Analyzing and Predicting Loan Approvals Using Statistical and Machine Learning Techniques

Hermann Diderot Ndamen Noungang
Middle East Technical University
Ankara, Türkiye

Abstract – This report conducts analysis and classification of loan requests, whether they will be rejected or approved. Exploratory Data Analysis is conducted to understand what are the variables that most likely determine the loan approval status. Logistic regression, Support Vector Machine, Artificial Neural Network, Random Forest and XGBoost are the supervised machine learning methods used in this analysis for classification. Accuracy, Precision, Recall, and F1 score are used to compare model performances. All the modelling except for Logistic regression are done with Python.

I. INTRODUCTION

A loan is something that is borrowed, especially a sum of money that is expected to be paid back with interest. In today's society, loans have almost become the heart of all activities, especially loans granted by financial institutions. It is therefore crucial to analyze such datasets to understand whether a loan will be classified as approved or denied, based on a given set of variables.

The classification of the loan status is done with Support Vector Machine, Artificial Neural Network, Random Forest and XGBoost, with Accuracy, Precision, Recall, and F1score values to compare their performances in the classification.

II. LITERATURE REVIEW

There are several other analyses which were made on this data, and most of them had the same research interest, which was to determine the loan status in various ways. In this report, we focus on using statistical approaches and modeling to reach this exact same goal.

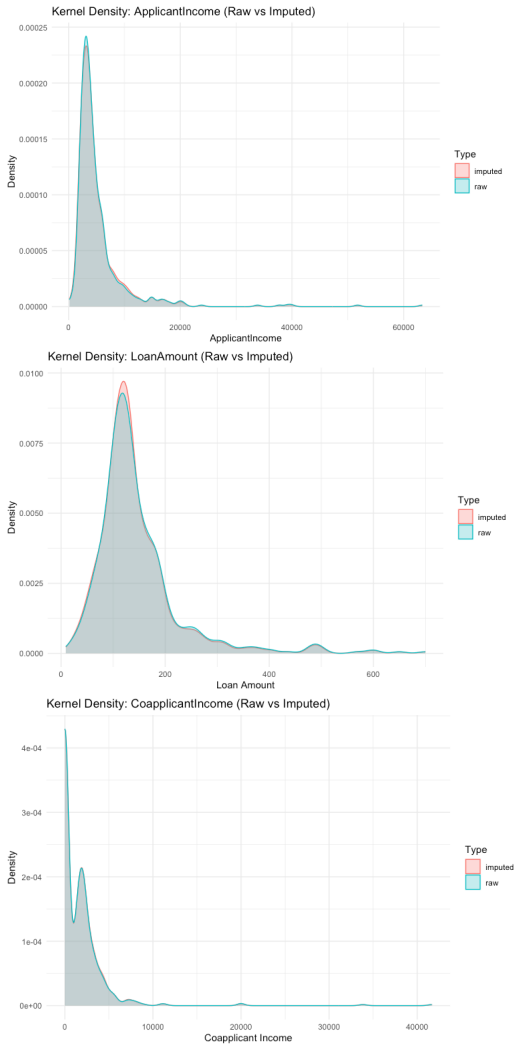
III. METHODOLOGY

A. *Dataset*

The dataset used in this analysis is from Kaggle, published by user altruistdelhite04, named: "Loan Predication Problem Dataset." The dependent variable is Loan_Status which indicates the loan approval status as "Y" if approved and "N" if not approved. The independent variables are: Loan_ID, Gender, Married, Dependents, Education, Self_Employed, ApplicantIncome, CoapplicantIncome, LoanAmount, Loan_Amount_Term, Credit_History, Property_Area, and Loan_Status.

B. *Data Imputation*

Our dataset has about 12% of missing values, and for imputation we used the Predictive Mean Method (pmm) in the Multiple Imputation by Chained Equations (MICE) package. After imputation, we obtain the following kernel densities and data summary:



As we can see, the data imputation was successful, since the Kernel density functions do not differ a lot from each other.

C. *Descriptive Statistics*

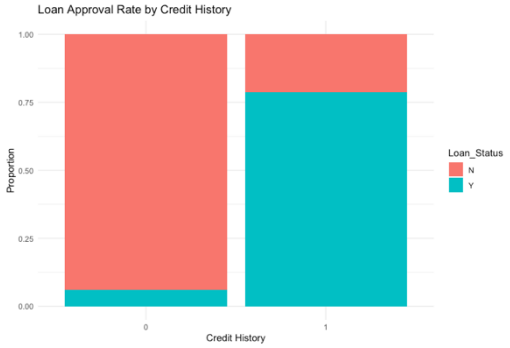
```
> summary(completed_data)
    Gender    Married   Dependents        Education   Self_Employed ApplicantIncome
 Female:118   No :211   0 :363     Graduate    :481   No :533       Min.   :  150
 Male  :496   Yes:403   1 :100     Not Graduate:133   Yes: 81       1st Qu.: 2808
                        2 : 95                                      Median : 3781
                        3+: 56                                      Mean   : 5254
                                                                    3rd Qu.: 5814
                                                                    Max.   :63337
>

CoapplicantIncome   LoanAmount     Loan_Amount_Term Credit_History   Property_Area Loan_Status
Min.   :    0     Min.   :  9.0   Min.   : 12.0    0: 84           Rural    :177   N:192
1st Qu.:    0     1st Qu.:100.0   1st Qu.:360.0    1:530           Semiurban:232   Y:422
Median : 1131     Median :127.0   Median :360.0                    Urban    :205
Mean   : 1620     Mean   :144.4   Mean   :340.1
3rd Qu.: 2297     3rd Qu.:164.8   3rd Qu.:360.0
Max.   :41667     Max.   :700.0   Max.   :480.0
```

Regarding the descriptive statistics, we can observe that Male applicants are dominant in our data, and most applicants are married. Most applicants report to have no dependents, and the number of applicants with 3 or more dependents is relatively low (56 of them). Additionally, graduates and self-employed people are more present in our data, most applicants seem to have a credit history, and most applicants come from the Semiurban area. Applicant's income ranges from 150 to 63,337 and the mean is 1620 which indicates a right skewed distribution, and the possible existence of outliers. Co-applicant's income seems to have a similar distribution, ranging from 0 to 41,667. The loan amount ranges from 9 to 700, with a relatively low mean and median, again suggesting a right skewed distribution. As for the loan term, the common loan term is about 360 months, equivalent to 10 years, therefore most probably house loans.

D. *Exploratory Data Analysis (EDA)*

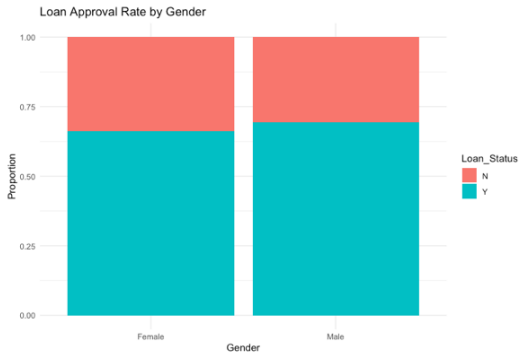
5 research questions were prepared for the EDA stage.

Question 1: Does an applicant's credit history significantly affect loan approval?



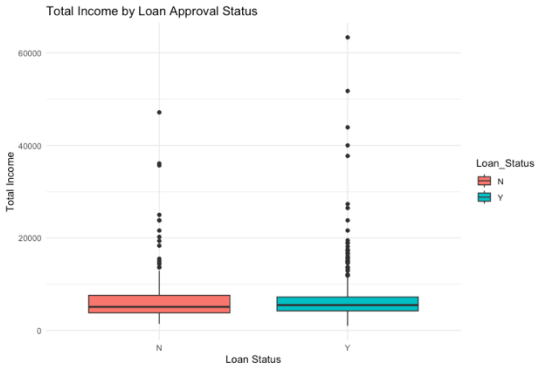
As we can observe on the plot, more loans are being approved when the applicant already has a credit history. This will be tested during the CDA.

Question 2: Is there a difference in loan approval rates based on gender?



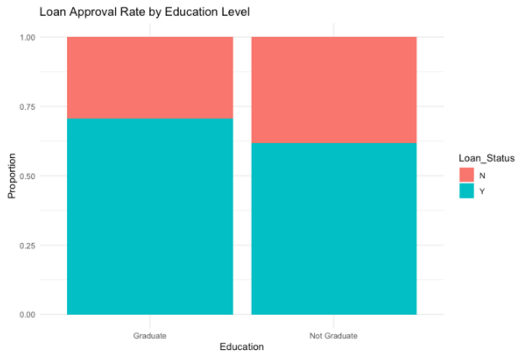
As observed on the graph showing the loan rate by gender, there doesn't seem to be a significant difference in loan approvals between males and females. We shall test this claim during CDA.

Question 3: How does Total Income relate to Loan Approval?



The median looks very similar in both the approved and not approved groups, and their distributions look similar. Therefore, total income may not have a great effect on loan approval.

Question 4: Are graduates more likely to get their loan approved than non-graduates?



It seems like the graduates and non-graduates have different approval rates.

Question 5: Does property location affect loan approval?

Semiurban areas seem to have more loan approvals compared to Rural and Urban areas who have similar approval rates. This might be the reason why majority of loan requests come from the Semiurban areas.

E. *Confirmatory Data Analysis (EDA)*

We will now evaluate our findings from the EDA.

Question 1: Does an applicant's credit history significantly affect loan approval?

H0: There is no association between credit history and loan approval status
H1: There is an association between them

Since we are testing the association between categorical variables, we will use Chi-sq test. We run the test and obtain a **p-value of 2.2e-16** which is less than 0.05, therefore **the test is significant**. Which means that there is an association between credit history and loan approval status.

Question 2: Is there a difference in loan approval rates based on gender?

H0: There is no association between gender and loan approval
H1: There is an association between them

Again, we are going to use Chi-sq test. We run the test and obtain a p-value of 0.6947, so we failed to reject the null hypothesis. There is no association between gender and loan approval as we suspected.

Question 3: How does Total Income relate to Loan Approval?

#H0: The mean total income is the same for approved and rejected applicants
#H1: It differs

Since we needed to use a T-test to test this, we first conducted a normality test, but it failed. Therefore, we proceeded with a non-parametric test, that is, Wilcoxon Rank Sum Test and obtained a **p-value of 0.567**, therefore we failed to reject the null hypothesis. As we suspected, there is no statistically significant difference in total income between approved and rejected loan applicants.

Question 4: Does Education Level Affect Loan Approval?

#H0: Education level and loan approval status are independent
#H1: They are not

We used Chi-sq test, and after making sure all the assumptions were satisfied, we ran the test and out **p value was 0.0597** which is slightly greater than 0.05 so we fail to reject the null hypothesis.

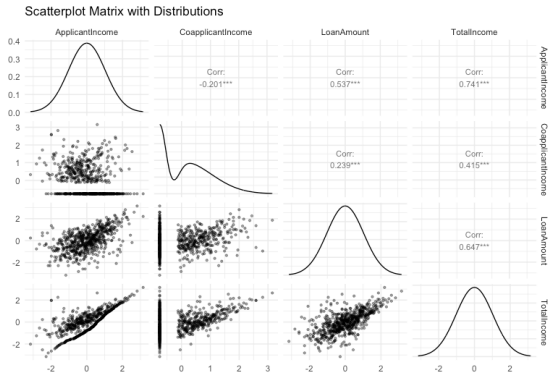
Question 5: Does property location affect loan approval?

#H0: Loan approval is independent of property area
#H1: It is not

Chi-sq test was used to test the hypothesis, and a **p-value of 0.01297** was obtained, hence we rejected the null hypothesis.

F. *Correlation Analysis*

In the correlation analysis, in the first graph, we notice that there is a strong positive correlation between applicant's income and total income.



G. *Train-Test Split and Cross Validation*

We used the stratified random sampling method based on the dependent variable to ensure class balance on the train and test data. Doing this, we obtained a perfectly balanced split, and here is the structure of our training set.

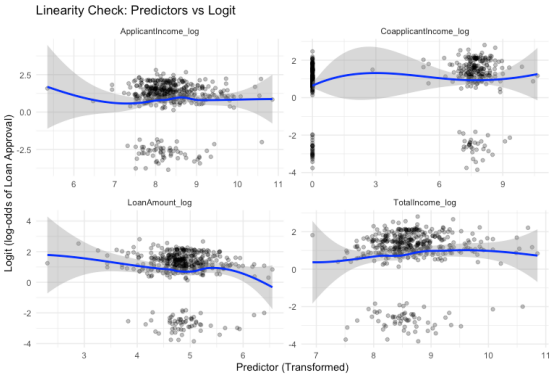
```
> str(train_data)
'data.frame':   431 obs. of  17 variables:
 $ Gender              : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
 $ Married             : Factor w/ 2 levels "No","Yes": 1 2 1 2 2 2 2 1 2 1 ...
 $ Dependents          : Factor w/ 4 levels "0","1","2","3+": 1 3 1 4 3 2 3 1 3 1 ...
 $ Education           : Factor w/ 2 levels "Graduate","Not Graduate": 1 1 2 1 1 1 1 1 1 1 ...
 $ Self_Employed       : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 1 ...
 $ ApplicantIncome     : num  0.743 0.57 -1.098 -0.509 0.1 ...
 $ CoapplicantIncome   : num  -0.756 1.395 0.137 -0.756 0.141 ...
 $ LoanAmount          : num  0.376 1.546 -0.855 0.561 0.708 ...
 $ Loan_Amount_Term    : int  360 360 360 360 360 360 360 360 120 360 ...
 $ Credit_History      : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
 $ Property_Area       : Factor w/ 3 levels "Rural","Semiurban",..: 3 3 3 3 2 3 2 3 1 3 3 ...
 $ TotalIncome         : num  0.2411 1.0336 -0.7481 -1.2994 0.0674 ...
 $ ApplicantIncome_log : num  8.7 8.6 7.76 8.02 8.3 ...
 $ CoapplicantIncome_log: num  0 8.34 7.32 0 7.33 ...
 $ LoanAmount_log      : num  4.96 5.59 4.56 5.07 5.13 ...
 $ TotalIncome_log     : num  8.7 9.17 8.26 8.02 8.62 ...
 $ Loan_Status_bin     : Factor w/ 2 levels "Rejected","Approved": 2 2 2 1 2 1 2 1 2 2 ...
>
```

H. *Modelling*

1. *Logistic Regression*

After splitting, it is now time to check the assumptions for the logistic regression model.

We firstly checked the VIF values which were all less than 10, so there was no sign of multicollinearity. As for the linearity assumption, we can observe on the graph below that both applicant income and
total income are approximately linear. However, co-applicant income and loan amount do not seem to be so. Since we already have total income in the model, which partly constitutes of co-applicant income, we will drop co- applicant income from the model, and to make
loan amount more linear, we will apply a poly transformation to it.



After handling that, we fit the model, and obtain the following model summary. And the main takeaway from it is that applicants with credit history are much likely to get a loan approval. But for now, all the other variables are not statistically significant.

```
> summary(final_model)

Call:
glm(formula = Loan_Status_bin ~ Gender + Married + Dependents +
    Education + Self_Employed + Loan_Amount_Term + Credit_History +
    Property_Area + TotalIncome_log + poly(LoanAmount_log, 2),
    family = "binomial", data = train_data)

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                 -3.549710   3.000404  -1.183   0.2368
GenderMale                   0.018230   0.333697   0.055   0.9564
MarriedYes                   0.422370   0.291060   1.451   0.1467
Dependents1                 -0.393226   0.355532  -1.106   0.2687
Dependents2                  0.085115   0.398359   0.214   0.8308
Dependents3+                -0.035982   0.447676  -0.080   0.9359
EducationNot Graduate       -0.370105   0.313632  -1.180   0.2380
Self_EmployedYes            -0.097667   0.397488  -0.246   0.8059
Loan_Amount_Term            -0.002526   0.002120  -1.191   0.2335
Credit_History1              3.983164   0.544748   7.312 2.63e-13 ***
Property_AreaSemiurban       0.534531   0.318821   1.677   0.0936 .
Property_AreaUrban          -0.241799   0.317010  -0.763   0.4456
TotalIncome_log              0.173610   0.314382   0.552   0.5808
poly(LoanAmount_log, 2)1 -3.112097   3.565796  -0.873   0.3828
poly(LoanAmount_log, 2)2 -2.438740   2.519012  -0.968   0.3330
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 535.87  on 430  degrees of freedom
Residual deviance: 401.59  on 416  degrees of freedom
AIC: 431.59

Number of Fisher Scoring iterations: 5
```

Investigating outliers, we did not find any but found 36 influential points. But since there is no instability in the model, we decided to run Ridge and Lasso regression, and after all our iterations, we decided to go for Lasso Regression because it had a smaller error, while maintaining our main predictors, and here is the model summary.

```
--- Lasso Coefficients ---
> print(coef(cv_lasso, s = "lambda.min"))
15 x 1 sparse Matrix of class "dgCMatrix"
                                s1
(Intercept)           -2.14417516
GenderMale             .
MarriedYes             0.12437381
Dependents1           -0.05232848
Dependents2            .
Dependents3+           .
EducationNot Graduate  .
Self_EmployedYes       .
Loan_Amount_Term       .
Credit_History1        3.20854389
Property_AreaSemiurban 0.28584310
Property_AreaUrban     .
ApplicantIncome_log    .
LoanAmount_log         .
TotalIncome_log        .
```

```
Best lambda for Ridge: 0.02969191
> cat("Ridge CV Error:", min(cv_ridge$cvm), "\n\n")
Ridge CV Error: 1.003283

> cat("Best lambda for Lasso:", cv_lasso$lambda.min, "\n")
Best lambda for Lasso: 0.02194446
> cat("Lasso CV Error:", min(cv_lasso$cvm), "\n\n")
Lasso CV Error: 0.9825804
```

Model Interpretation

As we can see, Lasso has a smaller error, and it keeps the main predictors in our model, which also proved to have a significant effect during the CDA. Therefore, we choose Lasso. In our Lasso Coefficients, Credit History is by far the stongest predictor, followed by the property area, married status, and dependents, which mostly go along with our suspicions at the EDA stage.

2. *Support Vector Machine (SVM)*

Model Type	Accuracy	Class 0 Recall	Class 1 Recall
Basic SVM	0.797	0.44	0.99
Balanced SVM	0.797	0.44	0.99
SMOTE + SVM	0.699	0.49	0.81

For the SVM, we ran 3 versions. A base model with nothing extra added or modified, a version with balanced class weights, and a SMOTE version. Both the basic and balanced versions had the same

results with an accuracy of about 80%, and a recall of 44% for class 0 (loan rejected). The SMOTE model on the other hand is a little more balanced, resulting in an increase in the recall for class 0 from 44 – 49%, at the cost of the accuracy dropping to about 70%. We therefore decide to favor accuracy, and choose the basic SVM model for further comparison.

3. *Random Forests*

Model Type	Accuracy	Class 0 Recall	Class 1 Recall
Base RF	0.78	0.47	0.95
Tuned RF	0.797	0.44	0.99
Tuned RF + balanced	0.772	0.47	0.94
SMOTE + RF	0.756	0.51	0.89

For the random forest, in the base model we had no tuning, and we had a train accuracy of 100%, hence overfitting. Then we had the tuned RF, the tuning was done with GridSearchCV, with a 5-fold CV. The tuned parameters were : n_estimators = 100, max_depth = 10, min_samples_leaf = 2, min_samples_split = 2, and with these we got the best generalizatioin with reduced overfitting. In the Tuned RF + balanced model, we used the same parameters as in the balanced model. This gave us a slightly lower accuracy, but improved class 0 recall to 0.47. Then we have the SMOTE + RF model which gave us the lowest accuracy, but the best recall for class 0. For the random forest, we choose the Tuned RF as the main model as it has the highest overall accuracy. As a balanced alternative, the tuned model with balanced class weights would be better, and if our goal was fairness, then we would go for the SMOTE + RF model.

4. *XGBoost*

Model Type	Accuracy	Class 0 Recall	Class 1 Recall
Base XGBoost	0.748	0.49	0.89
Tuned XGBoost	0.797	0.44	0.99
Balanced XGBoost	0.739	0.53	0.85

For XGBoost, the tuned model gives us the best overall accuracy and class 1 performance, but sacrifies the recall for class 0. On the other hand, the balanced model gets the recall for class 0 to it's max, at 0.53. The base model though is the midpoint in this case. We will therefore go ahead with the Tuned XGBoost for further comparisons.

5. *Artificial Neural Network (ANN)*

Model Type	Accuracy	Class 0 Recall	Class 1 Recall
Base ANN	0.715	0.47	0.85
Tuned ANN	0.707	0.44	0.85

Compared to the other models we tried, the ANN models have the least attractive results. The recall for class 0 is decent, but tuning does not improve it.

I. *Overall Comparison*

For the chosen sub-models in each technique we tried, we get the following table :

Model Type	Accuracy	Class 0 Recall	Class 1 Recall
Logistic Regression (Lasso)	0.814	0.429	0.992
Tuned SVM	0.797	0.44	0.99
Tuned Random Forest	0.797	0.44	0.99
Balanced XGBoost	0.739	0.53	0.85
Base ANN	0.715	0.47	0.85

The table above represents all the techniques that have been used in this project, namely: Logistic Regression, Support Vector Machine, Random Forests, XGBost, and Artificial Neural Networks. Every model was carefully optimized using hyperparameter tuning, and were not only evaluated based on overall accuracy, but also on specific recalls for class 0 (loan rejections) and class 1 (loan approvals), especially for class 0 since all the models had difficulties classifying its values. Logistic regression recorded the highest overall accuracy, followed by Tuned SVM and Tuned Random Forest. As for our main concern, class 0, Balanced XGBoost had by far the highest recall compared to all the other models. Although Logistic Regression and tuned models (RF, SVM) perform best in terms of raw accuracy, this model comparison shows that models such as Balanced XGBoost offer greater sensitivity to the minority class, making them more suitable for applications where fairness or equal error treatment is essential. The final model selection should be based on business priorities like accuracy, fairness, or a balanced trade-off.

IV. CONCLUSION

In this project, we used machine learning techniques like Support Vector Machines (SVM), Random Forest, XGBoost, Artificial Neural Networks (ANN), and Logistic Regression to create a predictive model for assessing loan approvals and denials. From missing data imputation to exploratory data analysis and confirmatory data analysis, we equally used hyperparameter tuning to enhance the performance of each model, and tested their effectiveness using a test set, focusing on overall accuracy and recall for the minority class (loan rejections) as evaluation metrics.

An accuracy of 79.7% was achieved with Tuned Random Forest and XGBoost. The Logistic regression on the other hand scored the best accuracy of 81.4%. Nevertheless, XGBoost is a better model is fairness is an essential criterion, since it recorded the highest recall for the minority class, at 53%.

All in all, these analysis made us come to the realization that while some models excell in accuracy, others do better when it comes to handling class imbalance. Which model should be picked for use should therefore be based on custom situations, and dependent on the context of application.

V. DISCUSSIONS

There are several more suggestions that can be done for the analysis of this dataset.

To begin with, SHapley Additive exPlanations could be used for feature importance and better model explainability.

A stratified K-fold cross-validation could be applied to make sure that the class imbalance is being handled in each fold.

Lastly, determine whether the models performs differently for specific groups like Gender, Marital Status, or even Property Area, by analyzing the recall for each subgroup.

VI. REFERENCES

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794. https://doi.org/10.1145/2939672.2939785

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems (Vol. 30, pp. 4765–4774). https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.), Proceedings of the 9th Python in Science Conference (pp. 51–56).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

OpenAI. (2025). ChatGPT (May 2024 version) [Large language model]. https://chat.openai.com