List 5
Data Analysis and Machine Learning

Description:

Welcome to the year 2912, where your data science skills are needed to solve a cosmic mystery. We've received a transmission from four lightyears away and things aren't looking good.

The Spaceship Titanic was an interstellar passenger liner launched a month ago. With almost 13,000 passengers on board, the vessel set out on its maiden voyage transporting emigrants from our solar system to three newly habitable exoplanets orbiting nearby stars.

While rounding Alpha Centauri en route to its first destination—the torrid 55 Cancri E—the unwary Spaceship Titanic collided with a spacetime anomaly hidden within a dust cloud. Sadly, it met a similar fate as its namesake from 1000 years before. Though the ship stayed intact, almost half of the passengers were transported to an alternate dimension!

To help rescue crews and retrieve the lost passengers, you are challenged to predict which passengers were transported by the anomaly using records recovered from the spaceship's damaged computer system.

Help save them and change history!

File and Data Field Descriptions

train.csv - Personal records for about two-thirds (~8700) of the passengers, to be used as training data

- *PassengerId* - A unique Id for each passenger. Each Id takes the form gggg_pp where gggg indicates a group the passenger is travelling with and pp is their number within the group. People in a group are often family members, but not always.
- *HomePlanet* - The planet the passenger departed from, typically their planet of permanent residence.
- *CryoSleep* - Indicates whether the passenger elected to be put into suspended animation for the duration of the voyage. Passengers in cryosleep are confined to their cabins.
- *Cabin* - The cabin number where the passenger is staying. Takes the form deck/num/side, where side can be either P for Port or S for Starboard.
- *Destination* - The planet the passenger will be debarking to.
- *Age* - The age of the passenger.
- *VIP* - Whether the passenger has paid for special VIP service during the voyage.
- *RoomService, FoodCourt, ShoppingMall, Spa, VRDeck* - Amount the passenger has billed at each of the Spaceship Titanic's many luxury amenities.
- *Name* - The first and last names of the passenger.
- *Transported* - Whether the passenger was transported to another dimension. This is the target, the column you are trying to predict.

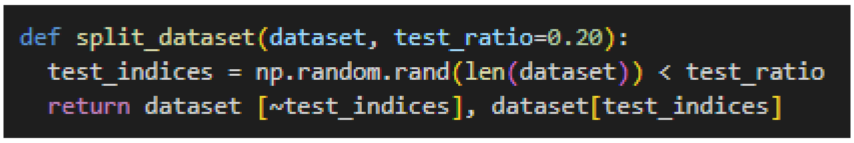
1. Create a Google Colab notebook:
    - Go to https://colab.research.google.com and log in with your student email.
    - Create a new notebook
    - Add the data to the session memory

    Install and import necessary libraries:

    - Install tensorflow_decision_forests
    - Import:
        - Tensorflow as tf
        - Tensorflow_decision_forests as tfdf
        - Pandas as pd
        - Numpy as np
        - Seaborn as sns
        - Matplotlib.pyplot as plt

    Load the dataset:

    - Read the csv file (you can use the read_csv function from pandas)
    - Display 5 first entries of the dataset to check whether everything loaded correctly
2. Explore the dataset:
    - Explore the statistics of the dataset using the describe() and info() methods
    - Make a bar chart for label column: Transported
    - Plot all the numerical columns and their value counts (sns.histplot(), preferably make one figure with subplots)
3. Prepare the dataset:
    - We don't need to know Name and PassengerId, so drop these columns
    - Check for the missing values (dataset_df.isnull().sum().sort_values(ascending=False))
    - TF-DF does not support boolean fields, convert those fields into int. To account for the missing values in the boolean fields, replace them with zero (use fillna() method)
    - Replace null value entries with zero for numerical columns
    - Since, TF-DF cannot handle boolean columns, adjust the labels in column Transported to convert them into the integer format that TF-DF expects
    - Convert the boolean fields CryoSleep and VIP to int
    - The value of column Cabin is a string with the format Deck/Cabin_num/Side. Split this column into 3: Deck, Cabin_num and Side and remove the original column
    - Split the data into training and testing datasets. Use the following function:

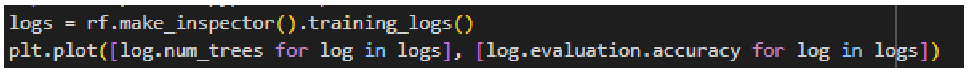


```python
def split_dataset(dataset, test_ratio=0.20):
  test_indices = np.random.rand(len(dataset)) < test_ratio
  return dataset [~test_indices], dataset[test_indices]
```

    - Convert the dataset from Pandas format (pd.Dataframe) into TensorFlow Datasets format (tf.data.Dataset). Use tfdf.keras.pd_dataframe_to_tf_dataset() function
4. Select a Model:
    - There are several tree-based models for you to choose from (use tfdf.keras.get_all_models() to see the list). To start, you will work with a Random Forest.
5. Configure and train the model:
    - Create a random forest, by default the model is set to train for a classification task
    - Include a list of eval metrics (use .compile(metrics=["accuracy])

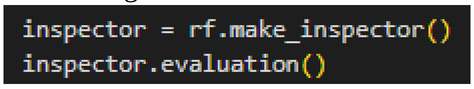
- Train the model
- Plot the trained model (tfdf.model_plotter.plot_model_in_colab(rf, tree_idx=0, max_depth=3))

6. Evaluate the model on the Out of bag (OOB) data and validation dataset



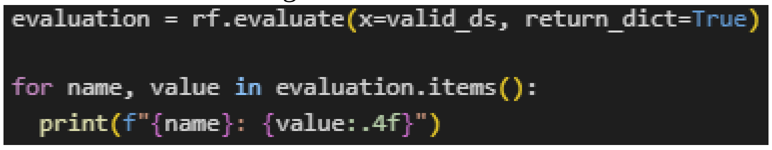
```
logs = rf.make_inspector().training_logs()
plt.plot([log.num_trees for log in logs], [log.evaluation.accuracy for log in logs])
```

- Plot the accuracy evaluated on the out-of-bag dataset according to the number of trees in the model.
- See some general statistics on the OOB dataset:



```
inspector = rf.make_inspector()
inspector.evaluation()
```

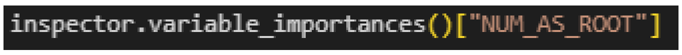
-
- Run the evaluation using the validation dataset



```
evaluation = rf.evaluate(x=valid_ds, return_dict=True)

for name, value in evaluation.items():
  print(f"{name}: {value:.4f}")
```

-
- Display the important features:



```
inspector.variable_importances()["NUM_AS_ROOT"]
```