Orange Assignment

Classification, Regression & Clustering

1 Part 1 Classification: Employee Churn Management

Employee churn, often referred to as attrition or turnover, is a critical concern for organizations across various industries. It entails the departure of valuable talent from an organization, which not only incurs significant financial costs but also disrupts productivity and can have a negative impact on team morale. Predicting employee churn has thus become a vital task for organizations seeking to proactively address this issue. This problem is inherently a binary classification task, it involves categorizing employees into two distinct groups: those likely to churn and those likely to stay with the organization, making it a quintessential example of binary classification in machine learning.

1.1 Objective

This assignment aims to develop classification models (KNN, Decision Tree and Random Forests) using Orange to predict employee churn. The primary objective is to utilize historical employee data to build a predictive model that can accurately identify individuals at risk of leaving the organization. By doing so, organizations can implement targeted retention strategies and interventions, ultimately reducing the overall churn rate and fostering a more stable and motivated workforce.

1.2 Dataset

For this assignment, you will be working with the churn.csv dataset, which focuses on employee churn prediction. The dataset contains information on various attributes related to employee performance and experiences within the company. This dataset comprises 14,999 samples, featuring 10 attributes. Among these, 6 are of integer type, 2 are of float type, and 2 are categorical objects. It's worth noting that none of the variables have missing or null values.
The attributes can be described in detail as follows:

1. `satisfaction_level`: This represents the level of employee satisfaction, ranging from 0 to 1.
2. `last_evaluation`: This indicates the performance evaluation conducted by the employer, also on a scale of 0 to 1.
3. `number_projects`: This denotes the number of projects assigned to an employee.
4. `average_monthly_hours`: This reflects the average number of hours worked by an employee in a month.
5. `time_spent_company`: This signifies the number of years an employee has spent in the company, indicating their level of experience.
6. `work_accident`: This binary attribute indicates whether an employee has experienced a work-related accident or not.
7. `promotion_last_5years`: This binary attribute indicates whether an employee has received a promotion in the last 5 years or not.
8. `Departments`: This categorical attribute represents the employee's working department or division.
9. `Salary`: This categorical attribute specifies the salary level of the employee, categorized as low, medium, or high.
10. `left`: This is the target variable, indicating whether the employee has left the company (1) or not (0).

The main focus of this assignment is to predict employee churn, with the `left` variable serving as the key target for classification. This prediction will be based on the information provided by the other attributes in the dataset.

1.3 Tasks

1.3.1 Task 1: Data Exploration

1. Import the dataset into Orange. Use `CSV file import` widget to import the churn.csv data frame. Connect the widget to the `Data Table` widget to view the imported data frame.

2. Display basic statistics about the dataset (e.g., number of instances, attributes, data types). Connect the `CSV file import` widget to the `Feature Statistics` widget and click on it to view the statistics.

3. Explore the distribution of the target variable (left vs not left). Connect the `CSV file import` widget to the `Distributions` widget and click on it to view the distributions (select the left variable).

1.3.2 Task 2: Data Preprocessing

1. Handle any missing values in the dataset (if applicable). No missing values are there as seen in the `Statistics` widget.

2. Use the select column widget to identify the target variable (left), keeping the remaining 9 as the features. Connect the `CSV file import` widget to the `Select Columns` widget and click on it to select left as the target variable.

3. Split the data into training and testing sets (e.g., 80% training, 20% testing) using Data Sampler. Connect the `Select Columns` widget to the `Data Sampler` widget and click on it to adjust the sampling type to be fixed proportion at 80%.

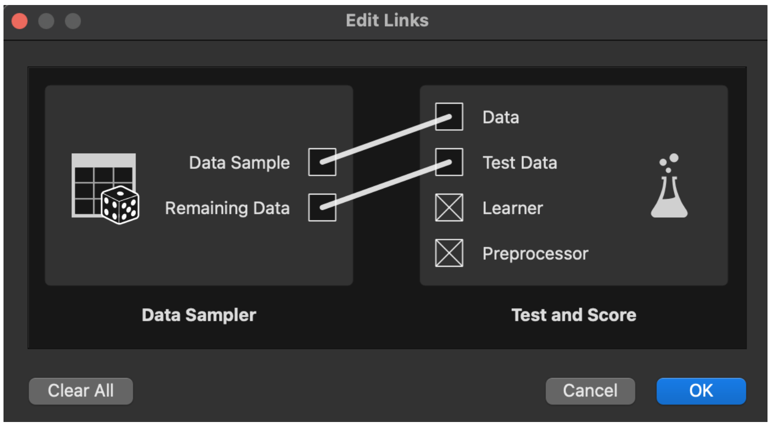
1.3.3 Task 3: k-NN Model

1. Apply the k-NN algorithm.

2. Evaluate the performance of the k-NN model on the data (e.g., accuracy, confusion matrix). You may connect both the data sampler and kNN model to the test and score widget. Connect the following to the `Test and Score` widget:

   (a) `Data Sampler` widget. Two Links: Data Sample to Data; Remaining Data to Test Data.

   (b) `kNN` widget: One Link: Learner to Learner.



   Make sure to click on the `Test and Score` widget and select "Test on Test Data".

3. Explore the impact of changing the distance metric (by clicking on the kNN Widget) on the model's predictions. Click on the `kNN` widget to adjust the type of regularization and the results should be available immediately.

4. Interpret the results and provide insights.

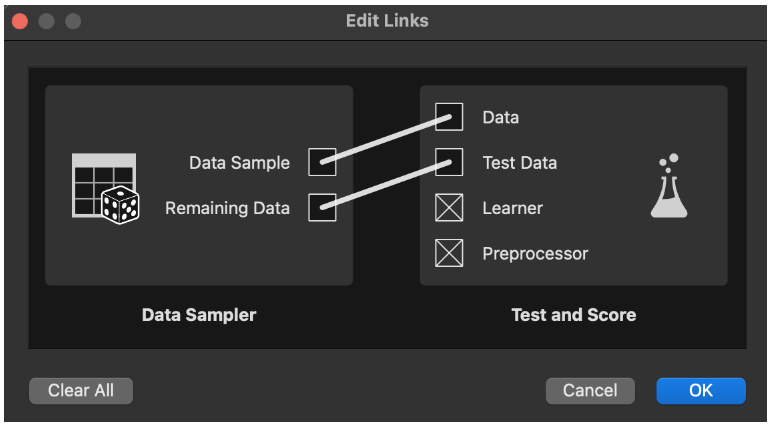
1.3.4 Task 4: Decision Tree Classifier

1. Apply the Decision Tree algorithm.

2. Evaluate the performance of the Decision Tree model on the data (accuracy, classification error). You may connect both the data sampler and Tree model to the test and score widget. Connect the following to the `Test and Score` widget:

(a) `Data Sampler` widget. Two Links: Data Sample to Data; Remaining Data to Test Data.

(b) `Tree` widget: One Link: Learner to Learner.



Make sure to click on the `Test and Score` widget and select “Test on Test Data”.

3. Try to change the value of maximal tree depth by clicking on the Tree widget and observe how it impacts the algorithm’s performance. Click on the `Tree` widget to adjust the value of maximal tree depth and the results should be immediately available under the `Test and Score` widget.

4. Interpret the results and provide insights.

1.3.5 Task 5: Random Forest Classifier

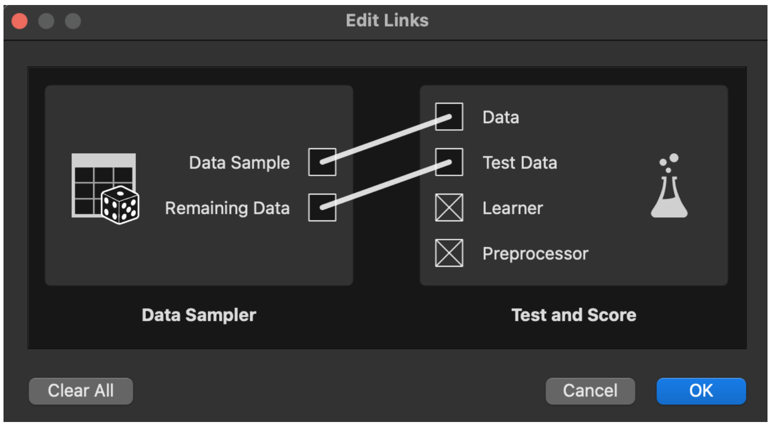
In this section, we will try to predict churn using Random Forest.

1. Apply the Random Forest algorithm.

2. Evaluate the performance of the Random Forest model on the data (accuracy, classification error). You may connect both the data sampler and Random Forest model to the test and score widget. Connect the following to the `Test and Score` widget:

   (a) `Data Sampler` widget. Two Links: Data Sample to Data; Remaining Data to Test Data as above.

   (b) `Random Forest` widget: One Link: Learner to Learner.



Make sure to click on the `Test and Score` widget and select “Test on Test Data”.

3. Try to change the number of trees in the Random Forest widget and observe how it impacts the algorithm’s performance. Click on the `Random Forest` widget to adjust the number of trees and the results should be immediately available under the `Test and Score` widget.

4. Interpret the results and provide insights.

2 Part 2 Regression: Automobile MPG Prediction

Predicting fuel efficiency, measured as miles per gallon (mpg), is a crucial task in the automotive industry. This assignment focuses on developing a linear regression model using Orange to predict the MPG of cars based on various features. The goal is to leverage the Auto MPG dataset, which contains information on displacement, cylinders, horsepower, weight, acceleration, model year, and origin, to build an accurate predictive model.

2.1 Objective

This assignment aims to create a linear regression model using Orange to predict the fuel efficiency (mpg) of cars. By utilizing historical data on various car features, the objective is to build a model that can effectively estimate the mpg for new cars. This predictive capability is valuable for designing fuel-efficient vehicles and making informed decisions in the automotive industry.

2.2 Dataset

For this assignment, the Auto MPG dataset will be used, featuring information on different car attributes. The dataset comprises 398 samples and includes the following attributes:

1. `displacement`: Continuous feature representing the engine displacement of the car.
2. `mpg`: Continuous target variable representing the miles per gallon.
3. `cylinders`: Integer feature indicating the number of cylinders in the car's engine.
4. `horsepower`: Continuous feature representing the horsepower of the car.
5. `weight`: Continuous feature indicating the weight of the car.
6. `acceleration`: Continuous feature representing the acceleration performance of the car.
7. `model_year`: Integer feature indicating the manufacturing year of the car.
8. `origin`: Integer feature indicating the origin of the car.
9. `car_name`: Categorical ID feature representing the name of the car.

2.3 Tasks

2.3.1 Task 1: Data Exploration

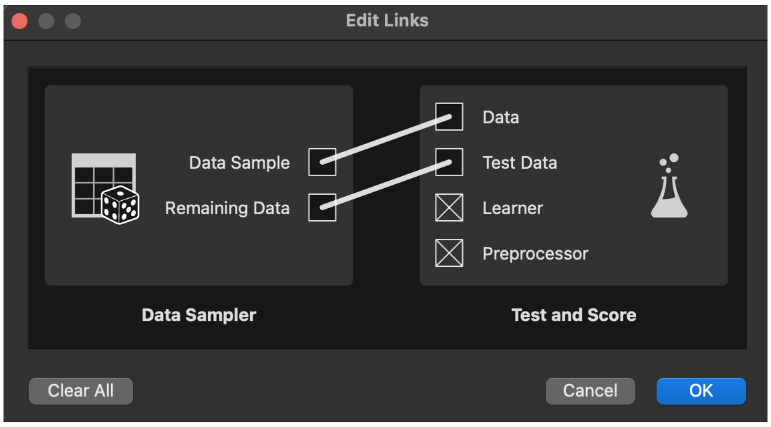
1. Import the dataset into Orange. Use `CSV file import` widget to import the auto-mpg.csv data frame. Connect the widget to `Data Table` widget to view the imported data frame.
2. Display basic statistics about the dataset (e.g., number of instances, attributes, data types). Connect the `Datasets` widget to the `Feature Statistics` widget and click on it to view the statistics.
3. Explore the distribution of the target variable (mpg). Connect the `Datasets` widget to the `Distributions` widget and click on it to view the distributions (select the mpg variable).

2.3.2 Task 2: Data Preprocessing

1. Handle any missing values in the dataset (if applicable). You may need to use `Impute` widget to handle the missing values of horsepower feature with average value.
2. Use the select column widget to identify the target variable (mpg), keeping the remaining 7 (excluding `car_names`) as the features. Connect the `CSV file import` widget to the `Select Columns` widget and click on it to select left as the target variable.
3. Split the data into training and testing sets (e.g., 80% training, 20% testing) using the Data Sampler. Connect the `Datasets` widget to the `Data Sampler` widget and click on it to adjust the sampling type to be fixed proportion at 80%.

2.3.3 Task 3: Linear Regression

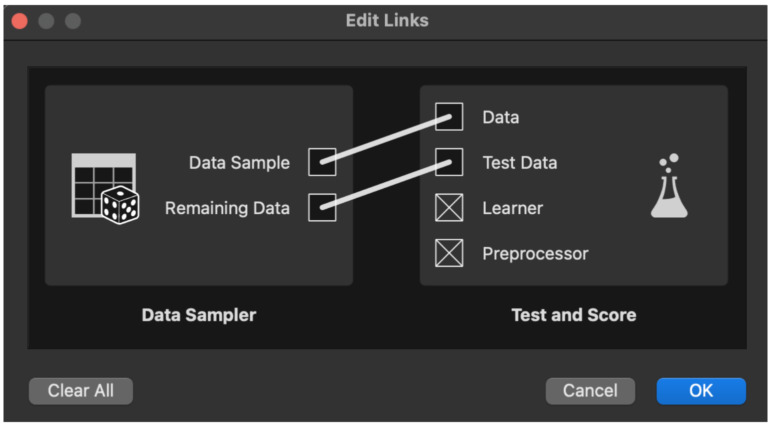
1. Apply the Linear Regression algorithm.
2. Evaluate the performance of the linear regression model on the data (e.g., mean squared error). You may connect both the data sampler and linear regression model to the test and score widget. Connect the following to the `Test and Score` widget:
   (a) `Data Sampler` widget. Two Links: Data Sample to Data; Remaining Data to Test Data.
   (b) `Linear Regression` widget: One Link: Learner to Learner.



   Make sure to click on the `Test and Score` widget and select Test on Test Data.
3. Explore the impact of regularization (by clicking on the Linear Regression Widget) on the model's predictions. Click on the `Linear Regression` widget to adjust the type of regularization and the results should be available immediately.
4. Interpret the results and provide insights.

2.3.4 Task 3: KNN Regression

1. Apply the KNN Regression algorithm.
2. Evaluate the performance of the KNN regression model on the data (e.g., mean squared error). You may connect both the data sampler and KNN model to the test and score widget. Connect the following to the `Test and Score` widget:
   (a) `Data Sampler` widget. Two Links: Data Sample to Data; Remaining Data to Test Data.
   (b) `KNN` widget: One Link: Learner to Learner.



   Make sure to click on the `Test and Score` widget and select Test on Test Data.
3. Explore the impact of changing the number of neighbors (by clicking on the KNN Widget) on the model's predictions. Click on the `KNN` widget to adjust the number of neighbors and the results should be available immediately.
4. Interpret the results and provide insights.

3 Part 3 Clustering: Iris Species Classification and Analysis

The Iris dataset provides a classic setting for exploring machine learning techniques. This assignment introduces unsupervised learning through k-means clustering. The goal is to leverage the features of sepal length, sepal width, petal length, and petal width to explore natural groupings in the data.

3.1 Objective

Explore unsupervised learning using k-means clustering to uncover natural groupings within the Iris dataset based solely on its features (sepal length, sepal width, petal length, and petal width), without relying on the actual species labels.

3.2 Dataset

For this assignment, the Iris dataset will be used, featuring information on various attributes of iris flowers. The dataset comprises 150 samples and includes the following attributes:

1. `sepal_length`: Continuous feature representing the sepal length of the iris flower.
2. `sepal_width`: Continuous feature representing the sepal width of the iris flower.
3. `petal_length`: Continuous feature representing the petal length of the iris flower.
4. `petal_width`: Continuous feature representing the petal width of the iris flower.
5. `species`: Categorical target variable indicating the species of the iris flower (Setosa, Versicolor, or Virginica).

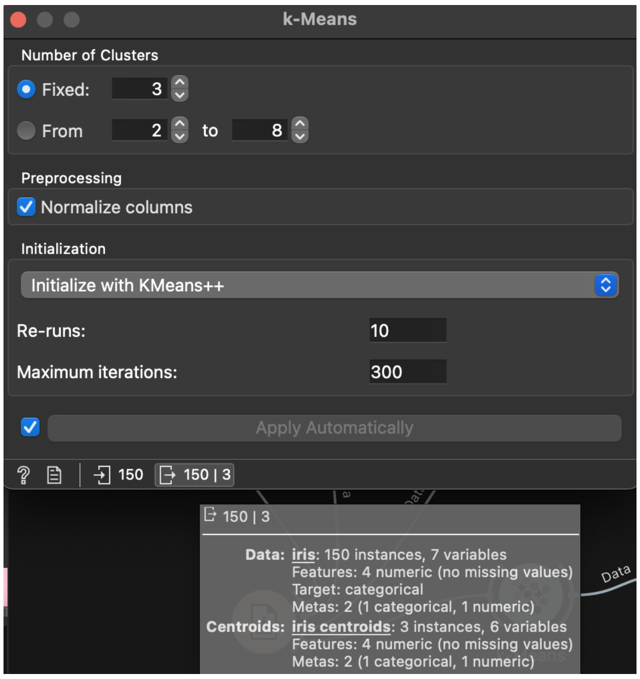
3.3 Tasks

3.3.1 Task 1: Data Exploration

1. Import the dataset into Orange. Use `CSV file import` widget to import the iris.csv data frame. Connect the widget to `Data Table` widget to view the imported data frame.
2. Display basic statistics about the dataset (e.g., number of instances, attributes, data types). Connect the `Datasets` widget to the `Feature Statistics` widget and click on it to view the statistics.

3.3.2 Task 2: Unsupervised Learning with k-Means Clustering

1. Implement k-means clustering using the k-means widget in Orange. Simply connect the `Datasets` widget to the `k-Means` widget and click on it to set the number of clusters to 3.
2. Obtain the clusters and their centroids and visualize them with sepal width and length along the x and y axis. Click on the `k-Means` widget and go to results to view the cluster centroids.



k-Means

Number of Clusters

● Fixed: 3

○ From 2 to 8

Preprocessing

✓ Normalize columns

Initialization

Initialize with KMeans++

Re-runs: 10

Maximum iterations: 300

✓ Apply Automatically

150 | 150 | 3

150 | 3

Data: iris: 150 instances, 7 variables
Features: 4 numeric (no missing values)
Target: categorical
Metas: 2 (1 categorical, 1 numeric)

Centroids: iris centroids: 3 instances, 6 variables
Features: 4 numeric (no missing values)
Metas: 2 (1 categorical, 1 numeric)

To obtain the visualization, simply connect the `k-Means` widget to the `Scatter Plot` widget and choose the x and y axis of your choice.

3. Interpret the results and provide insights into the structure of the data.