1 你所在的行业，常用的数据分析指标有哪些？请简述。

常用的数据分析指标有：用户的年龄、区域、在线时长、资产、交易偏好、风险偏好等等。

年龄、区域主要是描述用户的基本属性，了解用户的年龄分层以及主要用户在全国的分布情况，以便于针对性的运营投放，加强投顾。

资产、交易偏好以及风险偏好主要是描述用户的资产和投资能力，以便于对于不同的资产以及不同的风险偏好用户，精准推荐不同的理财产品，比如有保本收益型、基金、偏股票基金以及私募基金等。

2 Google搜索引擎是如何对搜索结果进行排序的？（请用自己的语言描述PageRank算法。）

Google的搜索引擎主要是使用PageRank算法对网页进行排序的，pageRank算法会计算每个网页的PageRank值，然后根据这个值的大小对网页的重要性进行排序。

```
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

/**
 * pagerank算法，计算每个PageNode的值
 */
public class PageRank
{
    public static void CalcPageRank(ArrayList<PageNode> graph)
    {
        double distance = 0.00001;
        double d = 0.85;// damping factor
        double common = (1 - d) / graph.size();
        while (true)
        {
            for (PageNode n : graph)
            {
                double sum = 0.0;
                for (int nodeId : n.getNeighbors())
                {
                    PageNode nb = getNodeById(nodeId,graph);
                    if (nb == null){
                        continue;
                    }
                    sum += nb.getPR() / nb.getDegree();
                }
                double newPR = common + d * sum;
                //如果尚未收敛，赋新值，否则结束迭代
                if (Math.abs(n.getPR() - newPR) > distance)
                    n.setPR(newPR);
                else
                    return;
            }
        }
    }
```

```
    public static PageNode getNodeById(int nodeId, ArrayList<PageNode> graph)
    {
        for(PageNode n:graph)
        {
            if (n.nodeId==nodeId)
                return n;
        }
        return null;
    }


    public static ArrayList<PageNode> buildGraph()
    {
        Random random = new Random();
        ArrayList<PageNode> graph = new ArrayList<PageNode>();//图以节点集合形式来
表示
        // 生成10PageNode节点
        for (int i = 0; i < 10; i++) {
            PageNode pageNode = new PageNode(i);
            List<Integer> neighors = new ArrayList<>();
            // 随机生成5个以内的邻居
            int loop = random.nextInt(5);
            while (neighors.size() < loop) {
                int r = random.nextInt(10);
                if (r == i){
                    continue;
                }else {
                    neighors.add(r);
                }
            }
            pageNode.setNeighbors(neighors);
            graph.add(pageNode);

        }
        return graph;
    }

    public static void main(String[] args)
    {
        ArrayList<PageNode> graph = buildGraph();
        CalcPageRank(graph);
        for(PageNode n:graph)
        {
            System.out.println(String.format("PageRank of %d is
%.2f",n.nodeId,n.getPR()));
        }
    }
}
```

```
import java.util.ArrayList;
import java.util.List;

/**
 * 页面节点
```

```
 */
public class PageNode implements Comparable<PageNode>
{
    public int nodeId;
    private List<Integer> neighbors = new ArrayList<Integer>();//以邻接表的形式表示
图结构【无向图】
    private double pr=1; //PageRank初始值设为1
    public PageNode(int nodeId)
    {
        this.nodeId = nodeId;
    }

    public int getDegree()
    {
        return this.neighbors.size();
    }

    public List<Integer> getNeighbors()
    {
        return this.neighbors;
    }
    public void setNeighbors(List<Integer> neighbors)
    {
        this.neighbors=neighbors;
    }

    public double getPR()
    {
        return pr;
    }
    public void setPR(double val)
    {
        this.pr=val;
    }

    // 按PageRank值排序
    public int compareTo(PageNode anotherPageNode)
    {
        if (this.neighbors != null && anotherPageNode.neighbors != null)
        {
            // 降序排列
            if (anotherPageNode.getPR() >this.getPR())
                return 1;
            else if (anotherPageNode.getPR() <this.getPR())
                return -1;
            else
                return 0;
        }
        return 0;
    }
}
```