Visualization of Election Data Using Feature Engineering for Result Prediction

Pratibha Chadha[1], **Sukhwinder Multani**[2]

[1]Department of Computer Science, C. T. University, India
pratibhachadha3[at]gmail.com

[2]Department of Computer Science, C. T University, India

Abstract: ***State election results in India yield a variegated and unique set of insights. Often it is hard to grasp the nuances of local electoral dynamics, and harder to communicate it to an audience not deeply engaged in the local political process, or to an audience from outside the state. While national news outlets do a fairly good job of analyzing and communicating the federal elections, the state elections in contrast have largely remained devoid of rigorous analysis and insightful communication. In this paper, using the data from the Assembly elections that took place in the state of Punjab in 2017, we present a process based on Feature Engineering and a set of interaction design and visualization methods to present complex insights and predictions. The general principles thus derived, will not only aid analysts and journalists to present their insights more effectively, but also empower the readers, depending on their level of interest and civic engagement, to go beyond what is presented and to discover new insights for themselves. Here the main focus is on how to clean and prepare the dataset, create new features out of the existing ones and then predict the results using a popular machine learning algorithm that may lead to more efficiency in prediction.***

Keywords: Feature Engineering, Data Visualization, Assembly elections, Result Predictions, Social Media, Journalism

1. Introduction

In today's era digital media has become one whole source of communication in itself, in every aspect. It is a blend of technology and content and plays a major role in the world of analytics and production. In the world of journalism, digital media is a weapon that will do wonders.

Digital media can incorporate interactions, movements, transitions, animations, sound effects, and dense data, along with the storyline. It helps to communicate data - driven insights very effectively in the knowledge discovery process. There are multiple windows of opportunity to derive inferences or make predictions from the unstructured data and to better present them for improved decision - making.

As humans, it is much easier to explore, compare things, identify patterns and relate other similar kinds of things encountered previously rather than analyzing any table or matrix format.

This paper holds the study on a dataset consisting of data from the Punjab Assembly Election 2017, the main aim is to predict the results using feature engineering algorithms and to put it in a better - visualized form. This requirement is in the interest of all those who are interested in politics and like to follow all the predictions and facts made in general or are part of journalism. Feature Engineering in specific is used to create the process leading to automation, a process to clear and maintain the data in all possible manners reaching better accuracy where it comes to data visualization, we have certain libraries that put the ready data frames into graphical representation or into catchy images that gather one's interest and increase the readability.

Feature Engineering

Feature engineering is the process of selecting, manipulating, and transforming raw data into features that can be used in supervised learning. To make machine learning work well on new tasks, it might be necessary to design and train better features. As you may know, a "feature" is any measurable input that can be used in a predictive model — it could be the color of an object or the sound of someone's voice. Feature engineering, in simple terms, is the act of converting raw observations into desired features using statistical or machine learning approaches. Feature engineering is a machine learning technique that leverages data to create new variables that aren't in the training set. It can produce new features for both supervised and unsupervised learning, to simplify and speed up data transformations while also enhancing model accuracy. Feature engineering is required when working with machine learning models. Regardless of the data or architecture, a terrible feature will have a direct impact on your model.
Feature engineering consists of various processes –

- **Feature Creation**: Creating features involves creating new variables which will be most helpful for our model. This can be adding or removing some features. As we saw above, the cost per sq. ft column was a feature creation.
- **Transformations**: Feature transformation is simply a function that transforms features from one representation to another. The goal here is to plot and visualize data, if something is not adding up with the new features, we can reduce the number of features used, speed up training, or increase the accuracy of a certain model.
- **Feature Extraction**: Feature extraction is the process of extracting features from a data set to identify useful information. Without distorting the original relationships or significant information, this compresses the amount of data into manageable quantities for algorithms to process.

- **Exploratory Data Analysis:** Exploratory data analysis (EDA) is a powerful and simple tool that can be used to improve your understanding of your data, by exploring its properties. The technique is often applied when the goal is to create new hypotheses or find patterns in the data. It's often used on large amounts of qualitative or quantitative data that haven't been analyzed before.
- **Benchmark:** A Benchmark Model is the most user - friendly, dependable, transparent, and interpretable model against which you can measure your own. It's a good idea to run test datasets to see if your new machine learning model outperforms a recognized benchmark. These benchmarks are often used as measures for comparing the performance between different machine learning models like neural networks and support vector machines, linear and non - linear classifiers, or different approaches like bagging and boosting. To learn more about feature engineering steps and processes, check the links provided at the end of this article. Now, let's have a look at why we need feature engineering in machine learning.

Feature Engineering Techniques for Machine Learning

Let's see a few feature engineering best techniques that you can use. Some of the techniques listed may work better with certain algorithms or datasets, while others may be useful in all situations.

1) Imputation

When it comes to preparing your data for machine learning, missing values are one of the most typical issues. Human errors, data flow interruptions, privacy concerns, and other factors could all contribute to missing values. Missing values have an impact on the performance of machine learning models for whatever cause. The main goal of imputation is to handle these missing values. There are two types of imputation:

- **Numerical Imputation**: To figure out what numbers should be assigned to people currently in the population, we usually use data from completed surveys or censuses. These data sets can include information about how many people eat different types of food, whether they live in a city or country with a cold climate, and how much they earn every year. That is why numerical imputation is used to fill gaps in surveys or censuses when certain pieces of information are missing.
  #Filling all missing values with 0
  data = data. fillna (0)
- **Categorical Imputation:** When dealing with categorical columns, replacing missing values with the highest value in the column is a smart solution. However, if you believe the values in the column are evenly distributed and there is no dominating value, imputing a category like "Other" would be a better choice, as your imputation is more likely to converge to a random selection in this scenario.
  #Max fill function for categorical columns
  data ['column_name']. fillna (data ['column_name']. value_counts (). idxmax (), inplace=True)

2) Handling Outliers

Outlier handling is a technique for removing outliers from a dataset. This method can be used on a variety of scales to produce a more accurate data representation. This has an impact on the model's performance. Depending on the model, the effect could be large or minimal; for example, linear regression is particularly susceptible to outliers. This procedure should be completed before model training. The various methods of handling outliers include:

- **Removal**: Outlier - containing entries are deleted from the distribution. However, if there are outliers across numerous variables, this strategy may result in a big chunk of the datasheet being missed.
- **Replacing values**: Alternatively, the outliers could be handled as missing values and replaced with suitable imputations.
- **Capping**: Using an arbitrary value or a value from a variable distribution to replace the maximum and minimum values.
- **Discretization:** Discretization is the process of converting continuous variables, models, and functions into discrete ones. This is accomplished by constructing a series of continuous intervals (or bins) that span the range of our desired variable/model/function.

3) Log Transform

Log Transform is the most used technique among data scientists. It's mostly used to turn a skewed distribution into a normal or less - skewed distribution. We take the log of the values in a column and utilize those values as the column in this transform. It is used to handle confusing data, and the data becomes more approximate to normal applications.
//Log Example
df [log_price] = np. log (df ['Price'])

4) One - hot encoding

A one - hot encoding is a type of encoding in which an element of a finite set is represented by the index in that set, where only one element has its index set to "1" and all other elements are assigned indices within the range [0, n - 1]. In contrast to binary encoding schemes, where each bit can represent 2 values (i. e.0 and 1), this scheme assigns a unique value for each possible case.

5) Scaling

Feature scaling is one of the most pervasive and difficult problems in machine learning, yet it's one of the most important things to get right. To train a predictive model, we need data with a known set of features that needs to be scaled up or down as appropriate. This blog post will explain how feature scaling works and why it's important as well as some tips for getting started with feature scaling.

After a scaling operation, the continuous features become similar in terms of range. Although this step isn't required for many algorithms, it's still a good idea to do so. Distance - based algorithms like k - NN and k - Means, on the other hand, require scaled continuous features as model input. There are two common ways of scaling:

- **Normalization**: All values are scaled in a specified range between 0 and 1 via normalization (or min - max normalization). This modification does not influence the feature's distribution; however, it does exacerbate the effects of outliers due to lower standard deviations. As a result, it is advised that outliers be dealt with before normalization.

- **Standardization**: Standardization (also known as z - score normalization) is the process of scaling values while accounting for standard deviation. If the standard deviation of features differs, the range of those features will likewise differ. The effect of outliers in the characteristics is reduced as a result. To arrive at a distribution with a 0 mean and 1 variance, all the data points are subtracted by their mean and the result is divided by the distribution's variance.

Data Visualization
Data visualization is the practice of translating information into a visual context, such as a map or graph, to make data easier for the human brain to understand and pull insights from. The main goal of data visualization is to make it easier to identify patterns, trends, and outliers in large data sets. The term is often used interchangeably with others, including information graphics, information visualization, and statistical graphics.
Data visualization is one of the steps of the data science process, which states that after data has been collected, processed, and modeled, it must be visualized for conclusions to be made. Data visualization is also an element of the broader data presentation architecture (DPA) discipline, which aims to identify, locate, manipulate, format, and deliver data in the most efficient way possible.

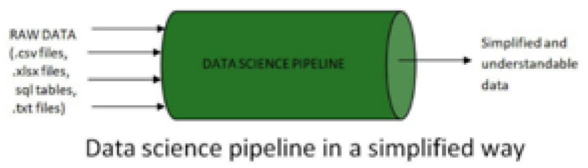


Figure 1: Representation of Data Visualization

Data visualization is a set of data points and information that are represented graphically to make it easy and quick for the users to understand. Data visualization is good if it has a clear meaning, and purpose, and is very easy to interpret, without requiring context. Tools of data visualization provide an accessible way to see and understand trends, outliers, and patterns in data by using visual effects or elements such as a chart, graphs, and maps.

Characteristics of Effective Graphical Visual:
- It shows or visualizes data very clearly in an understandable manner.
- It encourages viewers to compare different pieces of data.
- It closely integrates statistical and verbal descriptions of the data set.
- It grabs our interest, focuses our mind, and keeps our eyes on messages as the human brain tends to focus on visual data more than written data.
- It also helps in identifying the area that needs more attention and improvement.
- Using graphical representation, a story can be told more efficiently. Also, it requires less time to understand pictures than it takes to understand textual data.

Categories of Data Visualization;
Data visualization is very critical to market research where both numerical and categorical data can be visualized which helps in increasing in impacts of insights and also helps in reducing the risk of analysis paralysis. So, data visualization is categorized into the following categories:

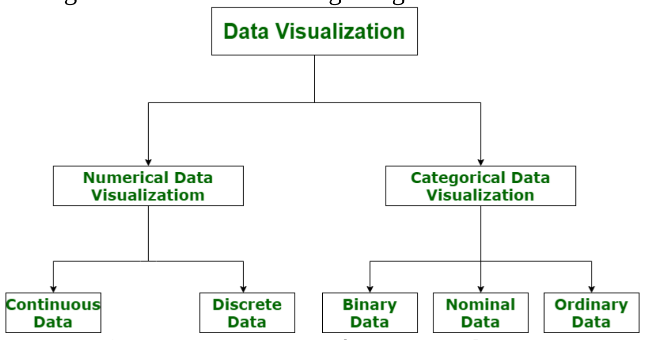


Figure 2: Categories of Data Visualization

1) Numerical Data:
Numerical data is also known as Quantitative data. Numerical data is any data where data generally represents an amount such as height, weight, age of a person, etc. Numerical data visualization is the easiest way to visualize data. It is generally used for helping others to digest large data sets and raw numbers in a way that makes them easier to interpret into action. Numerical data is categorized into two categories:
- **Continuous Data –**It can be narrowed or categorized (Example: Height measurements).
- **Discrete Data –**This type of data is not "continuous" (Example: Number of cars or children a household has).

The type of visualization techniques that are used to represent numerical data visualization is Charts and Numerical Values. Examples are Pie Charts, Bar Charts, Averages, Scorecards, etc.

2) Categorical Data
Categorical data is also known as Qualitative data. Categorical data is any data where data generally represents groups. It simply consists of categorical variables that are used to represent characteristics such as a person's ranking, a person's gender, etc. Categorical data visualization is all about depicting key themes, establishing connections, and lending context. Categorical data is classified into three categories:
- **Binary Data –**In this, classification is based on positioning (Example: Agrees or Disagrees).
- **Nominal Data –**In this, classification is based on attributes (Example: Male or Female).
- **Ordinal Data –**In this, classification is based on the ordering of information (Example: Timeline or processes).

The type of visualization techniques that are used to represent categorical data is Graphics, Diagrams, and Flowcharts. Examples are Word clouds, Sentiment Mapping, Venn diagrams, etc.

2. Problem Statement

The domain of journalism is dedicated to collecting facts and thoughts leading to certain predictions. These opinions are purely unbiased, based on observations or interviews conducted which are to be concisely written into print space.

In the digital era, new media and communication technologies have influenced the journalism techniques being used. The data has taken a much more complicated face. Thus, to catch the viewer's interest and to make this result prediction process more attractive data visualization is fixing its place.

The main problem in converting the data into a visualized form is to maintain its integrity and make out a much more accurate result.

The objective of this paper is to focus on 4 main aspects to maintain data integrity, thus the objectives are:

- Replacing missing values with valid methods for more accuracy.
- Deals with most of the features in a numeric format that create results easy to understand
- Transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data
- Using domain knowledge to extract features from raw data to improve the quality of results from a machine learning process.

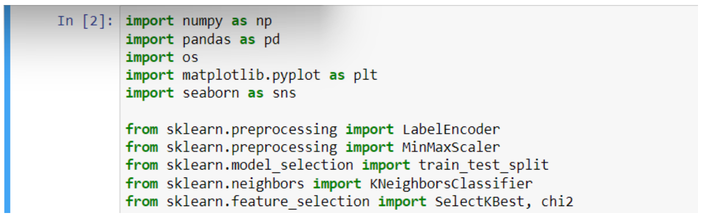
3. Research Methodology

To carry out the respective research, a sample dataset of 2263 records was taken from www.kaggle. come, the Election Commission's official website. The dataset consisted of data from Punjab Assembly Elections, 2017, which had many outliers to work upon.

```
dataset.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2263 entries, 0 to 2262
Data columns (total 19 columns):
STATE                                      2263 non-null object
CONSTITUENCY                               2263 non-null object
NAME                                       2263 non-null object
WINNER                                     2263 non-null int64
PARTY                                      2263 non-null object
SYMBOL                                     2018 non-null object
GENDER                                     2018 non-null object
CRIMINAL
CASES                             2018 non-null object
AGE                                        2018 non-null float64
CATEGORY                                   2018 non-null object
EDUCATION                                  2018 non-null object
ASSETS                                     2018 non-null object
LIABILITIES                                2018 non-null object
GENERAL
VOTES                              2263 non-null int64
POSTAL
VOTES                               2263 non-null int64
TOTAL
VOTES                                2263 non-null int64
OVER TOTAL ELECTORS
IN CONSTITUENCY          2263 non-null float64
OVER TOTAL VOTES POLLED
IN CONSTITUENCY      2263 non-null float64
TOTAL ELECTORS                             2263 non-null int64
dtypes: float64(3), int64(5), object(11)
memory usage: 336.0+ KB
```

Figure 3: Sample dataset profile

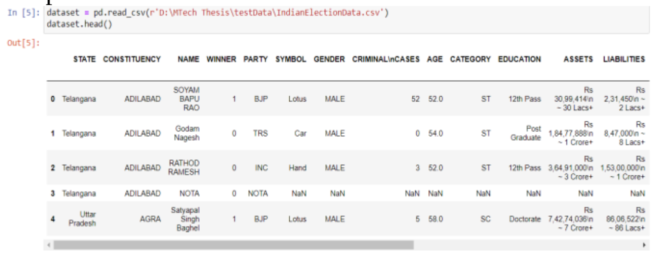
Further, the technologies for data visualization used are Pandas, NumPy, Matplotlib, and sklearn.



```
In [2]: import numpy as np
        import pandas as pd
        import os
        import matplotlib.pyplot as plt
        import seaborn as sns

        from sklearn.preprocessing import LabelEncoder
        from sklearn.preprocessing import MinMaxScaler
        from sklearn.model_selection import train_test_split
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.feature_selection import SelectKBest, chi2
```

Figure 4: The figure shows all the libraries being used for research

Sample Dataset



```
In [5]: dataset = pd.read_csv(r'D:\MTech Thesis\testData\IndianElectionData.csv')
        dataset.head()

Out[5]:
```

	STATE	CONSTITUENCY	NAME	WINNER	PARTY	SYMBOL	GENDER	CRIMINAL\nCASES	AGE	CATEGORY	EDUCATION	ASSETS	LIABILITIES
0	Telangana	ADILABAD	SOYAM BAPU RAO	1	BJP	Lotus	MALE	52	52.0	ST	12th Pass	Rs 30,99,414\n ~ 30 Lacs+	Rs 2,31,450\n ~ 2 Lacs+
1	Telangana	ADILABAD	Godam Nagesh	0	TRS	Car	MALE	0	54.0	ST	Post Graduate	Rs 1,84,77,888\n ~ 1 Crore+	Rs 8,47,000\n ~ 8 Lacs+
2	Telangana	ADILABAD	RATHOD RAMESH	0	INC	Hand	MALE	3	52.0	ST	12th Pass	Rs 3,64,91,000\n ~ 3 Crore+	Rs 1,53,00,000\n ~ 1 Crore+
3	Telangana	ADILABAD	NOTA	0	NOTA	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	Uttar Pradesh	AGRA	Satyapal Singh Baghel	1	BJP	Lotus	MALE	5	58.0	SC	Doctorate	Rs 7,42,74,036\n ~ 7 Crore+	Rs 86,06,522\n ~ 86 Lacs+

Figure 5: the figure gives a glimpse of a small part of the sample dataset to understand how the available data looks.

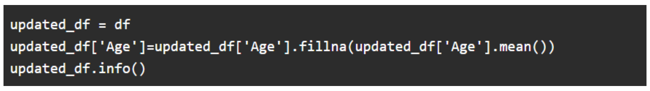
Objective 1: Searching for Missing data values in each column

```
dataset.isna().sum()

STATE                                       0
CONSTITUENCY                                0
NAME                                        0
WINNER                                      0
PARTY                                       0
SYMBOL                                    245
GENDER                                    245
CRIMINAL_CASES                            245
AGE                                       245
CATEGORY                                  245
EDUCATION                                 245
ASSETS                                    245
LIABILITIES                               245
GENERAL_VOTES                               0
POSTAL_VOTES                                0
TOTAL_VOTES                                 0
OVER_TOTAL_ELECTORS_IN_CONSTITUENCY         0
OVER_TOTAL_VOTES_POLLED_IN_CONSTITUENCY     0
TOTAL_ELECTORS                              0
dtype: int64
```

Figure 6: The figure shows the missing values

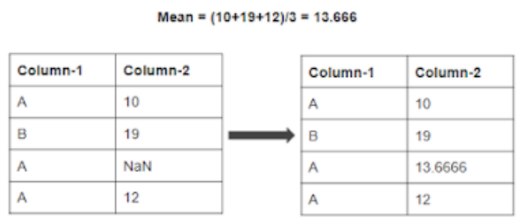
Methodologies to deal with missing values.

- There are multiple ways to treat missing values like deleting them, using back - fill or forward - fill, constant value imputation, mean/ median or mode imputation, etc.
- Since approximately only 10% of values were missing so we preferred to delete them directly for simplicity.
- Filling the missing values – Imputations – Although the filling could be done by any of the above - mentioned methods here, I am explaining only one of these i. e. filling with the mean value



```
updated_df = df
updated_df['Age']=updated_df['Age'].fillna(updated_df['Age'].mean())
updated_df.info()
```

Figure 7: The image represents the filling of missing values in the dataset with the mean value.



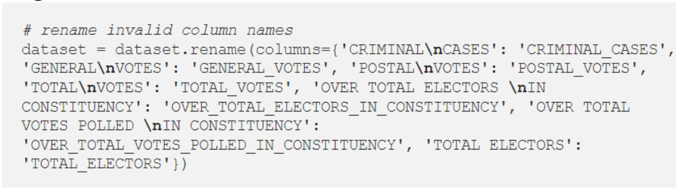
Mean = (10+19+12)/3 = 13.666

Column-1	Column-2
A	10
B	19
A	NaN
A	12

Column-1	Column-2
A	10
B	19
A	13.6666
A	12

Figure 8: Physical representation of filling of missing values with the mean value

Log Transform:

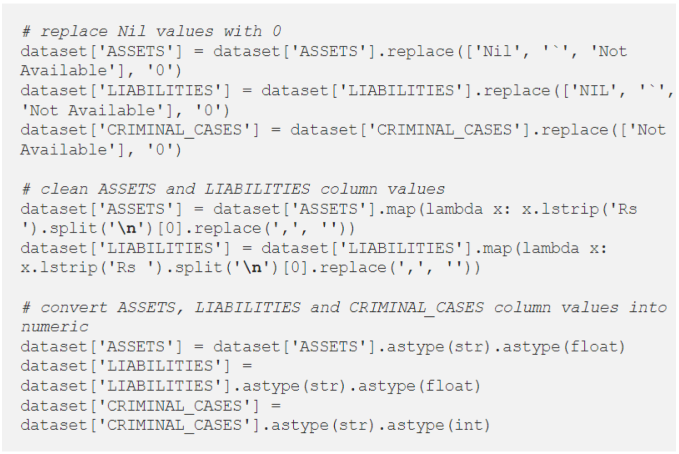


```
# rename invalid column names
dataset = dataset.rename(columns={'CRIMINAL\nCASES': 'CRIMINAL_CASES',
'GENERAL\nVOTES': 'GENERAL_VOTES', 'POSTAL\nVOTES': 'POSTAL_VOTES',
'TOTAL\nVOTES': 'TOTAL_VOTES', 'OVER TOTAL ELECTORS \nIN
CONSTITUENCY': 'OVER_TOTAL_ELECTORS_IN_CONSTITUENCY', 'OVER TOTAL
VOTES POLLED \nIN CONSTITUENCY':
'OVER_TOTAL_VOTES_POLLED_IN_CONSTITUENCY', 'TOTAL ELECTORS':
'TOTAL_ELECTORS'})
```

Figure 9: Renaming columns of the dataset

Objective 2: Transforming non - numeric data into numeric form

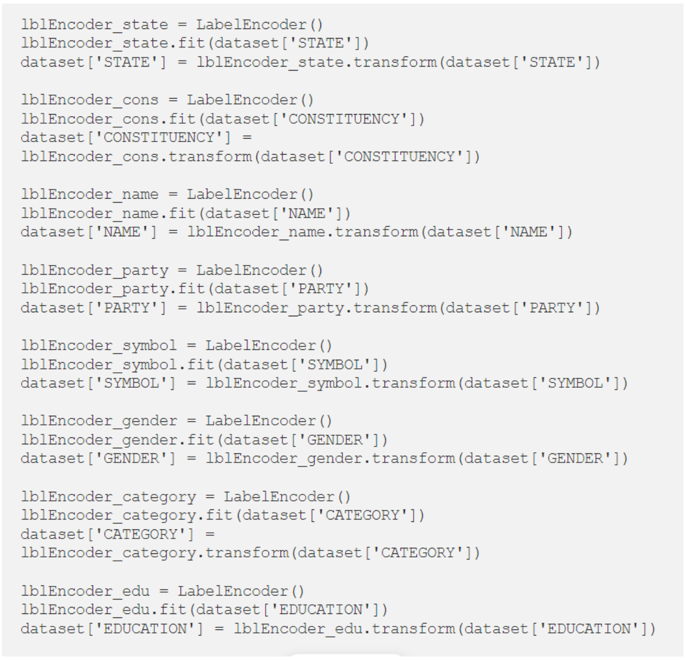
To implement this in the sample space we had ASSETS, LIABILITIES, and CRIMINAL_CASES columns which will be converted to numeric form because they represent money and count, and numeric will make sense to the models. To do that, we have to remove the 'Rs' sign, the '\n' character, and commas in each value field. But also, those columns contain 'Nil' and 'Not Available' as values. So before making them numeric, we have to replace those values with some meaningful values.



```
# replace Nil values with 0
dataset['ASSETS'] = dataset['ASSETS'].replace(['Nil', '`', 'Not
Available'], '0')
dataset['LIABILITIES'] = dataset['LIABILITIES'].replace(['NIL', '`',
'Not Available'], '0')
dataset['CRIMINAL_CASES'] = dataset['CRIMINAL_CASES'].replace(['Not
Available'], '0')

# clean ASSETS and LIABILITIES column values
dataset['ASSETS'] = dataset['ASSETS'].map(lambda x: x.lstrip('Rs
').split('\n')[0].replace(',', ''))
dataset['LIABILITIES'] = dataset['LIABILITIES'].map(lambda x:
x.lstrip('Rs ').split('\n')[0].replace(',', ''))

# convert ASSETS, LIABILITIES and CRIMINAL_CASES column values into
numeric
dataset['ASSETS'] = dataset['ASSETS'].astype(str).astype(float)
dataset['LIABILITIES'] =
dataset['LIABILITIES'].astype(str).astype(float)
dataset['CRIMINAL_CASES'] =
dataset['CRIMINAL_CASES'].astype(str).astype(int)
```

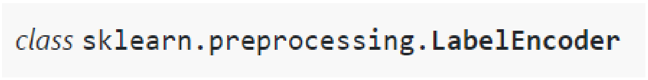
Figure 10: Transforming non - numeric data to numeric data features.

Objective 3: Transforming raw data into features.



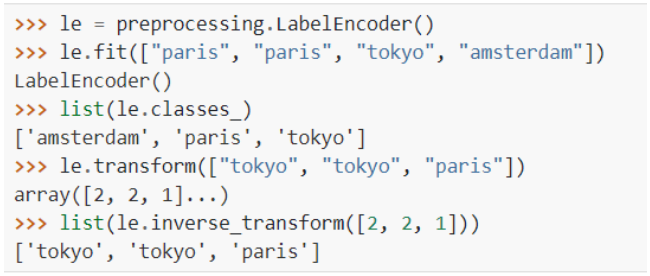
```
lblEncoder_state = LabelEncoder()
lblEncoder_state.fit(dataset['STATE'])
dataset['STATE'] = lblEncoder_state.transform(dataset['STATE'])

lblEncoder_cons = LabelEncoder()
lblEncoder_cons.fit(dataset['CONSTITUENCY'])
dataset['CONSTITUENCY'] =
lblEncoder_cons.transform(dataset['CONSTITUENCY'])

lblEncoder_name = LabelEncoder()
lblEncoder_name.fit(dataset['NAME'])
dataset['NAME'] = lblEncoder_name.transform(dataset['NAME'])

lblEncoder_party = LabelEncoder()
lblEncoder_party.fit(dataset['PARTY'])
dataset['PARTY'] = lblEncoder_party.transform(dataset['PARTY'])

lblEncoder_symbol = LabelEncoder()
lblEncoder_symbol.fit(dataset['SYMBOL'])
dataset['SYMBOL'] = lblEncoder_symbol.transform(dataset['SYMBOL'])

lblEncoder_gender = LabelEncoder()
lblEncoder_gender.fit(dataset['GENDER'])
dataset['GENDER'] = lblEncoder_gender.transform(dataset['GENDER'])

lblEncoder_category = LabelEncoder()
lblEncoder_category.fit(dataset['CATEGORY'])
dataset['CATEGORY'] =
lblEncoder_category.transform(dataset['CATEGORY'])

lblEncoder_edu = LabelEncoder()
lblEncoder_edu.fit(dataset['EDUCATION'])
dataset['EDUCATION'] = lblEncoder_edu.transform(dataset['EDUCATION'])
```

Figure 11: This figure is showing conversion of all the non - numeric data to numeric features.



```
class sklearn.preprocessing.LabelEncoder
```

Figure 12: The figure is showing LabelEncoder class. sklearn. preprocessing. LabelEncoder: LabelEncoder can be used to normalize labels.

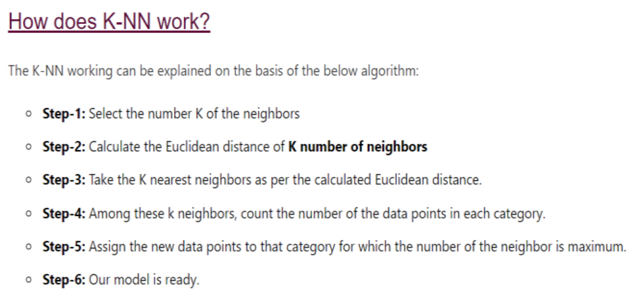


```
>>> le = preprocessing.LabelEncoder()
>>> le.fit(["paris", "paris", "tokyo", "amsterdam"])
LabelEncoder()
>>> list(le.classes_)
['amsterdam', 'paris', 'tokyo']
>>> le.transform(["tokyo", "tokyo", "paris"])
array([2, 2, 1]...)
>>> list(le.inverse_transform([2, 2, 1]))
['tokyo', 'tokyo', 'paris']
```

Figure 13: Example to understand LabelEncoder Transformation

Objective 4: Achieving better accuracy

- To judge this here we will dig into the K - Nearest Neighbors algorithm.
- KNN is a supervised machine learning model which is categorized under classification algorithms. The algorithm works by taking a data point and finding out the k closest data points.

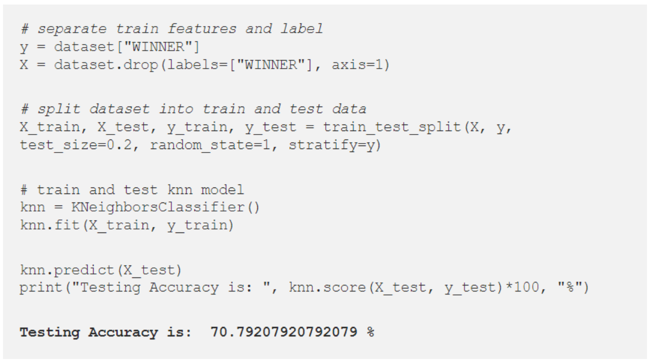


How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

Figure 14: Process how K - NN Model works

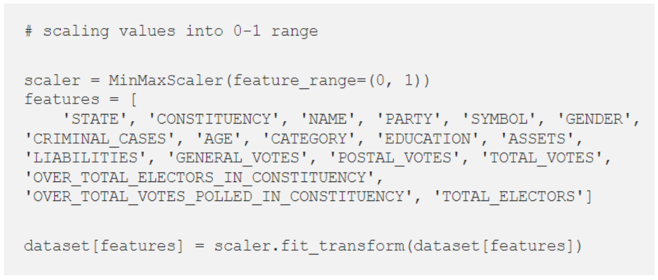


```
# separate train features and label
y = dataset["WINNER"]
X = dataset.drop(labels=["WINNER"], axis=1)

# split dataset into train and test data
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=1, stratify=y)

# train and test knn model
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)

knn.predict(X_test)
print("Testing Accuracy is: ", knn.score(X_test, y_test)*100, "%")

Testing Accuracy is:  70.79207920792079 %
```

Figure 15: This image shows an implementation of the K - NN Algorithm on our features to test the accuracy.

Normalizing data for improved accuracy

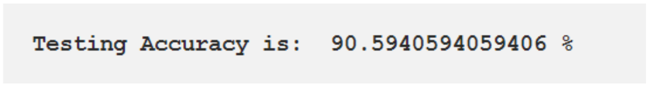
- MinMaxScaler from the scikit - learn library to scale down all the values into the 0–1 range.
- By doing this the accuracy levels can reach up to 85 – 90 %.



```
# scaling values into 0-1 range

scaler = MinMaxScaler(feature_range=(0, 1))
features = [
    'STATE', 'CONSTITUENCY', 'NAME', 'PARTY', 'SYMBOL', 'GENDER',
'CRIMINAL_CASES', 'AGE', 'CATEGORY', 'EDUCATION', 'ASSETS',
'LIABILITIES', 'GENERAL_VOTES', 'POSTAL_VOTES', 'TOTAL_VOTES',
'OVER_TOTAL_ELECTORS_IN_CONSTITUENCY',
'OVER_TOTAL_VOTES_POLLED_IN_CONSTITUENCY', 'TOTAL_ELECTORS']

dataset[features] = scaler.fit_transform(dataset[features])
```

Figure 16: This figure shows the normalizing of data

After reaching this stage of the dataset the accuracy has reached 90%.



```
Testing Accuracy is:  90.5940594059406 %
```

Figure 17: This figure shows retesting of the accuracy of the model

4. Conclusion & Future Scope

The main agenda of selecting Feature Engineering for Result Prediction is improving a supervised model's efficiency and accuracy. Its main feature is that itconverts the raw data into 'features' in a numeric format that can do wonders while making predictions.

Despite being in its nascent stages, feature engineering can reap the utmost benefits from the available data. It addresses both functional & non - functional aspects of a model. Feature engineering is a crucial step in data science. It ensures that relevant, reliable, and accurate data is fed to any predictive model.

Feature Engineering is still finding its way into different genres that will lead to the world of automation. Automated feature engineering will save you time, build better predictive models, create meaningful features, and prevent data leakage. Recent years have seen progress in automating the model selection and hyperparameter tuning, but the most important aspect of the machine learning pipeline, feature engineering, has largely been neglected. The most capable entry in this critical field is Featuretools, an open - source Python library.

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: //legal dictionary. The free dictionary. com/Electoral+ democracy

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

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

[12] Tim Verdonck, Bart Baesens, María Óskarsdóttir & Seppe vanden Broucke [2021], Special issue on feature engineering editorial

[13] J. Heaton, "An empirical analysis of feature engineering for predictive modeling, " *SoutheastCon 2016*, 2016, pp.1 - 6, DOI: 10.1109/SECON.2016.7506650.

[14] Ntakaris, Adamantios & Mirone, Giorgio & Kanniainen, Juho & Gabbouj, Moncef & Iosifidis, Alexandros. (2019). Feature Engineering for Mid - Price Prediction With Deep Learning. IEEE Access. PP.1 - 1.10.1109/ACCESS.2019.2924353.

[15] Wen Long, Zhichen Lu, Lingxiao Cui, Deep learning - based feature engineering for stock price movement prediction, Knowledge - Based Systems, Volume 164, 2019,

[16] Toasa G, Renato Mauricio & Maximiano, Marisa & Reis, Catarina & Guevara, David. (2018). Data visualization techniques for real - time information — A custom and dynamic dashboard for analyzing surveys' results.1 - 7.10.23919/CISTI.2018.8398641.

[17] Weiming Zhu, "A Study of Big - Data - Driven Data Visualization and Visual Communication Design Patterns", *Scientific Programming*, vol.2021, Article ID 6704937, 11 pages, 2021. https: //doi. org/10.1155/2021/6704937

[18] A Survey of Time Series Data Visualization ResearchYujie Fang[1], Hui Xu[1], and Jie Jiang[1]