Expected values, Surprise, Entropy, Cross-entropy loss, Softmax e True labels (one-hot encoded)

Instruções (requisitos) 3-1

Fazer cada atividade no Jupyter (Collab). Poste apenas os links.

Crie um novo repositório para a segunda avaliação. Coloque no repositório esta atividade e a Atividade 2.1. O repositório deve ser privado. Coloque-me como colaborador.

Para cada parte da atividade, colocar como figuras os slides do comando do problema antes de cada solução.

Instruções (requisitos) 3-2

Quando eu falar “faça a análise os resultados”, a análise deve estar em um célula de texto. Sem a análise detalhada, a tarefa não será avaliada. Na verdade, eu não preciso falar nada: cada parte da atividade sempre tem de ter análises de resultados e conclusão.

Instruções (requisitos) 3-3

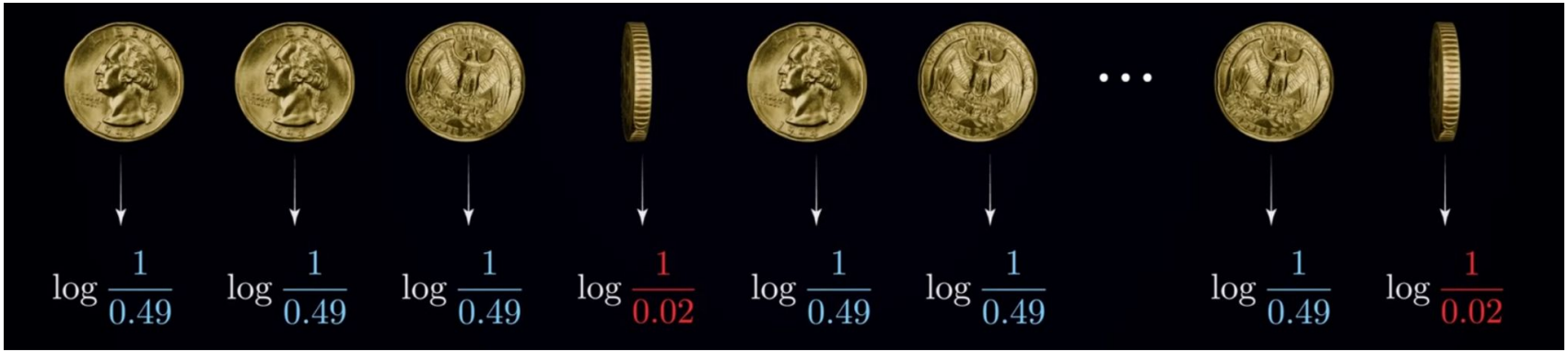
Caso algum requisito esteja faltando, sua tarefa não será analisada.

A - surpresa

Parte 4 - Surprise.pptx

Simule três moedas:

- Uma moeda justa
- Uma moeda 90/10
- A moeda do slide a seguir



$s = 0.95$

✓	✓	✓	✓	✓	✓	✓	✓	✗	✓
0.55	0.72	0.60	0.54	0.42	0.65	0.44	0.89	0.96	0.38

9

Para cada moeda:

- Execute a simulação 10, 100 e 1000 vezes
- Calcule a surpresa média
- Calcule usando a fórmula

Em um único gráfico de barras, coloque os valores teóricos, e os valores obtidos das simulações para todos os valores de repetição e para todas as moedas. Veja a melhor estratégia de exposição dos resultados. Analise os resultados.

B - entropia

Considere o cenário hipotético no qual as features e os parâmetros da rede neural não são apresentados, mas apenas a sua saída na forma de *raw logits* para um *3-class classification problem*, como no exemplo abaixo:

```
[[2.5, 0.3, 2.1],
 [1.2, 2.4, 0.1],
 [0.8, 0.5, 3.0]]
```

São usadas apenas três amostradas para treinar a rede. A primeira amostra da Classe 1 (1 0 0), a segunda da Classe 2 (0 1 0) e a terceira da Classe 3 (0 0 1)

True labels (one-hot encoded):

```
[[1, 0, 0],
 [0, 1, 0],
 [0, 0, 1]]
```

No seu programa será simulada iterativamente apenas a saída da rede na forma *raw logits*, sem considerar os seus parâmetros e as features. A primeira saída será aleatória no formato mostrado abaixo, com valores entre 0.1 a 3.0:

```
[[2.5, 0.3, 2.1],
 [1.2, 2.4, 0.1],
 [0.8, 0.5, 3.0]]
```

Use a semente gerada a partir do relógio do sistema

A partir desta saída, aplique a função softmax para gerar probabilidades

```
[[2.5, 0.3, 2.1],
 [1.2, 2.4, 0.1],
 [0.8, 0.5, 3.0]]
```

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

Inclua no trabalho alguns slides sobre a função softmax e a versão que você usar da função softmax

```python
# Softmax function
def softmax(logits):
    exp_logits = np.exp(logits - np.max(logits))  # Stability adjustment
    return exp_logits / np.sum(exp_logits, axis=-1, keepdims=True)
```

Nota: veja que cada linha na matriz se refere a uma das três amostras, e cada coluna a uma das classes

```
[[2.5, 0.3, 2.1],
 [1.2, 2.4, 0.1],
 [0.8, 0.5, 3.0]]
```

$$\begin{bmatrix} 0.5614 & 0.0622 & 0.3763 \\ 0.2149 & 0.7135 & 0.0715 \\ 0.0929 & 0.0688 & 0.8383 \end{bmatrix}$$

A partir das probabilidades geradas, calcule a *cross-entropy loss*

Compute the cross-entropy loss for a multi-class classification problem:

$$\text{Cross-Entropy Loss} = -\sum_{i=1}^{C} y_i \cdot \log(\hat{y}_i)$$

A partir das probabilidades geradas, calcule a *cross-entropy loss*

Compute the cross-entropy loss for a multi-class classification problem:

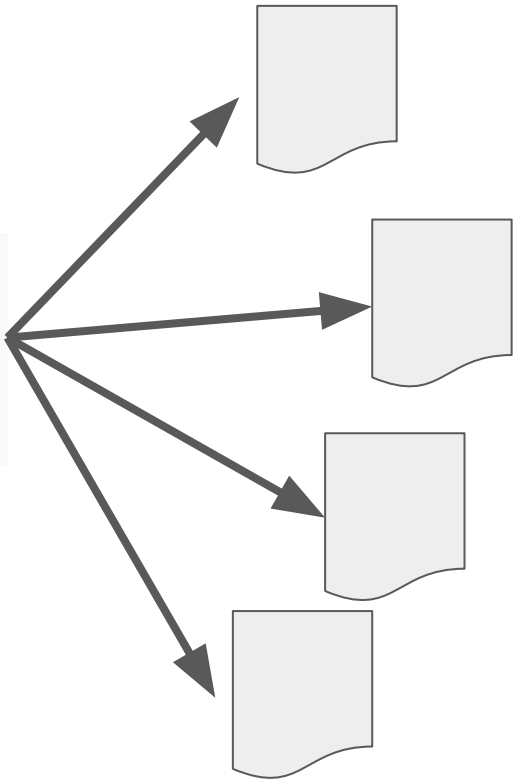
$$\text{Cross-Entropy Loss} = -\sum_{i=1}^{C} y_i \cdot \log(\hat{y}_i)$$

usando the true labels (one-hot encoded):

```
[[1, 0, 0],
 [0, 1, 0],
 [0, 0, 1]]
```

Agora começa o processo iterativo:

```
[[2.5, 0.3, 2.1],
 [1.2, 2.4, 0.1],
 [0.8, 0.5, 3.0]]
```



Usando uma distribuição Gaussiana cuja a média é cada um dos $raw\ logits$ originais e com o desvio padrão sendo um hiperparâmetro, gere outras quatro sequências.

Em cada uma delas aplique novamente a softmax e calcule a *cross-entropy loss*. Ordene as sequências, incluindo a original, de acordo com a entropia média, e descarte as quatro piores. Repita o processo 30 vezes. Mostre a evolução da entropia com o número de iterações.

C - Naive Bayes and Spam

Transforme o exemplo que inicia no slide 55 em código, plotando os histogramas, os cálculos de probabilidades e o resultado da classificação. Organize as probabilidades em tabelas usando Pandas.

Naive Bayes.pptx

Ao final, crie uma função na qual seja dada uma sequência como mostrada abaixo e ela retorne o resultado da classificação.

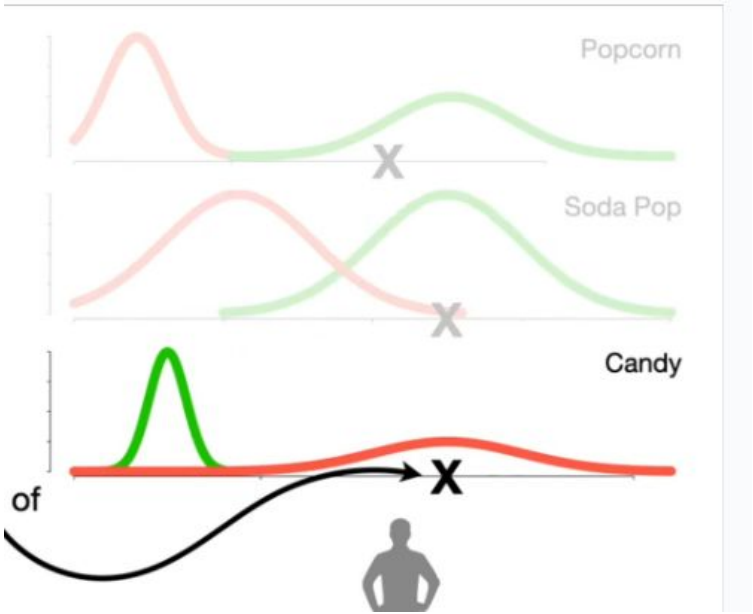
Lunch Money Money Money Money

D - Gauss Naive Bayes

Transforme o exemplo que inicia no slide 7 em código. Crie as tabelas para as três features a partir das distribuições reais usando médias e os desvios padrão dados nos slides. Plote essas distribuições em gráficos. No entanto, use os likelihoods dados nos exemplos para exemplificar em código a classificação.

Gaussian Naive Bayes.pptx

Ao final, crie uma função na qual seja dada uma valores como na figura e ela retorne a classificação.



E - Naive Bayes and Iris dataset/Acurácia

Apply Naive Bayes to the Iris Dataset

Da mesma forma que foi exemplificado no slide 208, aplique Naives Bayes para o dataset Iris: from sklearn.datasets import load_iris. Pesquise sobre a métrica de avaliação acurácia.

Naive Bayes.pptx

Iris - UCI Machine Learning Repository

Use o

```
from sklearn.metrics import classification_report
```

na análise dos resultados.

F - Simule os slides

Parte 3 - Expected Values.pptx

Valores esperados

Crie uma função que eu entre com os dados da tabela do slide 26 e os outcomes do slide 51, e o retorno da função seja o valor esperado.

Faça a simulação deste experimento e compare com os valores esperados teóricos. Faça o experimento para 10, 100, 1000 e 10000 repetições.