Watson Machine Learning Overview

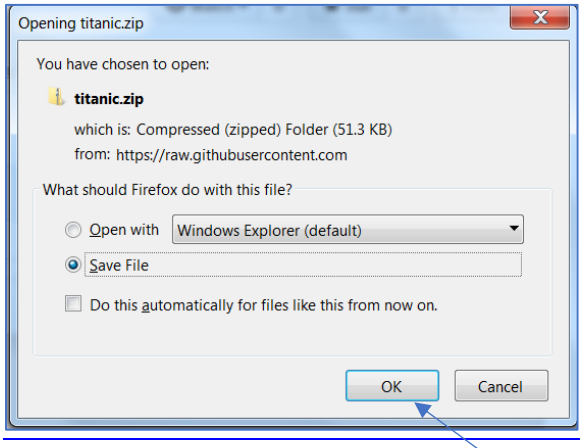
This lab will introduce the Watson Machine Learning capability using the Titanic dataset. The lab will consist of the following steps:

1. Adding a data asset to the Watson Studio Labs project
2. Creating a Model to predict whether a passenger would survive
3. Deploying and Testing the Model
4. Deploying a simple web front-end and connecting it to the Titanic deployed model.

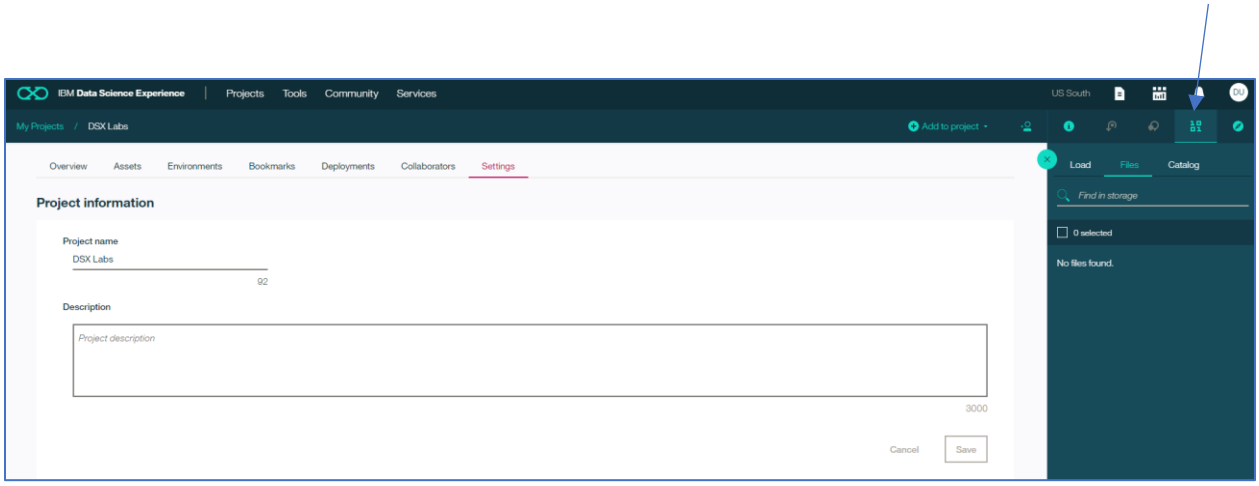
Step 1: Adding a Data Asset to the project

1. Download the Titanic data file from the following location by clicking on the link Titanic Data.

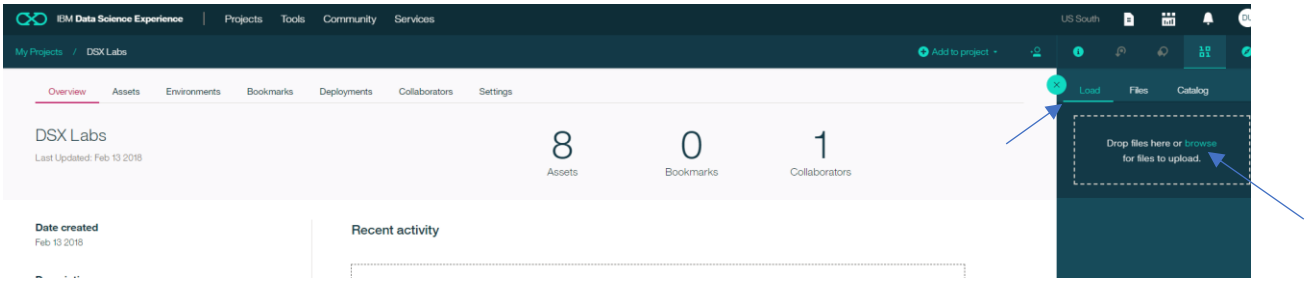
2. Click on the **OK** button in the pop-up dialog.



3. Navigate to the directory where the file has been downloaded. Unzip the titanic.zip file. There should be two files (1) titanic_cleansed.csv and (2) titanic.csv. You will use the **titanic_cleansed.csv** for this lab.
4. Go back to your Watson Studio Labs project. Click on the  icon.



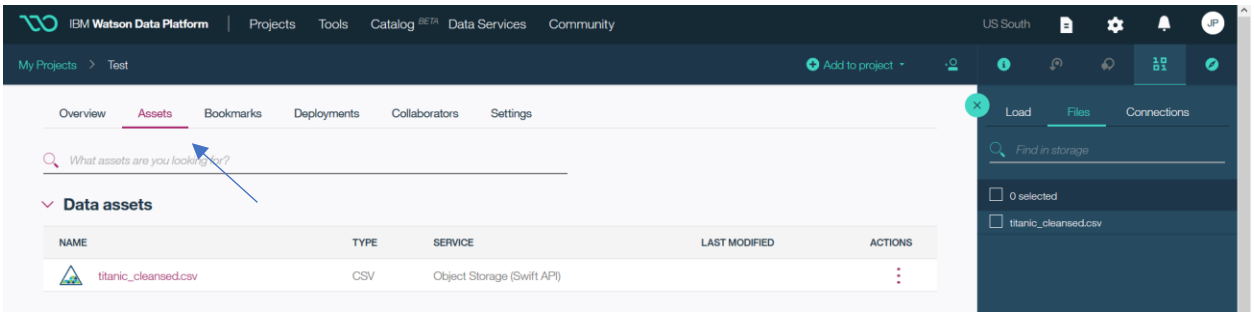
3. Click on **Load** and then **browse** and then go to the folder where the titanic_cleansed.csv is stored. Select titanic_cleansed.csv and then click Open.



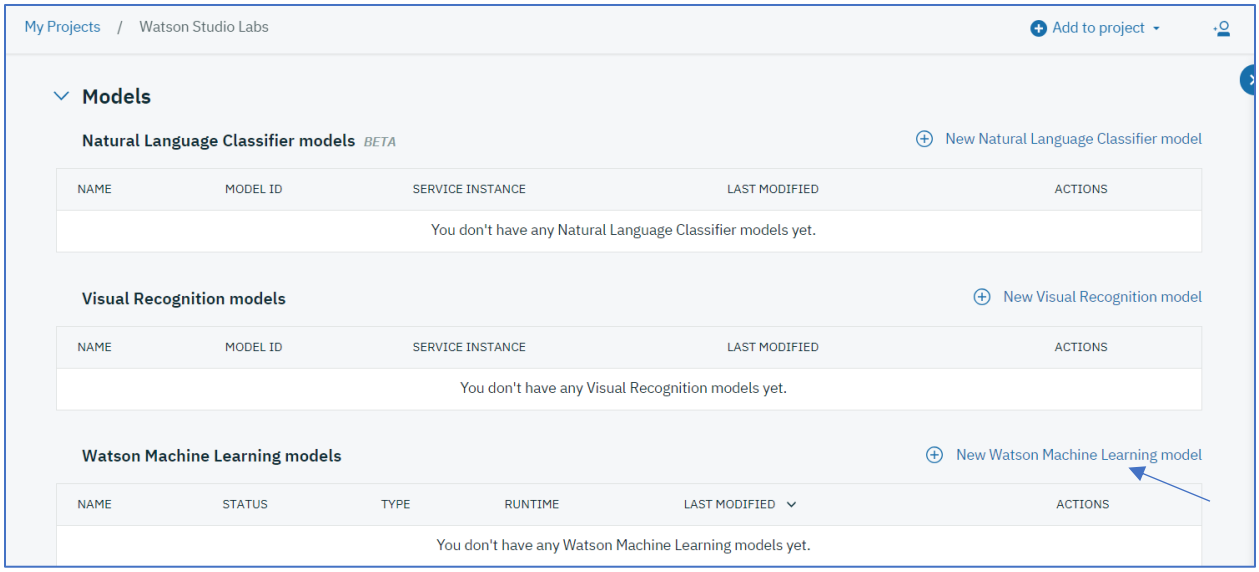
.

Step 2: Create a Model to predict survival

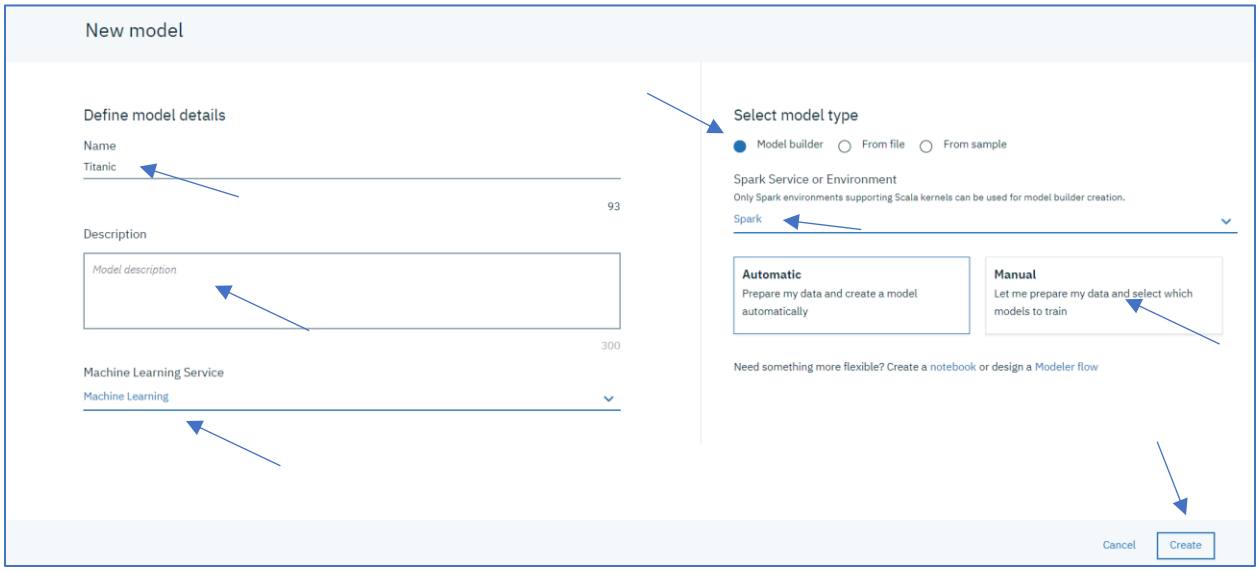
1. Click on the **Assets** Tab



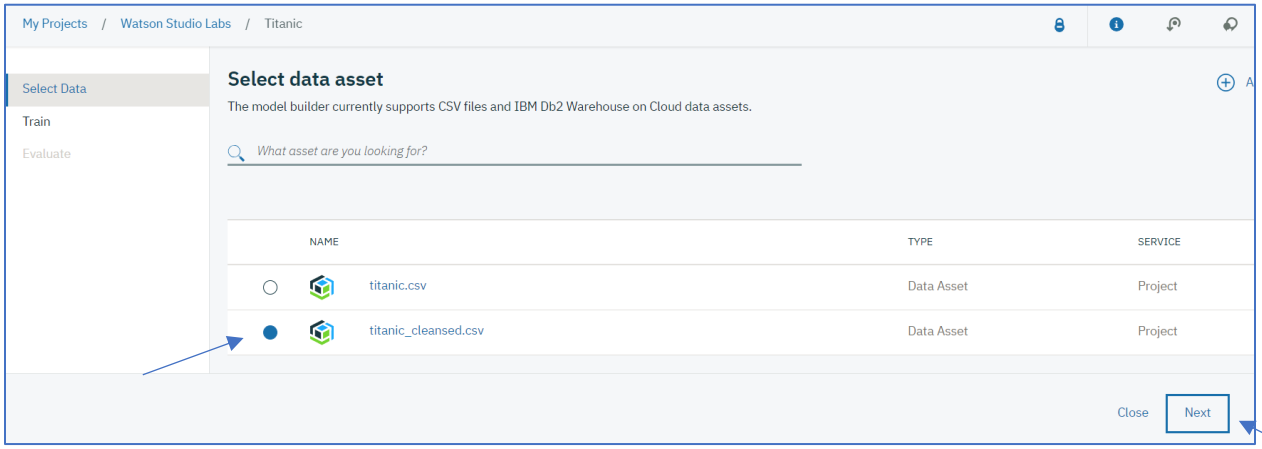
2. Click on **New Watson Machine Learning model**.



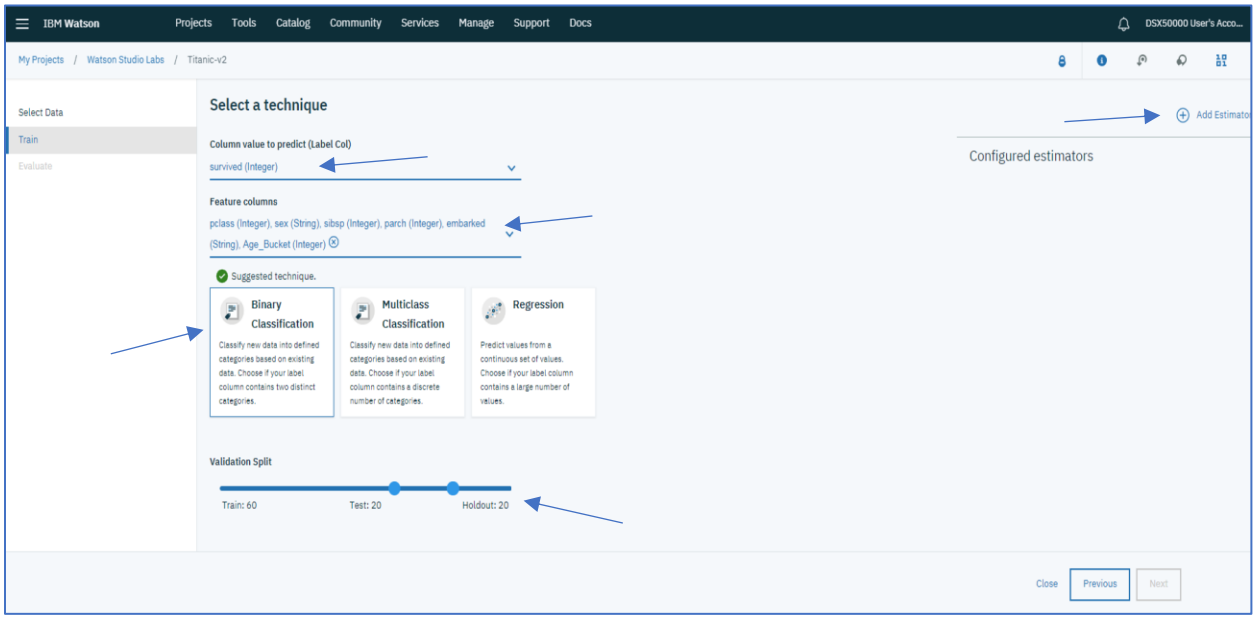
3. Enter a Model **Name** (eg Titanic), optionally a **Description**, leave the default for the **Machine Learning Service** (should be the one created in the pre-reqs), leave **Model Builder** selected, select the **Spark Service** created in the pre-reqs, select **Manual**, and click on **Create**.



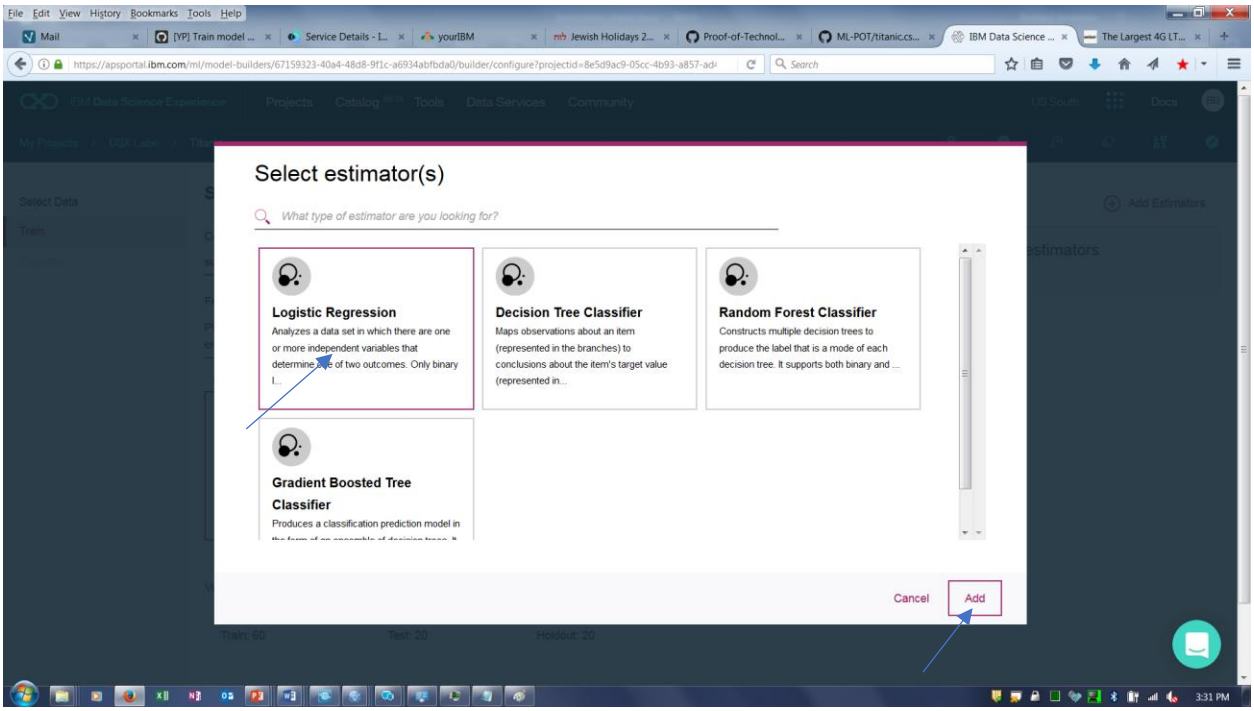
4. Click on the titanic_cleansed**.csv** and click on **Next**



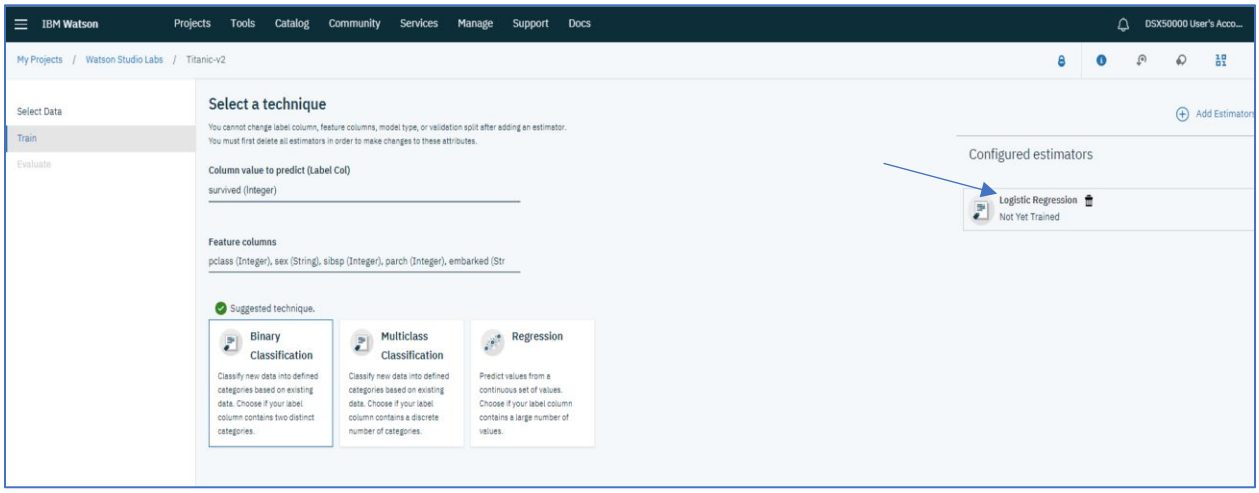
5. For **Column value to predict (Label Col)** select **survived**. For **Feature columns** select the following features (**pclass, sex, sibsp, parch, fare, embarked, and Age_Bucket**) . Click on the **Binary Classification** Box (which is suggested by the service). Adjust the **Validation Split** as desired. Click on **Add Estimators** to add the specific models to use.



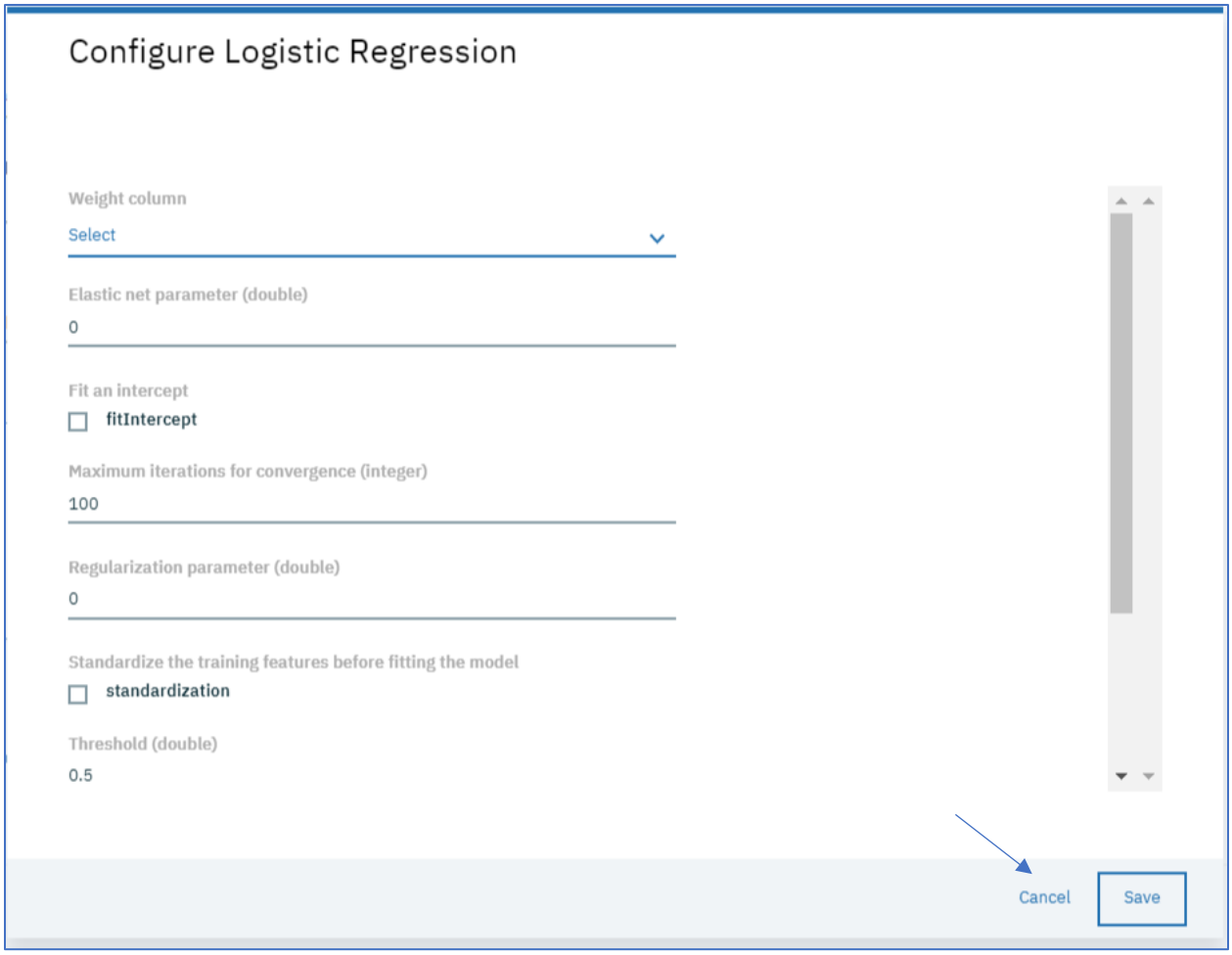
6. Select **Logistic Regression**. You can select more if you wish to see the results of multiple models. Select **Add**.



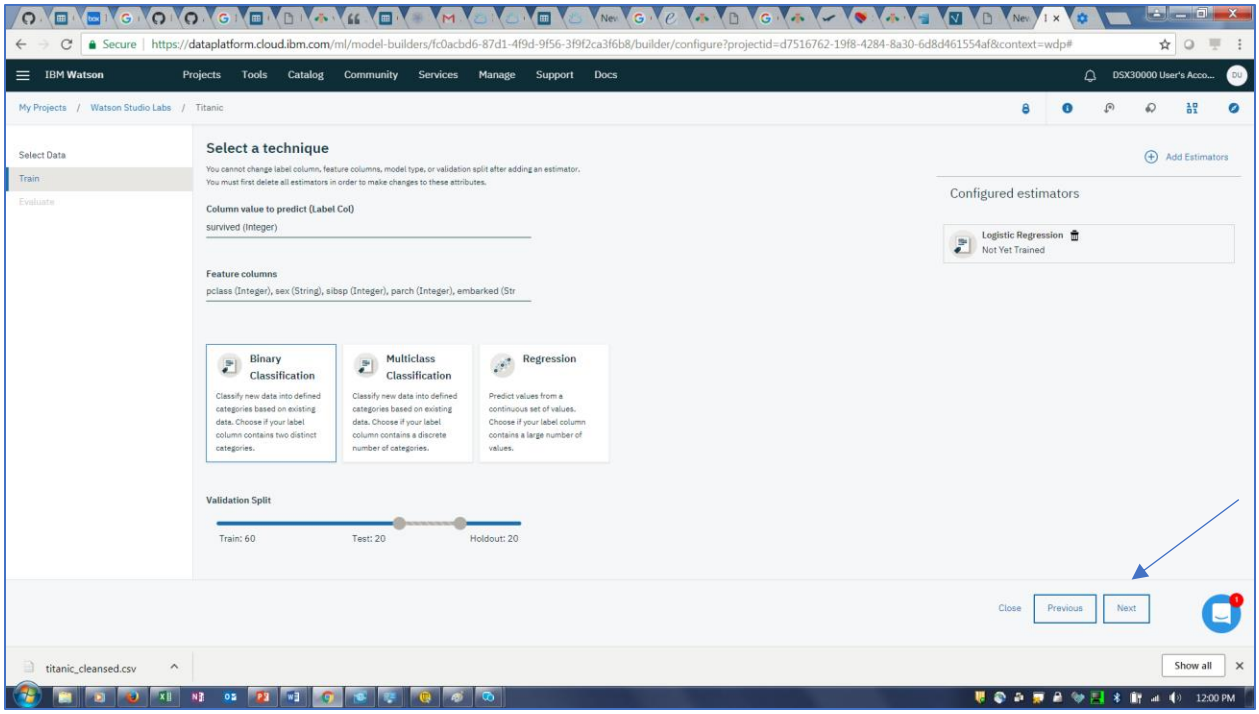
7. Note you can adjust the algorithm's hyperparameters by clicking on Logistic Regression.



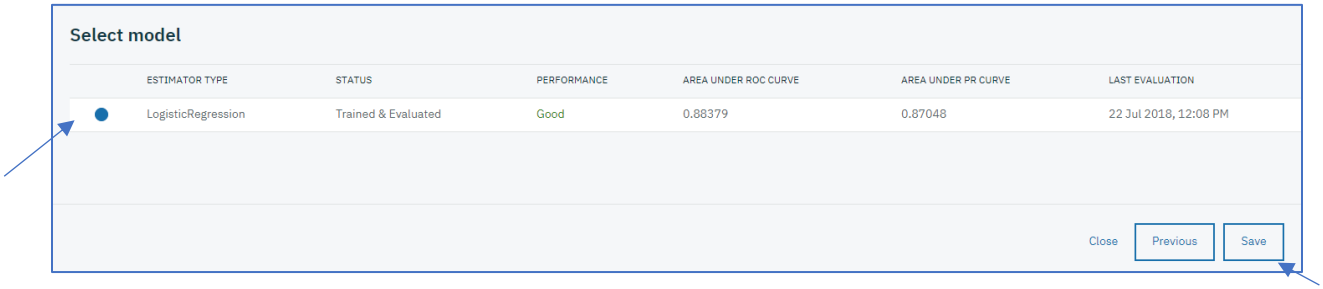
8. For now we will leave the hyperparameters unchanged, so click **Cancel**.



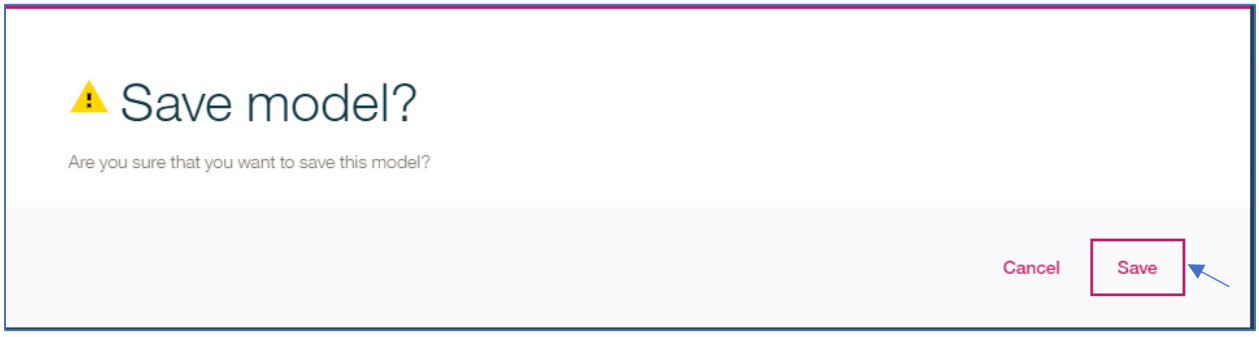
9. Select the **Next** button.



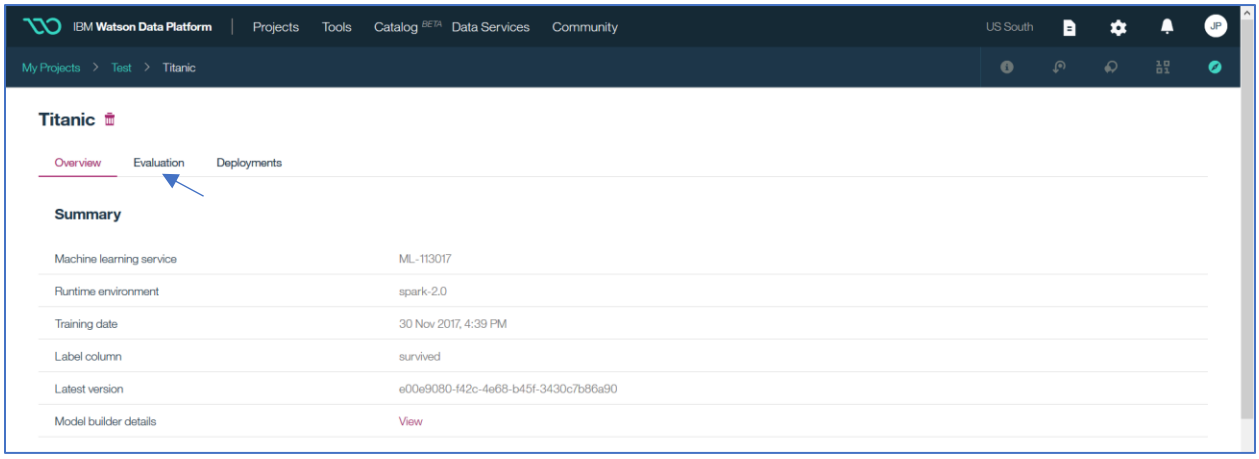
10. The system trains and evaluates each model. If more than one model was selected, the models would be listed in descending order of quality with the best result at the top. Note: if a model fails to run (rare, but happens), select Previous, delete the estimator, and re-add it. Then run again. Click on **Logistic Regression** (if it is the best) and then click **Save**.



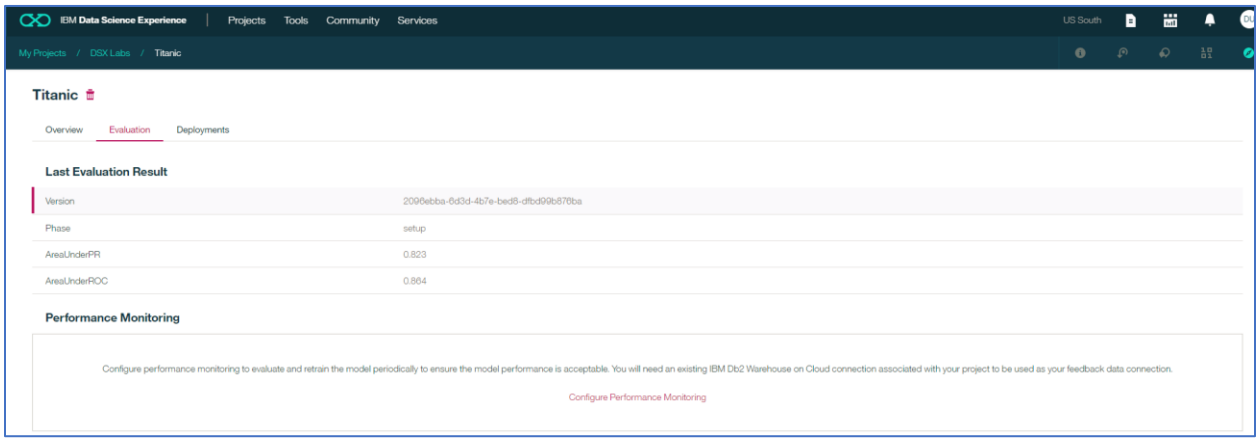
11. Click **Save** again on the next screen.



12. The system displays the model training summary. Click on Evaluation.



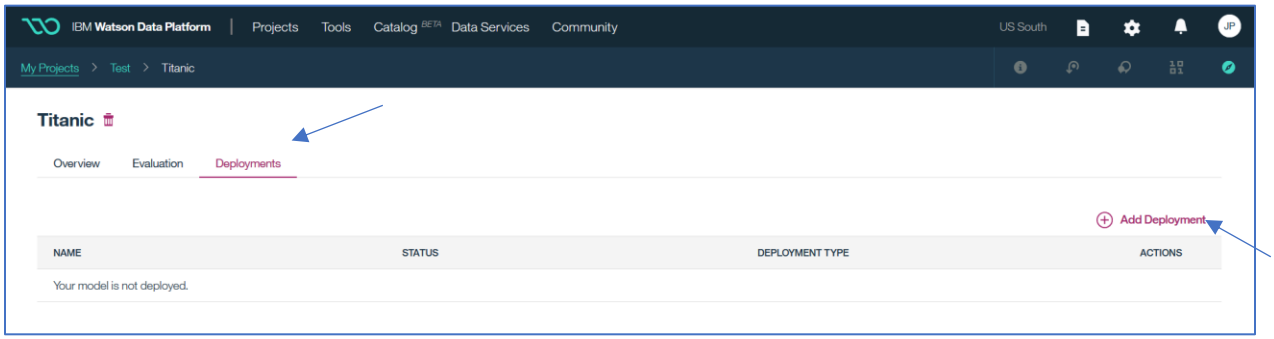
13. The system displays the recorded evaluation statistics for the run. You can also set up Continuous Learning (Performance Monitoring) on this screen. We will not do this now.



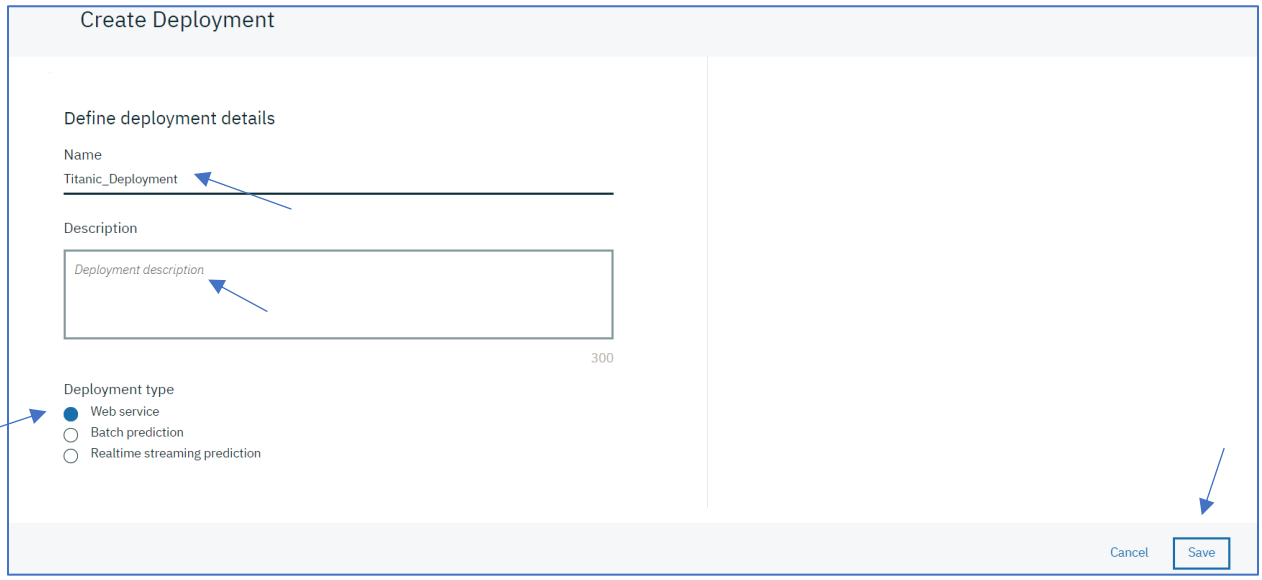
Step 3: Deploying a Model

We can deploy the model to enable applications to invoke it via an API call. This is a called a Web Service deployment or Online deployment.

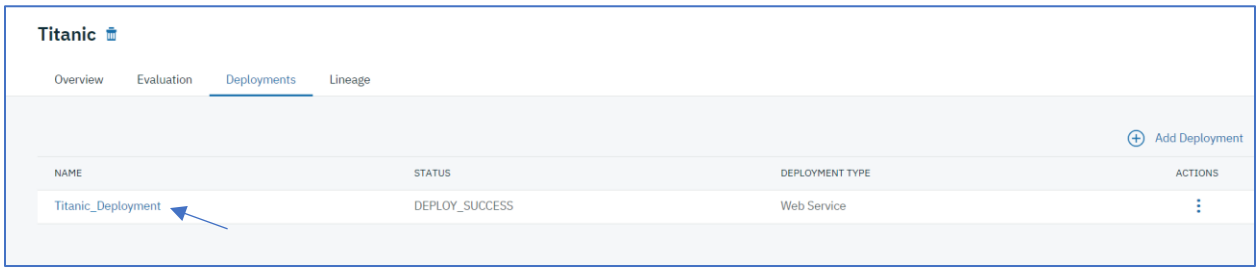
1. Select the **Deployments** Tab
2. Click on **Add Deployment**



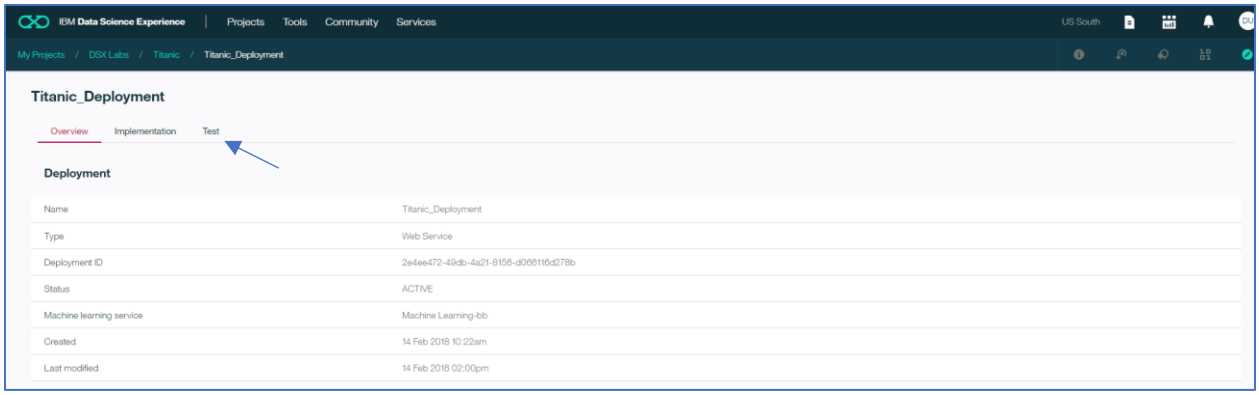
3. Three options for deployment are available. For this lab, we are going to embed the predictive model scoring in a web application. Enter Titanic_Deployment for **Name**, optionally a **Description**, select the **Web Service** radio button, and click on **Save**.



4. The system responds with an acknowledgement that the model was successfully deployed. Click on **Titanic_Deployment** to test the deployed API.



5. The system displays information about the deployed service. Click on **Test** to test out the API.

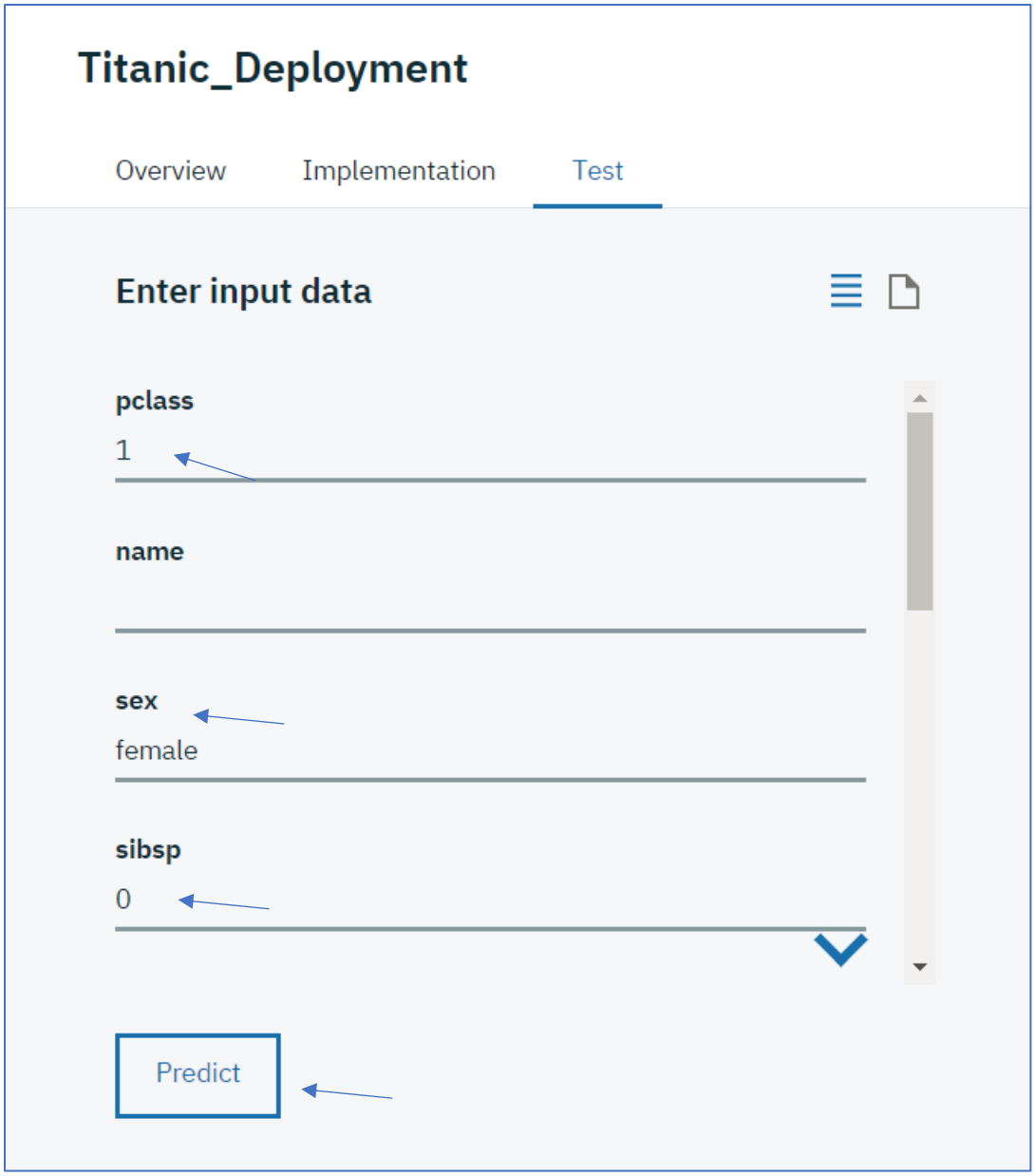


6. Enter values for the following fields:
   pclass - 1
   sex – female
   sibsp (number of siblings or spouses) – 0

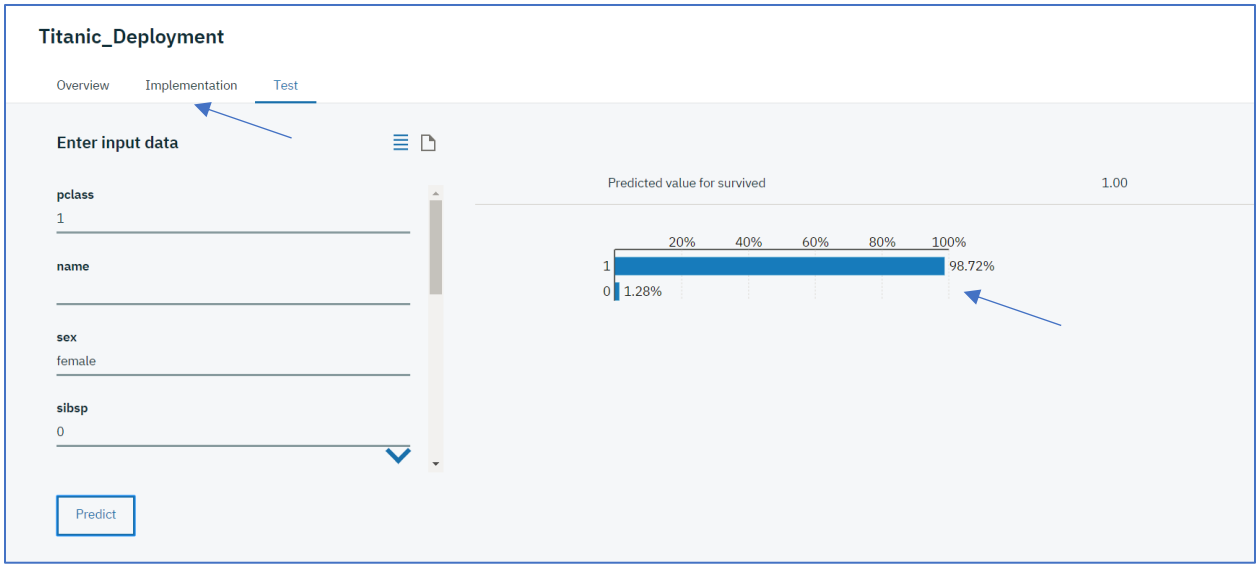
   Scroll down continue entering values for:

   parch (number of parents or children) – 0
   fare - 200
   embarked – S
   Age_Bucket - 1

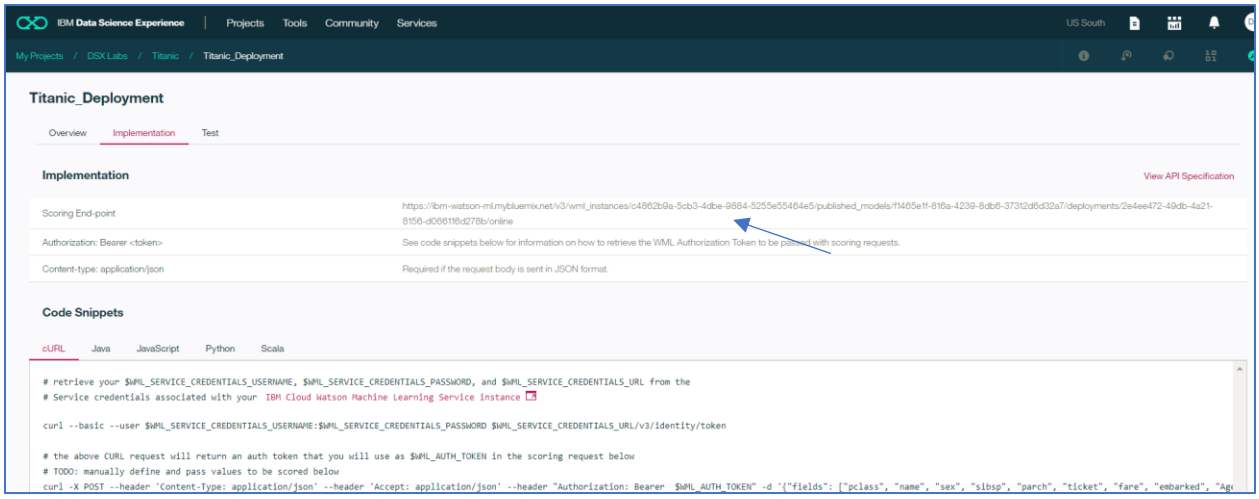
   and then click on **Predict**. Note that any values inputted for any of the fields not included in the model parameters (e.g. name) will not affect the prediction.



7. The predicted result should be returned which indicates that the model has been successfully deployed. Now click on the **Implementation** tab.



8. The Implementation panel provides information for the application developers to invoke the deployed model. It includes sample code in various programming languages and the scoring endpoint to be used when invoking the web service. Open Windows Notepad to copy and paste the scoring endpoint. We will be using this endpoint in Step 4.



Step 4: Deploy a simple web front-end to invoke the Watson Machine Learning service

This section will provide an example of a simple Python Flask web front-end application that invokes the Titanic scoring API demonstrating embedding machine learning in a web app. You

will click on a link below that will deploy the sample Python web application into your IBM Cloud account. A toolchain will be set up for continuous delivery of the application. The application code will be cloned from a public Git repository into a private Git repo in your account that will be set up as part of the toolchain. Each time you commit changes to the repo, the app will be built and deployed.

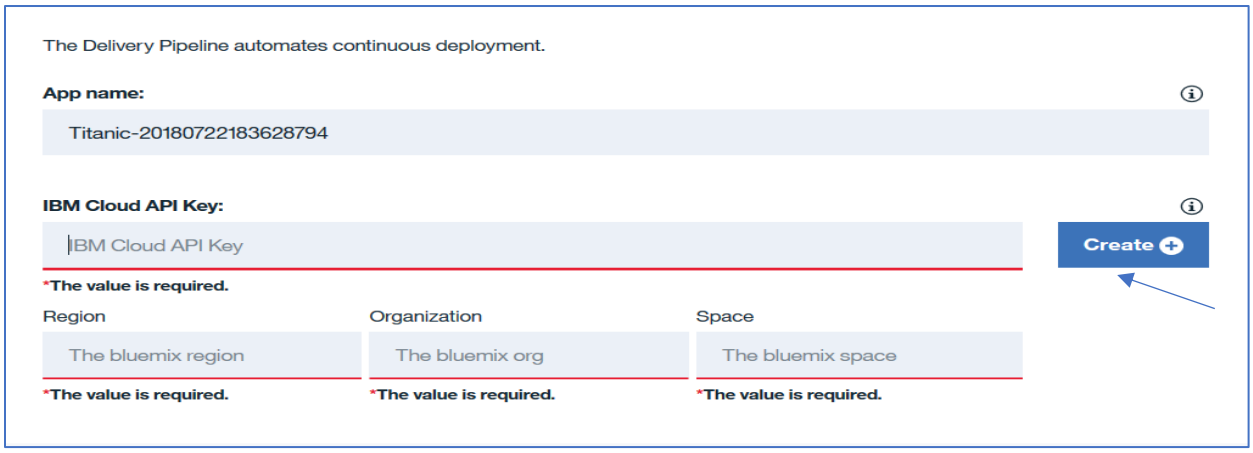
The toolchain uses tools that are part of the Continuous Delivery service. If an instance of that service isn't already in your account, when you click **Deploy**, it is automatically added with the free Lite plan selected.

You will need to configure the application to provide the credentials for your Watson Machine Learning service, and to provide the scoring endpoint.

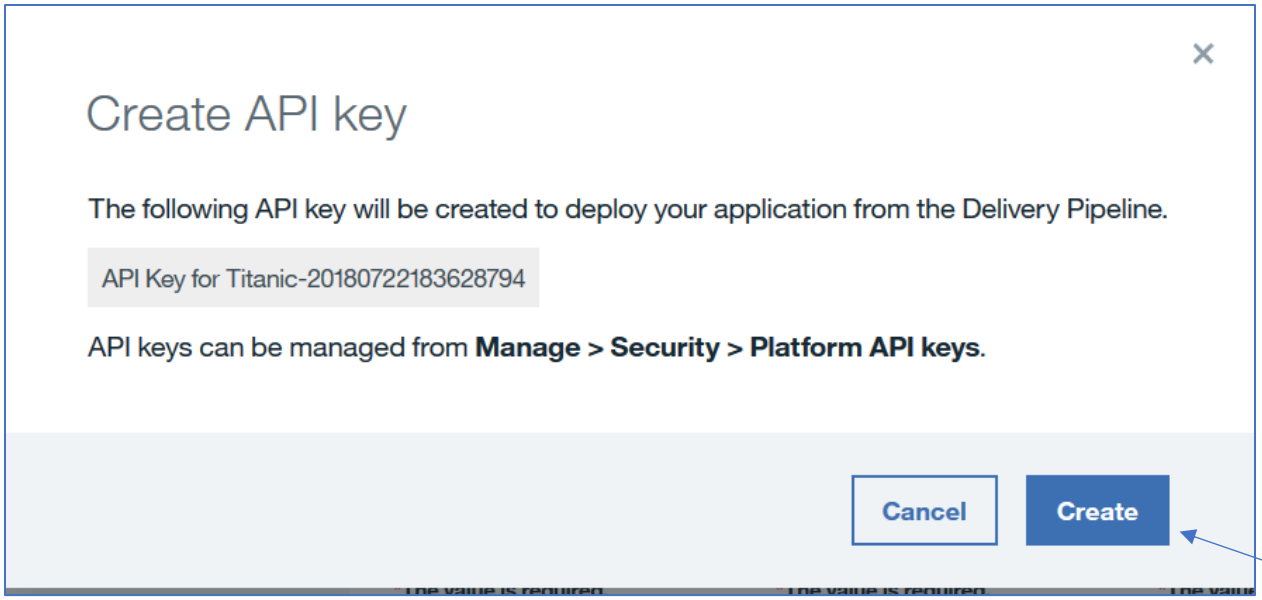
1. Click on the **Deploy to Bluemix** link below to deploy a sample Python Flash web application into your IBM Cloud account. Note you may get this message – "*An IBM Cloud account is required. To get started, click Log In or Sign Up at the top of this page*". If you get this message, click on **Log In.**

   Deploy to Bluemix

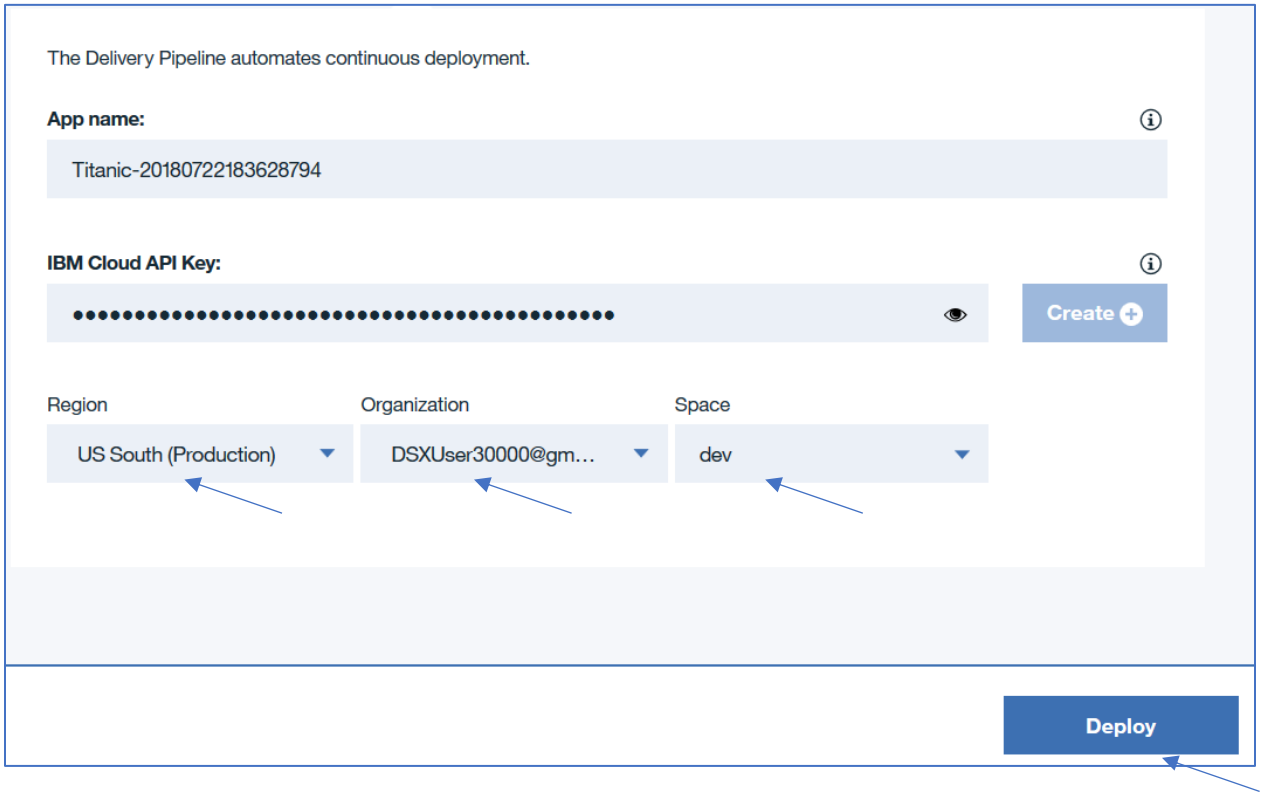
2. Scroll down to the bottom. Click on the **Create+** button to create an IBM Cloud API key.



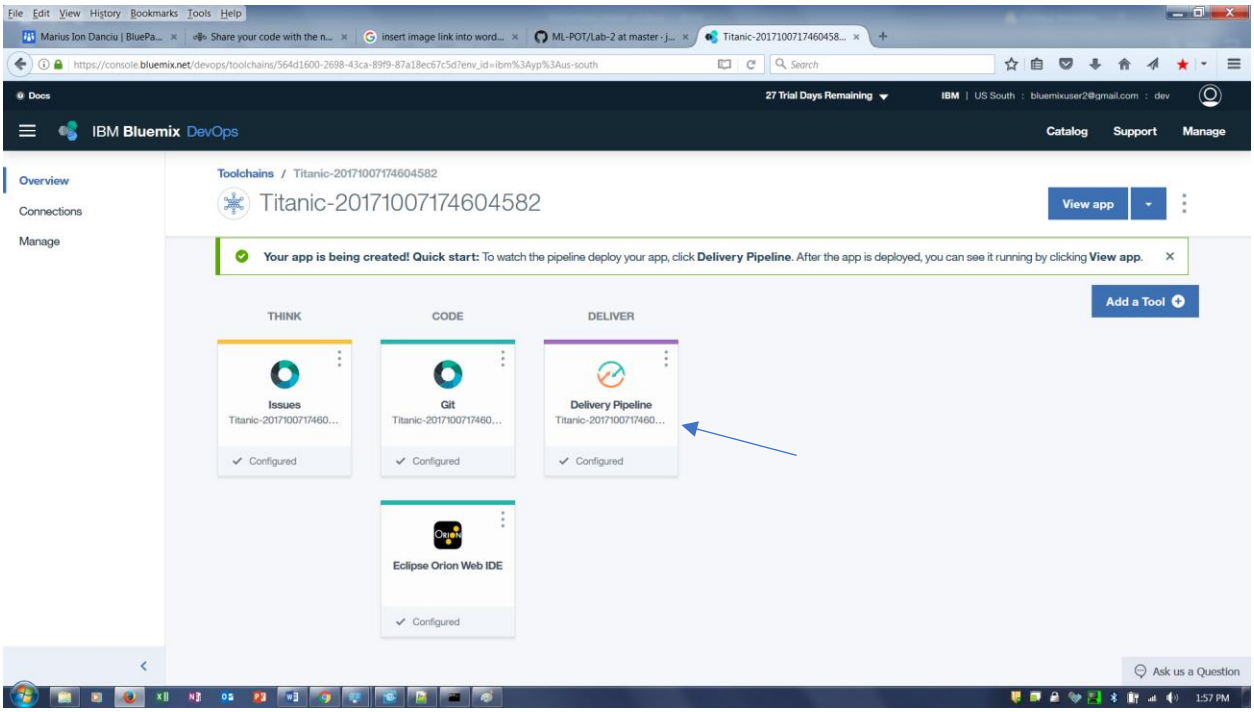
3. Click on the **Create** button.



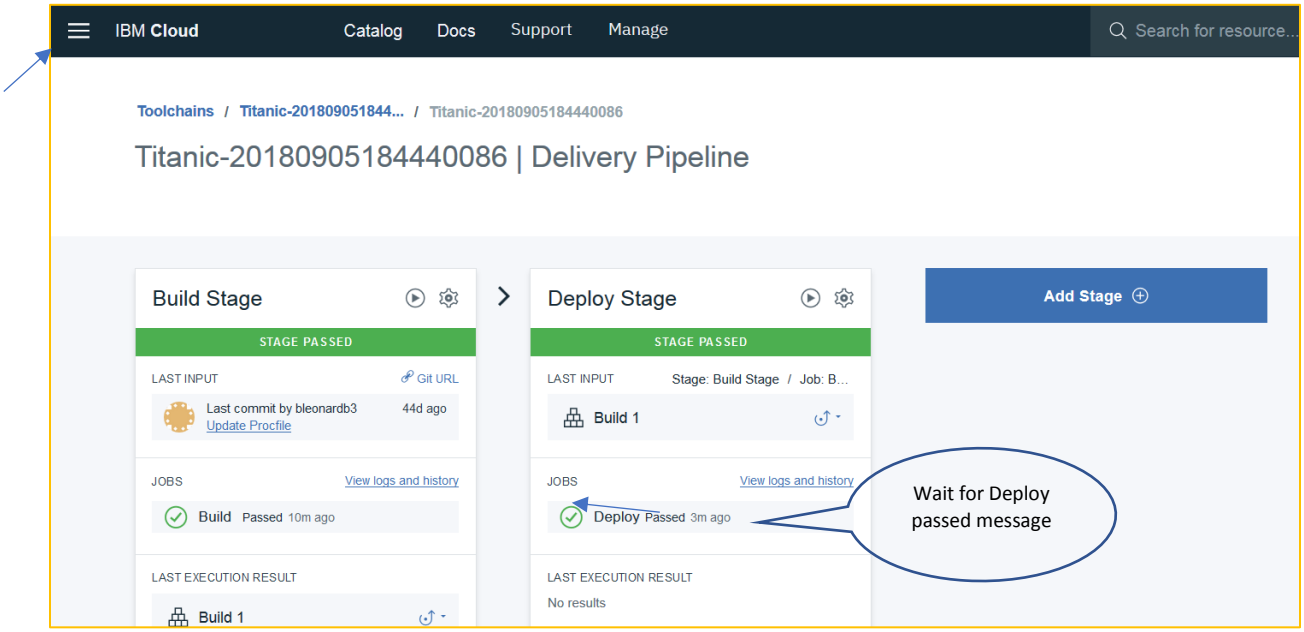
4. Please wait until the Region, Organization, and Space are filled in. **Note that these must match the corresponding Region, Organization, and Space where the Titanic model was deployed.** Unless you changed the default space (dev) on creating your Watson Studio account, they should match. Otherwise update the Space value, if needed (if you have multiple spaces defined you will see them listed when clicking on the down arrow in the Space field). Click on the **Deploy** button

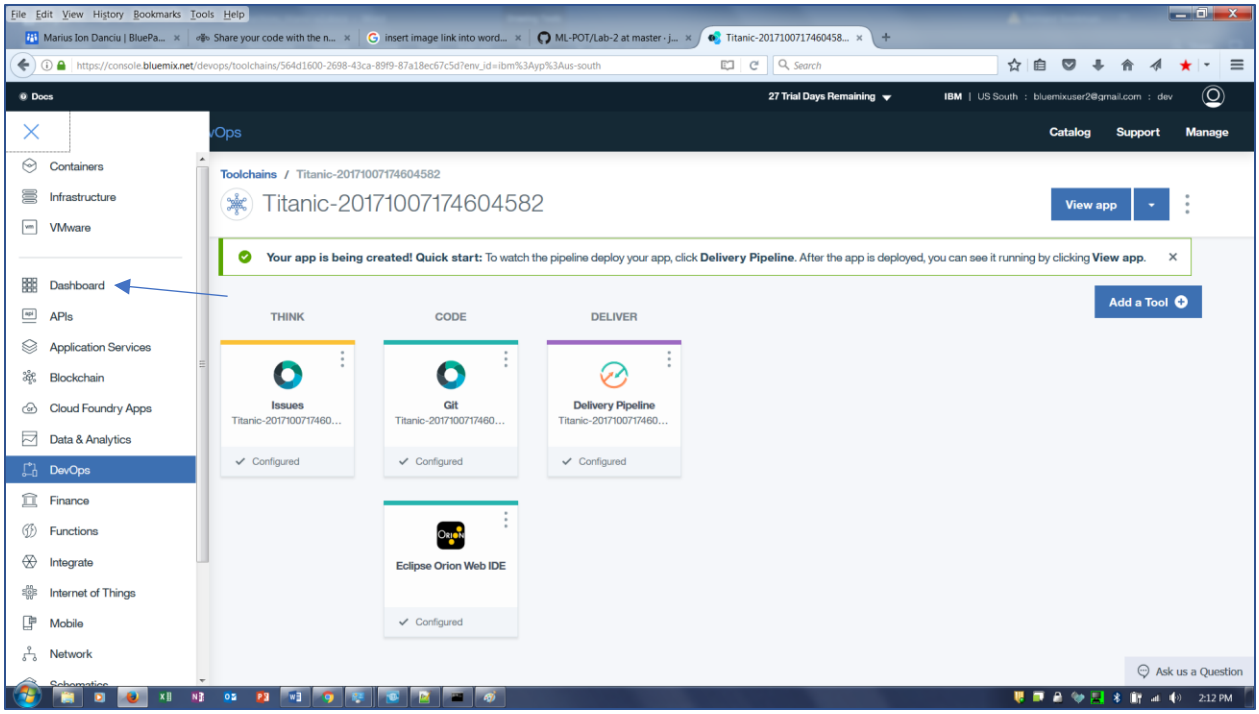


5. Your app is being created! To watch the pipeline deploy your app, click **Delivery Pipeline**.

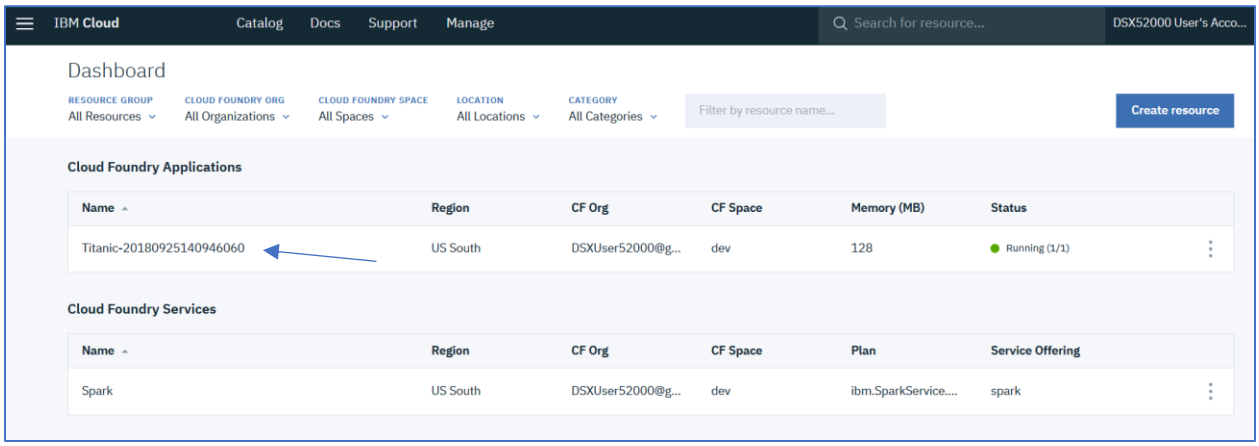


6. After the app is deployed successfully (should say Deploy Passed in the Deploy stage- may take about 2 minutes), view the running app by clicking the hamburger ☰ icon in the left-hand corner and **Dashboard** in the pulldown to navigate to the **Dashboard** where the running application should be listed.

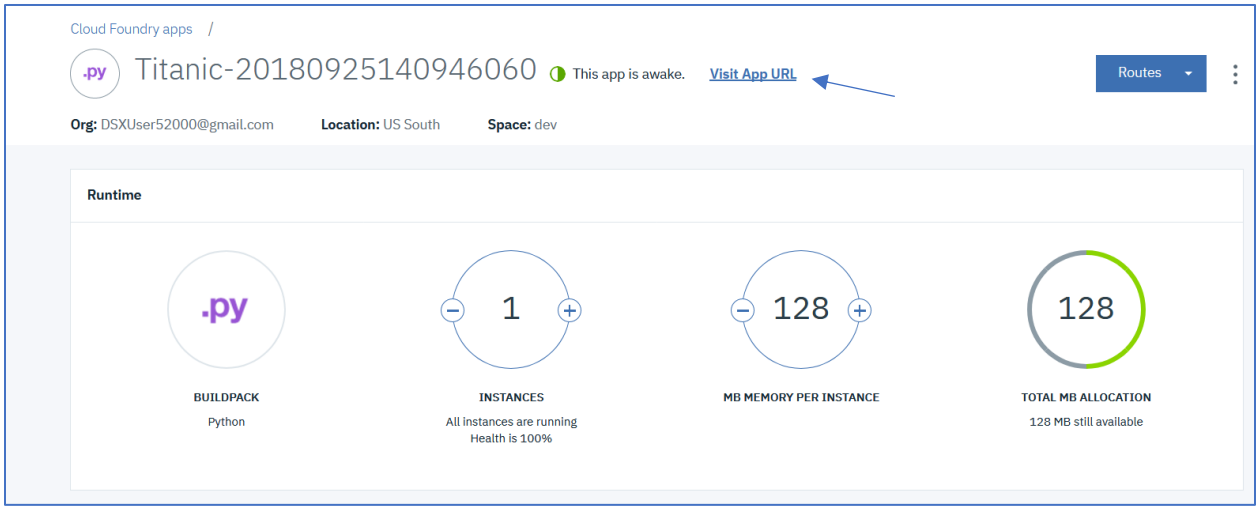




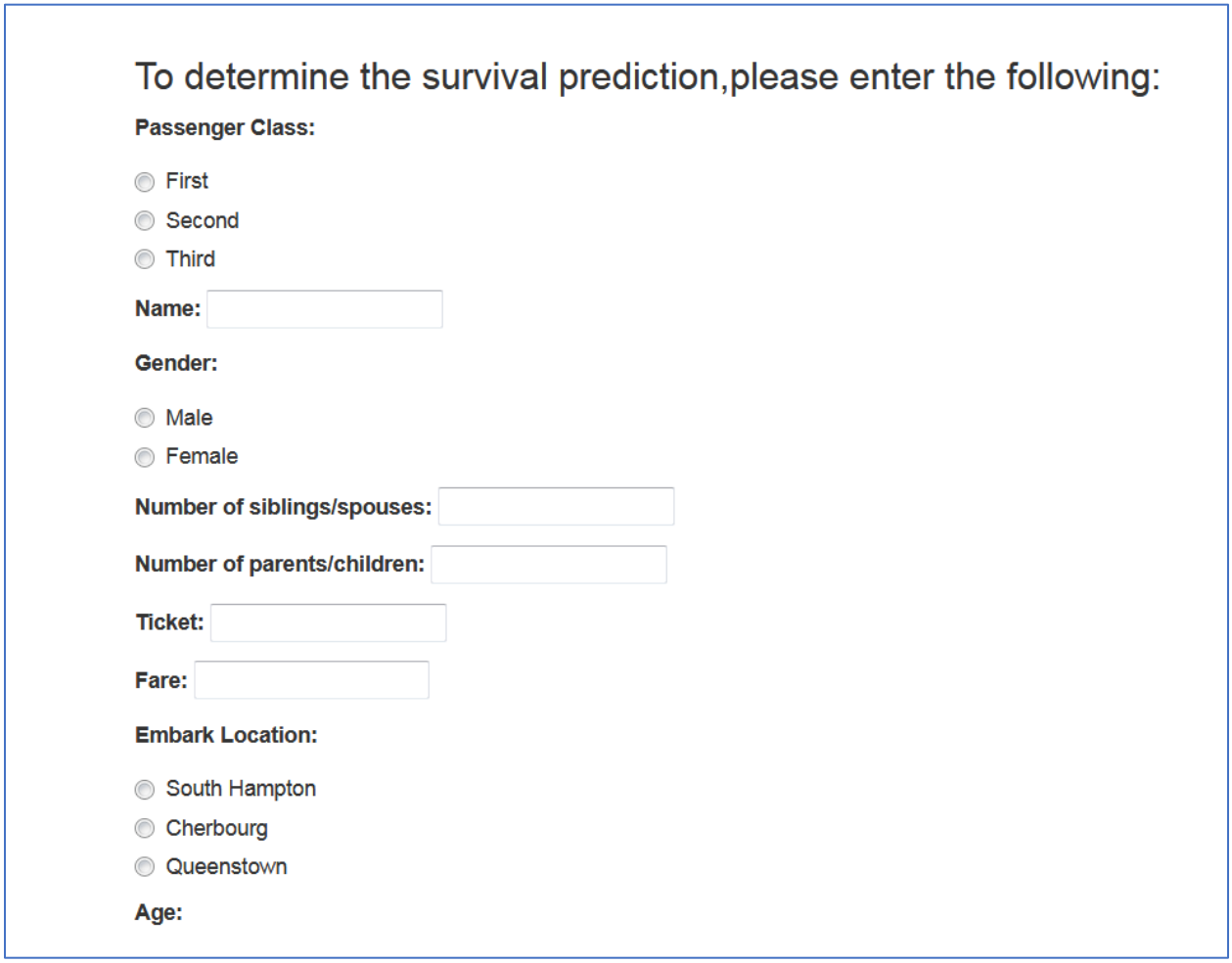
7. Click on the Titanic application name.



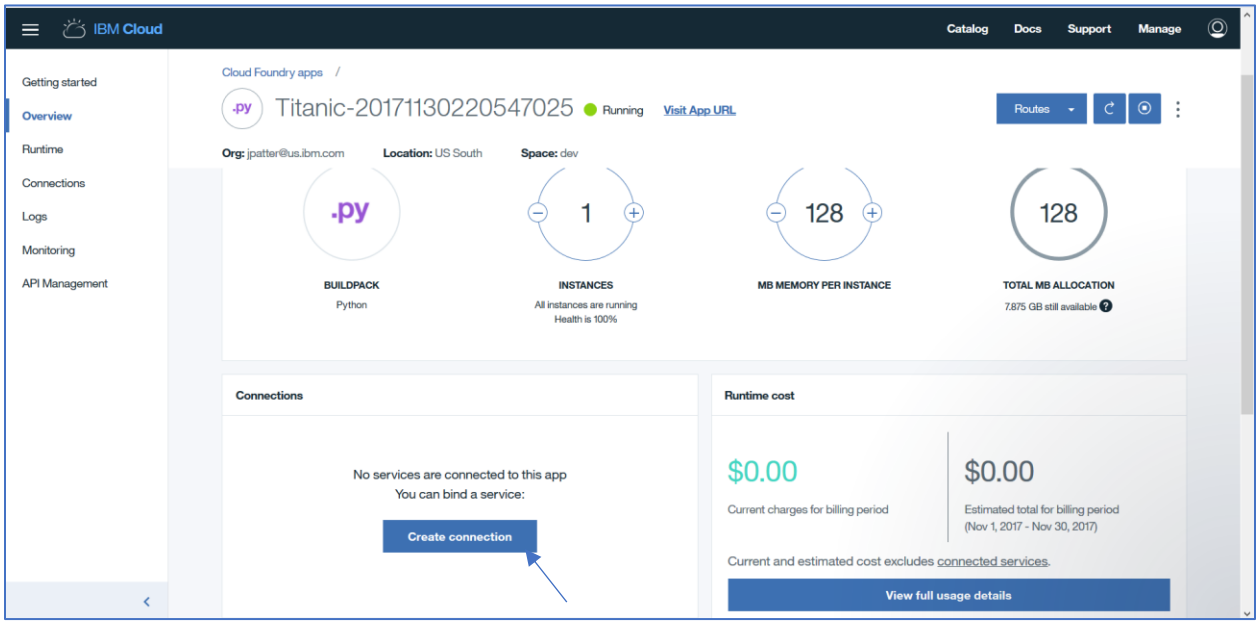
8. Click on **Visit App URL**



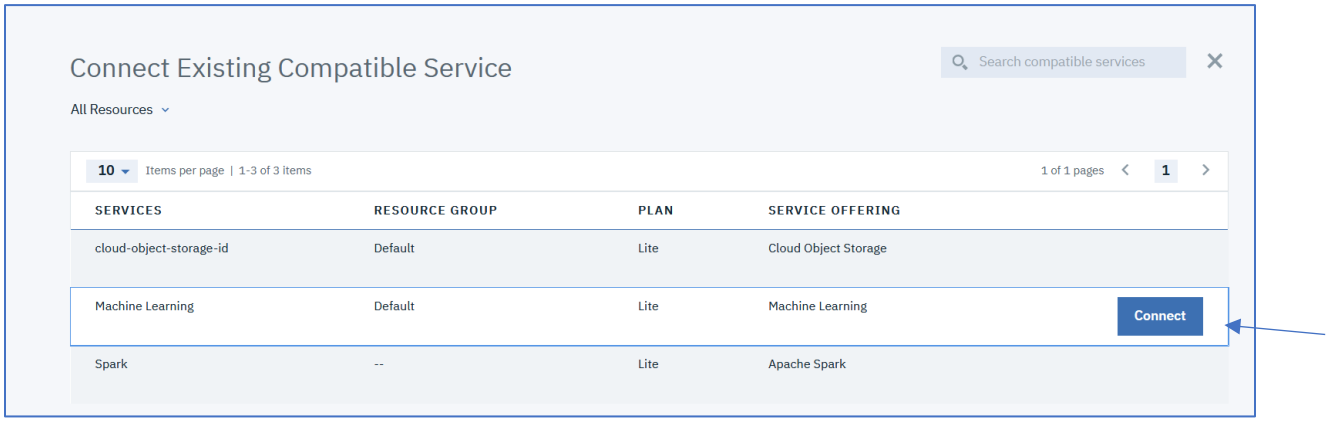
9. The web form collecting the Titanic passenger data should appear. Note that the application is not functional until we connect it to the Watson Machine Learning service so if you Submit you will get an error! Close the Titanic Prediction browser tab.



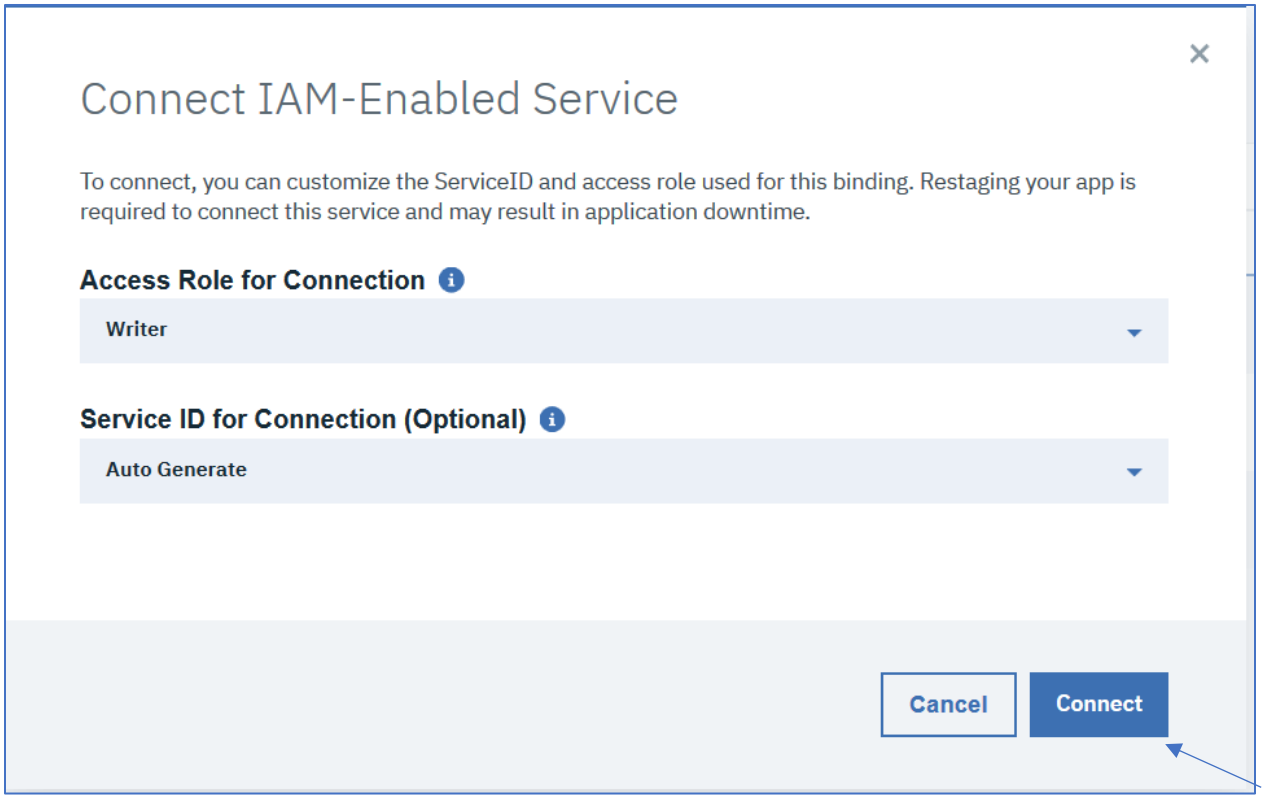
10. We are now going to connect the application to the Watson Machine Learning service that was created earlier. Scroll down until you see the Connections panel. Click on **Create Connection**.



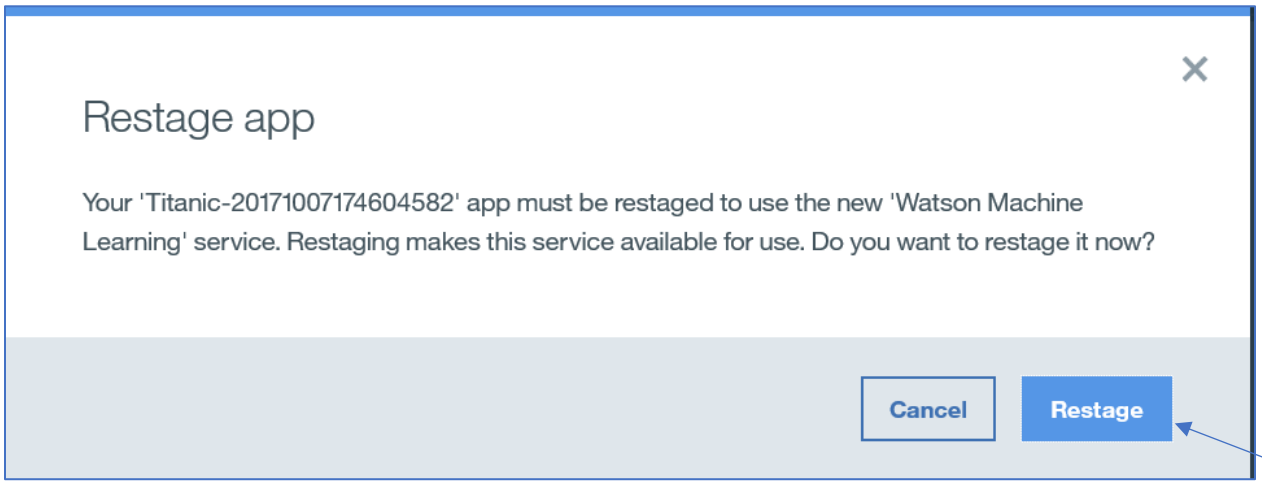
11. You should see at least 3 services listed, a Cloud Object Storage service, a Spark service, and a Watson Machine Learning service. Point the cursor on the **Machine Learning** service for your application, and then click on **Connect**.

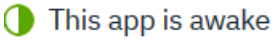


12. A **Connect IAM-enabled service** pop up will appear. Just click **Connect**.



13. A **Restage app** pop up will appear. Click on **Restage**.



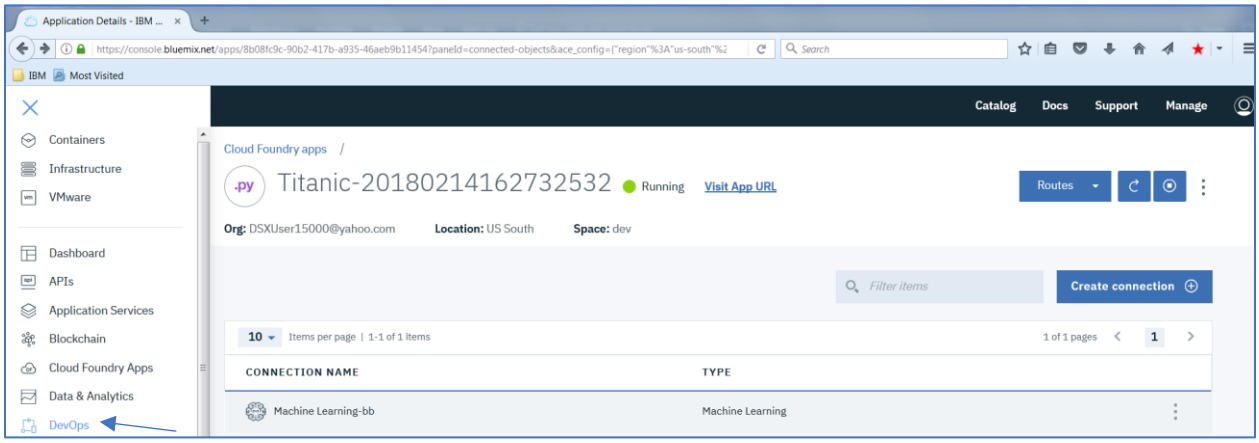
14. Wait for the application status to change to 

, or something similar.

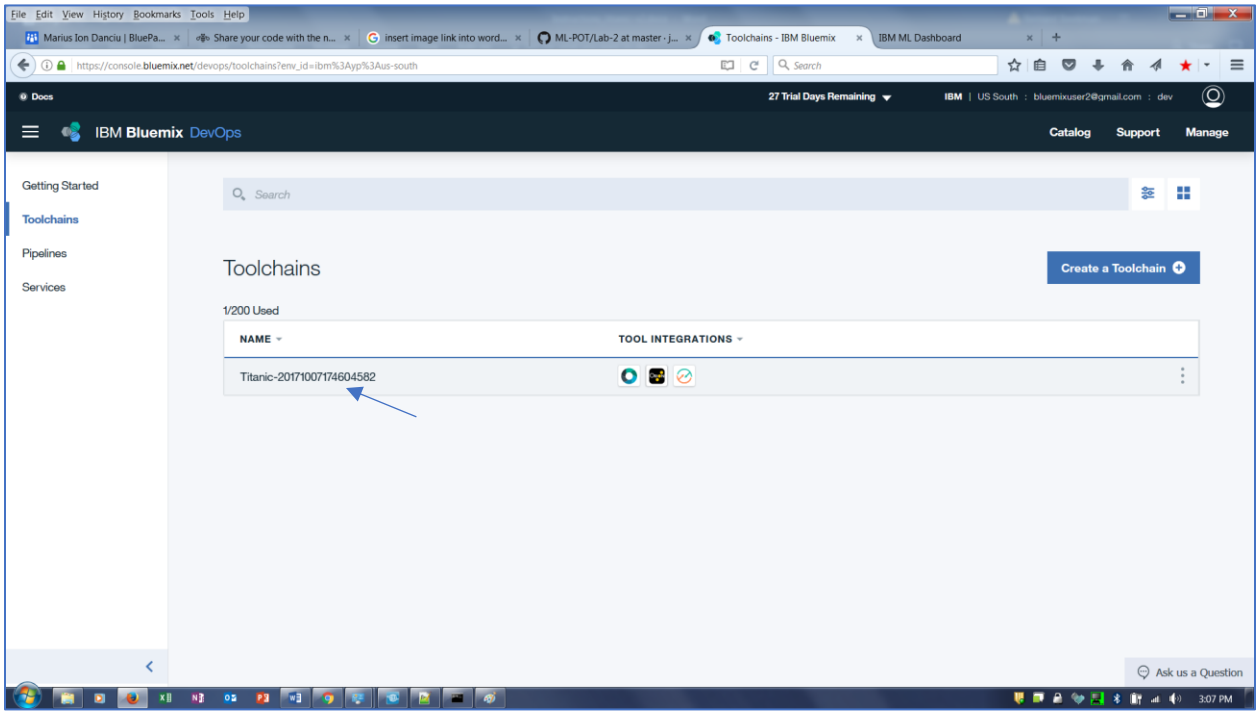
15. We now have tied the web application to the Watson Machine Learning service. Note that the Watson Machine Learning service could have more than one deployed model available to select and then embed in the web application. In our case, we have only one deployed model. We now need to copy the scoring endpoint of that deployed model

(previously copy and pasted into Notepad) and paste it in the web application code.

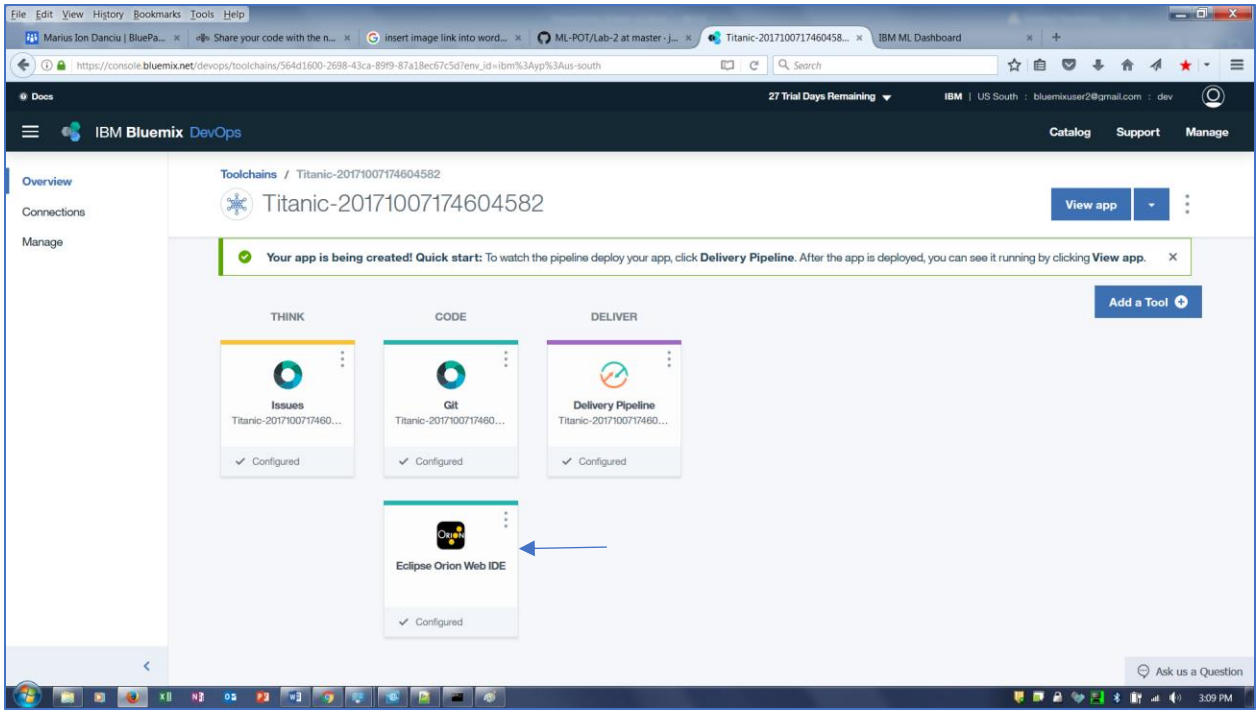
Click on the ☰ icon and click on DevOps in the pulldown to navigate to the Toolchain.



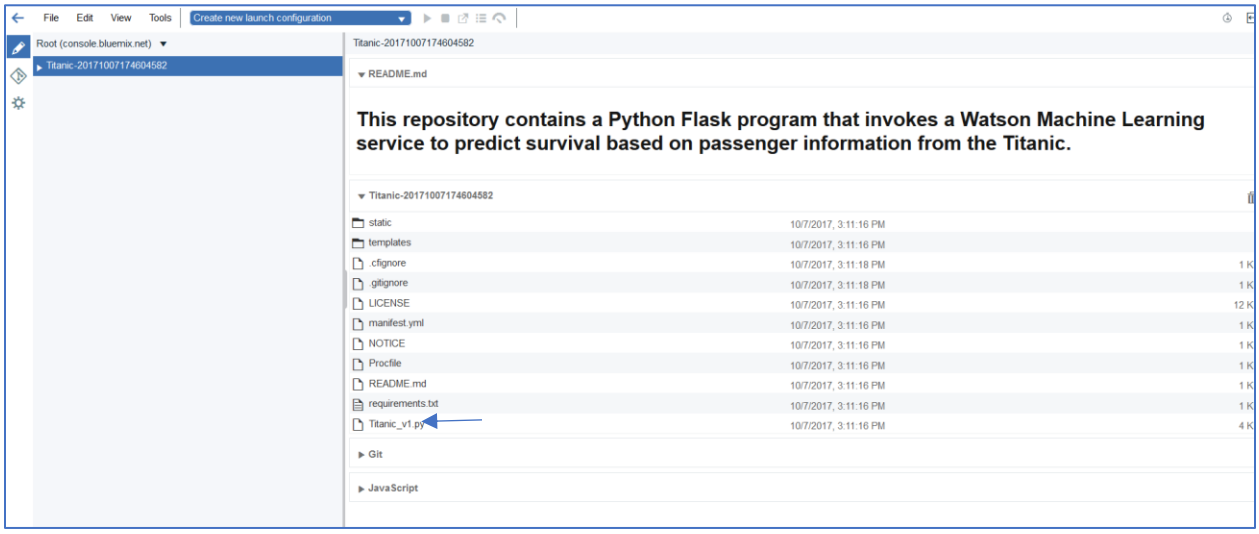
16. We are now going to paste the scoring endpoint into the application code. Click on the Toolchain (Titanic-2018xxxxx below).



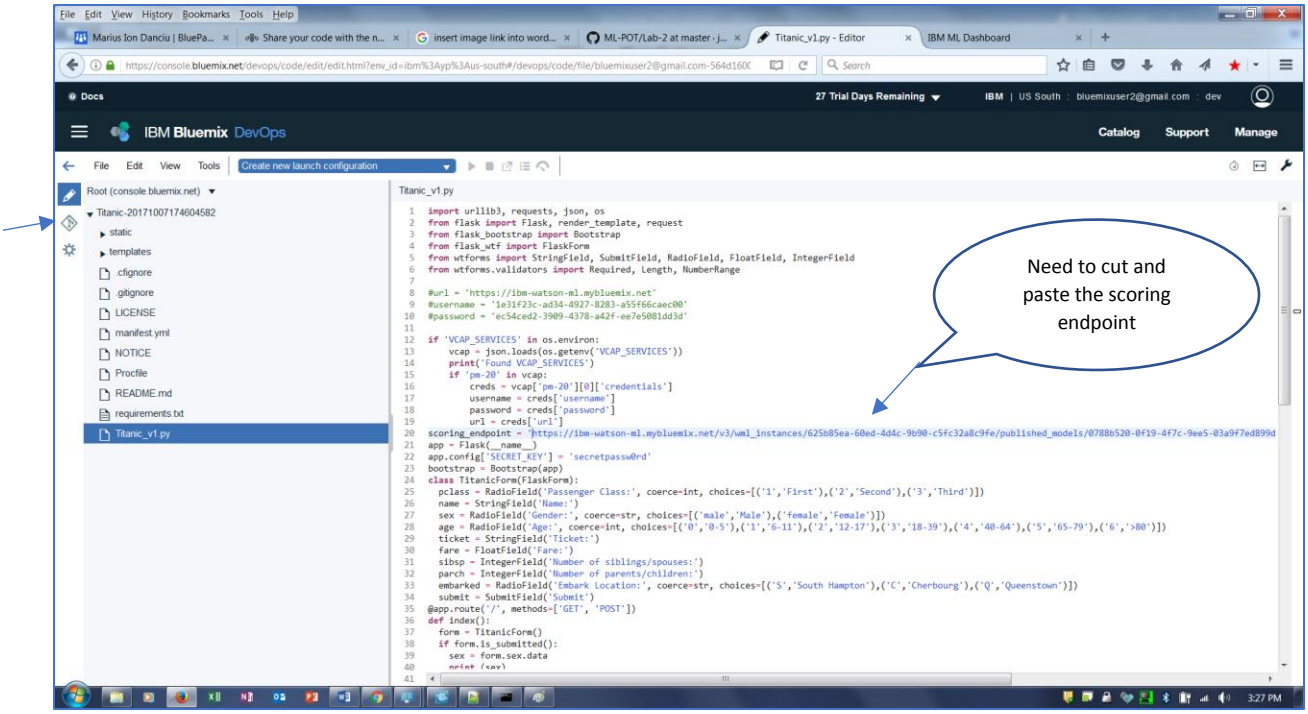
17. Click on the Eclipse Orion Web IDE.



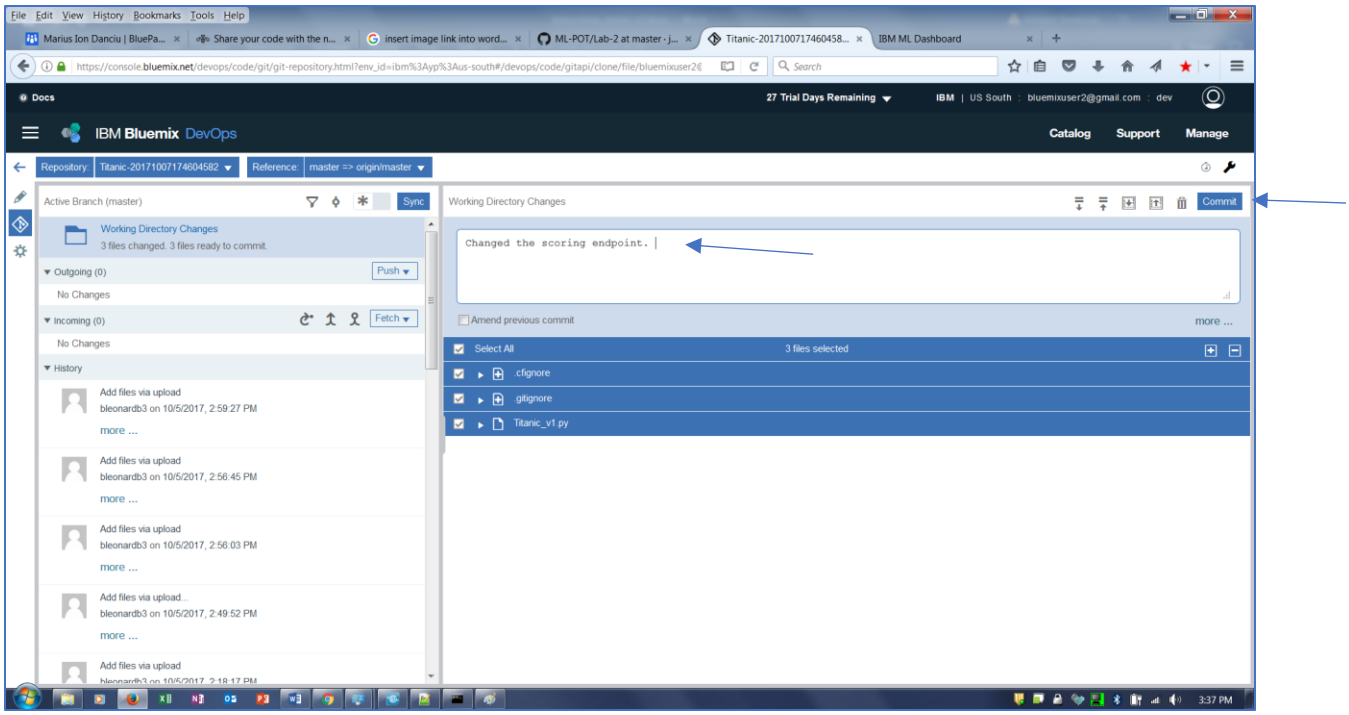
18. Click on the Titanic_v1.py file. This is a python source file.



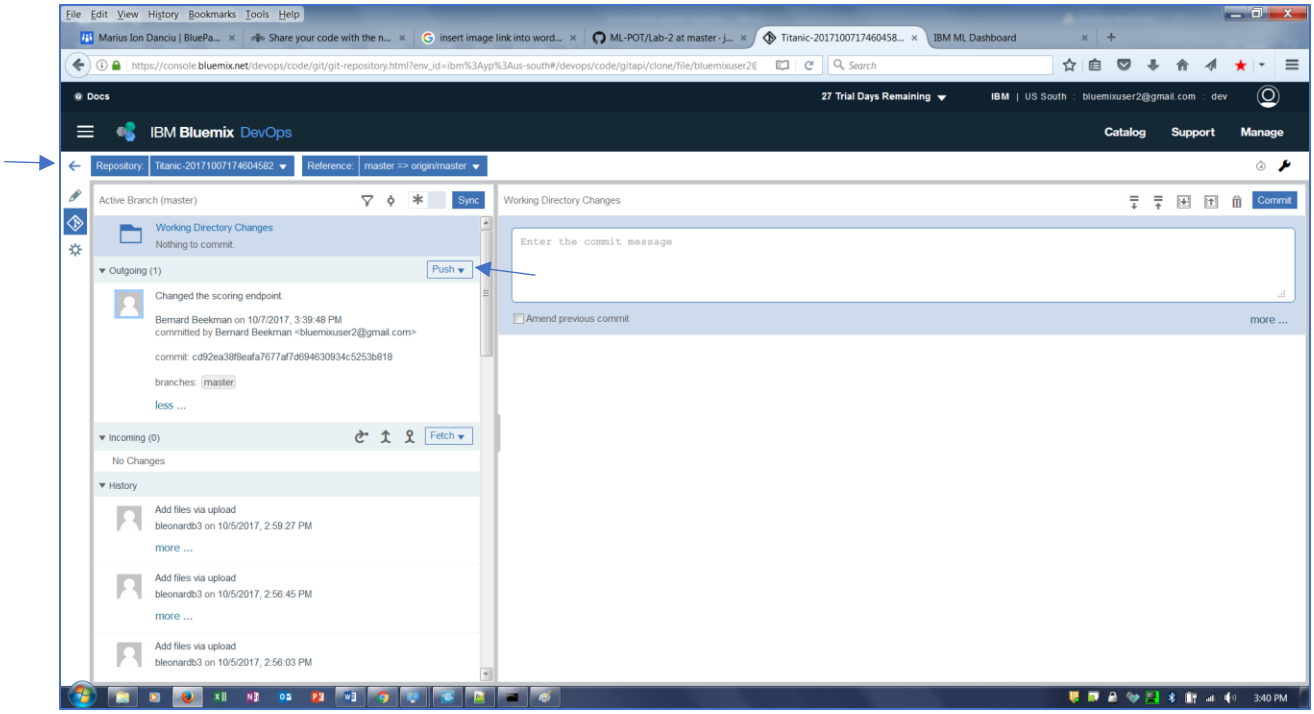
19. Go back to the Notepad file, and copy the scoring endpoint to the clipboard. Look around line 20 in the Titanic_v1.py file for the "scoring endpoint =". Select the scoring endpoint value in line 20 (starting with https:// may want to use Shift-End to get to the end of the line, and then back up one space to not select the endpoint quote – if you do just make sure to put it back in). Enter Ctrl-V to paste the new scoring endpoint from your Notepad file. Enter Ctrl-S or File > Save to save the file. Then click on the  icon on the top left.



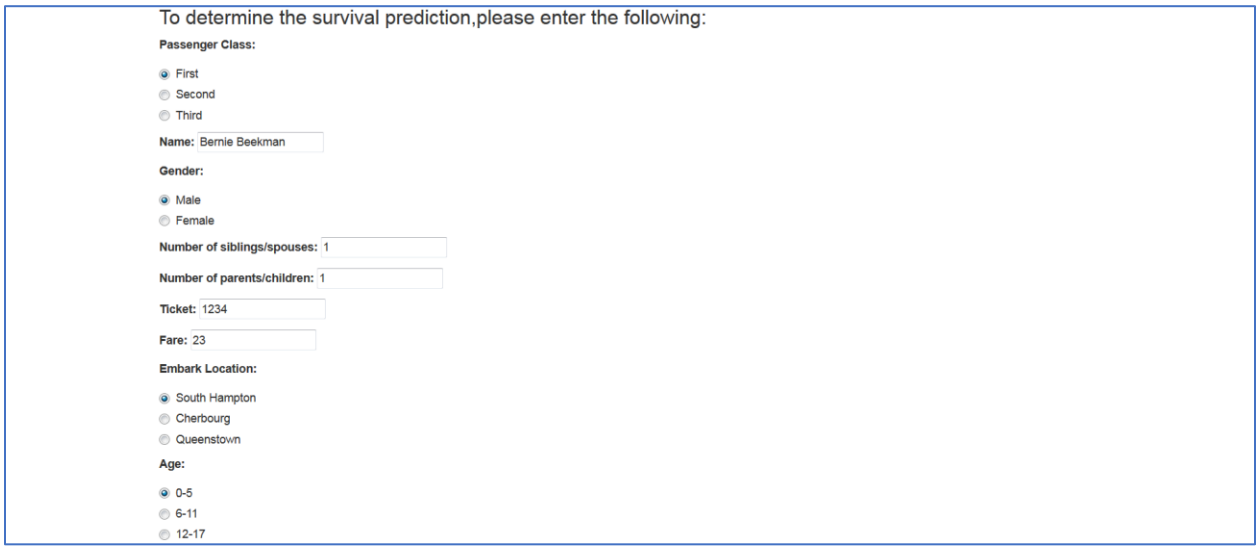
20. The next step is to commit the change to the git repository. Enter "Changed the Scoring Endpoint" in the Enter Commit Message field, and then click on **Commit**.



21. Then click on **Push** to push the changes to the central Git repo which will start the build and deploy of the application. Click on the left arrow to return to the Toolchain.



22. Click on the **Delivery Pipeline** to view status of the deployment as before. Once the Deployment status shows **Deploy passed now** it shouldn't take longer than 2 minutes so reload the browser in case the UI didn't update). You can see the running app by clicking  icon in the left-hand corner and Dashboard in the pulldown to navigate to the **Dashboard** where the running application should be listed. Click on the Titanic Application and then Visit App url. (See steps 6, 7 and 8 above).
23. The web form should appear. Enter data in all the fields and click on the **Submit** button. (the submit button is located at the bottom of the web form – you may need to scroll).



24. You should see something similar to the following depending on the values of the input fields that you entered. Click on the **Try Again!**, if you want to experiment with different inputs.