Міністерство освіти і науки Украіни

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО–МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

Система автоматичної оцінки ризиків своєчасного завершення проектів "Велике будівництво" на основі ретроспективного аналізу діяльності замовників та генеральних підрядників у системі Prozorro

Текстова частина до курсової роботи

за спеціальністю “прикладна математика” – 113

Керівник курсової роботи

Гороховський К. С.

_____

(підпис)

“____” _____ 2021 р.

Виконав студент факультету інформатики

спеціальності “Прикладна математика” – 3 курс

Кошмак І. О.

“____” _____ 2021 р.

Київ, 2021

Зміст

Анотація

У даній роботі розглядаються нейронні мережі та глибоке навчання. Покроково описується процес розробки системи оцінки ризиків своєчасного завершення проектів “*Велике будівництво*” за допомогою глибокого навчання на мові Python. Проводиться аналіз використання даних у сфері будівництва, розглядаються принципи та методи залучення цієї інформації у виробничий процес. Надаються рекомендації у рамках будівельних проектів на основі конкретних прикладів технологічних рішень з використанням Великих даних та Інтернету речей.

Вступ

Завдання пошуку ризиків, очевидно, є привабливим для людей з будь-якої сфери, тим більше, коли у ній задіяно багато людського та матеріального ресурсу. Ця тема тим паче стає наочнішою, коли ціллю є оптимізація державного бюджету, який наповнюється, в більшій мірі, за рахунок податків. Але як ті самі ризики передбачити? Недостатньо просто на рівні інтуїції зробити висновок, що той чи інший проект не вартий витрачених коштів та часу. Для коректної оцінки потрібні дані, або так звані Великі дані.

“Великі дані — дані, що занадто об’ємні, занадто динамічні та занадто складні для обробки…” [1] . Тим не менш, *Big data* вже задіяна майже у всіх сферах людської діяльності, в тому числі: медицина, державний сектор, торгівля, промисловість і так далі, при цьому збільшуючи свою значимість [2]. А за даними компанії IDC, «Глобальна датасфера» ще у 2018 році досягла 18 зетабайт (2^{70} байт) [6].

Але просто даних, звісно, вистачити не може, оскільки не менш важливим етапом після їх збирання є обробка. “Наука про дані включає принципи, процеси та методи для розуміння явищ за допомогою (автоматизованого) аналізу даних” [3].

Метою даної роботи є:

1. Розглянути Нейромережі та Глибоке навчання.

2. На прикладі даних державної програми “Велике будівництво” скласти систему оцінки ризиків своєчасного завершення проектів.

3. Розглянути існуючі методи використання big data у сфері будівництва та створити рекомендаційну систему на основі прийнятих світових практик.

Частина 1. Нейронні мережі та Глибоке навчання

1.1 Прийняття рішень на основі даних

Ще задовго до виникнення цивілізації, люди стикалися з проблемою прийняття рішень: куди краще піти на полювання, чи варто вживати в їжу якусь рослину, який дар слід піднести богам чи духам задля покращення погоди… Розвиваючись, людство вирішувало одні проблеми, але невпинно створювало нові і не менш складні. Важливим етапом у вирішенні проблеми вибору є якісь емпіричні дані, від яких можна відштовхуватись. Накопичення, систематизація і обробка цих знань дозволили людству розвиватись експоненційно. В цій частині річ піде про обробку інформації, а саме про машинне (і в тому числі глибоке) навчання.

Розробка та імплементація методів прийняття рішень на основі даних (*Data-driven decision making*) є однією з головних задач Науки про дані [3]. Але важливо зазначити, що існуючі методи часто є специфічними для конкретних випадків (наприклад модель представлення слів у вигляді векторів — *word2vec*) і тому дані часто визначають метод, або групу методів, за якими можна проводити аналіз. Одним із таких методів є Глибоке навчання (Deep learning), який в основному характеризується двома факторами: нелінійна багатошарова обробка та навчання (а) з вчителем або (б) без вчителя [4].

Вищезгаданий метод Глибокого навчання є лише підгалуззю Машинного навчання.

Машинне навчання цікаве в першу чергу тим, які завдання з його допомогою можна вирішити. Типовими сферами застосування [11] є:

- Задачі класифікації,
- Задачі регресії,
- Транскрипція,
- Переклад,
- Знаходження аномалій,
- Заповнення пропущених значень,
- Підвищення якості даних,
- Оцінка функцій щільності та ймовірності.

1.2 Нейрони і нейронні мережі

Основною архітектурною задачею саме Глибокого навчання є створення нейронних мереж. Ідейно вони повторюють аналогічні за назвою структури мозку (звідки і назва). Нейрон (рис. 1.1), атомарна одиниця обробки інформації, має наступну структуру: він отримує інформацію (за допомогою нейротрансмітерів та електромагнітних імпульсів) через дендриди (*dendrites*), обробляє її в тілі клітки (*soma* або *cell body*), і оброблену одиницю інформації передає через аксон (*axon*). Кожний окремо взятий нейрон має певний набір сусідів, з якими має контакти різної сили. Збуджуючись сила зв'язків нейронів може

посилюватись чи послаблюватись, що залежить від синхронності активації сусідів.

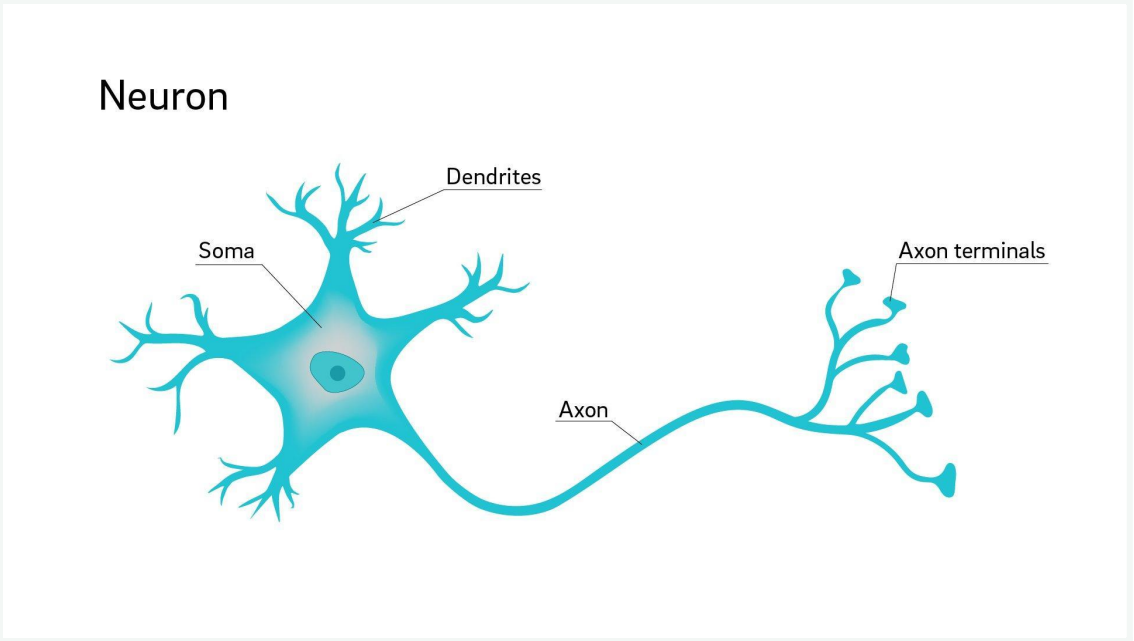

Рис. 1.1

(https://medicalxpress.com/news/2018-07-neuron-axons-spindly-theyre-optimizing.html)

Схожим за будовою є і нейрон в сенсі глибокого навчання (рис. 1.2): він так само отримує інформацію від сусідніх нейронів, “важливість” якої залежить від ваги зв’язку (аналогічно до сили зв’язку біологічних нейронів); обробляє за допомогою *функції активації* і виводить деякий результат. В загальному вигляді, результат обробки вхідних даних $x_1, x_2, ..., x_n$ з вагами $w_1, w_2, ..., w_n$, незалежною змінною b і функцією активації $\phi(x)$ буде наступним:

$$\phi(\sum_n x_n w_n + b)$$

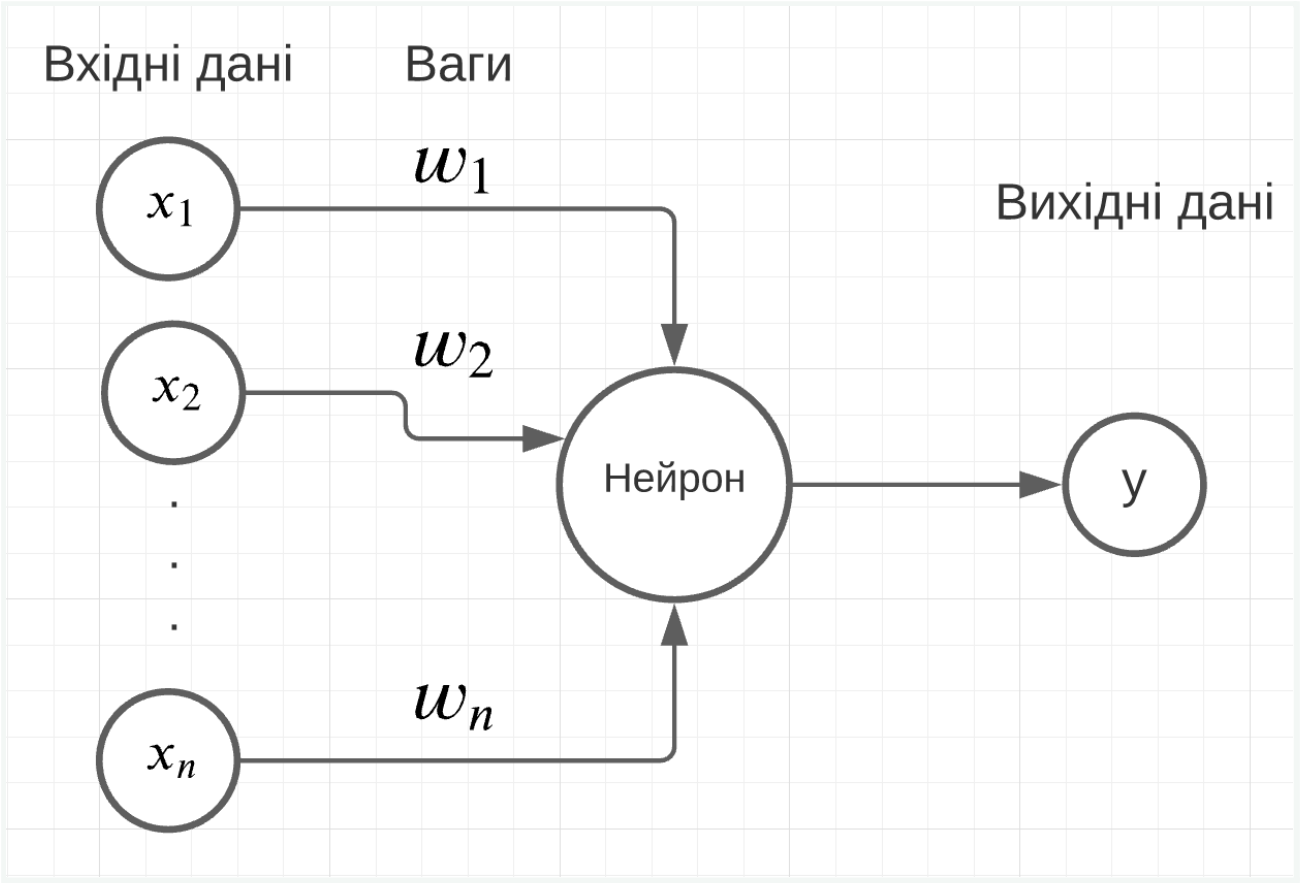

Рис. 1.2

Самі функції активації (Рис 1.3) умовно діляться на 2 типи: (а) лінійні та (б) нелінійні. Прикладами нелінійних є:

- $Sigmoid$, яка використовується, коли вихідний результат повинен належати проміжку (0, 1). Прикладом застосування є обрахування вірогідностей;
- $Tanh$, перевага якої полягає в тому, що негативні входи будуть відображені сильно негативними, а нульові входи будуть відображені біля нуля на графіку tanh (на відміну від $Sigmoid$).

Приклади лінійних:

- $Linear$, яка лінійно відображає вхідні дані;
- $ReLU$, яка використовується майже у всіх згорткових нейронних мережах або глибокому навчанні; вважається однією з найкращих та широко застосовуваних [10].

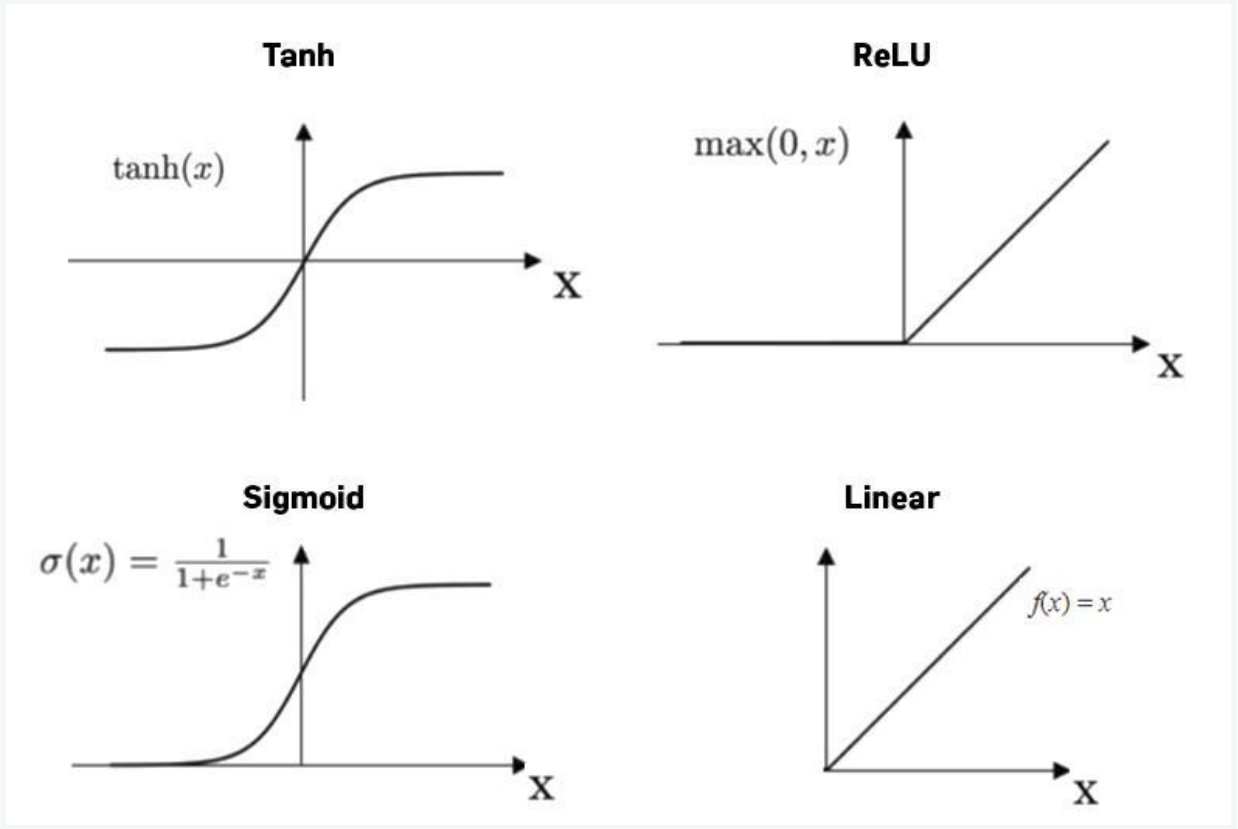

Рис 1.3

Нейрони, звісно, поодинці не застосовуються — вони є будівельним матеріалом нейронних мереж. У випадку deep learning, доцільно уточнити, що ці нейронні мережі багаторівневі. Їх глибина та кількість нейронів у кожному з рівнів залежить від конкретної задачі, для якої нейронна мережа спроектована. А сама задача побудови

архітектури нейронної мережі часто полягає у визначенні кількості рівнів, нейронів і функцій активації. Також існує практика використання вже готових архітектур (Рис 1.4).

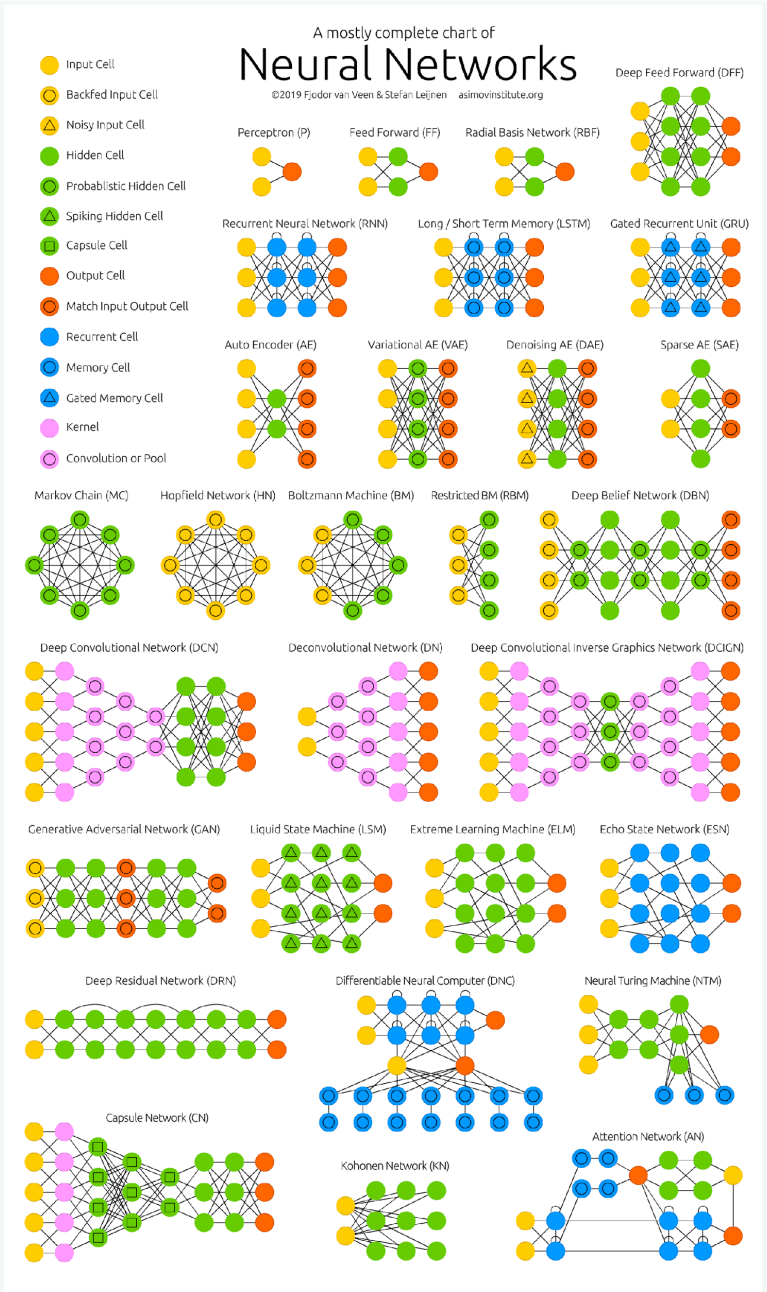

Рис 1.4 [14]

Рівні також поділяються за функціями (Рис 1.5): є так званий “вхідний рівень” (input layer), ціллю якого є отримання даних; “вихідний рівень” (output layer), який виводить результат; та “приховані рівні” (hidden layers), на яких, власне, і базується основна частина обчислень. Вхідний та вихідний рівні існують лише в єдиному екземплярі, а прихованих може бути будь-яка кількість (навіть 0, в цьому випадку нейронна мережа буде більш схожа на багатофакторну лінійну регресію).

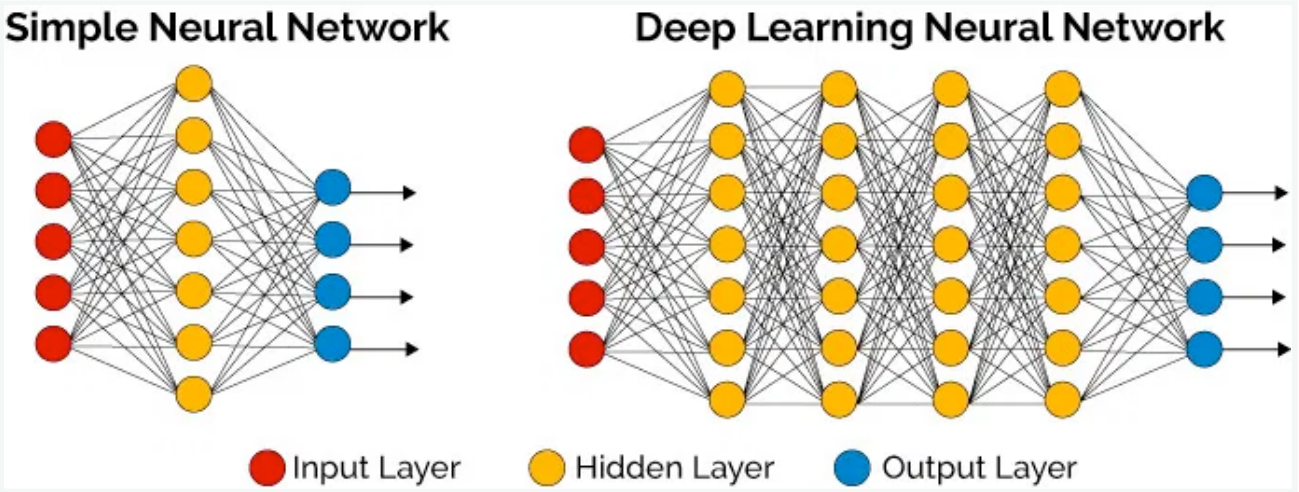

Рис 1.5 (https://thedatascientist.com/)

1.3 Навчання

Вищезгадана нелінійна багаторівнева обробка полягає в алгоритмі, за яким поточний рівень бере вихідні дані попереднього рівня як вхідні дані [4]. Між рівнями встановлюється чітка ієрархія для організації

важливості даних, які слід вважати корисними чи ні [4]. Процес встановлення цієї ієрархії називається навчанням.

Процесс навчання, здебільшого, полягає у зміні значень ваг так, щоб результат умовної функції $f: X \to Y$ (де X та Y — вхідні та вихідні дані відповідно) був найбільш точним. У випадку навчання з вчителем, то ці ваги калібруються за допомогою даних з завчасно відомим результатом (X та Y відомі завчасно). Алгоритми навчання з вчителем, в широкому сенсі, це група алгоритмів, які вчаться асоціювати деякі вхідні дані (X) з вихідними (Y) [11].

Частина 2. Розробка системи оцінки ризиків своєчасного завершення проектів державної програми “Велике будівництво”

2.1 Підготовка даних

Для створення та навчання моделі глибокого навчання було обрано бібліотеку Tensorflow [15], який реалізований на мові Python. Також використані бібліотекі pandas [16] (для зручної роботи з датасетом), scikit-learn [17] (для попередньої обробки датасету), matplotlib [18] (для візуалізації навчання) та transliterate [19] (для редагування назв змінних). Дані, що знаходяться у файлі *prozorro_data.xlsx*, імпортовано за допомогою бібліотеки pandas та збережено як об’єкт класу pandas.DataFrame [16] (Рис. 2.1).

```
data = pd.read_excel('prozorro_data.xlsx', sheet_name=None, header=1)
df = data['Data']
```

Рис. 2.1

Наступним кроком створено функцію для зміни назв колонок з кирилиці на латиницю, та застосовано на копії датасету (Рис. 2.2.1 і 2.2.2).

```
numeric_df = df_latin.select_dtypes(exclude=['object'])

not_date_cols = []
for col in numeric_df.columns:
    if not ('Data' in col or 'data' in col):
        not_date_cols.append(col)

numeric_df = numeric_df[not_date_cols]
numeric_df = numeric_df.fillna(0)
    for col in columns:
        col = translit(col, 'uk', reversed=True)
        for i, o in patterns.items():
            col = col.replace(i, o)
        eng_cols.append(col)

    return eng_cols
df_latin = df.copy(deep=True)
df_latin.columns = translate(df_latin.columns)
```

Рис. 2.2.1 і 2.2.2

Далі вилучено стовпчики із числовими даними, заповнено порожні значення нулями, оскільки це логічно виходить із суті цих колонок, оскільки це або (1) видатки і фінансування, або (2) відсоток виконаних робіт; тоді порожнє значення для (1) означає відсутність фінансування, а порожнє в (2) — відсутність просування у роботі (Рис. 2.3).

```
numeric_df = df_latin.select_dtypes(exclude=['object'])

not_date_cols = []
for col in numeric_df.columns:
    if not ('Data' in col or 'data' in col):
        not_date_cols.append(col)

numeric_df = numeric_df[not_date_cols]
numeric_df = numeric_df.fillna(0)
```

Рис. 2.3

Далі було обрано (Рис 2.4.1), відредаговано (Рис. 2.4.2 і 2.4.3) та закодовано (Рис 2.4.4) за допомогою One-Hot Encoding [20][21] деякі нечислові змінні (‘Назва місцевого бюджету’, ‘Назва бюджетної програми’, ‘Область’ і ‘Тип проекту’) для більш коректної їх репрезентації у вигляді чисел (1, якщо початкова змінна дорівнює назві колонки, і 0 в іншому випадку). Для цього використано класс OneHotEncoder з

бібліотеки scikit-learn [17]. Приклад застосування One-Hot Encoding на Рис. 2.4.5.

```
df_str = df_latin[['Oblast', 'Typ_proyektu',
                   'Kasovi_vydatky_sprjamovano_pydrjadnyku_u_2020_dzherelamy_derzhavnoho_bjudzhetu_nazva_bjudzhetnoyi_prohramy',
                   'Finansuvannja_peredbachene_u_2020_dzherelamy_mistsevoho_bjudzhetu_nazva_mistsevoho_bjudzhetu_']]
df_str.columns = ['Oblast', 'Typ_proyektu', 'Nazva_bjudzhetnoyi_prohramy', 'Nazva_mistsevoho_bjudzhetu']
```

```
df_str['Nazva_mistsevoho_bjudzhetu'].fillna('Невідомий', inplace=True)
combinatation_idx = df_str.query(
    "Nazva_mistsevoho_bjudzhetu not in ('Міський', 'Обласний', 'Районний', 'Сільський', 'Селищний', 'Невідомий')").index

df_str.loc[df_str.index.isin(combinatation_idx), 'Nazva_mistsevoho_bjudzhetu'] = 'Змішаний'
```

```
combinatation_idx = df_str.query(
    "Nazva_bjudzhetnoyi_prohramy not in ('ДФРР', 'Субвенція ПВ', 'Інша програма', \
    'Субвенція соц.-економ.', 'Надзвичайна кредитна програма для відновлення України', 'Спроможна школа', 'Невідома')").index

df_str.loc[df_str.index.isin(combinatation_idx), 'Nazva_bjudzhetnoyi_prohramy'] = 'Змішана'
```

```
enc = OneHotEncoder(sparse=False)
df_enc = pd.DataFrame(enc.fit_transform(df_str))
df_enc.index = df_str.index
df_enc.columns = enc.get_feature_names(df_str.columns)
```

Рис. 2.4.1 - 2.4.4

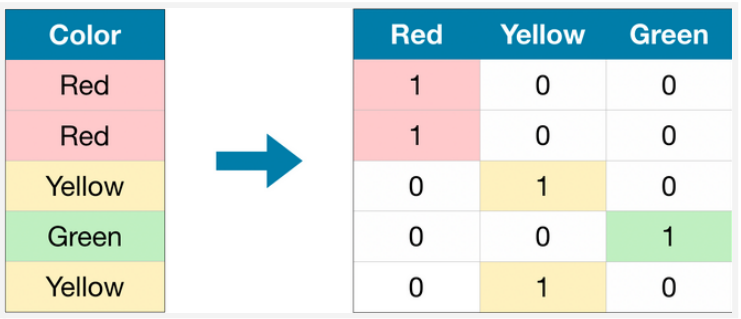

Рис. 2.4.5

Наступним кроком було обрано змінну ‘% Виконаних робіт відповідно плану робіт фактично’ для передбачення. Також було розділено датасет на 2 частини (Рис. 2.5): перша для тренування нейромережі, друга — для валідації (оцінки) нейромережі [20][21]. Для розділення було обрано функцію train_test_split з бібліотеки scikit-learn

[17], значення параметру test_size було визначено для того, щоб контролювати частку для валідації.

```
df_total.columns = translate(df_total.columns)

Y = df_total['%_vykonanykh_robit_vidpovidno_planu_robit_faktychno']
X = df_total.drop([
        '%_vykonanykh_robit_vid_zahalnoyi_vartosti_proyekta',
        '%_vykonanykh_robit_vidpovidno_planu_robit_faktychno',
        'Zaversheno_budivnytstvo_ta_vvedeno_v_ekspluatatsiju'
], axis=1)

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=1)
```

Рис 2.5

2.2 Робота з нейромережею

На цьому закінчується етап підготовки даних. Далі починається власне процес конструювання нейромережі. Першим кроком є створення функції з реалізованим класом для створення нейромережі (Рис 2.6) за допомогою бібліотеки tensorflow. В цьому класі однозначно визначена архітектура нейромережі типу Deep Feedforward Network з наступними рівнями:

1. Вхідний рівень (окремий нейрон для кожної зі змінних);
2. Перший прихований рівень з 32 нейронами;
3. Другий прихований рівень з 18 нейронами;
4. Вихідний рівень з 1 нейроном.

Функція похибки теж обрана однозначно — MeanAbsoluteError (Середня абсолютна похибка). Оптимізатор Adam [15] був обраний як стандартний [20]. Функція активації, крок навчання та список метрик — змінні.

```
def build_model(feature_layer_, learning_rate, metrics, activation):

    class Model(tf.keras.Model):
        def __init__(self):
            super(Model, self).__init__()
            self.dense1 = feature_layer_
            self.dense2 = tf.keras.layers.Dense(32, activation=activation)
            self.dense3 = tf.keras.layers.Dense(18, activation=activation)
            self.dense4 = tf.keras.layers.Dense(1)

        def call(self, inputs):
            """Go through every layer."""
            return self.dense4(self.dense3(self.dense2(self.dense1(inputs))))

    model = Model()
    model.compile(
        tf.keras.optimizers.Adam(
            learning_rate=learning_rate),
            loss=tf.keras.losses.MeanAbsoluteError(),
            metrics=metrics
    )
    return model
```

Рис. 2.6

Наступним кроком побудовано функцію для тренування нейромережі, яка просто використовує вбудований метод fit і повертає інформацію щодо значень похибки для наступної візуалізації (Рис. 2.7).

```
def train_model(model, x, y, epochs, batch_size):
    features = {str(name): np.array(value) for name, value in x.items()}
    label = np.array(y)

    history = model.fit(x=features, y=label, batch_size=batch_size, epochs=epochs)
    epochs = history.epoch
    hist = pd.DataFrame(history.history)

    return epochs, hist
```

Рис. 2.7

Функція для візуалізації, яка використовує функцію plot бібліотеки matplotlib [18] (Рис. 2.8).

```
def plot_curve(epochs, hist, metrics):
    plt.figure()
    plt.xlabel('Epochs')
    plt.ylabel('Value')

    for m in metrics:
        x = hist[m]
        plt.plot(epochs[1:], x[1:], label=m)

    plt.legend()
```

Рис. 2.8

Далі будується вхідний рівень, який потім буде передаватись у функцію build_model (Рис 2.9). Він будується за допомогою класу DenseFeatures бібліотеки tensorflow.

```
feature_columns = []
for col in X_train.columns:
    feature_columns.append(tf.feature_column.numeric_column(col))
feature_layer = tf.keras.layers.DenseFeatures(feature_columns)
```

Рис. 2.9

Далі частина з гіперпараметрами: learning_rate (крок навчання), num_epochs (кількість епох), batch_size (кількість даних (рядків), які будуть використані для тренування мережі за 1 раз), activation (функція активації) і metrics (метрики для візуалізації) [20] [21]. Значення для learning_rate, num_epochs, batch_size та activation налаштовані емпіричним шляхом, балансуючи між недонавчанням і перенавчанням (Рис. 2.10).

```
# Hyperparameters
learning_rate = 0.03
num_epochs = 50
batch_size = 128
activation = 'tanh'

metrics = [
    tf.keras.metrics.MeanAbsoluteError(name='MAE')
]

# Establish the model topography
model = build_model(feature_layer, learning_rate, metrics, activation)

# Train the model on the training set
epochs, hist = train_model(model, X_train, Y_train, num_epochs, batch_size)

list_of_metrics = ['MAE']

# Draw curve
plot_curve(epochs, hist, list_of_metrics)
```

Рис 2.10

Візуалізація навчання (значення MeanAbsoluteError) на графіку (Рис 2.11). Як видно із зображення, оцінка MeanAbsoluteError для моделі ~ 12.9.

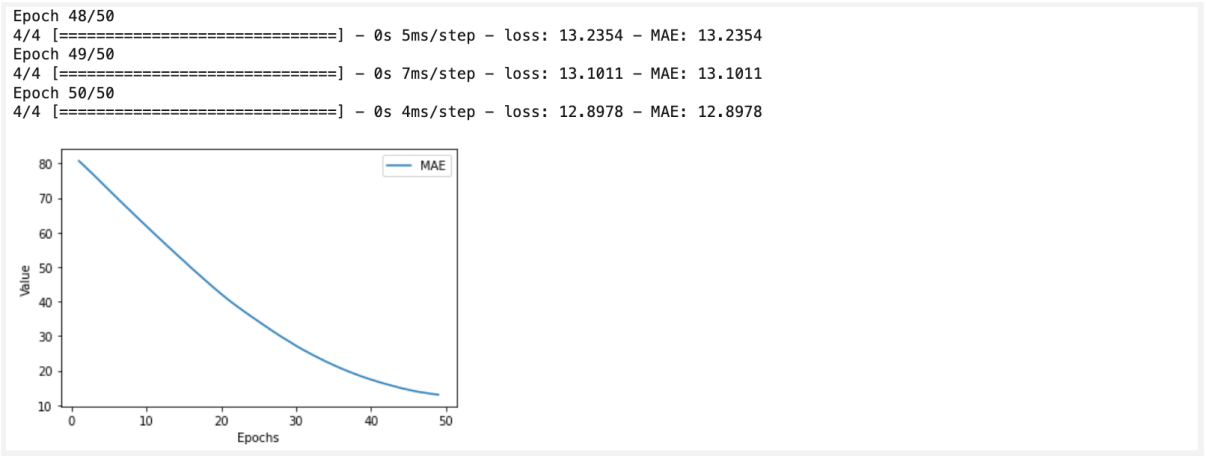

Рис 2.11

2.3 Висновки

Оцінка моделі на частині для валідації (Рис 2.12). Як видно із зображення, оцінка MeanAbsoluteError для моделі ~ 14,8, і різниця на тренувальній і валідаційній частинах невелика (~1,9), що свідчить про

відсутність або низький рівень перенавчання [20].

```
features = {str(name): np.array(value) for name, value in X_test.items()}
label = np.array(Y_test)

model.evaluate(x=features, y=label, batch_size=128)
```

```
2/2 [==============================] - 1s 7ms/step - loss: 14.7810 - MAE: 14.7810
[14.781038284301758, 14.781038284301758]
```

Рис 2.12

Хоча результат можна назвати відносно точним, але він все одно далекий від ідеалу, оскільки середня точність передбачення відсотку виконаних робіт відповідно плану робіт фактично ~14,8. Основну причину можна помітити навіть не проводячи детальний аналіз з використанням нейромереж, яка полягає в недостатній інформативності датасету. Маючи тільки описові дані, такі як тип проекту, джерела фінансування, суми фінансування, регіон і назву підрядної організації, очевидно, неможливо сказати про якість і швидкість проведення робіт. Причиною цього є те, що ці дані не вказують про якість та інтенсивність проведення робіт, не вказують про рівень економічності проекту, і тим більш не надає інформацію щодо екологічних показників (які хоч і не впливають на оцінку швидкості і якості проведення робіт, але все одно є цінними з точки зору планування).

Ці характеристики є вкрай цінними для побудови точної моделі. Методологія і приклади відстеження показників якості та швидкості проведення робіт описані в наступній частині роботи.

Частина 3. Великі дані в будівництві

3.1 Причини для використання

Технологічний розвиток сприяє все більшому накопиченню інформації, яка все частіше використовується у сферах, де ще кілька років тому її застосування було неочевидне [2]. Одним із прикладів деякий час назад став перетин Великих даних, робототехніки, Інтернету речей і сфери будівництва — так звана *Industry 4.0,* яку можна розпізнати за використанням дронів, будівельних 3D-принтерів і тд. Але існують також методи, які виходять за область поняття Індустрії 4.0, і при цьому також є технологічними і корисними, можуть істотно покращити процес прийняття рішень, мінімізувати ризики, збільшити доходи та розкрити цінну інформацію, яка в іншому випадку залишалася б прихованою [2].

Застосування сучасних методів допомагає будівельним компаніям не тільки збільшувати доходи, а також забезпечити вищий рівень безпеки на надійності своїх робітників [7]. Наразі [5] [9], основними сферами та прикладами використання великих даних в індустрії будівництва є:

1. Оптимізація ресурсів та відходів, покращення екологічних показників;

2. Моніторинг, планування та аналіз якості в режимі реального часу;

3. Управління інфраструктурою організації (в т.ч. моделювання робочого процесу на будівельних майданчиках за допомогою симуляцій, моделювання інфраструктури і тд);

4. Енергоефективність;

5. Прийняття рішень на основі аналізу раніше зібраних даних та визначення тенденцій і закономірностей;

6. Покращення комунікацій та інформаційних потоків між учасниками проектів, в тому числі за допомогою програм відстеження в реальному часі;

7. Контроль якості обладнання, інструментів та матеріалів і тд.

Сучасні рішення контролю виробництва стають не просто “технологічними забаганками”, а необхідністю. Традиційні методології моніторингу неспроможні впоратися з більшістю нагальних сценаріїв застосування, для яких залишається невирішеними проблеми зростаючих обсягів даних, їх інтеграції в виробничий процес, ефективний моніторинг та ін [7].

Сучасні реалізації, в свою чергу, використовують нові методи для збільшення об’ємів інформації. Прикладами є різні сенсори (температурні, тиску, кислотності, електропровідності, руху, тощо) [7] та трекери, які фіксують інформацію про стан обмежень (часових, ресурсних і тд) протягом усього терміну виконання завдання, починаючи від процесу планування виробництва, закінчуючи процесом підготовки до процесу виконання [8].

3.2 Планування на основі даних

Для початку треба зазначити, на яких саме етапах планування будівництва залучається інформація, в більш ширшому розумінні — як саме проходить планування на основі фітбеку. На рис. 3.1 зображена схема контролю робочого процесу на будівельних проектах, згідно з якою, планування формулює процеси попереднього виконання, тоді як засоби контролю оцінюють (що не є єдиною функцією), наскільки заплановане відповідає реальній ситуації для подальшого аналізу та імплементації [8].

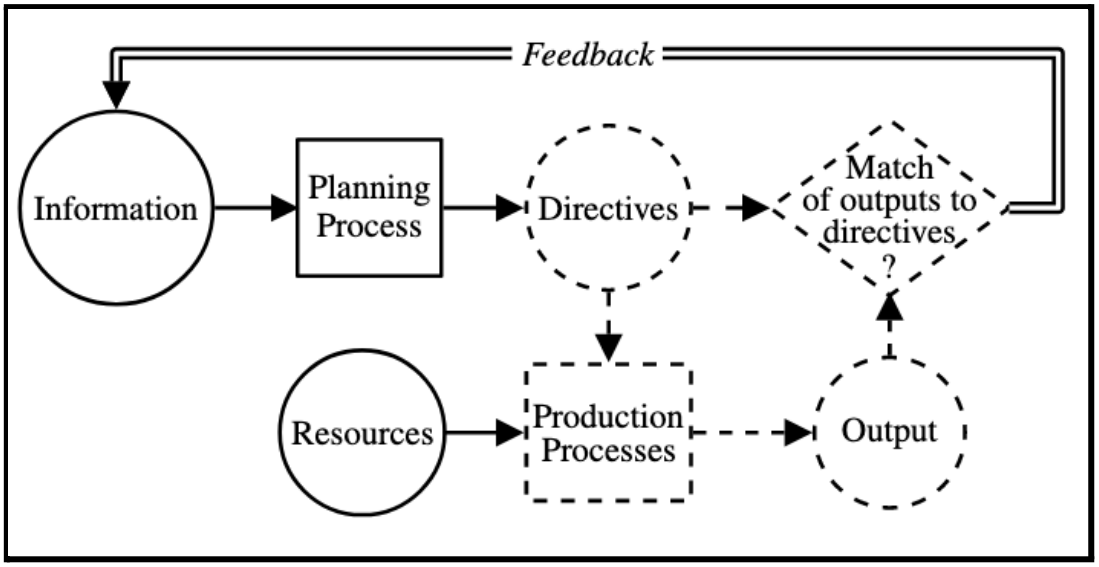

Рис 3.1 [8]

Щоб обійти обмеження класичних підходів, орієнтованих на в основному на виявленні, використовуються нові методології, що більше орієнтуються на причинно-наслідковий зв'язок процесів (не тільки для поліпшення виявлення, але в основному для діагностики та попередженню несправностей) [7]. Ці „структуровані” підходи можна

класифікувати як засновані на знаннях (knowledge-based) або на основі даних (data-driven), відповідно до походження інформації [7].

Класифікуючи за функціональністю, вирішальну роль в управлінні будівництвом мають інформаційні потоки [8]:

а) що дають можливість ефективно планувати (як короткостроково, так і довгостроково),

б) необхідні для виконання та контролю робочого процесу.

3.3 Приклади застосування

Існують багато різних прикладів застосування, які відрізняються ціною, складністю використання і видом отриманої вигоди (фінансова, прискорення процесу будівництва, підвищення рівня безпеки робітників і тд).

Для моніторингу поточного стану будівництва можна використовувати камери з пришвидшеною зйомкою або дрони, щоб оцінити або відповісти на вимогу оплати субпідрядниками. Наприклад, якщо “субпідрядник каже, що виконано 90% роботи, а з кадрів зрозуміло, що виконано не більше 40%, то субпідрядник не отримує оплату за пророблену роботу в повному об’ємі” [12]. Також, завдяки техніці реконструкції на основі зображень, можна визначити, чи відбувається будівництво відповідно до запланованого графіка [13].

Для моніторингу якості, можна за допомогою регулярних фотографій різних конструкцій виявляти можливі дефекти; “ми зазвичай

фотографуємо всю арматуру за день ... і іноді трапляється так, що у нас є кілька випадків, коли частини стійок лопаються, і ми дивимось раніше зроблені фотографії, на яких видно, що саме було встановлено в плиту і яким чином" [12].

Для оптимізації відходів можна вести облік кількості відходів, і робити планування на подібних проектах вже маючи деяких орієнтир щодо кількості відходів. "Ми збираємо дані про кількість відходів, які ми виробляємо на конкретному проекті ... і ми можемо використовувати ці дані для оцінки того, якою буде вартість наступної роботи на подібному проекті" [12].

Також можна застосовувати підхід до аналізу даних, заснований на правилах асоціації, який може бути використаний для вивчення причинно-наслідкових зв'язків аварій [13]. Іноді причиною цих аварій можуть стати природні катаклізми. Для них існують практики виявлення дефектів у будівництві після аварії за допомогою автоматизованого 3D-виявлення тріщин на основі зображень [13].

Висновки

В роботі показано важливість застосування даних у життєдіяльності людини. Розглянуто глибоке навчання, з описом нейронів і нейромереж. Побудована система оцінки ризиків своєчасного завершення проектів державної програми “Велике будівництво” з описом всіх кроків на основі готового датасету. Також зазначено про недостатню кількість зібраних даних для точного аналізу.

Була описана роль великих даних у сфері будівництва та процес залучення цих даних. Наведено приклади застосування технологічних рішень із залученням даних у сфері будівництва.

Використана література

1. 1. S. Madden, "From Databases to Big Data," in IEEE Internet Computing, vol. 16, no. 3, pp. 4-6, May-June 2012, doi: http://doi.org/10.1109/MIC.2012.50
2. McKinsey Global Institute, ‘Big Data: The Next Frontier for Innovation,Competition, and Productivity’, (2011), https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/McKinsey%20Digital/Our%20Insights/Big%20data%20The%20next%20frontier%20for%20innovation/MGI_big_data_full_report.pdf
3. Foster Provost and Tom Fawcett.Big Data.Mar 2013.51-59, doi: http://doi.org/10.1089/big.2013.1508
4. R. Vargas, A. Mosavi, L. Ruiz, Deep Learning: A Review, Advances in Intelligent Systems and Computing, (2017). https://eprints.qut.edu.au/127354/1/DEEP_LEARNING_A_REVIEW.pdf
5. A. Verdenhofs, I. Geipele, T. Tambovceva, Big data in construction industry: systematic literature overview, (2019), doi: https://doi.org/10.3846/mbmst.2019.062
6. D. Reinsel, J. Gantz, J. Rydning, ‘ The Digitization of the World’, https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf
7. Reis, M.S.; Gins, G. Industrial Process Monitoring in the Big Data/Industry 4.0 Era: from Detection, to Diagnosis, to Prognosis. *Processes* 2017, *5*, 35. https://doi.org/10.3390/pr5030035

8. B. Daveb, S. Kublerc, K. Fräṁling, L. Koskelab, ‘Opportunities for enhanced lean construction management using Internet of Things standards’, (2015), https://orbilu.uni.lu/bitstream/10993/23028/1/AiC_2015.pdf
9. Makram Bou Hatoum, Melanie Piskernik and Hala Nassereddine. Pages 1299-1306 (2020 Proceedings of the 37th ISARC, Kitakyushu, Japan, ISBN 978-952-94-3634-7), doi: https://doi.org/10.22260/ISARC2020/0178
10. P. Ramachandran, B. Zoph, Q. V. Le, ‘Searching for Activation Functions’, (2017), https://arxiv.org/abs/1710.05941
11. I. Goodfellow, Y. Bengio, A. Courville, ‘Deep Learning’, (2016), https://www.deeplearningbook.org
12. Atuahene, Bernard & Kanjanabootra, Sittimont & Gajendran, Thayaparan. (2020). Benefits of Big Data Application Experienced in the Construction Industry: A Case of an Australian Construction Company. https://www.researchgate.net/publication/344159534
13. Huang, Yao & Shi, Qian & Zuo, Jian & Pena-Mora, Feniosky & Chen, Jindao. (2021). Research Status and Challenges of Data-Driven Construction Project Management in the Big Data Context. Advances in Civil Engineering. 2021. 1-19. 10.1155/2021/6674980 https://www.researchgate.net/publication/350881833
14. https://www.asimovinstitute.org/neural-network-zoo/
15. https://www.tensorflow.org/
16. https://pandas.pydata.org/

17. https://scikit-learn.org/
18. https://matplotlib.org/
19. https://pypi.org/project/transliterate/
20. https://developers.google.com/machine-learning/crash-course/
21. https://www.kaggle.com/learn