Exercise 9: Visualization for ML Interpretability

Edward Gu

Here we are going to try using an out-of-the-box tool for some ML interpretability which supports some of the techniques we introduced in lecture.

The tool we will try out today is interpretML, an interpretability Python module maintained by Microsoft. See this blog post for an overview of the kinds of things you can do with interpretML, including code snippets that will be helpful for this exercise.

Preparing the dataset

For today's exercise, we'll be working with a dataset on emergency room visits and what factors predict whether a person is admitted or discharged. Before modeling this data, we need to do a bit of preprocessing. We provide this code since it's not the emphasis of our exercise.

We'll start by loading the dataset. The original is over 500,000 records, but we shared a subsampled version with 60,000 records via email.

In [ ]:
```
import pandas as pd

# students may need to change this path to whereever they downloaded the data
df = pd.read_csv("hospital-admissions.csv").drop(columns=['Unnamed: 0.1', 'Unnamed: 0'])
df.head()
```

Out[ ]:

	dep_name	esi	age	gender	ethnicity	race	lang	religion	maritalstatus	employstatus	...	cc_vaginaldischarge	cc_vaginalpain	cc_
0	A	2.0	48.0	Female	Hispanic or Latino	Other	English	Catholic	Single	Not Employed	...	0.0	0.0	
1	C	4.0	20.0	Female	Non-Hispanic	White or Caucasian	English	Catholic	Single	Full Time	...	0.0	0.0	
2	B	2.0	75.0	Female	Non-Hispanic	Black or African American	English	Pentecostal	Legally Separated	Retired	...	0.0	0.0	
3	B	4.0	50.0	Female	Hispanic or Latino	Other	English	Pentecostal	Married	Disabled	...	0.0	0.0	
4	A	4.0	26.0	Male	Non-Hispanic	Black or African American	English	NaN	Single	Full Time	...	0.0	0.0	

5 rows × 972 columns

The next block of code handles some preprocessing, namely, relabeling strings in our dataset as integers. This is necessary for modeling the data, and the way we do it here preserves a connection between the original string values and their "dummy coded" integers.

In [ ]:
```
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.utils import column_or_1d

# set up for relabeling
class MyLabelEncoder(LabelEncoder):

    def fit(self, y):
        y = column_or_1d(y, warn=True)
        self.classes_ = pd.Series(y).unique()
        return self

def code_string(df, colname):
    le = MyLabelEncoder()
    arr = df[colname].values
    le = le.fit(arr)
    df[colname] = le.transform(arr)

    return (df, le)

RECODE_SET = ['disposition', 'dep_name', 'gender', 'ethnicity', 'race', 'lang', 'religion', 'maritalstatus', 'employstatus', 'insura
```

In [ ]:
```
# prepare dataset:
# eliminate timeseries variables
for var in ['arrivalmonth', 'arrivalday', 'arrivalhour_bin']:
    df.pop(var)
# TODO: normalize X values?
numeric_columns = df.select_dtypes(include=['float64', 'int64']).columns
numeric_columns = [col for col in numeric_columns if col not in RECODE_SET]
scaler = StandardScaler()

# Apply normalization to numeric columns only
df[numeric_columns] = scaler.fit_transform(df[numeric_columns])
# make sure discharge maps to 0 when we use MyLabelEncoder
df.sort_values(by = 'disposition', ascending = True)
# call label encoder to code strings as numbers for modeling
label_encoders = {}
```

```
for var in RECODE_SET:
    df, label_encoders[var] = code_string(df, var)
# separate features and target
y = df.pop('disposition').values
X = df
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/extmath.py:1101: RuntimeWarning: invalid value encountered in divide
  updated_mean = (last_sum + new_sum) / updated_sample_count
/usr/local/lib/python3.11/dist-packages/sklearn/utils/extmath.py:1106: RuntimeWarning: invalid value encountered in divide
  T = new_sum / new_sample_count
/usr/local/lib/python3.11/dist-packages/sklearn/utils/extmath.py:1126: RuntimeWarning: invalid value encountered in divide
  new_unnormalized_variance -= correction**2 / new_sample_count
```

Modeling

Let's do the typical modeling workflow. I want you to try since the syntax is concise, and this is something you'll probably need to do often as a data scientist.

We'll start by using scikit-learn's test-train split, which is described well in this blog post. For our case, we already have clean input data separated into features `X` and classification labels `y`.

All you need to do is:

- Filter the features in `X` to only those that you would like to include in the model. You should pick 10 features to use as predictors in the model.
- Pass `X` and `y` to `sklearn.model_selection.train_test_split`
- Be sure to stratify on the outcome variable `y` so that the test and train sets have proportional numbers of people admitted to the emergency room

In [ ]:
```
# PROMPT: drop all but 10 features from X
selected_features = ['esi', 'age', 'gender', 'dep_name', 'race',
                     'insurance_status', 'arrivalmode', 'cc_chestpain',
                     'cc_shortnessofbreath', 'cc_abdominalpain']

X_filtered = X[selected_features]
```

In [ ]:
```
# PROMPT: split X, y into non-overlapping subsamples for testing and training our model
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X_filtered, y, test_size=0.2, random_state=42, stratify=y
)
```

PROMPT: What columns of X did you keep to include as predictors in your model? Why did you want to include these in particular?

I selected these 10 features as predictors because they represent a mix of clinical severity indicators, patient demographics, and healthcare system factors:

1. ESI (Emergency Severity Index): This is likely the strongest predictor as it directly measures the urgency of the patient's condition. Lower ESI scores indicate higher acuity cases that would likely require admission.
2. Age: Older patients typically have more complex health issues and less physiological reserve, often requiring admission even for conditions that might be managed outpatient in younger individuals.
3. Gender: There may be gender-based differences in disease presentation and treatment decisions that could affect admission rates.
4. Department name: Different departments likely have different admission thresholds and specialties.
5. Race: Healthcare disparities exist, and race might be associated with different admission patterns or disease presentations.
6. Insurance status: This can impact healthcare decisions, as financial considerations sometimes factor into admission decisions.
7. Arrival mode: How a patient arrives (ambulance vs. walk-in) is a strong indicator of severity and can influence admission decisions.
8. Chest pain: A serious symptom that often leads to admission to rule out cardiac events or pulmonary diseases that are often life-threatening.
9. Shortness of breath: Respiratory distress usually requires careful medical monitoring and may necessitate hospital admission.
10. Abdominal pain: A common presenting complaint that can indicate various serious conditions requiring admission.

Next, we will fit a model to our training data. Let's use a type of GAM called an Explainable Boosting Machine (EBM). This is basically a Generalized Additive Model fit using a variation on decision trees. We'll use the classifier version of this model, `interpret.glassbox.ExplainableBoostingClassifier`.

In [ ]:
```
!pip3 install interpret
```

```
Collecting interpret
  Downloading interpret-0.6.9-py3-none-any.whl.metadata (1.0 kB)
Collecting interpret-core==0.6.9 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading interpret_core-0.6.9-py3-none-any.whl.metadata (2.8 kB)
Requirement already satisfied: numpy>=1.25 in /usr/local/lib/python3.11/dist-packages (from interpret-core==0.6.9->interpret-core[ap
lr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.26.4)
Requirement already satisfied: pandas>=0.19.2 in /usr/local/lib/python3.11/dist-packages (from interpret-core==0.6.9->interpret-core
[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2.2.2)
Requirement already satisfied: scikit-learn>=0.18.1 in /usr/local/lib/python3.11/dist-packages (from interpret-core==0.6.9->interpre
t-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.6.1)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.11/dist-packages (from interpret-core==0.6.9->interpret-core[a
plr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.4.2)
Requirement already satisfied: psutil>=5.6.2 in /usr/local/lib/python3.11/dist-packages (from interpret-core[aplr,dash,debug,linear,
notebook,plotly,sensitivity,shap]==0.6.9->interpret) (5.9.5)
Requirement already satisfied: ipykernel>=4.10.0 in /usr/local/lib/python3.11/dist-packages (from interpret-core[aplr,dash,debug,lin
ear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (6.17.1)
Requirement already satisfied: ipython>=5.5.0 in /usr/local/lib/python3.11/dist-packages (from interpret-core[aplr,dash,debug,linea
r,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (7.34.0)
Requirement already satisfied: plotly>=3.8.1 in /usr/local/lib/python3.11/dist-packages (from interpret-core[aplr,dash,debug,linear,
notebook,plotly,sensitivity,shap]==0.6.9->interpret) (5.24.1)
Collecting SALib>=1.3.3 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading salib-1.5.1-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: shap>=0.28.5 in /usr/local/lib/python3.11/dist-packages (from interpret-core[aplr,dash,debug,linear,n
otebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.46.0)
Collecting dill>=0.2.5 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading dill-0.3.9-py3-none-any.whl.metadata (10 kB)
Collecting aplr>=10.6.1 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading aplr-10.8.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (2.1 kB)
Collecting dash>=1.0.0 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading dash-2.18.2-py3-none-any.whl.metadata (10 kB)
Collecting dash-core-components>=1.0.0 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpr
et)
  Downloading dash_core_components-2.0.0-py3-none-any.whl.metadata (2.9 kB)
Collecting dash-html-components>=1.0.0 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpr
et)
  Downloading dash_html_components-2.0.0-py3-none-any.whl.metadata (3.8 kB)
Collecting dash-table>=4.1.0 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading dash_table-5.0.0-py3-none-any.whl.metadata (2.4 kB)
Collecting dash-cytoscape>=0.1.1 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading dash_cytoscape-1.0.2.tar.gz (4.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.0/4.0 MB 81.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting gevent>=1.3.6 (from interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading gevent-24.11.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (13 kB)
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.11/dist-packages (from interpret-core[aplr,dash,debug,linea
r,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2.32.3)
Collecting Flask<3.1,>=1.0.4 (from dash>=1.0.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->inte
rpret)
  Downloading flask-3.0.3-py3-none-any.whl.metadata (3.2 kB)
Collecting Werkzeug<3.1 (from dash>=1.0.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpre
t)
  Downloading werkzeug-3.0.6-py3-none-any.whl.metadata (3.7 kB)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.11/dist-packages (from dash>=1.0.0->interpret-core[aplr,
dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (8.6.1)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.11/dist-packages (from dash>=1.0.0->interpret-core
[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (4.12.2)
Collecting retrying (from dash>=1.0.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret)
  Downloading retrying-1.3.4-py3-none-any.whl.metadata (6.9 kB)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.11/dist-packages (from dash>=1.0.0->interpret-core[aplr,dash,d
ebug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.6.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from dash>=1.0.0->interpret-core[aplr,dash,deb
ug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (75.1.0)
Collecting zope.event (from gevent>=1.3.6->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpre
t)
  Downloading zope.event-5.0-py3-none-any.whl.metadata (4.4 kB)
Collecting zope.interface (from gevent>=1.3.6->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->inter
pret)
  Downloading zope.interface-7.2-cp311-cp311-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl.m
etadata (44 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.4/44.4 kB 3.0 MB/s eta 0:00:00
Requirement already satisfied: greenlet>=3.1.1 in /usr/local/lib/python3.11/dist-packages (from gevent>=1.3.6->interpret-core[aplr,d
ash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.1.1)
Requirement already satisfied: debugpy>=1.0 in /usr/local/lib/python3.11/dist-packages (from ipykernel>=4.10.0->interpret-core[aplr,
dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.8.0)
Requirement already satisfied: jupyter-client>=6.1.12 in /usr/local/lib/python3.11/dist-packages (from ipykernel>=4.10.0->interpret-
core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (6.1.12)
Requirement already satisfied: matplotlib-inline>=0.1 in /usr/local/lib/python3.11/dist-packages (from ipykernel>=4.10.0->interpret-
core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.1.7)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from ipykernel>=4.10.0->interpret-core[aplr,das
h,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (24.2)
Requirement already satisfied: pyzmq>=17 in /usr/local/lib/python3.11/dist-packages (from ipykernel>=4.10.0->interpret-core[aplr,das
h,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (24.0.1)
Requirement already satisfied: tornado>=6.1 in /usr/local/lib/python3.11/dist-packages (from ipykernel>=4.10.0->interpret-core[aplr,
dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (6.4.2)
Requirement already satisfied: traitlets>=5.1.0 in /usr/local/lib/python3.11/dist-packages (from ipykernel>=4.10.0->interpret-core[a
plr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (5.7.1)
Collecting jedi>=0.16 (from ipython>=5.5.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpre
t)
  Downloading jedi-0.19.2-py2.py3-none-any.whl.metadata (22 kB)
Requirement already satisfied: decorator in /usr/local/lib/python3.11/dist-packages (from ipython>=5.5.0->interpret-core[aplr,dash,d
```

```
ebug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.11/dist-packages (from ipython>=5.5.0->interpret-core[aplr,das
h,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.7.5)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from ipython
>=5.5.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.0.50)
Requirement already satisfied: pygments in /usr/local/lib/python3.11/dist-packages (from ipython>=5.5.0->interpret-core[aplr,dash,de
bug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2.18.0)
Requirement already satisfied: backcall in /usr/local/lib/python3.11/dist-packages (from ipython>=5.5.0->interpret-core[aplr,dash,de
bug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.2.0)
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.11/dist-packages (from ipython>=5.5.0->interpret-core[aplr,das
h,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (4.9.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.19.2->interpret-cor
e==0.6.9->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.19.2->interpret-core==0.6.9->
interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.19.2->interpret-core==0.6.9
->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2025.1)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.11/dist-packages (from plotly>=3.8.1->interpret-core[aplr,d
ash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (9.0.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->interpret
-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->interpret-core[aplr,d
ash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->interpret-core
[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->interpret-core
[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2025.1.31)
Requirement already satisfied: matplotlib>=3.5 in /usr/local/lib/python3.11/dist-packages (from SALib>=1.3.3->interpret-core[aplr,da
sh,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.10.0)
Collecting multiprocess (from SALib>=1.3.3->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpre
t)
  Downloading multiprocess-0.70.17-py311-none-any.whl.metadata (7.2 kB)
Requirement already satisfied: scipy>=1.9.3 in /usr/local/lib/python3.11/dist-packages (from SALib>=1.3.3->interpret-core[aplr,dash,
debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.13.1)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn>=0.18.1->interpret
-core==0.6.9->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.5.0)
Requirement already satisfied: tqdm>=4.27.0 in /usr/local/lib/python3.11/dist-packages (from shap>=0.28.5->interpret-core[aplr,dash,
debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (4.67.1)
Requirement already satisfied: slicer==0.0.8 in /usr/local/lib/python3.11/dist-packages (from shap>=0.28.5->interpret-core[aplr,das
h,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.0.8)
Requirement already satisfied: numba in /usr/local/lib/python3.11/dist-packages (from shap>=0.28.5->interpret-core[aplr,dash,debug,l
inear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.60.0)
Requirement already satisfied: cloudpickle in /usr/local/lib/python3.11/dist-packages (from shap>=0.28.5->interpret-core[aplr,dash,d
ebug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.1.1)
Requirement already satisfied: Jinja2>=3.1.2 in /usr/local/lib/python3.11/dist-packages (from Flask<3.1,>=1.0.4->dash>=1.0.0->interp
ret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.1.5)
Requirement already satisfied: itsdangerous>=2.1.2 in /usr/local/lib/python3.11/dist-packages (from Flask<3.1,>=1.0.4->dash>=1.0.0->
interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (2.2.0)
Requirement already satisfied: click>=8.1.3 in /usr/local/lib/python3.11/dist-packages (from Flask<3.1,>=1.0.4->dash>=1.0.0->interpr
et-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (8.1.8)
Requirement already satisfied: blinker>=1.6.2 in /usr/local/lib/python3.11/dist-packages (from Flask<3.1,>=1.0.4->dash>=1.0.0->inter
pret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.9.0)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in /usr/local/lib/python3.11/dist-packages (from jedi>=0.16->ipython>=5.5.0->inte
rpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.8.4)
Requirement already satisfied: jupyter-core>=4.6.0 in /usr/local/lib/python3.11/dist-packages (from jupyter-client>=6.1.12->ipykerne
l>=4.10.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (5.7.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib>=3.5->SALib>=1.3.3->inte
rpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib>=3.5->SALib>=1.3.3->interpret
-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib>=3.5->SALib>=1.3.3->int
erpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (4.56.0)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib>=3.5->SALib>=1.3.3->int
erpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.4.8)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib>=3.5->SALib>=1.3.3->interpret-c
ore[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib>=3.5->SALib>=1.3.3->inte
rpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.2.1)
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.11/dist-packages (from pexpect>4.3->ipython>=5.5.0->interpr
et-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.7.0)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.11/dist-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0
->ipython>=5.5.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.2.13)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas>=0.19.2->int
erpret-core==0.6.9->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (1.17.0)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.11/dist-packages (from Werkzeug<3.1->dash>=1.0.0->interpr
et-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.0.2)
Requirement already satisfied: zipp>=3.20 in /usr/local/lib/python3.11/dist-packages (from importlib-metadata->dash>=1.0.0->interpre
t-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (3.21.0)
Requirement already satisfied: llvmlite<0.44,>=0.43.0dev0 in /usr/local/lib/python3.11/dist-packages (from numba->shap>=0.28.5->inte
rpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (0.43.0)
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.11/dist-packages (from jupyter-core>=4.6.0->jupyter-clien
t>=6.1.12->ipykernel>=4.10.0->interpret-core[aplr,dash,debug,linear,notebook,plotly,sensitivity,shap]==0.6.9->interpret) (4.3.6)
Downloading interpret-0.6.9-py3-none-any.whl (1.4 kB)
Downloading interpret_core-0.6.9-py3-none-any.whl (15.0 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 15.0/15.0 MB 83.0 MB/s eta 0:00:00
Downloading aplr-10.8.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (6.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.7/6.7 MB 94.0 MB/s eta 0:00:00
Downloading dash-2.18.2-py3-none-any.whl (7.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.8/7.8 MB 92.4 MB/s eta 0:00:00
Downloading dash_core_components-2.0.0-py3-none-any.whl (3.8 kB)
```

```
Downloading dash_html_components-2.0.0-py3-none-any.whl (4.1 kB)
Downloading dash_table-5.0.0-py3-none-any.whl (3.9 kB)
Downloading dill-0.3.9-py3-none-any.whl (119 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 119.4/119.4 kB 9.3 MB/s eta 0:00:00
Downloading gevent-24.11.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (6.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.8/6.8 MB 93.9 MB/s eta 0:00:00
Downloading salib-1.5.1-py3-none-any.whl (778 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 778.9/778.9 kB 37.9 MB/s eta 0:00:00
Downloading flask-3.0.3-py3-none-any.whl (101 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 101.7/101.7 kB 8.1 MB/s eta 0:00:00
Downloading jedi-0.19.2-py2.py3-none-any.whl (1.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 64.3 MB/s eta 0:00:00
Downloading werkzeug-3.0.6-py3-none-any.whl (227 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 228.0/228.0 kB 18.1 MB/s eta 0:00:00
Downloading multiprocess-0.70.17-py311-none-any.whl (144 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 144.3/144.3 kB 11.0 MB/s eta 0:00:00
Downloading retrying-1.3.4-py3-none-any.whl (11 kB)
Downloading zope.event-5.0-py3-none-any.whl (6.8 kB)
Downloading zope.interface-7.2-cp311-cp311-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25
9 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 259.8/259.8 kB 17.4 MB/s eta 0:00:00
Building wheels for collected packages: dash-cytoscape
  Building wheel for dash-cytoscape (setup.py) ... done
  Created wheel for dash-cytoscape: filename=dash_cytoscape-1.0.2-py3-none-any.whl size=4010716 sha256=d06d029c2f71b0b8d1823e936a5ee
c35791fd18c7be88193e3082f4d67da12ac
  Stored in directory: /root/.cache/pip/wheels/99/b1/ab/6c999ab288b4849d372e23c0a8f6ece7edb7ffeb8c97959ab0
Successfully built dash-cytoscape
Installing collected packages: dash-table, dash-html-components, dash-core-components, zope.interface, zope.event, Werkzeug, retryin
g, jedi, dill, aplr, multiprocess, gevent, Flask, SALib, interpret-core, dash, dash-cytoscape, interpret
  Attempting uninstall: Werkzeug
    Found existing installation: Werkzeug 3.1.3
    Uninstalling Werkzeug-3.1.3:
      Successfully uninstalled Werkzeug-3.1.3
  Attempting uninstall: Flask
    Found existing installation: Flask 3.1.0
    Uninstalling Flask-3.1.0:
      Successfully uninstalled Flask-3.1.0
Successfully installed Flask-3.0.3 SALib-1.5.1 Werkzeug-3.0.6 aplr-10.8.0 dash-2.18.2 dash-core-components-2.0.0 dash-cytoscape-1.0.
2 dash-html-components-2.0.0 dash-table-5.0.0 dill-0.3.9 gevent-24.11.1 interpret-0.6.9 interpret-core-0.6.9 jedi-0.19.2 multiproces
s-0.70.17 retrying-1.3.4 zope.event-5.0 zope.interface-7.2
```

In [ ]:
```python
# PROMPT: fit an EBM classifier to the training sample
from interpret.glassbox import ExplainableBoostingClassifier

ebm = ExplainableBoostingClassifier(random_state=42)
ebm.fit(X_train, y_train)
```

```
/usr/local/lib/python3.11/dist-packages/interpret/glassbox/_ebm/_ebm.py:813: UserWarning: Missing values detected. Our visualization
s do not currently display missing values. To retain the glassbox nature of the model you need to either set the missing values to a
n extreme value like -1000 that will be visible on the graphs, or manually examine the missing value score in ebm.term_scores_[term_
index][0]
  warn(
```

Out[ ]:
```
▾ ExplainableBoostingClassifier
ExplainableBoostingClassifier()
```

You are going to see a warning about missing data. It's best not to ignore that in practice.

Let's compute our test set accuracy using an AUC (Area Under the receiver operating characteristic Curve), which is a common measure of accuracy for classifiers. To do this, you'll need to use `sklearn.metrics.accuracy_score` (see API documentation).

In [ ]:
```python
# PROMPT: compute test set accuracy as a percentage.
# HINT: you'll need to round the output of ebm.predict() to get binary predictions
from sklearn.metrics import accuracy_score
import numpy as np

y_pred = np.round(ebm.predict(X_test)).astype(int)
accuracy_percentage = accuracy_score(y_test, y_pred) * 100

print(f"Test set accuracy: {accuracy_percentage:.2f}%")
```

```
Test set accuracy: 79.38%
```

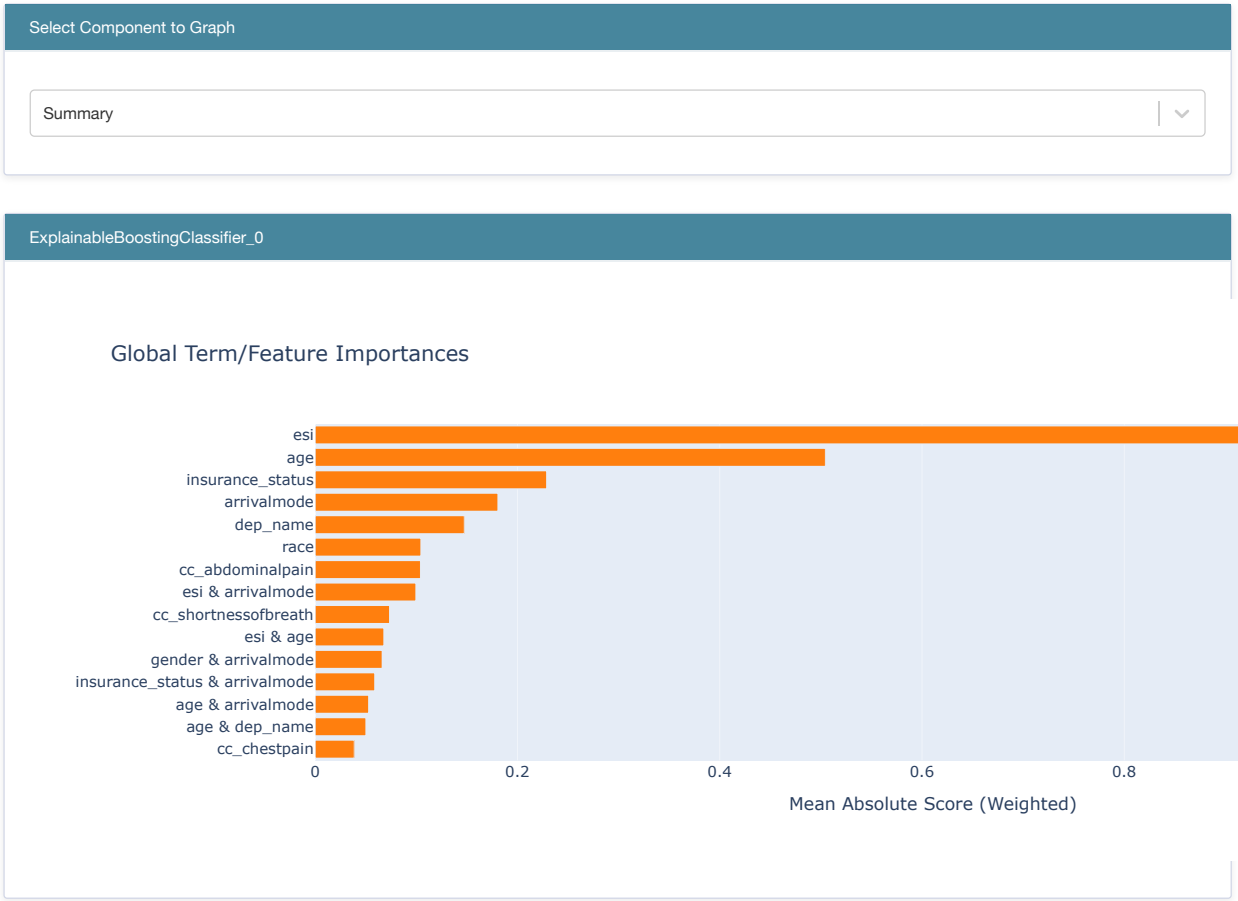
Here, the test set accuracy of the model with my selected features is 79.38%, suggesting moderate model performance.

Interpretability!

Now that we have a model fit to our data, we can use the interpretability tools provided by `interpretML` to explain what our model has learned. You'll need `interpret.show` along with the `ebm.explain_global()` and `ebm.explain_local()` methods.

In [ ]:
```python
# PROMPT: generate interpret's interactive widget for global explanations
from interpret import show

global_explanation = ebm.explain_global()
show(global_explanation)
```



PROMPT: Look at the "Summary" of feature importance. What are the top three predictors? Do these make sense to you? Are they what you expected?

Based on the summary of feature importance shown in the image, the top three predictors for hospital admission are:

1. ESI (Emergency Severity Index) - By far the most dominant predictor with a score of approximately 1.2
2. Age - The second most important predictor with a score around 0.5
3. Insurance_status - The third most important with a score of about 0.25

These top predictors align with clinical knowledge and what's going on in the healthcare system in general. ESI being the dominant predictor makes sense as it's specifically designed to categorize patients based on acuity and resource needs. Lower ESI scores (indicating higher urgency) naturally correlate with higher admission rates, so this is exactly what we would expect to see in a well-functioning model.

Age as the second most important factor also aligns with clinical experience - older patients typically have more comorbidities, decreased physiological reserve, and greater vulnerability to complications, all factors that increase hospitalization likelihood. What's interesting and perhaps slightly surprising is that insurance_status -- often considered a non-clinical factor -- ranks higher than clinical symptoms like chest pain or shortness of breath. This suggests that beyond purely clinical factors, healthcare system considerations play a significant role in admission decisions. This reflects a current problem with the US healthcare system where insurance coverage can influence care pathways. The model also identified several meaningful interactions between features (like "esi & arrivalmode" and "esi & age"), showing that the EBM is capturing more complex relationships than just individual feature effects.

PROMPT: Look through the shape functions for the features you've chosen to include in the model. Describe one or two unexpected patterns learned by the model. Reflect a bit on whether interpreting these charts feels rigorous to you.

Looking at the shape functions for some of the features included in the model, they appear to show some interesting patterns, with a couple particularly unexpected ones worth highlighting:

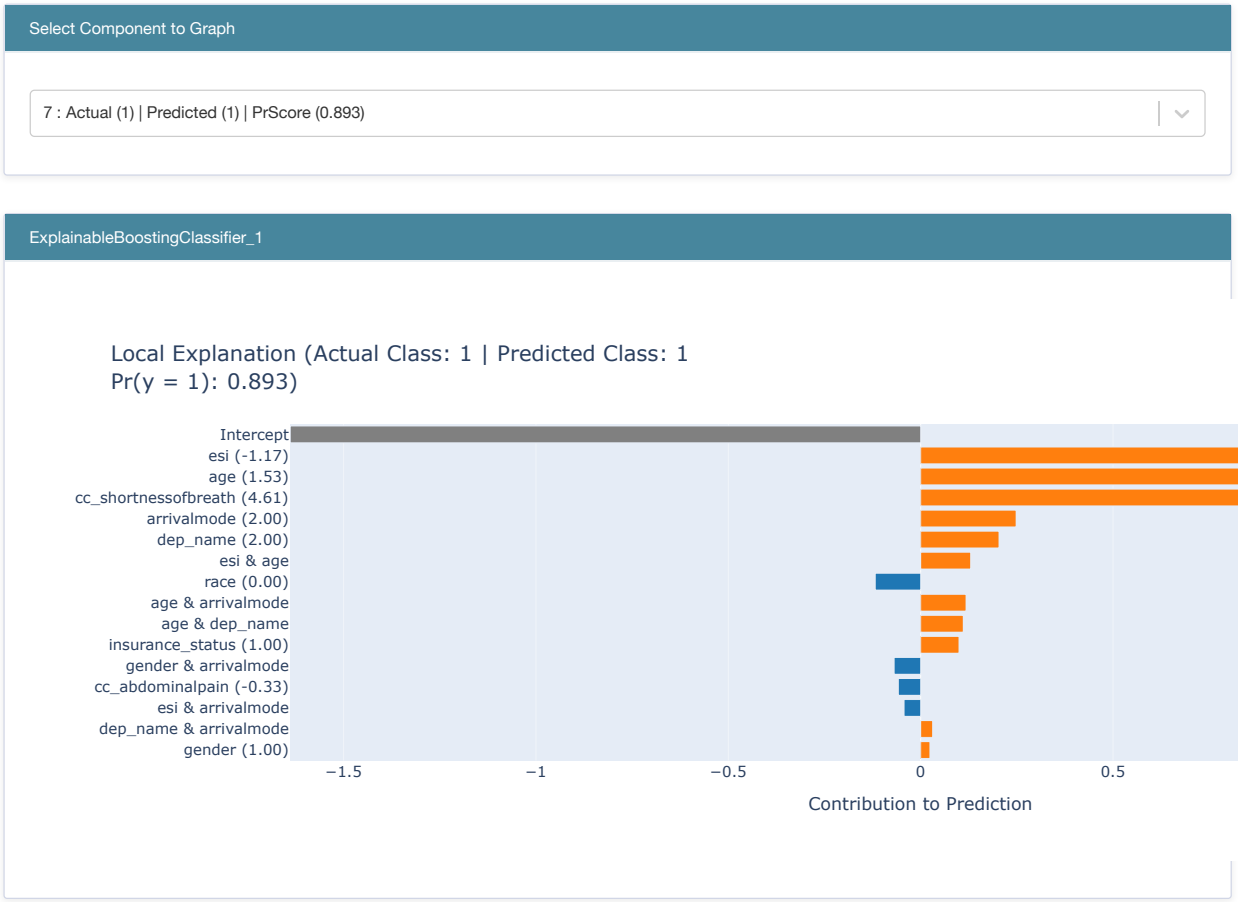
1. First, the insurance_status function shows a dramatic step pattern that shows significant disparities in admission likelihood based solely on insurance type. While most insurance categories have neutral to slightly negative effects on admission probability, there's a striking positive jump (approximately +5 score points) for status value 3.5-4.0. This suggests that patients with this particular insurance type are substantially more likely to be admitted regardless of their clinical presentation. This pattern raises important questions about potential systemic biases in healthcare delivery or differential admission thresholds for certain payer types. This non-clinical factor having such a pronounced effect was unexpected and merits further investigation from a health equity perspective.
2. Second, the relationship between ESI (Emergency Severity Index) and admission likelihood shows a clear step function rather than a smooth linear relationship. While it's expected that lower ESI scores (indicating higher acuity) would correlate with higher admission rates, the distinct plateaus

suggest that the model has identified threshold effects in clinical decision-making. This could reflect institutional protocols where patients below certain ESI thresholds are almost automatically admitted, while those above certain thresholds face a substantially different admission pathway. Similarly, the arrivalmode pattern shows an unexpectedly complex, non-monotonic relationship with several distinct steps across different modes, suggesting that arrival method influences admission decisions in ways that aren't simply related to patient acuity.

3. Regarding the rigor of interpreting these charts, there are both strengths and limitations. The visual representations from the global explanation output effectively communicate how each feature independently influences predictions, and the accompanying density plots provide us with important context about data distribution. However, these univariate views don't capture the interactions between features (though the model does include interaction terms as seen in the feature importance summary). Instead, the interaction terms are shown in the form of heatmaps. Additionally, the interpretation of encoded categorical variables requires domain knowledge about what each numeric value represents. Most concerning from a rigor perspective is that these charts show correlations, not causation, and may reflect systemic patterns in healthcare delivery rather than clinical necessity. While the interpretability is better than black-box models, these visualizations should be considered as starting points for investigation rather than definitive explanations of hospital admission decisions, in my opinion.

In [ ]:
```
# PROMPT: generate interpret's interactive widget for local explanations
local_explanation = ebm.explain_local(X_test, y_test)

show(local_explanation)
```



PROMPT: Pick two cases in the test set to examine using the dropdown menu. These should be one case where the model performs well and one case where the model performs poorly. In each case, describe how the model makes its prediction, focusing on which features contribute very differently to the two predictions. Please note if anything about the explanation doesn't make sense to you such that it might undermine your trust in the model.

Here, I picked two distinct cases from the dropdown menu, one with the model predicting an admission when the patient is actually admitted (true positive, so the model performs well) and the other with the model predicting a non-admission when the patient is actually admitted (false negative, so the model performs poorly).

Case 1: Successful Prediction of Hospital Admission

The first case shows a model success - correctly predicting admission with 89.3% confidence. Looking at the feature contributions shows a coherent clinical narrative. The patient's ESI score (-1.17, likely indicating high acuity) provides the strongest push toward admission with a substantial contribution of approximately 1.35 points. This is reinforced by three other significant factors: the patient's age (1.53, suggesting an older individual), presence of shortness of breath (4.61), and their arrival mode (2.00, likely via ambulance). Together, these create a consistent picture of a higher-risk patient. What's particularly compelling about this explanation is how it aligns with clinical reasoning - the combination of advanced age, respiratory distress, and high acuity appropriately drives the admission decision. The model also captures interaction effects, particularly between ESI and age, reinforcing the belief that older patients with higher acuity require more intensive care.

Case 2: Misclassified Admission Case

The second case shows where the model fails - incorrectly predicting discharge (with 74.9% confidence) for a patient who was actually admitted. The most notable difference between the cases is the role of age. Despite both patients being admitted in reality, age contributes positively in Case 1 but strongly negatively in Case 2 (-1.22, with approximately -0.6 contribution). Interestingly, both patients have identical ESI scores (-1.17), which contributes similarly positively toward admission in both cases, creating an internal contradiction. This suggests the model has learned that younger patients with the same ESI are less likely to require admission - potentially a valid statistical pattern but one that failed in this specific case. Additionally, Case 2 lacks the strong symptom indicators present in Case 1 (no significant contribution from shortness of breath), which might explain why the model underestimated the admission need.

What undermines trust in this explanation is the apparent contradiction between the ESI score strongly pushing toward admission while the age factor simultaneously pushes strongly toward discharge. This suggests the model may place too much weight on demographic factors relative to clinical indicators in borderline cases. Additionally, the absence of information about what specific clinical findings led to the actual admission decision raises questions about whether important variables are missing from the model entirely. This case highlights that while the model's transparency allows us to identify precisely where it went wrong, it also shows potential systematic gaps in how admission decisions are modeled, particularly for younger patients with serious conditions that might not be fully captured by the available features (after all, only 10 features are selected from the original dataset).

Export as PDF (Please Ignore)

In [ ]:
```
!dpkg --configure -a
```

In [ ]:
```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!pip3 install pypandoc
```

In [ ]:
```
!jupyter nbconvert --to HTML /content/DATA23700_Exercise9_Edward_v2.ipynb
```