Теоретические задачи

3.1 Знакомство с линейного классификатором
1) Бинарный линейный классификатор:

$$a(x) \quad = \quad \begin{cases} 1 & , f(x) > 0 \\ -1 & , f(x) < 0 \end{cases} \tag{1}$$

где $f(x) = w_0 + w_1 \cdot x_1 + ... + w_n \cdot x_n$

2) Отступ алгоритма $a(x) = sign\{f(x)\}$ на объекте x_i - величина $M_i = y_i \cdot f(x_i)$, где y_i - класс объекта x_i. Заметим, что

$$M_i \leq 0 \Leftrightarrow a(x) \neq y_i$$

$$M_i > 0 \Leftrightarrow a(x) = y_i$$

3) Классификаторы вида $a(x) = sign(< w, x > -w_0)$ можно свести к классификаторам вида $a(x) = sign(< w, x >)$, добавив к x ещё одну координату равную -1, тогда $x = (-1, x_1, x_2, ..., x_n)^T$, $w = (w_0, w_1, ...w_n)^T$.

4) Функция эмпирического риска через отступы: $Q(w) = \sum_{i=1}^{l} [M_i(w) < 0]$. Этот функционал минимален для наилучшего алгоритма классификации (в самом лучшем случае равен 0).

5) Если функционал эмпирического риска равен $Q(w) = \sum_{i=1}^{l} [M_i(w) < 0]$, то при w = 0 $f(x_i) = sign(< w, x >) = 0$ при любом x_i, значит $M_i(w) = 0$ для любого i и $Q(0) = 0$. Получим, что при w = 0 достигается минимум функционала для алгоритма $a(x) = sign(f(x))$.

6) Функционал аппроксимированного эмпирического риска - $\tilde{Q}(w) = \sum_{i=1}^{l} L(M_i(w))$, где L - функция потерь.

7) Функция потерь характеризует величину отклонения ответа алгоритма $y = a(x)$ от правильного ответа y^*. Через функцию потерь выражается функционал эмпирического риска, который равен ошибке алгоритма на заданной выборке. Среди заданных алгоритмов нужно найти тот, на котором функционал эмпирического риска минимален. Для того чтобы можно было применять градиентные методы оптимизации, берут непрерывные аппроксимации пороговой функции потерь: $Q(M) = (1 - M)^2, V(M) = (1 - M)_+, L(M) = log_2(1 + e^{-M})$... Обычно функции потерь монотонные, гладкие, равны 1 в 0, стремятся к 0 на $+\infty$ и к бесконечности при $-\infty$.

8) Негладкая функция потерь - пороговая функция потерь $Q(w) = [M(w) < 0]$.

9) Регуляризация - это метод для уменьшения переобучения модели, к функционалу аппроксимированного риска добавляется штраф на веса(уменьшение переобучения происходит из-за того, что веса не могут быть сильно большими)

l1-регуляризация: $\gamma \cdot \sum_{k=1}^{l} |w_k|$

l2-регуляризация: $\gamma \cdot \sum_{k=1}^{l} w_k^2$

10) Алгоритм обладает обобщающей способностью, если вероятность ошибки на тестовой выборке несильно отличается от вероятности ошибки на обучающей выборке.При переобучении же вероятность ошибки на тестовой выборке значительно больше вероятности ошибки на обучающей выборке. То есть если алгоритм переобучился, то у него низкая обобщающая способность. Так как регуляризация уменьшает переобучение, она увеличиает обобщающую способность алгоритма.

11)

!!12) Так как регуляризация ограничивает веса алгоритма, из-за неё часть весов обнуляться, причем чем больше параметр γ, тем больше весов обнулятся. Регуляризация ответственна за отбор признаков, значит по сути она уменьшает число переменных функции риска, которые имеют значение.

!!13) Регуляризация предотвращает переобучение, но увеличивает ошибку алгоритма на обучающей выборке (при переобучении вероятность ошибки на обучающей выборке меньше, так как алгоритм подстраивается под данные)

14)

15) Рассмотрим задачу бинарной классификации, каждый объект относится к одному из двух классов - 0 и 1. Вводятся следующие метрики качества алгоритма:
Accuracy - доля правильных ответов (число правильных ответов / число ответов)
Precision - точность ответа (число правильно угаданных 1 / число раз, когда алгоритм ответил "1")
Recall - полнота ответа (число правильно угаданных 1 / число 1)

TP(True Positive) - истинное значение - 1, ответ алгоритма - 1
TN(True Negative) - истинное значение - 0, ответ алгоритма - 0
FP(False Positive) - истинное значение - 0, ответ алгоритма - 1
FN(False Negative) - истинное значение - 1, ответ алгоритма - 0

Тогда $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$

16) Метрика AUC - area under the curve - считает площадь под кривой. Метрика ROC-AUC - площадь под ROC-кривой. ROC-кривая строится в координатах TPR(FPR), где $TPR = \frac{TP}{TP+FN}$, $FPR = \frac{FP}{FP+TN}$. ROC-кривая проходит через точки (0, 0) (если алгоритм всегда выдает 0, то TP =FP = 0) и (1, 1) - (если алгоритм всегда выдаёт 1, то FN = TN = 0 и TPR = FPR = 1).

17) В контесте про фамилии строится линейный классификатор. Переобозначим классы как {1, -1}(1 - фамилия, -1 - не фамилия). Тогда этот линейный классификатор можно записать в виде $a(x) = sign(< w, x > -w_0)$. Пусть после обучения веса алгоритма - $\hat{w}$. Заметим, что при $w_0 > \max\limits_x(< \hat{w}, x >)$ a(x) дает -1 на всех элементах выборки, значит TPR = FPR = 0, при $w_0 < \min\limits_x(< \hat{w}, x >)$ a(x) = 1 на любом x, и TPR = FPR = 1.Для построения ROC-кривой будем варьировать w_0 от $\min\limits_x(< \hat{w}, x >)$ до $\max\limits_x(< \hat{w}, x >)$, при каждом w_0 считаем TPR и FRP: TPR = число правильно угаданных фамилий / общее число фамилий в тесте; FPR = число не фамилий, на которых алгоритм ответил, что это фамилия / общее число не фамилий.