PLACEHOLDER PDF.

1. Only keep steps, no narrative
2. Add naming dataset
3. Add bucket options walkthrough

Use AutoML to Train a Linear Regression Model

The AutoML projects for this course will be implemented using Google's Vertex AI, the GUI-based AutoML and custom training framework the Authors are most familiar with. Note that the top three major cloud vendors (Google, Microsoft and AWS) all offer AutoML Tutorials. Google's guide can be found here, Microsoft's here, and Amazon's here. Many cloud vendors offer a trial period to explore their products without cost.

Given that Google offers a step-by-step tutorial on AutoML, some introductory steps are excluded.

Figure 4-15 shows a high-level overview of the AutoML no-code workflow for your business use case.

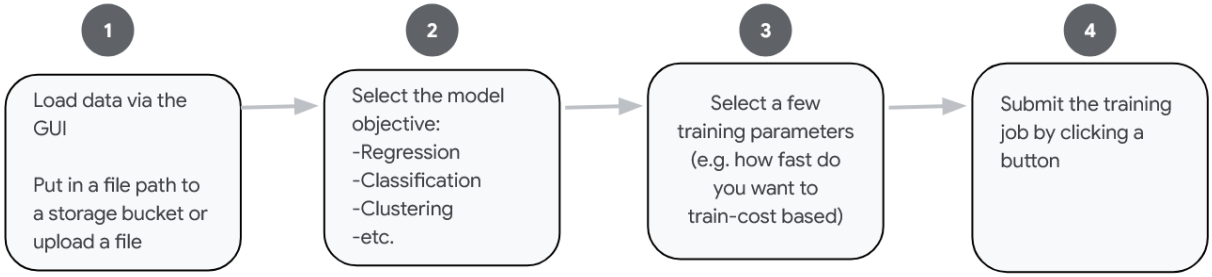


Figure 4-15. AutoML no-code workflow for your use case.

No-Code using Vertex AI

Figure X-X below shows the Vertex AI Dashboard. To create an AutoML machine learning model, turn on the Vertex AI API by clicking the *Enable All Recommended APIs* button. From the left-hand navigation menu, scroll down from *Dashboard* and select *Datasets*.

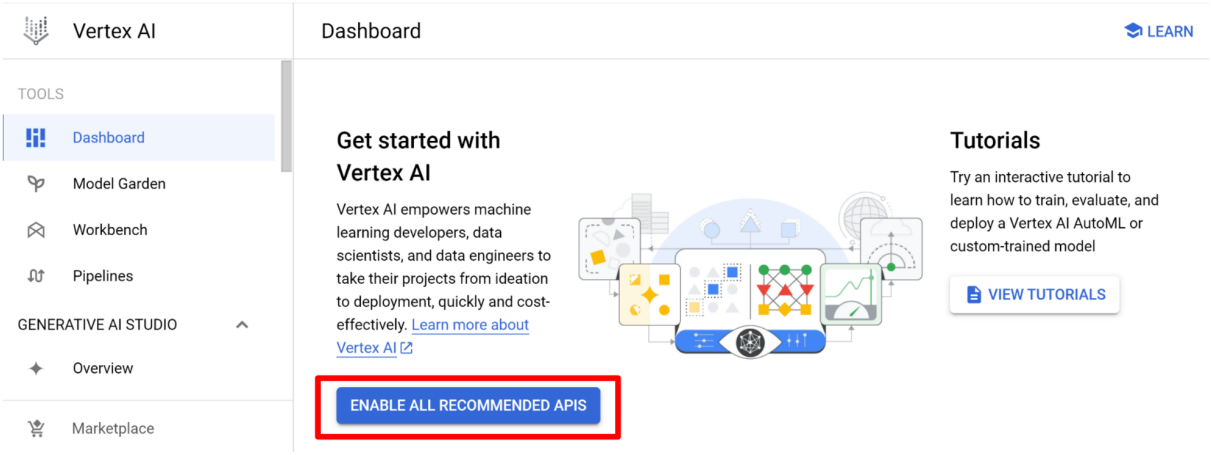


Figure 4-16. Vertex AI Dashboard showing the Enable All Recommended APIS button.

Create a managed dataset in Vertex AI

Vertex AI offers different AutoML models depending on data type and the objective you want to achieve with your model. When you create a dataset you pick an initial objective, but after a dataset is created you can use it to train models with different objectives. Keep the default region (us-central1), as shown in Figure 4-17.

Select the Create button at the top of the page then enter a name for the dataset. For example, you name the dataset *advertsing_automl*.

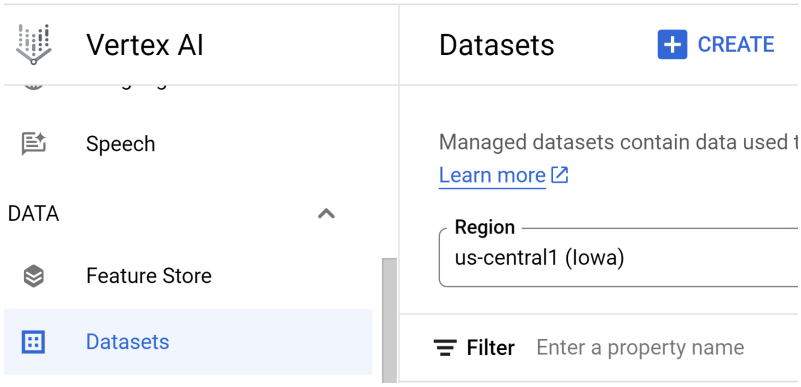


Figure 4-17. Vertex AI Dataset Navigation that allows you to create a dataset.

Select the Model Objective

Figure 4-18 shows Regression/Classification selected as the model objective under the *Tabular tab*. Given that you want to predict a target column's value (sales) this is the appropriate selection.

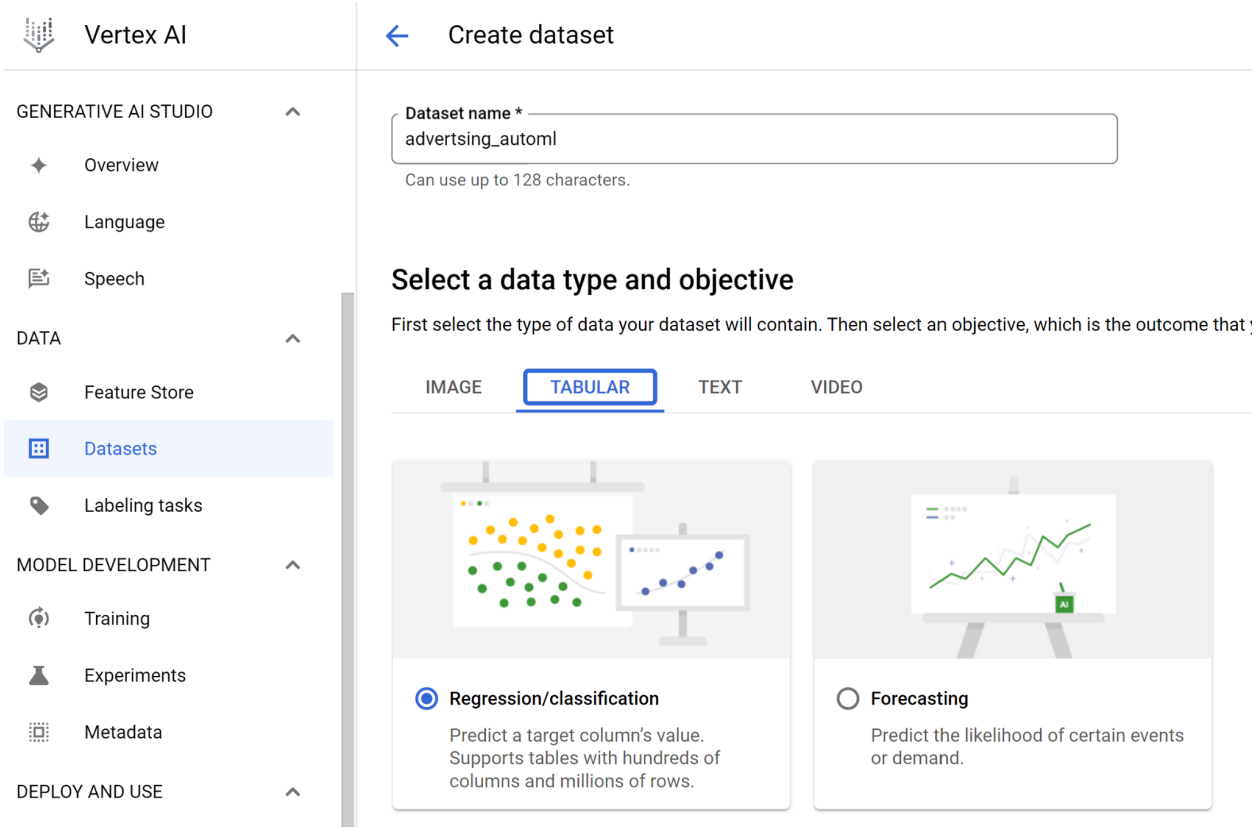


Figure 4-18. Regression/Classification selection for model objective.

You selected *Regression/Classification* as your objective. Let's cover some basic concepts to help you with future use cases. Regression is a supervised machine learning process. It is similar to classification, but rather than predicting a label for a classification, such as classifying SPAM from your email inbox, you try to predict a continuous value. Linear regression defines the relationship between a target variable (y) and a set of predictive features (x). If you need to predict a number, then use regression. In your use case, linear regression predicts a real value (sales) using some independent variables given in the dataset (digital, TV, radio, and newspaper).

Essentially, linear regression assumes a linear relationship with each feature. The "predicted" values are the data points on the line and the true values are in the scatter plot. The goal is to find the best fitting line so that when new data is input the model can predict where the new data point will be in relation to the line. The "evaluation" of how best that fit is includes an evaluation criteria - which is covered in the Evaluate Model section.

Figure 4-19 shows a “best-fit” line based on your dataset,where the model tries to “fit” the red line to your data points, which are the scatters in black.

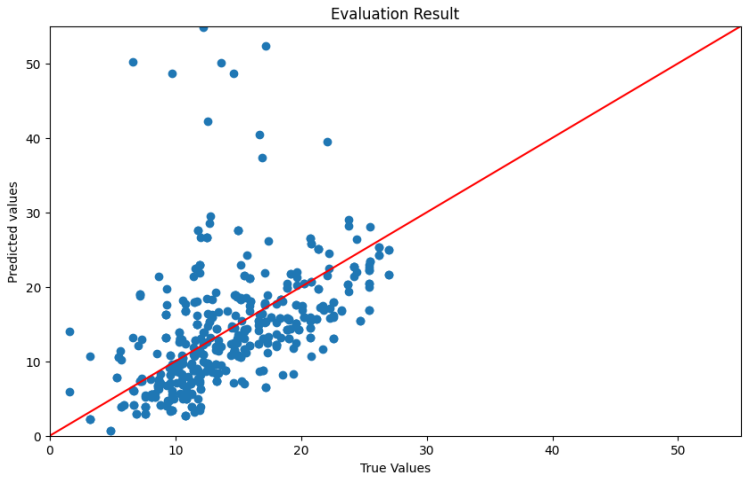


Figure 4-19. True and Predicted Values with a Best Fitted Line.

After making the *Regression/Classification* selection, scroll down and click the *Create* button. You are now ready to add data to your dataset. Vertex AI managed datasets are available for a variety of data types, including tabular, image, text, and video data.

Figure 4-20 shows the data source upload options - upload CSV from your computer, select CSV files from Cloud Storage or select a table or view from BigQuery (Google’s data warehouse).

To upload your advertising dataset, select *Upload CSV files from your computer*. Find the file on your local computer and load it**.**

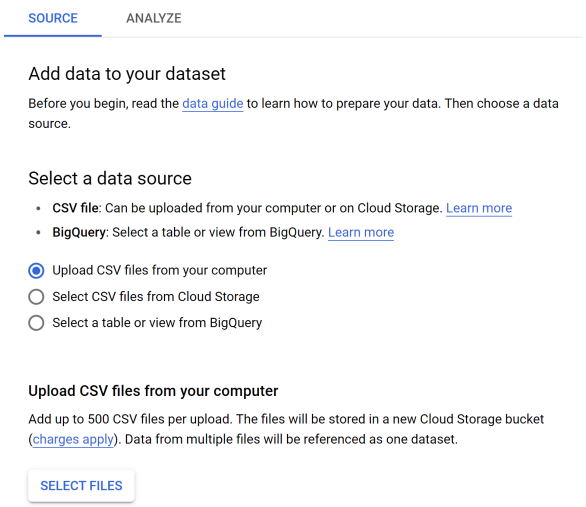


Figure 4-20. Data source upload options for your dataset file.

Scroll down the page and review the *Select a Cloud Storage Path* section, which requires that you store the file in a cloud storage bucket. Why do you need to store the file in a cloud storage bucket? There are many reasons: (1) When training a large-scale machine learning model, you may need to store terabytes or even petabytes of data; (2) Cloud storage buckets are scalable, reliable, and secure.

Figure 4-21 shows that the file Advertsing_automl.csv has been uploaded and a cloud storage bucket has been created to store the uploaded file. To see the entire step-by-step process of creating the storage bucket and the entire exercise, see the **PDF entitled *Chapter 4 AutoML Sales Prediction in the course repo here.***

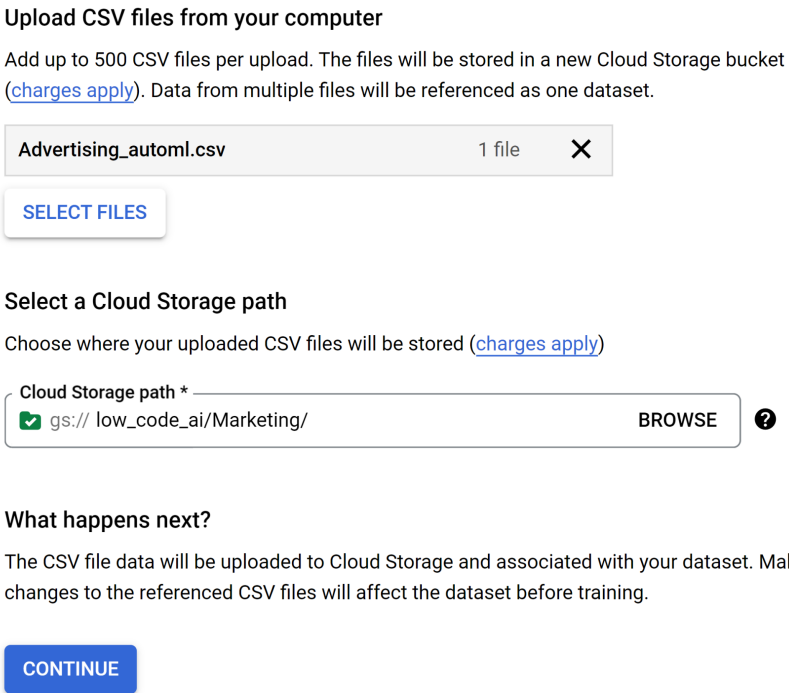


***Figure 4-X**. Data source options to load a CSV file and store it a cloud storage bucket.*

Some frameworks will generate statistics after the data loads. Other frameworks help minimize the need to manually clean data by automatically detecting and cleaning missing values, anomalous values, and duplicate rows and columns. Note that there are a few additional steps that you can employ, such as to review the data after it has loaded to check for missing values and view data statistics.

To generate statistics, click on *Generate Statistics* as shown in **Figure XX**. Note there are no missing values and the number of distinct values for each column is shown.

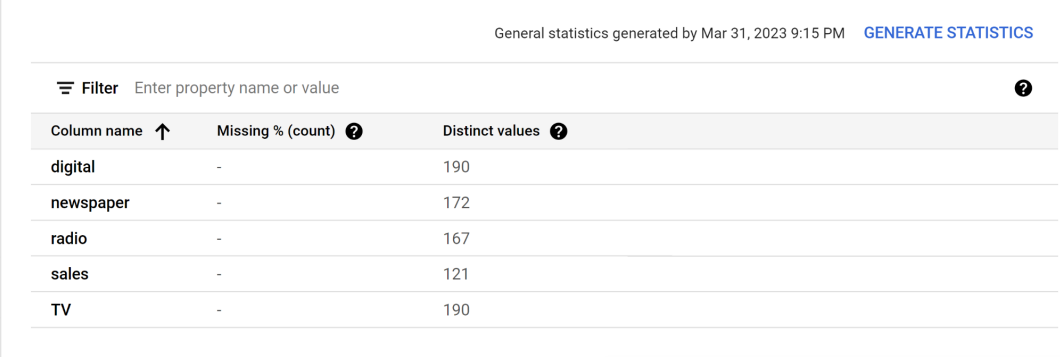


Figure XX. *The Output of the Generate Statistics Window.*

AutoML presents a data profile of each feature To analyze a feature, click the name of the feature. A page shows the feature distribution histograms for that feature.

Figure xxxxx shows the data profile for sales. Note that the mean is 14.014, which is the exact numeric value you received when you typed in the advertising_df.describe() code earlier in the chapter as you were exploring the dataset.

Column name: sales
Missing % (count): -
Distinct values: 121
Mean: 14.014
Standard Deviation: 5.2
Most common value (%): 9.7(4.862%)

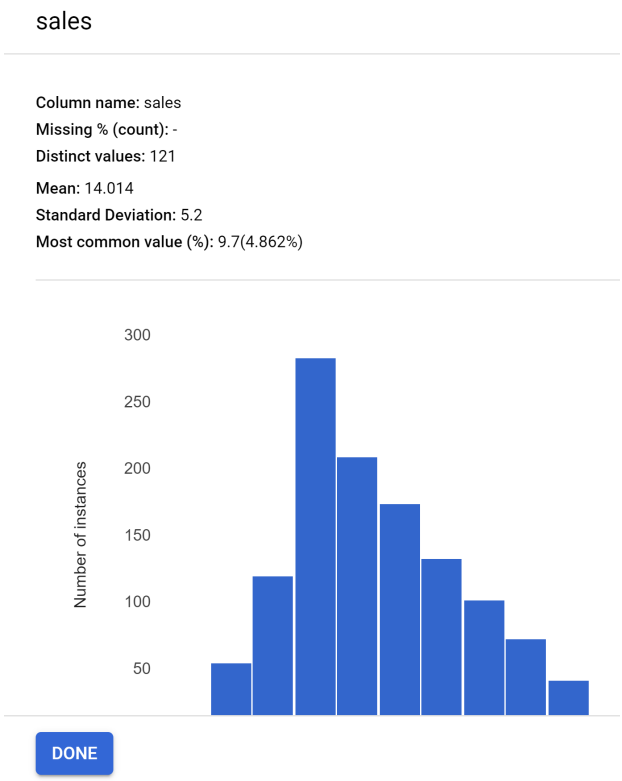


Figure XX. *Feature Profile for Sales from Vertex AI.*

Build the Training Model

Figure XX shows that the model is now ready to train. Select *Train New Model* under the *Training jobs and models section*. Select Other and not AutoML on Pipelines. AuotML on Pipelines, is a feature that allows you to specify the type of machine learning model you want to build and other parameters. It is beyond the scope of the book.

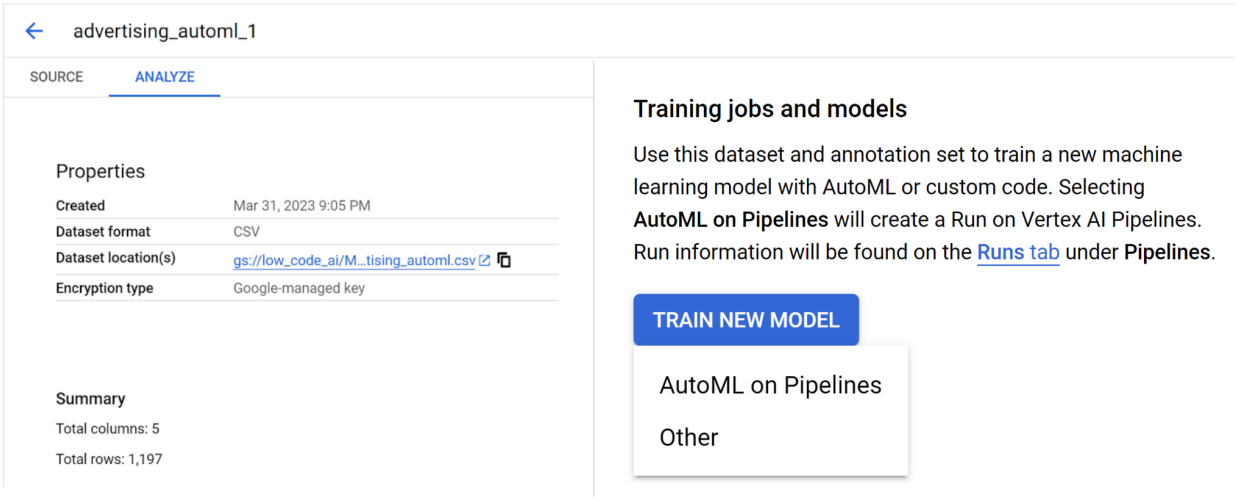


Figure XX. *Model is Now Ready to be Trained.*

The Train new model window appears. There are four steps.

1. Select the training method
2. Configure model details
3. Determine training options
4. Select compute and pricing

In Step 1, under the Objective, select the dropdown and choose Regression. Under Model Training Method, select AutoML (as shown in Figure 4-X). Click *Continue*.

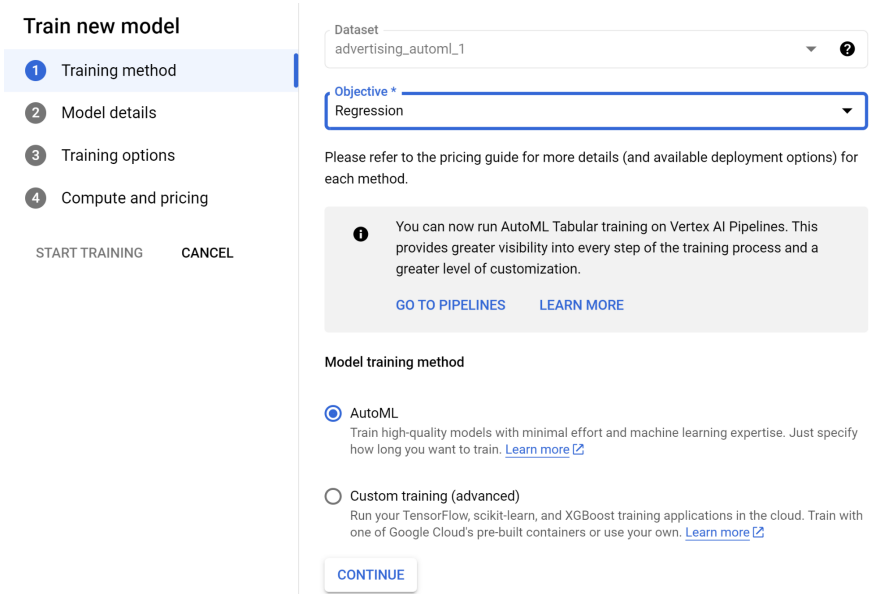


Figure XX. *Configure the training method in Step 1.*

In Step 2, under Model Details, name your model and give it a description. Under the Target column select the dropdown and select *sales*. (as shown in Figure 4-X). Click CONTINUE.

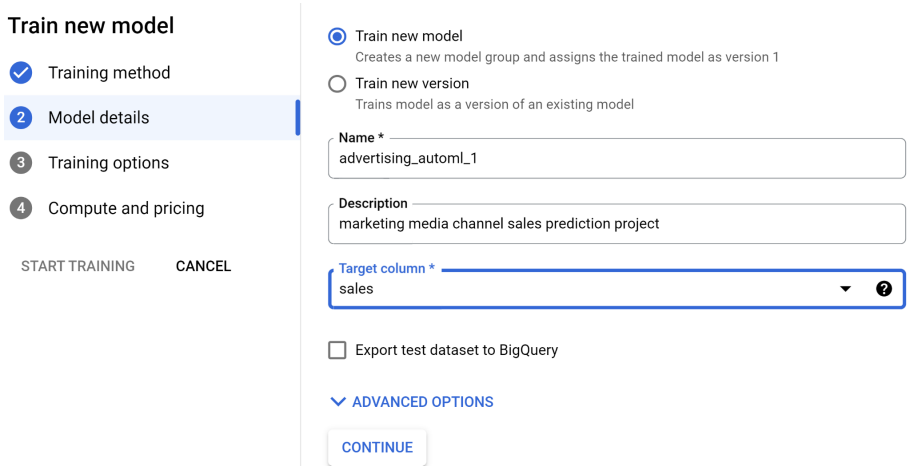


Figure XX. *Add model details in Step 2.*

In Step 3, review the training options. Note that any data transformations (or data processing) such as standardization is handled automatically (as shown in Figure 4-X). Click CONTINUE.

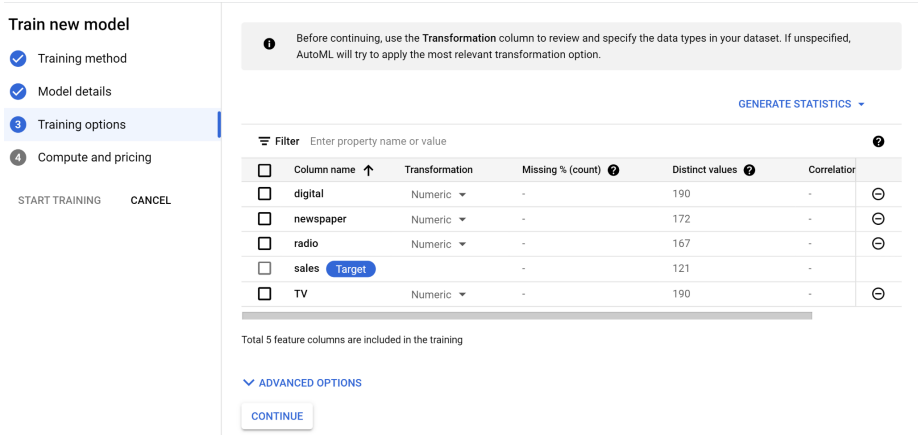


Figure XX. *Add training options in Step 3.*

In Step 4, you see [Compute and Pricing.](#)(as shown in Figure 4-X). The time required to train your model depends on the size and complexity of your training data. A node hour is one hour's usage of one node (think virtual machine) in the cloud, spread across all nodes. Enter three hours - this is just an estimate. You pay only for compute hours used; if training fails for any reason other than a user-initiated cancellation, you are not billed for the time. You are charged for training time if you cancel the operation.

Also under Compute and Pricing is Early Stopping. When you enable this option, this means that training will end when AutoML determines that no more model improvements can be made. If you disable Early Stopping, AutoML will train the model until the budget hours are exhausted.

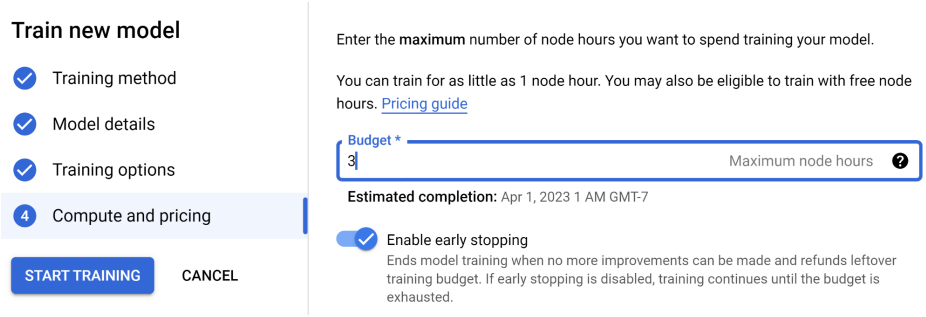


Figure XX. *Train New Model Step 4.*

Train the Model

The last option under Train the Model is compute and pricing. You pay for each model deployed to an endpoint, even if no prediction is made. Models that are not deployed or have failed to deploy are not charged. You pay only for compute hours used; if training fails for any reason other than a user-initiated cancellation, you are not billed for the time.

Once all the parameters are entered, you start the training job. Click START TRAINING.

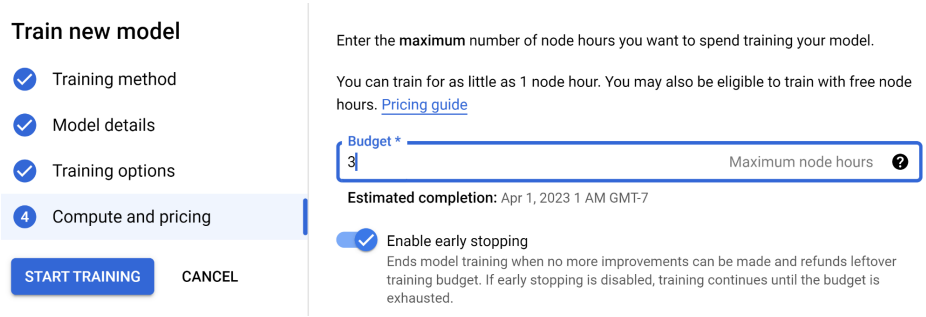


Figure XX. *Submit the training job for training.*

As previously mentioned, training can take up to several hours, depending on the size of your data and type of model objective you choose. Image and video data types may take much longer to process than a structured data type such as a *.csv* file. The number of training samples also impacts training time.

***Begin Note
AutoML is time intensive. AutoML algorithms need to train a variety of models, and this training process can be computationally expensive. This is because AutoML algorithms typically try a large number of different models and hyperparameters, and each model needs to be trained on the entire dataset. AutoML algorithms then need to select the best model from the set of trained models, and this selection process can also be time-consuming. This is because AutoML algorithms typically need to evaluate the performance of each model on a holdout dataset, and this evaluation process can be computationally expensive.
***End Note

After model training, the model is registered in the model registry.

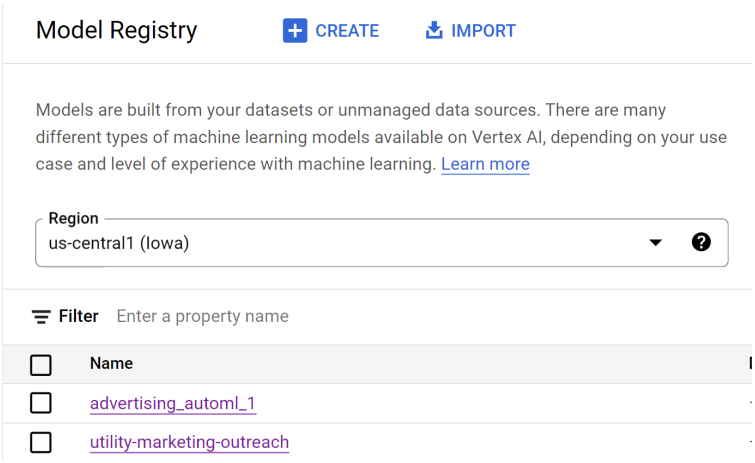


Figure XX. *Advertising_automl model showing model registry.*

Evaluate Model Performance

Model training results are presented after training. Figure 4-X shows training results.

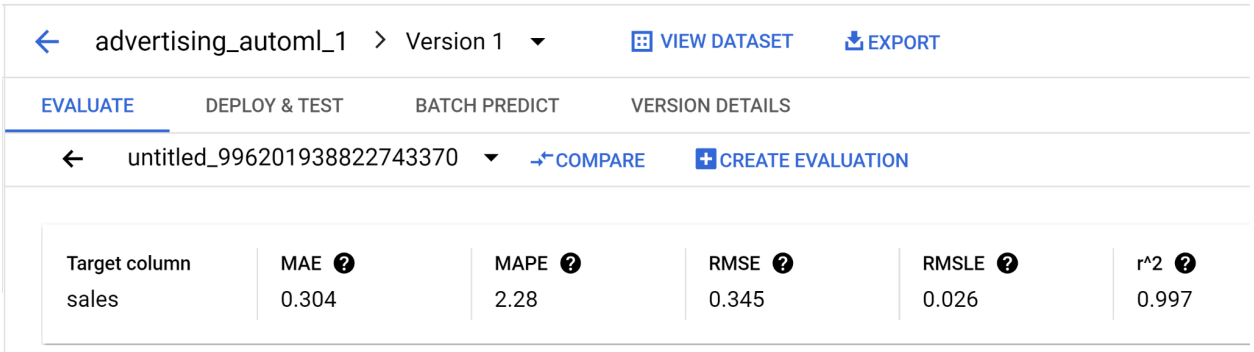


Figure 4-X. *Model training results.*

AutoML will output four evaluation criteria metrics as results.

There are a few factors that a practitioner should consider when weighing the importance of different linear regression evaluation metrics:

- *The purpose of the model*: The purpose of the model will determine which evaluation metrics are most important. For example, if the model is being used to make predictions, then the practitioner may want to focus on metrics such as mean squared error (MSE) or root mean squared error (RMSE). However, if the model is being used to understand the relationship between variables, then the practitioner may want to focus on metrics such as R-squared or adjusted R-squared.
- *The characteristics of the data*: The characteristics of the data will also affect the importance of different evaluation metrics. For example, if the data is noisy, then the practitioner may want to focus on metrics that are robust to noise, such as MAE. However, if the data is not noisy, then the practitioner may be able to focus on metrics that are more sensitive to changes in the model, such as MSE.
- *The practitioner's preferences*: Ultimately, the practitioner's preferences will also play a role in determining the importance of different evaluation metrics. Some practitioners may prefer metrics that are easy to understand, while others may prefer metrics that are more accurate. There is no right or wrong answer, and the practitioner should choose the metrics that are most important to them.

***Begin Note

Here are common linear regression evaluation metrics:

- *R-squared*: R-squared is a measure of how well the model fits the data. It is the square of the Pearson correlation coefficient between the observed and predicted values. It is calculated by dividing the sum of squared residuals by the total sum of squares. A higher R-squared value indicates a better fit. R squared ranges from 0 to 1, where a higher value indicates a higher-quality model. Your R^2 should be around 0.997.
- *Adjusted R-squared*: Adjusted R-squared is a modified version of R-squared that takes into account the number of independent variables in the model. It is calculated by dividing the sum of squared residuals by the total sum of squares minus the degrees of freedom. A higher adjusted R-squared value indicates a better fit, but it is less sensitive to the number of independent variables than R-squared.
- *Mean squared error (MSE)*: MSE is a measure of the average squared error between the predicted values and the actual values. A lower MSE value indicates a better fit.
- *Root mean squared error (RMSE)*: RMSE is the square root of MSE. It is a more interpretable version of MSE. A lower RMSE value indicates a better fit and a higher-quality model, where 0 means the model made no errors. Interpreting RMSE depends on the range of values in the series. Your RMSE should be around 0.345
- *Root mean squared log error (RMSLE)*: Interpreting RMSLE depends on the range of values in the series. RMSLE is less responsive to outliers than RMSE, and it tends to

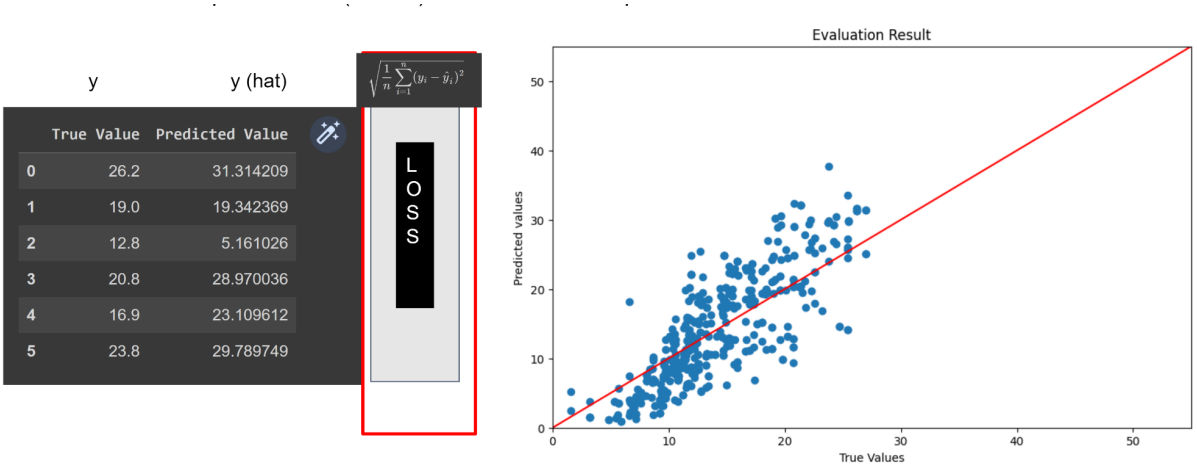
penalize underestimations slightly more than overestimations. Your RMSLE should be around 0.026.

- *Mean absolute error (MAE)*: MAE is a measure of the average absolute error between the predicted values and the actual values. A lower MAE value indicates a better fit.
- *Mean Absolute Percentage Error (MAPE)*: The mean absolute percentage error ranges from 0% to 100%, where a lower value indicates a higher quality model. MAPE is the average of absolute percentage errors. Your MAPE should be around 2.28.

*** End Note

You will evaluate the performance of the model using RMSE. The lower the RMSE, the better a given model is able to “fit” a dataset. So, what does that mean? RMSE takes the square root of the difference between the actual and predicted values squared. This is really the square root of the MSE. This is what is referred to as loss. The true values and predicted values are plotted in the scatter plot and a red line “fits” itself through the data.

Figure 4-X below shows the visualization of all true and predicted values plotted using the residuals (difference between the true and predicted values) shown.



	True Value	Predicted Value
0	26.2	31.314209
1	19.0	19.342369
2	12.8	5.161026
3	20.8	28.970036
4	16.9	23.109612
5	23.8	29.789749

Figure 4-X. *Loss formula for Root Mean Squared Error of True and Predicted Values.*

Model Feature Importance (Attribution)

Model feature importance tells you how much each feature impacted model training. **Figure 4-x** shows attribution values expressed as a percentage; the higher the percentage, the more strongly the correlation—that is, the more strongly that feature impacted model training. Feature attribution allows you to see which features contributed most strongly to the resulting prediction:

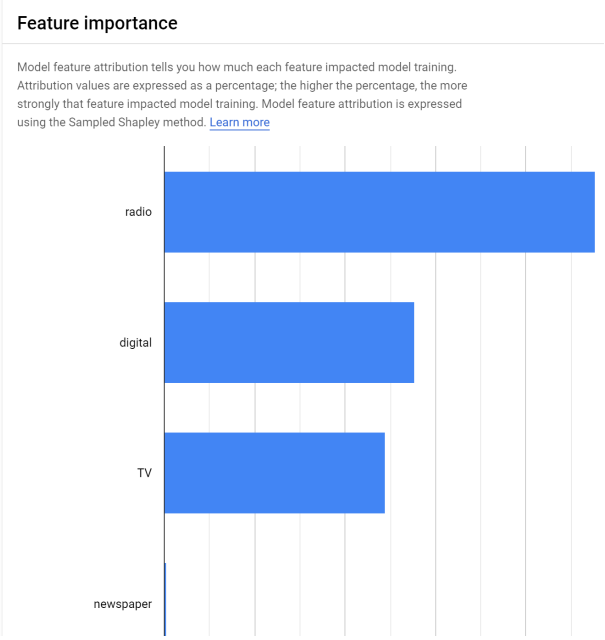


Figure 4-X. *Advertising Dataset Feature Importance Results*

If you were to hover over the newspaper feature shown in **Figure XX**, you would see that its contribution to model training is.2%. This supports what you discovered during the Exploratory Data Analysis phase you completed earlier - the correlation between sales and newspaper advertising is the weakest. These results mean that advertising on *Radio*, *Digital*, and *TV* contribute the most in *Sales*, and *Newspaper* advertisements have little effect on *Sales*.

Get Predictions From Your Model

In order to deploy your model, you will need to test it. You can deploy to your environment to test your model without building an application you would need to deploy to the cloud. After you train a machine learning model, you need to deploy the model so that others can use it to do inferencing. In Azure Machine Learning, you can use endpoints and deployments to do so.

There are four steps but for this Chapter, you'll only need the first two. For this exercise, it is not necessary to configure model monitoring or model objectives. Model monitoring adds an additional charge for logging while model objectives require you to choose from a variety of model objectives, depending on the type of model you are training and the application you are using it for.

1. Define your endpoint.
2. Configure Model Settings
3. Configure Model Monitoring
4. Configure Model Objectives

***Begin Note

What are endpoints and deployments? In machine learning, an endpoint is a service that exposes a model for online prediction. A deployment is the process of making a model available as an endpoint. An endpoint is an HTTPS path that provides an interface for clients to send requests (input data) and receive the inferencing (scoring) output of a trained model. Endpoints are typically used to make predictions in real time. For example, you could use an endpoint to predict the likelihood of a customer clicking on an ad or the risk of a loan defaulting.

Deployments are typically used to make a model available to a larger audience. For example, you could deploy a model to a production environment so that it can be used by your customers or employees.
***End Note

To deploy your model, go to Model Registry and select Deploy and Test and select your model.

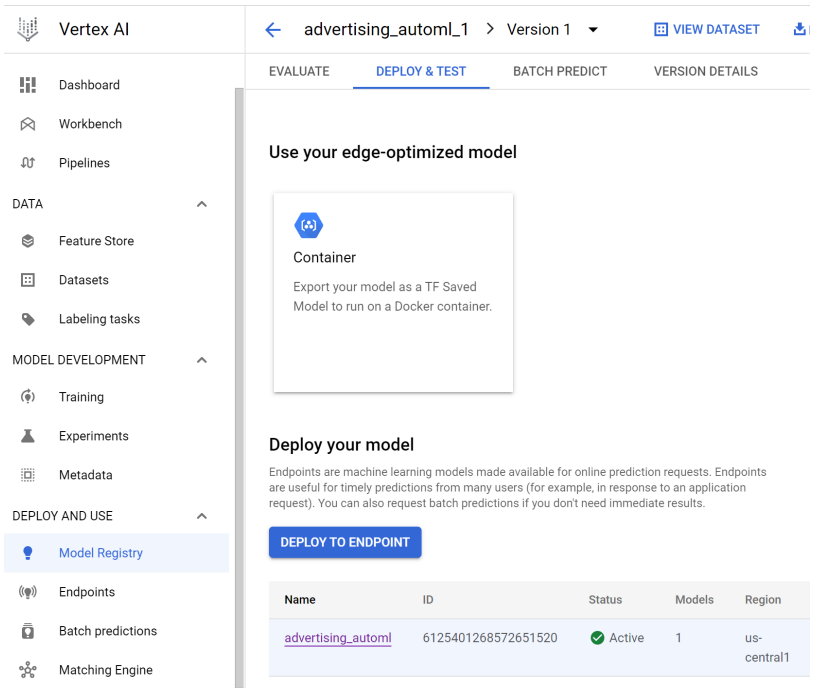


***Figure 4-XX**. Deploy your Model to an Endpoint Page*.To deploy your m
In Step 1 you define your endpoint. You select a region and determine how your endpoint will be accessed.

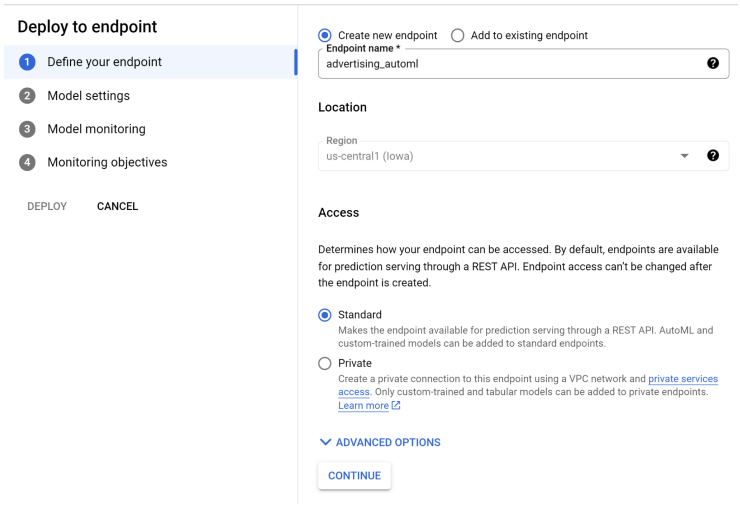


***Figure 4-XX**. Step 1 - Define your Endpoint.*

In Step 2 you add the model and add traffic split as shown in **Figure XX**. A traffic split in Vertex AI is a way to distribute traffic between multiple models that are deployed to the same endpoint. This can be useful for a variety of purposes, such as:

- *A/B testing*: Traffic splitting can be used to A/B test different models to see which one performs better.
- *Canary deployments*: Traffic splitting can be used to deploy a new model to a small percentage of users before deploying it to a larger audience. This can help to catch any problems with the new model before they affect too many users.
- *Rollouts*: Traffic splitting can be used to rollout a new model to users gradually. This can help to mitigate the impact of any problems with the new model.

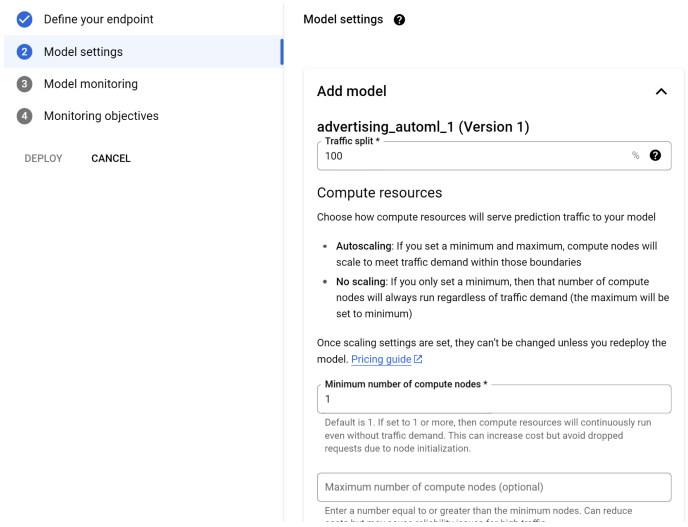


***Figure 4-XX**. Endpoint showing traffic split.*

Next, you choose how compute resources will serve the predictions to your model (shown in **Figure 4-X**). For this lab, use the minimum number of compute nodes (virtual machine servers). Under *Machine Type* select Standard.

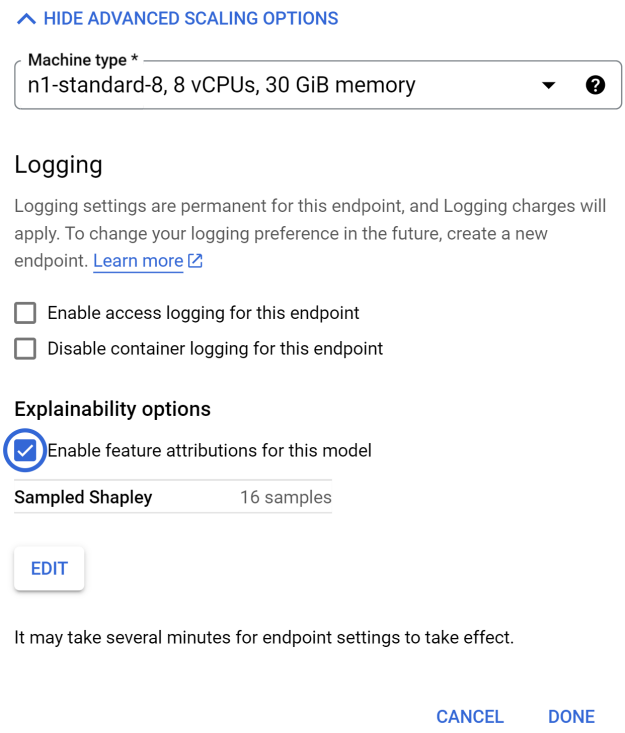


***Figure 4-XX**. Select Compute Resources*

Note: *Machine types* differ in a few ways: (1) Number of virtual CPUs (vCPUs) per node; (2) Amount of memory per node; and Pricing.

There are a few factors to consider when choosing compute resources for a prediction model:

- *The size and complexity of the model*: The larger and more complex the model, the more compute resources it will need. (This is more applicable to custom coding neural networks).
- *The number of predictions that will be made*: If you expect to make a large number of predictions, you will need to choose a compute resource that can handle the load.
- *The latency requirements*: If you need to make predictions in real time or with very low latency, you will need to choose a compute resource that can provide the necessary performance.
- *The cost*: Compute resources can vary in price, so you will need to choose one that fits your budget.

Once you have considered these factors, you can start to narrow down your choices. Here are a few examples of compute resources that can be used to serve prediction models:

- *CPUs*: CPUs are the most common type of compute resource and are a good choice for models that are not too large or complex.
- *GPUs*: GPUs are more powerful than CPUs and can be used to speed up the training and inference of large and complex models.
- *TPUs*: TPUs are specialized hardware accelerators that are designed for machine learning workloads. They are the most powerful option and can be used to train and serve the most demanding models.

As part of Step 2 under Model Settings, there is a Logging settings. If you enable endpoint logging, charges will apply. Thus, for this exercise, please do not enable it.

A variety of information is logged during model training, including:

- *Model hyperparameters*: The hyperparameters are the settings that control the model training process. These hyperparameters are logged so that you can track how they affect the performance of the model.
- *Model metrics*: The model metrics are the performance measures of the model during training. These metrics are logged so that you can track the progress of the model training process and identify any potential problems.
- *Model artifacts*: The model artifacts are the files that are generated during model training. These artifacts include the model itself, as well as the training data and the training logs.
- *Model metadata*: The model metadata is the information about the model, such as the name, the version, and the creation date.

Here are some of the benefits of logging during model training:

- *Troubleshooting*: Logs can be used to troubleshoot problems with model training. For example, if the model is not performing as expected, you can review the logs to see if there are any errors or warnings.
- *Monitoring*: Logs can be used to monitor the progress of model training. For example, you can track the training loss and the accuracy to see how the model is performing over time.
- *Reproducing results*: Logs can be used to reproduce the results of model training. This can be helpful if you need to retrain the model or if you need to share the model with others.
- *Auditing*: Logs can be used to audit the model training process. This can be helpful if you need to track who has access to the model training data or if you need to investigate a security incident.

The next setting is Explainability options, which do not carry charge. Check the Enable feature attributions for this model.

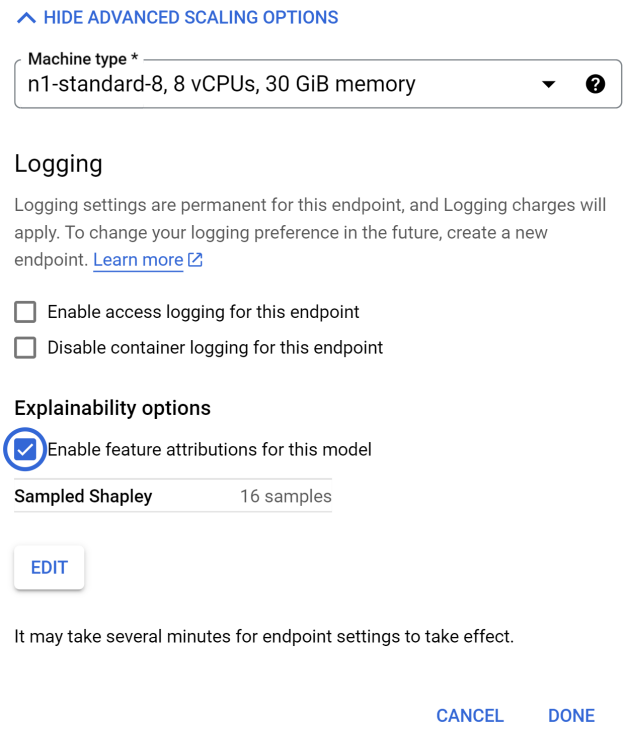


***Figure 4-XX**. Select Machine Type and Enability Options*

Step 3 is Model Monitoring. Do not enable it for this lab.

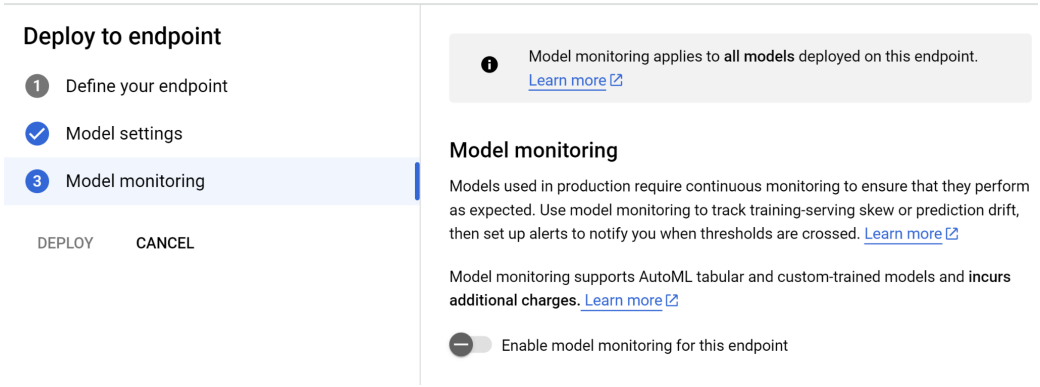


***Figure 4-XX**. Model Monitoring configuration window.*

Click Deploy to deploy your model to the endpoint.

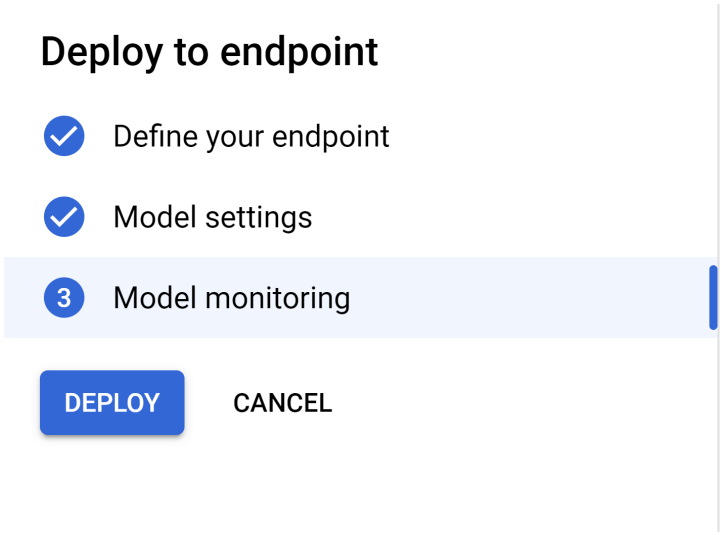


***Figure 4-XX**. Deploy to Endpoint without Model Monitoring*

Figure 4-X shows the message you should see to create the endpoint.

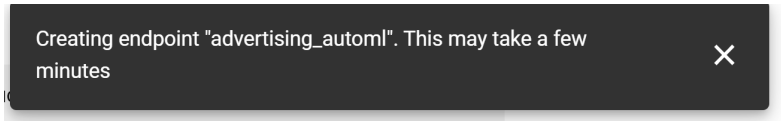


***Figure 4-XX**. Creating the endpoint popup message.*

After the endpoint is created and the model deployed to the endpoint, you should receive an email of the endpoint deployment status. If deployment was successful, you can start making predictions. There are four steps:

1. Go to Model Registry.
2. Select your model.
3. Select the model's version.
4. Scroll down until you see the Test your model pge.

Figure 4-X shows the Test your Model page. Note, this page could be an app or web page that looks like this - where you and your team input media channel values and predict sales volume.

Click the PREDICT button.

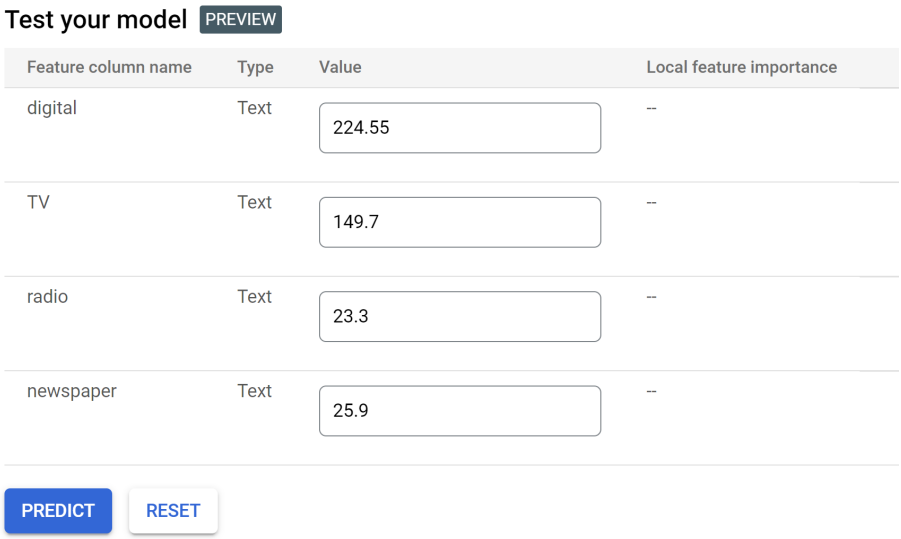


***Figure 4-XX**. Testing Page for Online Predictions*

After clicking the PREDICT button, you will get a prediction for your label (sales).

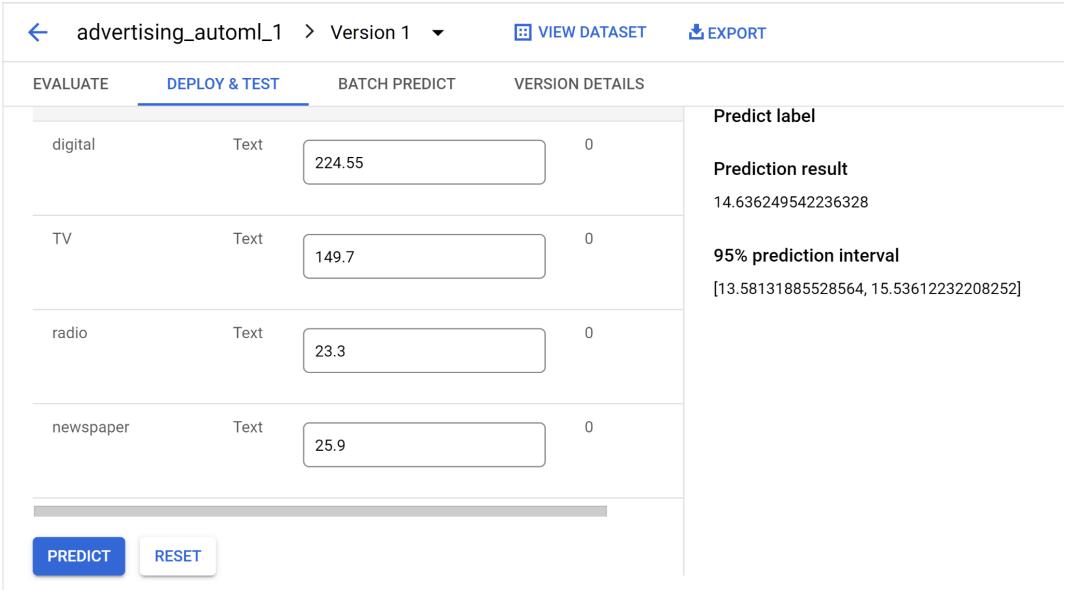


***Figure 4-XX**. Prediction of Sales Volume Based on Initial Values*

Regression models return a prediction value. **Figure 4-X** shows a sales prediction result value of 14.63 - which is very close to the mean from the sales histogram (**shown in Figure 4-X**). The prediction interval provides a range of values that the model has 95% confidence to contain the actual result. So, since the sales prediction result is 14.63, and the prediction interval is a range between 13.58 and 15.53, you can be 95% certain that any prediction result will fall within this range.

.

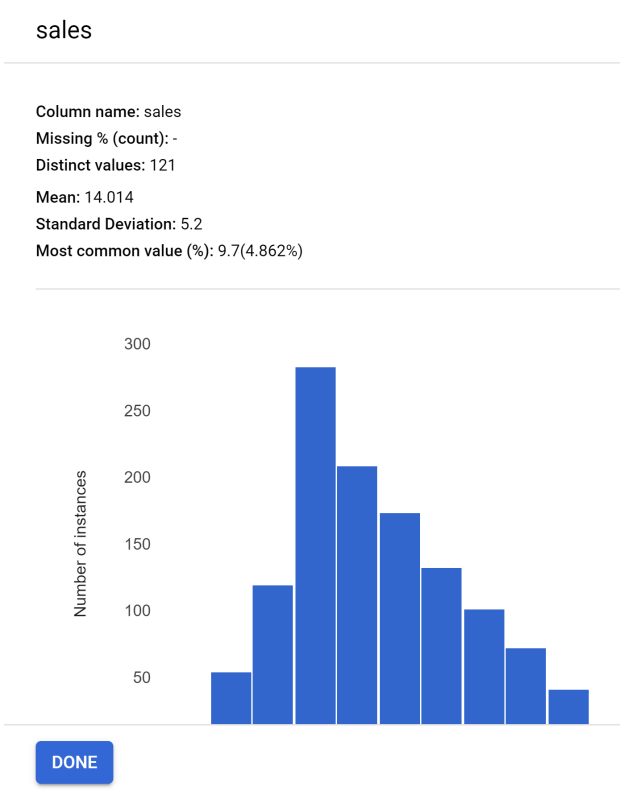


***Figure 4-XX**. Sales Histogram Showing Mean of 14.01*

***Begin Note
Prediction intervals are often used in regression analysis. A prediction interval is different from a confidence interval - though both are statistical measures that provide an estimate of the range of values within which a true value is likely to live. A confidence interval focuses on past or current events and is used to estimate a population parameter (e.g. standard deviation, mean).

A prediction interval is used to predict the value of a future observation - given what has already been observed. It provides an estimated range of values that may contain the value of a single new observation, based on previous data.

If you create a regression model, you can use it to develop a prediction interval that can determine where the next data point sampled may appear. Prediction intervals are wider than confidence intervals. This is because prediction intervals account for the uncertainty associated with predicting an individual value, as opposed to a population parameter.

Prediction intervals can be used to make decisions about future observations. For example, a company might use prediction intervals to decide whether to invest in a new product or service or to increase or decrease an advertising media channel budget.
***End Note

Now, let's answer those business questions.

The goal was to build a ML model to predict how much sales will be generated based on the money spent in each of the media channels.

1. **Can the model predict how much sales will be generated based on the money spent in each media channel?** Yes. Since Vertex AI allows you to input values for each media channel, you can now make decisions about future budget allocation. For example, your company's strategic media plan may now include increasing digital channel budget based upon the results obtained from a prediction.
2. **Is there a relationship between advertising spend and sales?** Yes. There is a positive linear relationship between advertising spend and sales for Digital, TV, and Radio. Newspaper spend has a weak relationship to sales.
3. **Which media channel contributes the most to sales?** TV contributes more to sales than the other media channels. How? The scatter plot you built during the exploratory data analysis section and your review of Vertex AI feature's attribution bar chart after the model was trained, show the contribution of TV to sales.
4. **How accurately can the model predict future sales?** The regression model returns a prediction value when media channel values are input into the *Predict* window to predict sales volume. The prediction results showed a sales prediction value and prediction interval. Prediction intervals can be used to make decisions about future observations so you can be 95% certain that any future sales prediction result will fall within the range shown in **Figure XX.**

Summary

In this chapter, you built an AutoML model to predict advertising media channel sales. You explored your data using Pandas, creating heatmaps, scatter plots, and histograms. After you exported the data file, you uploaded it into Google's Vertex AI framework. Then you learned how to use Google Cloud's AutoML to build, train, and deploy an ML Model to predict sales. You gained an overall understanding of the performance of your model using performance metrics and answered common business questions. You used the model to make online predictions and do a bit of budget forecasting! You are now ready to present to your Team!

***Begin Note

There are two types of linear regression: Univariate linear regression - Where you would predict sales (the dependent variable) using one independent variable (digital), as shown in **Figure XX:** and Multivariate linear regression - Where multiple independent variables are used to predict sales, as shown in **Fgure XX:**

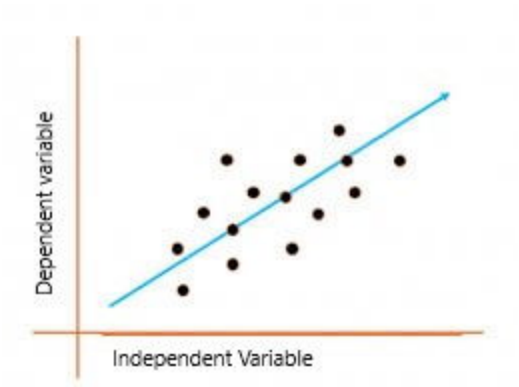


Figure 4-X. *Digital budget on the 'X" axis (independent variable) and sales on the "y" axis (dependent variable).*

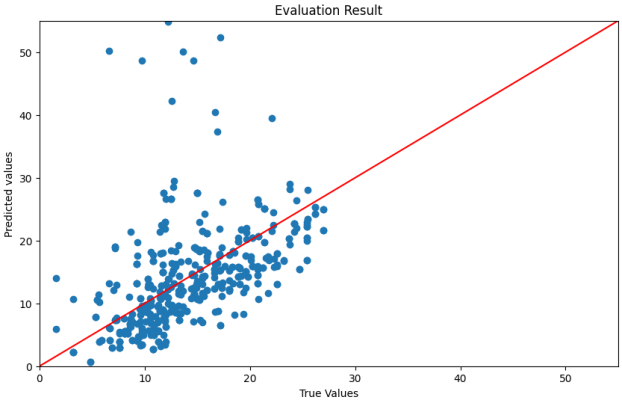


Figure 4-X. *Digital, newspaper, radio, and TV on the 'X" axis (independent variables) and sales on the "y" axis (dependent variable) showing the regression line.*

- *Univariate formula: Y = mx1 + b*

y (sales) = ***m*** *(slope)* ***x*** *(digital) budget) +* ***b*** *(y-intercept).*

- *Multivariate formula: y = mx1 + mx2 + mx3 + b*

*y = m * digital budget + m * radio budget + m * newspaper budget + m * TV budget + b*

*** End Note