```
import numpy as np
import pandas as pd
```

```
data = pd.read_csv('/content/drive/MyDrive/ColabNotebooks/ml_profit_all.csv').drop(['Unnam
```

```
print(data.shape)
data.head(10)
```

```
(1715, 3)
```

	PRIMARY_KEY	EXPENSES	PROCEEDS
0	2017_Speech synthesis	304177.0	1659028.0
1	2017_Watson	106780.0	720711.0
2	2017_OCR	297888.0	1369815.0
3	2017_Speech recognition	178571.0	958785.0
4	2017_Transformer	2072470.0	16546514.0
5	2017_Facial recognition	163253.0	1307986.0
6	2017_GPT-2	143542.0	142539.0
7	2017_AlphaGo	45945.0	420942.0
8	2017_Transformer	64749.0	0.0
9	2017_GPT-3	788420.0	5683949.0

```
data.dropna(inplace=True)
print(data.shape)
```

```
(1275, 3)
```

```
data['PROFIT'] = data['PROCEEDS'] - data['EXPENSES']
```

```
data['PRIMARY_KEY'] = data['PRIMARY_KEY'].apply(lambda x: x[5:])
```

```
KEY_TO_PROFIT = pd.DataFrame(data.groupby(by='PRIMARY_KEY', as_index=False)['PROFIT'].mean
```

```
KEY_TO_PROFIT.columns = ['PRIMARY_KEY', 'MEAN_PROFIT']
KEY_TO_PROFIT = KEY_TO_PROFIT.sort_values(by='MEAN_PROFIT', ascending=False)
```

```
print(KEY_TO_PROFIT.shape)
KEY_TO_PROFIT.head()
```

```
(16, 2)
```

	PRIMARY_KEY	MEAN_PROFIT
4	GPT-3	5.854825e+06
3	GPT-2	5.403160e+06
7	OCR	4.543448e+06

```
data = pd.merge(
    left=data,
    right=KEY_TO_PROFIT,
    on='PRIMARY_KEY',
    how='left'
)


print(data.shape)
data.head()
```

```
(1275, 5)
```

	PRIMARY_KEY	EXPENSES	PROCEEDS	PROFIT	MEAN_PROFIT
0	Speech synthesis	304177.0	1659028.0	1354851.0	3.227248e+06
1	Watson	106780.0	720711.0	613931.0	8.180296e+05
2	OCR	297888.0	1369815.0	1071927.0	4.543448e+06
3	Speech recognition	178571.0	958785.0	780214.0	1.877511e+06
4	Transformer	2072470.0	16546514.0	14474044.0	4.522072e+06

Bootstrap

```
GPT_3 = data[data['PRIMARY_KEY'] == 'GPT-3']['PROFIT'].values
GPT_2 = data[data['PRIMARY_KEY'] == 'GPT-2']['PROFIT'].values
OCR = data[data['PRIMARY_KEY'] == 'OCR']['PROFIT'].values


def get_bootstrap_samples(data, n_samples):
    indices = np.random.randint(0, len(data), (n_samples, len(data)))
    samples = data[indices]
    return samples


def stat_intervals(stat, alpha):
    boundaries = np.percentile(stat, [100 * alpha / 2., 100 * (1 - alpha / 2.)])
    return boundaries


np.random.seed(0)

ocr_mean_scores = list(map(np.mean, get_bootstrap_samples(OCR, 3000)))
```

```
gpt2_mean_scores = list(map(np.mean, get_bootstrap_samples(GPT_2, 3000)))
gpt3_mean_scores = list(map(np.mean, get_bootstrap_samples(GPT_3, 3000)))

print("95% confidence interval for the GPT_3 mean profit:",  stat_intervals(gpt3_mean_scor
print("95% confidence interval for the GPT_2 mean profit:",  stat_intervals(gpt2_mean_scor
print("95% confidence interval for the OCR mean profit:",  stat_intervals(ocr_mean_scores,
```

```
95% confidence interval for the GPT_3 mean profit: [5390351.09083333 6355341.3385
95% confidence interval for the GPT_2 mean profit: [3862895.987  6925948.5895]
95% confidence interval for the OCR mean profit: [3752293.10971429 5389446.18071429]
```