TensorFlow

TensorFlow API r1.4

Module: tf.contrib.learn

Defined in `tensorflow/contrib/learn/__init__.py`.

High level API for learning.

See the Learn (contrib) guide.

Modules

`datasets` module: Dataset utilities and synthetic/reference datasets.

`graph_actions` module: High level operations on graphs.

`head` module: Abstractions for the head(s) of a model.

`io` module: Tools to allow different io formats.

`learn_runner` module: Utilities to run and tune an Experiment.

`models` module: Various high level TF models.

`monitors` module: Monitors instrument the training process.

`ops` module: Various TensorFlow Ops.

`preprocessing` module: Preprocessing tools useful for building models.

`utils` module: TensorFlow Learn Utils.

Classes

`class BaseEstimator`: Abstract BaseEstimator class to train and evaluate TensorFlow models.

`class DNNClassifier`: A classifier for TensorFlow DNN models.

`class DNNEstimator`: A Estimator for TensorFlow DNN models with user specified _Head.

`class DNNLinearCombinedClassifier`: A classifier for TensorFlow Linear and DNN joined training models.

`class DNNLinearCombinedEstimator`: An estimator for TensorFlow Linear and DNN joined training models.

`class DNNLinearCombinedRegressor`: A regressor for TensorFlow Linear and DNN joined training models.

`class DNNRegressor`: A regressor for TensorFlow DNN models.

`class DynamicRnnEstimator`

`class Estimator` : Estimator class is the basic TensorFlow model trainer/evaluator.

`class Evaluable` : Interface for objects that are evaluatable by, e.g., `Experiment` .

`class Experiment` : Experiment is a class containing all information needed to train a model.

`class ExportStrategy` : A class representing a type of model export.

`class Head` : Interface for the head/top of a model.

`class InputFnOps` : A return type for an input_fn.

`class KMeansClustering` : An Estimator for K-Means clustering.

`class LinearClassifier` : Linear classifier model.

`class LinearEstimator` : Linear model with user specified head.

`class LinearRegressor` : Linear regressor model.

`class MetricSpec` : MetricSpec connects a model to metric functions.

`class ModeKeys` : Standard names for model modes.

`class ModelFnOps` : Ops returned from a model_fn.

`class NanLossDuringTrainingError`

`class NotFittedError` : Exception class to raise if estimator is used before fitting.

`class PredictionKey`

`class ProblemType` : Enum-like values for the type of problem that the model solves.

`class RunConfig` : This class specifies the configurations for an `Estimator` run.

`class SKCompat` : Scikit learn wrapper for TensorFlow Learn Estimator.

`class SVM` : Support Vector Machine (SVM) model for binary classification.

`class TaskType`

`class Trainable` : Interface for objects that are trainable by, e.g., `Experiment` .

Functions

`LogisticRegressor(...)` : Builds a logistic regression Estimator for binary classification.

`binary_svm_head(...)` : Creates a `Head` for binary classification with SVMs.

`build_parsing_serving_input_fn(...)` : Build an input_fn appropriate for serving, expecting fed tf.Examples.

`evaluate(...)` : Evaluate a model loaded from a checkpoint. (deprecated)

`extract_dask_data(...)` : Extract data from dask.Series or dask.DataFrame for predictors.

`extract_dask_labels(...)` : Extract data from dask.Series or dask.DataFrame for labels.

`extract_pandas_data(...)` : Extract data from pandas.DataFrame for predictors.

`extract_pandas_labels(...)` : Extract data from pandas.DataFrame for labels.

`extract_pandas_matrix(...)` : Extracts numpy matrix from pandas DataFrame.

`infer(...)` : Restore graph from `restore_checkpoint_path` and run `output_dict` tensors. (deprecated)

`infer_real_valued_columns_from_input(...)` : Creates `FeatureColumn` objects for inputs defined by input `x` .

`infer_real_valued_columns_from_input_fn(...)` : Creates `FeatureColumn` objects for inputs defined by `input_fn` .

`make_export_strategy(...)` : Create an ExportStrategy for use with Experiment.

`multi_class_head(...)` : Creates a `Head` for multi class single label classification.

`multi_head(...)` : Creates a MultiHead stemming from same logits/hidden layer.

`multi_label_head(...)` : Creates a Head for multi label classification.

`no_op_train_fn(...)`

`poisson_regression_head(...)` : Creates a `Head` for poisson regression.

`read_batch_examples(...)` : Adds operations to read, queue, batch `Example` protos.

`read_batch_features(...)` : Adds operations to read, queue, batch and parse `Example` protos.

`read_batch_record_features(...)` : Reads TFRecord, queues, batches and parses `Example` proto.

`read_keyed_batch_examples(...)` : Adds operations to read, queue, batch `Example` protos.

`read_keyed_batch_examples_shared_queue(...)` : Adds operations to read, queue, batch `Example` protos.

`read_keyed_batch_features(...)` : Adds operations to read, queue, batch and parse `Example` protos.

`read_keyed_batch_features_shared_queue(...)` : Adds operations to read, queue, batch and parse `Example` protos.

`regression_head(...)` : Creates a `Head` for linear regression.

`run_feeds(...)` : See run_feeds_iter(). Returns a `list` instead of an iterator. (deprecated)

`run_n(...)` : Run `output_dict` tensors `n` times, with the same `feed_dict` each run. (deprecated)

`train(...)` : Train a model. (deprecated)

Last updated November 2, 2017.

Stay Connected

Blog

GitHub

Twitter

Support

Issue Tracker

Release Notes

Stack Overflow