Machine Learning Exam

Name: Chitradevi Maruthavanan
PSU_ID: 950828319

1. (6 points) An online retailer like Amazon wants to determine which products to promote based on reviews. They only want to promote products that are likely to sell. For each product, they have past reviews. The reviews have both a numeric score (from 1 to 5) and text.

(a) To formulate this as a machine learning problem, suggest a target variable that the online retailer could use.

To formulate this as a machine learning problem, the online retailer could use a binary target variable indicating whether a product is likely to sell or not based on past reviews. This target variable could be represented as 1 for products that have a high probability of being purchased and 0 for those with a low probability of being purchased.

(b) For the predictors of the target variable, a data scientist suggests to combine the numeric score with frequency of occurrence of words that convey judgement like “bad", “good", and “doesn't work." Describe a possible linear model for this relation.

probability of product to sell = β_0+ β_1[numeric score]+ β_2[frequency of “bad”]+]+ β_3[frequency of “good”]+]+ β_4[frequency of “doesn’t work”]

$$y \approx \hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4$$

(c) Now, suppose that some reviews have a numeric score from 1 to 5 and others have a score from 1 to 10. How would this change your features?

If some reviews have a numeric score from 1 to 5 and others have a score from 1 to 10, it will affect the feature related to the numeric score. The feature related to the frequency of occurrence of words would remain the same. To handle this difference, we would need to normalize the scores to make them comparable. We can do this by using a range of 0 to 1. Then we can divide the 1-5 score range by 5 and the 1-10 score range by 10 so that they will be in a normalized range of 0 to 1

2. (3 points) In Support Vector Machine optimization problem, parameter C provides a balance between the hinge loss and margin. Discuss the impact of large or small C on model over/under fitting and how an ML engineer can find the optimal C?

In Support Vector Machine (SVM) optimization, the parameter C balance between the hinge loss and the inverse of margin. A small value of C allows for a larger margin but can result in misclassifying some training examples, leading to underfitting. A large value of C minimizes

the hinge loss at the expense of a smaller margin, which can lead to overfitting the training data. If the value of C is too small, the model will underfit and perform poorly on both the training and test data. If the value of C is too large, the model will overfit and perform well on the training data but poorly on the test data. Therefore, it is essential to choose an appropriate value of C that balances the trade-off between underfitting and overfitting.
To find the optimal value of C, an ML engineer can use cross-validation. Therefore, to correctly select the parameter C, we need to identify through cross-validation.

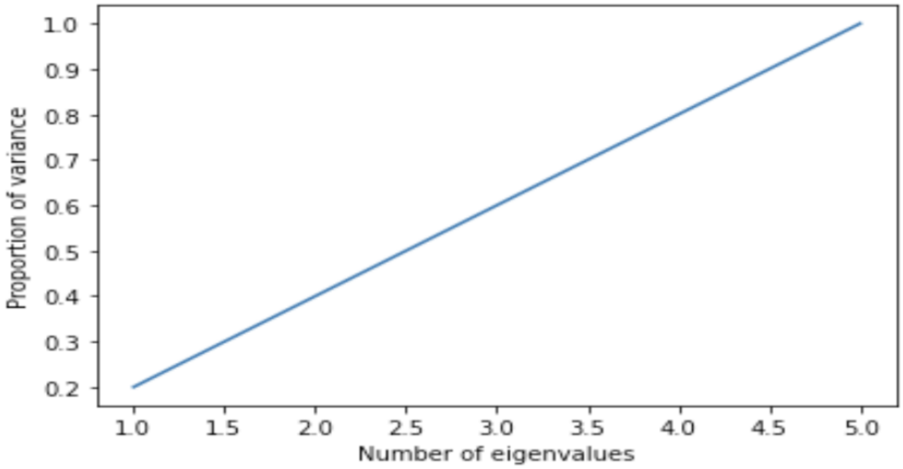
3. (3 points) Consider a dataset with five dimensions where each eigenvalue is equal to two.
Plot the proportion of variance as a function of number of eigenvalues? Is dimensionality reduction a good tool to reduce the data dimension and if so, how many dimensions would you choose for dimensionality reduction?

Since all eigenvalues are equal to 2 this means that each eigenvalue represents an equal proportion of the variance.

Number of Eigen values	**Proportion of variance**
1	**0.2**
2	**0.4**
3	**0.6**
4	**0.8**
5	**1.0**

It can be plotted like this:

y^ = 0.2x



Dimensionality reduction seems like a good tool to use, as we can reduce the dimension of the dataset while still retaining the same variation in data since all eigenvalues are equal. We could use a single dimension to accurately represent our data in this case.

4. (3 points) Let X be a dataset with number of rows corresponding to different samples and number of columns corresponding to different features. Let Y be the target variable vector. Write a few lines of code that gets a 70 to 30 train vs test split of the data and further removes the mean from each column and appropriately scales the data?

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3)
X_scaler = StandardScaler()
Y_scaler = StandardScaler()

X_train1 = X_scaler.fit_transform(X_train)
Y_train1 = Y_scaler.fit_transform(Y_train[:,None])

X_test1 = X_scaler.transform(X_test)
Y_test1 = Y_scaler.transform(Y_test[:,None])
```

5. (6 points) In this problem, we will look at how to exhaustively search over subsets of features. You are given three python functions:

```
model = LinearRegression() # Create a linear regression model object
model.fit(X,y) # Fits the model
yhat = model.predict(X) # Predicts targets given features
```

Given training data Xtr,ytr and test data Xts,yts, write a few lines of python code to:

(a) Find the best model using only one feature of the data (i.e. one column of Xtr and Xts).

```
from sklearn.linear_model import LinearRegression
import numpy as np
Xtr, ytr = np.load('Xtr.npy'), np.load('ytr.npy')
Xts, yts = np.load('Xts.npy'), np.load('yts.npy')
model = LinearRegression()

# Find the best model using only one feature
best_feature, best_score = None, -np.inf
for i in range(Xtr.shape[1]):

    # Select a single feature
    Xtr_i = Xtr[:, i].reshape(-1, 1)
    Xts_i = Xts[:, i].reshape(-1, 1)

    # Fit the model on the training data
    model.fit(Xtr_i, ytr)

    # Predict targets for the test data
    yhat = model.predict(Xts_i)

    # Compute the R-squared score
    score = model.score(Xts_i, yts)

    # Update the best feature and score
    if score > best_score:
        best_feature, best_score = i, score
print(f"Best feature: {best_feature}, R-squared score: {best_score}")
```

(b) Find the best model using only two features of the data (i.e. two columns of Xtr and Xts).

```
# Find the best model using only two features
best_features, best_score = None, -np.inf
for i in range(Xtr.shape[1]):
    for j in range(i+1, Xtr.shape[1]):
        Xtr_ij = Xtr[:, [i,j]]
        Xts_ij = Xts[:, [i,j]]

        model.fit(Xtr_ij, ytr)

        # Predict targets for the test data
        yhat = model.predict(Xts_ij)

        # Compute the R-squared score
        score = model.score(Xts_ij, yts)

        # Update the best features and score
        if score > best_score:
            best_features, best_score = (i,j), score

print(f"Best features: {best_features}, R-squared score: {best_score}")
```

6. Answer:

Compute the gradient $\partial f (x)/\partial x$ of the function
We have the below function which is of degree 2

$f (x) = x1 + x2x3.$

Taking partial derivatives, with respect to x1,x2 and x3 we get the below

$\partial f (x)/\partial x1 = 1$

$\partial f (x)/\partial x2 = x3$

$\partial f (x)/\partial x3 = x2$

For vector input W, and scalar function f(w)the gradient is

$\nabla f (w)= [\partial f (w)/\partial w1 \ldots\ldots. \partial f (w)/\partial wN]$

Therefore, we get the below for our question

the gradient of the function f (x) is:

$\nabla f (x) = [\partial f (x)/\partial x1, \partial f (x)/\partial x2, \partial f (x)/\partial x3] = [1, x3, x2]$

Since gradient is same size as the argument, and we have 3 arguments in our input function we get the below

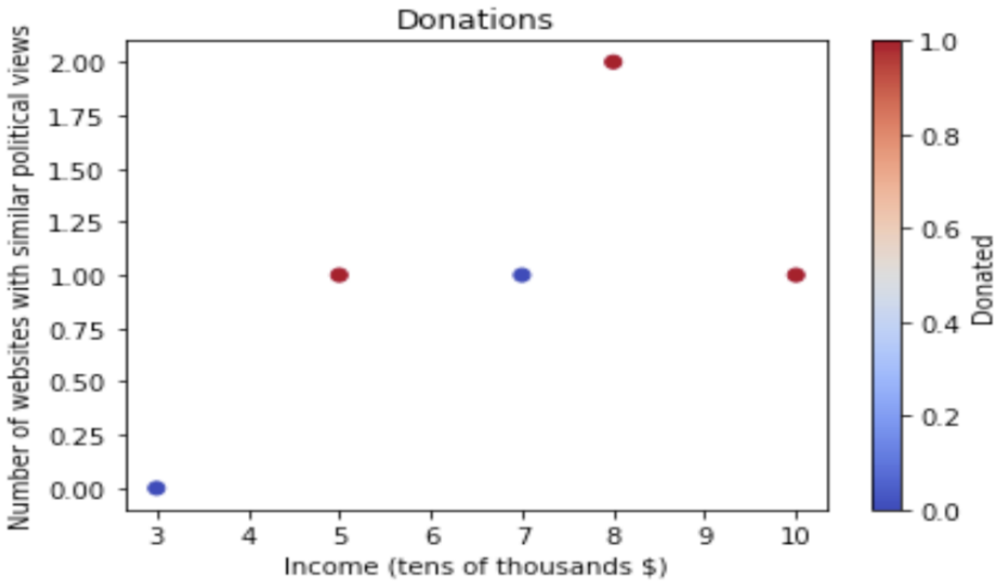
Size of gradient vector = 3

7. a) Answer

```
import matplotlib.pyplot as plt

# Define the data
income = [3, 5, 7, 8, 10]
num_websites = [0, 1, 1, 2, 1]
donate = [0, 1, 0, 1, 1]

# Create the scatter plot
plt.scatter(income, num_websites, c=donate, cmap='coolwarm')
plt.xlabel('Income (tens of thousands $)')
plt.ylabel('Number of websites with similar political views')
plt.title('Donations')
plt.colorbar(label='Donated')
plt.show()
```



---

7. b) Answer

$\mathbf{Z_i = w^T x_i + b}$
$\mathbf{Z_1 = 3w_1 + 0\ w_2 + b < 0}$
$\mathbf{Z_2 = 5\ w_1 + w_2 + b > 0}$
$\mathbf{Z_3 = 7\ w_1 + w_2 + b < 0}$
$\mathbf{Z_4 = 8\ w_1 + 2\ w_2 + b > 0}$
$\mathbf{Z_5 = 10\ w_1 + w_2 + b > 0}$

Consider,
W = [1, 1] and b = - 4

$\mathbf{Z_i = [1,1]^T x_i + b}$
$\mathbf{Z_i = x_1 + x_2 - 4}$

Case 1:
$\mathbf{Z_i = x_1 + x_2 - 4}$
$\mathbf{Z_1 = 3 + 0 - 4}$
$\mathbf{Z_1 = -1 < 0}$ **(true)**

Case 2:
$\mathbf{Z_2 = 5\ w_1 + w_2 + b}$
$\mathbf{Z_2 = 5 + 1 - 4}$
$\mathbf{Z_2 = 2 > 0}$ **(true)**

<u>Case 3:</u>

$Z_3 = 7 w_1 + w_2 + b$
$Z_3 = 7 + 1 -4$
$Z_3 = 4 > 0$ (false)

<u>Case 4:</u>

$Z_4 = 8 w_1 + 2 w_2 + b$
$Z_4 = 8 + 2 -4$
$Z_4 = 6 > 0$ (true)

<u>Case 5:</u>

$Z_5 = 10 w_1 + w_2 + b$
$Z_5 = 10 + 1 -4$
$Z_5 = 7 > 0$ (true)

Hence $w_1 = 1$, $w_2 = 1$ and b = -4 works for all the points, but we got one error in the case 3.

7. c) <u>Answer</u>

Using the logistic model, we have:
$P(y_i=1|x_i) = 1 / (1 + e\text{-}^{z_i})$

Using w = [1, 1] and b = -4 from the previous part, we can calculate Z_i for each sample:

$Z_1 = w^T x_1 + b = (1)(3) + (1)(0) - 4 = -1$
$z_2 = w^T x_2 + b = (1)(5) + (1)(1) - 4 = 2$
$z_3 = w^T x_3 + b = (1)(7) + (1)(1) - 4 = 4$
$z_4 = w^T x_4 + b = (1)(8) + (1)(2) - 4 = 6$
$z_5 = w^T x_5 + b = (1)(10) + (1)(1) - 4 = 7$

Now we can calculate the probability of each sample:
P(y_1=1|x1) = 1 / (1 + e^1) = 0.2689
P(y_2=1|x2) = 1 / (1 + e^-2) = 0.8808
P(y_3=1|x3) = 1 / (1 + e^-4) = 0.9820
P(y_4=1|x4) = 1 / (1 + e^-6) = 0.9975
P(y_5=1|x5) = 1 / (1 + e^-7) = 0.9991
Therefore, sample 1 ($x_1 = 3$ and $x_2 = 0$) has the smallest probability and is thus the least likely to donate according to this logistic model.