1)

I consider as most appropriate the BiLSTM model, which is a model composed by a forward LSTM and a backward LSTM, where the hidden state is computed as a combination of both the forward and backward LSTM hidden states.
I decided to use BiLSTM and not MLP or a CNN because CNN and MLP are a model which are not able to retain information over time, in fact after few iteration they lost informations, and so they are not suitable for this kind of input data. I could have also used an LSTM but I discard it because BiLSTM gets better performance and is more suitable for offline data (as in this case), even if the LSTM use the half of the parameters, keeping the model less complex.
After the BiLSTM I will use a MLP, I will concatenate the output of the BiLSTM with the other inputs and I will fed everything to the MLP.

2)
a)
On the data we will apply preprocessing on starting from:
- Hotel_name: each string is mapped to an integer number starting from 0
- Reviewer_numer_reviews: scale the number in [0,1]
- Review_date: split the date using "/" and I unify together the digits into an integer. I will obtain 3 vectors of integer: month, day, year.
- Review:
  - Split the words by whitespaces
  - Lowercase everything
  - Remove punctuation (create a list of stopwords)
  - Now we will have for any review a dictionary in which each entry is a word of the review. Then I'll create a dictionary of unique words, and I will number each unique word with integer starting from 0. Then I define an embedding for each word using an embedding layer which takes in consideration the context of the words.
- Review_score: I scale the score in [0, 1]

b) Input of the model will be:
- Hotel_name → vector of integers (13772 unique hotels)
- Reviewer_numer_reviews → vector with 13772 entries
- Review → (13772 x num_of_unique_words x embedding_size)
- Review_date →3 vectors of integers (month, day, year of integer) each with 13772 entries
- Hotel_numer_reviews: vector with 13772 entries

I decide to use the hotel name because I think that the review is based on the hotel in which we are. Also the date could be important and the number of reviews can determine if it's a frequented hotel or not. The review_numer could be important to understand if the reviewer is experienced.

I decided to drop the hotel_address and reviewer_nationality, which I think are irrelevant for this task. Also the review_type which could in a sort of way directly correspond to a score (meaning that the model could learn to predict the score from the classification in good or bad reviews).

3)
The output layer will be formed by one unit on which we will apply a sigmoid function and the output will be the score in [0,1]. If we want to have the output in [0, 10] we will apply a rescaling after the sigmoid. I decided to use just one unit because in this unit our aim is to output a real value using the sigmoid. The majority of the work for predicting this value will be done on the previous hidden layers of the MLP.

4)
I will use a MSE (Mean Squared Error) which calculate the squared average distance between the actual review_score and the predicted review_score.
Since we are dealing with a regression task, the most suitable loss is the MSE which is computed over real values.
If the dataset contains several outliers, we could also use the MAE (Mean Absolute Error).

5)
First we will apply an embedding layer. Then the model would be composed initially by a BiLSTM with n layers. As explained before the BiLSTM is formed by two LSTMs: a forward and a backward one. Each of the layers will compute the hidden state as a composition of the forward hidden state and backward hidden state.

The model will receive in input the embeddings of each review and its output will be concatenated with the hotel_name, Reviewer_numer_reviews, review_date, and hotel_number_reviews and it will directly be given as input to a MLP. The MLP will be formed by n hidden layers each containing n units which will try to predict the review_score. The final layer will be composed by a unit with a sigmoid function.
The activation function of the MLP hidden layers will be a sigmoid while the activation function of the BiLSTM are tanh and sigmoid, which are used in the LSTM gates.

———

5)

We will use as optimizer Adam which combine both momentum and RMSprop.
As initializer since we are using a sigmoid function I will use a glorot/Xavier initializer which help us with the weight initialization. I will use dropout over the hidden layers of the MLP to reduce overfitting with a dropout rate = 0.2 (can be tuned if regularization is not enough), also a batch normalization.
The scaling will be done using a minmax scaler for the input values, and the output value will be rescaled to a value in [0, 10].
The hyperparameters that I consider more relevant are:

- Learning rate
- Epochs
- Dropout rate
- Batch size
- Number of hidden layers of MLP
- Number of units per layer of MLP
- Number of BiLSTM layers

6)
Since the data are not splitted in training and testing, I will split them, and then I will split the training data into a smaller training data and into a validation set (e.g. 70% training, 15% testing, 15% validation).
Then I will apply a grid search on the validation set (using fewer hyperparameters, otherwise we would need to many parameters) to find the best values of hyperparameters for which we get the best performance over the validation set.
I would train the model over the training set and assess its generalization capabilities on unseen data: the test set.

I will use as metrics the MSE or if we have an high number of outliers the MAE, which are suitable for regression tasks like this one.