Predicting Loans with Machine Learning and Streamlit

Presented by Team#3: Emma, Lucas and Nav

6 March 2023

THE PROBLEM

Industry Overview

Most of the loan application process is laborious. The outcome is a simple 'Yes' or 'No' without further explanation.

Rejection is tough - don't let it make your business tougher. There is communication gap to be bridged.

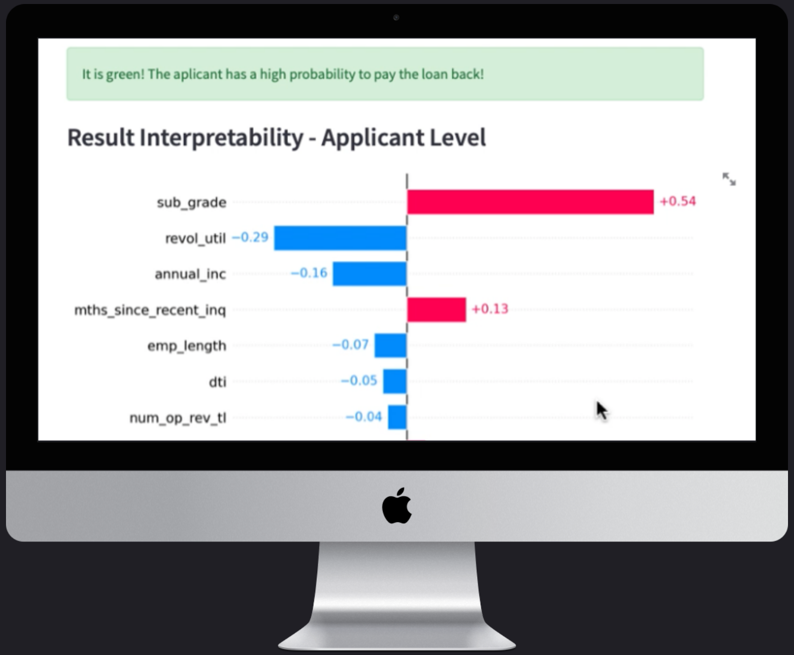
SOLUTION

Unlocking the Insights of Machine Learning

Despite its powerful capabilities, machine learning's charm was often hidden beneath technical jargon and the mystery of its "black box" operations.

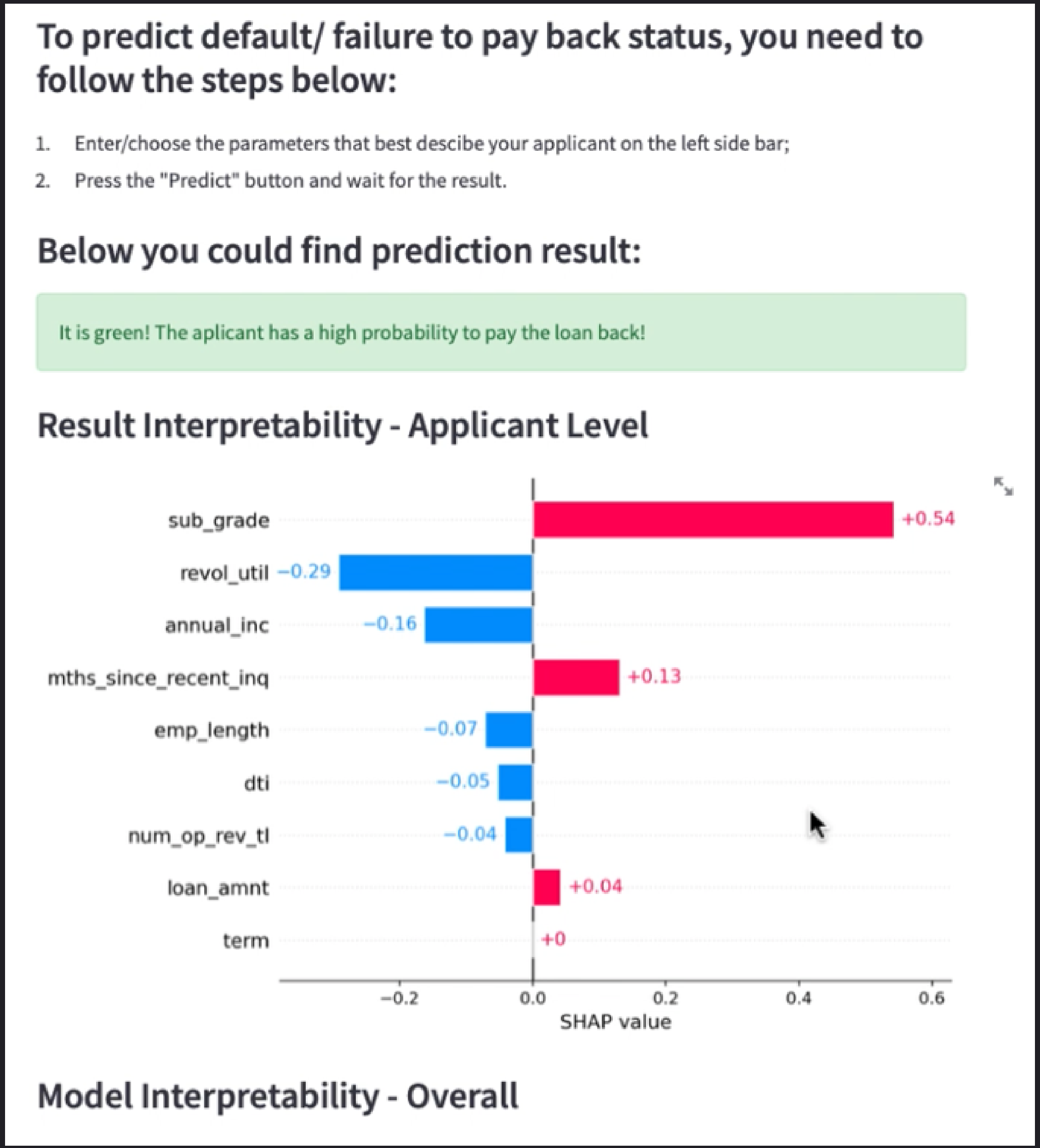
The web application created with Streamlit used visual exploration to demonstrate the power of machine learning, showcasing the dynamic input feature and its SHAP value that drove the decision.

This improved the application's transparency and helped to close the communication gap.



What are SHAP Values?

- Calculation that represents the relative influence of each input features within the algorithm.
- It can be thought of as the 'weighting' that each feature had upon the final result of creditworthiness that the model predicted.



Why Should I Care About SHAP Values?

Unveiling the Creditworthiness Prediction:

SHAP values provide an intuitive way to understand the impact of individual features on a model's predictions

SHAP values can be used to identify important features and detect potential bias in a model

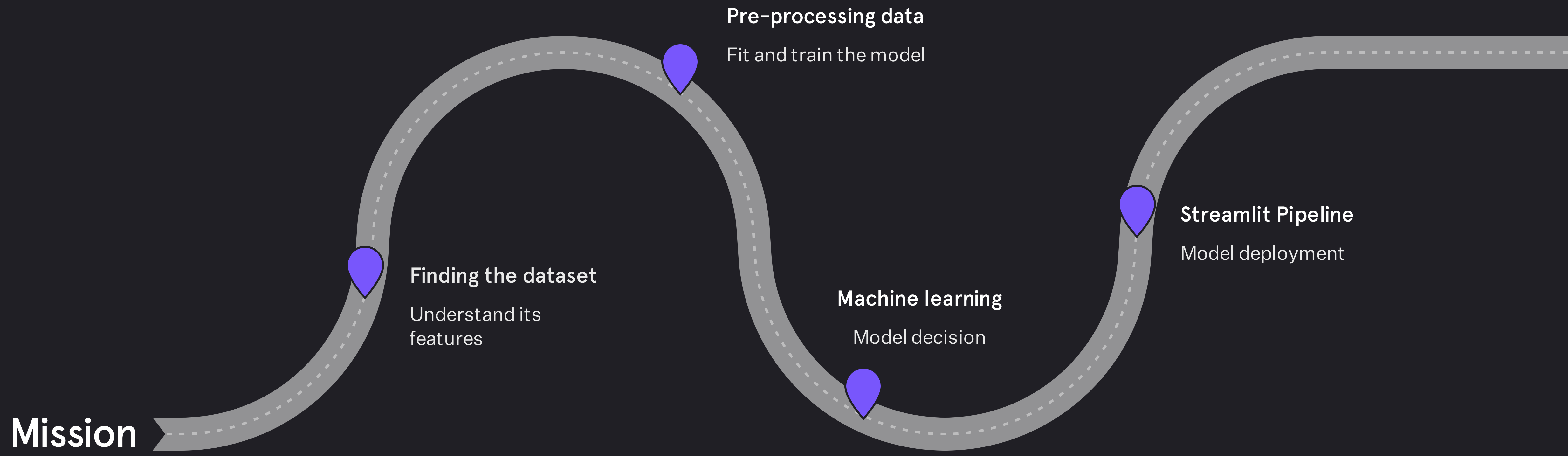
SHAP values can be used to improve model performance by identifying redundant or irrelevant features

Discover How Your Financial Profile Affects Your Risk with SHAP Values

MISSION

Helping customer to learn from their data

How did we get here?



“Teaching machine to think”

Lucas - machine learning specialist

Lending Club dataset is used for model training

Available from Kaggle

Total observations

2.26m

Original features

150

Annual income:

65k

Loan amount

$13k

Avg interest rate

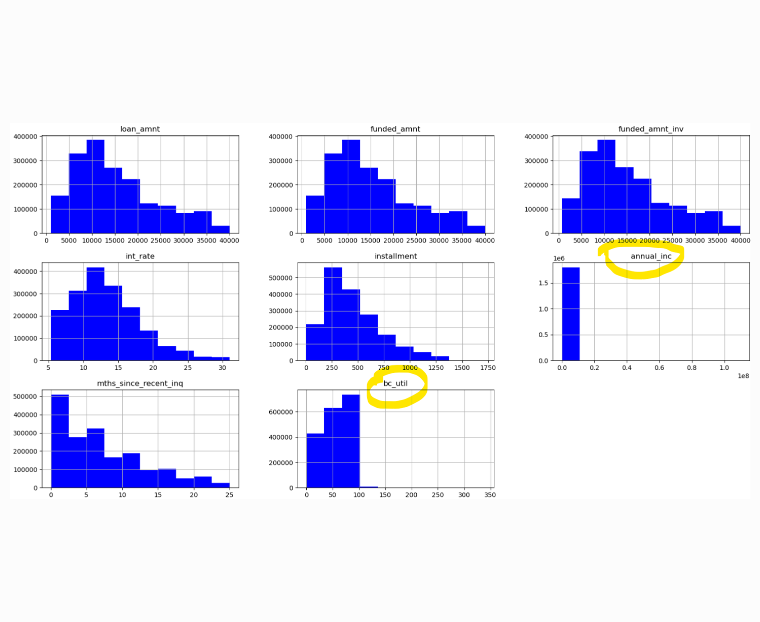
13%

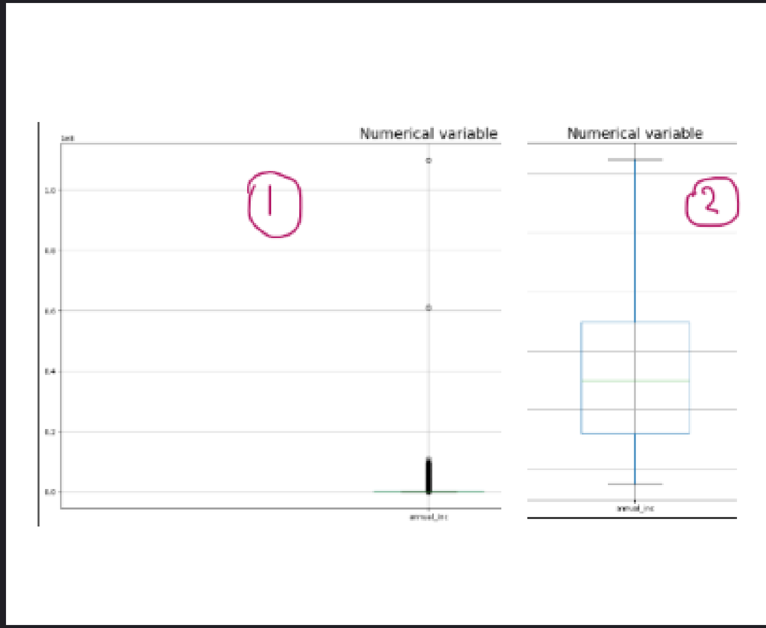
Selected features (aligned with pre-trained model)

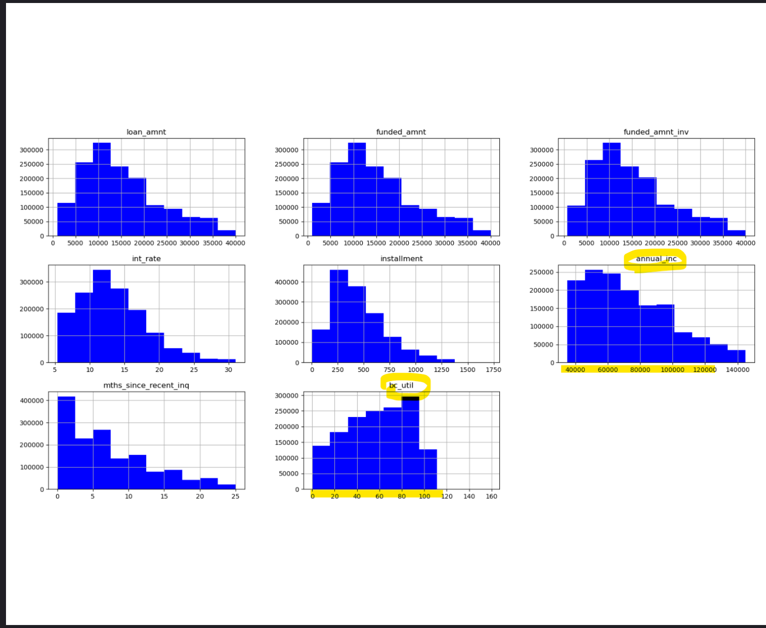
17

Pre-processing data

Dealing with outliers







Outlier

can significantly distort the feature distribution and ML algorithms such as: Linear Regression, Logistic Regression, Support Vector Machine

Remove outlier

low_q = df.quantile(0.08)

high_q = df.quantile(0.92)

new_df = (df< q_hi) & (df > q_low)]

Distribution after dropping outliers

If you have a really good sense of what range the data should fall in, like people's ages, you can safely drop values that are outside of that range.

```
[24]: # Create new labels for ordinal data
ordinal_transformation = {'term': {' 36 months': 1.0, ' 60 months': 2.0
                          'grade': {"A": 1.0, "B": 2.0, "C": 3.0, "D": 
                                    "F": 11.0, "G": 12.0},
                          'sub_grade': {"A1": 1.0, "A2": 2.0, "A3": 3.0
                                    "B1": 11.0, "B2": 12.0, "B3": 13.0, "
                                    "C1": 21.0, "C2": 22.0, "C3": 23.0, "
                                    "D1": 31.0, "D2": 32.0, "D3": 33.0, "
                                    "E1": 41.0, "E2": 42.0, "E3": 43.0, "
                                    "F1": 51.0, "F2": 52.0, "F3": 53.0, "
                                    "G1": 61.0, "G2": 62.0, "G3": 63.0, "
                                      },
                          "emp_length": {"< 1 year": 0.0, '1 year': 1.0
                                     '5 years': 5.0, '6 years': 6.0, '7 y
                                     '10+ years': 10.0 }
}

# Replace data
loans_df = loans_df.replace(ordinal_transformation)

# Preview data
loans_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1485538 entries, 0 to 2260698
Data columns (total 17 columns):
 #   Column                Non-Null Count    Dtype
---  ------                --------------    -----
 0   loan_amnt             1485538 non-null  float64
 1   funded_amnt           1485538 non-null  float64
 2   funded_amnt_inv       1485538 non-null  float64
 3   term                  1485538 non-null  float64
 4   int_rate              1485538 non-null  float64
 5   installment           1485538 non-null  float64
 6   grade                 1485538 non-null  float64
 7   sub_grade             1485538 non-null  float64
 8   emp_title             1485538 non-null  object
 9   emp_length            1485538 non-null  float64
 10  home_ownership        1485538 non-null  object
 11  annual_inc            1485538 non-null  float64
 12  verification_status   1485538 non-null  object
 13  loan_status           1485538 non-null  object
 14  pymnt_plan            1485538 non-null  object
 15  mths_since_recent_inq 1485538 non-null  float64
 16  bc_util               1485538 non-null  float64
dtypes: float64(12), object(5)
memory usage: 204.0+ MB
```

'EMP_LENGTH: 2YR'

FEATURES

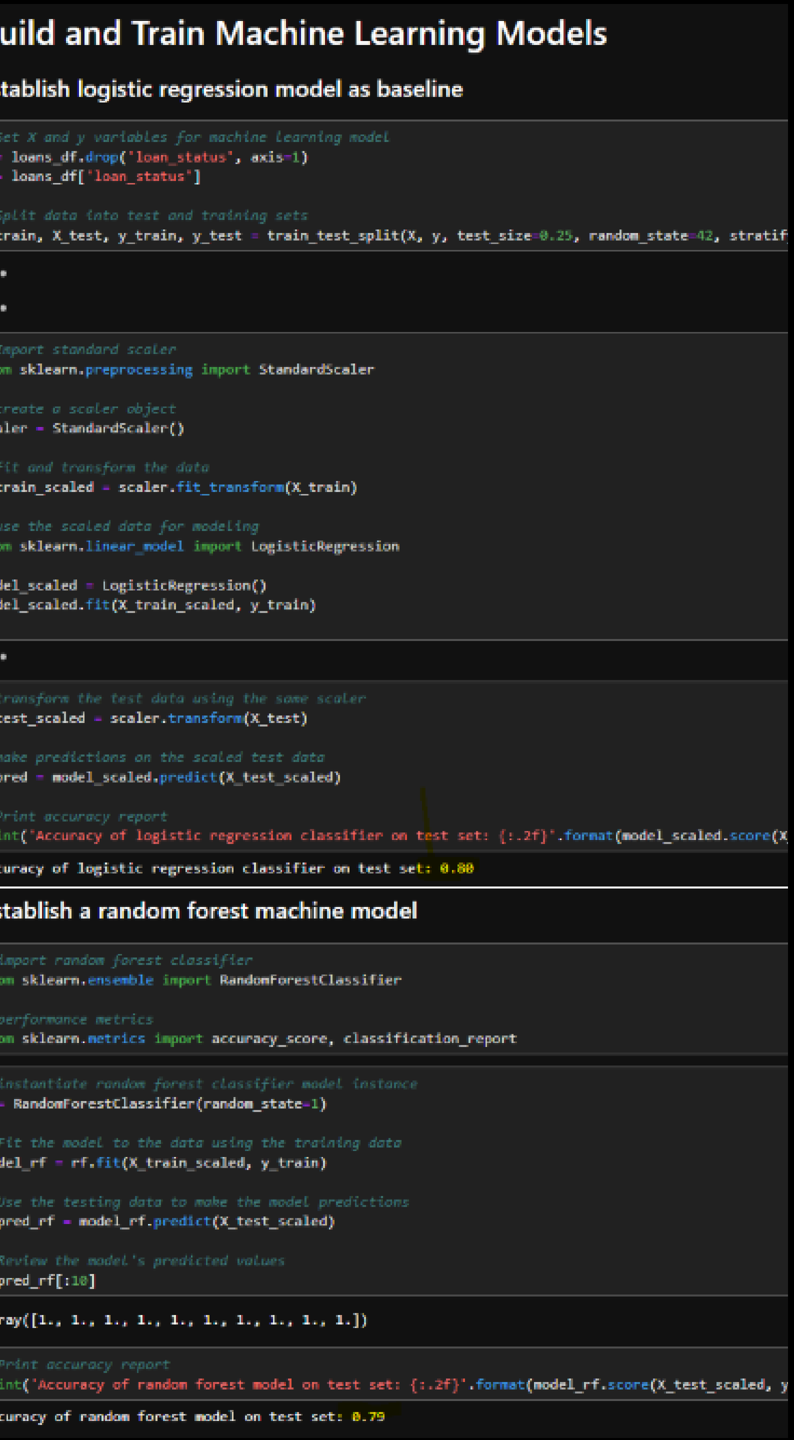
CATEGORICAL

'SUB_GRADE: A1'

'TERM: 6MOS'

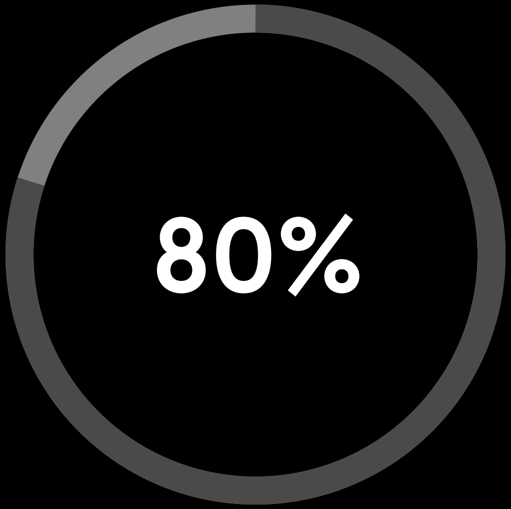
'GRADE: A'

ORDINAL_ENCODING

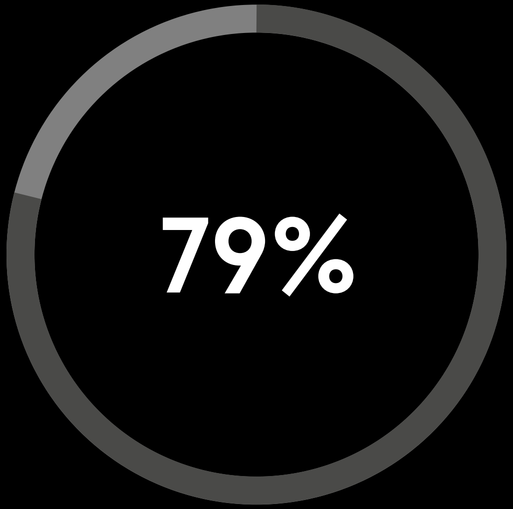


Model decision

Accuracy score



Logistic Regression



Random Forest

"A great user experience starts with great front-end code"

Nat - front end engineer

Streamlit

Streamlit Pipeline

Gather input

- Set up main page layout for instruction
- Set up sidebar layout for interactive user input

Transform input

- Create function to transform user input into "cleaner-type" that fit for the pre-trained model
- Activate function via "Go" button.

Predict

- Loaded pre-trained model
- Use method "predict_prob" based on user input
- Set up risky borderline at lower of 0.78

Prepare test dataset

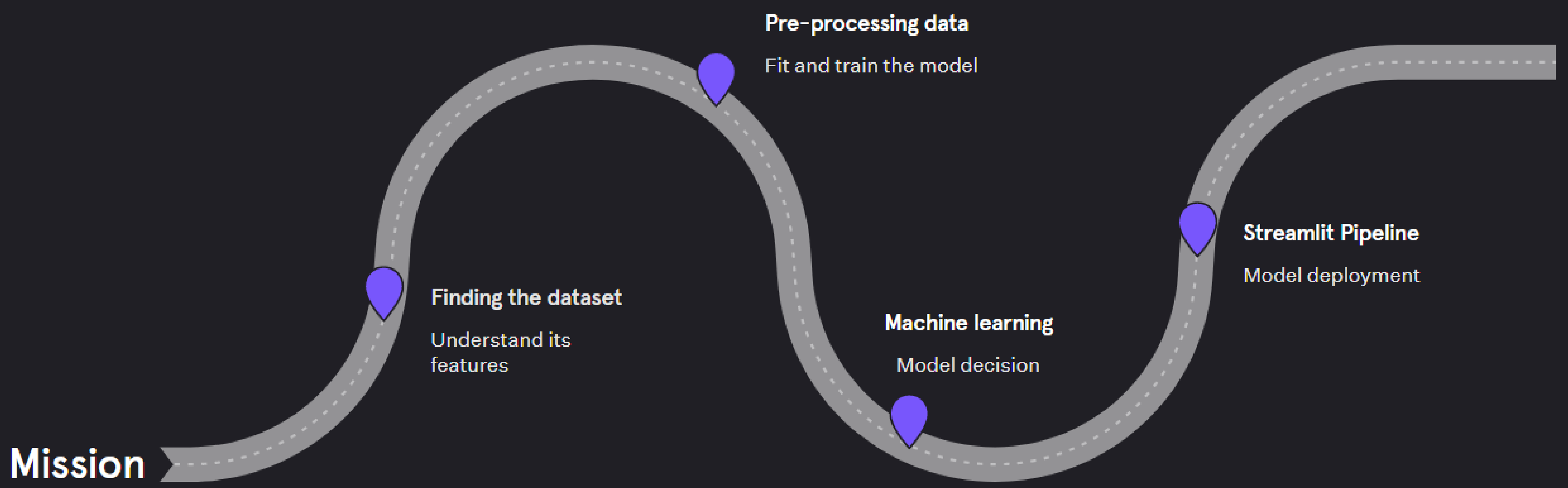
- Cache dataset
- train_test_split

Visualize SHAP values

- initialise JS visualisation code
- Create explainer object for the model used and X_train
- computes SHAP values per new user input

Demos


Deloitte.
log.md
index.js
var express = require('express');
var router = express.Router();
var User = require('../models/user');
// GET /register
router.get('/register', function(req, res, next) {
return res.render('register', { title: 'Sign Up' });
});
// POST /register
router.post('/register', function(req, res, next) {
if (req.body.email &&
req.body.name &&
req.body.favoriteBook &&
req.body.password &&
req.body.confirmPassword) {
// confirm that user typed same password twice
if (req.body.password !== req.body.confirmPassword) {
var err = new Error('Passwords do not match.');
err.status = 400;
return next(err);
}
// create object with form input
var userData = {
email: req.body.email,
name: req.body.name,
favoriteBook: req.body.favoriteBook,
password: req.body.password
};
// use schema's `create` method to insert document into Mongo
User.create(userData, function (error, user) {
if (error) {
JavaScript-Authentication-Mongo-Express/routes/index.js 1:1
LF UTF-8 JavaScript 0 files



Summary & Upgrade potential

- Feature engineering
- Hyperparameter tunning