**NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY GREATER NOIDA -201306**

**(An Autonomous Institute)**

**School of Computer Sciences & Engineering in Emerging Technologies**

Department of CSE (AI&ML)

**Session (2024 – 2025) LAB FILE**

**ON**

**INTERNSHIP ASSESSMENT-II**

**(ACSE0559)**

**(5TH SEMESTER/3RD YEAR)**

**Submitted To: Submitted By:**

**Ms. Renu**

**NAME: Sachin Vardhan**

**ROLL NO: 2201331530231 BRANCH & SECTION: AIML-A**





***Affiliated to Dr. A.P.J Abdul Kalam Technical University, Uttar Pradesh, Lucknow***

# PROJECT OVERVIEW

During my internship at Zetpeak, I had the opportunity to work on impactful projects in the field of Web Development and App Development.

Mobikart is a comprehensive web application designed for buying and selling mobile phones. It connects buyers and sellers on a single platform, offering a seamless user experience. Built using the latest web technologies, the application ensures ease of use, security, and efficiency.

**Tech Stack**

* **Frontend:** React.js with Tailwind CSS for styling.
* **Backend:** Supabase for database management and API services.
* **Authentication:** Clerk for secure user login and role management.
* **Hosting:** The application is deployed for public use, ensuring accessibility and scalability.

## Amazon Bestselling Books Analysis with JavaScript

* + *Eloquent JavaScript (3rd Edition)* by Marijn Haverbeke is an excellent choice for beginners, offering a deep dive into JavaScript fundamentals with interactive examples.
  + *JavaScript: The Good Parts* by Douglas Crockford focuses on the strengths of JavaScript, providing a concise and insightful guide for understanding its best features.
  + *You Don’t Know JS Yet* (Series) by Kyle Simpson is perfect for developers looking to master JavaScript, covering both core and advanced topics in detail.
  + *JavaScript: The Definitive Guide (7th Edition)* by David Flanagan is a comprehensive resource that serves as both a tutorial and a reference for developers of all levels.

In [1]:

**import** pandas **as** pd *# dataframe manipulation*

**import** numpy **as** np *# linear algebra*

*# data visualization*

**import** matplotlib.pyplot **as** plt

**%matplotlib** inline

**import** seaborn **as** sns

print('Seaborn verion', sns**.** version ) sns**.**set\_style('whitegrid')

*# text data*

**import** string

**import** re

df **=** pd**.**read\_csv('bestsellers with categories.csv')

Seaborn verion 0.12.2

In [ ]:

df**.**rename(columns**=**{"User Rating": "User\_Rating"}, inplace**=True**) df[df**.**Author **==** 'J. K. Rowling']

df[df**.**Author **==** 'J.K. Rowling']

df**.**loc[df**.**Author **==** 'J. K. Rowling', 'Author'] **=** 'J.K. Rowling'

df['name\_len'] **=** df['Name']**.**apply(**lambda** x: len(x) **-** x**.**count(" ")) *# subtract whit*

punctuations **=** string**.**punctuation

print('list of punctuations : ', punctuations)

*# percentage of punctuations*

**def** count\_punc(text):

"""This function counts the number of punctuations in a text""" count **=** sum(1 **for** char **in** text **if** char **in** punctuations)

**return** round(count**/**(len(text) **-** text**.**count(" "))**\***100, 3)

*# apply function*

df['punc%'] **=** df['Name']**.**apply(**lambda** x: count\_punc(x))

In [2]:

no\_dup **=** df**.**drop\_duplicates('Name')

g\_count **=** no\_dup['Genre']**.**value\_counts() fig, ax **=** plt**.**subplots(figsize**=**(8, 8))

**def** make\_autopct(values):

**def** my\_autopct(pct):

total **=** sum(values)

val **=** int(round(pct**\***total**/**100.0))

**return** '{p:.2f}%\n({v:d})'**.**format(p**=**pct,v**=**val)

**return** my\_autopct

genre\_col **=** ['navy','crimson']

*#genre\_col = ['khaki','plum']*

center\_circle **=** plt**.**Circle((0, 0), 0.7, color**=**'white')

plt**.**pie(x**=**g\_count**.**values, labels**=**g\_count**.**index, autopct**=**make\_autopct(g\_count**.**value startangle**=**90, textprops**=**{'size': 15}, pctdistance**=**0.5, colors**=**genre\_col

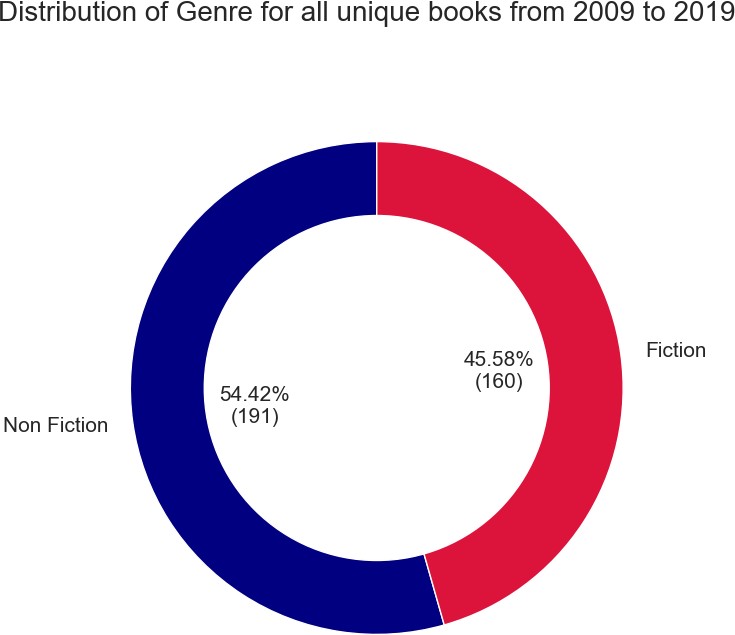
ax**.**add\_artist(center\_circle)

fig**.**suptitle('Distribution of Genre for all unique books from 2009 to 2019', fontsi fig**.**show()

C:\Users\HP\AppData\Local\Temp\ipykernel\_4992\1094739433.py:22: UserWarning: Matpl otlib is currently using module://matplotlib\_inline.backend\_inline, which is a non

-GUI backend, so cannot show the figure.

fig.show()



In [3]:

y1 **=** np**.**arange(2009, 2014) y2 **=** np**.**arange(2014, 2020)

g\_count **=** df['Genre']**.**value\_counts()

fig, ax **=** plt**.**subplots(2, 6, figsize**=**(12,6))

ax[0,0]**.**pie(x**=**g\_count**.**values, labels**=None**, autopct**=**'%1.1f%%',

startangle**=**90, textprops**=**{'size': 12, 'color': 'white'}, pctdistance**=**0.5, radius**=**1.3, colors**=**genre\_col)

ax[0,0]**.**set\_title('2009 - 2019\n(Overall)', color**=**'darkgreen', fontdict**=**{'fontsize

**for** i, year **in** enumerate(y1):

counts **=** df[df['Year'] **==** year]['Genre']**.**value\_counts()

ax[0,i**+**1]**.**set\_title(year, color**=**'darkred', fontdict**=**{'fontsize': 15}) ax[0,i**+**1]**.**pie(x**=**counts**.**values, labels**=None**, autopct**=**'%1.1f%%',

startangle**=**90, textprops**=**{'size': 12,'color': 'white'},

pctdistance**=**0.5, colors**=**genre\_col, radius**=**1.1)

**for** i, year **in** enumerate(y2):

counts **=** df[df['Year'] **==** year]['Genre']**.**value\_counts()

ax[1,i]**.**pie(x**=**counts**.**values, labels**=None**, autopct**=**'%1.1f%%',

startangle**=**90, textprops**=**{'size': 12,'color': 'white'}, pctdistance**=**0.5, colors**=**genre\_col, radius**=**1.1)

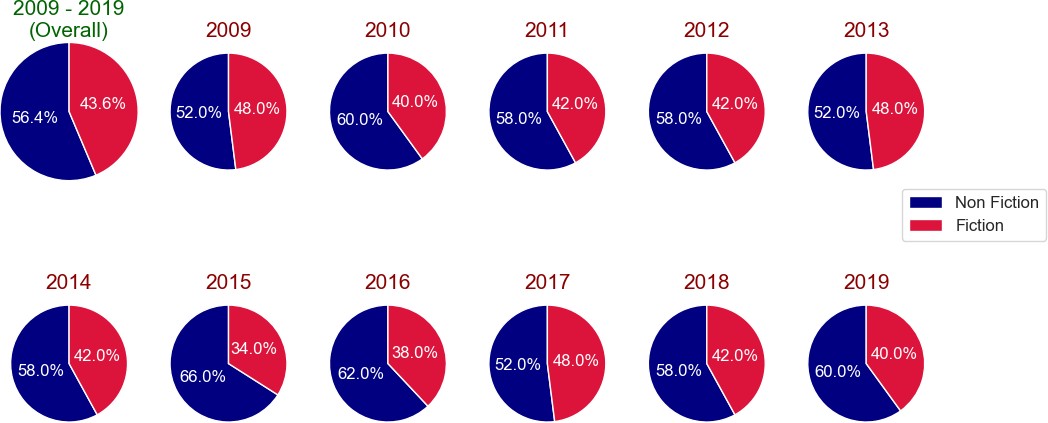
ax[1,i]**.**set\_title(year, color**=**'darkred', fontdict**=**{'fontsize': 15})

*#plt.suptitle('Distribution of Fiction and Non-Fiction books for every year from 20 #fontsize=25)*

fig**.**legend(g\_count**.**index, loc**=**'center right', fontsize**=**12) fig**.**show()

C:\Users\HP\AppData\Local\Temp\ipykernel\_4992\413210449.py:29: UserWarning: Matplo tlib is currently using module://matplotlib\_inline.backend\_inline, which is a non- GUI backend, so cannot show the figure.

fig.show()



In [4]:

best\_nf\_authors **=** df**.**groupby(['Author', 'Genre'])**.**agg({'Name': 'count'})**.**unstack() best\_f\_authors **=** df**.**groupby(['Author', 'Genre'])**.**agg({'Name': 'count'})**.**unstack()[

**with** plt**.**style**.**context('Solarize\_Light2'):

fig, ax **=** plt**.**subplots(1, 2, figsize**=**(8,8))

ax[0]**.**barh(y**=**best\_nf\_authors**.**index, width**=**best\_nf\_authors**.**values, color**=**genre\_col[0])

ax[0]**.**invert\_xaxis()

ax[0]**.**yaxis**.**tick\_left()

ax[0]**.**set\_xticks(np**.**arange(max(best\_f\_authors**.**values)**+**1))

ax[0]**.**set\_yticklabels(best\_nf\_authors**.**index, fontsize**=**12, fontweight**=**'semibold ax[0]**.**set\_xlabel('Number of appreances')

ax[0]**.**set\_title('Non Fiction Authors')

ax[1]**.**barh(y**=**best\_f\_authors**.**index, width**=**best\_f\_authors**.**values, color**=**genre\_col[1])

ax[1]**.**yaxis**.**tick\_right()

ax[1]**.**set\_xticks(np**.**arange(max(best\_f\_authors**.**values)**+**1))

ax[1]**.**set\_yticklabels(best\_f\_authors**.**index, fontsize**=**12, fontweight**=**'semibold' ax[1]**.**set\_title('Fiction Authors')

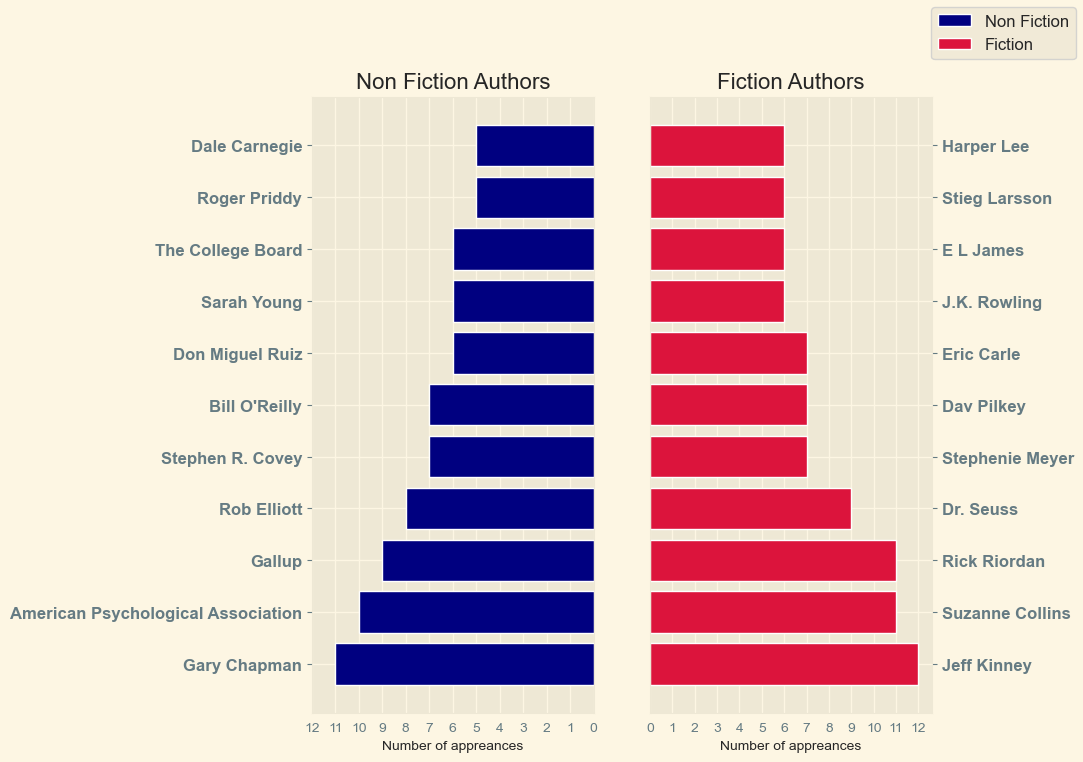
ax[1]**.**set\_xlabel('Number of appreances')

fig**.**legend(['Non Fiction', 'Fiction'], fontsize**=**12) plt**.**show()

C:\Users\HP\AppData\Local\Temp\ipykernel\_4992\2335528297.py:12: UserWarning: Fixed Formatter should only be used together with FixedLocator

ax[0].set\_yticklabels(best\_nf\_authors.index, fontsize=12, fontweight='semibold') C:\Users\HP\AppData\Local\Temp\ipykernel\_4992\2335528297.py:20: UserWarning: Fixed Formatter should only be used together with FixedLocator

ax[1].set\_yticklabels(best\_f\_authors.index, fontsize=12, fontweight='semibold')



In [5]:

n\_best **=** 20

top\_authors **=** df**.**Author**.**value\_counts()**.**nlargest(n\_best)

no\_dup **=** df**.**drop\_duplicates('Name') *# removes all rows with duplicate book names*

fig, ax **=** plt**.**subplots(1, 3, figsize**=**(11,10), sharey**=True**) color **=** sns**.**color\_palette("hls", n\_best)

ax[0]**.**hlines(y**=**top\_authors**.**index , xmin**=**0, xmax**=**top\_authors**.**values, color**=**color, l

ax[0]**.**plot(top\_authors**.**values, top\_authors**.**index, 'go', markersize**=**9) ax[0]**.**set\_xlabel('Number of appearences')

ax[0]**.**set\_xticks(np**.**arange(top\_authors**.**values**.**max()**+**1))

ax[0]**.**set\_yticklabels(top\_authors**.**index, fontweight**=**'semibold') ax[0]**.**set\_title('Appearences')

book\_count **=** []

total\_reviews **=** []

**for** name, col **in** zip(top\_authors**.**index, color):

book\_count**.**append(len(no\_dup[no\_dup**.**Author **==** name]['Name']))

total\_reviews**.**append(no\_dup[no\_dup**.**Author **==** name]['Reviews']**.**sum()**/**1000)

ax[1]**.**hlines(y**=**top\_authors**.**index , xmin**=**0, xmax**=**book\_count, color**=**color, linestyle ax[1]**.**plot(book\_count, top\_authors**.**index, 'go', markersize**=**9)

ax[1]**.**set\_xlabel('Number of unique books')

ax[1]**.**set\_xticks(np**.**arange(max(book\_count)**+**1)) ax[1]**.**set\_title('Unique books')

ax[2]**.**barh(y**=**top\_authors**.**index, width**=**total\_reviews, color**=**color, edgecolor**=**'black

**for** name, val **in** zip(top\_authors**.**index, total\_reviews): ax[2]**.**text(val**+**2, name, val)

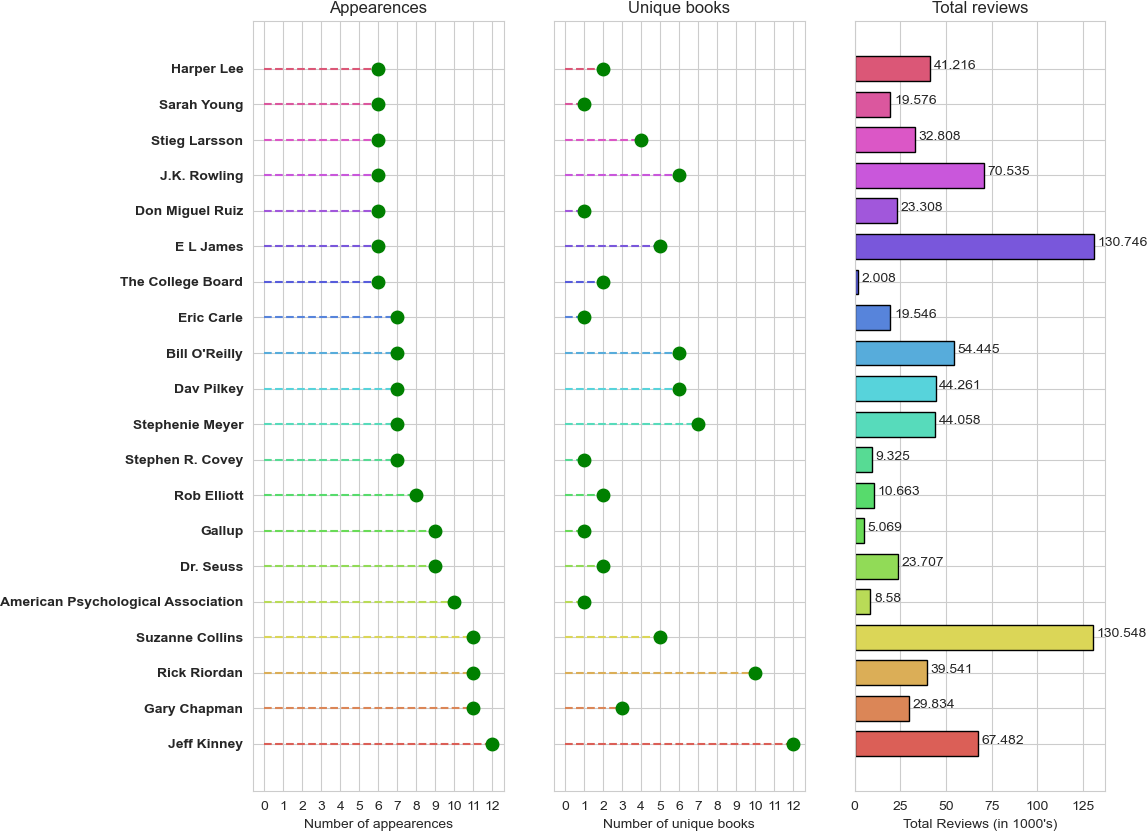
ax[2]**.**set\_xlabel("Total Reviews (in 1000's)") ax[2]**.**set\_title('Total reviews')

*#plt.suptitle('Top 20 best selling Authors (from 2009 to 2019) details', fontsize=1*

plt**.**show()

C:\Users\HP\AppData\Local\Temp\ipykernel\_4992\1934722200.py:14: UserWarning: Fixed Formatter should only be used together with FixedLocator

ax[0].set\_yticklabels(top\_authors.index, fontweight='semibold')



In [ ]:

Author Jeff Kinney **is** the best**-**selling author **with** 12 appearances **in** best**-**selling

# Predict US Elections with Python

* I will analyse the sentiments of people for the candidates and then at the end, I will conclude based on the most number of positive and negative tweets against the candidates.
* The datasets that I am using in this task to predict the US Elections are collected from twitter by the official twitter handles of Donald Trump and Joe Biden.
* I will get started with sentiment analysis. I will do it by using the Textblob package in Python. Here I will use this package to perform simple text classification in either positive or negative on the basis of sentiment analysis.
* Now before moving forward let’s understand what is Polarity. **Polarity** ranges from -1 to +1(negative to positive) and tells whether the text has negative sentiments or positive sentiments. Polarity tells about factual information.
* it is very clear that Joe Biden is getting more Positive tweets and less negative tweets as compared to Donald Trump. So it will not be wrong to conclude that Joe

Bined is more preffered by the people to win the US Presidential Elections than Donald Trump.

In [1]:

**import** pandas **as** pd **import** numpy **as** np **import** seaborn **as** sns

**import** matplotlib.pyplot **as** plt **from** textblob **import** TextBlob **from** wordcloud **import** WordCloud **import** plotly.graph\_objects **as** go **import** plotly.express **as** px

trump\_reviews **=** pd**.**read\_csv("Trumpall2.csv") biden\_reviews **=** pd**.**read\_csv("Bidenall2.csv")

In [2]:

print(trump\_reviews**.**head()) *#first 5 rows*

print(biden\_reviews**.**head())

|  |  |  |  |
| --- | --- | --- | --- |
|  | user |  | text |
| 0 | manny\_rosen | @sanofi | please tell us how many shares the Cr... |

1. osi\_abdul
2. Patsyrw

3seyedebrahimi\_m 4 James09254677

user

0 MarkHodder3

https://t.co/atM98CpqF7 Like, comment, RT #P... Your AG Barr is as useless &amp; corrupt as y... Mr. Trump! Wake Up! Most of the comments bel... After 4 years you think you would have figure...

text @JoeBiden And we’ll find out who won in 2026...

1 K87327961G

1. OldlaceA
2. penblogger
3. Aquarian0264

@JoeBiden Your Democratic Nazi Party cannot be...

@JoeBiden So did Lying Barr @JoeBiden It's clear you didnt compose this tw... @JoeBiden I will vote in person thank you.

|  |  |  |
| --- | --- | --- |
| In | [ ]: | *#analyzing sentiments* |
|  |  |  |
| In | [3]: | textblob1 **=** TextBlob(trump\_reviews["text"][10]) print("Trump :",textblob1**.**sentiment) |
|  |  | textblob2 **=** TextBlob(biden\_reviews["text"][500]) |
|  |  | print("Biden :",textblob2**.**sentiment) |
|  |  | Trump : Sentiment(polarity=0.15, subjectivity=0.3125) |
|  |  | Biden : Sentiment(polarity=0.6, subjectivity=0.9) |
| In | [4]: | **def** find\_pol(review): |
|  |  | **return** TextBlob(review)**.**sentiment**.**polarity  trump\_reviews["Sentiment Polarity"] **=**  trump\_reviews["text"]**.**apply(find\_pol) print(trump\_reviews**.**tail())  biden\_reviews["Sentiment Polarity"] **=**  biden\_reviews["text"]**.**apply(find\_pol) print(biden\_reviews**.**tail()) |

2783

user 4diva63

text \ @realDonaldTrump For the 1/100 time, absentee ...

In [ ]:

*#Polarity ranges from -1 to +1(negative to positive) and tells whether the*

*#text has negative sentiments or positive sentiments. Polarity tells about factual*

2784 hidge826 @realDonaldTrump If you’re so scared of losing... 2785 SpencerRossy @realDonaldTrump I rarely get involved with fo... 2786 ScoobyMcpherson @realDonaldTrump This is the moment when Trump... 2787 bjklinz @realDonaldTrump I’m sorry, Donald. No. #POTUS

|  |  |  |  |
| --- | --- | --- | --- |
|  | Sentiment Polarity | |  |
| 2783 | 0.000 | |
| 2784 | 0.000 | |
| 2785 | 0.225 | |
| 2786 | 0.000 | |
| 2787 | -0.500 | |
|  | user | | text \ |
| 2535 | meryn1977 @JoeBiden | | You'll just try to calm those waters... |
| 2536 | BSNelson114 @JoeBiden | | 96 days 96 dias #VoteJoeBiden2020 #... |
| 2537 | KenCapel @JoeBiden | | YOU THINK YOU CAN DO THAT??? YOU CAN... |
| 2538 | LeslyeHale @JoeBiden | | Trump wants our children back at sch... |
| 2539 | rerickre @JoeBiden | | ... and I know, because it’s much co... |
|  | Sentiment | Polarity | |
| 2535 |  | 0.15 | |
| 2536 |  | 0.00 | |
| 2537 |  | 0.00 | |
| 2538 |  | 0.10 | |
| 2539 |  | 0.20 | |

In [5]:

trump\_reviews["Expression Label"] **=** np**.**where(trump\_reviews["Sentiment Polarity"]**>**0, trump\_reviews["Expression Label"][trump\_reviews["Sentiment Pporilnatri(tyr"um]p**=**\_**=**r0e]v**=**i"eNwesut**.**rtali"l())

biden\_reviews["Expression Label"] **=** np**.**where(biden\_reviews["Sentiment Polarity"]**>**0 biden\_reviews["Expression Label"][trump\_reviews["Sentiment Polarity"]**==**0]**=**"Neutral print(biden\_reviews**.**tail())

,

"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | user |  | text | \ |
| 2783 | 4diva63 | @realDonaldTrump | For the 1/100 time, absentee ... |  |
| 2784 | hidge826 | @realDonaldTrump | If you’re so scared of losing... |  |
| 2785 | SpencerRossy | @realDonaldTrump | I rarely get involved with fo... |  |
| 2786 | ScoobyMcpherson | @realDonaldTrump | This is the moment when Trump... |  |

2787 bjklinz @realDonaldTrump I’m sorry, Donald. No. #POTUS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sentiment Polarity Expression Label | | | | | |
| 2783 |  | 0.000 Neutral | | | |
| 2784 |  | 0.000 Neutral | | | |
| 2785 |  | 0.225 positive | | | |
| 2786 |  | 0.000 Neutral | | | |
| 2787 | user meryn1977 | | -0.500 negative  text @JoeBiden You'll just try to calm those waters... | | \ |
| 2535 |
| 2536 | BSNelson114 | | @JoeBiden 96 days 96 dias #VoteJoeBiden2020 #... | |  |
| 2537 | KenCapel | | @JoeBiden YOU THINK YOU CAN DO THAT??? YOU CAN... | |  |
| 2538 | LeslyeHale | | @JoeBiden Trump wants our children back at sch... | |  |
| 2539 | rerickre | | @JoeBiden ... and I know, because it’s much co... | |  |
|  | Sentiment | Polarity | | Expression Label | |
| 2535 |  | 0.15 | | Neutral | |
| 2536 |  | 0.00 | | Neutral | |
| 2537 |  | 0.00 | | negative | |
| 2538 |  | 0.10 | | Neutral | |
| 2539 |  | 0.20 | | positive | |

C:\Users\HP\AppData\Local\Temp\ipykernel\_11212\168499711.py:2: SettingWithCopyWarn ing:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-

docs/stabl e/user\_guide/indexing.html#returning-a-view-versus-a-copy trump\_reviews["Expression Label"][trump\_reviews["Sentiment

Polarity"]==0]="Neutr al"

C:\Users\HP\AppData\Local\Temp\ipykernel\_11212\168499711.py:6: SettingWithCopyWarn ing:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas- docs/stabl e/user\_guide/indexing.html#returning-a-view-versus-a-copy

biden\_reviews["Expression Label"][trump\_reviews["Sentiment Polarity"]==0]="Neutr al"

In [6]:

reviews1 **=** trump\_reviews[trump\_reviews['Sentiment Polarity'] **==**

0.0000] print(reviews1**.**shape)

cond1**=**trump\_reviews['Sentiment Polarity']**.**isin(reviews1['Sentiment Polarity']) trump\_reviews**.**drop(trump\_reviews[cond1]**.**index, inplace **= True**) print(trump\_reviews**.**shape)

reviews2 **=** biden\_reviews[biden\_reviews['Sentiment Polarity'] **==**

0.0000] print(reviews2**.**shape)

cond2**=**biden\_reviews['Sentiment Polarity']**.**isin(reviews1['Sentiment Polarity']) biden\_reviews**.**drop(biden\_reviews[cond2]**.**index, inplace **= True**) print(biden\_reviews**.**shape)

|  |  |
| --- | --- |
| (1464, | 4) |
| (1324, | 4) |
| (1509, | 4) |
| (1031, | 4) |

In [7]:

*# Donald Trump* np**.**random**.**seed(10) remove\_n **=**324

drop\_indices **=** np**.**random**.**choice(trump\_reviews**.**index, remove\_n, replace**=False**) df\_subset\_trump **=** trump\_reviews**.**drop(drop\_indices) print(df\_subset\_trump**.**shape)

*# Joe Biden* np**.**random**.**seed(10) remove\_n **=**31

drop\_indices **=** np**.**random**.**choice(biden\_reviews**.**index, remove\_n, replace**=False**) df\_subset\_biden **=** biden\_reviews**.**drop(drop\_indices) print(df\_subset\_biden**.**shape)

(1000, 4)

(1000, 4)

In [8]:

count\_1 **=** df\_subset\_trump**.**groupby('Expression Label')**.**count() print(count\_1)

negative\_per1 **=** (count\_1['Sentiment Polarity'][0]**/**1000)**\***10 positive\_per1 **=** (count\_1['Sentiment Polarity'][1]**/**1000)**\***100

count\_2 **=** df\_subset\_biden**.**groupby('Expression Label')**.**count() print(count\_2)

negative\_per2 **=** (count\_2['Sentiment Polarity'][0]**/**1000)**\***100 positive\_per2 **=** (count\_2['Sentiment Polarity'][1]**/**1000)**\***100

Politicians **=** ['Joe Biden', 'Donald Trump']

lis\_pos **=** [positive\_per1, positive\_per2] lis\_neg **=** [negative\_per1, negative\_per2]

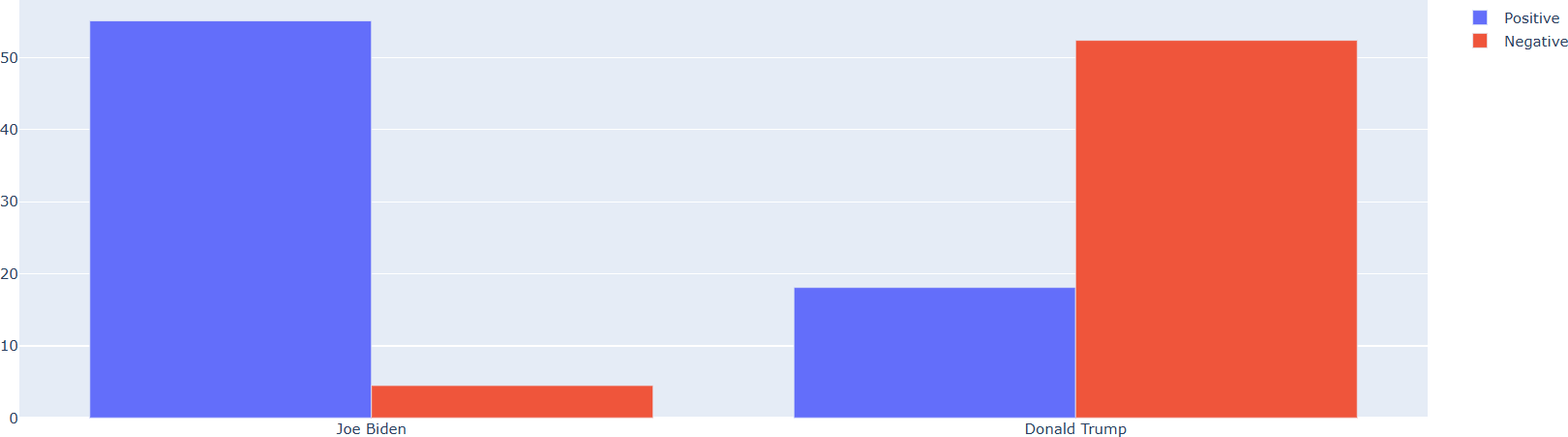
fig **=** go**.**Figure(data**=**[

go**.**Bar(name**=**'Positive', x**=**Politicians, y**=**lis\_pos), go**.**Bar(name**=**'Negative', x**=**Politicians, y**=**lis\_neg)

])

*# Change the bar mode* fig**.**update\_layout(barmode**=**'group') fig**.**show()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | user | text | Sentiment | Polarity |
| Expression | Label |  |  |  |
| negative | 449 | 449 |  | 449 |
| positive  Expression | 551  user | 551  text | Sentiment | 551  Polarity |
| Label |  |  |  |
| Neutral | 524 | 524 |  | 524 |
| negative | 181 | 181 |  | 181 |
| positive | 295 | 295 |  | 295 |



In [ ]:

**Electricity Price Prediction with Machine Learning**

Suppose that your business relies on computing services where the power consumed by your machines varies throughout the day. You do not know the actual cost of the

electricity consumed by the machines throughout the day, but the organisation has provided you with historical data of the price of the electricity consumed by the machines. Below is the information of the [data](https://raw.githubusercontent.com/amankharwal/Website-data/master/electricity.csv) we have for the task of forecasting electricity prices:

1. DateTime: Date and time of the record
2. Holiday: contains the name of the holiday if the day is a national holiday
3. HolidayFlag: contains 1 if it’s a bank holiday otherwise 0
4. DayOfWeek: contains values between 0-6 where 0 is Monday
5. WeekOfYear: week of the year
6. Day: Day of the date
7. Month: Month of the date
8. Year: Year of the date
9. PeriodOfDay: half-hour period of the day
10. ForcastWindProduction: forecasted wind production
11. SystemLoadEA forecasting national load
12. SMPEA: forecasted price
13. ORKTemperature: actual temperature measured
14. ORKWindspeed: actual wind speed measured
15. CO2 Intensity: actual C02 intensity for the electricity produced
16. ActualWindProduction: actual wind energy production
17. SystemLoadEP2: actual national system load
18. SMPEP2: the actual price of the electricity consumed (labels or values to be predicted)

So your task here is to use this data to train a machine

learning model to predict the price of electricity consumed by the machines.

In [16]:

**import** pandas **as** pd

**import** numpy **as** np

data **=** pd.read\_csv("https://raw.githubusercontent.com/amankharwal/Website-d print(data.head())

DateTime Holiday HolidayFlag DayOfWeek WeekOfYear Day Mont

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| h | \ |  | | | | | | |
| 0 | 01/11/2011 | 00:00 | None | 0 | 1 | 44 | 1 | 1 |
| 1 |  |  |  |  |  |  |  |  |
| 1 | 01/11/2011 | 00:30 | None | 0 | 1 | 44 | 1 | 1 |
| 1 |  |  |  |  |  |  |  |  |
| 2 | 01/11/2011 | 01:00 | None | 0 | 1 | 44 | 1 | 1 |
| 1 |  |  |  |  |  |  |  |  |
| 3 | 01/11/2011 | 01:30 | None | 0 | 1 | 44 | 1 | 1 |
| 1 |  |  |  |  |  |  |  |  |
| 4 | 01/11/2011 | 02:00 | None | 0 | 1 | 44 | 1 | 1 |
| 1 |  |  |  |  |  |  |  |  |

Year PeriodOfDay ForecastWindProduction SystemLoadEA SMPEA \ 0 2011 0 315.31 3388.77 49.26

1 2011 1 321.80 3196.66 49.26

2 2011 2 328.57 3060.71 49.10

3 2011 3 335.60 2945.56 48.04

4 2011 4 342.90 2849.34 33.75

ORKTemperature ORKWindspeed CO2Intensity ActualWindProduction SystemLoad

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EP2 | \ |  | | | | |
| 0 |  | 6.00 | 9.30 | 600.71 | 356.00 | 315 |
| 9.60 |  |  |  |  |  |  |
| 1 |  | 6.00 | 11.10 | 605.42 | 317.00 | 297 |
| 3.01 |  |  |  |  |  |  |
| 2 |  | 5.00 | 11.10 | 589.97 | 311.00 | 283 |
| 4.00 |  |  |  |  |  |  |
| 3 |  | 6.00 | 9.30 | 585.94 | 313.00 | 272 |
| 5.99 |  |  |  |  |  |  |
| 4 |  | 6.00 | 11.10 | 571.52 | 346.00 | 265 |
| 5.64 |  |  |  |  |  |  |

SMPEP2

0 54.32

1 54.23

2 54.23

3 53.47

4 39.87

C:\Users\HP\AppData\Local\Temp\ipykernel\_15892\3353965664.py:3: DtypeWarni ng: Columns (9,10,11,14,15,16,17) have mixed types. Specify dtype option o n import or set low\_memory=False.

data = pd.read\_csv("https://raw.githubusercontent.com/amankharwal/Websit e-data/master/electricity.csv")

In [17]:

data.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 38014 entries, 0 to 38013 Data columns (total 18 columns):

# Column Non-Null Count Dtype

* 1. DateTime 38014 non-null object
  2. Holiday 38014 non-null object
  3. HolidayFlag 38014 non-null int64
  4. DayOfWeek 38014 non-null int64
  5. WeekOfYear 38014 non-null int64
  6. Day 38014 non-null int64
  7. Month 38014 non-null int64
  8. Year 38014 non-null int64
  9. PeriodOfDay 38014 non-null int64
  10. ForecastWindProduction 38014 non-null object
  11. SystemLoadEA 38014 non-null object
  12. SMPEA 38014 non-null object
  13. ORKTemperature 38014 non-null object
  14. ORKWindspeed 38014 non-null object
  15. CO2Intensity 38014 non-null object
  16. ActualWindProduction 38014 non-null object
  17. SystemLoadEP2 38014 non-null object
  18. SMPEP2 38014 non-null object dtypes: int64(7), object(11)

memory usage: 5.2+ MB

In [18]:

data["ForecastWindProduction"] **=** pd.to\_numeric(data["ForecastWindProduction data["SystemLoadEA"] **=** pd.to\_numeric(data["SystemLoadEA"], errors**=** 'coerce' data["SMPEA"] **=** pd.to\_numeric(data["SMPEA"], errors**=** 'coerce')

data["ORKTemperature"] **=** pd.to\_numeric(data["ORKTemperature"], errors**=** 'coe data["ORKWindspeed"] **=** pd.to\_numeric(data["ORKWindspeed"], errors**=** 'coerce' data["CO2Intensity"] **=** pd.to\_numeric(data["CO2Intensity"], errors**=** 'coerce' data["ActualWindProduction"] **=** pd.to\_numeric(data["ActualWindProduction"], data["SystemLoadEP2"] **=** pd.to\_numeric(data["SystemLoadEP2"], errors**=** 'coerc data["SMPEP2"] **=** pd.to\_numeric(data["SMPEP2"], errors**=** 'coerce')

|  |  |  |
| --- | --- | --- |
| In [19]: | data.isnull().sum() |  |
| Out[19]: | DateTime | 0 |
|  | Holiday | 0 |
|  | HolidayFlag | 0 |
|  | DayOfWeek | 0 |
|  | WeekOfYear | 0 |
|  | Day | 0 |
|  | Month | 0 |
|  | Year | 0 |
|  | PeriodOfDay | 0 |
|  | ForecastWindProduction | 5 |
|  | SystemLoadEA | 2 |
|  | SMPEA | 2 |
|  | ORKTemperature | 295 |
|  | ORKWindspeed | 299 |
|  | CO2Intensity | 7 |
|  | ActualWindProduction | 5 |
|  | SystemLoadEP2 | 2 |
|  | SMPEP2 | 2 |
|  | dtype: int64 |  |
| In [20]: | data **=** data.dropna() |  |

In [21]:

**import** seaborn **as** sns

**import** matplotlib.pyplot **as** plt

correlations **=** data.corr(method**=**'pearson') plt.figure(figsize**=**(16, 12))

sns.heatmap(correlations, cmap**=**"coolwarm", annot**=True**) plt.show()

In [22]:

x **=** data[["Day", "Month", "ForecastWindProduction", "SystemLoadEA", "SMPEA", "ORKTemperature", "ORKWindspeed", "CO2Intensity", "ActualWindProduction", "SystemLoadEP2"]]

y **=** data["SMPEP2"]

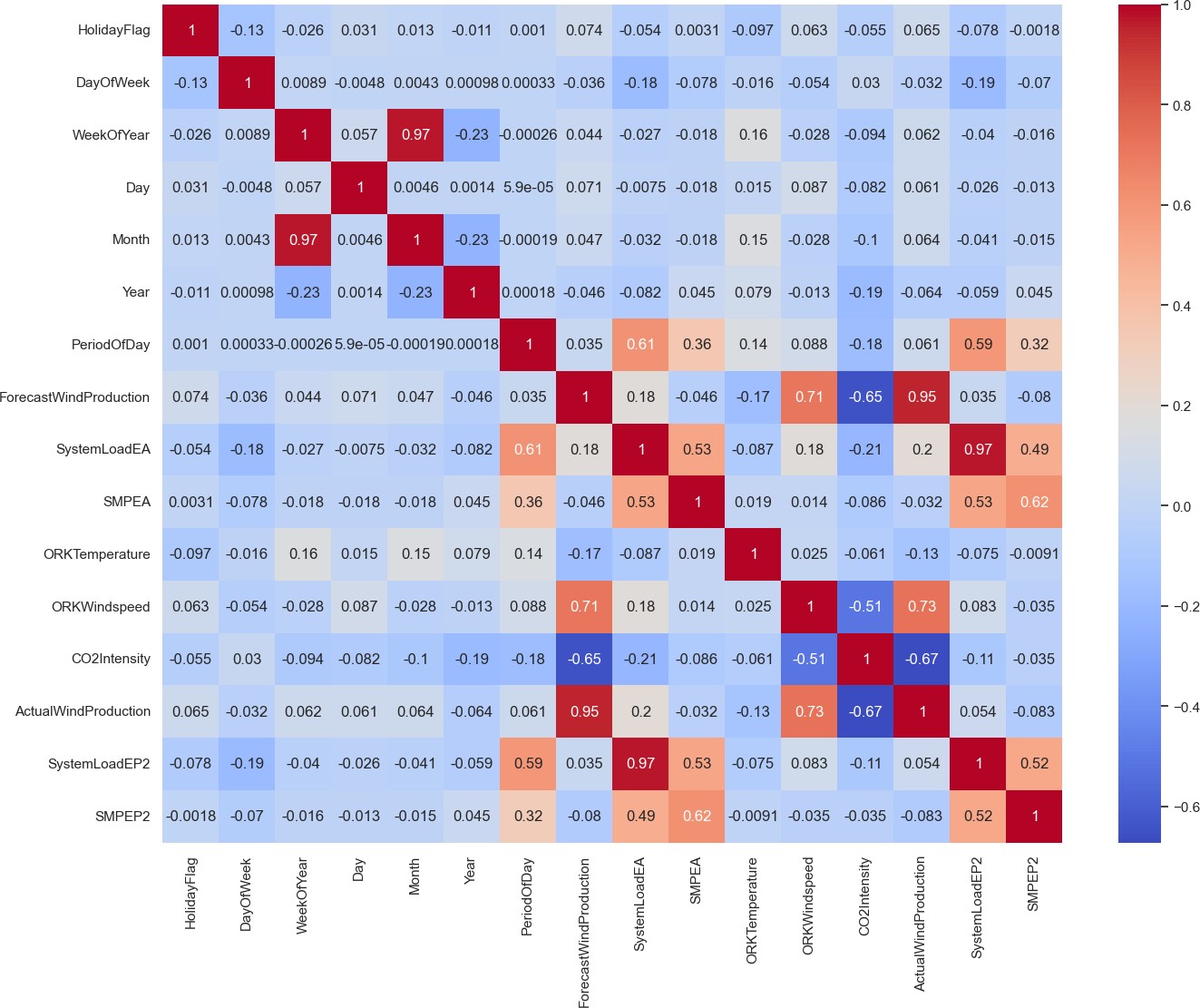
**from** sklearn.model\_selection **import** train\_test\_split xtrain, xtest, ytrain, ytest **=** train\_test\_split(x, y,

test\_size**=**0.2,

random\_state**=**42)

C:\Users\HP\AppData\Local\Temp\ipykernel\_15892\2280798960.py:3: FutureWarn ing: The default value of numeric\_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or s pecify the value of numeric\_only to silence this warning.

correlations = data.corr(method='pearson')



n [23]:

**from** sklearn.ensemble **import** RandomForestRegressor model **=** RandomForestRegressor()

model.fit(xtrain, ytrain)

Out[23]: RandomForestRegressor()

**In a Jupyter**

**environment, please rerun this cell to show the HTML representation or trust the no**

|  |  |  |
| --- | --- | --- |
| *#features = [["Day", "Month", "ForecastWindProduction", "SystemLoadEA", "SM*  features **=** np.array([[10, 12, 54.10, 4241.05, 49.56, 9.0, 14.8, 491.32, 54.  model.predict(features) | | |
|  |  |  |

# INTERNSHIP OFFER LETTER



**INTERNSHIP COMPLETION CERTIFICATE**

****