⑤ ドキュメント作成：MLモデル 成果物レポート（Sample）

1. 基本情報

- 受講者氏名：田中　太郎
- 作成日：2025年10月1日
- 対象工程：①～④ドキュメント作成（各種ドキュメント概要まとめ）
- 成果物ファイル名：「プロジェクト計画書.md, 要件定義書.md, リファクタリング報告書.md, リリースノート.md, 使用手順書.md
- 成果物バージョン：1.0

2. 概要

本報告書は、電力需要予測AI/ML構築プロジェクトにおいて作成された各種ドキュメントの概要をまとめたものである。本プロジェクトは、4つの異なる機械学習アルゴリズム（Keras, LightGBM, PyCaret, RandomForest）を用いたアンサンブル予測システムを開発し、電力需要の短期・中期予測精度の最大化を実現した。

3. プロジェクト基本情報

項目	内容
プロジェクト名	電力需要予測AIモデル構築
実施期間	2025年10月1日 ～ 2025年11月30日
対象データ	東京電力電力需要実績データ（2016-2025年）、気象庁気温データ
開発言語	Python 3.10.11
技術スタック	TensorFlow/Keras, LightGBM, PyCaret, Scikit-learn, Pandas, NumPy
予測対象	指定期間の電力需要量（MW単位）

4. ドキュメント作成実績

4.1. プロジェクト計画書

主要内容

- **目的・背景**: 電力需要予測の精度向上とエネルギー管理最適化を目的とした機械学習システム構築
- **プロジェクトスコープ**: 4種類のML/アルゴリズム（Keras, LightGBM, PyCaret, RandomForest）による包括的予測システム
- **成功指標**: 予測精度（RMSE < 200MW、R2スコア > 0.9）、処理性能（メモリ50%削減、学習時間30%短縮）

技術スタック構成

カテゴリ	技術	バージョン	用途
データ処理	Pandas	1.5.3	データ読み込み、前処理
数値計算	NumPy	1.24.4	配列操作、数値演算
深層学習	TensorFlow/Keras	2.15.0	ニューラルネットワーク
勾配ブースティング	LightGBM	3.3.5	高速機械学習
自動機械学習	PyCaret	3.0.0	自動モデル選択・最適化
可視化	Matplotlib	3.7.5	グラフ描画、結果可視化

4.2. 要件定義書

主要内容

- **ビジネス要件**: 短期予測（1日～1週間）精度95%以上、中期予測（1カ月～3カ月）精度90%以上
- **機能要件**: リアルタイム予測更新機能、異常値検知・アラート機能、複数シナリオ対応予測
- **非機能要件**: 予測処理時間10秒以内、データ更新頻度1時間毎、システム可用性99.9%以上

データ要件仕様

入力データ

- 過去5年間の電力使用量実績データ（東京電力）
- 気象データ（気温、湿度、風速等）
- カレンダー情報（平日/休日、季節性）
- イベント・特殊要因データ

出力データ

- 電力需要予測値（時間単位、日単位、月単位）
- 予測信頼区間
- 影響因子分析結果

4.3. リファクタリング報告書

主要内容

- **改修目的**: 従来の単一モデルアプローチから、4つのMLアルゴリズムによるアンサンブル手法への高度化
- **統一データクラス実装**: @dataclass設定管理、robust_model_operationアプローチによる統一エラーハンドリング
- **性能改善成果**: メモリ使用量50%削減、学習時間30%短縮、予測精度の大幅向上

統一データクラス設計

```
@dataclass
class ModelConfig:
    """モデル設定統一管理クラス"""
    INPUT_X: str = r"..\..\data\Xtrain.csv"
    INPUT_Y: str = r"..\..\data\Ytrain.csv"
    DATA_DTYPE: str = "float32"          # メモリ最適化
    ENCODING: str = 'shift_jis'          # 日本語データ対応
    RANDOM_STATE: int = 42               # 再現性確保
    TEST_SIZE: float = 0.1               # テストデータ比率
    CV_FOLDS: int = 5                    # クロスバリデーション
    N_JOBS: int = -1                     # 全利用可能最大活用
    MEMORY_OPTIMIZATION: bool = True     # メモリ最適化有効
```

4.4. リリースノート

主要内容

- **バージョン**: v1.0.0（4つのMLモデル統合版）
- **システム要件**: Python 3.10.11, メモリ8GB以上推奨, GPU推奨（TensorFlow用）
- **主要機能**: 4種類のMLアルゴリズム統合、統一設定管理、性能監視機能

モデル構成

```
AI/
├── train/
│   ├── keras_train.py（深層学習モデル）
│   ├── lightgbm_train.py（勾配ブースティング）
│   ├── pycaret_train.py（自動機械学習）
│   └── randomforest_train.py（アンサンブル学習）
├── tomorrow/
│   ├── keras_tomorrow.py（翌日予測）
│   ├── lightgbm_tomorrow.py
│   ├── pycaret_tomorrow.py
│   └── randomforest_tomorrow.py
└── data/（入力データ・出力結果）
```

4.5. 使用手順書

主要内容

- **システム概要**: 4つの機械学習アルゴリズムによる電力需要予測システム
- **環境セットアップ**: Python仮想環境構築、依存関係インストール手順
- **基本操作**: モデル訓練、予測実行、結果可視化の実行手順

実行可能スクリプト

訓練用

- keras_train.py（深層学習・ニューラルネットワーク）
- lightgbm_train.py（勾配ブースティング・高速学習）
- pycaret_train.py（自動機械学習・モデル選択）
- randomforest_train.py（アンサンブル学習・ランダムフォレスト）

5. 技術的成果

5.1. アルゴリズム性能比較

アルゴリズム	予測精度(R²)	学習時間	メモリ使用量	適用場面
Keras (LSTM)	0.95	45分	2.1GB	短期予測・時系列
LightGBM	0.93	8分	512MB	中期予測・高速処理
PyCaret	0.91	15分	1.2GB	自動最適化・プロトタイプ
RandomForest	0.89	12分	800MB	異常値検知・安定性

5.2. 統一データクラスの実現

アーキテクチャ成果

- **設定管理**: @dataclassによるModelConfig統一実装
- **エラーハンドリング**: robust_model_operation アプローチによる統一処理
- **性能監視**: 実行時間・メモリ使用量自動測定機能
- **可視化**: 日本語フォント対応による統一グラフ出力

品質向上効果

- **コード再利用率**: 75%向上（共通モジュール化）
- **保守効率**: 60%向上（統一データクラス）
- **エラー発生率**: 80%削減（統一エラーハンドリング）

6. 品質保証活動

6.1. モデル性能評価

モデル	RMSE	MAE	R2スコア	学習時間
Keras	< 200kW	< 150kW	> 0.90	5-10分
LightGBM	< 180kW	< 130kW	> 0.92	1-3分
PyCaret	< 190kW	< 140kW	> 0.91	3-5分
RandomForest	< 210kW	< 160kW	> 0.89	2-4分

6.2. システム品質保証

品質指標	目標値	達成値	改善率
コードカバレッジ	90%以上	95%	達成
メモリ使用量	50%削減	55%削減	110%達成
エラー率	0.1%以下	0.05%	200%改善
学習時間	30%短縮	35%短縮	117%達成

6.3. コード品質向上

品質項目	改善内容	達成状況
統一データクラス	@dataclass設定管理統一	100%完了
エラーハンドリング	robust_model_operationアプローチ	全モデル適用
型安全性	型ヒント完全実装	100%完了
ドキュメント	docstring・コメント充実	100%完了

7. システム運用・保守

7.1. 運用監視機能

リアルタイム監視

- **実行時間監視**: 各モデルの学習・予測時間自動測定
- **メモリ使用量監視**: プロセスメモリ使用量リアルタイム追跡
- **エラー検知**: 例外発生時の自動ログ記録・通知
- **性能劣化検知**: 予測精度低下の早期発見

ログ管理

```
# 統一ログフォーマット
logger.info(f"=== {operation_name} 開始 ===")
logger.info(f"実行時間: {execution_time:.1f}秒")
logger.info(f"メモリ使用量: {initial_memory:.1f}MB → {final_memory:.1f}MB")
logger.info(f"メモリ差分: {memory_diff:+.1f}MB")
```

7.2. 保守性向上

統一データクラスによる保守効率化

- **設定管理統一**: @dataclass による型安全な設定管理
- **エラーハンドリング統一**: robust_model_operation アプローチ
- **可視化統一**: 16ページスクリット・日本語フォント対応
- **ファイル名統一**: {model}_model.sav, {model}_Ypred.csv

拡張性確保

- **モジュール設計**: 各モデルの独立実装
- **プラグイン対応**: 新モデル追加時の最小変更
- **設定外部化**: 環境別パラメータ管理
- **API化対応**: 将来のWeb API化に対応

8. 総括・成果まとめ

本プロジェクトでは、機械学習技術を活用した包括的な電力需要予測システムを構築し、4つの異なるアルゴリズムによる高精度予測を実現した。統一データクラスの実装により、保守性・拡張性・運用効率の大幅な向上を達成した。

特に注目すべきは、@dataclass設定管理とrobust_model_operationアプローチによる統一実装により、各モデルの独立性を保ちながら、統一的な品質・性能を確保した点である。これにより、新規モデル追加や機能拡張において高い効率性を実現し、持続可能な開発基盤を構築した。

また、メモリ最適化・並列処理・リアルタイム監視機能により、エンタープライズレベルの要求に応える高品質なシステムを完成させ、電力需要予測業務の自動化・高度化に大きく貢献した。各モデルの特徴を活かした適材適所の運用により、様々な予測ニーズに対応可能な柔軟性も確保している。今後は継続的な改善とモデル最適化により、さらなる予測精度向上と業務効率化を追求していく。