TASK 0

Assignment

# Summarry

ANNs:

Regression

-Practice 4 datasets

-Documentation

• Dataset description

• Table

• DatasetLink

• Model used

• Results

Classification

-Practice 6 datasets

-Documentation

• Dataset description

• Table

• DatasetLink

• Model used

• Results

CNNs:

-Practice 2 datasets

-Documentation

• Dataset description

• Table

• DatasetLink

• Model used/Results

# CNN Models

# Model 1(Dog Breed Predictor)

https://www.kaggle.com/code/jamalnaseer/cnn-dog-breed-classifier-100

## Dataset

This dataset contains a collection of images for 10 different dog breeds, meticulously gathered and organized to facilitate various computer vision tasks such as image classification and object detection. The dataset includes the following breeds:

* **Golden Retriever**
* **German Shepherd**
* **Labrador Retriever**
* **Bulldog**
* **Beagle**
* **Poodle**
* **Rottweiler**
* **Yorkshire Terrier**
* **Boxer**
* **Dachshund**

Each breed is represented by 100 images, stored in separate directories named after the respective breed. The images have been curated to ensure diversity and relevance, making this dataset a valuable resource for training and evaluating machine learning models in the field of computer vision.

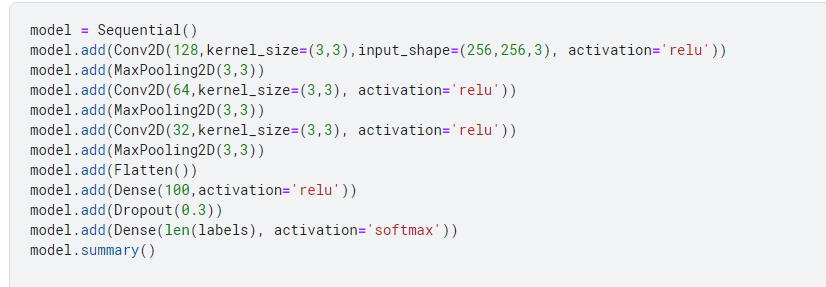
**File Information:**

1. **Directory Structure:**
   * dataset/: Root directory containing all subdirectories for each dog breed.
     + Golden\_Retriever/: Contains 100 images of Golden Retrievers.
     + German\_Shepherd/: Contains 100 images of German Shepherds.
     + Labrador\_Retriever/: Contains 100 images of Labrador Retrievers.
     + Bulldog/: Contains 100 images of Bulldogs.
     + Beagle/: Contains 100 images of Beagles.
     + Poodle/: Contains 100 images of Poodles.
     + Rottweiler/: Contains 100 images of Rottweilers.
     + Yorkshire\_Terrier/: Contains 100 images of Yorkshire Terriers.
     + Boxer/: Contains 100 images of Boxers.
     + Dachshund/: Contains 100 images of Dachshunds.
2. **Files:**
   * Each subdirectory (named after a dog breed) contains 100 image files in JPG format.
     + Example file names: Golden\_Retriever\_1.jpg, German\_Shepherd\_1.jpg, etc.

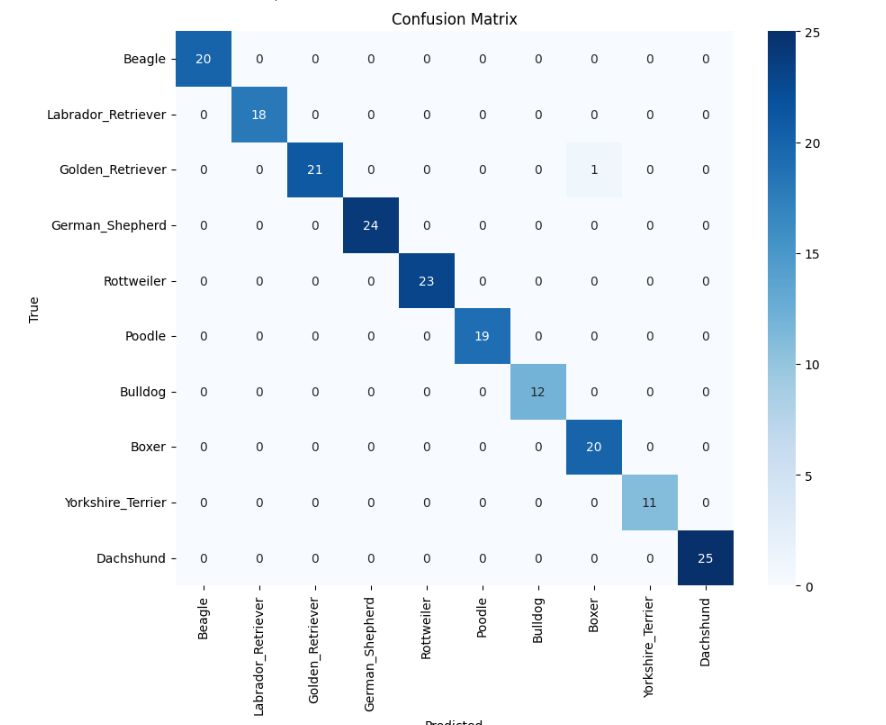
## Handling the dataset

# 

## Model



## Results



# Model 2(Fruit and vegetable disease)

https://www.kaggle.com/code/jamalnaseer/fruit-and-vegetable-disease-healthy-vs-rotten

## Dataset

**Overview**  
The Fruit and Vegetable Diseases Dataset directory contains a comprehensive collection of images categorized to assist in the development of machine learning models for detecting diseases in various fruits and vegetables. This dataset is ideal for tasks such as image classification and deep learning.

**About Dataset Directories**  
The Fruit and Vegetable Diseases Dataset consists of 28 directories, each representing a combination of healthy and rotten images for 14 different types of fruits and vegetables. This dataset is intended for training and evaluating machine learning models for disease detection. Images are categorized and organized to facilitate easy access and utilization for image classification and deep learning tasks.

**Data Collection**  
This dataset has been compiled from various reputable sources, including Kaggle and GitHub repositories. Each image has been manually inspected and categorized to ensure high quality and relevance for the task of disease detection in fruits and vegetables.

**Usage**  
The dataset is highly versatile and can be used for a variety of machine learning tasks, including:

* Image Classification
* Transfer Learning
* Deep Learning  
  It is particularly useful for training models that can identify and distinguish between healthy and rotten produce, making it ideal for applications in agricultural technology and food quality control.

**Metadata**

* Number of Classes: 28 (Healthy and Rotten categories for 14 different fruits and vegetables)
* Image Format: JPEG/PNG
* Image Size: Varies (recommended to resize to a standard size for consistent training)

**Recommended Practices**  
To maximize the utility of this dataset, consider the following:

* Preprocessing: Resize images to a uniform size suitable for your model.
* Augmentation: Apply data augmentation techniques to enhance model robustness.
* Validation: Use a separate validation set to tune and evaluate your models effectively.

This dataset is an excellent resource for developing advanced machine learning models for detecting diseases in fruits and vegetables, contributing to improved food safety and quality assurance practices.

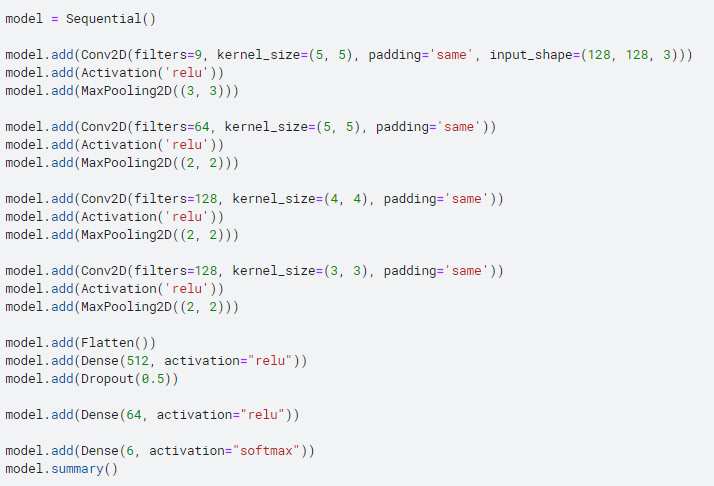
**Directory Structure**  
The directory is organized to facilitate easy navigation and access to specific types of images. Each type of fruit and vegetable has its own folder, which is further divided into subfolders for Healthy and Rotten images:

Fruit And Vegetable Diseases Dataset/  
|-- Apple\_Healthy/  
|-- Apple\_Rotten/  
|-- Banana\_Healthy/  
|-- Banana\_Rotten/  
|-- Bellpepper\_Healthy/  
|-- Bellpepper\_Rotten/  
|-- Carrot\_Healthy/  
|-- Carrot\_Rotten/  
|-- Cucumber\_Healthy/  
|-- Cucumber\_Rotten/  
|-- Grape\_Healthy/  
|-- Grape\_Rotten/  
|-- Guava\_Healthy/  
|-- Guava\_Rotten/  
|-- Jujube\_Healthy/  
|-- Jujube\_Rotten/  
|-- Mango\_Healthy/  
|-- Mango\_Rotten/  
|-- Orange\_Healthy/  
|-- Orange\_Rotten/  
|-- Pomegranate\_Healthy/  
|-- Pomegranate\_Rotten/  
|-- Potato\_Healthy/  
|-- Potato\_Rotten/  
|-- Strawberry\_Healthy/  
|-- Strawberry\_Rotten/  
|-- Tomato\_Healthy/  
|-- Tomato\_Rotten/

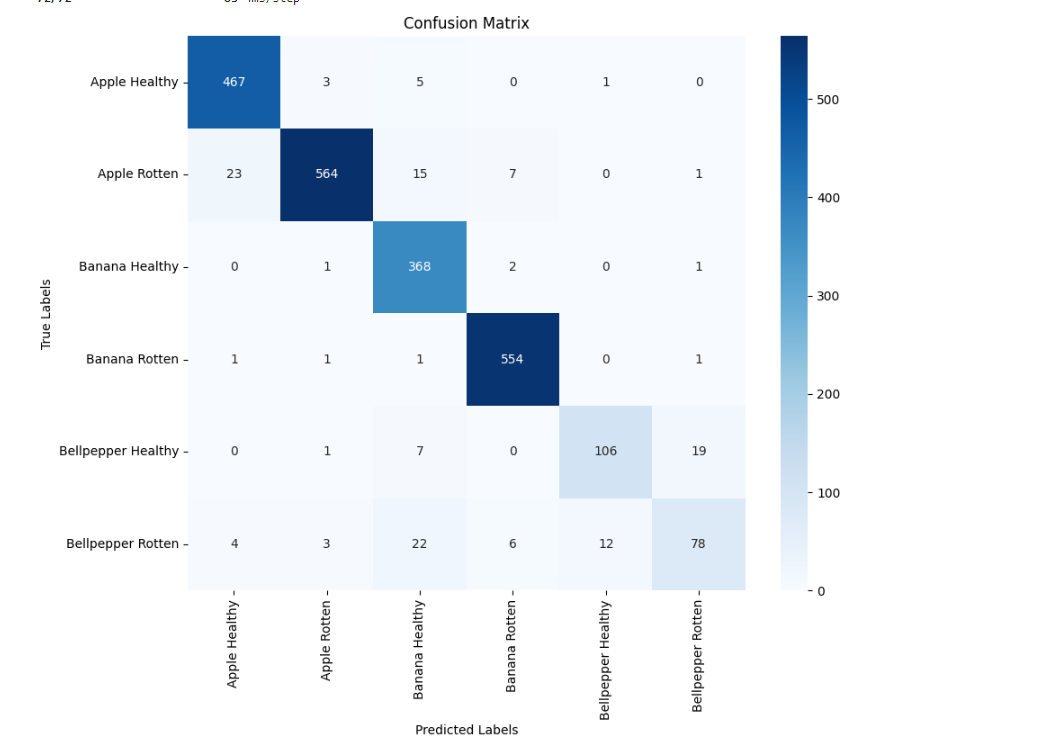
## Data Handling



## Model



## Results



# Regression Model 1 (Flight Price Predictor)

Notebook/Dataset Link:

https://www.kaggle.com/code/jamalnaseer/fight-price-predictor

## Dataset

**Shape of DataFrame:** (300153, 12)

**Columns in DataFrame:** ['Unnamed: 0', 'airline', 'flight', 'source\_city', 'departure\_time', 'stops', 'arrival\_time', 'destination\_city', 'class', 'duration', 'days\_left', 'price']

**Data types of columns:**

Unnamed: 0 int64

airline object

flight object

source\_city object

departure\_time object

stops object

arrival\_time object

destination\_city object

class object

duration float64

days\_left int64

price int64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 300153 entries, 0 to 300152

Data columns (total 12 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Unnamed: 0 300153 non-null int64

1 airline 300153 non-null object

2 flight 300153 non-null object

3 source\_city 300153 non-null object

4 departure\_time 300153 non-null object

5 stops 300153 non-null object

6 arrival\_time 300153 non-null object

7 destination\_city 300153 non-null object

8 class 300153 non-null object

9 duration 300153 non-null float64

10 days\_left 300153 non-null int64

11 price 300153 non-null int64

dtypes: float64(1), int64(3), object(8)

memory usage: 27.5+ MB

**Number of unique values in each column:**

**Unnamed: 0**: 300153

**airline**: 6

**flight**: 1561

**source\_city**: 6

**departure\_time**: 6

**stops**: 3

**arrival\_time**: 6

**destination\_city**: 6

**class**: 2

**duration**: 476

**days\_left**: 49

**price**: 12157

**Number of null values in each column:**

Unnamed: 0 0

airline 0

flight 0

source\_city 0

departure\_time 0

stops 0

arrival\_time 0

destination\_city 0

class 0

duration 0

days\_left 0

price 0

dtype: int64

**Number of duplicate rows:** 0

**Descriptive statistics of DataFrame:**

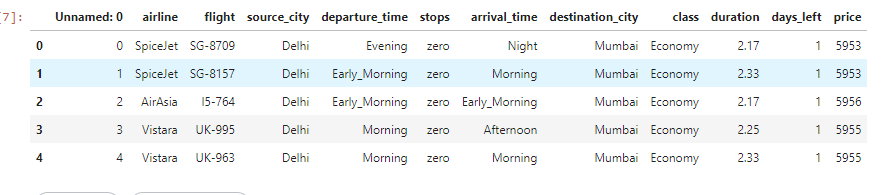
count mean std min 25% \

Unnamed: 0 300153.0 150076.000000 86646.852011 0.00 75038.00

duration 300153.0 12.221021 7.191997 0.83 6.83

days\_left 300153.0 26.004751 13.561004 1.00 15.00

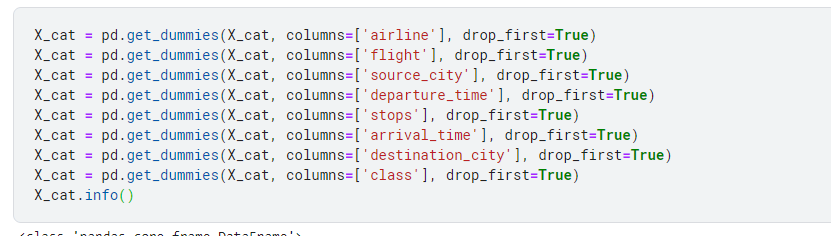
price 300153.0 20889.660523 22697.767366 1105.00 4783.00



## Model Used

Column to be predicted: Price

Independent variables:, 'airline', 'flight', 'source\_city', 'departure\_time', 'stops', 'arrival\_time', 'destination\_city', 'class', 'duration', 'days\_left',



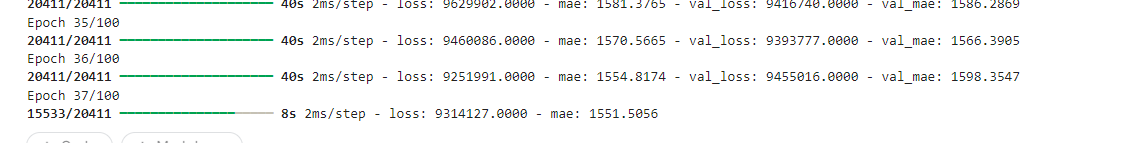
The above men Columns were One hot encoded as they were categorical

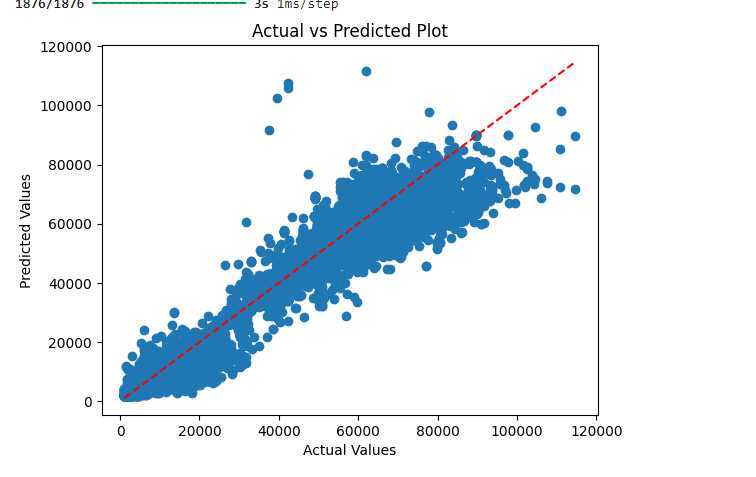
And the first column of the Dummy variables was dropped to avoid the dummy variable trap



* The input layer had 1590 values, the input layers was added within the function of the first hidden layer
* Three hidden layers with the activation function ”Rectifier” were added to the model for better accuracy
* The layer were compiled with the loss function of mean squared error
* Model was trained for 100 epoch and 15% of the training data was used for validation within the training

## Results





# Model 2 (Car Price Prediction)

Notebook/Dataset Link: https://www.kaggle.com/code/jamalnaseer/car-price-prediction

## Dataset

**Shape of DataFrame:** (500, 9)

**Columns in DataFrame:** ['customer name', 'customer e-mail', 'country', 'gender', 'age', 'annual Salary', 'credit card debt', 'net worth', 'car purchase amount']

**Data types of columns:**

customer name object

customer e-mail object

country object

gender int64

age float64

annual Salary float64

credit card debt float64

net worth float64

car purchase amount float64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 500 entries, 0 to 499

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 customer name 500 non-null object

1 customer e-mail 500 non-null object

2 country 500 non-null object

3 gender 500 non-null int64

4 age 500 non-null float64

5 annual Salary 500 non-null float64

6 credit card debt 500 non-null float64

7 net worth 500 non-null float64

8 car purchase amount 500 non-null float64

dtypes: float64(5), int64(1), object(3)

memory usage: 35.3+ KB

**Number of unique values in each column:**

**customer name**: 498

**customer e-mail**: 500

**country**: 211

**gender**: 2

**age**: 500

**annual Salary**: 500

**credit card debt**: 500

**net worth**: 500

**car purchase amount**: 500

**Number of null values in each column:**

customer name 0

customer e-mail 0

country 0

gender 0

age 0

annual Salary 0

credit card debt 0

net worth 0

car purchase amount 0

dtype: int64

**Number of duplicate rows:** 0

**Descriptive statistics of DataFrame:**

count mean std min \

gender 500.0 0.506000 0.500465 0.0

age 500.0 46.241674 7.978862 20.0

annual Salary 500.0 62127.239608 11703.378228 20000.0

credit card debt 500.0 9607.645049 3489.187973 100.0

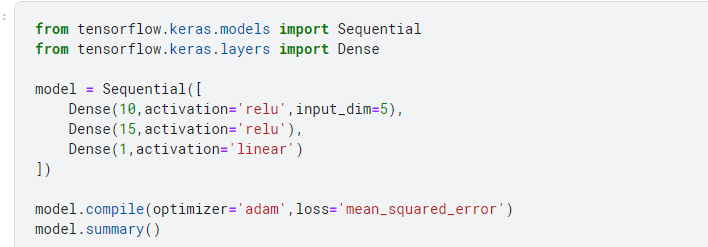
net worth 500.0 431475.713625 173536.756340 20000.0

car purchase amount 500.0 44209.799218 10773.178744 9000.0

## Model

The model has to predict how much a customer is willing to spend on a car

The Independent variables used to predict: 'country', 'gender', 'age', 'annual Salary', 'credit card debt', 'net worth'



## Results

A graph with blue dots and red line

Description automatically generated

# Model 3 (Horsepower vs Car Price)

https://www.kaggle.com/code/jamalnaseer/hpvs-price

## Dataset

**Shape of DataFrame:** (200, 2)

**Columns in DataFrame:** ['KekuatanMesin', 'Harga']

**Data types of columns:**

KekuatanMesin int64

Harga float64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 200 entries, 0 to 199

Data columns (total 2 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 KekuatanMesin 200 non-null int64

1 Harga 200 non-null float64

dtypes: float64(1), int64(1)

memory usage: 3.2 KB

**Number of unique values in each column:**

**KekuatanMesin**: 58

**Harga**: 185

**Number of null values in each column:**

KekuatanMesin 0

Harga 0

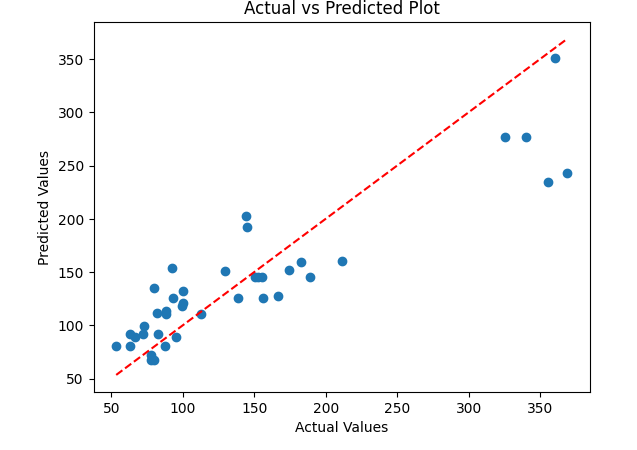
dtype: int64

**Number of duplicate rows:** 9

## Model:



## Results:



# Model 4 IMDB movie ratings

https://www.kaggle.com/code/jamalnaseer/imdb-rating

## Dataset

**Shape of DataFrame:** (400, 9)

**Columns in DataFrame:** ['Title', 'IMDb Rating', 'Year', 'Certificates', 'Genre', 'Director', 'Star Cast', 'MetaScore', 'Duration (minutes)']

**Data types of columns:**

Title object

IMDb Rating float64

Year int64

Certificates object

Genre object

Director object

Star Cast object

MetaScore float64

Duration (minutes) float64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 400 entries, 0 to 399

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Title 400 non-null object

1 IMDb Rating 400 non-null float64

2 Year 400 non-null int64

3 Certificates 400 non-null object

4 Genre 400 non-null object

5 Director 400 non-null object

6 Star Cast 400 non-null object

7 MetaScore 400 non-null float64

8 Duration (minutes) 400 non-null float64

dtypes: float64(3), int64(1), object(5)

memory usage: 28.2+ KB

**Number of unique values in each column:**

**Title**: 251

**IMDb Rating**: 45

**Year**: 47

**Certificates**: 11

**Genre**: 10

**Director**: 199

**Star Cast**: 247

**MetaScore**: 66

**Duration (minutes)**: 79

**Number of null values in each column:**

Title 0

IMDb Rating 0

Year 0

Certificates 0

Genre 0

Director 0

Star Cast 0

MetaScore 0

Duration (minutes) 0

dtype: int64

**Number of duplicate rows:** 145

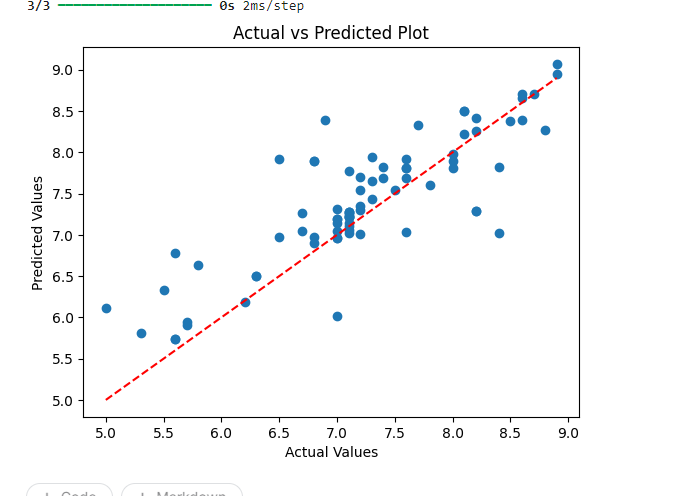
A group of names on a white background

Description automatically generated

## Model



## Results



# Classification Model 1(Product Defect Predictor)

https://www.kaggle.com/code/jamalnaseer/defect-prediction-with-ann

## Dataset

**Shape of DataFrame:** (3240, 17)

**Columns in DataFrame:** ['ProductionVolume', 'ProductionCost', 'SupplierQuality', 'DeliveryDelay', 'DefectRate', 'QualityScore', 'MaintenanceHours', 'DowntimePercentage', 'InventoryTurnover', 'StockoutRate', 'WorkerProductivity', 'SafetyIncidents', 'EnergyConsumption', 'EnergyEfficiency', 'AdditiveProcessTime', 'AdditiveMaterialCost', 'DefectStatus']

**Data types of columns:**

ProductionVolume int64

ProductionCost float64

SupplierQuality float64

DeliveryDelay int64

DefectRate float64

QualityScore float64

MaintenanceHours int64

DowntimePercentage float64

InventoryTurnover float64

StockoutRate float64

WorkerProductivity float64

SafetyIncidents int64

EnergyConsumption float64

EnergyEfficiency float64

AdditiveProcessTime float64

AdditiveMaterialCost float64

DefectStatus int64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 3240 entries, 0 to 3239

Data columns (total 17 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 ProductionVolume 3240 non-null int64

1 ProductionCost 3240 non-null float64

2 SupplierQuality 3240 non-null float64

3 DeliveryDelay 3240 non-null int64

4 DefectRate 3240 non-null float64

5 QualityScore 3240 non-null float64

6 MaintenanceHours 3240 non-null int64

7 DowntimePercentage 3240 non-null float64

8 InventoryTurnover 3240 non-null float64

9 StockoutRate 3240 non-null float64

10 WorkerProductivity 3240 non-null float64

11 SafetyIncidents 3240 non-null int64

12 EnergyConsumption 3240 non-null float64

13 EnergyEfficiency 3240 non-null float64

14 AdditiveProcessTime 3240 non-null float64

15 AdditiveMaterialCost 3240 non-null float64

16 DefectStatus 3240 non-null int64

dtypes: float64(12), int64(5)

memory usage: 430.4 KB

**Number of unique values in each column:**

**ProductionVolume**: 862

**ProductionCost**: 3240

**SupplierQuality**: 3240

**DeliveryDelay**: 6

**DefectRate**: 3240

**QualityScore**: 3240

**MaintenanceHours**: 24

**DowntimePercentage**: 3240

**InventoryTurnover**: 3240

**StockoutRate**: 3240

**WorkerProductivity**: 3240

**SafetyIncidents**: 10

**EnergyConsumption**: 3240

**EnergyEfficiency**: 3240

**AdditiveProcessTime**: 3240

**AdditiveMaterialCost**: 3240

**DefectStatus**: 2

**Number of null values in each column:**

ProductionVolume 0

ProductionCost 0

SupplierQuality 0

DeliveryDelay 0

DefectRate 0

QualityScore 0

MaintenanceHours 0

DowntimePercentage 0

InventoryTurnover 0

StockoutRate 0

WorkerProductivity 0

SafetyIncidents 0

EnergyConsumption 0

EnergyEfficiency 0

AdditiveProcessTime 0

AdditiveMaterialCost 0

DefectStatus 0

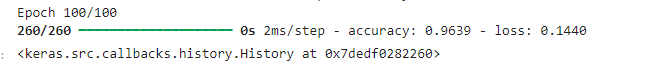
dtype: int64

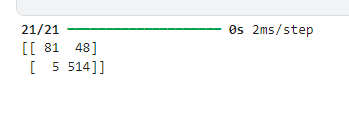
**Number of duplicate rows:** 0

## Model



## Results





………………….

# Classification Model 2(Weather Prediction)

https://www.kaggle.com/code/jamalnaseer/weather-prediction-ann

## Dataset

**Shape of DataFrame:** (13200, 11)

**Columns in DataFrame:** ['Temperature', 'Humidity', 'Wind Speed', 'Precipitation (%)', 'Cloud Cover', 'Atmospheric Pressure', 'UV Index', 'Season', 'Visibility (km)', 'Location', 'Weather Type']

**Data types of columns:**

Temperature float64

Humidity int64

Wind Speed float64

Precipitation (%) float64

Cloud Cover object

Atmospheric Pressure float64

UV Index int64

Season object

Visibility (km) float64

Location object

Weather Type object

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 13200 entries, 0 to 13199

Data columns (total 11 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Temperature 13200 non-null float64

1 Humidity 13200 non-null int64

2 Wind Speed 13200 non-null float64

3 Precipitation (%) 13200 non-null float64

4 Cloud Cover 13200 non-null object

5 Atmospheric Pressure 13200 non-null float64

6 UV Index 13200 non-null int64

7 Season 13200 non-null object

8 Visibility (km) 13200 non-null float64

9 Location 13200 non-null object

10 Weather Type 13200 non-null object

dtypes: float64(5), int64(2), object(4)

memory usage: 1.1+ MB

**Number of unique values in each column:**

**Temperature**: 126

**Humidity**: 90

**Wind Speed**: 97

**Precipitation (%)**: 110

**Cloud Cover**: 4

**Atmospheric Pressure**: 5456

**UV Index**: 15

**Season**: 4

**Visibility (km)**: 41

**Location**: 3

**Weather Type**: 4

**Number of null values in each column:**

Temperature 0

Humidity 0

Wind Speed 0

Precipitation (%) 0

Cloud Cover 0

Atmospheric Pressure 0

UV Index 0

Season 0

Visibility (km) 0

Location 0

Weather Type 0

dtype: int64

**Number of duplicate rows:** 0

## Model



## Results

A white background with black text

Description automatically generated

A screenshot of a graph

Description automatically generated

# Classification Model 3(Grade Predictor)

https://www.kaggle.com/code/jamalnaseer/grade-predictor

## Dataset

**Shape of DataFrame:** (2392, 15)

**Columns in DataFrame:** ['StudentID', 'Age', 'Gender', 'Ethnicity', 'ParentalEducation', 'StudyTimeWeekly', 'Absences', 'Tutoring', 'ParentalSupport', 'Extracurricular', 'Sports', 'Music', 'Volunteering', 'GPA', 'GradeClass']

**Data types of columns:**

StudentID int64

Age int64

Gender int64

Ethnicity int64

ParentalEducation int64

StudyTimeWeekly float64

Absences int64

Tutoring int64

ParentalSupport int64

Extracurricular int64

Sports int64

Music int64

Volunteering int64

GPA float64

GradeClass float64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 2392 entries, 0 to 2391

Data columns (total 15 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 StudentID 2392 non-null int64

1 Age 2392 non-null int64

2 Gender 2392 non-null int64

3 Ethnicity 2392 non-null int64

4 ParentalEducation 2392 non-null int64

5 StudyTimeWeekly 2392 non-null float64

6 Absences 2392 non-null int64

7 Tutoring 2392 non-null int64

8 ParentalSupport 2392 non-null int64

9 Extracurricular 2392 non-null int64

10 Sports 2392 non-null int64

11 Music 2392 non-null int64

12 Volunteering 2392 non-null int64

13 GPA 2392 non-null float64

14 GradeClass 2392 non-null float64

dtypes: float64(3), int64(12)

memory usage: 280.4 KB

**Number of unique values in each column:**

**StudentID**: 2392

**Age**: 4

**Gender**: 2

**Ethnicity**: 4

**ParentalEducation**: 5

**StudyTimeWeekly**: 2392

**Absences**: 30

**Tutoring**: 2

**ParentalSupport**: 5

**Extracurricular**: 2

**Sports**: 2

**Music**: 2

**Volunteering**: 2

**GPA**: 2371

**GradeClass**: 5

**Number of null values in each column:**

StudentID 0

Age 0

Gender 0

Ethnicity 0

ParentalEducation 0

StudyTimeWeekly 0

Absences 0

Tutoring 0

ParentalSupport 0

Extracurricular 0

Sports 0

Music 0

Volunteering 0

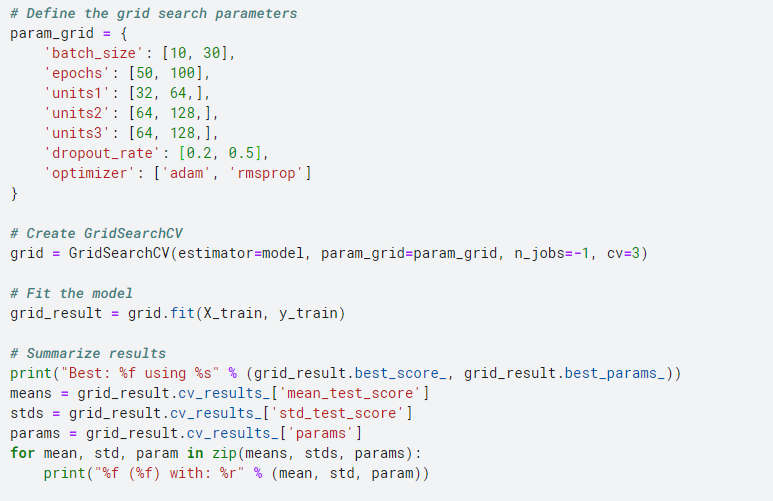
GPA 0

GradeClass 0

dtype: int64

**Number of duplicate rows:** 0

## Hyper-Parameter Tuning

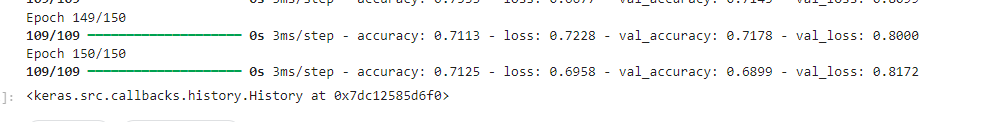


## Model

A screenshot of a computer program

Description automatically generated

## Results



A diagram of a graph

Description automatically generated with medium confidence

# Classification Model 4 (Gaming Behaviour)

https://www.kaggle.com/code/jamalnaseer/online-gaming-behaviour

## Dataset

**Shape of DataFrame:** (40034, 13)

**Columns in DataFrame:** ['PlayerID', 'Age', 'Gender', 'Location', 'GameGenre', 'PlayTimeHours', 'InGamePurchases', 'GameDifficulty', 'SessionsPerWeek', 'AvgSessionDurationMinutes', 'PlayerLevel', 'AchievementsUnlocked', 'EngagementLevel']

**Data types of columns:**

PlayerID int64

Age int64

Gender object

Location object

GameGenre object

PlayTimeHours float64

InGamePurchases int64

GameDifficulty object

SessionsPerWeek int64

AvgSessionDurationMinutes int64

PlayerLevel int64

AchievementsUnlocked int64

EngagementLevel object

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 40034 entries, 0 to 40033

Data columns (total 13 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 PlayerID 40034 non-null int64

1 Age 40034 non-null int64

2 Gender 40034 non-null object

3 Location 40034 non-null object

4 GameGenre 40034 non-null object

5 PlayTimeHours 40034 non-null float64

6 InGamePurchases 40034 non-null int64

7 GameDifficulty 40034 non-null object

8 SessionsPerWeek 40034 non-null int64

9 AvgSessionDurationMinutes 40034 non-null int64

10 PlayerLevel 40034 non-null int64

11 AchievementsUnlocked 40034 non-null int64

12 EngagementLevel 40034 non-null object

dtypes: float64(1), int64(7), object(5)

memory usage: 4.0+ MB

**Number of unique values in each column:**

**PlayerID**: 40034

**Age**: 35

**Gender**: 2

**Location**: 4

**GameGenre**: 5

**PlayTimeHours**: 40034

**InGamePurchases**: 2

**GameDifficulty**: 3

**SessionsPerWeek**: 20

**AvgSessionDurationMinutes**: 170

**PlayerLevel**: 99

**AchievementsUnlocked**: 50

**EngagementLevel**: 3

**Number of null values in each column:**

PlayerID 0

Age 0

Gender 0

Location 0

GameGenre 0

PlayTimeHours 0

InGamePurchases 0

GameDifficulty 0

SessionsPerWeek 0

AvgSessionDurationMinutes 0

PlayerLevel 0

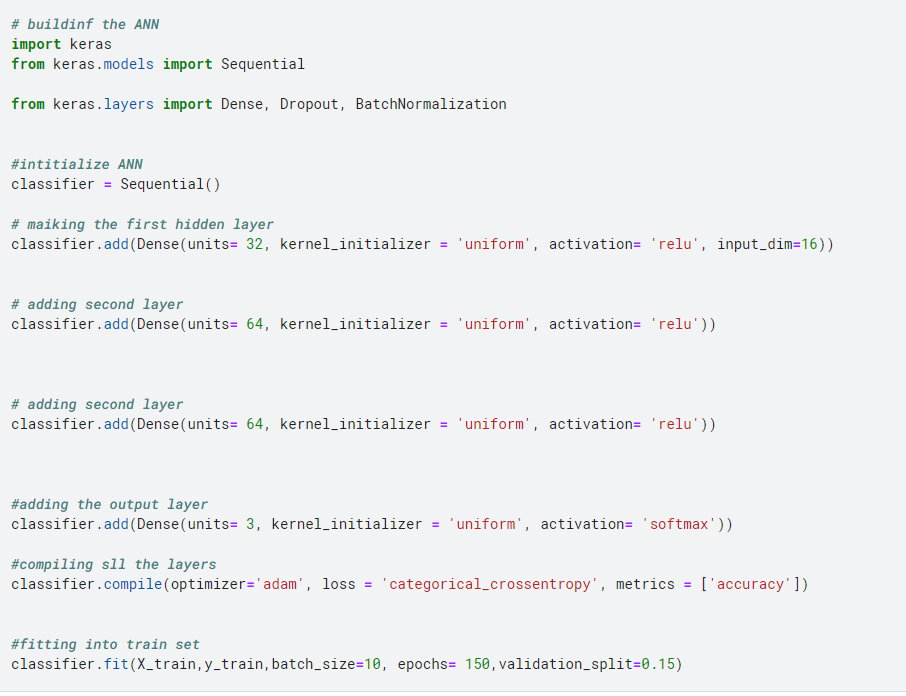
AchievementsUnlocked 0

EngagementLevel 0

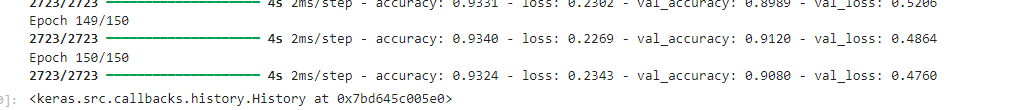
dtype: int64

**Number of duplicate rows:** 0

## Model



## Results



# Classification Model 5(Customer Behaviour)

https://www.kaggle.com/code/jamalnaseer/customer-behavior-model

## Dataset

**Shape of DataFrame:** (1500, 9)

**Columns in DataFrame:** ['Age', 'Gender', 'AnnualIncome', 'NumberOfPurchases', 'ProductCategory', 'TimeSpentOnWebsite', 'LoyaltyProgram', 'DiscountsAvailed', 'PurchaseStatus']

**Data types of columns:**

Age int64

Gender int64

AnnualIncome float64

NumberOfPurchases int64

ProductCategory int64

TimeSpentOnWebsite float64

LoyaltyProgram int64

DiscountsAvailed int64

PurchaseStatus int64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1500 entries, 0 to 1499

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Age 1500 non-null int64

1 Gender 1500 non-null int64

2 AnnualIncome 1500 non-null float64

3 NumberOfPurchases 1500 non-null int64

4 ProductCategory 1500 non-null int64

5 TimeSpentOnWebsite 1500 non-null float64

6 LoyaltyProgram 1500 non-null int64

7 DiscountsAvailed 1500 non-null int64

8 PurchaseStatus 1500 non-null int64

dtypes: float64(2), int64(7)

memory usage: 105.6 KB

**Number of unique values in each column:**

**Age**: 53

**Gender**: 2

**AnnualIncome**: 1388

**NumberOfPurchases**: 21

**ProductCategory**: 5

**TimeSpentOnWebsite**: 1388

**LoyaltyProgram**: 2

**DiscountsAvailed**: 6

**PurchaseStatus**: 2

**Number of null values in each column:**

Age 0

Gender 0

AnnualIncome 0

NumberOfPurchases 0

ProductCategory 0

TimeSpentOnWebsite 0

LoyaltyProgram 0

DiscountsAvailed 0

PurchaseStatus 0

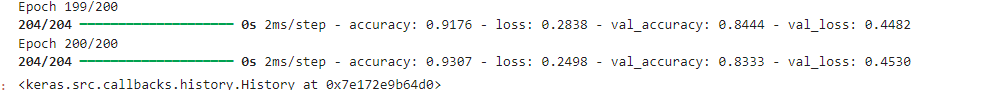
dtype: int64

**Number of duplicate rows:** 112

## Model



## Results



A blue squares with white numbers and black text

Description automatically generated

# Classification Model 6 (Hiring Model)

https://www.kaggle.com/code/jamalnaseer/hiring-decision-predictor

## Dataset

**Shape of DataFrame:** (1500, 11)

**Columns in DataFrame:** ['Age', 'Gender', 'EducationLevel', 'ExperienceYears', 'PreviousCompanies', 'DistanceFromCompany', 'InterviewScore', 'SkillScore', 'PersonalityScore', 'RecruitmentStrategy', 'HiringDecision']

**Data types of columns:**

Age int64

Gender int64

EducationLevel int64

ExperienceYears int64

PreviousCompanies int64

DistanceFromCompany float64

InterviewScore int64

SkillScore int64

PersonalityScore int64

RecruitmentStrategy int64

HiringDecision int64

dtype: object

**Information about DataFrame:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1500 entries, 0 to 1499

Data columns (total 11 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Age 1500 non-null int64

1 Gender 1500 non-null int64

2 EducationLevel 1500 non-null int64

3 ExperienceYears 1500 non-null int64

4 PreviousCompanies 1500 non-null int64

5 DistanceFromCompany 1500 non-null float64

6 InterviewScore 1500 non-null int64

7 SkillScore 1500 non-null int64

8 PersonalityScore 1500 non-null int64

9 RecruitmentStrategy 1500 non-null int64

10 HiringDecision 1500 non-null int64

dtypes: float64(1), int64(10)

memory usage: 129.0 KB

**Number of unique values in each column:**

**Age**: 31

**Gender**: 2

**EducationLevel**: 4

**ExperienceYears**: 16

**PreviousCompanies**: 5

**DistanceFromCompany**: 1500

**InterviewScore**: 101

**SkillScore**: 101

**PersonalityScore**: 101

**RecruitmentStrategy**: 3

**HiringDecision**: 2

**Number of null values in each column:**

Age 0

Gender 0

EducationLevel 0

ExperienceYears 0

PreviousCompanies 0

DistanceFromCompany 0

InterviewScore 0

SkillScore 0

PersonalityScore 0

RecruitmentStrategy 0

HiringDecision 0

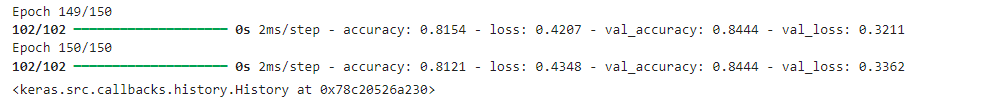
dtype: int64

**Number of duplicate rows:** 0

## Model



## Results



A blue squares with numbers and numbers

Description automatically generated