Aufgabe 1 – 20%

Betrachten Sie die logistische Regression:

$$\sum_i \log(1 + e^{-y^{(i)}((x^{(i)})^T w+b)}) \tag{1}$$

- In welchem Intervall befindet sich $e^{-y(x^T w+b)}$? In welchem Intervall befindet sich $\log(1 + e^{-y(x^T w+b)})$?
- Zeigen Sie, falls x bzgl. $f(x) = x^T w + b$ korrekt gelabelt wird, so gilt $-yf(x) < 0$.

Aufgabe 2 – 20%

Betrachten Sie folgende 2D-Datenpunkte, aufgeteilt in zwei unterschiedlich klassifizierte Mengen $X_0 = \{(1,0),(0,1),(-1,0),(0,-1)\}$ und $X_1 = \{(2,0),(0,2),(-2,0),(0,-2)\}$. Die Punkte in X_0 sind mit 0 gelabelt, die Datenpunkte in X_1 mit 1. Der Trainingsdatensatz X besteht aus allen Punkten $X = X_0 \cup X_1$.

- Wenn Sie eine Hard-Margin-SVM benutzen - wieviele Fehler machen sie minimal? Was wäre ein besserer Algorithmus zur Klassifikation?
- Beweisen Sie dass X nicht linear separierbar ist. Sie können den Beweis entweder anhand einer Grafik machen, oder den vollständigen mathematischen beweis. Betrachten Sie für letzteres die Möglichkeiten für $\{w,b|f(x) = x^T w + b\}$ und zeigen Sie, warum es keine Hyperebene (definiert durch w und b) geben kann, die X_0 und X_1 separiert. Eine Fallunterscheidung von $|b| > 2$, $2 > |b| > 1$ und $1 > |b| > 0$ kann hilfreich sein.
- Angenommen Sie transformieren die 2D-Punkte aus X mit der Funktion g in 3D-Punkte, wobei $g(a,b) = (a,b,a^2+b^2)$ in 3D-Punkte. Wie sieht dieser transformierte Datensatz aus (eine kurze Beschreibung reicht)? Ist der neue Datensatz linear trennbar, und wie würde die Hyperebene aussehen, die die Klassen trennt?

Aufgabe 3 – 10%

In der Vorlesung wurde erwähnt, dass ein großer Margin bei SVMs einer kleinen Modell-Komplexität entspricht. Sagen sie abstrakt, warum das so ist, und was das für den Bias-Variance-Tradeoff bedeutet. (Tipp: Wie kommt es zu einem großen Margin?)

Aufgabe 4 – 50%

In der Vorlesung haben Sie logistische Regression und lineare Support Vector Machines als Klassifikations-Algorithmen kennengelernt. In dieser Übung sollen Sie beide Algorithmen auf Klassifizierungsprobleme anwenden und sich näher mit der Bewertung der Performance von Modellen bei Klassifikationsproblemen auseinandersetzen. Für diese Übung werden Ihnen die Datensätze `iris_2D.csv`, `moons.csv` und `creditcard_20percent_sample.csv` vorgesehen. Den ersten Datensatz kennen Sie bereits aus einer vorherigen Übung. Der zweite Datensatz enthält einen Datensatz, der für einige Machine-Learning-Modelle eine Herausforderung darstellen kann. Der dritte Datensatz beschreibt ein typisches Klassifikationsproblem: Die Identifikation von Kreditkarten-Betrug. Setzen Sie für **alle** Methoden, die Zufallsgeneratoren verwenden, den `random_state` Parameter auf 20. Ihre **lineare** Support Vector Machine sollten Sie mit dem Parameter `dual="auto"` initialisieren.
Ihnen steht ein Jupyter Notebook `logistic_regression.ipynb` zur Verfügung, das ein grobes

Gerüst für die Aufgabe bereitstellt. Sie sind nicht verpflichtet das Notebook zu nutzen, es dient nur zur Orientierung und Hilfestellung.

Part 1 (15%)

Arbeiten Sie zunächst mit dem Datensatz `moon.csv`. Trainieren Sie sowohl einen logistischen Regressor als auch eine **lineare** Support Vector Machine auf den Daten. Für diese Aufgabe sollten Sie die Reihenfolge der Datenpunkte randomisieren und ein Test-Set mit 20% der Daten erstellen (`train_test_split`). Stellen Sie für beide Classifier in einem 2D-Plot dar, welche Bereiche der durch die zwei Features aufgespannten Fläche den verschiedenen Klasen zugeordnet sind. Visualisieren Sie außerdem die Vorhersagen für den von Ihnen erstellten Test-Datensatz. Was fällt Ihnen auf? Geben Sie ebenfalls *Precision* und *Recall* der Klassifikation an.
Bonus (nicht Teil der Bewertung): Probieren Sie mit der `SVC` Klasse von *scikit-learn* eine bessere Performance zu erreichen als mit den anderen beiden Klassifikationsalgorithmen.

Part 2 (15%)

Führen Sie nun dieselben Schritte wie in Part 1 für den Datensatz `iris_2D.csv` durch (damit ist auch die Darstellung der Ergebnisse gemeint!). Nutzen Sie die *One-vs-Rest* Strategie, um die Algorithmen auf das Problem der Multi-Class-Klassifizierung zu erweitern.

Part 3 (20%)

Zuletzt arbeiten Sie nun mit dem Datensatz `creditcard_20percent_sample.csv`. Verwenden Sie wieder beide Klassifikationsalgorithmen. Erstellen Sie wie in den vorherigen Aufgaben beschrieben ein Test-Set und geben Sie die beschriebenen Metriken an. Bewerten Sie diese kurz. Erstellen Sie nun eine Precision-Recall-Kurve, welchen Kompromiss aus *Precision* und *Recall* würden Sie in diesem Praxisfall empfehlen?

Bitte lösen Sie die Fragen und die Programmieraufgaben bis zum **8. Januar 2024**. Ihren Python-Code, sowie Visualisierungen können Sie in Form eines Jupyter Notebooks oder PDFs hochladen.