```
import numpy as np
import pandas as pd
# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter)
will list the files in the input directory
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

df= pd.read_csv("pseudo_facebook.csv")
df.head()
```

```
    userid  age  dob_day  dob_year  dob_month  gender  tenure
friend_count  \
0  2094382   14       19      1999         11    male   266.0
0
1  1192601   14        2      1999         11  female     6.0
0
2  2083884   14       16      1999         11    male    13.0
0
3  1203168   14       25      1999         12  female    93.0
0
4  1733186   14        4      1999         12    male    82.0
0

   friendships_initiated  likes  likes_received  mobile_likes  \
0                      0      0               0             0
1                      0      0               0             0
2                      0      0               0             0
3                      0      0               0             0
4                      0      0               0             0

   mobile_likes_received  www_likes  www_likes_received
0                      0          0                   0
1                      0          0                   0
2                      0          0                   0
3                      0          0                   0
4                      0          0                   0
```

```
 df.describe()
```

```
             userid           age       dob_day      dob_year
dob_month  \
count  9.900300e+04  99003.000000  99003.000000  99003.000000
99003.000000
mean   1.597045e+06     37.280224     14.530408   1975.719776
6.283365
std    3.440592e+05     22.589748      9.015606     22.589748
3.529672
min    1.000008e+06     13.000000      1.000000   1900.000000
1.000000
```

```
25%    1.298806e+06     20.000000      7.000000   1963.000000
3.000000
50%    1.596148e+06     28.000000     14.000000   1985.000000
6.000000
75%    1.895744e+06     50.000000     22.000000   1993.000000
9.000000
max    2.193542e+06    113.000000     31.000000   2000.000000
12.000000

             tenure  friend_count  friendships_initiated         likes
\
count  99001.000000  99003.000000           99003.000000  99003.000000

mean     537.887375    196.350787             107.452471    156.078785

std      457.649874    387.304229             188.786951    572.280681

min        0.000000      0.000000               0.000000      0.000000

25%      226.000000     31.000000              17.000000      1.000000

50%      412.000000     82.000000              46.000000     11.000000

75%      675.000000    206.000000             117.000000     81.000000

max     3139.000000   4923.000000            4144.000000  25111.000000


       likes_received  mobile_likes  mobile_likes_received
www_likes  \
count    99003.000000  99003.000000           99003.000000
99003.000000
mean       142.689363    106.116300              84.120491
49.962425
std       1387.919613    445.252985             839.889444
285.560152
min          0.000000      0.000000               0.000000
0.000000
25%          1.000000      0.000000               0.000000
0.000000
50%          8.000000      4.000000               4.000000
0.000000
75%         59.000000     46.000000              33.000000
7.000000
max     261197.000000  25111.000000          138561.000000
14865.000000

       www_likes_received
count        99003.000000
```

```
mean            58.568831
std            601.416348
min              0.000000
25%              0.000000
50%              2.000000
75%             20.000000
max         129953.000000
```

```
features = df.columns
features
```

```
Index(['userid', 'age', 'dob_day', 'dob_year', 'dob_month', 'gender',
'tenure',
       'friend_count', 'friendships_initiated', 'likes',
'likes_received',
       'mobile_likes', 'mobile_likes_received', 'www_likes',
       'www_likes_received'],
      dtype='object')
```

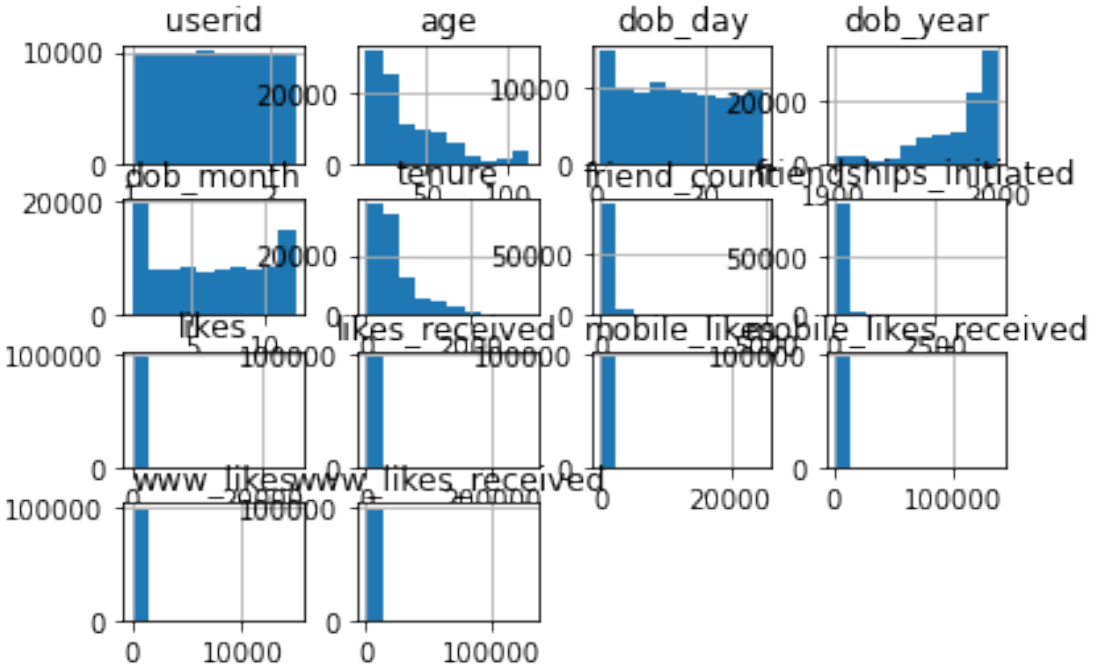
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99003 entries, 0 to 99002
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   userid                  99003 non-null  int64
 1   age                     99003 non-null  int64
 2   dob_day                 99003 non-null  int64
 3   dob_year                99003 non-null  int64
 4   dob_month               99003 non-null  int64
 5   gender                  98828 non-null  object
 6   tenure                  99001 non-null  float64
 7   friend_count            99003 non-null  int64
 8   friendships_initiated   99003 non-null  int64
 9   likes                   99003 non-null  int64
 10  likes_received          99003 non-null  int64
 11  mobile_likes            99003 non-null  int64
 12  mobile_likes_received   99003 non-null  int64
 13  www_likes               99003 non-null  int64
 14  www_likes_received      99003 non-null  int64
dtypes: float64(1), int64(13), object(1)
memory usage: 11.3+ MB
```

```
df.hist()
```

```
array([[<AxesSubplot:title={'center':'userid'}>,
        <AxesSubplot:title={'center':'age'}>,
        <AxesSubplot:title={'center':'dob_day'}>,
        <AxesSubplot:title={'center':'dob_year'}>],
       [<AxesSubplot:title={'center':'dob_month'}>,
```

```
       <AxesSubplot:title={'center':'tenure'}>,
       <AxesSubplot:title={'center':'friend_count'}>,
       <AxesSubplot:title={'center':'friendships_initiated'}>],
      [<AxesSubplot:title={'center':'likes'}>,
       <AxesSubplot:title={'center':'likes_received'}>,
       <AxesSubplot:title={'center':'mobile_likes'}>,
       <AxesSubplot:title={'center':'mobile_likes_received'}>],
      [<AxesSubplot:title={'center':'www_likes'}>,
       <AxesSubplot:title={'center':'www_likes_received'}>,
       <AxesSubplot:>, <AxesSubplot:>]], dtype=object)
```




```
Num_features = [feature for feature in features if df[feature].dtype !
= object]
Cat_features = [feature for feature in features if df[feature].dtype
== object]

Num_features
```

```
['userid',
 'age',
 'dob_day',
 'dob_year',
 'dob_month',
 'tenure',
 'friend_count',
 'friendships_initiated',
 'likes',
 'likes_received',
 'mobile_likes',
 'mobile_likes_received',
```

```
 'www_likes',
 'www_likes_received']
```

```
Cat_features
```

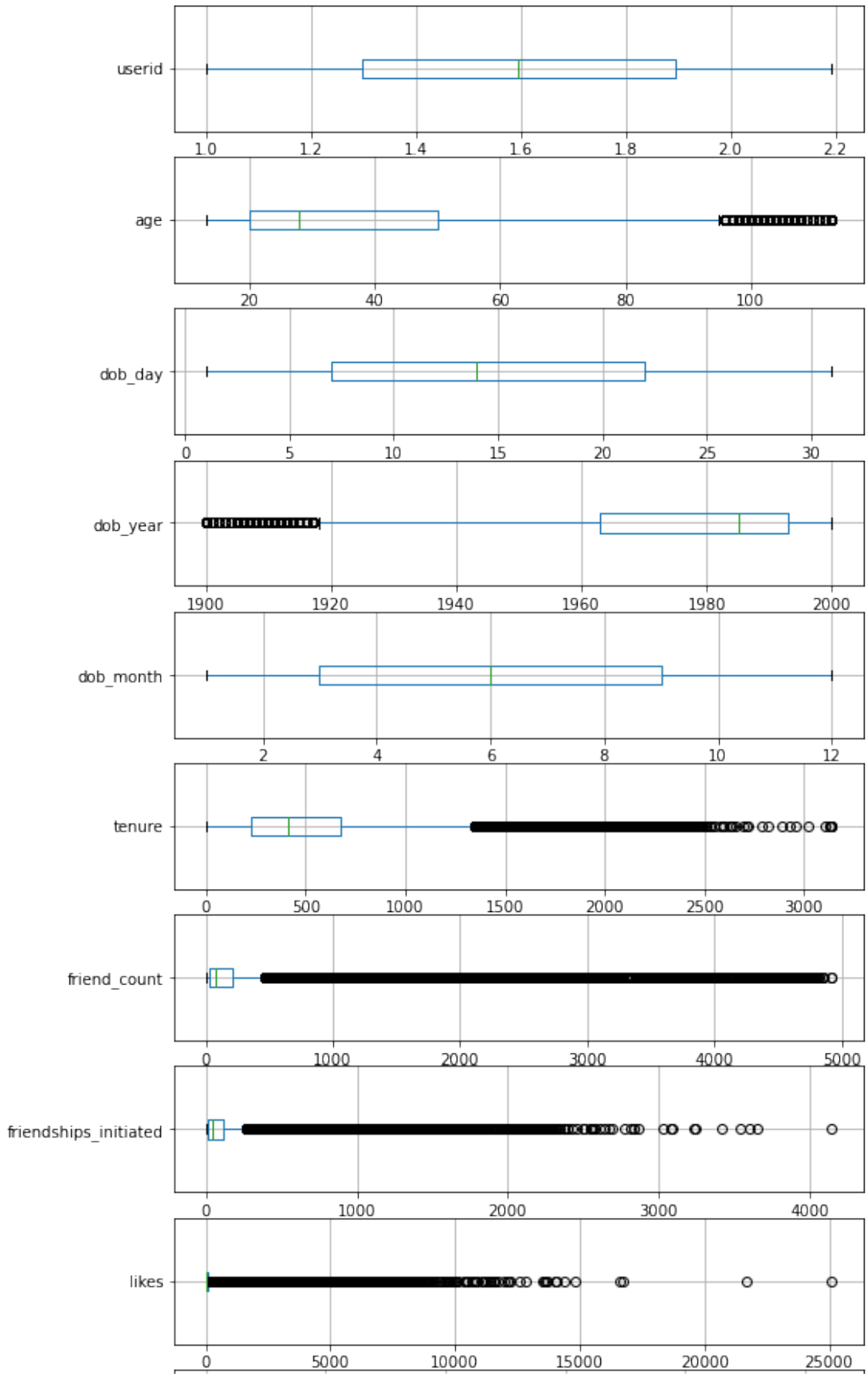
```
['gender']
```

```
df = df.fillna(method="bfill")
```

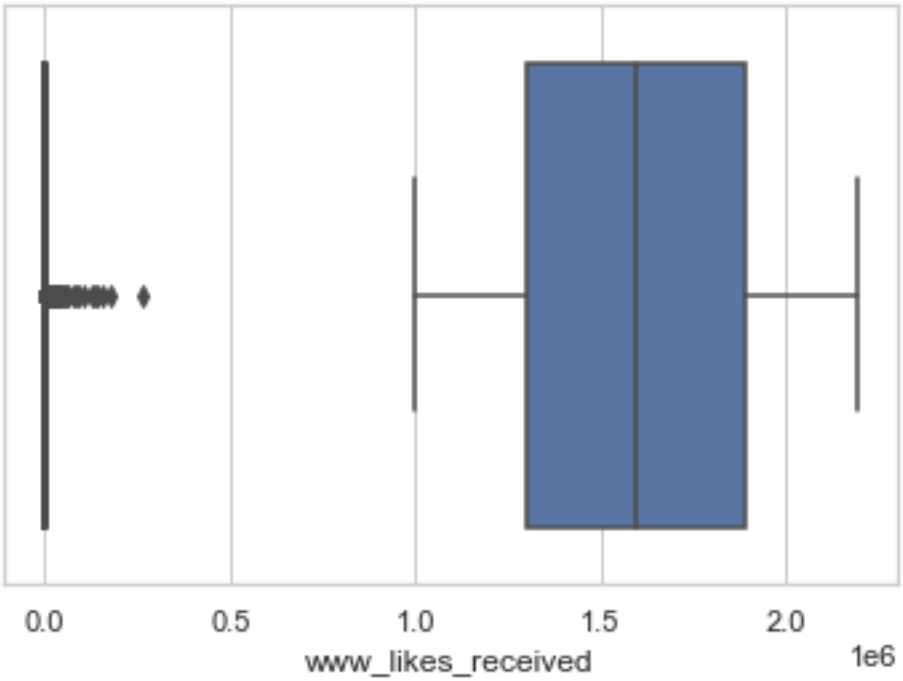
**Data exploration**

In data exploration, we'll plot histograms, boxplots , correaltion matrix,subplot of all numerical features and see countplot of the categorical variables

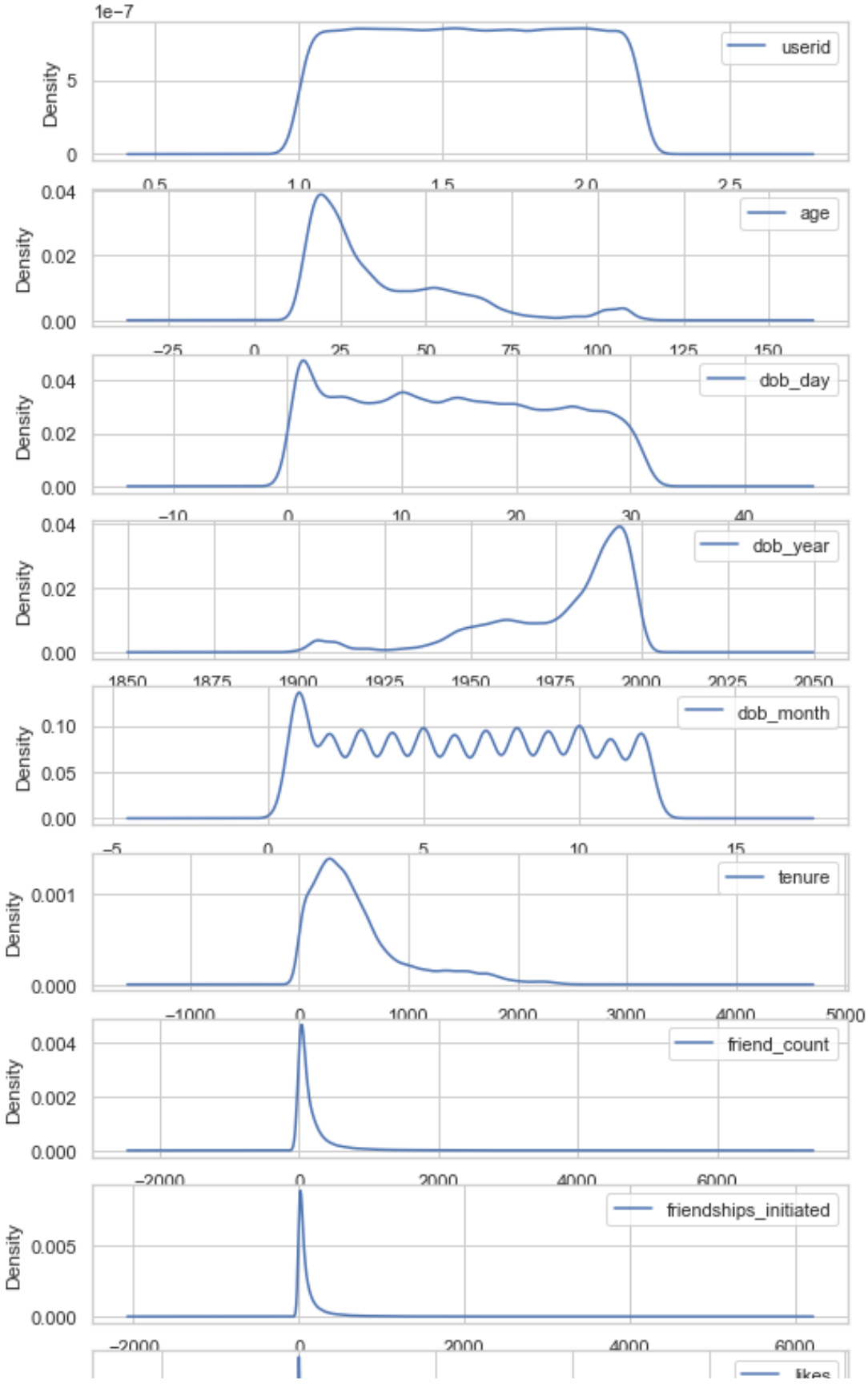
```
fig, axes = plt.subplots(14,1 ,figsize=(8,25))
for i,c in enumerate(Num_features):
 f = df[[c]].boxplot(ax=axes[i], vert=False)
```


userid
1.0
1.2
1.4
1.6
1.8
2.0
2.2
age
20
40
60
80
100
dob_day
0
5
10
15
20
25
30
dob_year
1900
1920
1940
1960
1980
2000
dob_month
2
4
6
8
10
12
tenure
0
500
1000
1500
2000
2500
3000
friend_count
0
1000
2000
3000
4000
5000
friendships_initiated
0
1000
2000
3000
4000
likes
0
5000
10000
15000
20000
25000

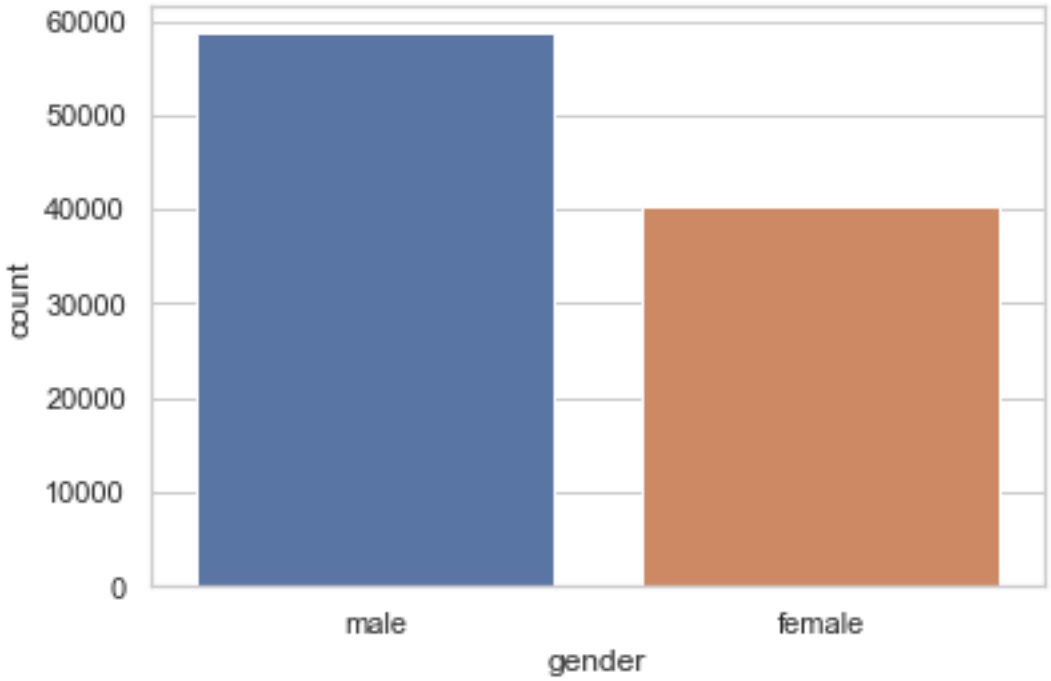
```
import seaborn as sns
sns.set_theme(style="whitegrid")
for i, c in enumerate(Num_features):
 ax = sns.boxplot(x = df[c])
```




```
fig, axes = plt.subplots(14,1 ,figsize=(8,25))
for i,c in enumerate(Num_features):
 f = df[[c]].plot(kind = 'kde', ax=axes[i])
```


1e−7
Density
userid
age
dob_day
dob_year
dob_month
tenure
friend_count
friendships_initiated
likes

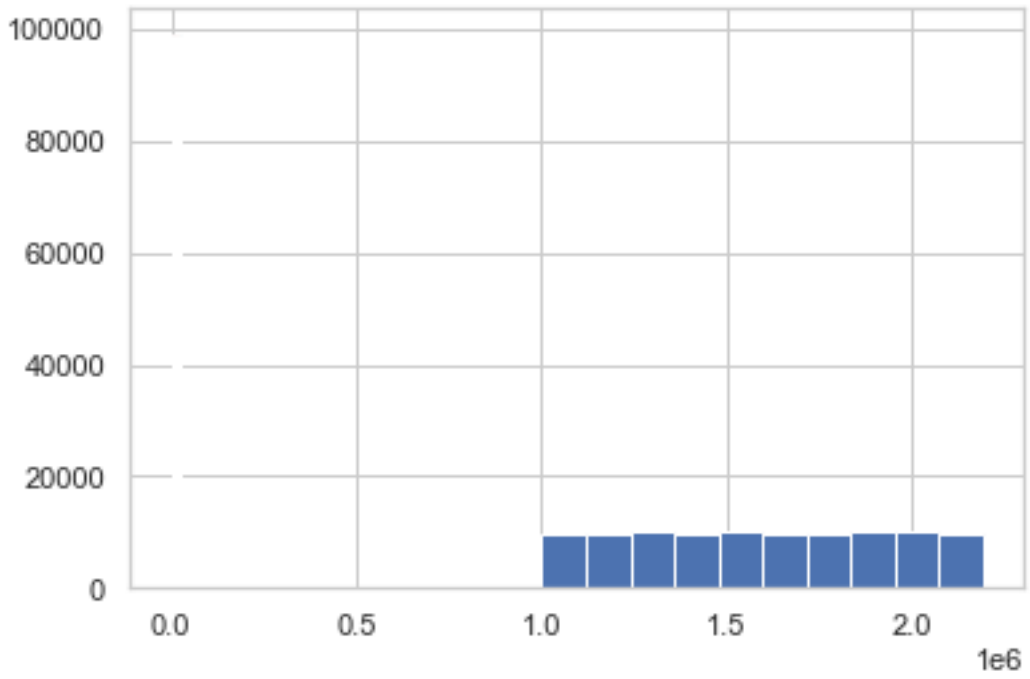
```
# For categorical features
countplot = sns.countplot(x="gender",data=df)
```



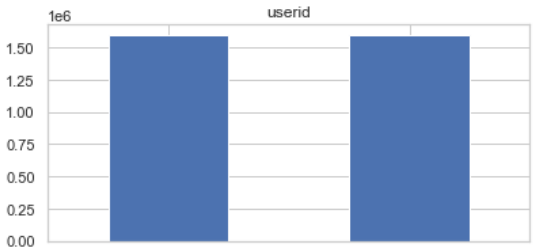

```
df_mean = df[Num_features].mean()

df_mean
```

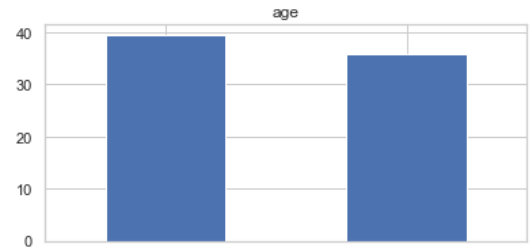
```
userid                   1.597045e+06
age                      3.728022e+01
dob_day                  1.453041e+01
dob_year                 1.975720e+03
dob_month                6.283365e+00
tenure                   5.378966e+02
friend_count             1.963508e+02
friendships_initiated    1.074525e+02
likes                    1.560788e+02
likes_received           1.426894e+02
mobile_likes             1.061163e+02
mobile_likes_received    8.412049e+01
www_likes                4.996243e+01
www_likes_received       5.856883e+01
dtype: float64
```

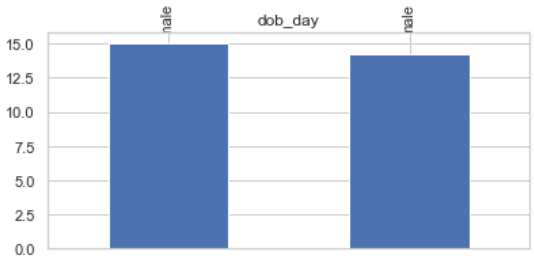
```
df_n = df.groupby('gender').mean()

for r in Num_features:
 df[r].hist()
```



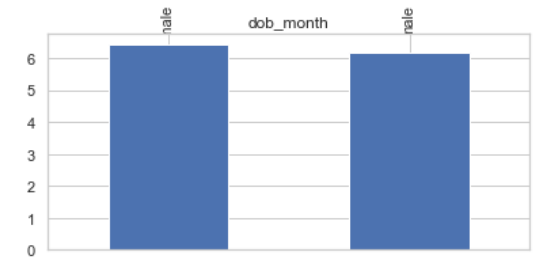

```
#df_col = df_n.columns
# Relationship between all features mean and our targer feature
fix, axes = plt.subplots(7,2, figsize=(14,24))
axes = [ax for axes_row in axes for ax in axes_row]
for i,c in enumerate(df[Num_features]):
 df_n = df.groupby('gender')[c].mean()
 plot = df_n.plot(kind='bar',title=c,ax=axes[i])
```

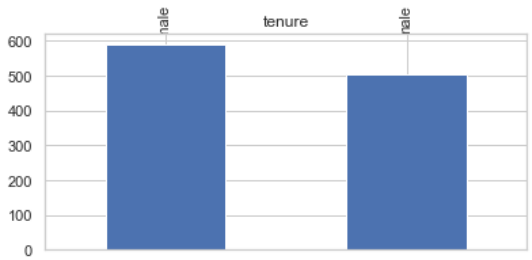

userid
1e6
1.50
1.25
1.00
0.75
0.50
0.25
0.00

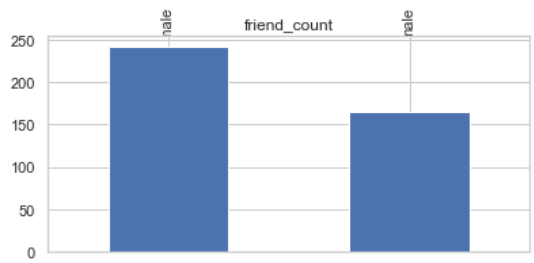

age
40
30
20
10
0

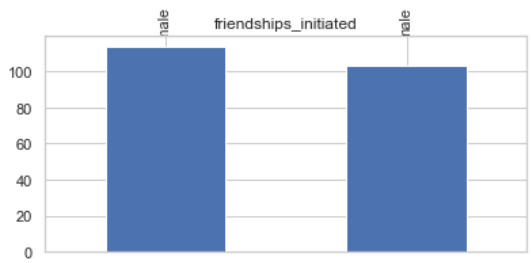

dob_day
15.0
12.5
10.0
7.5
5.0
2.5
0.0

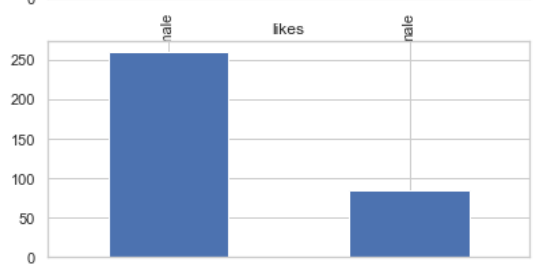

dob_year
2000
1500
1000
500
0

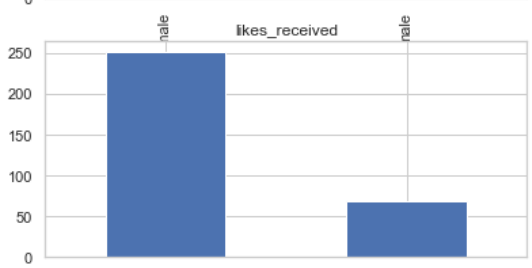

dob_month
6
5
4
3
2
1
0

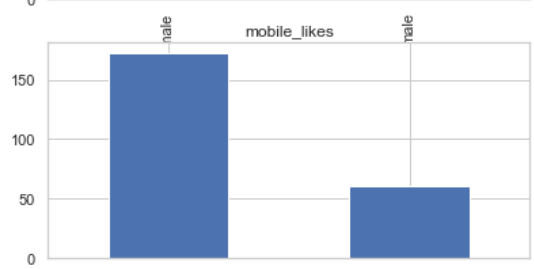

tenure
600
500
400
300
200
100
0

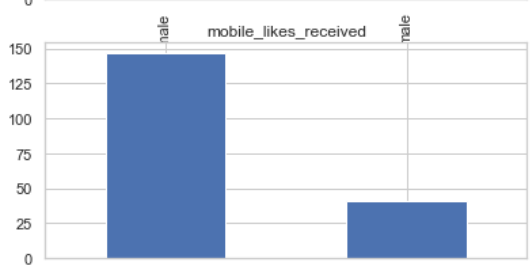

friend_count
250
200
150
100
50
0

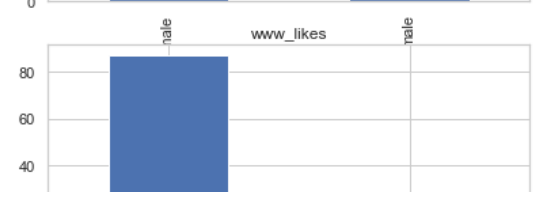

friendships_initiated
100
80
60
40
20
0

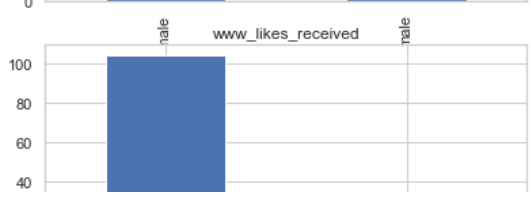

likes
250
200
150
100
50
0


likes_received
250
200
150
100
50
0

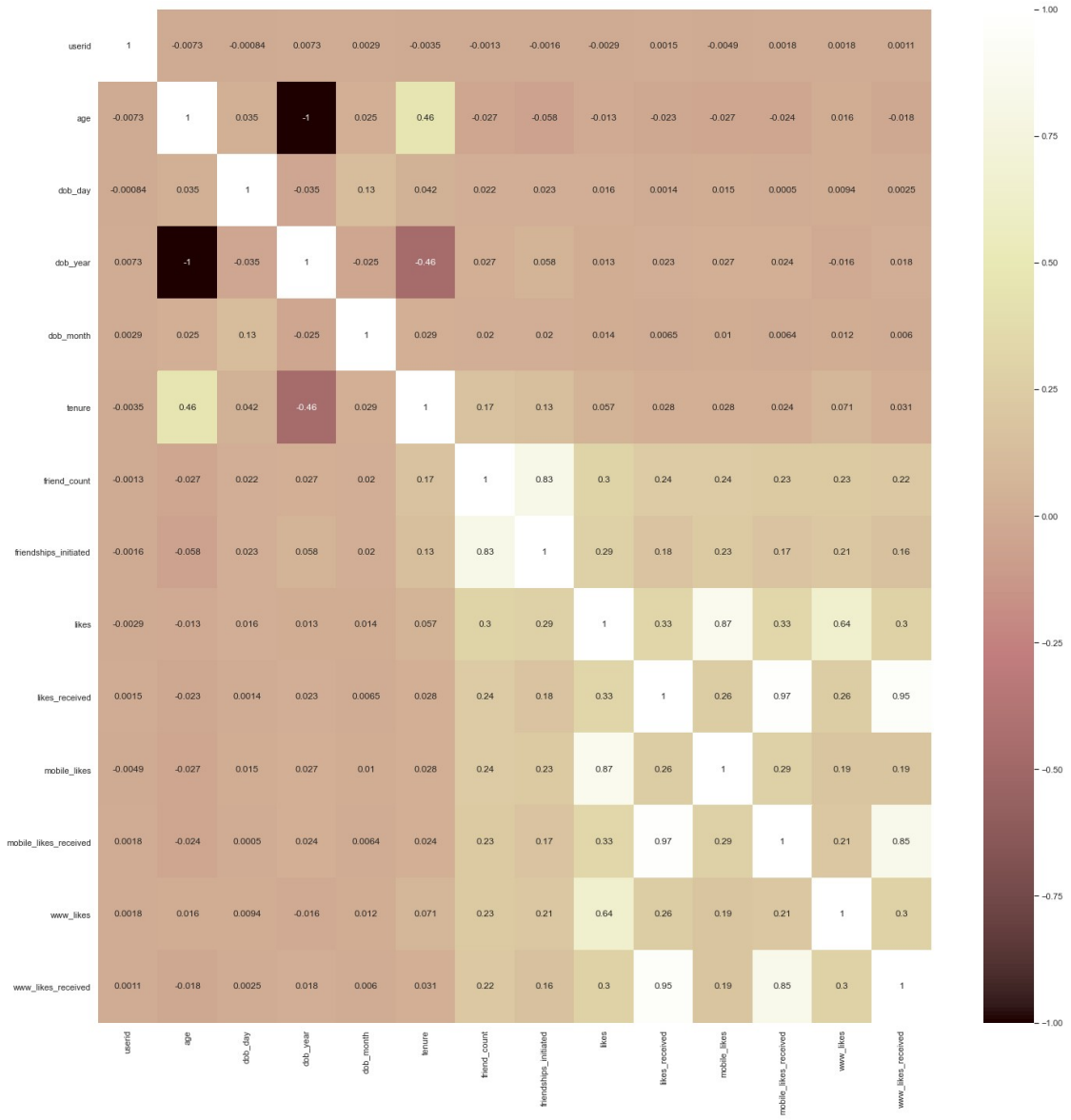

mobile_likes
150
100
50
0


mobile_likes_received
150
125
100
75
50
25
0


www_likes
80
60
40


www_likes_received
100
80
60
40

```
# Pearson Correlation matrix
corr_matrix = df[Num_features].corr(method='pearson')
plt.figure(figsize=(24,24))
correc = sns.heatmap(corr_matrix, annot=True, cmap = 'pink')
```




```
# Find features with high and low correlation
df['gender'] = df.gender.map({"male":0, "female":1})
df
```

```
        userid  age  dob_day  dob_year  dob_month  gender  tenure  \
0      2094382   14       19      1999         11       0   266.0
1      1192601   14        2      1999         11       1     6.0
2      2083884   14       16      1999         11       0    13.0
3      1203168   14       25      1999         12       1    93.0
4      1733186   14        4      1999         12       0    82.0
```

```
...        ...  ...      ...       ...         ...     ...     ...
98998  1268299   68        4      1945           4       1   541.0
98999  1256153   18       12      1995           3       1    21.0
99000  1195943   15       10      1998           5       1   111.0
99001  1468023   23       11      1990           4       1   416.0
99002  1397896   39       15      1974           5       1   397.0

       friend_count  friendships_initiated  likes  likes_received  \
0                 0                      0      0               0
1                 0                      0      0               0
2                 0                      0      0               0
3                 0                      0      0               0
4                 0                      0      0               0
...             ...                    ...    ...             ...
98998          2118                    341   3996           18089
98999          1968                   1720   4401           13412
99000          2002                   1524  11959           12554
99001          2560                    185   4506            6516
99002          2049                    768   9410           12443

       mobile_likes  mobile_likes_received  www_likes
www_likes_received
0                 0                      0          0
0
1                 0                      0          0
0
2                 0                      0          0
0
3                 0                      0          0
0
4                 0                      0          0
0
...             ...                    ...        ...
...
98998          3505                  11887        491
6202
98999          4399                  10592          2
2820
99000         11959                  11462          0
1092
99001          4506                   5760          0
756
99002          9410                   9530          0
2913

[99003 rows x 15 columns]
```

```
det = df.corr()
```

```
det['gender'].sort_values(ascending = False)
```

```
gender                    1.000000
likes                     0.150567
mobile_likes              0.124310
www_likes                 0.107918
friend_count              0.097638
tenure                    0.093523
age                       0.082228
likes_received            0.064988
www_likes_received        0.063122
mobile_likes_received     0.062193
dob_day                   0.046112
dob_month                 0.035472
friendships_initiated     0.028335
userid                    0.001480
dob_year                 -0.082228
Name: gender, dtype: float64
```

**Data cleaning and feature engineering**

1) All null values were found and replaced by the before fill method (as the percentage of null was less than 0.5%). 2) Numerical and categorical variables were seperated . 3) After plotting correlation matrix, we dropped least significant features(with relation to the feature'gender') . 4) Robust scalar was used to scale every feature{ also removing outliers}.

**Key Findings and Insights, which synthesizes the results of Exploratory Data Analysis in an insightful and actionable manner** 1) Correlation and significance of all features were found out. 2) distribution of values in all features were seen. 3) Relationship between all features mean and our targer feature was seen

```
df_final = df.drop('dob_day',1)
df_final = df_final.drop('dob_month',1)
df_final = df_final.drop('friendships_initiated',1)
df_final = df_final.drop('userid',1)
df_final
```

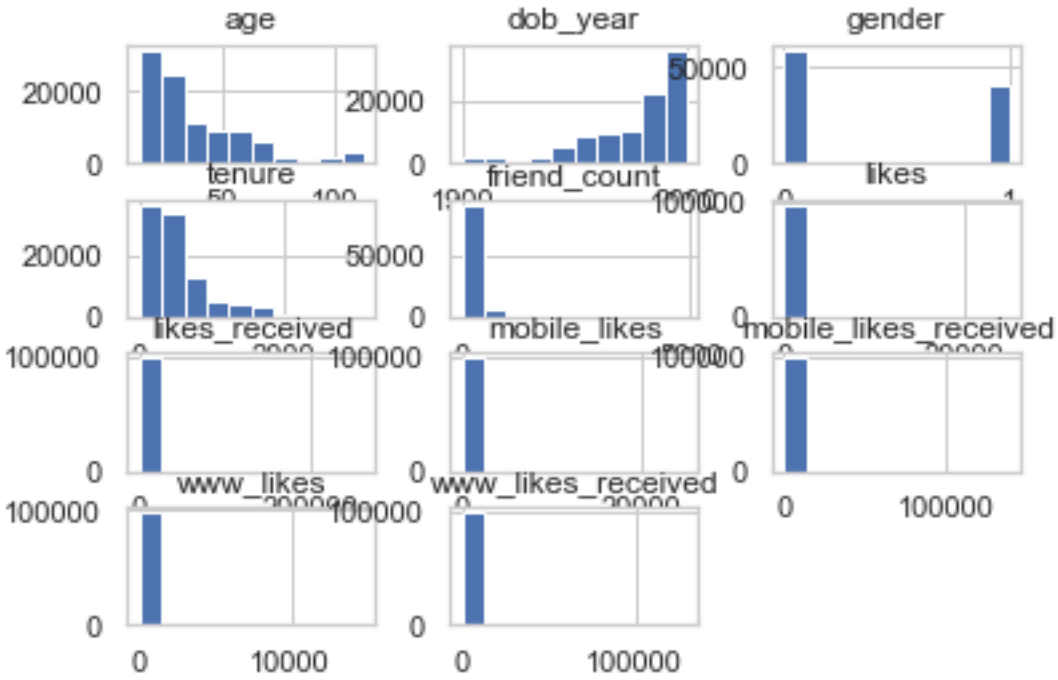
```
<ipython-input-32-705c7589e56e>:1: FutureWarning: In a future version
of pandas all arguments of DataFrame.drop except for the argument
'labels' will be keyword-only
  df_final = df.drop('dob_day',1)
<ipython-input-32-705c7589e56e>:2: FutureWarning: In a future version
of pandas all arguments of DataFrame.drop except for the argument
'labels' will be keyword-only
  df_final = df_final.drop('dob_month',1)
<ipython-input-32-705c7589e56e>:3: FutureWarning: In a future version
of pandas all arguments of DataFrame.drop except for the argument
'labels' will be keyword-only
  df_final = df_final.drop('friendships_initiated',1)
<ipython-input-32-705c7589e56e>:4: FutureWarning: In a future version
of pandas all arguments of DataFrame.drop except for the argument
'labels' will be keyword-only
  df_final = df_final.drop('userid',1)
```

```
       age  dob_year  gender  tenure  friend_count  likes
likes_received  \
0       14      1999       0   266.0             0      0
0
1       14      1999       1     6.0             0      0
0
2       14      1999       0    13.0             0      0
0
3       14      1999       1    93.0             0      0
0
4       14      1999       0    82.0             0      0
0
...    ...       ...     ...     ...           ...    ...
...
98998   68      1945       1   541.0          2118   3996
18089
98999   18      1995       1    21.0          1968   4401
13412
99000   15      1998       1   111.0          2002  11959
12554
99001   23      1990       1   416.0          2560   4506
6516
99002   39      1974       1   397.0          2049   9410
12443

       mobile_likes  mobile_likes_received  www_likes
www_likes_received
0                 0                      0          0
0
1                 0                      0          0
0
2                 0                      0          0
0
3                 0                      0          0
0
4                 0                      0          0
0
...             ...                    ...        ...
...
98998          3505                  11887        491
6202
98999          4399                  10592          2
2820
99000         11959                  11462          0
1092
99001          4506                   5760          0
756
99002          9410                   9530          0
2913
```

```
[99003 rows x 11 columns]
```

```
df_final.hist()
```

```
array([[<AxesSubplot:title={'center':'age'}>,
        <AxesSubplot:title={'center':'dob_year'}>,
        <AxesSubplot:title={'center':'gender'}>],
       [<AxesSubplot:title={'center':'tenure'}>,
        <AxesSubplot:title={'center':'friend_count'}>,
        <AxesSubplot:title={'center':'likes'}>],
       [<AxesSubplot:title={'center':'likes_received'}>,
        <AxesSubplot:title={'center':'mobile_likes'}>,
        <AxesSubplot:title={'center':'mobile_likes_received'}>],
       [<AxesSubplot:title={'center':'www_likes'}>,
        <AxesSubplot:title={'center':'www_likes_received'}>,
        <AxesSubplot:>]], dtype=object)
```

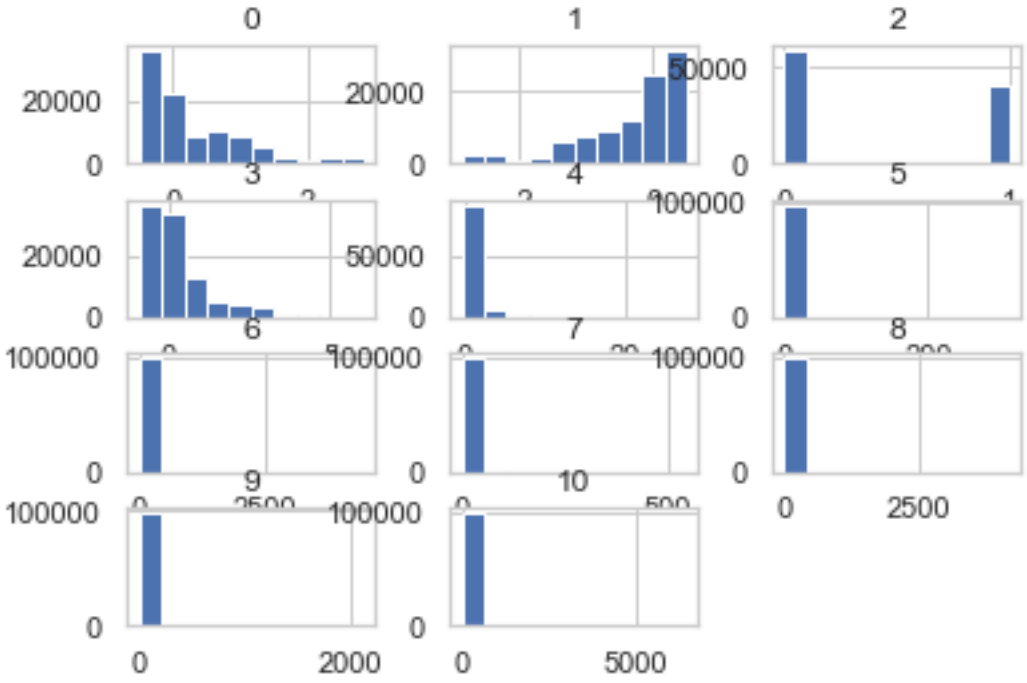



```
from sklearn.preprocessing import RobustScaler
from pandas import DataFrame
transformation = RobustScaler()
X = transformation.fit_transform(df_final)
dataset = DataFrame(X)
print(dataset.describe())
dataset.hist()
```

```
                0             1             2             3
4   \
count  99003.000000  99003.000000  99003.000000  99003.000000
```

```
99003.000000
mean       0.309341     -0.309341      0.407341      0.280393
0.653433
std        0.752992      0.752992      0.491342      1.019284
2.213167
min       -0.500000     -2.833333      0.000000     -0.917595     -
0.468571
25%       -0.266667     -0.733333      0.000000     -0.414254     -
0.291429
50%        0.000000      0.000000      0.000000      0.000000
0.000000
75%        0.733333      0.266667      1.000000      0.585746
0.708571
max        2.833333      0.500000      1.000000      6.073497
27.662857

                5             6             7             8
9   \
count  99003.000000  99003.000000  99003.000000  99003.000000
99003.000000
mean       1.813485      2.322230      2.219920      2.427894
7.137489
std        7.153509     23.929648      9.679413     25.451195
40.794307
min       -0.137500     -0.137931     -0.086957     -0.121212
0.000000
25%       -0.125000     -0.120690     -0.086957     -0.121212
0.000000
50%        0.000000      0.000000      0.000000      0.000000
0.000000
75%        0.875000      0.879310      0.913043      0.878788
1.000000
max      313.750000   4503.258621    545.804348   4198.696970
2123.571429

                10
count  99003.000000
mean       2.828442
std       30.070817
min       -0.100000
25%       -0.100000
50%        0.000000
75%        0.900000
max     6497.550000

array([[<AxesSubplot:title={'center':'0'}>,
        <AxesSubplot:title={'center':'1'}>,
        <AxesSubplot:title={'center':'2'}>],
       [<AxesSubplot:title={'center':'3'}>,
        <AxesSubplot:title={'center':'4'}>,
```

```
 <AxesSubplot:title={'center':'5'}>],
[<AxesSubplot:title={'center':'6'}>,
 <AxesSubplot:title={'center':'7'}>,
 <AxesSubplot:title={'center':'8'}>],
[<AxesSubplot:title={'center':'9'}>,
 <AxesSubplot:title={'center':'10'}>, <AxesSubplot:>]],
dtype=object)
```




**Findings** 1) likes, mobile_likes,, www_likes, friend_count, tenure. age, likes_recieved, www_likes_recieved, mobile_likes recieved are positively related to the target feature(Gender). 2) dob_year is negatively related to gender. 3) The rest features do not have any significant relationship/correlation with gender, thus can be dropped.

Chosen features for next ML/DL algorithm are likes, mobile_likes,, www_likes, friend_count, tenure. age, likes_recieved, www_likes_recieved, mobile_likes recieved, dob_year.

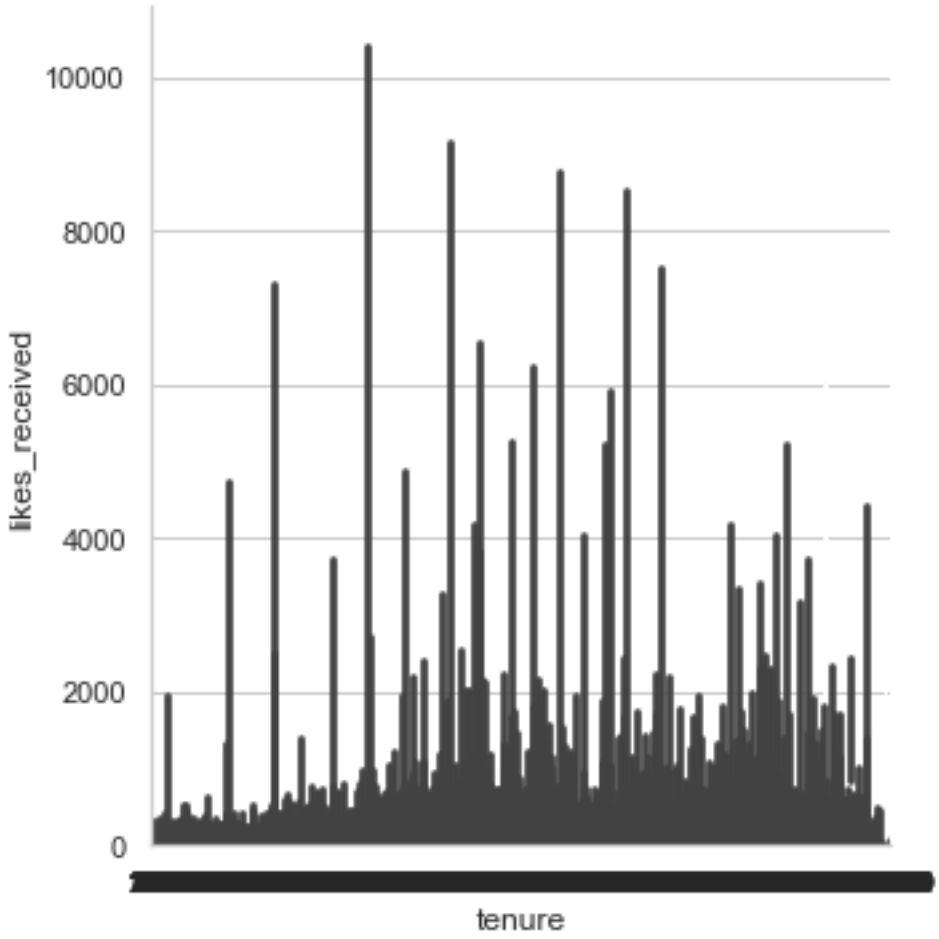
We haven't been asked to apply ML/DL algos in this question, so we'll jump to the Hypothesis part directly.

**Different hypothesis about the dataset**

1) The mean of likes recieved by the 2 genders are different. 2) Likes increase with the passage of tenure {likes_recieved is directly proportional to tenure}. 3) Female recieve more likes than male for the same friendship initiation.

```
sns.catplot(x='tenure', y='likes_received', kind="bar", data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x1ca3de5f250>
```

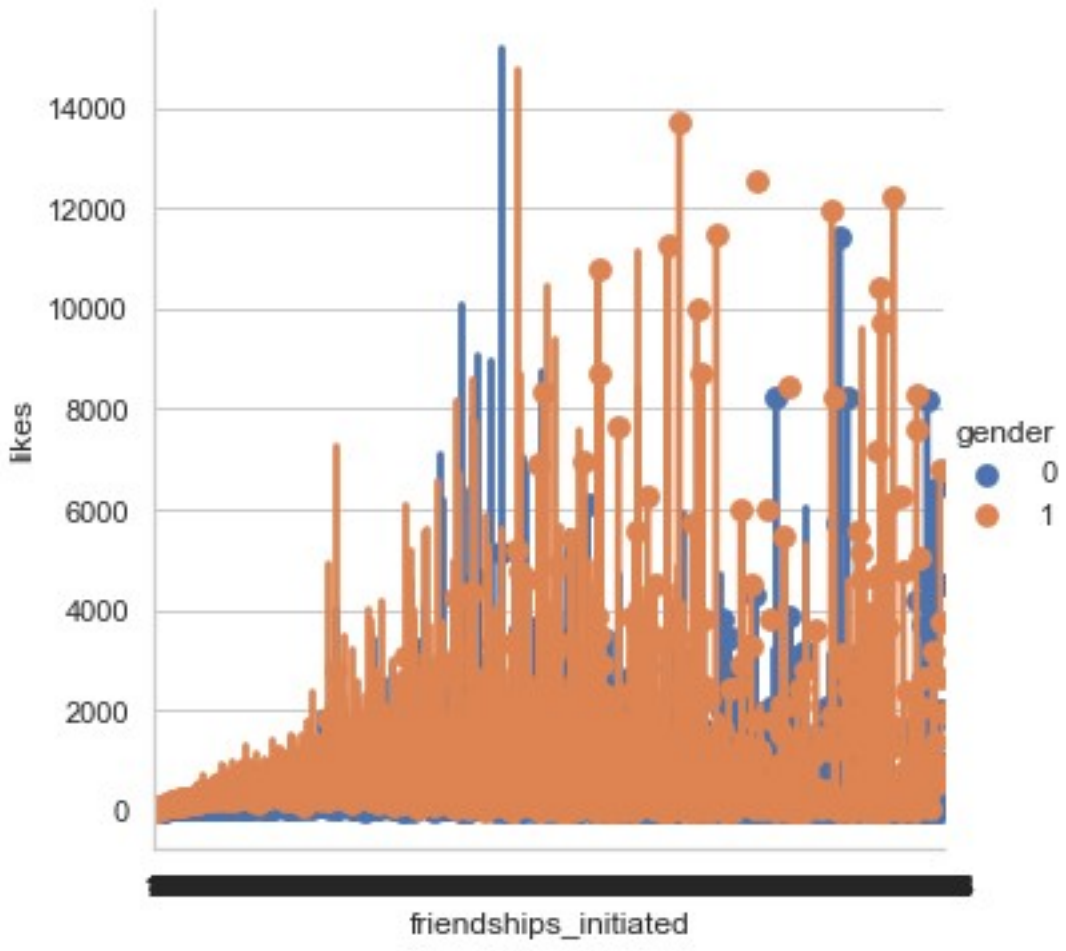



**Conducting a formal significance test for one of the hypotheses and discuss the results**

Hypothesis 2 was null hypothesis as we call nullify it with the results obtained from above plot. The likes do not necessarily increase with increase in tenure.

```
sns.catplot(x="friendships_initiated", y="likes", hue="gender",
kind="point", data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x1ca3bc29f70>
```




**Suggestions for next steps in analyzing this data**

1) Maybe power transformations can be tried to make data more gaussian like distribution.
2) Outlier removal algorithms Density based clustering algorithms can be used to clean the data.

**Summarizes the quality of this data set and a request for additional data if needed**

The quality of data was average. There weren't many highly coorelated features to our targer 'gender'. Also, the mean and median of the features was far away, thus indicating that outliers may be presesnt in it.(for that we used robust scaler). The dataset was more inclined towards male {the value of almost all features was more inclined towards males than females(subplots of averages)}.