MACHINE LEARNING ASSIGNMENT-1

1. In this set of practice exercises we'll be looking at a cool dataset of real passwords (made available from actual data breaches) sourced and compiled from Information is Beautiful and contributed to R's Tidy Tuesday project. These passwords are common ("bad") passwords that you should avoid using! But we're going to use this dataset to practice some regex skills.

➢
```
In [1]: import pandas as pd
```

2. The dataset has the following columns:

w	class	description
Rank	int	popularity in their database of released passwords
password	str	Actual text of the password
category	str	What category does the password fall in to?
value	float	Time to crack by online guessing
time_unit	str	Time unit to match with value
offline_crack_sec	float	Time to crack offline in seconds
rank_alt	int	Rank 2
strength	int	Strength = quality of password where 10 is highest, 1 is lowest, please note that these are relative to these generally bad passwords
font_size	int	Used to create the graphic for KIB

In these exercises, we're only interested in the password, value and time_unit columns so import only these two columns as a dataframe named df from this url: https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2020/2020-01-14/passwords.csv

➢
```
In [4]: import pandas as pd
df = pd.read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2020/2020-01-14/passwords.csv',
                 usecols=['password', 'value', 'time_unit'],
                 skipfooter = 7,
                 engine='python')
df.head()
```

Out[4]:

	password	value	time_unit
0	password	6.91	years
1	123456	18.52	minutes
2	12345678	1.29	days
3	1234	11.11	seconds
4	qwerty	3.72	days

3. An online password attack is when someone tries to hack your account by simply trying a very large number of username/password combinations to access your account. For each password in our dataset, the value column shows the amount of time it is estimated to take an “online password attack” to hack your account. The column time_unit shows the units of that time value (e.g., hours, days, years, etc.)

It would be much nicer if our values were of the same units so we can more easily compare the “online password guessing time” for each password. So your first task is to convert all of the values to units of hours (assume the conversion units I’ve provided below, e.g., 1 day is 24 hours, 1 week is 168 hours, etc).

```
units = {
"seconds": 1 / 3600,
"minutes": 1 / 60,
"days": 24,
"weeks": 168,
“months": 720,
"years": 8760,
}
```

In [5]:
```
units = {
    "seconds": 1 / 3600,
    "minutes": 1 / 60,
    "days": 24,
    "weeks": 168,
    "months": 720,
    "years": 8760,
}

for key, val in units.items():
    df.loc[df['time_unit'] == key, 'value'] *= val

df['time_unit'] = 'hours'
df.head()
```

Out[5]:

	password	value	time_unit
0	password	60531.600000	hours
1	123456	0.308667	hours
2	12345678	30.960000	hours
3	1234	0.003086	hours
4	qwerty	89.280000	hours

4. How many password begin with the sequence 123?

```
In [6]: df['password'].str.contains(r"^123").sum()
Out[6]: 9
```

5. What is the average time in hours needed to crack these passwords that begin with 123? How does this compare to the average of all passwords in the dataset?

```
In [7]: print(f"Avg. time to crack passwords beginning with 123: {df[df['password'].str.contains(r'^123')]['value'].mean():.0f} hrs")
        print(f"Avg. time to crack for all passwords in dataset: {df['value'].mean():.0f} hrs")

        Avg. time to crack passwords beginning with 123: 107 hrs
        Avg. time to crack for all passwords in dataset: 13918 hrs
```

6. How many passwords do not contain a number?

```
In [8]: df[df['password'].str.contains(r"^[^0-9]*$")].head()
Out[8]:
```

	password	value	time_unit
0	password	60531.60	hours
4	qwerty	89.28	hours
6	dragon	89.28	hours
7	baseball	60531.60	hours
8	football	60531.60	hours

7. How many passwords contain at least one number?

```
In [9]: df[df['password'].str.contains(r".*[0-9].*")].head()
Out[9]:
```

	password	value	time_unit
1	123456	0.308667	hours
2	12345678	30.960000	hours
3	1234	0.003086	hours
5	12345	0.030833	hours
11	696969	0.308667	hours

8. Is there an obvious difference in online cracking time between passwords that don't contain a number vs passwords that contain at least one number?

```
In [11]: print(f"Avg. time to crack passwords without a number: {df[df['password'].str.contains(r'^[^0-9]*$')]['value'].mean():.0f} hrs")
         print(f"Avg. time to crack passwords with at least one number: {df[df['password'].str.contains(r'.*[0-9].*')]['value'].mean():.0
```

```
Avg. time to crack passwords without a number: 8095 hrs
Avg. time to crack passwords with at least one number: 62005 hrs
```

9. How many passwords contain at least one of the following punctuations: [.!?\\-] (hint: remember this dataset contains *weak* passwords…)?

```
In [12]: df[df['password'].str.contains(r'[.!?\\-]')]
```

Out[12]:

password	value	time_unit

10. Which password(s) in the datasets took the shortest time to crack by online guessing? Which took the longest?

```
In [13]: df.query("value == value.min()")
```

Out[13]:

	password	value	time_unit
3	1234	0.003086	hours
19	2000	0.003086	hours
44	6969	0.003086	hours
76	1111	0.003086	hours
276	5150	0.003086	hours
314	2112	0.003086	hours
315	1212	0.003086	hours
324	7777	0.003086	hours
371	2222	0.003086	hours
373	4444	0.003086	hours
429	1313	0.003086	hours