List of Functions for the Midterm/Final

This document summarizes some important functions you need to memorize and understand for the purposes of the midterm and final exams in this course. This does not mean to overlook the rest of the material, this summary is meant to help you prepare for the code-related questions that you will find in the exam. For each of the listed functions, **here or in the rest of the code as illustrated in the class,** you need to:

1. Memorize the function, its name, input and output parameters
2. Understand the purpose of the function (i.e. what it is meant to do)
3. Understand the input and output parameters involved in the function.

PLEASE NOTE THAT THIS DOES NOT MEAN THE REST OF CODE SNIPPETS (OR OTHER PARTS OF THE MATERIAL) WE COVERED IN CLASS ARE NOT IMPORTANT. YOU MUST STUDY ALL THE MATERIAL COVERED IN CLASS. THIS SUMMARY IS JUST TO HELP YOU STUDY FOR CODE RELATED QUESTIONS.

If you have any questions do not hesitate to reach out either via Teams Course Channel, or Teams Chat.

Good Luck

I) Azure Hands-on Example

Function signature	Source
- `experiment = Experiment(workspace=ws, name="diabetes-experiment")`	from azureml.core import Experiment
- `X_train, X_test, y_train, y_test = train_test_split(x_df, y_df, test_size=0.2, random_state=66)`	**from** sklearn.model_selection **import** train_test_split
- `run = experiment.start_logging()` - `run.log(KEY, VALUE)` - `run.upload_file(name=model_name, path_or_stream=filename)` - `run.complete()`	Experiment object
- `model = Ridge(alpha=alpha)` - `model.fit(X=X_train, y=y_train)` - `y_pred = model.predict(X=X_test)`	Ridge → sklearn.linear_model
- `rmse = math.sqrt(mean_squared_error(y_true=y_test, y_pred=y_pred))`	- **import** math - **from** sklearn.metrics **import** mean_squared_error
- `model_name = STRING + ".pkl"` - `filename = "outputs/" + model_name` - `joblib.dump(value=model, filename=filename)`	**import** joblib

Also for the Azure example, you need to understand the main components of the main experiment page. A sample is illustrated in Fig[1]. 1.

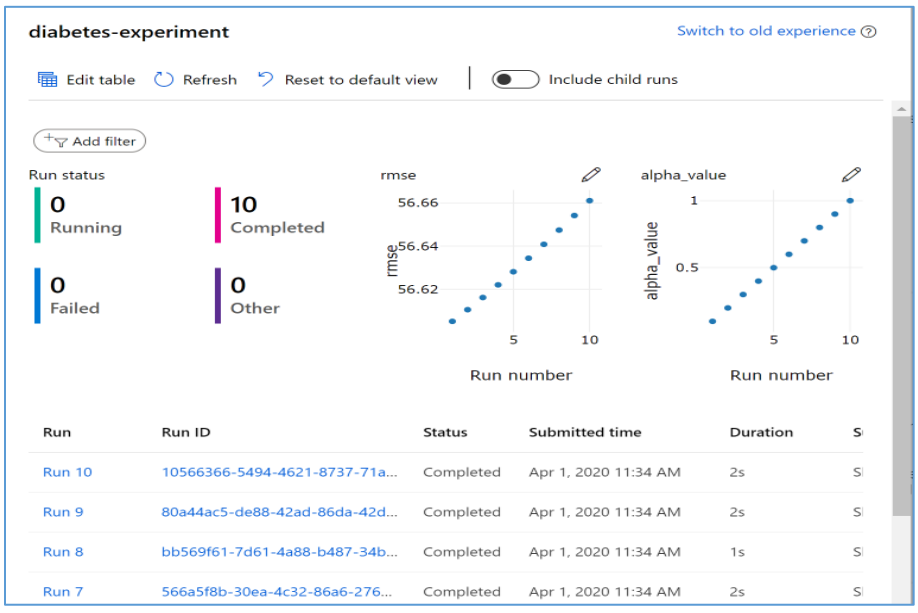

Fig. 1. Sample Experiments Page in Azure ML Studio.

[1] Columns beyond Duration the main experiment page are irrelevant for the purposes of this course.

II) Google Colab Hands-on Example (Random forest)

Function signature	Source
`wine_data = datasets.load_wine()`	`from sklearn import datasets`
- `df = pd.DataFrame(wine_data["data"], columns=wine_data["feature_names"])` - `df.head()` - `target = wine_data["target"]`	`import pandas as pd`
- `X_train, X_test, y_train, y_test = model_selection.train_test_split(df, target, test_size=0.2, random_state=0)`	**from** sklearn.model_selection **import** train_test_split
- `classifier = ensemble.RandomForestClassifier()` - `model = classifier.fit(X_train, y_train)`	`from sklearn import ensemble`
- `y_pred = model.predict(X_test)` - `report = metrics.classification_report(y_test, y_pred)`	`from sklearn import metrics`

III) Google Colab Hands-on Example (TensorFlow)

Function signature	Source
- `mnist = tf.keras.datasets.mnist` - `(x_train, y_train), (x_test, y_test) = mnist.load_data()`	`import tensorflow as tf`

- `model = tf.keras.models.Sequential([`
`tf.keras.layers.Flatten(input_shape=(28, 28)),`
`tf.keras.layers.Dense(128, activation='relu'),`
`tf.keras.layers.Dropout(0.2),`
`tf.keras.layers.Dense(10)`
`])`
	`import tensorflow as tf`

- `predictions = model(x_train[:1]).numpy()` - `tf.nn.softmax(predictions).numpy()`	`import tensorflow as tf`
- `loss_fn = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)`	`import tensorflow as tf`
- `model.compile(optimizer='adam',`	

loss=loss_fn, metrics=['accuracy'])	
- model.fit(x_train, y_train, epochs=5) - model.evaluate(x_test, y_test, verbose=2)	
- probability_model = tf.keras.Sequential([model, tf.keras.layers.Softmax()]) - probability_model(x_test[:5])	import tensorflow as tf

DOCUMENT ENDS HERE