RESEARCH ARTICLE OPEN ACCESS

A Survey on Feature Engineering and Data Visualization for Result Predictions

Pratibha Chadha

Department of Computer Science & Technology
Punjab Technical University – Jalandhar

ABSTRACT

Feature engineering refers to the process of using domain knowledge to select and transform the most relevant variables from raw data when creating a predictive model using machine learning or statistical modeling. The goal of feature engineering and selection is to improve the performance of machine learning (ML) algorithms.This technology has achieved great success in many applications such as image analysis and predictions. Recent developments in predictions have led Big Data to come into the role, which provides great methods to produce a result, but possesses many challenging issues in data mining and information processing owing to its characteristics of large volume, variety, velocity, and veracity. This can lead to the integrity of data at stake, thus in past few years deep learning and feature engineering have evolved together to not only generate appropriate meaningful data for predictions, out of raw data but to formulate the whole process towards automation as well.This paper evolves around the searches and predictions made with the help of feature engineering algorithms, which can be used to overcome certain drawbacks of Big Data and data mining such as dealing with missing values, dealing noises or inappropriate data, this process is called data cleaning, reaching more accuracy in making predictions and retaining the integrity of data.

Keywords: - Feature Engineering Machine Learning Data Visualization Matplotlib Pandas

I. INTRODUCTION

1. Feature Engineering

Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data. -Tomasz Malisiewicz. Feature engineering is the process of using domain knowledge to extract features (characteristics, properties, attributes) from the raw data. The motivation is to use these extra features to improve the quality of results from a machine learning process, compared with supplying only raw data to the machine learning process.

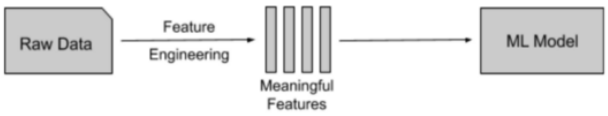

Figure 1: Feature Engineering Architecture

Dealing with Non-numerical Data
First, you have to convert non-numerical features into a numerical format.
There are two types of string data as **categorical data** and **continuous data.**

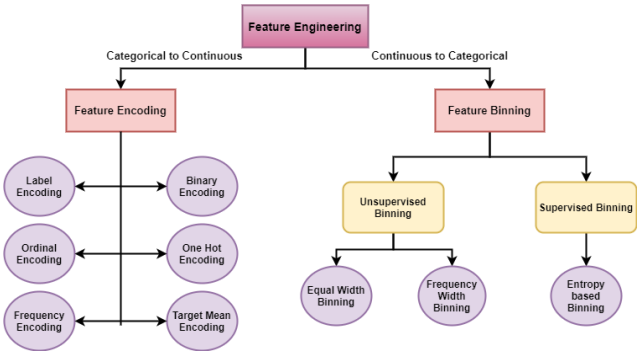

Figure 2: Types of methods to select features

II. CATEGORICAL FEATURES

Categorical data/ features are variables that have a finite number of labels or categories. Based on nature, categorical data can be classified into ordinal data and nominal data. An attribute is an ordinal attribute when there is a natural ordering of values. For example, the feature critical level can be either low, medium, or high and it has a certain order. When we cannot derive any order or relationship among categories, it is referred to as a nominal feature. For example, provinces or districts in a country have no order. Yet that is a categorical attribute as there is a fixed number of provinces or districts in a country.
There are a few popular techniques to convert categorical data into a numerical format. Even for algorithms like a decision tree and random forest which can work with categorical data as it is, encoding can make sense.

1. Label Encoding

Label encoding transforms non-numerical labels into numerical labels. This encoding type is better for ordinal

categorical data and if the number of categorical features in the dataset is huge, you may often use label encoding. Because an encoding technique like hot encoding can lead to high memory consumption. You can easily import label encoder from python scikit learn library. However, label encoding can make an ml model misunderstand that there is some kind of ordering in the data. To overcome this problem we may use one-hot-encoding.

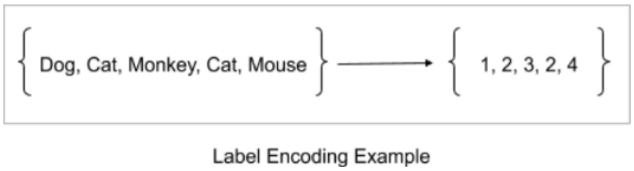

Figure 3: Label Encoding Example

2. One Hot Encoding

In one hot encoding, we make dummy columns for each class of a categorical attribute. For each dummy attribute, the presence of the class is represented as one, and absence is represented as zero. So, the numbers are represented by ones and zeroes. We use one hot encoding with nominal data. You can easily import one hot encoder from the python scikit learn library.

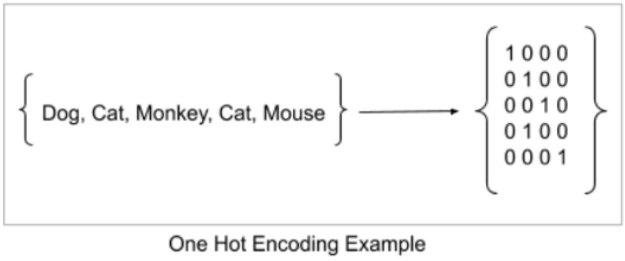

Figure 4: One Hot Encoding Example

3. Binary Encoding

In binary encoding, classes are first converted to the numerical format and then they are converted to their relevant binary strings. The binary digits are then split into separate columns.

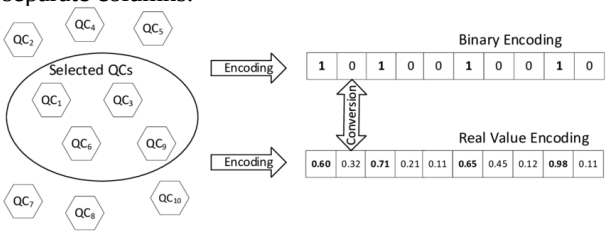

Figure 5: Binary Encoding Example

4. Tensor flow Feature Columns

Feature columns bridge input data with your machine learning model. Tensor flow provides a module that contains nine functions to deal with feature columns. These functions are listed below.

Numerical column
Bucketized column
Categorical identity column
Categorical vocabulary column (vocabulary file)
Categorical vocabulary column (vocabulary list)
Hashed column
Crossed column
Indicator column
Embedding column

Continuous Features

In mathematics, if a variable can take any value between a minimum and a maximum, then it is called a continuous variable. The same thing applies here. However, if the variable doesn't have a minimum and a maximum, then we might get into trouble as the variable may create sparse features for the machine learning model. Sparse features won't make any sense for a machine learning model and in my opinion, it's better to get rid of them. But the problem is dropping features from a dataset makes an ml algorithm less accurate. So, we should try every possibility to get that feature into a useful format. If there is no option, then it is better to drop the feature since it makes the machine learning model more accurate than without it.

In some cases, we might be able to derive more useful numerical features from string data. For example, consider a feature column with IP addresses. As a feature, IP address is string data. We can convert an IP address into an integer by breaking it into four octets and applying a simple mathematical operation. This integer data can be fed into an ml model and is expected to work well. But we can derive a more useful feature by getting longitude and latitude from the IP. You can use a free API or a database or any other technique to convert IP into relevant coordinates. In my opinion, we may be able to derive a more meaningful feature/s for a machine learning model by taking geolocation coordinates instead of simply converting IP into an integer.

As you might understand, handling continuous variables is a bit more difficult task. Yet there are certain popular techniques to deal with continuous variables. Listed below are only a few, and you may find more techniques.

1. Binning the Variable

Binning is dividing continuous numerical variables into groups. It is done to discover patterns of variables that are difficult to identify otherwise. For example, a list of ages can be divided into bins based on value ranges. However, the most important thing is the bin size. You may decide on an optimal bin size according to your hypothesis.

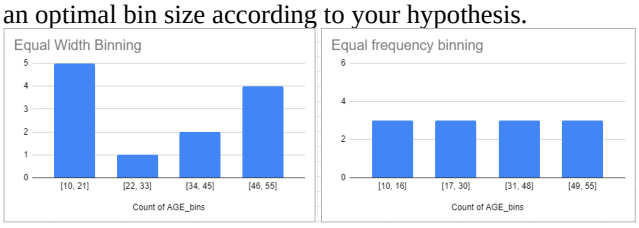

Figure 6: Binning Example

2. Normalization

Normalization is used to change the values of numerical columns into a common scale. In mathematics, you may find that normally distributed data is easy to read and interpret. Many machine learning algorithms assume a normal distribution among their data points. Especially normalization is very helpful in clustering techniques like K-means. A commonly used normalization technique is taking the z-score.

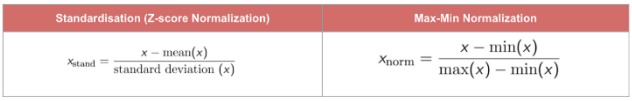

Figure 7: Example implementing normalization

3. Understanding the Business Logic

In some cases, data alone won't suggest any patterns. But understanding the business logic or the domain can help you to figure out more meaningful features. That is why you may often find one or more domain experts in a machine learning group. They know the business domain well and how it operates and can create more meaningful features out of existing features which can derive a recognizable pattern for a machine learning model. This technique is not only for continuous data and can be helpful for other types of data as well.

Feature Selection

After dealing with non-numerical features, you have to do the feature selection. Feature selection is the process of selecting attributes that can make the ml model more accurate and eliminate irrelevant attributes. Some of the most common feature selection techniques are listed below.

1. Missing Value Treatment

When we are collecting real-world data, it is possible to have missing values. This leads to including missing values in columns of a dataset. Missing values can lead a machine learning model to identify wrong patterns between data points and give wrong and less accurate predictions. You can easily find whether a dataset has missing values or not by reading the dataset as a Pandas data frame. Pandas library has a lot of inbuilt methods to deal with CSV data easily. Listed below are the few techniques to deal with missing values.

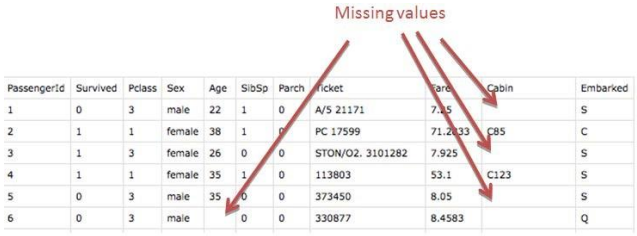

Figure 8: Example Table to show the missing values

1. Deletion

This is the most basic way to handle missing values. Basic idea is to drop any column which has missing values than a specified threshold. However, deleting columns will reduce the number of features for a machine learning model and leads to a less accurate model. In my opinion, it is better to keep data than to delete them.

2. Using back-fill or forward-fill

This method will propagate next or previous values respectively for the missing values. However, the NaN value will remain if the next or previous value is also NaN or not available.

3. Constant value imputation

This method will fill all the missing values with one global constant. Constant value imputation can be used when it doesn't make sense to try and predict missing values. The main drawback is the performance of the ml model can get decreased.

```
data['Cabin'].head(10)
0     NaN
1     C85
2     NaN
3    C123
4     NaN
5     NaN
6     E46
7     NaN
8     NaN
9     NaN
Name: Cabin, dtype: object
```

```
data['Cabin'].fillna('U').head(10)
0       U
1     C85
2       U
3    C123
4       U
5       U
6     E46
7       U
8       U
9       U
Name: Cabin, dtype: object
```

Figure 9: Example showing assignment of constant values

4. Mean, median, and mode imputation

Instead of filling missing values with one single constant, we can fill missing values with the mean or the median of the column. This method treats every variable individually ignoring any relationship with other variables.

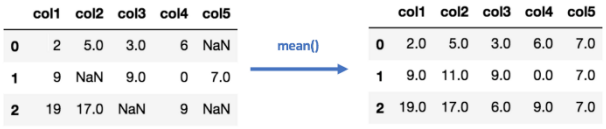

Figure 10: Example showing mean imputation

5. Using a prediction model

Sometimes we use a machine learning model to predict the missing values by identifying relationships among attributes. Considering the above-mentioned techniques, this is a much more effective way to fill missing values. But the accuracy entirely depends on how good your prediction model is.

2. Collinear Features

Collinear features are features that are highly dependent on another feature. Due to high variance and less model interpretability, collinear features lead to decreased generalized performance. The level of correlation is identified based on a specific correlation coefficient. We remove features that have a correlation coefficient greater than a specified threshold. However, we should remove only one feature from the two correlated features, because removing features will reduce the accuracy of the machine learning model. There are two famous coefficients, the Pearson correlation coefficient, and the Spearman correlation coefficient. Both of them can be imported from the SciPy python library. Pearson correlation can be used with continuous variables that have a linear relationship. Spearman correlation can be used with variables that have a non-linear relationship or with ordinal categorical variables. We can easily visualize the correlation between variables using the python seaborn heatmap.

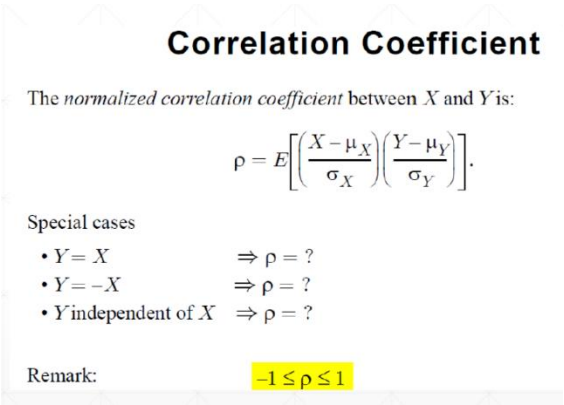

Figure 11: Example of normalization

3. Zero/ Low Important Features

Zero important and low important features don't make any sense for a machine learning model. The importance of a feature can be calculated using a gradient-boosting machine learning model (GBM). You may remove less important features because they are not important for an ml model to make predictions. However, this method is only applicable for supervised models which have non-deterministic labeled data and if we are going to use tree-based methods to make predictions. But more importantly, you should try to derive new meaningful features out of these features. There can be situations where it is possible to derive more meaningful features even though the feature itself is less important.

4. Single Unique Value Features

Here we remove columns that have a single unique value throughout the entire dataset because a machine learning model can't draw any conclusion out of these columns.

III. DATA VISUALIZATIONS

Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

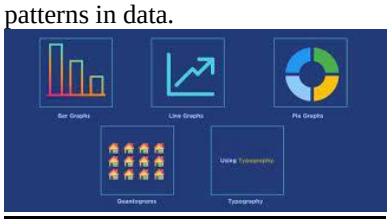
Figure 12: Types of data visualization

In the world of Big Data, data visualization tools and technologies are essential to analyze massive amounts of information and make data-driven decisions.

Our eyes are drawn to colors and patterns. We can quickly identify red from blue, and square from a circle. Our culture is visual, including everything from art and advertisements to TV and movies. Data visualization is another form of visual art that grabs our interest and keeps our eyes on the message. When we see a chart, we quickly see trends and outliers. If we can see something, we internalize it quickly. It's storytelling with a purpose. If you've ever stared at a massive spreadsheet of data and couldn't see a trend, you know how much more effective a visualization can be.

Types of data visualization:

Dashboards include common visualization techniques, such as:

- **Tables**: This consists of rows and columns used to compare variables. Tables can show a great deal of information in a structured way, but they can also overwhelm users that are simply looking for high-level trends.
- **Pie charts and stacked bar charts:** These graphs are divided into sections that represent parts of a whole. They provide a simple way to organize data and compare the size of each component to one other.
- **Line graphs and area charts:** These visuals show the change in one or more quantities by plotting a series of data points over time. Line graphs utilize lines to demonstrate these changes while area charts connect data points with line segments, stacking variables on top of one another and using color to distinguish between variables.
- **Histograms**: This graph plots a distribution of numbers using a bar chart (with no spaces between the bars), representing the quantity of data that falls within a particular range. This visual makes it easy for an end user to identify outliers within a given dataset.
- **Scatter plots**: These visuals are beneficial in revealing the relationship between two variables, and they are commonly used within regression data analysis. However, these can sometimes be confused with bubble charts, which are used to visualize three variables via the x-axis, the y-axis, and the size of the bubble.
- **Heat maps:** These graphical displays help visualize behavioral data by location. This can be a location on a map or even a webpage.
- **Treemaps**, which display hierarchical data as a set of nested shapes, typically rectangles. Tree

The benefits of data visualization

When considering business strategies and goals, data visualization benefits decision-makers in several ways to improve data insights. Let's explore seven major benefits in detail:

- Better analysis
- Quick action
- Identifying patterns
- Finding errors
- Understanding the story
- Exploring business insights
- Grasping the Latest Trends

Better analysis

Data visualization helps business stakeholders analyze reports regarding sales, marketing strategies, and product interest. Based on the analysis, they can focus on the areas that require attention to increase profits, which in turn makes the business more productive.

Quick action

As mentioned previously, the human brain grasps visuals more easily than table reports. Data visualizations allow decision-makers to be notified quickly of new data insights and take necessary actions for business growth.

Identifying patterns

Large amounts of complicated data can provide many opportunities for insights when we visualize them.

Visualization allows business users to recognize relationships between the data, providing greater meaning to it. Exploring these patterns helps users focus on specific areas that require attention in the data so that they can identify the significance of those areas to drive their business forward.

Finding errors

Visualizing your data helps quickly identify any errors in the data. If the data tends to suggest the wrong actions, visualizations help identify erroneous data sooner so that it can be removed from the analysis.

Understanding the story

Storytelling is the purpose of your dashboard. By designing your visuals in a meaningful way, you help the target audience grasp the story in a single glance. Always be sure to convey the story most simply, without excessively complicated visuals.

Exploring business insights

In the current competitive business environment, finding data correlations using visual representations is key to identifying business insights. Exploring these insights is important for business users or executives to set the right path to achieving the business goals.

Grasping the latest trends

Using data visualization, you can discover the latest trends in your business to provide quality products and identify problems before they arise. By staying on top of trends, you can put more effort into increased profits for your business.

IV. HOW DATA VISUALIZATION WORKS

Figure 13: A visualized form of a certain type of dataset

Data Visualization is one of the major pillars of Data Science. It uses shapes, lines, and other figurative approaches to portray numbers. Multiple techniques can be used to present data based on what kind of data is available. This article shall focus on five of them which are: bar chart, pie chart, scatter plot, histogram, and line chart.

Bar Chart

A bar Chart is an aspect of data visualization that presents categorical data's numerical values as bars to compare data points with one another. It does this by increasing or decreasing the size of the bar based on how big or small the numbers are. There are two types of a bar chart: horizontal bar chart and vertical bar chart.

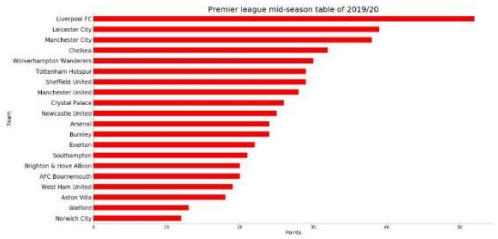

Figure 14: Example of the bar chart

Pie Chart

A pie chart is a chart that compares the magnitude of how different data point contributes to a whole. Each data point has a percentage that can be represented like a slice of pie that adds up to a full pie. That's why it is called a pie chart. The bigger the percentage, the bigger the slice and vice versa. All the percentages add up to 100%.

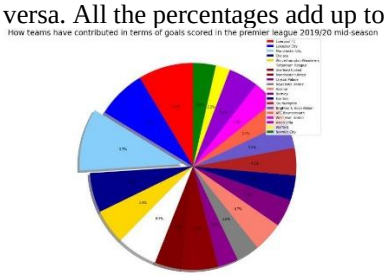

Figure 15: Example of Pie chart

Scatter Plot

A scatter plot is a plot that compares two numerical sets of values of two variables with one another. It places the numbers of one of the variables on the horizontal axis (x-axis) and the other on the vertical axis (y-axis). Then, it plots data points as symbols where the numerical values of the data point intersect on the graphs.

After plotting these symbols, further calculations can be carried out to measure the strength of the relationships between the two variables. This is done using the correlation coefficient. The correlation coefficient ranges from -1 to +1. A negative value signifies that they have an inverse relationship i.e as x increases, y decreases, and vice versa. A positive value signifies that they have a direct relationship i.e as x increases, y increases. The closer the correlation coefficient is to |1|, the more perfect it is.

Another useful ability of the scatter plot is to create a regression line.

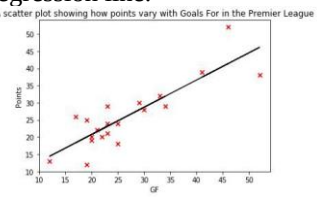

Figure 16: Example showcasing Scatter Plot

A scatter plot showing how points accumulated vary with goals scored by premier league teams by mid-season 2019/20

Histogram

A histogram is a graphing mechanism used to count the number of times a numerical value appears in a given data set. It does this by grouping the numerical values into different groups within a fixed width of certain numbers. It counts the number of times the numerical values in each

group appears and utilizes a bar to present its frequency. The more frequent the numerical values in a group the bigger the bar.

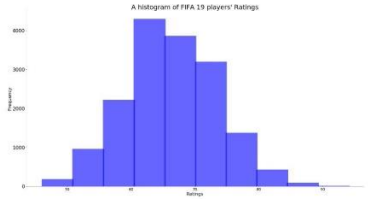

Figure 17: Histogram

Line Chart

A line chart is a chart that is mostly used to show how a certain variable has changed over some time. It does this by plotting time on the x-axis and plotting the variable on the y-axis. It connects preceding data points with the following data points using lines.

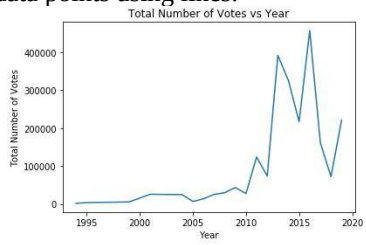

Figure 18: Graph showing Line Chart

V. CONCLUSION

As we conclude our brief study on data visualization, it is clear that the field is rich in potential applications in diverse disciplines, at the same time we need to be aware of its practical and ethical complexities.

The power of data visualization is evident in our modern world. Politicians, sports pundits, journalists, engineers, and accountants use data visualization techniques to convey comprehensive important messages to people with a better idea of what is going on.

Considering the world of the Internet of Things, Network and Complexity Theories, and recent developments in multidimensional visualization, we can see how data visualization has evolved along with the field to a better approach.

Data Visualization with Python

Data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends, and correlations that might not otherwise be detected can be exposed.

Python offers multiple great graphing libraries that come packed with lots of different features. No matter, if you want to create interactive, live, or highly customized plots python, has an excellent library for you.

To get a little overview here are a few popular plotting libraries:

- **Matplotlib:** low level, provides lots of freedom
- **Pandas Visualization:** easy-to-use interface, built on Matplotlib
- **Seaborn:** high-level interface, great default styles
- **ggplot:** based on R's ggplot2, uses Grammar of Graphics
- **Plotly:** can create an interactive plot

1. Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

- Create publication quality.
- Make interactive figures that can zoom, pan, and update.
- Customize visual style and layout.
- Export to many file formats.
- Embed in JupyterLab and Graphical User Interfaces.
- Use a rich array of third-party packages built on Matplotlib.

To install Matplotlib pip and conda can be used.

```
pip install matplotlib
```

or

```
conda install matplotlib
```

Matplotlib is specifically good for creating basic graphs like line charts, bar charts, histograms, and many more. It can be imported by typing:

```
import matplotlib.pyplot as plt
```

2. Pandas Visualization

Pandas is an open-source high-performance, easy-to-use library providing data structures, such as data frames, and data analysis tools like visualization tools.

Pandas Visualization makes it easy to create plots out of a Pandas data frame and series. It also has a higher-level API than Matplotlib and therefore we need less code for the same results.

Pandas is arguably the most popular data analysis and manipulation library. It makes it extremely easy to manipulate data in tabular form. The various functions of Pandas constitute a powerful and versatile data analysis tool. Pandas is not a data visualization library but it is capable of creating basic plots. If you are just creating plots for exploratory data analysis, Pandas might be highly useful and practical. You do not have to use an additional data visualization library for such tasks.

Pandas can be installed using either pip or conda.

```
pip install pandas
```

or

```
conda install pandas
```

3. Seaborn

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

Install Seaborn.

If you have Python and PIP already installed on a system, install them using this command:

```
C:\Users\Your Name>pip install seaborn
```

If you use Jupyter, install Seaborn using this command:

```
C:\Users\Your Name>! pip install seaborn
```

LITERATURE REVIEW

S. No.	Paper Title	Authors	Focused Area
1.	The emergence of social media data and sentiment	Priyavrat Chauhan, Nonita Sharma, Geeta Sikka	Predicting election results and open issues related to

	analysis in election prediction.		sentiment analysis
2.	Feature Engineering meets Deep Learning: A Case Study on Table Detection in Documents	Muhammad Ali Shahzad, Rabeya Noor, Sheraz Ahmad, Ajmal Mian, and Faisal Shafait	An approach to leverage deep learning algorithms to extract features with feature learning algorithms to increase the accuracy and efficiency of the model.
3.	Data Analysis using Python	Kiran Bala Nongthombam, Deepika Sharma	The very basic processes of data analysis like cleaning, transforming, and modeling of data are briefly explained in this paper and focus more on exploratory data analysis of an already existing dataset and finding insights.
4.	Visual Data Mining in Indian Election System	Prof. T. M. Kodinariya, Mr. Ravi Seta	The approach is divided into 5 phases: I) Data Pre-processing: II) Data Warehouse Creation: III) Task-Relevant Data Extraction; IV) Data Mining and V) Visualization.
5.	Visualization of Election Data: Using Interaction Design and Visual Discovery for Communicating Complex Insights	Kuhu Gupta, Shailaja Sampat, Manas Sharma, Venkatesh Rajamanickam	In this paper, using the data from the Assembly elections that took place in the state of Tamil Nadu in May 2016, we present a process and a set of interaction design and visualization methods to present complex insights.
6.	Visualizing E-Voting Results	O. Folorunso, O. S Ogunseye, J.O. Okesola2 And O.M Olaniyan	The paper proved that TreeMap algorithms can be configured and deployed on the central server to monitor effectively the voting transactions in real-time and hence enable transparency.
7.	Deep Learning and Visualization of Election Data	Garcia, Jorge A., Tao, Ng Ching, Betancourt, Frank, Wong, Kwai	This paper explores the feasibility of using a Deep Neural Network to predict bipartisan elections in the US and Hong Kong. The use of political opinion surveys allows for the training and testing of machine learning models while predicting with census data.
8.	LokDhaba: Acquiring, Visualizing, and Disseminating Data on Indian Elections	Mohit Kumar, Chinmay Narayan, Sudheendra Hangal, Priyamvada Trivedi	In this paper, the main focus was on data scraping, parsing, cleaning, consistency checking, and integration between multiple sources, with the help of some novel tools.
9.	Election Results Prediction System based on Fuzzy Logic	Harmanjit Singh, Gurdev Singh, Nitin Bhatia	The use of fuzzy logic in social science to evaluate the prediction is the core part of this paper. A toolbox from MATLAB software named fuzzy logic toolbox is used for this purpose
10.	Value-based prediction of election results using natural language processing: A case of the New Zealand General Election	Mathew Parackal, Damien Mather, David Holdsworth	The study concluded that the value-based prediction shows promise for improving the quality of political journalism and public engagement in the period of election campaigns, and will assist greatly in focusing public debate more on values that are influential on citizens' voting decisions.

SUMMARY AND PERSPECTIVES

With arising of large datasets their handling methods and technologies go hand in hand. Big data is a combination of structured, semi-structured, and unstructured data collected by organizations that can be mined for information and used in machine learning projects, modeling, and other advanced analytics applications. Big data is often characterized by the three V's:

- the large *volume* of data;
- the wide *variety* of data types frequently stored in big data systems;
- the *velocity* at which much of the data is generated, collected, and processed.

But despite being so powerful its results can be misleading. It is some part inefficient to deal and manipulate with missing or unmeaningful data which can lead to loss of integrity of data. Here comes Feature Engineering in the picture which is one of the beautiful arts which helps you to represent data in the most insightful possible way. It entails a skilled combination of subject knowledge, intuition, and fundamental mathematical skills which will effectively transform the data properties into data features. How you provide your data to your algorithm should effectively indicate the relevant structures/properties of the underlying information. It's a process of pre-processing data so that your model/learning algorithm may spend as little time as possible sifting through the noise. Any information that is unrelated to learning or forecasting concerning your final aim is known as noise. Feature Engineering can create a much better meaningful dataset to predict the results which further can be used to convert it into a visualization using Python Libraries.

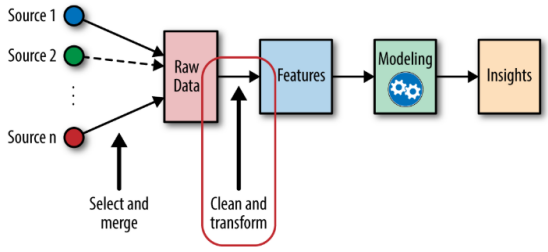

Figure 19: Process to be followed by Feature Engineering Algorithm

Feature Engineering is much more efficient in encapsulating various data engineering techniques such as selecting relevant features, handling missing data, encoding the data, and normalizing it. This is one of the most crucial tasks and plays a major role in determining the outcome of a model. To ensure that the chosen algorithm can perform to its optimum capability, it is important to engineer the features of the input data effectively.

An effective Feature Engineering implies:

- Higher efficiency of the model
- Easier Algorithms that fit the data
- Easier for Algorithms to detect patterns in the data
- Greater Flexibility of the features

Feature engineering is the development of new data features from raw data. With this technique, engineers analyze the raw data and potential information to extract a new or more valuable set of features. Feature engineering can be seen as a generalization of mathematical optimization that allows for better analysis.

This review has led to a study on a dataset consisting of data from Punjab Assembly Elections 2017, the main aim was to make result predictions beforehand and visualize them.

Here the main objectives are

- Dealing with missing values: To achieve better accuracy first the missing values were manipulated using some of the methods of feature engineering (either removing the values if the percentage of missing values is less than 10% or by applying certain methods of feature engineering to fill them with appropriate values.
- Dealing with noises: Any kind of inappropriate data and unmatched data from the other existing data in a column or dataset is known as noise. Thus, this needs to be manipulated for better accuracy, to do that, certain methods from feature engineering are used to replace the noise values with appropriate values (either replaced by constant values or mean, median, mode values, etc.)
- Achieving better accuracy while making predictions: After a proper approach to manipulating the dataset, a dataset with reformed data will be achieved, and then manipulating methods from the Pandas library will be used to make predictions and test the level of accuracy.
- Maintain Integrity of data: While deleting or replacing the values feature engineering algorithm ensures that the integrity of data retains and that it doesn't lose any important information that might mislead afterward.

REFERENCES

[1] Kuhu Gupta, Shailaja Sampat, Manas Sharma, Venkatesh Rajamanickam. 2016. Visualization of Election Data: Using Interaction Design and Visual Discovery for Communicating Complex Insights

[2] Thamindu Dilshan Jayawickrama.2019: Basic Feature Engineering to Reach More Efficient Machine Learning. (https://towardsdatascience.com/basic-feature-engineering-to-reach-more-efficient-machine-learning)

[3] "Diagnosis of Attention Deficit Hyperactivity Disorder (ADHD) Using CNN" by Karuna

[4] https://www.kaggle.com/learn/feature-engineering (for datasets to work on)

[5] https://eci.gov.in/statistical-report/statistical-reports (for datasets to work on)

[6] An Empirical Analysis of Feature Engineering for Predictive Modeling.

[7] Feature Engineering (FE) Tools and Techniques for Better Classification Performance

[8] Banerjee, M. (2014). Why India Votes (Exploring the Political in South Asia). New Delhi: Routledge Publications.

[9] Bouvier, J. (1856). Electoral democracy. A Law Dictionary, Adapted to the Constitution and Laws of the United States. Retrieved July 1, 2016, from http://legaldictionary.thefreedictionary.com/Electoral+ democracy

[10] On predicting elections with hybrid topic-based sentiment analysis of tweets by Barkha Bansal and Sangeeta Shrivastava

[11] A Reader on Data Visualization: *MSIS 2629 Spring 2019*