Compute Science Department
CS672 – Introduction to Deep Learning (CRN# 72463)
Fall 2023

Project #1 / Due 16-Oct-2023
This exercise will guide you through two important tasks in any Machine Learning / Deep Learning engagement:

- ➢ **Perform Exploratory Data Analysis**
- ➢ **Device a ML/DL Model (Regression Analysis)**

Performing Exploratory Data Analysis (EDA) on data is of paramount important for every Data Scientist / Data Analyst. Exploratory Data Analysis is often used to uncover various patterns present in the data and to draw conclusions from it. EDA is the core part when it comes to developing a Machine Learning model. This takes place through analysis and visualization of the data which will be fed to the Machine Learning Model. A Machine Learning Model is as good as the training data - you must understand it if you want to understand your model.

Prior commencing your efforts on coding, you must install the following libraries:

- pip install -q tensorflow_data_validation [visualization] (**)
  - https://pypi.org/project/tensorflow-data-validation/
- pip install apache-beam [interactive]
  - https://beam.apache.org/get-started/quickstart-py/
  - https://pypi.org/project/apache-beam/
- Install the GraphViz library
  - https://www.graphviz.org/download/

(A)
Perform an **explanatory data analysis** (**EDA**) on **Median House Prices** from California districts derived from the 1990 census. Besides the need to build a model based on the data provided, you are asked to look for issues in the data and find correlation among the various variables in order to improve median house price predictions.

Write **Python** scripts in order to complete the following tasks along with their output. All work should be done and submitted in a single **Notebook (Jupyter or Colab).**

1) Prep the data in order to be ready to be fed to a model.
   Look for missing, null, NaN records.
   Find outliers.
   Transform data – all entries should be numeric.
2) List all types of data, numeric, categorical, etc.
3) Perform EDA on data
Utilize both:
- Classic approach in EDA (Pandas, Numpy libraries)
- The TFDV (TensorFlow Data Validation) module with the powerful graphical statistics generated (apache beam library…)

Present dependencies and correlations among the various features in the data.
List the most variables (Feature Importance) that will affect the target label.

(B)
Build a **Deep Learning model** (based on **Tensorflow's regression models**) that provides reliable and improved accuracy on predicting median house prices.

Perform **regression modeling analysis** on the ready-to-be-fed data into the following Neural Network based algorithms:
1) **MLP** (Multi-Layer Perceptron)
2) **Linear Regression** (TF/Keras Sequential model w/ no hidden layers)
3) **DNN** (Deep Neural Network with at least 2 hidden layers)
Split your dataset into training and validation datasets in 80%/20% ratio.
(Note: Since dataset is time-sensitive, be extra careful on splitting…)

At the **compiling** phase, use:

- **loss function:** Mean Square Error (MSE), Mean Absolute Error (MAE).
- **optimizer**: SGD, Adam, RMSProp.
    - Use various values for **learning rate (lr)**

After each **fit** (run each one for a single value of **epoch**=100), present/plot the training_loss vs validation_loss (could utilize TensorBoard GUI module)

Present the **best Regression Deep Learning model** (compare the above 3) and its corresponding parameters. Assume bacth_size takes its default value of 32.

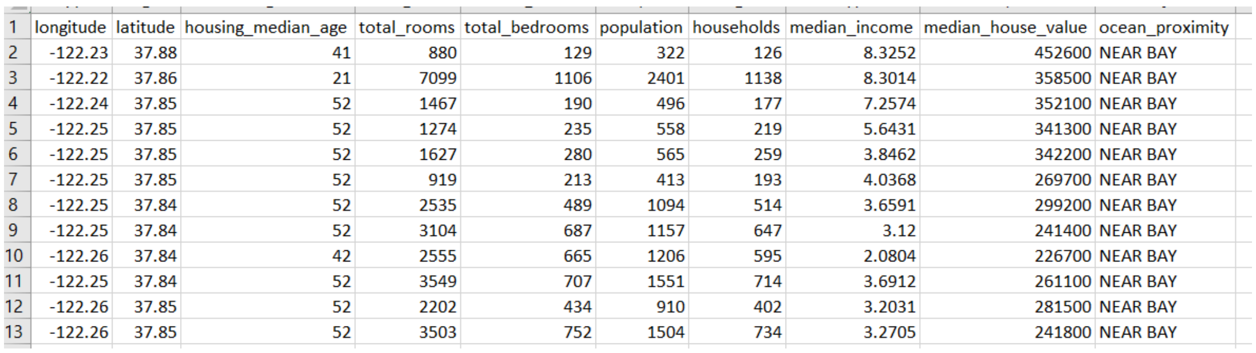
Using the best selected regression model, perform predictions by calling Keras **model.predict** module and review the loss.

Be aware of the following:
A) Mean square error (MSE) and mean absolute error (MAE) are common loss functions used for regression problems. MAE is less sensitive to outliers.
B) When numeric input data features have values with different ranges, each feature should be scaled independently to the same range.
C) Overfitting is a common problem for DNN models

Dataset / Content
The data pertains to the houses found in a given California district and some summary stats about them based on the 1990 census data. Be warned the data aren't cleaned so there are some preprocessing steps required!
The columns are as follows, their names are pretty self-explanatory:

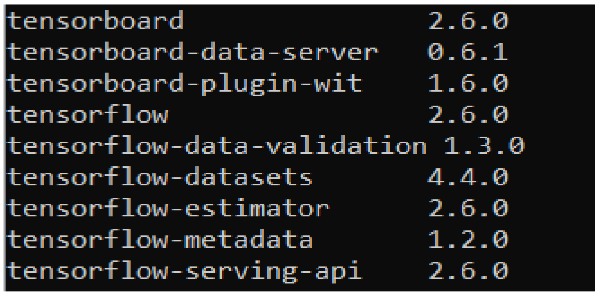
1. longitude: A measure of how far west a house is; a higher value is farther west
2. latitude: A measure of how far north a house is; a higher value is farther north

3. housing_median_age: Median age of a house within a block; a lower number is a newer building
4. total_rooms: Total number of rooms within a block
5. total_bedrooms: Total number of bedrooms within a block
6. population: Total number of people residing within a block
7. households: Total number of households, a group of people residing within a home unit, for a block
8. median_income: Median income for households within a block of houses (measured in tens of thousands of US Dollars)
9. median_house_value: Median house value for households within a block (measured in US Dollars)
10. ocean_proximity: Location of the house w.r.t ocean/sea



1	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
2	-122.23	37.88	41	880	129	322	126	8.3252	452600	NEAR BAY
3	-122.22	37.86	21	7099	1106	2401	1138	8.3014	358500	NEAR BAY
4	-122.24	37.85	52	1467	190	496	177	7.2574	352100	NEAR BAY
5	-122.25	37.85	52	1274	235	558	219	5.6431	341300	NEAR BAY
6	-122.25	37.85	52	1627	280	565	259	3.8462	342200	NEAR BAY
7	-122.25	37.85	52	919	213	413	193	4.0368	269700	NEAR BAY
8	-122.25	37.84	52	2535	489	1094	514	3.6591	299200	NEAR BAY
9	-122.25	37.84	52	3104	687	1157	647	3.12	241400	NEAR BAY
10	-122.26	37.84	42	2555	665	1206	595	2.0804	226700	NEAR BAY
11	-122.25	37.84	52	3549	707	1551	714	3.6912	261100	NEAR BAY
12	-122.26	37.85	52	2202	434	910	402	3.2031	281500	NEAR BAY
13	-122.26	37.85	52	3503	752	1504	734	3.2705	241800	NEAR BAY

(**) Highly recommend to have installed the whole gamma of TensorFlow's module.
Here is a 'base' list of them:



```
tensorboard                 2.6.0
tensorboard-data-server     0.6.1
tensorboard-plugin-wit      1.6.0
tensorflow                  2.6.0
tensorflow-data-validation  1.3.0
tensorflow-datasets         4.4.0
tensorflow-estimator        2.6.0
tensorflow-metadata         1.2.0
tensorflow-serving-api      2.6.0
```