A

PROJECT REPORT

ON

“**Style-sync** MACHINE LEARNING IN FASHION INDUSTRY”

SUBMITTED TO THE

SAVITRIBAI PHULE PUNE UNIVERSITY

IN THE PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE

OF

“BACHELOR OF ENGINEERING IN ARTIFICIAL INTELLIGENCE AND DATA SCIENCE”

BY

Mugdha Borse	Roll No. B1411009
Prabhat Manna	Roll No. B1411047

UNDER THE GUIDANCE OF

Prof. Dikshendra Sarpate

Department of Artificial Intelligence and Data Science

Zeal Education Society’s

ZEAL COLLEGE OF ENGINEERING AND RESEARCH

NARHE, PUNE-411041

2023-2024

CERTIFICATE

This is to certify that the project report entitled **“Style-sync** MACHINE LEARNING IN FASHION INDUSTRY**”**

SUBMITTED BY

Mugdha Borse	Roll No. B1411009
Prabhat Manna	Roll No. B1411047

is a bonafide work carried out by them under the supervision of Prof. Dikshendra Sarpate and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University for the award of the Degree of Bachelor of Engineering (Artificial Intelligence and Data Science). This Project report has not been earlier submitted to any other Institute or University for the award of any degree or diploma.

Prof.Dikshendra Sarpate
Internal Guide
Dept of Artificial Intelligence
and Data Science

Prof.Dikshendra Sarpate
Head of department
Dept of Artificial Intelligence
and Data Science

(Prof.________________)
External Examiner
SPPU

(Dr. A. M.Kate)
Principal,
Zeal College of Engineering &Research, Pune-41

Place: Pune
Date:

ACKNOWLEDGEMENT

Firstly, we take this privilege to thank sincerely of our college management in extending their incomparable support and facilities for the completion of our project stage-1 in the department.We would like to express our sincere gratitude to our project guide **Prof. Dikshendra Sarpate**, Assistant professor, Department of Artificial Intelligence and Data Science, for his valuable guidance and suggestions to complete our project stage-1. We are thankful for his valuable feedback and suggestions for the completion of this work. We would like to extend our wishes to our project coordinator **Prof. Shweta Kakade,** Department of Artificial Intelligence and Data Science, for her coordination, encouragement and valuable support throughout our project stage-1.We take this immense pleasure to thank our beloved Head of the Department, **Prof. Dikshendra Sarpate,** Department of Artificial Intelligence and Data Science, for his valuable guidance, support and encouragement and permitting us to use resources in the department. We are extremely thankful to our Principal **Dr. Ajit Kate** for his support and guidance and giving permission to use resources in the department and college.Last but not the least; we are very thankful to all the teaching and non-teaching staff members in the department for their kind cooperation and support given throughout the completion of our project.

Mugdha Borse(B1411009)

Prabhat Manna (B1411047)

ABSTRACT

Fashion analytics has been a hot topic, such as predicting trends and fashion recommendations. As one of the dominant clothing features, color dramatically influences people’s shopping behaviors. Understanding popular colors and color combinations are of high business value.

The AI-based stylist model intends to identify the exact shades of colors with a specific prediction of their name and predict the other colors harmonizing with the detected one. Mixing and matching colorful clothes is an essential feature of having a good fashion sense. A study is reliable that a typical human can perceive about 1 million different shades of colors. Nevertheless, in several instances, an individual with "encroma" can see only 1% of them (i.e., 10,000 colors). On the other hand, most human beings get confused about finding the best harmonizing colors for their attire, and may culminate dowdy.

In this thesis, we investigate compatible color combinations in fashion.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ABBREVIATION	ILLUSTRATION
MSE	Mean Squared Error
AI & ML	Artificial Intelligence and Machine Learning
RGB	Red Green Blue
CV	Computer Vision
PIL	Pillow
HEX	Hexadecimal
IDE	Integrated Development Environment

LIST OF FIGURES

INTRODUCTION

1.1 OVERVIEW

Fashion understanding and analysis have been a popular topic, and it has great value in business. Tasks such as predicting the most popular clothing colors combinations of the next season, making personal recommendations on clothing items, and online clothing retrieval are all of high interest in the fashion industry. In recent years, machine learning techniques have been shown to play a significant role in the fashion field, especially in trend forecasting, interactive search and color-combination recommendations.

Here we formulate an automatic suggestion model for retrieving aesthetic clothing colors using seaborn color pallets. This attempt gives an insight in fashion and shall be influencing how people shop and what they buy.

Three different colors Red, Green and Blue are being tracked by utilizing the fundamentals of computer vision. After successful compilation when we execute the code a window redirects to the image displayed on it whose path is given as an argument. Additionally, we obtain the color name of the pixel along with the composition of three different colors red, blue and green values.

Keywords: seaborn, color-compatibility, color-pallets, fashion field

1.2 MOTIVATION

Among all the features of clothing, such as texture, fabric and shape, color is one of the dominant features. A Canadian study showed that people's first impression on clothing is color, which is instantaneous and can last long. It is investigated that color has a higher influence on the clothing fashion update than some other attributes. To understand which color or which color combination is fashioned in the current season or predict popular colors in next period is much valued in the retail business. Learning the compatibility between colors, it is possible to make recommendations. For example, a person is wearing a red dress and wants to buy a hat, then based on the color combinations learned in fashion, it will be easy and credible to recommend a suitable color of a hat. Such recommendations improve customers' online shopping experiences and help them to find the right product from thousands of items in a shorter time, which is tailored for their styles.

1.3 PROBLEM DEFINITION

Based on the extracted colors, how to build a recommendation system to quantitatively learn color compatibility in fashion and recommend color combinations? To solve these questions, there are some subtasks needed to be taken into consideration. Due to the fact that the data availability in fashion is scarcer, how to achieve a high-quality, large-scale data and make the best use of it is a task of interest. Besides that, how to extract correct colors of each clothing item is also of vital importance

1.4 OBJECTIVE

The aim of this work is to implement ML libraries on color palettes' datasets and develop an application to automatically learn what kind of color combinations of clothing items are popular so that it can be further employed to recommend proper colors. This study investigates color compatibility quantitatively in the fashion field and possibly make color recommendations. To this end, it gives an insight in fashion trends in color as well as improves customers' online shopping experiences.

1.5 PROJECT SCOPE AND LIMITATIONS

A huge amount of color palettes with each palette with almost twenty color shades is imported. A sub-application where RGB coordinates of a clicked pixel on the screen are displayed identifying the exact shade.

Our application achieves state-of-the-art performance and is more generalizable comparing to other models proposed in the fashion field. We have constructed a color recommendation system to learn color compatibility in fashion quantitatively.

Normally, when fashion designers want to pick up some popular colors, they will go through available materials, such as famous magazines or authoritative books in order to gain some insights into the trend. Machine learning techniques serve as a way to automatically find possible patterns in fashion, which saves designers much time and energy. Designers could benefit from the heavy computations done by computers and focus on adjusting the design process based on their own experiences and understanding in fashion. Therefore, fashion trends in colors are always created by humans and it will not be taken over by machine learning techniques, which offers a guideline for color trends but not determines trends.

The primary targets recognized, which show the importance of the subject, are rattled off beneath.

1. Tedious and repeated work of choosing from lakhs of colors by manual method.
2. Wide range of areas to apply such a recommendation system.
3. Rapidly changing fashion trends and the color aesthetics.

1.6 Machine Learning and its types

1 Machine Learning

a. Definition:

Machine learning is a subset of AI, which enables the machine to automatically learn from data, improve performance from past experiences, and make predictions. Machine learning contains a set of algorithms that work on a huge amount of data. Data is fed to these algorithms to train them, and on the basis of training, they build the model & perform a specific task.

b. Types:

1. Supervised Machine Learning

2. Unsupervised Machine Learning

3. Semi-Supervised Machine Learning

4. Reinforcement Learning

c. Application areas:

- Virtual Personal Assistants.
- Predictions while commuting.
- Videos Surveillance.
- Social Media Services.
- Email Spam and Malware Filtering.
- Online Customer Support.
- Search Engine Result Refining.
- Product Recommendations.

2 ML libraries

a. Seaborn:

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It is built on the top of matplotlib library and also closely integrated into the data structures from pandas.

It provides sensible defaults for plots needed for machine learning, and most importantly, the plots are aesthetically better looking than those in Matplotlib. Seaborn uses matplotlib to draw its plots.

The colors stand out, the layers blend nicely together, the contours flow throughout, and the overall package not only has a nice aesthetic quality, but it provides meaningful insights to us as well.

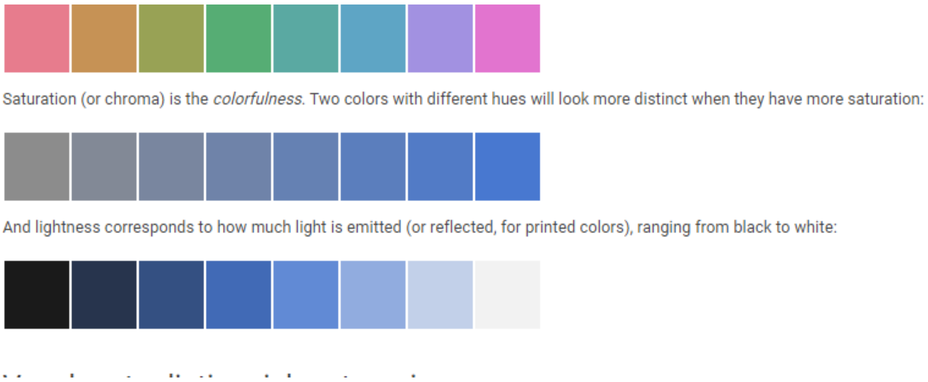
This is quite important in data science where we often work with a lot of messy data. Having the ability to visualize it is critical for a data scientist. Our stakeholders or clients will more often than not rely on visual cues rather than the intricacies of a machine learning model.

There are plenty of excellent Python visualization libraries available, including the built-in matplotlib. But seaborn stands out as it combines aesthetic appeal seamlessly with technical insights.

b. Color Palettes

Because of the way our eyes work, a particular color can be defined using three components. We usually program colors in a computer by specifying their RGB values, which set the intensity of the red, green, and blue channels in a display. But for analysing the perceptual attributes of a color, it’s better to think in terms of hue, saturation, and luminance channels.

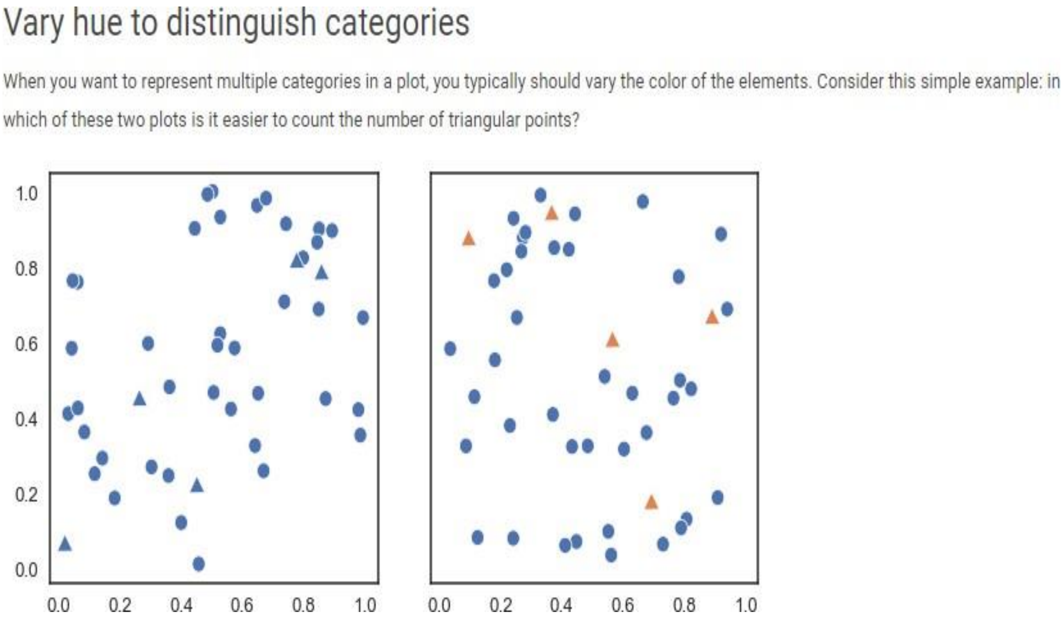
Hue is the component that distinguishes “different colors” in a non-technical sense. It’s property of color that leads to first-order names like “red” and “blue”



Saturation (or chroma) is the *colorfulness*. Two colors with different hues will look more distinct when they have more saturation:

And lightness corresponds to how much light is emitted (or reflected, for printed colors), ranging from black to white:

Fig 1.1: Demonstration of the way hue and saturation vary



Vary hue to distinguish categories

When you want to represent multiple categories in a plot, you typically should vary the color of the elements. Consider this simple example: in which of these two plots is it easier to count the number of triangular points?

Fig 1.2: Vary hue to distinguish categories

In the plot on the right, the orange triangles “pop out”, making it easy to distinguish them from the circles. This pop-out effect happens because our visual system prioritizes color differences.

The blue and orange colors differ mostly in terms of their hue. Hue is useful for representing categories: most people can distinguish a moderate number of hues relatively easily, and points that have different hues but similar brightness or intensity seem equally important. It also makes plots easier to talk about. Consider this example:

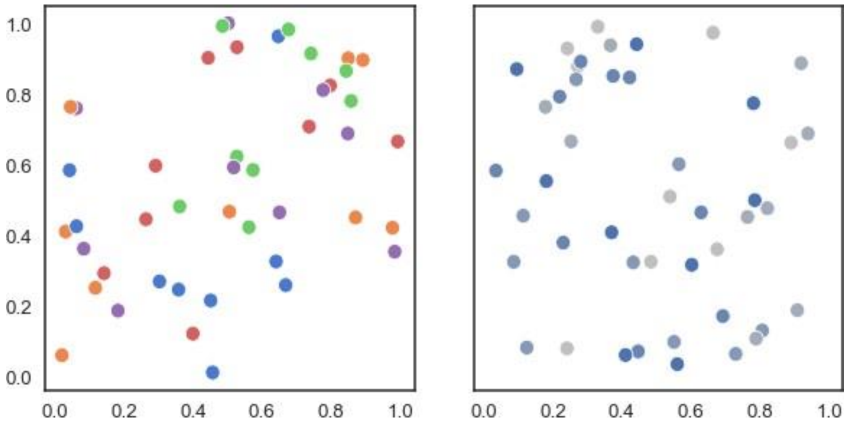


Fig 1.3: Vary hue to distinguish categories

Most people would be able to quickly ascertain that there are five distinct categories in the plot on the left and, if asked to characterize the “blue” points, would be able to do so.

With the plot on the right, where the points are all blue but vary in their luminance and saturation, it’s harder to say how many unique categories are present. And how would we talk about a particular category? “The fairly-but-not-too-blue points?” What’s more, the grey dots seem to fade into the background, de-emphasizing them relative to the more intense blue dots. If the categories are equally important, this is a poor representation.

So as a general rule, use hue variation to represent categories. With that said, here are few notes of caution. If you have more than a handful of colors in your plot, it can become difficult to keep in mind what each one means, unless there are pre-existing associations between the categories and the colors used to represent them. This makes your plot harder to interpret: rather than focusing on the data, a viewer will have to continually refer to the legend to make sense of what is shown. So you should strive not to make plots that are too complex. And be mindful that not everyone sees colors the same way.

Varying both shape (or some other attribute) and color can help people with anomalous color vision understand your plots, and it can keep them (somewhat) interpretable if they are printed to black-and-white.

c. Tools for choosing color palettes

The most important function for working with color palettes is aptly, color_palette() This function provides an interface to most of the possible ways that onecan generate color palettes in seaborn. And it’s used internally by any function that has palette argument.

The primary argument to **color_palette()** is usually a string: either a name of a specific palette or the name of a family and additional arguments to select a specific member. In the latter case, **color_palette()** will delegate to more specific function, such as **cubehelix_palette().** It’s also possible to pass a list of colors specified any way that matplotlib accepts (an RGB tuple, a hex code, or a name in the X11 table). The return value is an object that wraps a list of RGB tuples with a few useful methods, such as conversion to hex codes and a rich HTML representation.

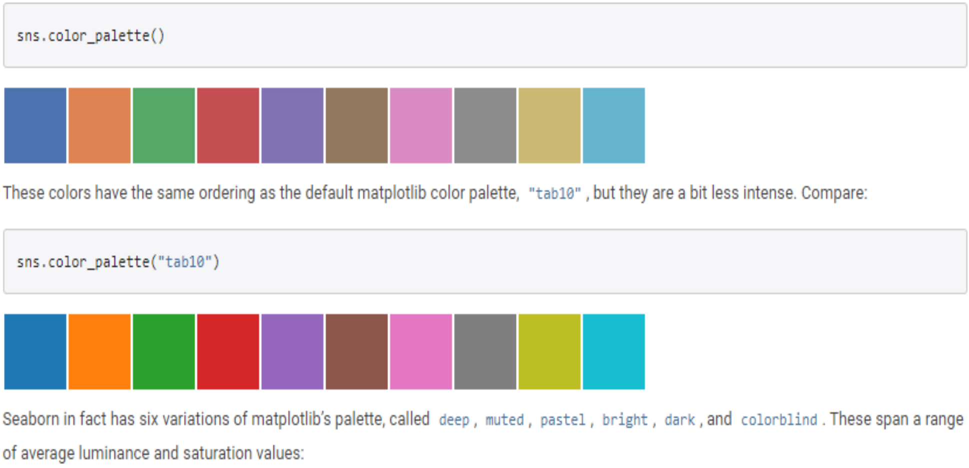
Calling **color_palette()** with no arguments will return the current default color palette that matplotlib (and most seaborn functions) will use if colors are not otherwise specified. This default palette can be set with the corresponding **set_palette()** function, which calls **color_palette()** internally and accepts the same arguments.

To motivate the different options that **color_palette()** provides, it will be useful to introduce a classification scheme for color palettes. Broadly, palettes fall into one of three categories:

- qualitative palettes, good for representing categorical data
- sequential palettes, good for representing numeric data
- diverging palettes, good for representing numeric data with a categorical boundary

d. Qualitative color palettes

Qualitative palettes are well-suited to representing categorical data because most of their variation is in the hue component. The default color palette in seaborn is a qualitative palette with ten distinct hues:



```
sns.color_palette()
```

These colors have the same ordering as the default matplotlib color palette, `"tab10"`, but they are a bit less intense. Compare:

```
sns.color_palette("tab10")
```

Seaborn in fact has six variations of matplotlib's palette, called `deep`, `muted`, `pastel`, `bright`, `dark`, and `colorblind`. These span a range of average luminance and saturation values:

Fig 1.4: Quantitative palettes

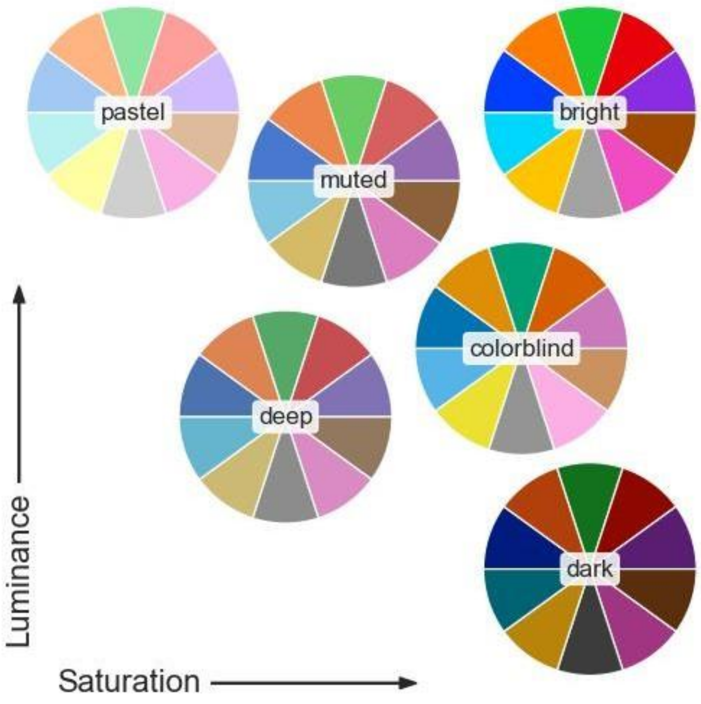


Fig 1.5: Luminance vs Saturation

f. OpenCv

Opencv is an open-source library which is veritably useful for computer vision operations similar as videotape analysis, CCTV footage analysis and image analysis. OpenCV is written by C and has further than optimized algorithms. When we produce operations for computer vision that we don't want to make from scrape we can use this library to start fastening on real world problems. There are numerous companies using this library moment similar as Google, Amazon, Microsoft and Toyota. Numerous experimenters and inventors contribute. We can fluently install it on any Operating System like Windows, Ubuntu and MacOS.

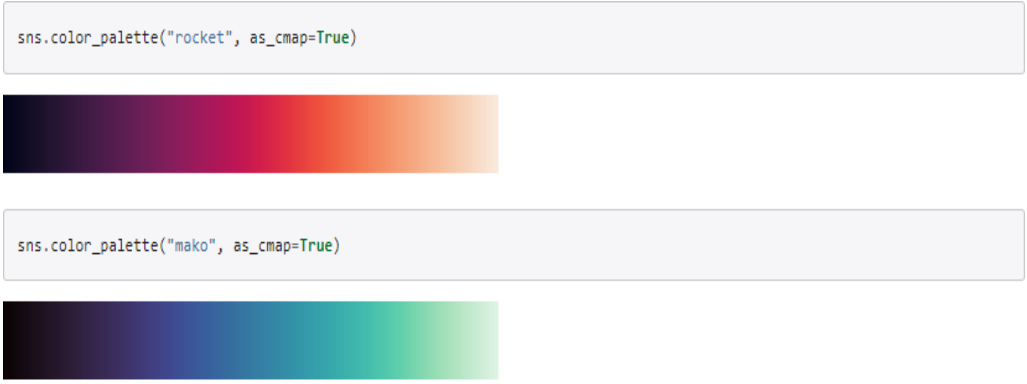
3 Other Color systems

a. Sequential Color palettes

The second major class of color palettes is called “sequential”. This kind of mapping is appropriate when data range from relatively low or uninteresting values to relatively high or interesting values (or vice versa). As we saw above, the primary dimension of variation in a sequential palette is luminance. Some seaborn functions will default to a sequential palette when you are mapping numeric data. (For historical reasons, both categorical and numeric mappings are specified with the hue parameter in functions like relplot() or displot(), even though numeric mappings use color palettes with relatively little hue variation)

b. Perceptually uniform palettes

Because they are intended to represent numeric values, the best sequential palettes will be *perceptually uniform*, meaning that the relative discriminability of two colors is proportional to the difference between the corresponding data values. Seaborn includes four perceptually uniform sequential colormaps: “rocket”,”meko”,” flare” and “crest”. The first two have a very wide luminance range and are well suited for applications such as heatmaps, where colors fill the space, they are plotted into:



```
sns.color_palette("rocket", as_cmap=True)
```

```
sns.color_palette("mako", as_cmap=True)
```

Fig 1.6: Perceptually Uniform palettes

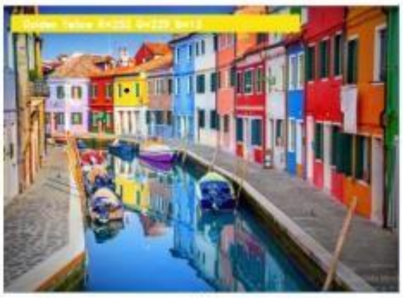
LITERATURE SURVEY

The main objective of the base thesis is to implement an application which is the methodology for identifying the shades of colors with an exact prediction with their names. A study says, a normal human is able to clearly identify nearly 1 million shades of colors. But in the case of humans having enchroma, they would be able to see only 1 percent (i.e.,10,000 colors) from the normal humans. While painting pictures, a painter needs to identify the color patterns exactly or else the reality of the image is not clear. In this paper we defined to get the required color field from an RGB image. In this various step are implemented using openCv platform. Below is the Literature Survey in Tabular form. It gives idea about all the supportive papers.

2.1 **Color Detection of RGB Images Using Python and OpenCv** *(P.Raguraman A.Meghana Y.Navya Sk.Karishma S.Ishwarya)*

The main objective of this paper is the methodology for identifying the shades of colors with an exact prediction with their names.

In this phase, the 3 layered colors are extracted from the input image. All the color images on screens such as televisions, computer, monitors, laptops and mobile screens are produced by the combination of Red, Green and Blue light. Each primary color takes an intensive value 0 (lowest) to 255 (highest). When mixing 3 primary colors at different intensity levels a variety of colors are produced. For Example: If the intensity value of the primary colors is 0, this linear combination corresponds to black. If the intensity value of the primary colors is 1, this linear combination corresponds to white.



(c)

Figure 3: (a) Original input image of Salt Lake (b) Output image with Color intensity RGB values as R=3 G=9 B=97 for Royal Blue (c) Output image with Color intensity RGB values as R=252 G=229 B=13 for Golden Yellow

Fig 2.1: Result from color detection method

2.2 Towards color compatibility in fashion using machine learning *(*Xinhui Wang*)*

In this thesis, we tackle the problem towards compatible color combinations of clothing items in fashion. We separate the task in two parts. First, we employ Deeplab V2 trained on ModaNet dataset to segment clothing items out. Due to the large-scale and high-quality dataset we use for training, our semantic segmentation model achieves state-of-the-art performance comparing to other models proposed in this field, with 0.64 mIoU and 0.96 accuracy of the test set. Then we propose two methods to learn color compatibility and possibly make recommendations. The first method doesn’t work well because of the relatively high loss of the model. Alternatively, the second method item-to-item collaborative f exploits the relationship between color to color and constructs a recommendation system to quantify color compatibility in fashion. Our system makes high-quality color recommendations with a hit-rate of 0.49 for top 5 recommendations.

Deeplab V2 trained on ModaNet dataset, which is a large-scale and high-quality fashion dataset. Having adequate training images makes our semantic segmentation model outperform other proposed models trained on currently common fashion datasets. Secondly, based on the segmentation maps, we employ K-means to extract the dominant color of each garment. It is shown that K-means has a good performance when the clothing item has a simple and pure pattern, while it sometimes doesn’t work well when extracting the color of a garment with complex patterns. Thirdly, after having the color palettes, we first propose matrix factorization to predict the color values of the missing garments in order to make recommendations. However, this method has a relatively high loss because of the sparsity of data and inner connections among R, G, B values of one clothing item. It turns out that using matrix factorization to predict ratings is different from using it to directly predict color values where the values are dependent on others.

2.3 Object color recognition and sorting robot based on OpenCV and machine vision

(Wenbin Zhang, Chengliang Zhang, Chengbin Li, He Zhang)*

This system takes the OpenCV image processing library as the core, uses image processing algorithms such as colorspace conversion, histogram equalization, filtering, Huff circle transformation, and combines the traditional three-axis truss mechanical structure to design an intelligent color recognition analysis Picking robot system. In order to realize the sorting function, the key technologies such as camera calibration, image filtering, object recognition, and positioning are studied. However, there are certain misjudgements of this system in special light and object angles, which need to be improved in the future research process.

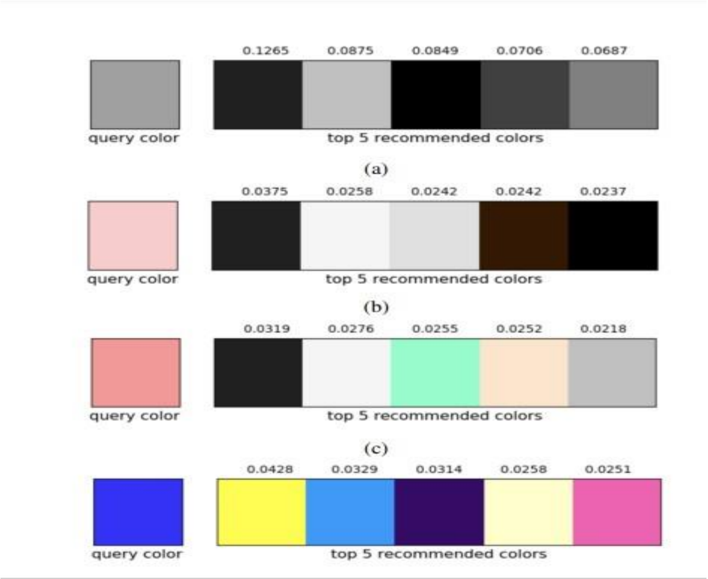


Fig 2.2: Top suggested colors

REQUIREMENT SPECIFICATIONS

INTRODUCTION

This project requires dictionary of seaborn color-palettes and color data-set along with IDE VS Code and Windows 7/8/10. The color detection and recommendation requires a specific operating system, amounts of memory, CPU cores, and disk space.

3. 1 Assumptions and Dependencies

There are three standard ways to implement the proposed application. Thus these three requirements must be fulfilled.

1) Using Seaborn color_palettes

2)Using Seaborn palettes based on varying hue and luminance

3) Dataset with RGB coordinates of about 900 colors

3. 2 Functional Requirements

3. 2. 1 System Feature

The key is that the functional requirements should be created at the undertaking level, which will surely impact the organization's way of life, climate, workers, security prerequisites, existing electronic security frameworks, and clients.

3. 3 System Requirements

The inconsistency discovery programming requires a particular working framework, measures of memory, CPU centres, and plate space. You should stick to specific necessities to execute the irregularity identification framework effectively.

a. Operating System:-

1) A PC running an authorized form of Windows 7/8/9.
2) An authorized form guarantees that the conditions expected for the establishment are settled naturally by the working framework.
3) Google Colab, Jupyter, Vs Code, or any other IDE.

b. Memory and CPU: -

1) Processor: An Intel Core i5 or more
2) RAM: 4 GB or 8 GB

c. Available disk space: -

The application requires minimum 4 GB of free disk space.

d. Permissions: -

Pseudo permissions are required to install the software required to implement a color-combination recommendation system.

3. 4 Database Requirements

A data set of RGB coordinates is required to implement and test the application, which will recommend colors.

SYSTEM DESIGN

4. 1 System Architecture:

At present, there are more than 250 programming languages in existence, according to the TIOBE index. Out of these, Python is one of the most popular programming languages that’s heavily used by developers/practitioners for Machine Learning. However, we can always switch to a language that suits the use case. Now, we’ll look at some of the frameworks that we utilise for various applications. OpenCV: OpenCV-Python is a library of Python bindings designed to solve computer vision problems. It's simple and super easy to use.

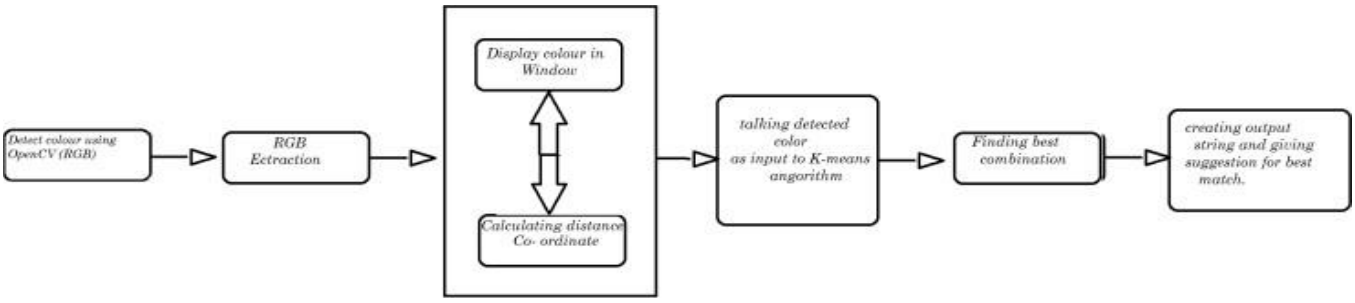

Detect colour using OpenCV (RGB)
RGB Ectraction
Display colour in Window
Calculating distance Co- ordinate
talking detected color as input to K-means angorithm
Finding best combination
creating output string and giving suggestion for best match.

Fig 4.1: System Architecture

4. 2 Data flow diagram:

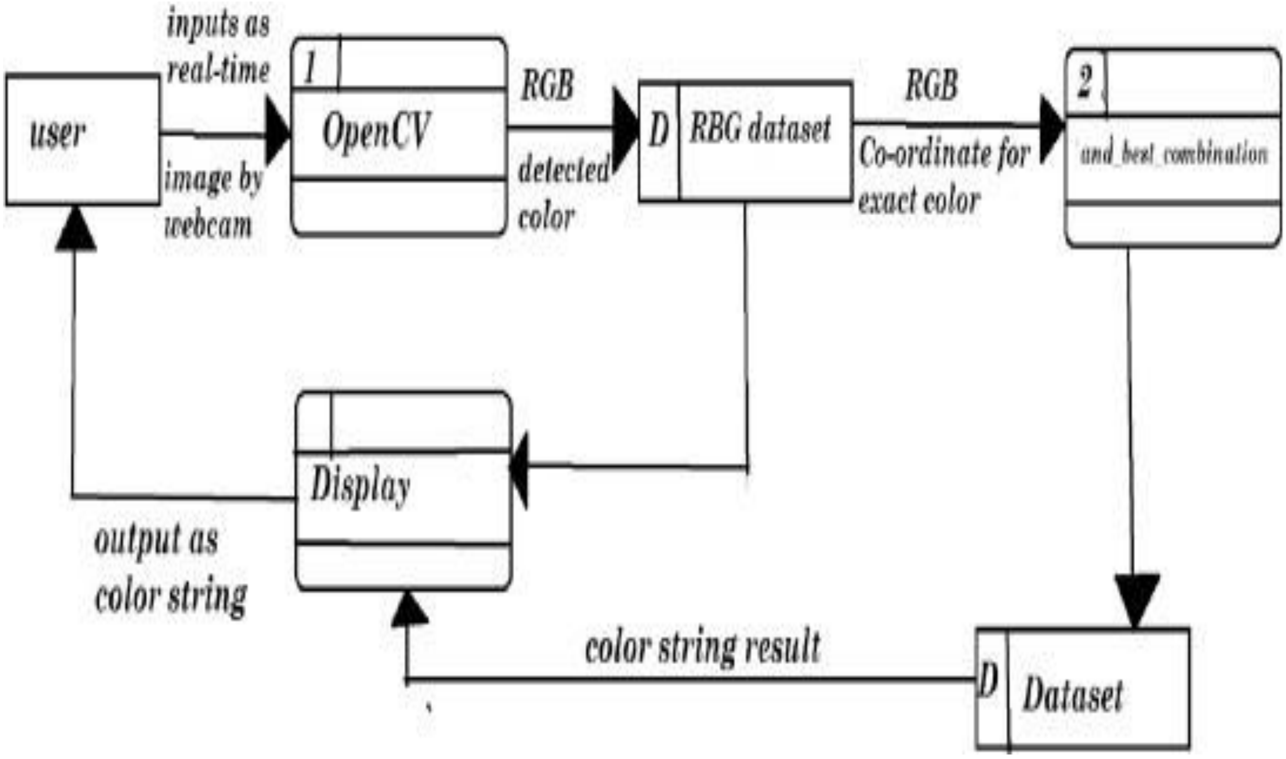


Fig 4.2: Data Flow Diagram

4.3 Framework:

The construction of application and the framework of the application is shown below:

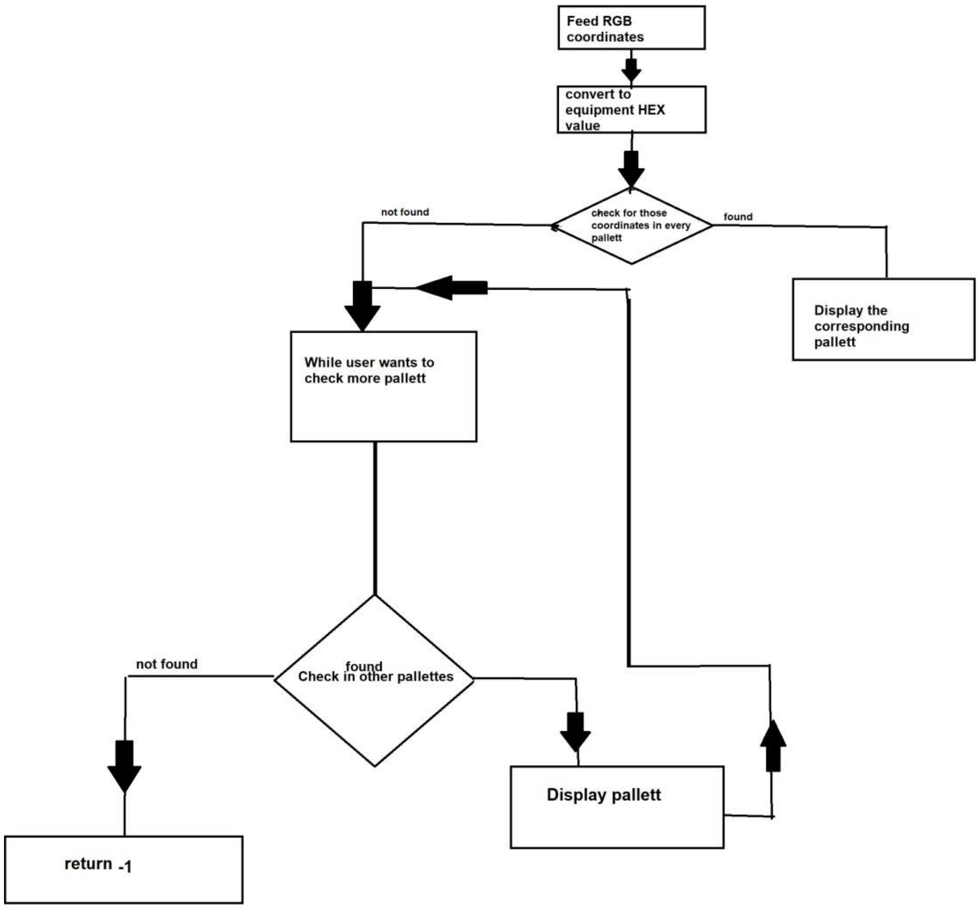

Feed RGB coordinates
convert to equipment HEX value
check for those coordinates in every pallett
not found
found
Display the corresponding pallett
While user wants to check more pallett
Check in other pallettes
not found
found
Display pallett
return -1

Fig 4.3: Framework

4. 3 Analysis Model:

In this project, the following phases are taken into consideration:

1. **Planning:**

   Nowadays, the requirement for a trendy and reliable system is desired by every individual in society. The most traditional system, manual selection, is being implemented everywhere, such as in fashion designing. However, this method requires continuous assessment. Hence to solve this issue, this project is constructed to automate color recommendation.

2. **Analysis:**

   In this project, a data set is considered for the implementaion and testing of the application. The data set in the form of RGB coordinates is further analysed for detecting colors in the provided image.

3. **Implementation:**

   Using OpenCv, extracting RGB coordinates of the pixel clicked by the user. These coordinates are fed to a function and converted to equivalent hexadecimal value. The obtained value is then searched for in all the palettes.

   If found, the entire palette is returned as recommended colours else the user is asked to enter palette name and choose from multiple colors.

4. **Maintenance:**

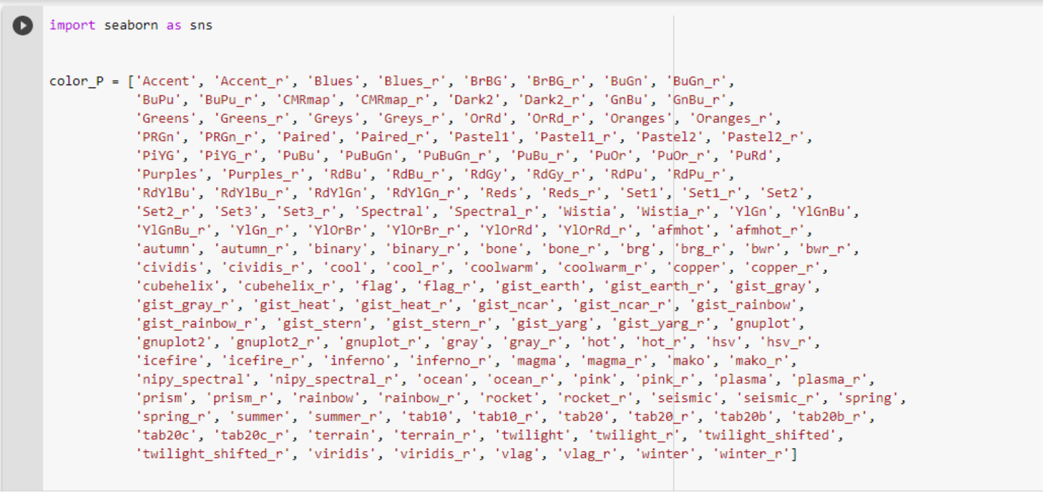
   This project can be used in malls and shopping destinations to help choose aesthetic and trending color combinations. The model should be trained and tested continuously to check its accuracy.

PROJECT PLAN

Sr. no.	Expected Target to be Achieved	First Semester			Second Semester		
		2 months	2 months	2 months	2 months	2 months	2 months
1	Literature Survey	🟨			🟨		
2	Prepare Detail Specification		🟨	🟨			
3	Procurement of equipment and other related material			🟨	🟨		
4	Project Report - Stage 1			🟨			
5	Coding and Development			🟨	🟨	🟨	
6	Project Report - Stage 2 /Research Paper generation					🟨	🟨
7	Consult Expert/Guide			🟨		🟨	🟨

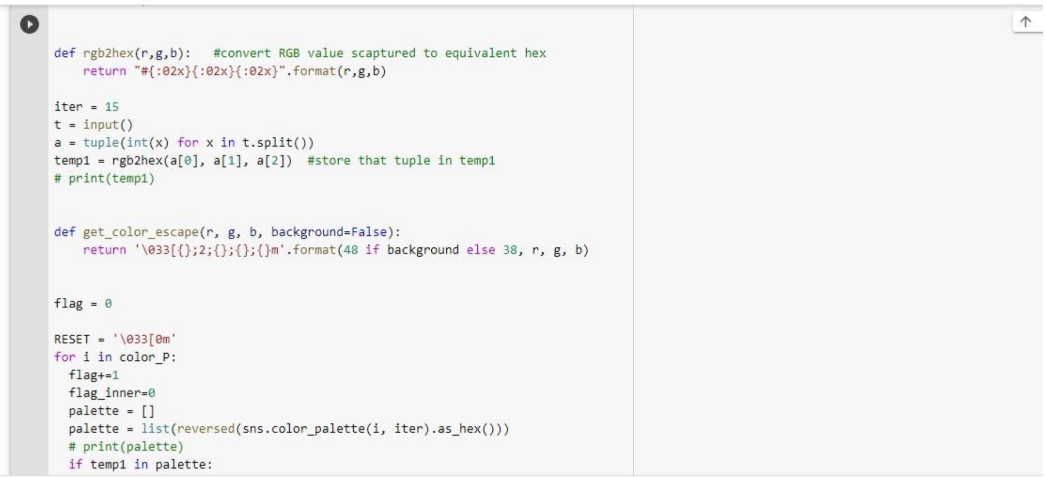
Table 5.1: Project Schedule

CODE SNIPPETS



```python
import seaborn as sns


color_P = ['Accent', 'Accent_r', 'Blues', 'Blues_r', 'BrBG', 'BrBG_r', 'BuGn', 'BuGn_r',
           'BuPu', 'BuPu_r', 'CMRmap', 'CMRmap_r', 'Dark2', 'Dark2_r', 'GnBu', 'GnBu_r',
           'Greens', 'Greens_r', 'Greys', 'Greys_r', 'OrRd', 'OrRd_r', 'Oranges', 'Oranges_r',
           'PRGn', 'PRGn_r', 'Paired', 'Paired_r', 'Pastel1', 'Pastel1_r', 'Pastel2', 'Pastel2_r',
           'PiYG', 'PiYG_r', 'PuBu', 'PuBuGn', 'PuBuGn_r', 'PuBu_r', 'PuOr', 'PuOr_r', 'PuRd',
           'Purples', 'Purples_r', 'RdBu', 'RdBu_r', 'RdGy', 'RdGy_r', 'RdPu', 'RdPu_r',
           'RdYlBu', 'RdYlBu_r', 'RdYlGn', 'RdYlGn_r', 'Reds', 'Reds_r', 'Set1', 'Set1_r', 'Set2',
           'Set2_r', 'Set3', 'Set3_r', 'Spectral', 'Spectral_r', 'Wistia', 'Wistia_r', 'YlGn', 'YlGnBu',
           'YlGnBu_r', 'YlGn_r', 'YlOrBr', 'YlOrBr_r', 'YlOrRd', 'YlOrRd_r', 'afmhot', 'afmhot_r',
           'autumn', 'autumn_r', 'binary', 'binary_r', 'bone', 'bone_r', 'brg', 'brg_r', 'bwr', 'bwr_r',
           'cividis', 'cividis_r', 'cool', 'cool_r', 'coolwarm', 'coolwarm_r', 'copper', 'copper_r',
           'cubehelix', 'cubehelix_r', 'flag', 'flag_r', 'gist_earth', 'gist_earth_r', 'gist_gray',
           'gist_gray_r', 'gist_heat', 'gist_heat_r', 'gist_ncar', 'gist_ncar_r', 'gist_rainbow',
           'gist_rainbow_r', 'gist_stern', 'gist_stern_r', 'gist_yarg', 'gist_yarg_r', 'gnuplot',
           'gnuplot2', 'gnuplot2_r', 'gnuplot_r', 'gray', 'gray_r', 'hot', 'hot_r', 'hsv', 'hsv_r',
           'icefire', 'icefire_r', 'inferno', 'inferno_r', 'magma', 'magma_r', 'mako', 'mako_r',
           'nipy_spectral', 'nipy_spectral_r', 'ocean', 'ocean_r', 'pink', 'pink_r', 'plasma', 'plasma_r',
           'prism', 'prism_r', 'rainbow', 'rainbow_r', 'rocket', 'rocket_r', 'seismic', 'seismic_r', 'spring',
           'spring_r', 'summer', 'summer_r', 'tab10', 'tab10_r', 'tab20', 'tab20_r', 'tab20b', 'tab20b_r',
           'tab20c', 'tab20c_r', 'terrain', 'terrain_r', 'twilight', 'twilight_r', 'twilight_shifted',
           'twilight_shifted_r', 'viridis', 'viridis_r', 'vlag', 'vlag_r', 'winter', 'winter_r']
```



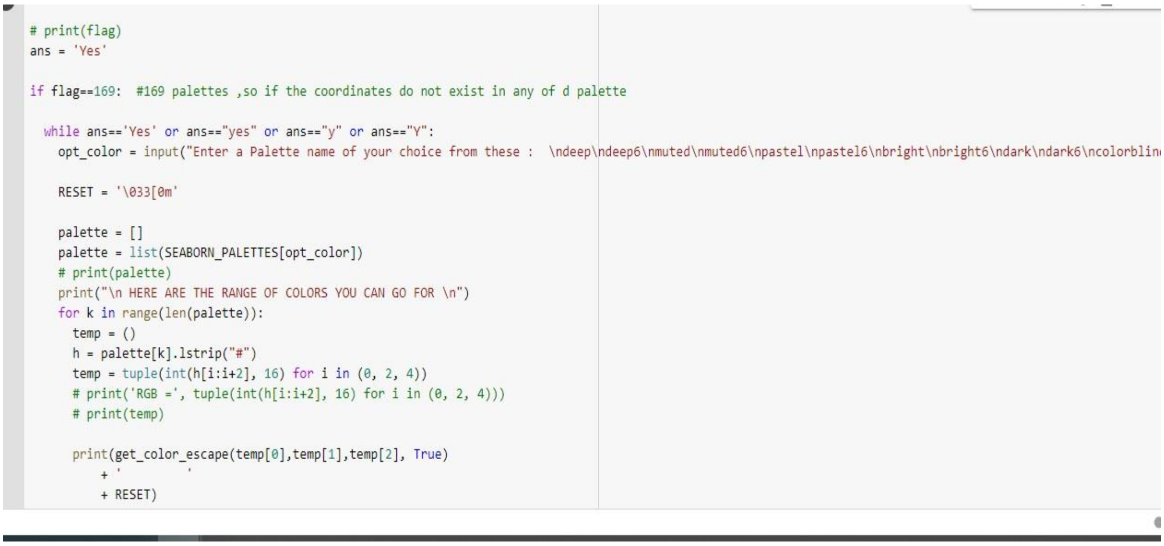
```python
def rgb2hex(r,g,b):   #convert RGB value scaptured to equivalent hex
    return "#{:02x}{:02x}{:02x}".format(r,g,b)

iter = 15
t = input()
a = tuple(int(x) for x in t.split())
temp1 = rgb2hex(a[0], a[1], a[2])  #store that tuple in temp1
# print(temp1)


def get_color_escape(r, g, b, background=False):
    return '\033[{};2;{};{};{}m'.format(48 if background else 38, r, g, b)


flag = 0

RESET = '\033[0m'
for i in color_P:
  flag+=1
  flag_inner=0
  palette = []
  palette = list(reversed(sns.color_palette(i, iter).as_hex()))
  # print(palette)
  if temp1 in palette:
```



```
# print(flag)
ans = 'Yes'

if flag==169:  #169 palettes ,so if the coordinates do not exist in any of d palette

  while ans=='Yes' or ans=="yes" or ans=="y" or ans=="Y":
    opt_color = input("Enter a Palette name of your choice from these :  \ndeep\ndeep6\nmuted\nmuted6\npastel\npastel6\nbright\nbright6\ndark\ndark6\ncolorblin

    RESET = '\033[0m'

    palette = []
    palette = list(SEABORN_PALETTES[opt_color])
    # print(palette)
    print("\n HERE ARE THE RANGE OF COLORS YOU CAN GO FOR \n")
    for k in range(len(palette)):
      temp = ()
      h = palette[k].lstrip("#")
      temp = tuple(int(h[i:i+2], 16) for i in (0, 2, 4))
      # print('RGB =', tuple(int(h[i:i+2], 16) for i in (0, 2, 4)))
      # print(temp)

      print(get_color_escape(temp[0],temp[1],temp[2], True)
          + '          '
          + RESET)
```

OTHER SPECIFICATIONS

7. 1 Advantages

This thesis focuses on automating color compatibility of garments and possibly making color recommendation. We go through the pipeline as shown which is a large-scale and high-quality fashion dataset. Having adequate RGB coordinates makes our color recommendation system outperform other proposed models trained on currently common fashion datasets..
Is cost-effective and scalable: Widely applicable in wide range of areas. As one's business grows, adding and integrating additional dataset and colors into an established one is easy and cost-effective.

7. 2 Limitations

- First, the system detects RGB coordinates from an image and not through the camera, thus making it currently not too reliable.

- Second, not necessary a user wants multiple color recommendations. The proposed system provides multiple color-combinations.

7. 3 Applications

The model can be used with variety of tasks. not only restricted to fashion industry but also industries involving color combination schema and pallet. For instance: 1. Use for Fashion Designing considering colour combination as major attribute 2. For designing of various painting artefacts. 3. Interior designing and decoration for focusing on this model's uses in Fashion industry can be used on a wide scale by Fashion Designers, Social media Influencers, Vloggers, Bloggers, Fashion advisor, and also can be used by any individual struggling to find a perfect match for their piece of attire.

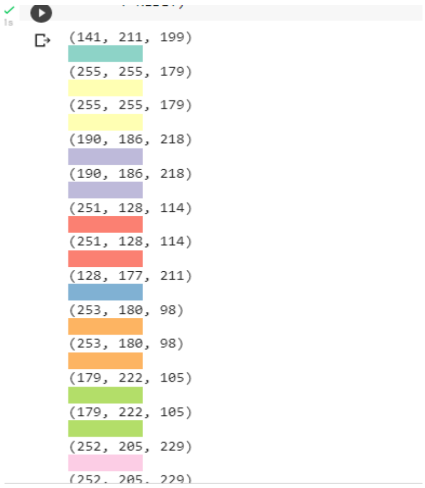
Other applications are, therefore, as follows:
1. Shopping centres

2. Malls

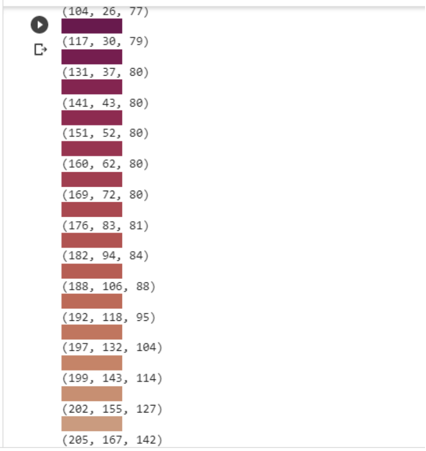
3. Fashion Designing

RESULTS

Output Frames:



```
(141, 211, 199)
(255, 255, 179)
(255, 255, 179)
(190, 186, 218)
(190, 186, 218)
(251, 128, 114)
(251, 128, 114)
(128, 177, 211)
(253, 180, 98)
(253, 180, 98)
(179, 222, 105)
(179, 222, 105)
(252, 205, 229)
(252, 205, 229)
```



```
(104, 26, 77)
(117, 30, 79)
(131, 37, 80)
(141, 43, 80)
(151, 52, 80)
(160, 62, 80)
(169, 72, 80)
(176, 83, 81)
(182, 94, 84)
(188, 106, 88)
(192, 118, 95)
(197, 132, 104)
(199, 143, 114)
(202, 155, 127)
(205, 167, 142)
```

Fig 8.1: Output Frames

CONCLUSION AND FUTURE WORK

The Project is developed using OpenCV a library of Python which provides vision to computers. The detection of color of attire is taken as input by the model and perfect combination for the color is printed in the form of color strings. It is capable of bringing revolution in the field of Fashion Industry introducing various aspects of Machine Learning and Artificial intelligence in it. The base paper used gets us idea about the detection of exact color by finding co-ordinates of color in RGB scale. The developed recommendation model uses color pallet dataset through Machine learning to give the output.

1) Commercial use in fashion malls: If combined with various aspects of IoT and hardware can be used for various Fashion Stores Local as well as Online helping customers to find perfect attire based on color scheme. 2) Social Media impact: Can be integrated with various social media platforms to so to make itself a tool for marketing. 3) Personal use: Various aspects of personal use can be exploited if combined with daily facilities.

In short, the techniques we have opted for can be integrated to achieve a highly reliable model to recommend combinations as well as to set new trends in apparels.

The fashion industry is ever increasing and expanding and the role of this application is an important one.

Appendix A

Plagiarism Report

Table 7.2: Plagiarism Report

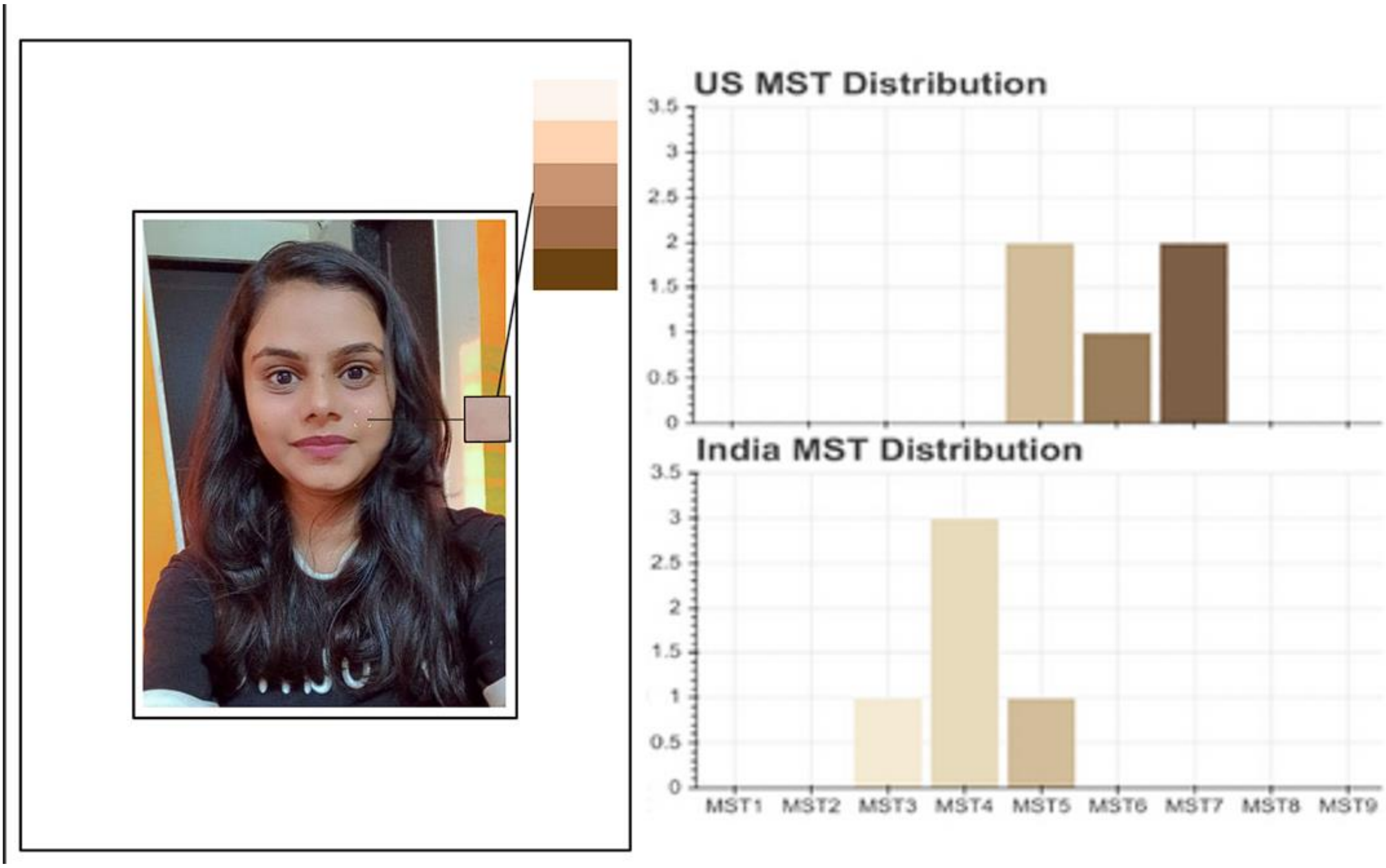
Sr. No	Chapter Name	Plagiarism Percentage
1.	Introduction	2%
2.	Background and literature Survey	7%
3.	Requirement and Analysis	0%
4.	System Architecture	0%
5.	Design	1%
6.	Implementation	0%
7.	Results and Evaluation	0%

Appendix B

Description	Expected Result	Actual Result	Status
Verify that the camera is on	Camera is on	Camera is on successfully	Pass
Verify that the user skin tone is detecting	User skin tone is detected	User skin tone successfully drtected	Pass
Verify that the skin tone should match with color palet	skin tone match with color palet	skin tone is matching with color palet sucessfully	Pass
Verify that the skin tone should extract by rgb extraction	skin tone should extract by rgb extraction	skin tone is extracted by rgb extraction successfully	Pass
Verify that the skin tone is matching with clothes	skin tone is matched with clothes	skin tone is matching with clothes successfully	Pass
display colors of clothes related to the skin tone	display color of clothes related to the skin tone	Successfully display color of clothes related to the skin tone	Pass
display clothes related to the color choosen by user	display clothes related to the color choosen by user	Successfully display clothes related to the color choosen by user	Pass
Gather users feedback	Gather users feedback	Gather users feedback successfully	Pass

Prefomance matrix:

Feature selection



REFERENCES

[1] P. Raguraman1*, A. Meghana2, Y. Navya3, Sk. Karishma4, S. Iswarya5 (Color Detection of RGB Images Using Python and OpenCv)

[2] Ying Li and Anshul Sheopuri (CREATIVE DESIGN OF COLOR PALETTES FOR PRODUCT PACKAGING).

[3] Yuli Liang1 , Seung-Hee Lee1, and Jane E. Workman1(Implementation of Artificial Intelligence in Fashion: Are Consumers Ready?)

[4] Hung-Cheng Tsai, Chia-Young Hung Fei-Kung Hung(Computer Aided Product Color Design with Artificial Intelligence)

[5] G.M. Snoek, Evaluating Color Descriptors for Object and Scene Recognition, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 32, NO. 9, SEPTEMBER 2010

[6] Claudia Nieuwenhuis, Spatially Varying Color Distributions for Interactive Multi Label Segmentation, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL 35, NO. 5, MAY 2013

[7] Kok -Meng Lee, Effects of Classification Methods on Color-Based Feature Detection with Food Processing Applications, IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, VOL. 4, NO. 1, JANUARY 2007

[8] Alexander Toet, Multisource Information and Fusion Architectures Algorithms Applications SPIE 6947 BELLINGHAM WA USA THE INTERNATIONAL SOCIETY FOR OPTICAL ENGINEERING, 1 12, 2008

[9] J. Van de Weijer, Curvature estimation in oriented patterns using curvilinear models applied to gradient vector fields, IEEE TRANS PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 23, N0. 9, PP. 1035- 1042, APRIL 2001.