给定一个字符串数组 `words`，找到 `length(word[i]) * length(word[j])` 的最大值，并且这两个单词不含有公共字母。你可以认为每个单词只包含小写字母。如果不存在这样的两个单词，返回 0。

示例 1:

输入： `["abcw","baz","foo","bar","xtfn","abcdef"]`
输出： `16`
解释： 这两个单词为 `"abcw", "xtfn"`。

示例 2:

输入： `["a","ab","abc","d","cd","bcd","abcd"]`
输出： `4`
解释： 这两个单词为 `"ab", "cd"`。

示例 3:

输入： `["a","aa","aaa","aaaa"]`
输出： `0`
解释： 不存在这样的两个单词。

分析，先对每个字符串进行字符出现的统计，然后每个字符串进行对比，从a比到z，有26次比较，在字符串比较中动态的记录最大值，并用两字符串相乘要大于最大值进行剪枝。

```
class Solution {
public:
    int maxProduct(vector<string>& words) {
        vector<map<char,int>> word_num;
        for(int i=0;i<words.size();i++){
            map<char,int> num;
            for(int j=0;j<words[i].size();j++){
                num[words[i][j]]=1;
            }
            word_num.push_back(num);
        }
        int num=0;
        for(int i=0;i<words.size();i++){
            for(int j=i+1;j<words.size();j++){
                if(words[i].size()*words[j].size()<=num){
                    continue;
                }
                for(char k='a';k<='z';k++){
                    if(word_num[i][k]==1&&word_num[j][k]){
                        break;
                    }
                    if(k=='z'){
                        num=words[i].size()*words[j].size();
                    }
                }
```

```
            }
        }
        return num;
    }
};
```