In [1]:

```
import pandas as pd
```

In [3]:

```
df=pd.read_csv("mobile_dataset.csv")
df
```

16	838	0	0.5	0	1	1	13	0.1	196	8	...
17	595	0	0.9	1	7	1	23	0.1	121	3	...
18	1131	1	0.5	1	11	0	49	0.6	101	5	...
19	682	1	0.5	0	4	0	19	1.0	121	4	...
20	772	0	1.1	1	12	0	39	0.8	81	7	...
21	1709	1	2.1	0	1	0	13	1.0	156	2	...
22	1949	0	2.6	1	4	0	47	0.3	199	4	...
23	1602	1	2.8	1	4	1	38	0.7	114	3	...
24	503	0	1.2	1	5	1	8	0.4	111	3	...
25	961	1	1.4	1	0	1	57	0.6	114	8	...
26	519	1	1.6	1	7	1	51	0.3	132	4	...
27	956	0	0.5	0	1	1	41	1.0	143	7	...
28	1453	0	1.6	1	12	1	52	0.3	96	2	...

In [4]:

```
x=df.iloc[:,:-1]
y=df['price_range']
```

In [7]:

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
```

In [8]:

```
df.shape
```

Out[8]:

```
(2000, 21)
```

```
#check target mera categorical ya nhi
# price_range is our target variable uske base pe featureselection krna h
#categorical h to chi2
#k ki value hum lenge jitna columns h utna lo ya to usse kam apne mann se lena h
```

In [9]:

```
#apply selectkbest algorithm
ordered_rank_features=SelectKBest(score_func=chi2,k=20)
ordered_feature=ordered_rank_features.fit(x,y)
```

In [10]:

```
#do data frames banae
dfscores=pd.DataFrame(ordered_feature.scores_,columns=["Score"])
dfcolumns=pd.DataFrame(x.columns)
```

In [11]:

```
#Dono dataframes ko merge karenge
features_rank=pd.concat([dfcolumns,dfscores],axis=1)
```

In [13]:

```
features_rank.columns=['Features','Score']
features_rank
```

Out[13]:

	Features	Score
0	battery_power	14129.866576
1	blue	0.723232
2	clock_speed	0.648366
3	dual_sim	0.631011
4	fc	10.135166
5	four_g	1.521572
6	int_memory	89.839124
7	m_dep	0.745820
8	mobile_wt	95.972863
9	n_cores	9.097556
10	pc	9.186054
11	px_height	17363.569536
12	px_width	9810.586750
13	ram	931267.519053
14	sc_h	9.614878
15	sc_w	16.480319
16	talk_time	13.236400
17	three_g	0.327643
18	touch_screen	1.928429
19	wifi	0.422091

In [14]:

```
features_rank.nlargest(10,'Score') #10 no. of features based on score
```

Out[14]:

	Features	Score
13	ram	931267.519053
11	px_height	17363.569536
0	battery_power	14129.866576
12	px_width	9810.586750
8	mobile_wt	95.972863
6	int_memory	89.839124
15	sc_w	16.480319
16	talk_time	13.236400
4	fc	10.135166
14	sc_h	9.614878

```
Feature Importance
This technique gives you a csore for each feature of your data,the higher the score more
relevant it is
```

In [16]:

```
from sklearn.ensemble import ExtraTreesClassifier
import matplotlib.pyplot as plt
model=ExtraTreesClassifier()
model.fit(x,y)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: F
utureWarning: The default value of n_estimators will change from 10 in versi
on 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[16]:

```
ExtraTreesClassifier(bootstrap=False, class_weight=None, criterion='gini',
           max_depth=None, max_features='auto', max_leaf_nodes=None,
           min_impurity_decrease=0.0, min_impurity_split=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
           oob_score=False, random_state=None, verbose=0, warm_start=False)
```

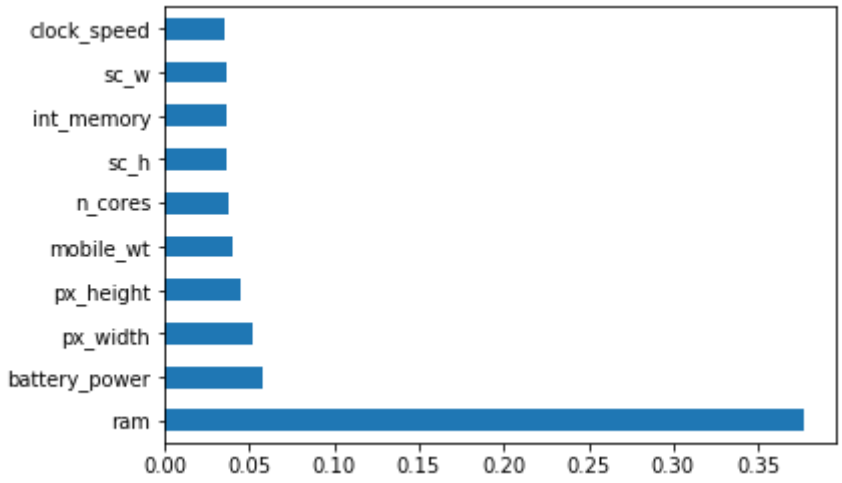
In [17]:

```
print(model.feature_importances_)
```

```
[0.05752582 0.02175379 0.03555947 0.01952948 0.03392488 0.01960454
 0.03604741 0.0340251  0.04013649 0.03775821 0.03369793 0.04529621
 0.05184943 0.37696318 0.0362946  0.03596961 0.03532534 0.01399432
 0.01218203 0.02256217]
```

In [18]:

```
ranked_features=pd.Series(model.feature_importances_,index=x.columns)#series= 1 dimension
ranked_features.nlargest(10).plot(kind='barh')
plt.show()
```



In [19]:

```
#correlation
df.corr()
```

Out[19]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_mem
battery_power	1.000000	0.011252	0.011482	-0.041847	0.033334	0.015665	-0.004
blue	0.011252	1.000000	0.021419	0.035198	0.003593	0.013443	0.041
clock_speed	0.011482	0.021419	1.000000	-0.001315	-0.000434	-0.043073	0.006
dual_sim	-0.041847	0.035198	-0.001315	1.000000	-0.029123	0.003187	-0.015
fc	0.033334	0.003593	-0.000434	-0.029123	1.000000	-0.016560	-0.029
four_g	0.015665	0.013443	-0.043073	0.003187	-0.016560	1.000000	0.008
int_memory	-0.004004	0.041177	0.006545	-0.015679	-0.029133	0.008690	1.000
m_dep	0.034085	0.004049	-0.014364	-0.022142	-0.001791	-0.001823	0.006
mobile_wt	0.001844	-0.008605	0.012350	-0.008979	0.023618	-0.016537	-0.034
n_cores	-0.029727	0.036161	-0.005724	-0.024658	-0.013356	-0.029706	-0.028
pc	0.031441	-0.009952	-0.005245	-0.017143	0.644595	-0.005598	-0.033
px_height	0.014901	-0.006872	-0.014523	-0.020875	-0.009990	-0.019236	0.010
px_width	-0.008402	-0.041533	-0.009476	0.014291	-0.005176	0.007448	-0.008
ram	-0.000653	0.026351	0.003443	0.041072	0.015099	0.007313	0.032
sc_h	-0.029959	-0.002952	-0.029078	-0.011949	-0.011014	0.027166	0.037
sc_w	-0.021421	0.000613	-0.007378	-0.016666	-0.012373	0.037005	0.011
talk_time	0.052510	0.013934	-0.011432	-0.039404	-0.006829	-0.046628	-0.002
three_g	0.011522	-0.030236	-0.046433	-0.014008	0.001793	0.584246	-0.009
touch_screen	-0.010516	0.010061	0.019756	-0.017117	-0.014828	0.016758	-0.026
wifi	-0.008343	-0.021863	-0.024471	0.022740	0.020085	-0.017620	0.006
price_range	0.200723	0.020573	-0.006606	0.017444	0.021998	0.014772	0.044

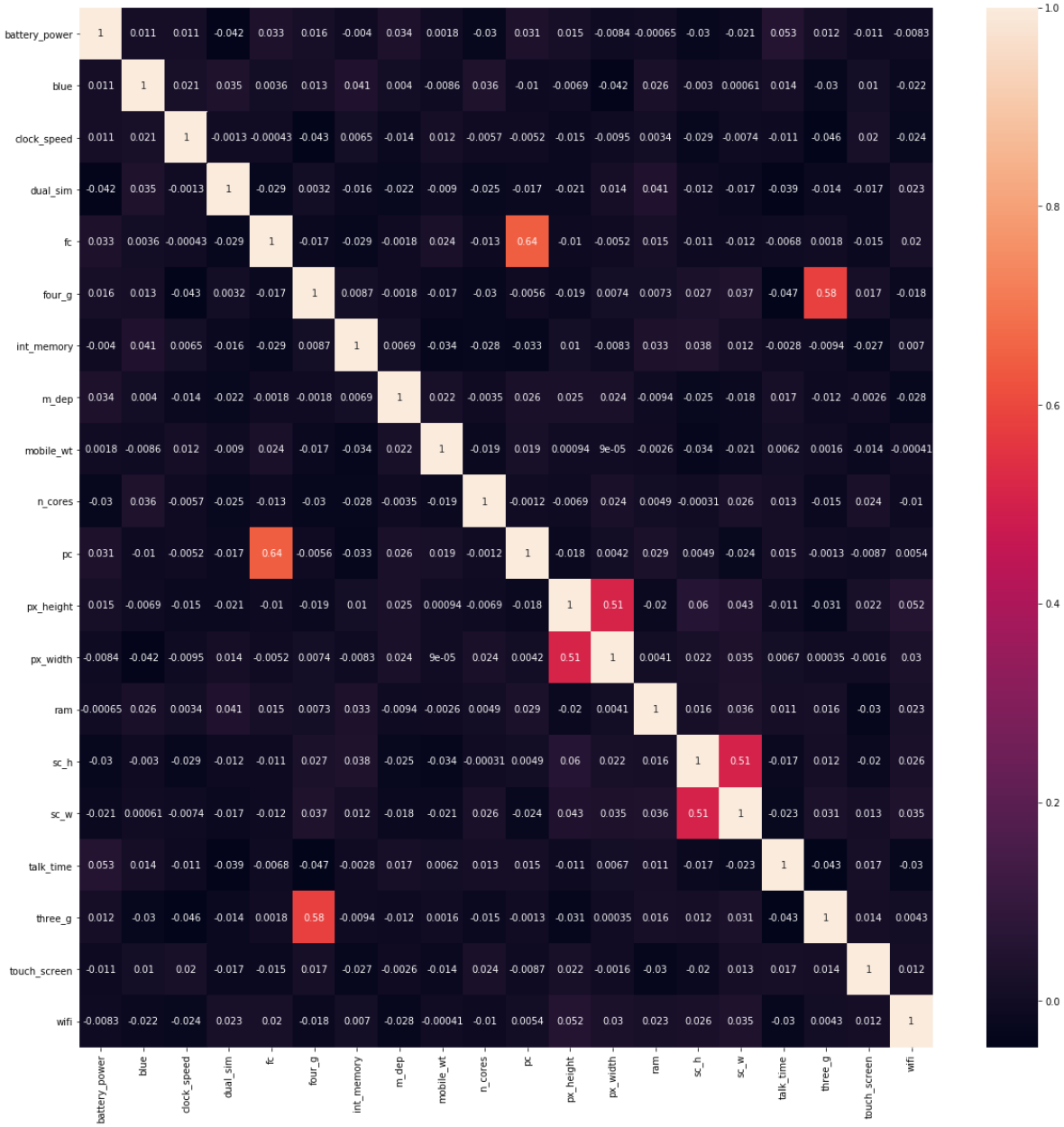
21 rows × 21 columns

In [22]:

```
import seaborn as sns
corr=df.iloc[:,:-1].corr()
top_features=corr.index
plt.figure(figsize=(20,20))
sns.heatmap(df[top_features].corr(),annot=True)
```

Out[22]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2090f543048>
```



Information Gain

In [23]:

```
from sklearn.feature_selection import mutual_info_classif
mutual_info=mutual_info_classif(x,y)
mutual_data=pd.Series(mutual_info,index=x.columns)
mutual_data.sort_values(ascending=False)
```

Out[23]:

```
ram              0.849337
sc_w             0.030523
clock_speed      0.029454
battery_power    0.027099
px_width         0.026736
px_height        0.022262
pc               0.016232
sc_h             0.014879
four_g           0.011764
three_g          0.003880
m_dep            0.002663
blue             0.002546
int_memory       0.001657
mobile_wt        0.000870
dual_sim         0.000000
wifi             0.000000
fc               0.000000
touch_screen     0.000000
talk_time        0.000000
n_cores          0.000000
dtype: float64
```

In [ ]: