In [52]:

```
import pandas
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB, BernoulliNB, MultinomialNB
import numpy as np
from sklearn import metrics
import matplotlib.pyplot as plt
```

In [4]:

```
df = pandas.read_csv("train.csv")
```

In [7]:

```
df.head()
```

Out[7]:

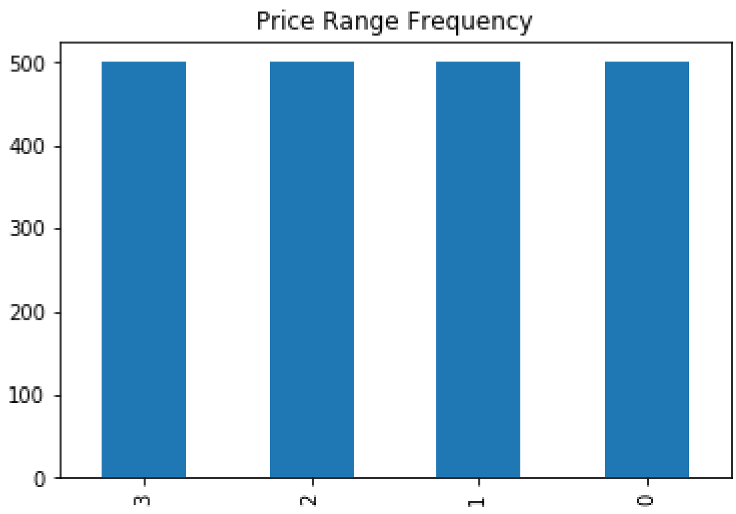
speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width
2.2	0	1	0	7	0.6	188	2	...	20	756
0.5	1	0	1	53	0.7	136	3	...	905	1988
0.5	1	2	1	41	0.9	145	5	...	1263	1716
2.5	0	0	0	10	0.8	131	6	...	1216	1786
1.2	0	13	1	44	0.6	141	2	...	1208	1212

In [20]:

```
df.price_range.value_counts().plot(kind='bar', title='Price Range Frequency')
```

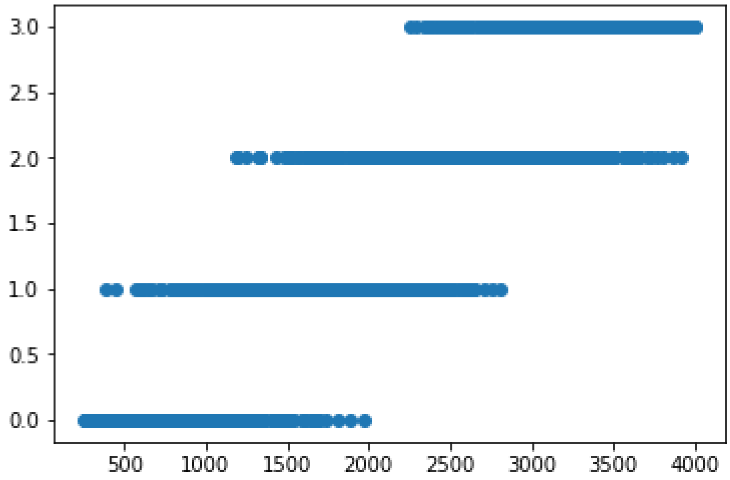
Out[20]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f61ad6c5ba8>
```



In [62]:

```
plt.scatter(x=df['ram'], y=df['price_range'])
plt.show()
```



In [ ]:

```
plt.scatter(x=df['ram'], y=df['price_range'])
plt.show()
```

In [42]:

```
#Dataset is pretty balanced, dividing the features_target dataset
target = np.array(df[["price_range"]].copy())
features = np.array(df.drop('price_range', axis = 1))
```

In [43]:

```
#Dividing the train/test dataset
train_features, test_features, train_labels, test_labels = train_test_split(features, targe
```

In [46]:

```
model = GaussianNB()
model.fit(train_features, train_labels.ravel())
```

Out[46]:

```
GaussianNB(priors=None, var_smoothing=1e-09)
```

In [47]:

```
y_pred = model.predict(test_features)
```

In [51]:

```
print('Expected performance')
print('Mean Absolute Error:', metrics.mean_absolute_error(test_labels, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(test_labels, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(test_labels, y_pred)))
```

```
Expected performance
Mean Absolute Error: 0.208
Mean Squared Error: 0.208
Root Mean Squared Error: 0.45607017003965516
```

In [57]:

```
errors = 0
for i in range(len(test_labels)):
    if test_labels[i] != y_pred[i]:
        errors += 1
```

In [59]:

```
print("Naive Bayes hit rate: "+  str((len(test_labels)-errors)/(len(test_labels)) * 100) +
```

```
Naive Bayes hit rate: 79.2%
```

In [ ]:

```

```