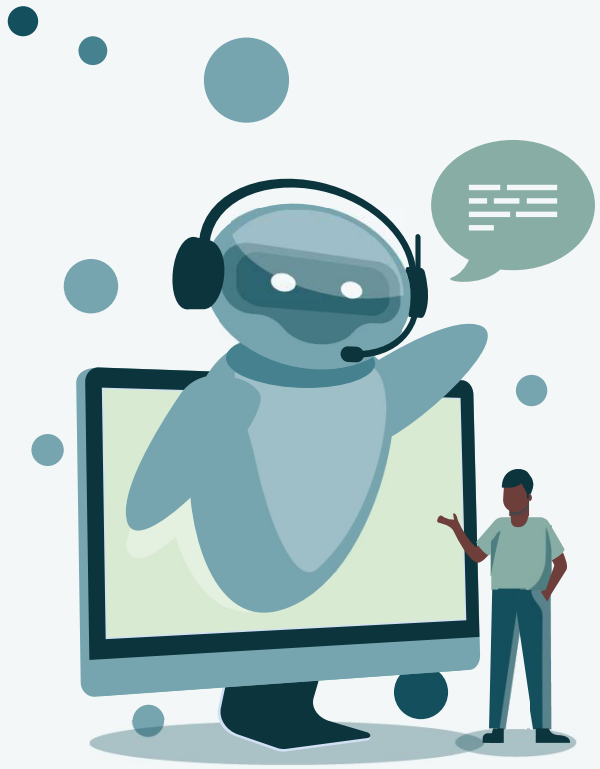


# Predicting Student Job Placements

Angie Achram

# Table of contents


01 Introduction

02 Pre-processing steps

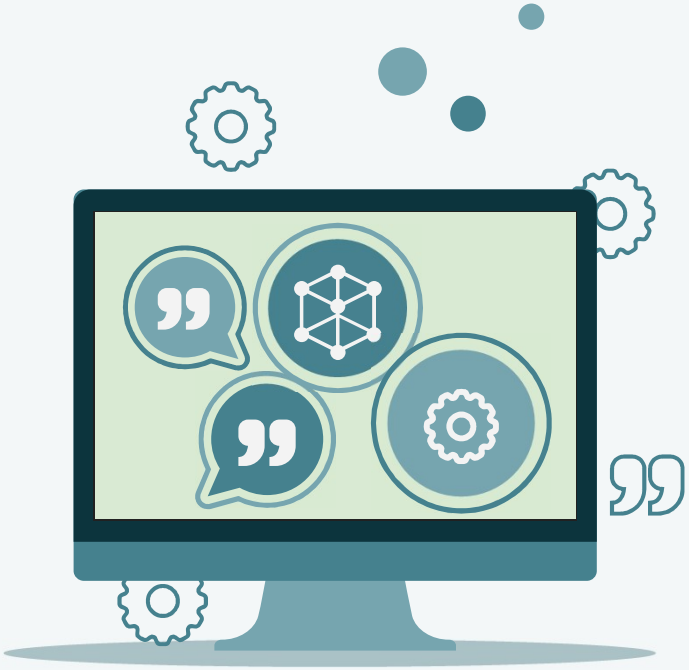
03 CatBoost and SMOTE

04 Results

# 01

# Introduction

Understanding the topic and data



# Introduction

The aim is to classify students into two classes, placed in a job(0) and not placed in a job(1).

To achieve this, we work with the training data by implementing several models and then predict whether the student will get placed or not using our test data.

But... we are dealing with imbalanced data ☹

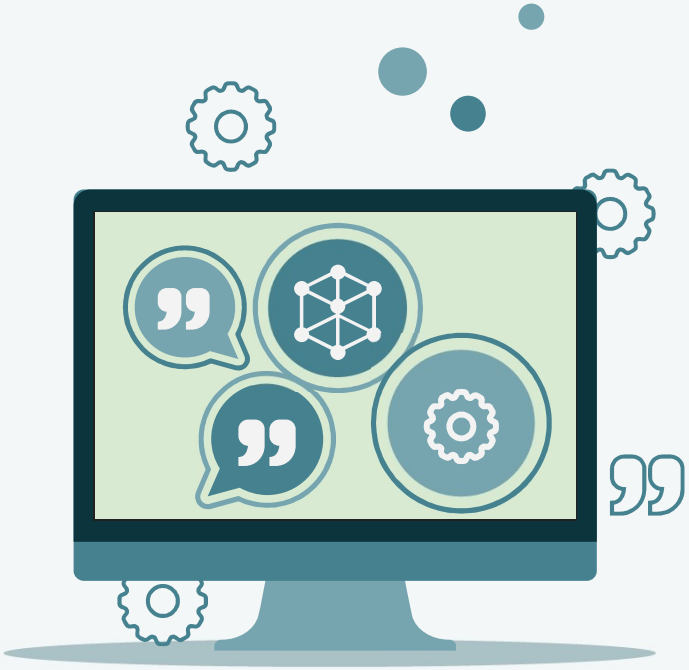
What is imbalanced data? How do we work with it?

| Placement | count |
|---|---|
| 0 | 239 |
| 1 | 61 |

# 02

# Pre-processing steps

Steps taken before implementing the models



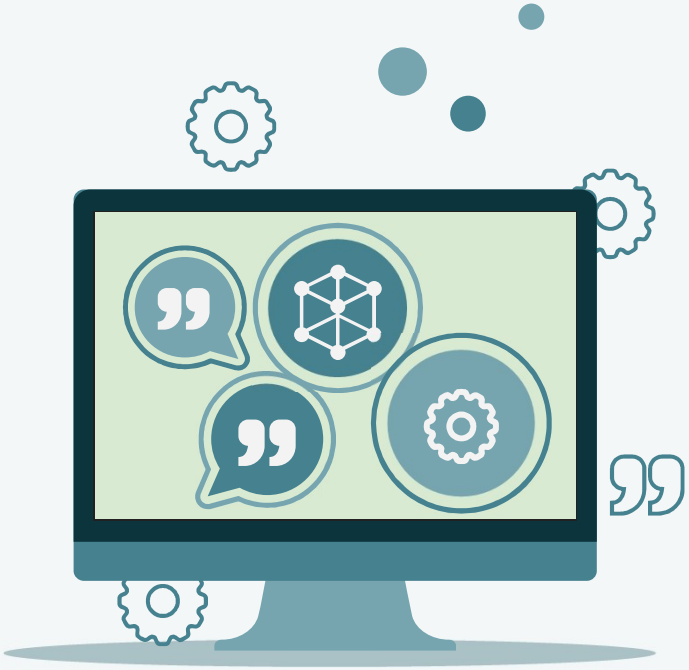
# Process before implementing models

1. Convert categorical features to numerical values. (I used binary and label encoding)
2. Dropped the ID column.
3. Implemented the same changes on the test data.
4. Split training data into train, test and cross validation (cv) with stratify.
    - Split it into 70% training and 30%  testing.
    - Split the training further into 80% training and 20% cv.

# 03

# CatBoost and SMOTE

Explaining the model



# CatBoost

**What is CatBoost?**

CatBoost is a gradient boosting algorithm that uses ordered boosting based on decision trees, but with unique features:

- Handles categorical features, no need for much pre-processing.
- More accurate with its default settings.
- Faster to train.

**Why CatBoost?**

- Handles missing data.
- Reduces overfitting.
- Good at handling bias and complex patterns.

# SMOTE (Synthetic Minority Over-sampling Technique)

**What is SMOTE?**

SMOTE adds more data to our minority class (oversampling).

How it works:

- Start with x (minority) find its K-nearest neighbor (only among the minority class)
- Randomly selects one or more of these neighbors.
- We get: new point= $x + \delta(x_{neighbor} - x)$.

**Why SMOTE?**

- Adds diversity to minority class.
- Does not copy existing data (adds variation).
- Model fairness.
- Prevents overfitting unlike the typical oversampling.

# Drawbacks

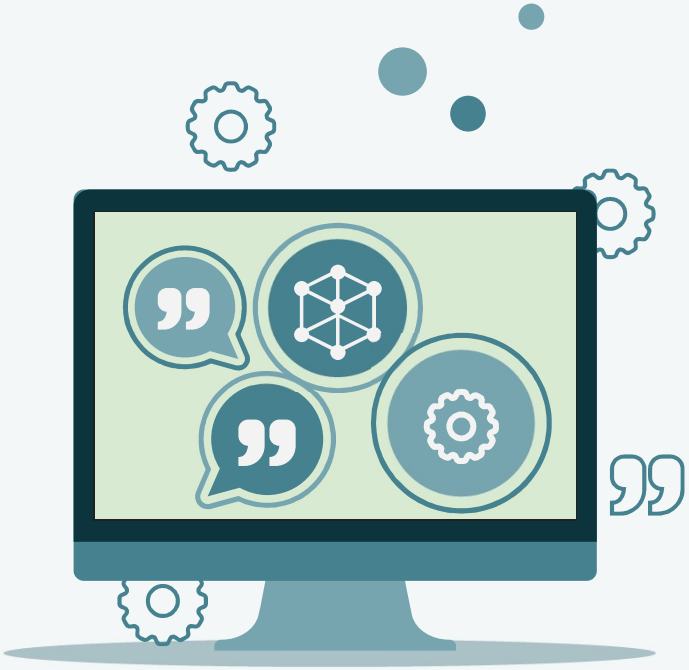
- **CatBoost:**
  - Black-box behavior, harder to interpret.
  - Risk of overfitting (without tuning).

- **SMOTE:**
  - Does not handle high dimensional data well.
  - Ignores majority class.
  - Can cause noise. (creates new points close to the majority class)

# 04

# Results

Results after implementing the model



# Results

**Classification report:**

How many predicted positives were actually correct

How many actual positives did the model catch

Balance between precision and recall (Harmonic mean between precision and recall)

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.83 | 0.79 | 0.81 | 72 |
| 1 | 0.29 | 0.33 | 0.31 | 18 |
| accuracy | | | 0.70 | 90 |
| macro avg | 0.56 | 0.56 | 0.56 | 90 |
| weighted avg | 0.72 | 0.70 | 0.71 | 90 |

# Formulas

| | Predicted: 0 | Predicted: 1 |
|---|---|---|
| **Actual: 0** | True Negative (TN) | False Positive (FP) |
| **Actual: 1** | False Negative (FN) | True Positive (TP) |

- **Precision:** $\frac{TP}{TP+FP}$
- **Recall:** $\frac{TP}{TP+FN}$
- **F1-score:** $\frac{2 * Precision * Recall}{Precision+Recall}$

# Thanks!