Business Analytics & AI Demo

Intelligent Customer Care

with Bot Framework, Cognitive Services, Big Data, Machine Learning, Dynamics

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

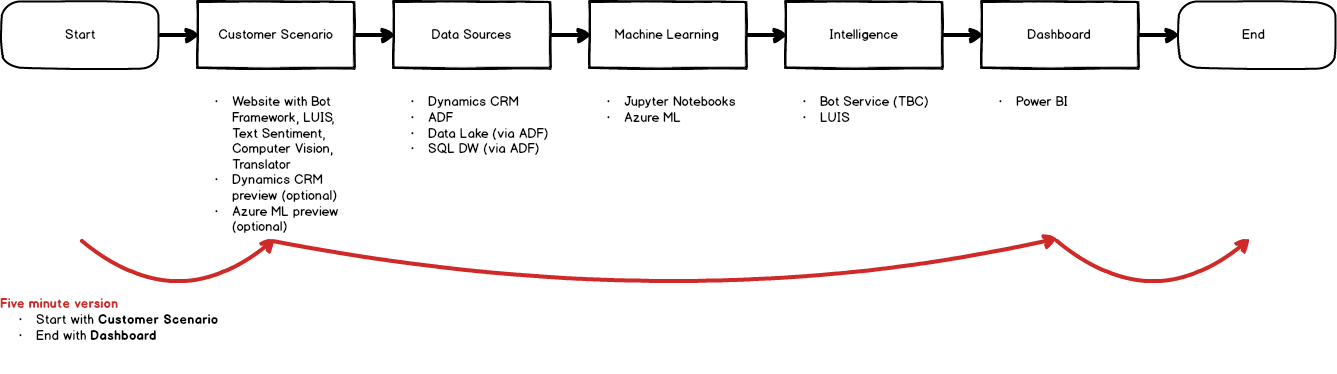
© 2017 Microsoft. All rights reserved.

# Introduction

This demo walks through a customer experience who is looking for insurance for her new car. The intelligent customer care bot will engage in a conversation with the customer and recommend quotes based on the customer profile and other factors. At some point, the bot will involve a human agent when it senses churn risk. The human agent can access the customer profile information and determine the best promotions and offers. Finally, the entire process metrics can be viewed from the operational dashboard and provide leaders on insights of the business. The demo has two sections; business demo and an extended technical demo.

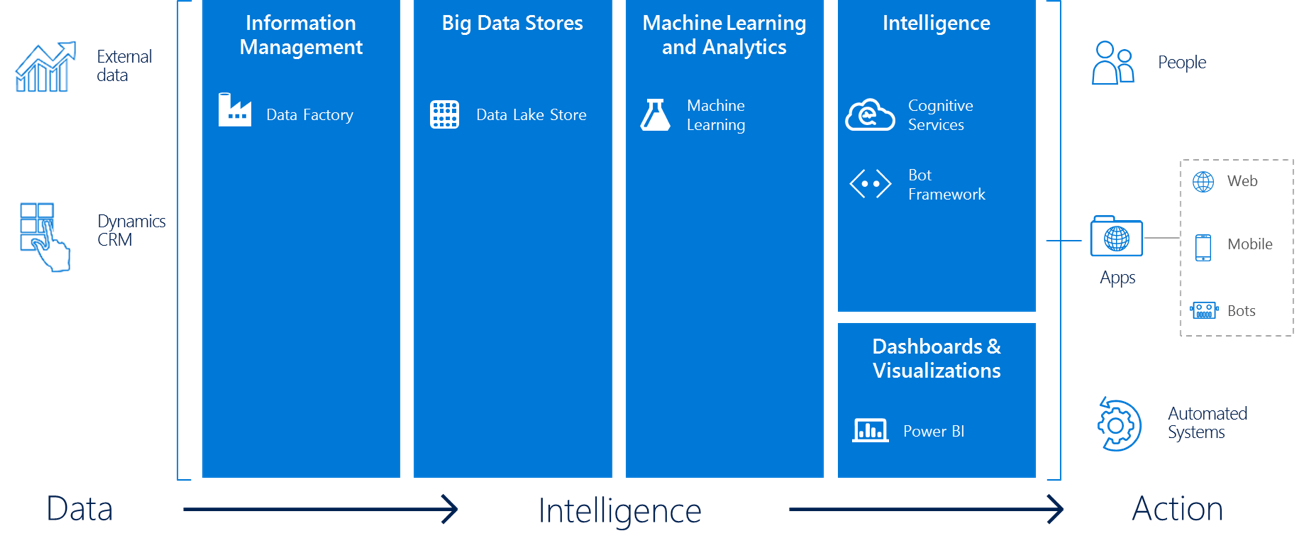
**Scenario walkthrough:**

1. The customer just purchased a new car and needs an insurance quote.
2. The customer goes to their insurance provider’s website for support via a “Get Started” link.
3. The customer service bot requests verification information.
4. After verification, the bot asks for information about the new car including a picture.
5. The bot also accesses customer’s profile in Dynamics CRM.
6. The bot recognizes that the customer’s dependent is going to turn 16 and up-sells to add the dependent to the policy.
7. Based on the CRM data + regional demographic data, the insurance provider uses machine learning to accurately assess the customer’s risk of churn and provides a quote.
8. The customer declines at which point the bot hands over to an agent.
9. Alternatively if the customer accepts the quote, the bot can directly transact on behalf of the customer showing integration and automation into the insurance company’s business process.



**Components:**

1. Insurance scenario: present the customer/bot interaction from initial contact through to agent handoff
2. Data: show CRM and Big Data sources powering the interaction
3. Machine Learning: show how the underlying model was built to support policy selection
4. Intelligence: show recommendations, translation, LUIS, Bot Framework offerings powering the interaction
5. Dashboard: show sentiment analysis output



# Setup Instructions

Once the above steps have been completed, open Microsoft Edge with the following tabs:

1. [Insurance website](https://cisbot-prod.azurewebsites.net/) (ensure AdBlock, Privacy Badger, and similar tools are disabled for the site)The login details are: Bot / Azure (note: you may also be prompted for these creds in CRM)
2. *[In Private Browsing Window]* [Dynamics CRM Online](https://litwareinsurancebotdemo1.crm.dynamics.com) - My Open Opportunities view (from home, click ‘Sales → Opportunities’)  
   The login details are: admin@litwareinsurancebotdemo1.onmicrosoft.com / @uzKg#R4n!S5
3. [Power BI](https://msit.powerbi.com/groups/586b07d3-931c-467b-9db2-45d90c2dc169/dashboards/e60bd6c3-984c-4fba-afef-105913049429) dashboard (under Workspaces > CisBotDemo)  
   The login details are: your microsoft.com credentials. If this is your first time accessing the dashboard, feel free to request for access and the owner will grant you access.

# Script

## Business Demo (15 minutes)

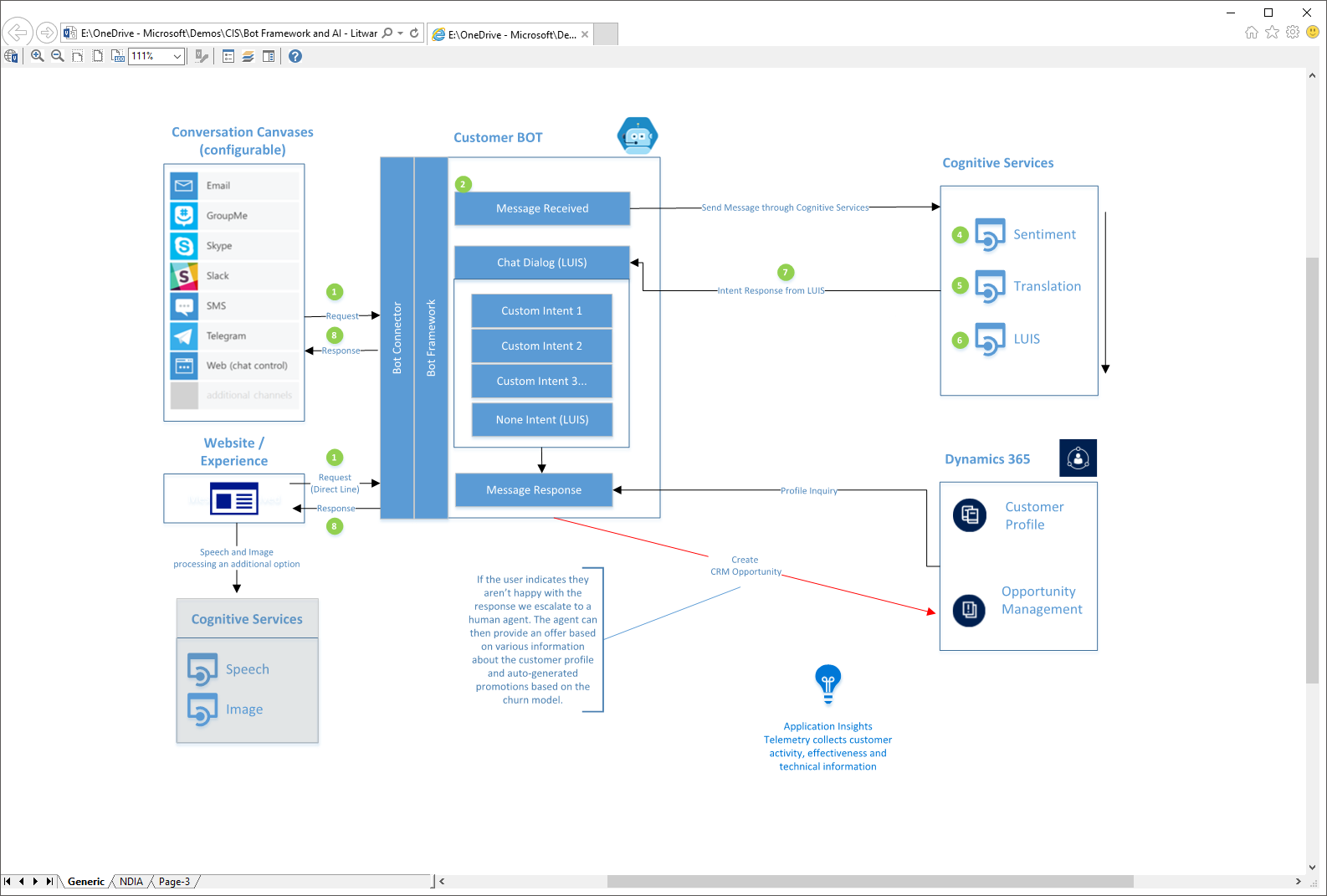
|  |  |  |
| --- | --- | --- |
| **Screenshot** | **Steps** | **Notes** |
| **Get Started** | | |
| cisbot-test.azurewebsites.net-.png | Switch to the Insurance website in Microsoft Edge.  Click ‘Get Started’. | *Alice just bought a new car yesterday. As she cannot drive without insurance, she goes to the website of her insurance company.*  *Alice browses to the website looking to get a new automotive policy for her new car.* |
| **Introduction to the Bot** | | |
| cisbot-dev.azurewebsites.net-.png | Type “What can you do?”  *OR* press the up arrow key to auto-type.  Type “Where is the company based?”  *OR* press the up arrow key to auto-type. | [COMMENT] This section is optional. You can skip directly to getting a policy if time is limited (next row).  *Bots can support many different patterns of interactions with your customers.*  *They represent an opportunity to empower customers to interact with your business in personal, natural ways.*    *Here we can ask the bot simple questions and get immediate responses. It’s able to identify the intent of the question regardless of how it’s phrased by the user.*    *The next interaction flow is more complex. Here, the bot maintains state as it leads the user through a series of steps to achieve a shared goal.* |
| **Engaging on the quote** | | |
| cisbot-dev.azurewebsites.net- (1).png | Type “I need insurance” *OR* press the up arrow key to auto-type.  Click “Auto”.  Flow #1: auth with voice Click “Yes”  Click the microphone and say the authentication phrase. Click again once you’re finished.  Flow #2: auth with PIN Click “No” Type “Alice Olson” *OR* press the up arrow key to auto-type. Type “1234” *OR* press the up arrow key to auto-type. | *As Alice is looking for a new insurance, let’s type: “I need insurance”.*    *The bot starts a conversation flow with Alice to help her with her purchase.*    *First, the bot works to understand what the customer needs. It uses LUIS to determine how to handle her inquiry. For example, if I typed “I need auto insurance”, it wouldn’t ask me what type of insurance I was wanting.*    *It then authenticates the customer using an experience more comparable to a traditional call center.* |
| **Infuse intelligence within business process** | | |
| cisbot-dev.azurewebsites.net- (2).png | Click “Yes”. | *Thanks to Dynamics CRM, the bot is able to connect to business processes and retrieve her customer information.*    *Using this information, the bot has been able to suggest adding her daughter to the policy. You can see there how you can take action through solutions that infuse intelligence within business process.* |
| **Services** | | |
| cisbot-dev.azurewebsites.net- (3).png | Type “Ford” *OR* press the up arrow key to auto-type.  Type “Fusion” *OR* press the up arrow key to auto-type.  Click “2015”.  Feel free to type any car or model to personalize this part of the demo. Your selection will appear in the CRM view in the next section of the demo. | *The bot now is collecting additional information related to the car Alice just acquired.*    *Not only can you provide the information by typing but you can also see beautiful cards, designed with buttons for common answers help accelerate the process.* |
| cisbot-dev.azurewebsites.net- (4).png | Click “Yes”.  Click on the picture icon and upload dog.jpg. Feel free to use any image you like to personalize this part of the demo.  Click on the picture icon and upload harry.jpg.  Click on the picture icon and upload ford\_fusion.jpg. You can download any image of the car from the internet for use. | *Here, Alice is asked to upload the picture of her car for the record. However, she uploads the wrong picture.*    *Uploading pictures is critical for Insurance companies, especially if you’ve had an accident.*  *Thanks to the Computer Vision API, it is now possible to understand what is inside an image, and verify if the content is appropriate.* |
| **Smart Insights** | | |
| cisbot-dev.azurewebsites.net- (4).png |  | *In the background, the bot is pulling together all the information Alice has provided and generates a quote.*    *This includes inputs from traditional risk assessment systems, as well as more modern sources like our Azure Machine Learning churn model.*  *We can truly capitalize on insights hidden within – and beyond – your unique data.* |
| cisbot-dev.azurewebsites.net- (5).png | Type: “No, that’s way too expensive!” *OR* press the up arrow key to auto-type. | *The bot presents his quote.*    *At this point, if the customer likes the quote, we can take them to a payment processing screen or even use a payment method already on file.*    *But let’s see what happens though if we respond negatively to the bot.* |
| cisbot-dev.azurewebsites.net- (6).png |  | *The bot can even detect the negative sentiment of the customer’s response thanks to Cognitive Services with the use of the Text Analytics API, or even leveraging an Azure ML churn model to measure if the threshold for continued interaction is too big.*    *In this scenario, the bot can reach out to a human agent to help finalizing the sale.* |
| cisbot-dev.azurewebsites.net- (7).png | Type “Bonjour”.  Type “Hola”. | [COMMENT] This section is optional. You can skip directly to behind the scenes if multi-lingual capabilities are not relevant.  *Before we leave the front-end and dig into the technologies that power this experience, it’s important to note that these capabilities are not just limited to the North American market.*  *Thanks to Translator APIs, the bot can detect the language being used by the customer and translate its responses to match.* |
| **Behind the scenes** | | |
| litwareinsurancebotdemo.crm.dynamics.com-main.aspx.png | Switch to Dynamics CRM:   * Show existing customer information including family details (left pane) * Show churn risk in activity feed (center pane) * Show inline chat window (right pane) | *From Dynamics CRM, we can see the opportunity the bot has built for the agent. It pulls in existing customer history and then appends information captured in the chat. It also leverages machine learning to identify customer churn risk and make smart suggestions regarding other products the customer may wish to purchase.*  *This ensures the opportunity is ready for a human agent to pick up and successfully close the deal.* |
| app.powerbi.com-.png | Switch to the Power BI Dashboard. | *From a customer profile perspective, we gathered a considerable amount of valuable data during this interaction. All of the information captured by the bot and shared by Alice has been added to CRM.*  *Contact centers will have a complete view of each person’s interaction in the CRM, but also all the customer requests aggregated in order to find smart insights, thanks to powerful visualization tools such as Power BI.* |
| app.powerbi.com- (1).png | Click on the map.  Click on the right red circle in Australia. | *By using technology such as sentiment analysis, we can identify and respond to trends in near real-time. For example, we can see our performance in Australia has dropped recently. If we drill in, we can quickly see why: our major competitor, Contoso Insurance, and the word expensive emerge indicating we’re no longer price competitive in that market.* |

End of business demo. The below section dives deeper into the technical details of the services on the back end.

## Technical Deep Dive Demo – Optional (30 minutes)

|  |  |  |
| --- | --- | --- |
| **Screenshot** | **Steps** | **Notes** |
| Open the browser with the following tabs. You will need to request access to the demo tenant admin for this section.   * [Azure Portal](https://portal.azure.com) with demo resource group blade open (the resource group is called cisbot) * Azure Data Factory Monitoring Portal (from cisbotadf in Azure, click ‘Monitor and Manage’) * [Azure Machine Learning workspace](https://studio.azureml.net) named cisbotml * IDEAR Jupyter Notebook (from the AML workspace); make sure to trust it (File > Trust Notebook) * Azure Machine Learning Web Service portal for the CisBot.2017.1.8.22.9.35.758 service * [LUIS Portal](https://www.luis.ai) | | |
| **Dynamics CRM** | | |
| litwareinsurancebotdemo.crm.dynamics.com-main.aspx.png | Switch to Dynamics CRM Online in Microsoft Edge.  Highlight the top pane. | *Jane can see Alice’s opportunity record in CRM. It shows her history with the company, as well as the engagement with the bot, so the agent is fully prepared to have a successful conversation.*  *On the top, we see information about the sales stage. As the bot has gathered more information, it’s advanced the sales stage from qualify, to develop, and now to proposal. We’ve also customized the sales stage so it provides an immediate guide to the agent on what they need to do and can mark each step complete.* |
| litwareinsurancebotdemo.crm.dynamics.com-main.aspx.png | Highlight the left pane. | *On the left, we see information about:*   * *Client: her dependents & her license status* * *Car: make, model, year, and an approximate valuation pulled in from an external system* * *History: payment & credit performance with the company*   *The agent knows at a glance the key information about the person they are about to chat with.* |
| litwareinsurancebotdemo.crm.dynamics.com-main.aspx.png | Highlight the middle pane. | *In the middle, we see a history of this opportunity. As the bot has progressed it, from initial interest, through identifying the type of policy, through to escalating it to an agent, entries have been added to the activity wall. It’s easy for the agent to understand how the escalation occurred and the bot continues to help by giving recommendations (like the risk of churn).*  *The risk of churn is calculated using an Azure ML model. It pulls together a variety of data sources to help understand and predict customers at risk of leaving. This allows agents to move more aggressively rather than unexpectedly lose a customer.* |
| litwareinsurancebotdemo.crm.dynamics.com-main.aspx.png | Highlight the right pane. | *On the right, we see the agent’s view of the chat embedded right in CRM. Rather than having another app or tool, it’s part of their view of the customer. It also shows the interaction the customer has already had with the bot so the agent has all the context they need to land the deal.* |
| litwareinsurancebotdemo.crm.dynamics.com-main.aspx (1).png | Highlight the lower pane.  Click on the ‘Suggestions’ link on the Auto Insurance product line item. | *If we scroll down, we can also drill into the price that has been provided via the bot on the website. Here we can see the additional family coverage added at the bot’s suggestion. We can also see other suggestions that the agent might like to make to the customer.*  *The suggestions are powered by the Recommendations API. This uses machine learning to build a model based on the data already in your CRM system in order to make better predictions of other products that might be relevant to your customers.* |
| **Azure Data Factory** | | |
| datafactory.azure.com--datafactory-edit-subscription-8d5f12c8-5826-4caf-be5b-b386b2daf671-resourceGroup-cisbot-dataFactory-cisbotadf--tenantId=72f988bf-86f1-41af-91ab-2d7cd011db47.png | Switch to the Azure Data Factory monitoring portal in Microsoft Edge. | *Building machine learning models to predict customer preferences or risk is no longer limited to historical business transactional data. It also includes data sets like census, clickstream, credit history, driving records, weather, demographics, crime stats, etc. Azure enables organizations to work with this diverse and evolving set of data through SQL DW, for traditional structured data, and Data Lake, for more diverse sources.*  *Azure Data Factory enables us to pull together data from multiple sources in preparation for use in our big data and machine learning services.* |
| datafactory.azure.com--datafactory-edit-subscription-8d5f12c8-5826-4caf-be5b-b386b2daf671-resourceGroup-cisbot-dataFactory-cisbotadf--tenantId=72f988bf-86f1-41af-91ab-2d7cd011db47 (1).png | Highlight the CrmDataset, ClaimsDataset, and AzureBlobCensusDataset. | *Here we’re capturing data from three different sources to build our ML model:*   * *CRM data is being pulled from Dynamics CRM Online via an OData feed.* * *Historic claims data is being pulled from our on premises SQL Server. We’re using the Data Gateway software which provides a secure connection from our network to Azure.* * *Census data is being pulled from a large CSV file stored in Azure Storage.* |
| datafactory.azure.com--datafactory-edit-subscription-8d5f12c8-5826-4caf-be5b-b386b2daf671-resourceGroup-cisbot-dataFactory-cisbotadf--tenantId=72f988bf-86f1-41af-91ab-2d7cd011db47 (5).png | Scroll right and select the HDInsightHive AggregationPipeline. | *The data is pulled in to Azure Storage from the three sources and then we use HDInsight to process it. All we need to do is provide a Hive script that aggregates and joins the data and ADF spins up an HDInsight cluster on demand for execution.* |
| datafactory.azure.com--datafactory-edit-subscription-8d5f12c8-5826-4caf-be5b-b386b2daf671-resourceGroup-cisbot-dataFactory-cisbotadf--tenantId=72f988bf-86f1-41af-91ab-2d7cd011db47 (6).png | Scroll right and select the CopyToDataLakeStorePipeline. | *Now we have a single data set that contains all of the information we have about a customer including personal details, aggregate claim history, and relevant census data. Finally, we store it in Azure Data Lake so we can analyze it further. First we’ll use Jupyter to explore and understand the data and then we’ll use Machine Learning to build predictive models.* |
| **Machine Learning** | | |
| notebooks.azureml.net-n-uEhKGYdl2Ms-notebooks-IDEAR.ipynb.png | Switch to the Jupyter Notebook in Microsoft Edge.  Run the notebook (Cell menu > Run All). | *Let’s look at how we built the customer churn model. It samples a range of data sets - from traditional CRM data through to things like claim history and census demographics - to provide a complete view of our customers.*  *The process of building a good model starts with exploratory data analysis. Jupyter is an open platform for interactive data science and supports a range of language and options for collaboration. It’s also built into Azure ML. Let’s execute this notebook and review the results.* |
| notebooks.azureml.net-n-dbbyRSMq2Pc-notebooks-IDEAR.ipynb.png | Scroll down to “Rank variables based on linear relationships with reference variable (on sampled data)”.  Change the “Top Num Vars” to 10. *Don’t click Export.* | *Here we have a notebook that uses Python 2. We also support Python 3 & R. The first snippet of code visualizes the distribution of the target in our data set. Understanding the patterns in the data will help in building the right ML model.*  *This is based on the IDEAR process, a publicly available methodology for understanding new data sets and the relative importance of different features. As you can see, it’s not just a static report: it uses Python modules to generate an interactive canvas to help us explore - in this case - the relationships between different variables and churn.* |
| studio.azureml.net-Home-ViewWorkspaceCached-5b439a0de13a4a76bca213be8d702cfd.png | Click on the Experiments tab.  Click on “CustomerChurnTraining”. | *Once we understand the data, we can build an experiment. Azure Machine Learning provides a visual interface to import, refine, & explore data and then train, refine, and score models. This churn model pulls data from our data sources in Azure and evaluates a range of algorithms to find the best choice for our data.*  *This model has several stages:*   * *Ingest data: supports many different sources incl. SQL, Hive, and plain flat files* * *Prepare data: add metadata, clean missing values, remove duplicates, and run inline R scripts* * *Build data sets: split training vs. test, as well as things like synthetic minority oversampling (i.e. increase incidence of low incidence examples before model building)* * *Build models: an experiment can consist of one or models, using out of the box algorithms or R* * *Evaluate: use an extensive array of statistical functions to analyze the performance of your models* |
| studio.azureml.net-Home-ViewWorkspaceCached-5b439a0de13a4a76bca213be8d702cfd (1).png | Click on the “Predictive Experiment” tab. | *Taking the model from the experiment phase to a production environment can be accomplished in just a few clicks.*  *First, we select the best model from our experiment. With a single click, other models and steps are removed. This is our predictive experiment.*  *Next, we click Deploy Web Service. In this case, we’re already deployed so it’s disabled.* |
| services.azureml.net-subscriptions-8d5f12c8-5826-4caf-be5b-b386b2daf671-resourceGroups-cisbot-providers-Microsoft.MachineLearning-webServices-CisBot.2017.1.8.22.9.35.758-dashboard.png | Switch to the Azure ML Web Service portal in Microsoft Edge. | *After the service is deployed, we can work with it from a dedicated management portal. It provides not only usage information, but also code samples (under Consume tab), an interactive test page (under Test tab), and detailed API definitions (under Swagger API tab).* |
| **Intelligence** | | |
| portal.azure.com-.png | Switch to Azure Portal in Microsoft Edge.  Click on the Azure Bot Service named “cisbot-prod-demo-svc” in the resource list. | *The Azure Bot Service combines all of the features of the Microsoft Bot Framework with the limitless scale of Azure.*  *It leverages Azure Functions, Microsoft’s serverless compute service, to allow your bot to be hosted without having to provision a dedicated host. You’re simply charged for the compute time you use so when you’re bot isn’t being called, you don’t incur any cost. And when your bot gets busy, Azure scales up seamlessly to handle the load.* |
| portal.azure.com-.png | Type “I need insurance” in the chat client. | *It also provides a quick way to get started building bots through a developer friendly web portal. Here we can write code to add capabilities to our bot and test them immediately through the web client.*  *As you can see, it behaves just like the full website.* |
| portal.azure.com- (1).png | Click on the “dialogs” folder.  Click on the “Dialog.csx” file. | *Here’s an example of a LUIS dialog. LUIS takes natural language and identifies the user’s intent. The attributes above each method indicate to LUIS which method to call for which intent. For example, if LUIS determines the user is saying hello, it calls the first method.*  *Under the covers, this is all powered by Azure Functions, so it benefits from the same seamless source control integration and local developer experience.* |
|  | Switch to LUIS Portal in Microsoft Edge.  Select the ‘cisBot’ bot. | *LUIS lets our app understand natural language. Through the portal, we can build and/or re-use models to understand the user’s intent and then easily integrate with our application.*  *After we sign in to authenticate to our LUIS account, we can see a list of all of our LUIS applications. From here, we can edit our application to add new intents and manage the state of our LUIS application.*  *On the left, we can see all of our pre-populated intents. We can type in a few utterances and see how well our LUIS app can match our plain human language with what it believes our intent is. This also enables our LUIS application to learn new dialog variations to better understand colloquialisms and phrasing used by our users.* |
|  | Type “I need insurance” into the utterance box.  Type “I want insurance for my new vehicle” into the utterance box. | *Typing an utterance into the text box, such as “I need insurance” shows that our LUIS app can match our utterance to an intent we have registered as* ***INeedInsurance****, with a (1) or 100% confidence. This indicates our LUIS application can understand our intent.*  *We can prove this by making a slight modification to the question: “I want insurance for my new vehicle”. We can see LUIS struggles a little more with the second phrasing, but we still get a 90% match on our intent.* |
|  | Click the “+” icon next to the “Intents” header on the left nav bar.  Complete the form:   * Introduction: “name” * Action Parameters:   + Name: “name”   + Type: “name”   Click “Save”. | *We can teach* ***new intents*** *to our LUIS app. Parameters are used to tell our intent to look for specific* ***entities*** *in any phrase that matches our intent. In this case, our* ***Introduction*** *intent will look for a* ***name*** *given by the user.* |
|  | Click the “+” icon next to the “Entities” header on the left nav bar.  Type “name”.  Click “Save”. | *We can register new entities to register with our intents.* |
|  | Type “My name is Daniel” in the utterance box. | *We can see that with our new* ***introduction*** *intent, the* ***name*** *parameter that we matched earlier is matched, as seen in* ***orange.*** *We also see our* ***introduction*** *intent is matched with a 98% confidence.* |
|  | (no action, visible on the right side of the screen) | *We can see how successfully LUIS is able to match our utterances with our intents. Green is good; it means that the app is correctly predicting our intent.* |
|  | Click “Publish” at the top of the left nav bar. | *Once we’re satisfied with whether our LUIS app is adequately matching our intents, we can publish the app to a live* ***https*** *endpoint, as we see here. Typing in a test query shows how the GET request is formatted via a url query.* |
|  | Type “what is my name” as the query.  Click the blue URL. | *Clicking the query link yields the results. We can see how our results are formatted in JSON, and can be consumed by our bot to help enhance our conversational experience.* |
| **Dashboard** | | |
| app.powerbi.com-.png | Switch to Power BI in Microsoft Edge. | *From a customer profile perspective, we gathered a considerable amount of valuable data during this interaction. All of the information captured by the bot and shared by Alice has been added to CRM.*  *Contact centers will have a complete view of each person’s interaction in the CRM, but also all the customer requests aggregated in order to find smart insights, thanks to powerful visualization tools such as Power BI.* |
| app.powerbi.com-.png |  | *Our customer churn model, for example, is not static. As we collect more data, we can improve the model and monitor its performance as well as the effectiveness of our intervention strategies.*  *We can also use the data we collect to identify surprising insights. For example, by using text analytics, we can understand the key topics our customers are talking about with the bot and identify subject matter gaps on our website.* |
| app.powerbi.com- (1).png | Click on the map.  Click on the right red circle in Australia. | *By using technology such as sentiment analysis, we can identify and respond to trends in near real-time. For example, we can see our performance in Australia has dropped recently. If we drill in, we can quickly see why: our major competitor, Contoso Insurance, and the word expensive emerge indicating we’re no longer price competitive in that market.* |

**Demo Architecture**



# Appendix 1: Technical Implementation

Litware Insurance is a demo and thus the code base is not designed to be ‘production ready’; however, much of the demo uses out of the box features and may serve as a useful getting started example:

* Website: uses the Direct Line API to communicate with the Bot Framework. The site is built in Angular 2 and uses bespoke components to render the chat surface.
* Dynamics CRM: uses the Organization Service to retrieve customer account information including dependents (i.e. related contacts) based on name/SSN and adds opportunity/product line items as the chat progresses. The ML recommendations model used for product suggestions is based on generated data in CRM.
* Translator service: all incoming client messages are passed to this service to detect the language. This is then used by the ‘Hello’ intent to determine the appropriate language to respond in. It could be used by other intents as well.
* LUIS service: all incoming client messages are passed to this service to map a user message to an intent/code for response.
* Computer Vision service: used to recognize images with vehicles in them. It’s also used to generate a vehicle thumbnail.
* Jupyter notebook: this is based on [IDEAR](https://github.com/Azure/Azure-TDSP-Utilities/tree/master/DataScienceUtilities/DataReport-Utils/Python2) project from Microsoft and uses the dataset from the ML model.
* Azure ML model: the churn model is based on existing Azure ML gallery sample and customized for this scenario. It is invoked via a web service and the results posted to the CRM opportunity activity feed.
* Text Sentiment service: used to interpret the final user response and, if negative, invite an agent to the chat.
* Speaker Recognition service: used to verify an audio sample from the user.

The following portions may require custom development:

* Website: authentication is partially implemented. The bot service authenticates the user based on the information provided in the chat via a CRM query - and optionally a voice recognition check - but this isn’t passed through to the Angular 2 client components as they are hard coded to the ‘Alice Olson’ persona. One option would be to share this information back to the client components via the [channelData](https://docs.botframework.com/en-us/csharp/builder/sdkreference/channels.html) field. Also, the agent/multi-user chat is faked for demo purposes. Instead of [sending](https://docs.botframework.com/en-us/restapi/directline3/#!/Conversations/Conversations_PostActivity) the message as a new user in the conversation, it sends the message as the bot but updates the image on the front end. After the new user was added to the conversation, the bot would need to ignore messages between the customer and agent.
* Dynamics CRM: information regarding the opportunity - e.g. payment history & car value - is generated rather than being fetched from an API. Once a suitable API had been identified, adding it to flow would be straightforward as CRM connectivity is already in place. Also, while the bot can post to the opportunity activity feed, it isn’t configured to listen to any responses. To add this, the server could either poll CRM or potentially use a CRM plugin to detect the change.
* Dynamics CRM website embed: this IFRAME shows the chat history by querying the Bot Framework. To operationalize this control, the implementer would need to: a) store the data in a durable store rather than querying from the Bot Framework (i.e. once the chat is complete, call the Direct Line API and store the messages in Azure Storage); b) implement corporate identity (i.e. wrap the page hosted in the IFRAME with [Azure AD](https://docs.microsoft.com/en-us/azure/app-service-web/web-sites-dotnet-lob-application-azure-ad)); c) enable auto-refresh to allow updates in real-time (i.e. update TypeScript to poll the Direct Line API in the same way the full website does); and d) allow the agent to respond (i.e. send messages like the full website but use the logged in user’s identity in the from field).
* Azure Data Factory: the pipeline is real and will pull data from CRM, on premises, & blob before aggregating in Hive and writing to Data Lake; however, the resulting data set is not comprehensive enough for Machine Learning model generation so a pre-generated data set is used instead.
* Power BI: this dashboard uses a static, generated dataset.

# Appendix 2: Re-deployment Notes

These instructions are for those setting up their own deployment of this demo (from scratch). It assumes a reasonable level of familiarity with CRM, LUIS & Azure. *This part will be extended with further details in the next revision.*

**Azure - part 1**

1. Open the Azure Portal.
2. Create a new resource group (e.g. CisBot).
3. Create the following Cognitive Services in the resource group (free tier is fine):
   1. Computer Vision
   2. Speaker Recognition
   3. Text Analytics
   4. Recommendations
   5. Translator Text
   6. LUIS
4. Create a new Azure Web App in the resource group.
5. Create a new storage account in the resource group.
6. Create a new Azure Machine Learning workspace in the resource group.

**CRM**

1. Create new CRM trial with user name ‘Admin User’ (admin@...) and company name ‘Litware Insurance’.
2. Add extra user called ‘Customer Service Bot’ (bot@...) and assign a CRM license
3. Log into CRM
4. Edit Admin User and set profile photo to `jane.jpg`
5. Edit Customer Service Bot and set profile photo to `chat\_person\_120x120\_whitebackground.jpg` + add Admin role
6. Import ‘[CisBot\_1\_0\_0\_0.zip](https://cisbotstore.blob.core.windows.net/resources/CisBot_1_0_0_0.zip?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=9NR%2BKLCH%2FCXxf39%2FFkIJYt5QwDsaVpV9Zydh9df9qNE%3D)’ and then publish all customizations
7. In the solution, edit the Form, update the three IFRAME URLs to point to your website address, and then publish
8. Create a new Account:
   1. Name: Alice Olson
   2. Photo: shelly-profile\_100x100.jpg
   3. SSN (last four): 1234
   4. Address: 1 Microsoft Way, Redmond WA 98052
9. Create a new Contact:
   1. Name: Alice Olson
   2. Account: Alice Olson
   3. Gender: Female
10. Create another new Contact:
    1. Name: Vicky Lane
    2. Account: Alice Olson
    3. Job Title: Daughter
    4. DOB: 9/2/2000
    5. Gender: Female
11. Create a new Product:
    1. Name: Auto Insurance
    2. ID: INS-1
    3. Unit Group: Default Unit
    4. Default Unit: Primary Unit
    5. Default Price List: Price List
    6. Decimals Supported: 2
    7. Price List Items: Price List - Primary Unit - $1200
    8. Status: Active
12. Create another new Product:
    1. Name: Family Coverage
    2. ID: INS-2
    3. Unit Group: Default Unit
    4. Default Unit: Primary Unit
    5. Default Price List: Price List
    6. Decimals Supported: 2
    7. Price List Items: Price List - Primary Unit - $125
    8. Status: Active
13. Create another new Product:
    1. Name: Personal Injury Protection
    2. ID: INS-3
    3. Unit Group: Default Unit
    4. Default Unit: Primary Unit
    5. Default Price List: Price List
    6. Decimals Supported: 2
    7. Price List Items: Price List - Primary Unit - $195
    8. Status: Active
14. Enable Cross-sell recommendations preview
15. Configure Azure Machine Learning Recommendation Service:
    1. URL: https://westus.api.cognitive.microsoft.com/recommendations/v4.0 (or similar; check Azure Portal)
    2. Key: <from recommendations service in Azure>
16. Delete all but three opportunities. Each of the remaining should have est. value of $1325 (i.e. the demo was run completely). If you do not have three opportunities, run through the demo until you do.
17. For each opportunity, add Personal Injury Protection and remove Family Coverage from the product lines.
18. Under Product Recommendations, remove Order & Quote from the Basket Data Entities list and edit Opportunity to remove the Status = 'Won' condition from the filter.
19. Build a new model version and activate it.

**LUIS**

1. Download [cisBot.json](https://cisbotstore.blob.core.windows.net/resources/cisBot.json?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=gew0HKwmreJ1dpBM0cefVGn2e5NJr19XkTuztkoNmUc%3D). This file contains the LUIS model.
2. Sign into the [luis.ai portal](https://www.luis.ai).
3. Click New App > Import Existing Application.
4. Choose the cisBot.json file and click Import.
5. Train the model by clicking on the button in the lower left corner. Wait for it to complete.
6. Publish the web endpoint by clicking on the Publish button in the upper left corner. You don’t need to enable Action Binding for the Microsoft Bot Framework.
7. In the dialog, click Publish Web Service.
8. [Only required for full deployment] Update the app key from the LUIS subscription in the Azure Portal.

**Machine Learning**

1. Sign into the new [Machine Learning](https://studio.azureml.net) workspace.
2. Create a container called ‘jupyter’ in the storage account with public blob access.
3. Upload [all the files](https://cisbotstore.blob.core.windows.net/resources/ML.zip?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=zaqf9l5PwOEP%2Bl7QU%2B3zLWydrkV%2FKKflNPHezysRVxQ%3D) in the ML folder to the container (\*.py, \*.yaml, \*.csv).
4. Create a new Jupyter Notebook by importing [IDEAR.ipynb](https://cisbotstore.blob.core.windows.net/resources/IDEAR.ipynb?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=9Qdyh903fPStorp%2Bz2Ucxe7ZjP3goSDXM8RmF91t%2BTk%3D). Once complete, update the storage account name in the first cell.
5. Create a new ML dataset from [CustomerChurnTrainingSample.csv](https://cisbotstore.blob.core.windows.net/resources/CustomerChurnTrainingSample.csv?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=9xVWP9LNoBlE54lCqlnTxjHIoTggQ62Eo4J%2FXKuWDsU%3D).
6. Import the ML experiment using [AzureMLPS](https://github.com/hning86/azuremlps) from [exp.json](https://cisbotstore.blob.core.windows.net/resources/exp.json?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=Cy1jh8LcX%2Fe96%2FeNF6yEtYb0E3m5%2F9jNtoSpcyf388w%3D).
7. Run the ML experiment.
8. Select the right hand model training step and publish it as a predictive experiment.
9. Run the new predictive ML experiment.
10. Publish the predictive ML experiment to Azure (i.e. ‘new’ web service deployment model).

**Azure - part 2**

1. Return to the Azure Portal.
2. Create a new Azure Bot Service in the resource group.
3. Upload [census.csv](https://cisbotstore.blob.core.windows.net/resources/census.csv?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=TrSfp2OaJgn8Xhzof5Bh7wQKoGJTJtDXkn%2Fot%2BR5%2BB4%3D) to container litware and folder census-data.
4. Open the Bot Service.
5. Create the Microsoft App ID & Password when prompted. Remember to copy the password so you can paste it into the Azure portal.
6. Return to the Azure Portal and create a ‘Basic’ bot (we’ll override whatever gets created).
7. Under Settings, open App Settings and create the following keys:
   1. LuisAppId: <app ID from LUIS portal>
   2. LuisAPIKey: <app key from LUIS portal; same as LUIS service key>
   3. TranslatorSubscriptionKey: <your translator sub key>
   4. CrmUserName: admin@<your crm org>.onmicrosoft.com
   5. CrmPassword: <your trial password>
   6. CrmBotUserName: bot@<your crm org>.onmicrosoft.com
   7. CrmTargetSalesStage: Develop,Propose
   8. OrganizationServiceUrl: https://<your crm org>.api.crm.dynamics.com/XRMServices/2011/Organization.svc
   9. TextSentimentKey: <your text sentiment key>
   10. SpeakerRecognitionServiceKey: <your speaker recognition key>
   11. ComputerVisionKey: <your computer vision API key>
   12. MachineLearningServiceUrl: <your ML service URL>
   13. MachineLearningServiceKey: <your ML service key>
   14. VoiceAuthEnabled: true to enable voice auth; false otherwise
   15. VoiceAuthWithLocalResources: true to use a pre-recorded phrase rather than user mic; false otherwise
   16. ImageHostUrl: https://<your web app name>.azurewebsites.net/images
   17. StorageConnectionString: DefaultEndpointsProtocol=https;AccountName=<your storage account name>;AccountKey=<your storage account key>
8. Use WebDeploy to publish BotService.zip to your Bot Service. You can install Web Deploy from the Microsoft Web Platform installer and then invoke from the command line:   
   "c:\Program Files (x86)\IIS\Microsoft Web Deploy V3\msdeploy.exe" -verb:sync -source:package=BotService.zip -dest:auto,ComputerName='https://<bot service name>.scm.azurewebsites.net:443/msdeploy.axd?site=<bot service name>',UserName='$<bot service name>',Password='<bot service password>',AuthType='Basic' -setParam:name='IIS Web Application Name',value='<bot service name>'  
   *N.B. The password is available from the publish profile. You can download this from the Azure Portal.*
9. Return to the Web App and under App Settings create the following keys:
   1. DirectLineSecret: <from the Bot Portal, select your bot, enable Direct Line, and grab the embed secret>
   2. CrmUserName: <as per Bot Service>
   3. CrmPassword: <as per Bot Service>
   4. CrmBotUserName: <as per Bot Service>
   5. CrmTargetSalesStage: <as per Bot Service>
   6. OrganizationServiceUrl: <as per Bot Service>
   7. StorageConnectionString: <as per Bot Service>
10. Use WebDeploy to publish Server.zip to your Web App.
11. Create a new Azure Data Factory in the resource group.
12. Create a new ADF Data Gateway called OnPremisesDataGateway. OPTIONAL: create a SQL Server Developer VM with login creds theadmin / Pass@word333 (incl. for SQL Auth), install the agent, and create the database using [claims.sql](https://cisbotstore.blob.core.windows.net/resources/claims.sql?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=49Yg9qCKR0khGKPc4Ui0AVe%2FiSqaGMbPmBX26cpzIfs%3D).
13. From Visual Studio, publish to the new Azure Data Factory. This may take several attempts. Make sure to update the connection string for the Azure Storage linked service as well as updating sub ID, resource group name, & URL for CRM & Azure Data Lake. You’ll need to re-authorize ADL + CRM from the Azure Portal. Make sure to exclude both from future publishes.

**Speaker Recognition**

1. Record your verification phrase three times using an audio recording tool. A list of verification phrases are available [here](https://www.microsoft.com/cognitive-services/en-us/speaker-recognition-api). The recognition API requires recordings to be formatted as WAV files (16-bit sample, 16k bit rate, mono channel, PCM encoding). To accomplish this task you have a couple of options:
   1. Use the Insurance website. Go through the demo flow and authenticate using voice. It won’t work as expected but you will be able to download the recording from inside the chat flow.
   2. Use QuickTime or Windows Voice Recorder to record and then convert to WAV. [Here’s](http://www.screencast.com/t/K2h6t4zNOuwV) an example of how to convert a recording using iTunes.
   3. Use a pre-recorded WAV file ([SpeakerRecognitionProfileExample.wav](https://cisbotstore.blob.core.windows.net/resources/SpeakerRecognitionProfileExample.wav?st=2017-01-19T06%3A12%3A00Z&se=2019-01-20T06%3A12%3A00Z&sp=r&sv=2015-12-11&sr=b&sig=M6LnWDGE0kkWf5Gsch0P0HzXtshGk1X9C8qeJCeFung%3D)). If you opt for this, make sure you set the VoiceAuthWithLocalResources configuration property to true in the previous section.
2. In order to register a profile with the service, you’ll need a tool capable of sending HTTP requests. Examples include [Postman](https://www.getpostman.com/) (recommended), PowerShell, and curl. You’ll also need the API key for the Speaker Recognition service you created earlier in Azure.
3. Using your selected tool, create a new profile by sending a post command to the verification profile API. Details [here](https://dev.projectoxford.ai/docs/services/563309b6778daf02acc0a508/operations/563309b7778daf06340c9652).
4. Copy the profile ID you receive from the API. You’ll need this later.
5. Using your selected tool, submit the first recording by sending a POST command to the verification enrolment API. Details [here](https://dev.projectoxford.ai/docs/services/563309b6778daf02acc0a508/operations/56406930e597ed20c8d8549c). The verification profile ID is the ID you created earlier.
6. Repeat the above step two more times with the other two records you made.
7. Sign into Dynamics CRM and update the ‘Alice Olson’ account by setting the Audio Profile ID to the ID you created earlier. *This means only your voice will work with this demo environment.*