Credit Card Fraud Detection

Machine Learning Project

By Herman Lin &
Mahika Jain

OVERVIEW

INTRODUCTION 01.

The problem and dataset

SUPERVISED ANALYSIS 02.

Logistic Regression, SVM, and Neural Network

RESULTS 03.

Data obtained from model types

CONCLUSION 04.

Experimental findings and conclusions

01.

INTRODUCTION

In 2020, the Consumer Sentinel Network took in over 4.7 million reports; 2.2 million (46%) of those were of fraud.

Of the nearly 2.2 million fraud reports, 34% indicated losing money. This totaled to more than $3.3 billion lost.

Federal Trade Commission. *Consumer Sentinel Network Data Book 2020.*

From a list of transactions, we want to be able to determine which of those are fraudulent and which are not.

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

DATASET: CREDIT CARD FRAUD DETECTION

- Transactions made by credit cards in Sept. 2013 by european cardholders
- Timeframe of data collection: 2 days
- 30 features: [Time, V1,..., V28, Amount]
  - PCA transformation applied on features V1-V28
- Target: Class {0: if not fraud, 1: if fraud}
- 492 frauds out of 284,807 transactions
  - Highly imbalanced dataset
  - Frauds make up 0.172% of all transactions

https://www.kaggle.com/mlg-ulb/creditcardfraud

02.

SUPERVISED LEARNING

Logistic Regression, Support Vector Machine (SVM), Neural Network

Metrics for Analysis

Accuracy

The number of correctly predicted data points out of all the data points

Precision

The ratio between the number of true positive results to the number of all positive results

Recall

The ratio between true positive results to the number of all samples that should have been identified as positive

F1-Score

The harmonic mean of the precision and recall

01 02 03 04 05 06

01
02
03
04
05
06

Metric Equations

Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$
Precision	$\frac{TP}{TP + FP}$
Recall	$\frac{TP}{TP + FN}$
F1-Score	$\frac{TP}{TP + {}^{1}/_{2}(FP + FN)} = 2 \cdot \frac{precision \cdot recall}{precision + recall}$

More Metrics for Analysis

Precision-Recall Curve

Shows the tradeoff between precision and recall for different thresholds

AP score

Summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold

Loss Curves

Loss function is used to calculate the cost that is added to gradient calculations

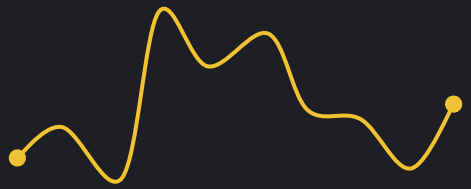
Logistic Regression

Model 1

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

Logistic Regression

Standardization



Polynomial Feature Transformation

No Regularization

Penalty set to ‘none’ to disregard the regularization term

L1 Regularization

L2 Regularization

Used a range of C Values to alter the regularization term

Support Vector Machines

Model 2

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

SVM Kernels

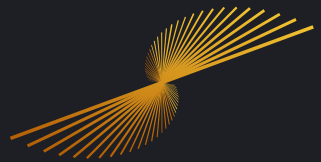
Standardization

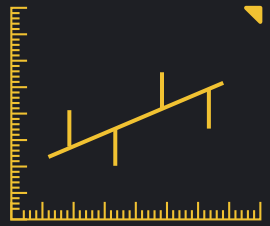
Linear

RBF

Poly

A range of C Values that add different amounts of L2 regularization





Neural Networks

Model 3

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

Neural Networks

Standardization

Activation Function

Logistic/Sigmoid

tanh

ReLU

Iterations

10,000

Hidden Layers

(22)

(22, 22)

(22, 22, 22)

(30)

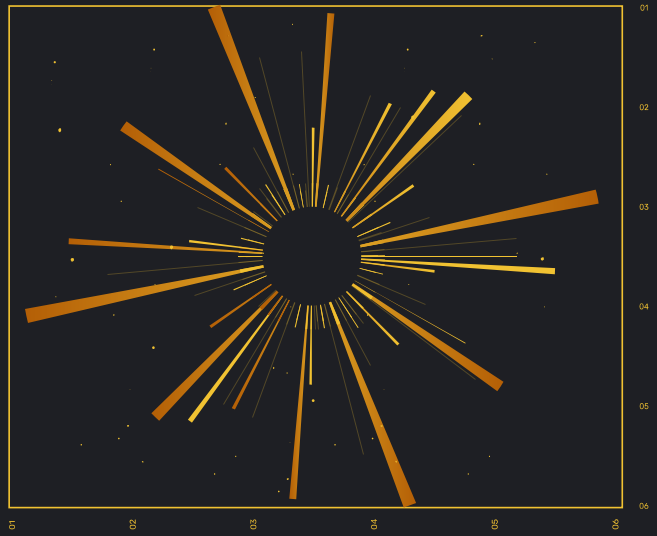
Alpha

0.01

0.001

0.0001

0.00001



03.

Results

Logistic Regression, Support Vector Machine (SVM), Neural Network

Logistic Regression

Best Performed Models from both sets of transformed data and regularization:

- L1 and L2 Regularization models for the StandardScaler data

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

Logistic Regression

StandardScaler

cVal: 1		
	0	1
precision	0.975099	0.983278
recall	0.997106	0.869822
f1-score	0.98598	0.923077

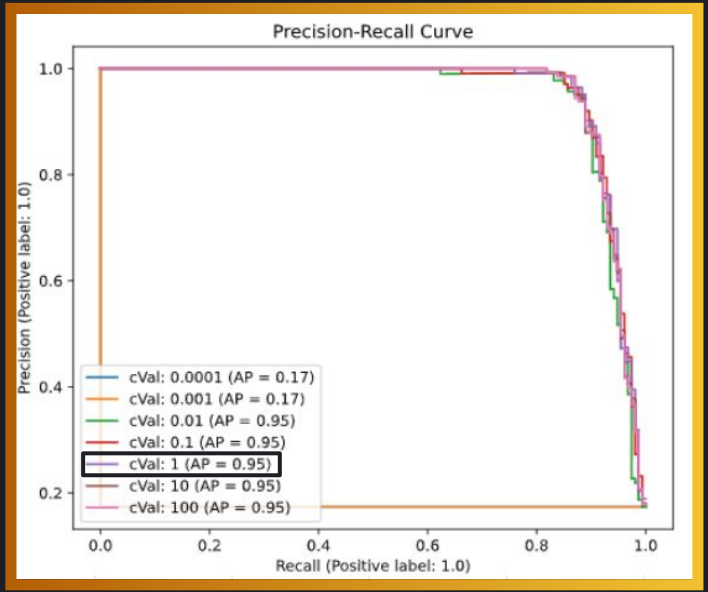
L1 Regularization
<----------------

L2 Regularization
---------------->

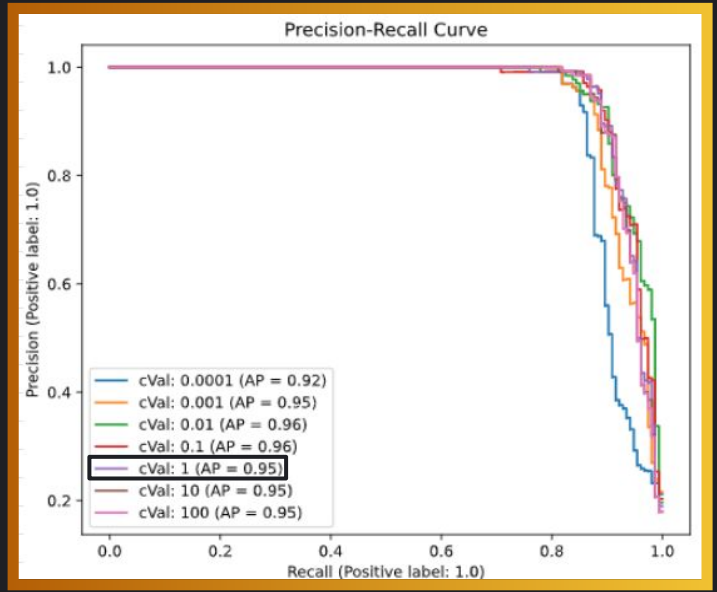
cVal: 1		
	0	1
precision	0.975085	0.98
recall	0.996528	0.869822
f1-score	0.98569	0.92163

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

Logistic Regression with StandardScaler Data



L1 Regularization



L2 Regularization

Support Vector Machines

Best Performed Models out of the three kernels:

- Linear Kernel SVM model
- RBF Kernel SVM model

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

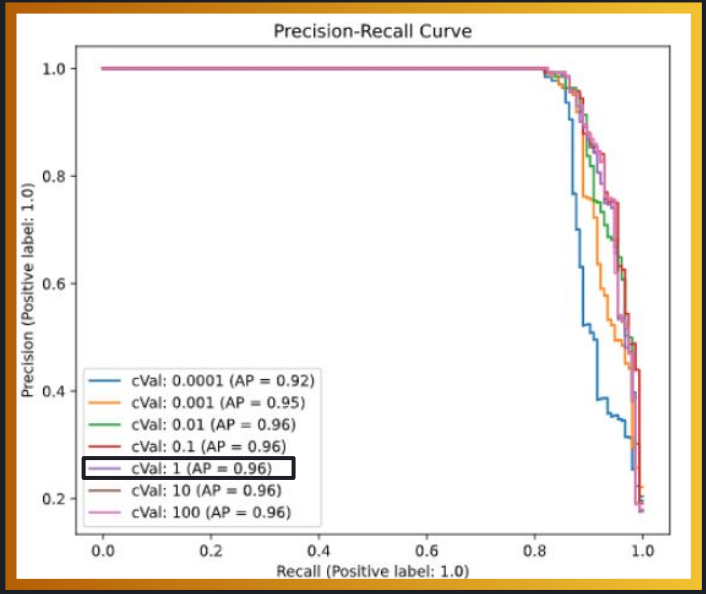
Support Vector Machines

Linear Kernel

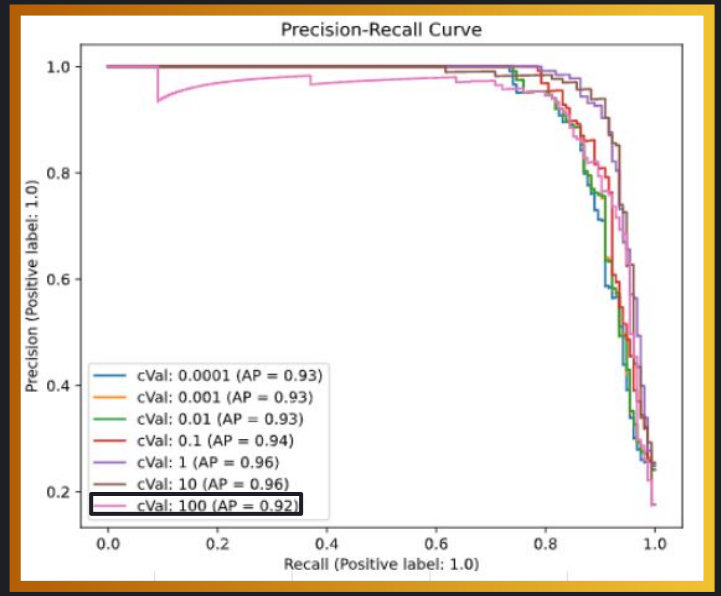
cVal: 1		
	0	1
precision	0.974026	0.989831
recall	0.998264	0.863905
f1-score	0.985996	0.922591

Radial Basis Function Kernel

cVal: 100		
	0	1
precision	0.999422	1
recall	1	0.997041
f1-score	0.999711	0.998519



SVM with Linear Kernel



SVM with RBF Kernel

Neural Networks

Best Performed Model for different hidden layers and regularization:

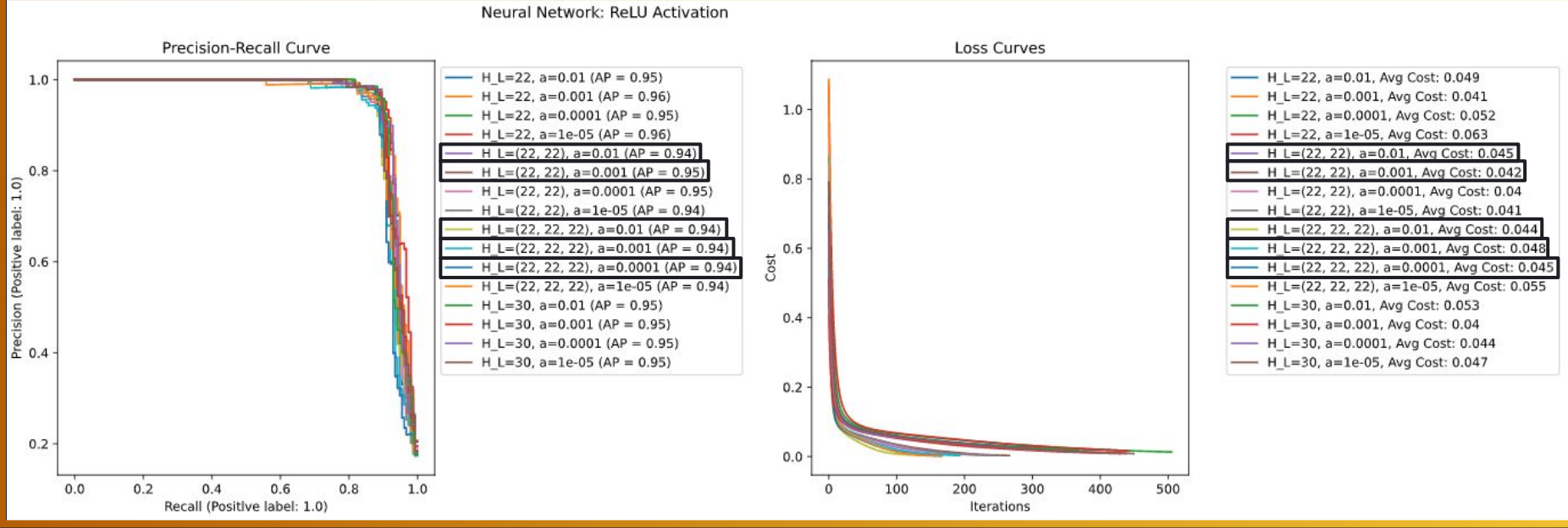
- ReLU Activation function NN model

003-1040559 1250 003-77156.8 1760 0009-14563.7 73273

Neural Networks

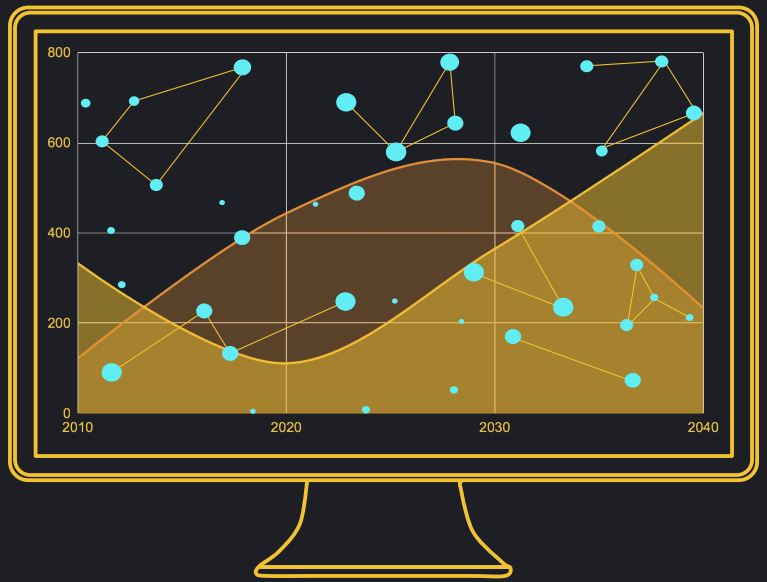
	ReLU Activation	
	0	1
precision	1	1
recall	1	1
f1-score	1	1

Hidden Layers	Alpha Value
(22, 22)	0.01
(22, 22)	0.001
(22, 22, 22)	0.01
(22, 22, 22)	0.0001
(22, 22, 22)	0.00001



NN with ReLU Activation Function

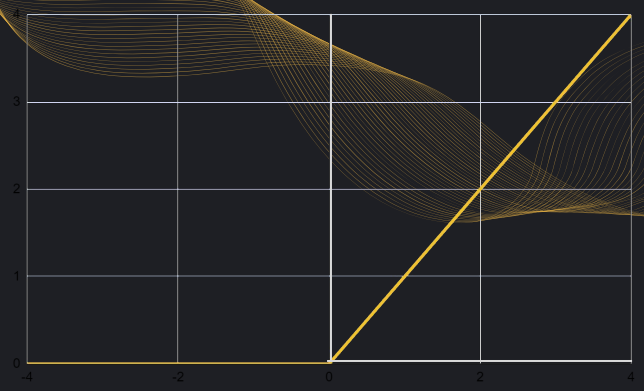
- Hidden Layers: (22), (22, 22), (22, 22, 22), (30)
- Alphas: 0.01, 0.001, 0.0001, 0.00001



04.

Conclusion

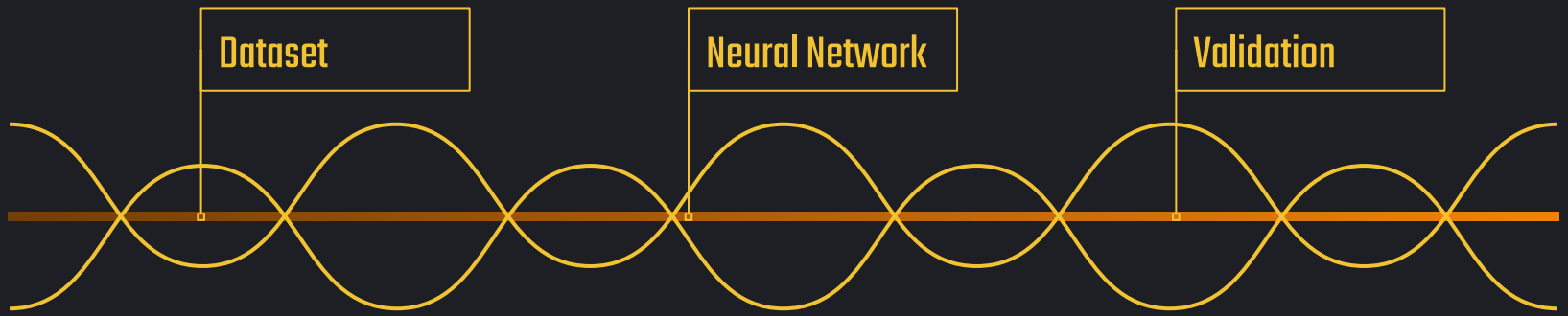
Selected Model: ReLU Neural Network Model



Reasons:

- Lower average cost from the loss curve graph
- Overall high AP score in the precision-recall curve
- Optimal precision, recall, and f1-score
- Range of alphas for regularization term to penalize overfitting

Improvements on Selected Model



Test on different dataset sizes and ratios

Test varying number of neurons and hidden layers or try to use dropout layers

Use cross validation to reduce error and improve model performance

THANK YOU

Questions?