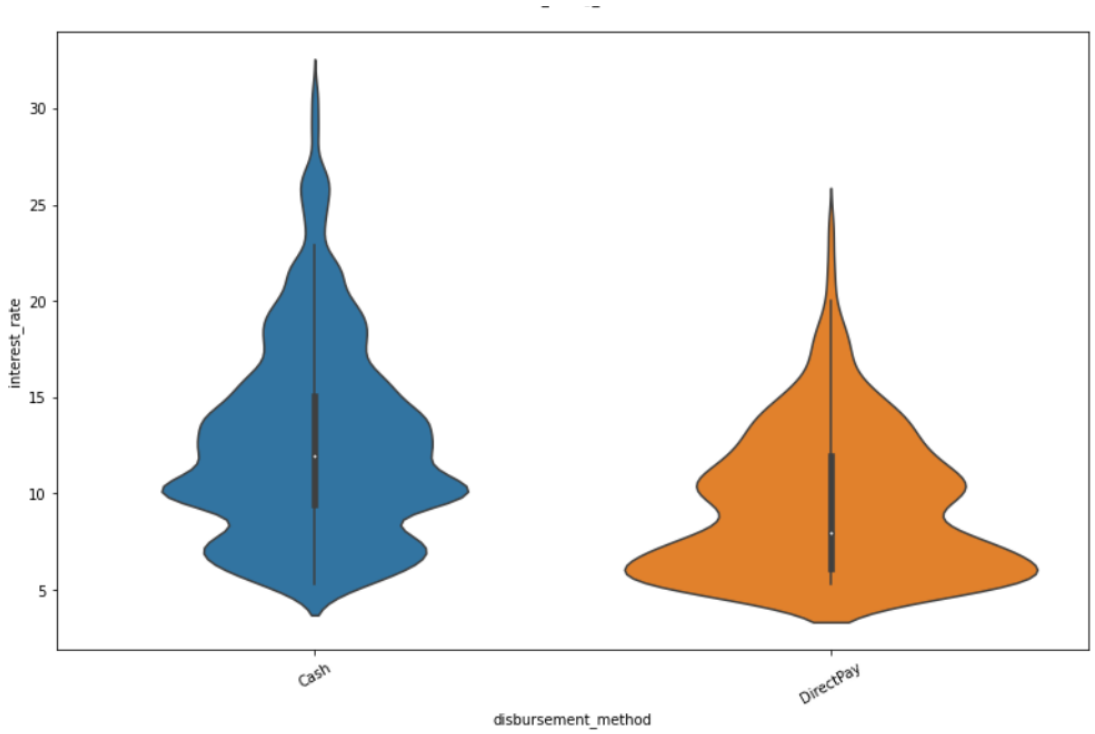
Internship Project Report

Case 1

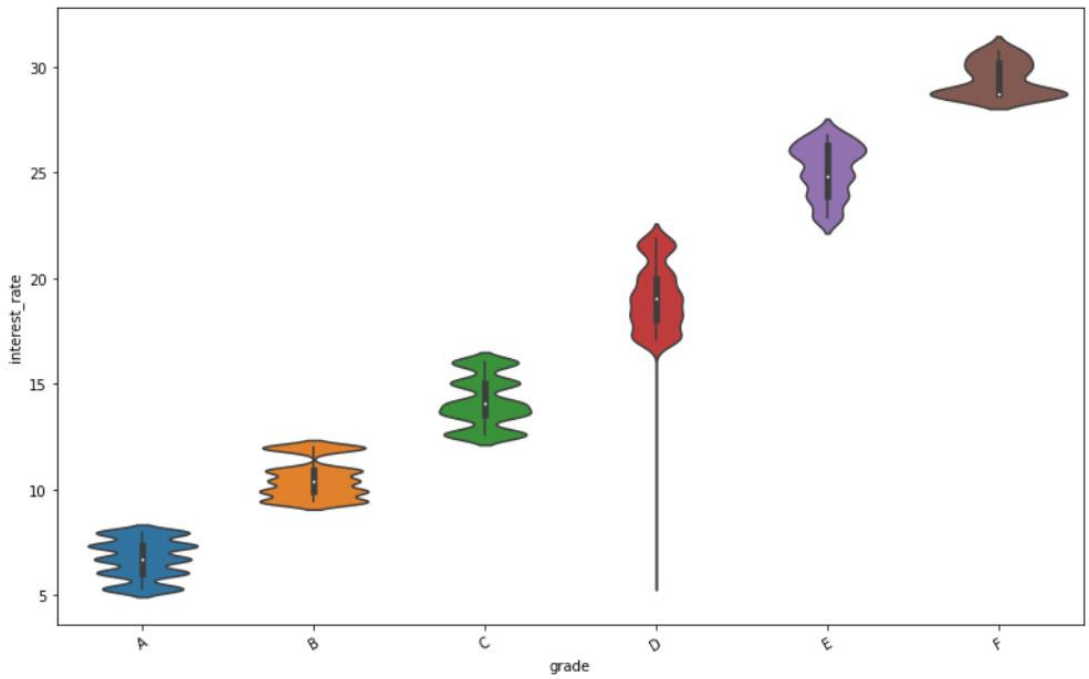
1. Description: This is a dataset containing many people's data with plenty of features, and we need to use these data to train a model to predict the interest_rate.
   Problem:
   (1) In the case study 1, I strongly think there are at least two data inside the dataset (row 3, row 15) are wrong. I think this is an error because there is a long tail in the violin graph when I analyze grade and interest rate
   (2) In verified_income and verification_income_joint we have three categories(source verified, verified, not verified), but we only need two verifications. (Verified, not verified)
   (3) The num_accounts_120d_past_due column is useless. It only has 0 and NULL. num_accounts_30d_past_due is almost useless, it only has a 1.
   (4) There are many NaN values inside the dataset. We must clean it before we want to push our dataset to the neural network
   (5) Both categorical and numerical values are inside the dataset. We need to apply one hot encoder to generate the feature set.
2.

(1)



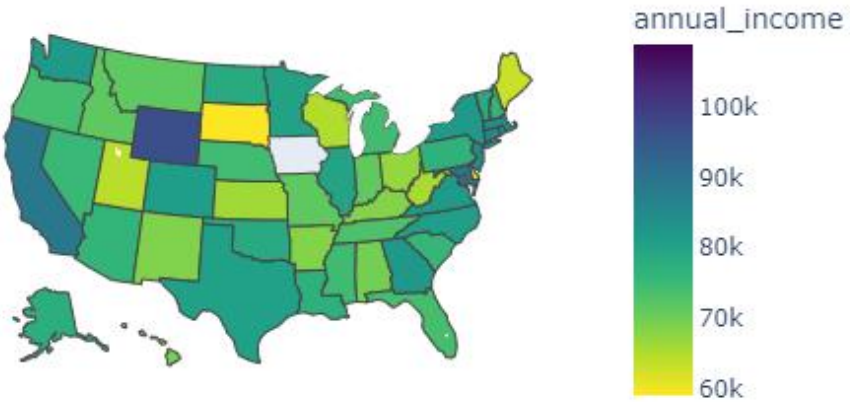
As we can see from the violin plot for the disbursement_method, we can find that there is a strong difference in the interest_rate if people apply different disbursement_method.

(2)



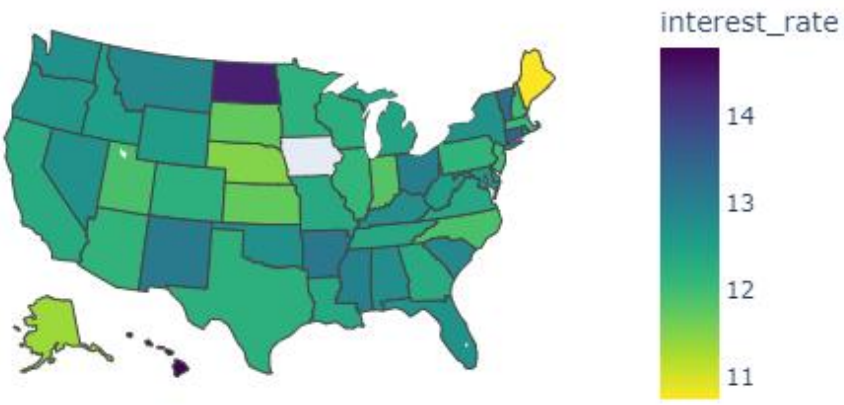
As we can see, different credit grades strongly affect the interest_rate. This means if we use grade as an input value inside our neural network, it looks like a cheat.

(3)



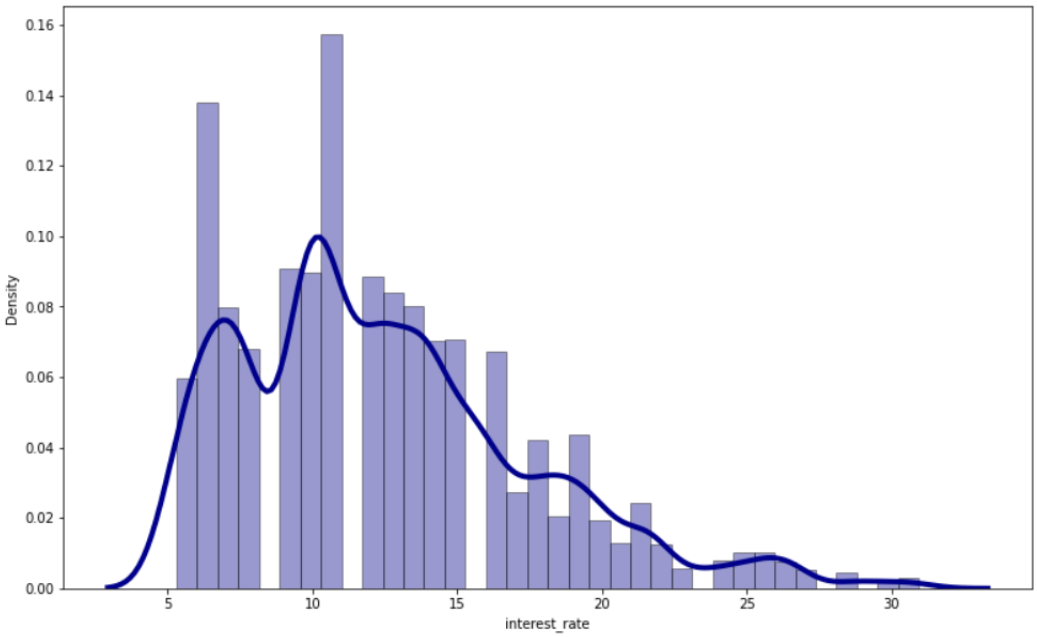
As we can see, the mean of annual_income in this dataset in WY is the top in the USA.

(4)



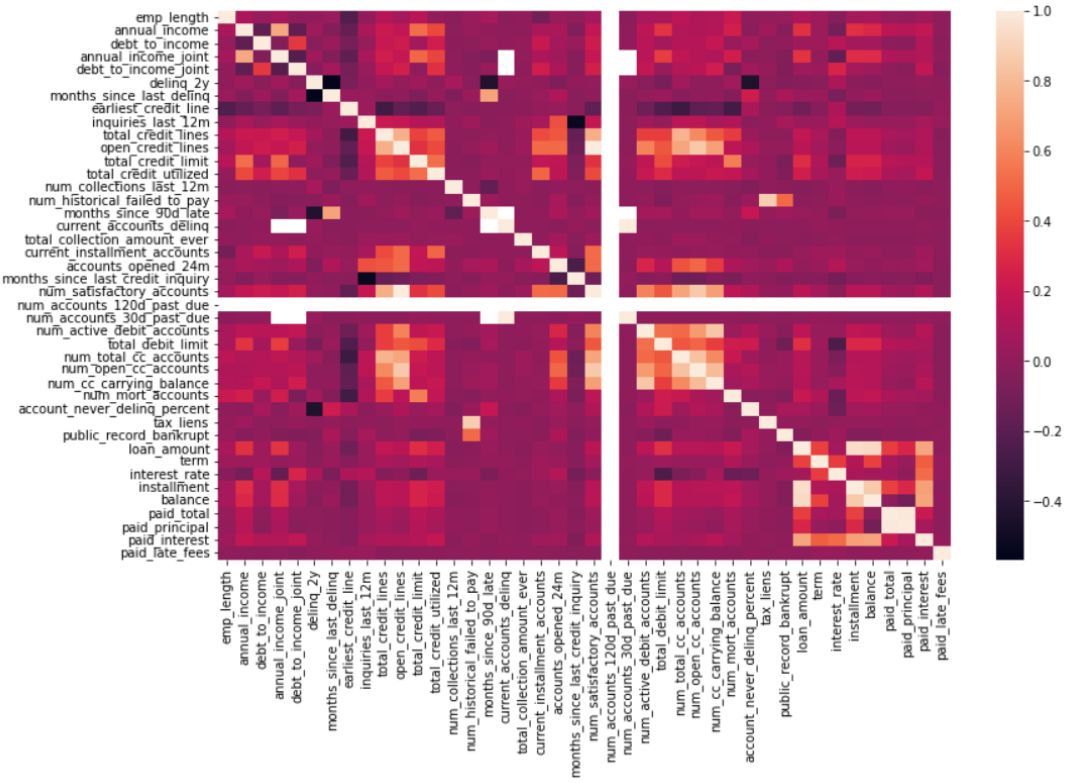
At first, I did not think I should input the state as an input when I trained the neural network. However, according to this graph, it is a great idea not to remove state from the input parameters.

(5)



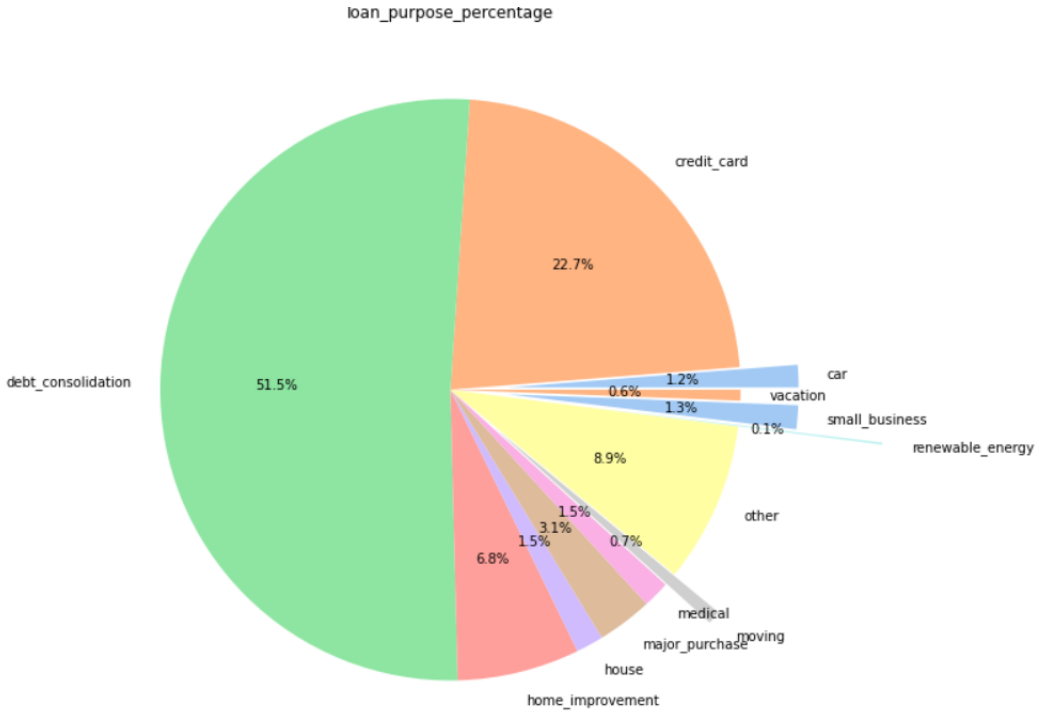
This graph both contains the histogram and density graph. As we can see, most people get an interest rate of around ten percentage points.

(6)



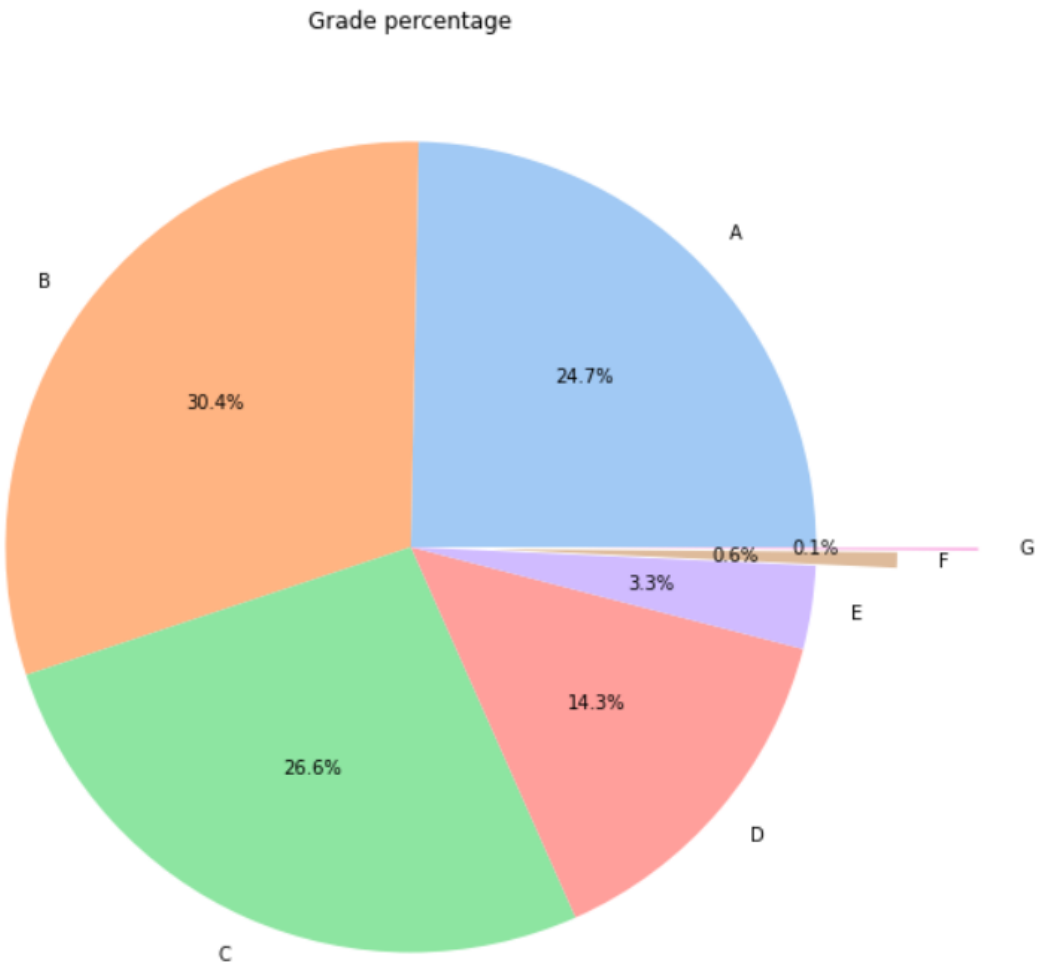
This is a heatmap, and we can learn which variables have a high correlation with the interest rate. The term seems important.

(7)



This is a pie chart. We can see from this graph that more than half of the people loan money because they need debt_consolidation.

(8)



This is a pie chart. We can see from this graph that more than half of the people get credit grade higher than B (Included).

3. Reference: https://stackoverflow.com/questions/46433588/pandas-drop-rows-columns-if-more-than-half-are-nan

(0) I have also tried to use linear regression to predict the result, but the result is too terrible because we have ignored too many categorical values. Then I tried to use ordinal encoding to preprocess the categorical consequence, but the result was awful. Finally, I decided to use one-hot encoding to preprocess the neural network.

(1) I remove those columns that have more than half the NULL value.

(2) I replace the cell whose column is a categorical value with the mode value in the column.

(3) I replace the cell whose column is a numerical value with the mean value in the column.

(4) After that, I remove these columns ["num_accounts_30d_past_due", "num_accounts_120d_past_due", "emp_title", "issue_month", "sub_grade"], some of them are useless and some of them may make my results become too better, for example sub_grade.

(5) I divide my features into 3 categories:

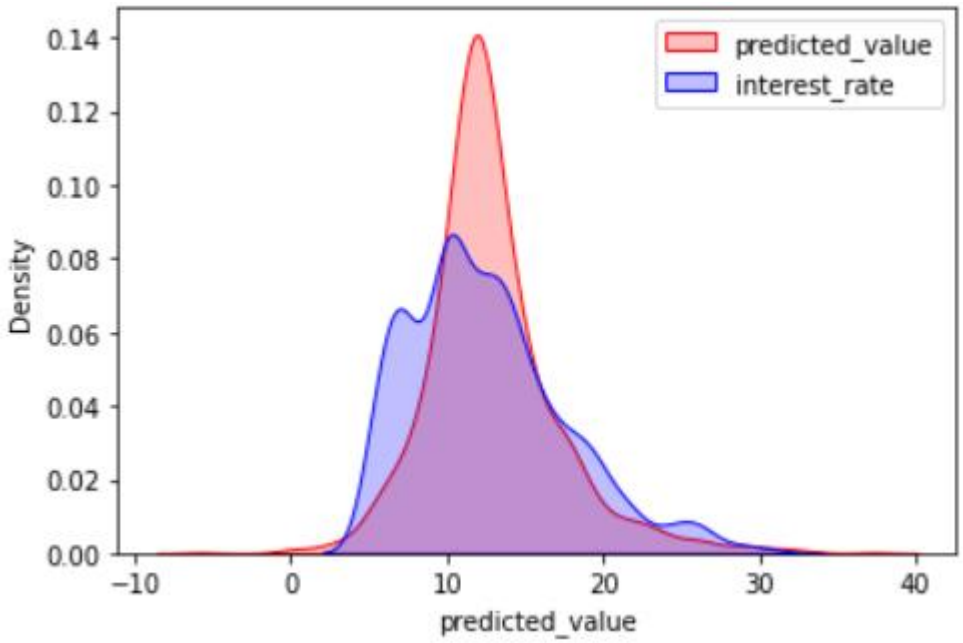
Numerical_columns

Bucketized_columns: ["emp_length", "earliest_credit_line", "term"]

Categorical_column: ["homeownership", 'verified_income', 'loan_purpose', 'application_type', "grade", 'loan_status', "initial_listing_status", "disbursement_method",

"state"]
There are a total of 44 features input.
(6) Use DenseFeatures to transform feature_columns to feature layers for the future easier input.
(7) Use sequential to create the neural network, and use Epochs = 300 and batch_size = 16 to train the model.
4. Final training and testing result:

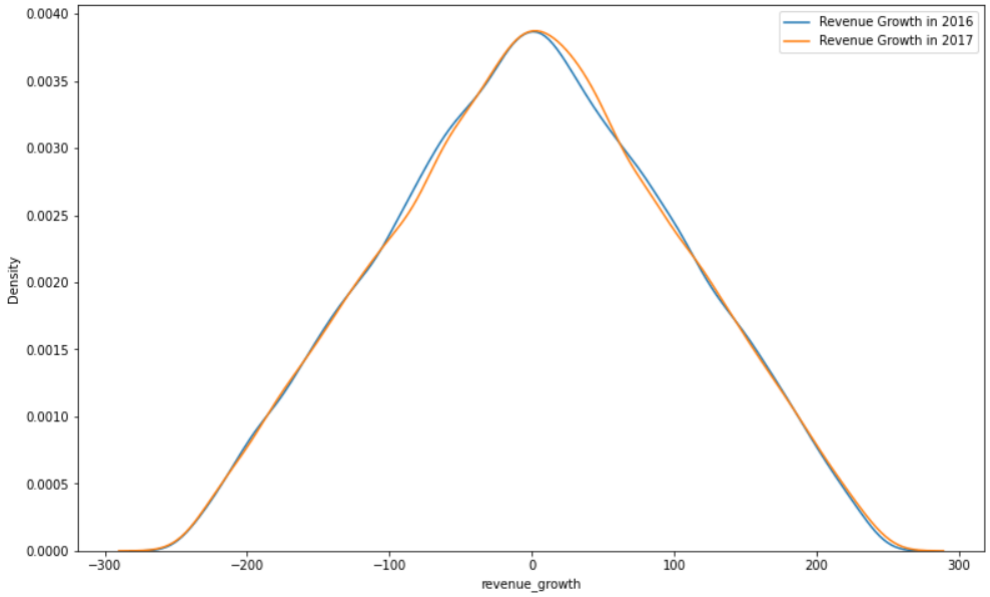
```
Epoch 298/300
500/500 [==============================] - 1s 2ms/step - loss: 11.7856 - mse: 11.7856 - val_loss: 10.9337 - val_mse: 10.9337
Epoch 299/300
500/500 [==============================] - 1s 2ms/step - loss: 11.6906 - mse: 11.6906 - val_loss: 10.7721 - val_mse: 10.7721
Epoch 300/300
500/500 [==============================] - 1s 2ms/step - loss: 11.8270 - mse: 11.8270 - val_loss: 11.1422 - val_mse: 11.1422
```

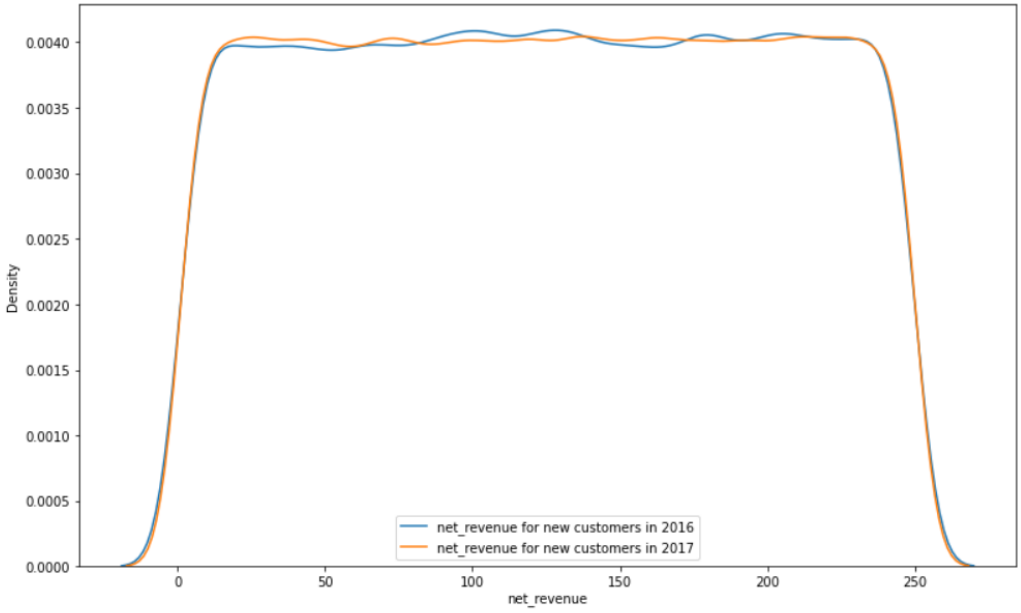


If I had more time, I would adjust each layer inside the neural network. I may even add one drop off layer to prevent the overfitting. Also, I will spend more time changing epochs. Also, I would not replace the NaN value with the mode of the column. This seems to be a bad practice. I need to handle the NaN value more carefully, and I can even make the NaN value become one category under the categorical variables. Also, the bound for the Bucketized_columns also needs to be adjusted to get a better prediction.

Case 2

Questions ? I Feel confused about the revenue lost from attrition.





As we found, the density distribution for the revenue growth or the net revenue for new customers is quite similar in 2016 and 2017.