
from sklearn import cross_validation, metrics

lr_accuracy = round(regressor.score(X_train,y_train) * 100,2)
lr_accuracy



# In[19]:



r2_score(y_train, regressor.predict(X_train))



# In[20]:



#Perform cross-validation:
cv_score = cross_val_score(regressor, X_train, y_train, cv=5,
scoring='mean_squared_error')

print(np.sqrt(np.abs(cv_score)))

print("RMSE : %.4g" % np.sqrt(metrics.mean_squared_error(y_train,
regressor.predict(X_train))))



# In[21]:
```

```
submission = pd.DataFrame({
'Item_Identifier':test2['Item_Identifier'],
'Outlet_Identifier':test2['Outlet_Identifier'],
'Item_Outlet_Sales': y_pred
},columns=['Item_Identifier','Outlet_Identifier','Item_Outlet_Sales'])

submission.to_csv('submission1.csv',index=False)


# In[22]:


#K-means Clustering

data_c = pd.read_csv('train.csv')
data_c.head()

x= range(1,101)
y= data_c.Item_Outlet_Sales[:100]

X = np.array(list(zip(x, y)))
plt.scatter(x, y, c='black', s=7)


# In[23]:


# Euclidean Distance Caculator
def dist(a, b, ax=1):
    return np.linalg.norm(a - b, axis=ax)
# Number of clusters
k = 3
```

```
# X coordinates of random centroids
C_x = np.random.randint(0, np.max(X)-20, size=k)
# Y coordinates of random centroids
C_y = np.random.randint(0, np.max(X)-20, size=k)
C = np.array(list(zip(C_x, C_y)), dtype=np.float32)
print(C)
# Plotting along with the Centroids
plt.scatter(C_x, C_y, marker='*', s=200, c='g')


# In[24]:


from sklearn.cluster import KMeans

# Number of clusters
kmeans = KMeans(n_clusters=3)
# Fitting the input data
kmeans = kmeans.fit(X)
# Getting the cluster labels
labels = kmeans.predict(X)
# Centroid values
centroids = kmeans.cluster_centers_
# Comparing with scikit-learn centroids
print(C) # From Scratch
print(centroids) # From sci-kit learn
```

Sales Prediction.ipynb


```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "metadata": {},
```

"outputs": [],
"source": [
 "import pandas as pd\n",
 "import numpy as np\n",
 "import matplotlib.pyplot as plt\n",
 "import os\n",
 "os.chdir(\"...\")"
]
},
{
"cell_type": "code",
"execution_count": 2,
"metadata": {},
"outputs": [],
"source": [
 "train = pd.read_csv(\"train.csv\")\n",
 "test = pd.read_csv(\"test.csv\")"
]
},
{
"cell_type": "code",
"execution_count": 6,
"metadata": {},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "(8523, 13) (5681, 12) (14204, 13)\n"
  ]
 },
 {
  "name": "stderr",
  "output_type": "stream",

```
     "text": [
      "C:\\Users\\Lenovo\\Anaconda3\\lib\\site-packages\\ipykernel_launcher.py:4: FutureWarning: Sorting because non-concatenation axis is not aligned. A future version\n",
      "of pandas will change to not sort by default.\n",
      "\n",
      "To accept the future behavior, pass 'sort=False'.\n",
      "\n",
      "To retain the current behavior and silence the warning, pass 'sort=True'.\n",
      "\n",
      "  after removing the cwd from sys.path.\n"
     ]
    }
   ],
   "source": [
    "#Combine test and train into one file\n",
    "train['source']='train'\n",
    "test['source']='test'\n",
    "data = pd.concat([train, test],ignore_index=True)\n",
    "print(train.shape, test.shape, data.shape)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 7,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
```

```
"\n",
"    .dataframe tbody tr th {\n",
"        vertical-align: top;\n",
"    }\n",
"\n",
"    .dataframe thead th {\n",
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>Item_Fat_Content</th>\n",
"      <th>Item_Identifier</th>\n",
"      <th>Item_MRP</th>\n",
"      <th>Item_Outlet_Sales</th>\n",
"      <th>Item_Type</th>\n",
"      <th>Item_Visibility</th>\n",
"      <th>Item_Weight</th>\n",
"      <th>Outlet_Establishment_Year</th>\n",
"      <th>Outlet_Identifier</th>\n",
"      <th>Outlet_Location_Type</th>\n",
"      <th>Outlet_Size</th>\n",
"      <th>Outlet_Type</th>\n",
"      <th>source</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>Low Fat</td>\n",
"      <td>FDA15</td>\n",
"      <td>249.8092</td>\n",
```

```
"    <td>3735.1380</td>\n",
"    <td>Dairy</td>\n",
"    <td>0.016047</td>\n",
"    <td>9.30</td>\n",
"    <td>1999</td>\n",
"    <td>OUT049</td>\n",
"    <td>Tier 1</td>\n",
"    <td>Medium</td>\n",
"    <td>Supermarket Type1</td>\n",
"    <td>train</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>1</th>\n",
"    <td>Regular</td>\n",
"    <td>DRC01</td>\n",
"    <td>48.2692</td>\n",
"    <td>443.4228</td>\n",
"    <td>Soft Drinks</td>\n",
"    <td>0.019278</td>\n",
"    <td>5.92</td>\n",
"    <td>2009</td>\n",
"    <td>OUT018</td>\n",
"    <td>Tier 3</td>\n",
"    <td>Medium</td>\n",
"    <td>Supermarket Type2</td>\n",
"    <td>train</td>\n",
"  </tr>\n",
"  <tr>\n",
"    <th>2</th>\n",
"    <td>Low Fat</td>\n",
"    <td>FDN15</td>\n",
"    <td>141.6180</td>\n",
"    <td>2097.2700</td>\n",
"    <td>Meat</td>\n",
```

```
"      <td>0.016760</td>\n",
"      <td>17.50</td>\n",
"      <td>1999</td>\n",
"      <td>OUT049</td>\n",
"      <td>Tier 1</td>\n",
"      <td>Medium</td>\n",
"      <td>Supermarket Type1</td>\n",
"      <td>train</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
"      <td>Regular</td>\n",
"      <td>FDX07</td>\n",
"      <td>182.0950</td>\n",
"      <td>732.3800</td>\n",
"      <td>Fruits and Vegetables</td>\n",
"      <td>0.000000</td>\n",
"      <td>19.20</td>\n",
"      <td>1998</td>\n",
"      <td>OUT010</td>\n",
"      <td>Tier 3</td>\n",
"      <td>NaN</td>\n",
"      <td>Grocery Store</td>\n",
"      <td>train</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>4</th>\n",
"      <td>Low Fat</td>\n",
"      <td>NCD19</td>\n",
"      <td>53.8614</td>\n",
"      <td>994.7052</td>\n",
"      <td>Household</td>\n",
"      <td>0.000000</td>\n",
"      <td>8.93</td>\n",
```

"   <td>1987</td>\n",
"   <td>OUT013</td>\n",
"   <td>Tier 3</td>\n",
"   <td>High</td>\n",
"   <td>Supermarket Type1</td>\n",
"   <td>train</td>\n",
"   </tr>\n",
" </tbody>\n",
"</table>\n",
"</div>"
],
"text/plain": [
" Item_Fat_Content Item_Identifier Item_MRP Item_Outlet_Sales \\\n",
"0   Low Fat   FDA15 249.8092   3735.1380   \n",
"1   Regular   DRC01   48.2692   443.4228   \n",
"2   Low Fat   FDN15 141.6180   2097.2700   \n",
"3   Regular   FDX07 182.0950   732.3800   \n",
"4   Low Fat   NCD19   53.8614   994.7052   \n",
"\n",
"   Item_Type Item_Visibility Item_Weight \\\n",
"0   Dairy   0.016047   9.30   \n",
"1   Soft Drinks   0.019278   5.92   \n",
"2   Meat   0.016760   17.50   \n",
"3 Fruits and Vegetables   0.000000   19.20   \n",
"4   Household   0.000000   8.93   \n",
"\n",
"   Outlet_Establishment_Year Outlet_Identifier Outlet_Location_Type \\\n",
"0   1999   OUT049   Tier 1   \n",
"1   2009   OUT018   Tier 3   \n",
"2   1999   OUT049   Tier 1   \n",
"3   1998   OUT010   Tier 3   \n",
"4   1987   OUT013   Tier 3   \n",
"\n",
" Outlet_Size   Outlet_Type source \n",

```
    "0      Medium  Supermarket Type1  train  \n",
    "1      Medium  Supermarket Type2  train  \n",
    "2      Medium  Supermarket Type1  train  \n",
    "3         NaN      Grocery Store  train  \n",
    "4        High  Supermarket Type1  train  "
   ]
  },
  "execution_count": 7,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "data.head()"
]
},
{
"cell_type": "code",
"execution_count": 8,
"metadata": {},
"outputs": [
 {
  "data": {
   "text/html": [
    "<div>\n",
    "<style scoped>\n",
    "    .dataframe tbody tr th:only-of-type {\n",
    "        vertical-align: middle;\n",
    "    }\n",
    "\n",
    "    .dataframe tbody tr th {\n",
    "        vertical-align: top;\n",
    "    }\n",
    "\n",
```

```
"    .dataframe thead th {\n",
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>Item_MRP</th>\n",
"      <th>Item_Outlet_Sales</th>\n",
"      <th>Item_Visibility</th>\n",
"      <th>Item_Weight</th>\n",
"      <th>Outlet_Establishment_Year</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>count</th>\n",
"      <td>14204.000000</td>\n",
"      <td>8523.000000</td>\n",
"      <td>14204.000000</td>\n",
"      <td>11765.000000</td>\n",
"      <td>14204.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>mean</th>\n",
"      <td>141.004977</td>\n",
"      <td>2181.288914</td>\n",
"      <td>0.065953</td>\n",
"      <td>12.792854</td>\n",
"      <td>1997.830681</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>std</th>\n",
```

```
"      <td>62.086938</td>\n",
"      <td>1706.499616</td>\n",
"      <td>0.051459</td>\n",
"      <td>4.652502</td>\n",
"      <td>8.371664</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>min</th>\n",
"      <td>31.290000</td>\n",
"      <td>33.290000</td>\n",
"      <td>0.000000</td>\n",
"      <td>4.555000</td>\n",
"      <td>1985.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>25%</th>\n",
"      <td>94.012000</td>\n",
"      <td>834.247400</td>\n",
"      <td>0.027036</td>\n",
"      <td>8.710000</td>\n",
"      <td>1987.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>50%</th>\n",
"      <td>142.247000</td>\n",
"      <td>1794.331000</td>\n",
"      <td>0.054021</td>\n",
"      <td>12.600000</td>\n",
"      <td>1999.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>75%</th>\n",
"      <td>185.855600</td>\n",
"      <td>3101.296400</td>\n",
```

"   <td>0.094037</td>\n",
"   <td>16.750000</td>\n",
"   <td>2004.000000</td>\n",
"   </tr>\n",
"   <tr>\n",
"   <th>max</th>\n",
"   <td>266.888400</td>\n",
"   <td>13086.964800</td>\n",
"   <td>0.328391</td>\n",
"   <td>21.350000</td>\n",
"   <td>2009.000000</td>\n",
"   </tr>\n",
" </tbody>\n",
"</table>\n",
"</div>"
],
"text/plain": [
"   Item_MRP Item_Outlet_Sales Item_Visibility   Item_Weight \\\n",
"count 14204.000000   8523.000000   14204.000000 11765.000000   \n",
"mean   141.004977   2181.288914   0.065953   12.792854   \n",
"std   62.086938   1706.499616   0.051459   4.652502   \n",
"min   31.290000   33.290000   0.000000   4.555000   \n",
"25%   94.012000   834.247400   0.027036   8.710000   \n",
"50%   142.247000   1794.331000   0.054021   12.600000   \n",
"75%   185.855600   3101.296400   0.094037   16.750000   \n",
"max   266.888400   13086.964800   0.328391   21.350000   \n",
"\n",
"   Outlet_Establishment_Year \n",
"count   14204.000000 \n",
"mean   1997.830681 \n",
"std   8.371664 \n",
"min   1985.000000 \n",
"25%   1987.000000 \n",
"50%   1999.000000 \n",

"75%   2004.000000 \n",
"max   2009.000000 "
]
},
"execution_count": 8,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"#Numerical data summary:\n",
"data.describe()"
]
},
{
"cell_type": "code",
"execution_count": 9,
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"Item_Fat_Content   0\n",
"Item_Identifier   0\n",
"Item_MRP   0\n",
"Item_Outlet_Sales   5681\n",
"Item_Type   0\n",
"Item_Visibility   0\n",
"Item_Weight   2439\n",
"Outlet_Establishment_Year   0\n",
"Outlet_Identifier   0\n",
"Outlet_Location_Type   0\n",
"Outlet_Size   4016\n",
"Outlet_Type   0\n",

"source   0\n",
"dtype: int64"
]
},
"execution_count": 9,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"#Data Cleaning\n",
"#Check missing values:\n",
"data.apply(lambda x: sum(x.isnull()))"
]
},
{
"cell_type": "code",
"execution_count": 10,
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"Item_Fat_Content   0\n",
"Item_Identifier   0\n",
"Item_MRP   0\n",
"Item_Outlet_Sales   0\n",
"Item_Type   0\n",
"Item_Visibility   0\n",
"Item_Weight   0\n",
"Outlet_Establishment_Year   0\n",
"Outlet_Identifier   0\n",
"Outlet_Location_Type   0\n",
"Outlet_Size   0\n",

"Outlet_Type 0\n",
"source 0\n",
"dtype: int64"
]
},
"execution_count": 10,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"#Filling missing values\n",
"data.Item_Outlet_Sales =
data.Item_Outlet_Sales.fillna(data.Item_Outlet_Sales.mean())\n",
"data.Item_Weight = data.Item_Weight.fillna(data.Item_Weight.mean())\n",
"data['Outlet_Size'].value_counts()\n",
"data.Outlet_Size = data.Outlet_Size.fillna('Medium')\n",
"\n",
"data.apply(lambda x: sum(x.isnull()))"
]
},
{
"cell_type": "code",
"execution_count": 11,
"metadata": {},
"outputs": [
{
"name": "stdout",
"output_type": "stream",
"text": [
"<class 'pandas.core.frame.DataFrame'>\n",
"RangeIndex: 14204 entries, 0 to 14203\n",
"Data columns (total 13 columns):\n",
"Item_Fat_Content 14204 non-null object\n",

"Item_Identifier   14204 non-null object\n",
"Item_MRP   14204 non-null float64\n",
"Item_Outlet_Sales   14204 non-null float64\n",
"Item_Type   14204 non-null object\n",
"Item_Visibility   14204 non-null float64\n",
"Item_Weight   14204 non-null float64\n",
"Outlet_Establishment_Year   14204 non-null int64\n",
"Outlet_Identifier   14204 non-null object\n",
"Outlet_Location_Type   14204 non-null object\n",
"Outlet_Size   14204 non-null object\n",
"Outlet_Type   14204 non-null object\n",
"source   14204 non-null object\n",
"dtypes: float64(4), int64(1), object(8)\n",
"memory usage: 1.4+ MB\n"
]
}
],
"source": [
"data.info()"
]
},
{
"cell_type": "code",
"execution_count": 12,
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"Food   10201\n",
"Non-Consumable   2686\n",
"Drinks   1317\n",
"Name: Item_Type_Combined, dtype: int64"
]

```
    },
    "execution_count": 12,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "#Item type combine:\n",
   "data['Item_Identifier'].value_counts()\n",
   "data['Item_Type_Combined'] = data['Item_Identifier'].apply(lambda x: x[0:2])\n",
   "data['Item_Type_Combined'] = data['Item_Type_Combined'].map({'FD':'Food',\n",
   "                                                             'NC':'Non-Consumable',\n",
   "                                                             'DR':'Drinks'})\n",
   "data['Item_Type_Combined'].value_counts()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 13,
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/html": [
      "<div>\n",
      "&lt;style scoped&gt;\n",
      "    .dataframe tbody tr th:only-of-type {\n",
      "        vertical-align: middle;\n",
      "    }\n",
      "\n",
      "    .dataframe tbody tr th {\n",
      "        vertical-align: top;\n",
      "    }\n",
      "\n",
```

```
"    .dataframe thead th {\n",
"        text-align: right;\n",
"    }\n",
"</style>\n",
"<table border=\"1\" class=\"dataframe\">\n",
"  <thead>\n",
"    <tr style=\"text-align: right;\">\n",
"      <th></th>\n",
"      <th>Item_Identifier</th>\n",
"      <th>Item_MRP</th>\n",
"      <th>Item_Outlet_Sales</th>\n",
"      <th>Item_Type</th>\n",
"      <th>Item_Visibility</th>\n",
"      <th>Item_Weight</th>\n",
"      <th>Outlet_Establishment_Year</th>\n",
"      <th>Outlet_Identifier</th>\n",
"      <th>source</th>\n",
"      <th>Item_Fat_Content_0</th>\n",
"      <th>...</th>\n",
"      <th>Outlet_0</th>\n",
"      <th>Outlet_1</th>\n",
"      <th>Outlet_2</th>\n",
"      <th>Outlet_3</th>\n",
"      <th>Outlet_4</th>\n",
"      <th>Outlet_5</th>\n",
"      <th>Outlet_6</th>\n",
"      <th>Outlet_7</th>\n",
"      <th>Outlet_8</th>\n",
"      <th>Outlet_9</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
```

```
"    <td>FDA15</td>\n",
"    <td>249.8092</td>\n",
"    <td>3735.1380</td>\n",
"    <td>Dairy</td>\n",
"    <td>0.016047</td>\n",
"    <td>9.30</td>\n",
"    <td>1999</td>\n",
"    <td>OUT049</td>\n",
"    <td>train</td>\n",
"    <td>0</td>\n",
"    <td>...</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>1</td>\n",
"   </tr>\n",
"   <tr>\n",
"    <th>1</th>\n",
"    <td>DRC01</td>\n",
"    <td>48.2692</td>\n",
"    <td>443.4228</td>\n",
"    <td>Soft Drinks</td>\n",
"    <td>0.019278</td>\n",
"    <td>5.92</td>\n",
"    <td>2009</td>\n",
"    <td>OUT018</td>\n",
"    <td>train</td>\n",
"    <td>0</td>\n",
```

```
"      <td>...</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>FDN15</td>\n",
"      <td>141.6180</td>\n",
"      <td>2097.2700</td>\n",
"      <td>Meat</td>\n",
"      <td>0.016760</td>\n",
"      <td>17.50</td>\n",
"      <td>1999</td>\n",
"      <td>OUT049</td>\n",
"      <td>train</td>\n",
"      <td>0</td>\n",
"      <td>...</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
```

```
"    <td>1</td>\n",
"   </tr>\n",
"   <tr>\n",
"    <th>3</th>\n",
"    <td>FDX07</td>\n",
"    <td>182.0950</td>\n",
"    <td>732.3800</td>\n",
"    <td>Fruits and Vegetables</td>\n",
"    <td>0.000000</td>\n",
"    <td>19.20</td>\n",
"    <td>1998</td>\n",
"    <td>OUT010</td>\n",
"    <td>train</td>\n",
"    <td>0</td>\n",
"    <td>...</td>\n",
"    <td>1</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"    <td>0</td>\n",
"   </tr>\n",
"   <tr>\n",
"    <th>4</th>\n",
"    <td>NCD19</td>\n",
"    <td>53.8614</td>\n",
"    <td>994.7052</td>\n",
"    <td>Household</td>\n",
"    <td>0.000000</td>\n",
"    <td>8.93</td>\n",
```

```
"      <td>1987</td>\n",
"      <td>OUT013</td>\n",
"      <td>train</td>\n",
"      <td>0</td>\n",
"      <td>...</td>\n",
"      <td>0</td>\n",
"      <td>1</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"      <td>0</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>5 rows × 37 columns</p>\n",
"</div>"
],
"text/plain": [
"  Item_Identifier  Item_MRP  Item_Outlet_Sales             Item_Type  \\\n",
"0           FDA15  249.8092          3735.1380                 Dairy   \n",
"1           DRC01   48.2692           443.4228           Soft Drinks   \n",
"2           FDN15  141.6180          2097.2700                  Meat   \n",
"3           FDX07  182.0950           732.3800  Fruits and Vegetables   \n",
"4           NCD19   53.8614           994.7052             Household   \n",
"\n",
"   Item_Visibility  Item_Weight  Outlet_Establishment_Year Outlet_Identifier  \\\n",
"0         0.016047         9.30                       1999            OUT049   \n",
"1         0.019278         5.92                       2009            OUT018   \n",
"2         0.016760        17.50                       1999            OUT049   \n",
"3         0.000000        19.20                       1998            OUT010   \n",
```

"4   0.000000   8.93   1987   OUT013   \n",
"\n",
" source Item_Fat_Content_0 ... Outlet_0 Outlet_1 Outlet_2 Outlet_3 \\\n",
"0 train   0 ...   0   0   0   0   \n",
"1 train   0 ...   0   0   0   1   \n",
"2 train   0 ...   0   0   0   0   \n",
"3 train   0 ...   1   0   0   0   \n",
"4 train   0 ...   0   1   0   0   \n",
"\n",
"   Outlet_4 Outlet_5 Outlet_6 Outlet_7 Outlet_8 Outlet_9 \n",
"0   0   0   0   0   0   1 \n",
"1   0   0   0   0   0   0 \n",
"2   0   0   0   0   0   1 \n",
"3   0   0   0   0   0   0 \n",
"4   0   0   0   0   0   0 \n",
"\n",
"[5 rows x 37 columns]"
]
},
"execution_count": 13,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"#Numerical and One-Hot Coding of Categorical variables\n",
"#Import library:\n",
"from sklearn.preprocessing import LabelEncoder\n",
"le = LabelEncoder()\n",
"#New variable for outlet\n",
"data['Outlet'] = le.fit_transform(data['Outlet_Identifier'])\n",
"var_mod =
['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Item_Type_Combined','Outlet_Type','Outlet']\n",

```
    "le = LabelEncoder()\n",
    "for i in var_mod:\n",
    "    data[i] = le.fit_transform(data[i])\n",
    "\n",
    "#One Hot Coding:\n",
    "data = pd.get_dummies(data,
columns=['Item_Fat_Content','Outlet_Location_Type','Outlet_Size','Outlet_Type','Item_Type
_Combined','Outlet'])\n",
    "\n",
    "data.head()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 14,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "Item_Identifier              object\n",
       "Item_MRP                    float64\n",
       "Item_Outlet_Sales           float64\n",
       "Item_Type                    object\n",
       "Item_Visibility             float64\n",
       "Item_Weight                 float64\n",
       "Outlet_Establishment_Year     int64\n",
       "Outlet_Identifier            object\n",
       "source                       object\n",
       "Item_Fat_Content_0            uint8\n",
       "Item_Fat_Content_1            uint8\n",
       "Item_Fat_Content_2            uint8\n",
       "Item_Fat_Content_3            uint8\n",
       "Item_Fat_Content_4            uint8\n",
```

"Outlet_Location_Type_0 uint8\n",
"Outlet_Location_Type_1 uint8\n",
"Outlet_Location_Type_2 uint8\n",
"Outlet_Size_0 uint8\n",
"Outlet_Size_1 uint8\n",
"Outlet_Size_2 uint8\n",
"Outlet_Type_0 uint8\n",
"Outlet_Type_1 uint8\n",
"Outlet_Type_2 uint8\n",
"Outlet_Type_3 uint8\n",
"Item_Type_Combined_0 uint8\n",
"Item_Type_Combined_1 uint8\n",
"Item_Type_Combined_2 uint8\n",
"Outlet_0 uint8\n",
"Outlet_1 uint8\n",
"Outlet_2 uint8\n",
"Outlet_3 uint8\n",
"Outlet_4 uint8\n",
"Outlet_5 uint8\n",
"Outlet_6 uint8\n",
"Outlet_7 uint8\n",
"Outlet_8 uint8\n",
"Outlet_9 uint8\n",
"dtype: object"
]
},
"execution_count": 14,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"data.dtypes"
]

```
},
{
 "cell_type": "code",
 "execution_count": 15,
 "metadata": {},
 "outputs": [],
 "source": [
  "#Exporting data\n",
  "import warnings\n",
  "warnings.filterwarnings('ignore')\n",
  "#Drop the columns which have been converted to different types:\n",
  "data.drop(['Item_Type','Outlet_Establishment_Year'],axis=1,inplace=True)\n",
  "\n",
  "#Divide into test and train:\n",
  "train = data.loc[data['source']==\"train\"]\n",
  "test = data.loc[data['source']==\"test\"]\n",
  "\n",
  "#Drop unnecessary columns:\n",
  "test.drop(['Item_Outlet_Sales','source'],axis=1,inplace=True)\n",
  "train.drop(['source'],axis=1,inplace=True)\n",
  "\n",
  "#Export files as modified versions:\n",
  "train.to_csv(\"train_modified.csv\",index=False)\n",
  "test.to_csv(\"test_modified.csv\",index=False)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 16,
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
```

```
"text": [
 "   Item_MRP  Item_Visibility  Item_Weight  Item_Fat_Content_0  \\\n",
 "0  249.8092         0.016047         9.30                   0   \n",
 "1   48.2692         0.019278         5.92                   0   \n",
 "2  141.6180         0.016760        17.50                   0   \n",
 "3  182.0950         0.000000        19.20                   0   \n",
 "4   53.8614         0.000000         8.93                   0   \n",
 "\n",
 "   Item_Fat_Content_1  Item_Fat_Content_2  Item_Fat_Content_3  \\\n",
 "0                   1                   0                   0   \n",
 "1                   0                   1                   0   \n",
 "2                   1                   0                   0   \n",
 "3                   0                   1                   0   \n",
 "4                   1                   0                   0   \n",
 "\n",
 "   Item_Fat_Content_4  Outlet_Location_Type_0  Outlet_Location_Type_1  ...  \\\n",
 "0                   0                       1                       0  ...   \n",
 "1                   0                       0                       0  ...   \n",
 "2                   0                       1                       0  ...   \n",
 "3                   0                       0                       0  ...   \n",
 "4                   0                       0                       0  ...   \n",
 "\n",
 "   Outlet_0  Outlet_1  Outlet_2  Outlet_3  Outlet_4  Outlet_5  Outlet_6  \\\n",
 "0         0         0         0         0         0         0         0   \n",
 "1         0         0         0         1         0         0         0   \n",
 "2         0         0         0         0         0         0         0   \n",
 "3         1         0         0         0         0         0         0   \n",
 "4         0         1         0         0         0         0         0   \n",
 "\n",
 "   Outlet_7  Outlet_8  Outlet_9  \n",
 "0         0         0         1  \n",
 "1         0         0         0  \n",
 "2         0         0         1  \n",
 "3         0         0         0  \n",
```

"4   0   0   0 \n",
"\n",
"[5 rows x 31 columns]\n",
"0   3735.1380\n",
"1   443.4228\n",
"2   2097.2700\n",
"3   732.3800\n",
"4   994.7052\n",
"Name: Item_Outlet_Sales, dtype: float64\n"
]
}
],
"source": [
"#Model Building\n",
"# Reading modified data\n",
"train2 = pd.read_csv(\"train_modified.csv\")\n",
"test2 = pd.read_csv(\"test_modified.csv\")\n",
"\n",
"train2.head()\n",
"\n",
"X_train = train2.drop(['Item_Outlet_Sales', 'Outlet_Identifier','Item_Identifier'], axis=1)\n",
"y_train = train2.Item_Outlet_Sales\n",
"\n",
"X_test = test2.drop(['Outlet_Identifier','Item_Identifier'], axis=1)\n",
"\n",
"print(X_train.head())\n",
"\n",
"print(y_train.head())\n"
]
},
{
"cell_type": "code",
"execution_count": 17,

```
   "metadata": {},
   "outputs": [],
   "source": [
    "#Linear Regression Model\n",
    "# Fitting Multiple Linear Regression to the training set\n",
    "from sklearn.linear_model import  LinearRegression\n",
    "regressor = LinearRegression()\n",
    "regressor.fit(X_train, y_train)\n",
    "\n",
    "# Predicting the test set results\n",
    "y_pred = regressor.predict(X_test)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 18,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "56.36"
      ]
     },
     "execution_count": 18,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "import warnings\n",
    "warnings.filterwarnings('ignore')\n",
    "# Measuring Accuracy\n",
    "from sklearn.metrics import accuracy_score, r2_score, mean_squared_error\n",
```

```
   "from sklearn.model_selection import cross_val_score\n",
   "from sklearn import cross_validation, metrics\n",
   "\n",
   "lr_accuracy = round(regressor.score(X_train,y_train) * 100,2)\n",
   "lr_accuracy"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 19,
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "0.563589277727048"
     ]
    },
    "execution_count": 19,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "r2_score(y_train, regressor.predict(X_train))"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 20,
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
```

```
     "output_type": "stream",
     "text": [
      "[1150.93927648 1118.68414103 1112.89657923 1126.30724065 1140.59735737]\n",
      "RMSE : 1127\n"
     ]
    }
   ],
   "source": [
    "#Perform cross-validation:\n",
    "cv_score = cross_val_score(regressor, X_train, y_train, cv=5,
scoring='mean_squared_error')\n",
    "\n",
    "print(np.sqrt(np.abs(cv_score)))\n",
    "\n",
    "print(\"RMSE : %.4g\" % np.sqrt(metrics.mean_squared_error(y_train,
regressor.predict(X_train))))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 21,
   "metadata": {},
   "outputs": [],
   "source": [
    "submission = pd.DataFrame({\n",
    "'Item_Identifier':test2['Item_Identifier'],\n",
    "'Outlet_Identifier':test2['Outlet_Identifier'],\n",
    "'Item_Outlet_Sales': y_pred\n",
    "},columns=['Item_Identifier','Outlet_Identifier','Item_Outlet_Sales'])\n",
    "\n",
    "submission.to_csv('submission1.csv',index=False)"
   ]
  },
  {
```

```
"cell_type": "code",
"execution_count": 22,
"metadata": {},
"outputs": [
 {
  "data": {
   "text/plain": [
    "<matplotlib.collections.PathCollection at 0xd6f0d68>"
   ]
  },
  "execution_count": 22,
  "metadata": {},
  "output_type": "execute_result"
 },
 {
  "data": {
   "image/png":
   "text/plain": [
    "<Figure size 432x288 with 1 Axes>"
   ]
  },
  "metadata": {},
  "output_type": "display_data"
 }
],
"source": [
 "#K-means Clustering\n",
 "\n",
 "data_c = pd.read_csv('train.csv')\n",
 "data_c.head()\n",
 "\n",
 "x= range(1,101)\n",
 "y= data_c.Item_Outlet_Sales[:100]\n",
 "    \n",
```

```
    "X = np.array(list(zip(x, y)))\n",
    "plt.scatter(x, y, c='black', s=7)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 23,
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "[[4802. 3196.]\n",
      " [3345. 2114.]\n",
      " [ 342. 1195.]]\n"
     ]
    },
    {
     "data": {
      "text/plain": [
       "<matplotlib.collections.PathCollection at 0xd788be0>"
      ]
     },
     "execution_count": 23,
     "metadata": {},
     "output_type": "execute_result"
    },
    {
     "data": {
      "image/png":       "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
```

"metadata": {},
"output_type": "display_data"
}
],
"source": [
"# Euclidean Distance Caculator\n",
"def dist(a, b, ax=1):\n",
"   return np.linalg.norm(a - b, axis=ax)\n",
"# Number of clusters\n",
"k = 3\n",
"# X coordinates of random centroids\n",
"C_x = np.random.randint(0, np.max(X)-20, size=k)\n",
"# Y coordinates of random centroids\n",
"C_y = np.random.randint(0, np.max(X)-20, size=k)\n",
"C = np.array(list(zip(C_x, C_y)), dtype=np.float32)\n",
"print(C)\n",
"# Plotting along with the Centroids\n",
"plt.scatter(C_x, C_y, marker='*', s=200, c='g')"
]
},
{
"cell_type": "code",
"execution_count": 24,
"metadata": {},
"outputs": [
{
"name": "stdout",
"output_type": "stream",
"text": [
"[[4802. 3196.]\n",
" [3345. 2114.]\n",
" [342. 1195.]]\n",
"[[50.52631579 4732.26110526]\n",
" [51.6   855.699476]\n",

" [48.70967742 2569.08594839]]\n"
]
}
],
"source": [
"from sklearn.cluster import KMeans\n",
"\n",
"# Number of clusters\n",
"kmeans = KMeans(n_clusters=3)\n",
"# Fitting the input data\n",
"kmeans = kmeans.fit(X)\n",
"# Getting the cluster labels\n",
"labels = kmeans.predict(X)\n",
"# Centroid values\n",
"centroids = kmeans.cluster_centers_\n",
"# Comparing with scikit-learn centroids\n",
"print(C) # From Scratch\n",
"print(centroids) # From sci-kit learn"
]
},
{
"cell_type": "code",
"execution_count": null,
"metadata": {},
"outputs": [],
"source": []
}
],
"metadata": {
"kernelspec": {
"display_name": "Python 3",
"language": "python",
"name": "python3"
},

```
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.7.4"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 2
}
```