1. Read in JSON file:

```
import json
```

```
with open("E:\\Machine Learning\\U.W\\Classification\\important_words.json", "r") as f:
    important_words = json.load(f)

important_words = [str(s) for s in important_words]
```

```
important_words
```

```
['baby',
 'one',
 'great',
 'love',
 'use',
```

2. The function *String.split().count(word)* splits a String and count the number of each word:

```
products["review_clean"][0].split().count("and")
```

```
2
```

3. The function *pd.DataFrame({"Name": column})* creates a new DataFrame:

```
table = pd.DataFrame({"word": ["(intercept)"] + important_words})
```

```
table
```

	word
0	(intercept)
1	baby
2	one
3	great
4	love
5	use

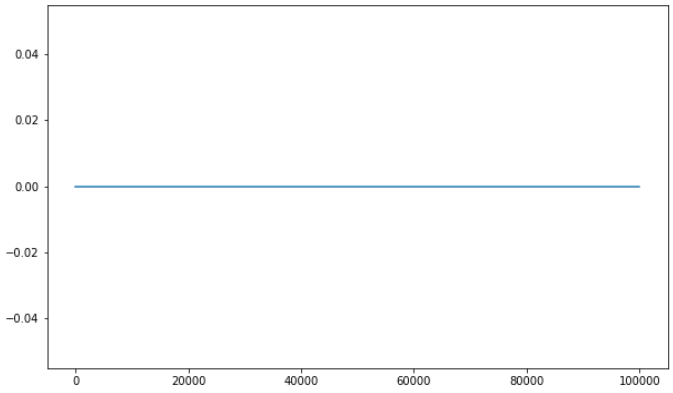
4. The function *list*len(XX)* outputs a list *[list, list, list]* of length *len:*

```
[0.]*5
```

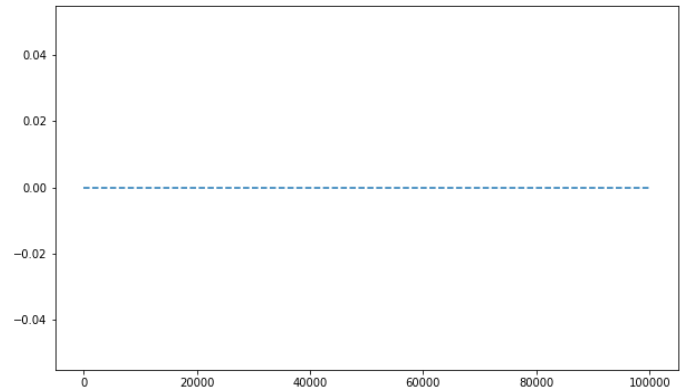
```
[0.0, 0.0, 0.0, 0.0, 0.0]
```

5. Visualization:

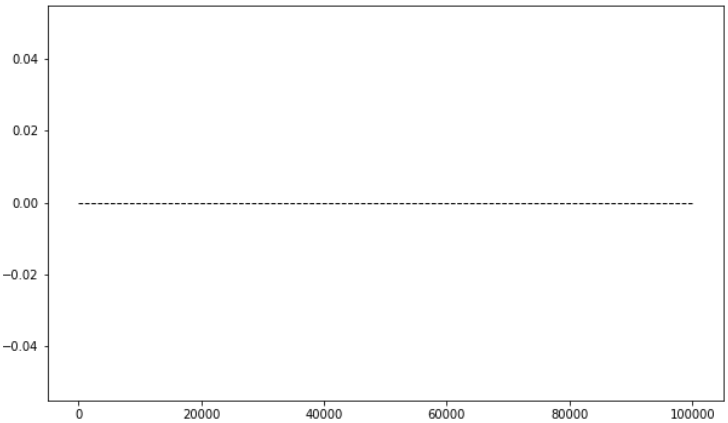
```
xx = [0, 4, 10, 1e2, 1e3, 1e5]
plt.plot(xx, [0.]*len(xx))
```



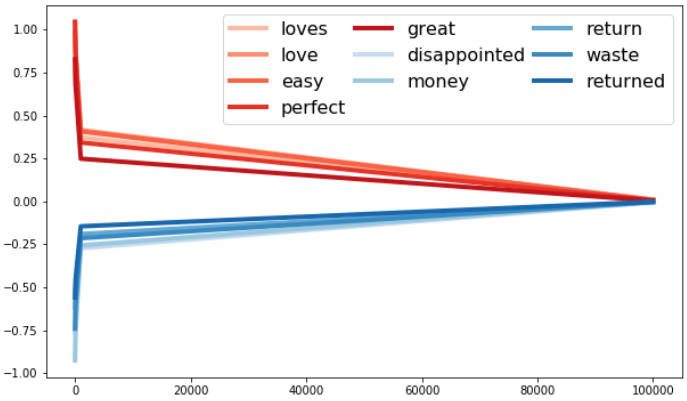

```
xx = [0, 4, 10, 1e2, 1e3, 1e5]
plt.plot(xx, [0.]*len(xx), "--")
```



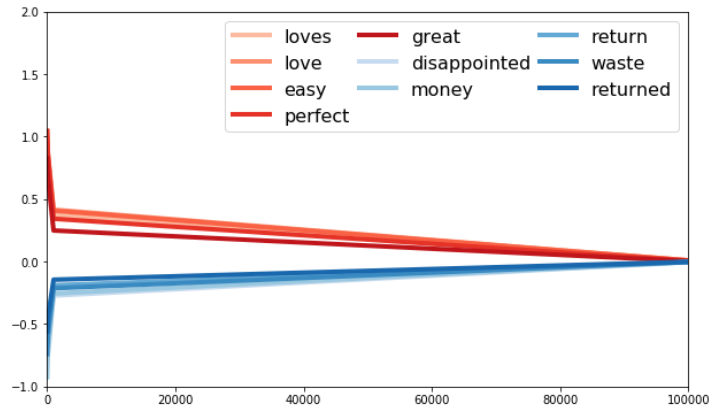

```
xx = [0, 4, 10, 1e2, 1e3, 1e5]
plt.plot(xx, [0.]*len(xx), "--", lw=1, color="k")
```



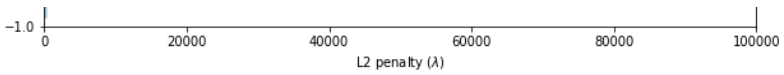
```
plt.legend(loc="best", ncol=3, prop={"size":16}, columnspacing=0.5)
```




```
plt.axis([1, 1e5, -1, 2])
```




```
plt.xlabel("L2 penalty ($\lambda$)")
```



6. The function *dataframe.isin()* filter the DataFrame by value:

```
table[table["word"].isin(positive_words)]
```

	word	coefficients [L2=0]	coefficients [L2=4]	coefficients [L2=10]	coefficients [L2=1e2]	coefficients [L3=1e3]	coefficients [L2=1e5]
3	great	0.793059	0.788378	0.781486	0.693890	0.372431	0.008916
4	love	1.042101	1.034556	1.023456	0.883468	0.414689	0.009011
8	easy	0.983293	0.976446	0.966371	0.839295	0.407048	0.008979
23	loves	1.045100	1.036585	1.024044	0.864734	0.342611	0.006040
34	perfect	0.826638	0.819590	0.809206	0.676986	0.248442	0.003970

7. Function *sorted()*:

```
sorted(validation_accuracy.keys())
```

```
[0, 4, 10, 100.0, 1000.0, 100000.0]
```

8. Function *items()* and *sorted*:

```
train_accuracy
```

```
{0: 0.7847233671714912,
 4: 0.7847704736556987,
 10: 0.7846527074451798,
 100.0: 0.7839696634241703,
 1000.0: 0.7723343618249052,
 100000.0: 0.7119438490708246}
```

```
train_accuracy.items()
```

```
[(0, 0.7847233671714912),
 (100000.0, 0.7119438490708246),
 (100.0, 0.7839696634241703),
 (1000.0, 0.7723343618249052),
 (10, 0.7846527074451798),
 (4, 0.7847704736556987)]
```

```
sorted(train_accuracy.items())
```

```
[(0, 0.7847233671714912),
 (4, 0.7847704736556987),
 (10, 0.7846527074451798),
 (100.0, 0.7839696634241703),
 (1000.0, 0.7723343618249052),
 (100000.0, 0.7119438490708246)]
```

```
sorted(train_accuracy.items(), key=lambda x:x[0])
```

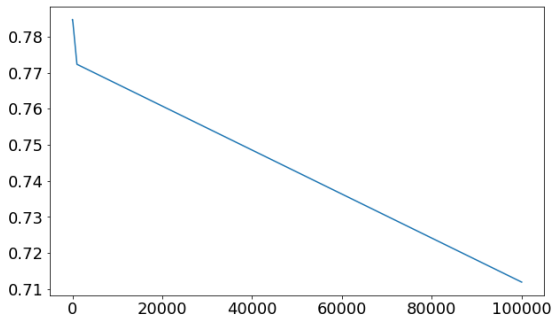
```
[(0, 0.7847233671714912),
 (4, 0.7847704736556987),
 (10, 0.7846527074451798),
 (100.0, 0.7839696634241703),
 (1000.0, 0.7723343618249052),
 (100000.0, 0.7119438490708246)]
```

```
sorted(train_accuracy.items(), key=lambda x:x[1])
```

```
[(100000.0, 0.7119438490708246),
 (1000.0, 0.7723343618249052),
 (100.0, 0.7839696634241703),
 (10, 0.7846527074451798),
 (0, 0.7847233671714912),
 (4, 0.7847704736556987)]
```

9. Visualization:

```
plt.plot([p[0] for p in sorted_list], [p[1] for p in sorted_list])
```




```
plt.plot([p[0] for p in sorted_list], [p[1] for p in sorted_list], "bo-")
```