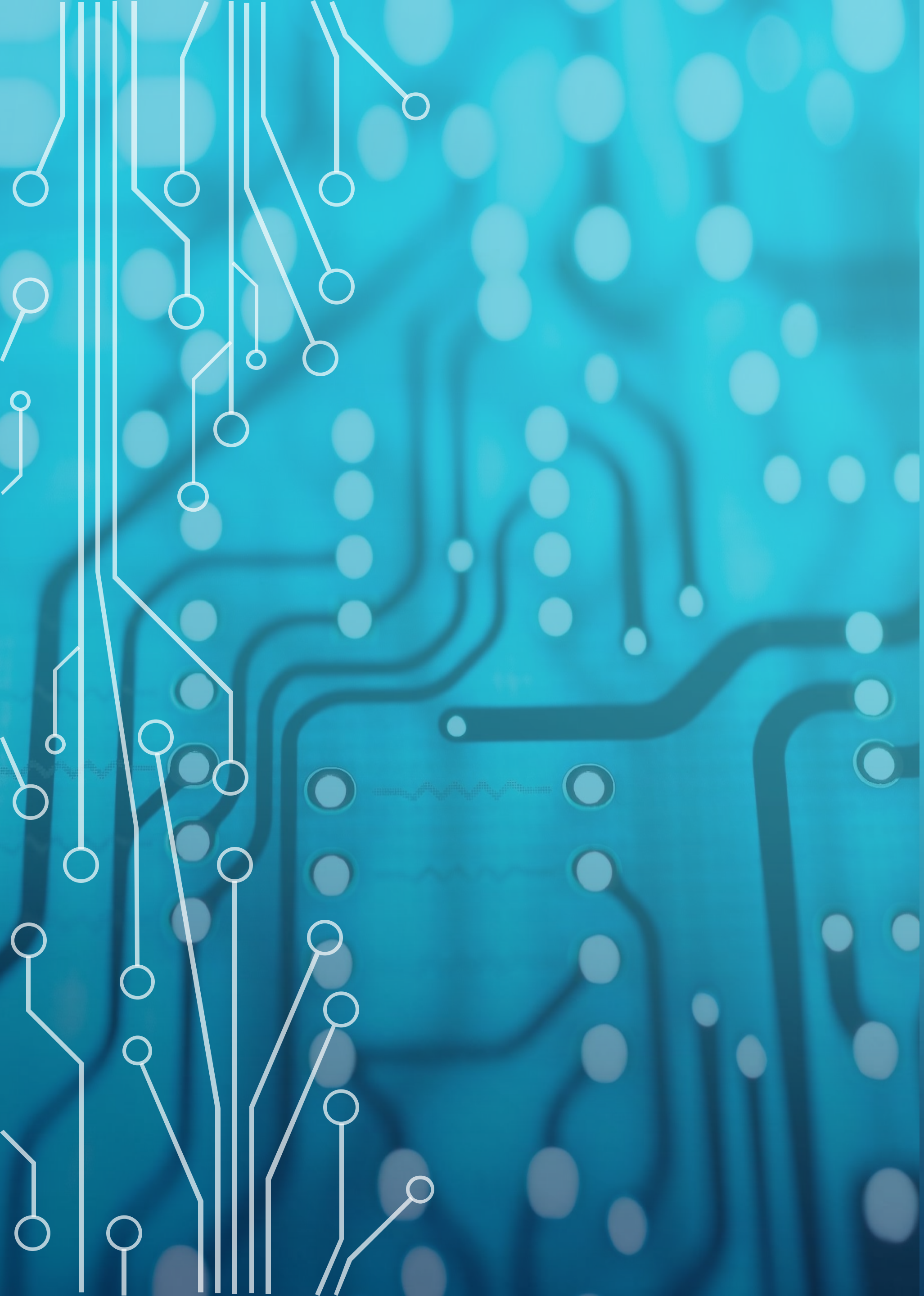
CLASSIFICATION OF PASSWORD STRENGTH

Rebecca Conti, Antonella Mei, Jacopo Orsini, Lorenzo Pannacci



FEATURE ENGINEERING:

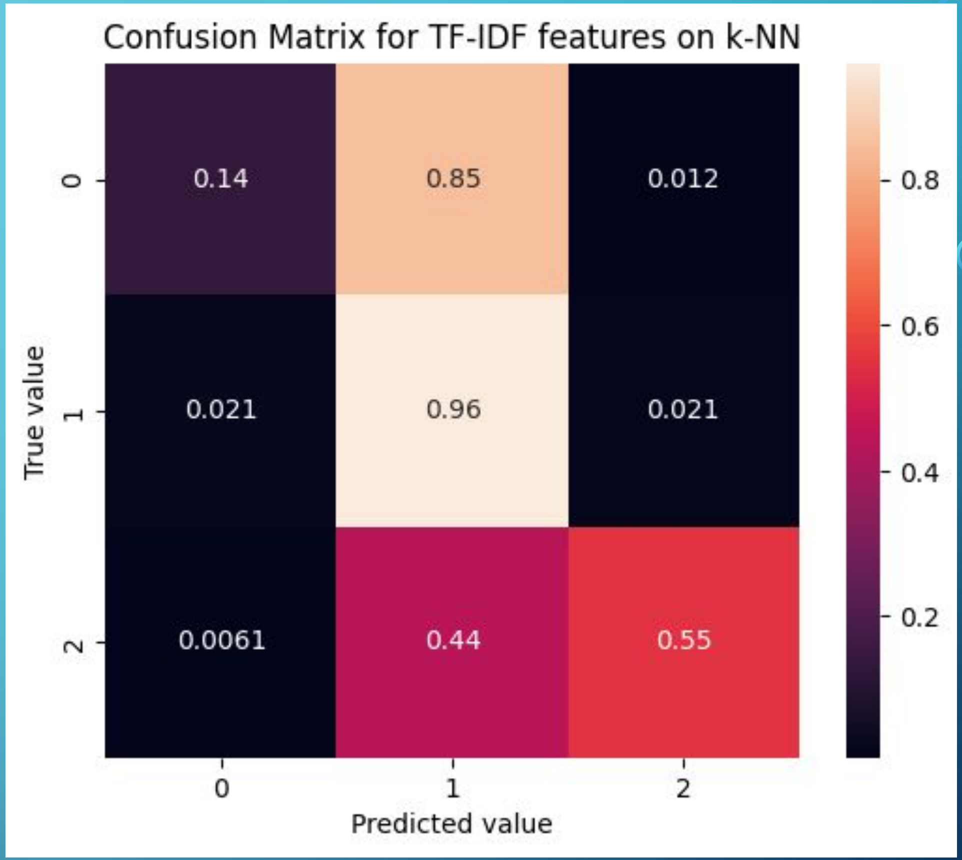
- Statistical features
  - Length, count of character types, percentage of character types, presence of special characters, of existing words or date.
- TFIDF features
  - This will be used to identify characters that are both frequently occurring within a specific password and relatively rare across the entire dataset.
  - This dataset does not contain informations about the length of the passwords.
- Per character features
  - We created a dataset that categorize each position in the password with the family of that character (lowercase, uppercase, number, symbol, empty).

DATASET

- We took our dataset from Kaggle: https://www.kaggle.com/datasets/bhavikbb/password-strength-classifier-dataset
- It has 669’880 samples but contains some dirty values, so we needed to delete some of them and adjust others.
- Final number of samples: 669’806
- We split into 20% test set and 80% train set, of which 20% is used for validation.
- The dataset is highly unbalanced and the baseline would be around 74% accuracy, however rebalancing the dataset we noticed only minimal decreases in overall performances

MODELS (1/4) – k-NN

- We identified which is the best value of k, according to the accuracy on the validation set.
- The model performed poorly on TF-IDF features, we thought that would be caused by unbalance of the classes towards the #1, however if balancing the dataset increases the performances of the two other classes, it also decreases in accuracy (around 70%).



TF-IDF	STATISTICAL WITH LENGTH	STATISTICAL W/O LENGTH	PER CHARACTER
79,67%	99,95%	95,45%	99,89%

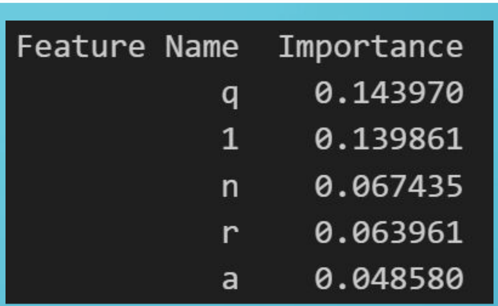
MODELS (2/4)

DECISION TREE

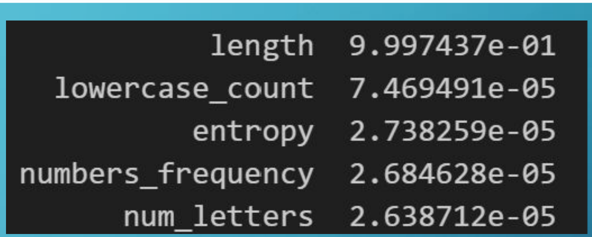
- We created a decision tree using the model built in sklearn. Then we applied the decision tree on each dataset in order to study the metrics as the accuracy, confusion matrix...

- We used the attribute ‘feature_importances_’ of the Decision Tree to analyze which are the most important characters for each dataset.

TF-IDF	STATISTICAL WITH LENGTH
92,59%	99,96%

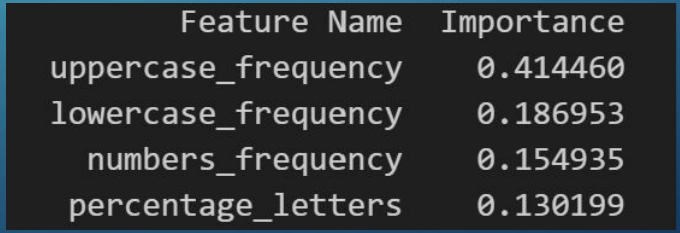


```
Feature Name   Importance
           q     0.143970
           l     0.139861
           n     0.067435
           r     0.063961
           a     0.048580
```

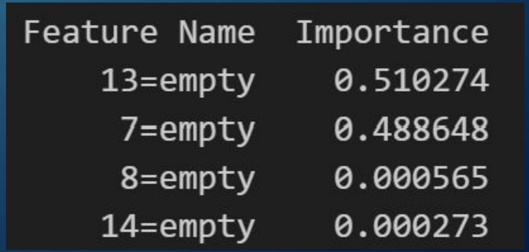


```
           length   9.997437e-01
  lowercase_count   7.469491e-05
          entropy   2.738259e-05
numbers_frequency   2.684628e-05
      num_letters   2.638712e-05
```

STATISTICAL W/O LENGTH	PER CHARACTER
95,88%	99,97%



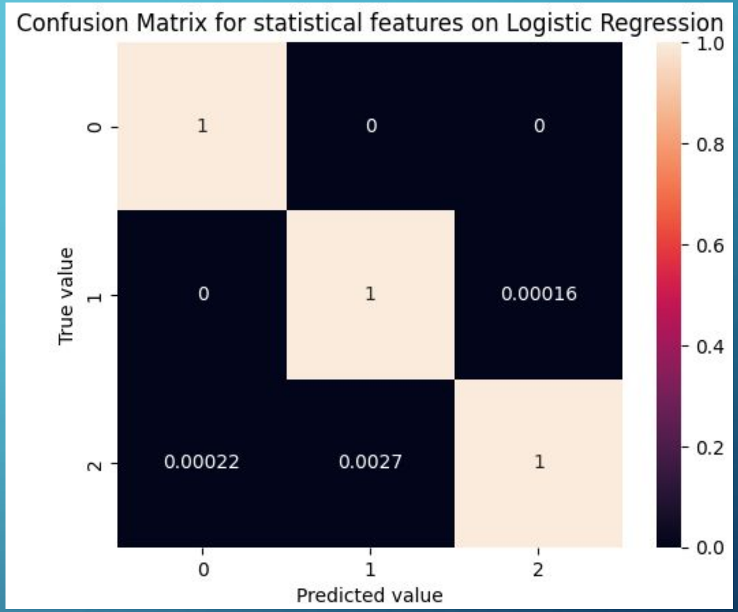
```
        Feature Name  Importance
uppercase_frequency     0.414460
lowercase_frequency     0.186953
  numbers_frequency     0.154935
 percentage_letters     0.130199
```



```
Feature Name   Importance
    13=empty     0.510274
     7=empty     0.488648
     8=empty     0.000565
    14=empty     0.000273
```

MODELS (3/4) Logistic Regression (one vs all)

- We implemented the Logistic Regression through the sklearn library.
- This model has been implemented for all 4 the different kind of features: Statistical, Statistical W/O Length, TF-IDF and CHAR.
- Performances greatly vary from one dataset to the other



TF-IDF	STATISTICAL WITH LENGTH	STATISTICAL W/O LENGTH	PER CHARACTER
81,84%	99,95%	83,56%	99,96%

MODELS (4/4) Neural Networks

Given the good results yielded by the previous models the use of Neural Networks may be unnecessary, however we got some meaningful results.

- Length continues to show itself as the most powerful feature we can extract, models trained over datasets that contains in some way the password length reach a near-perfect score.
- Datasets without length informations manage to reach much higher scores when compared to other models: in particular TF-IDF reach a near-perfect score
- This bring us to the conclusion that while length might be enough to distinguish between good and bad password it is not the only approach we can take, as there seems to be a correlation between password strength and other factors.

TF-IDF	STATISTICAL WITH LENGTH	STATISTICAL W/O LENGTH	PER CHARACTER
98,76%	99,96%	95,67%	99,97%

SUMMARY OF RESULTS:

- Every model that uses data containing informations about length has near-perfect results, so we want to focus on performances over the two datasets that don't contain length informations (Statistical w/o length and TF-IDF)
- k-NN performs poorly on TF-IDF but greatly on the other, however its poor running time makes it a sub-par choice compared to other models.
- Logistic regression has mediocre performances on both, maybe because of its generalization ability is limited to only linear relationships between variables
- Decision tree is a good model for our analysis both for performance and running time
- Neural networks yielded the best results overall even when using a small two layers network as we did

CONCLUSIONS:

While the variability of the target label could almost completely be explained by the password length (as it is in real scenarios, given that the best defense against brute-force approaches to password stealing is having a long password), is it interesting to see that even without this information we managed to create near-perfect model.

Those results could be explained by a correlation between password length and other features: people who creates long passwords are more likely to be conscious about security and might employ other good practises in password creation.

RELATED WORKS:

- https://github.com/gouravbarkle/Password-Strength-Classifier
- https://github.com/faizann24/Machine-Learning-based-Password-Strength-Classification/tree/master
- https://www.kaggle.com/code/burakergene/predict-password-strength-details-nlp-logistic-reg/notebook#Apply-TF-IDF-on-data
- https://www.kaggle.com/code/kaushalkrishna2000/password-strength-classifier-notebook

Thank you for your attention