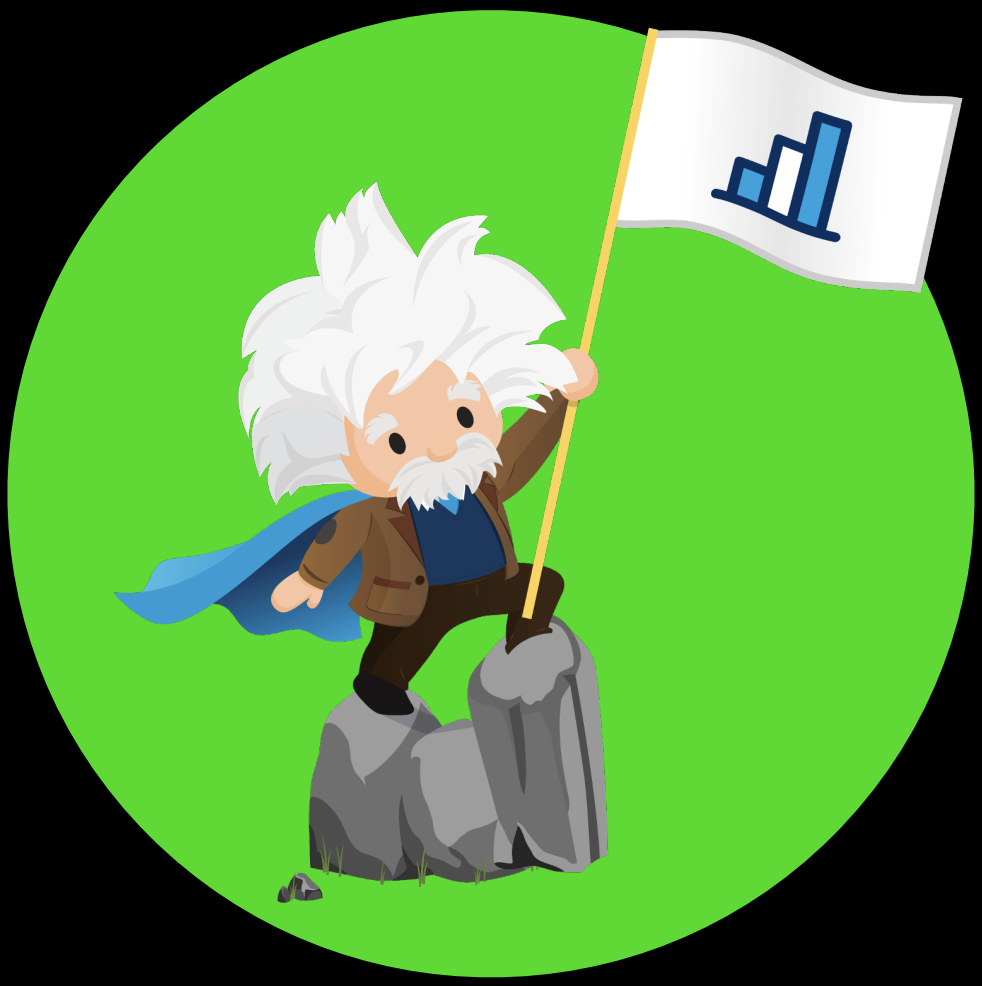


Short Report on Salesforce Einstein Discovery

September 2021

Introduction

The Salesforce (SF) "Tableau CRM (TCRM) Growth" license offers companies the ability to leverage cloud data to create dashboards that can help them extract business insights. However, the insight extraction process can be time-consuming and requires costly human input.
In many scenarios, companies judge the ROI on an analysis too small and turn their back to potentially valuable information. To reduce these investments, an increasing number of businesses rely on AI algorithms to automate the information extraction process.
This solution remains expensive because the pipeline to create an AI model includes many steps and requires solid data-science expertise.
Einstein Discovery (ED) proposes a solution: simplify the model creation pipeline to reduce the skillset and amount of work required to create a model. This report aims to evaluate ED's strengths and weaknesses and to assess to what extent this type of solution is useful.

Simple Model Creation Pipeline

Salesforce (SF) says that "Business Scientists" (aka "Citizen Data Scientist") are the primary target of ED. Although being marketing jargon, this quote sheds light on what ED strategy is: erase as much as possible the technicalities associated with model creation and focus on the business interpretability of models. The entire model creation pipeline can be achieved through drags-and-drops. Simply tell ED which field is your target and which columns are relevant to the problem, and it will automatically create a model.

Create a Dataset

Using Dataflows or SF Recipes create a dataset used to train a model.

The data sources can be ext. CSV files, ext. cloud data or SF data objects.

Datasources can be configured to be refreshed everyday.

Create a Story

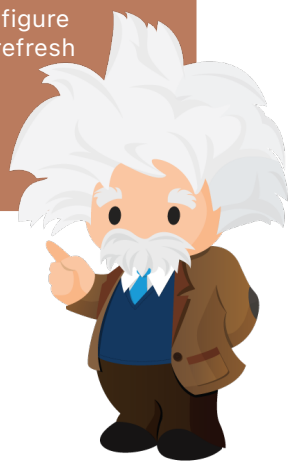
In SF, models are referenced as "Story". Simply link a story to a training dataset, and configure a few parameter, ED takes care of the training process.

Verify that the insights collected by ED are relevant and correct.

Deploy & Assess

Deploy the model. Predictions are written to a SF object.

Using the Model Manager monitor the daily performance of the model. Configure the model refresh rate.



Model Creation Pipeline

Highly Interpretable Models

To balance the lack of control one has over the model design process, ED development teams spent a lot of time designing an interface allowing the business mindset to understand how the model perceives the problem. In that respect, models are not validated through scores but by checking that the model picked up the correct trends in the data.
Let's take an example to explain the aforementioned point. Imagine a user creates a model to predict the hotel reservation cancelation likelihood. To evaluate the model's performance, the user can check that the model correctly picked up that clients who made a deposit rarely cancel. Scores such as AUC, R2 ... are deemed less relevant.

Following the same transparency effort, ED informs the end-user (who consumes the prediction) on the reasons behind the model's prediction. This information helps build trust between the end-users and the model, as humans inherently trust what they understand.

The Downside of Having Highly Interpretable Models

The pursuit of high interpretability allows (1) non-data scientist users to create models and also (2) helps to build trust between end consumers and the model. However, this quality comes at the expense of the models' complexity.
ED models are linear, trained under the assumption that the optimum can be found in sub-linear space (see the section about technicalities for more information). For that reason, the range of applications in which ED can be applied is reduced. ED models work well to solve simple problems, where a significant margin of error is acceptable. However, for complex, tight problems, ED won't work.

Furthermore, the model-building process simplification means that there is very little room for data-scientist to get involved. From a technical standpoint, the most advanced decision one is allowed to make during the model design process is the choice of the model (GLM, XGBoost, or Random Forests), and this limitation can be frustrating. For instance, if one detects that the model overfits the training data the model's hyper-parameters can't be changed.

To address the simplicity of ED models, the potential downfall of the model should be investigated (and not only at the feature level) and reported in the ED model's card. This verification step can take the form of a comparison with a custom-made model build by a data scientist. This comparison can shed light on the potential weaknesses of ED's model.
In this situation, ED is a self-driving car, and the data-scientist is the driver who controls that the car properly steers.

The Need of a Data Scientist

As with all AI technologies, ED models can return accurate predictions on data that exhibit a clear pattern. It is paramount that the development team checks the existence of such trends in the data. This verification process can either be achieved by talking to the business — for instance, asking a hotel manager if he can predict the cancelation likelihood of a client based on the client's record — or by calling a data scientist that will build a custom model to verify the existence of such patterns.

Even though one could think that ED eclipses the need for a data scientist, its presence on the model development team is required, to avoid having pseudo-random models.

Imagine a pharmaceutical company that wishes to develop a model that predicts the success likelihood of a drug on patients. They can turn to doctors and ask them based on their experience if they know when the drug works. But the issue is that the doctors have not seen enough patients and thus haven't enough experience to return a prediction. The pharmaceutical company can aggregate the patients' data collected by 100 of doctors and ask a data scientist to crunch the numbers and see if a model catches any pattern in the patient's data. If the data-scientist detects a pattern, a model can be created in ED.

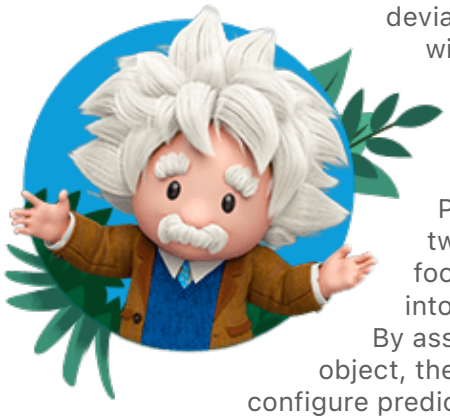
The Technical Aspects of Einstein Discovery

ED uses four kinds of algorithms implemented by the H2O.ai open-source library: GLM (either a piecewise linear model with ridge regression or a piecewise logistic model with ridge regression), XGBoost, Gradient Boosting, or Random Forests. It is worth noting that linear models would be far too simple, hence, a piecewise algorithm fits a linear model for each decile of a continuous variable.
To ensure proper training, models are trained using 4-fold cross-validation.

An essential part of a model design is feature engineering, and although this step is delegated to ED, the user can embed some feature transformation in the design of a Salesforce dataflow (created before the model's training).

ED approaches datasets with the notion that most business data is either implicitly or explicitly categorical, even when the data elements superficially appear to be numbers, dates, or text. Datapoints are grouped using the bucketing algorithm Kernel Density Estimation.

ED can deal with the following problems: duplicates detection, outlier detection (no clustering algorithms, ED detects values greater than five times the standard deviation), strong predictors (fields with an R2 value greater than 0.3), identical values, dominant values, disparate impact.



Predictions are consumed using two channels. The first one focuses on integrating the model into everyday business operations. By associating the model to an SF object, the system administrator can configure predictions to appear on the record pages of the object.
The second way of consuming a prediction is using TCRM. In this case, predictions can be consumed in the form of dashboards or lenses.

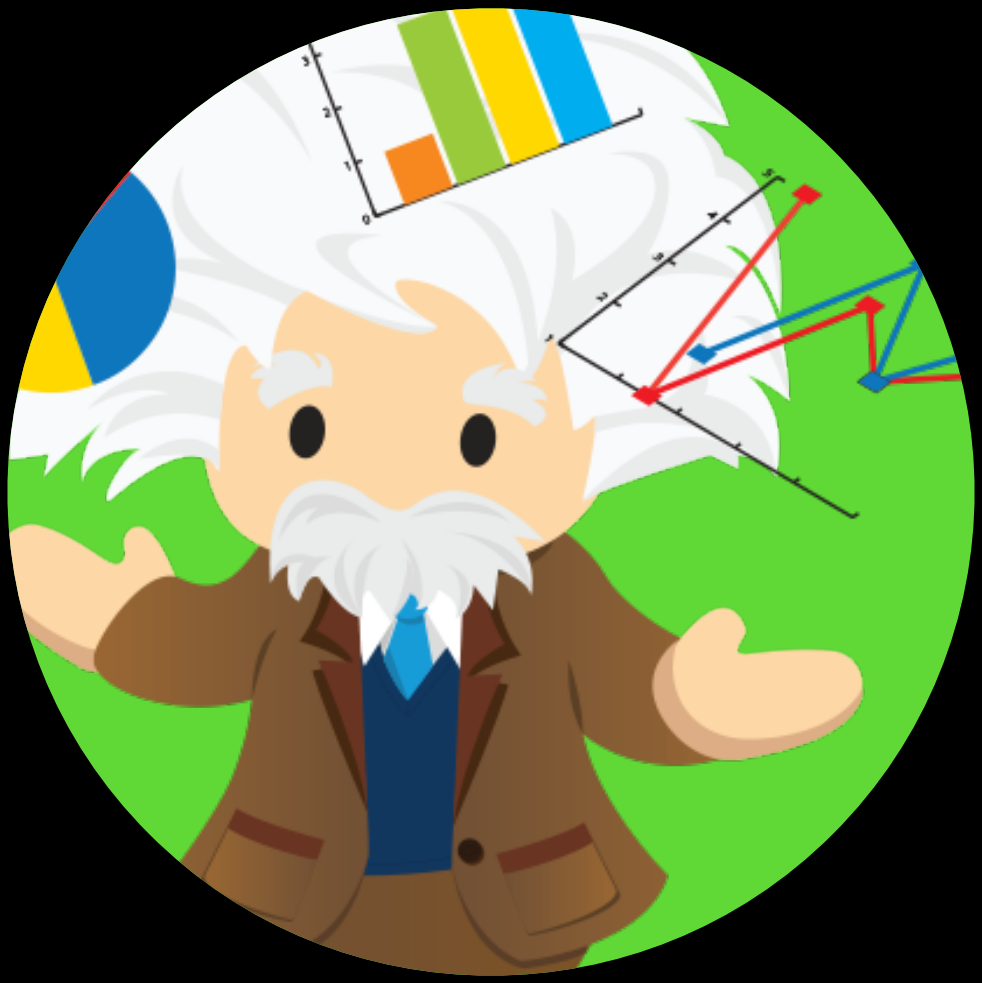
Personal Experiments

Experiments to verify ED performances were performed. All experiments code can be found in the Git repository mentioned in the reference section of this report.

Name	Experiment Description	Score	Comments
Real Estate Prices	Given listings and their properties predict the selling price. Interesting case because the number of available fields is large. To get a comprehensive pricing model, the model should select fields that cover different properties of the listing, and avoid fields that describe the same properties (eg. having two or more fields on the Garage). In this experiment, the model was not deployed.	R2: 0.84 RMSE: 32183 R2: 0.91 RMSE: 22181	The custom model best performance was achieved using a TabNet based on +15 features. ED was able to support only 12 features, out of which, 3 where dedicated to the garage. Feature selection using WiSe methodology.
Lead Prediction	Given a list of fields about leads, predict which leads is most likely to be closed. Relatevely small dataset. This is a simple toy example to test how accurate ED is compated to a custom made model.	AUC: 0.96 AUC: 0.98	This toy example simply proves the overall good performance of ED in simple settings.
Hotel Bookings	Given the reservation list of a hotel chain predict which client is most likely to cancel his reservation. Interesting case because the target classes are unbalanced. Again the number of features is quite large. In this experiment, the entire ED pipeline from getting raw data, to configuring dataflows and recipes, to train and deploy model was tested. A record page was configure to display the prediction, and a small dashboard was created using the prediction made.	AUC: 0.87 AUC: 0.92	Both models are able to achieved good results. However, seems to place all of its prediction on whether or not the the client made a deposit, while TabNet (custom model) balances more the feature importance, which yields to a more comprehensive model.

Conclusion

ED eclipses the need for lengthy AI projects and puts augmented intelligence at the finger-tip of each companies' workforce. This solution is especially relevant in today's fast-moving business world, where reactivity is essential to drive success.
However, one should not be blindsided by the overall simplicity offered by ED's model creation pipeline, inspecting and monitoring created models remains essential to ensure the overall project's success.
Furthermore, ED has fundamental technical limitations and cannot be used to solve all business problems. Only simple supervised learning problems can be solved using ED.



Sources:

Understanding the Differentiating Capabilities and Unique Features of Salesforce Einstein (2021)

Discovery within the Machine Learning Space, Darvish Lee Shadravan

How to Build An AI Prediction, A Step by Step Guide to Success with ED, Salesforce (2021)

Tableau CRM Reference Guide, Salesforce (2021)

Trailblazer, Salesforce (2021)

TabNet: Attentive Interpretable Tabular Learning, Sercan O. Arik, Tomas Pfister (2020)

All experiments can be found on the following GitHub repository:
https://github.com/PaulMansat/Salesforce-ED.git