“Custom Regression Model (Salary Prediction)” Module Transcript

Chapter 1

Intro, Topics Covered, & Learning Outcomes

Hey. My name is Hayley, and I'm on the One AI team here at One Model. In previous modules, we covered the fundamentals of machine learning and best practices for building models. Now we'll apply that knowledge to create a salary prediction regression model using the custom One AI guided framework. This will demonstrate how easy it is to build models outside of our predefined recipes list when you have the right data in One Model.

Throughout this module, we will cover the custom advanced modeling in One AI, an overview of a salary prediction regression model, some key considerations before beginning the model-building process, and we'll close with step-by-step instructions for building a Custom / Advanced model in One AI using salary prediction as our example.

After completing this module, you will recognize scenarios and business needs that make a salary prediction regression model valuable, such as compensation planning; understand and address important considerations before beginning the model building process, ensuring effective and accurate predictions; and you will be able to confidently create Custom / Advanced regression models in One AI using the available guided framework.

Chapter 2

Custom / Advanced Modeling in One AI

Section 2 - Custom / Advanced Modeling in One AI

Since One AI currently only offers one predefined regression recipe, the group attrition regression, most regression models are built using the guided custom / advanced framework that looks nearly identical to a recipe.

Similar to the predefined recipes, the custom / advanced framework in One AI simplifies most settings by automatically selecting defaults, allowing you to focus primarily on data framing. However, a key difference is that the tooltips in this framework do not always recommend specific metrics or columns because we don't know exactly what you're

predicting. They do explain what each field is used for and provide general recommendations to help you make informed selections, though.

Another important difference is that in the first step, you must select the type of prediction problem - classification or regression - which would be automatically determined on the backend if you were using a predefined recipe. You will select regression when you want the model to predict continuous numerical values.

Creating a model with the custom advanced framework allows you to predict on whatever you want, provided you have the appropriate data to train the model already validated in One Model and a defined prediction metric.

For custom regression models, it's important to ensure you're predicting continuous values only, not constant values or fixed values. This requirement is why time-to modeling is complex and typically not possible without deliberate manipulation of the data and creating the model with a data destination instead of our guided recipe framework.

Some other custom / advanced regression models that we have seen successfully created in One AI include employee attrition regressions, which help you predict the likelihood of an employee turning over; employee performance regressions, which help you analyze performance reviews to predict future numerical performance scores for employees; and employee engagement regressions, which help you analyze survey responses, sentiment analysis, and feedback to predict employees' numerical engagement levels and satisfaction. Like I said, any already defined continuous metric could potentially be used to build a custom regression model.

Chapter 3

Salary Prediction Regression Model Overview

Section 3 - Salary Prediction Regression Model Overview

While I'm just using a salary prediction regression model as an example to demonstrate custom modeling outside the predefined One AI recipe list, I did want to briefly explain this model because it's one of the most commonly created custom regression models. Feel free to skip this section if it's not relevant to you at this moment.

A salary prediction regression model estimates an employee's future salary at a specified time, such as within the next year. By analyzing attributes like experience,

education, job role, location, performance, and any other relevant variables, the model helps you understand how these factors contribute to salary projections.

The model outputs a continuous salary estimate for individual employees, where each employee represents an instance in the model, typically identified by their unique person ids from the employee table. The model creator selects the population, prediction period, and input features, which include core attributes as well as individual and team generative attributes.

Additionally, the model results highlight the top factors influencing salary predictions, providing insights into key drivers of compensation within your organization. This enables more informed decisions about salary structures and equity, supporting competitive and fair compensation practices.

Some key use cases for a salary prediction regression models are compensation planning and budget forecasting. So this model can be used to ensure that salary structures are aligned with organizational goals and industry industry standards by predicting future salary needs and identifying potential discrepancies. In a similar vein, financial planners can use the model to forecast future salary expenditures, helping to plan budgets and allocate resources more effectively across the organization.

Additionally, this model can be used for performance-based adjustment analysis. So leaders can use the model to predict how factors like performance ratings or skill development might influence future salary increases, allowing for more targeted and justified compensation adjustments.

Additionally, pay equity analysis can be performed using the model results, and this helps identify pay inequities and create strategies to address these inequities by comparing predicted salaries with actual compensation, ensuring fair and competitive pay practices.

Recruitment and hiring could use the insights from this model to inform salary offers for new hires by predicting competitive salaries based on similar roles within the organization and core attributes about the new hire, ensuring that offers are attractive enough to land top talent, but are also aligned with internal compensation structures.

And finally, retention strategies. If the salary prediction model estimates that an employee's salary should be significantly higher than what they are currently earning, it could be a signal that the employee might feel under compensated. This discrepancy can increase the risk of turnover as the employee may seek opportunities elsewhere that offer compensation closer to their predicted value. By identifying these gaps,

human resources can take proactive measures to retain valuable employees before they consider leaving the organization.

Chapter 4

Considerations Before Model Building

Section 4 - Key Considerations Before Beginning the Model-Building Process

Before creating any custom / advanced regression model in One AI, there are some important factors that should be carefully considered to ensure its effectiveness.

First, and I know I've mentioned this before, but it's crucial, make sure you're predicting continuous values, not fixed or constant values. For example, predicting how many days are in a week wouldn't make sense because it's always seven, so there's no variability to predict. Regression models are meant to predict outcomes that can change based on different input variables, so using them to predict a constant or fixed value is not appropriate.

A more relevant example of this in One Model might be when predicting hours worked for an hourly employee. If you're only capturing the current days count, which would be eight on any given day, rather than the total hours over a span of time, you're dealing with a fixed value, not a continuous one. This would lead to inaccurate predictions since the model can't learn from fixed data.

Next, as always, you should clearly define the objectives of the model. Determine if the model aims to confirm hypotheses, conduct exploratory analysis to learn more, or replicate a previously built model outside of One AI. This will help guide your approach.

Also, you should consider potential biases in the training data. For example, if the historical salary data used to train a salary prediction model reflects existing biases, such as disparities based on gender, race, or age, the model is likely to learn and replicate these biases in its predictions. This means that if certain groups have historically been paid less for similar roles, the model might continue to predict lower salaries for these groups, perpetuating those inequities. If this is a concern, you may want to consider bias detection and removal.

And finally, always ensure you have accessible, reliable, and maybe most importantly, relevant data to train the model. Verify that the necessary data sources are integrated and validated in One Model before beginning.

Next, we'll move over to One Model for a demo on building a custom / advanced salary prediction regression model in One AI.

Chapter 5

Building a Custom / Advanced Regression Model in One AI

Section 5 - Building a Custom / Advanced Regression Model in One AI

Before you can start building models in One AI, you will need the CanAccessOneAIMenu application access role. If you can see the One AI tab in the main ribbon menu, you should be all set.

Click One AI in the main ribbon menu, and this will bring you to the page where you can create or edit models. If your organization has previously created any models, they will be listed here. To add a new model, click "Add Machine Learning Model" in the upper right-hand corner.

First, give your model a unique name in the display name field. Then click ‘One AI Recipe’ under 'Using Data From' and then "Configure One AI Recipe." This brings you to the One AI Query Builder. Under "What are you interested in predicting?", select “Custom / Advanced Model” from the list of available recipes. Once selected, the problem statement is populated with the information that we need to provide in order to build the model.

In step one, we define the type of prediction problem we are working with. The options for problem type are classification or regression. In this case, you will select 'Regression'.

In step two, we select the column or metric that defines the continuous value that we want the model to predict. First, I'm picking 'Metric' from this dropdown because columns should only be used if they were created by your data engineer specifically for One AI purposes.

Then I'm selecting the 'EOP Salary - Average' metric, and this average metric works well because when viewed on a per-employee basis, it effectively reflects each employee's actual salary at the end of that day, and it's a common metric among One Model sites.

If you do not have a similar metric on your One Model site, you can create one by using the annual salary column from the employee or compensation table. Something like this works well, but chat with your CS team if you need help.

Additionally, if you wanted to filter the target metric, you could do so here.

In the next step, select the population that One AI will make predictions on. In this case, this will be which employees we want to predict salaries for. You can use a general headcount end of period metric and filter down or a specific headcount metric like US headcount or salaried headcount for your population metric in order to get to your group of interest.

Then you must choose a unique identifier for each instance in the model. Ignore the 'Select via Dimension Level' button when not working with group models. When using a headcount population metric, typically person ID from the employee table is best, but employee ID, worker ID, or any other unique ID will work. The unique ID must come from the same table as the population metric. One AI will verify the ID's uniqueness for the model.

Next, the population date anchors the query with the predict frame and the train/test dataset offset from it. Common choices are 'Today', 'End of Last Month', or a static date. Avoid dates too far in the past since we already know the outcomes. I'm selecting 'Today', which means that this model will use history going back starting from today and will predict a year in the future also starting from today.

And then here you can add filters to refine your headcount population. Instead of running the model on the entire organization, segment it by factors like department, country, or job level since drivers tend to vary by group. For example, I included only engineering employees in this model's population by filtering the org unit dimension.

Next, select how far in the future you want to predict from the population date. This defaults to one year, but can be adjusted to fit your custom modeling needs.

Then choose how much historical data you want to use for training the model. We have found that best performance results from enough training intervals to give us one to two years of historical data. So because I selected one year in the previous step, each interval equals one year. Therefore, I will enter one year so that the model has one year of historical data to learn from. In most cases, a good strategy is starting with one year and increasing to two or three years to see if it improves performance.

Next, select core attributes, which are the input variables that form the dataset that the model learns from and uses to make predictions. So what attributes about an employee

that might impact their predicted salary for our case? We want to aim for balance, enough attributes for robust predictions, but not so many that the results become difficult to interpret.

If you don't intend to manually select each column into your model dataset, I recommend using a ‘Narrow’ or ‘Balanced’ scope to capture attributes that sit in the employee table and/or one join away from the employee table. Sometimes performance, survey, or compensation data sits in its own table, which is when the Balanced scope comes in handy.

Then go through all of the Included columns and exclude anything you don't want One AI to use using this 'X' button here. Once you've gone through all the included columns and excluded anything you don't want One AI to use, go through each of the excluded columns and bring in additional features of interest using this check mark button here.

Then select the generative attributes you want to use in your prediction. Use the checkbox to the left of each attribute to bring in that generative attribute as an input variable, or select all by checking the box at the top. You can also create, edit, and delete generative attributes in this step. For more information on how they work and how to build them, check out our module on generative attributes.

Next, click 'Generate Data Statistics' to verify that all of your previous selections are valid. This helps identify errors that cause the model to error while running. You will also receive valuable data exploration information about the trained test dataset, such as row count, standard deviations, and min and max values. We received the green light "Success" so we can proceed. If One AI finds a problem, it will display action needed with details about the errors that need to be corrected.

In the final step, you can download the Train / Test dataset, Predict dataset, or both that were generated by this recipe.

Once you're done with the final step, review each step in the recipe. Once you're happy with your selections, click the save icon in the upper right to return to the machine learning model screen. Here you can manually configure your model settings. Refer to our module series on advanced configuration and global settings if interested. Once complete, scroll to the bottom and click 'Create'. If you don't click create and navigate elsewhere in One Model, you will lose your work from the One AI Query Builder screen.

And just like that, you can now create Custom / Advanced regression models in One AI. The next step is to run your model, evaluate your results, refine the model, and then deploy and share its insights with stakeholders.

Chapter 6

Conclusion & Thanks

We've explored how to build custom regression models in One AI using salary prediction as a relevant example. You now have the skills to configure, validate, and refine custom models to suit various needs. By understanding when to use these models and addressing potential biases, you can enhance your predictive modeling capabilities. Happy modeling!