Loan Approval – Maximum Loan Amount

Cristian Renato Leyton Medina

PART A: Loan Approval Status

Domain Understanding: Classification

Variable Name	Retain or Drop	Brief justification for retention or dropping
ID	DROP	Identifier. Not to be considered.
Sex	DROP	99.6% of the values are null.
Age	RETAIN	It represents (Fernandez-Corrugedo and Muellbauer, 2006, p. 6) an indicator to consider.
Education Qualifications	DROP	Over 99.5% of the data is ‘Unknown’.
Income	RETAIN	Borrower characteristics such as annual income is a relevant variable (Serrano-Cinca, Gutiérrez-Nieto and López-Palacios, 2015).
Home Ownership	RETAIN	The current housing situation is a relevant variable (Serrano-Cinca, Gutiérrez-Nieto and López-Palacios, 2015).
Employment Length	RETAIN	It’s concluded (Castellanos et al., 2025) that job loss is common among new borrowers.
Loan Intent	RETAIN	“Loan purpose is considered as one of the factors explaining the probability of default” (Serrano-Cinca, Gutiérrez-Nieto and López-Palacios, 2015).
Loan Amount	RETAIN	Sifrain (2023) shows that larger loan amounts are positively associated with default risk in P2P lending models.
Loan Interest Rate	PART A DROP / PART B RETAIN	It’s a value determined by the bank, not a variable of a client. Mucci (n.d.) explains that using a variable in the ML model that is not available during the prediction in a real-world scenario is cause for overfitting.
Loan to Income Ratio (LTI)	DROP	It’s an arithmetic operation between Loan Amount and Income. Considering it will lead to multicollinearity. “Multicollinearity occurs when there is a fairly strong linear relationship among a set of explanatory variables.” (Albright and Winston, 2019, p.485).
Payment Default on File	RETAIN	By logic, having defaulted on a payment in the past may prevent banks from giving a new loan to a client.
Credit History Length	RETAIN	The longer the credit history, the higher the confidence in the ‘Payment Default’ history.
Loan Approval Status	TARGET	Shows whether a loan was approved or not. Is the equivalent of the ‘Credit Application Acceptance’ feature
Maximum Loan Amount	TARGET	Shows the maximum amount. Will be dropped in PART A but will be TARGET in PART B.
Credit Application Acceptance	DROP	Equivalent to the ‘Loan Approval Status’ feature.

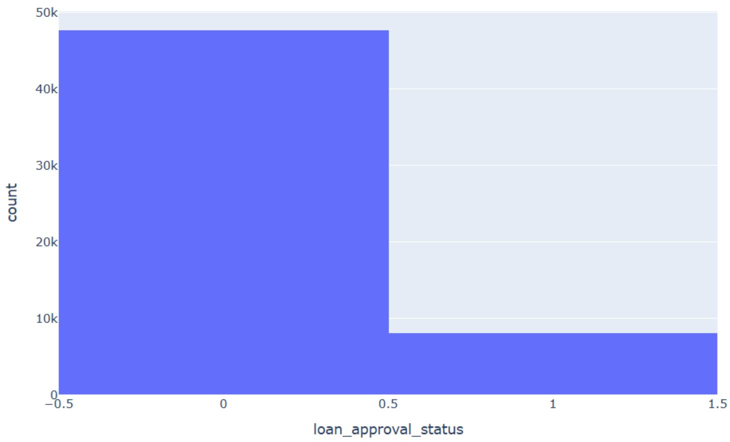
Data Understanding: Producing Your Experimental Design

Data Description | Variable Type

	count	mean	std	min	25%	50%	75%	max
age	55696.0	27.070598	5.235316	20.0	23.0	26.0	29.0	52.0
income	55696.0	60085.816989	24955.929170	4200.0	41000.0	55660.0	75000.0	175500.0
home_ownership	55696.0	1.361247	0.583486	0.0	1.0	1.0	2.0	3.0
emplyment_length	55696.0	4.571441	3.667228	0.0	2.0	4.0	7.0	19.0
loan_intent	55696.0	2.182975	1.657235	0.0	1.0	2.0	4.0	5.0
loan_amount	55696.0	8960.305982	5201.178463	500.0	5000.0	8000.0	12000.0	26000.0
payment_default_on_file	55696.0	0.148592	0.355689	0.0	0.0	0.0	0.0	1.0
credit_history_length	55696.0	5.523377	3.588046	2.0	3.0	4.0	8.0	17.0
loan_approval_status	55696.0	0.855088	0.352015	0.0	1.0	1.0	1.0	1.0

```
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   age                      55696 non-null  float64
 1   income                   55696 non-null  int64
 2   home_ownership           55696 non-null  int64
 3   emplyment_length         55696 non-null  int64
 4   loan_intent              55696 non-null  int64
 5   loan_amount              55696 non-null  int64
 6   payment_default_on_file  55696 non-null  float64
 7   credit_history_length    55696 non-null  int64
 8   loan_approval_status     55696 non-null  float64
dtypes: float64(3), int64(6)
```

Loan Approval Status (0: approved; 1: not approved)



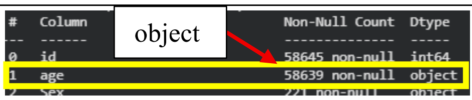
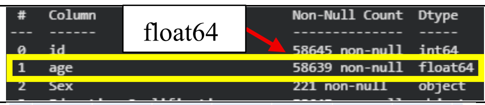
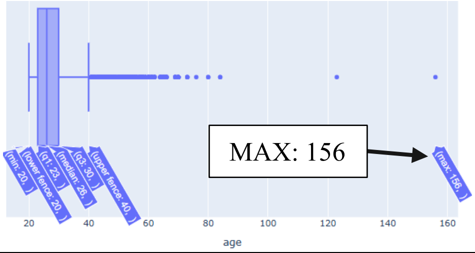
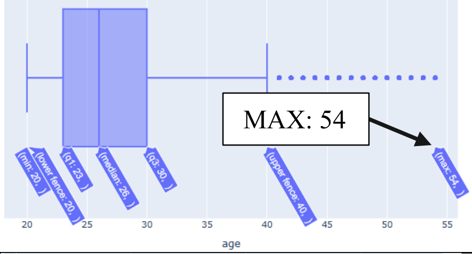
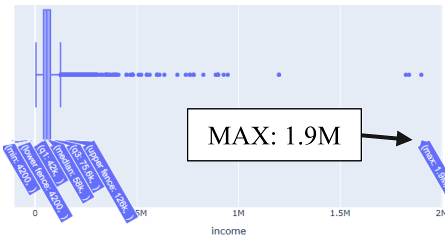
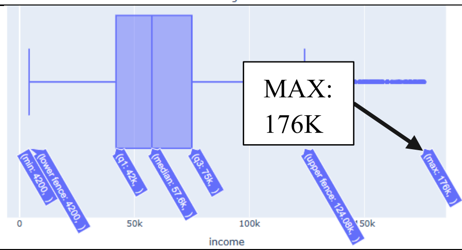
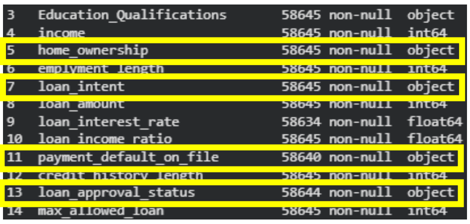
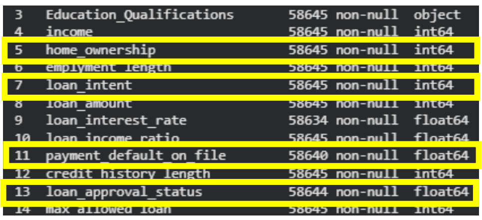
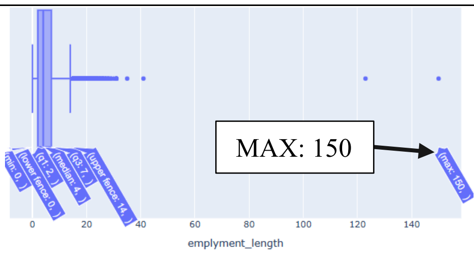
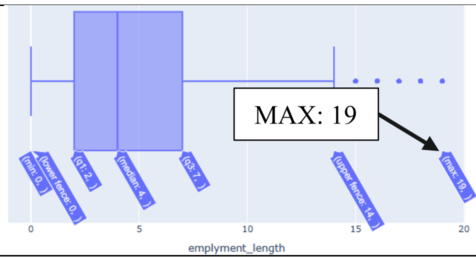
Data Preparation: Cleaning and Transforming your data

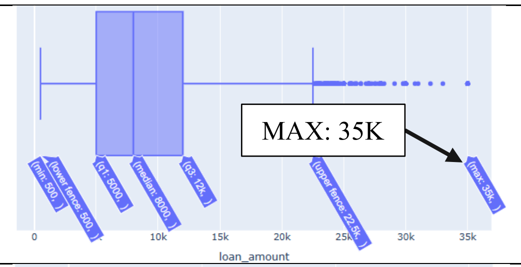
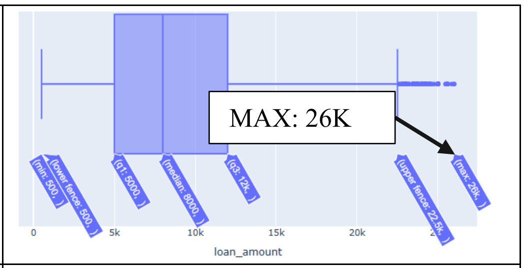
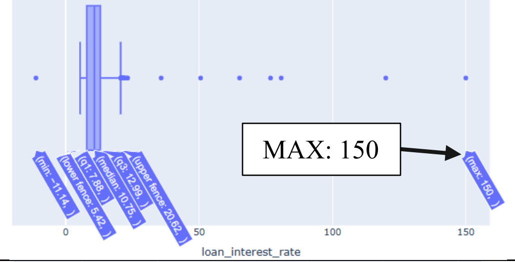
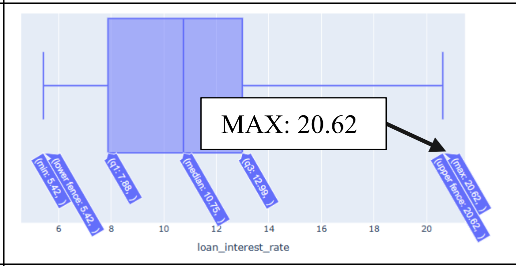
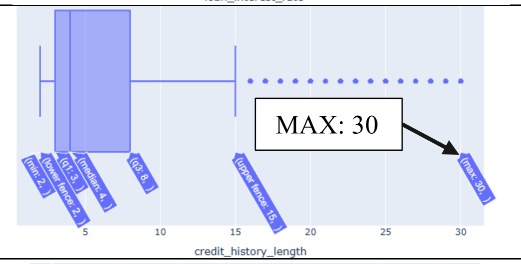
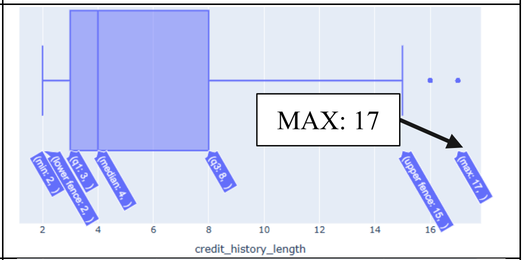
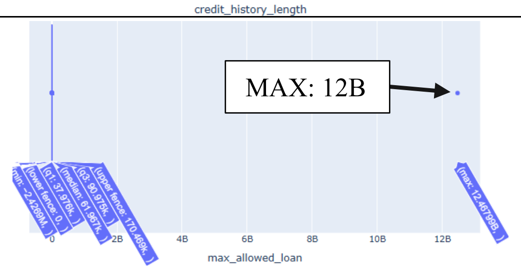
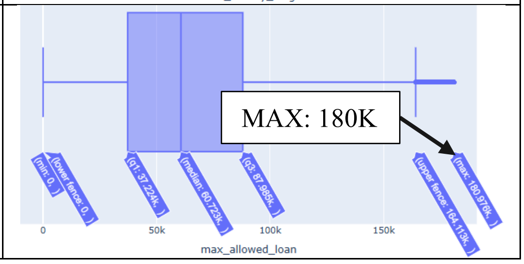
a) Issues with the RETAINED features

Variable Name	Issue Description	Proposed Mitigation	Justification for used mitigation
Age	Contained “object” type values; Outliers	Use “replace” function to reassign values. Drop outliers.	Volume of data to replace was manageable.
Income	Outliers	Drop outliers	There’s more data available.
Home Ownership	Data type was “object”	Use “map” function to reassign values.	Categorical variables need to be mapped into numeric values.
Employment Length	Outliers	Drop outliers	There’s more data available.
Loan Intent	Data type was “object”	Use “map” function to reassign values.	Categorical variables need to be mapped into numeric values.
Loan Amount	Outliers	Drop outliers	There’s more data available.
Loan Interest Rate	Outliers	Drop outliers	There’s more data available.

Payment Default on File	Data type was "object"	Use "map" function to reassign values.	Categorical variables need to be mapped into numeric values.
Credit History Length	Outliers	Drop outliers	There's more data available.
Loan Approval Status	Data type was "object"	Use "map" function to reassign values.	Categorical variables need to be mapped into numeric values.
Maximum Loan Amount	Outliers	Drop outliers	There's more data available.

b) Implementation of mitigations

Variable Name	Before Mitigation	After Mitigation
Age		
Age		
Income		
- Home Ownership - Loan Intent - Payment Default on File - Loan Approval Status		
Employment Length		

<table>
<tr><td>Loan Amount</td><td>
</td><td>
</td></tr>
<tr><td>Loan Interest Rate</td><td>
</td><td>
</td></tr>
<tr><td>Credit History Length</td><td>
</td><td>
</td></tr>
<tr><td>Maximum Loan Amount</td><td>
</td><td>
</td></tr>
</table>

Modelling: Creating Predictive Classification Models

a) Algorithm Overview

Algorithm Name	Algorithm Type	Learnable Parameter	Possible Hyperparameters	Python Package to use the Algorithm
Logistic Regression	Parametric	intercept, slope	penalty, tolerance, fit_intercept, intercept_scaling, class_weight	PyCaret Scikit-learn
Random Forest	Nonparametric	parameters estimated during training	n_estimators, criterion, max_features, max_depth, min_samples_split, min_samples_leaf, max_leaf_nodes	PyCaret Scikit-learn
Naïve Bayes	Both Parametric and nonparametric. In this case we use its **Parametric** form.	conditional probability	Smoothing parameter	PyCaret Scikit-learn

b) Train-Test split

i.

```
['age',
 'income',
 'home_ownership',
 'emplyment_length',
 'loan_intent',
 'loan_amount',
 'payment_default_on_file',
 'credit_history_length',
 'loan_approval_status']
```

Explanatory Variables

Target Variable

Shape

```
(55696, 9)
```

ii. Pradhan and Kumar (2019) when splitting the dataset for a similar model (credit classification) suggest “split the dataset into 70:30 (or 80:20) ratio”. Geron (2023, p.31) mentions that “is common to use 80% of the data for training and hold out 20% for testing”. Therefore, considering both approaches, we will split the data in an 80:20 ratio.

iii. We use a Training – Test approach because our data volume allows us to do so. K-fold CV is used in cases where data is scarce, therefore making it necessary to iterate between multiple splits.

iv. Evidence of reproducibility

```
loan = setup(data,target = 'loan_approval_status',
            max_encoding_ohe = 100,
           #fold_strategy = 'kfold',
           #fold = 5,
            data_split_stratify = True ,
            transformation = False,
            train_size = 0.8,
            numeric_features= ['age', 'income', 'emplyment_length', 'loan_amount', 'credit_history_length'],
            categorical_features = ['home_ownership', 'loan_intent', 'payment_default_on_file'],
            normalize = True,
            normalize_method ='minmax',
            session_id = 87)
```

Stratify = True

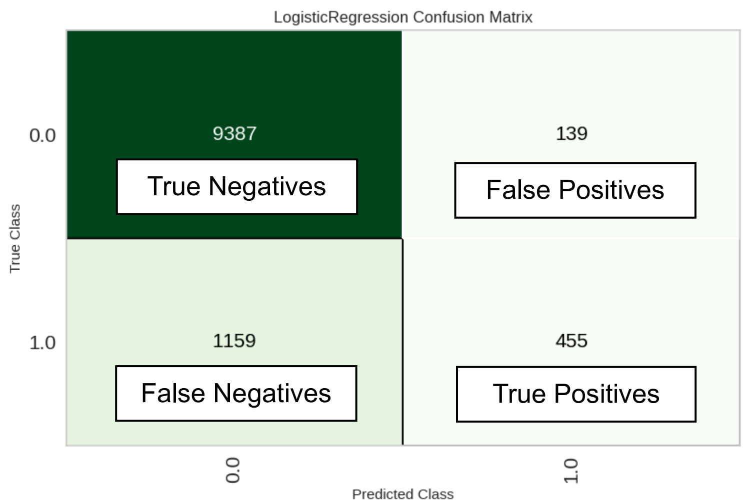
Session ID = 87

Evaluation: How Good are the models

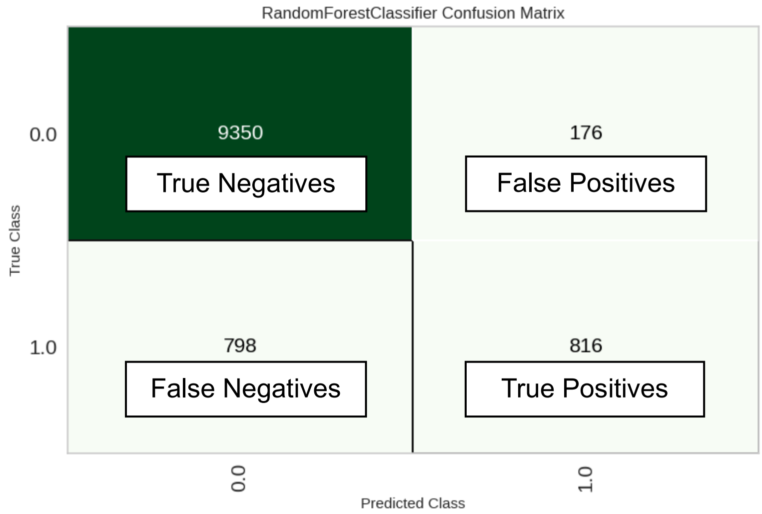
According to the financial analysts, the model should aim to predict rejects correctly to lower the risk of defaulted payments. Therefore, the model should aim to **increase the number of true positives**, and in consequence, **lower the number of false negatives** and **false positives**.

a) Confusion Matrix

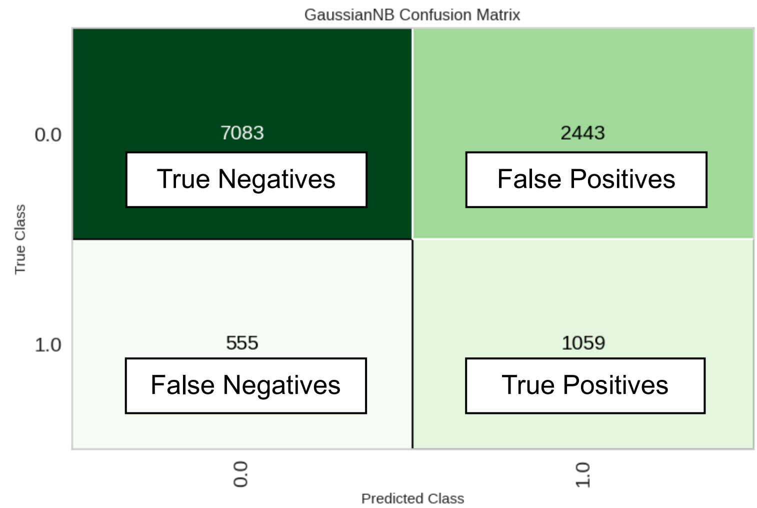
- Logistic Regression



- Random Forest



- Naïve Bayes



b) Test Performance Results

Metrics	USE or DO NOT USE	Justification in relation to the success criteria	Model Name	Test Score
Accuracy	DO NOT USE	Considers True Negatives, which we are not interested in.	LR	0.8835
			RF	0.9126
			NB	0.7309
Recall	USE	We aim to detect the highest number of true positives.	LR	0.2819
			RF	0.5056
			NB	0.6561
Precision	USE	We don't want to have many false positives as this will affect the business.	LR	0.766
			RF	0.8226
			NB	0.3024
F-Score	USE	Balance between Recall and Precision.	LR	0.4121
			RF	0.6262
			NB	0.414
AUC-ROC	DO NOT USE	Values fluctuate around 80% (the model is better than random).	LR	0.8348
			RF	0.8758
			NB	0.7869

c) Suggested model based on USED metrics

Random Forest is the suggested model, as it has the highest Precision and F-Score. While NB predicts a higher quantity of True Positives, it also detects a huge amount of False Positives which costs money to the business. **Random Forest** has a more balanced ratio between Recall and Precision.

d) Establishing if the model is or not a good fit.

Since we didn't save a validation split, our approach to test generalisation is to retrain our RF this time using K-folds. The following figure shows how through different iterations, our metrics don't vary by much, therefore we can affirm **RF is a good fit**.

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.9105	0.8789	0.4954	0.8142	0.6160	0.5687	0.5912
1	0.9119	0.8801	0.4926	0.8303	0.6184	0.5722	0.5972
2	0.9097	0.8758	0.4833	0.8189	0.6079	0.5606	0.5856
3	0.9089	0.8766	0.4926	0.8020	0.6104	0.5621	0.5835
4	0.9135	0.8877	0.4954	0.8432	0.6241	0.5789	0.6051
Mean	0.9109	0.8798	0.4919	0.8217	0.6153	0.5685	0.5925
Std	0.0016	0.0042	0.0044	0.0141	0.0058	0.0067	0.0079

e) **Random Forest** Tuning

i. 5 Folds used. (These parameters were chosen after multiple iterations with different values).

```
param_grid = {
    'n_estimators': [200, 600],
    'max_depth': [None],
    'min_samples_split': [5],
    'min_samples_leaf': [2],
    'max_features': ['sqrt']
}

tuned_rf = tune_model(
    rf,
    search_library='scikit-learn',
    search_algorithm='grid',
    custom_grid=param_grid,
    fold=5,
    optimize='Recall',
    choose_better = False
)
```

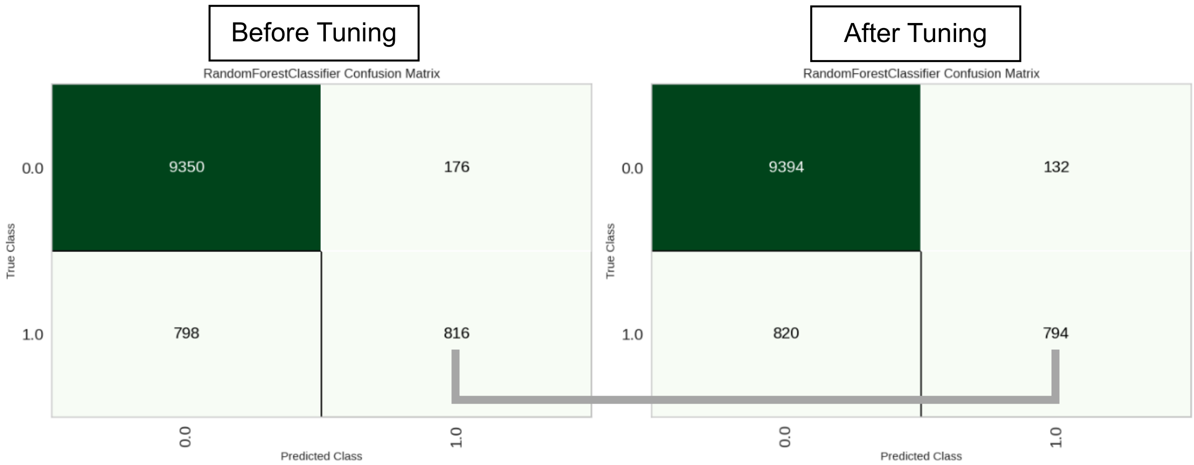
GridSearchCV

K-Folds

ii. Difference in Hyper Parameters

Hyperparameters	Original	Tuned
min_samples_leaf	1	2
min_samples_split	2	5
n_estimators	100	200, 600

iii. Confusion Matrix – Before vs After Tuning



iv. New scores of performance metrics after tuning.

Performance Metrics	Before Tuning	After Tuning
Recall	0.5056	0.484
Precision	0.8226	0.849
F-Score	0.6262	0.617

The Tuned Model aimed to increase Recall as it aimed to predict as many True Positives as possible. After trying with different hyperparameters, Recall decreased. However, with this configuration of hyperparameters, while Recall decreased, Precision improved. This means the model is predicting fewer False Positives while sacrificing a few True Positives.

v. Analysing if the tuned model performs better.

For this, we will simulate a business scenario (as done in Seminar 5):

Let's speak in simple terms and say for instance that every True Positive prevents the business from losing revenue; every False Positive prevents the business from earning profit. As well, True Negatives mean profit and False Negatives means loss of revenue.

Let's assign arbitrary values of +1 for True Positives and True Negatives as they affect profit positively and -3 for False Positives and False Negatives as they affect profit negatively.

- Before Tuning:

Random Forest	Customers	Value	Total
True Positives + True Negatives	10,166	+1	10,166
False Positives + False Negatives	974	-3	-2,922
			7,244

- After Tuning

Random Forest	Customers	Value	Total
True Positives + True Negatives	10,188	+1	10,188
False Positives + False Negatives	952	-3	-2856
			7,332

Speaking business, **the second model** although reducing True Positives, increases profit. Therefore, we can say it **serves the purpose better**.

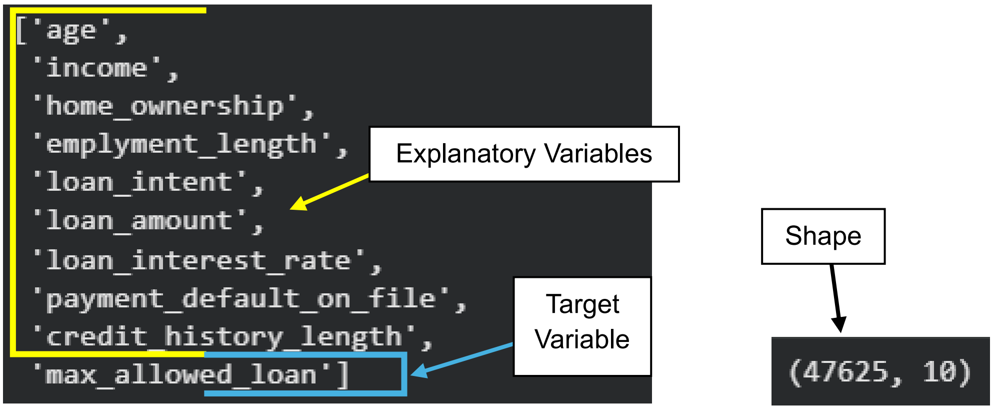
Notice that we could assign a value of -10 for negative profits and even then, the second scenario would be better, although this doesn't mean any of the models is a potential good candidate as we'll discuss in detail in the next section.

f) The research question, critique and ethics.

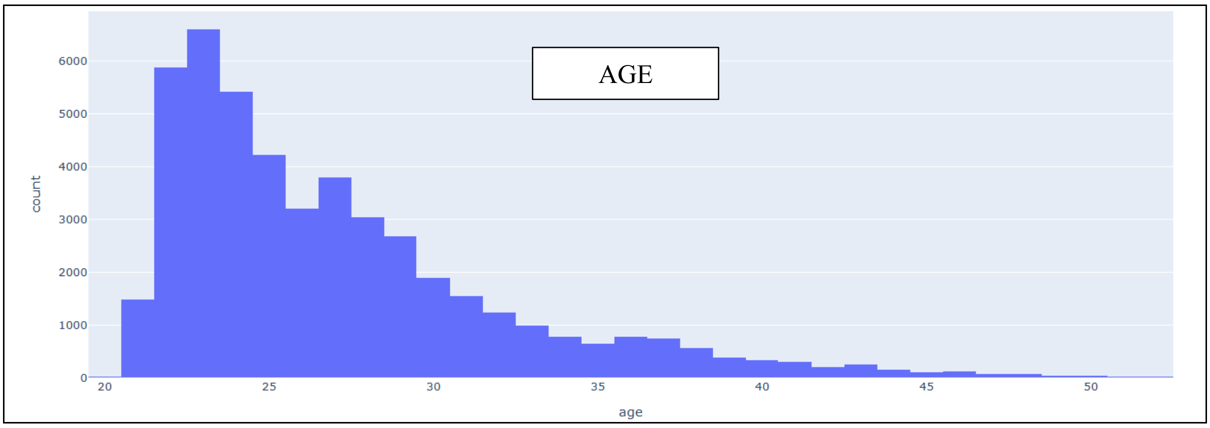
The best model (Tuned RF) DOES NOT have the potential to perform in a real-case scenario due to having misclassified 812 rejected clients as approved. From my professional perspective as a former Collections Coordinator, follow-up on defaulters demands a great number of resources and doesn't guarantee recovery of the defaulted amounts. The reason behind RF outperforming the other candidates is that it considers relationships between variables that the other models don't. Speaking of ethics, approving a loan for an individual who isn't in the condition to repay it will put them in serious financial difficulties.

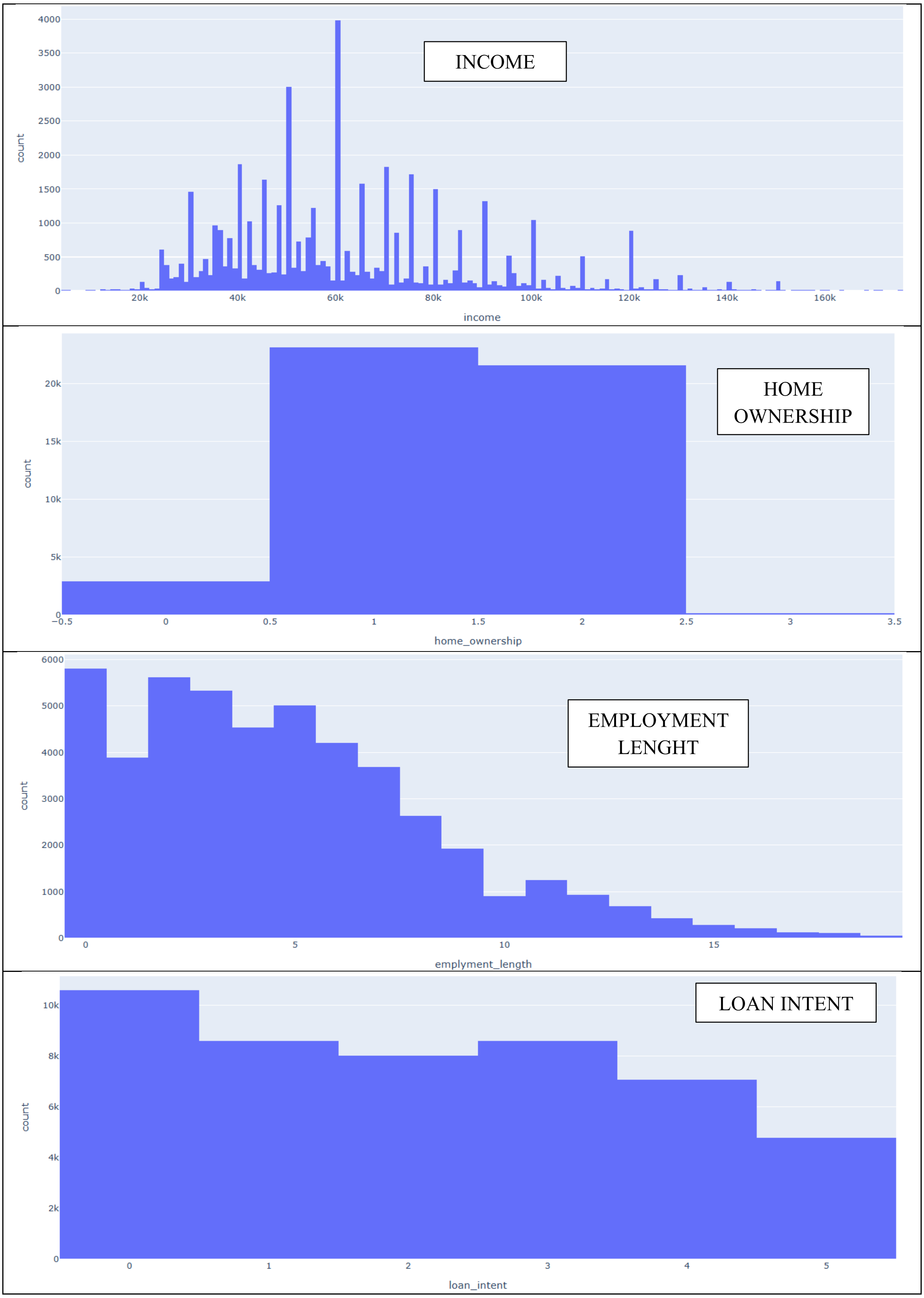
Part B: Maximum Loan Amount Prediction

Domain Understanding: Regression

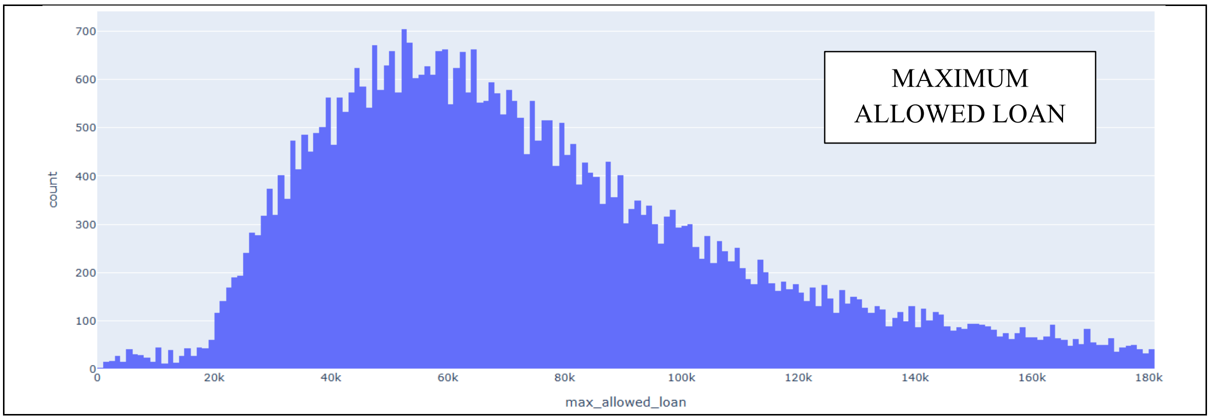


Data Understanding : Experimental Design




INCOME
count
income
HOME OWNERSHIP
count
home_ownership
EMPLOYMENT LENGHT
count
emplyment_length
LOAN INTENT
count
loan_intent


LOAN AMOUNT
count
0
1000
2000
3000
4000
5000
6000
5k
10k
15k
20k
25k
loan_amount
INTEREST RATE
count
0
500
1000
1500
2000
6
8
10
12
14
16
18
20
loan_interest_rate
PAYMENT DEFAULT ON FILE
count
0
5k
10k
15k
20k
25k
30k
35k
40k
−0.5
0
0.5
1
1.5
payment_default_on_file
CREDIT HISTORY LENGTH
count
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
2
4
6
8
10
12
14
16
credit_history_length



Data Preprocessing

a) At first sight, we may think we need to scale the values as there's a big difference in magnitudes as we see in the figure:

	count	mean	std	min	25%	50%	75%	max
age	47625.00	27.06	5.20	20.00	23.00	26.00	29.00	52.00
income	47625.00	62232.34	24987.38	4200.00	44000.00	60000.00	75000.00	175500.00
home_ownership	47625.00	1.39	0.60	0.00	1.00	1.00	2.00	3.00
emplyment_length	47625.00	4.73	3.69	0.00	2.00	4.00	7.00	19.00
loan_intent	47625.00	2.15	1.65	0.00	1.00	2.00	3.00	5.00
loan_amount	47625.00	8623.34	4961.95	500.00	5000.00	7875.00	11800.00	26000.00
loan_interest_rate	47625.00	10.25	2.82	5.42	7.51	10.39	12.42	20.11
payment_default_on_file	47625.00	0.12	0.33	0.00	0.00	0.00	0.00	1.00
credit_history_length	47625.00	5.52	3.57	2.00	3.00	4.00	8.00	17.00
max_allowed_loan	47625.00	74160.27	35037.68	232.00	48135.00	67473.00	93972.00	180976.00

However, as we will see in the next point, Decision Trees suppose an exception to the rule.

b) Scaling variables

According to Geron (2019, p.69) "scaling the target values is generally not required" when modelling Decision Trees. As well, Muller and Guido (2017, p.83) mention one of the advantages of Decision Trees is that "no preprocessing like normalization or standardization of features is needed". Therefore, we will not scale any variables.

In general terms, most ML algorithms are sensitive to magnitudes, therefore is recommended to scale the numeric variables.

Modelling: Build Predictive Regression Models

a) The main benefits of Decision Trees are that they are easy to visualize and understand by non-experts, and they don't require preprocessing like variable scaling (Muller and Guido, 2017, p.83).
b) Creating Model 1 and Model 2

i. Reproducibility

```python
X = dataset.drop(['max_allowed_loan'], axis=1)
y = dataset['max_allowed_loan']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=87)
```

Random State = 87

ii. Dimensions of Model 1 and Model 2

```
Model 1 X Train: (38100, 6)
Model 1 y Train: (38100,)
Model 1 X Test: (9525, 6)
Model 1 y Test: (9525,)
```

```
Model 2 X Train: (38100, 9)
Model 2 y Train: (38100,)
Model 2 X Test: (9525, 9)
Model 2 y Test: (9525,)
```

```
['age',
 'income',
 'emplyment_length',
 'loan_amount',
 'loan_interest_rate',
 'credit_history_length',
 'max_allowed_loan']
```

```
['age',
 'income',
 'home_ownership',
 'emplyment_length',
 'loan_intent',
 'loan_amount',
 'loan_interest_rate',
 'payment_default_on_file',
 'credit_history_length',
 'max_allowed_loan']
```

Evaluation: How Good are the models

According to the financial analysts, while a degree of error is expected in estimating the maximum loan amount, the selected model should contain input features that better explain the target variable values. Therefore, our defining metric is R-Square.

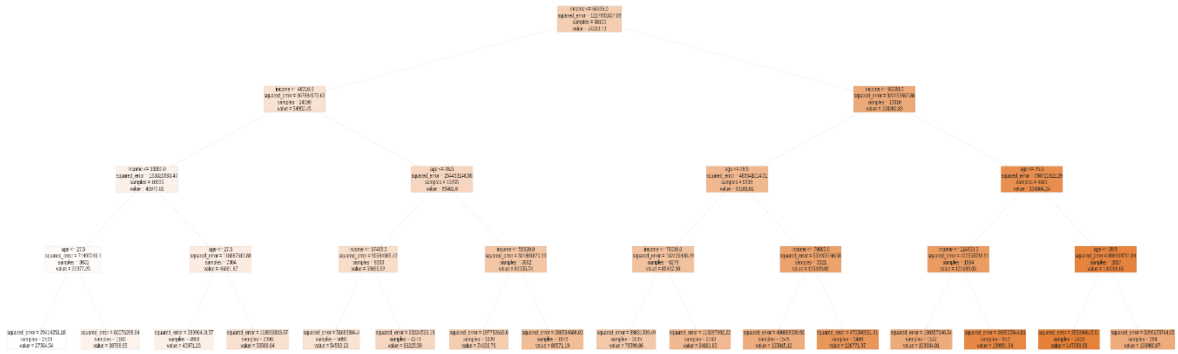
a) Metrics

Metrics	USE or DO NOT USE	Justification in relation to the success criteria	Model Name	Test Score
MSE	USE	Big differences from the actual value are highly penalised, therefore it's useful as it allows us to determine there may be cases where customers are getting small loan amounts, which represents a loss in revenue. It also helps us notice there are customers getting amounts higher than they should be allowed, risking profit as well as risking the client's financial status.	DT1	2592956.09
			DT2	2719232.72
MAE	USE	Easier to interpret compared to MSE. Measures the average size of errors.	DT1	703.96
			DT2	734.37
R-Square	USE	Shows how well our model explains the target variable. Part of the research question.	DT1	0.9978
			DT2	0.9977

b) When preparing data, we purposely left some outliers in the dataset to avoid dropping many observations. MSE is very likely being affected by said outliers but at the same time, acknowledges us of the existence of predictions that vary greatly from the actual values.

c) **DT1** (only numeric values) is our choice as it's performed better in every metric. From R-Square we can conclude that it explains almost perfectly how our chosen variables affect the target. MAE is relatively low, which indicates that our model's predictions are in average, very close to the actual values. MSE aware us of cases with big differences between predicted and actual values.

d) Pruned DT1



Metrics	Model Name	Test Score
MSE	DT1	2553662.27
	DT1 Pruned	160487165.28
MAE	DT1	695.17
	DT1 Pruned	8330.75
R-Square	DT1	0.9979
	DT1 Pruned	0.8691

Pruning our DT worsened its performance by much, as shown in the table above. The difference in the MAE is very big speaking of loans: To loan a person 700£ less or more than they should receive, depending on the requested amount, is not a very big problem. On the other hand, allowing a customer to receive 8,300£ more than they should be allowed, represents a risk for both parties.

e) Predicting on a new customer

Attached evidence of the prediction.

Features not included in the model were commented in the code so that the model runs. As well, Age was input as 56 instead of "56 Years old".

Please see next page.

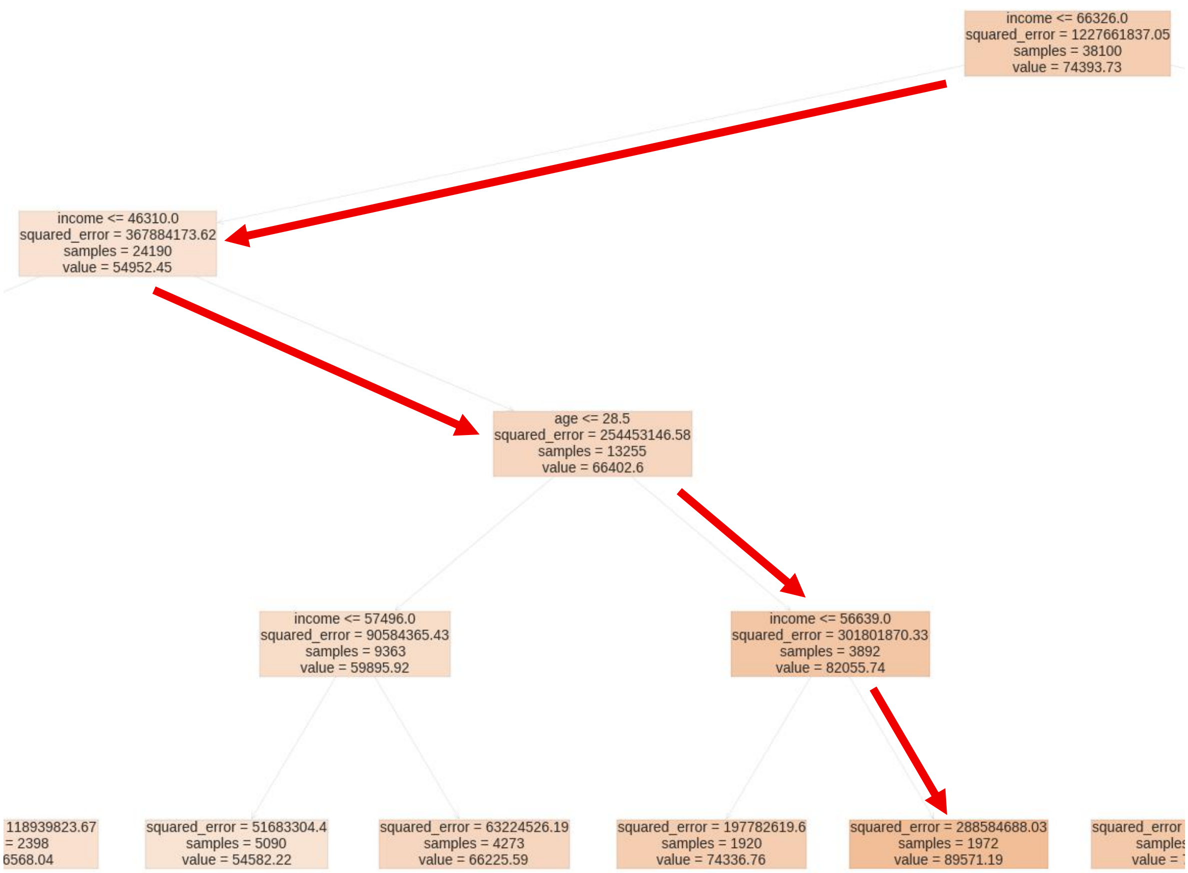
```python
# Create a new DataFrame from scratch to predict Max Loan Amount
new_customer = []
new_customer.append( {
            #"id":60256,
             "age":56,
            #"Sex":"F",
            #"Education_Qualifications":"Unknown",
             "income":57000,
            #"home_ownership":"rent",
             "emplyment_length":15,
            #"loan_intent":1,
             "loan_amount":25700,
             "loan_interest_rate":23,
            #"loan_income_ratio":10,
            #"payment_default_on_file":"No",
             "credit_history_length":35,
            #"loan_approval_status":"Approved",
            #"max_allowed_loan":value,
            #"Credit_Application_Acceptance":0
            } )
customer_to_predict = pd.DataFrame(new_customer)

# Add a new column to customer_to_predict with the predicted prices:
customer_to_predict["max_allowed_loan"] = pruned_regressor.predict(customer_to_predict)
customer_to_predict.head()
```

Max Allowed Loan

	age	income	emplyment_length	loan_amount	loan_interest_rate	credit_history_length	max_allowed_loan
0	56	57000	15	25700	23	35	89571.188641

Path followed in the tree (image cropped to improve readability):



REFERENCES

Albright, S.C. and Winston, W.L., 2019. *Business Analytics: Data Analysis and Decision Making*. 7th ed. Boston, MA: Cengage Learning.

Castellanos, S.G., Jiménez-Hernández, D., Mahajan, A., Alcaraz Prous, E. and Seira, E., 2025. *Contract Terms, Employment Shocks, and Default in Credit Cards*. Review of Economic Studies, p.rdaf079.

Fernandez-Corugedo, E. and Muellbauer, J., 2006. *Consumer credit conditions in the United Kingdom*. London: Centre for Banking Studies, Bank of England, and Nuffield College.

Geron, A., 2019. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 3rd ed. Sebastopol, CA: O'Reilly Media.

Mucci, T. (n.d.) *What is data leakage in machine learning? IBM Think*. Available at: https://www.ibm.com/think/topics/data-leakage-machine-learning. (Accessed: 31 October 2025).

Müller, A.C. and Guido, S., 2017. *Introduction to Machine Learning with Python: A guide for data scientists*. Sebastopol, CA: O'Reilly Media.

Pradhan, M. and Kumar, U.D., 2019. *Machine Learning Using Python*. Singapore: Springer.

Serrano-Cinca, C., Gutiérrez-Nieto, B. and López-Palacios, L. (2015) *'Determinants of default in P2P lending*', PLOS ONE, 10(10), e0139427. Available at: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0139427. (Accessed: 31 October 2025).

Sifrain, R., 2023. *Predictive Analysis of Default Risk in Peer-to-Peer Lending Platforms: Empirical Evidence from LendingClub*. Journal of Financial Risk Management, 12(1), pp.28–49. Available at: https://www.scirp.org/journal/paperinformation?paperid=123509 (Accessed 12 November 2025).