業務のためのデータを
活用可能なデータへ

DataVehicle

「活用できる状態のデータ」ってなんだろう

ないようであるデータ
あるようで使えないデータ

まずは、企業がデータを活用する際、最初のボトルネックとなりうる「データの問題」について考えていきたいと思います。

うちにはあまりデータなんてないと思っている企業でも実は意外にたくさんデータを持っています。一方で、うちは大量のビッグデータを持っていると思っている企業でも、いざ活用しようとするとそれだけでは使い物にならないことがわかったりします。なぜこのようなギャップが生まれてしまうのでしょうか？

その答えは「業務のためのデータ」と「活用のためのデータ」の違いにあります。

現代的な経営を行っているほとんどの企業は、業務を行えば必ずどこかにデータが蓄積されます。たとえば、あなたの会社の従業員について、どのような経歴の人間が、どのようなプロセスで採用され、その過程でどのような評価を受け、どの部署に配置され、いつ休んで、今までにいくらの賃金を受け取ったのかというデータを会社のどこかに持っているはずです。あるいは、工場で使う機械について、どのメーカーのどの型番のものを、いついくらで誰から調達し、どこに置いているか、というデータを持っている会社もあるでしょう。

このようなデータは、社内の IT システムの中に蓄積されたものもあれば、特定の部署だけで共有されるエクセルシートにのみ存在しているかもしれません。このような「業務のためのデータ」は、社内の情報を確認して管理する上で、あるいは社内外で必要な手続き（ たとえば人事異動や給料の支払、減価償却の計算など）をする上で、誰かの記憶や紙に頼るよりもとても便利です。

ただ、業務のためのデータは「人が後から中身を見て確認できる」「エクセル上に該当箇所を読み込んで簡単な計算ができる」「特定の業務に用いられるシステムが止まらなければよい」というだけでその目的を達成できてしまいます。いざこのようなデータを活用しようとすると、そうした「業務のため」には十分だったはずのデータの問題点に気づくことになります。

活用のためのデータの基本構造

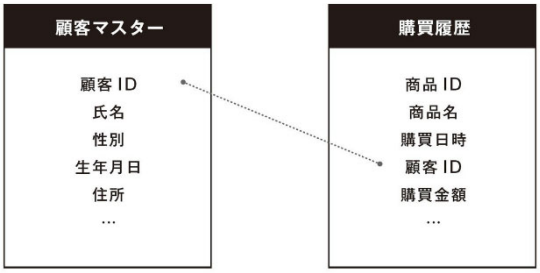
では「活用のためのデータ」とはどのようなものなのでしょうか？分析するにしても、AI に利用するにしても、活用のためのデータは基本的に次のような条件を満たしていなければいけません。

① 最終的には一枚の表にまとまっていなければいけない
② 最終的に用いられる項目は「数値の大小を示す」ものか「(せいぜい数十個程度の) 有限な状態に分類する」もののみ
③ 行数は最低数十行以上
④ 列数は自由だが多ければ多いほど分析や予測の価値が増す
⑤ 中身のセルに抜け・漏れは基本的に許されない

この条件に当てはまらないものの一例に、業務用のデータベース内で、いくつもの表にまたがって管理されているデータや、フリーテキストで詳細な情報が記入されたデータなどを挙げることができます。こうしたデータはそのままでは活用できません。

具体例で考える活用のための加工法

この問題をもう少し具体的に掘り下げてみましょう。たとえば現在、世界中で営業するスーパーマーケットの多くは ID-POS（図表 1-1）と呼ばれるシステムを導入しています。読者の皆様に、一度もスーパーマーケットでレジを打つところを見たことがないとか、ポイントカードという仕組みについてまったく想像がつかない、という人はいないでしょう。



図表1-1　ID-POSのデータ構造

それほど私たちの生活にとって身近な「業務のためのデータ」がスーパーマーケットの ID-POS に蓄積されているわけですが、これを「活用のためのデータ」にしようとする場合、どのようなところに注目したらよいでしょうか？

まず条件①についてみると、多くの場合、ID-POS には少なくとも 2 種類の表が含まれていることが考えられます。1 つは顧客マスター、つまり、ポイントカードを作ってくれた顧客 1 人につき 1 行、という形で、その中には氏名、性別や生年月日、住所といった個人情報が含まれます。一方、販売のトランザクションすなわち、レシートに印字される行ごとに、何の商品を、いつ、誰が、いくらで買ったのか、という形式の表で管理されるデータも存在します。これを購買のトランザクションまたは、購買履歴と呼びます。現代の業務システムでは多くの場合、リレーショナルデータベースが使われていますが、リレーショナルデータベースでは「顧客」と「購買」というように異なる粒度のデータを別々の表として管理することで、データ量を節約したり、整合性を取ったりしています。

ただし、別々の表とは言っても、多くの場合、レシート側のデータにおける「誰が」、というところは顧客１人ずつに振り分けられたＩＤで特定できるようになっています。したがって、丁寧に確認すれば表をまたいだ業務の処理、たとえば特定の顧客が昨日お買い物をしたかどうか、ということも理論上わかるようになっています。

しかし、このままの状態では顧客マスターと購買履歴は別々の表です。このままではデータ分析をするにしても単純な集計しかできませんし、AI のアルゴリズムを適用することもできません。

次に条件②について考えてみると、これらのデータの中には「数字の大小」と「（ せいぜい数十個程度への）分類」ではない項目が多数含まれていることがわかります。たとえば顧客マスターは、性別は「（ せいぜい数十個程度への）分類」ですが、氏名や住所には数十個よりはるかに数多くの種類があり、生年月日もそのままでは「数字の大小」というわけではありません。

レシート側の購買履歴についても、「いくらで」買ったかという項目は数字の大小ですが、「何の商品を」や「いつ」といった項目は、「数字の大小」でも「（ せいぜい数十個程度への）分類」でもありません。

つまり、ID-POS データで、そのまま使えるような項目はごくわずかで、「男女別にどちらの客単価が高いか」といった見える化ぐらいの活用しかできないということになります。

実際、私たちはこれまで、高額なシステムやツールを導入して大量のデータを蓄積しながら、この程度のわずかなデータ活用しかできていないという企業をたくさん目にしてきました。

データの規模が大きいということは最終的には、条件③である「行数の多さ」を満たしやすくなりますが、これは数十行～数万行あればそれ以上はデータ活用の価値を向上させるわけではありません。どれだけのビッグデータを集めたとしても、「数字の大小」、「有限な状態への分類」といった活用可能な列をたくさん用意できるかどうか、という条件④を満たせなければそれほど大した活用にはなりません。

有意義な分析結果を導き出すための２つの方法

このような「業務のためのデータ」と「活用のためのデータ」のギャップを克服するために、行なうべきことは大きく分けて、２つあります。（ 図表１−２）

項目	含まれる値の例	「活用のための」形式
顧客ID	000001, 000002, 000003, …	−
(顧客の)氏名	高橋一郎, 渡辺和子, 鈴木謙二, …	−
(顧客の)性別	男性, 女性, 男性, …	（数十個以内への）分類
(顧客の)生年月日	1963/3/4, 1958/10/21, 1974/11/3, …	−
(顧客の)住所	横浜市青葉区藤が丘X-X-X, 木更津市太田2丁目…	−
商品ID	4903110021322, 4902705103030, 49013305…	−
商品名	あんぱん, 低脂肪乳, ポテトチップス, …	−
購買日時	2018/4/1/ 9:58:12, 2018/4/1/ 10:01:49, …	−
購買金額	112, 208, 124, 358, 98, …	（大小が意味を持つ）数字

図表1-2　データ項目の例と「活用のための」形式

１つは「結合・集計」で、２つ目が「数値化と再分類」です。

「結合・集計」とはたとえば、エクセルの VLOOKUP 関数とピボットテーブルを使ったり、SQL の JOIN 句や集計関数を使ったりする操作などです。「顧客」と「購買」という、共通した ID（ 今回で言えば顧客 ID）を使って 2 枚の表を紐づけ、その後もし「顧客ごと」という切り口で活用したければ、1 人の顧客につき複数存在する購買履歴を集計して「１人の顧客に対して１行ずつ」という形のデータに加工します。こうした操作によって、条件①の問題はクリアできるはずです。

また、「数値化」とはたとえば生年月日や購買日時など、そのままでは大小を意味する数値ではない項目から、「年齢」や「直近の購買日からの経過日数」という数値を計算で示すことです。あるいは住所や商品名などそのままでは数十個をはるかに超える、フリーテキストのデータをうまく再分類して、「居住エリア」や「商品ジャンル」などの新たな項目を見出すこともできます。

データがキレイに管理されていれば、少しのコツがわかるだけでこうした作業はすぐに実行できます。しかし問題は、多くの企業の「業務のためのデータ」は、業務が円滑に回ることを目的としているため、結合や、数値化を阻む障壁が含まれている場合があります。

次回は、活用のための加工を阻む障壁も含めて、どのように作業を進めていくかを見ていきましょう。

データ活用のための結合と集計

データ活用のための結合作業

引き続き、スーパーマーケットの ID-POS を題材にして、条件①を満たせるためにどうすればよいかを考えていきます。結合を行なう際には次のような手順で考えていくとよいでしょう。

手順①　表をまたいでデータをつなげるための「キー」を確認
手順②　つなげる前に「データを含む対象」にズレがないか確認
手順③　「最終的に何毎に一行にまとめるか」を決定
手順④　複数行になる場合は適切に集計

では手順①について考えてみます。顧客マスターはポイントカードを作ってくれた顧客１人につき１行、という形で、その中には氏名、性別や生年月日、住所といった個人情報が含まれていました。一方、販売履歴はレシートに印字される行ごとに、どんな商品を、いつ、誰が、いくらで買ったのか、という形式でした。

両者の間は顧客 ID という項目でつなげることができます。これは顧客１人につき必ず１つずつの値で、複数の顧客の間で「ID がかぶる」ということは基本的にありません。このことを専門用語では「ユニーク」あるいは「一意」といいます。たとえば顧客 ID が 123456 番というデータがレシート側にあった際に、顧客マスターから顧客 ID が 123456 番のデータを検索すれば、それが男性による購買かどうか、といったことが判別できます。このように複数の表をつなぐための項目を専門用語で「キー」と呼びます。適切な「キー」が存在していれば、つまりエクセル上で VLOOKUP 関数を使ったり、データベースに対して JOIN 句を含む SQL 文を実行したりすれば、少なくとも「１枚の表」というデータ活用のための条件を満たすこと

ができます。

これはあくまで「理想的な状況」であり、実際はそう上手く運ばないケースも多々あります。

結合を阻む「不完全なキー」

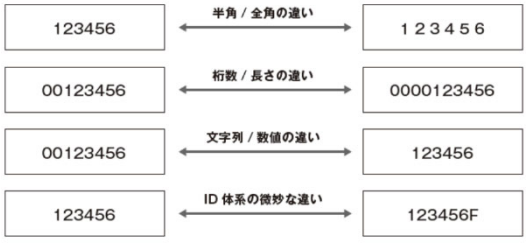
そのようなケースは、データを連結するための「キー」が不完全である場合を挙げることができます。最悪なケースは、キーの定義が不確かで、データの管理がアナログすぎるために「フリーテキストで入力された氏名」を使って、顧客マスター側のデータとレシート側のデータを紐づけなければいけない、という状況などです。これは「顧客 ID＝123456」という形で互いのデータを参照しあうのではなく、「顧客氏名が渡辺和子さん」といった情報をもとに、参照しあわなければならない状態です。

大手スーパーマーケットなどではあまり見かけませんが、個人商店や中小の B to B 企業などでは、ユニークな顧客 ID を確実に振らずに、エクセルなどで管理された「フリーテキストだらけの顧客台帳」と、「フリーテキストだらけの販売履歴」という形で、業務のためのデータを管理しているところも数多くあります。「フリーテキストだらけ」でも「顧客台帳」があれば、顧客への連絡業務はでき、「販売履歴」があれば、未収金のチェックや税務署への売上申告という業務はこなせるというわけです。

また大きな問題はこのような「キー」が顧客にとってユニークなものとは限らない場合です。ユニークとは、より正確には、顧客間で ID がかぶることもなく、1 人の顧客が複数の ID を持つこともなく、顧客 1 人と 1 つの ID が、完全に 1 対 1 の関係を持つことです。「氏名」も確かに「個人を特定するもの」ですが、厳密に 1 対 1 かというとそうではありません。

たとえば、「同姓同名」の人間がたまたま 2 人以上、顧客マスターにあれば、購買履歴側のデータにおいて、そのどちらを指すのか特定することができません。また、「表記の揺れ」という問題が生じることもあります。つまり、名字が「渡辺」で、正確な表記が「渡邊」だった場合に、本人やスタッフの気分次第で「渡辺」と略することもあれば、読み間違って「渡部」と違う書き方をしてしまうことがあるかもしれません。名字と名前の間の空白の有無や、その空白は全角か半角かが違う場合もあります。さらに結婚や離婚によって名字自体が変わってしまう、というケースも考えられます。

このような状況は「キー」の管理として最悪なパターンですが、一応顧客を特定する ID が設定されていた場合でも、同様に「連結できない」という問題が生じ得ます。一方のデータでは ID が全角、他方では半角ということがあります。あるいは、一方のデータでは「00123456」という 8 桁の文字列が、他方のデータでは「0000123456」と 10 桁になってしまっているかもしれません。さらに、数字文頭に 0 が並ぶ文字列ではなく「123456」という数値として管理されている状況もあります。あるいは特定の業務などの理由で「男性／女性を判別するため末尾にM／F をつけて管理する」という仕組みを導入し、ID が「123456F」と表示されるという場合も見かけます（図表 1 - 3）。



図表1-3　データの結合を阻む「キー」

これらは多くの場合「業務のため」に使う限り、それほど問題にはなりません。スタッフの記憶をたどったり、住所などを参照すれば「どちらの渡辺さん」かを特定できるかもしれませんし、名前の表記を間違っても少し失礼ですが、郵便物は届きます。また、名字と名前の間が全角であれ半角であれ、人が見れば「同じ名前」だと認識するでしょう。ID についても全角で書かれた「00123456F」という文字列と「123456」と書かれた数値が同じ意味だ、とシステム管理者なら理解できるかもしれません。

しかし、データ活用ではそうはいきません。個別の業務をこなそうというのではなくデータをまとめて活用しようという場合、たとえば数万ものデータを一括して、加工しなければならないわけです。このような場合に「１つ１つ丁寧に見て何とかする」というのは、活用までに全体としてとんでもない手間がかかる、というのと同じ意味になるからです。

活用のためのデータに思わぬ間違いが含まれないように、つなげるための「キー」はきちんと互いに「まったく同じ」となっているかどうか確認しましょう。

データを含む対象のズレ

「キー」の確認が終わり、もし何か問題があっても何とか互いに「まったく同じ」という状態に整えることができたら、次につなげようとするデータ間で「その中に含む対象」にズレがないかを確認しましょう。

このようなズレがある状況の例として、顧客マスターについては「登録されているすべての顧客のデータ」を使い、販売履歴については「本店のデータ」だけを使う、といったことが考えられます。なぜこんなことをしてしまうのかというと、多くの場合、１人の顧客は何回も来店して、その度にいくつもの商品を買うため、購買履歴の方が顧客マスターよりも何倍もデータが多いからです。そのため「とりあえず本店の購買履歴だけでやってみよう」とか「とりあえず東京の購買履歴だけでやってみよう」「とりあえず直近１か月の購買履歴だけでやってみよう」と考える人はしばしばいます。

もちろん「一部のデータだけで試しにやってみる」という考え方自体は間違いではありませんが、その際注意すべきなのは、組み合わせて活用する顧客マスターについても、同じように「一部」でなければならないことです。

たとえば全顧客のデータと、本店だけの購買履歴を紐づけた場合、どのようなことが起こるでしょうか？「どのような顧客がたくさん買ってくれるか」という分析をした際に、このデータからわかることを正確にいえば、「すべての顧客のうち、本店でたくさん買ってくれるのはどういう人か」という情報でしかありません。当然、「本店の近くに住んでいる顧客はよく本店で買ってくれる」というどうでもいい情報が得られてしまいます。また、このことに気づかず、同じデータで試しに「優良顧客を発見する AI」を作ってみた場合、おそらく同様に本店の遠くに住んでいる人はどれだけロイヤルカスタマーであっても「（ 本店では）ほとんど購買してくれない」と判定されてしまうかもしれません。

このようなことがないように、使う購買履歴が「本店における過去１か月分の購買履歴」であれば、使う顧客マスターも「本店において過去１か月に一度でも購買したことのある顧客」あるいは少なくとも「理論上本店において過去１か月に購買できると考えられる顧客」に絞り込まなければいけ

ません。

データの行は活用したい切り口

「キー」と含む対象の確認が終わったら、エクセルの VLOOKUP 関数なり、SQL の JOIN 句なりを使って、いつでもデータを結合することができますが、次に考えるべきは最終的にデータをどう活用するかという形式です。つまり、結合した後に「何毎に 1 行ずつにするのか」ということを決めてその形にしていかなければなりません。今回のデータの場合どのような形式が考えられるでしょうか？

顧客マスターの情報を使って、「顧客ごとに 1 行ずつ」ということもでき、購買履歴の情報を使って「商品ごとに 1 行ずつ」にしても構いません。たとえば前者の形式であれば「たくさん購買してくれる顧客とそうでない顧客の違いはどこにあるか」と分析することができます

また「優良顧客を自動で見つけてくれる AI」というものを作ることもできます。同様に、商品ごとのデータにすれば「売れる商品とそうでない商品の違いは何か」を分析したり、「商品情報を入力すればそれが今後いくつ売れるかを教えてくれる AI」を作ったりすることもできるかもしれません。

つまり、最終的なデータを「何毎に 1 行ずつにするのか」ということは、「どのような切り口でデータを活用するのか」と考えることとほぼ同じ意味です。分析や予測について考える章で後述しますが、ざっくり言えば、マーケティングや営業などで顧客のことをよく考えなければいけない場合は顧客ごとに 1 行、仕入れや企画などで商品のことをよく考えなければいけない場合は商品ごとに 1 行、というデータ形式がよいでしょう。

ただし、どんな切り口でもいいかというとそうでもありません。すでに皆さんは活用のためのデータの条件として、「数十行以上必要」というものを学んで来たはずです。したがって「男女ごとに 1 行ずつ」というのではたった 2 行にしかなりませんので、このような形式のデータは少なくとも本書が考える「活用のためのデータ」ではありません。なぜなら、どんな高度な統計手法を使う分析も、どんなアルゴリズムを使う AI も、私の知る限りたった 2 行のデータから価値を生むことはないからです。

データ活用のための集計作業

顧客なり商品なりで 1 行ずつのデータを作ろうとすると、最後に必要となるのが集計作業です。一般的に集計作業というと「今月の売上は合計いくらだったか」とか「登録している顧客の男女比は何％ずつか」といったビジネスの概観をつかむためのものと考えられているかもしれませんが、より高度な分析や AI 開発のためのデータを用意するためにも集計作業は必要になってきます。

たとえば顧客マスターと販売履歴を結合した状態から、顧客 1 人に対して 1 行ずつ、というデータを用意しようとする場合、 1 人の顧客に対して複数行存在しうる購買履歴はどのように扱えばよいのでしょうか？ 1 人の顧客が何度も買い物をする可能性は十分にあり、また、 1 回の買い物ごとに複数の商品を買う可能性もあります。このうち 1 つだけの購買履歴を残し、ほかは捨ててしまう、というのはあまりにもったいないことです。そこで、複数行をまとめて 1 行ずつの項目にするために集計を行なうわけです。

エクセルでもデータベースでも、多くの IT ツールには図表 1 - 4 にまとめているような、集計のための関数が用意されています。

集計方法	エクセルの関数	SQLの関数
件数のカウント	COUNTIF関数	COUNT関数
複数の数値の合計	SUMIF関数	SUM関数
複数の数値の平均値	AVERAGEIF関数	AVG関数
複数の数値の最大値	MAXIFS関数	MAX関数
複数の数値の最小値	MINIFS関数	MIN関数

図表1-4　集計方法とExcelやSQLで使う関数

たとえば１人の顧客に対して複数存在する「(購買した商品の) 金額」という数値に対して、どのような集計が考えられるでしょうか？この件数をカウントしたものは「総購買商品点数」すなわち過去に何個の商品を買ったかという情報になります。またこの合計は顧客の「総購買金額」や「ライフタイムバリュー」と考えることができます。

さらに、平均を取れば「平均購買商品単価」と呼ぶことができます。この数値が高い人ほど平均的に高額商品を買ってくれた優良顧客だと解釈できるでしょう。ただし、過去に 101 商品の購買履歴があって、そのうち 100 個が 100 円で、１個だけが 100 万円だった場合に「平均購買商品単価が１万円」という形で集計してしまうと「100 万円という異常な高額商品を買ってくれたことがある」という情報が埋もれてしまいます。この場合、最大値という集計方法を用いて「最高額商品単価」と呼ぶ集計を行なってもよいかもしれません。また、もちろん「最低額商品単価」という集計を考えることもできます。

このようにさまざまな集計方法を考え、「顧客１人つき１行」として求められるデータの形式に加工すると、「(購買した商品の) 金額」という１つの項目から複数の列を生み出すことができます。この列はデータ分析における「説明変数」あるいは、AI における「特徴量」と呼ばれる素材となり、数多くあればあるほど、興味深い分析結果や高い精度での動作の可能性が高まります。

顧客ごとの購買金額の違いを「説明するかもしれない変数」として分析に用いたり、優良顧客を AI が識別するための「特徴」として考えたりしようというわけです。複数の項目を組み合わせてから集計したり、集計した項目同士で計算したりしても構いません。

ちなみに今回は「(購買した商品の) 金額」という、もともと「大小が意味を持つ数値」という項目を例に説明しました。これ以外の項目はどう扱えばいいでしょうか？購買履歴の方には「どんな商品を」「いつ」買ったか、という項目も含まれていますが、前述の通りこれらは「(大小が意味を持つ) 数字」でも、「(せいぜい数十個程度への) 分類」でもありません。次節ではこうした項目をどう扱うかを説明します。このような項目についても「数値化と再分類」という手順を踏めば、データ分析のための「説明変数」や、AI に用いる「特徴量」という形で利用できるようになります。

数値化と再分類でデータをもっとリッチに

活用できるデータの項目

前節では「顧客ごと」「レシート１行ごと」という粒度の異なる形式のデータを結合し、集計することで「活用のためのデータ」に加工する考え方を学びました。「活用のためのデータ」は顧客ごとあるいは商品ごとに一行ずつ、という形式にそろえた１枚の表となる必要があり、こうした作業が必要になります。

また、１枚の表になっているというだけではなく、データ分析で使える説明変数や AI で活用できる特徴量は、基本的に「（大小が意味を持つ）数値」であるか「（数十個程度への）分類」といったものでなければなりません。ここまで題材としたデータにおける、顧客の生年月日と住所、購買商品と購買日時といった項目はまだこの形式ではないので、「活用のためのデータ」にはなっていません。

せっかく蓄積されたデータを利用しない、あるいは捨ててしまうというのはたいへんもったいないことですので、こうした項目についても何とか、数値や分類という状態に加工することを考えてみましょう。

「大小が意味を持つ数値」とは

念のためこの「大小が意味を持つ」という意味を少し補足しておきましょう。たとえば身長が 170 ㎝だとか体重が 60 ㎏だとかの数値は「大小が意味を持つ」という条件を満たしています。170 ㎝の人よりも 180 ㎝の方が長身だとか、60 ㎏の人より 50 ㎏の人の方が軽い、という比較を私たちは日常的に行なっています。もちろん前節で取り扱った、商品の金額だとか、その件数をカウントした「総購買商品点数」といった集計値も大小が意味を持つ数値です。

先ほど「顧客の生年月日」という項目については、このままでは大小が意味を持つ数値ではない、と述べましたが、皆さんは生年月日から「年齢」というその大小が意味を持つ数値を計算するための方法をご存じのはずです。生年月日に限らず、現在の日付との差分を考えれば数値化できる日付データはたくさんあります。たとえばポイントカードをいつ作ったか、という登録日の情報があった時、現在日付との差分は「お店と顧客の付き合いの長さ」を表わすと考えられるでしょう。また、顧客ごとに複数ある購買日時に対して最大値（ つまり最新のもの）という集計を行った後、現在日付との差分を考えれば「最近来店してくれているか」を表わす数値だと考えることもできます。

これが郵便番号や野球の背番号だとどうでしょうか？たとえばハイフンなしで書くと、東京都庁の郵便番号は 1638001、仙台市役所の郵便番号は 9808671、鳥取市役所については 6808571 という数になります。これを仮に数字として扱った場合に「大小が意味を持つ」比較は考えられるでしょうか？地域の人口や所在地の緯度経度を示しているわけでもなく、単に郵便物を円滑に配達するための記号でしかありません。

同様に、高校野球であれば、レギュラーメンバーについてはキャッチャーが 2 番で、ショートが 6 番というように背番号が割り振られますが、「数が小さければ小さいほど偉い」というわけでもありません。異なるポジションを数字で区別しているだけです。

日本のスーパーマーケットの多くが、「何の商品を買ったか」という情報を JAN コードで管理しているかと思います。それ以外のビジネスでも商品を識別するための ID は、整数を用いていることが多いかもしれません。こうした数値は郵便番号や背番号と同様に「大小が意味を持つ」わけではないため、数値だからといってそのまま活用のためのデータにはなりません。

数字を使っていようが、文字を使っていようが、商品や場所などを「区別するための項目」であれば、そのように扱わなければいけません。ここで注意しておきたいのが「あまりに細かく区別するような項目は活用のためのデータにおいて役に立たない」というポイントです。

数十個程度に分類する意味と方法

活用のためのデータにおいて「もっとも細かく区別する項目」は、顧客ごとなら顧客 ID、商品ごとなら商品 ID、というように 1 行ずつの形式にまとめられた際のユニークな ID です。1 行ごとに重複なく 1 つずつの値が割り振られており、これ以上細かく分けることはできません。

このような ID は慣例的に、後々データを確認できるよう最終的なデータの左端に、「念のため」つけられることがありますが、実際にいざデータ分析をする、あるいは AI に用いようとする場合に説明変数や特徴量として使われることはありません。なぜでしょうか？

ここでは、顧客ごとに 1 行という「活用のためのデータ」を作って、「どのような顧客の総購買金額が高いか」という分析を行なう際のことを考えてみましょう。顧客ごとの購買金額の差を「説明するかもしれない変数」の候補として顧客 ID を含めていれば、「顧客 ID が 123456 の顧客の総購買金額は平均よりも 1 万円高い」という結果が得られるかもしれませんが、そんな情報は役に立つでしょうか？

おそらく役には立ちません。その条件を満たす顧客はこの世にたった 1 人しかおらず、今後別の顧

客にゲンがいいからと「顧客 ID を 123456 に割り振る」といったことをしても売上はまったくあがりません。単に顧客 ID がユニークでなくなり、データが活用できなくなるだけの話です。

優良顧客をリストアップするというだけなら、データ分析の必要などありません。後の章で詳述しますが、データ分析をする意味とは、データの背後に存在する「なぜか購買金額の高い顧客に共通した条件」を見つけることです。

仮にたった 1 人の優良顧客だけが当てはまる条件を見つけても、それは「たまたまその人が当てはまっているだけ」で今後役に立たない情報かもしれません。しかし、たとえば数十名以上の優良顧客には高い割合で当てはまって、数十名以上のそうでない顧客についてはほとんど当てはまらない、という条件を見つければ、統計学は「これがたまたまと言えるような傾向なのかどうか」を判断することができます。したがって、「顧客 ID が 123456 の顧客の総購買金額は平均よりも 1 万円高い」というような結果は、統計学を使ったデータ分析を行なうと「たまたまかもしれない」と判定されてしまうだけの無意味なものになってしまうでしょう。

これは AI でも同じことです。現在主流の AI 技術では「統計的機械学習」と呼ばれている計算が行なわれていますが、たった 1 人にしか該当しないような「顧客 ID がいくつか」という特徴を使って、顧客の購買金額を予測させるような AI を作ったとしても、新しい顧客が来た時にその情報は役に立ちません。なぜなら顧客 ID をユニークに管理し続ける限り、二度と「顧客 ID が 123456」という新規顧客が現れることはないからです。

このように、活用のためデータを 1 行ずつ完全に区別するような項目は「後々確認するため」以外の目的で使うことはできません。これが「あまりに細かく区別するような項目は活用のためのデータにおいて役に立たない」という状況の一番極端な例です。ではどれぐらいの細かさであれば活用のためのデータにおいて役に立つのでしょうか？

技術的な最低ラインの目安は、活用のためのデータにおいて「最低数十行程度は該当するもの／しないものがあるように」というものです。つまり、顧客ごとに 1 行ずつ、という形式に加工しているならば、最低数十人程度以上の該当する顧客もいるし、最低数十人程度以上該当しない顧客もいる、という分け方なら無意味というわけではありません。この水準を満たせば、統計学や AI が「偶然とは言いがたい」と、分析結果や AI のアルゴリズムに採用する可能性がないわけではない、ということです。

ただし、最初からとにかく細かく分けておけばよいかというとそうとも言い切れません。細かく分ければ分けるほど分析結果に目を通す手間が増え、その解釈から価値を生むアクションにつなげる際に困難が伴います。たとえば全国展開するスーパーマーケットのチェーンにおいて、最低数十人くらいは該当するだろうと、顧客の住所を細かく「何丁目」という区分で分析すれば、途方もない数の「大事なエリアがどこか」という情報が出力されることになります。その確認はたいへんな手間を生み、折り込みチラシなどの広告媒体をこの「何丁目」というレベルで管理するのもとんでもない手間です。

また AI を開発する場合にも、このような細かい粒度で区分するような特徴量を用いては計算量が増え、その分開発の時間やコストが跳ね上がってしまいます。だからといって実用上、今後の優良顧客の識別に役立つかというとそうでもありません。数十万人の既存顧客のうち、数十人だけが該当す

るような特徴を AI が利用したとして、そのメリットが得られるのは 1 万人の新規顧客のうちわずか 1 人だけ、ということになるでしょう。

分析結果が理解しやすくなるように、という私たちの慣例的な目安は「まずは数十個程度に分類」というものです。たとえば全国の顧客の住所を分類するのであれば、まずは都道府県のレベルに分けて考えます。あるいは同じ項目から「政令指定都市かどうか」「首都圏かどうか」という分類を考えても構いません。また同じ元データからこれら複数の分類が考えられる際に、同時に「活用のためのデータ」内に含めてしまっても構いません。個別の商品 ID も、ひとまずは生鮮食料品か、惣菜か、というように数十個程度の大分類を用いましょう。商品 ID として JAN コードを使っているのであれば、JICFS 分類という標準的な分け方を採用するのもよいでしょう。

また、地域を区別するような文字列（住所）や商品を区別するような数字（商品 ID）だけでなく、生年月日や購買日時といった日時についても、分類すれば説明変数や特徴量として使うことができます。「その日付が何月か」という分け方であれば 12 に分類できます。「何曜日か」であれば 7 つ、「月の上旬か中旬か下旬か」と 3 つに分けられます。いずれにしても数十個程度以内に分けることができれば、興味深い説明変数やよい特徴量として使うことができるでしょう。

なお顧客ごとに 1 行ずつという形式に加工すれば、購買した商品のジャンルを数十程度に再分類しても、その情報は 1 人につき複数存在しえます。顧客が複数回来店していれば、お菓子を買うことも雑貨を買うこともあるでしょう。このような場合も集計が必要で、「過去に何個のお菓子を買ったか」というカウントや「過去に買った商品数に占めるお菓子の割合は何％か」というように割合を計算します。

顧客 1 人につき複数行存在しうる購買日時についても同様で、「それが何月か」と分類した上で、「過去 12 月に購買した商品は何個あるか」とカウントしたり、「過去の購買に占める 8 月の割合は何％か」と割合を求めたりすることができます。このように分類したものを集計すれば、立派な「大小が意味を持つ数値」となるわけです。ただし、このような数値化や再分類ができるのも「データがキレイに管理されているかどうか」に大きく依存します。現実にはデータの抜け漏れや異常値、結合のキーと同様の表記の揺れ、という問題などにより、数値化や再分類が難しくなる場合があります。次節ではこれらの対処法について詳しく学んでいきましょう。

抜け漏れ・異常値・表記の揺れにどう対処するか

数値化や再分類を阻む「データの汚れ」

前節では日付やフリーワード、「あまりに細かすぎる区分をするID」などは、数値化や再分類といった加工をすれば「活用のためのデータ」に採用できることを学びました。この作業を困難にし、活用の際に意図しない誤りを生み出すのが、「データの汚れ」です。データが抜けていたり、異常値が含まれていたり、表記が揺れていたりする場合に、問題が生じます。本節でこの問題について詳しくみていきましょう。

データの抜け漏れが生みだす問題と対処方法

第1回目の記事で、最終的な「活用のためのデータの条件⑤」として「中身のセルに抜け・漏れは基本的に許されない」ということを挙げました。データ分析でも、AIのアルゴリズムでも、基本的に1か所でもデータが空白となっていれば、その行は丸々使うことができません。データ分析やAI開発に使うツールによっては、こうした空白がデータに含まれているだけでエラーになって動かない、というものもあります。

そうでなくても分析や予測の精度が大きく低下してしまうことがあります。その理由は「1か所でもデータが空白となっていれば、その行は丸々使うことができない」という状況でエラーとならないように、ツール側で行われる対処法にあります。すなわち、「使えない行」があれば内部で削除してしまう、という処理によってエラーにならないようにしているからです。

元々のデータのサイズが大きく、たとえば100万人以上の顧客データがあるから、多少抜け漏れのある行を削除してもかまわないのではと考えられるかもしれませんが、怖いのは「1か所でも」という部分です。元のデータに利用可能な1000項目の情報が入っていたとして、そのうちわずか1%だけがまったくの偶然で「抜け漏れ」だったとし

ましょう。これらを使って 100 万人の顧客に対して 1 行ずつのデータを、抜け漏れに何も対処もせずに加工していった場合、「1 か所も空白にならない」顧客は何人いるでしょうか？答えは 100 万人× 0.99 の 1000 乗で 43 人だけ、ということになります。

せっかく顧客が 100 万人いて、1000 項目もの利用可能な情報が含まれる素晴らしいビッグデータを持ちながら、知らず知らずのうちにツールの内部でそのほとんどが捨てられ、わずか 43 人だけの「たまたまったく抜け漏れのない顧客」だけが使われていることになってしまいます。これでは大した分析結果は得られませんし、AI の性能も出ません。このような問題に対してどう対処すればよいでしょうか？

活用のために数値化するのか、再分類するのかということにより、データの抜け漏れへの対処方法は異なってきますが、再分類の場合は比較的単純なやり方があります。それは「不明」とか「元データなし」といった形に分類するという方法です。

つまり、なぜか購買した商品 ID が空白のままのデータとして存在していた場合、ふつうに商品大分類という形に再分類しようとエクセルを操作したり、プログラムを書いたりすると「どの商品大分類でもない」ということで最終的な「活用のためのデータ」においても空白、あるいは内部的に「エラー」という扱いのままのデータが含まれてしまう可能性があります。これをそのままにしてしまうと、前述のような問題にぶつかってしまうわけです。

このような場合、商品 ID が空白なら明示的に「商品大分類が不明」というように分類するわけです。物ではなく何かのサービスを提供したなど、データの空白が生じる理由が何かしら、あるのかもしれませんが、少なくともいったんエラーにならないように処理すれば分析や AI での利用は可能になります。

抜け漏れを含むデータを再分類する際にはこのような対処法が考えられます。数値化ではどうでしょうか？「抜け漏れ」という分類は考えられても、「抜け漏れ」という数値がいくつかはわかりませんのでそう単純にはいきません。統計学の専門用語ではこのような状況を「欠測データ」と呼んで、その対処法だけで 1 冊の本が作られるぐらいなので、興味がある方はそちらを参照してみるとよいでしょう。比較的初歩的な方法としては「エラーにならないように元データの時点で除外する」あるいは「何かそれらしい値を補完する」という考え方があり、後者は専門用語では単一補完法と呼ばれます。

このうちどちらを使ったらよいか、また「それらしい値」とは何かというのはケースバイケースです。購買した商品の金額が抜けているとは、無料のノベルティグッズの受取りなどで「0 円」という取引の意味を示しているのだとしましょう。この場合「そういうへんな購買履歴はノーカウントにした方がいい」と考えるのであれば、集計の前に除外してしまうというのが前者の「除外」という処理です。あるいは、0 という値を補完しても構いません。

基本的にこのような抜け漏れがほとんどないようであればどちらの処理をしてもそう結果に影響はしませんが、細かく言うと「購買商品の平均単価」という集計をする際などには、微妙な差異が生じます。すなわち、ノベルティグッズを除外した上で平均値を求めた方が、「0 円」という扱いで平均値の計算に用いた場合よりも多少大きな値となるでしょう。極端なケースを挙げると、1 度ノベルティグッズを受け取るついでに、1 万円の商品を

買った、という購買履歴だけが存在する顧客について、前者の処理をすれば「平均１万円」、後者の処理をすれば「平均５千円」ということになります。判断に困ったら、このような極端なケースを考えて、どちらがイメージに近いか考えてみてもよいでしょう。

また「０」という値での補完が問題という状況も実際には起こります。さすがに現在のスーパーマーケットのレジなどではほとんど発生しませんが、店員の不注意でお金は受け取っているものの、データとして入力を忘れている、という状況ではどうでしょうか？このような場合「０円」ということは基本的にないはずです。あるいはもっと現実的にこうしたデータの抜け漏れが起こりうる状況を挙げましょう。顧客に回答してもらったアンケートを分析しようとする場合に、年齢や予算といった数値情報について未回答、ということはしばしばあります。このような場合も「０才」「予算０円」と扱ってしまうのは不適切でしょう。

こうした場合、「それらしい値」として、抜け漏れのない他のデータの平均値を採用するという処理がしばしば行われます。たとえばスーパーマーケットの店員が入力し忘れた商品の金額として、１円といった値は非現実的ですし、100 万円という金額も非現実的です。スーパーマーケットであれば１商品の単価は概ね百数十円程度から数百円程度のものではないか、というのが「それらしい」気もします。高級感を売りにしているのか、お買得感を売りにしているのかというお店の方向性でも変わってくるでしょう。そこで、抜け漏れのない他のデータから、「顧客が購買する商品の平均単価」を計算すればそれが「それらしい値」であると考えられるわけです。

顧客が年齢を記入していなかった場合も同様です。それが５歳児と考えたり、100 歳のご長寿と考えたりするのも非現実的ですが、その中間の「それらしい値」として、いったん回答の得られた他のデータで平均年齢を計算し、それが 40 歳なら未回答者の値を「40」と補完する、というのが現実的な対処方法です。

もちろんデータの欠測についての本が書かれるぐらいなので、このような取り扱いの問題がないわけではありませんが、抜け漏れがごくわずか（ たとえば１％未満）であれば、分析結果や AI の精度にそれほど大きな影響はありません。

また、何割ものデータが抜けている、という数値の項目が存在していた場合、そもそもこの項目自体を信用できないものとして、除外してしまうというのも一つの方法です。いっそ数値を敢えて「再分類」してしまうという考え方もあります。

たとえば何割もの回答者について年齢が未回答となっているのであれば、まず年齢を 10 代、20 代、30 代、…70 代以上、年齢未回答といったように再分類してしまった方が、ムリに平均値を補完するよりよい結果が得られる、という考え方です。

また、ここで述べたことは「元々数値の入った項目」だけでなく、「生年月日から年齢を算出」といった日付型の数値化の際にも当てはまります。生年月日が空白になっていれば、そこから自動的に年齢を計算しようとしてもやはり空白やエラーになってしまうでしょう。このような場合も、平均年齢で補完したり、（ 計算された）年齢を最終的なデータから削除したり、「年齢不明」というカテゴリーを含む再分類を行うことができます。

異常値による問題と対処方法

元データに抜け漏れが存在している場合だけでなく「あり得ない値が入ってしまっている」という

状況も、データ活用の障害になることがあります。商品の金額として、操作ミスなどの事情で「999,999 円」というデータが入力された場合どのようなことが起こるでしょうか？

こうした異常値も、人間が目視で確認するという作業をはさめば「業務のためのデータ」としてはそれほど問題にならないかもしれません。スーパーマーケットに約 100 万円の商品が存在することはほぼありえず、9 という数字が並んでいればそれだけで目立ちます。したがって、日々の業務に携わる人たち同士では誤りは明らかで、その日のレジを〆る時に事情を話して帳簿には残らないようにすれば経営上問題はないかもしれません。しかし、「活用のためのデータ」にこうしたミスが少なからず含まれてしまうと話は変わってきます。

データ結合の時に述べたように、活用のためには何万行だとか時に何百万行ものデータを扱うことがあります。このとき、どこかの店舗で「この日入力ミスをしてしまった」という情報は、たいていの場合分析や AI 開発をする人と共有されず、一行一行個別に目を通すこともできません。

そうすると、優良顧客の特徴を見つけるための分析や、優良顧客を識別する AI は、実は正しい「優良顧客」ではなく「入力ミスをされやすい顧客」を見てしまっていることになってしまうかもしれません。毎週約 1 万円ずつ購買してくれる顧客の 1 年間の購買金額は 50 万円ほどですが、そうした顧客が一度こうした入力ミスに遭遇しただけで、いきなり客単価が 3 倍に増えてしまうわけです。その結果、「入力の管理がいい加減な店舗の近くに住んでいる顧客は優良顧客になりやすい」といった分析結果や AI が得られてしまう危険性に注意する必要があります。

また数値だけでなく、日付からの数値化についても、こうした異常値の問題は起こり得ます。典型的な例としてはシステム内のテストデータとして「西暦００００年１月１日」という日付に生まれた、あるいはポイントカードを作ったという架空の顧客が、活用のためのデータに紛れ込んでくることがあります。私たちはこうした状況を指して「キリスト世代」と呼んでいます。

こうした異常な日付に対して何も対処せずに年齢や取引年数を算出すれば、当然「2 千歳以上のご長寿」や「2 千年以上取引のあるお得意様」といったあり得ないデータが得られてしまいますが、もちろんこれは適切ではありません。データの加工の際にはそれぞれの項目の最大値と最小値を確認して、異常な値が含まれていれば抜け漏れと同様に対処しましょう。つまり、排除するか、それらしい値で埋めるか、「不明」という分類にするか、ということです。

表記の揺れへの対処方法

データ結合の際に ID が適切に設定されていない例として「表記の揺れ」について述べました。氏名というフリーテキストの項目をキーとして使おうとすると、「渡辺か渡邊か」「姓と名とのスペースは半角か全角か」のような表記の違いにより、適切にデータを結合できません。このような表記の揺れは、数値化や再分類する場合にもやはり問題になります。

数値化の際に問題となる表記の揺れの典型例には「フリーテキストで書かれた日付」というものがあります。たとえば「業務のため」に人間が生年月日を読み取って、重要な顧客の誕生日を祝おうというだけなら、同じ日付に対して次表内のどのような表記を使っても問題はないかもしれません。

日付の表記	書き方の規準
1963/3/4	日本国内では最もよく見られる形式
1963-3-4	システムによってはスラッシュではなくハイフン区切りのものも
1963年3月4日	年や月を漢字で表記し数字が全角という場合も
昭和三十八年三月四日	世代によっては和暦や漢数字で書く人も
March 4th, 1963	アメリカ式の丁寧な書き方
3/4/'63	アメリカ式のカジュアルな書き方
4 March 1963	イギリス式の丁寧な書き方(アメリカと順番が違う)
4/3/'63	イギリス式のカジュアルな書き方(アメリカと順番が違う)

スラッシュで区切られていようが、ハイフンで区切られていようが、漢字を使っていようが、和暦であろうが構いませんし、外国人などであれば月を英語の略称で表現することも、日本人とは異なる順番で書く人もいます。いずれであっても人が見れば「あ、今日はこの顧客の誕生日だ」と判断することができます。その判断に自信がなくても「ひょっとして今日お誕生日ですか？」という会話が顧客との間で生まれることはそう悪いことではありません。

しかし、個別のデータを参照するのではなく、数万人以上の生年月日から年齢を計算しようとすれば、何らかのツールを使う必要が出てきます。こうした場合、いちいちデータの中身を確認し、どのような表記方法が使われているかの仕分けをしながら日付の形式をそろえて、そこから年齢計算をするプログラムを書かなければなりません。たいていの場合こうした作業は試行錯誤を伴い、見落としている例外的な状況のせいでエラーになっているところがないかを確認する必要があります。たとえば同じ英語圏でもアメリカ式とイギリス式で日付を書く順番が異なり、「25 月 12 日」という不正な日付の値が現れることもあります。
日付だけではなく、住所などについても同じことがいえます。住所というフリーテキストから都道府県ごとに分類をしたい場合を考えてみましょう。すべてのデータが必ず都道府県から始まっていれば「住所の最初の 2 文字が何か」というだけで都道府県は特定できますが、必ずしもそうではありません。政令指定都市などの住所は都道府県を省略して書くことが多いですし、自宅近辺の店で住所を登録する際には「どうせ近所だから郵便物も届くだろう」と、市町村すら省略して住所を記入する人もいます。大阪市福島区の人が市を省略して書けば「住所の最初の 2 文字が何か」という判定では福島県民ということになってしまいます。
こうした住所の表記についても、業務のデータという観点では「郵便物が届くから大丈夫」ということで問題になりませんが、活用のときにはエラーや抜け漏れのリスクが生じたり、それを回避するためのたいへんな手間がかかったりすることになります。

このような表記の揺れに対して、私たちはどう対処すればよいでしょうか？

正攻法としては、活用に耐えられるようにすべてのデータをきっちり構造化しておくべきです。都道府県という情報が必要なら、市区町村以下の住所をフリーテキストで記入し、都道府県については必須入力で選択させる、という形式にすればよいでしょう。また日付についても年、月、日、をそれぞれ別々に選択させたり、形式のチェックを厳密にしたり、といった対処をしておきます。それ以外の情報も、データとして分類できるように、数値として使いたいのであれば半角の数字だけで、といったように最初からデータ活用をしやすい形でデータを入力しておけば、いざ活用しようとする際に余計な手間がかかりません。

ただ、次のシステム改修を待たずにデータ活用を進める必要がある場合は、そうした正攻法は難しいかもしれません。現実的にどう進めていくべきでしょうか？
こうした場合、ある種の割り切りが役に立ちます。データの形式をそろえることや、正確な数値化や分類に労力がかかりすぎるようであれば、８～９

割程度の数値化や分類が終わった状態で、それ以外を抜け漏れと同じように対処してもよいでしょう。もちろんこうした項目を丸々「いったん後回しにする」という判断をしても構いません。

これは抜け漏れや、異常値についても同様のことが言えます。データをキレイにすることはあくまで活用のための手段です。ある程度の割り切った判断をしながら「ひとまず今あるデータを使って、できる範囲で分析や AI 開発を進めてみる」ことがおすすめです。

次節は締めくくりとして、現実的なデータ活用のプロセスのために、どのような姿勢でデータ整備を進めていけばよいかを説明しましょう。

継続的なデータ活用プロセスにおける データ整備の位置づけ

データ整備のサグラダファミリア

ここまで「業務のためのデータ」をどう「活用のためのデータ」に加工するかを詳しく説明してきました。複数の表を結合するためのキーを確認し、それぞれの表の中に含まれる対象のデータを確認し、最終的にどのような切り口で１行ずつにまとめるのかを決めて、それぞれの項目を数値化したり分類したり、必要に応じてさまざまな集計をしたものが活用のためのデータです。その過程では、抜け漏れや異常値、表記の揺れなど、適切に対処しなければいけないことがあります。

データ分析においても、AI 開発においても、多くの場合、実はこうしたデータの加工やそのためのクリーニングなどの作業に８～９割もの時間や工数が費やされます。つまり「高度な統計学や機械学習の知見」を持った専門家に、不完全な「業務のためのデータ」を丸投げしてしまったのでは、専門的知識を発揮させることなく、本章で述べた泥臭い作業にその労力のほとんどをかけさせるという大きなムダが生まれることになります。

一方で、データをキレイにできさえすればよいか、というとそうでもありません。私たちが今まで見てきた企業の中には、データ活用を進める前に、何年もの期間と何億円もの予算を投じて「データ自体を完璧にする」ことに心血を注がれていることがあります。多くの場合、そうした努力がもたらしてしまう残念な状況のことを、私たちは「データのサグラダファミリア」と呼んでいます。

キーとなるIDも含めたすべての項目について、抜け漏れもなく、異常値もなく、表記の揺れもない完全な状態というのはもちろん美しいものです。その一方で業務やITシステムというものは日々変化していくものです。ある時点で過去に存在するデータをすべて詳細に目視で確認し、表記の差異や含まれうる異常値、抜け漏れ、またIDの体系などを確認するためには、最低数ヶ月から1年ほどの時間と、それなりの人員を必要とし、そうした問題と無縁なシステムを設計し、導入するためにもやはり数ヶ月から1年以上の時間がかかるでしょう。この間にも、新製品を扱うようになったり、他社と合併したり、組織構造や社内外のルールが変わったり、社内に新たなシステムが導入されたりします。そうすると、「最初に確認した時点では存在しなかった整合性の問題」が新たに生まれてしまうかもしれません。

そうなるとまた、詳細に確認して、どのようにシステムを改修すべきかを整理して…という作業が必要になってきます。これが100年経っても、設計者が亡くなっても、なかなか完成しない「サグラダファミリア」のようだというわけです。

データをキレイにすることの価値とは

幸いにしてこの「サグラダファミリア」の完成にこぎつけた企業もあります。だからといってそのキレイなデータを上手く使って、利益につなげられたかというとそうとも言い切れません。せっかく何億円もかけてデータを整備し終わったというのに、そこから大した活用アイデアが生まれず、どうしたらよいだろうか?といった相談も、私たちのところにはしばしば舞い込みます。

このようなことになぜなるかと考えると、そのような企業にとっても、データ整備を依頼された企業にとっても、データ整備自体が自己目的化してしまったからでしょう。先ほど「統計学や機械学習の専門知識とは別の」と述べましたが、一定レベルのIT技術者であれば、データ活用のことを知らなくてもデータをキレイにすることはできるでしょう。

ただ、いざデータをキレイにした上でそれをどう使うか、どう活用すれば価値が生まれるか、といったことを必ずしもイメージできるわけではありません。

社内に存在するさまざまな項目について、データをキレイにすることの価値は等しいわけではありません。当該企業の事業内容や、その中でデータを活用しようとするユーザーが誰なのか、どのような分析、あるいはAI開発をしようとするのか、といった目的によって大きく異なります。そうした事情によって、何としてでもキレイにした方がいい項目もあれば、まったく使い道のない項目もあるわけです。これはデータ分析やAI開発の視点がある人間にとっては比較的容易に判断できることですが、一般的な技術者にはそうでもありません。

まずは重要な項目だけに集中して、さっさと活用のフェーズに進んだ方が効率的ですが、整備はできても活用のノウハウのない外部のIT企業にとっては「とにかく全部キレイにしましょう」と提案した方が大きな売上につながることになります。したがってIT予算が潤沢な企業ほど、つい「サグラダファミリア」に陥りやすいのかもしれません。

では、内部にデータ分析やAI開発のできる人材がいない企業はどうやってこのような判断を行なえばよいのでしょうか?その答えは「まずは試しにやってみる」というものです。システムを改修する前に、1サイクル分のデータ分析をしたり、AIのプロトタイプを作ってみたりしましょう。本章の

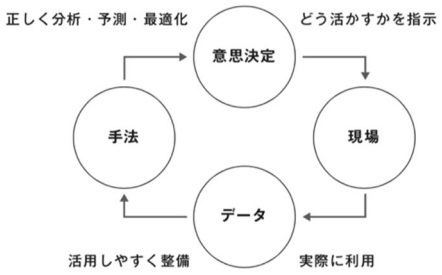
知識があれば、その過程でデータにどのような問題があるか気づくことができます。また多少の手作業は必要になったとしても、「サグラダファミリア」の完成を待つより遥かに早く、一定の成果を見ることができるはずです。

ここまで来れば、分析や AI の活用のためにとても役に立った項目と、そうでもない項目がわかります。したがって、この時点で継続的に役に立つであろう重要な項目を重点的に、抜け漏れや異常値、表記の揺れなどが生じないようシステムや運用のルールに対策を講じれば、データ整備を効率的に進められます。

また、「サグラダファミリア」どころか、データをキレイに整備しようという機運がまったく感じられない組織においても、このような「試しに現状のデータでやってみる」というアプローチは有効です。

このような組織では多くの場合、誰かが「データを整備しましょう」と提案すると「整備することでどのような、いくら位のメリットが得られるのか」と質問されます。しかし、活用のイメージがない技術者は「どのようなメリットが得られるか」という問いには答えられません。データの形式を見れば、すぐに活用のイメージがいくらでも浮かぶ私たちでも「いくら位のメリットかあるか」と正確に約束することはできません。なぜなら、実際にデータ分析をしてみなければ、「どうすればどれくらい儲かりそう」という結果は得られないからです。同様に、「どれくらいの精度で動作する AI ができるか」「それによってどれくらい経営上のムダが省けるか」といったことを予めわかる人はどこにもいません。「とにかく現状のデータでやってみる」ことで、こうした成果の目安が見え、データ整備のコストと見合うかどうかという根拠が提示できるようになるわけです。

データが窮屈になる瞬間



図表0-1　データを活かす組織の理想（再掲）

データ活用というのは継続的なプロセスです。最初の 1 周目は、現状のデータを何とか分析や AI のために役立つように加工して、とにかく早い段階で分析手法やアルゴリズムで処理ができるようにしましょう。

その成果をどう活かすか意志決定し、現場に届けて、どの程度の利益につながったか、ということをデータで評価できるようにしておきます。

このように、適切にサイクルを回していくと、やがて多くの人はデータに対して不満を覚えるようになります。

それは本章で説明した「データの汚れ」に対してもですが、「こんな項目も取っておけばいいのに」「こんな情報があればいいのに」という、元々のデータに「何が存在していないか」ということが気になってくるわけです。この状態を私たちは「データが窮屈になる」と表現しています。本章ではスーパーマーケットにおける顧客と販売履歴というデータについて考えてきました。これらを使って一通りの分析を終えると、「JAN コードがあっても、その商品ジャンルをいちいち分けるのが手間」「商品ジャンルはわかってもその中にどのような成分が入っているのかわからない」「店舗の

特徴や周辺地域の環境によって顧客の購買に違いが出るはず」といったデータに対する「欲」が出てきます。

昨今は IoT や AI を使い、データを収集する仕組みが商品化されていますが、こうした「仕組み」の導入は、このような「欲」が出てきてからでいいかもしれません。すでに存在する社内のデータさえ活用できていない状態で新しいデータを収集しても、たいてい宝の持ち腐れとなります。高度な仕組みを使い、大量のデータを収集しておきながら、「これをどう活かしていいかわからない」という相談を私たちはしばしば受けます。また、いざデータを活用しようとしたタイミングで、既存のデータと新しく収集されたデータを合わせる際に、データ整備の余計な手間が生じることもあります。

データが活用できるめどが明確について、その中の課題がわかっている状態であれば、欲しいデータを簡単に収集できる仕組みの価値は、正確に判断することができるでしょう。

さらに、自前でデータを収集するよりも、すでに外部に存在しているデータを買う方が効率的な場合もあります。世の中にはさまざまなデータを収集し、販売している事業者がたくさんあり、より速く正確に、活用しやすいデータを提供しています。

たとえば eBASE という会社は全国で流通する食品や日用雑貨についての成分情報やパッケージの材質などのデータを持っています。ゼンリンデータコムや昭文社という地図の会社は、全国のエリアを細かく(町丁字ごとや250mずつのメッシュなどに)区切ったエリアごとに、どのような施設があるか、といったデータを提供しています。ソフトバンク系列の Agoop や、ドコモインサイトマーケティングでは、携帯電話から収集したデータを用いて、メッシュごとにどれぐらいの人通りがあるのか、といったデータを販売しています。BB においても、取引先や見込み顧客の法人についてのデータが欲しければ、帝国データバンクや東京商工リサーチという会社に相談することができます。

自らの店舗や施設の周辺にカメラをつけて、人通りがどのくらいあるかをリアルタイムに計測する AI、というのは現代の技術力で可能になっています。そうしたシステムを販売している企業も１つや２つではありませんが、最終的な活用方法が「何曜日の何時ぐらいにタイムセールや呼び込みを行なえばいいか知りたい」ということなら、１秒ごとの正確な数字は不要で、継続的に測定する仕組みもいりません。ドコモインサイトマーケティングや Agoop、ゼンリンデータコムでも、このような活用に必要なデータは販売してくれます。自分たちの店舗以外の地域についてのデータも持っているため、今後の出店や進出にも役立つかもしれません。

ぜひ、以上のような知恵をもとに、活用可能なデータの幅を広げてみてください。

date Ferry を使ったデータプレパレーション

第 1 章では「どのような形式にすればデータが活用可能な状態になるか」を説明しました。活用のためのデータを用意するには、キーの形式をそろえて複数のデータを結合し、何毎に 1 行ずつのデータになるかを考えて集計して、数値化したり分類をまとめたり、抜け漏れや異常値、表記の揺れをそろえたりといった作業が必要です。これらは数千行程度のデータであればエクセル上でもできる作業ですが、数万行を超えてくると徐々に動作が重くなり、数十万行以上になってくるとエクセル上での作業はあまり現実的なものではなくなります。

もともとビッグデータという言葉も「一般的なパソコンのメモリ上で処理しきれない」というところから生まれた概念ですが、多くの企業にとって顧客や販売、ウェブサイトのアクセスや工場にある機械の動作の履歴といったデータは十分に大きい「ビッグデータ」です。

このような規模のデータを処理しようとする際には、ふつうデータベース（ 正確にいうとデータベースマネジメントシステム)が必要になります。たとえば Oracle やマイクロソフトの SQL Server、オープンソースなものでは Postgre SQL や MySQL といった製品などです。おそらく皆さまの会社にある多くの業務データもこうしたデータベースの中に存在していることでしょう。

これらの製品名についている「SQL」とは、Structured Query Language すなわちデータを取り出す「問い合わせ(クエリ)」を行なうための構造化された言語という意味です。この SQL を知っていれば蓄積されたデータを必要な形式で取り出すことができ、本章で述べたような結合や集計、抜け漏れの補完などの作業を行なうこともできます。一時期ビッグデータという言葉が脚光を浴び始めていたときには、これらのデータベースよりも高速に大規模なデータが処理できるという触れ込みで、Hadoop や NoSQL といった新しいデータ管理の仕組みが提案されましたが、現在ではこうした新しい仕組みについても SQL による データの操作ができるようになっています。

しかし、こうしたデータベースに対して「 活用のためのデータ」を作成するために SQL を直接入力する方法は、少なくとも 3 点ほどの問題があり、あまり現実的なものではありません。

1 つめは、「活用のためのデータ」を加工できるレベルで SQL を使いこなせる人材が限られることです。SQL は広く普及したプログラミング言語ですが、その形式は C や JAVA、Python といった一般的なソフトウェア開発に使われる言語とは異なり、エンジニアにとって「一応使えるけどそれほど詳しくはない」というレベルが一般的です。活用のためのデータを作る際には、時に数百行から数千行というとんでもない量の複雑な SQL を書く場合もあります。データを蓄積するとか、売上高の合計を定期的に集計して取り出す仕組みを作るレベルのことを理解しているだけのエンジニアでは、こうした複雑な SQL を書くにはとても時間がかかります。したがって、現在、企業がデータ活用を進めようとすると、データ取得のためだけに、データベースの操作に長けた専門のエンジニアを 1~2 名外注することが必要になります。これだけで毎月 200 万程度のコストが発生しますので、それをペイできるだけのアイデアがなければデータ

活用は進みません。データを取り出す専門家と、データを活用する現場との間で認識に齟齬があり、データを取得するまでに時間がかかり過ぎて、活用の商機を逃してしまうこともあります。

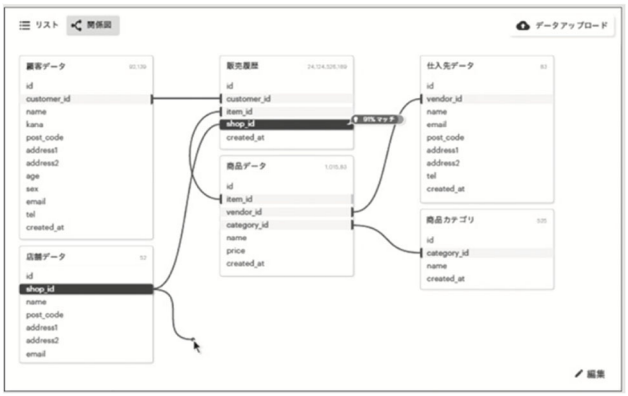
2 つめの問題は、SQL の操作ができる技術者がいたとしても「業務のためのデータベース」に対して負荷をかける作業は望ましいものではありません。こうした業務用データベースは、会計処理や、生産管理といった目的のために作られています。万一、データベースが止ってしまえば、お金の受取りや、支払い、工場を回すことができなくなってしまうリスクがあります。業務のためのデータベースは、顧客がたくさん買い物をしようが、工場が繁忙期を迎えようが止ることのない性能で設計されているはずです。しかし、このデータベース上で複雑な集計処理をする SQL 実行したり、誤ればとんでもない計算量の SQL を実行したりすると、想定していた以上の負荷がかかってしまうことになります。

ではまず、最低限のデータを業務用データベースから取り出し、それを別のコンピュータに持ってきて、そちらで活用のためのデータを準備するのはどうでしょうか?現在データ活用を進めている企業の多くがこうした方法を採っています。SQL が得意なエンジニアがいれば、別途分析用のデータベースを用意し、その上で加工することになるでしょう。JAVA や Python などの、エンジニアが得意な言語での加工をするという会社もあります。

ただ、このやり方にも「もったいない」という 3 つめの問題があります。業務負荷を考え、データベースを設計し、その上でどのような加工が適切かを考えてプログラムを書く、というのはそれなりに手間のかかるエンジニアの仕事です。業務のためのシステムやソフトウェア開発と違い、「最終的にどのような加工が必要か」という判断を事前に行なうことは困難です。実際にデータを加工して分析したり、機械学習手法を適用したりしないと、その加工にどれだけの価値があるのかわかりません。実際にはまったく別のデータが必要だったということもあります。したがって、せっかく構築した環境や、大量の工数をかけて書いたプログラムの多くが「使い捨て」になってしまうこともあります。

「じゃあどうすればいいんだ」となりますが、私たちはこうした現状を解決するために「date Ferry(データ・フェリー)」というツールを開発しました。データプレパレーションツールすなわち「データを準備するためのツール」と呼ばれる製品の一種で、誰でも簡単に素早く、「活用のためのデータ」を取得することができます。

情報システム部門に確認してデータベースに接続する情報を入力すれば、それだけで必要なデータを取得することができます。またファイル形式で入手したデータを加工していくこともできます。(df1-1)



顧客のデータか、購買履歴か、というデータベースの中身の「どの列が必要なのか」も簡単に選べ、顧客のうち女性だけ、あるいは年齢が一定以上だけ、というような活用のためのデータに欲しい データの条件を指定することもできます。(df 1-2)

複数のデータに対して操作を行なった後、データを結合したければ、レゴブロックを結合するように、キーを指定するだけで結合操作も可能です。特定のキーごとに集計処理を行なうことも簡単にできます。(df 1-3)

日付の表示形式を変えたり、差分を取って経過日数を計算したり、数値を丸めたり、四則演算や異常値の排除、空白箇所を別の値で置き換えるなどの必要な機能がいくつも用意されていますので、それらを使ってデータをクレンジングしていきます。その処理後は、加工済みのデータを自分の手元にダウンロードできます。

ここまでの処理はすべてセキュアなクラウド上で完結しますので、システム構築やプログラムを書く手間を大きく省くことができます。また社内のオンプレミス環境でデータを扱う必要がある場合、サーバーを立てる必要はありますが、それでも専用のデータ処理基盤を構築するよりは素早く、低コストで実現することができます。

私たちはこのようなツールを通して、データ活用のための準備がよりスムーズに運んでいくことを望んでいます。