Lista de exercícios 02

Aprendizado profundo

João Honorato
Maria Raquel
Samila Garrido

QUESTÃO 01

Titanic

Dadas características dos passageiros e a informação de quem morreu ou não, é possível prever a sobrevivência de alguém?

Dados

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

SibSp: número de irmãos/cônjuges a bordo

Parch: número de pais/filhos a bordo do Titanic

Embarked: porto de embarque, Cherbourg, Queenstown ou Southampton

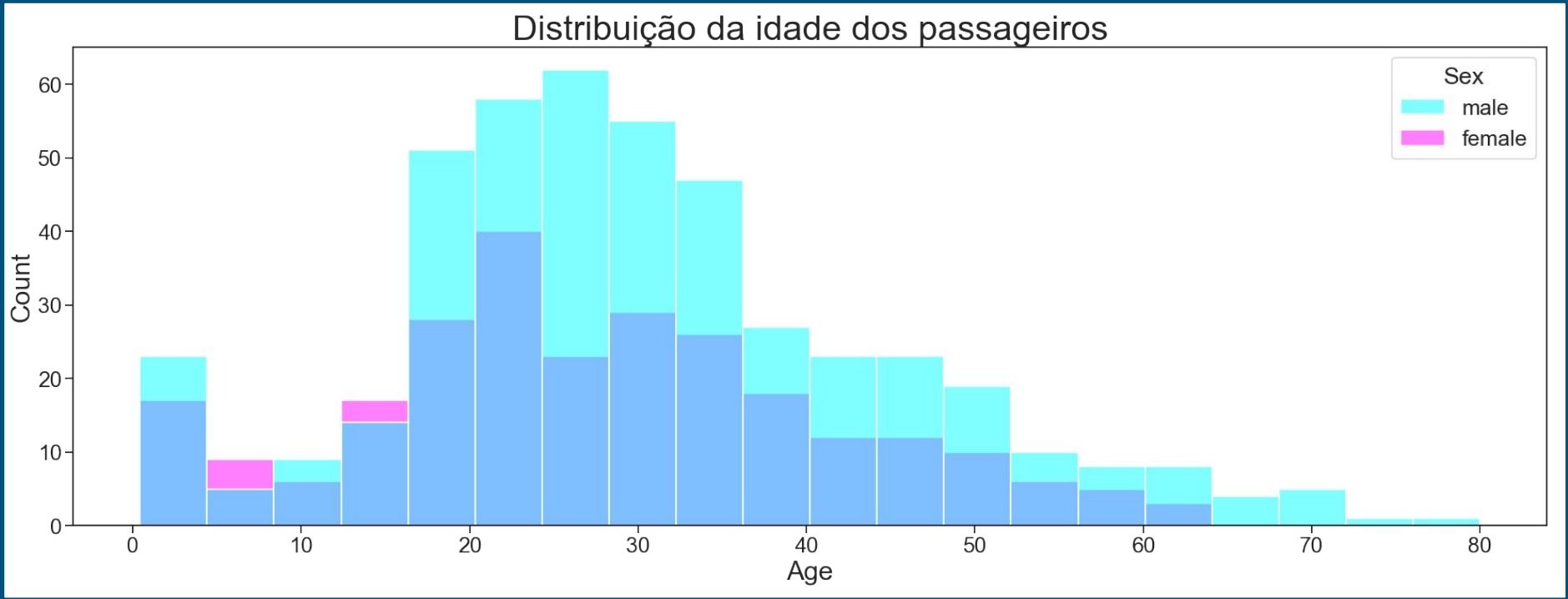
Tratamento: valores nulos

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

Cabin: Removemos a coluna

Embarked: Removemos as instâncias

Variável idade:



Variável idade:

```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

Tratamento de tipo

```
Survived      int64
Pclass        int64
Sex          object
Age         float64
SibSp         int64
Parch         int64
Fare        float64
Embarked     object
dtype: object
```

Sex: Male = 0

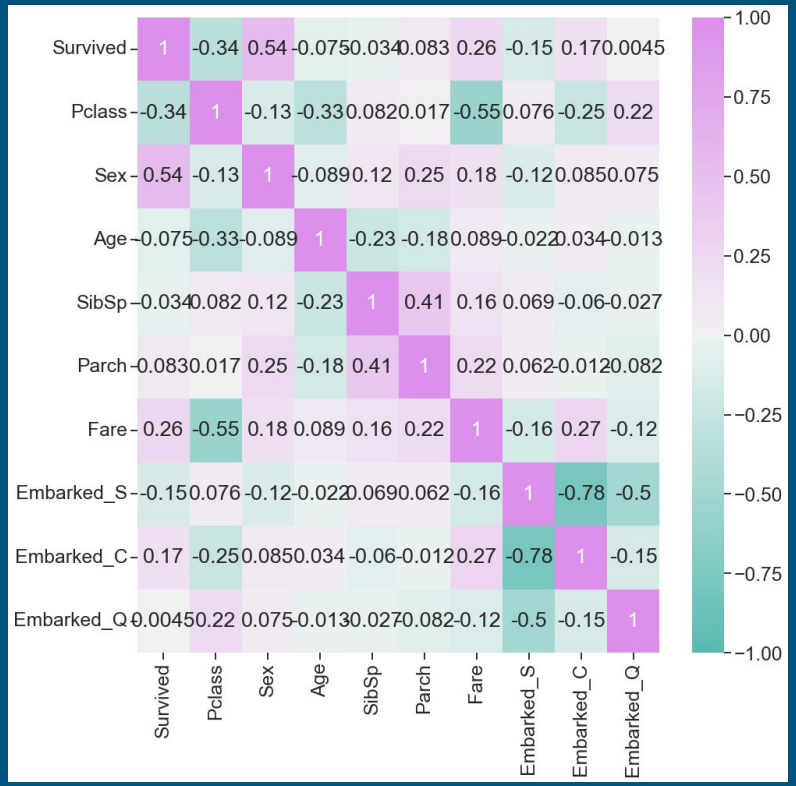
Female = 1

Embarked: Q = 0

S = 1

C = 2

Análise de correlação



Normalização

```
count    889.000000
mean       0.062649
std        0.097003
min        0.000000
25%        0.015412
50%        0.028213
75%        0.060508
max        1.000000
Name: Fare, dtype: float64
```

```
count    889.000000
mean      32.096681
std       49.697504
min        0.000000
25%        7.895800
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

Arquitetura

3 neurônios de entrada

6 neurônios na camada oculta

1 neurônio de saída

```
class TitanicNN(nn.Module):
    def __init__(self,n_in=3,n_hid=6,n_out=1):
        super(TitanicNN,self).__init__()
        self.n_in  = n_in
        self.n_hid = n_hid
        self.n_out = n_out

        self.linear_layer =  nn.Sequential(nn.Linear(n_in,n_hid),
                             nn.ReLU(),
                             nn.Linear(n_hid,n_out),
                             nn.Sigmoid())

    def forward(self,x):
        x = self.linear_layer(x)
        return x
```

Treinamento

Validação: 20% do teste

Batch size: 64

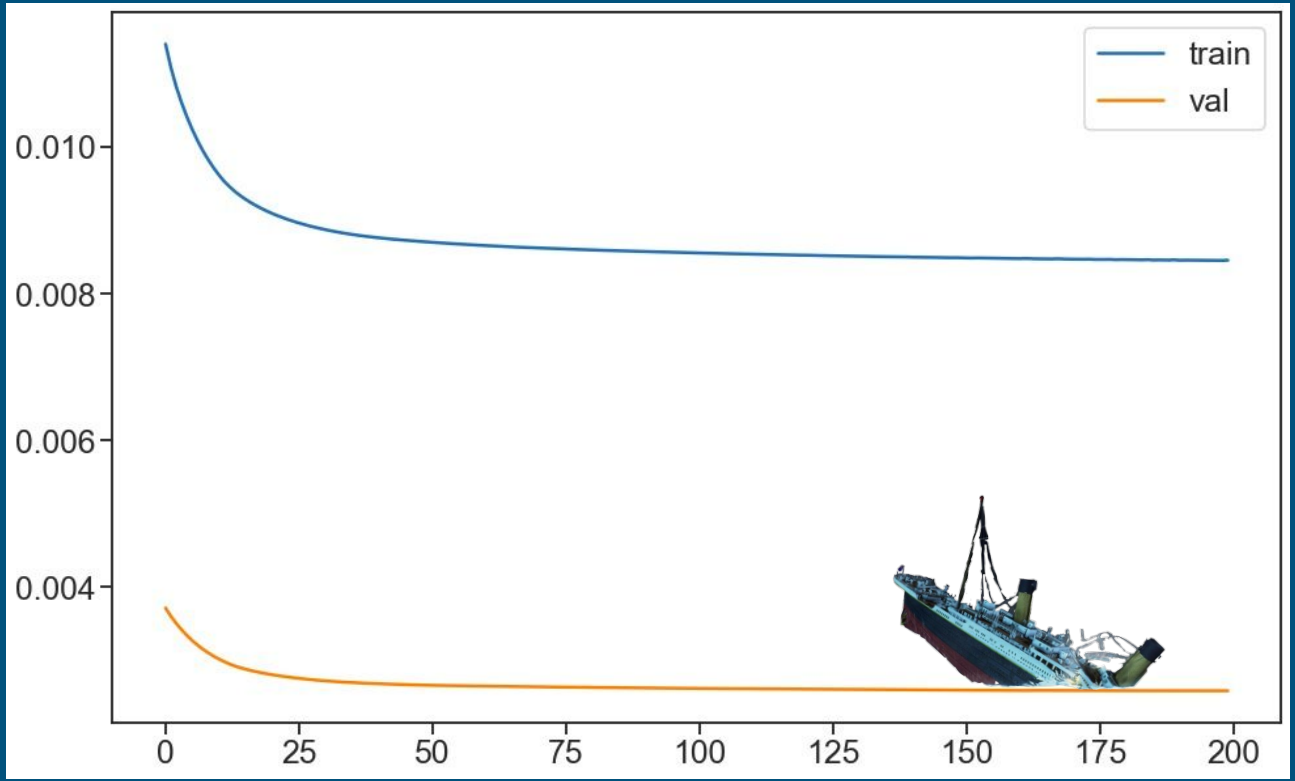
Épocas: 200

Função de erro: BCE

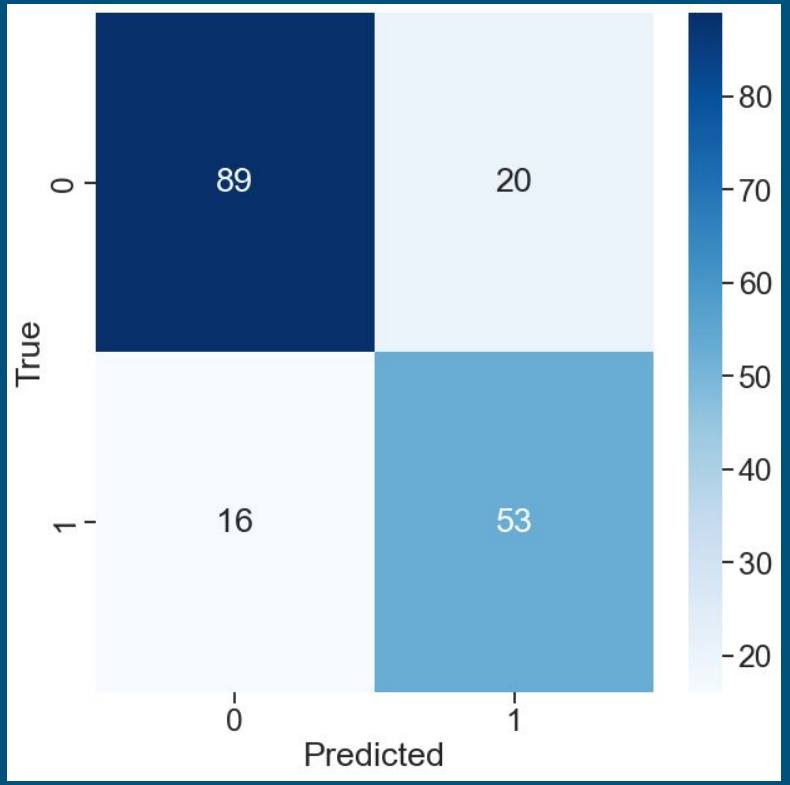
Otimizador: Adam

LR: 0.005

Resultados



Treinamento



QUESTÃO 02

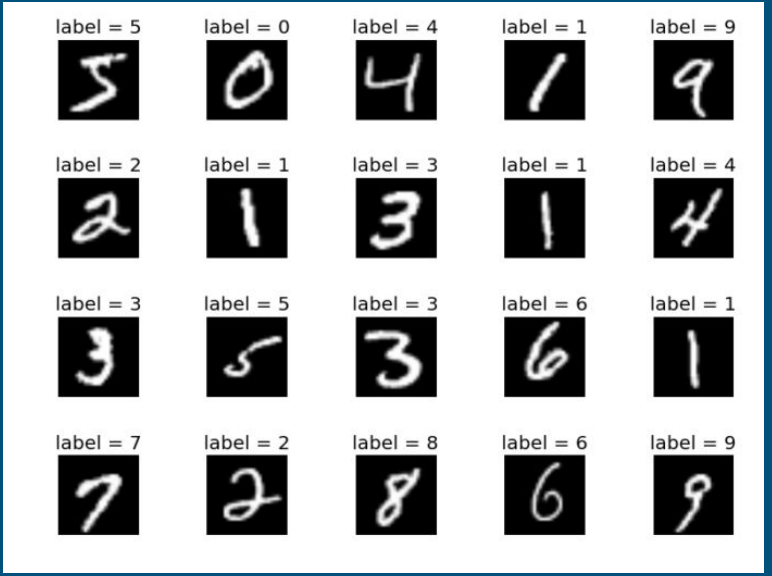
Questão 02

Implementar duas redes neurais artificiais (MLP e CNN) para o problema de classificação de dígitos escritos à mão utilizando a base de dados do MNIST.

Dados:

- 60 mil imagens de **treino**
- 10 mil imagens de **teste**

10 classes: **0, 1, 2, 3, 4, 5, 6, 7, 8, 9**



Amostra dos dados de entrada

Questão 02

Normalização

Dados com valores entre 0 e 1

```
#normalização
x_train = np.array(x_train).reshape(-1,784).astype('float32') / 255.0
x_test =  np.array(x_test).reshape(-1,784).astype('float32') / 255.0
```

One-hot encoding

De categórico para binário

```
def one_hot(true_labels,num_classes):
    labels = keras.utils.to_categorical(true_labels, num_classes)
    return labels

y_train_one_hot = one_hot(y_train, 10)
y_test_one_hot = one_hot(y_test, 10)
```

MLP

Neurônios:

- Layer 1: 10 neurônios
- Layer 2: 40 neurônios
- Layer 3: 10 neurônios
- Layer de saída: 10 neurônios

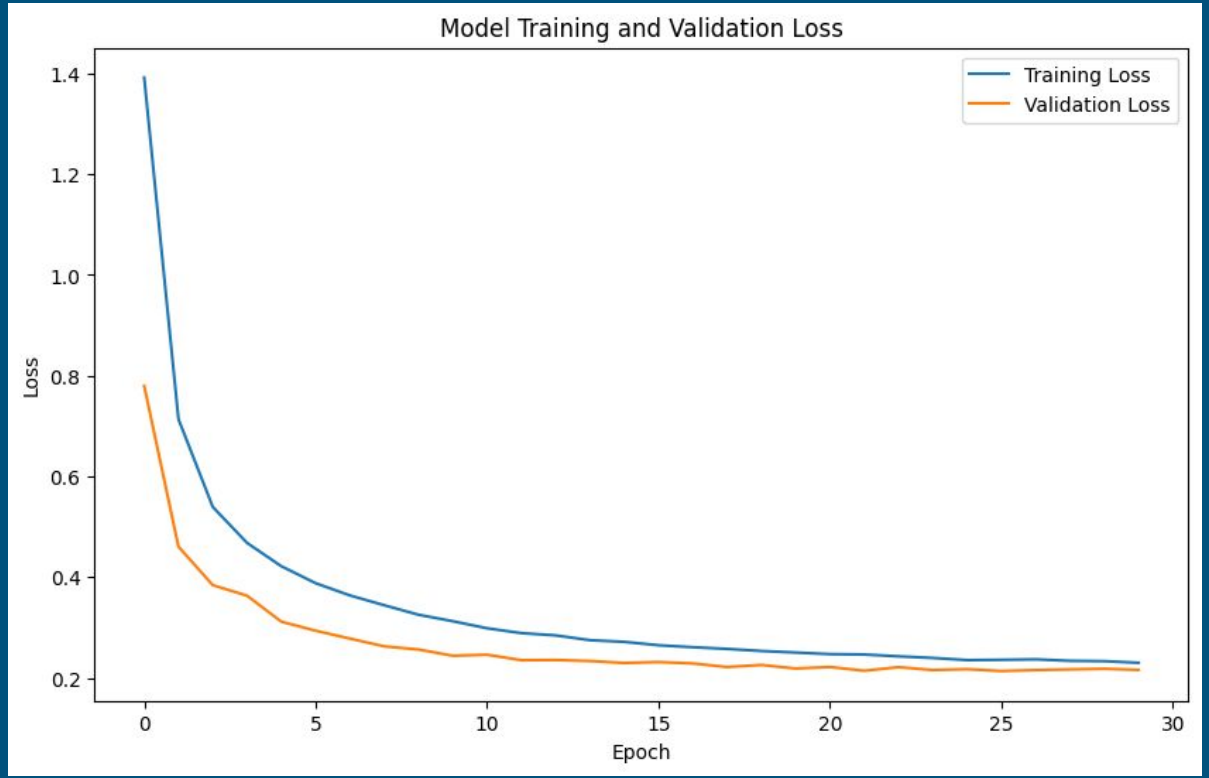
Número de camadas: 4

Funções de Ativação:

- Layer 1: ReLU
- Layer 2: Tangente Hiperbólica
- Layer 3: ReLU
- Layer de saída: Softmax
- Otimizador: SGD
- Função de Custo: Categorical Crossentropy
- Número de épocas: 30

Questão 02 - MLP

Curva de erro médio

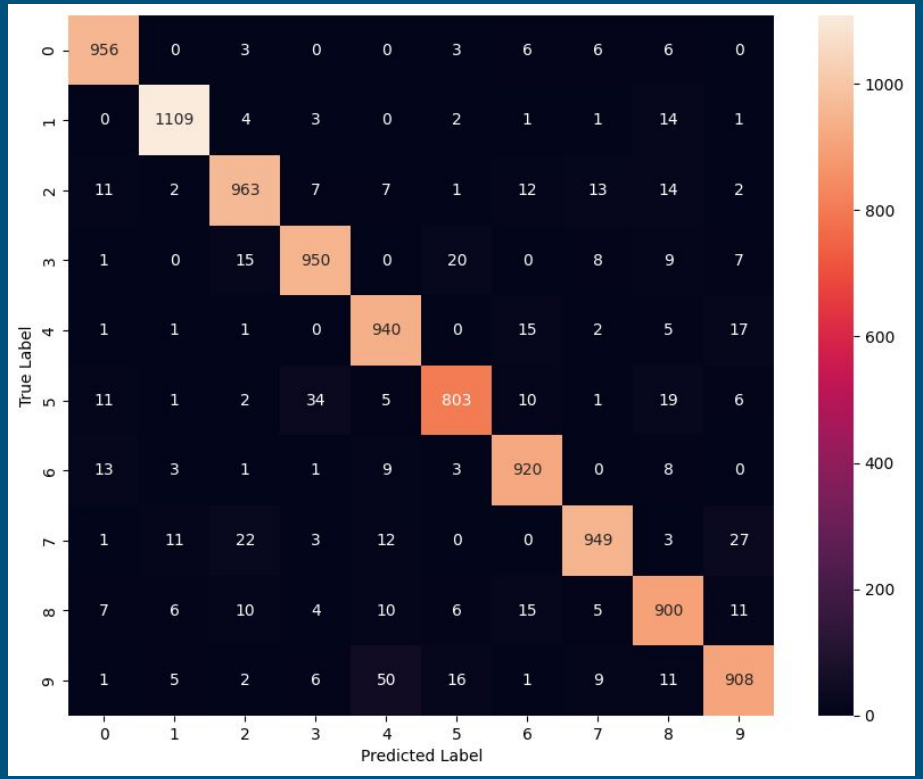


Curva de erro médio por época para treino e validação

Acurácia: 0.93980

Questão 02 - MLP

Matriz de Confusão



CNN

Neurônios:

- **Layer 1: 32 neurônios**
- **Layer 2: 64 neurônios**
- **Layer 3: 128 neurônios**
- **Layer de saída: 10 neurônios**

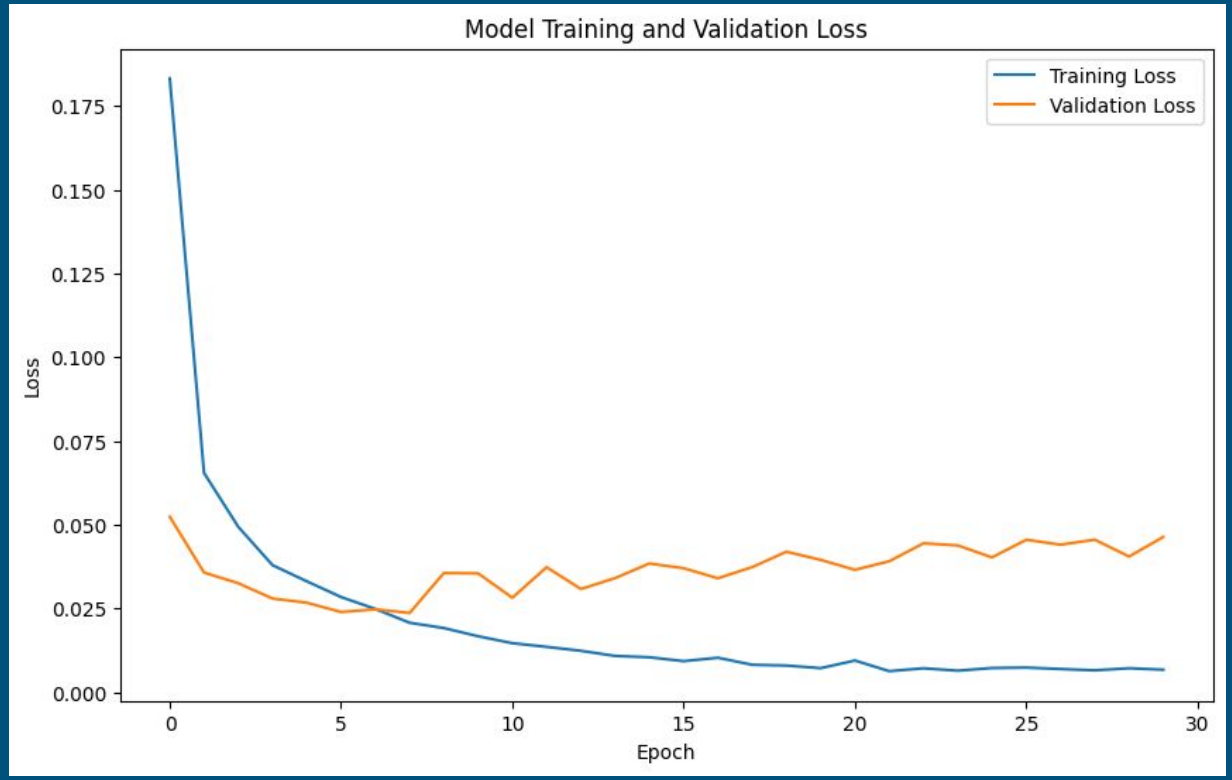
Número de camadas: 4

Funções de Ativação:

- **Layer 1: ReLU**
- **Layer 2: ReLU**
- **Layer 3: Dropout (0.5)**
- **Layer de saída: Softmax**
- **Otimizador: Adam**
- **Função de Custo: CrossentropyLoss**
- **Número de épocas: 30**

Questão 02 - CNN

Curva de erro médio

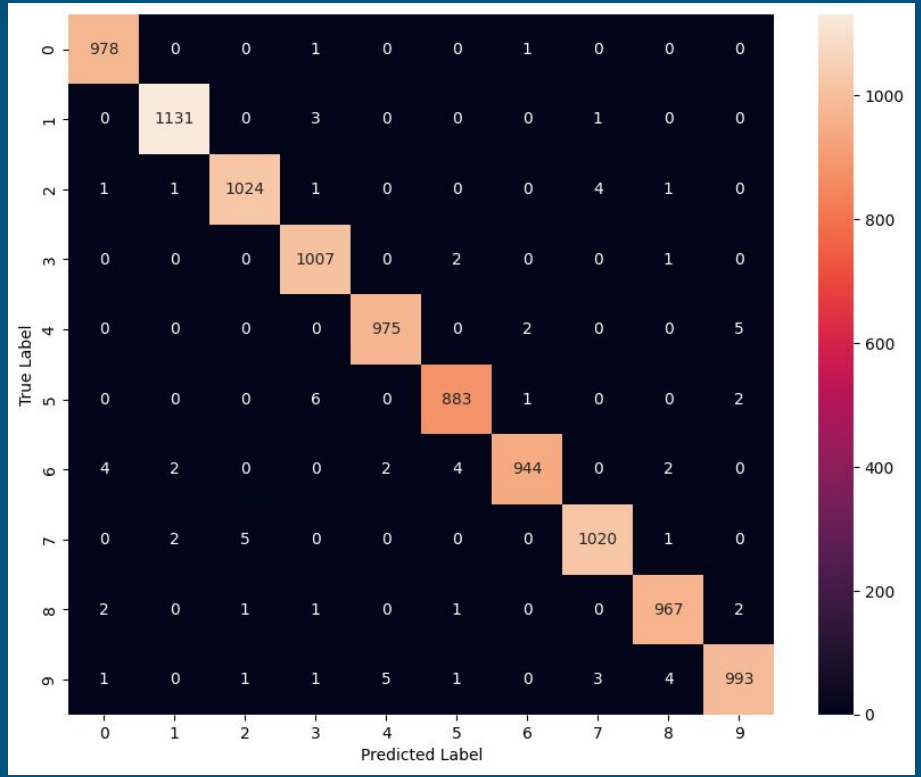


Curva de erro médio por época para treino e validação

Acurácia: 0.9922

Questão 02 - CNN

Matriz de Confusão



QUESTÃO 03

Questão 03

Rede Neural Convolucional para classificação de imagens do dataset **CIFAR-10**

60 mil imagens RGB 32x32 pixels:

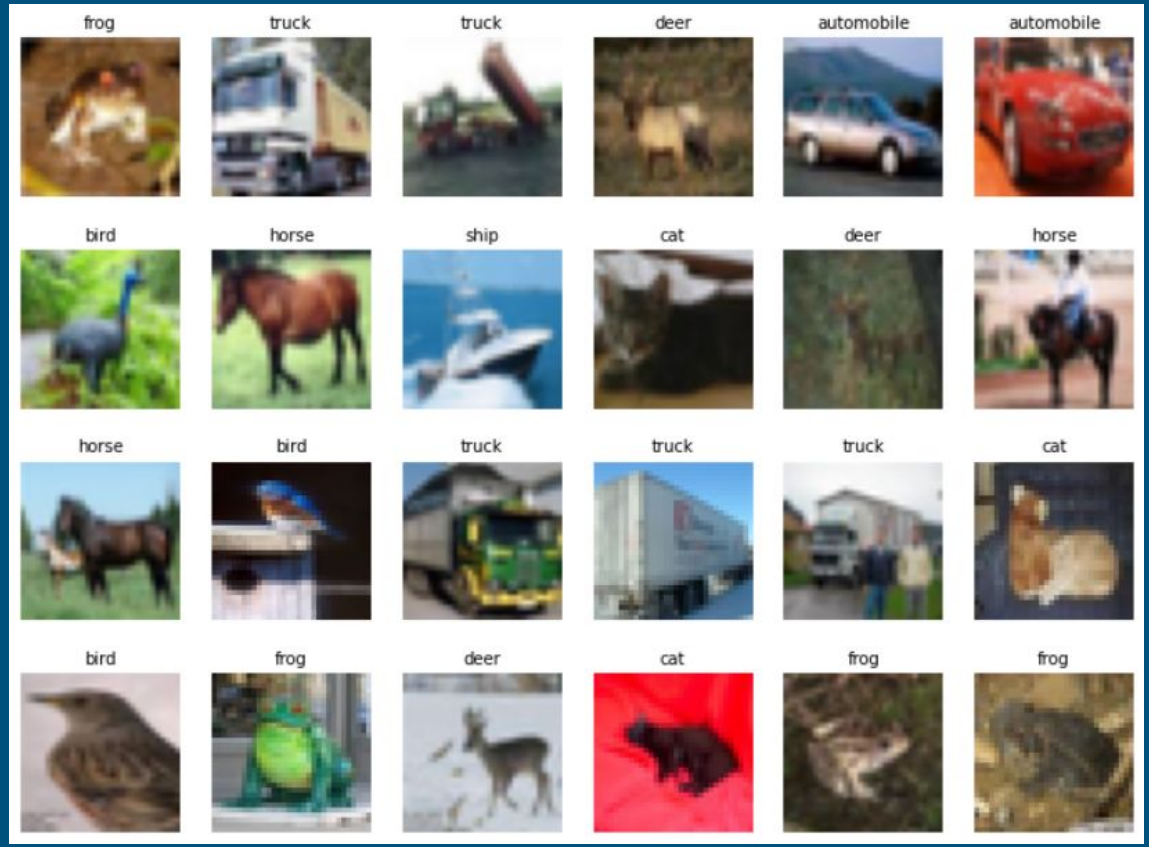
- 40 mil imagens de **treino**
- 10 mil imagens de **validação**
- 10 mil imagens de **teste**

10 grupos:

airplane	**automobile**
bird	**cat**
deer	**dog**
frog	**horse**
ship	**truck**

Questão 03

Alguns exemplos de imagens e rótulos do dataset CIFAR-10



Questão 03

Manipulação dos dados

Normalização

Dados com valores entre 0 e 1

One-hot encoding

De categórico para binário

Validação

Avaliar antes de testar

```
## Os valores dos componentes de cada pixel RGB variam de 0 a 255
## Vamos normalizar os valores das componentes dos pixels
x_train = x_train/255
x_test = x_test/255

## Transformando os labels em one-hot encoding
y_train_en = to_categorical(y_train, 10)
y_test_en = to_categorical(y_test, 10)

## Fazendo um recorte de dados para treino e validação
x_val = x_train[-10000:]
y_val_en = y_train_en[-10000:]
x_train = x_train[:-10000]
y_train_en = y_train_en[:-10000]
```

Questão 03

Camadas do modelo

Conv2D

Duas camadas convolucionais com 64 filtros

MaxPooling

Redução das dimensões dos mapas de características

Dropout

Desativação temporária de neurônios

```
## Instanciando o modelo sequencial
model = Sequential()

## Adicionando camadas ao modelo
## 1) Conv2D: Duas camadas com 64 filtros de 4x4 e função de ativação ReLU
## Gerará 64 mapas de características em cada camada,
## cada um com 28x28 pixels (tamanho da imagem após a aplicação dos filtros)
model.add(Conv2D(64,(4,4),input_shape=(32,32,3),activation='relu'))
model.add(Conv2D(64,(4,4),input_shape=(32,32,3),activation='relu'))

## 2) MaxPooling2D: reduz as dimensões da saída da camada
## transformando cada 2x2 pixels em 1 pixel
model.add(MaxPooling2D(pool_size=(2,2)))

## 3) Dropout: desativa aleatoriamente 40%
## dos neurônios da camada para evitar overfitting
model.add(Dropout(0.4))
```

Questão 03

Camadas do modelo

Conv2D

Duas camadas convolucionais com 128 filtros

MaxPooling

Redução das dimensões dos mapas de características

Dropout

Desativação temporária de neurônios

```
## 4) Conv2D: Duas camadas com 128 filtros de 4x4 e função de ativação ReLU
model.add(Conv2D(128,(4,4),input_shape=(32,32,3),activation='relu'))
model.add(Conv2D(128,(4,4),input_shape=(32,32,3),activation='relu'))

## 5) MaxPooling2D: reduz novamente as dimensões da saída da camada
model.add(MaxPooling2D(pool_size=(2,2)))

## 6) Dropout: desativa aleatoriamente 40%
## dos neurônios da camada para evitar overfitting
model.add(Dropout(0.4))
```

Questão 03

Camadas do modelo

Flatten

Matriz —> vetor 1d

Camadas densamente conectadas

Duas camadas com 1024 neurônios e função de ativação relu

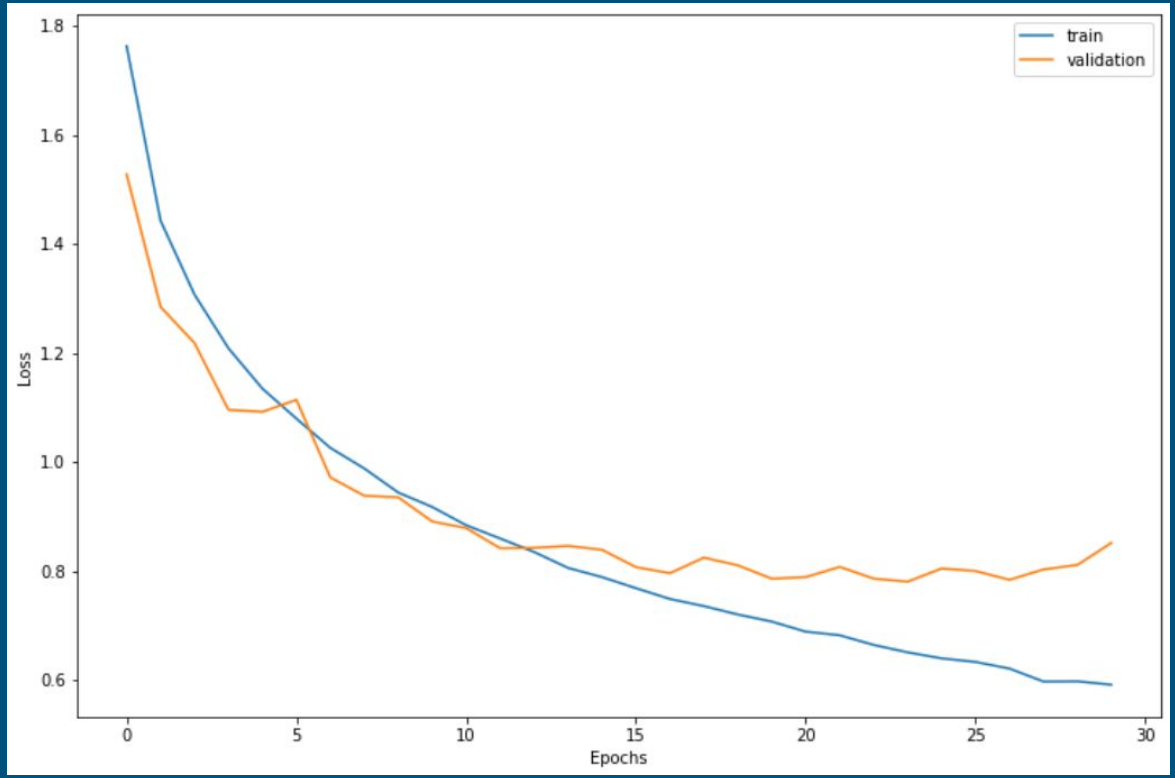
Saída

10 neurônios com função de ativação softmax

```
## 7) Flatten: transforma a matriz em um vetor 1D
model.add(Flatten())

## 8) Duas camadas de 1024 neurônios com função de ativação ReLU
model.add(Dense(1024,activation='relu'))
model.add(Dense(1024,activation='relu'))

## 9) 10 neurônios com função de ativação softmax
model.add(Dense(units = 10, activation = 'softmax'))

## Compilando o modelo:
    ## loss: função de erro categorical_crossentropy
    ## optimizer: algoritmo de otimização Adam
    ## metrics: métrica de avaliação accuracy
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Questão 03

Curva de erro médio

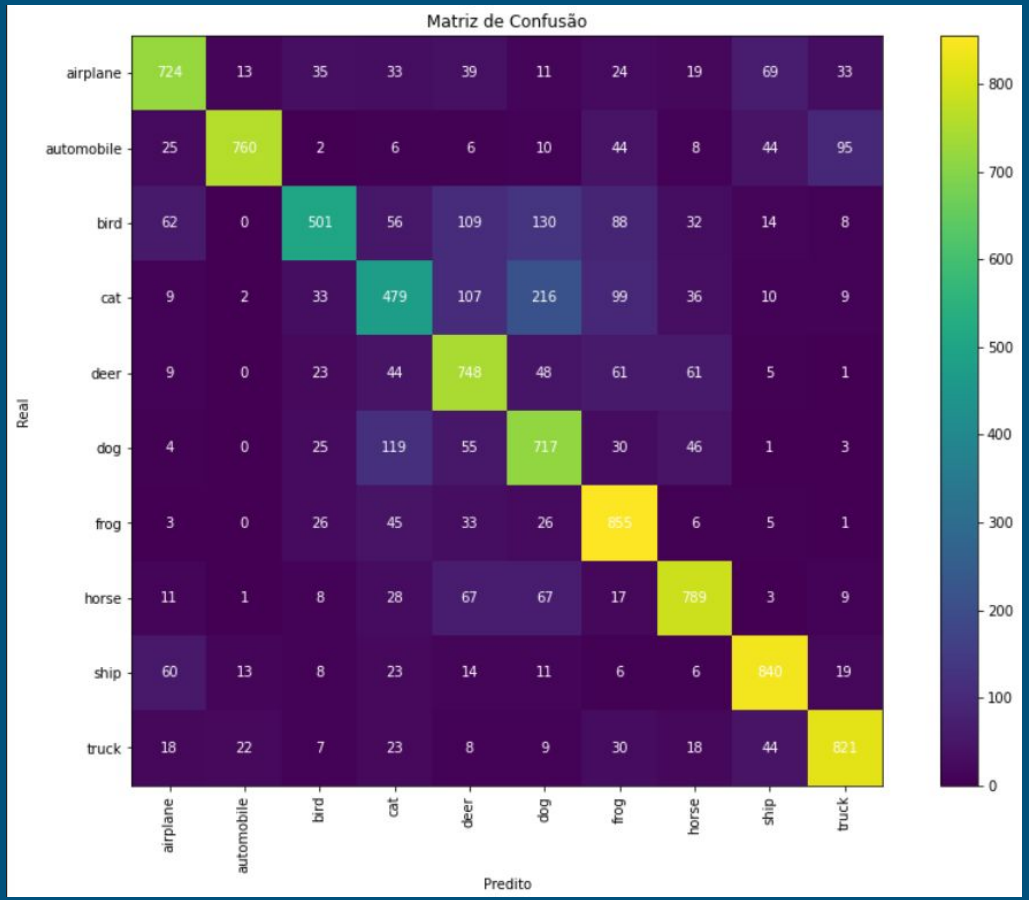


Curva de erro médio por época para treino e validação

Questão 03

Matriz de confusão

Matriz de confusão dos dados de teste



Questão 03

Classificação

Questão 03

Classificação