Contents

[Utilizing InterpretML for Model Explainability 1](#_Toc108787095)

[Step 1: InterpretML : Glassbox models and explainers 2](#_Toc108787096)

[Step 2: Initial Pre-processing of the data using pandas 2](#_Toc108787097)

[Step 3: Using InterpretML’s functions to obtain descriptive stats (visually) 3](#_Toc108787098)

[Step 4: Using Glassbox model from InterpretML 4](#_Toc108787099)

[Step 5: Using global explainers from Glass box models 5](#_Toc108787100)

[Step 6: Using local explainers from Glass box models 5](#_Toc108787101)

[Step 7: Glassbox Model : Performance metrics 6](#_Toc108787102)

[Step 8: Glassbox Model : Training other models 6](#_Toc108787103)

[Step 9: Glassbox Model : Generating global and local explanations for all models 7](#_Toc108787104)

[Step 10: Dashboard view for all models 8](#_Toc108787105)

[Step 11 : Blackbox explainers using sklearn model 9](#_Toc108787106)

[Step 12 : Training blackbox model: 9](#_Toc108787107)

[Step 13: Generating model performance from Blackbox model using interpret library 10](#_Toc108787108)

[Step 14: LIME for local explanations on blackbox model 10](#_Toc108787109)

[Step 15: SHAP for local explanations on blackbox model 11](#_Toc108787110)

[Step 16: Global explanations: Morris sensitivity on blackbox model 12](#_Toc108787111)

[Step 17: Global explanations: Partial Dependence plots on blackbox model 13](#_Toc108787112)

[Step 18: Comparing all explainer Dashboard: Blackbox model 13](#_Toc108787113)

# Utilizing InterpretML for Model Explainability

**Below is an excerpt** that I had sent earlier to summarize information on InterpretML. Later in this document I would try to elaborate each section with screenshots to make it easier to understand

“Here are few findings around the same :-

1. **InterpretML** is a simpler library as compared to **Alibi** and easy to understand
2. **InterpretML** is managed by **Microsoft Research team** and has active collaboration with the **developer community of SHAP and LIME**
3. **InterpretML** comes inbuilt with **Azure SDK and** also has a **text version for NLP.**
4. Here are few additional details around InterpretML :-

**A.      InterpretML** supports : **Glassbox Models** and **Blackbox Explainers**

**B.      Glassbox models** - models that are meant to be interpretable.

·       Thus there are 4 models which are created inhouse as part of IntepretML : Explainable Boosting Machines, Decision Tree, Decision Rule List, Linear/Logistic Regression

·       **EBM : explainable as Linear models, but accurate as RF or any ensemble models**

·       Basic idea behind building these models is to make general additive models more explainable even with Boosting (explainable boosting)

·       These also provide both Local as well as Global explainability.

·       This doesn’t  work for a multi class classification problem

**C.      Blackbox explanations** : These are explanations that can be given for any models coming from any other library such as sklearn

a.       They consider **only input and output values** and assume that based on these values they need to **identify the explainability.**

b.       In this process, changes are made to inputs and passed through model to analyse the change in model output and thus provide **explainability.**

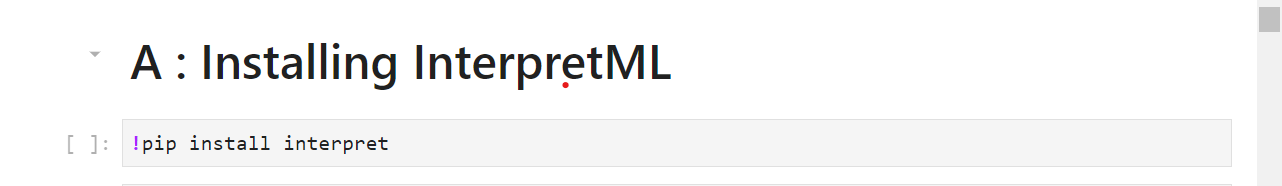
c.       **Local explainers** in blackbox explanations include:**SHAP, LIME**

d.       **Global explainers** in Blackbox explanations include : **Partial Dependency Plot** and **Morris Sensitivity.**

e.       This can work very well on deep neural nets or on complex ml pipelines

I have used the loan dataset shared earlier by Ankita and implemented both Glassbox model and blackbox explainers. “

# Step 1: InterpretML : Glassbox models and explainers



**Dataset** being used :- <https://www.kaggle.com/arashnic/banking-loan-prediction>

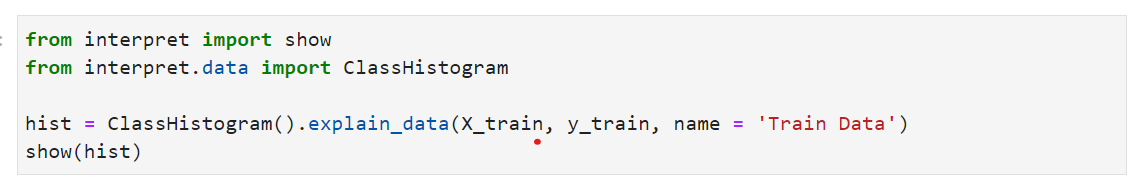
* **Although this was a multi class classification dataset, but InterpretML currently only supports binary classification.**
* **Hence the encoding for target variable will be made to consider only 2 classes**

## Step 2: Initial Pre-processing of the data using pandas

* 1. Understanding the data using descriptive statistics
  2. Looking at the null values present in the data
  3. Treating the null values based on frequency presence in the data
  4. Label encoding the target variable based on Step 1 mentioned.

## Step 3: Using InterpretML’s functions to obtain descriptive stats (visually)

* Code:



* **Visualization:**  by selecting features from drop down:

We can select different features from the dropdown generated and look at the distribution of target binary classes across these continuous as well as categorical features

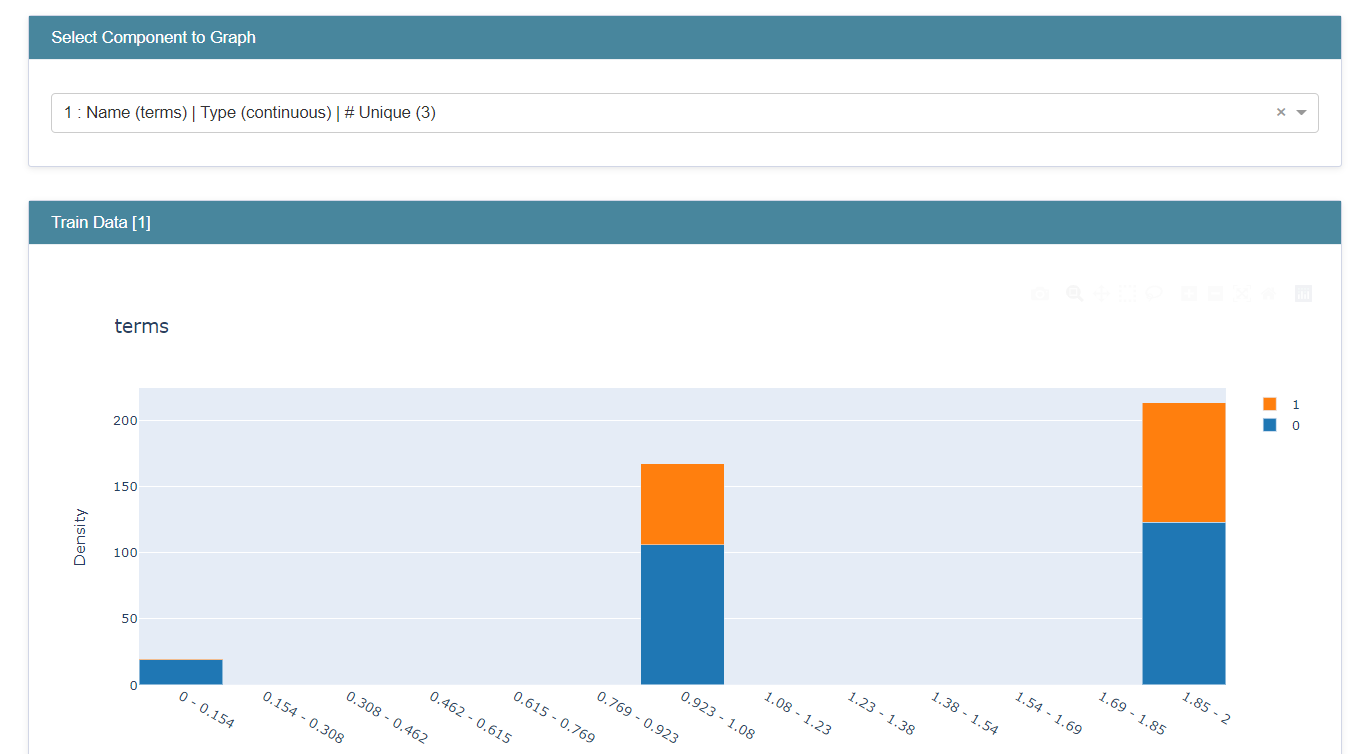


Figure : Distribution of Term (continuous column vs target variable)

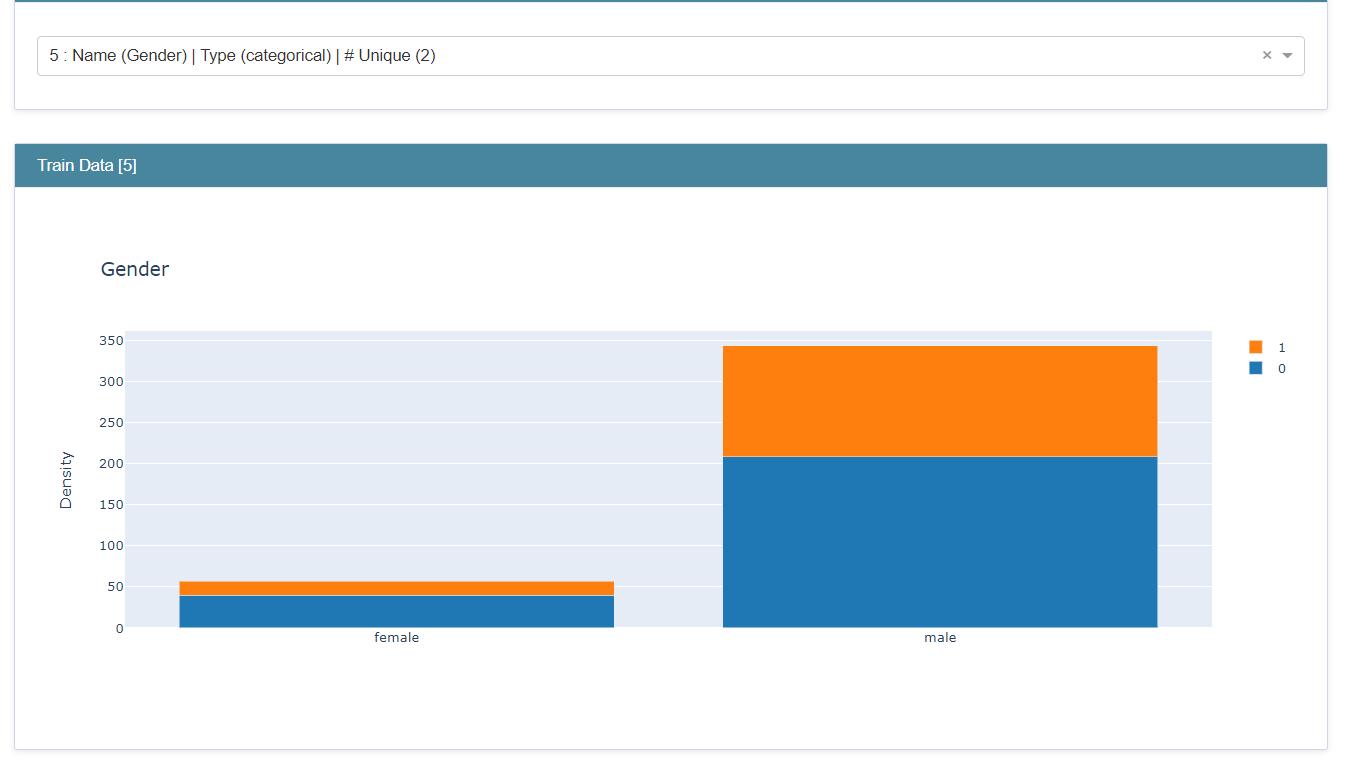


Figure Distribution of Gender (categorical column vs target variable)

## Step 4: Using Glassbox model from InterpretML

**Glassbox models**

1. Models that are meant to be interpretable and are internally present within interpretML.
2. There are 4 models which are created inhouse as part of IntepretML : Explainable Boosting Machines, Decision Tree, Decision Rule List, Linear/Logistic Regression
3. Basic idea behind building these models is to make general additive models more explainable even with Boosting (explainable boosting)
4. These also provide both Local as well as Global explainability.
5. This doesn’t work for a multi class classification problem

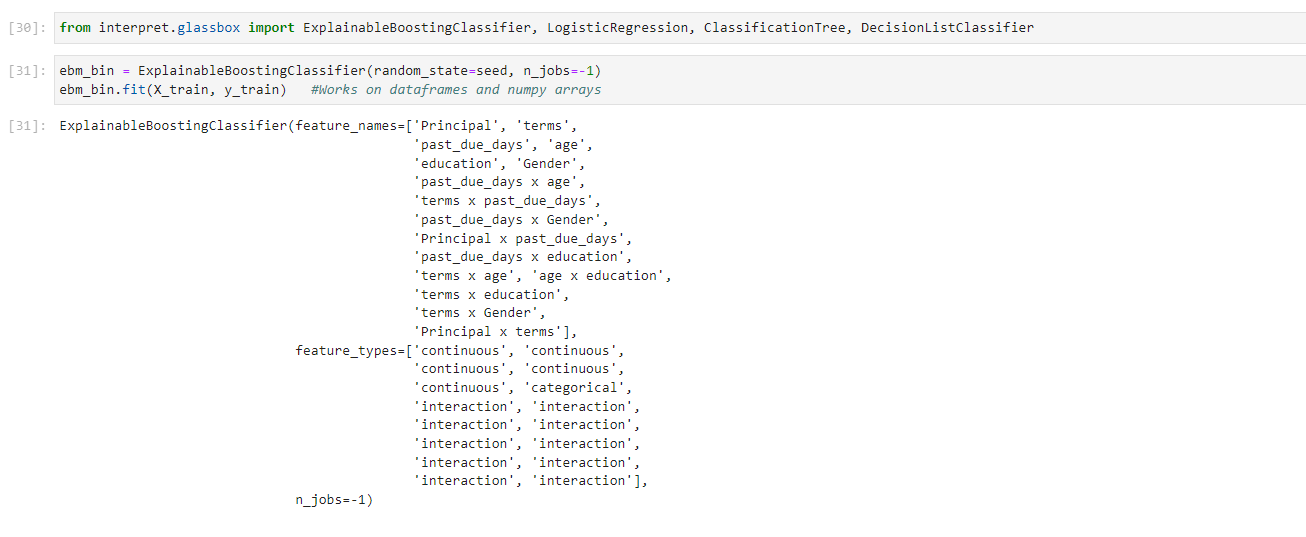


Figure : ML model fitting from InterpretML library­

## Step 5: Using global explainers from Glassbox models

* **An overall summary plot provides details on how each feature impacts overall model outcome**
* Individual plots can help us understand how model prediction varies for different range of values of each individual feature
* Higher you are on y axis in this individual plot, the higher chances that you will either have a collection or collection\_paidoff
* Looking at each segment in an individual feature helps in saving: - \*\*Sampling Bias , Overfitting

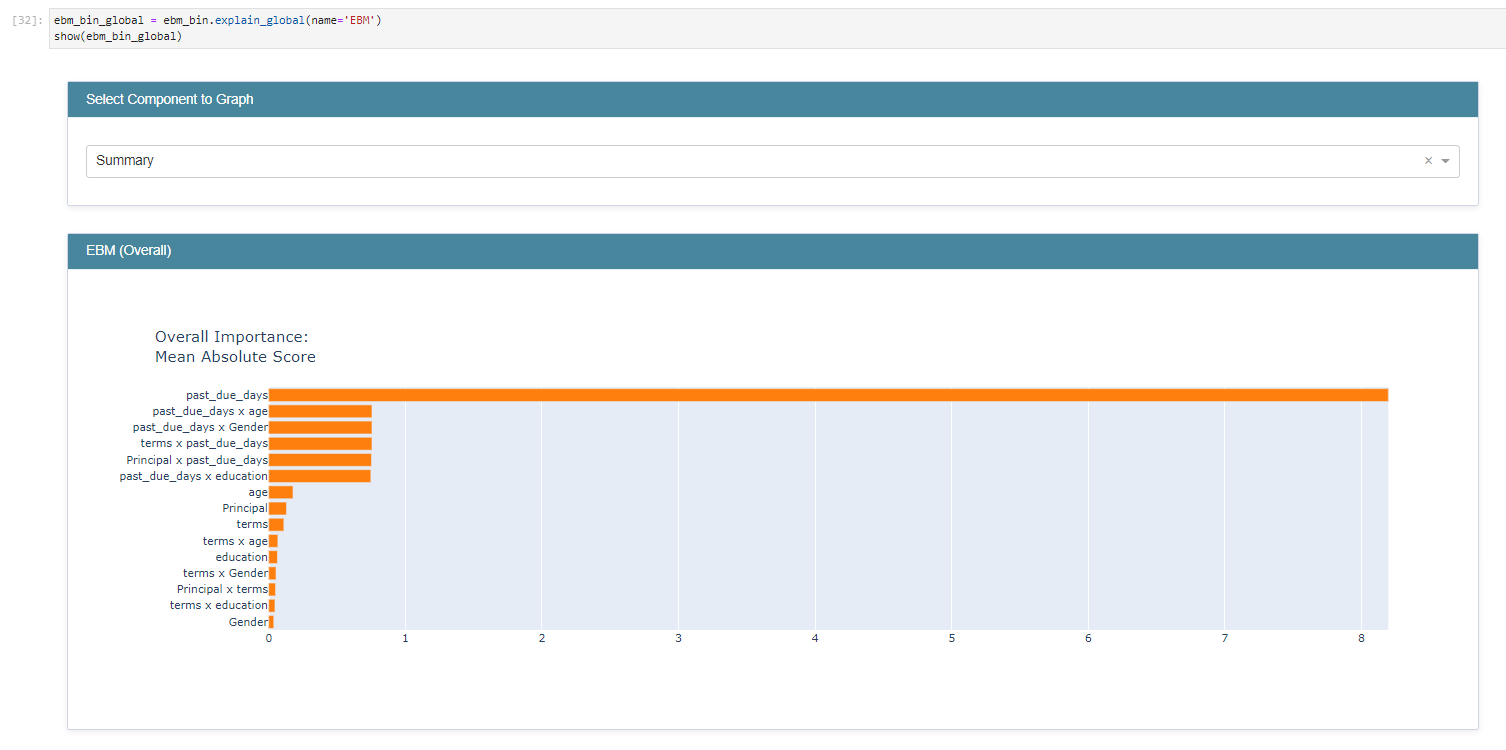


Figure - Global explainers from InterpretML : Provides feature importance at an overall model level.

## Step 6: Using local explainers from Glass box models

* Local explainers help in looking at each row item and understand feature contribution towards prediction.
* Select each row from the drop down to look at feature contribution

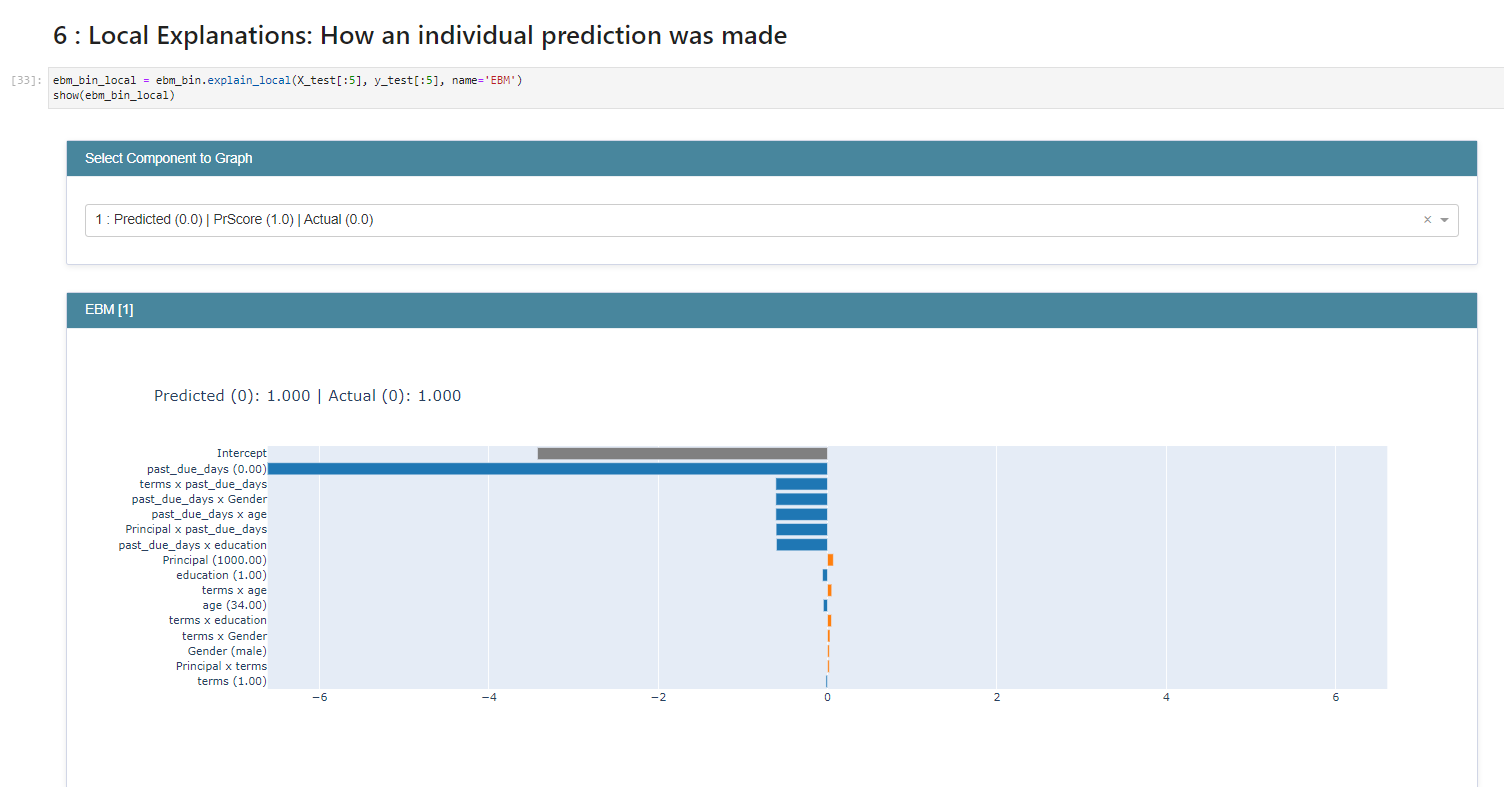


Figure - InterpretML local explanation

## Step 7: Glassbox Model: Performance metrics

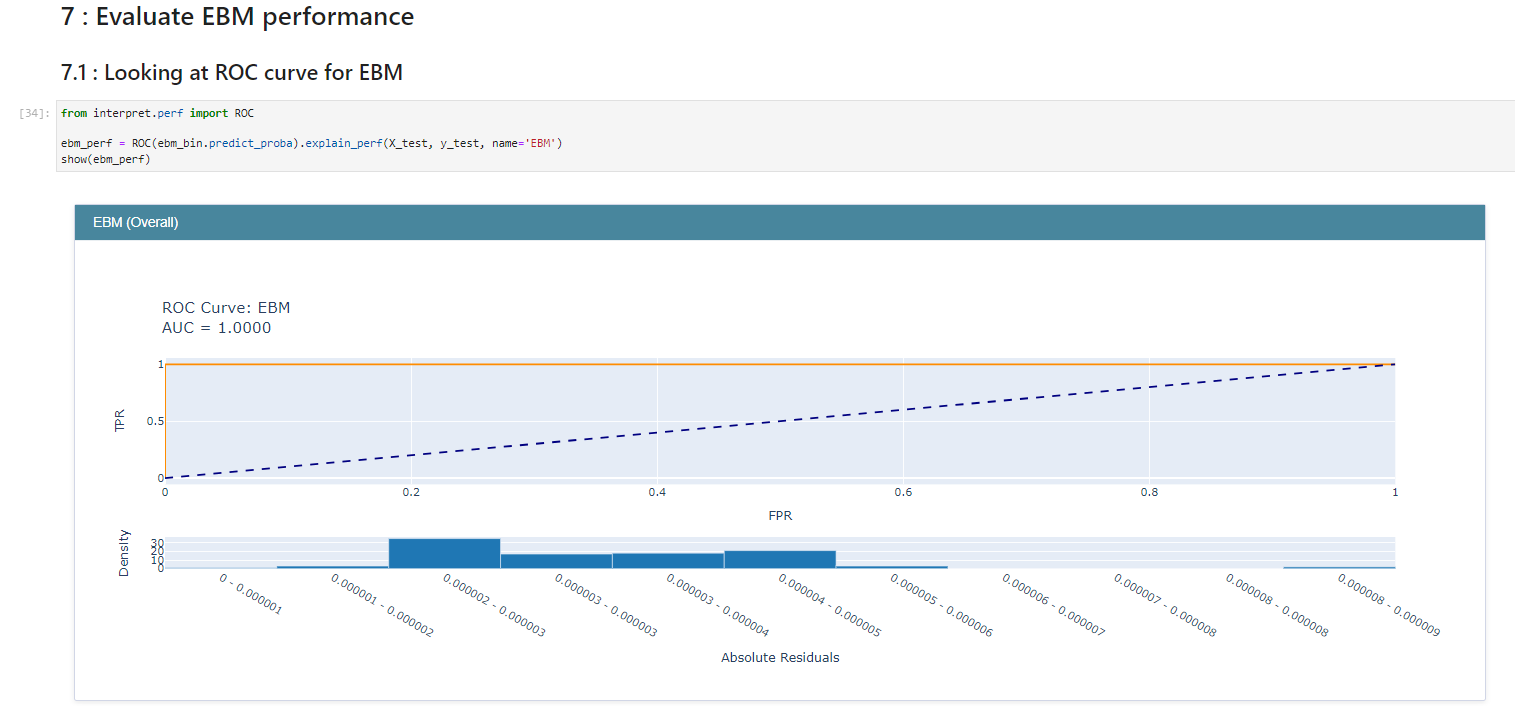


Figure :- ROC curve from the interpret Explainable Boosting Machine

## Step 8: Glassbox Model: Training other models



* Comparing the performance metrics for each model

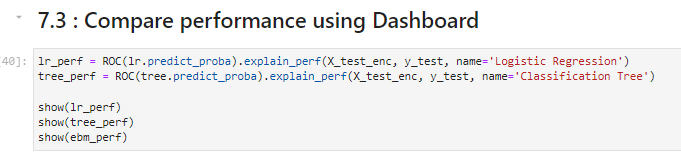


Figure - Code to generate performance metrics for each model

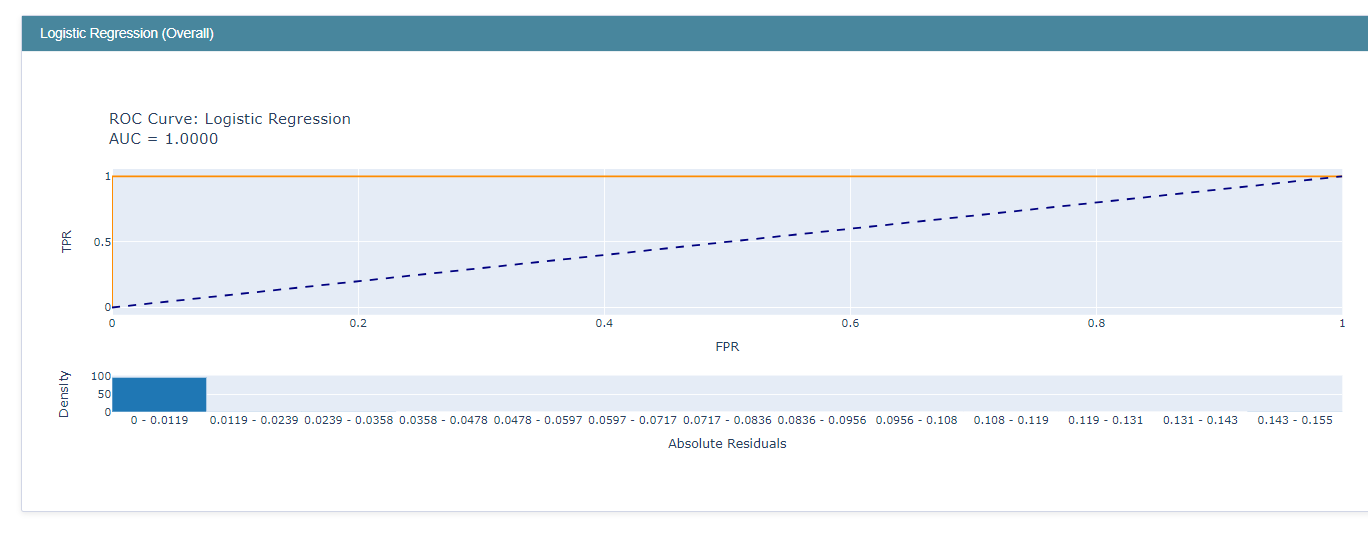
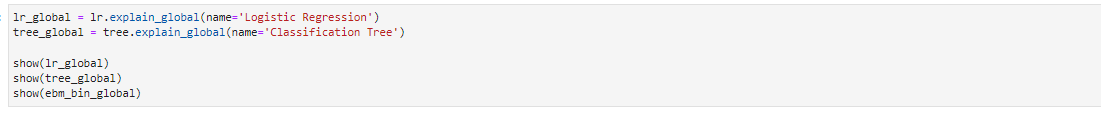
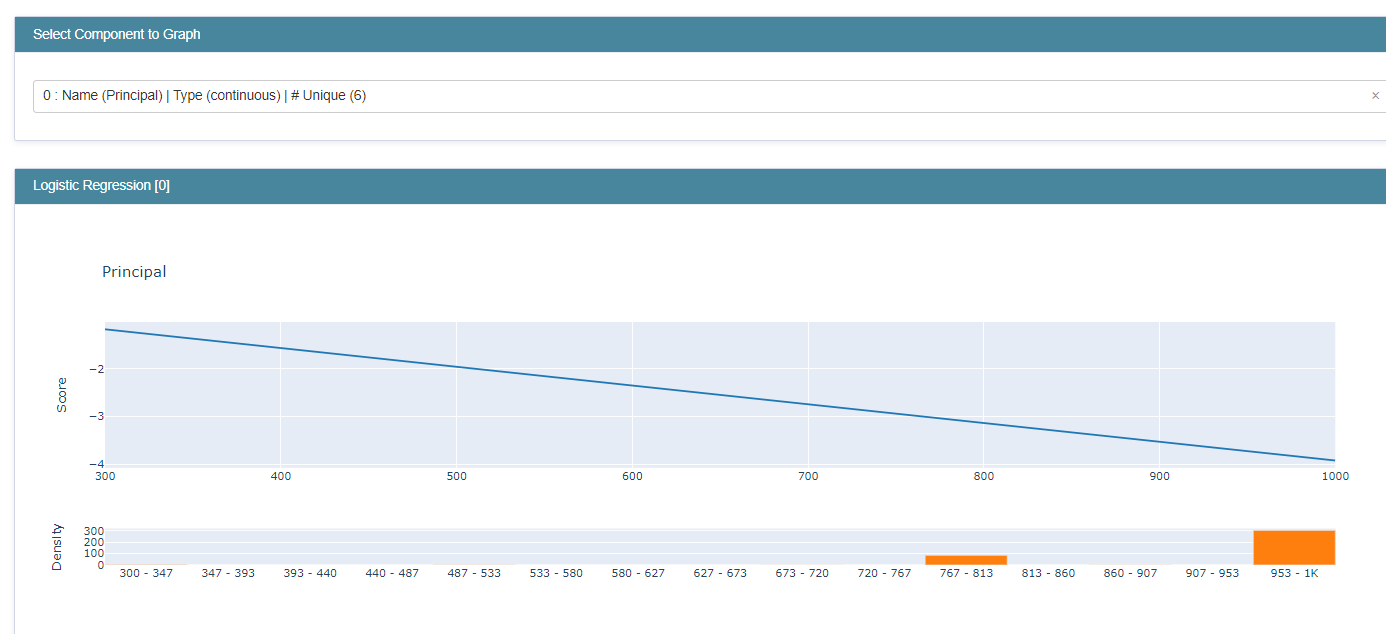


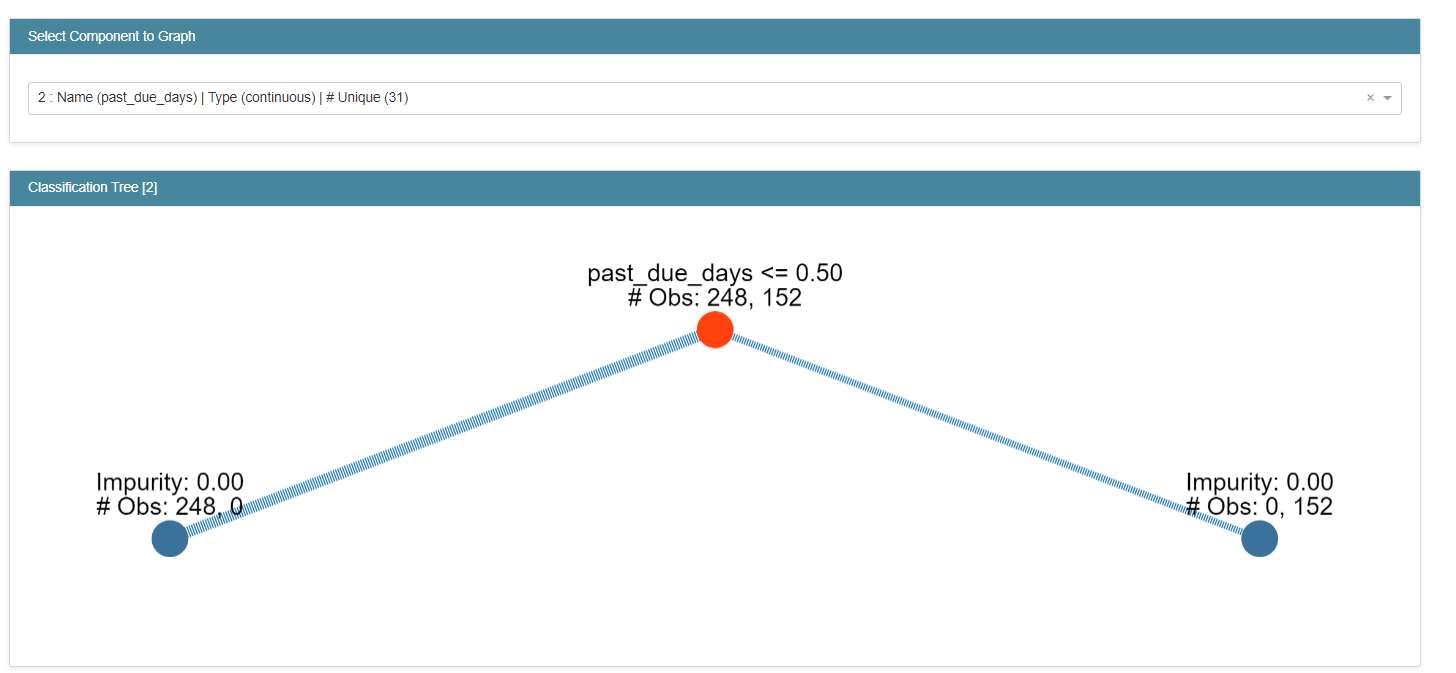
Figure InterpretML: Logistic regression: ROC

## Step 9: Glassbox Model: Generating global and local explanations for all models

## 







## Step 10: Dashboard view for all models



1. The above code would generate a dashboard with 4 tabs:
   * Overview: explains about all sections in the dashboard
   * Data: Select the data/features
   * Performance: Model performance by selecting model type
   * Global: provides global explanations by selecting model type
   * Local: provides row wise explanations

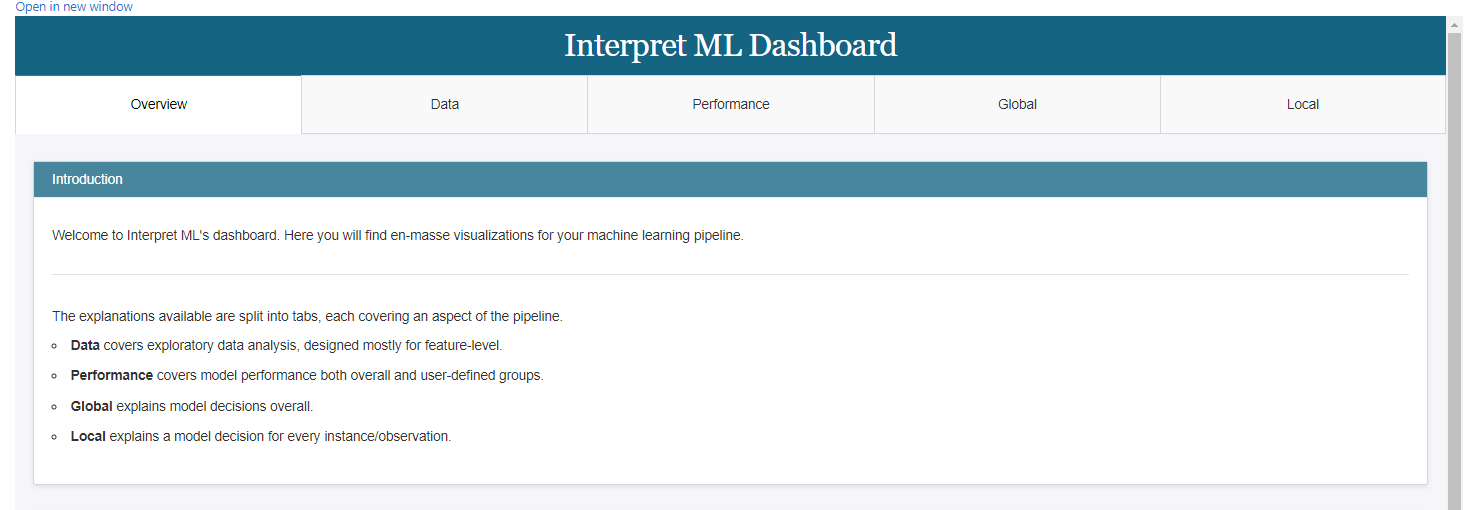


Figure - Dashboard view

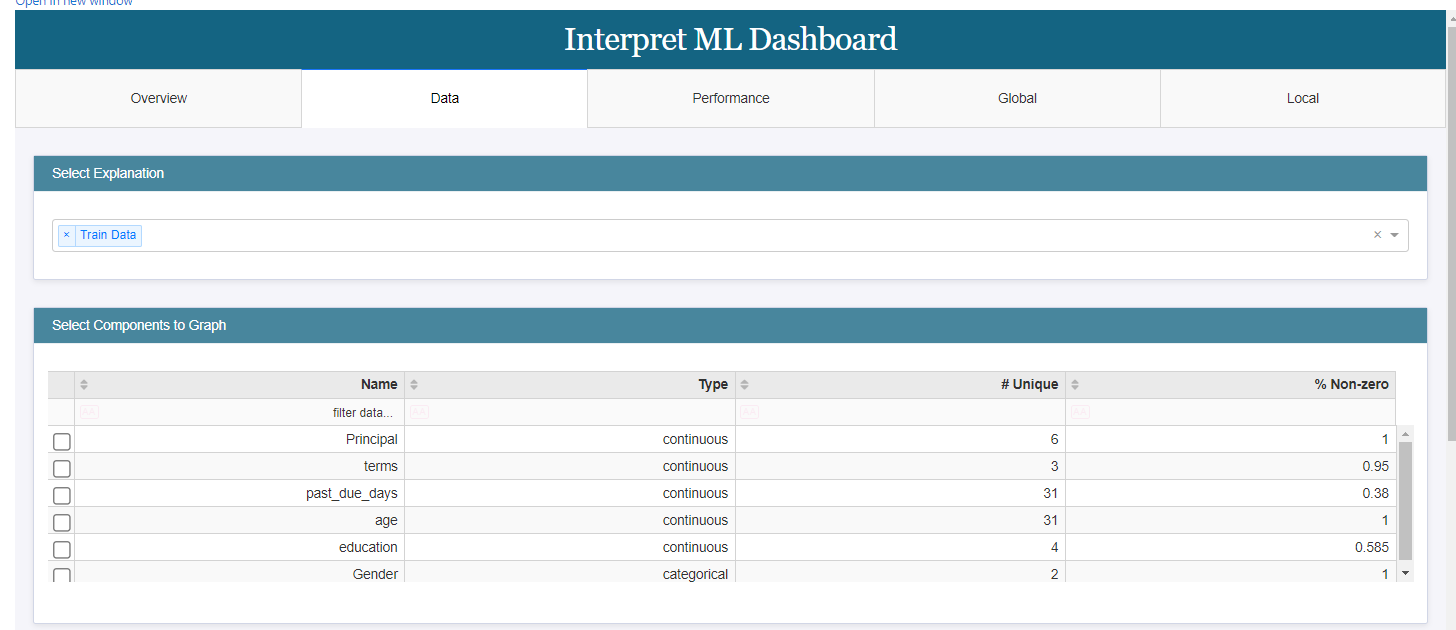


Figure - Data tab selected

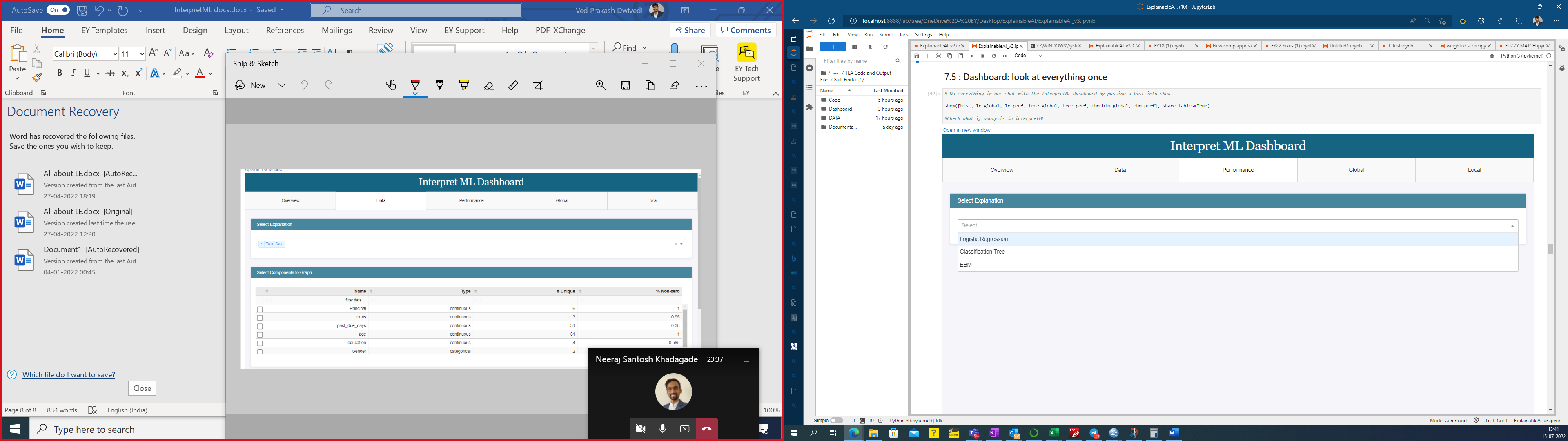


Figure - Select model type from Performance tab

# Step 11: Blackbox explainers using sklearn model

**Blackbox explanations**: These are explanations that can be given for any models coming from any other library such as sklearn

a.       They consider **only input and output values** and assume that based on these values they need to **identify the explainability.**

b.       In this process, changes are made to inputs and passed through model to analyse the change in model output and thus provide **explainability.**

c.       **Local explainers** in blackbox explanations include:**SHAP, LIME**

d.       **Global explainers** in Blackbox explanations include: **Partial Dependency Plot** and **Morris Sensitivity.**

e.       This can work very well on deep neural nets or on complex ml pipelines

## Step 12: Training blackbox model:

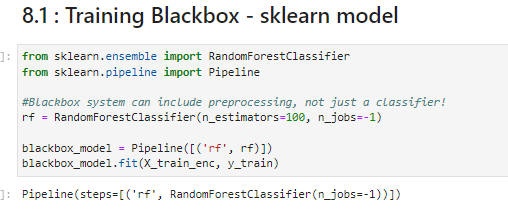


Figure - Training Sklearn based model

## Step 13: Generating model performance from Blackbox model using interpret library

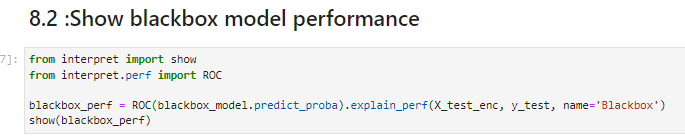
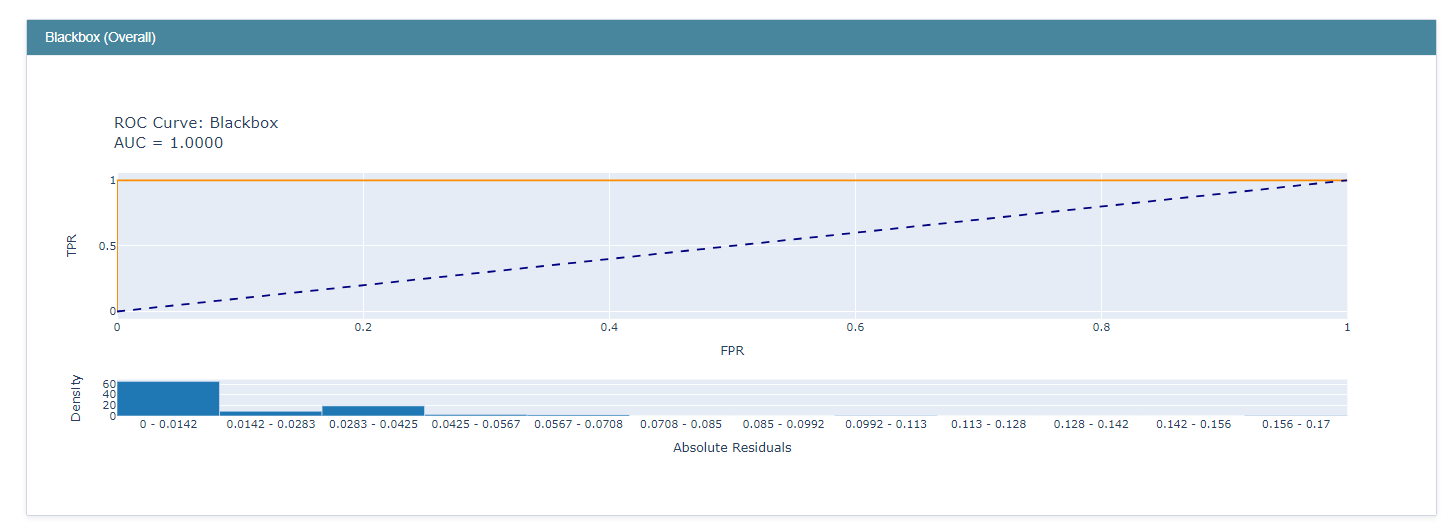


Figure - Using ROC method from interpret to understand model performance for sklearn based model

Using the code above helps in generating the model performance and showing the ROC curve (refer below) :



## Step 14: LIME for local explanations on blackbox model

* Lime tabular comes inbuilt with Interpret library and can provide local explanations for any tabular data:

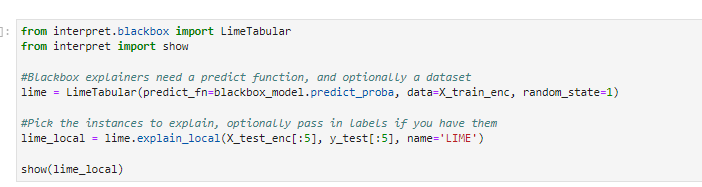


Figure - Generating Lime value for tabular data (sklearn models)

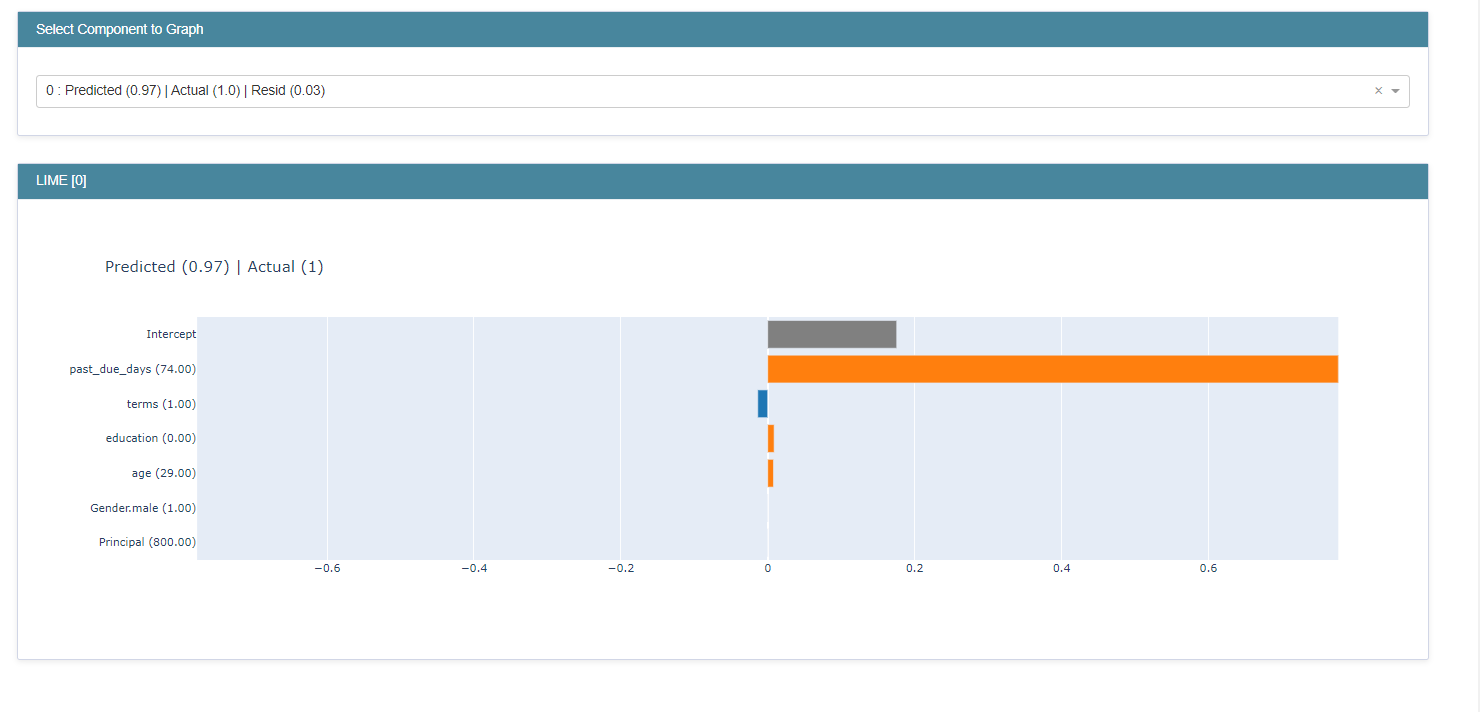


Figure : For 1st rows prediction: individual feature importance

* 'We can select from a dropdown row of the data (here first row = row 0 has been selected)
* **“Past\_Due\_days”:**  is the major contributor in prediction for this row.

## Step 15: SHAP for local explanations on blackbox model

* **Code: -**

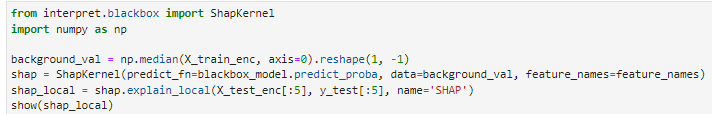


Figure - Code to generate Shap Value

* **Output: -**
  + Output provides a UI to select the dropdown (row number from the data)
  + Later we can look at how each feature contributes to row wise prediction

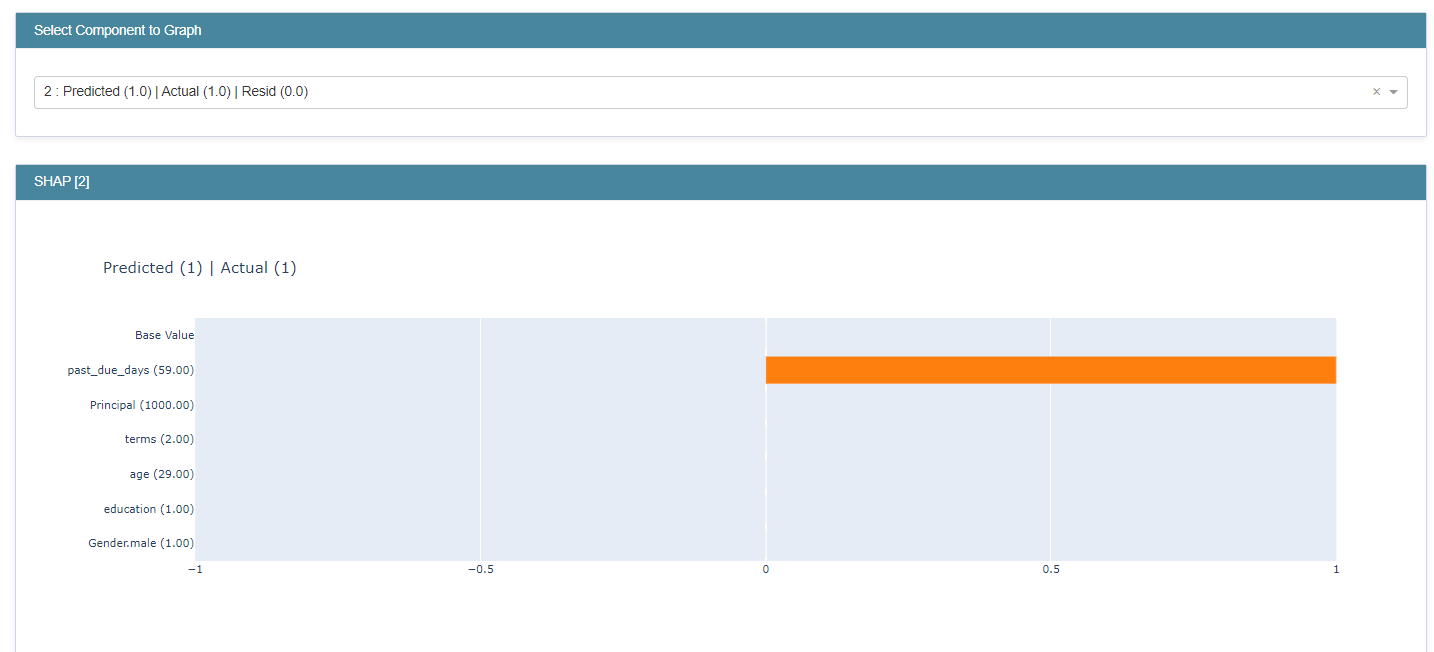
****

Figure - SHAP value: blackbox model

## Step 16: Global explanations: Morris sensitivity on blackbox model



Figure - Code to generate Morris sensitivy using interpret Library

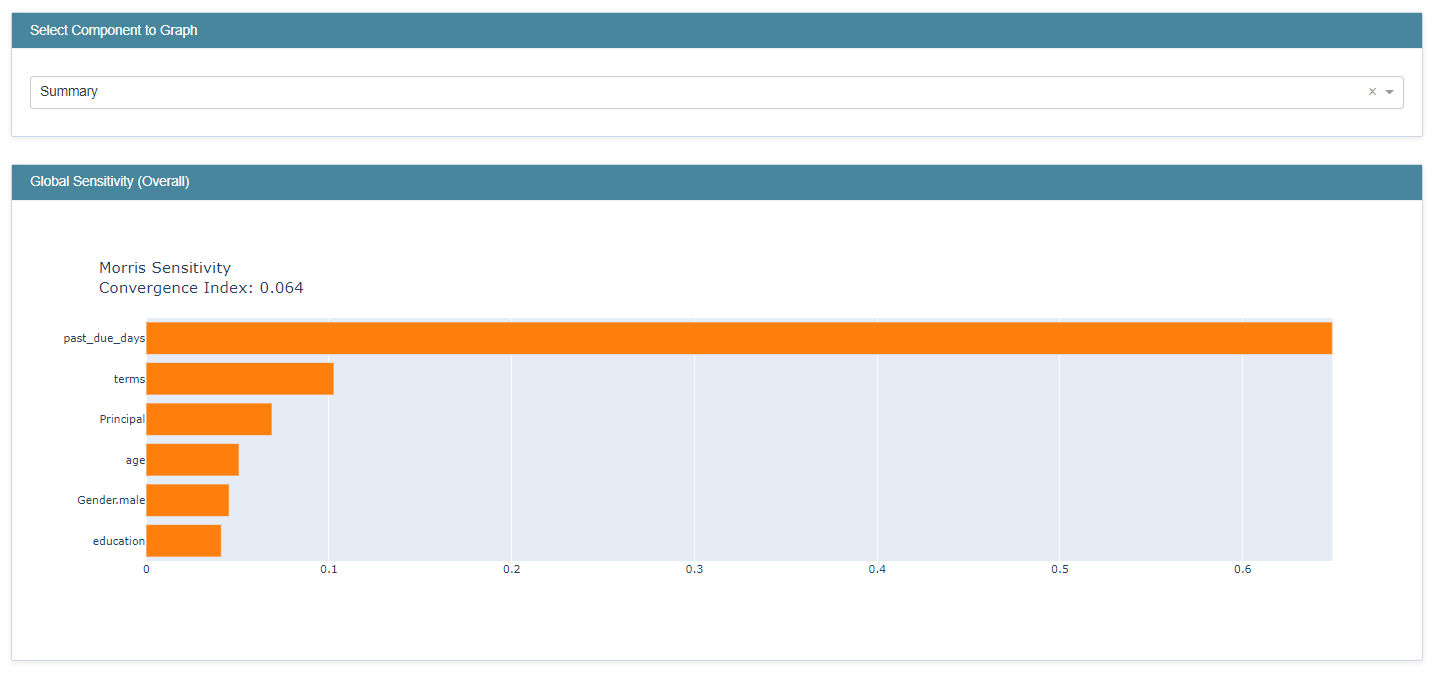


Figure - Output for Morris sensitivity show at a Model level : feature importance

## Step 17: Global explanations: Partial Dependence plots on blackbox model

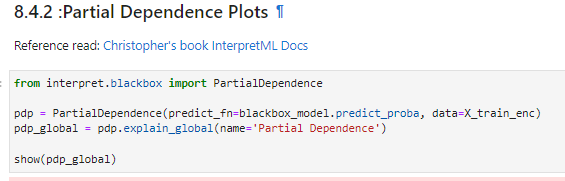


Figure - Code to use interpret library to generate PDP

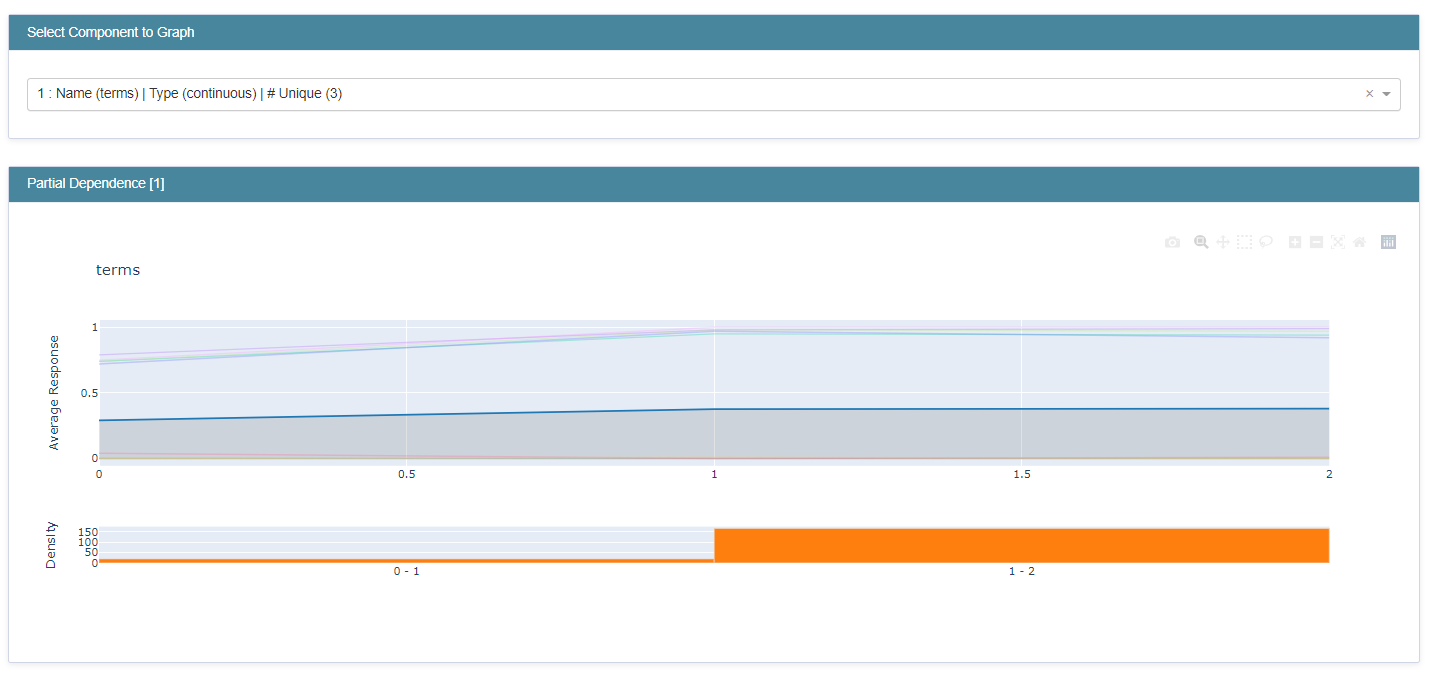


Figure : Select features from dropdown

* For different values of a feature, we can see how much it is contributing to the final model
* This would help in doing a **what-if analysis** later on
* *In above output: The model prediction gets impacted with values between 1-2 for the feature: Term.*

## Step 18: Comparing all explainer Dashboard: Blackbox model



Figure : Code to generate dashboard using interpret library

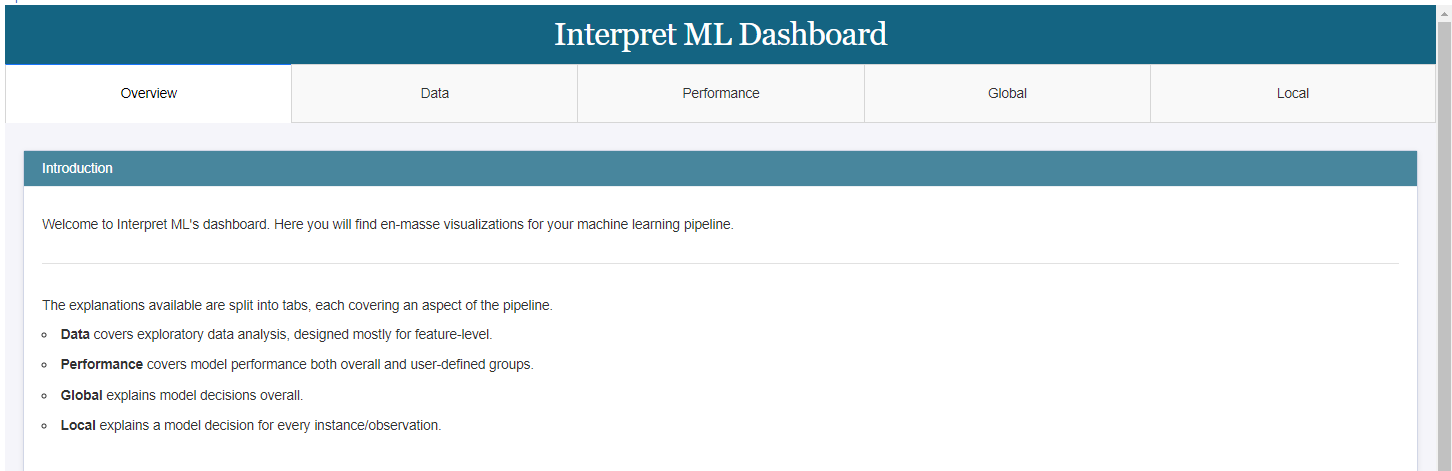


Figure - 5 tabs are generated

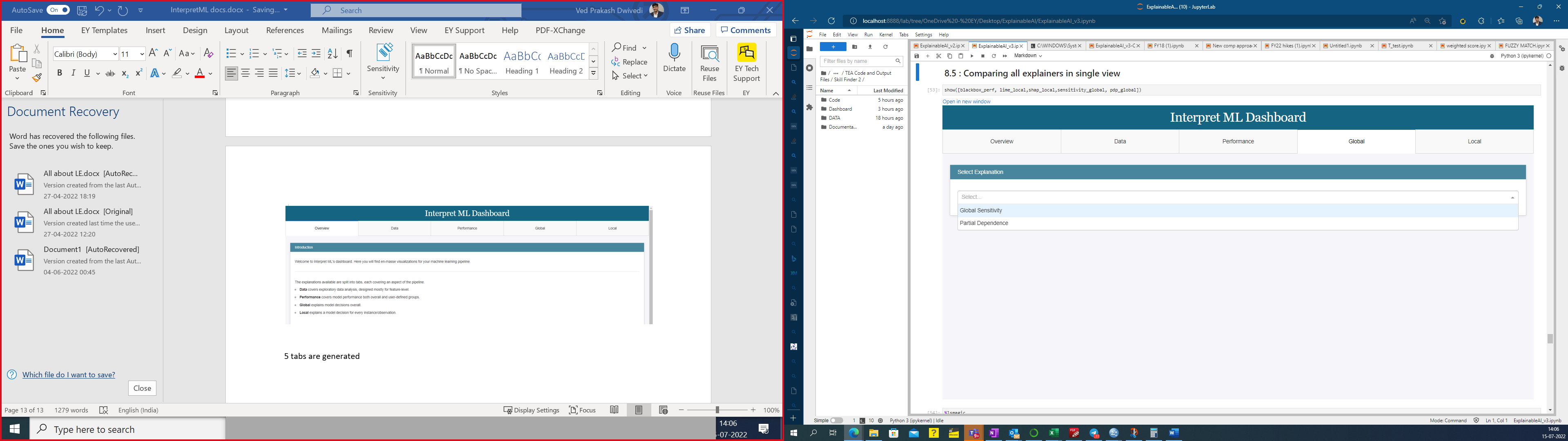


Figure : Option to select different global explainers from dropdown

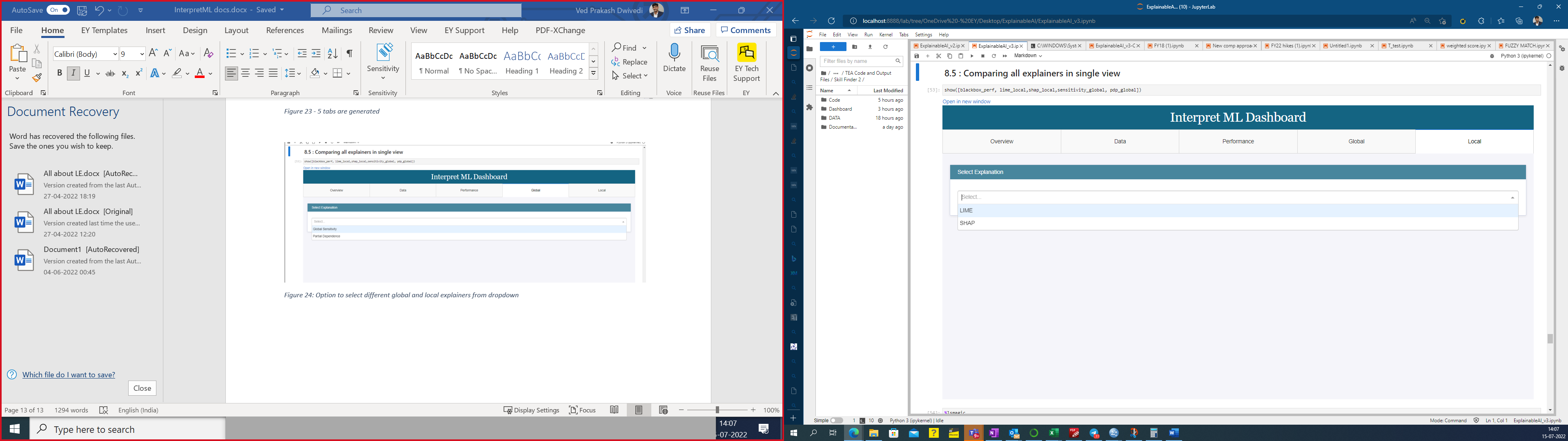


Figure - Option to select different local explainers from dropdown (SHAP, LIME)