D8TA Final Project - Credit Card Approval

Shreya Balaji, Rohit Manimaran, Vishal Menon

Credit Approval Process

Goal and Dataset Summary

- Goal: Check for data privacy and fairness for the credit card dataset. We hope to remove any unfair discrimination when it comes to the credit card approval process.
    - Fair Equality of Opportunity (FEO)

- Application Record table contains general information on each applicant such as number of children, marital status, car ownership, etc

- Credit Record table contains information on the applicant's payment history and how long they have had an account

1. Application: Borrower applies for credit
2. ***Creditworthiness Assessment: Lender evaluates ability to repay and assesses risk***
3. Terms and Conditions: Lender sets interest rate, repayment schedule, and collateral requirements
4. Approval/Rejection: Lender approves or rejects based on creditworthiness and risk.
5. Disbursement: Lender disburses funds if approved

Preparing the Data

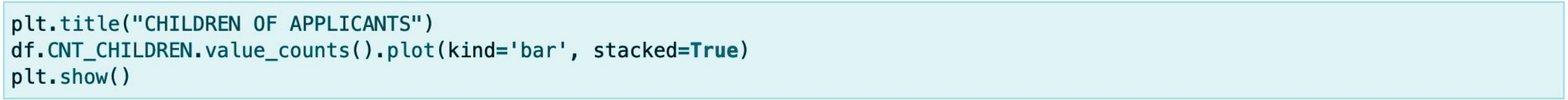
- Adjusted the status values to improve the assessment of applicant fitness

- Eliminated duplicate entries for more accurate analysis

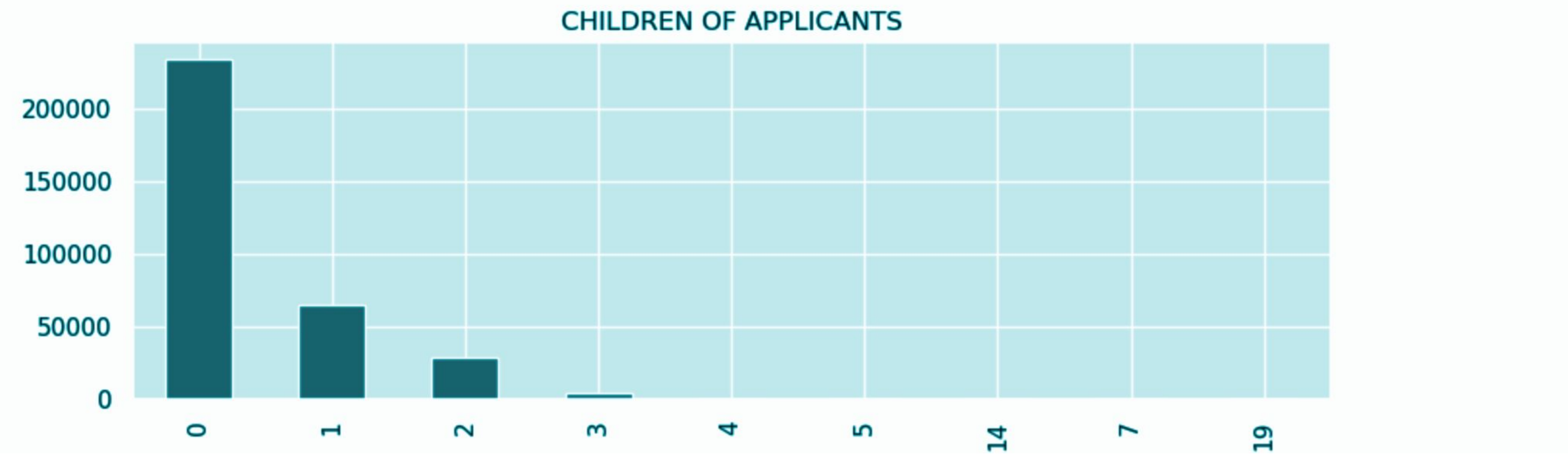
- Converted all "yes" or "no" responses to binary values (0/1)

- Addressed missing values to ensure complete and reliable data for analysis

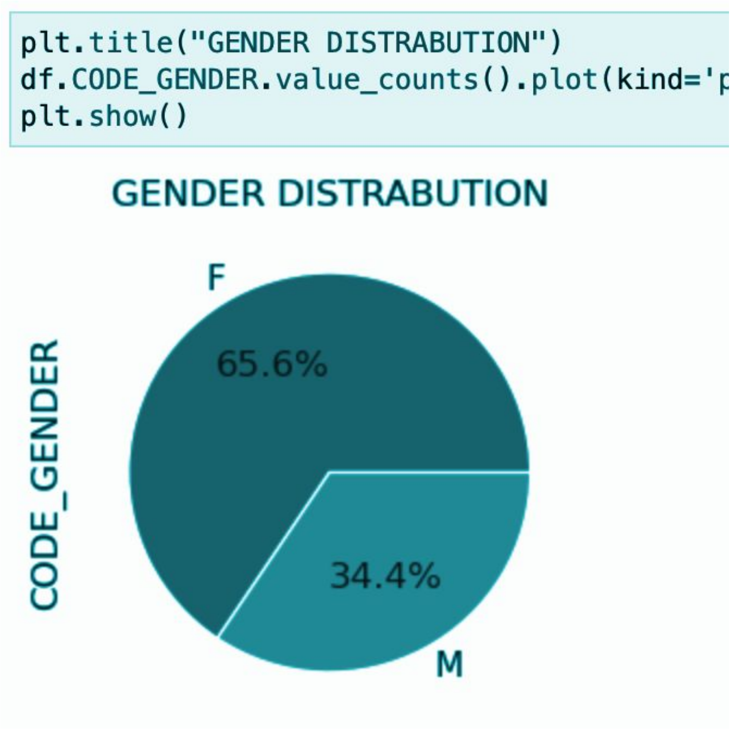
Exploratory Analysis - 1



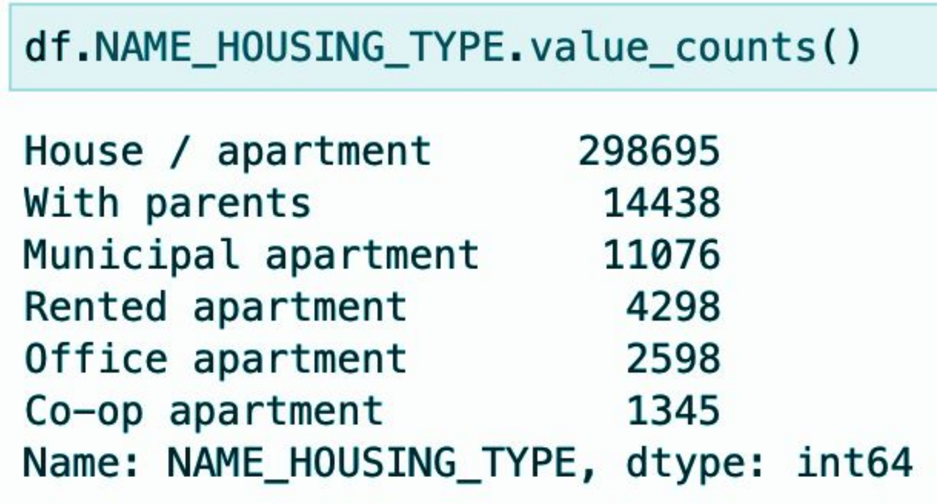
```
plt.title("CHILDREN OF APPLICANTS")
df.CNT_CHILDREN.value_counts().plot(kind='bar', stacked=True)
plt.show()
```



Exploratory Analysis - 2



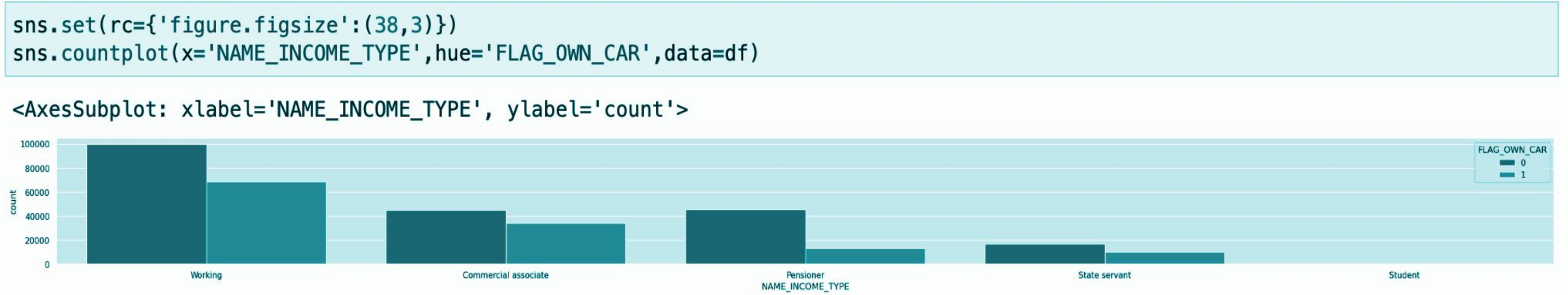
```
plt.title("GENDER DISTRABUTION")
df.CODE_GENDER.value_counts().plot(kind='p
plt.show()
```



```
df.NAME_HOUSING_TYPE.value_counts()
```

```
House / apartment      298695
With parents            14438
Municipal apartment     11076
Rented apartment         4298
Office apartment         2598
Co-op apartment          1345
Name: NAME_HOUSING_TYPE, dtype: int64
```

Exploratory Analysis - 3



```
sns.set(rc={'figure.figsize':(38,3)})
sns.countplot(x='NAME_INCOME_TYPE',hue='FLAG_OWN_CAR',data=df)
```

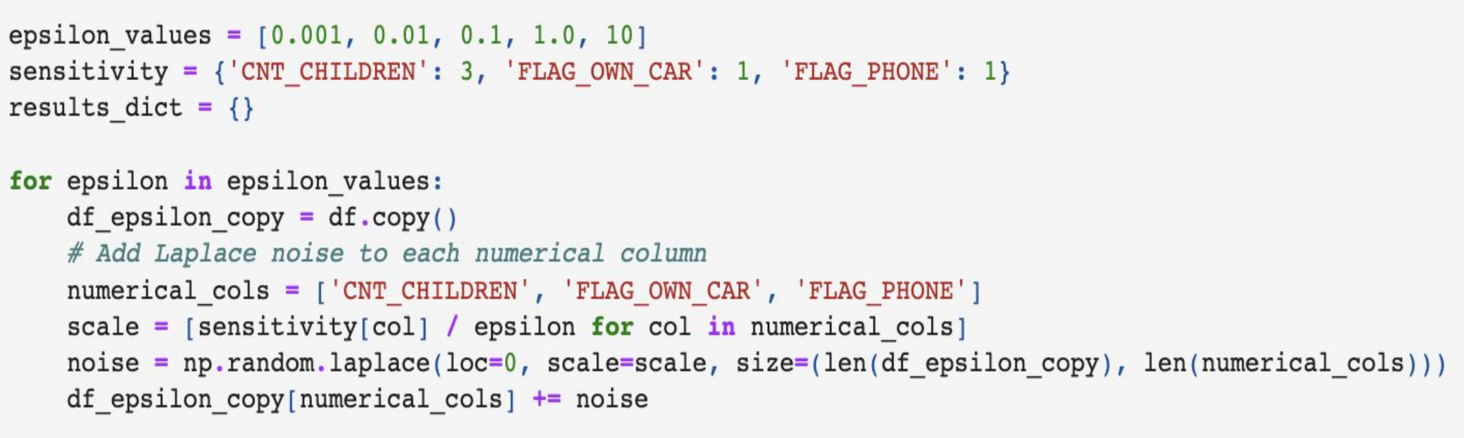
```
<AxesSubplot: xlabel='NAME_INCOME_TYPE', ylabel='count'>
```

Labeling

- Developed a method to classify applicants as "good" or "bad"
- Ran a pearson correlation against Status to determine what factors we need to use to create an equation that calculates available funds after accounting for expenses such as children, car, phone, and housing
- Dividing this value by total income yields the percentage of funds available for other expenses
- A higher percentage indicates a better potential credit card owner
- Compared this to status attribute to give us a baseline on each applicant

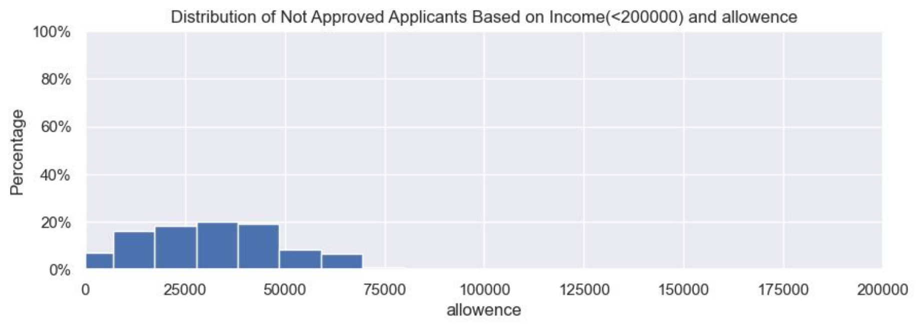
```
Income - ((Avg cost of child * # of children) + Avg cost of car ownership (flag) + Avg cost of owning a phone (flag) + Avg cost of housing)
```

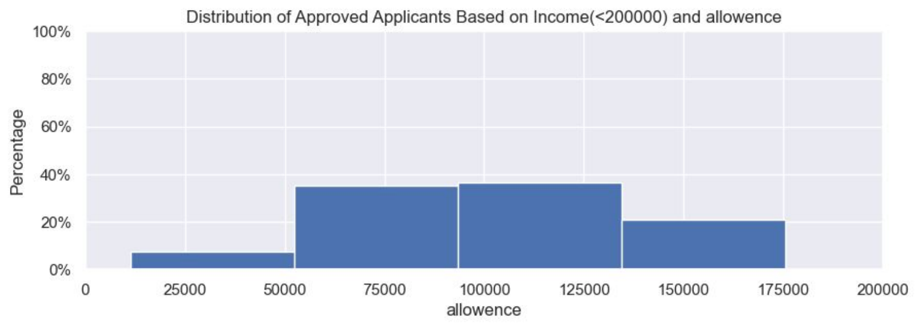
Differential Privacy

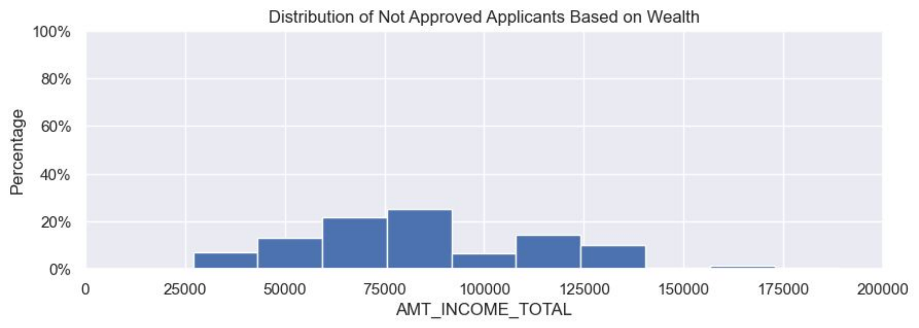


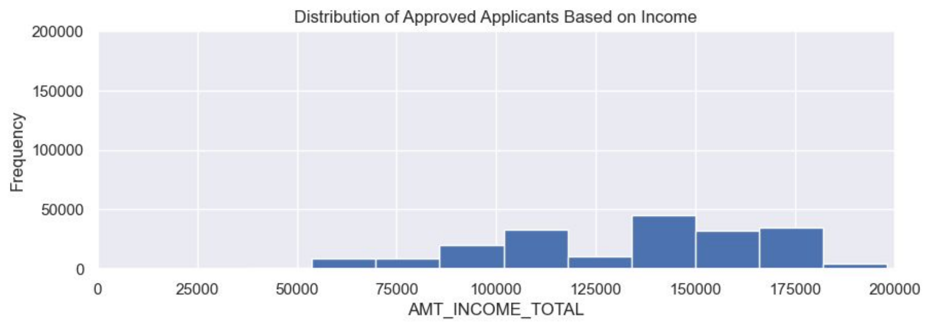
```python
epsilon_values = [0.001, 0.01, 0.1, 1.0, 10]
sensitivity = {'CNT_CHILDREN': 3, 'FLAG_OWN_CAR': 1, 'FLAG_PHONE': 1}
results_dict = {}

for epsilon in epsilon_values:
    df_epsilon_copy = df.copy()
    # Add Laplace noise to each numerical column
    numerical_cols = ['CNT_CHILDREN', 'FLAG_OWN_CAR', 'FLAG_PHONE']
    scale = [sensitivity[col] / epsilon for col in numerical_cols]
    noise = np.random.laplace(loc=0, scale=scale, size=(len(df_epsilon_copy), len(numerical_cols)))
    df_epsilon_copy[numerical_cols] += noise
```

Columns with added Laplace noise: # of children, car ownership, phone ownership


Distribution of Not Approved Applicants Based on Income(<200000) and allowence
100%
80%
60%
40%
20%
0%
Percentage
0
25000
50000
75000
100000
125000
150000
175000
200000
allowence


Distribution of Approved Applicants Based on Income(<200000) and allowence
100%
80%
60%
40%
20%
0%
Percentage
0
25000
50000
75000
100000
125000
150000
175000
200000
allowence


Distribution of Not Approved Applicants Based on Wealth
100%
80%
60%
40%
20%
0%
Percentage
0
25000
50000
75000
100000
125000
150000
175000
200000
AMT_INCOME_TOTAL

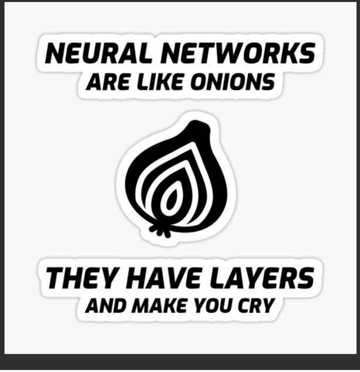

Distribution of Approved Applicants Based on Income
200000
150000
100000
50000
0
Frequency
0
25000
50000
75000
100000
125000
150000
175000
200000
AMT_INCOME_TOTAL

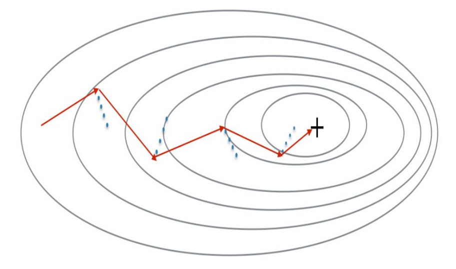
Model Evaluation

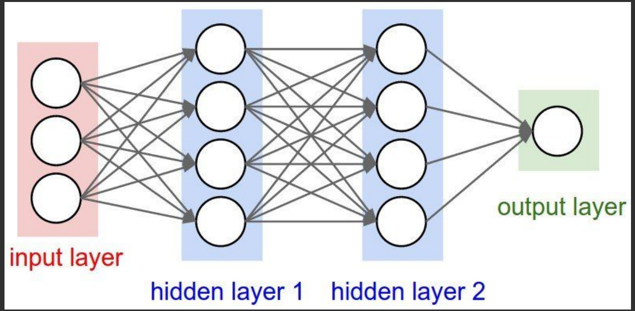
- Loss: difference between predicted output and actual during training
- Accuracy: percentage correct/total predictions
- Val_loss: loss measure but with unseen data
- Val_Accuracy: accuracy but with unseen data

Neural Network Prediction Model

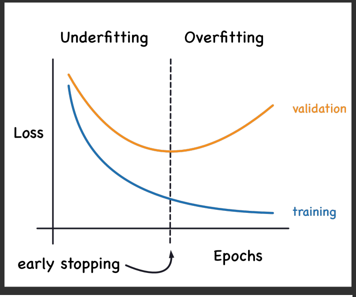
- Three layer neural network
  - 32, 16, and 1 output layer
- Optimization:
  - Adam optimizer: combines momentum and learning rate decay to minimize the loss function
  - L2 Regularization: adds a penalty to the loss function, lowers the weight of certain nodes
- Loss graph shows how well the model is performing during training
- Inputs:
  - AMT_INCOME_TOTAL: Applicant's total income
  - CNT_CHILDREN: Number of children the applicant has
  - FLAG_OWN_CAR: Whether the applicant owns a car or not (binary: 0 or 1)
  - FLAG_PHONE: Whether the applicant provided a mobile phone number (binary: 0 or 1)
- Output:
  - Predicts the credit quality of the applicant (binary: 0 or 1)

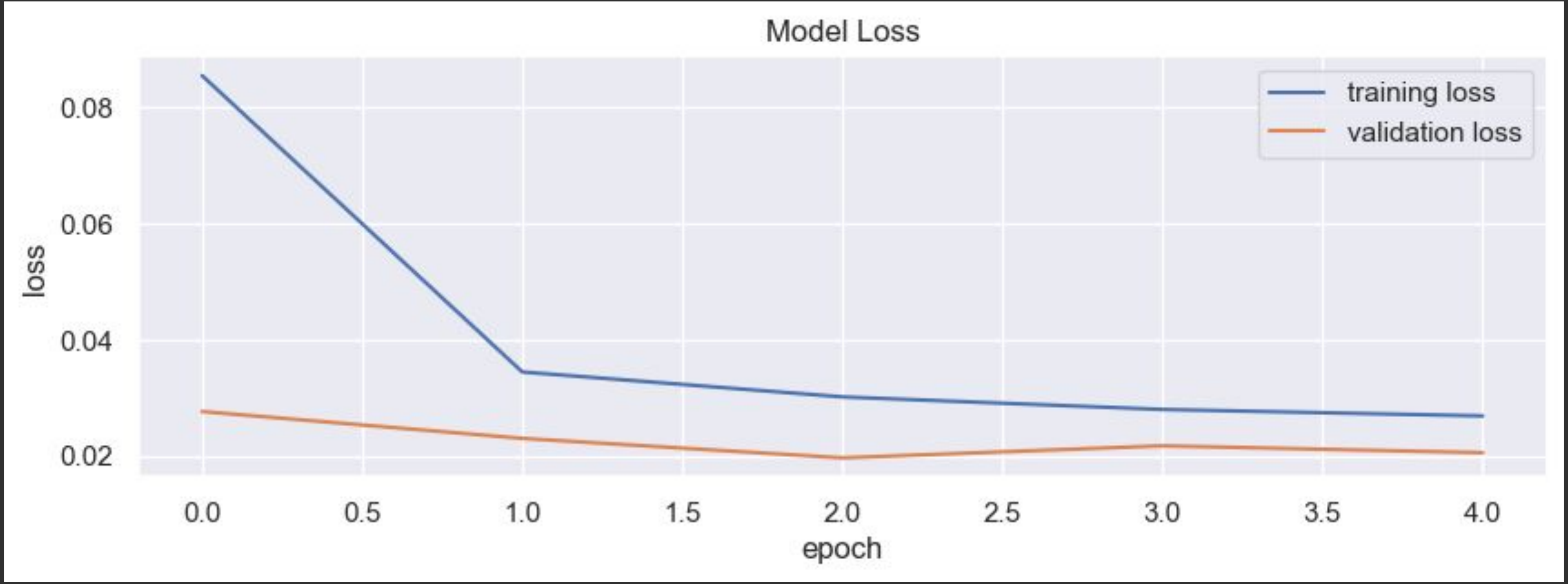






Loss Plot (w/out DP)



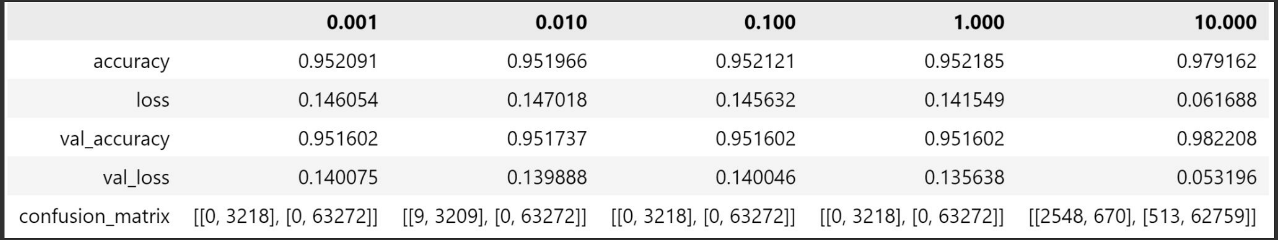


- Training loss decreases faster than the validation loss, indicating that the neural network is not overfitting

Determining the best epsilon value for DP

```
(Without DP loss): 0.0269 - accuracy: 0.9947 - val_loss: 0.0206 - val_accuracy: 0.9971
```

- as epsilon increases the accuracy and loss values were stable
- shows us that the DP added is not significantly affecting the model’s performance
- **0.001:** model misclassified 3218 neg samples as positive, correctly classified all positive samples
  - model is more prone to false positives without DP
- **0.01:** fewer false positives than 0.001
  - increasing epsilon can help reduce the number of false positives
    - see similar trend for the other values as you increase epsilon
- **10:**
  - Accuracy and validation accuracy are significantly higher than without differential privacy
  - Loss and validation loss are much lower than without differential privacy
  - Confusion matrix shows some true positive predictions, indicating improved performance



	0.001	**0.010**	**0.100**	**1.000**	**10.000**
accuracy	0.952091	0.951966	0.952121	0.952185	0.979162
loss	0.146054	0.147018	0.145632	0.141549	0.061688
val_accuracy	0.951602	0.951737	0.951602	0.951602	0.982208
val_loss	0.140075	0.139888	0.140046	0.135638	0.053196
confusion_matrix	[[0, 3218], [0, 63272]]	[[9, 3209], [0, 63272]]	[[0, 3218], [0, 63272]]	[[0, 3218], [0, 63272]]	[[2548, 670], [513, 62759]]

Takeaways

- Epsilon trade-offs: Selecting an epsilon value involves balancing privacy and accuracy
  - Smaller epsilon value: Provides stronger privacy but may result in less accuracy
  - Greater epsilon value: Provides less privacy but may lead to more accuracy
- DP protects personal information: Adding differential privacy makes it harder for personal information to be used for discriminatory or malicious purposes
- EDA is a pivotal step in Data Analysis… prevents downstream issues
- Coursera is free if you have a California public library card
- Kids are expensive

Thanks for tuning in.