Exercise 3 – K-Nearest Neighbors (KNN)

**Πρόβλημα**

Πρόβλεψη του τύπου των κρασιών καθώς και της ποιότητας τους

**Περιγραφή**

Το dataset περιλαμβάνει 3 τύπους διαφορετικών κρασιών (Λευκό, Κόκκινο, Ροζέ), όπου για κάθε δείγμα κρασιού αναγράφονται τα χαρακτηριστικά του που προέκυψαν έπειτα από φυσικοχημικές διεργασίες. Στη συνέχεια, τα δείγματα αξιολογήθηκαν για την ποιότητα τους από ειδικούς. Τα χαρακτηριστικά του dataset είναι τα παρακάτω:

1. *Type*
2. *Fixed Acidity*
3. *Volatile Acidity*
4. *Critic Acid*
5. *Residual Sugar*
6. *Chlorides*
7. *Free Sulfur Dioxide*
8. *Total Sulfur Dioxide*
9. *Density*
10. *pH*
11. *Sulphates*
12. *Alcohol*
13. *Quality*

**Οδηγίες**

1. Κατεβάστε το dataset από το φάκελο του μαθήματος
2. Φορτώστε το dataset σε pandas **DataFrame**.
3. Χωρίστε το dataset σε 70-30 train-test.

**Άσκηση 1 – Πρόβλεψη Type (Red/White)**

1. Αφαιρέστε τη στήλη “**quality**” χρησιμοποιώντας την εντολή df.drop(columns=[‘quality’], καθώς δεν επηρεάζει τον τύπου του κρασιού. Επιπλέον, χρησιμοποιήστε την εντολή df = df.dropna() για να αφαιρέσετε τις στήλες με missing values (nans). Στη συνέχεια, εκπαιδεύστε στο train set και μετρήστε το accuracy score για τις προβλέψεις του test set για καθένα από τους παρακάτω συνδυασμούς:
   1. : **5, 15, , 100**, όπου το πλήθος των παραδειγμάτων του train set.
   2. : ‘**uniform**’, ‘**distance**’.
   3. p: 1,2 (Για p=1 χρησιμοποιεί **Manhattan**, p=2 χρησιμοποιεί **Euclidean**).

Δώστε ένα πινακάκι με τα αποτελέσματα για κάθε συνδυασμό

1. Σε ποιες περιπτώσεις οι τιμές ενός χαρακτηριστικού επηρεάζουν την απόδοση του KNN? Υπάρχουν τέτοια χαρακτηριστικά μέσα στο dataset? Να τα αναφέρετε ονομαστικά και να εμφανιστεί το αντίστοιχο ιστόγραμμα τους
2. Δοκιμάστε να αξιοποιήσετε τα feature importances για την εκπαίδευση ενός KNN. Πιο συγκεκριμένα:
   1. Δοκιμάστε να εκπαιδεύσετε ένα **DecisionTree** με και να πάρετε τα **feature importances** .
   2. Δημιουργήστε bar-plot με τη σημαντικότητα κάθε χαρακτηριστικού.
   3. Αφαιρέστε από το dataset το λιγότερο σημαντικό χαρακτηριστικό και υπολογίστε το accuracy score του KNN για τον προηγούμενο καλύτερο συνδυασμό που βρήκατε.
   4. Είναι πολύ πιθανό το λιγότερο σημαντικό χαρακτηριστικό να συσχετίζεται (correlated) με κάποιο από τα υπόλοιπα χαρακτηριστικά του dataset και για αυτό να έχει χαμηλό importance score. Υπολογίστε το διάγραμμα correlation με την εντολή **matrix = df.corr()** και εμφανίστε το με τη βιβλιοθήκη matplotlib με την εντολή **plt.plot(matrix).**
3. Δοκιμάστε να κάνετε normalization τα δεδομένα και στη συνέχεια να εκπαιδεύσετε έναν ΚΝΝ. Πιο συγκεκριμένα:
   1. Δοκιμάστε να κάνετε Normalization (ή αλλιώς “***scaling***”) στα inputs. Για να το κάνετε αυτό, χρησιμοποιείστε τον **MinMaxScaler,** εισάγοντας τον με την εντολή “”. Χρησιμοποιήστε την εντολή scaler.fit(x\_train) στα δεδομένα του train set και ύστερα scaler.transform(x\_train) και scaler.transform(x\_test) για τα παραδείγματα του train και test αντίστοιχα. Με αυτόν τον τρόπο, κάθε χαρακτηριστικό θα παίρνει τιμές από 0 ως 1. Έτσι, κάθε χαρακτηριστικό θα συμβάλει ισάξια στις προβλέψεις.
   2. Μετρήστε το **accuracy score** του ΚΝΝ από το ερώτημα 1, δοκιμάζοντας μόνο τις δύο διαφορετικές μετρικές απόστασης (**Manhattan**, **Euclidean**).

**Άσκηση 2 – Πρόβλεψη Ποιότητας (0-10)**

Επαναλάβετε το ερώτημα (1) της άσκησης 1, αυτή τη φορά για την πρόβλεψη της ποιότητας (**quality**) του κρασιού. Για τον τύπο (**type**) του κρασιού, θα πρέπει να μετατρέψετε τις ονομαστικές τιμές “white”, “red”, “rose” σε διάνυσμα one-hot διανύσματα. Μπορείτε να χρησιμοποιήσετε τη συνάρτηση όπως φαίνεται παρακάτω, για να αντικαταστήσετε την στήλη “type” με το αντίστοιχο one-hot vector. Υπολογίστε το **MAE** για κάθε συνδυασμότων παραμέτρων και δώστε το αντίστοιχο πινακάκι.

Εικόνα που περιέχει κείμενο, γραμματοσειρά, στιγμιότυπο οθόνης

Περιγραφή που δημιουργήθηκε αυτόματα

**Bonus Άσκηση – Μείωση Διάστασης (LDA, UMAP)**

1. Πάρτε τα x\_train, y\_train της άσκησης 1 και εφαρμόστε τη μέθοδο Linear Discriminant Analysis (LDA) για τη μείωσης διάστασης του x\_train. Χρησιμοποιήστε τον υλοποιημένο αλγόριθμο **sklearn.discriminant\_analysis.LinearDiscriminantAnalysis(n\_components=2)** καλώντας την εντολή lda.fit(x\_train, y\_train) και στη συνέχεια τις εντολές lda.transform(x\_train) και lda.transform(x\_test). Μπορείτε να συμβουλευτείτε το παράδειγμα στο documentation: <https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html>
2. Δημιουργήστε scatter plot με τα σημεία των x\_train και x\_test. Ορίστε ένα διαφορετικό χρώμα για τα White & Red αντίστοιχα (πχ μπλε κόκκινο). Επίσης, για να ναι πιο καθαρό το plot, μπορείτε να ορίσετε το opacity (alpha) των σημείων του train στο 0.5. Μπορείτε να συμβουλευτείτε το παράδειγμα στο documentation: [https://matplotlib.org/stable/gallery/lines\_bars\_and\_markers/scatter\_with\_legend.html#sphx-glr-gallery-lines-bars-and-markers-scatter-with-legend-py](https://matplotlib.org/stable/gallery/lines_bars_and_markers/scatter_with_legend.html)
3. Χρησιμοποιείστε ΚΝΝ Classifier (χωρίς να πειράξετε τις υπέρ-παραμέτρους) και υπολογίστε το accuracy score.
4. Επαναλάβετε τη διαδικασία χρησιμοποιώντας την τεχνική Uniform Manifold Approximation and Projection (UMAP). Θα χρειαστεί να την εγκαταστήσετε στο colab με την εντολή !pip install umap-learn. Θα χρειαστείτε την εντολή umap.UMAP(n\_components=2).fit\_transform(x\_train, y=y\_train) για τα train και test αντίστοιχα. Μπορείτε να συμβουλευτείτε το documentation: https://umap-learn.readthedocs.io/en/latest/supervised.html