Instagram Data-Science Project

Instagram

Made by : Yossef Davarashvili

IDEs used:

- **PyCharm (Community edition)**
- **Jupyter Notebook**

Research questions:

1)Does colors have an effect on instagram post behaviour (Likes , Engagement etc) ?
2)Can we predict the amount of the next posts likes by knowing the dominant colors and the amount of followers ?

Visual analytics and marketing platform Curalate took things a bit further and analyzed 8 million Instagram photos to figure out which colors get more likes than others. Mostly-blue images got 24 percent more likes than ones that had red as the most prominent color.

Marissa Laliberte

Updated: Jun. 11, 2017

Steps:

Data crawling->Data scrubbing->EDA and Visualization ->Exploring and modeling Data-> ML

Web crawling - collecting and obtaining data

Data cleaning , removing duplicates and na values.

Data cleaning , removing duplicates , na values , outliers etc

Visualizing and deeply understanding the data

Getting a conclusion and model predicting

Step 1 : Data Crawling

Modules used:
selenium -
logging instagram via selenium webdriver.
Searching and getting links to posts that are related to the Photography & Travel niche.

colorgram -
For every photo - exports the 10 most dominant colors in RGB .

instascrape - For getting the follower/like/comment counts on each post.

Python Imaging Library (PIL) - Getting the colors from each photo without needing to save them.

Step 1 : Data Crawling

Step 1 : Collecting links to posts :

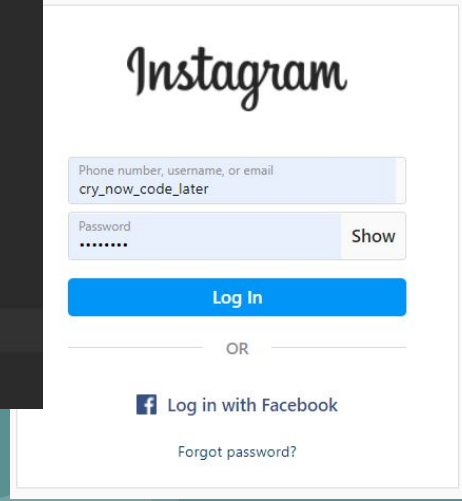
1) **Searching words and profiles related to “photography” and “traveling”**
2) **Saving the links to scrape them.**

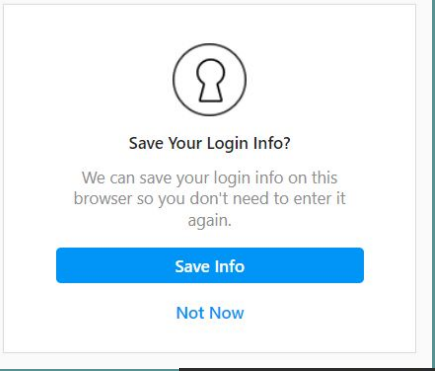
In this step I used selenium web driver to bypass login and “not now” messages when logging to instagram.

```
def post_links_crawler(words_to_search):
    '''Gets a list of words/acounts/hashtags to search and get the link to the posts on
    that particular page ---> return a list of the links
    using selenium '''
```

```
user_name = WebDriverWait(driver, 10).until(
    expected_conditions.element_to_be_clickable((By.CSS_SELECTOR, "input[name='username']")))
password = WebDriverWait(driver, 10).until(
    expected_conditions.element_to_be_clickable((By.CSS_SELECTOR, "input[name='password']")))

user_name.clear()
password.clear()
user_name.send_keys(user_name_2)
password.send_keys(password_2)
```






```
login_button = WebDriverWait(driver, 10).until(
    expected_conditions.element_to_be_clickable((By.CSS_SELECTOR, "button[type='submit']"))).click()
not_now_bottun = WebDriverWait(driver, 10).until(
    expected_conditions.element_to_be_clickable((By.XPATH, "//button[contains(text(),'Not Now')]"))).click()
not_now_bottun_2 = WebDriverWait(driver, 10).until(
    expected_conditions.element_to_be_clickable((By.XPATH, "//button[contains(text(),'Not Now')]"))).click()
```

Step 1 : Data Crawling

Step 2: Crawling the data and building Data-Set:

1) This step is separated to 1,2,...,8 chunks of links in order to bypass account/IP blocking from instagram .
2) For every link I scraped the data needed .
3) data_insta_files - folder with 8 data frames . And the we concatenated them to 1 Data-Set.
4) Every photo of the post gone through COLORGRAM module in order to get the 10 most dominant colors in that photo .

```
links_chunk1 = postlinks[0:500]
links_chunk2 = postlinks[501:1000]
links_chunk3 = postlinks[1001:1500]
links_chunk4 = postlinks[1501:2000]
links_chunk5 = postlinks[2001:2500]
links_chunk6 = postlinks[2501:3000]
links_chunk7 = postlinks[3001:3500]
links_chunk8 = postlinks[3501:4305]
```

```
def make_and_save_df(data_links, file_to_save):
    '''gets a chunks 1,2,3...,8 of links to posts , and make a df from the data we want
    using the instascrape to take likes and comment and photos ,
     the mudole colorgram takes the 10 domminant colors in that photo
     and then saves the df '''
```

Step 1 : Data Crawling

*At this part Instagram blocked my IP , and even banned one account i opened - “cry_now_code_later” because of too many requests (4305 links) . So a new account was needed “code_now_cry_later” .

And more than that - I needed to bypass the need to login every time i request a new post to crawl through.
To achieve that i had to copy a code (Below) in order to send “session_id” , - thats a Web site's server assigns a specific user for the duration of that user's visit (session). Shortly - Cookie .

```
session_id = '50837202909%3AwZz6MuC4fhu2yi%3A10'
headers = {
    "user-agent": "Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Mobile Safari/537.36 Edg/87.0.664.57",
    "cookie": f"sessionid={session_id};"}
```

Copied from stackoverflow

Step 2 : Data Scrubbing

- Dealing with NaN values .
- Dealing with duplicates .
- Clustering the RGB colors with KMeans Clustering Algorithm - to handle and use colors for ML models , we can't keep colors as RGB tuples .

```
def cleanDataFrame():
    '''cleans duplicates , wrong values etc (DATA CLEANING)'''
    df = pandas.read_csv('instaDataFinal.csv')
    df.drop_duplicates(inplace=True)
    df.dropna(axis=0)
    df.drop(df[df.likes < 1].index, inplace=True)
    df.to_csv('instaDataCleanedAndReady.csv')
```

```
# df = pandas.read_csv('instaDataCleanedAndReady.csv')
# df = df.copy(deep=True)
#
# colorCols = [f'color{i}' for i in range(1,11)]
# allPossibleColors = np.zeros((len(colorCols) * df.shape[0], 3))
#
# matRowIdx = 0
# for rowIdx, row in df.iterrows():
#     for col in colorCols:
#         RGBCurr = re.findall("[0-9]+", row[col])
#         for colIdx in [0,1,2]:
#             allPossibleColors[matRowIdx, colIdx] = float(RGBCurr[colIdx])
#         matRowIdx +=1
#
# kmeans = KMeans(n_clusters=numberOfMainColorsNeeded)
# kmeans.fit(allPossibleColors)
#
# def predictUsingKmeans(x):
#     RGBCurr = re.findall("[0-9]+", x)
#     return kmeans.predict(np.array([int(RGBCurr[0]), int(RGBCurr[1]), int(RGBCurr[2])]).reshape(1,3))[0]
```

Data Frame:

```
for idx in range(0,10):
    KMeans.cluster_centers_[idx]
```

Likes - Amount of likes on the post .
Followers - Amount of followers of the post publisher.
Comments - Amount of comments on the post .
Color 1 - The most dominant color in the photo.
Color 2 - 2nd most dominant color in the photo.
Color 3 - 3rd most dominant color in the photo.
Color 4 - 4th most dominant color in the photo.
Color 5 - 5th most dominant color in the photo.
Color 6 - 6th most dominant color in the photo.
Color 7 - 7th most dominant color in the photo.
Color 8 - 8th most dominant color in the photo.
Color 9 - 9th most dominant color in the photo.
Color 10 - 10th most dominant color in the photo.

Colors map :
0 -[234, 234, 234]
1 - [97, 94 , 91]
2 - [150, 158, 167]
3 - [30, 26 , 25]
4 - [65, 125, 168]
5 - [39, 58, 73])
6 - [177, 125, 74]
7 - [204 , 205, 205]
8 - [113, 63, 40]
9 - [194, 162 , 127]

Data Frame :

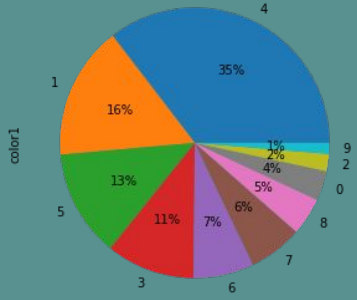
	likes	followers	comments	color1	color2	color3	color4	color5	color6	color7	color8	color9	color10
0	187	1290	1	7	6	4	0	6	0	1	9	2	7
1	306	761	5	3	7	8	8	7	2	2	1	1	1
2	487	2995	3	3	0	4	7	6	0	7	9	7	8
3	409	1904	4	4	6	5	7	4	3	2	9	3	7
4	420	2032	5	4	0	5	9	6	0	0	6	4	7
...	...	...	...	...	...	...	...	...	...	...	...	...	...
3612	2384	18597	59	6	1	8	4	5	4	4	7	3	3
3613	488	3169	33	7	5	3	3	2	9	4	3	5	4
3614	4110	17541	198	6	5	4	7	9	3	6	7	4	2
3615	635	5481	40	8	4	6	8	6	3	1	9	1	5
3616	156	1160	5	5	4	1	6	3	5	7	1	1	9

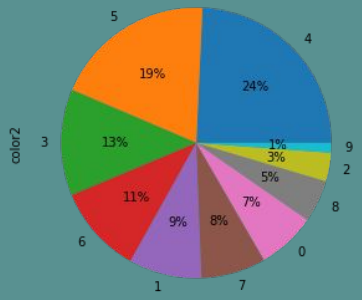
At this point - once Ive got the full DataFrame , I moved from coding in PyCharm , to Jupyter Notebook in order to facilitate the work.

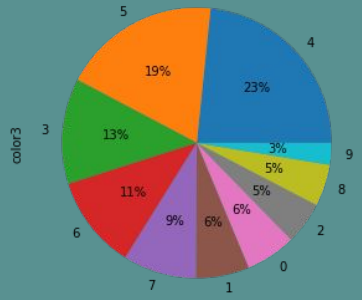
After collecting the data we want to explore -
There are 13 column : Likes , Followers , Comments and 10 columns of the most dominant colors in the photo .

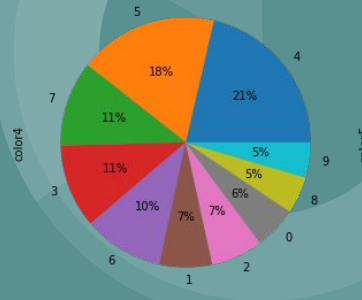
EDA & Visualization

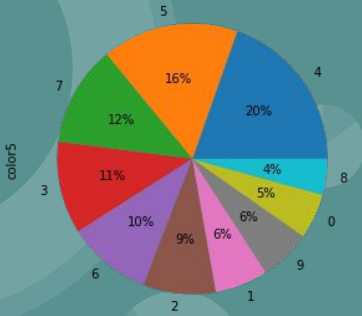
Common colors for each color raw (1-10):

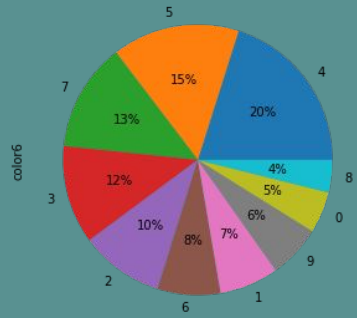


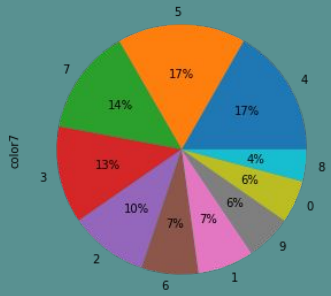


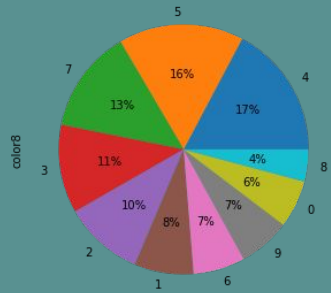


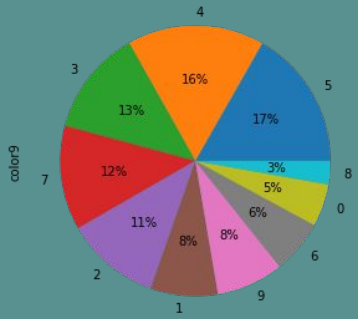


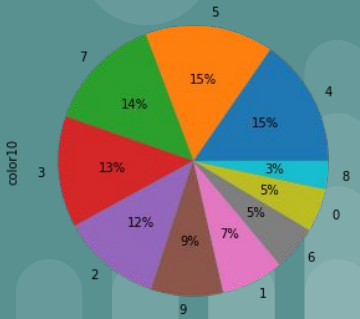






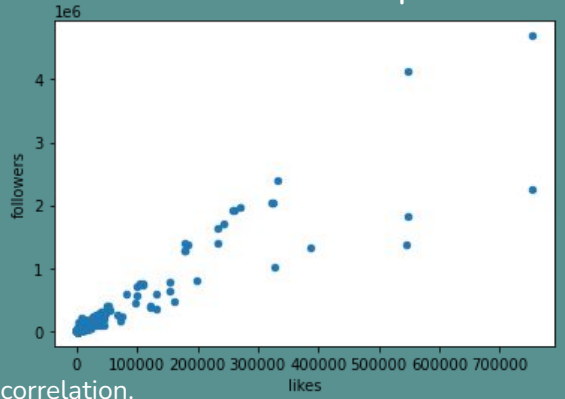






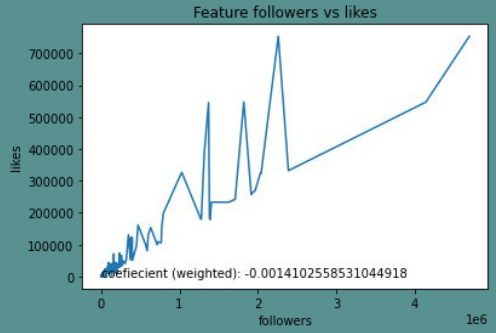
EDA & Visualization

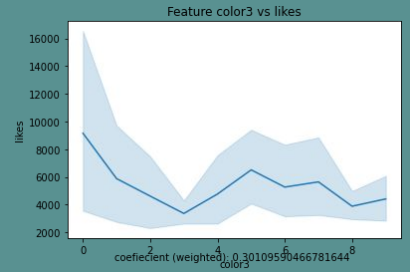
At this point - The correlation between the colors and likes is really low.
But once we add followers and comments , we can see a positive correlation that can lead to prediction.

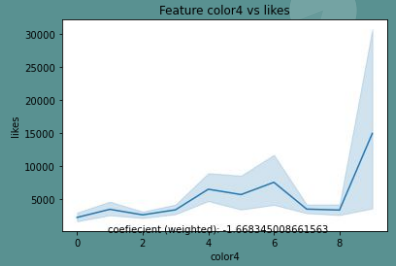


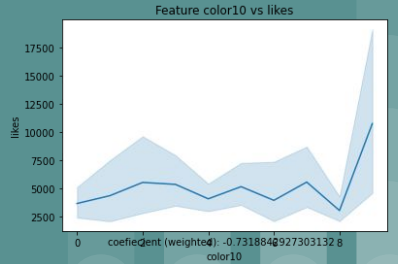
High correlation.

	likes	followers	comments	color1	color2	color3	color4	color5	color6	color7	color8	color9	color10
likes	1.000000	0.932621	0.592144	-0.010440	0.012289	-0.012469	0.048353	0.015004	-0.012543	0.011445	-0.004087	-0.000610	0.032257
followers	0.932621	1.000000	0.599525	-0.003057	0.016058	-0.010640	0.038682	0.015634	-0.008070	0.010359	0.001717	0.004145	0.026604
comments	0.592144	0.599525	1.000000	-0.020911	-0.005620	0.011502	0.018401	0.007139	0.022406	-0.000061	-0.004807	0.012194	0.040093
color1	-0.010440	-0.003057	-0.020911	1.000000	-0.027893	0.013363	0.016867	0.065288	0.039669	0.032818	0.052512	0.049666	0.029935
color2	0.012289	0.016058	-0.005620	-0.027893	1.000000	-0.065682	0.056801	0.004018	0.033865	0.047878	0.037345	-0.008931	0.022351
color3	-0.012469	-0.010640	0.011502	0.013363	-0.065682	1.000000	-0.004129	-0.008345	0.021101	0.006090	0.027965	0.043749	0.026349
color4	0.048353	0.038682	0.018401	0.016867	0.056801	-0.004129	1.000000	-0.025436	-0.001608	0.007344	0.017904	0.011521	0.015519
color5	0.015004	0.015634	0.007139	0.065288	0.004018	-0.008345	-0.025436	1.000000	-0.041403	-0.049987	-0.021851	0.023066	0.009826
color6	-0.012543	-0.008070	0.022406	0.039669	0.033865	0.021101	-0.001608	-0.041403	1.000000	-0.002115	-0.031236	-0.026557	0.020857
color7	0.011445	0.010359	-0.000061	0.032818	0.047878	0.006090	0.007344	-0.049987	-0.002115	1.000000	0.004675	-0.029843	-0.016044
color8	-0.004087	0.001717	-0.004807	0.052512	0.037345	0.027965	0.017904	-0.021851	-0.031236	0.004675	1.000000	-0.007022	-0.006295
color9	-0.000610	0.004145	0.012194	0.049666	-0.008931	0.043749	0.011521	0.023066	-0.026557	-0.029843	-0.007022	1.000000	-0.018018
color10	0.032257	0.026604	0.040093	0.029935	0.022351	0.026349	0.015519	0.009826	0.020857	-0.016044	-0.006295	-0.018018	1.000000









Low correlation.

ML and conclusion :

Splitting the Data-Frame to Xtrain , Ytrain , Xtest , Ytest.
In total we have 3617 Rows :

```
In [38]: df.dropna(axis=0)
         dataA = df[featureCols]
         Y = df['likes']
         Xtrain = dataA[:3000]
         Xtest = dataA[3000:]
         Ytrain = Y[:3000]
         Ytest = Y[3000:]
```

After training and testing ML Algorithms , The best score came from a Linear Regression .

```
MSE (train) for linear regression is = 2104.2259589317277
MSE (test) for linear regression is = 1117.6064880489052
```

The colors were not enough to predict a accurate likes prediction , but once we added the “Follower” and “comments” to the feature columns we had a better correlation and score .

ML and conclusion :

Unfortunately the best score I've been able to achieve (decreasing amount of K-Means of the colors to 10 & adding "followers" and "comment" to the feature cols) :

```
In [42]: R2_scr = sklearn.metrics.r2_score(Ytest, Ypred)
         print(f"R2 Score For LinearRegMod : {R2_scr}")

         R2 Score For LinearRegMod : 0.48920570400396124
```

At this point - i wanted to check the actual numbers to have a better understanding :
I wanted to see the error on different thresholds of the likes (100,300,500,.....,9900).

And then i found out the following conclusion :
The error for predicting the likes amount is between 520-660.

So lets talk numbers :

ML and conclusion :

After couple google searches of how much percentage of errors is ok for data-science , i assumed average of 7.5% top is ok .

```
error (abs) for test set for threshold 100 Likes for linear regression is = 516.187711690036
error (abs) for test set for threshold 300 Likes for linear regression is = 505.1161737013218
error (abs) for test set for threshold 500 Likes for linear regression is = 508.5282095502067
```

Above , we can see that the error is too high , more then 100% !

```
error (abs) for test set for threshold 6700 Likes for linear regression is = 590.2770484735718
error (abs) for test set for threshold 6900 Likes for linear regression is = 591.7948546938463
error (abs) for test set for threshold 7100 Likes for linear regression is = 595.0067802360655
```

But as the popularity of the post (High amount of Followers/Likes - the error percentage decreases to ~14% error)

```
error (abs) for test set for threshold 9500 Likes for linear regression is = 661.3892342275961
error (abs) for test set for threshold 9700 Likes for linear regression is = 661.3892342275961
error (abs) for test set for threshold 9900 Likes for linear regression is = 661.3892342275961
```

But once we passing “Very viral” content - 10K likes and above .
The error rate drops to ~5% .

ML and conclusion :

Final Conclusion :
1- Predicting amount of likes only by dominant colors is not possible , we need more data.

2- Once we add more data - Followers and comments , we can predict amount of likes ,
But , The prediction will be accurate in case we have a account with over 10K followers .

3- Colors have a strong psychological effect , By hundreds of researches ;
But when it comes to Social-Networking - colors are not a big effect on popularity and virality . The history of likes-comments-followers on each photo have a bigger influence.

4- More data of posts for specific Profiles and their engagement will predict a better results and more accurate ML models , in comparing to searching randomly In my opinion .