# Visualizing Data

The first step is to get an understanding for which variables are important, view summary statistics, and visualize the data. The lending_loan_club data set is being used for this project. First, the panda's library is imported to read and manipulate the data from the csv file. Then a simple count plot is used to see the loan_status column. This plot shows how many of the borrowers fully paid back their loans and how many of them were charged off. This will eventually be the target class for the model that will be created to predict the person who will pay back the loan depending on the other columns.

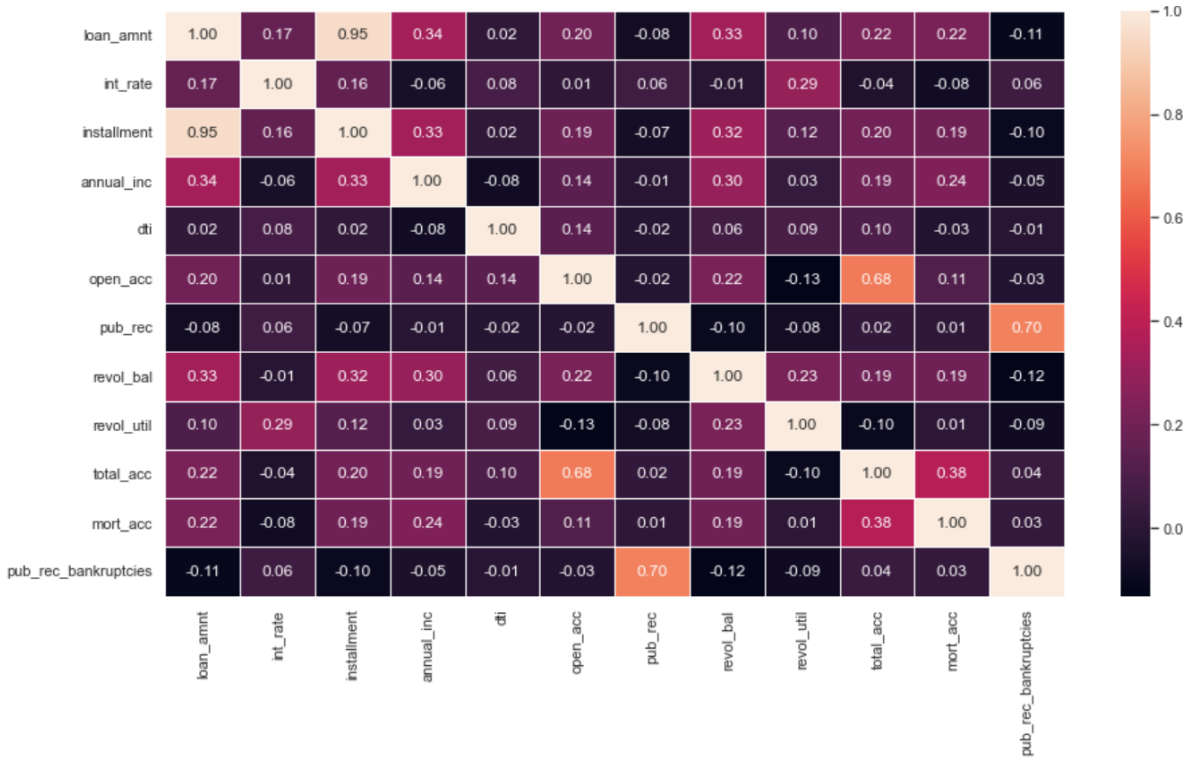


**Figure 1: Correlation heat-map of all numerical columns.**

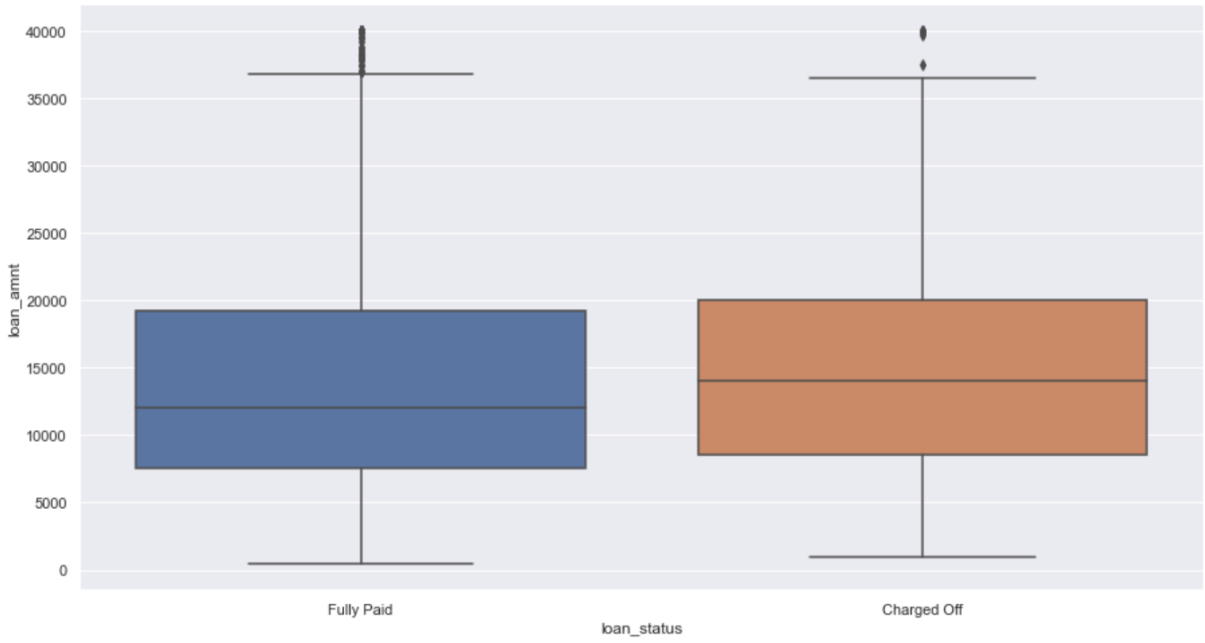


**Figure 2: Boxplot of loan_amnt grouped by loan_status**

A correlation map is created (*Fig. 1*) to see which columns are related and how closely they are related. Based on the correlation map created, we can conclude that the loan_amnt and the installment are very closely related at 95%, the next closest is pub_rec_bankruptcies and pub_rec at 70%, the last closest is total_acc and open_acc at 68%. The data for loan_amnt and installment are likely duplicated. An analysis of the relationship between the loan_status and loan_amnt is done using a boxplot (*Fig. 2*). Summary statistics for loan_amount is displayed in a group with the loan_status (*Table. 1*). This table gives us an idea of how many people have taken loan and fully paid or got charged off, the mean of the loan amount, standard deviation as well as the max. This allows us to visualize the dataset better.

| loan_status | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **Charged Off** | 77673.0 | 15126.300967 | 8505.090557 | 1000.0 | 8525.0 | 14000.0 | 20000.0 | 40000.0 |
| **Fully Paid** | 318357.0 | 13866.878771 | 8302.319699 | 500.0 | 7500.0 | 12000.0 | 19225.0 | 40000.0 |

**Table 1: Loan_amount grouped by loan_status**

Grade and the sub_grade columns are analyzed. The unique grades that are possible are the following - ['A', 'B', 'C', 'D', 'E', 'F', 'G']. A simple count plot is made of each grade to see the distribution of the rows across the different grades (Fig. 3). From the analysis of the count plot the following conclusions are drawn:

- B and C are the most common grades.
- The amount of A and D grades are almost equal with D being a little higher.
- E, F and G are all significantly lower than each other and lower than grade D and A.

Keeping this information in mind, the count plot of sub_grade is created. With the count plot of the sub_grade column the differences between each of the grades is clearer. The trend on the differences between the grades is more vibrant as seen in Fig. 4. The graph starts steady in the beginning, slowly rises upwards until grade B3, then it starts to slowly decrease all the way to the end of the graph.

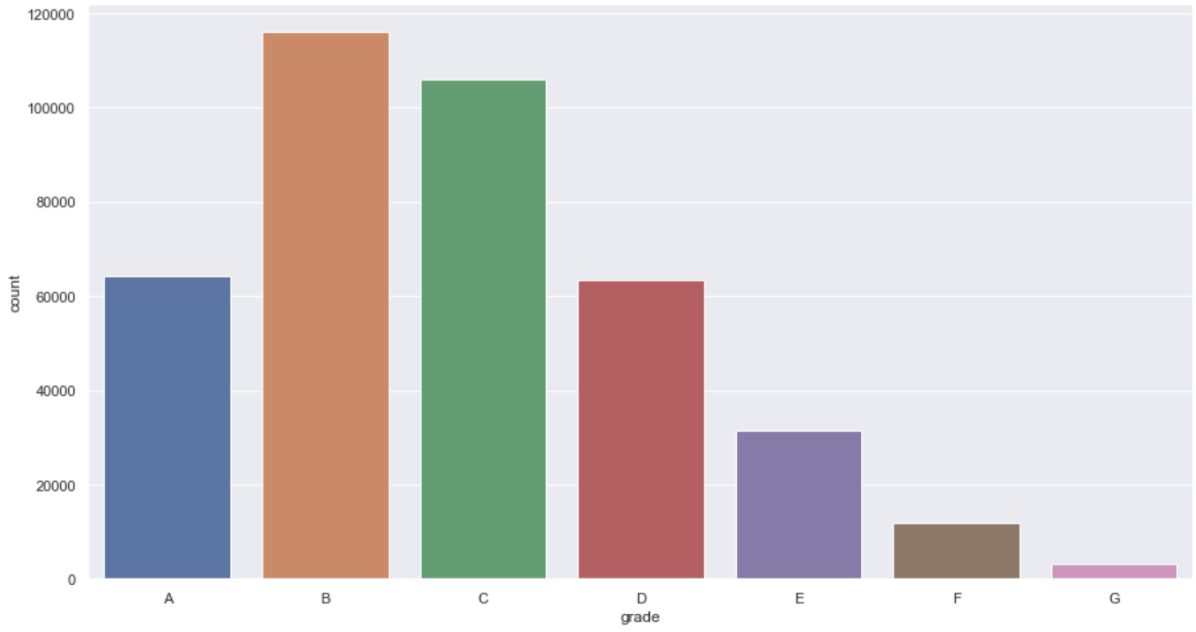


**Figure 3: Count plot of the grade column.**

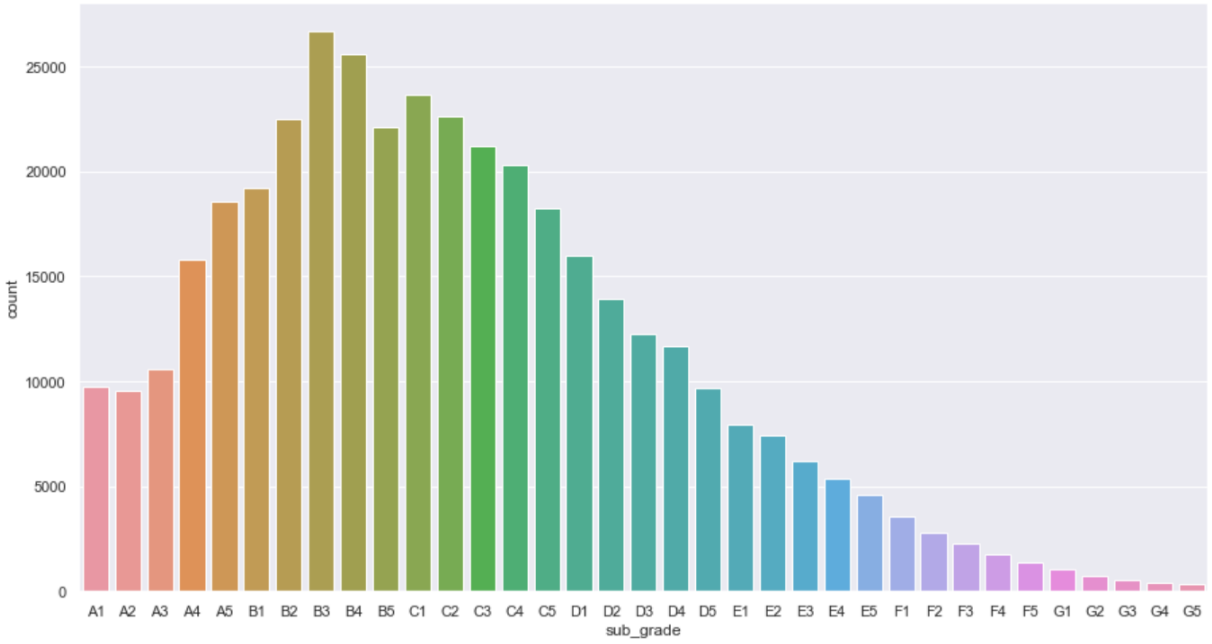


**Figure 4: Count plot of the sub_grade column.**

A count plot per subgrade is made to see the number of loans given out to each subgrade as well as a distinction made between the loans that were paid and the loans that were charged off. This gives a distinctive idea of how many people per grade did not pay back their loans versus how many people per subgrade received loans. Following these steps, another graph is made to only include the subgrades which do not get paid back often, these are from grades F and G as seen from Fig. 5. Fig. 6 depicts the loan count by subgrade that does not get paid back often.

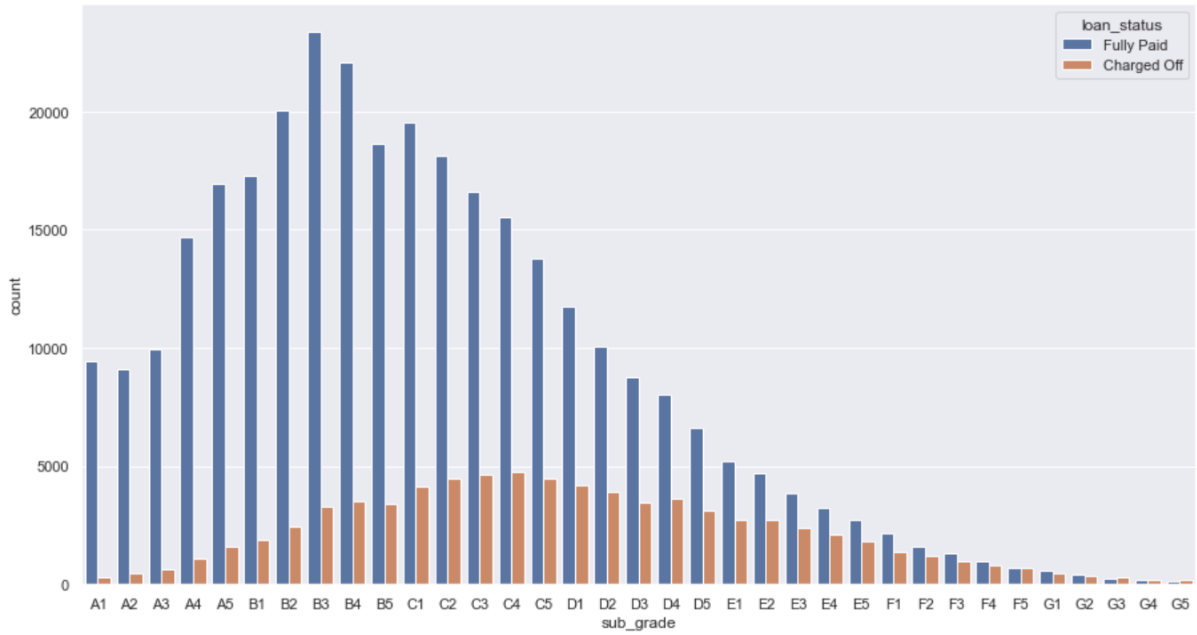


**Figure 5: Loans made per subgrade separated based on loan_status.**

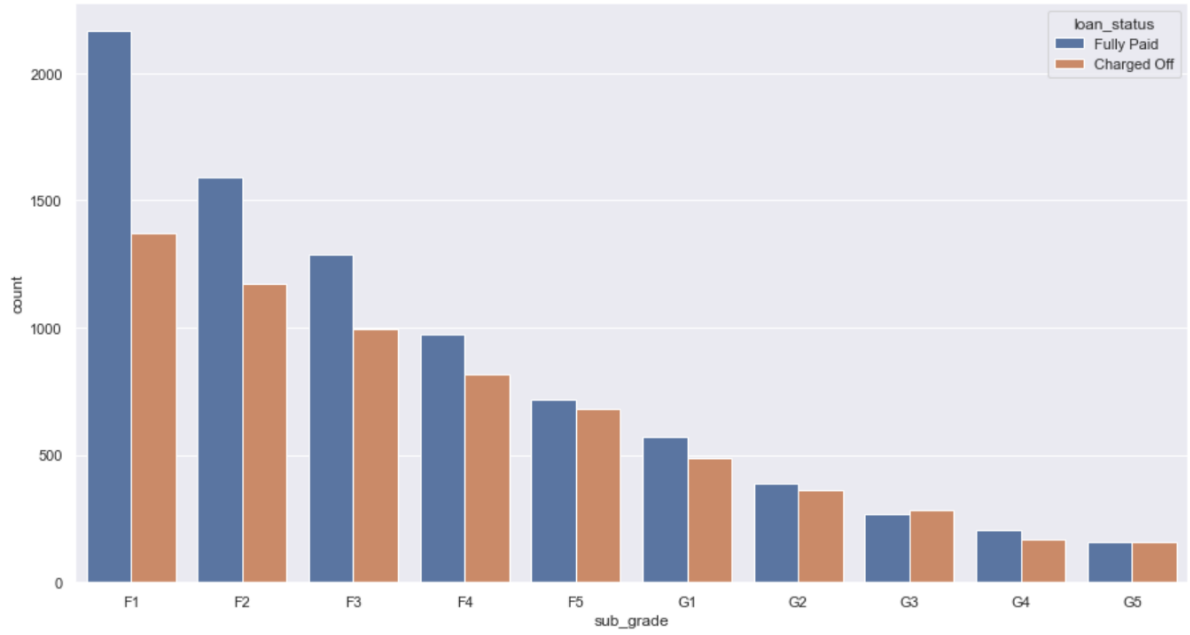


**Figure 6: Loan count by subgrade that does not get paid back often.**

A new column called loan_repaid is made with 1's and 0's based on fully paid or charged off. These values are used to plot a bar graph on correlation between each numerical column in

sorted. As seen in Fig. 7, the most correlated column to loan_repaid is mort_acc.

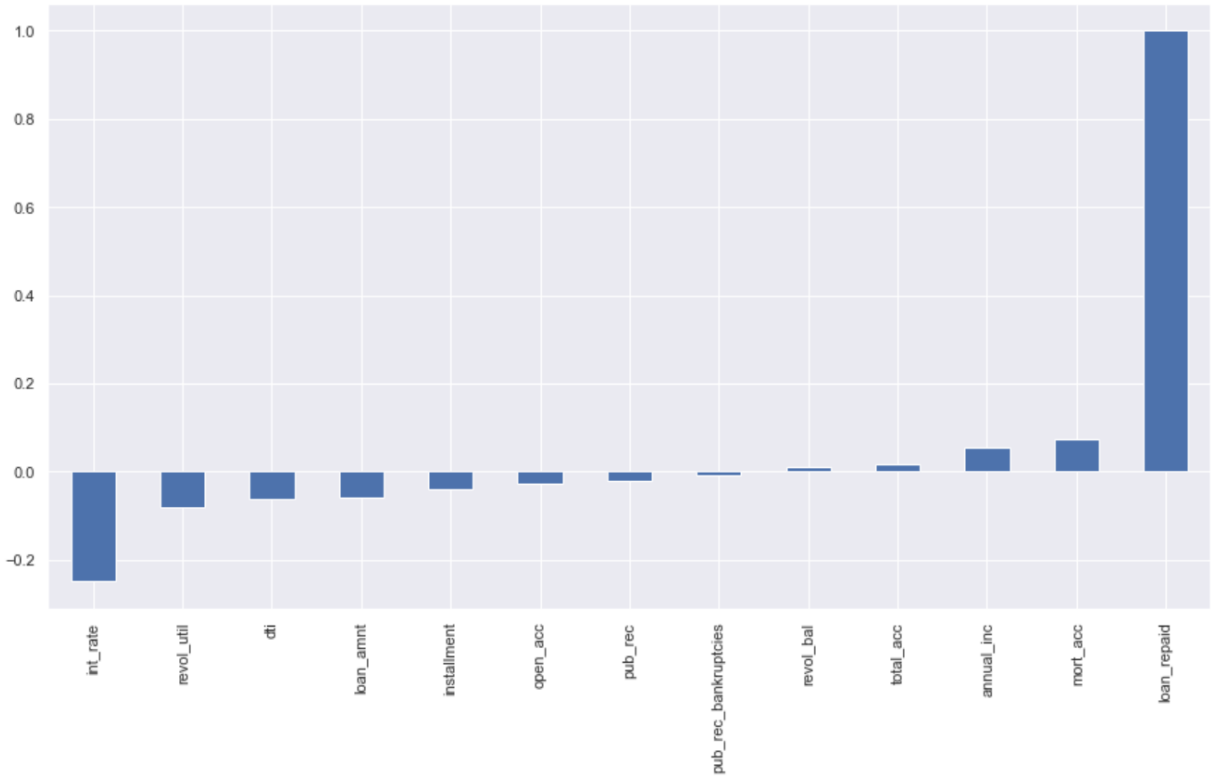


**Figure 7: Correlation bar graph between loan_repaid and all other numerical columns.**

# Data Preprocessing

The goal in this stage is to remover or fill any missing data, remove unnecessary and repetitive features and convert categorical string features into dummy variables. The total count of missing values per column and show the same in percentage. This gives an idea on which columns need to be edited. From the count, the columns that needs to be filled in is emp_title, emp_length, title, revol_util, mort_acc and pub_rec_bankruptcies.

First, the emp_title column is looked at. There seems to be way too many different types of employment title which means they will do little to help as a feature when classifying by the model, therefore it is safe to drop this column. A unique count of the emp_length is taken, there are 11 different lengths starting with less than 1, ending with more than 10. Then a count plot of the emp_length is made to see the distribution of the length over the number of employees.

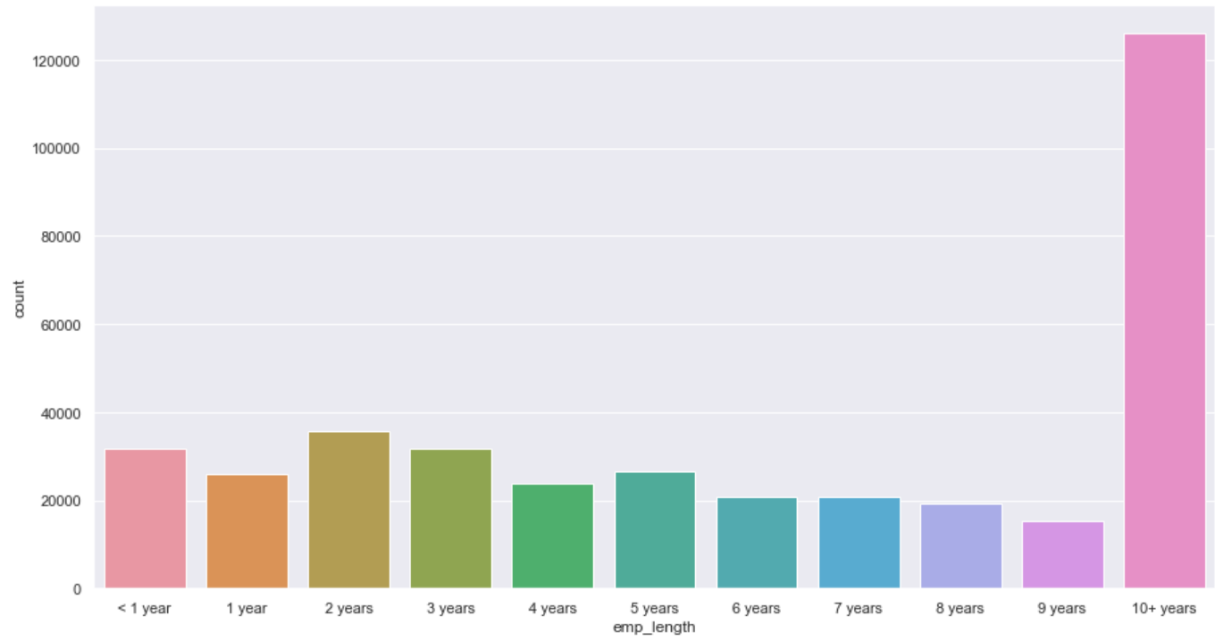


**Figure 8: Count plot of emp_length.**

To judge the relationship between the emp_length and the loan_status a count plot is made. From the plot in Fig. 9, it is clear that there is some relationship with the number of years employed versus the length an employee has been employed.

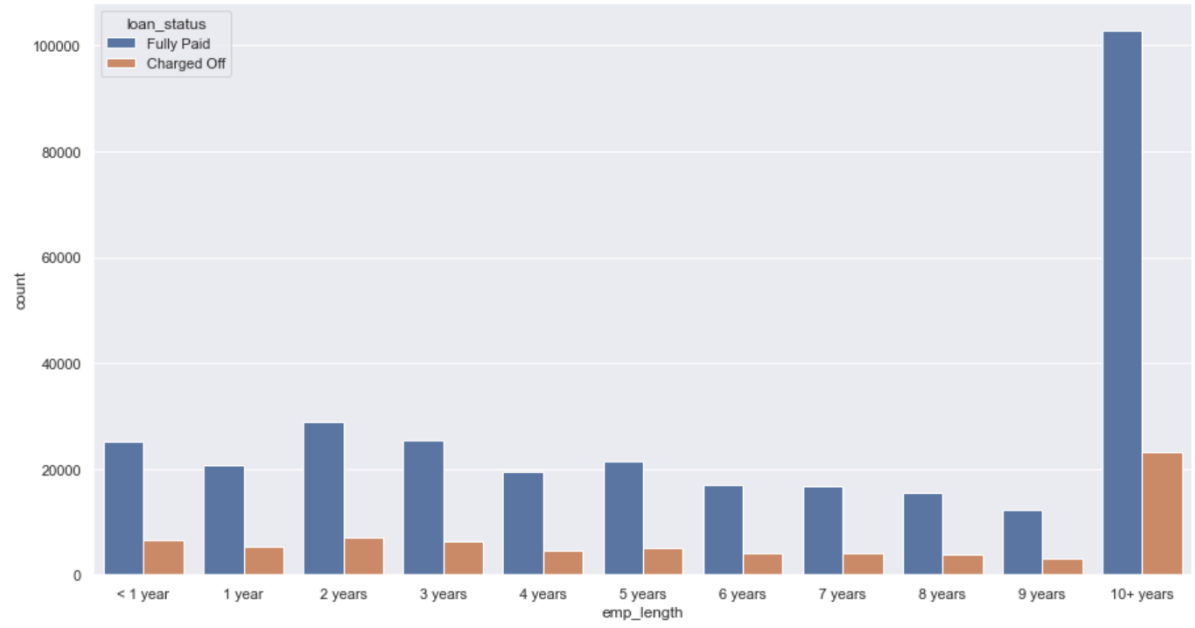


**Figure 9: Count plot of emp_length grouped by loan_status.**

Next, the percentage of charge off per employment category is analyzed by creating a bar plot. From the bar plot it is evident that Employment length doesn't seem to vary much between different charge off rates. Since employment length does not make any significant contribution to the outcome, we can remove this column.

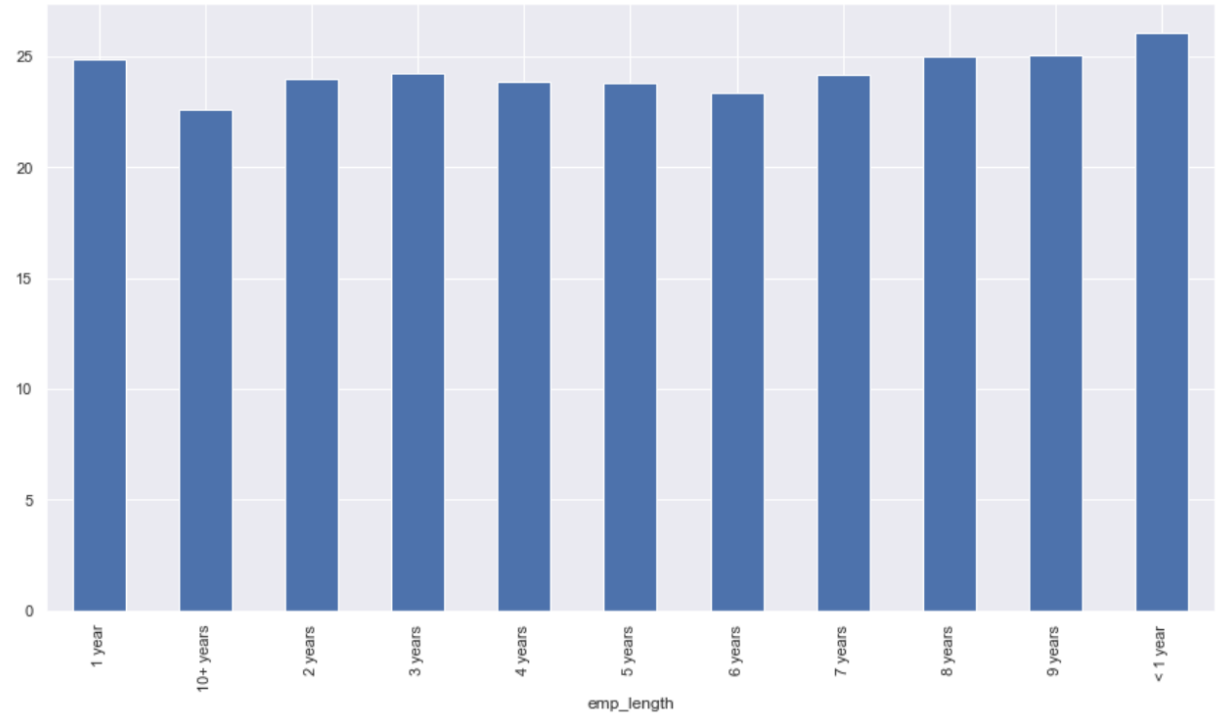


**Figure 10: Bar plot of emp_length grouped by percentage of charge off's by fully paid.**

A simple analysis of printing the unique values in purpose and title shows that title is a much more detailed version of the purpose. It does not give us a lot of insight for categorizing since every column has a different value for title, therefore, the 'title' column can be dropped. The term is taken care of by keeping only an integer of the actual string, so '36 months' became '36'. Grade column can be dropped since it can be found from the sub_grade. Home_ownership column unique values were printed to see the distinction, NONE and OTHER categories were then converted to OTHER and dummy variables were created to add to the original data frame. From the address, the zip codes were extracted and added to a new column called zip_code. A dummy value of all the zip codes were created and added to the data frame. Only the year from the earliest_cr_line is retained, and the month is discarded.

## Train Test Split

The first step is to import the train_test_split model is from the sklearn model_selection library. Then from some analysis it can be seen that issue_d needs to be converted to int, so only the year is now retained. Loan_status column is dropped because it was converted to 1 or 0 in a new column called loan_repaid. The X is set as all the columns except the 'loan_repaid' column and the y is set as 'loan_repaid' column. Then the train_test_split is called on the X and y with 20% test size and 101 random states. A scaler object is created which is used to fit transform and transform the x train and test sets.

## Creating The Model

A sequential model is created with the input layer having 78 with relu activation. Then three more dense layers are added with the relu activation function with 58, 38 and 18 neurons and a final dense layer is added with the sigmoid activation function for binary classification. In between dropout layers are added to avoid the overfitting problem. Model is then compiled using binary cross entropy loss function and the adam optimizer. Model is then fitted with the training data and the test data as the validation data with 25 epochs and 256 batch size. The model is then saved as loan_data_model.hf, for later access in the notebook.

## Evaluating Model Performance

The model is evaluated by plotting the loss and validation loss from the model history. From the plot we can see that validation loss stays constant as the loss decreases steadily forming a downward sloping concave curve as seen in Fig. 10.

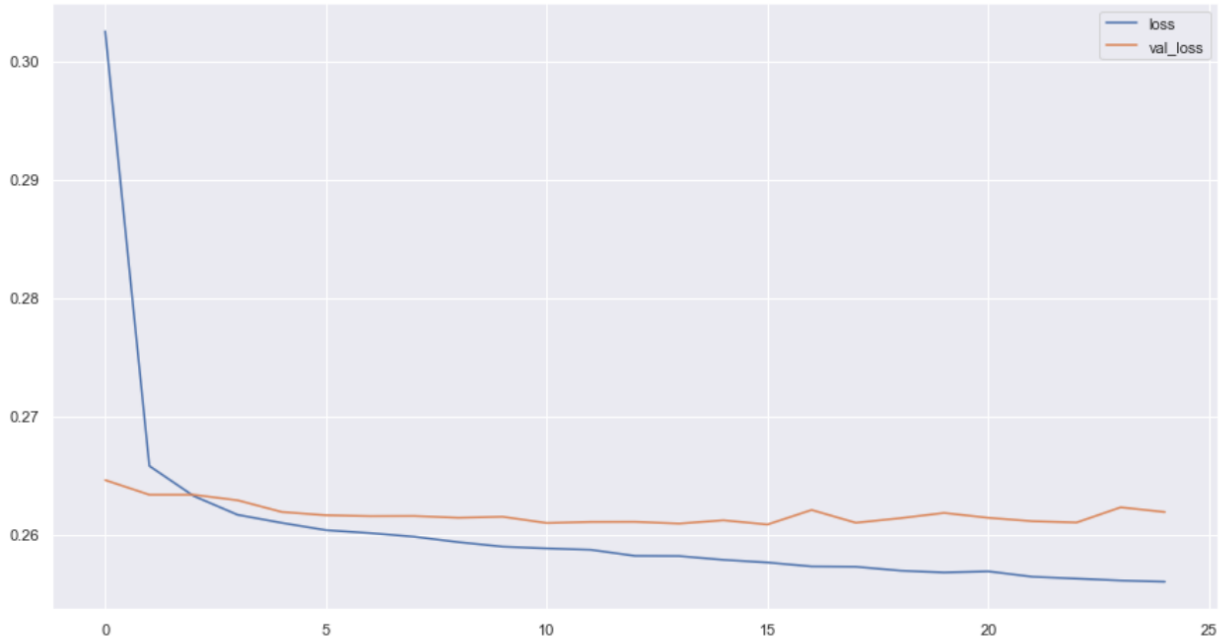


**Figure 11: Loss and validation loss curves.**

The predict function is then called on the model, and the predicted values are converted from a series of decimal values to 0 or 1 depending on the 0.5 threshold. A classification report and confusion matrix are made which indicates 89% accuracy. To further test out this mode a random row from the data frame is selected and prediction is made on the row by the model to get see if the person should be approved for receiving loans. On running the function, a couple different times, the predictions are made correctly. Therefore, this model is good for use on new data.