**Predict Employee Turnover with scikit-learn**

Now, we you will apply decision trees and random forests using scikit-learn and Python to build an employee churn prediction application with interactive controls. We will accomplish this with the help of following tasks in the project:

* Introduction and Import Libraries
* Exploratory Data Analysis
* Encode Categorical Features
* Visualize Class Imbalance
* Create Training and Test Sets
* Build a Decision Tree Classifier with Interactive Controls
* Build a Random Forest Classifier with Interactive Controls
* Feature Importance Plots and Evaluation Metrics

While you are watching the instructions, you will get a cloud desktop with all the required software pre-installed and you will be able to follow along the instructions to complete the above-mentioned tasks.

Click on the button below to launch the project on Rhyme.

**Course Objectives**

In this course, we are going to focus on three learning objectives:

1. Apply decision trees and random forests with scikit-learn to classification problems.
2. Interpret decision trees and random forest models using feature importance plots.
3. Tune model hyperparamters to improve classification accuracy.
4. Create interactive, GUI components in Jupyter notebooks using widgets.

By the end of this course, you will be able to apply decision trees and random forest models using scikit-learn and Python to build an employee churn prediction application with interactive controls.

**Course Structure**

This course is divided into 4 parts:

1. Course Overview: This introductory reading material.
2. Predict Employee Turnover with scikit-learn: This is the hands-on project that we will work on in Rhyme.
3. Graded Quiz: This is the final assignment that you need to pass in order to successfully complete the course and earn a Course Certificate.

**Project Structure**

The hands on project on Predict Employee Churn with Decision Trees and Random Forests is divided into the following tasks:

**Task 1: Introduction and Import Libraries**

* Introduction to the data set and the problem overview.
* See a demo of the final product you will build by the end of this project.
* Introduction to the Rhyme interface.
* Import essential modules and helper functions from [NumPy](https://numpy.org/), [Matplotlib](https://matplotlib.org/), and [scikit-learn](https://scikit-learn.org/stable/).

**Task 2: Exploratory Data Analysis**

* Load the employee dataset using [pandas](https://pandas.pydata.org/)
* Explore the data visually by graphing various features against the target with Matplotlib.

**Task 3: Encode Categorical Features**

* The dataset contains two categorical variables: Department and Salary.
* Create dummy encoded variables for both categorical variables.

**Task 4: Visualize Class Imbalance**

* Use Yellowbrick's [Class Balance](https://www.scikit-yb.org/en/latest/api/target/class_balance.html) visualizer to create a frequency plot of both classes.
* The presence or absence of a class balance problem will inform your sampling strategory while creating training and validation sets.

**Task 5: Create Training and Validation Sets**

* Split the data into a 80/20 training/validation split.
* Use a stratified sampling strategy

**Tasks 6 & 7: Build a Decision Tree Classifier with Interactive Controls**

* Use the [interact](https://ipywidgets.readthedocs.io/en/latest/examples/Using%20Interact.html) function to automatically create UI controls for function arguments.
* Build and train a decision tree classifier with scikit-learn.
* Calculate the training and validation accuracies.
* Display the fitted decision tree graphically.

**Task 8: Build a Random Forest Classifier with Interactive Controls**

* Use the [interact](https://ipywidgets.readthedocs.io/en/latest/examples/Using%20Interact.html) function again to automatically create UI controls for function arguments.
* To overcome the variance problem associated with decision trees, build and train a random forests classifier with scikit-learn.
* Calculate the training and validation accuracies.
* Display a fitted tree graphically.

**Task 9: Feature Importance Plots and Evaluation Metrics**

* Many model forms describe the underlying impact of features relative to each other.
* Decision Tree models and Random Forest in scikit-learn, feature\_importances\_ attribute when fitted.
* Utilize this attribute to rank and plot the features.