Assignment 2 Report: Neural Networks

Course: ECE 9063 Data Analytics and Foundations

Submitted to: Katarina Grolinger

Student Name: Chunyang Jia

Student Number: 251001556

October 30, 2018

Table of Contents

Description of the forecasting problem

Different people may hold vastly different opinions on the fair value of a particular house. While this statement may sound general, in real life there are certain common factors that people prefer, say the overall lot area, the number of bedrooms, total square feet, neighborhood.

This dataset collected 79 attributes and sales price for 1460 cases of housing sales. In my Assignment, I am going to divide them into training and testing sets to train a neural network model for the predictions and see how closely my predictions are to the real sales price. Attached table below are the 19 features selected among the 79 attributes.

Feature Name	Detail Description	Feature Name	Detail Description
SalePrice	the property's sale price in dollars	GrLivArea	Above grade (ground) living area square feet
LotFrontage	Linear feet of street connected to property	FullBath	Full bathrooms above grade
LotArea	Lot size in square feet	BedroomAbvGr	Number of bedrooms above basement level
OverallQual	Overall material and finish quality	KitchenAbovGr	Number of kitchens
OverallCond	Overall condition rating	TotRomsAbvGrd	Total rooms above grade (does not include bathrooms)
YearBuilt	Original construction date	Fireplaces	Number of fireplaces
YearRemodAdd	Remodel date	GarageCars	Size of garage in car capacity
TotalBsmtSF	Total square feet of basement area	WoodDeckSF	Wood deck area in square feet
1stFlrSF	First Floor Square Feet	PoolArea	Pool area in square feet
2ndFlrSF	Second Floor Square Feet	YrSold	Year Sold

Overview of network architecture

Briefly speaking, I have used the Keras library to construct my neural network structure. For any given neural networks, there are many important parameters we need to specify. I started my neural network from the "default" model for beginners. My **basic model** contains the following features:

1. **Number of Layers**: 12. My basic model starts from 12 layers in total including 1 input layer, 1 output layer, 10 hidden layers.
2. **Input Layer**: My input layer has 19 input dimensions which corresponds to the 19 features selected from the list. These 19 features are all numeric types which does not require to be encoded and also are the most significant factors buyers tend to focus when they purchase a home.
3. **Hidden Layer**: My basic model has 10 hidden layers and each hidden layer contains 60 neurons in the middle. The dataset contains 1460 training sets in total which is considered to be a small dataset. Such a hidden layer is big enough to learn the algorithms and over complicated structure would easily over-fitting problems and largely increase the computation load.
4. **Output Layer**: 1 output neuron corresponds to our target feature, "Sale Price". That's the goal of our neural network to predict.
5. **Initial Weight and Bias of neurons**: Keras provide the method initializer class to enable the users on initialize weights and biases before training starts. By default, Keras initialize all weights and biases automatically.
6. **Activation Function**: For the hidden layer neurons, I have chosen the "relu" function as the default activation function. Relu function can be written as $f(x) = \max(x,0)$. The

advantage is that its derivative is always 1 when input is positive hence resulting faster learning than the sigmoid activation functions. The sigmoid functions can be slow at the far end as the slope of the function is very small approaching to 0.

7. **Optimizer** (Adam vs SGD): I am using the "Adam" optimizer proposed researchers and recommended by Keras as the default optimizers. In general, different optimizers tend to control the learning rate. Learning rate can is the speed we update the neural networks weight and biases. So, it can be written as $W = W - \eta * \nabla f(x)$, $B = B - \eta * \nabla f(x)$, where W is the weight, B is the bias and η is the learning rate, $\nabla f(x)$ is the derivative of our cost function.
    - For SGD optimizer, there are learning rate, decay, momentum and nestrov momentum. For this assignment, I only tuned the learning rate on SGD. Details on the implementation will be discussed in the next section.
    - For Adam optimizer, there are learning rate, beta_1, beta_2, epsilon, decay and AMSGrad variant. For this assignment, in order to simply this approach, I only tuned the learning for Adam optimizer. Further work on can be done on the beta_1, beta_2, decay, epsilon and AMSGrad variant.

8. **Loss Function**: Predicting housing sales price is a regression problem. Two most common ones are listed below:
    - Mean Square Error (Quadratic Loss, L2 loss): the MSE is the sum of the squared differences between the label value and predicted value then divided by the number of samples. The equation can be written as:

$$MSE = \frac{\sum_{i=1}^{n} (y_i - y_i^p)^2}{n}$$

- Mean Absolute Error (L1 loss): the MAE is the sum of the absolute differences of the label value and predicted value then divided by the number of samples. The equations can be written as:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - y_i^p|}{n}$$

- **Comparison**: In general, MSE is better because at the bottom, the gradient of MSE is smaller, making it easier to find the lowest point. The gradient of MAE is always the same and hence harder to get to the bottom. However, the MSE will square the error which makes it bigger and when our data is corrupted with abnormal data points, MAE should be chosen over the MSE.

9. **Batch Size**: Batch Size refers to number of samples per gradient update. If unspecified, Keras defaults to 32.
    - When the batch size is set to the total number of samples, it's called batch gradient descent.
    - When batch size is set to 1 sample per update, it's called stochastic gradient descent. In my model, I also implemented SGD method, details will be discussed in the next section.
    - When batch size is between 1 and total number of samples, it's called mini-batch gradient descent.

10. **Epochs**: Epochs refer to number of iterations to train the model. One epoch is an iteration over the entire training data set provided. Lower epochs can lead to underfitting while higher epochs will tend to overfitting problems. This can be observed by the decrease of loss errors but an increase in validation errors.
11. **Validation Split factor**: Keras provided this parameter for us to further divide the datasets to training and validation sets. It greatly helps us observe the potential of overfitting problems. Ideally the losses of both training and validation sets should decrease at the same time. In an overfitting situation, the losses of training sets continue to decrease while the losses of validation sets start to tick up. That's where we should stop. In my code, I used 0.1, equivalent to 10%, of my datasets to be the validation sets.

Description of implementation of Neural Network

1. **Feature Selection**:
2. with a unique direction on the same day, but it can pass through many stations on its route;
3. Each train should get to its destination on the same day of departure.

Entity - Relationship Diagram

Figure 1 Entity-relationship diagram

Relational Model

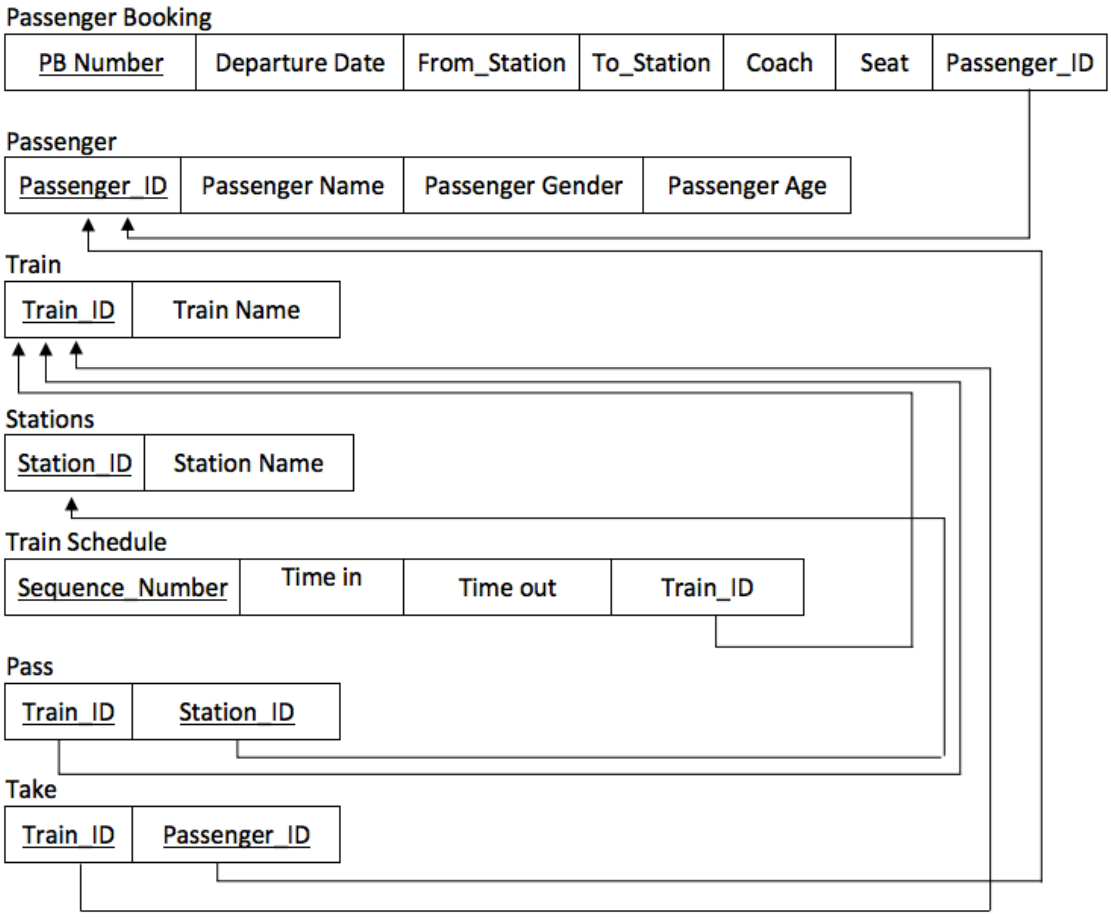

Figure 2 Relational Model

Python Code

See below and check file attached *Assignment_2_NN.py*.

```
use Task_1_ECE9014;

/*1. create Trains entity and insert the data*/
create table Trains(
        Train_ID INT NOT NULL Primary Key,
        Train_Name varchar(50) not null
);

insert into trains(train_ID, train_name)
                values (1, 'Toronto_london_express');

insert into trains(train_ID, train_name)
                values (2, 'Windsor_london_express');

/*2. create Stations entity and insert the data*/
create table Stations(
           Station_ID INT NOT NULL primary key,
           station_name varchar(50) not null
);

insert into stations(Station_ID, Station_Name)
                values (50, 'London');

insert into stations(Station_ID, Station_Name)
                values (60, 'Toronto');

insert into stations(Station_ID, Station_Name)
                values (70, 'Windsor');

/*3. create Passenger entity and insert the data*/
create table Passenger(
            Passenger_ID INT NOT NULL primary key,
            Passenger_Name varchar(50) not null,
            Passenger_Gender varchar(50),
            Passenger_Age INT

);

insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
                 values(126, 'Chunyang Jia','M',26);

 insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
                 values(127, 'Jicong Guo','M',26);

 insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
                 values(128, 'Liwei Wang','M',22);

 insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
                 values(129, 'Ali Rmeity','M',22);

/*4. create Train_Schedule entity and insert the data*/
 create table Train_Schedule(
                    Sequence_Number INT NOT NULL Primary key,
                    Time_in datetime not null,
```

```
55.                      Time_out datetime not null,
56.                      Train_ID int not null,
57.
58.                      foreign key(Train_ID ) references Trains(Train_ID)
59.  );
60.
61.  insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
62.                    values(31356,'2018-10-23 13:00:00','2018-10-23 16:00:00',1);
63.
64.  insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
65.                    values(31398,'2018-10-23 10:00:00','2018-10-23 18:00:00',1);
66.
67. insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
68.                    values(32345,'2018-10-23 6:00:00','2018-10-23 10:00:00',2);
69.
70.  insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
71.                    values(33498,'2018-10-23 3:00:00','2018-10-23 7:00:00',2);
72.
73.
74. /*5. create Passenger_Booking entity and insert the data*/
75. create table Passenger_Booking(
76.                       Passenger_Booking_Number int not null primary key,
77.                       Booking_Date date not null,
78.                       From_Station varchar(50),
79.                       To_Station varchar(50),
80.                       Coach int not null,
81.                       Seat int not null,
82.                       Passenger_ID int not null
83.
84. );
85.
86. alter table passenger_booking add foreign key (passenger_id) references passenger(passenge
    r_ID);
87.
88. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
    on,Coach,Seat,Passenger_ID)
89.                        values (235,'2018-10-23','London','Toronto',3,4,126);
90. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
    on,Coach,Seat,Passenger_ID)
91.                        values (236,'2018-10-23','London','Toronto',3,5,127);
92. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
    on,Coach,Seat,Passenger_ID)
93.                        values (237,'2018-10-23','London','Toronto',3,6,128);
94. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
    on,Coach,Seat,Passenger_ID)
95.                        values (238,'2018-10-23','London','Windsor',4,2,129);
96.
97. alter table passenger_booking change column Booking_date  Departure_Date date not null;
98.
99. /*6. create Pass relationship and insert the data*/
100.create table Pass(
101.        Train_ID int not null,
102.                       Station_ID int not null,
103.
104.
105.        primary key(Train_ID,Station_ID),
106.                       foreign key(Train_ID ) references Trains(Train_ID),
107.         foreign key(Station_ID ) references Stations(Station_ID)
108.);
109.
110.insert into Pass(Train_ID,Station_ID) values(1,50);
```

```
111. insert into Pass(Train_ID,Station_ID) values(1,60);
112. insert into Pass(Train_ID,Station_ID) values(1,70);
113. insert into Pass(Train_ID,Station_ID) values(2,50);
114. insert into Pass(Train_ID,Station_ID) values(2,60);
115. insert into Pass(Train_ID,Station_ID) values(2,70);
116.
117.
118. /*7. create Take relationship and insert the data*/
119.
120. create table Take(
121.         Train_ID int not null,
122.                         Passenger_ID int not null,
123.
124.
125.          primary key (Train_ID,Passenger_ID),
126.                          foreign key(Train_ID ) references Trains(Train_ID),
127.          foreign key(Passenger_ID ) references Passenger(Passenger_ID)
128. );
129.
130. insert into Take(train_id, passenger_id) values(1,126);
131. insert into Take(train_id, passenger_id) values(1,127);
132. insert into Take(train_id, passenger_id) values(1,128);
133. insert into Take(train_id, passenger_id) values(2,129);use Task_1_ECE9014;
134.
135. /*1. create Trains entity and insert the data*/
136. create table Trains(
137.         Train_ID INT NOT NULL Primary Key,
138.         Train_Name varchar(50) not null
139. );
140.
141. insert into trains(train_ID, train_name)
142.                 values (1, 'Toronto_london_express');
143.
144. insert into trains(train_ID, train_name)
145.                 values (2, 'Windsor_london_express');
146.
147. /*2. create Stations entity and insert the data*/
148. create table Stations(
149.             Station_ID INT NOT NULL primary key,
150.             station_name varchar(50) not null
151. );
152.
153. insert into stations(Station_ID, Station_Name)
154.                 values (50, 'London');
155.
156. insert into stations(Station_ID, Station_Name)
157.                 values (60, 'Toronto');
158.
159. insert into stations(Station_ID, Station_Name)
160.                 values (70, 'Windsor');
161.
162. /*3. create Passenger entity and insert the data*/
163. create table Passenger(
164.              Passenger_ID INT NOT NULL primary key,
165.              Passenger_Name varchar(50) not null,
166.              Passenger_Gender varchar(50),
167.              Passenger_Age INT
168.
169. );
170.
171. insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
```

```
172.                    values(126, 'Chunyang Jia','M',26);
173.
174.  insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
175.                    values(127, 'Jicong Guo','M',26);
176.
177.  insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
178.                    values(128, 'Liwei Wang','M',22);
179.
180.  insert into Passenger (Passenger_ID, Passenger_Name, Passenger_Gender,Passenger_Age)
181.                    values(129, 'Ali Rmeity','M',22);
182.
183. /*4. create Train_Schedule entity and insert the data*/
184.  create table Train_Schedule(
185.                       Sequence_Number INT NOT NULL Primary key,
186.                       Time_in datetime not null,
187.                       Time_out datetime not null,
188.                       Train_ID int not null,
189.
190.                       foreign key(Train_ID ) references Trains(Train_ID)
191.  );
192.
193.  insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
194.                    values(31356,'2018-10-23 13:00:00','2018-10-23 16:00:00',1);
195.
196.  insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
197.                    values(31398,'2018-10-23 10:00:00','2018-10-23 18:00:00',1);
198.
199. insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
200.                    values(32345,'2018-10-23 6:00:00','2018-10-23 10:00:00',2);
201.
202.  insert into Train_Schedule(sequence_number, time_in, time_out, train_ID)
203.                    values(33498,'2018-10-23 3:00:00','2018-10-23 7:00:00',2);
204.
205.
206. /*5. create Passenger_Booking entity and insert the data*/
207. create table Passenger_Booking(
208.                      Passenger_Booking_Number int not null primary key,
209.                      Booking_Date date not null,
210.                      From_Station varchar(50),
211.                      To_Station varchar(50),
212.                      Coach int not null,
213.                      Seat int not null,
214.                      Passenger_ID int not null
215.
216. );
217.
218. alter table passenger_booking add foreign key (passenger_id) references passenger(passenge
     r_ID);
219.
220. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
     on,Coach,Seat,Passenger_ID)
221.                       values (235,'2018-10-23','London','Toronto',3,4,126);
222. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
     on,Coach,Seat,Passenger_ID)
223.                       values (236,'2018-10-23','London','Toronto',3,5,127);
224. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
     on,Coach,Seat,Passenger_ID)
225.                       values (237,'2018-10-23','London','Toronto',3,6,128);
226. insert into Passenger_Booking(Passenger_Booking_Number, Booking_Date,From_Station,To_Stati
     on,Coach,Seat,Passenger_ID)
227.                       values (238,'2018-10-23','London','Windsor',4,2,129);
```

```
228.
229.alter table passenger_booking change column Booking_date  Departure_Date date not null;
230.
231./*6. create Pass relationship and insert the data*/
232.create table Pass(
233.        Train_ID int not null,
234.                        Station_ID int not null,
235.
236.
237.        primary key(Train_ID,Station_ID),
238.                        foreign key(Train_ID ) references Trains(Train_ID),
239.         foreign key(Station_ID ) references Stations(Station_ID)
240.);
241.
242.insert into Pass(Train_ID,Station_ID) values(1,50);
243.insert into Pass(Train_ID,Station_ID) values(1,60);
244.insert into Pass(Train_ID,Station_ID) values(1,70);
245.insert into Pass(Train_ID,Station_ID) values(2,50);
246.insert into Pass(Train_ID,Station_ID) values(2,60);
247.insert into Pass(Train_ID,Station_ID) values(2,70);
248.
249.
250./*7. create Take relationship and insert the data*/
251.
252.create table Take(
253.        Train_ID int not null,
254.                        Passenger_ID int not null,
255.
256.
257.         primary key (Train_ID,Passenger_ID),
258.                         foreign key(Train_ID ) references Trains(Train_ID),
259.         foreign key(Passenger_ID ) references Passenger(Passenger_ID)
260.);
261.
262.insert into Take(train_id, passenger_id) values(1,126);
263.insert into Take(train_id, passenger_id) values(1,127);
264.insert into Take(train_id, passenger_id) values(1,128);
265.insert into Take(train_id, passenger_id) values(2,129);
```