Feature Engineering (L'Art de Créer des Variables Pertinentes)

Objectif Général : Créer de nouvelles variables qui capturent des informations non directement évidentes dans les données brutes, mais qui sont cruciales pour prédire les ventes. Ces variables aideront notre modèle à comprendre les tendances, la saisonnalité, l'impact des événements, et les caractéristiques uniques de chaque magasin.

Outil Principal : Alteryx est excellent pour ça ! Nous utiliserons des outils comme ***Formula, DateTime, Multi-Row Formula, Generate Rows, Join, Summarize.***

Catégories de Feature Engineering et Exemples pour Rossmann :

Je vais diviser le Feature Engineering en catégories logiques pour bien comprendre le type de variables que nous allons créer.

Catégorie 1 : Caractéristiques Temporelles (Issues de la colonne Date)

Les ventes sont très sensibles au temps. Il faut extraire toutes les informations pertinentes de la date.

Variables à créer :

Année : Year([Date]) -> Pour capter les tendances annuelles.
Mois : Month([Date]) -> Pour la saisonnalité mensuelle (ex: décembre pour Noël).
Jour du mois : Day([Date])
Jour de la semaine : DayOfWeek([Date]) -> Déjà présente mais peut-être à retraiter (numérique 1-7).
Semaine de l'année : Week([Date]) -> Pour la saisonnalité hebdomadaire/mensuelle.
Jour de l'année : ToNumber(DateTimeFormat([Date], "%j")) -> Pour les cycles annuels plus fins.
Est un Week-end ? :IF DayOfWeek([Date]) IN (6,7) THEN 1 ELSE 0 ENDIF -> Impact important sur les ventes.
Est le Début/Fin du mois ? :IF Day([Date]) <= 5 OR Day([Date]) >= 25 THEN 1 ELSE 0 ENDIF -> Les gens sont souvent payés/achètent plus en début et fin de mois.
Nombre de jours depuis le début du dataset : Utile pour modéliser une tendance linéaire globale. DateTimeDiff([Date], Min([Date]) over (), 'days') (c'est un exemple plus complexe qui nécessite un Multi-Row Formula ou un Summarize initial pour le Min(Date)).

Outils Alteryx : Principalement l'outil **DateTime** pour extraire jour, mois, année, semaine. Et l'outil **Formula** pour créer des variables booléennes comme IsWeekend.

Catégorie 2 : Caractéristiques Liées aux Promotions (promo & promo2)

Les promotions sont un moteur clé des ventes. Il faut les analyser en détail.

Variables à créer **(à partir de Promo2SinceWeek, Promo2SinceYear, PromoInterval après imputation) :**

Durée depuis le début de Promo2 : DateDiff entre Date et la date de début de Promo2 (si Promo2 est actif).

Ceci est plus complexe : il faut reconstruire la date de début de Promo2 à partir de Promo2SinceYear et Promo2SinceWeek, puis calculer la différence.

Exemple (concept) :IF [Promo2]=1 THEN DateTimeDiff([Date], DateTimeParse(ToString([Promo2SinceYear]) + "-" + ToString([Promo2SinceWeek]) + "-1", "%Y-%W-%w"), 'weeks') ELSE 0 ENDIF (Le DateTimeParse avec semaine est délicat). Une approche plus simple est de créer une "Promo2Active" qui indique si une promo2 est active à la date donnée.

Indicateurs d'intervalle de Promo2 : La colonne PromoInterval ("Feb,May,Aug,Nov") est une chaîne. Il faut en extraire l'information.

IsPromo2Month : IF [PromoInterval] LIKE "%" + DateTimeFormat([Date], "%b") + "%" THEN 1 ELSE 0 ENDIF (Utilise le format court du mois, ex: Jan, Feb).

DaysSinceLastPromo2Start : Nombre de jours depuis le début du cycle de la Promo2 (si applicable). Très avancé, peut-être pour plus tard.

Outils Alteryx :Formula (avec des fonctions de manipulation de chaînes et de dates), DateTime pour la construction de dates.

Catégorie 3 : Caractéristiques Liées à la Concurrence

Variables à créer (à partir de CompetitionDistance, CompetitionOpenSinceMonth, CompetitionOpenSinceYear après imputation) :

Temps depuis l'ouverture du concurrent :DateDiff entre Date et la date d'ouverture du concurrent.

CompetitionOpenDurationDays : IF [CompetitionOpenSinceMonth] != 0 THEN DateTimeDiff([Date], DateTimeCreate([CompetitionOpenSinceYear], [CompetitionOpenSinceMonth], 1), 'days') ELSE 0 ENDIF (Prend le 1er du mois comme date d'ouverture). Les Nulls imputés à 0 serviront de base pour le ELSE.

HasCompetition :IF [CompetitionDistance] != 100000 THEN 1 ELSE 0 ENDIF -> C'est une variable binaire qui indique la présence de concurrence. C'est plus explicite pour le modèle que juste la distance.

Outils Alteryx :Formula, DateTimeCreate, DateTimeDiff.

Catégorie 4 : Caractéristiques des Magasins et Environnement

Variables à créer (à partir de StoreType, Assortment, StateHoliday, SchoolHoliday) :

One-Hot Encoding des catégories : Les modèles ML ont du mal avec les catégories textuelles (a, b, c, d). Il faut les transformer en variables numériques binaires (0 ou 1).

StoreType_a, StoreType_b, StoreType_c, StoreType_d.

Assortment_a, Assortment_b, Assortment_c.

StateHoliday_a, StateHoliday_b, StateHoliday_c. (Garder 0 comme catégorie de référence ou la convertir aussi).

Combiner StateHoliday et SchoolHoliday : IsAnyHoliday (si l'une ou l'autre est un jour férié).

Outils Alteryx :

One-Hot Encode (ou Dummy tool) : Alteryx a des outils spécifiques pour l'encodage des variables catégorielles. C'est le plus simple.
Formula pour créer IsAnyHoliday.

Catégorie 5 : Caractéristiques Lagged (Décalées) et Rolling (Fenêtres Glissantes) - PLUS AVANCÉ

Ces variables sont très puissantes pour les séries temporelles, mais plus complexes à implémenter.

Variables à créer (nécessite de trier les données par Store et Date) :

Ventes de la veille : Lag([Sales], 1) -> Les ventes du jour précédent.
Moyenne des ventes sur 7 jours : MovingAvg([Sales], 7) -> Tendance hebdomadaire.
Ventes de la même semaine de l'année précédente : Lag([Sales], 365) -> Saisonnalité annuelle.
Différence de ventes d'une semaine à l'autre :[Sales] - Lag([Sales], 7) -> Pour la dérive.

Outils Alteryx :

Sort (Trier) : TRÈS IMPORTANT de trier par Store puis Date AVANT ces opérations.
Multi-Row Formula (Formule Multi-Lignes) : L'outil parfait pour calculer les lags, les différences, et les moyennes glissantes (en cochant Group By Store).
Running Total (Total Cumulé) : Peut aussi être utile pour certaines agrégations.