Identifying Delinquent Loans.

RICARDO ALAMO SALGADO - PROJECT PORTFOLIO

What is Freddie Mac?

- Freddie Mac was chartered by the US Congress in 1970 to support the U.S. housing finance system and to help ensure a reliable and affordable supply of mortgage funds across the country. Rather than lending directly to borrowers, Freddie Mac operates in the U.S. secondary mortgage market, buying loans that meet our standards from approved lenders.
- Those lenders are then, in turn, able to provide more loans to qualified borrowers and keep capital flowing into the housing market. Freddie Mac then pools the mortgages it buys into securities, which they sell to investors around the world.

- Source: https://www.freddiemac.com/about

Role and Objective

I’ve been hired from Freddie Mac to create a model that identifies potential delinquent loans. That way it can price those risks when buying the securities and get a fair deal. This is crucial since Freddie mac is trying to keep the mortgage market healthy for all US citizens who want to buy a property.

- A good industry standard to follow is available from Turiel, J. & Aste, T. (2019) who are researchers at the University College London. They got a recall score of **77.4%**, a similar score must be possible.

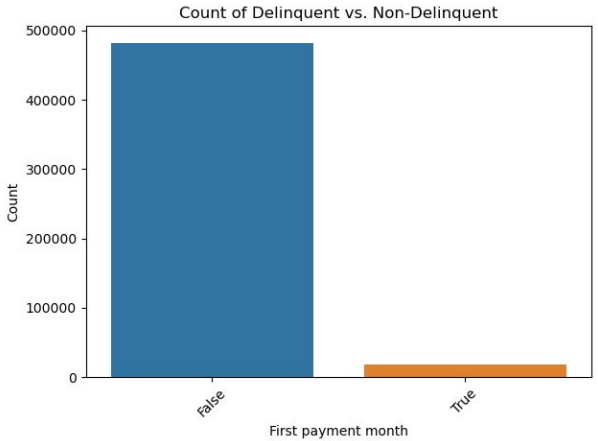
-My objective is to develop an ML model that can achieve the highest possible recall score for predicting delinquent loans. By doing so, Freddie Mac can buy mortgage securities at the correct price and avoid the negative impact of delinquent loans on the market."

Data

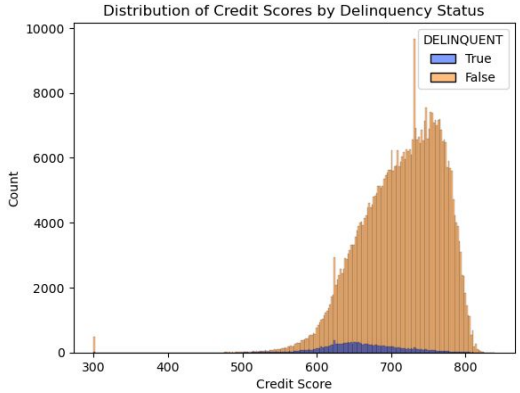
```
RangeIndex: 500137 entries, 0 to 500136
Data columns (total 32 columns):
 #   Column                            Non-Null Count   Dtype
---  ------                            --------------   -----
 0   CREDIT_SCORE                      497426 non-null  float64
 1   FIRST_PAYMENT_DATE                500137 non-null  datetime64[ns]
 2   FIRST_TIME_HOMEBUYER_FLAG         369578 non-null  object
 3   MATURITY_DATE                     500137 non-null  datetime64[ns]
 4   METROPOLITAN_STATISTICAL_AREA     429988 non-null  float64
 5   MORTGAGE_INSURANCE_PERCENTAGE     449089 non-null  float64
 6   NUMBER_OF_UNITS                   500134 non-null  float64
 7   OCCUPANCY_STATUS                  500137 non-null  object
 8   ORIGINAL_COMBINED_LOAN_TO_VALUE   500124 non-null  float64
 9   ORIGINAL_DEBT_TO_INCOME_RATIO     485208 non-null  float64
 10  ORIGINAL_UPB                      500137 non-null  int64
 11  ORIGINAL_LOAN_TO_VALUE            500128 non-null  float64
 12  ORIGINAL_INTEREST_RATE            500137 non-null  float64
 13  CHANNEL                           500137 non-null  object
 14  PREPAYMENT_PENALTY_MORTGAGE_FLAG  494959 non-null  object
 15  PRODUCT_TYPE                      500137 non-null  object
 16  PROPERTY_STATE                    500137 non-null  object
 17  PROPERTY_TYPE                     500042 non-null  object
 18  POSTAL_CODE                       500106 non-null  float64
 19  LOAN_SEQUENCE_NUMBER              500137 non-null  object
 20  LOAN_PURPOSE                      500137 non-null  object
 21  ORIGINAL_LOAN_TERM                500137 non-null  int64
 22  NUMBER_OF_BORROWERS               499890 non-null  float64
 23  SELLER_NAME                       500137 non-null  object
 24  SERVICER_NAME                     500137 non-null  object
 25  PREPAID                           500137 non-null  bool
 26  DELINQUENT                        500137 non-null  bool
 27  temp                              500137 non-null  object
 28  FIRST_PAYMENT_DATE_y              500137 non-null  int64
 29  FIRST_PAYMENT_DATE_m              500137 non-null  int64
 30  MATURITY_DATE_y                   500137 non-null  int64
 31  MATURITY_DATE_m                   500137 non-null  int64
```

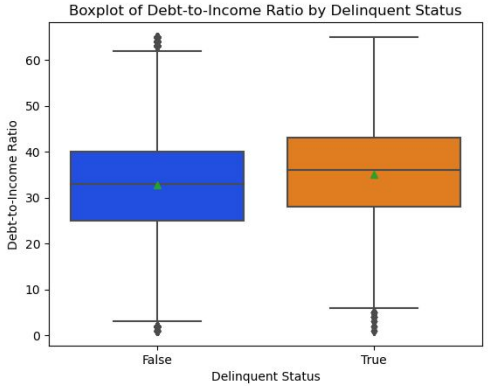
Dataset contains 32 columns and 500,137 rows with numerical, categorical and datetime columns.

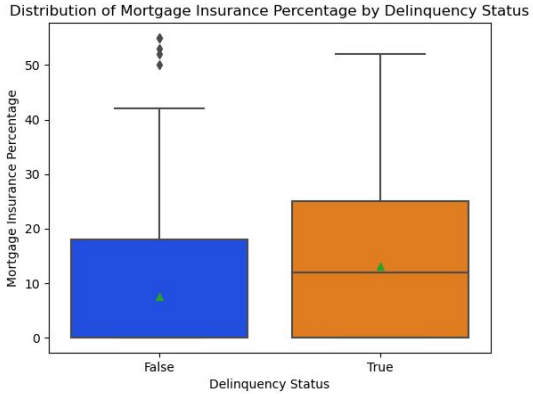
An imbalance label with 96% non delinquent and 4% delinquent.(payment delay of more than 90 days)

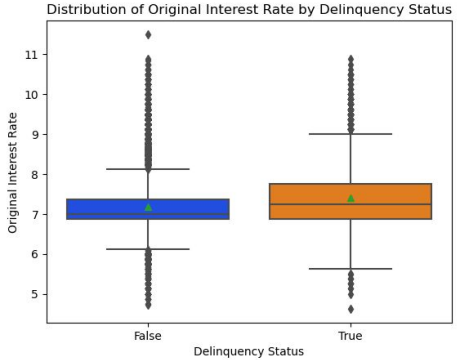


EDA





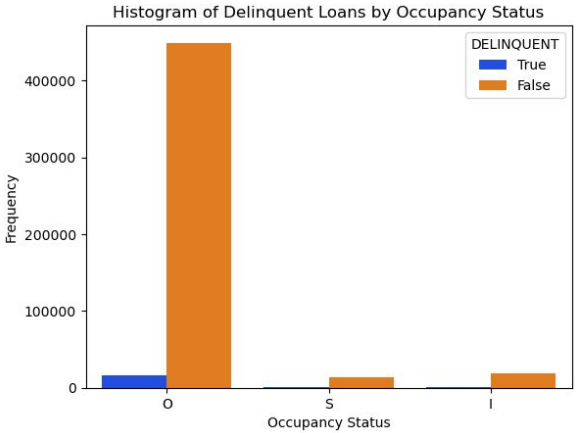


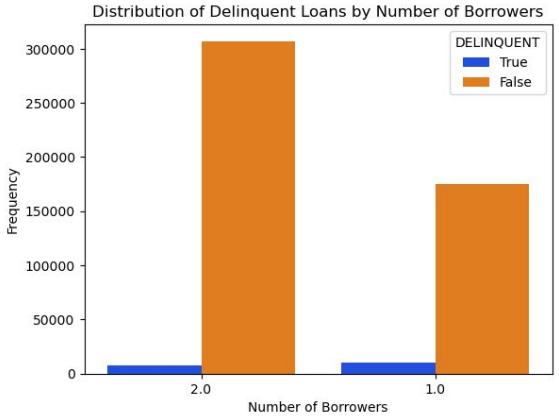


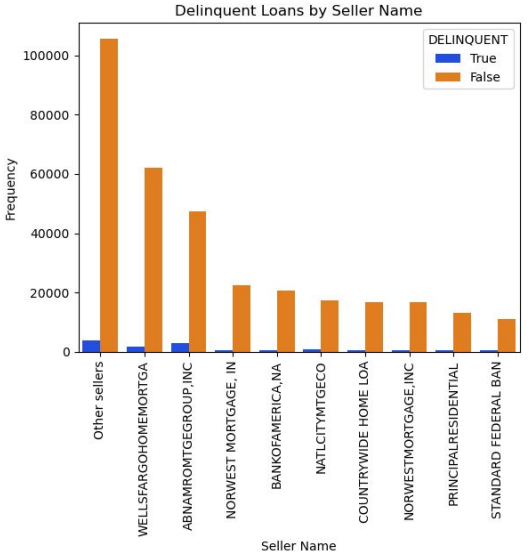
A delinquent customer has the following characteristics in general:

- Lower credit score
- Higher Debt-to-Income
- A larger mortgage insurance
- But not necessarily a higher interest rate.
- Riskier loans should have higher interest but the data is not showing that.

EDA





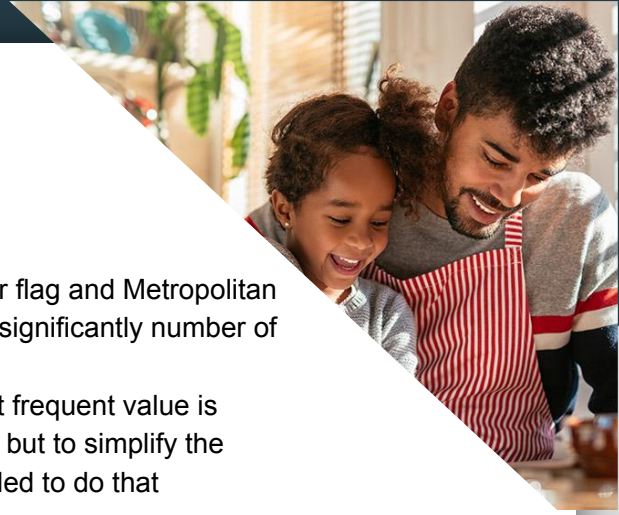


- Bigger defaults for Occupied by owner than Secondary or Investment properties
- Loans are safer when 2 people co-sign.
- Some banks are riskier than others.

NULL VALUES

```
In [57]: df.isna().sum().sort_values(ascending=False)
```

```
Out[57]: FIRST_TIME_HOMEBUYER_FLAG          130559
         METROPOLITAN_STATISTICAL_AREA       70149
         MORTGAGE_INSURANCE_PERCENTAGE       51048
         ORIGINAL_DEBT_TO_INCOME_RATIO       14929
         PREPAYMENT_PENALTY_MORTGAGE_FLAG     5178
         CREDIT_SCORE                         2711
         NUMBER_OF_BORROWERS                   247
         PROPERTY_TYPE                          95
         POSTAL_CODE                            31
         ORIGINAL_COMBINED_LOAN_TO_VALUE        13
         ORIGINAL_LOAN_TO_VALUE                  9
         NUMBER_OF_UNITS                         3
```



- First time home buyer flag and Metropolitan Statistical Area have significantly number of nulls
- Filling it with the most frequent value is probably not the best but to simplify the preprocessing I decided to do that

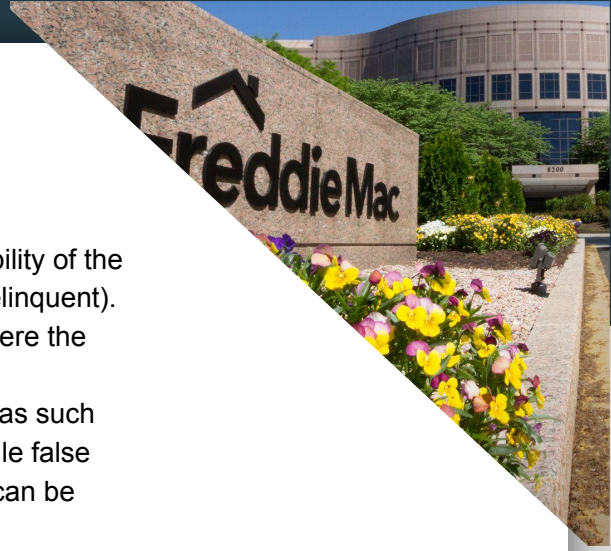
ENCODERS AND PREPROCESSING

- One hot col encoder for categories with few options.
- Target encoder for categories with plenty of options to avoid high dimensionality.
- Robust Scaler for the numerical columns.
- Smote for the imbalance label.

In [473]:
```
preprocessor = ColumnTransformer(transformers=[
    ('one_hot_encoder', one_hot_encoder, one_hot_cols),
    ('target_encoder', target_encoder, target_encode_cols),
    ('robust_scaler',RobustScaler(),numerical_cols) #Robust Scaler for linear models
])
```

In [383]:
```
LR_pipeline = Pipeline([
    ('preprocessor', preprocessor), #calling preprocessor
    ('oversampler', SMOTE()), #Using SMOTE as oversampler
    ('model', LogisticRegression(max_iter=1000)) # Base Model no Hyper Tuning
])
```

Score Metric - Recall

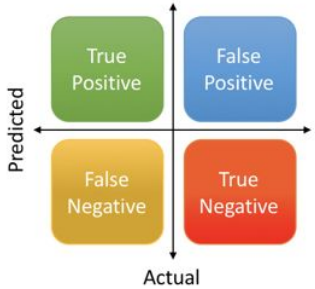


- Recall is an important metric for identifying delinquent loans because it measures the ability of the model to correctly identify all positive cases (i.e., the cases where the loan is actually delinquent). In other words, recall measures the model's ability to avoid false negatives, or cases where the model incorrectly predicts that a loan is not delinquent when it actually is.
- This is because false negatives (loans that are actually delinquent but are not identified as such by the model) can have serious consequences for both the lender and the borrower, while false positives (loans that are incorrectly identified as delinquent) are usually less costly and can be addressed through further investigation.

$$\textbf{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

$$\textbf{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

$$\textbf{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total}}$$



ML Models - XGBoost and Random Forest

- XGBoost and Random Forest perform poorly when taking in consideration recall for the delinquent customers.
- XGboost improved significantly after hypertuning with a Grid Search

	precision	recall	f1-score	support
False	0.97	1.00	0.98	96426
True	0.45	0.08	0.14	3602
accuracy			0.96	100028
macro avg	0.71	0.54	0.56	100028
weighted avg	0.95	0.96	0.95	100028

Base Model

	precision	recall	f1-score	support
False	0.98	0.92	0.95	96426
True	0.17	0.43	0.24	3602
accuracy			0.90	100028
macro avg	0.57	0.67	0.59	100028
weighted avg	0.95	0.90	0.92	100028

Model with hyperparameters tuned

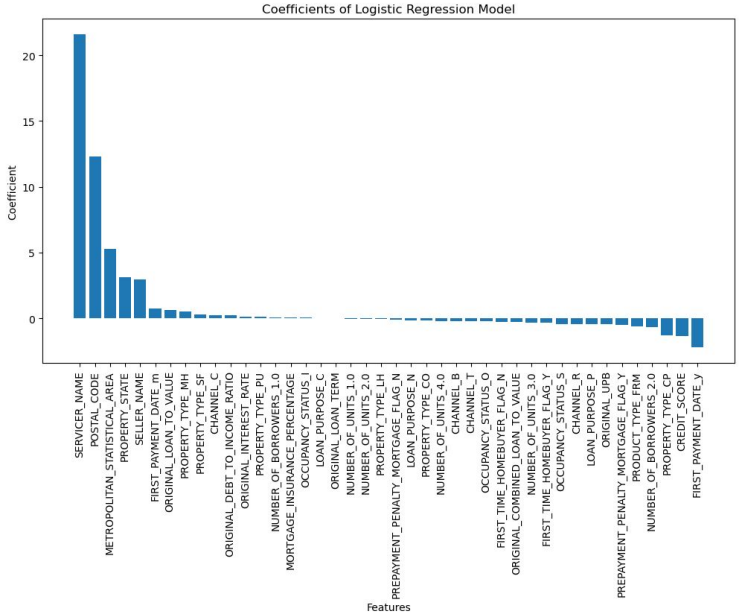
ML Models - Logistic Regression.

- The logistic regression model is great at recall.
- The feature with the biggest coefficients are servicer name (bank), Postal Code, Metropolitan Area, First Payment year, Credit Score, Property Type.
- The penalty is taking some elements almost to 0.

```
In [810]: print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
False	0.99	0.77	0.87	96426
True	0.11	0.77	0.20	3602
accuracy			0.77	100028
macro avg	0.55	0.77	0.53	100028
weighted avg	0.96	0.77	0.84	100028

Model with
hyperparameters tuned



Voting Classifier and Stacking

- The voting classifier and Stacking Classifier did not improved significantly the recall score.

STACKING

In [714]:
```
clf = StackingClassifier(estimators=[
    ('XGB', XGBClassifier(eta= 0.05,max_depth= 3, objective= 'binary:logistic')), # Best XG boost
    ('RF', RandomForestClassifier()), # Trying again with Random Forest
    ('LR', LogisticRegression(max_iter=1000,solver='lbfgs',C=0.01)), # LR since is the best until now
    ('LR2', LogisticRegression(max_iter=1000)) # LR a base model logistic regresion
    ],
    final_estimator=LogisticRegression(max_iter=1000,solver='liblinear', penalty='l1',C=0.68))
    # perhaps using a logistic regression as a final estimator will help the model

clf_pipe = Pipeline([
    ('preprocessor',preprocessor),
    ('oversampler', SMOTE()),
    ('model', clf)
])
```

	precision	recall	f1-score	support
False	0.97	0.99	0.98	96426
True	0.33	0.11	0.16	3602
accuracy			0.96	100028
macro avg	0.65	0.55	0.57	100028
weighted avg	0.94	0.96	0.95	100028

LIME - Logistic Regression

- Customer 88346 was predicted as non-delinquent when it was delinquent.
- With a credit score of 724 which is very good is really hard to predict that this customer would default.
- Also it seems that the postal code is also full of non_delinquents so again the model struggles to identify it as Non Delinquent.

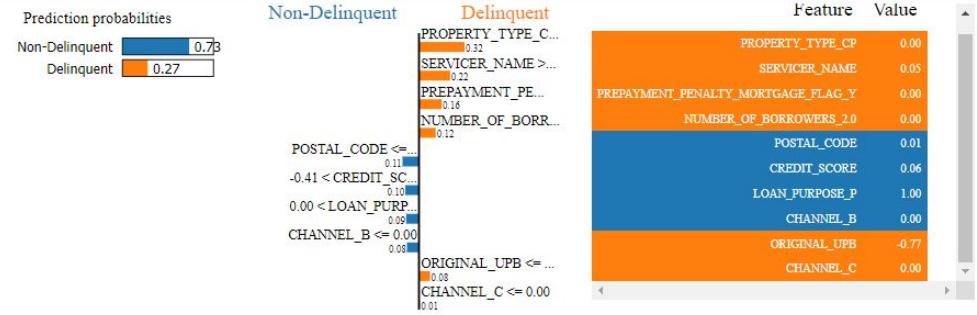
```
In [763]: X.iloc[88346]

Out[763]: FIRST_TIME_HOMEBUYER_FLAG                      Y
          NUMBER_OF_UNITS                              1.0
          OCCUPANCY_STATUS                             O
          CHANNEL                                      R
          PREPAYMENT_PENALTY_MORTGAGE_FLAG             N
          PRODUCT_TYPE                               FRM
          PROPERTY_TYPE                               CO
          LOAN_PURPOSE                                 P
          NUMBER_OF_BORROWERS                          1.0
          METROPOLITAN_STATISTICAL_AREA            35644.0
          PROPERTY_STATE                              NJ
          POSTAL_CODE                              7600.0
          SELLER_NAME                GMAC MORTGAGE CORPOR
          SERVICER_NAME                   Other servicers
          FIRST_PAYMENT_DATE_y                       1999
          FIRST_PAYMENT_DATE_m                          3
          CREDIT_SCORE                              724.0
          MORTGAGE_INSURANCE_PERCENTAGE               0.0
          ORIGINAL_COMBINED_LOAN_TO_VALUE            80.0
          ORIGINAL_DEBT_TO_INCOME_RATIO              47.0
          ORIGINAL_UPB                              61000
          ORIGINAL_LOAN_TO_VALUE                     80.0
          ORIGINAL_INTEREST_RATE                      6.5
          ORIGINAL_LOAN_TERM                          360
          Name: 88346, dtype: object
```

```
In [862]: Customer_index= 88346

exp= explainer.explain_instance(X_test_preprocessed.loc[Customer_index],LR_best_model.predict_proba)
exp.show_in_notebook()

C:\Users\Richard\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LogisticRegression was fitted with feature names
  warnings.warn(
```



Deep Learning MLP

- Almost as good as the Logistic Regression. But it would be hard to comply with:
- Fair Housing Act: The Fair Housing Act, 42 U.S.C. 3601 et seq., prohibits discrimination by direct providers of housing, banks or other lending institutions and homeowners insurance companies whose discriminatory practices make housing unavailable to persons because of race or color, religion, sex ,national origin,familial status, or disability.

MLP with 6 neurons instead of 3

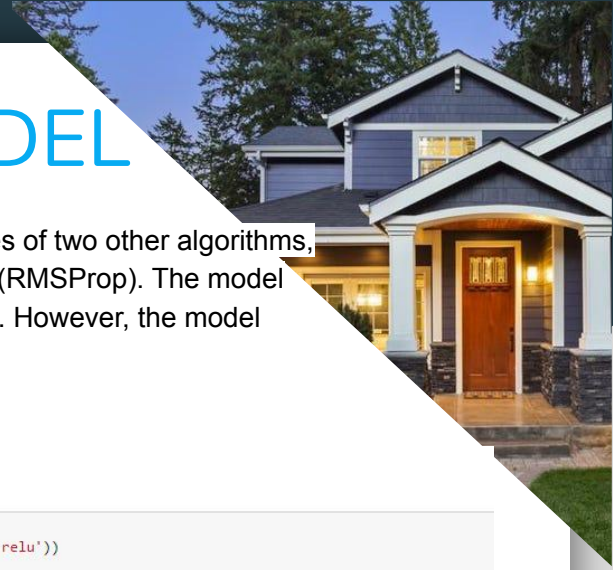
```
In [512]: mlp_6 = MLPClassifier(hidden_layer_sizes=(6, 2), activation='relu', solver='adam', random_state=42)
```

```
In [514]: MLP_6_pipeline = Pipeline([
              ('preprocessor', preprocessor), #calling preprocessor
              ('oversampler', SMOTE()), #Using SMOTE as oversampler
              ('model', mlp_6)
          ])
```

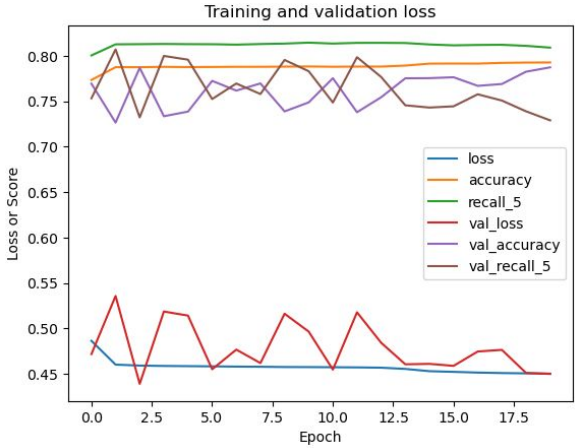
```
In [534]: scores
Out[534]: array([0.75747047, 0.74079222, 0.75191105, 0.69041001, 0.6903024 ])

In [535]: scores.mean(), scores.std()
Out[535]: (0.7261772274078062, 0.029736817865622604)
```

Deep Learning TF- KERAS MODEL



- I decided to use the Adam optimization algorithm because it combines the advantages of two other algorithms, namely Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp). The model achieved a good score with three layers and six, three, and one neurons respectively. However, the model cannot be used due to the constraints of the Fair Housing Act.



2nd Keras model with 3 layers

In [656]:
```
keras_model_9_3 = Sequential()
keras_model_9_3.add(Dense(6, input_dim=42, activation='relu'))
keras_model_9_3.add(Dense(3, activation='relu'))
keras_model_9_3.add(Dense(1, activation='sigmoid'))
keras_model_9_3.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy', Recall()])
```

In [657]:
```
keras_model_9_3.summary()
# In this case we have 3 layers, one with 6 neurons, second with 3 nad the last for the classification decision
```

```
Model: "sequential_11"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_23 (Dense)            (None, 6)                 258

 dense_24 (Dense)            (None, 3)                 21

 dense_25 (Dense)            (None, 1)                 4
```

In [651]:
```
keras_model_9_3.evaluate(X_test_preprocessed,y_test)
```

```
3126/3126 [==============================] - 2s 739us/step - loss: 0.4506 - accuracy: 0.7888 - recall_5: 0.7404
```

Out[651]:
```
[0.4506392180919647, 0.788759171962738, 0.7404220104217529]
```

Bonus

What if we only have the numeric columns?

In my experience working with clients, not all the information is always available, especially for categorical columns as it often depends on the loan type and seller. However, the numeric features play an essential role in loan approval. Therefore, a simplistic model that only includes numeric values makes sense in this case.

The logistic regression still delivers a good recall with just the numeric columns. Recall: 0.75

```
In [838]: Simplistic_LR_pipeline.fit(X_train,y_train)

Out[838]: Pipeline(steps=[('impute', SimpleImputer()), ('scaler', RobustScaler()),
                          ('oversampler', SMOTE()),
                          ('model', LogisticRegression(max_iter=1000))])

In [839]: y_pred= Simplistic_LR_pipeline.predict(X_test)

In [840]: print(classification_report(y_test,y_pred))
          # Great recall of 0.75
```

	precision	recall	f1-score	support
False	0.99	0.72	0.83	96426
True	0.09	0.75	0.16	3602
accuracy			0.72	100028
macro avg	0.54	0.73	0.49	100028
weighted avg	0.95	0.72	0.81	100028

Conclusion.

- The logistic regression model is the best model for predicting delinquent loans with a recall score of 0.76.
- In this case, recall is more important because Freddie Mac wants to avoid purchasing delinquent loans at the incorrect price. By prioritizing recall, the model is better able to identify loans that are actually delinquent and Freddie Mac can correctly charge a premium for them.
- In addition to the best score, the interpretability of logistic regression makes it compliant with the Fair Housing Act. Its transparency allows lenders to identify the features driving the model's predictions and avoid discrimination against any protected groups.
- The model performs close to the benchmark model made by Turiel, J. & Aste, T which had a recall score of 77%.

Thank you!