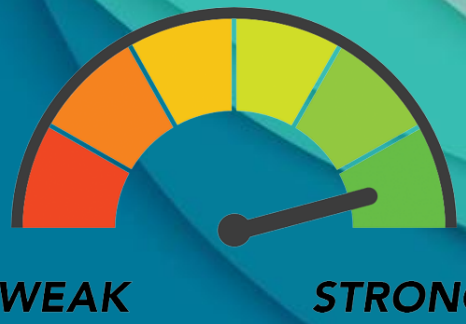
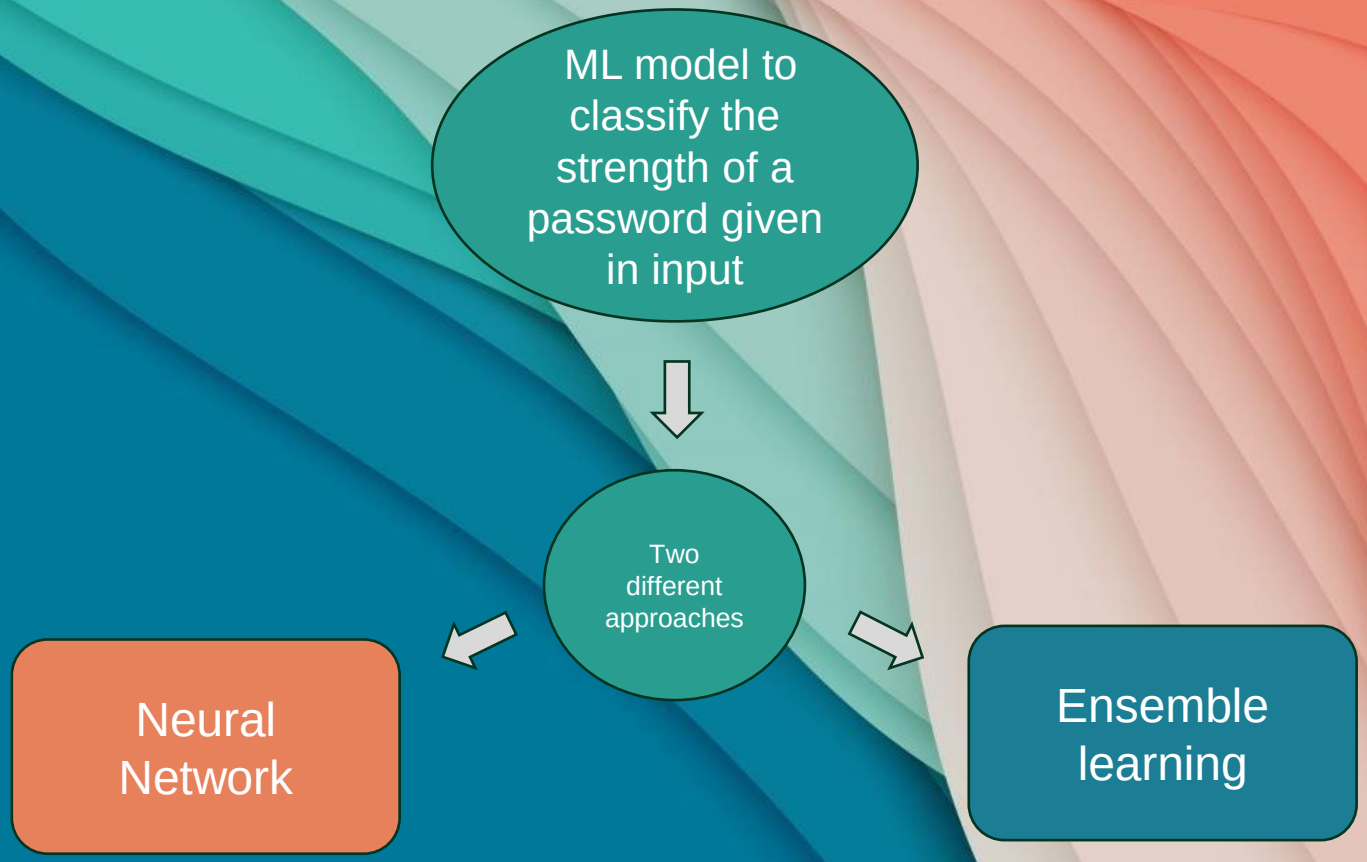


PASSWORD STRENGTH CLASSIFIER

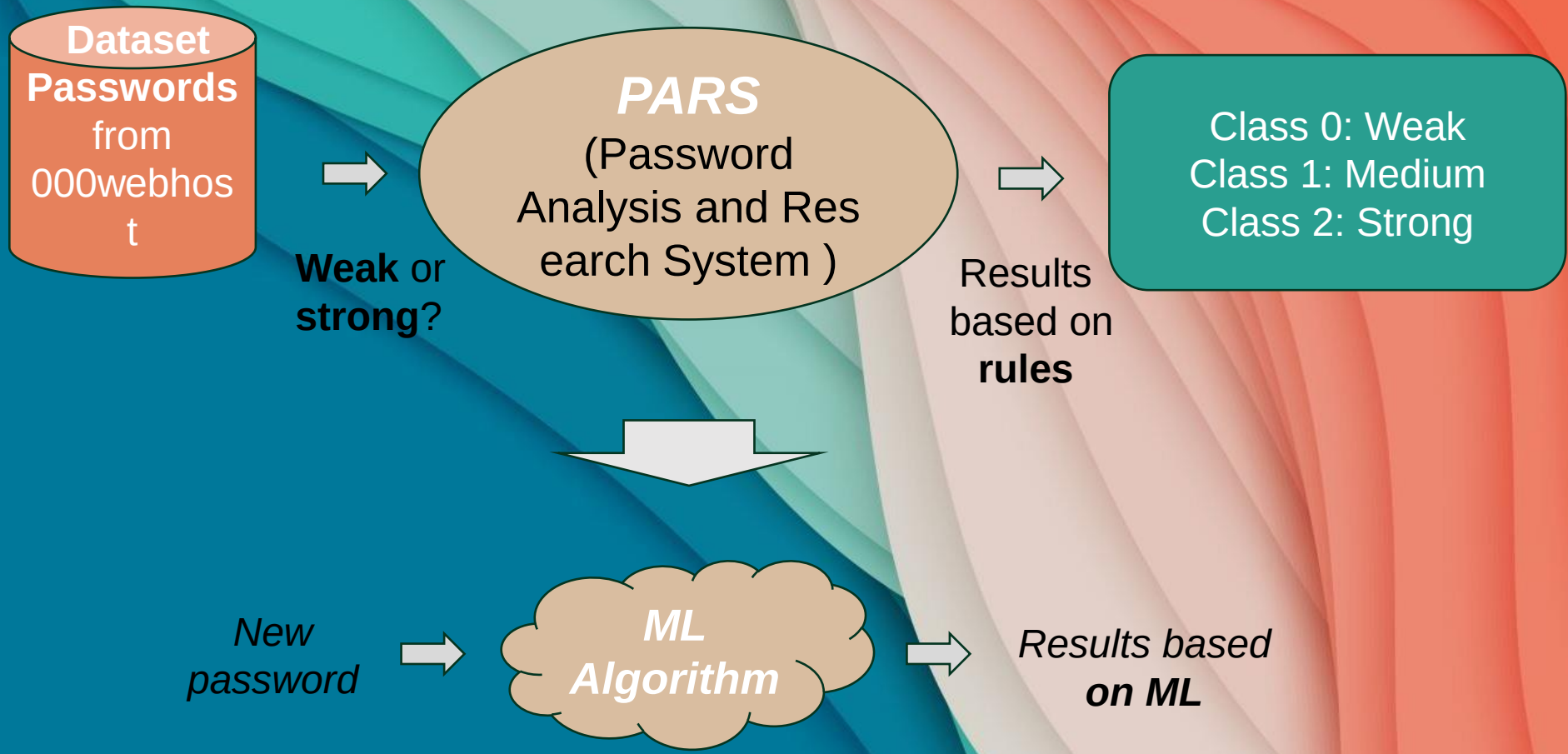
MACHINE LEARNING PROJECT

Davide Scintu
2023

Introduction



Dataset and password classification



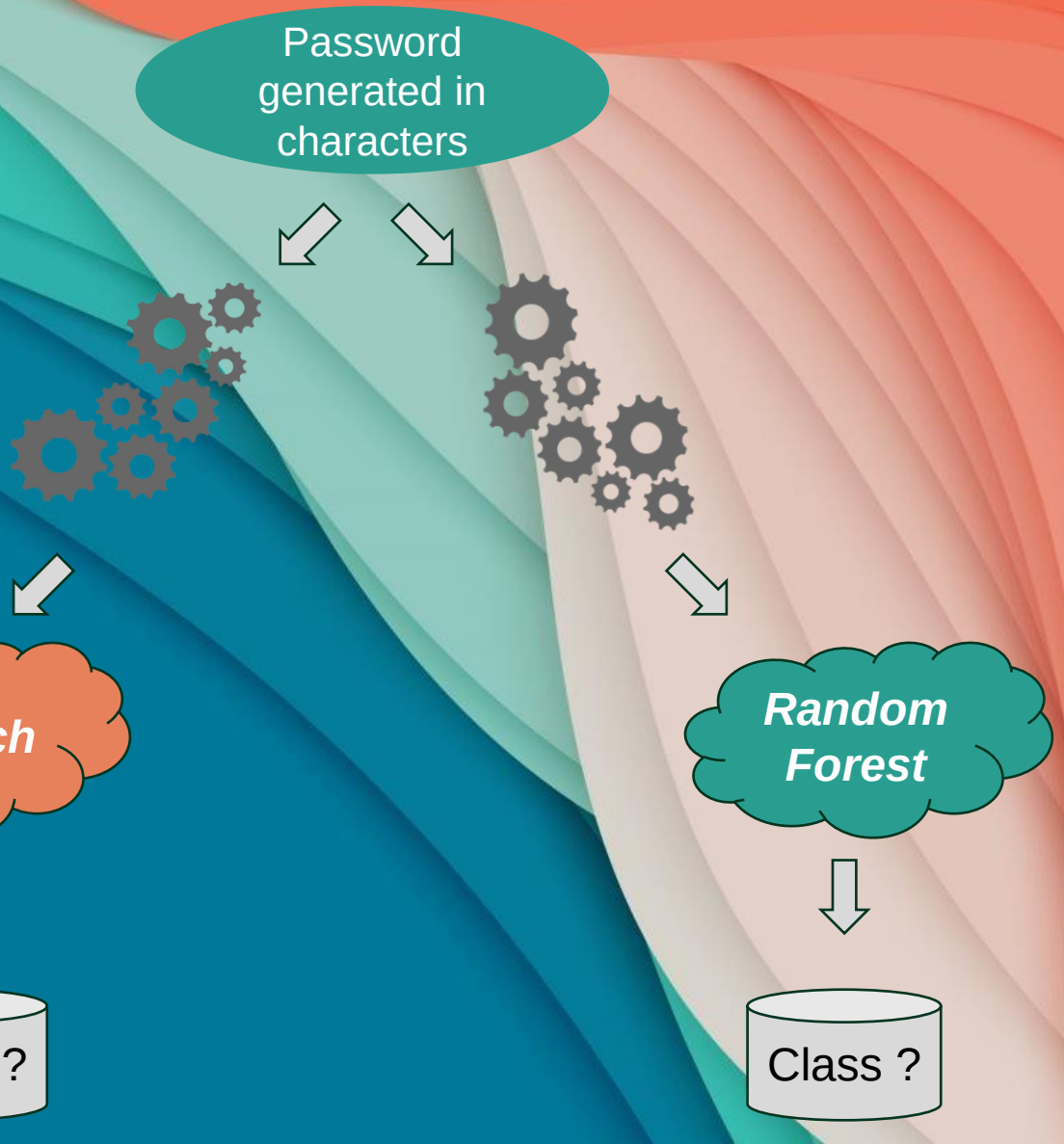
NNs vs ensemble learning

INPUT

DATA PROCESSING

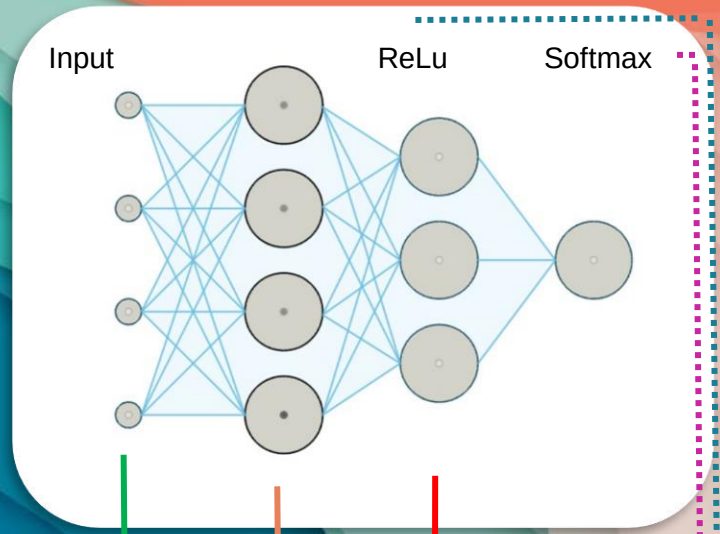
MODEL

RESULTS



Pytorch model - first simple NN

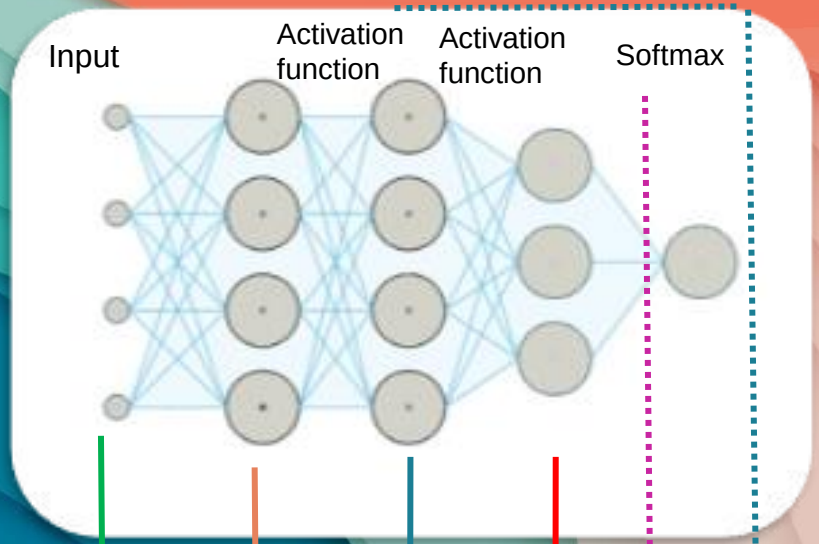
Davide Scintu –
ML Project 2023




```
input_size = X_train_tensor.shape[1]
hidden_size = 64
output_size = len(label_encoder.classes_)

model = PasswordClassifier(input_size, hidden_size, output_size)

criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
num_epochs = 2
batch_size = 32
```

```
class PasswordClassifier(nn.Module):
    def __init__(self, input_size, hidden_size, output_size):
        super(PasswordClassifier, self).__init__()
        self.fc1 = nn.Linear(input_size, hidden_size)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(hidden_size, output_size)
        self.softmax = nn.Softmax(dim=1)
```

Pytorch model - more complex NN

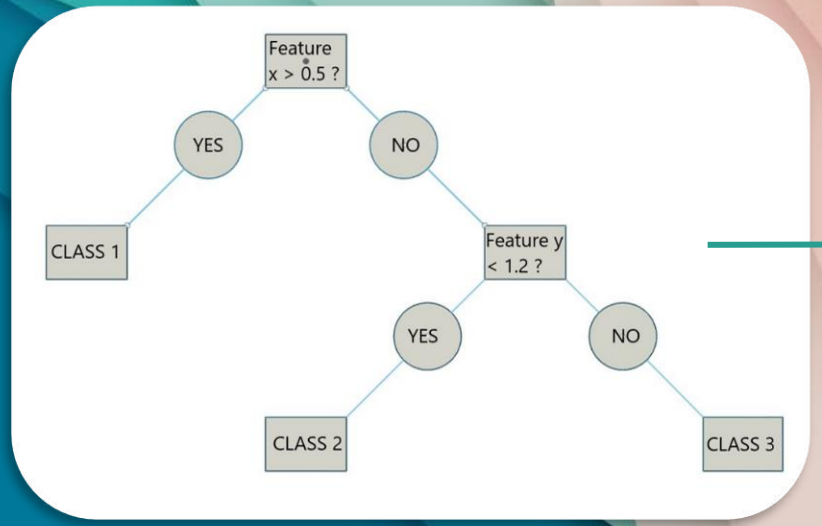



```
input_size = X_train_tensor.shape[1]
hidden_size1 = 64
hidden_size2 = 32
output_size = len(label_encoder.classes_)

model = PasswordClassifier(input_size, hidden_size1, hidden_size2, output_size)

criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(model.parameters(), lr=0.01, momentum=0.9)
num_epochs = 5
batch_size = 64
```

```
class PasswordClassifier(nn.Module):
    def __init__(self, input_size, hidden_size1, hidden_size2, output_size):
        super(PasswordClassifier, self).__init__()
        self.fc1 = nn.Linear(input_size, hidden_size1)
        self.activation = nn.Tanh()  # or nn.ReLu(), nn.LeakyReLU() ecc.
        self.fc2 = nn.Linear(hidden_size1, hidden_size2)
        self.fc3 = nn.Linear(hidden_size2, output_size)
        self.softmax = nn.Softmax(dim=1)
```

Random Forest model

```
model = RandomForestClassifier(random_state=42)
model.fit(X_train_vec, y_train)
```



Each tree is different from the others.

They are independent

Pytorch model vs RandomForest model

Number of samples = 100k

TRAINING

```
Epoch [1/2]: 100%|██████████| 2500/2500 [00:12<00:00, 200.02it/s, Loss=0.8422]
Epoch [2/2]: 100%|██████████| 2500/2500 [00:13<00:00, 186.61it/s, Loss=0.7086]
Test Accuracy: 0.8776
```

PyTorch

```
Training: 100%|██████████| 100000/100000 [03:28<00:00, 478.62samples/s]
Accuracy: 0.8614703422734604
```

RandomForest

TEST new passwords

```
test_password = "To#rkw1zxXzfY7$4^*89bHs7Xb5!#A!4Ve8xGb4jW9arQdU61k" # strong
test_password = "LIserEptInguISEd" # medium
test_password = "qwerty" # weak
```

RESULTS

Weak, Medium, Strong

Conclusion

Which one is better

References

- Password strenght image: https://www.hostpapa.com/blog/security/how-to-create-strong-passwords-to-secure-your-website/
- Password weakness: https://www.abecarolina.com/blog/test-your-password-strength
- Dataset source: https://www.kaggle.com/code/alperkaraca1/password-strength-analysis
- Gears image: https://it.vecteezy.com/arte-vettoriale/5511912-ingranaggi-lavoratore-meccanismo-simbolo-disegno-vettoriale
- Gear image: https://it.freepik.com/vettori-premium/gear-wheel-icon-illustrazione-vettoriale_32232346.htm#query=ingranaggio&position=6&from_view=keyword&track=sph
- Lamp image: https://www.istockphoto.com/it/vettoriale/lampadina-idea-vettore-icona-contorno-lampada-gm960813498-262365917
- RandomForest: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
- PyTorch: https://pytorch.org/tutorials/beginner/basics/tensorqs_tutorial.html